

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

U.S. Pat. No.: 7,103,640

Issue Date: September 5, 2006

Appl. No.: 09/661,222

Filing Date: September 13, 2000

Pat. Title: NETWORK DISTRIBUTED TRACKING
WIRE TRANSFER PROTOCOL

Control No.: To be assigned

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

**REQUEST FOR *EX PARTE* REEXAMINATION OF
U.S. PATENT NO. 7,103,640**

Dear Commissioner:

Pursuant to 35 U.S.C. § 302 and 37 C.F.R. § 1.510, the undersigned, on behalf of Amazon Web Services, Inc. (the "Requester") requests *ex parte* reexamination of claims 17, 18, and 24 (the "Challenged Claims") of U.S. Patent No. 7,103,640 ("the '640 Patent," Exhibit 1), issued on September 5, 2006. The '640 Patent is assigned to Kove IO, Inc. ("Patent Owner").

Ex Parte Patent Reexamination Filing Requirements

Pursuant to 37 C.F.R. § 1.510(b)(1), this Request provides statements pointing out each substantial new question of patentability based on material, non-cumulative reference patents and printed publications for the Challenged Claims of the '640 Patent (*see* Section II). In particular, this Request raises three substantial new questions of patentability for obviousness under 35 U.S.C § 103 based on: (1) U.S. Pat. No. 6,473,781 ("Skagerwall") alone or in view of U.S. Pat.

No. 6,185,601 (“Wolff”); (2) Oracle Corporation’s manual *Oracle Names Administrator’s Guide, Release 2.0* (“OracleNamesAdminGuide”) in view of Maarten van Steen’s IEEE article “Locating Objects in Wide-Area Systems” (“Steen”) and/or Joseph Greene’s book *Oracle 8 Server: Unleashed* (“OracleUnleashed”); and (3) Steen alone or in view of Skagerwall and/or Wolff.

Pursuant to 37 C.F.R. § 1.510(b)(2), this Request requests reexamination of the Challenged Claims of the ‘640 Patent, and provides a detailed explanation of the pertinence and manner of applying the cited references to the Challenged Claims (*see* Section III).

Pursuant to 37 C.F.R. § 1.510(b)(3), the Request provides copies of every patent or printed publication relied upon or referred to in the statement pointing out each substantial new question of patentability or in the detailed explanation of the pertinence and manner of applying the cited references (see Exhibits 2-6).

Pursuant to 37 C.F.R. § 1.510(b)(4), the Request provides a copy of the ‘640 Patent as Exhibit 1, along with a copy of any disclaimer, certificate of correction, and reexamination certificate issued corresponding to the patent.

Pursuant to 37 C.F.R. § 1.510(b)(5), the attached Certificate of Service indicates that a copy of this Request in its entirety has been served on Patent Owner at the following address of record for Patent Owner, in accordance with 37 C.F.R. § 1.33(c):

Reichman Jorgensen Lehman & Feldberg LLP
100 Marine Parkway, Suite 300
Redwood City, CA 94065

Also submitted herewith is the fee set forth in 37 C.F.R. § 1.20(c)(1).

Pursuant to 37 C.F.R. § 1.510(b)(6), Requester hereby certifies that the statutory estoppel provisions of 35 U.S.C. § 315(e)(1) and 35 U.S.C. § 325(e)(1) don't prohibit Requester from filing this *ex parte* patent reexamination request.

TABLE OF CONTENTS

I.	Introduction	1
II.	Substantial New Questions Of Patentability	7
A.	‘640 PATENT OVERVIEW	7
B.	THE CHALLENGED CLAIMS AREN’T ENTITLED TO CLAIM PRIORITY TO THE FILING DATE OF THE ‘896 PROVISIONAL APPLICATION.	9
C.	LISTING OF PRIOR ART PATENTS AND PRINTED PUBLICATIONS	11
D.	STATEMENT SETTING FORTH EACH SUBSTANTIAL NEW QUESTION OF PATENTABILITY	11
E.	CO-PENDING LITIGATION	14
F.	NEW AND NON-CUMULATIVE PRIOR ART	16
1.	<i>In re Vivint</i> Isn’t Applicable Here.	19
2.	<i>Becton, Dickinson</i> Isn’t Applicable Here.	20
G.	ESTOPPEL	24
III.	Detailed Explanation Of The Pertinence And Manner Of Applying The Prior Art References To Every Claim For Which Reexamination Is Requested	24
A.	LEVEL OF SKILL IN THE ART	24
B.	CLAIM CONSTRUCTION	25
C.	OVERVIEW OF THE PRIOR ART	28
1.	Skagerwall (Ex. 2) Overview	28
2.	Wolff (Ex. 3) Overview	31
3.	Steen (Ex. 4) Overview	32
4.	Oracle Overview	37
5.	OracleNamesAdminGuide (Ex. 5) Overview	42
6.	OracleUnleashed (Ex. 6) Overview	60

D.	SNQ 1: SKAGERWALL IN VIEW OF WOLFF RENDERS OBVIOUS INDEPENDENT CLAIMS 17 AND 18 AND DEPENDENT CLAIM 24.	70
1.	Skagerwall in View of Wolff Renders Obvious Independent Claims 17 and 18.	73
2.	Skagerwall Alone or in View of Wolff Renders Obvious Dependent Claim 24.	93
E.	SNQ 2: CLAIMS 17, 18, AND 24 ARE OBVIOUS BASED ON ORACLE NAMES ADMIN GUIDE IN VIEW OF ORACLE UNLEASHED AND STEEN.	96
1.	Patent Owner Distinguished Its Claimed Invention from Hierarchical and Mixed Topology Embodiments.	98
2.	Routing Efficiency Benefits Hierarchical, Non-Hierarchical, And Mixed Topology Embodiments.	99
3.	OracleNamesAdminGuide in View of OracleUnleashed and Steen Render Obvious Independent Claim 17.	104
4.	Independent Claim 18 is Obvious Over OracleNamesAdminGuide in View of OracleUnleashed and Steen.	159
5.	OracleNamesAdminGuide in View of OracleUnleashed and Steen Renders Obvious Dependent Claim 24.	167
F.	SNQ 3: STEEN IN VIEW OF WOLFF AND/OR SKAGERWALL RENDERS OBVIOUS INDEPENDENT CLAIMS 17 AND 18 AND DEPENDENT CLAIM 24.	177
1.	Steen Alone or in View of Skagerwall and/or Wolff Renders Obvious Independent Claims 17 and 18.	182
2.	Steen, Skagerwall, and Wolff Render Obvious Dependent Claim 24.	215
G.	SECONDARY CONSIDERATIONS	220
IV.	Conclusion.....	226

TABLE OF EXHIBITS

Exhibit	Description
1	U.S. Patent No. 7,103,640 (“the ’640 Patent”)
2	U.S. Patent No. 6,473,781 to Roger Skagerwall and Kaj Nygren, filed October 27, 1998, issued October 29, 2002 (“Skagerwall”)
3	U.S. Patent No. 6,185,601 to James Wolff, filed April 15, 1998, issued February 6, 2001 (“Wolff”)
4	Maarten van Steen et al., “Locating Objects in Wide-Area Systems,” IEEE Communications Magazine, Vol. 36, Issue 1, pp. 104-09 (Jan. 1998) (“Steen”)
5	<i>Oracle Names Administrator’s Guide, Release 2.0</i> (Oracle Corp., Redwood City, CA), published 1996 (“OracleNamesAdminGuide”)
6	Joseph B. Greene, <i>Oracle8 Server Unleashed</i> , (Sams Publishing, Indianapolis, Ind.), published August 2, 1998 (“OracleUnleashed”) (divided into Part 1 (front cover through page 214), Part 2 (pages 215 through 454), Part 3 (pages 455 through 695), and Part 4 (page 696 through the back cover))
7	U.S. Patent No. 7,233,978 to Overton et al., filed on June 1, 2001, issued on June 19, 2007 (“the ’978 Patent”)
8	U.S. Patent No. 7,814,170 to Overton et al., filed on February 13, 2006, issued on October 12, 2010 (“the ’170 Patent”)
9	Original Prosecution History of the ’640 Patent (“’640 History”)
10	Original Prosecution History of the ’978 Patent (“’978 History”)
11	Original Prosecution History of the ’170 Patent (“’170 History”)
12	Departments of Informatics, <i>Oracle V2</i> , Virtual Exhibitions in Informatics, University of Klagenfurt (September 30, 2019), http://cs-exhibitions.uni-klu.ac.at/index.php?id=403 (last accessed October 4, 2021)
13	<i>Oracle7 Server Concepts, Release 7.3</i> (Oracle Corp., Redwood City, CA), published 1996
14	George Koch & Kevin Loney, <i>Oracle: The Complete Reference, Electronic Edition</i> (Oracle Press), ISBN: 0078822858, published 1996 (excerpts)
15	Kevin Loney, <i>Oracle DBA Handbook, 7.3 Edition</i> (McGraw-Hill Osborne Media) ISBN: 0078822890, published 1997 (excerpts)

Exhibit	Description
16	<i>Kove IO, Inc. v. Amazon Web Services, Inc.</i> , 1:18-cv-08175 (N.D. Ill.), Kove’s Memorandum in Opposition to Amazon Web Services, Inc.’s Motion for Judgment on the Pleadings, D.I. 145 (May 29, 2020)
17	<i>Kove IO, Inc. v. Amazon Web Services, Inc.</i> , 1:18-cv-08175 (N.D. Ill.), Joint Claim Construction Chart (Updated), D.I. 382 (August 11, 2021)
18	Declaration of Sylvia D. Hall-Ellis, Ph.D. Under 37 C.F.R. § 1.132 Regarding Publication of <i>Oracle Names Administrator’s Guide, Release 2.0</i> (Exhibit 2)
19	Declaration of Joseph B. Greene Under 37 C.F.R. § 1.132 (“Greene Decl.”)
20	<i>Amazon Web Services, Inc., v. Kove IO, Inc.</i> , IPR2020-00276, Paper 11 (Patent Owner’s Preliminary Response) (PTAB March 19, 2020)
21	<i>Amazon Web Services, Inc., v. Kove IO, Inc.</i> , IPR2020-00276, Paper 23 (Decision) (PTAB March 19, 2020)
22	Oracle - Investor Relations – FAQ, Frequently Asked Questions, https://investor.oracle.com/faq/default.aspx (last accessed October 4, 2021)
23	<i>Kove IO, Inc. v. Amazon Web Services, Inc.</i> , 1:18-cv-08175 (N.D. Ill.), Kove IO, Inc.’s Final Enforceability and Validity Contentions (August 13, 2020)
24	NSN 7610-01-448-8459, 7610014488459, ISO Group Defense and Aerospace Supply Chain Partner, https://www.iso-group.com/NSN/7610-01-448-8459 (last accessed on October 12, 2021)
25	<i>Kove IO, Inc. v. Amazon Web Services, Inc.</i> , 1:18-cv-08175 (N.D. Ill.), Claim Construction Order (Updated), D.I. 484 (December 17, 2021)
26	<i>Kove IO, Inc. v. Amazon Web Services, Inc.</i> , 1:18-cv-08175 (N.D. Ill.), Motion For Addl Claim Construction, D.I. 539 (November 28, 2022)
27	<i>Kove IO, Inc. v. Amazon Web Services, Inc.</i> , 1:18-cv-08175 (N.D. Ill.), Minute Entry, D.I. 580 (January 14, 2023)
28	Reexamination History of the ’170 Patent (Control Number 90/019,035)
29	Reexamination History of the ’640 Patent (Control Number 90/019,036)
30	Reexamination History of the ’978 Patent (Control Number 90/019,034)
31	Reexamination History of the ’978 Patent (Control Number 90/019,109)

Exhibit	Description
32	ACM Digital Library TOC
33	Yigdal Bejerano et al., “An Efficient Mobility Management Strategy for Personal Communication Systems,” Department of Electrical Engineering Technion - Israel Institute of Technology, pp. 215-222 (1998)
34	C. Ballintijn et al., “Lightweight Crash Recovery in a Wide-area Location Service,” VU Research Portal, pp. 1-13 (October 1998)
35	Evaggelia Pitoura et al., “An Efficient Hierarchical Scheme for Locating Highly Mobile Users,” Department of Computer Science, University of Iowa, pp. 218-225 (1998)
36	Amin Vahdat et al., “Active Naming: Programmable Location and Transport of Wide-Area Resources” (November 1998)
37	U.S. Patent No. 6,122,664 to Boukobza, filed on June 27, 1997, issued on September 19, 2000 (“Boukobza”)
38	U.S. Patent No. 6,230,183 to Yocum, filed on March 11, 1998, issued on May 8, 2001 (“Yocum”)
39	U.S. Pat. Application No. 09/111,896 to Overton and Roizen, filed July 8, 1998 (‘896 Application”)

USE OF EMPHASIS IN QUOTATIONS

All emphases in quotations and exhibit citations have been added, unless otherwise indicated.

APPLICABLE STATUTES

References to 35 U.S.C. §§102 and 103 are to the pre-AIA versions applicable to the ‘640 Patent.

I. INTRODUCTION

Claims 17, 18, and 24 of the ‘640 Patent should be rejected for obviousness under 35 U.S.C. § 103. The (1) Skagerwall alone or with Wolff, (2) OracleNamesAdminGuide and Steen and/or OracleUnleashed, and (3) Steen and Skagerwall and/or Wolff combinations each render obvious independent claims 17 and 18 and the additional elements of claim 24, which depends from claim 18.

The ‘640 Patent is directed to techniques for locating data stored in a distributed network.¹ To do so, the patent describes storing the network addresses of data in a network of location servers, then having clients query one of the location servers to retrieve the network address of the desired data.² The ‘640 Patent’s specification explains that the location server network may have a non-hierarchical (e.g., flat or cluster) architecture, a hierarchical (e.g., tree) architecture, or a mixture of the two.³ But Patent Owner argued in Reexamination Control No. 90/019,036 that the claims don’t encompass the hierarchical and mixed architectures disclosed by the prior art in that proceeding. Specifically, Patent Owner stated that the claims require that (1) the location servers be organized in a non-hierarchical cluster structure, (2) each location server be able to respond to any location request with either the location of the data or the location of a server known to contain the requested location data without an intervening traversal, and (3) each request must be resolved in two or fewer traversals.⁴ After Patent Owner made

¹ See ‘640 Patent at Abstract.

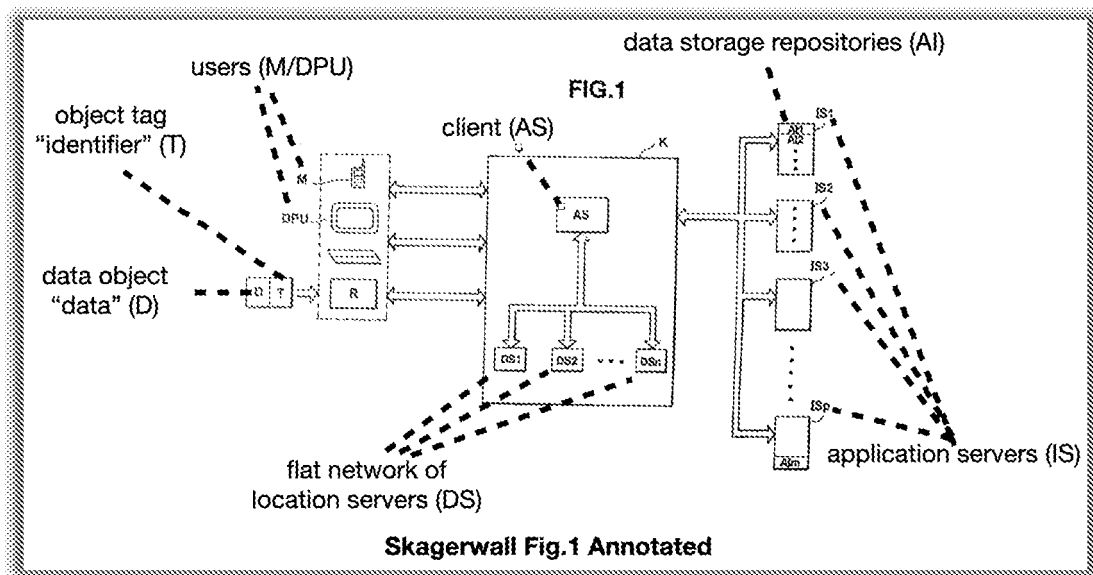
² See *id.* at Abstract, 2:12-27.

³ *Id.* at 14:29-31 (“The redirection mechanisms allow cluster and hierarchical topologies, and mixtures of such topologies...”)

⁴ See 08/04/2022 Patent Owner’s Reply to Office Action in Reexamination Control No. 90/019,036 at 23-25.

those arguments, the Office withdrew its rejection and confirmed the claims.⁵ As shown in SNQs 1-3, the prior art here teaches all elements of claims 17, 18, and 24, even under this narrowed claim scope.

Skagerwall provides an example of a location server network with a flat architecture. Like the '640 Patent, Skagerwall describes a system of directory servers “for communicating information, related to a plurality of tagged objects, in a network.”⁶ Skagerwall teaches that its system associates “data objects” (i.e. data entities) stored on application servers with respective “object data” stored on directory servers, including retrieval information (i.e., location information) using an “object tag” (i.e., identifier).⁷



⁵ 09/22/2022 Notice of Intent to Issue *Ex Parte* Reexamination Certificate in Reexamination Control No. 90/019,036 at 4.

⁶ Skagerwall at 1:9-13.

⁷ *Id.* at 4:12-20.

Skagerwall explains that its system resolves requests for data faster when organizing the directory servers according to a hash function. It states: “To avoid unnecessary look up operations for object tags at directory servers, as outlined above, it is preferred that object data is stored on directory servers according to the hashfunction.”⁸ This organization allows the system to determine which directory server contains the desired information based on the object tag (i.e., identifier) and quickly resolve the request.⁹

OracleNamesAdminGuide teaches location server networks with hierarchical, non-hierarchical, and mixed architectures. Like the ‘640 Patent, OracleNamesAdminGuide answers the question in a distributed data storage system: “where is my data?” OracleNamesAdminGuide describes a technique that separates the “where” (i.e., location of the data repository) from the “what” (i.e., the data itself).¹⁰ OracleNamesAdminGuide does this by using specialized servers called Names Servers, which serve as intermediaries between clients requesting data and the corresponding data storage servers.¹¹ A client seeking to locate data on OracleNamesAdminGuide’s network will query a Names Server to retrieve the data’s network address, then use that address to connect to the entity housing the data. OracleNamesAdminGuide illustrates this network architecture in Figure 3-11, shown below.¹²

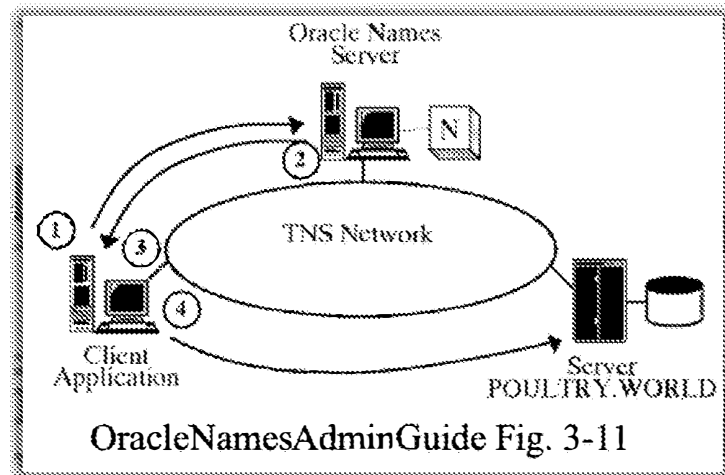
⁸ *Id.* at 10:39-40.

⁹ *Id.* at 9-18-31.

¹⁰ OracleNamesAdminGuide at 20, 23, 33, 43-44, 52-53.

¹¹ *See id.* at 20, 22.

¹² Steps 1-4 appear in OracleNamesAdminGuide’s original Figure 3-11.

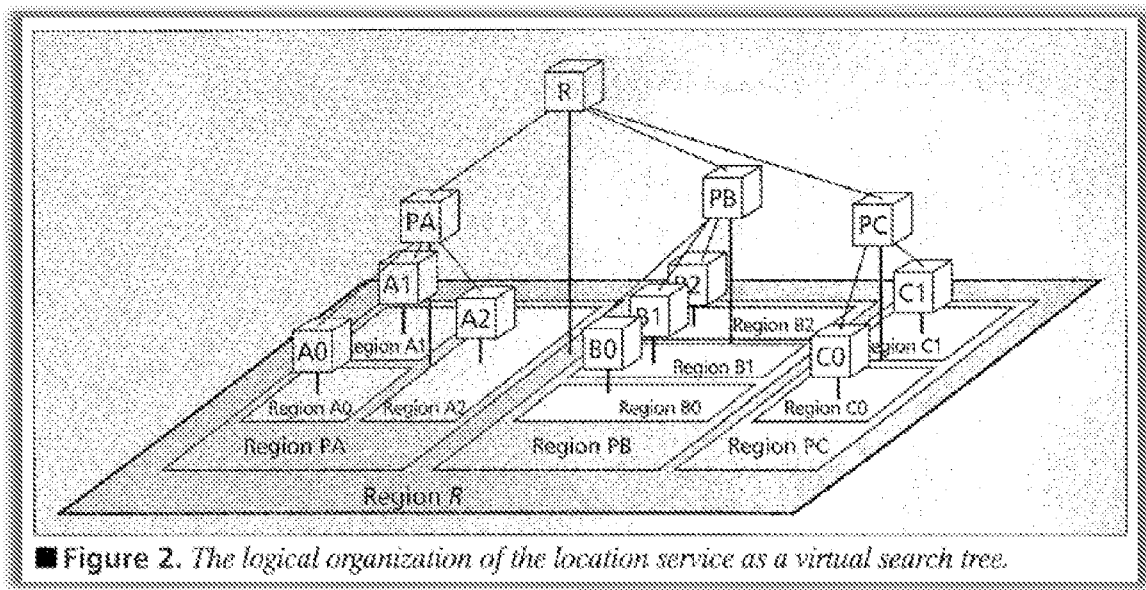


At step ①, the client submits a request for the data located at POULTRY.WORLD. OracleNamesAdminGuide uses its networking utility, SQL*Net, to send the client's request to the Names Server. At step ②, the Names Server receives the request, looks up the location information associated with the POULTRY.WORLD identifier, then sends that location information to client. The client receives the location information from the Names Server at step ③, then the client uses that location information to contact the server housing the database instance associated with POULTRY.WORLD. Finally, at step ④, the client contacts the server and establishes a client-server connection.¹³

Steen teaches the use of hierarchical, non-hierarchical, and mixed topologies in its location server networks. Like the '640 Patent, Steen describes a system for managing and locating data on a wide-area distributed network. As Steen states, its system "present[s] a scalable location service" to "optimize lookups."¹⁴ In Figure 2, Steen provides a two-level system for managing and locating an object's contact addresses.

¹³ See *id.* at 52-53; see also *id.* at 128-129 (describing location information syntax includes host name, port, and database instance along with the connection information).

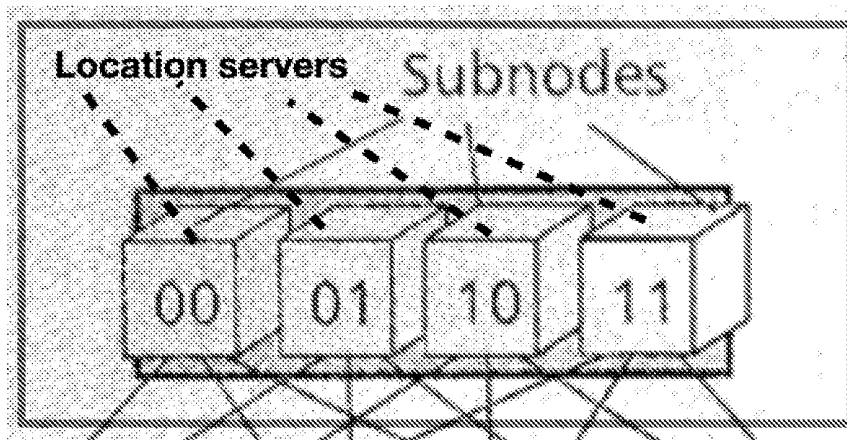
¹⁴ Steen at 104.



In the first level, Steen describes using a directory system for resolving an object name to an “object handle.”¹⁵ The object handle includes location information for a contact record. Then, in the second level, Steen teaches that the client/user uses the object handle to obtain object contact addresses.

In Steen’s top level, Steen teaches the use of a non-hierarchical architecture. Steen explains that a directory node $\text{dir}(R)$ stores object names (i.e., identifiers) associated with object handles (i.e., location information). Steen explains that these high-level servers can be partitioned into multiple subnodes, creating a non-hierarchical cluster of servers to handle a distributed workload, as shown below in annotated Figure 6.

¹⁵ *Id.* at Abstract.



Steen Figure 6 Annotated

For a partitioned $\text{dir}(\mathcal{R})$, each subnode is configured and functions as its own server, and contains its own subset of location information. Steen explains that it's important for each subnode to be "aware" of how the location information is distributed amongst the subnodes, allowing for requests to be re-routed to the subnode containing the correct location information. Steen provides an exemplary hashing method and explains that more complicated methods can be applied as the system scales up.¹⁶ Then, in the second level, Steen teaches that the client/user uses the object handle to obtain object contact addresses from a hierarchical location server network. Thus, Steen's teachings provide a non-hierarchical top layer, hierarchical bottom layer, and in their combination, a mixed topology location server network.

This Request presents substantial new questions of patentability ("SNQs") based on prior art that teaches the narrowed scope of the claims consistent with Patent Owner's arguments in the prior reexamination. Specifically, this Request details obviousness combinations of Skagerwall in view of Wolff (SNQ 1), OracleNamesAdminGuide in view of Steen and Oracle-Unleashed (SNQ 2), and Steen in view of Wolff and/or Skagerwall (SNQ 3). In SNQs 1 and 3,

¹⁶ *Id.* at 108.

Requester relies solely on new references—Skagerwall, Wolff, and Steen—that the Office didn’t consider during the original examination or the prior reexamination. And in SNQ 2, Requester relies on new references Steen and OracleUnleashed and new portions of OracleNamesAdminGuide that disclose a flat, non-hierarchical system.

Indeed, during the original examination and the prior reexamination of the ‘640 claims, the Examiner wasn’t presented with and didn’t consider the Skagerwall, Steen, OracleUnleashed, or Wolff references. Further, the Office wasn’t presented with and didn’t consider the OracleNamesAdminGuide reference during the original examination. And though the Office did consider portions of OracleNamesAdminGuide toward hierarchical configurations in the prior reexamination, it wasn’t presented with and didn’t consider OracleNamesAdminGuide’s teachings toward non-hierarchical configurations.

Accordingly, this Request raises SNQs for claims 17, 18, and 24.

II. SUBSTANTIAL NEW QUESTIONS OF PATENTABILITY

Requester respectfully requests reexamination for claims 17, 18, and 24 of the ‘640 Patent under 35 U.S.C. § 302 and 37 C.F.R. § 1.510.

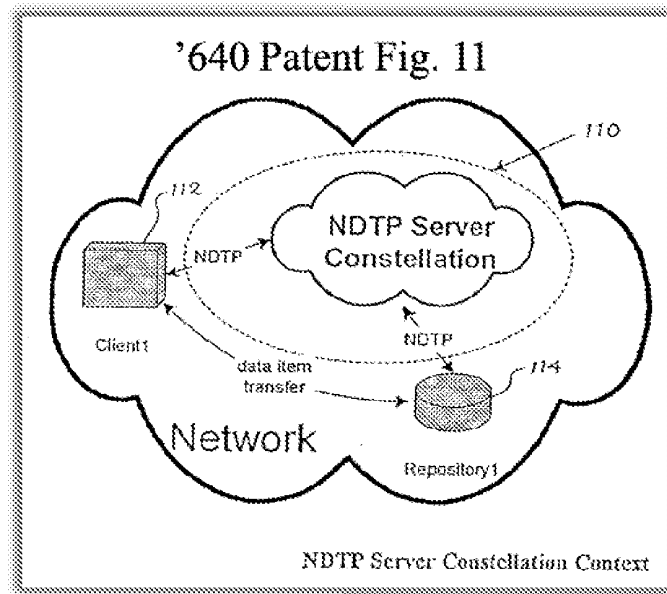
A. ‘640 Patent Overview

The ‘640 Patent was filed on September 13, 2000.¹⁷ The ‘640 Patent is directed to well-known techniques for locating data entities stored in a distributed networking environment.¹⁸ To

¹⁷ See ‘640 Patent at cover.

¹⁸ See *id.* at Abstract.

do so, the patent describes using an identifier associated with the location of a respective data entity.¹⁹ And the patent uses data location servers to store location information associated with those identifiers as shown below in Figure 11.²⁰



The '640 Patent states that it was known in the art that distributed databases allow physical storage of a database to exist across multiple locations and that, to access such data, a client would send a request to the server.²¹ And the patent acknowledges that hashing algorithms were known in the art to index location tables, such as the “hashpjw function presented by Aho, Sethi and Ullman in their text *Compilers, Principles, Techniques and Tools*.”²²

During the original examination, the Examiner twice rejected the claims based on prior art. This prior art showed it was known for a distributed data system to have a client requesting

¹⁹ See *id.* at Abstract, 2:12-27.

²⁰ See *id.* at 16:53-17:7.

²¹ See *id.* at 1:34-36, 1:56-65.

²² *Id.* at 15:1-20; Greene Decl. (Ex. 19) ¶ 20.

the location of data associated with an identifier, and a location server retrieving the location information for the entity based on the identifier and sends that information to the client.²³ And this prior art showed it was known that “use of a [] hashing operation in a computer networking environment was well known in the art at the time the invention was made.”²⁴

In response, Applicant added to the claims the limitation that, if the location server lacks location information for the requested data, the location server will send a *redirect message* to the client.²⁵ The amended claims further recite that the redirect message includes information useable by the client to locate another location server having the location information for the request entity.²⁶

Below, Requester establishes that the limitations of issued claims 17, 18, and 24 were already known in the art, as shown by OracleNamesAdminGuide’s distributed database storage system and other art.

B. The Challenged Claims Aren’t Entitled To Claim Priority To The Filing Date Of The ‘896 Provisional Application.

Claims 17, 18, and 24 aren’t entitled to the benefit of the July 8, 1998 filing date of U.S. Pat. Application No. 09/111,896 (the “‘896 Application”). 35 U.S.C §120 (pre-AIA) only grants this benefit when the subject matter of the claimed invention is disclosed by the prior application “in the manner provided by §112(a).” MPEP 2163 discloses that, “[t]o satisfy the [§112] written description requirement, a patent specification must describe the claimed invention in sufficient detail that one skilled in the art can reasonably conclude that the inventor had possession of the

²³ See ‘640 History at 440-449, 488-498 (12/25/2004, 02/09/2005 Office Actions).

²⁴ See *id.* at 447-448 (02/24/2004 Office Action).

²⁵ See *id.* at 514-526 (08/09/2005 Amendments).

²⁶ See *id.*

claimed invention.” As detailed below, the ‘896 Application doesn’t provide support under §112 for at least the following six limitations of ‘640 claim elements 17.c, 17.f.2, 18.d, 18.g, and 18.h, and claim 24:

1. A location string specifying data entity location, as required by elements 17.c, 18.d, and 18.h;²⁷
2. The contents of the location servers be indexed, as required by claim 24;²⁸
3. The contents of the location servers be organized according to a hash table, as required by claim 24;²⁹
4. Each location server having information to calculate which location server contains the requested location information, as required by claim elements 17.f.2 and 18.h;³⁰
5. Returning/sending a redirect message if the location server lacks the requested location information, as required by elements 17.f.2 and 18.h;³¹ and
6. The redirect message comprises information for calculating the location of a location server known to have the requested location information, as required by claim elements 17.f.2 and 18.h.³²

Thus, the ‘896 Application fails to disclose at least the above subject matter, and claims 17, 18, and 24 aren’t entitled to the benefit of the July 8, 1998 filing date of the ‘896 Application. As detailed in Section (III.C), even if the Challenged Claims are entitled to their earliest priority date (September 14, 1999), Skagerwall, Wolff, Steen, and OracleNamesAdminGuide, and OracleUnleashed all predate the ‘640 Patent and qualify as prior art.

²⁷ See Greene Decl. ¶¶ 42-43.

²⁸ *Id.*

²⁹ *Id.*

³⁰ *Id.*

³¹ *Id.*

³² *Id.*

C. Listing of Prior Art Patents and Printed Publications

Pursuant to 37 C.F.R. § 1.510(b)(3), this Request provides a copy of every patent or printed publication relied upon or referred to herein:

- **Exhibit 2:** *Oracle Names Administrator's Guide, Release 2.0* (Oracle Corp., Redwood City, CA), published 1996 ("OracleNamesAdminGuide");
- **Exhibit 3:** Joseph B. Greene, *Oracle 8 Server: Unleashed*, 1st ed. (Sams Publishing, Indianapolis, Ind.) ISBN: 0672312076, January 1998 ("OracleUnleashed");
- **Exhibit 4:** Maarten van Steen, et al., "Locating Objects in Wide-Area Systems," IEEE Communications Magazine, Vol. 36, Issue 1, pp. 104-09, ISBN: 0163-6804 (January 1998) ("Steen");
- **Exhibit 5:** U.S. Pat. No. 6,473,781 to Roger Skagerwall et al. ("Skagerwall"); and
- **Exhibit 6:** U.S. Pat. No. 6,185,601 to James J. Wolff ("Wolff").

A Form SB-08 and copies of the foregoing references are submitted herewith.

D. Statement Setting Forth Each Substantial New Question of Patentability

For claims 17, 18, and 24, this Request presents a substantial new question of patentability based on obviousness under 35 U.S.C. § 103 over Skagerwall in view of Wolff ("SNQ 1"), OracleNamesAdminGuide in view of Steen and/or OracleUnleashed ("SNQ 2"), and Steen in view of Wolff and/or Skagerwall ("SNQ 3").

Requester submits that the prior art references raise a new "substantial question of patentability" because "the teaching of the (prior art) patents and printed publications is such that a reasonable examiner would consider the teaching to be important in deciding whether or not the claim is patentable."³³ For example, these references, discussed in further detail in Section III,

³³ MPEP 2242.

when considered as a combination in view of their shared teachings, teach each limitation of the claims, including:

- OracleNamesAdminGuide discloses, or renders obvious in view of either Steen or OracleUnleashed, claim 17's "in response to receiving a data location request from a client to retrieve a location string associated with an identification string provided in the data location request, transmitting a redirect message to the client if the identification string is not associated with a location string at the data location server, wherein the redirect message contains information for use by the client to calculate a location of a different data location server in the plurality of data location servers, wherein the different data location server contains the location string," and a similar limitation in claim 18;
- OracleNamesAdminGuide discloses, or renders obvious in view of either Steen or OracleUnleashed, claim 24's "the location information comprises a portion of a hash table distributed over the plurality of data location servers."
- Skagerwall discloses, or renders obvious in view of Wolff, claim 17's "in response to receiving a data location request from a client to retrieve a location string associated with an identification string provided in the data location request, transmitting a redirect message to the client if the identification string is not associated with a location string at the data location server, wherein the redirect message contains information for use by the client to calculate a location of a different data location server in the plurality of data location servers, wherein the different data location server contains the location string," and a similar limitation in claim 18;
- Skagerwall discloses, or renders obvious in view of Wolff, claim 24's "the location information comprises a portion of a hash table distributed over the plurality of data location servers."
- Steen discloses, or renders obvious in view of Skagerwall and/or Wolff, claim 17's "in response to receiving a data location request from a client to retrieve a location string associated with an identification string provided in the data location request,

transmitting a redirect message to the client if the identification string is not associated with a location string at the data location server, wherein the redirect message contains information for use by the client to calculate a location of a different data location server in the plurality of data location servers, wherein the different data location server contains the location string,” and a similar limitation in claim 18; and

- Steen discloses, or renders obvious in view of Skagerwall and/or Wolff, claim 24’s “the location information comprises a portion of a hash table distributed over the plurality of data location servers.”

Further, these prior art references are new because the “same question of patentability as to the claim has not been decided by the Office in an earlier concluded examination or review of the patent.”³⁴ None of these references were raised, considered, or formed the basis of a rejection during the original prosecution of the ‘640 Patent. And only OracleNamesAdminGuide was raised or considered during the prior reexamination of the ‘640 Patent.

Overall, this Request provides the detailed mapping and explanation regarding the combination of references and raises a SNQ because the Office hasn’t previously considered the teaching of the combinations of references in SNQ 1, SNQ 2, or SNQ 3.

Further, Mr. Greene’s declaration (Ex. 19) reinforces the Request’s analyses. Mr. Greene has more than 30 years of professional experience implementing, managing, and analyzing Oracle products. Based on this experience, he has published six books on Oracle’s database management system and its associated tools and utilities, including OracleUnleashed.³⁵ Mr. Greene concludes that the combinations of OracleNamesAdminGuide in view of OracleUnleashed and/or

³⁴ *Id.*

³⁵ A full description of Mr. Greene’s professional background and qualifications can be found at paragraphs 4-13 of his declaration.

Steen, Skagerwall in view of Wolff, and Steen in view of Skagerwall and/or Wolff render obvious claims 17 and 18, and claim 18's dependent claim 24.³⁶

E. Co-Pending Litigation

Requester is currently engaged in a pending litigation concerning the '640 Patent, namely *Kove IO, Inc. v. Amazon Web Services, Inc.*, 1:18-cv-08175 (N.D. Ill.). Requester is unaware of any pending prosecution concerning the '640 Patent.

In Reexamination Control No. 90/014,552, Aneeta Mishra, who has no relationship with Requester, filed a request for *ex parte* reexamination of claims 17, 18, and 24 of the '640 Patent. The Office denied that request on September 9, 2020. None of the prior art references presented in this Request were before the Office in the 90/014,552 reexamination.

In Reexamination Control No. 90/019,036, Requester Amazon Web Services, Inc. (i.e., the same requester here) filed a request for *ex parte* reexamination of claims 17, 18, and 24 of the '640 Patent. The Office granted the request on December 23, 2021. The Reexamination Specialist issued an Office Action on May 4, 2022 rejecting claims 17, 18, and 24. Patent Owner filed a reply to that Office Action on August 4, 2022. Seeking to distinguish the claims over the prior art, Patent Owner limited the scope of the claims to a system of location servers organized in a non-hierarchical cluster structure, wherein each location server must be able to respond to any location request with either the location of the data or the location of a server known to contain the requested location data without an intervening traversal, and wherein each request must be resolved in two or fewer traversals.³⁷ Patent Owner then distinguished that narrowed scope from

³⁶ See Greene Decl. ¶¶ 40, 134, 194, and 367.

³⁷ See 08/04/2022 Patent Owner's Reply to Office Action in Reexamination Control No. 90/019,036 at 10, 23-25.

the presented primary prior art reference, OracleNamesAdminGuide, asserting that the Oracle system uses messages that can only traverse up and down a hierarchy of location servers.³⁸ Building on that incorrect premise, Patent Owner contended that Oracle’s system can’t meet the “redirect message” and “hash” limitations recited in the challenged claims.³⁹ Based on those arguments, the Reexamination Specialist issued a Notice of Intent to Issue *Ex Parte* Reexamination Certificate on September 22, 2022. And on December 1, 2022, the Office issued an *Ex Parte* Reexamination Certificate confirming claims 17, 18, and 24.

Other than Reexamination Control No. 90/019,036, Requester is unaware of any other pending prosecution concerning the ‘640 Patent.

Patent Owner also holds U.S. Patent No. 7,233,978 (“’978 Patent,” Ex. 7), which shares many elements of the ’640 Patent. In IPR2020-00276, Requester filed a petition seeking *inter partes* review of ’978 claims 1, 3, 6, 10, 14, 17, 23, 24, 30, and 31. The Board denied this petition on June 16, 2020 in Paper 23. None of the prior art references presented in this Request were before the Board in IPR2020-00276.⁴⁰ In Reexamination Control Nos. 90/019,034 and 90/019,109, Requester challenged a subset of claims of the ’978 Patent. The ’034 Reexamination resulted in a Final Office action cancelling claims 1 and 31 of the ’978 Patent, while claims 17, 23, 24, and 31 are currently under review in the ’109 Reexamination. Steen, Skagerwall, and

³⁸ *Id.*

³⁹ *Id.* at 42-43.

⁴⁰ *Contra In re Vivint, Inc.*, 2021 WL 4448620, at *2, *8 (Fed. Cir. Sept. 29, 2021) (ordering USPTO to dismiss an *ex parte* reexamination request that simply “repackaged” arguments raised in a prior IPR petition). In the instant Request, none of the relied upon prior art references or obviousness arguments were presented previously in an IPR or reexamination challenging the ‘640 Patent or any related patent.

Wolff weren't considered or and didn't form part of any SNQ in either the '034 or '109 Reexamination.

F. New and Non-Cumulative Prior Art

This Request raises new and non-cumulative questions of patentability. This Request bases each SNQ on one or more prior art references and obviousness combinations that the Office hasn't reviewed in a previous proceeding or examination.

In Reexamination Control No. 90/019,109 of the '978 Patent, the Office confirmed that a request like this one, presenting both old and new prior art references, contained proper SNQs and wasn't cumulative of prior proceedings before the Office. Patent Owner had petitioned for termination of that reexamination. Patent Owner argued that, because the request used one prior art reference from a previous reexamination, it should be denied as cumulative under §325(d) and the *In Re Vivint* decision.⁴¹ The Office determined that the three new references provided substantial new material to consider and denied Patent Owner's petition for termination.⁴² For the same reasons set forth in that decision, the instant Request contains new and non-cumulative art that should be considered, and any petition for termination should be denied.

The decision denying the petition in the '109 Reexamination applied both Federal Circuit precedent and the MPEP, which establish that an *ex parte* reexamination request may be denied if it's based on "old art."⁴³ The MPEP defines "old art" as references "previously cited/considered in an earlier concluded Office examination of the patent."⁴⁴ And the Federal Circuit explained in *In re Hiniker* that a reexamination request based on "old art" could only be denied

⁴¹ 10/06/2022 Petition to the Director to Terminate Reexamination Control Number 09/019,109 at 1-2.

⁴² 11/22/2022 Order Instituting Reexamination Control No. 09/019,109 at 26-32.

⁴³ MPEP 2258.01; *In re Hiniker Co.*, 150 F.3d 1362, 1365-66 (Fed. Cir. 1998).

⁴⁴ MPEP 2258.01.

when that previously cited art is the sole basis for the proposed substantial new question of patentability.⁴⁵ Thus, as the *Hiniker* decision explains, a request basing an obviousness combination on one new reference with two pieces of “old art” still raises a substantial new question of patentability:

The examiner and the Board here applied East [reference] in finding the claims under reexamination obvious. East was never before the examiner during the original prosecution and is thus new art. There is no indication that East ... was not material to the question of obviousness vel non or that it was cumulative with any old art. Thus, both the examiner’s and the Board’s decisions were based on a substantial new question of patentability.⁴⁶

This Request bases each SNQ on an obviousness combination of one or more prior art references that the Office hasn’t previously reviewed. Each of the references in SNQs 1 and 3 (Skagerwall, Wolff, and Steen) hasn’t been cited or considered by the Office in any previous proceeding in connection with the ’640 Patent. For SNQ 2, the Office has previously reviewed the OracleNamesAdminGuide reference (Ex. 5), finding it discloses elements of the challenged claims.⁴⁷ But SNQ 2 relies on new references OracleUnleashed and Steen in combination with OracleNamesAdminGuide to establish the challenged claims are obvious. Accordingly, this Request doesn’t base any SNQ on old art.

Further, this Request’s new references aren’t cumulative of any old art. Under the MPEP, prior art is cumulative only if it “substantially reiterates verbatim the teachings of a reference”

⁴⁵ *In re Hiniker Co.*, 150 F.3d at 1367.

⁴⁶ *Id.*

⁴⁷ 05/04/2022 Non-Final Office Action in Reexamination Control No. 90/019,036 at 3.

that the Office previously considered “even though the title or the citation of the reference may be different.”⁴⁸ As the MPEP further states, denials based on cumulative prior art are rare:

However, it is expected that a repetitive reference which cannot be considered by the Office during reexamination will be a rare occurrence since most references teach additional information or present information in a different way than other references, even though the references might address the same general subject matter.⁴⁹

None of the teachings of Skagerwall, Steen, Wolff, or OracleUnleashed are cumulative of previously considered art. Skagerwall, Steen, and Wolff have no connection to art that the Office considered in prior proceedings. Further, while Mr. Greene wrote both the OracleUnleashed reference at issue in this Request and the OracleSG reference at issue in Reexamination Control No. 90/019,036, OracleUnleashed describes system configurations and features beyond those taught in the secondary references (e.g., OracleSG) in the prior reexamination. For example, OracleUnleashed describes features that improve performance, including improved parallelism, partitioning, and optimization, which also mirror the teachings in the '640 Patent. OracleUnleashed further teaches system configurations for updated versions of Oracle's product—through Oracle version 8.1, compared to Oracle7.2 for OracleSG. Thus, OracleUnleashed doesn't “substantially reiterate[] verbatim” the teachings of art previously considered by the Office. As a result, Skagerwall, Steen, Wolff, and OracleUnleashed “teach additional information or present information in a different way than other references,” and thus aren't cumulative under the MPEP's guidance.

⁴⁸ MPEP 2258.01.

⁴⁹ *Id.*

1. *In re Vivint* Isn't Applicable Here.

This Request is also consistent with *In re Vivint, Inc.*⁵⁰ In that decision, the Federal Circuit vacated an *ex parte* reexamination, reasoning that it was improper for the Office to deny an IPR for relying on previously-considered prior art but later grant a reexamination request based on that same art.⁵¹ As a result, the court found that the reexamination request should have been denied because it copied “vast swaths” of the already abusive previous petition, using three of the same references, and two of the same combinations.⁴⁸ But unlike the request in *Vivint*, this Request relies on the Skagerwall, Steen, Wolff, and OracleUnleashed new prior art references and three distinct SNQs that have never been reviewed or determined to be abusive by the Office. As a result, this Request shouldn't be rejected under the guidance of *Vivint*.

Indeed, the Federal Circuit stated that *Vivint*'s “holding ... is narrow,” and requires only that “the Patent Office, when applying § 325(d), cannot deny institution of an IPR based on abusive filing practices and then grant a nearly identical reexamination request that is even more abusive.”⁵² Unlike *Vivint*, the Office has never previously denied any IPR petition or other challenge to the '640 Patent on the basis of undesirable, incremental, or abusive petitioning under § 325(d). Further, this Request doesn't contain “nearly identical” grounds to any previous challenge. Each SNQ contains new references and obviousness arguments that the Office hasn't considered in any proceeding. While SNQ 2 includes OracleNamesAdminGuide that the Office has previously considered, SNQ 2 relies on Steen and OracleUnleashed that are new references and form new obviousness arguments that the Office hasn't previously considered. Thus, *Vivint* doesn't apply here.

⁵⁰ 14 F.4th 1342 (Fed. Cir. 2021).

⁵¹ *Id.* at 1347.

⁵² *Id.* at 1342.

Also, this Request’s SNQs apply new prior art that discloses limitations that the Reexamination Specialist determined were missing from the SNQs raised in the prior Reexamination Control Nos. 90/014,552 and 90/019,036. Both the prior art and arguments presented in this Request aren’t the same or substantially the same as those previously presented in Reexamination Control Nos. 90/014,552 and 90/019,036.

To the extent Patent Owner asserts this Request relies on prior Office decisions as a “roadmap” to overcome previous deficiencies, this assertion lacks merit. The Office has found that concerns regarding a party making serial filings to cure deficiencies noted in prior filings relate to the Board’s goal of ensuring the efficient use of the IPR review process—not toward *ex parte* reexamination proceedings.⁵³

2. *Becton, Dickinson* Isn’t Applicable Here.

The Office has stated that it need not consider the *Becton, Dickinson* factors relating to cumulativeness for *ex parte* reexaminations:

The [*Becton, Dickinson*] factors are considered by the Board when determining whether to institute a trial. When determining whether to order *ex parte* reexamination, however, the Office may not necessarily consider these factors. An *ex parte* reexamination proceeding is not a trial proceeding, and the considerations with respect to issues involving 35 U.S.C. § 325(d) may differ due to the different nature of an *ex parte* reexamination proceeding.⁵⁴

⁵³ See, e.g., *General Plastic Industrial Co., Ltd. v. Canon Kabushiki Kaisha*, IPR2016-01357, Paper 19 at 17 (P.T.A.B. Sept. 6, 2017); 11/22/2022 Order Granting Request for *Ex Parte* Reexamination in Reexamination Control. No. 90/019,109 at 33-34.

⁵⁴ PTAB Consolidated Trial Practice Guide (Nov. 2019) at 63); see *ex parte Briggs*, Reexamination No. 90/014,813, Decision on Petitions at 6 (quoting Trial Practice guide in finding “*Becton* factors were specifically formulated to apply to AIA trial proceedings, not to *ex parte* reexamination proceedings”).

If the Office nonetheless were to consider the *Becton, Dickinson* factors, those factors don't support discretionarily denying the instant request.⁵⁵ The Board divides the factors into “a two-part framework, first considering *Becton, Dickinson* factors (a), (b), and (d) to determine whether the same or substantially the same art or arguments were previously presented to the Office, and if so, evaluating *Becton, Dickinson* factors (c), (e), and (f) to determine whether [a p]etitioner has demonstrated that the Office erred in a manner material to the patentability of challenged claims.”⁵⁶

Here, the first part's factors all weigh against exercising discretion under § 325(d) because every ground of the request raises prior art and obviousness arguments that aren't the same or substantially the same as those previously considered by the Office. As the *Amazon.com, Inc. v. M2m Solutions LLC* decision explains, “[t]he Board frequently holds that a reference that ‘was neither applied against the claims nor discussed by the Examiner does not weigh in favor of exercising the Board's discretion under § 325(d) to deny a petition.’”⁵⁷ Consistent with that, the Office

⁵⁵ Specifically, the factors include: “(a) the similarities and material differences between the asserted art and the prior art involved during examination; (b) the cumulative nature of the asserted art and the prior art evaluated during examination; (c) the extent to which the asserted art was evaluated during examination, including whether the prior art was the basis for rejection; (d) the extent of the overlap between the arguments made during examination and the manner in which Petitioner relies on the prior art or Patent Owner distinguishes the prior art; (e) whether Petitioner has pointed out sufficiently how the Examiner erred in its evaluation of the asserted prior art; and (f) the extent to which additional evidence and facts presented in the Petition warrant reconsideration of the prior art or arguments.” *Becton*, No. IPR2017-01586, Paper 8 at 17-18.

⁵⁶ *Solvay USA Inc. v. Worldsource Enters., LLC*, No. IPR2020-00768, Paper 12 at 7 (PTAB Sept. 25, 2020) (citing *Advanced Bionics LLC v. MED-EL Elektromedizinische Geräte GmbH*, IPR2019-01469, Paper 6 at 7-11 (PTAB Feb. 13, 2020) (precedential)).

⁵⁷ No. IPR2019-01205, Paper 14 at 16 (PTAB Jan. 27, 2020) (quoting *Zip-Top LLC v. Stasher, Inc.*, IPR2018-01216, Paper 14 at 35-36 (PTAB Jan. 17, 2019), and citing *Shenzhen Zhiyi Tech. Co. v. iRobot Corp.*, IPR2017-02137, Paper 9 at 9-10 (PTAB Apr. 2, 2018)).

routinely rejects arguments that factors (a), (b), and (d) favor exercising discretion to deny a challenge that, like the request here, relies on new prior art.⁵⁸

And when a challenge presents some overlapping prior art, the Office’s analysis of factors (a), (b), and (d) gives particular weight to whether newly cited art is material to limitations found to be missing from prior art addressed during prosecution or a previous challenge. For instance, in *Ty Inc. v. Softbelly’s, Inc.*, the Board considered an IPR petition raising invalidity grounds where the patent owner had established during a previous reexamination that the cited prior art didn’t teach “a wiping tool made from two different fabrics.”⁵⁹ Because the IPR petition relied on a new reference that “teaches the specific element the Board found missing” in the art cited during reexamination, the Board concluded “the arguments presented in the [p]etition are not substantially the same as those raised in reexamination.”⁶⁰ For this reason, the Board found it “need not reach the issue of error” under the second part of the *Becton, Dickinson* framework.⁶¹ Here, too, there’s no basis to reach the second part of the *Becton, Dickinson* framework, as the Request’s art and obviousness arguments aren’t substantially the same as those raised in any prior proceeding or examination of the ’640 Patent.

And the new references “teach the specific element[s] ... found missing” in the prior Request. While one of this Request’s SNQs relies on the OracleNamesAdminGuide reference that

⁵⁸ See, e.g., *Apple Inc. v. Corephotonics, Ltd.*, No. IPR2020-00487, Paper 13 at 24 (PTAB Nov. 5, 2020) (finding that factors (a), (b), and (d) didn’t support exercising discretion to deny IPR, as new reference was “not cumulative to the references considered and applied during prosecution of the [patent at issue] because it is cited ... to teach the features that led to allowance”) (citing *Oticon Medical AB v. Cochlear Ltd.*, IPR2019-00975, Paper 15 at 20 (PTAB Oct. 16, 2019) (precedential)); *Amazon*, IPR2019-01205, Paper 14 at 15 (PTAB Jan. 27, 2020) (rejecting argument that all “*Becton* factors counsel in favor of denying [IPR] under § 325(d)”).

⁵⁹ No. IPR2020-00689, Paper 10 at 13 (PTAB Sept. 15, 2020).

⁶⁰ *Id.* at 13-14.

⁶¹ *Id.*; *accord Vestas-Am.*, No. IPR2018-01015, Paper 9 at 23.

was previously presented in the Reexamination Control No. 90/019,036, it also relies on newly presented references Steen and OracleUnleashed. As explained above, both Steen and OracleUnleashed disclose teachings that render obvious the limitations in claims 17 of “transmitting a redirect message to the client if the identification string is not associated with a location string” and “wherein the redirect message contains information for use by the client to calculate a location of a different data location server”; and in claims 18 of “if the data location server does not contain the location string associated with the identification string, the location server transmits a redirect message to the client” and “wherein the redirect message contains redirect information for use by the client to calculate a location of a different data location server in the plurality of data location servers.” The Reexamination Specialists identified those limitations as being patentable over the art of record in the Reexamination Control Nos. 90/014,552 and 90/019,036.⁶² In SNQs 1 and 3, the new references Skagerwall, Steen, and Wolff also address those same claim limitations. Thus, each of this Request’s SNQs apply new prior art that discloses limitations the specialists determined were missing from the SNQs raised in the prior Reexamination Control Nos. 90/014,552 and 90/019,036.

To the extent Patent Owner asserts this Request should be denied because it relies on prior art that could have been raised in a prior proceeding, that’s incorrect. The Office has found that concerns over whether a party could have relied on prior art in a prior proceeding are relevant to the Board’s authority under 35 U.S.C. § 314(a) for discretionary denials of IPRs—not to

⁶² 09/29/2022 Supplemental Notice of Intent to Issue *Ex Parte* Reexamination Certificate in Reexamination Control No. 90/019,036 at 3; 08/10/2020 Order Denying Request for *Ex Parte* Reexamination in Reexamination Control No. 90/014,552 at 19-20, 22-23, 29-33.

ex parte reexaminations.⁶³ Similarly, the Office has found such concerns are relevant to estoppel limitations under §§ 315(e) and 325(e) stemming from final written decisions issued in IPRs and PRGs—not to *ex parte* reexaminations.⁶⁴

Accordingly, SNQs 1-3 are new and non-cumulative because they rely on prior art and arguments that aren't the same or substantially the same as those previously presented in Reexamination Control Nos. 90/014,552 and 90/019,036.

G. Estoppel

The statutory estoppel provisions of 35 U.S.C. § 315(e)(1) and 35 U.S.C. § 325(e)(1) don't prohibit Requester from filing this *ex parte* reexamination request.

III. DETAILED EXPLANATION OF THE PERTINENCE AND MANNER OF APPLYING THE PRIOR ART REFERENCES TO EVERY CLAIM FOR WHICH REEXAMINATION IS REQUESTED

In accordance with 37 C.F.R. § 1.510(b)(2), Requester provides the following detailed explanation of the pertinence and manner of applying the prior art to the Challenged Claims.

A. Level of Skill in the Art

At the time of the purported invention, a POSITA would have had a bachelor's degree in computer science (or a similar field) and at least two years of experience working with database management software systems. Alternatively, a POSITA would have had a Master's degree or similar post-graduate studies in the field of database management software systems with fewer years of experience.⁶⁵

⁶³ See, e.g., *General Plastic Industrial Co., Ltd. v. Canon Kabushiki Kaisha*, IPR2016-01357, Paper 19 at 9 (P.T.A.B. Sept. 6, 2017); 11/22/2022 Order Granting Request for *Ex Parte* Reexamination in Reexamination Control. No. 90/019,109 at 36-37.

⁶⁴ See, e.g., *General Plastic Industrial Co., Ltd. v. Canon Kabushiki Kaisha*, IPR2016-01357, Paper 19 at 9 (P.T.A.B. Sept. 6, 2017); 11/22/2022 Order Granting Request for *Ex Parte* Reexamination in Reexamination Control. No. 90/019,109 at 36-37.

⁶⁵ See Greene Decl. ¶ 31.

B. Claim Construction

In the pending district court matter, Patent Owner has stated that the '640 Patent expired on April 8, 2020.⁶⁶ Because the '640 Patent has expired, the claims should be given their ordinary and customary meaning as set forth in *Phillips v. AWH Corporation*. The MPEP states:

During reexamination . . . claims are given the broadest reasonable interpretation consistent with the specification and limitations in the specification are not read into the claims.... In a reexamination proceeding involving claims of an expired patent, claim construction pursuant to the principle set forth by the court in *Phillips* ... (words of a claim 'are generally given their ordinary and customary meaning' as understood by a person of ordinary skill in the art in question at the time of the invention) should be applied since the expired claim are not subject to amendment."⁶⁷

Thus, this Request applies the *Phillips* claim construction standard.

In the district court matter, Patent Owner proposed constructions for certain claim terms and agreed to constructions for other terms.⁶⁸ In this reexamination, Requester shows the prior art invalidates the Challenged Claims using the following constructions proposed or adopted by Patent Owner:

- "location" means "an encoding that is a member of a set of associations with an identifier in a location server, and that specifies where data pertaining to the entity identified by the identifier is stored" (claims 17-18, 24);
- "location server" means "a network-attached component that maintains a set of identifier/location mappings that are modified or returned in response to location request messages from clients" (claims 17-18, 24);

⁶⁶ See Ex. 16 (Memo in Opposition) at 8.

⁶⁷ MPEP 2258 (citing *In re Yamamoto*, 740 F.2d 1569 (Fed. Cir. 1984) and *Phillips v. AWH Corp.*, 415 F.3d 1303, 1316 (Fed. Cir. 2005)).

⁶⁸ See Ex. 17 (Joint Claim Construction Chart) at 4-6.

- “location information” means “information pertaining to one or more locations of data and/or the identities of one or more location servers” (claims 17-18, 24);
- “identifier”/ “identifier string” / “identification string” means “a unique encoding that identifies an individual entity, and with which zero or more location strings are associated in a location server” (claims 17-18);
- “client” means “a network-attached component that initiates update or lookup of identifier/location mappings from a location server with location request messages” (claims 17-18, 24);
- “hash table” means “a data structure that stores values in a table, where values are stored and retrieved by applying a hash function to an input and using the function result as an index into the table” (claim 24); and
- “data location sewer network” means “data location server network” (claim 17).⁶⁹

Additionally, in the related district court, the court issued the following constructions:⁷⁰

Claim Term	Construction
“location information”	‘978: “one or more identifiers and their associated locations” ‘170/‘640: “information pertaining to one or more locations of data and/or the identities of one or more location servers”
“location server”	“a network-attached component that maintains a set of identifier/location mappings that are modified or returned in response to location request messages from clients”
“client”	“a network-attached component (which may be software or hardware) that initiates update or lookup of identifier/location mappings from a location server with location request messages”
“based on a hash function used to organize the data location information across the plurality of data location servers . . . based on the hash function applied to the identifier string”	“the portion of the data location information included in a corresponding one of the data location servers is based on a hash function that maps identifier strings to one or more of the data location servers, and each one of the data location servers is configured to determine the at least one of the plurality of data location servers based on the hash function applied to the identifier string”
“data pertaining to the entity”	“data pertaining to a person or thing (real, digital, or abstract) distinct from that data”

⁶⁹ *Id.*

⁷⁰ See Ex. 25 (Claim Construction Order) at 35.

These constructions are nearly identical to the above-detailed constructions that Patent Owner proposed or agreed to, with no substantive differences that would impact the below mapping of the prior art to the Challenged Claims.

Additionally, in the related district court case, Requester is seeking further claim construction based on Patent Owner's statements narrowing the scope of the claims in the prior reexaminations of the '640 and related patents. Those proposed constructions are:⁷¹

APPENDIX A		
Claims	Terms	Proposed Constructions
'640: 17, 18, 24 '170: 1, 2, 6, 8, 9, 12, 15 '978: 1, 3, 6, 10, 14, 17, 23, 24, 30, 31	"network" / "system"	{network/system} arranged in a non-hierarchical, cluster topology
'640: 17, 18, 24 '170: 1, 2, 6, 8, 9, 12, 15 '978: 1, 3, 6, 10, 14, 17, 23, 24, 30, 31	"location server"	a network-attached component that maintains, for satisfying every request for the location of data on the network, a set of at least one of (1) identifier/location mappings or (2) information about the location of such mappings that are modified or returned in response to all location request messages from clients
'640: 17, 18 '170: 1, 2, 5, 6, 8, 9, 12 '978: 1, 3, 6, 10, 17, 23, 24, 30, 31	"providing" / "determin[e]ing" / "retrieving" location information or servers	providing, without performing any request iterations, ... location information / determin[e]ing, without performing any request iterations, ... location servers / retrieving in two or fewer request iterations location information
'640: 17, 18 '170: 6, 15 '978: 3, 10, 14	"return" / "sending" / "transmit[s]ing" a redirect message	return / sending / transmit[s]ing a redirect message without performing any request iterations

The district court has announced that it will consider these proposed claim constructions during summary judgment briefing.⁷² Here, and as detailed more specifically below, the Requester provides prior art that invalidates the Challenged Claims under the narrowed claim scope reflected in these proposed constructions.

⁷¹ Ex. 26 (Dkt. No. 539 at Appendix A).

⁷² Ex. 27 (Dkt. No. 580).

Requester doesn't concede the scope of any claim term in litigation or waive any argument in any proceeding that the claim terms are indefinite or the claims are invalid or unpatentable. For the purposes of this proceeding, Requester construes all other terms as having their ordinary and customary meanings. As shown below in Sections III.D-III.F, the prior art renders obvious the Challenged Claims under any reasonable construction of the claim terms.

C. Overview of the Prior Art

This Request establishes that the Challenged Claims are obvious based on the teachings of OracleNamesAdminGuide, Steen, Skagerwall, Wolff, and OracleUnleashed. During the original examination of the '640 Patent, the Examiner did not cite, consider, or rely on any of the references.

1. Skagerwall (Ex. 2) Overview

U.S. Patent No. 6,473,781 ("Skagerwall") is titled "Communication System and Method." Skagerwall issued on October 29, 2002 from an application filed on October 27, 1998. Skagerwall is prior art under at least 35 U.S.C. §§ 102(a) and (b) because it was filed more than one year before the '640 Patent's filing date of September 13, 2000. Further, even if the '640 Patent were entitled to priority to the Provisional filing date of September 14, 1999, Skagerwall would still pre-date this by 11 months. During the original examination of the '640 Patent, the Examiner didn't cite or rely on Skagerwall's disclosures in any Office action.

Skagerwall teaches a method for optimizing performance of a class of servers based on specific user goals: "The present invention relates to a system and method of communicating information in a network, and in particular, the present invention relates to a system for communicating information, related to a plurality of tagged objects, in a network."⁷³

⁷³ Skagerwall at 1:9-13.

Skagerwall describes that its system associates data with an object tag, and stores that data on application servers.⁷⁴ Skagerwall explains that a plurality of directory servers are a “key resource” for retrieving, i.e., locating data, stored based on the object tag, i.e., identifier:

Advantageously, each of a plurality of directory servers stores part of an arbitrarily scaleable data base containing object data associating each of the objects with at least one of the application data packets. The plurality of directory servers serve as key resource in retrieving an application data packet related to an object. Means for receiving a read object tag from a reading means and for identifying the directory server for storing object data related to the read object tag and retrieving the object data are provided.⁷⁵

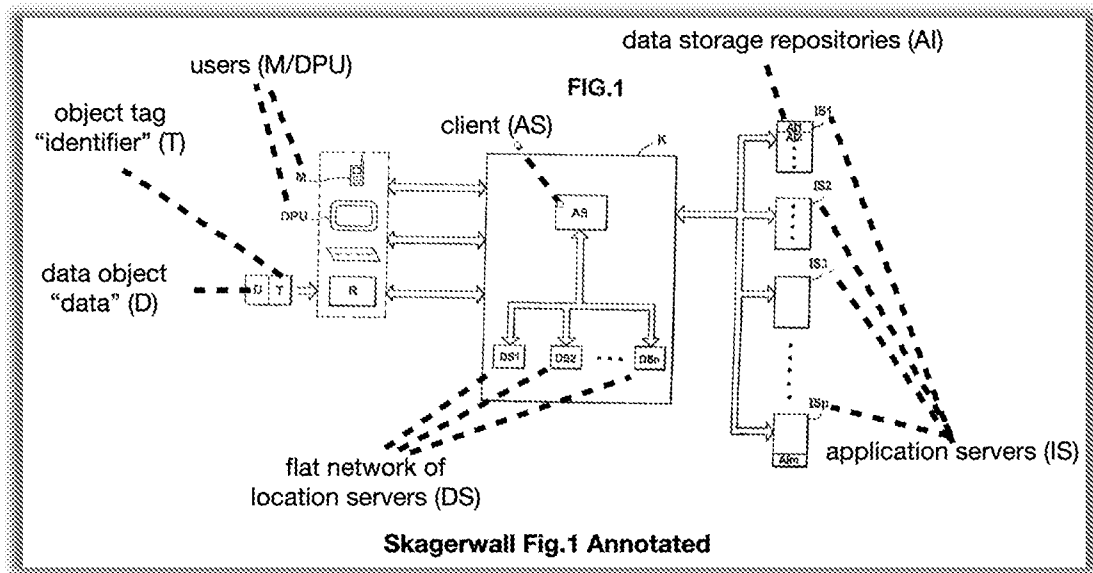
In Skagerwall’s system, there is a “Kernel,” containing at least one access server (“AS”) and the plurality of directory servers (“DS”). As shown below in Figure 1, Skagerwall states that users will request data from the access server, which in turn requests retrieval information, i.e., location information, from the directory servers:

This object of the invention is solved by a communication system, comprising: a plurality of objects (D) each identified by an object tag (T); a plurality of application servers each having an address (IS; IS1-ISp) and each including memory means for storing a plurality of application data packets, each related to one of the objects and identified by an application identifier (AI1-AIm); reading means (R) for reading one of the plurality of object tags (T); a system kernel (K), including: a plurality of directory servers (DS; DS1-DSn) each having an address and including memory means (TM; AM) for storing in a distributed data base object data associating each of the objects with at least one of the application data packets; means for receiving the read object tag (T) from the reading means (R) and means for identifying the directory server for storing object data related to the read object tag and retrieving the object data; and means for retrieving at least one of the application

⁷⁴ *Id.* at 4:12-20.

⁷⁵ *Id.*

data packets based on the object data or transmitting at least part of the object data to a receiving unit (M, DPU).⁷⁶



Skagerwall explains that its system resolves requests for data faster when organizing the directory servers according to a hash function: “To avoid unnecessary look up operations for object tags at directory servers, as outlined above, it is preferred that object data is stored on directory servers according to the hashfunction.”⁷⁷ This organization allows the system to determine, i.e., “know,” which directory server contains the desired information based on the object tag, i.e., identifier:

This preferably includes sending the request including the object tag T; T1-Tw to one of the directory servers DS1; DS2; . . . ; DSn, which in turn, using the hash-function and hashtable H, determines the directory server for storing the object tag T; T1-Tw and forwards the request to that particular directory server. However, it

⁷⁶ *Id.* at 2:51-3:2.

⁷⁷ *Id.* at 10:39-40.

is also possible, that the access server AS; AS1-ASr or other components of the system kernel K perform steps for identifying the correct directory server.⁷⁸

2. Wolff (Ex. 3) Overview

U.S. Patent No. 6,185,601 (“Wolff”) is titled “Dynamic Load Balancing Of A Network Of Client And Server Computers.” Wolff issued on February 6, 2001 from an application filed on April 15, 1998, which was a continuation-in-part of an application filed on August 1, 1997, and claimed priority to provisional applications filed on March 6, 1998 and August 6, 1996. Wolff is prior art under at least 35 U.S.C. §§ 102(a) and (b) because it was filed more than 1 year before ’640 Patent’s filing date of September 13, 2000. Further, even if the ’640 Patent were entitled to priority to the Provisional filing date of September 14, 1999, Wolff would still pre-date this date by over a year. During the original examination of the ’640 Patent, the Examiner didn’t cite or rely on Wolff’s disclosures in any Office action.

Wolff teaches methods for load rebalancing on a distributed network. For example, Wolff explains that its “[m]ethods for load rebalancing by clients in a network are disclosed. Client load rebalancing allows the clients to optimize throughput between themselves and the resources accessed by the nodes.”⁷⁹

As Wolff describes, servers will not always be able to fulfill each request received from a given client. For example, Wolff describes an “overloaded” server which has received too many requests and lacks the processing power to fulfill them all:

In operation at time T=0, normal client 100A is shown accessing memory resource 118 via path 70 through overloaded server 104. At the same time, aware client 102A is shown accessing memory resource 18, via path 74, through overloaded server

⁷⁸ *Id.* at 9-18-31.

⁷⁹ Wolff at Abstract.

104A. At time T=1, process 102P1, implemented on aware client 102A, detects the overload condition of server 104A, and accesses memory resource 118 via an alternate path 76 through server 106A. Thus, in this subsequent state, the load on server 104A is reduced and the access by aware client 102A to memory resource 118 is enhanced. Normal client 100A cannot initiate the processes 35 discussed above in connection with the aware client 102A and is unable to select an alternate path 72 to the underutilized server 106A.⁸⁰

Wolff explains that the network will optimize performance by redirecting requests from an overloaded server, e.g., a server that can't satisfy the request, to another server that can satisfy the request with a "redirection command" from the overloaded server:

Client load rebalancing refers to the ability of a client, enabled with processes in accordance with the current invention, to re-map a path through a plurality of nodes to a resource. *The re-mapping may take place in response to a redirection command emanating from an overloaded node, e.g. server.* The embodiments disclosed allow more efficient, robust communication between a plurality of clients and a plurality of resources, via a plurality of nodes. Resources can include, but are not limited to, computers, memory devices, imaging devices, printers, and data sets. A data set can include a database or a file system.⁸¹

3. Steen (Ex. 4) Overview

Steen is a printed publication written by Maarten van Steen. It was published in the January 1998 edition of IEEE Communications Magazine with the title "Locating Objects in Wide-Area Systems." Mr. Greene confirms that IEEE Communications Magazine was well known to those of ordinary skill in the art at the time of its publishing in January 1998.⁸² The ACM Digital library confirms Steen's inclusion in the January 1998 issue of IEEE Communications Magazine

⁸⁰ *Id.* at 5:25-38.

⁸¹ *Id.* at 5:25-38.

⁸² Greene Decl. ¶ 54.

by listing it in the table of contents for the issue.⁸³ Those in the art were reading and citing to Steen throughout 1998, confirming its public availability.⁸⁴ Thus, Steen was a publicly available printed publication no later than January 31, 1998. As a result, Steen is a printed publication that is prior art under 35 U.S.C. § 102(b) because it published more than one year prior to the earliest possible priority date of the '640 Patent and under § 102(a) because an author other than the named inventors published it before that date. During the original examination of the '640 Patent, the Examiner didn't cite or rely on Steen's disclosures in any Office action.

Steen describes a system for managing and locating data on a wide-area distributed network. As Steen states, its system “present[s] a scalable location service” to “optimize lookups.”⁸⁵ Steen describes a two level system for managing and locating an object's contact addresses. In the first level, Steen describes using a directory system for resolving an object name to an “object handle.”⁸⁶ The object handle includes location information for a contact record. Then, in the second level, Steen teaches that the client/user uses the location information (i.e., object handle) to obtain object contact addresses. Thus, Steen summarizes its two levels as: “An object's name is bound to its unique object handle, which, in turn, is mapped to the addresses where the object can be contacted.”⁸⁷

a) Steen's Object Name

Steen teaches an “object name” that is “well known to its users.”⁸⁸ Steen explains that, in distributed systems, users need names that are separated from locations: “What is needed is a

⁸³ Ex. 32 (ACM Digital Library TOC)

⁸⁴ Exs. 33-36 (Articles citing to Steen throughout 1998).

⁸⁵ Steen at 104.

⁸⁶ Steen at Abstract.

⁸⁷ *Id.*

⁸⁸ *Id.* at 104.

naming facility that hides all aspects of an object's location. Users should not be concerned where an object is located or whether it can move.”⁸⁹ Steen explains that the user uses this “object name” to identify the desired service and obtain the object handle: “A client has the service's name resolved to the object handle.”⁹⁰ Steen explains that this object-name approach simplifies the resolution process for users: “Our approach allows an object to update its contact addresses independent of how users have named the object.”⁹¹

b) Steen's Data Location Information

Steen teaches two levels of data location information. In the first level of its system, Steen teaches location information called object handles: “[The] object handle may contain information that can be used to assist in locating the object.”⁹² For example, Steen describes that the object handle can include subnode identification for a given contact record: “As an example, we can use the first n bits of an object's handle to identify the subnode responsible for that object.”⁹³

Then, Steen provides a second level of data location information called contact records. It states: “Addresses and forwarding pointers are stored in contact records.”⁹⁴ Steen explains that the object “handle” doesn't rely on the addresses in the contact record., describing it as “location independent.”⁹⁵

c) Steen's Partitioned Subnodes

As shown below in Figure 6, Steen describes that each directory server dir(R) may be partitioned into multiple subnodes to handle greater volume of entries and processing requests:

⁸⁹ *Id.* at 105.

⁹⁰ *Id.*

⁹¹ *Id.* at 105.

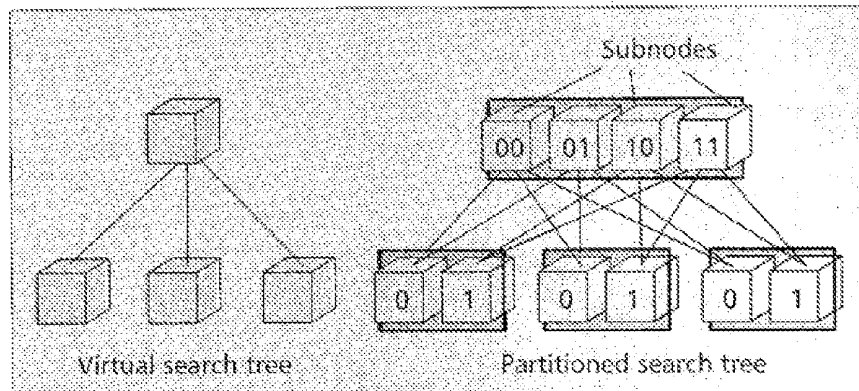
⁹² *Id.*

⁹³ *Id.* at 108.

⁹⁴ *Id.* at 105.

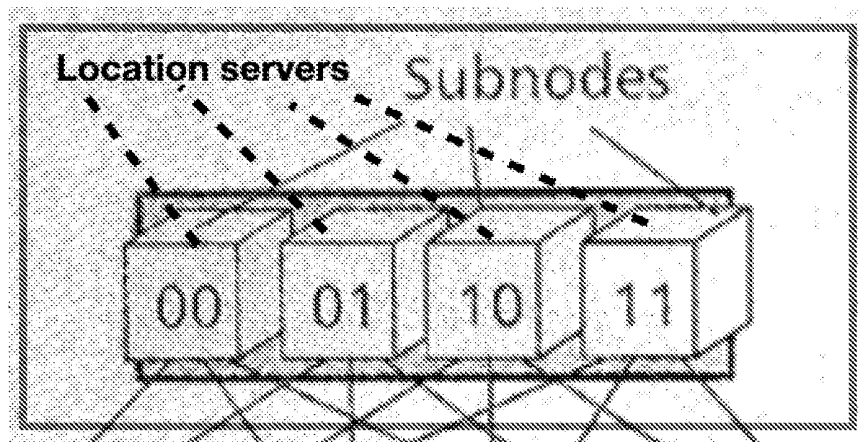
⁹⁵ *Id.*

In particular, higher-level directory nodes not only have to handle a relatively large number of requests, they also have enormous storage demands. Our solution is to partition a directory node into one or more directory subnodes such that each subnode is responsible for a subset of the records originally stored at the directory node.⁹⁶



■ **Figure 6.** A search tree and a corresponding logical tree after partitioning the directory nodes into subnodes.

For a partitioned $\text{dir}(R)$, each subnode is configured as and functions as its own server and contains its own subset of location information as shown below in an annotated portion of Steen's Figure 6.



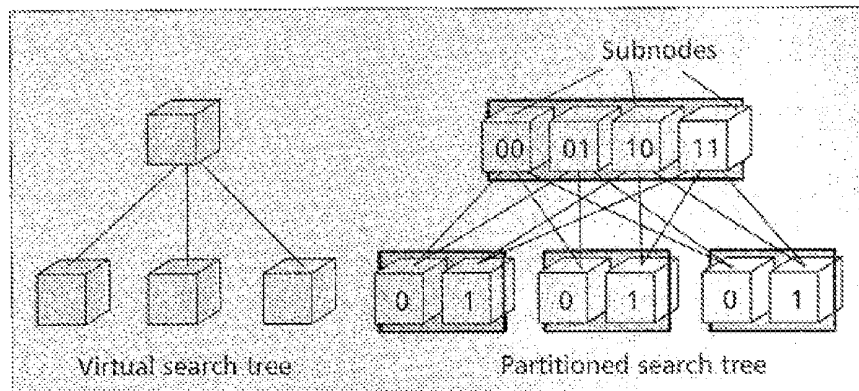
Steen Figure 6 Annotated

⁹⁶ *Id.* 107-08.

d) *Steen's Hashing of Subnodes*

As further shown in Figure 6, Steen discloses that the first n-bits of an objects handle may be used to identify the subnode responsible in a basic hashing method:

As an example, we can use the first n bits of an object's handle to identify the subnode responsible for that object...To illustrate, Fig. 6 shows a search tree in which the root node has been partitioned into four subnodes based on the first two bits of the object handle ($n = 2$), and each of the leaf nodes into two subnodes ($n = 1$).⁹⁷



■ **Figure 6.** A search tree and a corresponding logical tree after partitioning the directory nodes into subnodes.

A POSITA would recognize that Steen's description of using the first n-bits of a handle to identify the correct subnode would be achieved with a very basic hash rule/function. As a result, A POSITA would recognize this description as a basic hashing method.

Steen explains that it is important to organize hashing for each subnode to know how the directory node is partitioned with partitioning and mapping information that can be cached on each subnode:

⁹⁷ *Id.* at 108.

Because communication between directory nodes in the original search tree now takes place between their respective subnodes, each subnode should be aware of how the directory node with which it communicates is actually partitioned. This information is contained in a separate tree management service. This service also maintains the mapping of subnodes to physical nodes. Partitioning and mapping information is assumed to be relatively stable so that it can be easily cached by subnodes.⁹⁸

Steen further teaches that more complex hashing methods would likely be developed for a system based on its specific needs: “We note that we have developed more realistic hashing-based methods than explained here. For example, we also have to take into account that the total number of links between parent and children subnodes remains manageable.”⁹⁹

4. Oracle Overview

Oracle Corporation develops and manufactures commercial database systems. Oracle released its first commercially available database in 1979, Oracle v2.¹⁰⁰ By 1992, Oracle had released Oracle version 7.¹⁰¹

Mr. Greene details how Oracle7 included full functionality for distributed databases.¹⁰² For example, the *Oracle7 Server Concepts* manual explains the concept of a distributed database as follows:

A distributed database is a network of databases managed by multiple database servers that appear to a user as a single logical database. The data of all databases

⁹⁸ *Id.* at 108.

⁹⁹ *Id.*

¹⁰⁰ *See* Greene Decl. ¶ 64 (citing Departments of Informatics’ Oracle v2 timelines (Ex. 12)).

¹⁰¹ *See* Greene Decl. ¶ 65.

¹⁰² *See id.* ¶ 65.

in the distributed database can be simultaneously accessed and modified. The primary benefit of a distributed database is that the data of physically separate databases can be logically combined and potentially made accessible to all users on a network.¹⁰³

To operate as a distributed database system, Mr. Greene explains that a system includes a management utility stored on each client and each database server for sending requests to a database instance and for returning responses and data from the database to the client over the network. An Oracle7 system may be hosting dozens of database instances on any given database server. So to make a connection, the management utility must know the client's and database server's IP address or "host name," the ports used, the database instance, and the transport protocol.¹⁰⁴

Further, Mr. Greene explains that the Oracle7 system included two advances—namely, the expansion of location transparency and the use of hashing in distributed databases.¹⁰⁵ In distributed databases, location transparency hides the actual location of data from the users, allowing them to "see through" the location and internal storage information to the data itself. Location transparency is an important feature for maintaining and updating a distributed database on an ongoing basis. As the book *Oracle: The Complete Reference* details:

Over the life span of an application, its data will very likely move from one database to another, or from one host to another. Therefore it will simplify maintenance of an application if the exact physical location of a database object is shielded from the user (and the application).¹⁰⁶

¹⁰³ Ex. 13 (Oracle7 Server Concepts) at 71-72.

¹⁰⁴ See Greene Decl. ¶ 67.

¹⁰⁵ See *id.* ¶ 68.

¹⁰⁶ Ex. 14 (Oracle: The Complete Reference) at 5.

And Mr. Greene details that, in 1996, Oracle introduced further distributed database capabilities, including the release of Oracle version 7.3, the Oracle tools and utilities SQL*Net release 2.3.3, and Oracle Names release 2.0.¹⁰⁷

SQL*Net is the name for Oracle's add-on networking utility. This utility provides the network underpinnings for distributed databases.¹⁰⁸ The 1994 *Oracle DBA Handbook* explains that SQL*Net makes it "possible to create synonyms that give applications true network transparency; the user who submits the query will not know the location of the data that is used to resolve it."¹⁰⁹ SQL*Net Client runs on each client and interfaces with applications running on the client. SQL*NET Listener runs on database servers, listens on multiple network ports and IP addresses for client requests, and returns requested information to clients.¹¹⁰

Oracle Names is the moniker for Oracle's distributed database name server utility, and provides a directory of databases on Names Servers.¹¹¹ Names Servers are intermediaries that provide connection information, including location, to clients requesting data from corresponding data storage servers. Using Oracle Names, a client seeking data requests the location of the data from a Names Server, receives a location response from the Names Server, then requests the data itself from the data storage server identified in the location response.¹¹²

¹⁰⁷ See Greene Decl. ¶ 71.

¹⁰⁸ See *id.* ¶ 72.

¹⁰⁹ See Ex. 15 (Oracle DBA Handbook) at 13.

¹¹⁰ See Greene Decl. ¶ 72.

¹¹¹ At times, those in the field refer to Oracle Names as "Oracle*Names." See *id.* ¶ 73 n.57.

¹¹² See *id.* ¶ 73.

With Oracle Names, Mr. Greene explains that all tasks for managing distributed databases are handled using a global naming service, available to all hosts in a network. As the book *Oracle: The Complete Reference* explains:

This changes the way in which database links are resolved. Before, when a database link was specified, the database first looked at the user's private database links. If none with the matching name was found, then the available public database links were checked.

Oracle*Names adds an additional level to this. Now, if the first two checks do not return a match for the database link name, then the Oracle*Names server's list of global database link names is searched for the database link. If the link name is found there, Oracle*Names will return the link's specifications and resolve the query.

This greatly simplifies the administration of *location transparency*—the ability to mask the physical location of data-in a distributed environment. The information related to remote data access is now stored in a central location. The impact of this is felt every time an object's location is changed. For example, if there were multiple links using explicit connections to a single demote database, then a pre-Oracle*Names modification to the user's password would require dropping and re-creating multiple database links.¹¹³

Oracle maintains a separate manual for Oracle Names release 2.0, which is titled *Oracle Names Administrator's Guide, Release 2.0* (Exhibit 2).¹¹⁴

Mr. Greene explains that Oracle Names and SQL*Net allow customers to create unique customer identifiers associated with a data set's network address, and to use those identifiers to query Names Servers to locate where in the system the data set resides.¹¹⁵

¹¹³ See Ex. 14 (Oracle: The Complete Reference) at 8.

¹¹⁴ See OracleNamesAdminGuide at 4.

¹¹⁵ See Greene Decl. ¶ 75.

Like most relational database management system (“RDBMS”) products at the time, Oracle7 included hashing capability. Hashing was a known approach at the time to improve performance by processing certain types of queries more quickly.¹¹⁶ The *Oracle DBA Handbook* explains Oracle7’s use of this hashing capability as follows:

In Oracle7, a second type of cluster is supported. Hash clusters use hashing functions on the row’s cluster key to determine the physical location where the row should be stored. This will yield the greatest performance benefit for queries of the form shown in the following listing.

```
SELECT name
FROM employee
WHERE empno = 123;
```

In this example, the EMPLOYEE table is queried for an exact match for the EMPNO column. If that table was part of the hash cluster, and EMPNO was part of the cluster key, then the database could use the hashing function to quickly determine where the data was physically located.¹¹⁷

And as the *Oracle DBA Handbook* illustrates, it was understood in the art that hashing reduced the processing cost and time for query results:

Queries in which a column’s value is compared to an exact value are called equivalence queries. In Oracle7, a new type of cluster, called a hash cluster, is made available. This type of cluster stores a row in a specific location based on its value in the cluster key column. Every time a row is inserted, its cluster key value is used to determine which block it should be stored in; this same logic can be used during queries to quickly find data blocks that are needed for retrieval.¹¹⁸

¹¹⁶ See *id.* ¶¶ 76-77.

¹¹⁷ Ex. 15 (*Oracle DBA Handbook*) at 5-6.

¹¹⁸ *Id.* at 9.

Today, Oracle is up to version 21.¹¹⁹ Patent Owner didn't provide any Oracle publications to the Patent Office during the original examination of the '640 Patent.

5. OracleNamesAdminGuide (Ex. 5) Overview

OracleNamesAdminGuide is a printed publication from the Oracle Corporation. On its face, OracleNamesAdminGuide shows that it was published by the Oracle Corporation in 1996 as Part No. A42482-1.¹²⁰ Oracle Corporation made OracleNamesAdminGuide accessible to the public on Oracle's website.¹²¹ Mr. Greene confirms that, as of 1996, Oracle Corporation "was an established publisher that was well known to those interested in the field of database technologies."¹²²

In Exhibit 18, Dr. Sylvia Hall-Ellis corroborates that OracleNamesAdminGuide was publicly accessible by at least May 7, 1997.¹²³ Dr. Hall-Ellis identifies a College of William & Mary library catalog entry that shows that OracleNamesAdminGuide was entered into that library's collection on May 7, 1997, where it was indexed by subject matter for public access.¹²⁴ As Dr.

¹¹⁹ See Greene Decl. ¶ 80.

¹²⁰ See OracleNamesAdminGuide at 4.

¹²¹ See, e.g., https://docs.oracle.com/pdf/A42482_1.pdf (last accessed October 13, 2021).

¹²² Greene Decl. ¶ 81; see also *Vidstream LLC v. Twitter, Inc.*, 981 F.3d 1060, 1065 (Fed. Cir. 2020) ("A reference will be considered publicly accessible if it was 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.... When there is an established publisher there is a presumption of public accessibility as of the publication date.").

¹²³ See Ex. 18 (Hall-Ellis Decl.) ¶ 19. Dr. Hall-Ellis is an Adjunct Professor in the School of Information at San Jose State University in San Jose, California. She holds a Ph.D. in Library Science from the University of Pittsburgh. See *id.* at pp.12-14 (Attachment B). In the proceeding Dr. Hall-Ellis originally opined on OracleNamesAdminGuide's publication date, it was listed as Exhibit 2, but in this request it is Exhibit 5.

¹²⁴ See *id.* ¶ 31.

Hall-Ellis explains, this catalog entry establishes that OracleNamesAdminGuide was publicly accessible no later than May 7, 1997.¹²⁵

Exhibit 24 further corroborates the public accessibility of OracleNamesAdminGuide. Exhibit 24 is a National Stock Number (“NSN”) entry, which shows that the U.S. government assigned OracleNamesAdminGuide the NSN 7610-01-448-8459 on September 11, 1997.¹²⁶ The U.S. government assigns an NSN to publications, among other items in commerce, to process their procurement for the Department of Defense.¹²⁷

Thus, OracleNamesAdminGuide is a printed publication that is prior art under 35 U.S.C. § 102(b) because it published more than one year prior to the application date of the ’640 Patent and under § 102(a) because authors other than the named inventors published it before the application date. And, even if Patent Owner were able to establish the ’640 Patent is entitled to the benefit of the earliest possible priority date of July 8, 1998, OracleNamesAdminGuide is §§ 102(a) and (b) prior art based on its publication date.

OracleNamesAdminGuide describes a real product, Oracle Names, which is a part of an Oracle7 database system. OracleNamesAdminGuide discloses a distributed database system, fo-

¹²⁵ See *id.* ¶ 26.

¹²⁶ Ex. 24 at 2-3.

¹²⁷ See, e.g., https://www.dla.mil/Portals/104/Documents/J3LogisticOperations/Brochures/J3_nsnbooklet_200323.pdf?ver=2020-03-24-105311-683. As shown in Exhibit 24, OracleNamesAdminGuide was assigned a National Stock Number on September 17, 1997, corroborating the ongoing public accessibility of OracleNamesAdminGuide.

cusing on the Oracle Names utility that provides connection information for service locations residing on different clients and servers.¹²⁸ And OracleNamesAdminGuide explains that its Oracle7 system includes a Names Server that responds to a client's data requests using unique customer identifiers associated with the data's network address:

An Oracle Names Server *stores names and addresses for network services* such as databases or MultiProtocol Interchanges, database link definitions, and object aliases.

The Names Server stores the following types of objects, which are described below:

- database service names—*The service name is the global database name that is mapped to the SQL*Net connect descriptor*. The combination of a listener address and connect data make up a connect descriptor that locates a database service. Gateways to non-Oracle databases and Oracle RDB databases are also stored....
- V1 connect strings—A V1 *connect string can be mapped to a service name in the Oracle Names database*. The service name you enter acts as an alias for the SQL*Net V1 connect string, making it easy to access a server in a network that still uses SQL*Net V1....

After services have been defined in Oracle Names, *users only need to refer to them by name to connect to a database or network service*. The necessary exchange of details occurs between the requester and the Oracle Names Server.¹²⁹

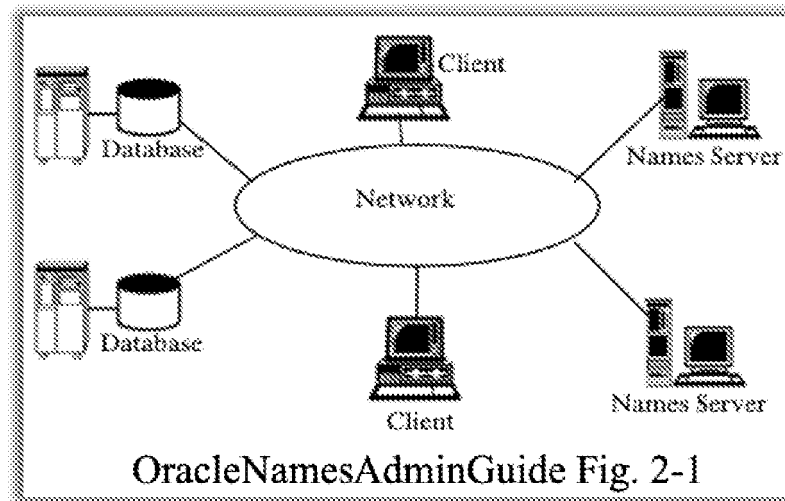
Further, OracleNamesAdminGuide describes an Oracle7 system having multiple Names Servers functioning as a directory service storing addresses for all services on the network and making them available to clients wishing to make a connection.¹³⁰ As shown below in Figure 2-

¹²⁸ See OracleNamesAdminGuide at 5, 22, 52-54.

¹²⁹ *Id.* at 23-24.

¹³⁰ See *id.* at 5, 22, 52-54.

1, OracleNamesAdminGuide explains that an Oracle7 system supports multiple clients, databases, and Oracle Names database instances connected on a network.¹³¹



OracleNamesAdminGuide discloses that servers include a “listener” application that listens for client requests and hands off requests to servers.¹³²

a) OracleNamesAdminGuide’s Names Servers

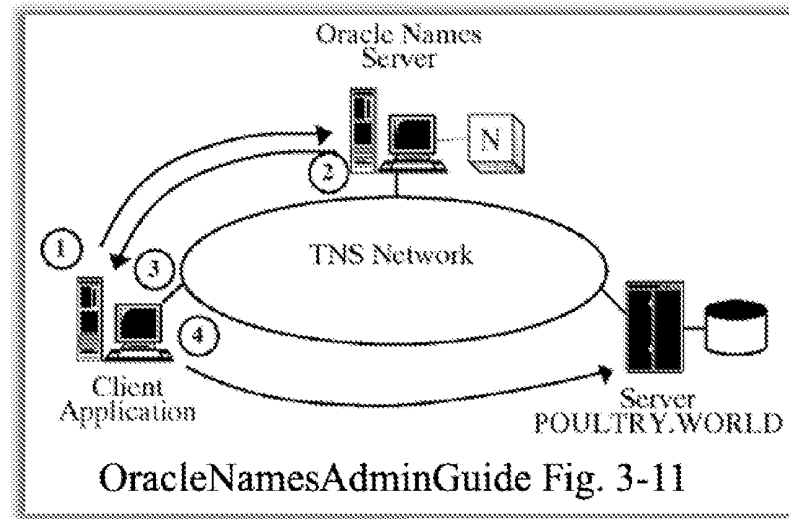
OracleNamesAdminGuide explains a “Names Server has a single purpose: to resolve, or assist in resolving, a client-initiated name request. It interprets the request, then looks up the name in its cache, or it calls a Names Server in another region. The response [is] passed back to the client.”¹³³ As shown below in Figure 3-11, OracleNamesAdminGuide describes exchanging messages that include ① the client issues a data query to Names Server, ② the Names Server receives the request, looks up the location of the data, then sends the database connection and location information to the client, ③ the client uses the location information data to contact the

¹³¹ See *id.* as 32-33.

¹³² See *id.* at 32-33, 70; Greene Decl. ¶¶ 50-51.

¹³³ OracleNamesAdminGuide at 52; see also *id.* at 202 (defining an Oracle Names Server as “[t]he name resolution software for efficiently answering name requests”).

server hosting the desired database instance, and ④ the client establishes a client-server connection.¹³⁴



And OracleNamesAdminGuide explains that large installations benefit from having a central directory service that captures connection information in one system for all clients to use.¹³⁵ OracleNamesAdminGuide teaches storing connection information (e.g., host name, port, and database instance) associated with a client-side alias identifier.¹³⁶

As discussed above in Section III.B, Patent Owner has asserted that the claim term “location server” means “a network-attached component that maintains a set of identifier/location mappings that are modified or returned in response to location request messages from clients.” A POSITA would have understood that OracleNamesAdminGuide teaches that each Names Server is a location server.¹³⁷ This is because each Names Server attaches to the network and maintains

¹³⁴ See *id.* at 52-53.

¹³⁵ See *id.* at 21-22.

¹³⁶ See, e.g., *id.* at 23, 52-53, 128-29.

¹³⁷ See Greene Decl. ¶ 91.

data location identifiers and returns those identifiers in response to query messages from clients. And OracleNamesAdminGuide explains that each Names Server is capable of modifying and updating data location identifiers.¹³⁸

b) OracleNamesAdminGuide's Non-Hierarchical Configuration

The '640 Patent discloses that its purported invention covers systems having either a hierarchical or non-hierarchical configuration. For example, the '640 Patent states that its Server Redirection Mechanism allows for clusters and hierarchical topologies: "NDTP provides two mechanisms for server redirection. The redirection mechanisms allow *cluster and hierarchical topologies, and mixtures of such topologies*"¹³⁹

Further, the '640 Patent claims the very hierarchical architecture that Patent Owner asserts the Oracle prior art requires. Specifically, '640 claim 26 expressly recites that its claimed system covers hierarchical configurations where each location server functions as a "client" for retransmitting data location requests:

The system of claim 18, further comprising a plurality of servers *related in a logical hierarchy* between the client and the data location servers, wherein *each of the plurality of servers is configured to function as a next client* and retransmit the data location request to a next logically associated server until a data location server receives the data location request.

Like the '640 Patent, OracleNamesAdminGuide discloses some embodiments with systems having hierarchical configurations and other embodiments with systems having non-hierar-

¹³⁸ *E.g.*, OracleNamesAdminGuide at 67 (explaining Names Servers store modified connect data), 101 (explaining administrator modifying data then refreshing Name Server).

¹³⁹ '640 Patent at 14:28-30.

chical configurations. As detailed in Section III.C.5, OracleNamesAdminGuide teaches an embodiment having a system with a non-hierarchical configuration for its Names Servers. For example, OracleNamesAdminGuide teaches a system for small-scale applications that uses a non-hierarchical “flat naming model.”¹⁴⁰ As Mr. Greene explains, a POSITA would understand that OracleNamesAdminGuide’s non-hierarchical configuration has no local region and doesn’t require a client’s location query to travel up or down a network “tree” to reach a root server.¹⁴¹

Further, as shown below in Figure 3-5, OracleNamesAdminGuide specifies that its small-scale system is non-hierarchical because “[a]ll names are unique within a single domain—the WORLD domain. The WORLD domain is predefined in the Oracle Network Manager for customers choosing a flat naming model.”¹⁴² For a client query such as for data in location CONFIG, OracleNamesAdminGuide teaches that the system will append WORLD to the query as in CONFIG.WORLD.¹⁴³ OracleNamesAdminGuide discloses that, “[i]n the most common case of a single, central administrative region, all Names Servers contain identical data, and give identical results.”¹⁴⁴

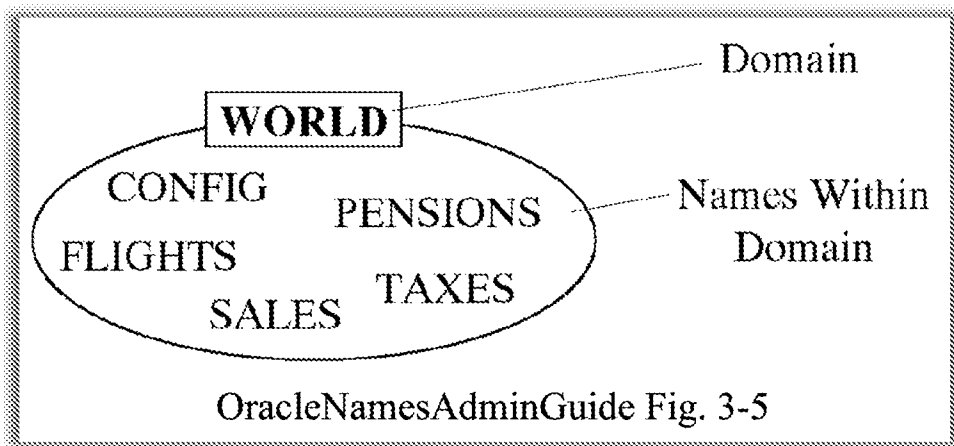
¹⁴⁰ OracleNamesAdminGuide at 40-41.

¹⁴¹ See Greene Decl. ¶ 195.

¹⁴² OracleNamesAdminGuide at 40.

¹⁴³ *Id.* at 41.

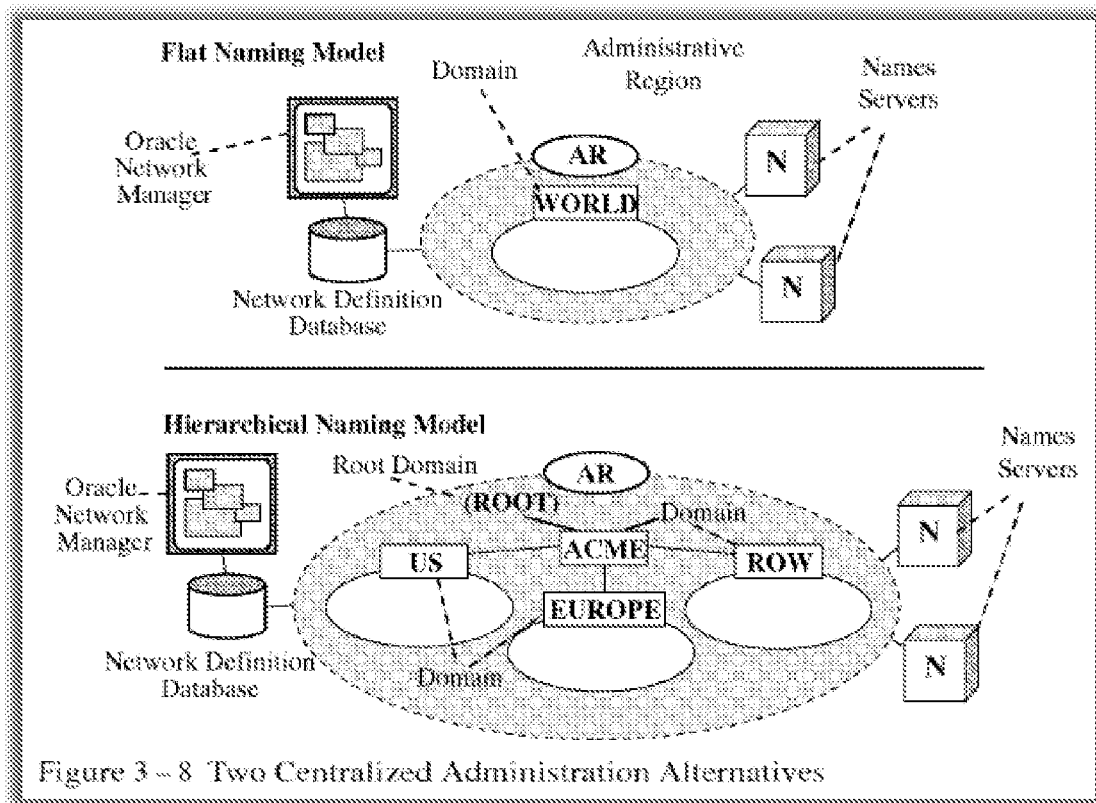
¹⁴⁴ *Id.* at 45.



And OracleNamesAdminGuide details that its flat non-hierarchical configuration is different from a hierarchical configuration for large-scale applications of more than 100 services.¹⁴⁵ Below, OracleNamesAdminGuide's Figure 3-8 illustrates how a system having the "flat" non-hierarchical configuration differs from other systems having a hierarchical configuration.¹⁴⁶

¹⁴⁵ *Id.* at 40-41.

¹⁴⁶ *Id.* at 44-45.



1. *OracleNamesAdminGuide's SQL*Net Utility*

OracleNamesAdminGuide teaches that SQL*Net is Oracle's networking utility that enables remote data access between programs and databases. SQL*Net allows users to distribute applications and databases to different machines while continuing to communicate as if they were located on the same machine.¹⁴⁷ OracleNamesAdminGuide teaches that SQL*Net is an application that "works with the Oracle Server and enables two or more computers ... to exchange data through a network. SQL*Net supports distributed processing and distributed database capability.

¹⁴⁷ See *id.* at 29-30, 36, 53-54, 203.

SQL*Net runs over and interconnects many communications protocols.”¹⁴⁸ And OracleNamesAdminGuide teaches that, using SQL*Net, Names Servers communicate network address information to clients.¹⁴⁹

2. *OracleNamesAdminGuide*’ Location Information

For a given data entity (e.g., a database), OracleNamesAdminGuide teaches that the entity’s location information includes its host name,¹⁵⁰ port, and database instance. OracleNamesAdminGuide explains that its Names Servers contain this location information to respond to a client’s request to connect to a data entity.

As discussed above in Section III.B, Patent Owner has asserted that the claim term “location” means “an encoding that is a member of a set of associations with an identifier in a location server, and that specifies where data pertaining to the entity identified by the identifier is stored.” Patent Owner has also asserted that the claim term “location information” means “information pertaining to one or more locations of data and/or the identities of one or more location servers.”

OracleNamesAdminGuide explains that its SQL*Net uses location information to form a “connection descriptor,” also called a connect string or dbstring. The client uses this information to establish a client-server connection.¹⁵¹ OracleNamesAdminGuide explains that its connection descriptor is a “specially formatted description of the destination for a network connection. Connect descriptors are constructed using a set of keywords and values. They are mapped to *service*

¹⁴⁸ *Id.* at 203.

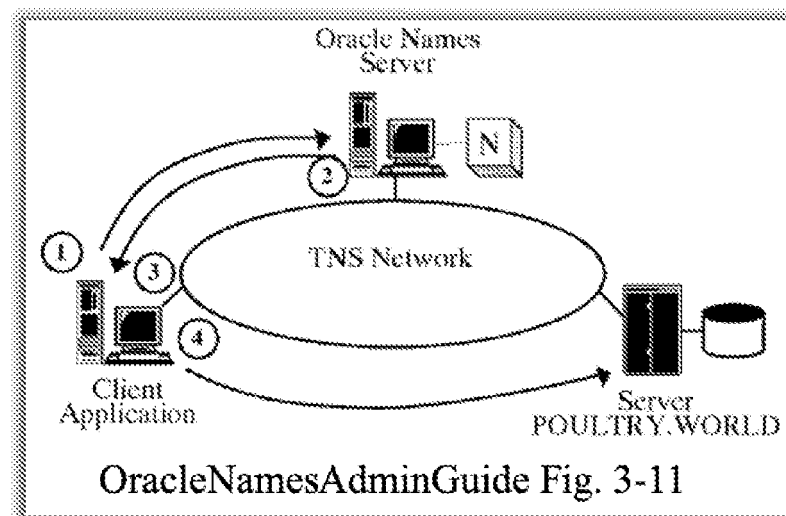
¹⁴⁹ *See id.* at 20, 23, 52-53; Greene Decl. ¶ 92.

¹⁵⁰ A POSITA would have understood that a “host name” may include a direct IP address or a host name that uses DNS to obtain an IP address. *See* Greene Decl. ¶ 93 n.85.

¹⁵¹ *See* OracleNamesAdminGuide at 23 (explaining the prior version of Oracle Names used the term “connect string”), 100 (teaching prior Oracle versions refer to descriptors as “dbstring [which] is a database name defined in the Names Server”).

names to provide more convenient reference.”¹⁵² And OracleNamesAdminGuide explains that each “unique network location [is] used to identify a network object such as a database service, client, Interchange, or Names Server. TNS addresses have a specific format. Addresses must be unique.”¹⁵³

As shown below in Figure 3-11, OracleNamesAdminGuide teaches that the client device sends to the Names Server a request for the POULTRY.WORLD database instances. The Names Server uses the identifier POULTRY.WORLD to identify the location information (e.g., host name, port, and database instance) and sends this location information to the client. Using this location information, the client contacts the server housing the POULTRY.WORLD database instance, then uses SQL*Net to make the connection.¹⁵⁴



¹⁵² *Id.* at 200; *see also id.* at 202 (describing that a connect descriptor may include a system identifier (“SID”), which is a “unique name for an Oracle database instance.... The SID is included in the CONNECT DATA parts of the connect descriptors in a TNSNAMES.ORA file, and in the definition of the network listener in the LISTENER.ORA file”).

¹⁵³ *Id.* at 199; Greene Decl. ¶ 94.

¹⁵⁴ *See* OracleNamesAdminGuide at 52-53.

Similarly, OracleNamesAdminGuide provides an exemplary request to the Names Server for connection and location information using “QUERY BONES.DEM.MEDICINE A.SMD” to the Oracle7 system.¹⁵⁵ Like the above POULTRY.WORLD example, OracleNamesAdminGuide teaches that the SQL*Net client sends a request to the Names Server for the desired database using the corresponding BONES.DEM.MEDICINE identifier. The Names Server uses the identifier BONES.DEM.MEDICINE from the request to identify the location information (e.g., host name, port, and database instance) and send this location information to the client. The client then uses this location information to connect to the server storing the BONES.DEM.MEDICINE database. OracleNamesAdminGuide details the resulting query metrics and location information sent to the client from the Name Server as follows:

Total response time:	0.04 seconds
Response status:	normal, successful completion
Authoritative answer:	yes
Number of answers:	1
Canonical name:	bones.dem.medicine
TTL:	1 day
Alias translations:	
from:	bones.dem.medicine
to:	bones.dem.medicine

Answers:

data type is ‘a.smd’

Syntax is ADDR: ... (DESCRIPTION=(ADDRESS=(COMMUNITY=tcp) (PROTOCOL=TCP) (Host=cowboy) (Port=1522)) (CONNECT_DATA=(SID=rodeo))) [.]”¹⁵⁶

¹⁵⁵ See *id.* at 129.

¹⁵⁶ *Id.*

This illustrates that OracleNamesAdminGuide's Names Server completed the response in an *a.smd* output format in 0.4 seconds, and labeled the response status as normal and successfully completed. And the Names Server responds (i.e., "answers") to the client's query by providing the location information for BONES.DEM.MEDICINE.¹⁵⁷ This location information includes the host name "cowboy,"¹⁵⁸ port 1522, and database instance "rodeo" along with the connection information ("protocol=TCP").¹⁵⁹

Using this location information, OracleNamesAdminGuide teaches that the client contacts the server housing the BONES.DEM.MEDICINE database, then uses SQL*Net to connect to the database using one of the available network transport protocols, TCP/IP.¹⁶⁰ And using the below annotated version of OracleNamesAdminGuide's Figure 3-11, Mr. Greene demonstrates the sequence of events to resolve the client's data query BONES.DEM.MEDICINE A.SMD.¹⁶¹

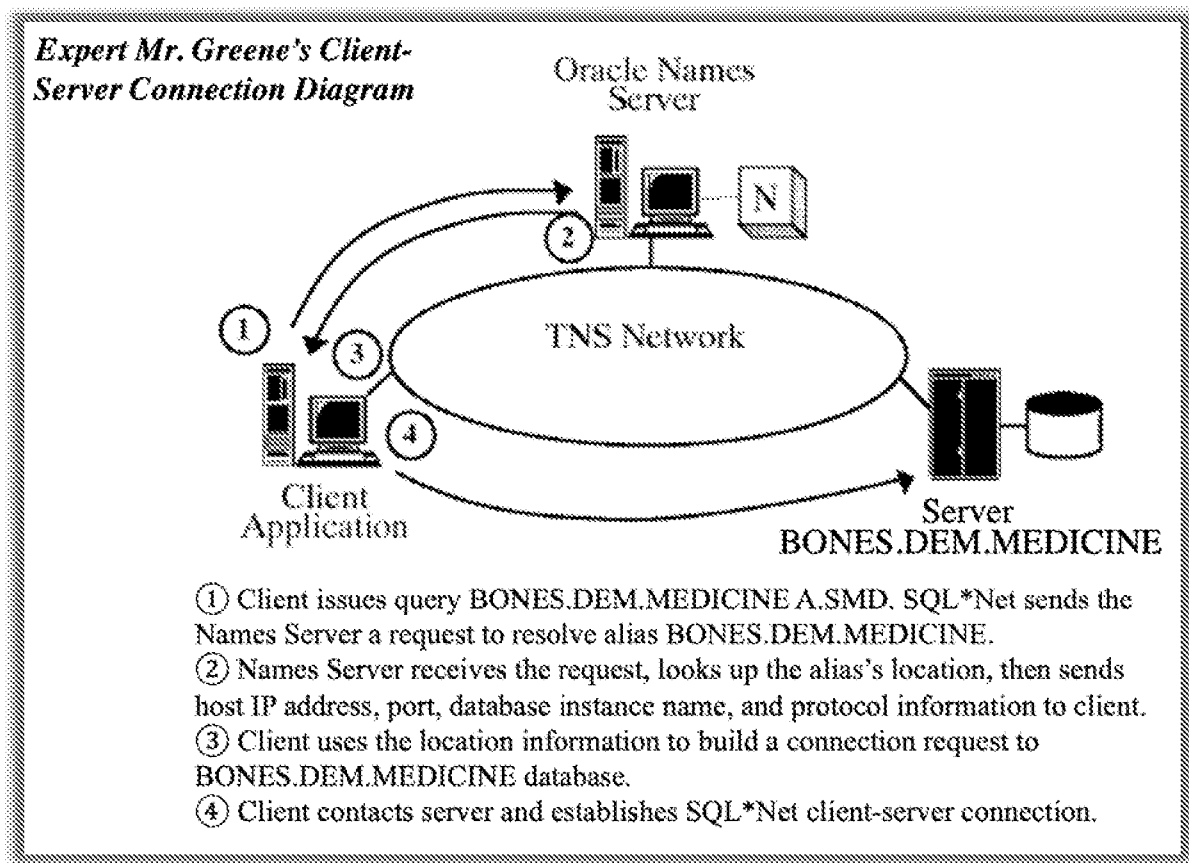
¹⁵⁷ *See id.*

¹⁵⁸ As Mr. Greene explains, the host name would be converted to an IP address using a DNS server before the connection string was compiled. Greene Decl. ¶ 97 n.93.

¹⁵⁹ *See* OracleNamesAdminGuide at 129.

¹⁶⁰ *See id.* at 39, 201 (describing other available transport protocols, like DECnet).

¹⁶¹ Diagram from Mr. Greene's Declaration. *See* Greene Decl. ¶ 98 (citing OracleNamesAdminGuide at 52, 129).



Based on these teachings, a POSITA would have understood that OracleNamesAdminGuide's connection descriptor is built using the claimed "location information" (i.e., a data encoding that specifies the storage location of data and that is associated with an identifier for the data (e.g., network-service-name-alias) in a Names Server).¹⁶²

3. *OracleNamesAdminGuide's Identifier String*

OracleNamesAdminGuide discloses associating location information (i.e., connection descriptor) to an identifier. As discussed above in Section III.B, Patent Owner has asserted the claim term "identification string" means "a unique encoding that identifies an individual entity, and with which zero or more location strings are associated in a location server."

¹⁶² See *id.* ¶ 99.

On the client-side of a client-server connection, OracleNamesAdminGuide uses a network-service-name-alias as an identifier string to easily refer to desired entities on the distributed database and to obtain connection information from the Names Server. For example, OracleNamesAdminGuide explains that aliases allow a user to “connect to a database or use a database link by using a simple name instead of having to specify the global database name or global database link name (that is, the fully-qualified name).”¹⁶³ As Mr. Greene explains, a POSITA would have understood that Oracle literature uses “alias” in other contexts, such as server-side aliases and object aliases.¹⁶⁴ Because those other types of aliases are different and distinct from aliases used on the client-side of an Oracle7 system, the Request and Mr. Greene refer to client-side identifiers as network-service-name-aliases.¹⁶⁵

As shown in the above example, OracleNamesAdminGuide’s client uses the network-service-name-alias BONES.DEM.MEDICINE to identify a specific database associated with location information (e.g., host name, port, database instance) in the Names Server. And as shown in that example, the network-service-name-alias BONES.DEM.MEDICINE refers to one, and only one, database in the Oracle7 system.¹⁶⁶

Based on these teachings, a POSITA would have understood that OracleNamesAdminGuide’s network-service-name-alias is the claimed “identifier string” because it is a unique en-

¹⁶³ OracleNamesAdminGuide at 101.

¹⁶⁴ See Greene Decl. ¶ 101.

¹⁶⁵ *Id.* ¶¶ 101-02.

¹⁶⁶ As OracleSG explains, this single location address may correspond to a virtual address for multiple servers working together for load balancing. See OracleSG at 690-96.

coding that identifies an individual data entity (e.g., BONES.DEM.MEDICINE) and is associated with the location information (i.e., host name, port, and database instance used to build a connection descriptor) in a Names Server.

In an IPR proceeding concerning claims of the related '978 Patent, Patent Owner argued that the specification requires a "1-to-1 relationship between an identifier and an entity and [a] 1-to-many relationship between an identifier and locations."¹⁶⁷ But Patent Owner's argument lacks sufficient support in the disclosures of the '978 Patent or '640 Patent that would limit an entity to one, and only one, identifier, or that would require a one-to-many relationship between identifier and location. Patent Owner appears to base its 1-to-1 argument on the position that its construction's use of the word "unique" means that it requires exclusivity. But this interpretation is inconsistent with the specification, which refer to multiple identification strings specifying an entity and that the relationships between identifiers, locations, and entities are dynamic:

The network distributed tracking wire transfer protocol includes two basic components: *identification strings for specifying the identity of an entity* in the distributed data collection, and location strings for specifying network locations of data associated with an entity. The protocol accommodates variable length identifier and location strings. *Relationships between identification strings and location strings can be dynamically and spontaneously manipulated thus allowing the corresponding data relationships also to change dynamically*, spontaneously, and efficiently.¹⁶⁸

¹⁶⁷ Ex. 20 (IPR2020-00276 Preliminary Resp.) at 26.

¹⁶⁸ '640 Patent at 2:15-24.

And indeed, in its IPR decision denying institution, the Board stated that it “disagree[s] based on the record before us with Patent Owner’s arguments that its construction requires as ‘a 1-to-1 relationship between an identifier and the entity that it is identifying.’”¹⁶⁹

Yet even if Patent Owner’s 1-to-1 argument were correct, OracleNamesAdminGuide and OracleUnleashed describe specific configurations in which an identifier identifies a single entity. For example, as shown in Figure 3-11, OracleNamesAdminGuide discloses that the identifier POULTY.WORLD identifies a single entity, the POULTRY.WORLD database.¹⁷⁰ As another example, OracleNamesAdminGuide discloses that the identifier BONES.DEM.MEDICINE identifies a single entity, having location Host=cowboy, Port=1522, CONNECT_DATA=(SID=rodeo).¹⁷¹

Similarly, even if Patent Owner’s 1-to-many argument were correct, OracleNamesAdminGuide and OracleUnleashed describe specific configurations in which an identifier is associated with multiple locations. For example, OracleNamesAdminGuide explains the need for fault tolerance in an Oracle7 system in which the data entities are duplicated in the system.¹⁷² In such a configuration, a POSITA would have understood an entity’s identifier has a 1-to-many relationship with two or more locations.

¹⁶⁹ Ex. 21 (IPR2020-00276 Decision) at 12 (declining to construe “identifier,” “location,” and “location information” because decision didn’t turn on those constructions for ’978 Patent concerning art not at issue here).

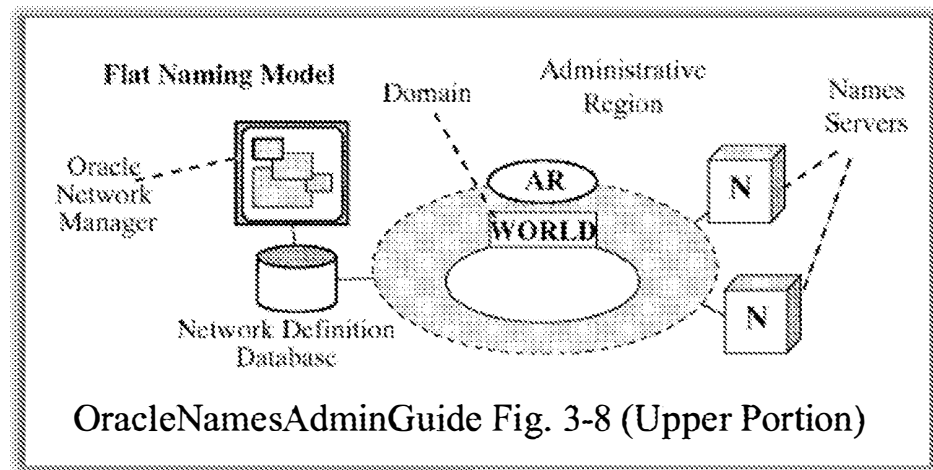
¹⁷⁰ See OracleNamesAdminGuide at 52-53.

¹⁷¹ See *id.* at 129.

¹⁷² See *id.* at 32, 72; 129.

4. *OracleNamesAdminGuide's Redirect Messaging*

Using location and identifier strings, OracleNamesAdminGuide teaches that a client submits a query for the location of a data entity in an Oracle7 system having multiple Names Servers, such as in below Figure 3-8. If the upper Names Server receives that query but lacks the location information to answer that query, OracleNamesAdminGuide teaches the option of the upper Names Server forwarding the request to the lower Names Server. But relevant to the '640 claims, OracleNamesAdminGuide also teaches that it is often desirable to disable the upper Names Server's forwarding function to lessen the workload of the Names Servers, especially when that server is a performance bottleneck in processing requests.¹⁷³ When the forwarding function is disabled, OracleNamesAdminGuide explains that the upper Names Server "simply tells the requestor the address of the Names Server that can answer the request."¹⁷⁴



¹⁷³ *E.g., id.* at 137; *see also id.* at 119-20, 186-87.

¹⁷⁴ *Id.* at 137.

Using this message from the upper Names Server, a POSITA would have understood that the requester, i.e., client, then determines the location of the lower Names Server that has the requested location information.¹⁷⁵ In sum, OracleNamesAdminGuide teaches that, if a Names Server doesn't have the client's requested location string, the Names Server can be configured to transmit to a redirect message to the client that allows the client to calculate the location of a different Names Server containing the location string.

6. OracleUnleashed (Ex. 6) Overview

OracleUnleashed is a printed publication authored by Mr. Greene.¹⁷⁶ As Mr. Greene attests, OracleUnleashed was published by Sams Publishing and made available in print by at least January 1998.¹⁷⁷ Mr. Greene confirms that, as of 1998, Sams Publishing “was an established publisher that was well known to those interested in the field of database technologies.”¹⁷⁸ Corroborating evidence establishes that OracleUnleashed was publicly available and indexed by subject matter in a library no later than February 2, 1998 in its translated German version.¹⁷⁹

Thus, OracleUnleashed is a printed publication that is prior art under 35 U.S.C. § 102(b) because it published more than one year prior to the application date of the '640 Patent and under § 102(a) because an author other than the named inventors published it before the application date. And, even if Patent Owner were able to establish the '640 Patent is entitled to the benefit of

¹⁷⁵ See Greene Decl. ¶ 110.

¹⁷⁶ See *id.* ¶ 111.

¹⁷⁷ See *id.*

¹⁷⁸ *Id.*; see also *Vidstream LLC v. Twitter, Inc.*, 981 F.3d 1060, 1065 (Fed. Cir. 2020) (“A reference will be considered publicly accessible if it was 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.... When there is an established publisher there is a presumption of public accessibility as of the publication date.”).

¹⁷⁹ See Hall-Ellis Decl. ¶¶ 56-59.

the earliest possible priority date of July 8, 1998, OracleUnleashed is § 102(a) prior art because it was published more than six months prior to that date. OracleUnleashed hasn't been previously considered in connection with the '640 Patent.

OracleUnleashed describes versions of the Oracle system up through Oracle version 8.1 and related products. An earlier book Mr. Greene authored, Oracle DBA Survival Guide was published in 1995 and only describes Oracle features available at the time (those included in up until version 7.2). Both books describe the use of the real Oracle system products, along with other applications that work with those products, but OracleUnleashed describes the most up to date features as of its publication in 1997.¹⁸⁰ OracleUnleashed states that it is a “quick reference machete” that cuts “through to the complete world of the Oracle Server and its components.”¹⁸¹

OracleUnleashed explains that its teachings are compatible with systems having Oracle Names Servers,¹⁸² such as OracleNamesAdminGuide. This is because the Oracle8 system includes all features of prior versions, including those of Oracle Names Servers.¹⁸³

OracleUnleashed provides a description of the key details of the Oracle8 system:

Instead of getting lost in endless reams of manuals, by first referring to this book, one can focus on the issues and hotpoints surrounding the complex tasks faced in purchasing, installing, and implementing the appropriate Oracle technology.¹⁸⁴

Accordingly, OracleUnleashed aims to help skilled artisans identify a system-related issue and find the solution:

¹⁸⁰ See Greene Decl. ¶ 113.

¹⁸¹ OracleUnleashed at 29.

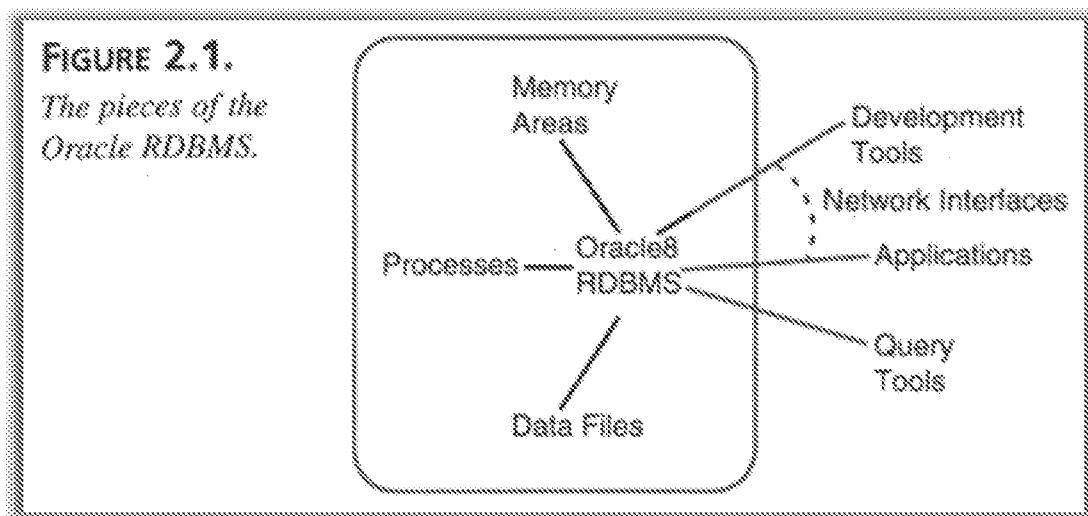
¹⁸² *Id.* at 424, 488.

¹⁸³ See Greene Decl. ¶ 114.

¹⁸⁴ OracleUnleashed at 30.

This book is more like a companion. For instance, if one found himself or herself in the predicament of needing to connect and synchronize an Oracle8 Server to an IBM main- frame running DB2, he or she probably couldn't do it with this book alone. Instead, the constant reader would first study this book to gain an understanding of the many issues surrounding this Herculean task, using the expert knowledge and examples as a template when referring to specific documentation regarding the many tasks that they faced. Once the large picture has been grasped, the reader might then dive into specific Oracle manuals in order to extract useful details needed to complete the task.¹⁸⁵

Further, OracleUnleashed provides information on some specific areas of technology. For example, OracleUnleashed explains the basic structure of the relational database management system (“RDBMS”) with three main parts: background processes, shared memory areas, and disk data storage areas, as shown below in Figure 2.1.¹⁸⁶



A key feature of Oracle’s system is network transparency. OracleUnleashed explains that Oracle’s SQL*Net (or “Net8” as it’s called in Oracle8) provides this transparency:

¹⁸⁵ *Id.*

¹⁸⁶ *Id.* at 45.

Network Transparency

This really should be called SQL*Net Transparency. You still need Oracle's standard network layer running over all your different networks to shelter each database from network differences and to provide a common protocol. SQL*Net is the portion of the gateway that handles network communication. When SQL*Net is working, there is network transparency between databases.¹⁸⁷

OracleUnleashed further describes the use of location transparency, where "not even the physical location of the data needs to be known by the user. Oracle's gateway will map a request to the different physical machines and databases on which the data resides."¹⁸⁸ As OracleUnleashed explains, "[t]o realize Location Transparency, an administrator needs to set up SQL*Net on both machines and create the proper database links, grants, and synonyms."¹⁸⁹

OracleUnleashed also explains that Net8 uses improved Name Services to map these requests:

Net8 also has improved the Name Services. SQL*Net has a Names Server to resolve names to locations. This method of naming makes it very easy to move Oracle servers and databases with no impact on the end-user community. The Net8 Name Services has the ability to cache names at the client for much faster resolution, and the various listeners and connection managers register as available with Name Services.¹⁹⁰

¹⁸⁷ *Id.* at 179.

¹⁸⁸ *Id.*

¹⁸⁹ *Id.*

¹⁹⁰ *Id.* at 452.

OracleUnleashed further provides that users utilize SQL to query the location information on a relational database like Oracle8. OracleUnleashed demonstrates how SQL enables such queries “to determine the location of your data files,” as shown below in the SQL diagram.¹⁹¹

```
SQL> select * from sys.dba_data_files;

FILE_NAME
.....
FILE_ID TABLESPACE_NAME      BYTES    BLOCKS STATUS
.....
C:\ORAWIN\OR8S\wdborbs.ora
      3  ROLLBACK_DATA          3145728    1536 AVAILABLE
C:\ORAWIN\OR8S\wdbtemp.ora
      4  TEMPORARY_DATA          2097152    1024 AVAILABLE
C:\ORAWIN\OR8S\wdbuser.ora
      2  USER_DATA                3145728    1536 AVAILABLE
C:\ORAWIN\OR8S\wdbsys.ora
      1  SYSTEM                    10485760    5120 AVAILABLE

SQL>
```

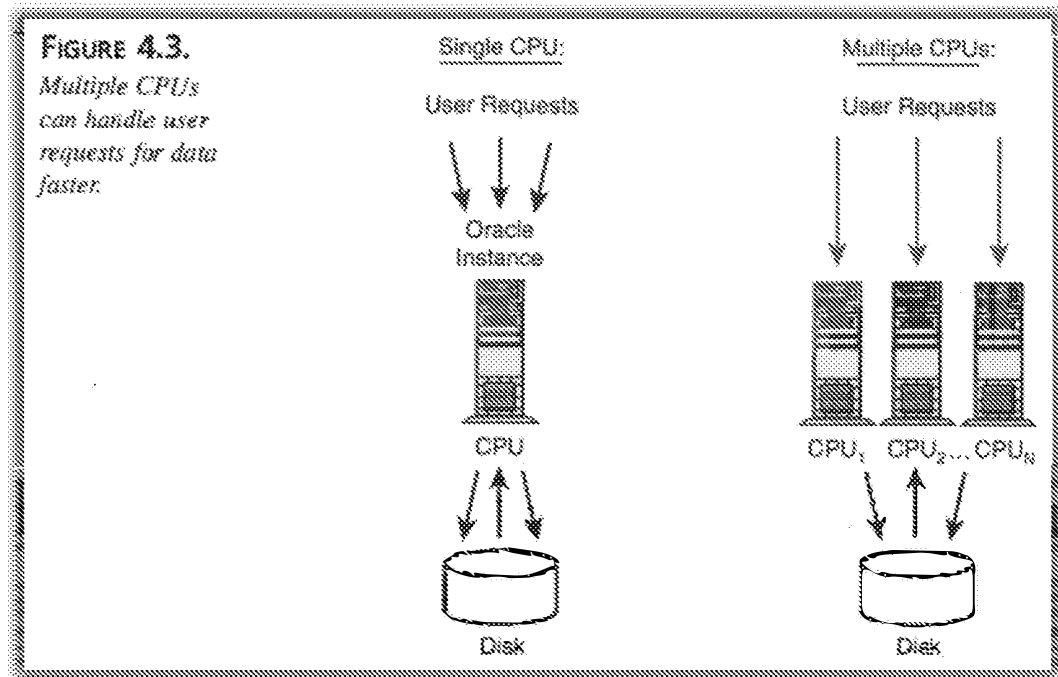
OracleUnleashed teaches ways to improve the performance of the servers themselves, including parallelism and partitioning.¹⁹² As OracleUnleashed explains, parallelism provides for multiple queries to run at the same time by cloning the database into identical copies which can process tasks separately:

This ability to run parallel queries is vital for any database in today’s market, if the database product is to utilize the full capacity of today's multiple CPU architecture. Aside from parallel queries, a database engine itself, the brain of the database, is better optimized if multiple copies of the database “brain” are each running off of different CPUs. This is a higher-level of parallelism. In Oracle, the “brain” of the database is the database instance; by running many database instances on different CPUs, you obviously can yield faster results for the many users of larger systems

¹⁹¹ *Id.* at 232.

¹⁹² *Id.* at 38-40.

that are requesting data from the database and demanding the attention of the Oracle RDBMS (see Figure 4.3).¹⁹³



Further, OracleUnleashed teaches that partitioning data can improve performance for large datasets. OracleUnleashed gives multiple examples of partitioning, starting with table partitioning:

Table partitioning. This is a big improvement when it comes to very large tables. Here you can divide a table into multiple sections based on some logical index value (divide an Orders table into sections based on month ordered, for example). Each section can be placed on a separate disk drive. One advantage to this arrangement is that Oracle is smart enough to know which sections are needed. When performing a query, it ignores the partitions that are not needed, thereby reducing the amount of data that must be scanned. It can also enhance reliability in that you can operate the database with a given section of a table down due to a failed disk drive or some other reason. You cannot issue queries that require information from the missing

¹⁹³ *Id.* at 85; Greene Decl. ¶ 122.

section; however, you can issue queries against the other section and add new data to another section.¹⁹⁴

OracleUnleashed further describes the use of index partitioning, where servers “divide an index into multiple partitions (disks) based on the values of that index. It can be very useful for large indexes in splitting disk input/output.”¹⁹⁵

Before taking these actions, OracleUnleashed teaches that the DBA must monitor databases to identify performance issues.¹⁹⁶ OracleUnleashed explains that DBAs should monitor as needed when problems arise or proactively to prevent problems from occurring:

Monitoring can be done on a continual basis or on an “as needed” basis. The more servers there are to administrate, the greater the need for constant, background-based monitoring. The more servers there are to administrate, the more global the monitoring becomes. Overworked administrators want to know only about impending disasters such as tablespaces about to fill or objects about to reach their MAXEXTENTS and about the status of certain processes. *Many items that can cause database downtime can be monitored and corrected proactively.*

There are three main reasons to monitor a UNIX and/or a database server. The first is to be proactive rather than reactive with error resolution. Downtime can be minimized if administrators know, for example, that certain dynamic objects are about to reach their MAXEXTENTS. This can easily be monitored and corrected without the end user even being aware that there was almost an application stopping problem. A reactive approach to this example would be the end user getting the MAXEXTENTS error, calling the help desk or administrator, and waiting.

The second reason is the ability to tune both the server and database environments using real-time information. Disk I/O usage, memory usage, swapping, and dozens

¹⁹⁴ OracleUnleashed at 40.

¹⁹⁵ *Id.*; Greene Decl. ¶ 124.

¹⁹⁶ OracleUnleashed at 273.

of other related processes can be monitored and adjustments made accordingly. Resource-intensive SQL, locking situations, and never-ending tasks can all be monitored and tuned accordingly.

The final reason to monitor is for capacity planning, monitoring the growth of objects and machine utilization, times of day of heavy traffic, and so on. There are sometimes contractual response time commitments, license agreements on the number of concurrent users, and the like. Monitoring can ensure compliance with various vendor and departmental agreements.¹⁹⁷

OracleUnleashed teaches DBAs should automate performance monitoring:

Just as important as monitoring and regular maintenance is auditing. Auditing is another way of monitoring the database, in that the DBA can easily measure events that occur in it. After you have set up processes to audit the database's activity and monitor its elements, you can use job queues to schedule jobs at the database level. In other words, *much of the monitoring and auditing process can be automated*.¹⁹⁸

OracleUnleashed further explains that DBAs should automate many tasks to optimize the database's performance:

Manual intervention still has its place for many tasks, but the automation of most database systems has drastically increased.¹⁹⁹

The scheduling of periodic tasks allows database users (especially the DBA) the opportunity to organize database activities effectively and often allows for the automation of many of our otherwise mundane tasks.²⁰⁰

One feature of Oracle8 that became available with Oracle 7.3 is the automatic sizing of data files option. One of the biggest problems for database administrators in the past was running out of space for a new extent within a tablespace. You had to be

¹⁹⁷ *Id.* at 471-72.

¹⁹⁸ *Id.* at 476; Greene Decl. ¶ 126.

¹⁹⁹ OracleUnleashed at 476.

²⁰⁰ *Id.* at 511.

constantly vigilant to ensure you had both enough space and large enough blocks of contiguous space to accommodate table and index growth.²⁰¹

OracleUnleashed explains that this automation of tasks is greatly preferred. It states: “Dialing in to run jobs is no fun and completely unnecessary. Even if you’re on an operating system that does not adequately support job scheduling, you can let the database take care of itself by using the DBMS_JOB package. Spend the weekend skiing instead.”²⁰²

OracleUnleashed also teaches the benefits of the well, known hashing techniques and implementing them into distributed database servers. For example, OracleUnleashed provides that SQL statements used for data requests should use hashing and indices to optimize performance. “The well-tuned SQL statement uses index- es or hashing as available. If possible, the application should also take advantage of features such as array processing and discrete transactions.”²⁰³

OracleUnleashed further describes the performance decision a DBA makes when considering hashing for various tables.

The decision to use hash clusters is an important one. Hash clusters can be beneficial because, when they are effectively used, the requested data can be retrieved in just one J/O. Unfortunately, if a hash cluster is used on a table that is not a good candidate for hashing, performance can be severely degraded.²⁰⁴

Because hashing uses the value of the data to calculate the data block the desired data is in, hashing is best used on tables that have unique values for the cluster key and that are queried primarily by equality queries on the cluster key. In the case of equality queries, the data is usually retrieved in one read operation. The cluster key

²⁰¹ *Id.* at 241.

²⁰² *Id.* at 591.

²⁰³ *Id.* at 372.

²⁰⁴ *Id.* at 394.

need not be a single column; if the typical query uses an equality on a set of columns, use these columns to create a composite key.²⁰⁵

OracleUnleashed describes how hashing data on a server results in a hash table used to look up its location based on 'unique values' for entries.

Because hashing uses the value of the data to calculate the data block the desired data is in, hashing is best used on tables that have unique values for the cluster key and that are queried primarily by equality queries on the cluster key. In the case of equality queries, the data is usually retrieved in one read operation. The cluster key need not be a single column; if the typical query uses an equality on a set of columns, use these columns to create a composite key.²⁰⁶

Finally, OracleUnleashed provides various examples of how redirect messages can help resolve requests in different scenarios. For example, OracleUnleashed describes redirects for URL requests, for maintaining password privacy, and in the OWA utility package.

The WAS has some powerful features for redirecting incoming URL requests in some unique ways.²⁰⁷

There are some security considerations with the ODBC cartridge. Since calls to the cartridge include the sql parameter, which allows any valid DML statement to be used, then a user could theoretically build his or her own URL to invoke the cartridge, with a SQL parameter such as DELETEcontacts. There are some steps you can take to prevent this, such as only allowing certain IP addresses to access the system, or turning on password security for the ODBC cartridge. If password security is used, application developers must embed the password in their URL's that call the ODBC cartridge. Therefore, developers must take extra steps to ensure that their applications' calls to the ODBC cartridge do not include the password in any

²⁰⁵ *Id.* at 395.

²⁰⁶ *Id.* at 638.

²⁰⁷ *Id.* at 698.

HTML that is written to the browser. Instead, any embedded hyperlinks to executables must first refer to a procedure that, in turn, redirects the call to the ODBC cartridge. Otherwiser, a savvy user can click the View-> View Source menu option in his or her browser, and in a couple of minutes identify the password to the ODBC cartridge from the HTML code.²⁰⁸

OWA_UTIL Very useful package that includes various utilities, including those that get CGI environment variables, generate MIME type responses, redirect URLs, print preformatted tables of data, and so forth. Lets you deal directly with HTTP.²⁰⁹

By providing skilled artisans with these key features of the Oracle8 system and explaining the considerations necessary to maximize its benefits, OracleUnleashed helps artisans to merge various functions and features and implement the full range of capabilities contained within an Oracle system.²¹⁰

D. SNQ 1: Skagerwall in View of Wolff Renders Obvious Independent Claims 17 and 18 And Dependent Claim 24.

Claims 17 and 18 recite a system having a data repository storing data and a location server storing the data's location information wherein a client may request the location of data using an identifier associated with the entity's location information. Claims 17 and 18 also recite using a redirect message to redirect a client to another location server if the initial location server lacks the location information responsive to the client's query. And claim 24 provides that the redirect message contains portions of a hash table mapping the location servers.

Additionally, based on Patent Owner's statements in the previous reexaminations, claims 17, 18, and 24 also require that the location servers be organized in a non-hierarchical cluster

²⁰⁸ *Id.* at 729.

²⁰⁹ *Id.* at 725.

²¹⁰ Greene Decl. ¶ 133.

structure, that each location server be able to respond to any location request with either the location of the data or the location of a server known to contain the requested location data, and that each request must be resolved in two or fewer traversals. As described below, Skagerwall alone or in combination with Wolff renders obvious each claim regardless of Patent Owner's arguments based on additional limitations.

Skagerwall discloses a non-hierarchical system of directory servers storing location information (i.e., location servers) wherein a client may request the location of data using an object tag (i.e., identifier) associated with the data object's server addresses (i.e., location strings/location information).²¹¹ And Skagerwall describes using a hash function to organize data according to object tags and mapping server content according to a hash table.²¹² Skagerwall further teaches or renders obvious resolving any location request within two traversals by responding to each location request with either the requested location data or a redirect message to redirect a client to another location server that contains the requested location data without an intervening traversal if the initial location server lacks the location information responsive to the client's query.

To the extent Skagerwall doesn't disclose all elements of claims 17 and 18, Wolff renders obvious using a redirect message to redirect a client to another location server. Skagerwall and Wolff both describe managing distributed networks using client-server relationships.²¹³ Skagerwall explains that minimizing "time consuming" processes is necessary to optimize performance for networks like these.²¹⁴

²¹¹ See, e.g., Skagerwall at 3:45-67.

²¹² *Id.* at 4:2-5:10.

²¹³ Wolfe at 1:41-54; Skagerwall at Abstract.

²¹⁴ Skagerwall at 2:8-12.

Similarly, Skagerwall and Wolff describe the challenges of expanding networks, while maintaining performance.²¹⁵ Wolff explains that such swift growth can overload a system, making a server a “bottleneck” in the system.²¹⁶ Skagerwall echoes this same bottleneck concern, emphasizing the need to reduce processing load whenever possible on a variety of servers.²¹⁷

So, a POSITA implementing Skagerwall’s system would have been motivated to incorporate teachings like Wolff’s toward optimizing performance by reducing processing load on network servers.²¹⁸ As Mr. Greene explains, Wolff’s “redirection command” offers one such teaching toward reducing the processing load on Skagerwall’s directory servers by moving the processing from the directory servers to the access server.²¹⁹ Thus, a POSITA would have identified Wolff’s redirection command as a technique for optimizing Skagerwall’s directory servers to use additional processing power to resolve requests intended for other directory servers. Accordingly, a POSITA would have been motivated to modify Skagerwall’s directory servers to use Wolff’s redirection command to identify another directory server having the requested data then sending a redirect message containing the location of that other server.²²⁰

Further, a POSITA would have had a reasonable expectation of success combining Skagerwall’s system with Wolff’s redirection messages.²²¹ As Mr. Greene details, a POSITA would have expected that the access servers like those in Skagerwall could calculate the correct

²¹⁵ *E.g., id.* at 1:59-62.

²¹⁶ Wolff at 2:1-2.

²¹⁷ Skagerwall at 5:31-36, 18:54-58.

²¹⁸ *See* Greene Decl. ¶ 140.

²¹⁹ *Id.*

²²⁰ *Id.*

²²¹ *Id.* ¶ 141.

directory server just as well as the originally queried directory server, because Skagerwall teaches the identification step can be completed by any server in the kernel.²²²

1. Skagerwall in View of Wolff Renders Obvious Independent Claims 17 and 18.

During the original examination, the Examiner indicated that the prior art of record didn't teach or suggest the "hash function" or "redirect" limitations in elements of claims 17, 18, and 24.²²³ Skagerwall discloses or renders obvious alone or in view of Wolff these limitations because Skagerwall discloses using a hash function to organize location information and generating a hash table and Wolff discloses the use of redirection to prevent network failure.

- a) *Claim elements 17.pre and 18.pre: "A system for retrieving data location information for data stored in a distributed network, the system comprising:"*

Claim elements 17.pre and 18.pre are the same. To the extent any preamble is limiting, Skagerwall discloses a database system for managing data stored in a distributed network and method of using the same. Skagerwall describes a "kernel" which retrieves data objects (D) from storage repositories (AI) on application servers (IS).²²⁴ As Skagerwall describes, the directory servers are a "key resource" in retrieving data:

Advantageously, each of a plurality of directory servers stores part of an arbitrarily scaleable data base containing object data associating each of the objects with at least one of the application data packets. *The plurality of directory servers serve as key resource in retrieving an application data packet related to an object.* Means

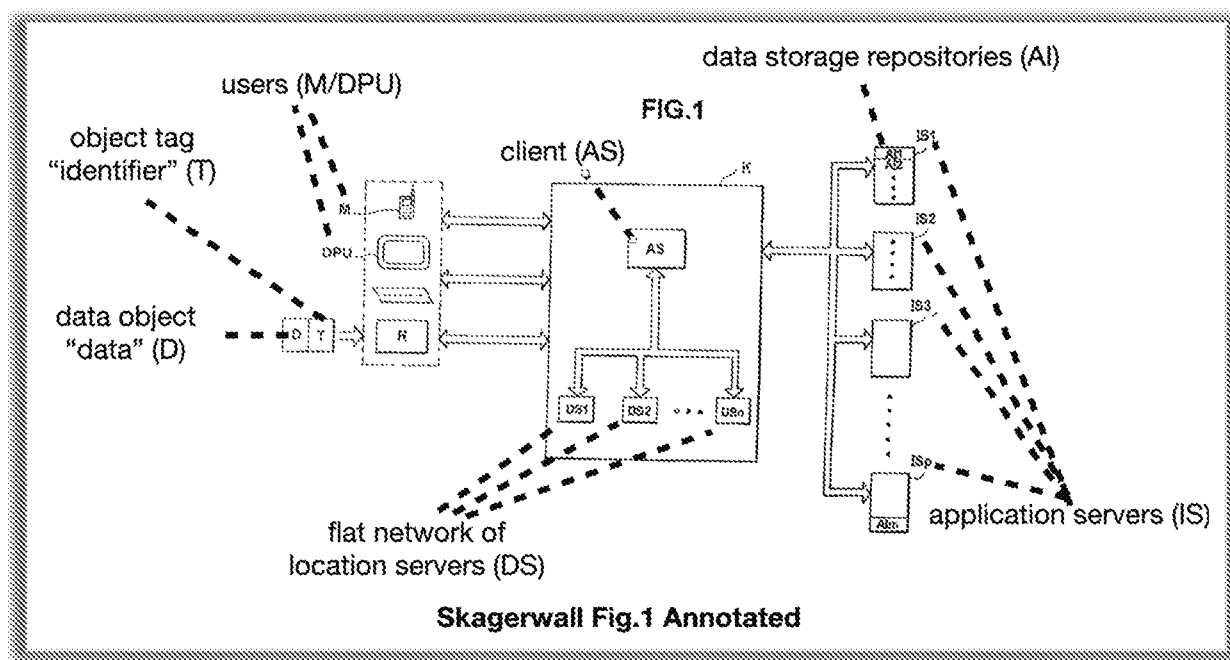
²²² *Id.*

²²³ Ex. 9 ('640 History) at 442, 460.

²²⁴ Skagerwall at 2:51-3:2.

for receiving a read object tag from a reading means and for identifying the directory server for storing object data related to the read object tag and retrieving the object data are provided.²²⁵

Further, as shown below in annotated Figure 1, Skagerwall teaches that user devices (M/DPU) make requests for data object (D) to an “access server” (AS) inside the kernel using an identifying object tag (T).²²⁶ Skagerwall explains that the access server then requests retrieval information from a plurality of directory servers (DS) to locate the requested data.²²⁷ A POSITA would have understood Skagerwall’s retrieval information to be “location information” because it contains information pertaining to the location of data.²²⁸



²²⁵ *id.* at 4:12-20.

²²⁶ *Id.* at 2:51-3:2.

²²⁷ *Id.*

²²⁸ See Greene Decl. ¶ 144; Ex. 17 (Joint Claim Construction Chart) at 4-6.

As Mr. Greene explains, a POSITA would have understood Skagerwall's kernel to be a system for retrieving data location information for data stored in a distributed network.²²⁹

Thus, based on the above teachings, Skagerwall discloses or renders obvious the preambles of claims 17 and 18.

b) *Claim elements 17.a and 18.a: "a plurality of data repositories/a data repository configured to store data," / "a data repository configured to store data,"*

Claim elements 17.a and 18.a are substantially similar. Skagerwall discloses or alone renders obvious these claim elements. Skagerwall teaches that its kernel retrieves data from "a plurality of application servers" storing data objects (D).²³⁰ As Skagerwall describes, each application server (IS) contains "memory means for storing a plurality of application data packets, each related to one of the objects."²³¹ As Mr. Greene explains, a POSITA would understand the "object" to be a "data entity and the data entity to be the collection of associated application data packets. Further, a POSITA would have understood that such application servers with storage means would in turn constitute "data repositories configured to store data."²³²

A POSITA would have understood the servers and databases to be "data repositories" in Skagerwall's system because they are locations to store data.²³³ Based on the above teachings, Skagerwall discloses or alone renders obvious claim elements 17.a and 18.a's data repository configured to store a data entity.

²²⁹ See Greene Decl. ¶ 145.

²³⁰ *Id.* at 2:51-3:2.

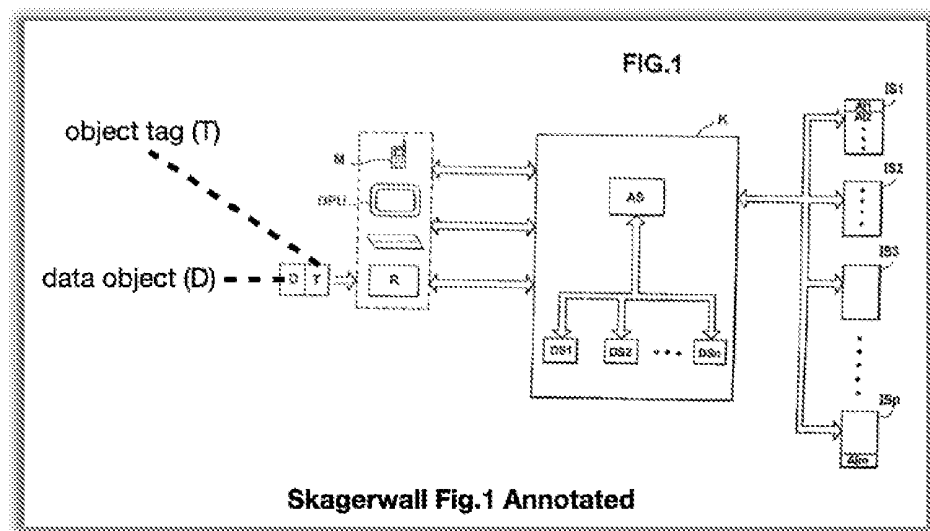
²³¹ *Id.* at 2:60-63.

²³² See Greene Decl. ¶ 147.

²³³ *Id.*

- c) *Claim elements 17.b and 18.b: “wherein the data is associated with a respective identifier string in each data repository;”/” wherein the data is associated with an identifier string;”*

Claim elements 17.b and 18.b are substantially similar. Skagerwall discloses or alone renders obvious these claim elements. Skagerwall teaches storing its data objects anywhere on the network in arbitrary locations, which makes locating them more complex.²³⁴ As explained above regarding claims 17 and 18’s preambles, Skagerwall provides the plurality of directory servers to manage this process. Skagerwall teaches an “object tag” associated with each data object and used to identify the object on the network. As shown below in annotated Figure 1, Skagerwall’s object tag and object data are associated with each other upon entry to the system.



Skagerwall describes using the object tag to identify and organize the data objects across the system: “The object data is distributed according to a defined scheme onto the plurality of directory servers in order to be able to identify the directory server storing object data associated with a particular read object tag.”²³⁵ As Mr. Greene explains, a POSITA would have understood

²³⁴ Skagerwall at Abstract.

²³⁵ *Id.*

the object tag to be the claimed “identifier string” because each object tag is a unique encoding identifying a data entity, and is associated with an application server address (i.e., a location string).²³⁶

Thus, based on the above teachings, Skagerwall discloses or renders obvious claim elements 17.b and 18.b’s location server storing a plurality of identifier strings each identifying a data entity.

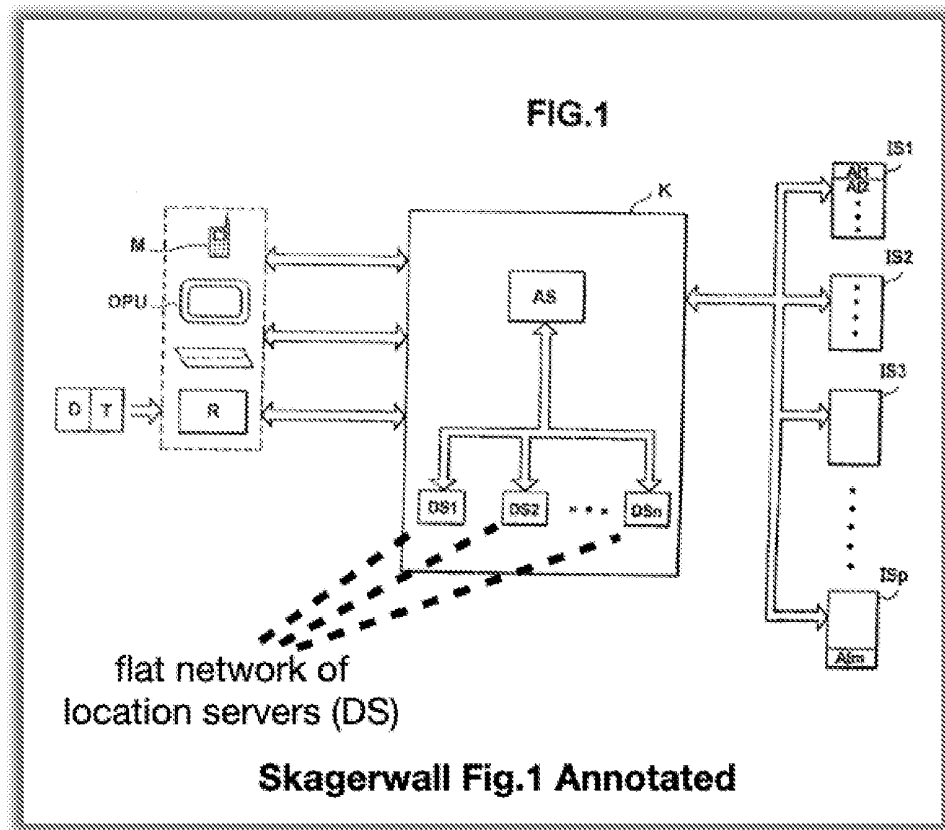
d) *Claim elements 17.c and 18.d: “a data location [server/network] [having/comprising] a plurality of data location servers,”*

Claim elements 17.c and 18.d are substantially similar. Skagerwall discloses or alone renders obvious these claim elements. Skagerwall discloses that the kernel contains “a plurality of directory servers.”²³⁷ As shown below in annotated Figure 1, Skagerwall’s directory servers (DS1 ... DS_n) have a flat, non-hierarchical architecture with no hierarchical or tree structure.²³⁸

²³⁶ See Greene Decl. ¶ 150.

²³⁷ Skagerwall at 2:59-63.

²³⁸ See Greene Decl. ¶ 152.



As Figure 1 illustrates, the directory servers have a flat configuration next to one another, with no hierarchical arrangement.²³⁹ And Figure 1 teaches that each directory server communicates with the access server (AS) with no intermediary servers in between. Further, Skagerwall doesn't describe any "root" or "parent" directory servers that could indicate a hierarchy.²⁴⁰ Nor is there any discussion of separate regions or domains for different directory servers. Accordingly, a POSITA would understand that the directory servers in Skagerwall are non-hierarchical and the directory server network is flat.²⁴¹

²³⁹ *Id.*

²⁴⁰ *Id.*

²⁴¹ *Id.*

Further, Skagerwall explains that the directory servers contain information for locating the data objects stored on the data repositories of the application servers (IS1 ... ISp):

It is a key aspect of the invention that the data base of object data is distributed over a plurality of directory servers DS; DS1-DSn, allowing that the number of objects can be scaled almost arbitrarily by adding new directory servers to the existing ones for storing new object data. Each of the directory servers DS; DS1-DSn preferably stores object data related to a small number of the range of available object tags T; T1-Tw. As mentioned above, object data related to an object tag preferably includes application identifiers AI1-Aim of application data packets related to the object *as well as information about storage locations of the application data packets, e.g., the addresses of application servers IS; IS1-IsP storing the application data packets.*²⁴²

As Mr. Greene explains, because the directory servers function to store location information for the object data, a POSITA would understand those directory servers to be the claimed “location servers.”²⁴³

Thus, based on the above teachings, Skagerwall discloses or renders obvious claim elements 17.c and 18.d’s server network comprising a plurality of data location servers.

e) *Claim elements 17.d and 18.e: “each of the plurality of data location servers containing location strings associated with respective identifier strings and” / “at least one of the plurality of data location servers containing location information associated with the identifier string,”*

Claim elements 17.d and 18.e are substantially similar. Skagerwall discloses or alone renders obvious these claim elements. As explained above regarding claim elements 17.c and 18.d, Skagerwall teaches a plurality of data location servers storing object data according to object

²⁴² Skagerwall at 8:15-27.

²⁴³ See Greene Decl. ¶ 154.

tags. Skagerwall explains that each object tag (i.e., identifier string) is associated with the application server address (i.e., location strings) for the relevant data object:

As mentioned above, object data related to an object tag preferably includes application identifiers AI1-Aim of application data packets related to the object as well as information about storage locations of the application data packets, e.g., the addresses of application servers IS; IS1-Isp storing the application data packets.²⁴⁴

Further, as Mr. Greene explains, a POSITA would have understood Skagerwall's application server addresses (i.e., location strings) to be a form of location information that the system uses to retrieve requested data.²⁴⁵ Thus, as Mr. Greene explains, a POSITA would have understood that each of Skagerwall's directory servers stores application server addresses (i.e., location strings and location information) associated with the respective object tags (i.e., identifier string).²⁴⁶

Based on the above teachings, Skagerwall discloses or renders obvious claim elements 17.d and 18.e's at least one of the plurality of data location servers includes location information associated with the identifier string.

f) *Claim element 18.c: "a client responsive to a data query to query a data location server for location information associated with the identifier string"*

Skagerwall discloses or alone renders obvious claim element 18.c. As described above regarding claim elements 17.d and 18.e, Skagerwall discloses a plurality of directory servers (i.e., data location servers) storing application server addresses (i.e., location string/location in-

²⁴⁴ Skagerwall at 8:15-27.

²⁴⁵ See Greene Decl. ¶ 157.

²⁴⁶ *Id.*

formation) associated with object tags (i.e. identifier strings) for data entities. Skagerwall explains that a user device, such as a mobile phone, will request a data entity from the kernel using that entity's object tag (i.e., identifier string).²⁴⁷ Skagerwall teaches that the access server responds to this request by querying a directory server (i.e., location server) for the requested data entity's location where that query includes the object tag (i.e., identifier string):

After receiving a request for information including the object tag T, the access server AS; AS1-Asr performs operations to identify the directory server DS; DS1-DSn storing information concerning application data packets related to the object. This preferably includes sending the request including the object tag T; T1-Tw to one of the directory servers DS1; DS2; . . . ; DSn...²⁴⁸

As Mr. Greene explains, Skagerwall's interaction between the user device, access server, and directory servers is a form of "client-server" interaction, where the access server (AS) functions as the client to the directory servers by requesting the data object's address on the application server (IS).²⁴⁹ Thus, a POSITA would have understood that Skagerwall's access server corresponds to the claimed "client" because it is a network-attached component that uses location request messages to initiate the lookup of location mappings from the directory servers (i.e., location servers).²⁵⁰

Based on the above teachings, Skagerwall discloses or renders obvious claim element 18.c's storing and retrieving data location information for a plurality of data entities in the data location server network.

²⁴⁷ Skagerwall at 9:18-28.

²⁴⁸ *Id.*

²⁴⁹ See Greene Decl. ¶ 160.

²⁵⁰ *Id.*

- g) *Claim elements 17.e and 18.f: “each of the plurality of data location servers having computer executable code configured to execute the following steps:” / “wherein each of the plurality of data location servers comprises computer executable code configured to execute the following steps in response to receiving a data location request from the client”*

Claim elements 17.e and 18.f are substantially similar. Skagerwall discloses or alone renders obvious these claim elements. Skagerwall teaches that its directory servers have a means to execute a variety of processes including the storing of location data, receiving requests, identifying correct directory servers, and retrieving stored data:

[Directory servers]...including memory means (TM; AM) for storing in a distributed data base object data associating each of the objects with at least one of the application data packets; means for receiving the read object tag (T) from the reading means I and means for identifying the directory server for storing object data related to the read object tag and retrieving the object data; and means for retrieving at least one of the application data packets based on the object data or transmitting at least part of the object data to a receiving unit (M, DPU).²⁵¹

As disclosed, the directory servers receive requests from clients for object data for retrieving data objects, i.e., location information, associated with object tags, i.e., identifiers.

Further, as Mr. Greene explains, for elements 17.e and 18.f’s claimed processor, a POSITA would have understood that, for the directory servers to do any of these tasks, there must be executable code to complete the processes.²⁵² Thus, a POSITA would have understood that each directory server (i.e., location server) includes computer executable code for performing Skagerwall’s disclosed tasks.²⁵³

²⁵¹ Skagerwall, 2:60-3:2.

²⁵² See Greene Decl. ¶ 163.

²⁵³ *Id.*

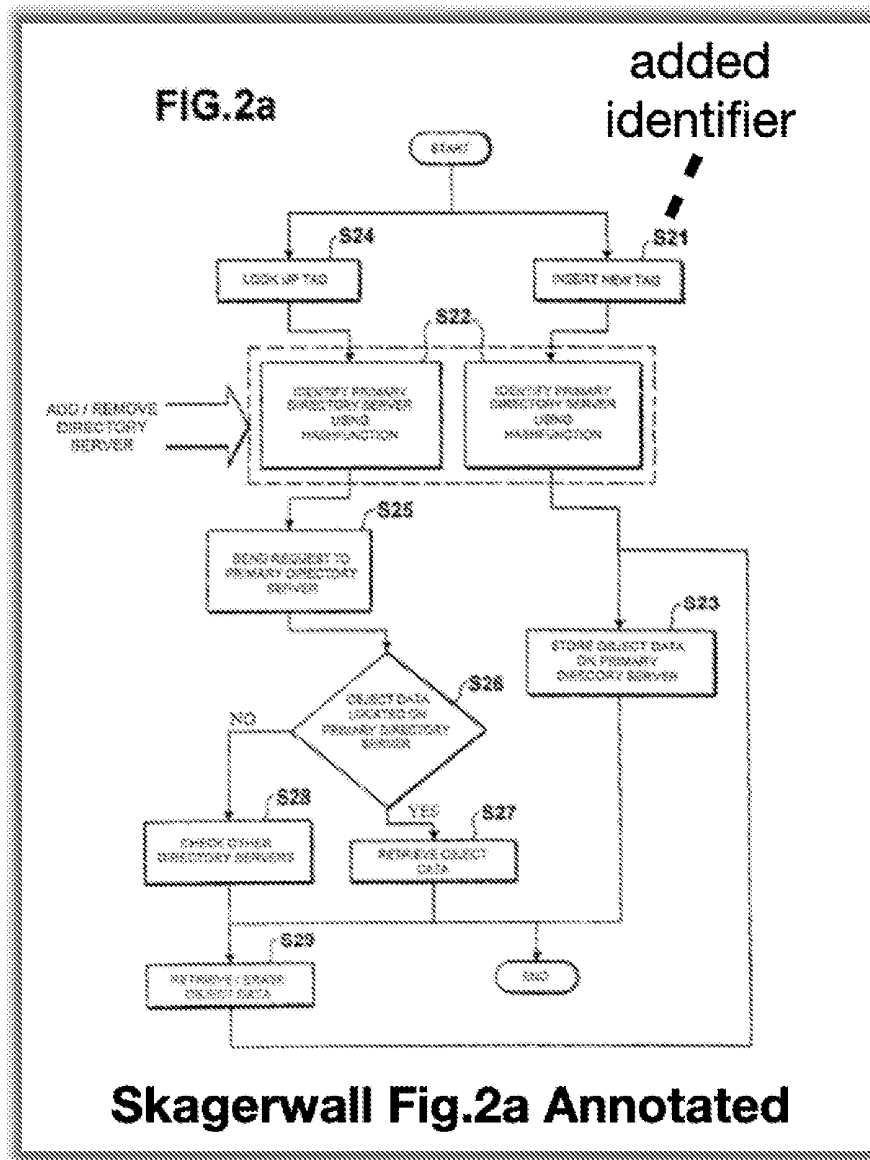
Skagerwall also discloses or renders obvious claim 17.e's and 18.f's "transport protocol" limitation. Specifically, Skagerwall teaches the use of standard protocols with the system.

It is further possible, that the system and method according to the invention uses standard and/or amended protocols of a known datacom- or telecom network for retrieving information associated with an object.²⁵⁴

As Mr. Greene explains, a POSITA would have understood a transport protocol to be one of these standard protocols. As he further explains, such transport protocols were well-known, routine, and well-understood to be used for sending messages and data over a network. Thus, to any extent that Skagerwall does not disclose the claimed transport protocol, it would have been obvious to a POSITA at this time to use one.

Further, Skagerwall discloses claim element 18.c's "location addition request" limitation. Specifically, in Fig. 2a, Skagerwall teaches the addition of a new object tag (i.e., identifier) into the network of directory servers in S21.

²⁵⁴ Skagerwall, 9:59-62.



Skagerwall teaches that in “step S21 the system is advised to insert new object data into its database distributed on the plurality of the directory servers.”²⁵⁵ As Mr. Greene explains, a POSITA would have understood that the system would be advised with a request, such as the claimed location addition request. Accordingly, a POSITA would understand the new object data

²⁵⁵ *Id.* at 12:5-8.

(i.e., new location information) from S21 would be included in a request with the object tag (i.e., identifier) shown in S21 of Fig. 2a. Thus, Skagerwall teaches a location addition request including location information and an associated identifier to be added to the location server network.

Skagerwall also discloses or renders obvious claim elements 17.e's and 18.f's "location addition request" limitation. Specifically, Skagerwall explains that the access server (i.e., client) sends a request to the directory server and that the directory server will respond "in turn" to the request with specific steps:

This preferably includes sending the request including the object tag T; T1-Tw to one of the directory servers DS1; DS2; . . . ; DS_n, which in turn, using the hash-function and hashtable H, determines the directory server for storing the object tag T; T1-Tw and forwards the request to that particular directory server. However, it is also possible, that the access server AS; AS1-AS_r or other components of the system kernel K perform steps for identifying the correct directory server.²⁵⁶

Thus, based on the above teachings, Skagerwall discloses or renders obvious claim elements 17.e and 18.f's data location requests with identifiers to location servers storing associated location information.

*h) Claim element 18.g: "if the data location server contains the location string associated with the identification string provided in the data location request, the data location server transmits location information for use by the client to calculate a location of the data associated with the identification string;"*²⁵⁷

Skagerwall discloses or alone renders obvious these claim elements. Skagerwall explains that, when a directory server has retrieved the address for the requested object tag, the directory server returns that address to the access server, i.e., client:

²⁵⁶ *Id.* at 9:18-28.

²⁵⁷ While unclear, it is assumed that the identification string is the same as the identifier string in this analysis.

The obtained addresses IS; IS1-ISp and optionally other information can be returned to the access server AS; AS1-ASr, which in the following retrieves the specified application data packets from the identified application servers or transmits the addresses IS; IS1-ISp and possibly other information directly to the user, i.e., the mobile telephone M or the data processing unit DPU.²⁵⁸

As Mr. Greene explains, a POSITA would have understood that Skagerwall's "retriev[ing] the specified application data packets from the identified application servers" includes returning the location information to the client. To "retrieve the specified data packets" associated with an object tag (i.e., identification string), the access server (i.e., client) makes the request for data "from the identified application servers."²⁵⁹ A POSITA would have understood that, to make such a request, the client must use the location information associated with the object tag to calculate a location of the data associated with the identification string.²⁶⁰

Based on the above teachings, Skagerwall discloses or renders obvious claim elements 18.g's responding to data location requests with the associated location information if contained by the location server.

- i) *Claim element 18.h: "if the data location server does not contain the location string associated with the identification string, the location server transmits a redirect message to the client, wherein the redirect message contains redirect information for use by the client to calculate a location of a different data location server in the plurality of data location servers, wherein the different data location server contains the location string."*

Skagerwall discloses this claim element or renders it obvious alone or in view of Wolff. As described regarding claim element 18.g, for each data request, Skagerwall teaches that a directory server identifying if it is the primary directory server (i.e., location server) containing the

²⁵⁸ Skagerwall at 9:42-47.

²⁵⁹ *Id.* at 9:42-47.

²⁶⁰ See Greene Decl. ¶ 171.

application server address (i.e., location string) for the request. And Skagerwall provides instructions that, if the queried directory server doesn't contain the object tag for the requested data, the system progresses to flowchart step S28 where the request goes to other directory servers:

If in step S26 it has been determined that object data related to the object tag is not located at the identified primary directory server, in a step S28, the lookup request is forwarded to other directory servers. In case the number of directory servers has been increased recently, the lookup request may be forwarded to a directory server which is determined using the hashfunction based on a previous number of available directory servers. The same holds true, if the number of directory servers has been decreased recently. If none of the directory stores the desired object data, an error message is generated and the operation ends.²⁶¹

Skagerwall discloses this claim element through its use of the "hashfunction." As discussed above, Skagerwall teaches that "[i]ntermediate steps are possible, where the look up request is relayed to further devices before it reaches the correct directory server, as outlined later."²⁶² As Mr. Greene explains, when providing the access server (i.e., client) with a hashfunction for identifying the primary directory server to the access server, a POSITA would have understood Skagerwall's "intermediate steps" include a redirect message having information to calculate the location of the directory server containing the location string for the requested data.²⁶³ This arises when system efficiency dictates lessening the processing burden of the directory servers.²⁶⁴

Further, to the extent Patent Owner argues Skagerwall's optional intermediate steps don't require redirection, a POSITA would have found it obvious for Skagerwall's system to look to

²⁶¹ Skagerwall at 12:63-13:6.

²⁶² *Id.* at 9:39-41.

²⁶³ *See* Greene Decl. ¶ 174.

²⁶⁴ *Id.*

other directory servers to resolve a data request because failure to resolve this request is contrary to the system's objectives and desired efficiency.²⁶⁵ Skagerwall provides the exemplary forwarding step from the initial primary directory server to the identified directory server having the location of the requested data:

After identification of the correct directory server DS; DS1-DSn, the request, including the object tag T; T1-Tw, is forwarded to the identified directory server. The directory server, in a subsequent step, performs operations to determine all addresses of application servers IS; IS1-ISp storing application data packets associated with the read object tag T; T1-Tw and thus object D. Intermediate steps are possible, where the look up request is relayed to further devices before it reaches the correct directory server, as outlined later.²⁶⁶

Skagerwall teaches that exactly which kernel component performs the identification steps is a design choice by explaining that any component can complete this step: "[O]ther components of the system kernel K may serve as means for identifying the correct directory servers."²⁶⁷ Accordingly, as Mr. Greene describes, a POSITA would have understood that either the queried directory server, i.e., location server, or the access server (i.e., client) could perform the identification step depending on the needs of the system and its specific design.²⁶⁸

Thus, it would have been an obvious matter of design choice for Skagerwall's queried directory server to send to the access server (i.e., client) information for contacting another directory server having the location of the requested data rather than having the queried directory server forward the request to the other directory server. For example, Mr. Greene explains that

²⁶⁵ See *id.* ¶ 175.

²⁶⁶ Skagerwall at 9:32-49.

²⁶⁷ *Id.* at 10:21-26.

²⁶⁸ See Greene Decl. ¶ 176.

for a system where there is only one access server, it may have limited processing power so that any extra burden should be minimized. Mr. Greene explains that, in such a system, a POSITA would have understood the benefits of having the queried directory server find the correct directory server and obtain the desired location information, and return the addresses (i.e., location strings) to the access server.²⁶⁹ Conversely, as Mr. Greene describes, for a system with multiple access servers, the processing power may be more limited on the directory servers. In such a system, Mr. Greene explains, a POSITA would have understood that it's more efficient to minimize the processing requirements of the queried directory server. Thus, for such a system with multiple access servers, a POSITA would have been motivated to design the system such that the queried directory server returns to the pertinent access server (i.e., client) the information to calculate the location of the primary directory server.²⁷⁰ As Mr. Greene explains, it would have been understood to a POSITA that the system would return such information in a "redirect message."

271

Further, Wolff's teachings render obvious the claimed "redirect message" limitation. Wolff explains that servers will not always be able to fulfill each request received from a given client. For example, Wolff describes an "overloaded" server which has received too many requests and lacks the processing power to fulfill them all:

In operation at time T=0, normal client 100A is shown accessing memory resource 118 via path 70 through overloaded server 104. At the same time, aware client 102A is shown accessing memory resource 18, via path 74, through overloaded server 104A. At time T=1, process 102P1, implemented on aware client 102A, detects the

²⁶⁹ See *id.* ¶ 177.

²⁷⁰ See *id.*

²⁷¹ See *id.*

overload condition of server 104A, and accesses memory resource 118 via an alternate path 76 through server 106A. Thus, in this subsequent state, the load on server 104A is reduced and the access by aware client 102A to memory resource 118 is enhanced. Normal client 100A cannot initiate the processes 35 discussed above in connection with the aware client 102A and is unable to select an alternate path 72 to the underutilized server 106A.²⁷²

And Wolff teaches optimizing network performance by redirecting requests from an overloaded server (e.g., a server that can't satisfy the request) to another server that can satisfy the request with a "redirection command" to the client from the overloaded server:

Client load rebalancing refers to the ability of a client, enabled with processes in accordance with the current invention, to re-map a path through a plurality of nodes to a resource. *The re-mapping may take place in response to a redirection command emanating from an overloaded node, e.g. server.* The embodiments disclosed allow more efficient, robust communication between a plurality of clients and a plurality of resources, via a plurality of nodes. Resources can include, but are not limited to, computers, memory devices, imaging devices, printers, and data sets. A data set can include a database or a file system.²⁷³

As Mr. Greene explains, a POSITA would have understood that Wolff's client re-maps the request path according to a redirection message from the overloaded server to the client.²⁷⁴ Thus, for any server that can't resolve a request, Wolff teaches that the server sends the client a redirect message pointing the client to another server that will resolve the request.²⁷⁵

So, based on Wolff's teachings, a POSITA would have found it obvious to have Skagerwall's queried location server transmit a redirect message to the access server (i.e., client)

²⁷² Wolff at 5:25-38.

²⁷³ *Id.* at Abstract.

²⁷⁴ See Greene Decl. ¶ 181.

²⁷⁵ See *id.*

if that location server doesn't contain the location string associated with the identification string wherein the redirect message has information for use by the access server to calculate a location of another directory server containing the location string. A POSITA would have been motivated to do so because this would optimize performance of Skagerwall's system to send the retrieval instructions to the client in the redirect message rather than letting the request fail at the first directory server.²⁷⁶ Doing so would further Skagerwall's goal to "facilitate the retrieval of information."²⁷⁷ Without incorporating Wolff's redirection command message, the request for location information would fail if the directory server didn't contain the requested data, thus frustrating Skagerwall's goal of "facilitating the retrieval of information."²⁷⁸ Allowing failures to occur would be inefficient for the system and cause less reliability for successful data retrieval. A POSITA would have been motivated to incorporate Wolff's redirection techniques in Skagerwall's system to provide the user with the most reliable and beneficial system of data retrieval possible.²⁷⁹

Because each location server can determine the location of location information on the location server network (i.e., "knows" where to find any requested location data), Skagerwall alone or in combination with Wolff renders obvious that each location server can resolve a data location request in two or fewer traversals.²⁸⁰ In the first traversal (i.e., first step), the client requests its desired location information from any location server, which responds with either the

²⁷⁶ See *id.*.

²⁷⁷ Skagerwall at Abstract.

²⁷⁸ See Greene Decl. ¶ 181.

²⁷⁹ See *id.*

²⁸⁰ See *id.* ¶ 182.

location information of the data or the location information for a location server known to contain the requested location information.²⁸¹ In the second traversal (i.e., second step), if necessary, the client uses the redirect message to make an alternate request (e.g., Wolff's redirected alternate path²⁸²) for the location information from another location server (e.g., Skagerwall's S28 checking other servers²⁸³) known to contain the desired location information. Thus, Skagerwall alone or in combination with Wolff renders obvious resolving any client's data location request in two or fewer traversals.²⁸⁴

Thus, based on the above teachings, Skagerwall and Wolff disclose or render obvious claim element 18.h.

- j) *Claim element 17.f: "in response to receiving a data location request from a client to retrieve a location string associated with an identification string provided in the data location request, transmitting a redirect message to the client if the identification string is not associated with a location string at the data location server, wherein the redirect message contains information for use by the client to calculate a location of a different data location server in the plurality of data location servers, wherein the different data location server contains the location string."*

Claim elements 17.f is substantially similar to claim elements 18.g and 18.h. Accordingly, Skagerwall alone or in view of Wolff discloses this claim element or renders it obvious alone or in view of Wolff.²⁸⁵ As described regarding claim element 18.g, Skagerwall discloses servicing a data location query for the address (i.e., location string) associated with an object tag (i.e., identifier string) from an access server (i.e., client) to a directory server (i.e., location

²⁸¹ Skagerwall at 12:9-53.

²⁸² Wolff at 5:25-38.

²⁸³ Skagerwall at 12:63-13:6.

²⁸⁴ See Greene Decl. ¶ 182.

²⁸⁵ See *id.* ¶ 184.

server).²⁸⁶ And as explained regarding claim element 18.h, Skagerwall and Wolff disclose or render obvious that, if the directory server doesn't contain the desired addresses, it sends a redirect message to the client having information for the client to identify the directory server containing the address associated with requested data.²⁸⁷

Thus, based on the above teachings, Skagerwall and Wolff disclose or render obvious claim element 17.f's in response to receiving a data location request from a client to retrieve a location string associated with an identification string provided in the data location request, transmitting a redirect message to the client if the identification string isn't associated with a location string at the data location server, wherein the redirect message contains information for use by the client to calculate a location of a different data location server in the plurality of data location servers, wherein the different data location server contains the location string.

In sum, Skagerwall and Wolff render obvious independent claims 17 and 18.

2. Skagerwall Alone or in View of Wolff Renders Obvious Dependent Claim 24.

Skagerwall alone or in view of Wolff renders obvious this claim. Claim 24 recites: "The system of claim 18, wherein the location information comprises a portion of a hash table distributed over the plurality of data location servers." As described above, Skagerwall and Wolff render obvious claim 18.

During the prior reexamination, Patent Owner asserted that the prior art's hierarchical configuration fails to teach the requirements of claim 24. Skagerwall teaches a system having a

²⁸⁶ See *id.*

²⁸⁷ See *id.*

non-hierarchical configuration that renders obvious claim 24 alone or in view of Wolff's teachings.

Skagerwall further teaches that object tags on its directory servers are organized according to hashing methods, including a hash table based on a hash function. For example, Skagerwall describes "storing a list of a plurality of available directory servers in a hashtable; mapping all object tags employing a hashfunction and the hashtable onto the directory servers."²⁸⁸

As explained in claim element 18.h, Skagerwall alone or in view of Wolff, renders obvious sending a redirect message with information necessary to identify the directory server containing the object data for a given object tag. As Mr. Greene explains, the mapping information or the relevant portion of the hash table, would be necessary for the access server (i.e., client) to identify the correct directory server and send the request to it.²⁸⁹

As shown below in annotated Figure 2, Skagerwall further provides examples of a hashtable showing the available directory servers and their contents:

In the example of FIG. 2 the hashtable comprises 8 entries, addresses of available servers are stored as records in respective sequential entries of the hashtable. Thus, the hashtable constitutes an ordered list of all available directory servers, as shown in FIG.2. The records stored in the hashtable H are indicated by A, B, C and D stand for addresses of four directory servers DS; DS1-DSn. However, any other order is possible, e.g., D, B, A, C.

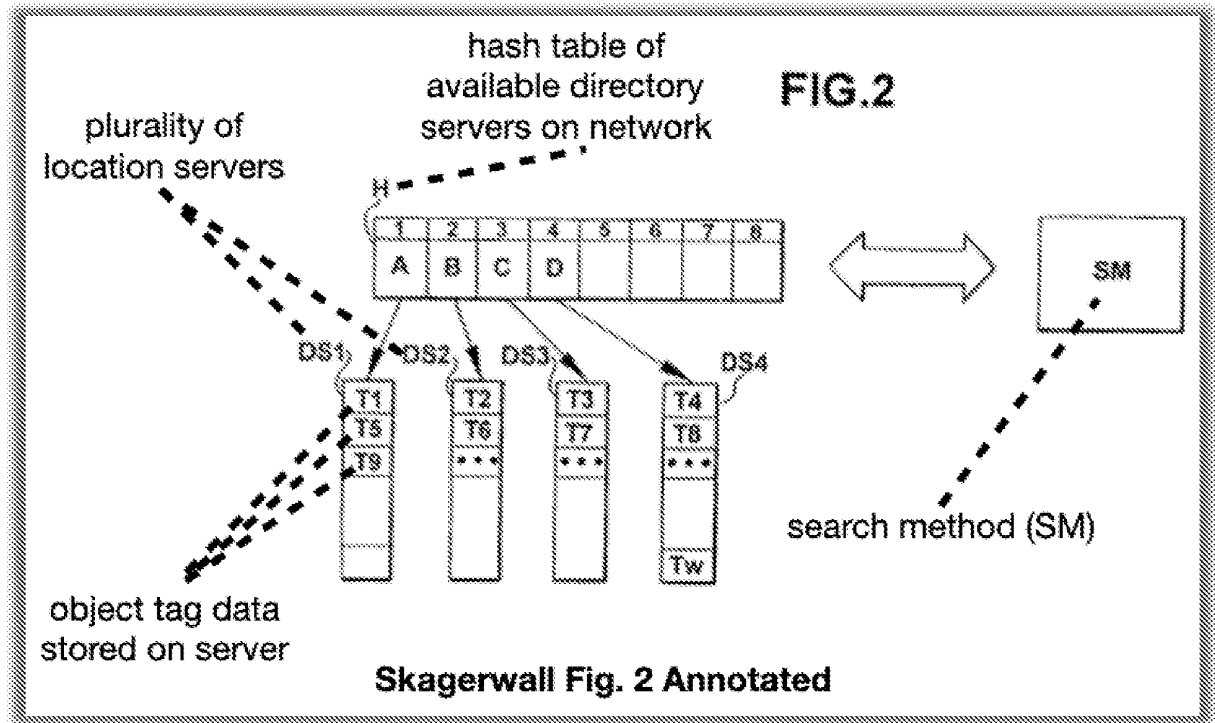
To avoid unnecessary lookup operations for object tags at directory servers, as outlined above, it is preferred that object data is stored on directory servers according to the hashfunction, which maybe executed by search means SM indicated in FIG.2.

The search means SM groups object tags, preferably in correspondence with the

²⁸⁸ Skagerwall at 3:51-54.

²⁸⁹ See Greene Decl. ¶ 190.

number of objects and the number of available directory servers, i.e., a target directory server for storing as well as for retrieving object data is determined based on an identifier of an object tag, which, e.g., maybe a number assigned to the tag, and the number of available directory servers DS;DS1-DSn.²⁹⁰



Based on Skagerwall's teachings, a POSITA would have appreciated that the directory servers store the hash table and that the hash table contains the list of available directory servers with their respective contents. Further, a POSITA would have understood that the access server (i.e., client) uses portions of the hash table to identify which directory server to send its data request based on the information in the redirect message. As Mr. Greene explains, a POSITA would have concluded that sending the entire hash table or storing the entire hash table at the client would be too cumbersome.²⁹¹ This would have frustrated Skagerwall's goal of facilitating

²⁹⁰ Skagerwall at 10:31-49.

²⁹¹ See Greene Decl. ¶ 192.

data retrieval by lowering the efficiency of the system and Wolff's goals for optimizing network performance and reliability. As a result, a POSITA would have found it obvious to break the hash table into portions, to distribute those portions to the directory server, and to send only the relevant portion to the client. Further, a POSITA would have understood that this portion includes at least the portion describing the object tag requested by the client. As a result, a POSITA would have understood that Skagerwall teaches or renders obvious that all relevant directory server contents be stored in a hash table across the directory server network.²⁹²

In sum, as Mr. Greene explains, Skagerwall, alone or in view of Wolff, renders obvious claim 24.

E. SNQ 2: Claims 17, 18, and 24 Are Obvious Based on OracleNamesAdmin-Guide in View of OracleUnleashed and Steen.

Claims 17 and 18 recite a system having a data repository storing data identified by an identifier string, and multiple location servers storing location information for multiple data entities wherein at least one location server includes location information associated with the identifier string and each location server stores a portion of the location information based on a hash function. Claim 24 depends from claim 18 and recite the distribution of location information across servers using an index location store organized according to a hash table.

Claims 17 and 18 recite a system having a data repository storing data and a location server storing the data's location information wherein a client may request the location of data using an identifier associated with the entity's location information. Claims 17 and 18 also recite using a redirect message to redirect a client to another location server if the initial location server

²⁹² See *id.*

lacks the location information responsive to the client's query. Claim 24 provides that the redirect message contains portions of a hash table mapping the location servers.

Additionally, based on Patent Owner's statements in the previous reexaminations, claims 17, 18, and 24 also require that the location servers be organized in a non-hierarchical cluster structure, that each location server be able to respond to any location request with either the location of the data or the location of a server known to contain the requested location data without an intervening traversal, and that each request must be resolved in two or fewer traversals. As described below, OracleNamesAdminGuide alone or in combination with Steen or OracleUnleashed renders obvious each claim regardless of Patent Owner's arguments based on additional limitations.

OracleNamesAdminGuide discloses systems with varied structures, including hierarchical, non-hierarchical, and mixed topologies, each having data repositories and location Names Servers storing location information wherein a client may request the location of a data entity using a network-service-name-alias (i.e., identifier) associated with the entity's host name, port, and database instance used to build a connection descriptor (i.e., location information).²⁹³

OracleNamesAdminGuide discloses the systems having data repositories and location Names Servers storing location information wherein a client may request the location of data using a network-service-name-alias (i.e., identifier) associated with the data's host name, port, and database instance used to build a connection descriptor (i.e., location information).²⁹⁴ OracleNamesAdminGuide further teaches or renders obvious resolving any location request within two

²⁹³ See OracleNamesAdminGuide at 28, 32, 36.

²⁹⁴ See *id.*

traversals by responding to each location request with either the requested location data or a redirect message to redirect a client to another location server that contains the requested location data if the initial location server lacks the location information responsive to the client's query.

To the extent OracleNamesAdminGuide doesn't disclose all elements of claims 17, 18, and 24, OracleUnleashed's and Steen's teachings render obvious partitioning the contents of one high volume location server onto multiple location servers with a flat, non-hierarchical architecture, organizing the contents according to a hash function and mapping it with a hash table, and using a redirect message to redirect a client to another location server.

1. Patent Owner Distinguished Its Claimed Invention from Hierarchical and Mixed Topology Embodiments.

In the '036 reexamination, Patent Owner distinguished the '640 claims from the prior art by differentiating between system architectures. Specifically, Patent Owner distinguished the claims' non-hierarchical structure from the prior art's hierarchical and mixed topologies, and asserted that distributed hashing can't be used in hierarchical configurations.²⁹⁵ More particularly, Patent Owner argued that, "[a]lthough the '640 specification discloses hierarchical applications for NDTP as possible embodiments, claims 17 and 18 require *non-hierarchical* structure."²⁹⁶

But it's worth noting that the '640 Patent specification teaches that its "Network Distributed Transport Protocol" (NDTP) is applicable to the architectures Patent Owner distinguished as incompatible to overcome the Oracle system prior art. When describing how the NDTP hash-

²⁹⁵ 08/04/2022 Patent Owner's Reply to Office Action in Reexamination Control No. 90/019,036 at 23, n.2 (emphasis in original); *see* '640 Pat. at 4:5-7 (stating NDTP stands for network distributed tracking protocol).

²⁹⁶ 08/04/2022 Patent Owner's Reply to Office Action in Reexamination Control No. 90/019,036 at 23, n.2 (emphasis in original); *see* '640 Pat. at 4:5-7 (stating NDTP stands for network distributed tracking protocol).

based redirection mechanisms work, the '640 Patent specification explains that the NDTP invention is “suitable for diverse topology deployments, [and] some applications might still benefit from *a specifically hierarchical server topology*.”²⁹⁷ In the prior reexamination, Patent Owner = acknowledged that the '640 Patent's similar specification includes embodiments with systems having hierarchical configurations, before arguing that the claimed invention only works with a flat, non-hierarchical architecture.²⁹⁸ Regardless, and as further detailed below, OracleNamesAdminGuide includes embodiments with systems having hierarchical configurations and other embodiments with systems having non-hierarchical configurations.

2. Routing Efficiency Benefits Hierarchical, Non-Hierarchical, And Mixed Topology Embodiments.

It's beneficial to provide the most efficient pathways for network communication in all topologies. Hierarchical structures don't limit the network from using more efficient pathways to connect its nodes.²⁹⁹ Instead, the “tree” paths in hierarchical structures provide fault-tolerance in case more efficient means fail. It'll always be possible to travel up and down the network structure until the desired data is found, but more efficient data retrieval mechanisms are possible and at times preferred.³⁰⁰ Steen describes this “worst case” fault tolerant benefit of hierarchies, while identifying the need to provide more efficient routing:

²⁹⁷ '640 Patent, 16:18-21.

²⁹⁸ 08/04/2022 Patent Owner's Reply to Office Action in Reexamination Control No. 90/019,034 at 23, n.2 (Although the '640 specification discloses hierarchical applications for NDTP as possible embodiments, claims 17 and 18 require *non-hierarchical* structure.”); *see* '978 Pat. at 4:20-21 (stating NDTP stands for network distributed tracking protocol).

²⁹⁹ Greene Decl. ¶ 207.

³⁰⁰ Steen at 105.

An implication of this [hierarchical] design is that in the worst case it is always possible to locate every object by following the chain of pointers from the root node. In practice, we can do much better than this, as described later.³⁰¹

Thus, it's possible and sometimes preferred to apply methods like hashing and redirection to any network topology, including the '640 Patent's distinguished hierarchical and mixed topology structures.³⁰² Whether additional routing protocols are applied to any given structure is a matter of design choice based on the individual needs of a given network.

Thus, like the '640 Patent specification, OracleNamesAdminGuide includes embodiments with systems having hierarchical configurations and other embodiments with systems having non-hierarchical configurations. As detailed below, OracleNamesAdminGuide's system having a non-hierarchical configuration discloses or renders obvious each limitation of independent claims 17 and 18.

During the prior reexamination of the '640 Patent, the Requester didn't rely on OracleNamesAdminGuide's non-hierarchical configuration, focusing instead on OracleNamesAdminGuide's hierarchical configuration that mirrors the '640 Patent's hierarchical configuration.³⁰³ Thus, the Office is considering OracleNamesAdminGuide's non-hierarchical configuration for the first time in the present reexamination.

OracleNamesAdminGuide teaches a system for small-scale applications that uses a non-hierarchical "flat naming model."³⁰⁴ The below illustration shows how OracleNamesAdminGuide's non-hierarchical configuration mirrors the '640 Patent's non-hierarchical configuration.

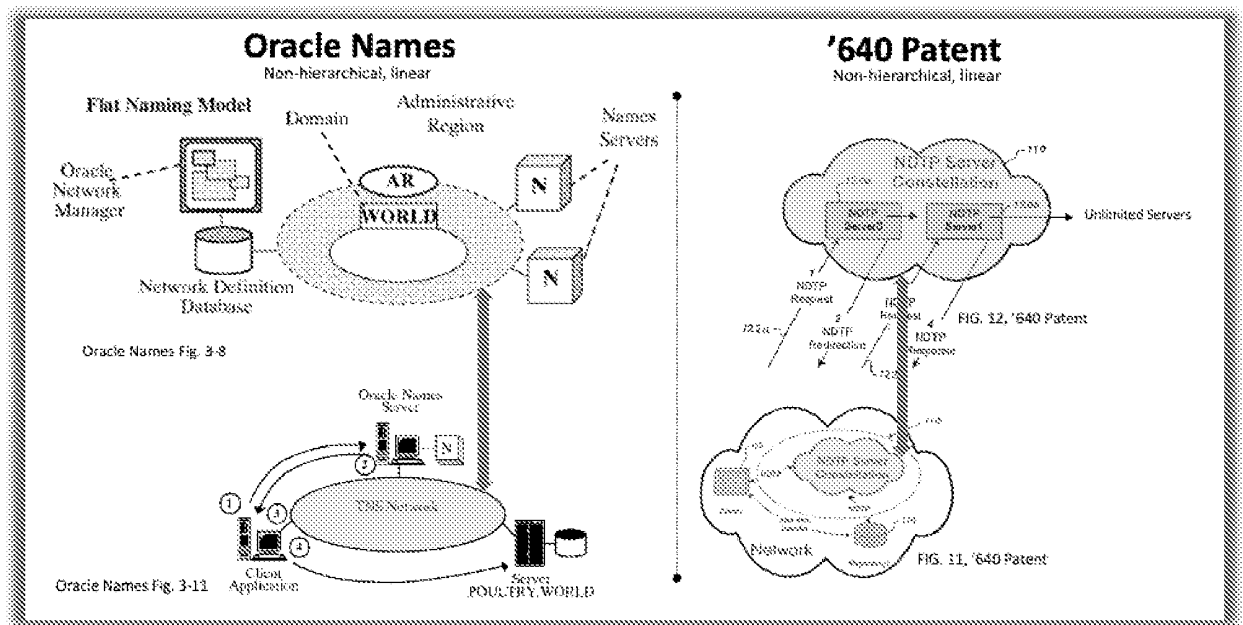
³⁰¹ Steen at 105.

³⁰² '640 Patent, 16:18-21; Green Decl. ¶ 208.

³⁰³ See, e.g., 11/19/21 Request in Reexamination Control No. 90/019,034 at 21-22, 49-50, 57, 74, 82, 92.

³⁰⁴ OracleNamesAdminGuide at 40-41.

In each of these non-hierarchical configurations, the network is flat, resulting in a client being able to send a query to any location server in the network. Each of these non-hierarchical configurations doesn't have a local region and doesn't require a location query to travel up or down a network "tree" to reach a root server.³⁰⁵



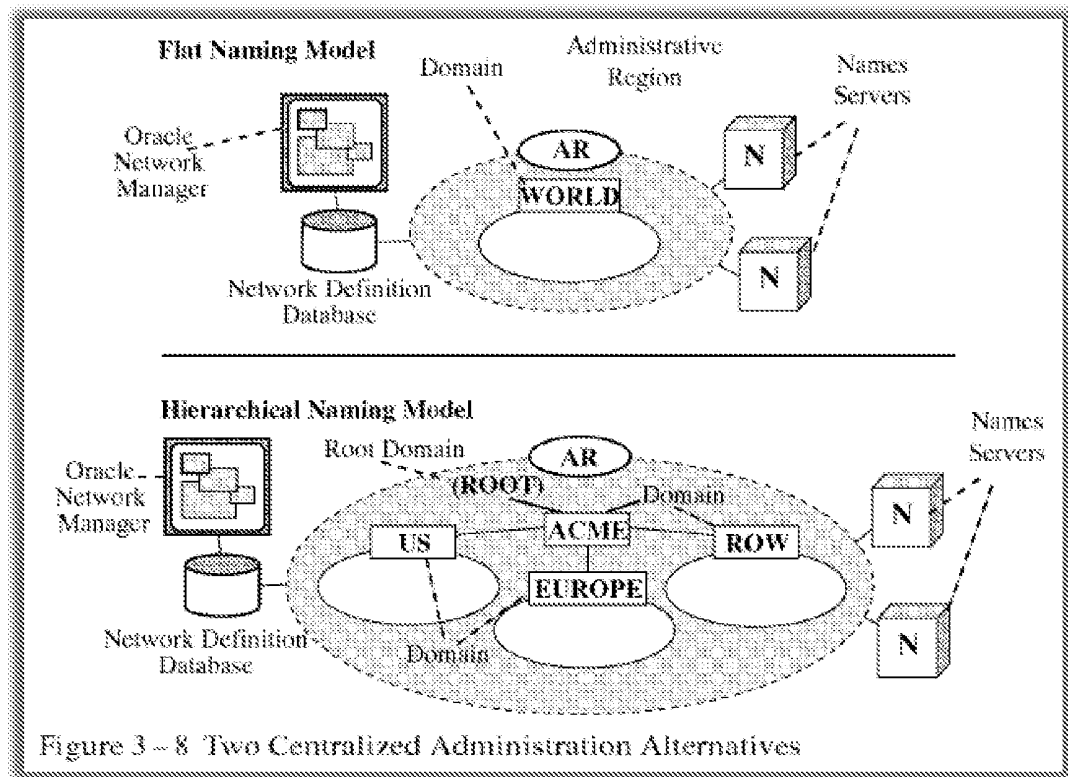
OracleNamesAdminGuide specifies that this system is non-hierarchical because “[a]ll names are unique within a single domain—the WORLD domain. The WORLD domain is predefined in the Oracle Network Manager for customers choosing a flat naming model.”³⁰⁶ And OracleNamesAdminGuide details that its flat non-hierarchical configuration is different from a hierarchical configuration for large-scale applications of more than 100 services.³⁰⁷ Below, Oracle-

³⁰⁵ Compare OracleNamesAdminGuide at 45 & 52 with 08/04/2022 Patent Owner’s Reply to Office Action in Reexamination Control No. 90/019,036 at 22-23.

³⁰⁶ OracleNamesAdminGuide at 40.

³⁰⁷ *Id.* at 40-41.

NamesAdminGuide's Figure 3-8 illustrates how a system having the "flat" non-hierarchical configuration differs from other systems having a hierarchical configuration.³⁰⁸ Mr. Greene explains that POSITAs using real-world Oracle7 systems employed OracleNamesAdminGuide's non-hierarchical configuration because such systems typically involved less than 100 services.³⁰⁹



Further, OracleNamesAdminGuide's non-hierarchical configuration meets the purported requirements for the "redirect messages" limitations. As shown above in Figure 3-8, OracleNamesAdminGuide's non-hierarchical system has an upper Names Server and a lower Names Server that cumulatively store all the location strings known in the system. When the upper

³⁰⁸ OracleNamesAdminGuide at 44-45.

³⁰⁹ Greene Decl. ¶ 212.

Names Server receives a client's query but lacks the location string to answer that query, OracleNamesAdminGuide teaches the upper Names Server transmits a message to the client where the message includes in the address of the lower Names Server housing the location string. The client uses that address to calculate the location of the lower Names Server, then the client sends its data location request to the lower Names Server.³¹⁰

As Mr. Greene explains, a POSITA would understand that a similar scenario transpires when the lower Names Server receives a client's query but lacks the location string to answer that query. This results in the lower Names Server transmitting a message to the client containing the address of the upper Names Server housing the location string. The client uses that address to calculate the location of the upper Names Server, then the client sends its data location request to the upper Names Server.³¹¹

As shown above in Figure 3-8, OracleNamesAdminGuide teaches an embodiment with a non-hierarchical system having just two Names Servers. In the embodiment with two Names Servers, a POSITA would understand that the first queried Names Server will return address information that the client uses to determine the second Names Server that contains the sought-after location string.³¹² Thus, if the first Names Server doesn't have the location string for client's query, the first Names Server will always transmit a redirect messages to the client having information to calculate the location of the second Names Server containing the location string.³¹³

³¹⁰ See OracleNamesAdminGuide at 137; Greene Decl. ¶ 213.

³¹¹ See OracleNamesAdminGuide at 137; Greene Decl. ¶ 214.

³¹² See OracleNamesAdminGuide at 44-45, 137; Greene Decl. ¶ 215.

³¹³ See Greene Decl. ¶ 215.

**3. OracleNamesAdminGuide in View of OracleUnleashed and Steen
Render Obvious Independent Claim 17.**

During the original examination, the Examiner indicated that the prior art of record didn't teach or suggest the "hash function" and "redirection" limitations of claims 17, 18, and 24.³¹⁴ OracleNamesAdminGuide in view of OracleUnleashed and/or Steen renders obvious these limitations because those references teach the capability of using a hash function to generate a hash table to organize location information across location servers.³¹⁵

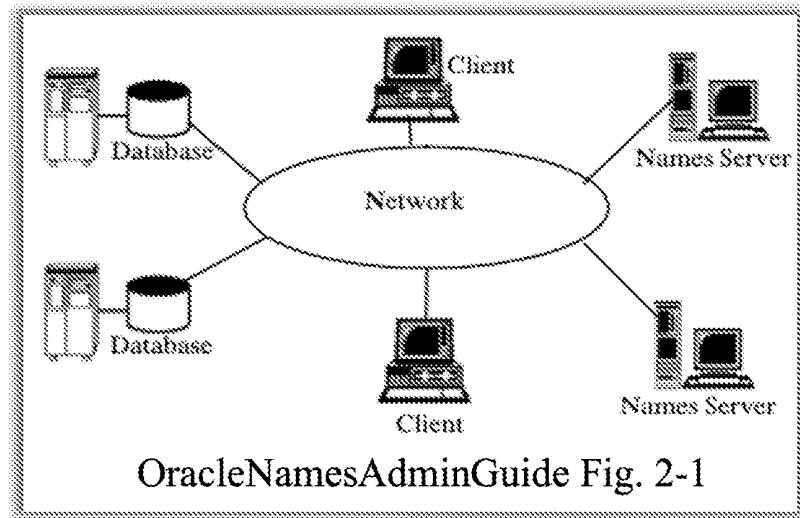
- a) *Claim element 17.pre: "A system for retrieving data location information for data stored in a distributed network, the system comprising:"*

To the extent any preamble is limiting, OracleNamesAdminGuide discloses an Oracle database system for managing data stored in a distributed network and method of using the same. As shown below in Figure 2-1, OracleNamesAdminGuide teaches an Oracle database system having multiple Oracle Names Servers for retrieving location information for data stored in a distributed network. For instance, OracleNamesAdminGuide describes an Oracle7 system that provides user access to a distributed database network. OracleNamesAdminGuide teaches the system components include SQL*Net functionalities that allow users to request data that is located in the system using Names Servers in a flat, non-hierarchical configuration.³¹⁶

³¹⁴ Ex. 9 ('640 History) at 442, 460.

³¹⁵ To establish obviousness, there is no requirement that the prior art be enabling. *See ABT Sys., LLC v. Emerson Elec. Co.*, 797 F.3d 1350, 1360 (Fed. Cir. 2015); *Symbol Tech., Inc. v. Opticon, Inc.*, 935 F.2d 1569, 1578 (Fed. Cir. 1991). But even if the prior art references did have to be enabling, both OracleNamesAdminGuide and OracleUnleashed describe Oracle7 systems which were fully functional and operational products that existed before the '640 Patent. Manuals, and specifically software manuals, have consistently been found to be enabling, even when there are differences between the manual and the related product. *See Stored Value Sols., Inc. v. Card Activation Techs., Inc.*, 499 F. App'x 5, 15 (Fed. Cir. 2012); *Microstrategy Inc. v. Bus. Objects Americas*, 410 F. Supp. 2d 348, 362-63 (D. Del. 2006), *aff'd*, 238 F. App'x 605 (Fed. Cir. 2007).

³¹⁶ *See, e.g.*, OracleNamesAdminGuide at 5, 32; Greene Decl. ¶ 217.



Further, OracleNamesAdminGuide teaches that its system receives a client's query for location information associated with a network-service-name-alias (i.e., identifier string). As detailed above in the overview of OracleNamesAdminGuide (§III.D.I.c), a POSITA would have understood that OracleNamesAdminGuide's connection descriptor is "location information" because each descriptor pertains to a location of data and identifies that location using the host name, port, and database instance:

Oracle Names product ... makes network address and database link information in an enterprise-wide distributed database network available to all nodes throughout the network.³¹⁷

[A]dress [is a] unique network location used to identify a network object such as a database service, client, Interchange, or Names Server. TNS addresses have a specific format. Addresses must be unique.³¹⁸

[C]onnect descriptor [is a] specially formatted description of the destination for a network connection. Connect descriptors are constructed using a set of keywords

³¹⁷ OracleNamesAdminGuide at 5.

³¹⁸ *Id.* at 199.

and values. They are mapped to *service names* to provide more convenient reference.³¹⁹

As OracleNamesAdminGuide explains, a database instance is a virtual collection of data where a server may store multiple database instances and the network may connect to multiple servers storing database instances. A POSITA would have understood that the database server will store multiple databases where each server connects to OracleNamesAdminGuide's network.³²⁰

Further, even if a POSITA would not have understood this from OracleNamesAdminGuide alone, it would have been obvious based on Steen's teachings. Steen explains that, in distributed systems, there can be multiple instances of the same data or service:

In distributed systems, this way of locating a service is also known as anycasting: *a client requires a particular service, but is really not interested in which server will handle the request.* Using our approach, the service is assigned a unique object handle, and each server registers its network address under that object handle. A client has the service's name resolved to the object handle, which is then subsequently resolved to the address of *any server that can handle the request.*³²¹

A POSITA would have appreciated that Steen's principle of "anycasting" allows for faster resolution of clients' requests because multiple servers can resolve any given query.³²² As a result, a POSITA would have been motivated by OracleNamesAdminGuide's goal of "lightning fast" res-

³¹⁹ *Id.* at 200 (emphasis in original); *see also id.* at 129 (BONES.DEM.MEDICINE identifier example that includes host name, port, and database instance). As Mr. Greene explains, a POSITA would have understood that a "host name" may include a direct IP address or a host name that uses DNS to obtain an IP address. Greene Decl. ¶ 218.

³²⁰ *See* Greene Decl. ¶ 219.

³²¹ Steen at 105.

³²² *See* Greene Decl. ¶ 221.

olution to incorporate anycasting to its data storage. As Mr. Greene explains, this would have resulted in multiple data repositories where each server connects to OracleNamesAdminGuide's Names Service, i.e., location server network.³²³

Based on the above teachings, OracleNamesAdminGuide discloses or renders obvious in light of Steen the preambles' system for managing data stored in a distributed network.

b) *Claim element 17.a: "a plurality of data repositories configured to store data,"*

OracleNamesAdminGuide discloses this claim element or renders obvious it alone or in view of OracleUnleashed and Steen. OracleNamesAdminGuide teaches that its system includes a plurality of data repositories in the form of databases that store data. For instance, OracleNamesAdminGuide states that its system includes multiple databases storing data: "[A] network using the Oracle Names Service consists of ... one or more Oracle Names Servers [and] one or more databases."³²⁴ And OracleNamesAdminGuide details that its system's databases store data objects: "For a given network, all of these objects are stored together in the Network Manager network definition database."³²⁵

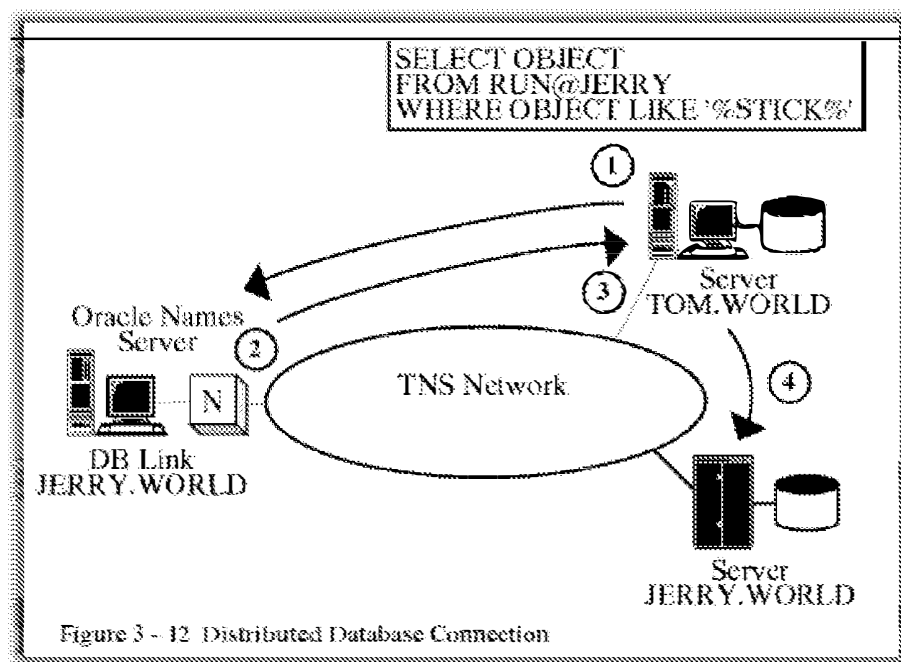
Further, OracleNamesAdminGuide explains that its system connects each database together in the network: "A global database link is a link that connects *each database in a network to all other databases*. This link is created automatically by the Network Manager for use with Oracle Names. Each global database link created by Network Manager connects you to every

³²³ See *id.*

³²⁴ OracleNamesAdminGuide at 32; see also *id.* at 28 ("Oracle Names 1.0 required administrators to enter the names and addresses of all databases in the network into Oracle Network Manager. Network Manager then created a network definition which it *stored in a database*.").

³²⁵ *Id.* at 51; *id.* at 35 ("... Oracle Names 2.0 architecture is designed to function on most SQL*Net networks composed of clients and databases.").

other database server in the network.”³²⁶ In the example shown below in Figure 3-12, OracleNamesAdminGuide describes its system connecting two data repositories (e.g., databases TOM.WORLD within one server and JERRY.WORLD within another server).³²⁷



To the extent Patent Owner asserts that OracleNamesAdminGuide doesn't expressly disclose "a plurality of data repositories configured to store data," a POSITA would have found obvious this limitation. OracleNamesAdminGuide teaches the Oracle Names location service assists in location management. OracleUnleashed and Steen both teach the use of a location service, like OracleNamesAdminGuide's Names location service, to manage data stored in multiple repositories.

Specifically, Steen provides a location service to manage stored objects, including data files. "Locating mobile objects in a worldwide system requires a scalable location service. An

³²⁶ *Id.* at 99.

³²⁷ *Id.* at 53-54.

object can be...software or [a]data object, such as a file or an electronic document.”³²⁸ As Mr. Greene explains, to the extent that Patent Owner argues that Steen doesn’t disclose a data repository, it would have been obvious to a POSITA that the data object would have to be stored somewhere. Further, as Mr. Greene describes, it would also be obvious that Steen discloses a plurality of data repositories, because if there were only one, it would be simple to manage the locations, rendering the location service unnecessary. Further, even if it didn’t require a plurality of data repositories, Steen describes its location service as “scalable,” e.g., expanding.³²⁹ Thus, even if there were only one data repository to begin with, Steen teaches adding data repositories moving forward.

OracleUnleashed also provides for using a location service, like Oracle Names, to manage data on a plurality of data repositories. The Oracle product was designed to store massive quantities of data across multiple servers and databases:

SQL*Net has a Names Server to resolve names to locations. This method makes it very easy to move Oracle servers and data-bases with no impact on the end-user community. The Net8 Name Services has the ability to cache names at the client for much faster resolution, and the various listeners and connection managers register as available with Names Services.³³⁰

A POSITA would have understood the servers and databases to be “data repositories” in OracleUnleashed’s system because they are locations to store data.³³¹ Based on the above teachings, OracleNamesAdminGuide, OracleUnleashed, and Steen disclose or render obvious claim element 17.a’s plurality of data repositories configured to store data.

³²⁸ Steen at Abstract.

³²⁹ *Id.*

³³⁰ OracleUnleashed at 220.

³³¹ See Greene Decl. ¶ 228.

- c) *Claim element 17.b: “wherein the data is associated with a respective identifier string in each data repository;”*

OracleNamesAdminGuide discloses this claim element, or renders it obvious alone or in view of OracleUnleashed. A POSITA would have understood that OracleNamesAdminGuide’s network-service-name-alias is the claimed “identifier string.” As discussed above (§III.B), Requester and Patent Owner construe a “identifier string” as “a unique encoding that identifies an individual entity, and with which zero or more location strings are associated in a location server.”³³²

(1) OracleNamesAdminGuide discloses or alone renders obvious the data being associated with a respective identifier string.

OracleNamesAdminGuide teaches that each data entity is located at a network address associated with a network-service-name-alias, which is an identifier string. OracleNamesAdminGuide uses a network-service-name-alias as an identifier string to easily refer to desired data entities on the distributed database network and to obtain connection information from the Names Server. OracleNamesAdminGuide explains that using such aliases allow customers to request data entities from databases using a simple name rather than a lengthy network location address:

Oracle Names makes network address and database link information available to all nodes throughout the network. *Each database server’s network address is stored with a name that is used to identify it. Client applications can then request a database connection with a simple name rather than a lengthy address.*³³³

³³² Ex. 17 at 6.

³³³ OracleNamesAdminGuide at 20.

[A]liases [are] so that users can connect to a database or use a database link by using a simple name instead of having to specify the global database name or global database link name (that is, the fully-qualified name).³³⁴

A POSITA would have understood OracleNamesAdminGuide's network-service-name-alias refers to the alias that the system uses on the client-side of a client-server connection.³³⁵ As Mr. Greene explains, a POSITA would have understood that Oracle literature uses "alias" in other contexts, such as server-side aliases and object aliases.³³⁶ Because those other types of aliases are different and distinct from aliases used on the client-side of an Oracle7 system, the Request and Mr. Greene refer to client-side identifiers as network-service-name-aliases.³³⁷

Further, OracleNamesAdminGuide explains that each data entity is associated with a respective identifier string in each data repository (e.g., a database server). For instance, OracleNamesAdminGuide specifies each data entity is associated with a network-service-name-alias, connect string, and database service name:

An Oracle Names Server stores names and addresses for network services such as databases or MultiProtocol Interchanges, database link definitions, and object aliases.

The Names Server stores the following types of objects, which are described below:

- *database service names*—*The service name is the global database name that is mapped to the SQL*Net connect descriptor.* The combination of a listener address and connect data make up a connect descriptor that locates a database service. Gateways to non-Oracle databases and Oracle RDB databases are also stored.

³³⁴ *Id.* at 101.

³³⁵ *See* Greene Decl. ¶ 231.

³³⁶ *See id.*

³³⁷ *See id.*

- database links—Global database links are automatically created to each defined database from all database servers in the network. These links are made available to all users.
- *aliases*—*You can assign service name aliases as alternate names for any defined database service or database link. Service name aliases are resolved as if they were the original object.*
- V1 connect strings—*A V1 connect string can be mapped to a service name in the Oracle Names database. The service name you enter acts as an alias for the SQL*Net V1 connect string, making it easy to access a server in a network that still uses SQL*Net V1.*³³⁸

Thus, a POSITA would have understood that each network-service-name-alias is the claimed “identifier string” because it is a unique encoding that identifies an entity and that is associated with a connection descriptor (i.e., location information) in a Names Server.³³⁹

Further, a POSITA would have understood that each identifier string in OracleNamesAdminGuide’s system is associated with a data entity, and not simply a data repository (e.g., a database server). As discussed above, each OracleNamesAdminGuide’s database server is a physical device that houses multiple virtual database instances. In Reexamination Control No. 90/019,034 toward the related ’978 Patent, the Examiner noted that Patent Owner’s assertion that OracleNamesAdminGuide “does not teach an identifier that uniquely identifies a single entity is not factual.”³⁴⁰ The Examiner stated that, in its flat, non-hierarchical configuration, OracleNamesAdminGuide’s system has one domain that contains all identifier strings (e.g., names) where each

³³⁸ See OracleNamesAdminGuide at 23.

³³⁹ See Greene Decl. ¶ 233.

³⁴⁰ 10/27/2022 Final Office Action in Reexamination Control No. 90/019,034 at 21.

string uniquely identifies a single entity (e.g., object). In support, the Examiner included OracleNamesAdminGuide's Figure 3-5 and its quote: "All names in the WORLD domain must be unique. The Network Manager has a validation feature that ensures that all names you enter are unique."³⁴¹

To the extent Patent Owner asserts that OracleNamesAdminGuide doesn't expressly disclose "the data is associated with a respective identifier string in each data repository," a POSITA would have found obvious this limitation. OracleNamesAdminGuide teaches using identifier strings to identify location on a network. A POSITA would have appreciated that the network would include data repositories. And a POSITA would have understood that all data entities stored on the data repositories are associated with respective identifier strings, and that the identifier strings are all stored collectively in the location server network.³⁴²

(2) OracleNamesAdminGuide and OracleUnleashed render obvious the data being associated with a respective identifier string.

Further, OracleNamesAdminGuide and OracleUnleashed render obvious this claim element. Based on OracleUnleashed's teachings, a POSITA would have been motivated and found it obvious to have OracleNamesAdminGuide's system associate each data repository's data with a respective identifier string. OracleUnleashed explains that SQL*Net, which OracleNamesAdminGuide utilizes, makes it easier for users to connect to various databases and find their data, which is advantageous because "[y]our users do not want to write complex queries."³⁴³ A POSITA would have understood that associating respective identifier strings with data improves

³⁴¹ *Id.* at 17 (citing and quoting OracleNamesAdminGuide at 40-41).

³⁴² *See* Green Decl. ¶ 235.

³⁴³ *See* OracleUnleashed at 255.

the users' experience and to ease the ability to find desired data since that is one of the objectives of an Oracle DBA.³⁴⁴ The benefits of associating data with a respective identifier string are directly perceived by the end user, and a DBA would have been motivated to simplify the users' experience in this manner.³⁴⁵

Thus, as Mr. Greene explains, a POSITA would have found it obvious to have each of OracleNamesAdminGuide's data repositories' data be associated with a respective identifier string for use in optimizing query efficiency, based on OracleUnleashed's teachings.³⁴⁶ During the relevant timeframe, Oracle DBAs were often asked to help improve query efficiency, as OracleUnleashed details:

It seems pretty obvious that a search method that works well for one of these types of queries is probably not the best method for the others. In fact, Oracle and most other databases have a number of search algorithms in their systems that are designed to handle different needs. This is what enables them to work in such a large number of environments and applications. When an application is performing slowly on a well-tuned database, the first thing you should *check is how the query is being processed to ensure that it is as efficient as it can be.*³⁴⁷

Based on the commonplace need for efficiency in running queries, a POSITA would have been motivated to associate a respective identifier string with data because this allows OracleNamesAdminGuide's Names Server to quickly return one or more locations associated with the identifier string to the query requestor with minimized cost.³⁴⁸

³⁴⁴ Greene Decl. ¶ 236.

³⁴⁵ *Id.*

³⁴⁶ See OracleUnleashed at 452; Greene Decl. ¶ 237.

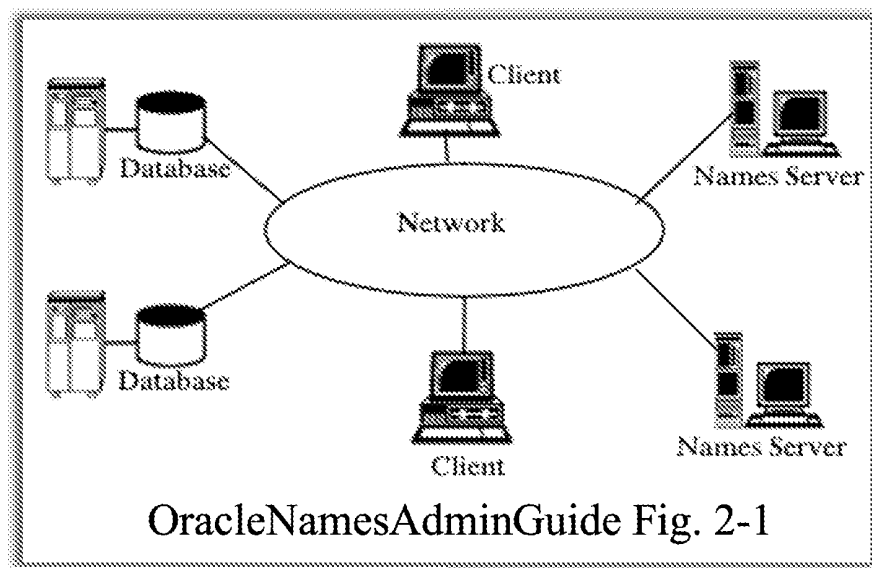
³⁴⁷ OracleUnleashed at 652; Greene Decl. ¶ 237.

³⁴⁸ See OracleUnleashed at 452.

Based on the above teachings, OracleNamesAdminGuide and OracleUnleashed disclose or render obvious claim element 17.b's identifier string identifier string that identifies that data entity.

d) *Claim element 17.c: "a data location se[rv]er network having a plurality of data location servers,"*

OracleNamesAdminGuide discloses this claim element or renders it obvious alone or in view of OracleUnleashed and/or Steen. As shown below in Figure 2-1, OracleNamesAdminGuide explains that its system includes a network of Names Servers.³⁴⁹ A POSITA would have understood that these Names Servers are the claimed "location servers." As discussed above (§III.B), Patent Owner has asserted a "location server" is "a network-attached component that maintains a set of identifier/location mappings that are modified or returned in response to location request messages from clients."



³⁴⁹ Joint Claim Construction Chart at 4.

As explained above in the overview of OracleNamesAdminGuide (§III.D.1.a,) a POSITA would have understood that OracleNamesAdminGuide's Names Server is a "location server" because it is a network-attached server that maintains data location mappings to service client requests. For instance, OracleNamesAdminGuide states that each Names Server attaches to the network and maps data locations to objects that clients may request:

Oracle Names makes network address and database link information available to all nodes throughout the network. *Each database server's network address is stored with a name that is used to identify it. Client applications can then request a database connection with a simple name rather than a lengthy address.*³⁵⁰

The Names Server has a single purpose: to resolve, or assist in resolving, a client-initiated name request. It interprets the request, then looks up the name in its cache, or it calls a Names Server in another region. The response or error conditions are passed back to the client.³⁵¹

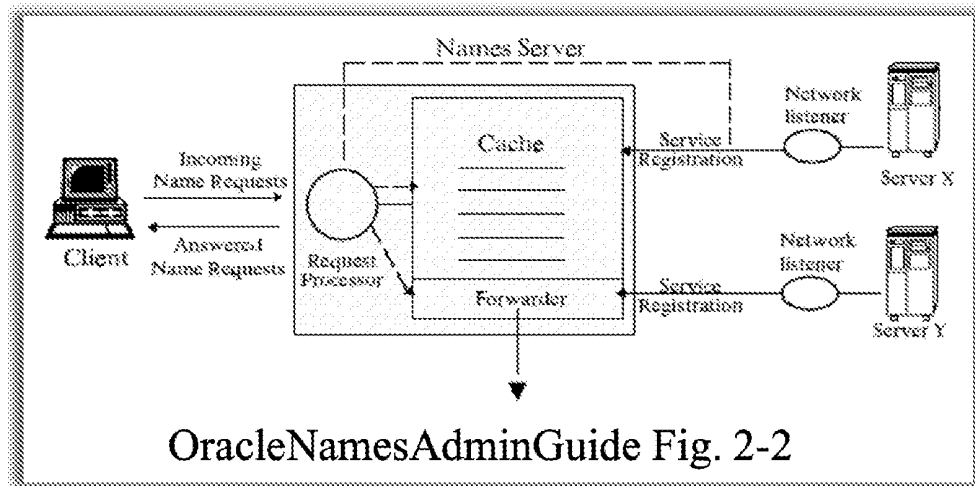
And as shown below in Figure 2-2, each Names Server connects to a client and includes a processor and a cache memory for resolving location queries from clients.³⁵² OracleNamesAdminGuide explains that the "user and application developer are exposed only to the name of a network object. No knowledge of the connection path is required.... When making a connection, the client specifies the name of a destination and Oracle Names provides the address."³⁵³

³⁵⁰ OracleNamesAdminGuide at 20.

³⁵¹ *Id.* at 5, 32 ("We recommend that you set up at least two Names Servers for fault tolerance."), 36 (explaining that each region has "one or more Oracle Names Servers").

³⁵² *Id.* at 32-33.

³⁵³ *Id.* at 22.



Further, OracleNamesAdminGuide discloses that its system has at least two Names Servers on the network for fault tolerance and load balancing: "Oracle recommends that each administrative region should have at least two Names Servers. If one Names Server is unavailable, clients will automatically forward requests to the other.... There is also an issue of network load- ing, or load balancing. Although any given Names Server is only engaged for a short period of time when a client initiates a connection, the goal is to keep name resolutions lightning fast.... [L]oad balancing across multiple Names Servers can increase system performance."³⁵⁴

Regarding the claimed "processor" and "programming logic," OracleNamesAdminGuide explains that each Names Server includes a processor and memory.³⁵⁵ As Mr. Greene explains, a POSITA would have understood that, for this processor to function, the memory contains pro- gramming logic to execute steps.³⁵⁶

³⁵⁴ *Id.* at 60.

³⁵⁵ *Id.* at 32-33.

³⁵⁶ *See* Greene Decl. ¶ 265.

Further, OracleNamesAdminGuide teaches that each Names Server loads and executes software to preform steps in response to a data location request: “At startup, the Names Server reads its configuration files to set up its operating parameters, then loads all data for the administrative region.”³⁵⁷ Mr. Greene explains that a POSITA would have understood that each Names Server includes programming logic to read configuration files and set up operating parameters.³⁵⁸

And OracleNamesAdminGuide discloses that each Names Servers is a resource running as software on a computer and is executable on operating systems, such as UNIX, Windows, and OS/2:

An Oracle Names Server is portable software that runs on a variety of computer platforms.... In general, Oracle Names runs on:

- A variety of UNIX and VAX/VMS minicomputers and workstations.
- Intel-based computers (PCs) running operating systems such as OS/2, NetWare, Windows NT, and UNIX.

The computer does not have to be dedicated as an Oracle Names Server. In fact, Oracle Names typically requires relatively little CPU and memory to function.

Your choice of computer to use as a Names Server should be based on ... [s]oftware availability—While Oracle Names runs on many operating systems, it does not run on all. *As a shared resource, Oracle Names only needs to be on one type of operating system to serve all clients and servers.* It does not have to be available on all of the operating systems you use....”³⁵⁹

³⁵⁷ See OracleNamesAdminGuide at 160; *id.* at 104 (“The STARTUP command loads the Names Server into memory and tells it to begin executing. At startup, the Names Server loads its configuration, loads its data, then becomes available to answer requests.... The SHUTDOWN command stops the execution of a Names Server. The program stops executing, and releases any machine resources being used.”).

³⁵⁸ See Greene Decl. ¶ 266.

³⁵⁹ OracleNamesAdminGuide at 62-63.

As Mr. Greene explains, a POSITA would have understood that the computer acting as the Names Server contains programming logic to execute steps.³⁶⁰

OracleNamesAdminGuide provides an example command NAMESCTL for each Names Server that results in operations executing on the computer: “NAMESCTL is loaded and *all operations are executed within it*. . . . You can also execute most commands from the operating system command line by running the NAMESCTL program with a complete command as a parameter to the program. In this case, NAMESCTL *will load and execute the command*, then return the operating system prompt.”³⁶¹

Further, OracleUnleashed and/or Steen teaches it would have been obvious to have OracleNamesAdminGuide’s system include a plurality of location servers. For instance, as illustrated below in Figure 19.4, OracleUnleashed describes the benefits of partitioning a dataset into “smaller, easier-to-manage pieces”:

A new feature of Oracle8 is that of partitioning. Partitioning enhances Oracle’s support for very large databases as *partitioning breaks tables and indexes into smaller, easier-to-manage pieces*, storing these pieces at the tablespace level. Partitioned tables/indexes are nothing more than a logical collection of tablespaces (see Figure 19.4).

Partitioning greatly enhances database performance. This performance can be monitored and adjusted in a variety of manors. Read/write activity can be load-balanced among the disk drives/disk arrays. The Oracle optimizer recognizes partitions and can select from partitions that contain the data being selected. The Oracle Parallel Server can divide work based on the number of partitions.

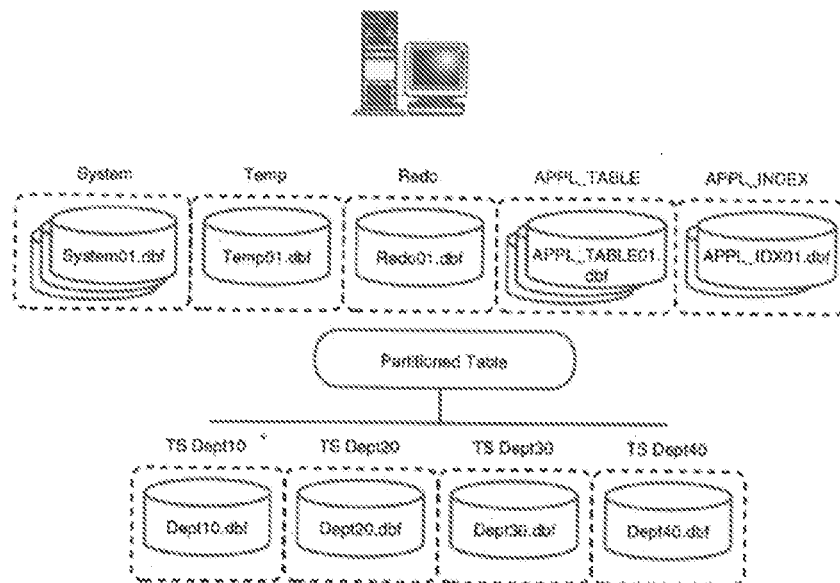
³⁶⁰ See Greene Decl. ¶ 268.

³⁶¹ OracleNamesAdminGuide at 116.

Availability is higher, backups are easier, and recovery is shorter. High availability is the single goal of most computer systems. Recovery and time to recover are always an issue. Downtime for maintenance or backups is not possible with some applications. Machine failures are inevitable. Partitions can be backed up online, individually, and only the ones that have changes. This makes the backup/recovery process much quicker. If recovery is needed, the other partitions not affected are still available online.

Administration and tuning is much easier. *Oracle8 partitioning has many features that make partitions easy to create, move, split, and so on.* One can even swap in an existing non-partitioned table structure and data to an existing table partition. Tuning is enhanced with additional indexing features available.³⁶²

FIGURE 19.4.
Oracle tablespace layout example.



³⁶² OracleUnleashed at 424-25.

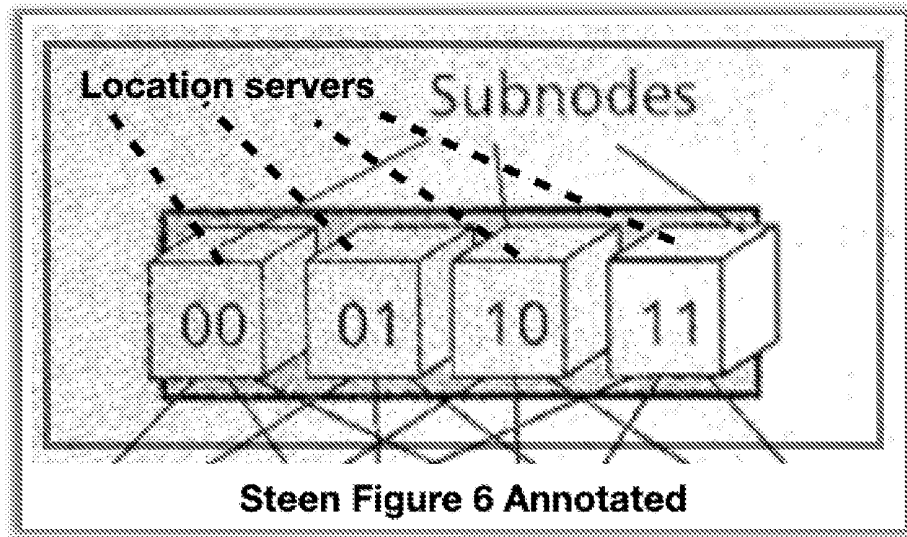
As Mr. Greene explains, a POSITA would have been motivated by OracleUnleashed's descriptions of the performance benefits of partitioning to apply its teachings to any of OracleNamesAdminGuide's underperforming Names Servers.³⁶³ A POSITA would have had a reasonable expectation of success because both OracleUnleashed and OracleNamesAdminGuide describe the same product lines and are intended to be used in conjunction with each other.

Further, Steen describes a system for managing the location of data on a distributed network. As Steen explains, a single server providing location information can be overwhelmed with too much data or too many requests: "higher-level directory nodes not only have to handle a relatively large number of requests, they also have enormous storage demands."³⁶⁴ To avoid this potential strain on a system, Steen teaches partitioning a directory server with location information into a flat, non-hierarchical group of "subnode" location servers: "Our solution is to partition a directory node into one or more directory subnodes such that each subnode is responsible for a subset of the records originally stored at the directory node."³⁶⁵ Below, a portion of Steen's Figure 6 includes annotations illustrating this non-hierarchical group of locations servers.

³⁶³ See Greene Decl. ¶ 245.

³⁶⁴ Steen at 107-08.

³⁶⁵ *Id.*



As Mr. Greene explains, a POSITA would have understood that Steen’s partitioned group of location servers organized in a non-hierarchical configuration results in each server containing a subset of the data location information originally stored on the first location server. Further, a POSITA would have been motivated to adopt Steen’s partitioning approach for grouping OracleNamesAdminGuide’s Names Servers to maintain “lightning fast” resolutions to data queries.³⁶⁶ Thus, applying either (1) OracleUnleashed’s teachings toward partitioning servers and datasets or (2) Steen’s teachings toward partitioning a location server into a group of servers to OracleNamesAdminGuide’s Names Servers would have rendered obvious that the data location server network has a plurality of data location servers.

Based on the above teachings, OracleNamesAdminGuide, OracleUnleashed, and Steen disclose or render obvious claim element 17.c’s data location server network comprising a plurality of data location servers.

³⁶⁶ See Greene Decl. ¶ 247.

- e) *Claim element 17.d: “each of the plurality of data location servers containing location strings associated with respective identifier strings and”*

OracleNamesAdminGuide discloses this claim element or renders it obvious alone or in view of OracleUnleashed. A POSITA would have understood that OracleNamesAdminGuide’s connection descriptors are the claimed “location strings.”³⁶⁷ Patent Owner has asserted that the claimed “location strings” refers to a “location.”³⁶⁸ As discussed above (§III.B), Requester and Patent Owner construe “location” as “an encoding that is a member of a set of associations with an identifier in a location server, and that specifies where data pertaining to the entity identified by the identifier is stored.”³⁶⁹

(1) OracleNamesAdminGuide discloses or renders obvious location strings associated with respective identifier strings.

OracleNamesAdminGuide discloses that each Names Server, i.e., location server, contains connection descriptors, i.e., location strings, associative with respective network-service-name-aliases, i.e., identifier strings. For instance, OracleNamesAdminGuide explains that its SQL*Net uses a data entity’s location information to build a “connection descriptor,” also called a connect string or dbstring.³⁷⁰ OracleNamesAdminGuide teaches that its connection descriptor is a “specially formatted description of the destination for a network connection. Connect descriptors are constructed using a set of keywords and values. They are mapped to *service names*

³⁶⁷ See *id.* ¶ 250.

³⁶⁸ See 08/04/2022 Patent Owner’s Reply to Office Action in Reexamination Control No. 90/019,036 at 18, 35 (Patent Owner stating: “‘Location strings’ are also known as ‘location’ or ‘data location,’ which are construed as specifying where ‘data pertaining to the entity’ is stored.”).

³⁶⁹ Joint Claim Construction Chart at 6.

³⁷⁰ OracleNamesAdminGuide at 23, 100.

to provide more convenient reference.”³⁷¹ And OracleNamesAdminGuide explains that each “unique network location [is] used to identify a network object such as a database service, client, Interchange, or Names Server. TNS addresses have a specific format. Addresses must be unique.”³⁷²

As explained above in the overview of OracleNamesAdminGuide (§III.D.1.c), a POSITA would have understood that OracleNamesAdminGuide’s connection descriptors are “location strings” because each descriptor pertains to a location of data and identifies that location using the host name, port, and database instance.³⁷³ For instance, OracleNamesAdminGuide specifies that its system uses SQL*Net to map each connection descriptor to the location where each data entity is stored:

The Names Server stores the following types of objects ...

- database service names—*The service name is the global database name that is mapped to the SQL*Net connect descriptor.* The combination of a listener address and connect data make up a connect descriptor that locates a database service. Gateways to non-Oracle databases and Oracle RDB databases are also stored....
- aliases—*You can assign service name aliases as alternate names for any defined database service or database link.* Service name aliases are resolved as if they were the original object.
- V1 connect strings—*A V1 connect string can be mapped to a service name in the Oracle Names database.* The service name you enter acts as an alias for the

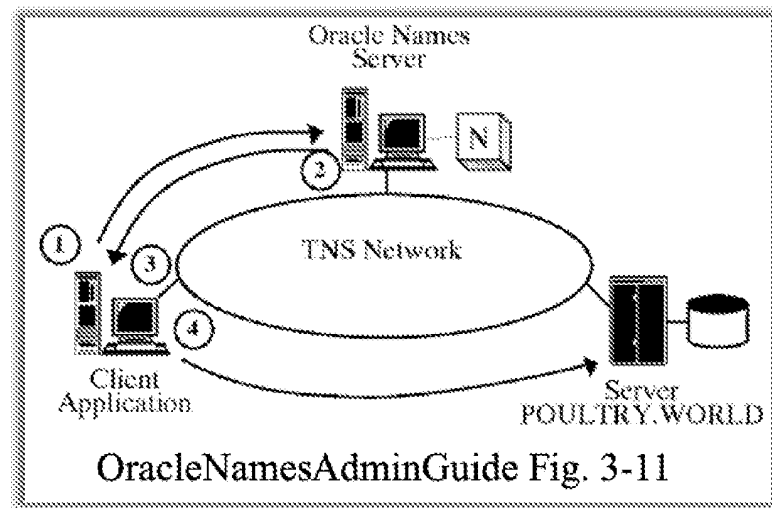
³⁷¹ *Id.* at 200; *see also id.* at 202 (describing that a connect descriptor may include a system identifier (“SID”), which is a “unique name for an Oracle database instance.... The SID is included in the CONNECT DATA parts of the connect descriptors in a TNSNAMES.ORA file, and in the definition of the network listener in the LISTENER.ORA file”).

³⁷² *Id.* at 199; Greene Decl. ¶ 251.

³⁷³ *See* Greene Decl. ¶ 252.

SQL*Net V1 connect string, making it easy to access a server in a network that still uses SQL*Net V1.³⁷⁴

Below, OracleNamesAdminGuide's Figure 3-11 provides an example of how each Names Server (i.e., location server) includes connection descriptors (i.e., location strings) associated with respective network-service-name-aliases (i.e., identifier strings). At step 1, the client sends a data location request for POULTRY to the Names Server. Based on this request, at step 2, the Names Server identifies the connection descriptor (e.g., host name, port, and database instance) for the POULTRY.WORLD database, and sends this connection descriptor to the client. And at steps 3-4, the client receives connection descriptor from the Names Server then uses that connection descriptor to connect to the POULTRY.WORLD database.³⁷⁵



Similarly, OracleNamesAdminGuide provides another example of how each Names Server (i.e., location server) includes connection descriptors (i.e., location strings) associated

³⁷⁴ OracleNamesAdminGuide at 23-24.

³⁷⁵ See *id.* at 52-53.

with respective network-service-name-aliases (i.e., identifier strings). Here, the user of client device issues the command “QUERY BONES.DEM.MEDICINE A.SMD.”³⁷⁶ Like the above POULTRY example, the client sends a request for the BONES.DEM.MEDICINE data to the Names Server. The Names Server uses the identifier BONES.DEM.MEDICINE to identify the connection descriptor, and sends that connection descriptor to the client. OracleNamesAdminGuide discloses that the connection descriptor includes the host name “cowboy,” port 1522, and database instance “rodeo” along with “protocol=TCP.”³⁷⁷ The client uses the connection descriptor to connect to the BONES.DEM.MEDICINE database.³⁷⁸

To the extent Patent Owner asserts that OracleNamesAdminGuide doesn’t expressly disclose “each of the plurality of data location servers containing location strings associated with respective identifier strings,” a POSITA would have found this limitation obvious. For instance, a POSITA would have found it obvious to associate OracleNamesAdminGuide’s location strings (e.g., connection descriptor) with respective identifier strings (e.g., network-service-name-alias) and to respond to queries with location messages for use in optimizing query efficiency.³⁷⁹

(2) OracleNamesAdminGuide and OracleUnleashed render obvious locating strings associated with respective identifier strings.

Further, OracleNamesAdminGuide and OracleUnleashed render obvious this claim element. Before the ‘640 Patent, it was well known and ubiquitous for computer network systems to

³⁷⁶ See OracleNamesAdminGuide at 128-29.

³⁷⁷ See *id.* at 129; see *id.* at 39, 201 (describing other available transport protocols, like DECnet). As Mr. Greene explains, the host name would be converted to an IP address using a DNS server before the connection string was compiled. See Greene Decl. at ¶ 254 n.237.

³⁷⁸ See Greene Decl. ¶ 254.

³⁷⁹ See *id.* ¶ 255.

associate location information with identifiers. With respect to Oracle systems like those in OracleNamesAdminGuide, OracleUnleashed teaches that SQL *Net associates the data's network address to an "easy name" alternate:

A synonym is merely a way of assigning an alias with which users can reference a table or view. This makes it easier to access tables that you do not own. When you own a table, you use its name in your SQL queries. When you want to access a table owners by an-other user, you need to specify both the owner of the table and the table name (in the owner.table_name format). When you reference tables located in a different database using SQL*Net, the format can be rather complicated. Synonyms simplify this access because you can create a simple synonym that contains the details of a complex reference string.³⁸⁰

Based on OracleUnleashed's teachings, a POSITA would have found it obvious for OracleNamesAdminGuide to associate location strings (i.e., connection descriptor) with respective identifier strings (i.e., network-service-name-alias). For instance, OracleUnleashed explains that OracleNamesAdminGuide's SQL*Net uses executable code to make it easier for users to connect to various databases and find their data:

SQL*Net has a Names Server to resolve names to locations. This method makes it very easy to move Oracle Servers and databases with no impact on the end-user community. The Net8 Name Services has the ability to cache names at the client for much faster resolution, and the various listeners and connection managers register as available with Names Services.³⁸¹

³⁸⁰ OracleUnleashed at 228.

³⁸¹ *Id.* at 424.

Indeed, before the '640 Patent, POSITAs were combining these teachings in Oracle systems in the 1990s.³⁸² And OracleUnleashed explains that it should be studied in conjunction with Oracle manuals, like OracleNamesAdminGuide:

Instead, the constant reader would first study this book to gain an understanding of the many issues surrounding this Herculean task, using the expert knowledge and examples as a template when referring to specific documentation regarding the many tasks that they faced. Once the large picture has been grasped, the reader might then dive into specific Oracle manuals in order to extract useful details needed to complete the task.

Using [this book] in this way, the reader will save a great deal of time. Instead of getting lost in endless reams of manuals, by first referring to this book, one can focus on the issues and hotpoints surrounding the complex tasks faced in purchasing, installing, and implementing the appropriate Oracle technology.³⁸³

Further, OracleUnleashed specifically directs DBAs to combine its teachings with reference manuals published by Oracle Corporation—like OracleNamesAdminGuide—to fully utilize Oracle database systems and all their capabilities.³⁸⁴ Further, beyond the traditional Oracle reference manuals, as Mr. Greene explains, DBAs would have been expected to combine their systems with various technologies and tools as they develop to maximize functionality and effectiveness.³⁸⁵ As OracleUnleashed explains: “Technology surpasses itself each year, especially in the world of Oracle and relational database systems. New tools are constantly made available to the DBA to aid in overall job effectiveness.”³⁸⁶

³⁸² See Greene Decl. ¶ 258.

³⁸³ See OracleUnleashed at 30.

³⁸⁴ See *id.* at 235, 527, 558, 565, 591.

³⁸⁵ See Greene Decl. ¶ 259.

³⁸⁶ OracleUnleashed at 476.

Maintaining a working knowledge of the state of the art would be imperative for managing a distributed system, especially an Oracle system, because a POSITA would have been expected to combine the system with related technologies to improve efficiency and optimization.³⁸⁷ OracleUnleashed outlines these motivations, stating: “However, all these responsibilities revolve around the common goals of protecting the data, providing desirable functionality, and optimizing performance as much as possible.”³⁸⁸

For those reasons, a POSITA would have been motivated to associate identifier strings with OracleNamesAdminGuide’s data and location information for use in optimizing query efficiency, based on OracleUnleashed’s teachings. As described in OracleUnleashed, SQL*Net makes it easier for users to connect to various databases and find their data, which is advantageous because “[y]our users do not want to write complex queries.”³⁸⁹ A POSITA would have understood that making use of location strings associated with identifier strings improves the users’ experience and eases the ability to find desired data since that is one of the objectives of an Oracle DBA.³⁹⁰ The benefits of associating a location string with an identifier string are directly perceived by the end user, and a DBA would have been motivated to simplify the users’ experience in this manner.³⁹¹

It was understood in the art that a POSITA was often asked to help improve query efficiency. During the relevant timeframe, Oracle DBAs were often asked to help improve query efficiency, as OracleUnleashed details:

³⁸⁷ See Greene Decl. ¶ 260.

³⁸⁸ OracleUnleashed at 510.

³⁸⁹ *Id.* at 255.

³⁹⁰ See Greene Decl. ¶ 261.

³⁹¹ See *id.*

It seems pretty obvious that a search method that works well for one of these types of queries is probably not the best method for the others. In fact, Oracle and most other databases have a number of search algorithms in their systems that are designed to handle different needs. This is what enables them to work in such a large number of environments and applications. When an application is performing slowly on a well-tuned database, the first thing you should ***check is how the query is being processed to ensure that it is as efficient as it can be.***³⁹²

Based on the commonplace need for efficiency in running queries, a POSITA would have been motivated to associate location strings with respective identifier strings because this allows OracleNamesAdminGuide's Names Server to quickly return one or more locations associated with the identifier string to the query requestor with minimized cost.³⁹³ Further, this would have resulted in the effective use of SQL*Net functionality, and improve query efficiency and optimization of Oracle systems to ease users' ability to find their desired data. Based on the commonplace need for efficiency in running queries, a POSITA would have been motivated to associate the identifier string with data and location information because this allows OracleNamesAdminGuide's Names Server to contain location strings associated with respective identifier strings to service data requests with minimized cost.³⁹⁴

Based on the above teachings, OracleNamesAdminGuide and OracleUnleashed disclose or render obvious claim element 17.d's at least one of the plurality of data location servers includes location information associated with the identifier string.

f) *Claim element 17.e: "each of the plurality of data location servers having computer executable code configured to execute the following steps:"*

³⁹² OracleUnleashed at 652; Greene Decl. ¶ 262.

³⁹³ See OracleUnleashed at 452.

³⁹⁴ See Greene Decl. ¶ 263.

OracleNamesAdminGuide discloses this claim element or renders it obvious alone or in view of Steen. OracleNamesAdminGuide explains that each Names Server includes a processor and memory.³⁹⁵ As Mr. Greene explains, a POSITA would have understood that, for this processor to function, the memory contains computer executable code to execute steps.³⁹⁶

Further, OracleNamesAdminGuide teaches that each Names Server loads and executes software to preform steps in response to a data location request: “At startup, the Names Server reads its configuration files to set up its operating parameters, then loads all data for the administrative region.”³⁹⁷ Mr. Greene explains that a POSITA would have understood that each Names Server includes executable code to read configuration files and set up operating parameters.³⁹⁸

And OracleNamesAdminGuide discloses that each Names Servers is a resource running as software on a computer and is executable on operating systems, such as UNIX, Windows, and OS/2:

An Oracle Names Server is portable software that runs on a variety of computer platforms.... In general, Oracle Names runs on:

- A variety of UNIX and VAX/VMS minicomputers and workstations.
- Intel-based computers (PCs) running operating systems such as OS/2, NetWare, Windows NT, and UNIX.

The computer does not have to be dedicated as an Oracle Names Server. In fact, Oracle Names typically requires relatively little CPU and memory to function.

³⁹⁵ OracleNamesAdminGuide at 32-33.

³⁹⁶ See Greene Decl. ¶ 265.

³⁹⁷ See OracleNamesAdminGuide at 160; *id.* at 104 (“The STARTUP command loads the Names Server into memory and tells it to begin executing. At startup, the Names Server loads its configuration, loads its data, then becomes available to answer requests.... The SHUTDOWN command stops the execution of a Names Server. The program stops executing, and releases any machine resources being used.”).

³⁹⁸ See Greene Decl. ¶ 266.

Your choice of computer to use as a Names Server should be based on ... [s]oftware availability—While Oracle Names runs on many operating systems, it does not run on all. *As a shared resource, Oracle Names only needs to be on one type of operating system to serve all clients and servers.* It does not have to be available on all of the operating systems you use....”³⁹⁹

As Mr. Greene explains, a POSITA would have understood that the computer acting as the Names Server contains computer executable code to execute steps.⁴⁰⁰

OracleNamesAdminGuide provides an example command NAMESCTL for each Names Server that results in operations executing on the computer: “NAMESCTL is loaded and *all operations are executed within it*.... You can also execute most commands from the operating system command line by running the NAMESCTL program with a complete command as a parameter to the program. In this case, NAMESCTL *will load and execute the command*, then return the operating system prompt.”⁴⁰¹

To the extent Patent Owner asserts that OracleNamesAdminGuide doesn’t expressly disclose “each one of the plurality of data location servers comprises a processor and a portion of the data location information” steps, a POSITA would have found this limitation obvious based on Steen’s teachings. Steen describes a system for managing the location of data on a distributed network. As Steen explains, a single server providing location information can be overwhelmed with too much data or too many requests: “higher-level directory nodes not only have to handle a relatively large number of requests, they also have enormous storage demands.”⁴⁰² To avoid this

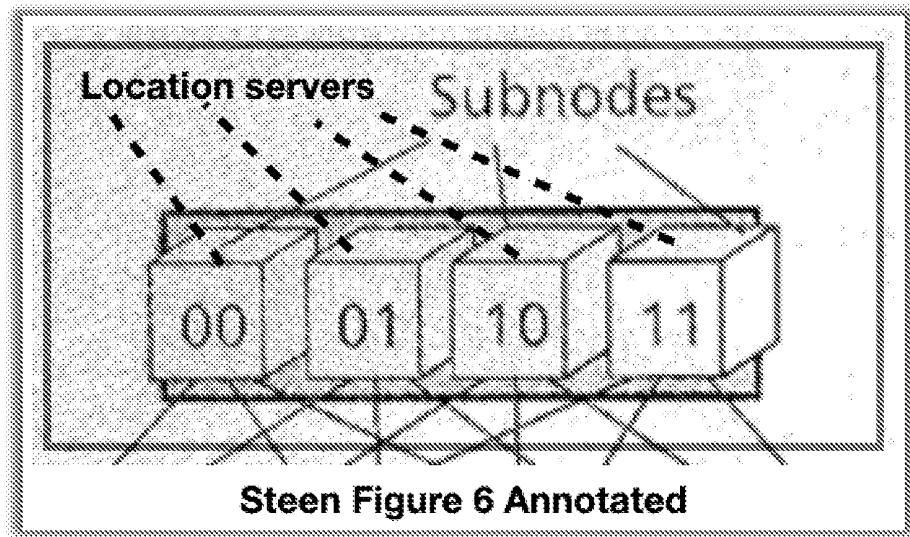
³⁹⁹ OracleNamesAdminGuide at 62-63.

⁴⁰⁰ See Greene Decl. ¶ 268.

⁴⁰¹ OracleNamesAdminGuide at 116.

⁴⁰² Steen at 107-08.

potential strain on a system, Steen teaches partitioning a directory server functioning as a location server into a flat, non-hierarchical group of “subnode” location servers: “Our solution is to partition a directory node into one or more directory subnodes such that each subnode is responsible for a subset of the records originally stored at the directory node.”⁴⁰³ Below, a portion of Steen’s Figure 6 includes annotations illustrating this non-hierarchical group of locations servers.



As Mr. Greene explains, a POSITA would have understood that Steen’s partitioned group of location servers organized in a non-hierarchical configuration results in each server containing a subset of the data location information originally stored on the first location server. Further, a POSITA would have been motivated to adopt Steen’s partitioning approach for grouping Oracle-NamesAdminGuide’s Names Servers to maintain “lightning fast” resolutions to data queries.⁴⁰⁴

⁴⁰³ *Id.*

⁴⁰⁴ See Greene Decl. ¶ 271.

Based on the above teachings, OracleNamesAdminGuide and Steen disclose or render obvious claim element 17.e's each of the plurality of data location servers having computer executable code configured to execute the steps set forth in the remaining claim elements.⁴⁰⁵

g) *Claim element 17.f.1: "in response to receiving a data location request from a client to retrieve a location string associated with an identification string"⁴⁰⁶ provided in the data location request,"*

OracleNamesAdminGuide discloses this claim element, or renders it obvious alone or in view of OracleUnleashed and Steen. Claim element 17.f.1 builds on claim element 17.e to require each location server be configured to receive a client's data location request. The prior art discloses or renders obvious claim element 17.f.1.

(1) OracleNamesAdminGuide discloses or renders obvious a client sending a data location request.

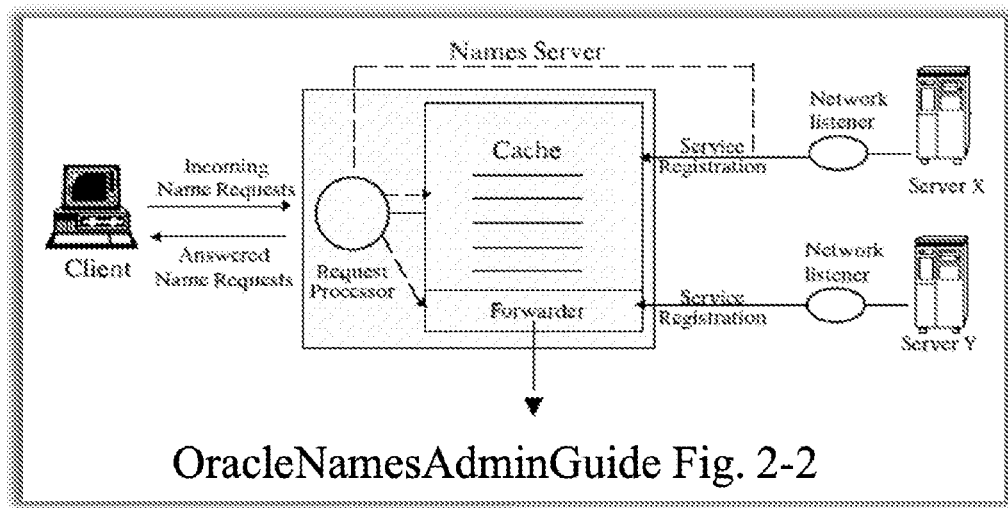
OracleNamesAdminGuide teaches that each client connects to the system and submits data location requests. As discussed above (§III.B), Patent Owner has asserted that the claim term "client" means "a network-attached component that initiates update or lookup of identifier/location mappings from a location server with location request messages." A POSITA would have understood that OracleNamesAdminGuide's client is the claimed "client." For instance, OracleNamesAdminGuide explains that its client is a software component attaching to the network for looking up identifiers from the Names Servers: "The client is the software that requests the name to be resolved. The name itself typically comes from the user's application, or the actual

⁴⁰⁵ See *id.* ¶ 272.

⁴⁰⁶ Claim element 17.d recites "identifier strings." For purposes of this request only, Requester assumes claim element 17.f's "an identification string" refers to one of claim element 17.d's "identifier strings."

user.... When the name is successfully resolved, the client uses it to connect to the intended destination. All interaction with the Names Server and use of the results is transparent to the user.”⁴⁰⁷

And as shown below in Figure 2-2, OracleNamesAdminGuide’s client initiates data location request messages and receives answers from Names Servers: “The true user of Oracle Names is always either *a client application in a client–server connection* Oracle Names is not visible to the user or the application programmer.”⁴⁰⁸



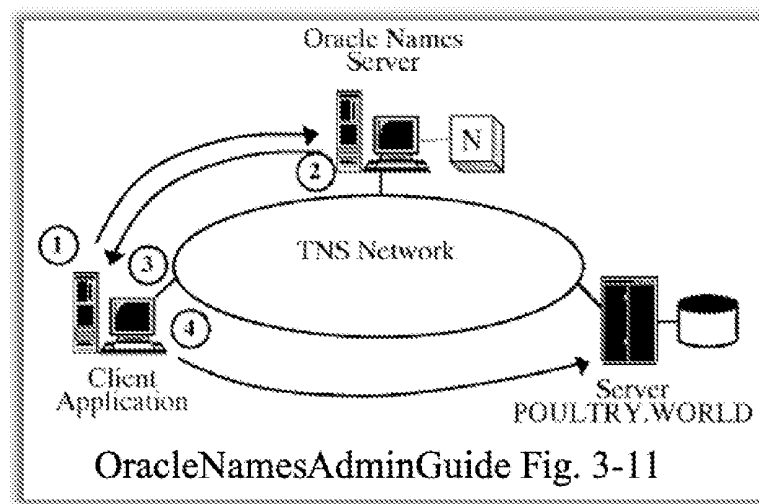
As discussed regarding claim element 17.d, OracleNamesAdminGuide teaches that its system receives a data location request from a client to retrieve a location string associated with an identification string provided in the data location request. And OracleNamesAdminGuide explains that such systems have flat, non-hierarchical configurations. For example, as shown below in the non-hierarchical configuration of Figure 3-11, the client sends the data location request

⁴⁰⁷ OracleNamesAdminGuide at 51-52.

⁴⁰⁸ *Id.* at 24; *see id.* at 33 (“Architecturally, Oracle Names is a network service that provides name resolution to Oracle clients and servers.”), 43-44, 91-92 (similar).

“sqlplus scott/tiger@POULTRY” to retrieve the connection descriptor (i.e., location string) associated with the request’s identifier string POULTRY:

[Step] 1. The client application issues a connection request to SQL*Net of the form: sqlplus scott/tiger@POULTRY where POULTRY is a database service name defined in the Oracle Names Server. SQL*Net determines from the client configuration in the SQLNET.ORA file that the client’s default domain is WORLD and the preferred Names Server is the one shown. SQL*Net sends the Names Server a request to resolve the name POULTRY.WORLD.⁴⁰⁹



To the extent Patent Owner asserts that OracleNamesAdminGuide doesn’t expressly disclose each Names Server being configured to “respon[d] to receiving a data location request from a client to retrieve a location string associated with an identification string provided in the data location request,” a POSITA would have found this limitation obvious. For instance, a POSITA would have found it obvious to have each Names Server in OracleNamesAdminGuide’s non-hierarchical configuration be able to respond to a client’s data location request to retrieve a

⁴⁰⁹ *Id.* at 52-53; *see id.* at 128-29 (describing client sending data location request to Names Server where request includes identifier string for BONES.DEM.MEDICINE).

location string to optimize query efficiency.⁴¹⁰ As Mr. Greene explains, failure to resolve a request causes wasted resources on a network. To avoid waste and optimize performance, a POSITA would have designed servers to minimize failure.⁴¹¹ Accordingly, it would have been beneficial for a POSITA to design each Names Server to be able to handle any request it receives, even when it doesn't contain the requested location information. Accordingly, a POSITA would have found it obvious to configure each Names Server to respond to any data location request to optimize efficiency.⁴¹²

Further, even if OracleNamesAdminGuide alone doesn't render this obvious, it would have been obvious based on Steen's teachings. As described above, Steen teaches the use of "anycasting" to provide a single service from multiple servers. A POSITA would have understood that such a system results in each server being configured to resolve any request it received. As Mr. Greene explains, a POSITA would have been motivated to incorporate Steen's anycasting into OracleNamesAdminGuide's Names Servers to improve the location request responsiveness of the system.⁴¹³

(2) OracleNamesAdminGuide discloses or alone renders obvious a non-hierarchical system with each Names Server being configured to receive the data location request from the client.

Patent Owner has asserted that "claims 17 and 18 recite a *non-hierarchical* cluster configuration. A client can send a query to *any* server in the network—not just its local region."⁴¹⁴

⁴¹⁰ See Greene Decl. ¶ 277.

⁴¹¹ See *id.*

⁴¹² See *id.*

⁴¹³ See *id.* ¶ 278.

⁴¹⁴ 08/04/2022 Patent Owner's Reply to Office Action in Reexamination Control No. 90/019,036 at 23 (emphasis in original) (citation omitted).

Patent Owner uses “non-hierarchical” and “cluster” synonymously.⁴¹⁵ To the extent Patent Owner argues that claim element 17.f.1 requires a non-hierarchical configuration where a client can query any location server, OracleNamesAdminGuide discloses or renders this obvious. As detailed below, OracleNamesAdminGuide teaches a system having a flat, non-hierarchical configuration where a client can send a data location query to each and every Names Server (i.e., location server) in the network. A POSITA would understand this occurs without the client having to navigate a tree structure.⁴¹⁶

OracleNamesAdminGuide teaches that its system has a flat, non-hierarchical configuration, resulting in “all clients” and Names Servers residing on the same network using one naming convention. For example, OracleNamesAdminGuide states that its non-hierarchical configuration has a single WORLD domain for “all clients and services”:

In a flat naming model, when defined by Network Manager, the default domain for all clients is the WORLD domain; that is, all clients and services reside in the WORLD domain. (Without SQLNET.ORA, the default domain is [ROOT] or ”.”) For example, a global database service name in a flat model might be SALES.WORLD.⁴¹⁷

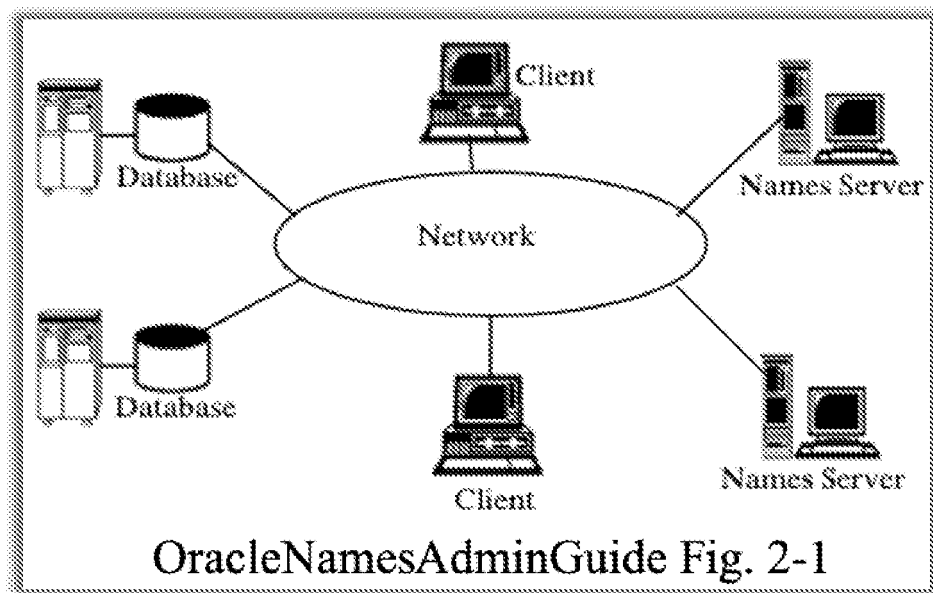
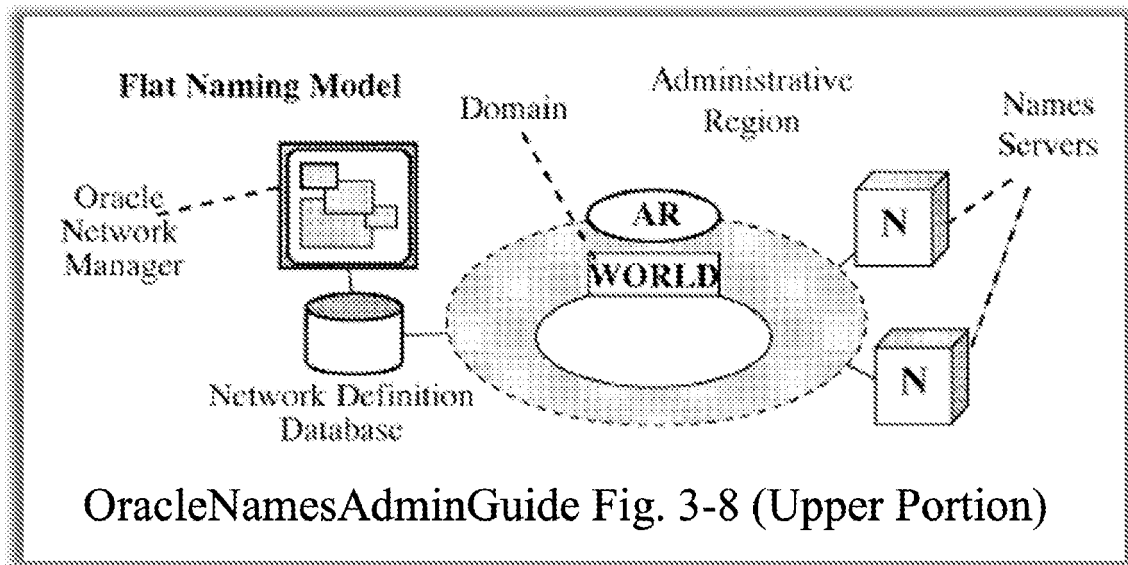
As shown below, OracleNamesAdminGuide’s Figure 3-8 shows this non-hierarchical configuration with a single region AR housing the WORLD domain and directly connected to every

⁴¹⁵ See e.g., Ex. 29 (08/04/2022 Patent Owner’s Reply to Office Action in Reexamination Control No. 90/019,036 at 27-28 (“requires a non-hierarchical cluster configuration”).

⁴¹⁶ See Greene Decl. ¶ 279.

⁴¹⁷ OracleNamesAdminGuide at 91.

Names Server.⁴¹⁸ And as shown below, OracleNamesAdminGuide's Figure 2-1 shows each Names Server and client being connected to the same non-hierarchical network.⁴¹⁹



⁴¹⁸ *Id.* at 45.

⁴¹⁹ *Id.* at 32-33.

Further, OracleNamesAdminGuide discloses that its non-hierarchical configuration works well “until you have several hundreds or thousands of names.”⁴²⁰ Mr. Greene explains that POSITAs using real-world Oracle7 systems employed OracleNamesAdminGuide’s non-hierarchical configuration because such systems typically involved data location information employing less than several hundred names.⁴²¹ Requester hasn’t presented evidence toward OracleNamesAdminGuide’s non-hierarchical configuration in any previous proceeding or reexamination.

Patent Owner has asserted that a “hierarchical configuration restricts the ability of a client to find the authoritative server in the network without undergoing the traversals (iterations) required in navigating a tree structure.”⁴²² OracleNamesAdminGuide discloses that its non-hierarchical configuration doesn’t have a root server or result in the client needing to travel up or down a network tree. As Mr. Greene explains, a POSITA would understand that OracleNamesAdminGuide’s flat, non-hierarchical configuration doesn’t use a tree structure of clients to navigate because it describes name resolution as being “lightning fast.”⁴²³ A POSITA would have appreciated that, while hierarchical structures provide fault tolerance for organization, the response times for queries may not follow the most efficient routes. Accordingly, the lightning fast re-

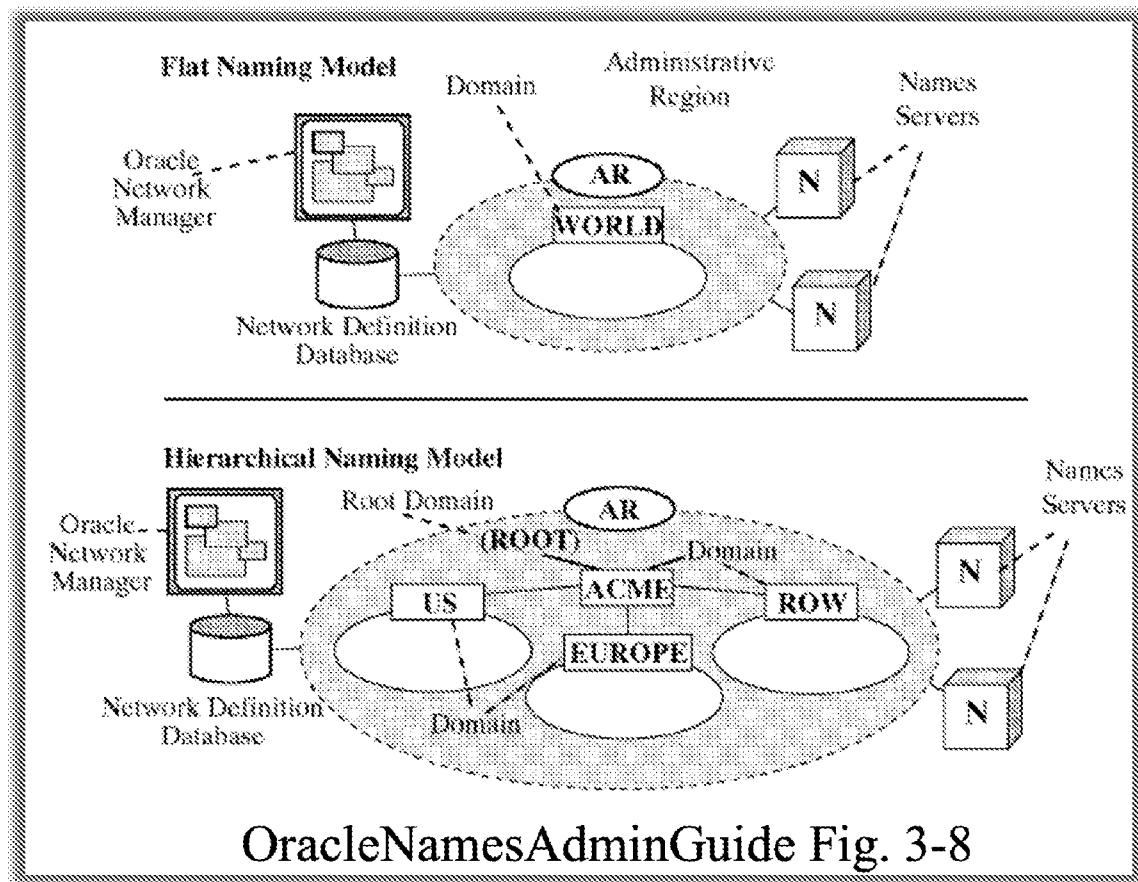
⁴²⁰ *Id.* at 57.

⁴²¹ Greene Decl. ¶ 282.

⁴²² 08/04/2022 Patent Owner’s Reply to Office Action in Reexamination Control No. 90/019,036 at 27-28 (emphasis in original).

⁴²³ Greene Decl. ¶ 283.

sponse times of OracleNamesAdminGuide require more direct pathways to resolve location queries.⁴²⁴ As shown below, OracleNamesAdminGuide's Figure 3-8 contrasts a hierarchical configuration having a root server and sub-domains US, Europe, and Row, against a flat, non-hierarchical configuration having no root server.⁴²⁵



OracleNamesAdminGuide's non-hierarchical system is different that OracleNamesAdminGuide's other embodiments having hierarchical configurations in which a client's request

⁴²⁴ *Id.*

⁴²⁵ OracleNamesAdminGuide at 45.

message are sent up and down branches of a network tree to a root server, such as the example network shown in the reference's Figure C-4 using the "Dynamic Discovery Option."⁴²⁶

Further, Patent Owner may attempt to distinguish claim 17 from OracleNamesAdminGuide's flat, non-hierarchical configuration by asserting that OracleNamesAdminGuide uses a data location mapping approach similar to the Domain Name Service ("DNS") protocol.⁴²⁷ But this assertion lacks merit for at least two reasons. First, OracleNamesAdminGuide details that its flat, non-hierarchical configuration uses a simpler naming convention than DNS where the configuration has a single domain, e.g., WORLD, with each location string having a unique name in the single domain.⁴²⁸ Second, even if OracleNamesAdminGuide were to use a naming convention like DNS, the '640 Patent's specification describes its system as using the NDTP protocol to associate data entities with data location information. The specification discloses that NDTP employs functions in the same way as the DNS protocol for mapping data entities to location information:

NDTP is designed for use in the larger context of a distributed data collection. As such, it supports an architecture, in which information about where data associated with particular application entities, can be managed and obtained independently of the data itself. *One way to understand this is as a highly dynamic DNS for data. DNS maintains a mapping between names and machines.* NDTP and its associated

⁴²⁶ See 08/04/2022 Patent Owner's Reply to Office Action in Reexamination Control No. 90/019,036 at 23-25 (citing OracleNamesAdminGuide at 28, 32, 60, 190-92 toward a system having a hierarchical configuration); '640 Pat. at 16:15-39, 18:5-8, 18:46-52, Figs. 10, 12-13 (describing system with hierarchical configuration with messages being sent up and down network tree).

⁴²⁷ See 08/04/2022 Patent Owner's Reply to Office Action in Reexamination Control No. 90/019,036 at 25-26 (citing OracleNamesAdminGuide at 56-57 toward using a data location mapping standard based on DNS).

⁴²⁸ OracleNamesAdminGuide at 40-41.

servers maintain a mapping between entity identifiers and data locations. *The identifier/location mapping maintained by NDTP servers is much more dynamic (more frequent updates), than the domain name/IP address mapping maintained by DNS.* NDTP is designed to support very fast, very diverse, and very large scale mapping manipulations.⁴²⁹

To any extent OracleNamesAdminGuide alone doesn't render obvious the non-hierarchical cluster structure, it renders it obvious in further view of OracleUnleashed or Steen. As Mr. Greene details, both Steen and OracleUnleashed teach the idea of partitioning servers to improve performance. As a result of combining OracleNamesAdminGuide's Names Servers with Steen and OracleUnleashed's server partitioning, a POSITA would have found it obvious that the Names Servers be arranged in non-hierarchical clusters.

Further, even if they weren't organized into these non-hierarchical clusters, Patent Owner's arguments that hierarchical structures can't employ redirect messages is incorrect. While hierarchical structures *can* find data by following the hierarchies, they aren't limited to doing so. As Steen states, "[a]n implication of [hierarchical designs] is that in the worst case it's always possible to locate every object by following the chain of pointers from the root node. *In practice, we can do much better than this...*"⁴³⁰ As Mr. Greene explains, a POSITA would have been motivated to "do much better than this" to make resolutions "lightning fast" as OracleNamesAdminGuide's stated goal explains.⁴³¹ Hence, as Mr. Greene details, network hierarchies provide *fault tolerance* to ensure data can always be found, not the optimized path for finding it

⁴²⁹ '640 Pat. at 20:4-17.

⁴³⁰ Steen at 105.

⁴³¹ Greene Decl. ¶ 287.

quickly.⁴³² As a result, Patent Owner's alleged distinction over the prior art doesn't reflect the function of the technology itself.

Based on the above teachings, OracleNamesAdminGuide discloses or renders obvious in view of Steen or OracleUnleashed claim element 17f.1.'s each location server being configured to respond to receiving a data location request from a client to retrieve a location string associated with an identification string provided in the data location request.

(3) OracleNamesAdminGuide, OracleUnleashed, and Steen render obvious a client sending a data location request.

Further, OracleNamesAdminGuide, OracleUnleashed, and Steen render obvious claim element 17.f.1. OracleNamesAdminGuide teaches that clients may request network services.

A client can request a network service within its default domain using the service's simple, unqualified name, that is, without specifying a domain name. If a user requests a name without a '.' character in it, the default domain name will be automatically appended to the database service or data-base link name requested.⁴³³

As Mr. Greene explains, a POSITA would have understood these network services include data requests and that data requests would require a data location request.⁴³⁴

Further, even if OracleNamesAdminGuide doesn't disclose this explicitly, a POSITA would have been motivated to have a client be able to connect to any Names Server where each Names Server is configured to respond to receiving a data location request from a client to retrieve a location string associated with an identification string provided in the data location request. For instance, this would have resulted in the effective use of SQL*Net functionality, and

⁴³² *Id.*

⁴³³ OracleNamesAdminGuide at 43-44.

⁴³⁴ See Greene Decl. ¶ 290.

improve query efficiency and optimization of Oracle7 systems to ease users' ability to find their desired data.⁴³⁵

As discussed above, OracleNamesAdminGuide discloses a flat, non-hierarchical configuration in which "[a]ll names are unique within a single domain—the WORLD domain."⁴³⁶ And OracleNamesAdminGuide explains that, "[i]n the most common case of a single, central administrative region, all Names Servers contain identical data, and give identical results."⁴³⁷ Thus, a POSITA would have found it obvious to have a non-hierarchical system where a client can send a query to any Names Server in the network based on OracleNamesAdminGuide and Steen's teachings.

As explained above, Steen teaches a location service to manage and locate data for users. Steen further teaches the use of single services fulfilled by a multitude of servers:

In distributed systems, this way of locating a service is also known as anycasting: *a client requires a particular service*, but is really not interested in which server will handle the request. Using our approach, the service is assigned a unique object handle, and each server registers its network address under that object handle. A client has the service's name resolved to the object handle, which is then subsequently resolved to the address of any server that can handle the request.⁴³⁸

As Mr. Greene explains, in such a system, any server can receive any request and it can still be resolved. A POSITA would have been motivated to apply this to OracleNamesAdminGuide's

⁴³⁵ See *id.* ¶ 291.

⁴³⁶ OracleNamesAdminGuide at 40.

⁴³⁷ *Id.* at 45.

⁴³⁸ Steen at 105.

Names Servers to maximize the servers capable of resolving location queries to provide users with the “lightning fast” results that OracleNamesAdminGuide aspires to.⁴³⁹

Further, as described by OracleUnleashed’s partitioning methods, each Names Server may contain a portion of the location information in the total dataset. As a result, a POSITA would have found it obvious that each of the partitioned subsets must be able to receive and respond to location questions.⁴⁴⁰

Based on the above teachings, OracleNamesAdminGuide and Steen and OracleUnleashed render obvious claim element 17.f.1.

- h) *Claim element 17.f.2: “transmitting a redirect message to the client if the identification string is not associated with a location string at the data location server, wherein the redirect message contains information for use by the client to calculate a location of a different data location server in the plurality of data location servers, wherein the different data location server contains the location string.”*

OracleNamesAdminGuide discloses this claim element, or renders it obvious alone or in view of OracleUnleashed and Steen. Claim element 17.f.2 addresses the scenario of how the system will respond if a location server doesn’t have a location string associated with a client’s requested identification string. In that scenario, this claim element requires that the system transmit a redirect message to the client where the message contains information useable by the client to calculate the location of a different location server that has the relevant location string. The prior art discloses or renders obvious this claim element.

⁴³⁹ See Greene Decl. ¶ 294.

⁴⁴⁰ See *id.* ¶ 295.

(1) OracleNamesAdminGuide discloses or renders obvious transmitting a redirect message to the client.

OracleNamesAdminGuide teaches that each Names Server is configured to transmit a redirect message to the client if the identification string isn't associated with a location string at the Names Server which received the client's data location request message. And OracleNamesAdminGuide discloses that the redirect message contains information for use by the client to calculate a location of a different Names Server where that different Names Server contains the location string.

Like the '640 Patent, OracleNamesAdminGuide discloses that its system will periodically have changes in location information when the network administrator adds a new database or Names server and/or alters a database's address.⁴⁴¹ When those changes occur, the administrator will refresh the memories of the system's Names Servers which results in updates to the identifier strings associated with location strings of the databases.⁴⁴² At times, this process of refreshing the memories will result in some, but not all, Names Servers having the location string for a particular data entity.⁴⁴³

As shown in the below table, OracleNamesAdminGuide discloses two configurations for addressing how the system will respond if a Names Server doesn't have a location string associated with a client's requested identification string.⁴⁴⁴

⁴⁴¹ *E.g.*, '640 Patent at 10:27-59; OracleNamesAdminGuide at 45.

⁴⁴² *See* OracleNamesAdminGuide at 45; Greene Decl. ¶ 299.

⁴⁴³ *See* OracleNamesAdminGuide at 45, 104 ("A method for requesting to add, change, or delete network objects and their associated names from the Names Server is required.... The administrator can then modify the data and refresh the Names Servers at regular intervals."); *see id.* at 37, 199 (describing different refresh frequencies in Names Servers' cache memories); Greene Decl. ¶ 299.

⁴⁴⁴ OracleNamesAdminGuide at 137.

Configuration 1 (Forwarding Enabled)	Configuration 2 (Forwarding Disabled)
The initial Names Server that received the request will forward the request to a second Names Server containing the requested location string, the second Names Server will return the location string to the initial Names Server, then the initial Names Server will return the location string to the client.	The initial Names Server that received the request will send a message to the client where the message contains the address of a second Names Server containing the requested location string, then the client will send its request to the address of the second Names Server.

OracleNamesAdminGuide explains that its system will alternate between these two configurations based on the network administrator turning on and off the SET FORWARDING_AVAILABLE command.⁴⁴⁵ OracleNamesAdminGuide teaches that it's desirable to disable the forwarding function to lessen the workload on each Names Server by having the client, rather than the initially-requested Names Server, calculate the location of the different Names Server that has the requested location string.⁴⁴⁶ OracleNamesAdminGuide states the administrator is able to turn off this command in all Names Servers on the network.⁴⁴⁷

When the administrator has turned off the forwarding command, OracleNamesAdminGuide discloses that the initially-requested Names Server that doesn't have the answer to the client's query "tells the requestor [i.e., client] the address of the Names Server that can answer the

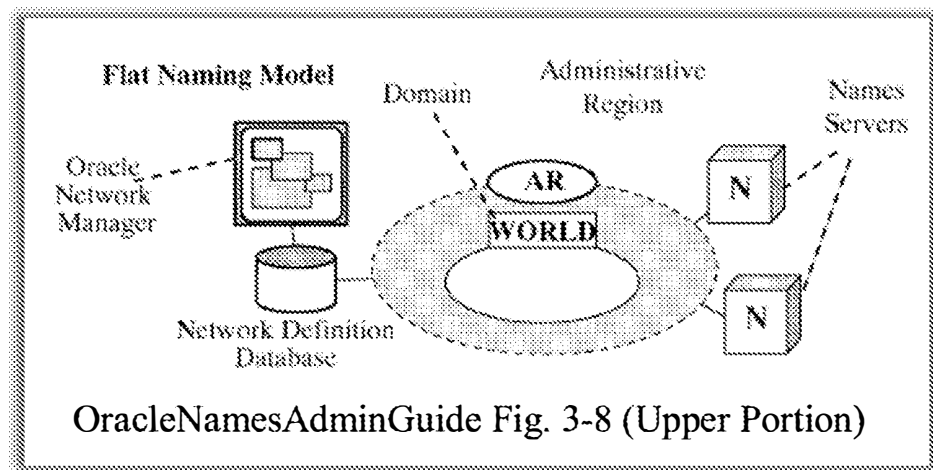
⁴⁴⁵ *Id.* at 137.

⁴⁴⁶ *E.g., id.* at 137; *see also id.* at 119-20 (explaining forwarding command will "Turn on or off name request forwarding for a Names Server).

⁴⁴⁷ *See id.* at 119-20, 137; Greene Decl. ¶ 301.

request.”⁴⁴⁸ The message containing the address of the different Names Server corresponds to the claimed “redirect message.”⁴⁴⁹

Below, OracleNamesAdminGuide’s Figure 3-8 illustrates how this redirect message meets all the requirements of claim element 17.f.2. Figure 3-8 depicts a non-hierarchical system with an upper Names Server and a lower Names Server that cumulatively store all the location strings known in the system. As OracleNamesAdminGuide explains, when the upper Names Server receives a client’s query but lacks the location string to answer that query, a POSITA would understand the upper Names Server transmits a message to the client where the message includes in the address of the lower Names Server housing the location string. The client uses that address to calculate the location of the lower Names Server, then the client sends its data location request to the lower Names Server.⁴⁵⁰



As Mr. Greene explains, a POSITA would understand that a similar scenario transpires when the lower Names Server receives a client’s query but lacks the location string to answer

⁴⁴⁸ OracleNamesAdminGuide at 137.

⁴⁴⁹ See Greene Decl. ¶ 302.

⁴⁵⁰ See OracleNamesAdminGuide at 137; Greene Decl. ¶ 303.

that query. This results in the lower Names Server transmitting a message to the client containing the address of the upper Names Server housing the location string. The client uses that address to calculate the location of the upper Names Server, then client sends its data location request to the upper Names Server.⁴⁵¹

To the extent Patent Owner asserts that OracleNamesAdminGuide doesn't expressly disclose each Names Server "transmitting a redirect message to the client if the identification string is not associated with a location string at the data location server, wherein the redirect message contains information for use by the client to calculate a location of a different data location server in the plurality of data location servers, wherein the different data location server contains the location string," a POSITA would have found this limitation obvious. For instance, to optimize query efficiency, a POSITA would have found it obvious to have each Names Server in OracleNamesAdminGuide's non-hierarchical configuration be able to transmit a redirect message to the client having information the client can use to calculate the location of a different Names Server that contains the requested location string.⁴⁵² A POSITA would have been motivated to do so because this would further the use of OracleNamesAdminGuide's SQL *Net and Names Servers to respond to certain data requests and lessen the workload of the initially-requested Names Server because it no longer needs to perform forwarding functions to other servers containing the requested information.⁴⁵³

⁴⁵¹ See OracleNamesAdminGuide at 137; Greene Decl. ¶ 304.

⁴⁵² See Greene Decl. ¶ 305.

⁴⁵³ See *id.*

Moreover, a POSITA would have been motivated to do so because providing a redirect message to calculate the location of a different Names Server would improve the ability of OracleNamesAdminGuide's system to locate the client's requested information. To accomplish this, a POSITA would have designed OracleNamesAdminGuide's system to detect and resolve problems without placing the burden on the user. This is the DBA's inherent role, and the clear benefits for the user would have motivated a POSITA to modify OracleNamesAdminGuide in this way.⁴⁵⁴

(2) Patent Owner's Assertions Regarding OracleNamesAdminGuide's Redirect Messages Lack Merit.

Patent Owner has asserted that, "[b]y its plain language, the claim [17] requires that each (i.e., every) location server must be configured to return information that the *client uses* to determine a second location server that contain the sought-after location string, if the first queried location does not contain that location string."⁴⁵⁵ OracleNamesAdminGuide discloses or alone renders obvious Patent Owner's asserted requirement for claim 17. This is because a POSITA would understand that OracleNamesAdminGuide's teachings toward redirect messages and a non-hierarchical system with two Names Servers (Figure 3-8) establish that the first queried Names Server will return address information that the client uses to determine the second Names Server that contains the sought-after location string.⁴⁵⁶ Requester hasn't presented evidence toward OracleNamesAdminGuide's non-hierarchical configuration, e.g., Figure 3-8, in any previous proceeding or reexamination.

⁴⁵⁴ See *id.* ¶ 306.

⁴⁵⁵ 08/04/2022 Patent Owner's Reply to Office Action in Reexamination Control No. 90/019,036 at 24 (emphasis in original).

⁴⁵⁶ See OracleNamesAdminGuide at 44-45, 137; Greene Decl. ¶ 307.

Further, Patent Owner has asserted that OracleNamesAdminGuide doesn't disclose claim element 17.f.2 because the initially-requested Names Server returns the address of "next server above it in the network path," rather than the Names Server containing the location string.⁴⁵⁷ But this assertion lacks merit for at least two reasons. First, OracleNamesAdminGuide describes an embodiment having a non-hierarchical configuration, e.g., Figure 3-8, that is a flat network architecture without any Names Servers connecting upward toward a root server.⁴⁵⁸ Second, even if OracleNamesAdminGuide's system were limited to hierarchical configurations, the '640 Patent acknowledges that those configurations still provide redirect messages sent to a second Names Servers functioning as the claimed "client." For instance, like claim 17, claim 18 recites "transmit[ing] a redirect message to the client." And claim 25 depends from claim 18 and recites "a plurality of servers related in a *logical hierarchy* between the client and the data location servers, wherein *each of the plurality of servers is configured to function as a next client and retransmit the data location request to a next logically associated server until a data location server receives the data location request.*" Thus, the '640 Patent expressly teaches that returning the address of next location server above it in the network path would meet claim element 17.f.2 because that server would "function as a next client."

Patent Owner has asserted that OracleNamesAdminGuide doesn't disclose claim element 17.f.2 because disabling the SET FOWARDING_AVAILABLE command results in the client being "not capable of redirecting their requests as Names Servers would" for "foreign names."⁴⁵⁹

⁴⁵⁷ See 08/04/2022 Patent Owner's Reply to Office Action in Reexamination Control No. 90/019,036 at 32.

⁴⁵⁸ See OracleNamesAdminGuide at 44-45, 137; Greene Decl. ¶ 308.

⁴⁵⁹ See 08/04/2022 Patent Owner's Reply to Office Action in Reexamination Control No. 90/019,036 at 30-31.

But this argument is incorrect. In the flat, non-hierarchical configuration, OracleNamesAdminGuide has only one domain, e.g., WORLD, where there aren't any foreign names.⁴⁶⁰ OracleNamesAdminGuide specifies all names in the domain of the non-hierarchical configuration will be unique and accessible to all services and clients connected to the system.⁴⁶¹ As Mr. Greene explains, there would be no foreign names in this configuration and no risk of a client being unable to resolve a foreign name.⁴⁶² Rather, OracleNamesAdminGuide's system functions just like the '640 Patent's non-hierarchical embodiment where each location server is able to transmit to the client a redirect message containing information for use by the client to calculate the location of the different location sever that contains the location string.⁴⁶³

OracleNamesAdminGuide discloses that its forwarding command (enabled or disabled) is preferably for Names Servers having "no local clients" that are servicing request inquiries from "foreign Names Servers."⁴⁶⁴ To the extent Patent Owner asserts this preference teaches away from the claimed "redirect message," this is incorrect for at least three reasons. First, OracleNamesAdminGuide states this is an "intended" preference rather than a requirement. Illustrating this isn't a requirement, OracleNamesAdminGuide specifies that a Names Server with this forwarding command disabled may have "clients who rely directly on that Names Server."⁴⁶⁵ Mr. Greene explains this is particularly relevant to OracleNamesAdminGuide's non-hierarchical configurations with all Names Servers and location strings located on the same network resulting in

⁴⁶⁰ See OracleNamesAdminGuide at 40-41, 44-45.

⁴⁶¹ See *id.*

⁴⁶² Greene Decl. ¶ 309.

⁴⁶³ *Id.*

⁴⁶⁴ OracleNamesAdminGuide at 137.

⁴⁶⁵ *Id.*

no foreign location information and no foreign Names Servers.⁴⁶⁶ OracleNamesAdminGuide warns that, when forwarding is disabled and the Names Server sends redirect messages to the client, the server can't resolve request inquiries for "foreign names."⁴⁶⁷ But this concern is irrelevant to OracleNamesAdminGuide's non-hierarchical configuration because, as explained above, all the data entities are locating in the same region, thus resulting in no foreign names.

Second, even assuming the system has a "foreign Names Server," a POSITA would have understood such a server would be the claimed "client" that is sending data location requests. As discussed above, Patent Owner proposes that a "client" is "a network-attached component that initiates update or lookup of identifier/location mappings from a location server with location request messages." Thus, in this context, a foreign Names Server is a client because it's a network-attached component initiating a lookup of an identifier/location mapping from a Names Server via a location request message. This is consistent with the '640 Patent's specification, which expressly teaches that another location server that sends request messages is the claimed "client." For instance, '640 claim 25 states "each of the plurality of servers is configured to function *as a next client* and retransmit the data location request to a next logically associated server." Thus, the Names Server servicing a data request from a foreign Names Server is servicing a request from the claimed "client" and sending a redirect message back to that client if the server doesn't have a location string associated with the requested identifier.⁴⁶⁸

Third, as Mr. Greene explains, a POSITA would have been motivated to have OracleNamesAdminGuide's non-hierarchical configuration provide each Names Server with code that allows the server to receive requests from and send redirect messages to both clients and foreign

⁴⁶⁶ Greene Decl. ¶ 310.

⁴⁶⁷ OracleNamesAdminGuide at 137.

⁴⁶⁸ See Greene Decl. ¶ 311.

Names Servers.⁴⁶⁹ For those reasons, a POSITA would have found it obvious to modify OracleNamesAdminGuide's system such that every Names Server (not just those with no local clients) has code for sending redirect messages to clients that include information for the client to calculate the location of a different Names Server.⁴⁷⁰

Based on the above teachings, OracleNamesAdminGuide discloses or renders obvious claim element 17.f.2.

(3) OracleNamesAdminGuide, OracleUnleashed, and Steen render obvious the “redirect message” limitations.

Further, OracleNamesAdminGuide and OracleUnleashed or Steen render obvious claim element 17.f.2. OracleNamesAdminGuide describes a distributed database system, focusing on Names Servers that respond to a client's data requests using unique identifiers associated with the data's network address. Similarly, both OracleUnleashed and Steen describe a distributed network configuration that uses well-known name resolution techniques.

As explained above for claim element 17.c, Steen and OracleUnleashed teach partitioning network servers to improve performance. Each of Steen and OracleUnleashed further teach methods of organizing datasets across servers, including hashing.

In particular, OracleUnleashed teaches multiple hashing methods providing the benefit of organizing data using hash clusters and hash joins:

A hash cluster is similar to a cluster but uses a hash function rather than an index to reference the cluster key. A hash cluster stores the data based on the result of a hash function (a numeric function that determines the data block in the cluster based on the value of the cluster key).⁴⁷¹

⁴⁶⁹ See *id.* ¶ 312.

⁴⁷⁰ See *id.*

⁴⁷¹ OracleUnleashed at 394.

Hash joins greatly enhance the performance of two joined tables, where one table is significantly larger than the other. Hash joins replace the "sort-merge" join algorithm and is only supported by the cost-based optimizer. The hash join works by splitting two tables into partitions. A partition is read into memory and a hash table is created. This hash table is then used to map join columns from the other table, eliminating the sometimes resource-expensive sort-merge. In the hash join method, both tables are scanned only once.⁴⁷²

As Mr. Greene details, hashing datasets makes it easier to find a piece of data at a later point.⁴⁷³ As a result, Mr. Greene explains that a POSITA would have been motivated to hash the partitioned location servers taught by the combination of OracleNamesAdminGuide and Oracle-Unleashed. Doing so would allow any Names Servers on the network to "know" the location of a desired piece of location information.⁴⁷⁴ As a result of this knowledge, as Mr. Greene explains, it would have been obvious to a POSITA that, if a Names Server received a request it could not resolve because it lacked the location data, it would redirect the client to a Names Server containing the desired location data.⁴⁷⁵

Likewise, Steen also teaches hashing, specifically in connection with partitioned location servers. As Steen provides, this hashing ensures each subnode is aware of (i.e., knows) the location of any requested location information regardless of which server it's contained on:

Because communication between directory nodes in the original search tree now takes place between their respective subnodes, each subnode should be aware of how the directory node with which it communicates is actually partitioned. This information is contained in a separate tree management service. This service also

⁴⁷² *Id.* at 638.

⁴⁷³ *See* Greene Decl. ¶ 318.

⁴⁷⁴ *See id.*

⁴⁷⁵ *See id.*

maintains the mapping of subnodes to physical nodes. Partitioning and mapping information is assumed to be relatively stable so that it can be easily cached by subnodes.⁴⁷⁶

Steen explains that this awareness minimizes the need to request partition information from other services to identify the location server storing the requested data. For instance, Steen explains that minimizing such requests improves system efficient by decreasing the need for repeated requests: “This assumption is necessary to avoid having to query the management service each time a subnode needs to communicate with its parent or children, which would turn the management service into a potential communication bottleneck.”⁴⁷⁷

Thus, based on OracleUnleashed’s and/or Steen’s teachings, a POSITA would have found it obvious for OracleNamesAdminGuide’s first Names Server, if it lacks an association to the requested location string, to send a redirect message to the client containing redirect information for the client to calculate the location of a different Names Server having the request location string. A POSITA would have been motivated to do so because this would further the use of OracleNamesAdminGuide’s SQL *Net and Names Servers to respond to certain data requests and lessen the workload of the initially-requested Names Server because it no longer needs to perform forwarding functions to other servers containing the requested information.⁴⁷⁸

Further, if a Names Server can’t resolve a client’s information query, a POSITA would have been motivated to have that Names Server transmit a redirect message so the client can cal-

⁴⁷⁶ Steen at 108.

⁴⁷⁷ *Id.*

⁴⁷⁸ See Greene Decl. ¶ 321.

culate the location of another Names Server housing the requested location string. This is because using such redirect messages was an established method to address unresolved location requests, as taught by OracleUnleashed and Steen.⁴⁷⁹

Moreover, a POSITA would have been motivated to do so because providing a redirect message to calculate the location of a different Names Server would improve the ability of OracleNamesAdminGuide's system to locate the client's requested information. To accomplish this, a POSITA would have designed OracleNamesAdminGuide's system to detect and resolve problems without placing the burden on the user. This is the DBA's inherent role, and the clear benefits for the user would have motivated a POSITA to modify OracleNamesAdminGuide in this way.⁴⁸⁰

Because each server can determine the location of location information on the location server network (i.e., "knows" where to find any requested location data), OracleNamesAdminGuide alone or in combination with Steen and/or OracleUnleashed teach that each location server can resolve a data location request in two or fewer traversals.⁴⁸¹ In the first traversal, the client requests its desired location information from any location server, which responds with either the location information of the data or the location information for a location server known to contain the requested location information. In the second traversal, if necessary, the client uses the redirect message to request the location information from the location server known to contain the desired location information. Thus, OracleNamesAdminGuide alone or in combination

⁴⁷⁹ See *id.* ¶ 322.

⁴⁸⁰ See *id.* ¶ 323

⁴⁸¹ See *id.* ¶ 328.

with Steen and/or OracleUnleashed renders obvious resolving any client's data location request in two or fewer traversals.⁴⁸²

Based on the above teachings, OracleNamesAdminGuide, OracleUnleashed, and Steen render obvious claim element 17.f.2.

And in sum, OracleNamesAdminGuide, OracleUnleashed, and Steen render obvious independent claim 17.

4. Independent Claim 18 is Obvious Over OracleNamesAdminGuide in View of OracleUnleashed and Steen.

Claim 18 recites a system similar to the system recited in claim 17. As such, during the prior reexamination, Patent Owner asserted that OracleNamesAdminGuide's hierarchical configuration fails to teach the requirements of claim 18 toward redirect messages. This Request focuses on OracleNamesAdminGuide's teachings toward a system having a non-hierarchical configuration that, even under Patent Owner's interpretation of claim 18, discloses or renders obvious all limitations alone or in view of OracleUnleashed and Steen.

- a) *Claim element 18.pre: "A system for retrieving data location information for data stored in a distributed network, the system comprising:"*

Claim 18's preamble is the same as claim 17's preamble. To the extent the preamble is limiting, for all the reasons shown for claim 17's preamble, OracleNamesAdminGuide discloses an Oracle database system for retrieving location information for data stored in a distributed network.⁴⁸³

- b) *Claim element 18.a: "a data repository configured to store data,"*

⁴⁸² See *id.*

⁴⁸³ See *id.* ¶329.

Claim element 18.a is substantially similar to claim element 17.a. For all the reasons shown for claim element 17.a, OracleNamesAdminGuide discloses claim element 17.a or renders it obvious alone or in view of OracleUnleashed and Steen.⁴⁸⁴

c) *Claim element 18.b: “wherein the data is associated with an identifier string;”*

Claim element 18.b is substantially similar to claim element 17.b. For all the reasons shown for claim element 17.b, OracleNamesAdminGuide discloses claim element 18.b or renders it obvious alone or in view of OracleUnleashed and Steen.⁴⁸⁵

d) *Claim element 18.c: “a client responsive to a data query to query a data location server for location information associated with the identifier string;”*

Claim element 18.c is substantially similar to claim elements 17.c and 17.f.1. Claim 18.c is the first instance of a limitation requiring “location information.” As shown above regarding claim 17’s preamble, OracleNamesAdminGuide’s connection description is “location information” because each connection description includes information—such as the host name, port, and database instance—that pertains to the location of data in the network.⁴⁸⁶ In the BONES.DEM.MEDICINE example, OracleNamesAdminGuide teaches the location information includes the host cowboy, port 1522, database rodeo, connection type TCP that pertains to the location of data associated with the BONES.DEM.MEDICINE alias.⁴⁸⁷

⁴⁸⁴ See *id.* ¶ 330.

⁴⁸⁵ See *id.* ¶ 331.

⁴⁸⁶ *Id.* at 23-24, 52-53, 128-29.

⁴⁸⁷ See *id.* at 128-29; Greene Decl. ¶ 332.

Further, for all the reasons shown for claim elements 17.c and 17.f.1, OracleNamesAdminGuide discloses claim element 18.c or renders it obvious alone or in view of OracleUnleashed and/or Steen.⁴⁸⁸

- e) *Claim element 18.d: “a data location server network comprising a plurality of data location servers,”*

Claim element 18.d is substantially similar to claim element 17.c. For all the reasons shown for claim element 17.c, OracleNamesAdminGuide discloses claim element 18.d or renders it obvious alone or in view of OracleUnleashed and Steen.⁴⁸⁹

- f) *Claim element 18.e: “at least one of the plurality of data location servers containing location information associated with the identifier string,”*

Claim element 18.e is substantially similar to claim element 17.d. For all the reasons shown for claim element 17.d, OracleNamesAdminGuide discloses claim element 18.e or renders it obvious alone or in view of OracleUnleashed and Steen.⁴⁹⁰

- g) *Claim element 18.f: “wherein each of the plurality of data location servers comprises computer executable code configured to execute the following steps in response to receiving a data location request from the client:”*

Claim element 18.f is substantially similar to claim element 17.e. For all the reasons shown for claim element 17.e, OracleNamesAdminGuide discloses claim element 18.f or renders it obvious alone or in view of OracleUnleashed and/or Steen.⁴⁹¹

- h) *Claim element 18.g: “if the data location server contains the location string associated with the identification string provided in the data location request, the data location server transmits location*

⁴⁸⁸ See Greene Decl. ¶ 333.

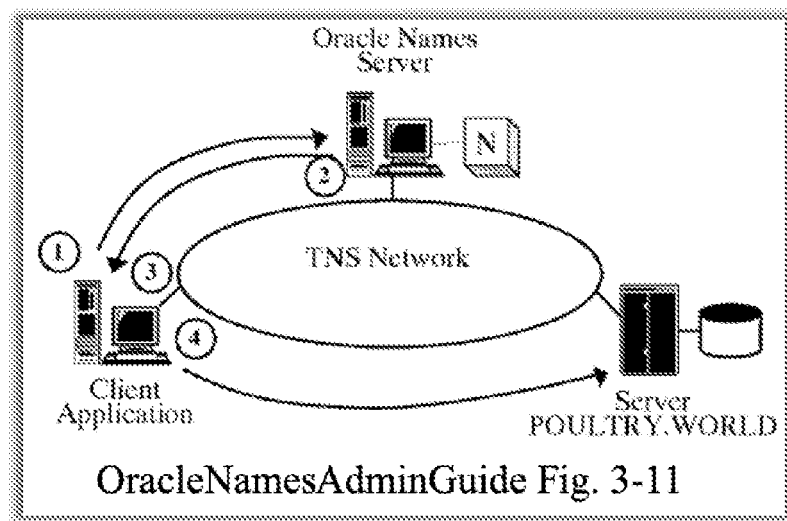
⁴⁸⁹ See *id.* ¶ 334.

⁴⁹⁰ See *id.* ¶ 335.

⁴⁹¹ See *id.* ¶ 336.

information for use by the client to calculate a location of the data associated with the identification string;"

OracleNamesAdminGuide discloses this claim element or renders it obvious alone or in view of OracleUnleashed and Steen. As shown regarding claim elements 17.c, 17.f.1, and 18.c, OracleNamesAdminGuide teaches its Names Server (i.e., location server) receives a query for a connection descriptor (i.e., location information) associated with a network-service-name-alias (i.e., identification string). As shown below in the example for Figure 3-1, OracleNamesAdminGuide explains that, in response to the POULTRY query (step 1), the Names Server checks whether it contains location information associated with the identification string for POULTRY.WORLD (step 2). If it does, the Names Server sends the location information, including the host name, port, and database instance, to the client (step 2). As shown below in Figure 3-1, steps 3-4 include the client using the location information to calculate a location of the data associated with the POULTRY.WORLD alias.⁴⁹²



⁴⁹² See OracleNamesAdminGuide at 52-53; see also *id.* at 22 (“When making a connection, the client specifies the name of a destination and Oracle Names provides the address.”), 51-52 (“When the name is successfully resolved, the client uses it to connect to the intended destination.”).

Further, as shown regarding claim elements 17.c, 17.f.1, and 18.c, OracleNamesAdminGuide teaches another example to illustrate the QUERY command in which the client sends a data location request containing the identification string for BONES.DEM.MEDICINE. If the queried Names Server contains the location string associated with that identification string, the Names Server transmits location information, including “ADDR:... (DESCRIPTION=(ADDRESS= (COMMUNITY=tcp)(PROTOCOL=TCP) (Host=cowboy) (Port=1522))(CONNECT_DATA=(SID=rodeo)))” to the client.⁴⁹³ Like the example in Figure 3-1, the client uses the location information (e.g., host cowboy, port 1522, database rodeo, connection type TCP) to calculate the location of the data associated with the BONES.DEM.MEDICINE alias.⁴⁹⁴

As shown regarding claim elements 17.c, 17.f.1, and 18.c, OracleUnleashed or Steen’s teachings further render obvious claim element 18.g. In each of Steen and OracleUnleashed, the purpose of a location service is getting the desired location information to the requesting client. As a result, it would have been appreciated by a POSITA that any system designed to help locate data, like Steen’s and OracleUnleashed’s, would return the requested information to the client if it contained it.⁴⁹⁵

Thus, based on OracleUnleashed or Steen, a POSITA would have been motivated to have OracleNamesAdminGuide’s Names Server, if it contains the location information associated with the network-service-name-alias (i.e., identifier string) provided in the data location request, transmit location information for use by the client to calculate a location of the data associated

⁴⁹³ See *id.* at 128-29.

⁴⁹⁴ See *id.*; Greene Decl. ¶ 338.

⁴⁹⁵ See Greene Decl. ¶ 339.

with the alias. For instance, this would have resulted in the effective use of SQL*Net functionality, and improve query efficiency and optimization of Oracle7 systems to ease users' ability to find their desired data.

Based on the above teachings, OracleNamesAdminGuide and OracleUnleashed and Steen disclose or render obvious claim element 18.g.

- i) *Claim element 18.h: "if the data location server does not contain the location string associated with the identification string, the location server transmits a redirect message to the client, wherein the redirect message contains redirect information for use by the client to calculate a location of a different data location server in the plurality of data location servers, wherein the different data location server contains the location string."*

Claim element 18.h is substantially similar to claim element 17.f.2. For all the reasons shown for claim element 17.f.2, OracleNamesAdminGuide discloses claim element 18.h or renders it obvious alone or in view of OracleUnleashed and Steen.⁴⁹⁶

OracleNamesAdminGuide alone or in view of Steen or OracleUnleashed renders obvious these claim elements. OracleNamesAdminGuide describes a distributed database system, focusing on Names Servers that respond to a client's data requests using unique identifiers associated with the data's network address. Similarly, both OracleUnleashed and Steen describe a distributed network configuration that uses well-known name resolution techniques.

As explained above for claim elements 17.c and 18.d, Steen and OracleUnleashed teach partitioning network servers to improve performance. Each of Steen and OracleUnleashed further teach methods of organizing datasets across servers, including hashing.

⁴⁹⁶ See *id.* ¶ 342.

In particular, OracleUnleashed teaches multiple hashing methods providing the benefit of organizing data using hash clusters and hash joins:

A hash cluster is similar to a cluster but uses a hash function rather than an index to reference the cluster key. A hash cluster stores the data based on the result of a hash function (a numeric function that determines the data block in the cluster based on the value of the cluster key).⁴⁹⁷

Hash joins greatly enhance the performance of two joined tables, where one table is significantly larger than the other. Hash joins replace the "sort-merge" join algorithm and is only supported by the cost-based optimizer. The hash join works by splitting two tables into partitions. A partition is read into memory and a hash table is created. This hash table is then used to map join columns from the other table, eliminating the sometimes resource-expensive sort-merge. In the hash join method, both tables are scanned only once.⁴⁹⁸

As Mr. Greene details, hashing datasets makes it easier to find a piece of data at a later point.⁴⁹⁹ As a result, Mr. Greene explains that a POSITA would have been motivated to hash the partitioned location servers taught by the combination of OracleNamesAdminGuide and OracleUnleashed. Doing so would allow any Names Servers on the network to "know" the location of a desired piece of location information.⁵⁰⁰ As a result of this knowledge, as Mr. Greene explains, it would have been obvious to a POSITA that, if a Names Server received a request it could not resolve because it lacked the location data, it would redirect the client to a Names Server containing the desired location data.⁵⁰¹

⁴⁹⁷ OracleUnleashed at 394.

⁴⁹⁸ *Id.* at 638.

⁴⁹⁹ *See* Greene Decl. ¶ 318.

⁵⁰⁰ *See id.*

⁵⁰¹ *See id.*

Likewise, Steen also teaches hashing, specifically in connection with partitioned location servers. As Steen provides, this hashing ensures each subnode is aware, i.e., knows, the location of any requested location information regardless of which server it is contained on:

Because communication between directory nodes in the original search tree now takes place between their respective subnodes, each subnode should be aware of how the directory node with which it communicates is actually partitioned. This information is contained in a separate tree management service. This service also maintains the mapping of subnodes to physical nodes. Partitioning and mapping information is assumed to be relatively stable so that it can be easily cached by subnodes.⁵⁰²

Steen explains that this awareness minimizes the need to request partition information from other services to identify the location server storing the requested data. For instance, Steen explains that minimizing such requests improves system efficient by decreasing the need for repeated requests: “This assumption is necessary to avoid having to query the management service each time a subnode needs to communicate with its parent or children, which would turn the management service into a potential communication bottleneck.”⁵⁰³

Thus, based on OracleUnleashed’s and/or Steen’s teachings, a POSITA would have found it obvious for OracleNamesAdminGuide’s first Names Server, if it lacks an association to the requested location string, to send a redirect message to the client containing redirect information for the client to calculate the location of a different Names Server having the request location string. A POSITA would have been motivated to do so because this would further the use of OracleNamesAdminGuide’s SQL *Net and Names Servers to respond to certain data requests

⁵⁰² Steen at 108.

⁵⁰³ *Id.*

and lessen the workload of the initially-requested Names Server because it no longer needs to perform forwarding functions to other servers containing the requested information.⁵⁰⁴

Further, if a Names Server can't resolve a client's information query, a POSITA would have been motivated to have that Names Server transmit a redirect message so the client can calculate the location of another Names Server housing the requested location string. This is because using such redirect messages was an established method to address unresolved location requests, as taught by OracleUnleashed and Steen.⁵⁰⁵

Moreover, a POSITA would have been motivated to do so because providing a redirect message to calculate the location of a different Names Server would improve the ability of OracleNamesAdminGuide's system to locate the client's requested information. To accomplish this, a POSITA would have designed OracleNamesAdminGuide's system to detect and resolve problems without placing the burden on the user. This is the DBA's inherent role, and the clear benefits for the user would have motivated a POSITA to modify OracleNamesAdminGuide in this way.⁵⁰⁶

In sum, OracleNamesAdminGuide and OracleUnleashed and Steen render obvious independent claim 18.

5. OracleNamesAdminGuide in View of OracleUnleashed and Steen Renders Obvious Dependent Claim 24.

OracleNamesAdminGuide renders obvious claim 24, alone or in view of OracleUnleashed and/or Steen. Claim 24 recites that "[t]he system of claim 18, wherein the location information comprises a portion of a hash table distributed over the plurality of data location servers."

⁵⁰⁴ See Greene Decl. ¶ 321.

⁵⁰⁵ See *id.* ¶ 322.

⁵⁰⁶ See *id.* ¶ 323.

During the prior reexamination, Patent Owner asserted that OracleNamesAdminGuide's hierarchical configuration fails to teach the requirements of claim 24 toward hash tables. This Request focuses on OracleNamesAdminGuide's teachings toward a system having a non-hierarchical configuration that renders obvious claim 24 based on OracleUnleashed's and Steen's teachings.

a) The '640 Patent Establishes the Obviousness of Having Hash Table Distributed Over the Names Servers.

The '640 Patent establishes that it was already well known in the art for a computer network system to use a hash table to locate information stored in the system. The '640 Patent specifies that it uses one, and only one, hash table configuration based on the "well-known function" hashpjw for distributing the hash table over the locations servers (e.g., NDTP servers):

The appropriate NDTP server is selected from the table in the NDTP_RDR_RSP message by applying a **well-known function** to the identifier string and using the function result as an index into the NDTP server table.

The well-known function preferably applied is the hash pjw function presented by Aho, Sethi and Ullman in their text *Compilers, Principles, Techniques and Tools*:

```
uint32_t
hash (uint8_t *s, uint32_t slen, uint32_t size)
{
    uint32_t g;
    uint32_t i;
    uint32_t h = 0;
    uint8_t c;
    for (i = 0; i < slen; i++) {
        c = s[i];
        h = (h << 4) + c;
        g = (h & 0xf0000000);
        if (g) {^
            h ^= g >> 24;
            h = g;
        }
    }
```

```
    }  
    return h % size;  
}.507
```

Further, during the original examination of the ‘640 Patent, the Examiner took official notice that “use of a [] hashing operation in a computer networking environment was well known in the art at the time the invention was made.”⁵⁰⁸ This was confirmed in the related reexamination for the ‘170 Patent, where the Reexamination Specialist found that “the use of hash functions was well known in the database art long before the date of the invention.”⁵⁰⁹ Similarly, during the original examination, the Applicant admitted that the ‘640 Patent’s “disclosure illustrates this using the standard hashpjw function from a reference computer science text book.... [T]he system distributes hash-function computation, or creates a ‘distributed hash table’.”⁵¹⁰

Thus, the ‘640 Patent acknowledges that motivation existed in a POSITA’s general knowledge before the earliest priority date to have OracleNamesAdminGuide’s location information comprise a portion of a hash table distributed over the plurality of Names Servers. This establishes that claim 24 is obvious over the prior art. And a POSITA would have had a reasonable expectation of success in modifying OracleNamesAdminGuide’s redirect message to include a portion of a hash table distributed over the Names Servers because system messages containing such fields were well known in the art. As Mr. Greene explains, a POSITA would have under-

⁵⁰⁷ See ‘640 Patent at 14:64-15:21.

⁵⁰⁸ ‘640 History at 447-48.

⁵⁰⁹ 09/27/2022 ‘170 Patent Notice of Intent to Issue Reexamination Certificate at 11.

⁵¹⁰ *Id.* at 522.

stood a redirect message includes information needed by the client to make its request to the intended other location server.⁵¹¹ A POSITA would have understood that, for a location server network organized by hashing, the portion of the hash table identifying which location server contains the requested location information would constitute such necessary information. Thus, it would have been obvious to a POSITA that for the client to know where it should redirect its request, the redirect message should include the necessary portion of the hash table informing it which location server to send its redirected message to.⁵¹²

To the extent Patent Owner argues that claim 24 requires “a single hash table” that is “divided into portions that are distributed over the data location servers in the data location server network,”⁵¹³ a POSITA would have found this configuration obvious. For example, each location server would only need to use the hash table for location information that it didn’t store. Accordingly, as Mr. Greene explains, it would have been obvious to a POSITA that only the portions of the hash table describing the contents of other location servers would need to be on the server. As a result, in a scenario where there were only two location servers, each location server would contain a half portion of a single hash table describing the data location server network.⁵¹⁴

b) *OracleNamesAdminGuide, Steen, and OracleUnleashed Render Obvious Having a Hash Table Distributed Over the Names Servers.*

Further, Steen’s and OracleUnleashed’s teachings render obvious OracleNamesAdminGuide’s location information comprising a portion of a hash table distributed over the plurality of

⁵¹¹ See Greene Decl. ¶ 349.

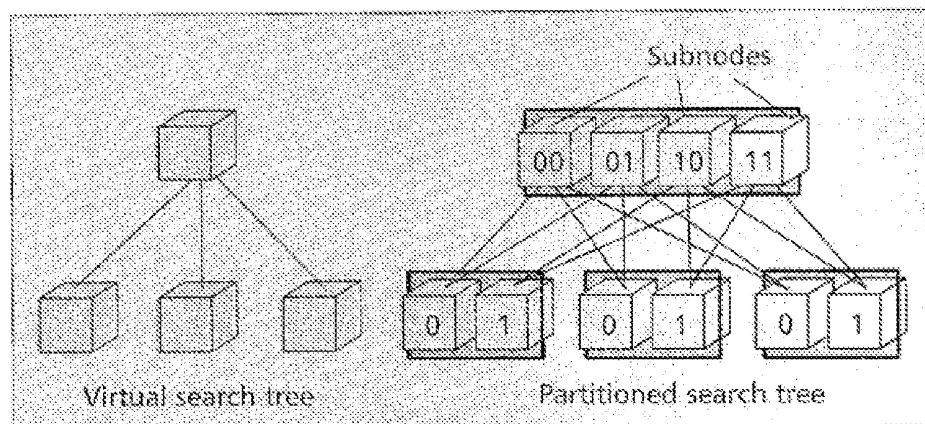
⁵¹² See *id.*

⁵¹³ 08/04/2022 Patent Owner’s Reply to Office Action in Reexamination Control No. 90/019,036 at 42.

⁵¹⁴ See Greene Decl. ¶ 350.

Names Servers. Steen teaches organizing location information by distributing portions of a hash table over multiple location servers. For instance, Steen teaches hashing to organize identifier string-location information on its directory subnode location servers. As shown below in Figure 6, Steen describes using the first n-bits of an object's handle to identify the subnode responsible in a hashing method:

As an example, we can use the first n bits of an object's handle to identify the subnode responsible for that object...To illustrate, Fig. 6 shows a search tree in which the root node has been partitioned into four subnodes based on the first two bits of the object handle ($n = 2$), and each of the leaf nodes into two subnodes ($n = 1$).⁵¹⁵



■ **Figure 6.** A search tree and a corresponding logical tree after partitioning the directory nodes into subnodes.

As Mr. Greene explains, a POSITA would have understood that Steen's use of bits to organize the object handle contents across servers applies a basic hash function.⁵¹⁶ Accordingly, a POSITA would see Steen's method as a simple hashing technique for data organization.

⁵¹⁵ Steen at 108.

⁵¹⁶ See Greene Decl. ¶ 352.

Steen further teaches that more complex hashing methods exist and should be used in systems like OracleNamesAdminGuide's: "We note that we have developed more realistic hashing-based methods than explained here. For example, we also have to take into account that the total number of links between parent and children subnodes remains manageable."⁵¹⁷ A POSITA would have understood that such "realistic hashing-based methods" includes more complex hashing methods, including hash functions. Accordingly, it would have been obvious to a POSITA that, if the division of location information data was more complex, it would be beneficial to organize its distribution with a hash function.⁵¹⁸

And Steen describes that a POSITA would find important for each location server to be aware of how the location information has been partitioned and for each location server to easily store the hashing information:

Because communication between directory nodes in the original search tree now takes place between their respective subnodes, each subnode should be aware of how the directory node with which it communicates is actually partitioned. This information is contained in a separate tree management service. This service also maintains the mapping of subnodes to physical nodes. Partitioning and mapping information is assumed to be relatively stable so that it can be easily cached by subnodes.⁵¹⁹

As Mr. Greene explains, a POSITA would have understood the location servers will store this partitioning and mapping information and use the hash function to determine which location information should be stored on which location server.⁵²⁰ Steen emphasizes the need to minimize each location server's need to request partition information from other services. Steen explains

⁵¹⁷ Steen at 108.

⁵¹⁸ See Greene Decl. ¶ 353.

⁵¹⁹ Steen at 108.

⁵²⁰ See Greene Decl. ¶ 355.

that excessive requests can make the system less efficient by demanding repeated requests: “This assumption is necessary to avoid having to query the management service each time a subnode needs to communicate with its parent or children, which would turn the management service into a potential communication bottleneck.”⁵²¹

Similarly, OracleUnleashed describes the use of hashing to organize datasets across multiple servers. OracleUnleashed teaches multiple hashing methods that help organize data, including hash clusters and hash joins:

A hash cluster is similar to a cluster but uses a hash function rather than an index to reference the cluster key. A hash cluster stores the data based on the result of a hash function (a numeric function that determines the data block in the cluster based on the value of the cluster key).⁵²²

Hash joins greatly enhance the performance of two joined tables, where one table is significantly larger than the other. Hash joins replace the “sort-merge” join algorithm and is only supported by the cost-based optimizer. The hash join works by splitting two tables into partitions. A partition is read into memory and a hash table is created. This hash table is then used to map join columns from the other table, eliminating the sometimes resource-expensive sort-merge. In the hash join method, both tables are scanned only once.⁵²³

As Mr. Greene provides, hashing tables makes it easier to find a piece of data when needed.⁵²⁴ As a result, Mr. Greene explains that a POSITA would have been motivated to hash the partitioned location servers taught by the combination of OracleNamesAdminGuide and OracleUnleashed. Doing so would have allowed any component on the network to “know” where a

⁵²¹ Steen at 108.

⁵²² OracleUnleashed at 394.

⁵²³ *Id.* at 638.

⁵²⁴ See Greene Decl. ¶ 357.

desired piece of location information was located.⁵²⁵ As a result of this knowledge, as Mr. Greene explains, it would have been obvious to a POSITA for OracleNamesAdminGuide's location information to include a portion of a hash table distributed over the Names Servers.⁵²⁶

As explained regarding claim element 18.h, OracleNamesAdminGuide, OracleUnleashed, and Steen render obvious sending a redirect message with information necessary to identify the Names Server containing the object data for a given object tag. As Mr. Greene explains, the mapping information or the relevant portion of the hash table would be necessary for the client to identify the correct Names Server and send the request to it.

Because OracleNamesAdminGuide and OracleUnleashed share the common goal of parsing data sets quickly, a POSITA would have been motivated to use OracleUnleashed's hashing table techniques in OracleNamesAdminGuide's Names Servers to quickly and efficiently determine the location of the requested data.⁵²⁷ It was understood in the art that DBAs would actively search for resources to solve problems that may arise in the specific database environment that they are working in.⁵²⁸

Mr. Greene explains that, as a system's configuration becomes more complex requiring more information to uniquely identify the data's location, the storage and processing requirements for the OracleNamesAdminGuide's Names Servers increase along with it.⁵²⁹ A DBA would have been motivated to minimize these burdens and operate the Names Servers at optimal

⁵²⁵ *See id.*

⁵²⁶ *See id.*

⁵²⁷ *See* Greene Decl. ¶ 359.

⁵²⁸ *See id.*

⁵²⁹ *See id.* ¶ 360.

efficiency. This would have furthered OracleNamesAdminGuide's goal of having its Names Servers identify the requested data's location "lightning fast."⁵³⁰

Mr. Greene explains that, based on the desire to maximize efficiency in running queries, a POSITA would have been motivated to combine OracleNamesAdminGuide's and OracleUnleashed's techniques of locating data using hashing location tables to quickly return location information associated with the requested data.⁵³¹

Thus, a POSITA would have been motivated and found it obvious to have OracleNamesAdminGuide's location information include a portion of a hash table distributed over multiple Names Servers. This would have furthered OracleNamesAdminGuide's goal of having its Names Servers identify the requested data's location "lightning fast."⁵³²

A POSITA would have had a reasonable expectation of success in modifying OracleNamesAdminGuide's location server identification, i.e., the contents of a redirect message, to include a portion of a hash table distributed over the Names Servers because system messages containing such fields was well known in the art. As Mr. Greene explains, a POSITA would have understood a redirect message to include information needed by the client to make its request to the intended other location server.⁵³³ For example, as Steen describes, this information for locating the requested data on a location server should be easily cached on each location server: "Partitioning and mapping information is assumed to be relatively stable so that it can be *easily cached by subnodes*."⁵³⁴ Accordingly, a POSITA would have appreciated that, for a location

⁵³⁰ OracleNamesAdminGuide at 60; *see* Greene Decl. ¶ 560

⁵³¹ *See* Greene Decl. ¶ 361.

⁵³² OracleNamesAdminGuide at 60.

⁵³³ *See* Greene Decl. ¶ 363.

⁵³⁴ Steen at 108.

server network organized by hashing, the portion of the hash table identifying which location server contained the requested location information would constitute such necessary information, e.g., the cache mapping and partitioning information. Thus, it would have been obvious to a POSITA that, for the client to know where it should redirect its request to, the redirect message will include the necessary portion of the hash table informing it which location server to send its redirected message to.⁵³⁵

To the extent Patent Owner argues that claim 24 requires a single hash table that is “divided into portions that are distributed over the data location servers in the data location server network,”⁵³⁶ Steen’s and OracleUnleashed’s teachings would have rendered this configuration obvious. For example, each location server would only need to use the hash table for location information that it did not store. Accordingly, as Mr. Greene explains, it would be obvious to a POSITA that only the portions of the hash table describing the contents of other location servers would need to be on the server.⁵³⁷ As a result, in a scenario where there were only two location servers, one server would contain a portion of a single hash table describing the data location server network, and the other server would contain the rest of the hash table.⁵³⁸

In sum, OracleNamesAdminGuide alone or in view of Steen or OracleUnleashed renders obvious dependent claim 24.

⁵³⁵ See Greene Decl. ¶ 363.

⁵³⁶ 08/04/2022 Patent Owner’s Reply to Office Action in Reexamination Control No. 90/019,036 at 42.

⁵³⁷ See Greene Decl. ¶ 364.

⁵³⁸ See *id.*

F. SNQ 3: Steen in View of Wolff and/or Skagerwall Renders Obvious Independent Claims 17 and 18 And Dependent Claim 24.

Claims 17 and 18 recite a system having a data repository storing a data identified by an identifier string, and multiple location servers storing location information for multiple data entities wherein at least one location server includes location information associated with the identifier string and each location server stores a portion of the location information based on a hash function. Claim 24 depends from claim 18 and recites the distribution of location information across servers using an index location store organized according to a hash table.

Additionally, based on Patent Owner's statements in the previous reexaminations, claims 17, 18, and 24 also require that the location servers be organized in a non-hierarchical cluster structure, that each location server be able to respond to any location request with either the location of the data or the location of a server known to contain the requested location data without an intervening traversal, and that each request must be resolved in two or fewer traversals. As described below, Steen alone or in combination with Wolff and/or Skagerwall renders obvious each claim regardless of Patent Owner's arguments based on additional limitations.

Steen teaches a mixed topology (i.e., both hierarchical and non-hierarchical) system for managing and retrieving data stored in a distributed network using an object name.⁵³⁹ In Steen's system, there are two levels for managing and locating an object's contact addresses. In the first level, Steen describes using a directory system with partitioned non-hierarchical subnodes for resolving an object "name" to an object "handle."⁵⁴⁰ The object handle includes location information for a contact record. Thus, in the first level, Steen describes a directory system for locating data (i.e., a data location server network) that resolves object names, which correspond to the

⁵³⁹ Steen at Abstract.

⁵⁴⁰ *Id.* at 105.

claimed “identifiers,” to object handles, which correspond to the claimed “location information.” And Steen describes that high traffic directory nodes (i.e., location servers) should be partitioned into a plurality of servers to optimize performance, forming non-hierarchical clusters.⁵⁴¹

Further, Steen discloses or renders obvious that the partitioned servers should be organized using hashing methods where the server-content is tracked with a hash table.⁵⁴² Steen further teaches or renders obvious using a redirect message to redirect a client to another location server if the initial location server lacks the location information responsive to the client’s query.⁵⁴³ Accordingly, Steen’s first level constitutes a system for resolving any location request within two traversals by responding to each location request with either the requested location data or a redirect message to redirect a client to another location server that contains the requested location data without an intervening traversal if the initial location server lacks the location information responsive to the client’s query. Then, in the second level, Steen teaches a hierarchical system whereby the user uses the location information (i.e., object handle) to obtain the object addresses from the contact record.⁵⁴⁴ Thus, Steen teaches a two-level, mixed topology system for retrieving location information in a distributed system.

To the extent Patent Owner asserts that Steen doesn’t disclose all elements of the challenged claims, Steen explains that making location services scalable is a primary goal for its system.⁵⁴⁵ As a result, a POSITA would have been motivated to look at other location systems

⁵⁴¹ *Id.* at 104.

⁵⁴² *Id.* at 108.

⁵⁴³ *E.g., id.* at 108.

⁵⁴⁴ *Id.* at 105.

⁵⁴⁵ *Id.* at Abstract.

providing additional scalability. Skagerwall teaches such scalability: “It is a key aspect of the invention that the database of object data is distributed over a plurality of directory servers DS₁, DS₂, ..., DS_n, allowing that the number of objects can be scaled.”⁵⁴⁶

Specifically, as described above for SNQ 1, Skagerwall discloses a system having servers storing location information wherein a client may request the location of data using an object tag (i.e., identifier) associated with the data object’s server addresses (i.e., location strings/location information).⁵⁴⁷ And Skagerwall describes using a hash function to organize data according to object tags and mapping server-content according to a hash table.⁵⁴⁸ Skagerwall further teaches or renders obvious using a redirect message to redirect a client to another location server if the initial location server lacks the location information responsive to the client’s query. Skagerwall explains that minimizing “time consuming” processes is necessary to optimize performance for networks like these.⁵⁴⁹ As a result, a POSITA would be motivated to look to Skagerwall for any advantages to incorporate into Steen’s system to improve its location service.

Further, Wolff’s teachings render obvious using a redirect message to redirect a client to another location server. Steen, Skagerwall, and Wolff describe managing distributed networks operating using client-server relationships.⁵⁵⁰ Skagerwall explains there are benefits to minimizing this “time consuming” processes because it reduces unnecessary strain on network performance, thus optimizing the system.⁵⁵¹ Further, for Steen, a POSITA would have understood that

⁵⁴⁶ Skagerwall at 8:15-18.

⁵⁴⁷ *See, e.g., id.* at 3:45-67.

⁵⁴⁸ *Id.* at 4:2-5:10.

⁵⁴⁹ *Id.* at 2:8-12.

⁵⁵⁰ Steen at Abstract; Wolff at 1:41-54; Skagerwall at Abstract.

⁵⁵¹ Skagerwall at 2:8-12.

redirecting requests would help avoid failure and provide “fault tolerance.”⁵⁵² Steen explains that fault tolerance for failures is critical for location services: “The location service should be resilient to node and link failures, and should continue to operate in the presence of network partitions.”⁵⁵³

Steen, Skagerwall and Wolff describe the challenges of scalability-expanding the networks and maintaining performance.⁵⁵⁴ For instance, Steen explains its system is scalable to service clients and objects located around the world:

The service should allow clients and objects to be located anywhere in the world, and be able to support a huge number of objects. We anticipate that eventually, one billion users with 1000 objects each will be registered with the location service, adding up to 1012 objects.⁵⁵⁵

Similarly, Wolff explains that such swift growth risks overloading a system, making a server a “bottleneck” in the system.⁵⁵⁶ And Skagerwall echoes this same bottleneck concern for scaling up a system, emphasizing the need to reduce processing load whenever possible on a variety of servers.⁵⁵⁷

Thus, a POSITA implementing Steen’s system would have been motivated to incorporate teachings like Skagerwall’s scalable system organized with hashing and Wolff’s teachings toward optimizing performance by reducing processing load on network servers.⁵⁵⁸ As Mr. Greene explains, Wolff’s redirection command offers one such teaching toward reducing the processing

⁵⁵² Steen at 105; Greene Decl. ¶ 375.

⁵⁵³ Steen at 105.

⁵⁵⁴ *E.g.*, Skagerwall at 1:59-62; Steen at 105.

⁵⁵⁵ Steen at 105.

⁵⁵⁶ Wolff at 2:1-2.

⁵⁵⁷ Skagerwall at 5:31-36, 18:54-58.

⁵⁵⁸ *See* Greene Decl. ¶ 378.

load on Steen's or the Steen-Skagerwall directory servers by moving the processing to the access server.⁵⁵⁹ Thus, a POSITA would have identified Wolff's redirection command as a technique for optimizing Steen's directory servers to use processing power to resolve requests intended for other directory servers. Accordingly, a POSITA would have been motivated to modify Steen's or Steen-Skagerwall's directory servers to use Wolff's redirection command to identify another directory server having the requested data then sending a redirect message containing the location of that other server.⁵⁶⁰

Further, a POSITA would have had a reasonable expectation of success combining Steen and Skagerwall's systems with Wolff's redirection command messages.⁵⁶¹ As Mr. Greene describes, a POSITA would have expected that any server with a processor and the necessary information about the way location information is organized on the directory server network, i.e., the basic hashing method described in Steen or the more complex versions taught by Steen and Skagerwall. Steen makes it clear that any server with processing capability and the necessary information can identify where the requested location information is stored on the partitioned servers because the mapping is stable:

Partitioning and mapping information is assumed to be relatively stable so that it can be easily cached by subnodes. This assumption is necessary to avoid having to query the management service each time a subnode needs to communicate with its parent or children, which would turn the management service into a potential communication bottleneck.⁵⁶²

⁵⁵⁹ *See id.*

⁵⁶⁰ *See id.*

⁵⁶¹ *See id.* ¶ 379.

⁵⁶² Steen at 108.

As Mr. Greene explains, maintaining stable mapping allows more servers to calculate locations with the necessary information. And stable mapping would support the ability to redirect a client with confidence because the location would be unlikely to change before the client resolves its request. Further, even if a POSITA would seek a more robust redirecting technique than Steen's, Skagerwall teaches completing the identification step by any server in the kernel.⁵⁶³ Accordingly, by providing the client with the information about how the location information is partitioned, a POSITA would have a reasonable expectation of success in having the client use the information to request the desired location information from the server known to contain the location information.⁵⁶⁴

1. Steen Alone or in View of Skagerwall and/or Wolff Renders Obvious Independent Claims 17 and 18.

During the original examination, the Examiner indicated that the prior art of record didn't teach or suggest the "hash function" or "redirect" limitations in elements of claims 17, 18 and 24.⁵⁶⁵ Steen in view of Skagerwall or Wolff renders obvious these limitations because Steen discloses partitioning a location server into a flat network and using a hash function to organize location information, Skagerwall discloses the organization of location information across a flat network of directory servers with hash functions and tables, and Wolff discloses the use of redirection to prevent network failure.

- a) *Claim elements 17.pre and 18.pre: "A system for retrieving data location information for data stored in a distributed network, the system comprising:"*

⁵⁶³ See Greene Decl. ¶ 379; Steen at 108; Skagerwall at 4:17-26.

⁵⁶⁴ See Greene Decl. ¶ 379.

⁵⁶⁵ Ex. 9 ('640 History) at 442, 460.

Claim elements 17.pre and 18.pre are the same. To the extent they are limiting, Steen discloses these elements. For instance, Steen describes a location service in a system for “locating mobile objects in a worldwide system”:

Locating mobile objects in a worldwide system requires a scalable location service. An object can be a telephone or a note-book computer, but also a software or data object, such as a file or an electronic document. Our service strictly separates an object’s name from the addresses where it can be contacted. This is done by introducing a location-independent object handle. An object’s name is bound to its unique object handle, which, in turn, is mapped to the addresses where the object can be contacted. To locate an object, we need only its object handle. We present a scalable location service based on a worldwide distributed search tree that adapts dynamically to an object’s migration pattern to optimize lookups and updates.⁵⁶⁶

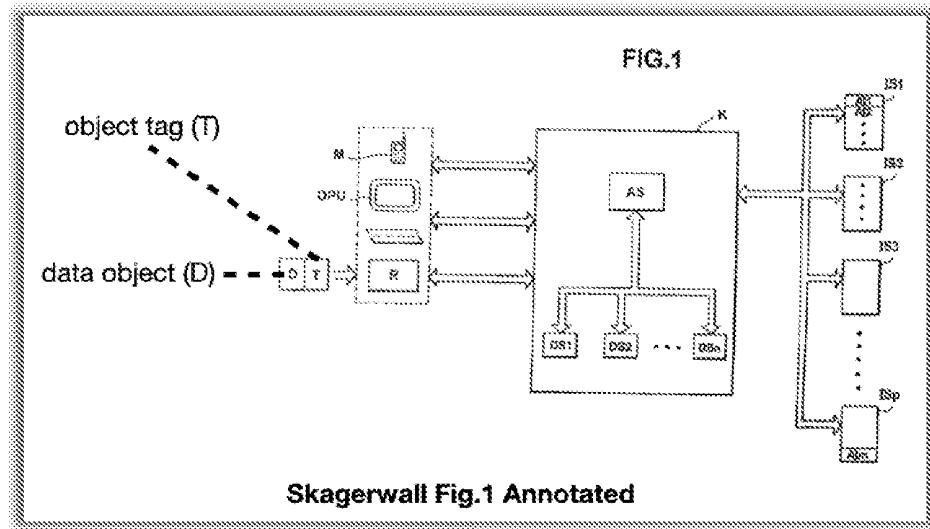
Accordingly, Steen discloses or render obvious the claim preambles.⁵⁶⁷

Further, as described above for SNQ 1, Skagerwall’s teachings render obvious these preambles. Skagerwall provides a plurality of directory servers to manage this process. Skagerwall teaches an “object tag” associated with each data object and used to identify the object on the network.⁵⁶⁸ As shown below in annotated Figure 1, Skagerwall’s object tag and object data are associated with each other upon entry to the system.

⁵⁶⁶ Steen at Abstract.

⁵⁶⁷ See Greene Decl. ¶ 382.

⁵⁶⁸ Skagerwall at 4:17-26.



Skagerwall describes using the object tag to identify and organize the data objects across the system: “The object data is distributed according to a defined scheme onto the plurality of directory servers in order to be able to identify the directory server storing object data associated with a particular read object tag.”⁵⁶⁹ As Mr. Greene explains, a POSITA would have understood the object tag to be the claimed “identifier string” because each object tag is a unique encoding identifying a data entity, and is associated with an application server address (i.e., a location string).⁵⁷⁰

Further, a POSITA would have been motivated to look to Skagerwall’s system for locating data objects based on Steen’s shared goal of locating files on a distributed system. Because Skagerwall and Steen describe similar technologies, a POSITA would also have a reasonable expectation that Skagerwall’s teachings could be incorporated and applied to Steen’s system for the predictable result of effectively locating data on a distributed database.⁵⁷¹

⁵⁶⁹ *Id.* at Abstract.

⁵⁷⁰ See Greene Decl. ¶ 384.

⁵⁷¹ See *id.* ¶ 385.

Thus, based on the above teachings, Steen discloses or renders obvious in view of Skagerwall the preambles of claims 17 and 18.

b) *Claim elements 17.a and 18.a: “a plurality of data repositories/a data repository configured to store data,” / “a data repository configured to store data,”*

Claim elements 17.a and 18.a are substantially similar. Steen discloses these claim elements or render them obvious alone or in view of Skagerwall. For example, Steen provides that a location service to manage stored objects, including data files:

Locating mobile objects in a worldwide system requires a scalable location service.

An object can be a telephone or a note-book computer, but also a software or data object, such as a file or an electronic document.⁵⁷²

As Mr. Greene explains, to the extent Patent Owner argues that Steen doesn't disclose a data repository, it would have been obvious to a POSITA that Steen's system stores the data object somewhere. Further, as Mr. Greene describes, it would have been obvious that Steen teaches or renders obvious a plurality of data repositories, because if Steen's system had only one repository, the system would have no need for location services because managing the locations would be simple.⁵⁷³ Further, even if it did not require a plurality of data repositories to begin with, Steen describes its location service as “scalable,” e.g., expanding.⁵⁷⁴ Thus, even if there were only one data repository originally, Steen provides for additional data repositories to be added moving forward upon scaling the system.

Further, as described above for SNQ 1, Skagerwall's teachings render obvious these claim elements. Steen explains the need for data repositories to handle a huge number of objects

⁵⁷² Steen at Abstract.

⁵⁷³ See Greene Decl. ¶ 387.

⁵⁷⁴ Steen at Abstract.

all over the world.⁵⁷⁵ Steen and Skagerwall both describe the challenges of scalability-expanding the networks and maintaining performance.⁵⁷⁶ For example, Steen states that its system is scalable to service clients and objects located around the world:

The service should allow clients and objects to be located anywhere in the world, and be able to support a huge number of objects. We anticipate that eventually, one billion users with 1000 objects each will be registered with the location service, adding up to 1012 objects.⁵⁷⁷

Similarly, Skagerwall addresses concerns for scaling up a system, emphasizing the need to reduce the load on server whenever possible by the work it over a plurality of servers.⁵⁷⁸ Because Skagerwall and Steen describe similar distributed databases, a POSITA would be motivated to have Steen's system include a plurality of data repositories configured to store data. A POSITA also would have had a reasonable expectation of success that Skagerwall's teachings could be incorporated and applied to Steen's system for the predictable result of effectively storing data on a plurality of data repositories.⁵⁷⁹

Accordingly, Steen and Skagerwall disclose or render obvious in combination claim elements 17.a and 18.a.”

- c) *Claim elements 17.b and 18.b: “wherein the data is associated with a respective identifier string in each data repository;”/“ wherein the data is associated with an identifier string;”*

Claim elements 17.b and 18.b are substantially similar. Steen discloses these claim elements or renders them obvious alone or in view of Skagerwall. For example, Steen teaches the

⁵⁷⁵ *Id.* at 105.

⁵⁷⁶ *E.g.*, Skagerwall at 1:59-62; Steen at 105.

⁵⁷⁷ Steen at 105.

⁵⁷⁸ Skagerwall at 5:31-36, 18:54-58.

⁵⁷⁹ Greene Decl. ¶ 389.

use of an “object name” that is “well known to its users.”⁵⁸⁰ Steen explains that, in distributed systems, users need names that are separated from locations: “What is needed is a naming facility that hides all aspects of an object’s location. Users should not be concerned where an object is located or whether it can move.”⁵⁸¹ Steen explains that the user uses this “object name” to identify the desired service and obtain the object handle: “A client has the service’s name resolved to the object handle.”⁵⁸² Steen explains that this object-name approach simplifies the resolution process for users: “Our approach allows an object to update its contact addresses independent of how users have named the object.”⁵⁸³

Further, as described above for SNQ 1, Skagerwall’s teachings render obvious these claim elements. Skagerwall describes the use of identifiers to manage data in a system. Skagerwall teaches an “object tag” associated with each data object and used to identify the object on the network. As shown below in annotated Figure 1, Skagerwall’s object tag and object data are associated with each other upon entry to the system.⁵⁸⁴

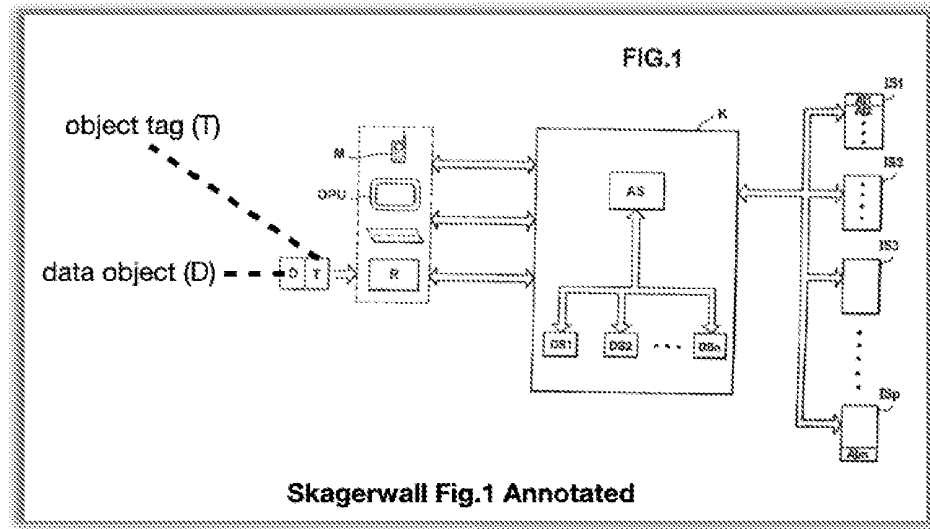
⁵⁸⁰ Steen at 104.

⁵⁸¹ *Id.* at 105.

⁵⁸² *Id.*

⁵⁸³ *Id.*

⁵⁸⁴ Skagerwall at 7:53-61.



Skagerwall describes using the object tag to identify and organize the data objects across the system: “The object data is distributed according to a defined scheme onto the plurality of directory servers in order to be able to identify the directory server storing object data associated with a particular read object tag.”⁵⁸⁵ As Mr. Greene explains, a POSITA would have understood the object tag to be the claimed “identifier string” because each object tag is a unique encoding identifying a data entity, and is associated with an application server address (i.e., a location string).⁵⁸⁶

Because Skagerwall and Steen describe similar distributed databases, a POSITA would have been motivated to have Steen’s data be associated with a respective identifier string in each data repository. A POSITA also would have had a reasonable expectation that Skagerwall’s teachings could be successfully incorporated and applied to Steen’s system for the predictable result of effectively storing data on a plurality of data repositories.⁵⁸⁷ Associating data with an

⁵⁸⁵ *Id.* at Abstract.

⁵⁸⁶ *See* Greene Decl. ¶ 393.

⁵⁸⁷ *Id.*

identifier was a routine practice within the field so a POSITA would have expected that Steen's system could readily incorporate such an identifier, like Skagerwall's object tag.⁵⁸⁸ And a POSITA would have understood the similarities between Steen's and Skagerwall's distributed server directory systems and have reasonably expected that those systems operate similarly enough to incorporate such an adjustment from Steen to Skagerwall's system.⁵⁸⁹

Accordingly, Steen and Skagerwall disclose or render obvious in combination claim elements 17.b's and 18.b's location server storing a plurality of identifier strings each identifying a data entity..

d) *Claim elements 17.c and 18.d: "a data location [sewer/server] network [having/comprising] a plurality of data location servers,"*

Claim elements 17.pre and 18.pre are substantially similar. Steen discloses or alone renders obvious these claims elements. For instance, Steen describes a two-level system for managing and locating an object's contact addresses. In the first level, Steen describes using a directory system for resolving an object name to an "object handle." The object handle includes location information for a contact record.⁵⁹⁰ Thus, as Mr. Greene explains, in the first level, Steen describes a data location server network that resolves "object names" (i.e., the claimed "identifiers") to object handles (i.e., the claimed "location information").⁵⁹¹ Accordingly, Steen's first level constitutes a data location server network comprising a plurality of data location servers. Then, in the second level, Steen teaches that the user uses the location information (i.e., object handle) to obtain object addresses from the contact record.

⁵⁸⁸ See *id.* ¶¶ 393-94.

⁵⁸⁹ See *id.*

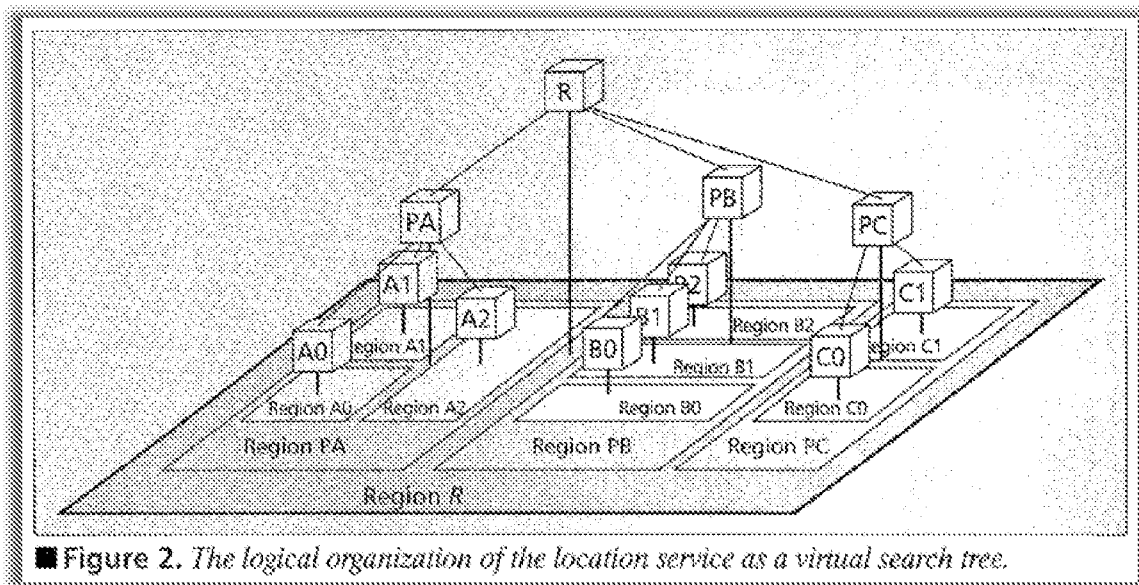
⁵⁹⁰ Steen at 105.

⁵⁹¹ Greene Decl. ¶ 396.

Further, Steen teaches using a plurality of servers to provide services. Steen provides the exemplary description of *anycasting* to show how multiple servers provide services like the disclosed location service.

In distributed systems, this way of locating a service is also known as anycasting: a client requires a particular service, but is really not interested in which server will handle the request. Using our approach, the service is assigned a unique object handle, and each server registers its network address under that object handle. A client has the service's name resolved to the object handle, which is then subsequently resolved to the address of any server that can handle the request.⁵⁹²

To manage the first level of the system, Steen discloses the use of “directory nodes,” which correspond to the claimed “location servers.” As shown below in Figure 2, Steen's system includes a directory node Dir(R), for resolving object names to object handles, e.g., location information for addresses. Steen describes that, in systems with multiple regions, there may be multiple directory nodes, e.g., location servers.⁵⁹³



■ Figure 2. The logical organization of the location service as a virtual search tree.

⁵⁹² Steen at 105.

⁵⁹³ *Id.* at 105-06; Greene Decl. ¶ 398.

In our model for tracking objects, we assume hierarchical decomposition of a (worldwide) network into regions. This decomposition is relevant only to the location service. With each region we associate a directory node capable of storing addresses that lie within that region. This leads to a logical tree-based organization, as shown in Fig. 2.⁵⁹⁴

Steen teaches that the system assumes addresses be location-dependent, i.e., the region in which an address lies is encoded in the address itself:

The location service normally stores new addresses at the leaf node representing the region in which the address lies. For each new object, it constructs a path of forwarding pointers from the root to each leaf node where an address is stored. Addresses and forwarding pointers are stored in contact records. An implication of this design is that in the worst case it is always possible to locate every object by following the chain of pointers from the root node. In practice, we can do much better than this, as described later.⁵⁹⁵

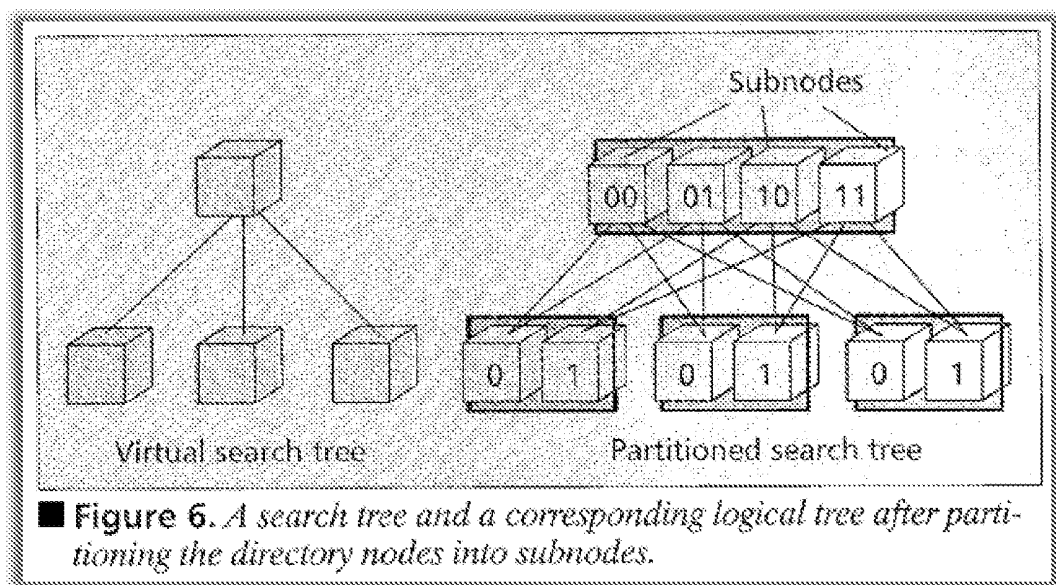
Further, Steen also describes that $\text{dir}(R)$ may be partitioned into multiple subnodes, as shown below in Figure 6:

In particular, higher-level directory nodes not only have to handle a relatively large number of requests, they also have enormous storage demands. Our solution is to partition a directory node into one or more directory subnodes such that each subnode is responsible for a subset of the records originally stored at the directory node.⁵⁹⁶

⁵⁹⁴ Steen at 105.

⁵⁹⁵ *Id.* at 105.

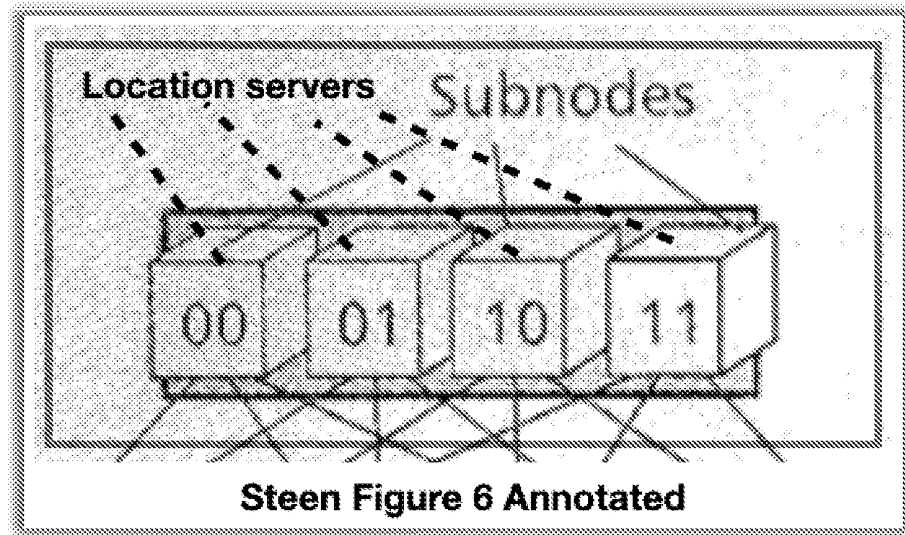
⁵⁹⁶ *Id.* at 107-08.



For a partitioned $\text{dir}(\mathbf{R})$, Steen teaches its system configures each subnode to function as its own server and contains its own subset of location information.⁵⁹⁷ For those reasons, as Mr. Greene explains, a POSITA would understand that each subnode is a location server containing location information.⁵⁹⁸

⁵⁹⁷ *Id.*

⁵⁹⁸ Greene Decl. ¶ 401.



Thus, Steen discloses a system of flat location servers which each contain a portion of a total dataset of location information.

Further, and as described for SNQ 1, Skagerwall's teachings render obvious these claim elements. Skagerwall teaches a plurality of data location servers storing object data according to object tags. Skagerwall explains that each object tag (i.e., identifier string) is associated with the application servers address (i.e., location strings) for the relevant data object:

As mentioned above, object data related to an object tag preferably includes application identifiers AII-Aim of application data packets related to the object as well as information about storage locations of the application data packets, e.g., the addresses of application servers IS; IS1-Isp storing the application data packets.⁵⁹⁹

Skagerwall addresses concerns for scaling up a system, emphasizing the need to use multiple location servers.⁶⁰⁰ Because Skagerwall and Steen describe similar location management processes on similar distributed databases, a POSITA would have been motivated to have Steen's system include a plurality of data location servers. A POSITA also would have had a

⁵⁹⁹ Skagerwall at 8:15-27.

⁶⁰⁰ *Id.* at 5:31-36, 18:54-58.

reasonable expectation that Skagerwall's teachings could be incorporated and applied to Steen's system for the predictable result of effectively storing data on a plurality of data repositories.⁶⁰¹

Thus, Steen and Skagerwall disclose or render obvious claim elements 17.c's and 18.d's location server network comprising a plurality of data location servers..

- e) *Claim elements 17.d and 18.e: "each of the plurality of data location servers containing location strings associated with respective identifier strings and" / "at least one of the plurality of data location servers containing location information associated with the identifier string,"*

Claim elements 17.d and 18.e are substantially similar. Steen discloses or alone renders obvious these claim elements. For example, Steen provides that its location service resolves names to "object tags" containing location information. Steen teaches that, in the lower level, the system uses data files called "contact records": "Addresses and forwarding pointers are *stored in contact records*."⁶⁰² Thus, Steen's contact record constitutes stored data associated with an object. In the lower level, the system resolves object handles to contact records. Steen teaches that the object handle doesn't rely on the addresses contained in the contact record, describing it as "location independent."⁶⁰³ Rather, Steen explains that the system uses that object handle for locating stored "contact record" data: "[The] object handle may contain information that can be *used to assist in locating the object*."⁶⁰⁴ For example, Steen describes that the object handle can include subnode identification for a given contact record: "As an example, we can use the first n bits of an object's handle to identify the subnode responsible for that object."⁶⁰⁵

⁶⁰¹ Greene Decl. ¶ 403.

⁶⁰² Steen at 105.

⁶⁰³ *Id.* at Abstract, 105.

⁶⁰⁴ *Id.* at 105.

⁶⁰⁵ *Id.* at 108.

Steen illustrates the role of the object handle as location information by comparing it to a “lifetime telephone number”:

As an illustration, consider an officeless company whose employees are located across the country, normally working at home or visiting customers. Using our approach, we assign a lifetime location-independent telephone number to the company. This number corresponds to an object handle. A naming or directory service would maintain a mapping between the company’s name and its lifetime telephone number.⁶⁰⁶

Accordingly, Steen’s object handle contains location information for finding the contact record. Thus, a POSITA would understand Steen’s object handle to be the claimed location information.⁶⁰⁷

Further, the claim construction that Patent Owner proposed and the district court adopted provides that location information means “information pertaining to one or more locations of data and/or *the identities of one or more location servers*.”⁶⁰⁸ To the extent Patent Owner argues that Steen’s “object handle” doesn’t contain location information for the final data entity, the object handle provides information that identifies which lower level location servers contain that information.⁶⁰⁹ Thus, to the extent Patent Owner asserts that the object handle isn’t “information pertaining to the location of data,” it’s location information because it pertains to “the identities of one or more location servers.” And a POSITA would understand that the lower level of Steen contains directory servers that contain addresses for the stored data.⁶¹⁰ Accordingly, Steen’s

⁶⁰⁶ *Id.* at 105.

⁶⁰⁷ Greene Decl. ¶ 407.

⁶⁰⁸ Joint Claim Construction Chart at 4-6.

⁶⁰⁹ *See* Greene Decl. ¶ 407.

⁶¹⁰ *See id.*

lower level has “one or more location servers” and the object handle identifies which lower-level servers contains the desired object location information.⁶¹¹

Accordingly, Steen discloses or renders obvious claim elements 17.d’s and 18.e’s at least one of the plurality of data location servers includes location information associated with the identifier string.

f) *Claim element 18.c: “a client responsive to a data query to query a data location server for location information associated with the identifier string”*

Steen discloses this claim element or renders it obvious alone or in view of Skagerwall. Steen provides that its system receives requests from users and that the servers handle those requests:

In distributed systems, this way of locating a service is also known as anycasting: a client requires a particular service, but is really not interested in which server will handle the request. Using our approach, the service is assigned a unique object handle, and each server registers its network address under that object handle. A client has the service’s name resolved to the object handle, which is then subsequently resolved to the address of any server that can handle the request.⁶¹²

As Mr. Greene explains, a POSITA would have appreciated that, when a user needs information, the user makes a data query on its device in Steen’s system.⁶¹³ A POSITA would further understand that the device would go to the directory server for location information, i.e., query, a location server.⁶¹⁴

⁶¹¹ *See id.*

⁶¹² Steen at 105.

⁶¹³ *See* Greene Decl. ¶ 410.

⁶¹⁴ Steen at 105-06; Greene Decl. ¶ 410.

Further, and as described for SNQ 1, Skagerwall's teachings render obvious this claim element. Skagerwall discloses a plurality of directory servers (i.e., data location servers) storing application server addresses (i.e., location string/location information) associated with object tags (i.e. identifier strings) for data entities. Skagerwall explains that a user device, such as a mobile phone, will request a data entity from the kernel using that entity's object tag (i.e., identifier string).⁶¹⁵ Skagerwall teaches that the access server responds to this request by querying a directory server (i.e., location server) for the requested data entity's location where that query includes the object tag (i.e., identifier string):

After receiving a request for information including the object tag T, the access server AS; AS1-Asr performs operations to identify the directory server DS; DS1-DSn storing information concerning application data packets related to the object. This preferably includes sending the request including the object tag T; T1-Tw to one of the directory servers DS1; DS2; . . . ; DSn...⁶¹⁶

As Mr. Greene explains, Skagerwall's interaction between the user device, access server, and directory servers is a form of "client-server" interaction, where the access server (AS) is the client to the directory servers by requesting the data object's address on the application server (IS).⁶¹⁷ As a result, a POSITA would have understood that Skagerwall's access server corresponds to the claimed "client" because it is a network-attached component that uses location request messages to initiate the lookup of location mappings from the directory servers (i.e., location servers).⁶¹⁸

⁶¹⁵ Skagerwall at 9:18-28.

⁶¹⁶ *Id.*

⁶¹⁷ See Greene Decl. ¶ 412.

⁶¹⁸ See *id.*

As Mr. Greene explains, both Steen and Skagerwall describe the same type of interaction, i.e., a “client-server” interaction. To any extent Steen alone doesn’t teach a client making a query to a data location server in response to a data query, it would have been obvious to a POSITA that in order to provide requested data, Steen’s client would need to locate it and would make a query to the location service in response.⁶¹⁹

Based on the above teachings, Steen and Skagerwall disclose or render obvious claim element 18.c’s storing and retrieving data location information for a plurality of data entities in the data location server network.

g) *Claim elements 17.e and 18.f: “each of the plurality of data location servers having computer executable code configured to execute the following steps:” / “wherein each of the plurality of data location servers comprises computer executable code configured to execute the following steps in response to receiving a data location request from the client”*

Claim elements 17. e and 18.f are substantially similar. Steen discloses these claim elements or renders them obvious alone or in view of Skagerwall. As Mr. Greene explains, any component of a network must have executable code to function properly, including data location servers. As a result, it would have been understood by a POSITA that each of Steen’s data location server in the data location server network has executable code configured to executes certain steps to respond to a request.⁶²⁰

To the extent Patent Owner asserts that Steen doesn’t disclose or render obvious this element alone, and as described for SNQ 1, Skagerwall teaches that its directory servers have a

⁶¹⁹ See *id.* ¶ 413.

⁶²⁰ *Id.* ¶ 415.

means to execute a variety of processes including the storing of location data, receiving requests, identifying correct directory servers, and retrieving stored data:

[Directory servers]...including memory means (TM; AM) for storing in a distributed data base object data associating each of the objects with at least one of the application data packets; means for receiving the read object tag (T) from the reading means I and means for identifying the directory server for storing object data related to the read object tag and retrieving the object data; and means for retrieving at least one of the application data packets based on the object data or transmitting at least part of the object data to a receiving unit (M, DPU).⁶²¹

As Mr. Greene explains, a POSITA would have understood that, in order for the directory servers to do any of these tasks, there must be executable code for how to complete the processes.⁶²² Thus, a POSITA would have understood that each directory server (i.e., location server) includes computer executable code for performing Skagerwall's disclosed tasks.⁶²³

Similarly, with respect to claim element 18.f's requirement for responding to a client request, Skagerwall's teachings render this obvious. Skagerwall explains that the access server (i.e., client) sends a request to the directory serve and that the directory server will respond "in turn" to the request with specific steps:

This preferably includes sending the request including the object tag T; T1-Tw to one of the directory servers DS1; DS2; . . . ; DS_n, which in turn, using the hash-function and hashtable H, determines the directory server for storing the object tag T; T1-Tw and forwards the request to that particular directory server. However, it

⁶²¹ Skagerwall at 2:60-3:2.

⁶²² See Greene Decl. ¶ 417.

⁶²³ See *id.*

is also possible, that the access server AS; AS1-ASr or other components of the system kernel K perform steps for identifying the correct directory server.⁶²⁴

Thus, based on Skagerwall's teachings, a POSITA would have been motivated to have each of Steen's location server include computer executable code configured to execute location-related steps in response to receiving a data location request from the client.⁶²⁵

Steen and Skagerwall also each disclose or render obvious claim 17.e's and 18.f's "transport protocol" limitation. As Mr. Greene explains, requests like these would conform to network protocols. Network protocols are necessary to enable different components to send and receive messages to execute basic network functions, including transporting data and information. Thus, as Mr. Greene explains, a POSITA would understand Steen and Skagerwall's networks to include a transport protocol.⁶²⁶ Further, Skagerwall specifically teaches the use of standard protocols with its system:

It is further possible, that the system and method according to the invention uses standard and/or amended protocols of a known datacom- or telecom network for retrieving information associated with an object.⁶²⁷

As Mr. Greene attests, a POSITA would have understood a transport protocol to be one of these standard protocols.⁶²⁸ As he further explains, such transport protocols were well-known, routine, and well-understood to be used for sending messages and data over a network. Thus, to any extent that Skagerwall doesn't disclose the claimed transport protocol, it would have been obvious to a POSITA at this time to use one.

⁶²⁴ Skagerwall at 9:18-28.

⁶²⁵ See Greene Decl. ¶ 419.

⁶²⁶ See *id.* ¶ 420.

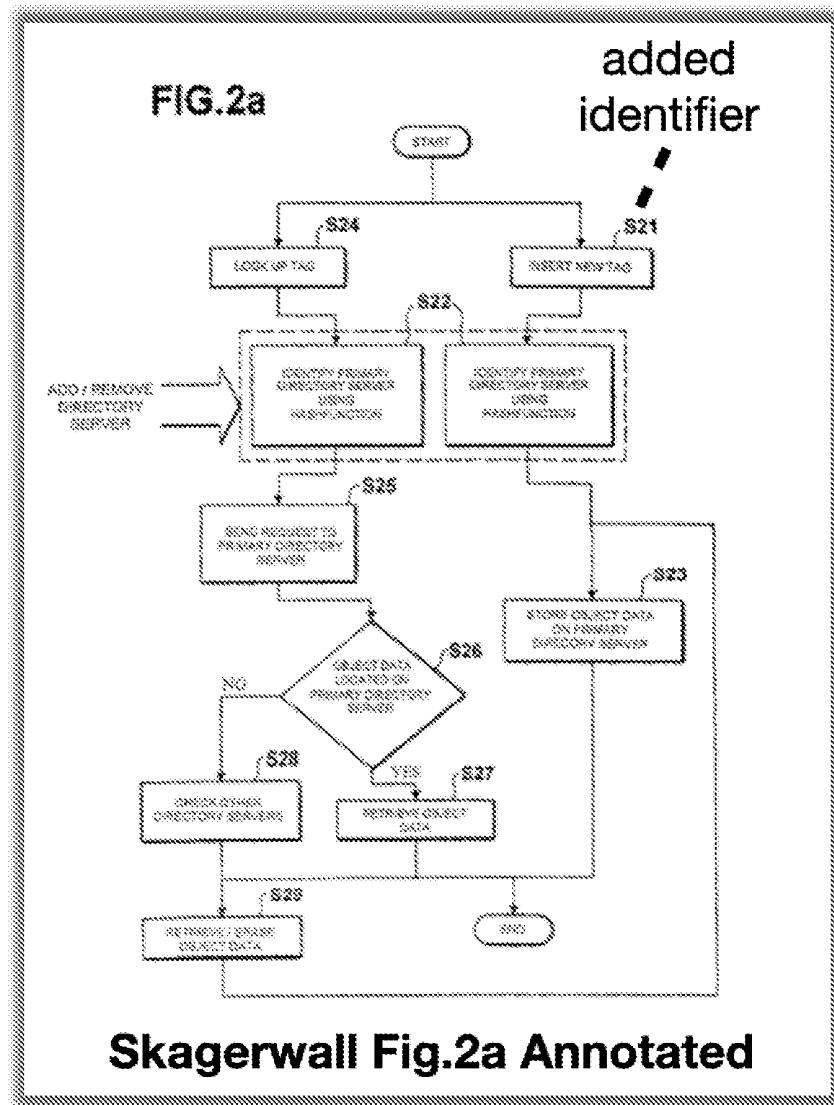
⁶²⁷ OracleNamesAdminGuide at 43-44. Skagerwall, 9:59-62.

⁶²⁸ See Greene Decl. ¶ 421.

Steen and Skagerwall further each disclose or render obvious claim element 18.c's "location addition request" limitation. Steen specifically teaches the addition of new data locations into the lower level of its system. "The location service normally stores new addresses at the leaf node representing the region in which the address lies."⁶²⁹ To any extent it's not disclosed that data may be added to the top level, it would be obvious to a POSITA that in order to add data to the bottom layer, a corresponding addition must be made to the top layer. As a result, it would have been obvious to a POSITA that a location addition request be made with location information (i.e., object handle) and an associated identifier (i.e., object name).

To any extent Steen doesn't alone render obvious the location addition request, it would be obvious in view of Skagerwall. In Fig. 2a, Skagerwall teaches the addition of a new object tag (i.e., identifier) into the network of directory servers in S21.

⁶²⁹ Steen at 105.



Skagerwall teaches that in “step S21 the system is advised to insert new object data into its database distributed on the plurality of the directory servers.”⁶³⁰ As Mr. Greene explains, a POSITA would have understood that the system would be advised with a request, such as the claimed location addition request.⁶³¹ Accordingly, a POSITA would understand the new object data (i.e., new location information) from S21 would be included in a request with the object tag,

⁶³⁰ Skagerwall at 12:5-8.

⁶³¹ See Greene Decl. ¶ 424.

(i.e., identifier) shown in S21 of Fig. 2a. Thus, Skagerwall teaches a location addition request including location information and an associated identifier to be added to the location server network.

As Mr. Greene affirms, a POSITA would have been motivated to incorporate the location addition request from Skagerwall into Steen's system to allow Steen's system to add new data to its directory system.⁶³² A POSITA would have understood the benefits of this capability to add new data into the system and help users retrieve it easily with the directory system. This would have supported Steen goals for a scalable system. As Mr. Greene explains, a POSITA would have understood a scalable system would grow with the addition of new data. And, as a result, providing a way to add new location information to the system would have enabled Steen's top level to scale up.⁶³³ Thus, a POSITA would have been motivated to incorporate Skagerwall's ability to add location information into Steen's system.

Accordingly, Steen and Skagerwall disclose or render obvious in combination claim elements 17.e's and 18.f's data location requests with identifiers to location servers storing associated location information.

- h) *Claim element 18.g: "if the data location server contains the location string associated with the identification string provided in the data location request, the data location server transmits location information for use by the client to calculate a location of the data associated with the identification string"*⁶³⁴;

⁶³² See *id.* ¶ 425.

⁶³³ See *id.* ¶ 425.

⁶³⁴ While unclear, it's assumed that the identification string is the same as the identifier string in this analysis.

Steen discloses this claim element or renders it obvious alone or in view of Skagerwall. For example, as described for element 17.pre, Steen provides that a location service to manage stored objects, including data files. As Mr. Greene describes, a POSITA would have appreciated that any data location directory server returns the requested information if it was contained on the requested server. A POSITA would have understood this to be the primary function of a location server.⁶³⁵

To the extent Patent Owner asserts that Steen doesn't disclose or render obvious this element, as described for SNQ 1, Skagerwall explains designing its system so that any component can identify whether a given directory server contains the necessary location information in response to a data inquiry, but provides that the preferred component is a directory server:

[I]t is preferred, that one of the directory servers DS1; DS2; . . . ; DS_n identifies the directory servers storing information related to a requested object tag. However, other components of the system kernel K may serve as means for identifying the correct directory servers.⁶³⁶

Further, Skagerwall describes tasking a "primary" directory server with storing the data for a given object tag. But Skagerwall teaches the benefits of changing the primary directory server and using a hash function to identify each data's locations when adding and removing directory servers since the location in the system where the data is stored can change:

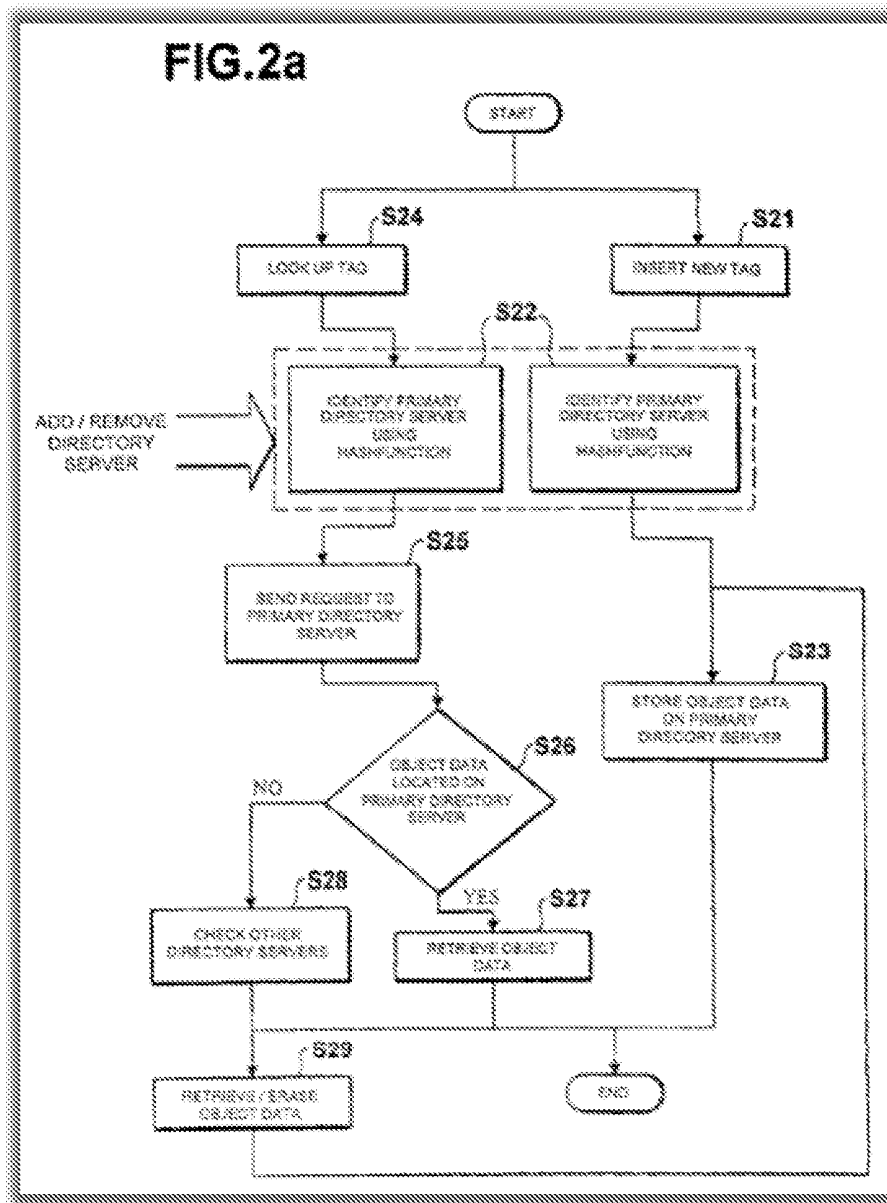
However, in the case the number of available directory servers changed, previously stored object data was mapped onto the plurality of directory servers using a hash-function based on the previous number of directory servers, whereas a target directory server is now identified based on a new number of directory servers. This will

⁶³⁵ See Greene Decl. ¶ 427.

⁶³⁶ Skagerwall at 10:21-26.

sometimes not lead to the directory server actually storing object data related to the desired object.⁶³⁷

As shown below in Figure 2a, Skagerwall describes a flow chart to identify what to do when the requested data isn't found where it should be according to the object tag.⁶³⁸



⁶³⁷ *Id.* at 11:37-45.

⁶³⁸ *Id.* at 11:27-13:15.

As shown in Figure 2a, Skagerwall's step S22 teaches that a directory server receives a request for data location information associated with an object tag and looks it up to find the primary server.⁶³⁹ As Mr. Greene explains, a POSITA would have understood that the system performs these flowchart steps taken in the course of receiving a data request from an access server (i.e., client).⁶⁴⁰ As Mr. Greene explains, Skagerwall's step S25 ("Send Request To Primary Directory Server") is only necessary if the request isn't already at the primary directory server.⁶⁴¹ As a result, if the directory server identifies itself as the primary server, the next relevant step is retrieving the requested data at step S27 ("Retrieve Object Data").⁶⁴²

Further, Skagerwall explains that, once the primary directory server has retrieved the address for the requested object tag, the directory server returns that address to the access server (i.e., client):

The obtained addresses IS; IS1-ISp and optionally other information can be returned to the access server AS; AS1-ASr, which in the following retrieves the specified application data packets from the identified application servers or transmits the addresses IS; IS1-ISp and possibly other information directly to the user, i.e., the mobile telephone M or the data processing unit DPU.⁶⁴³

As Mr. Greene explains, a POSITA would have understood that Skagerwall's "retriev[ing] the specified application data packets from the identified application servers" includes

⁶³⁹ Skagerwall at 12:9-53.

⁶⁴⁰ See Greene Decl. ¶ 431.

⁶⁴¹ See *id.*

⁶⁴² See *id.*

⁶⁴³ Skagerwall at 9:42-47.

the claim element 18.f's "use by the client to calculate a location of the data associated with the identification string."⁶⁴⁴

Thus, based on Skagerwall's teachings, a POSITA would have found it obvious to configure each of Steen's location servers such that, if the location server contains the location string associated with the identification string provided in the data location request, the location server transmits location information for use by the client to calculate a location of the data associated with the identification string. A POSITA would have been motivated to do so because Steen specifically directs the reader to use "more realistic hashing-based methods than explained here."⁶⁴⁵ A POSITA would recognize that Skagerwall specifically teaches how to use hashing to identify the location server containing the requested location information. A POSITA would be motivated to include this server identification in a redirect message because a failure to resolve a request causes inefficiency and poor performance.

Thus, motivation exists to provide as much information for resolving a request as possible to the client, even if it doesn't include the requested information itself and only a location of a server where it can be found.⁶⁴⁶ In fact, as Mr. Greene attests, successful distributed databases at the time implemented these very ideas from Steen and Skagerwall to perform all of the functions needed.⁶⁴⁷

Accordingly, Steen and Skagerwall disclose or render obvious in combination claim elements 18.g's responding to data location requests with the associated location information if contained by the location server.

⁶⁴⁴ See Greene Decl. ¶ 433.

⁶⁴⁵ Steen at 108.

⁶⁴⁶ See *id.*

⁶⁴⁷ See *id.*

- i) *Claim element 18.h: “if the data location server does not contain the location string associated with the identification string, the location server transmits a redirect message to the client, wherein the redirect message contains redirect information for use by the client to calculate a location of a different data location server in the plurality of data location servers, wherein the different data location server contains the location string.”*

Steen discloses this claim element or renders it obvious alone or in view of Skagerwall and/or Wolff. As described in SNQ 2, Steen discloses this claim element. For instance, Steen teaches hashing, specifically in connection with partitioned location servers. As Steen provides, this hashing ensures each subnode is aware, i.e., knows, the location of any requested location information regardless of which server it is contained on or has the information to calculate the location of the server containing the desired location information:

Because communication between directory nodes in the original search tree now takes place between their respective subnodes, each subnode should be aware of how the directory node with which it communicates is actually partitioned. This information is contained in a separate tree management service. This service also maintains the mapping of subnodes to physical nodes. Partitioning and mapping information is assumed to be relatively stable so that it can be easily cached by subnodes.⁶⁴⁸

Steen explains that this awareness minimizes the need to request partition information from other services to identify the location server storing the requested data. For instance, Steen explains that minimizing such requests improves system efficiency by decreasing the need for repeated requests: “This assumption is necessary to avoid having to query the management service each time a subnode needs to communicate with its parent or children, which would turn the management service into a potential communication bottleneck.”⁶⁴⁹

⁶⁴⁸ Steen at 108.

⁶⁴⁹ *Id.*

Further, as described for SNQ 1, Skagerwall's teachings render obvious this claim element. For each data request, Skagerwall teaches a directory server identifying if it is the primary directory server (i.e., location server) containing the application server address (i.e., location string) for the request. And Skagerwall provides instructions that, if the queried directory server doesn't contain the object tag for the requested data, the system progresses to flowchart step S28 where the request goes to other directory servers:

If in step S26 it has been determined that object data related to the object tag is not located at the identified primary directory server, in a step S28, the lookup request is forwarded to other directory servers. In case the number of directory servers has been increased recently, the lookup request may be forwarded to a directory server which is determined using the hashfunction based on a previous number of available directory servers. The same holds true, if the number of directory servers has been decreased recently. If none of the directory stores the desired object data, an error message is generated and the operation ends.⁶⁵⁰

As described for SNQ 1, Skagerwall's hashfunction teaches the requirements of this claim element. For instance, Skagerwall teaches that "[i]ntermediate steps are possible, where the look up request is relayed to further devices before it reaches the correct directory server, as outlined later."⁶⁵¹ As Mr. Greene explains, a POSITA would have understood Skagerwall's "intermediate steps" include a redirect message having information to calculate the location of the directory server containing the location string for the requested data.⁶⁵² This arises when system efficiency dictates lessening the processing burden of the directory servers.⁶⁵³

⁶⁵⁰ Skagerwall at 12:63-13:6.

⁶⁵¹ *Id.* at 9:39-41.

⁶⁵² *See* Greene Decl. ¶ 438.

⁶⁵³ *See id.*

Further, a POSITA would have found it obvious for Skagerwall's system to look to other directory servers to resolve a data request because failure to resolve this request is contrary to the system's objectives and desired efficiency.⁶⁵⁴ Skagerwall provides the exemplary forwarding step from the initial primary directory server to the identified directory server having the location of the requested data:

After identification of the correct directory server DS; DS1-DSn, the request, including the object tag T; T1-Tw, is forwarded to the identified directory server. The directory server, in a subsequent step, performs operations to determine all addresses of application servers IS; IS1-ISp storing application data packets associated with the read object tag T; T1-Tw and thus object D. Intermediate steps are possible, where the look up request is relayed to further devices before it reaches the correct directory server, as outlined later.⁶⁵⁵

Skagerwall teaches that exactly which kernel component performs the identification steps is a design choice by explaining that any component can complete this step: "[O]ther components of the system kernel K may serve as means for identifying the correct directory servers."⁶⁵⁶ Accordingly, as Mr. Greene describes, a POSITA would have understood that either the queried directory server (i.e., location server) or the access server (i.e., client) could perform the identification step depending on the needs of the system and its specific design.⁶⁵⁷

So, it would have been an obvious matter of design choice for Skagerwall's queried directory server to send to the access server (i.e., client) information for contacting another directory server having the location of the requested data rather than having the queried directory

⁶⁵⁴ See *id.* ¶ 439.

⁶⁵⁵ Skagerwall at 9:32-49.

⁶⁵⁶ *Id.* at 10:21-26.

⁶⁵⁷ See Greene Decl. ¶ 440.

server forward the request to the other directory server. For example, Mr. Greene explains that for a system where there is only one access server, it may have limited processing power so that any extra burden should be minimized. Mr. Greene explains that, in such a system, a POSITA would have understood the benefits of having the queried directory server find the correct directory server and obtain the desired location information, and return the addresses (i.e., location strings) to the access server.⁶⁵⁸ Conversely, as Mr. Greene describes, for a system with multiple access servers, the processing power may be more limited on the directory servers. In such a system, Mr. Greene explains, a POSITA would have understood that it's more efficient to minimize the processing requirements of the queried directory server.⁶⁵⁹ Thus, for such a system with multiple access servers, a POSITA would have been motivated to design Steen's system such that the queried directory server returns to the pertinent access server (i.e., client) the information to calculate the location of the primary directory server.⁶⁶⁰ As Mr. Greene explains, it would have been understood to a POSITA that the system would return such information in a "redirect message."⁶⁶¹

Thus, based on Steen's teachings, a POSITA would have found it obvious for Steen directory server, if it lacks an association to the requested location string, to send a redirect message to the client containing redirect information for the client to calculate the location of a different directory server having the request location string. A POSITA would have been motivated to do so because this would further the goals of the location service itself to respond to certain

⁶⁵⁸ See *id.* ¶441.

⁶⁵⁹ See *id.*

⁶⁶⁰ See *id.*

⁶⁶¹ See *id.*

data requests and to lessen the workload of the initially-requested directory servers because it no longer needs to perform forwarding functions to other servers containing the requested information.⁶⁶²

Further, if a directory can't resolve a client's information query, a POSITA would have been motivated to have that directory server transmit a redirect message so the client can calculate the location of another directory server housing the requested location string. This is because using such redirect messages was an established method to address unresolved location requests, as a POSITA would have been generally aware.⁶⁶³

Moreover, a POSITA would have been motivated to do so because providing a redirect message to calculate the location of a different directory server would improve the ability of Steen's system to locate the client's requested information. To do so, a POSITA would have designed Steen's system to detect and resolve problems without placing the burden on the user. The clear benefits for the user would have motivated a POSITA to modify Steen or Steen-Skagerwall in this way. Based on the above teachings, Steen and Skagerwall render this element obvious.⁶⁶⁴ Further, Wolff's teachings render obvious the claimed "redirect message" limitations. Wolff explains that servers will not always be able to fulfill each request received from a given client. For example, Wolff describes an "overloaded" server which has received too many requests and lacks the processing power to fulfill them all:

In operation at time T=0, normal client 100A is shown accessing memory resource 118 via path 70 through overloaded server 104. At the same time, aware client 102A is shown accessing memory resource 18, via path 74, through overloaded server

⁶⁶² See *id.* ¶ 442 .

⁶⁶³ See *id.* ¶ 443.

⁶⁶⁴ See *id.* ¶ 444.

104A. At time T=1, process 102P1, implemented on aware client 102A, detects the overload condition of server 104A, and accesses memory resource 118 via an alternate path 76 through server 106A. Thus, in this subsequent state, the load on server 104A is reduced and the access by aware client 102A to memory resource 118 is enhanced. Normal client 100A cannot initiate the processes 35 discussed above in connection with the aware client 102A and is unable to select an alternate path 72 to the underutilized server 106A.⁶⁶⁵

And Wolff teaches optimizing network performance by redirecting requests from an overloaded server, e.g., a server that can't satisfy the request, to another server that can satisfy the request with a "redirection command" from the overloaded server:

Client load rebalancing refers to the ability of a client, enabled with processes in accordance with the current invention, to re-map a path through a plurality of nodes to a resource. *The re-mapping may take place in response to a redirection command emanating from an overloaded node, e.g. server.* The embodiments disclosed allow more efficient, robust communication between a plurality of clients and a plurality of resources, via a plurality of nodes. Resources can include, but are not limited to, computers, memory devices, imaging devices, printers, and data sets. A data set can include a database or a file system.⁶⁶⁶

Accordingly, based on Wolff's teachings, a POSITA would have found it obvious to have Steen or Steen-Skagerwall's queried location server transmit a redirect message to the access server (i.e., client) if that location server doesn't contain the location string associated with the identification string wherein the redirect message has information for use by the access server to calculate a location of another directory server containing the location string.⁶⁶⁷ A POSITA would have been motivated to do so because this would optimize performance of Steen-

⁶⁶⁵ Wolff at 5:25-38.

⁶⁶⁶ *Id.* at 2:41-51.

⁶⁶⁷ *See* Greene Decl. ¶ 446.

Skagerwall's system by having the queried directory server send a redirect message to the client rather than the server determine and forward the request to another directory server.⁶⁶⁸

Because each location server can determine the location of location information on the location server network (i.e., "knows" where to find any requested location data), Steen alone or in combination with Skagerwall and/or Wolff renders obvious that each location server can resolve a data location request in two or fewer traversals.⁶⁶⁹ In the first traversal (i.e., first step), the client requests its desired location information from any location server, which responds with either the location information of the data or the location information for a location server known to contain the requested location information.⁶⁷⁰ In the second traversal (i.e., second step), if necessary, the client uses the redirect message to make an alternate request (e.g., Wolff's redirected alternate path⁶⁷¹) for the location information from another location server (e.g., Skagerwall's S28 checking other servers and Steen's 'awareness' and pointers)⁶⁷² known to contain the desired location information. Thus, Steen alone or in combination with Skagerwall and/or Wolff renders obvious resolving any client's data location request in two or fewer traversals.⁶⁷³

Accordingly, Steen and Skagerwall in view of Wolff, disclose or render obvious in combination claim elements 18.h.

- j) *Claim element 17.f: "in response to receiving a data location request from a client to retrieve a location string associated with an identification string provided in the data location request, transmitting a redirect message to the client if the identification string is not associated with a location string at the data location server, wherein the redirect message contains information for use by the client to calculate a location of a different data location server in*

⁶⁶⁸ See *id.*

⁶⁶⁹ See *id.* ¶ 447.

⁶⁷⁰ Skagerwall at 12:9-53; Steen at 105-07.

⁶⁷¹ Wolff at 5:25-38.

⁶⁷² Skagerwall at 12:63-13:6; Steen at 105-07.

⁶⁷³ See Greene Decl. ¶ 447.

the plurality of data location servers, wherein the different data location server contains the location string.”

Claim elements 17.f is substantially similar to the combination of element 18.g and 18.h. Steen discloses this claim element or renders it obvious alone or in view of Skagerwall and/or Wolff.⁶⁷⁴ As described above for claim element 18.g, each of Steen, Skagerwall, and Wolff teaches a data location query for location strings associated with an identification string from a client to a location server.⁶⁷⁵ And as explained for claim element 18.h, Steen discloses, or renders obvious in combination with Skagerwall and/or in further view of Wolff that, if the queried location server doesn't contain the desired location string, it sends a redirect message to the client containing any necessary information for the client to identify the location server containing the location string associated with the desired identifier string.⁶⁷⁶

Thus, based on the above teachings, Steen, Skagerwall, and Wolff disclose or render obvious claim element 17.f. As a result, and in sum, Steen alone or in combination with Skagerwall and in further view of Wolff render obvious independent claims 17 and 18.

2. Steen, Skagerwall, and Wolff Render Obvious Dependent Claim 24.

Steen alone or in view of Skagerwall and/or Wolff render obvious the additional limitations of claim 24. Claim 24 recites: “The system of claim 18, wherein the location information comprises a portion of a hash table distributed over the plurality of data location servers.”

During the prior reexamination, Patent Owner asserted that the prior art's hierarchical configuration failed to teach the requirements of claims 2, 9, and 12. This Request focuses on

⁶⁷⁴ See *id.* ¶ 449.

⁶⁷⁵ See *id.*

⁶⁷⁶ See *id.*

Steen and Skagerwall's teachings toward a system having a non-hierarchical configuration that renders obvious claims 2, 9, and 12.

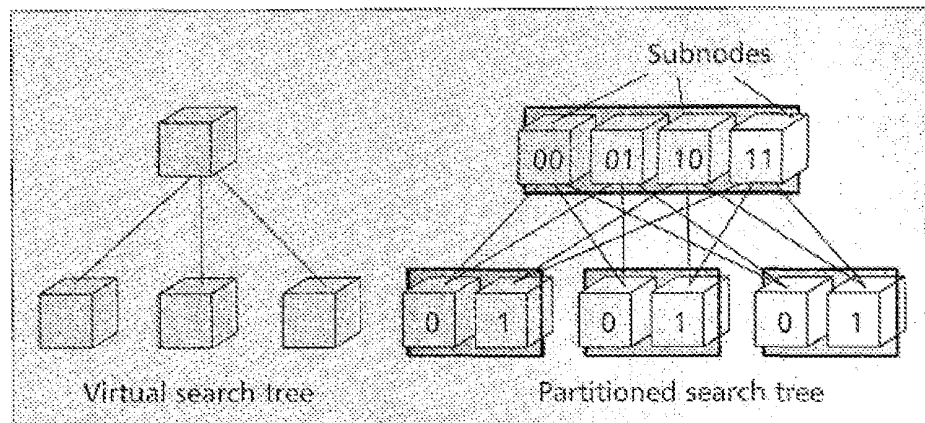
As described above, Steen in view of Skagerwall and Wolff render obvious each element of claim 18. Steen, Skagerwall, and Wolff further teach that organizing the disclosed location servers according to hashing methods, including a hash table based on a hash function. For example, Skagerwall describes "storing a list of a plurality of available directory servers in a hashtable; mapping all object tags employing a hashfunction and the hashtable onto the directory servers."⁶⁷⁷

Steen similarly provides the use of hashing to organize the identifier string-location information on its directory subnode location servers. As shown below in Figure 6, Steen explains using the first n-bits of an object's handle to identify the subnode responsible in a hashing method:

As an example, we can use the first n bits of an object's handle to identify the subnode responsible for that object...To illustrate, Fig. 6 shows a search tree in which the root node has been partitioned into four subnodes based on the first two bits of the object handle ($n = 2$), and each of the leaf nodes into two subnodes ($n = 1$).⁶⁷⁸

⁶⁷⁷ Skagerwall, 3:51-54.

⁶⁷⁸ Steen at 108.



■ **Figure 6.** *A search tree and a corresponding logical tree after partitioning the directory nodes into subnodes.*

Steen further notes that more complex hashing methods exist and should be used in some applications: “We note that we have developed more realistic hashing-based methods than explained here. For example, we also have to take into account that the total number of links between parent and children subnodes remains manageable.”⁶⁷⁹ A POSITA would understand that such “realistic hashing-based methods” include more complex hashing methods, including hash functions. Accordingly, it would have been obvious to a POSITA that, if the division of location information data was more complex, it would be beneficial to organize its distribution with a hash function.

And Steen describes that a POSITA would find important for each location server to be aware of how the location information has been partitioned and for each location server to easily store the hashing information:

Because communication between directory nodes in the original search tree now takes place between their respective subnodes, each subnode should be aware of how the directory node with which it communicates is actually partitioned. This information is contained in a separate tree management service. This service also

⁶⁷⁹ *Id.*

maintains the mapping of subnodes to physical nodes. Partitioning and mapping information is assumed to be relatively stable so that it can be easily cached by subnodes.⁶⁸⁰

As Mr. Greene explains, a POSITA would have understood the location servers will store this partitioning and mapping information and use the hash function to determine which location information should be stored on which location server.⁶⁸¹ Steen emphasizes the need to minimize each location server's need to request partition information from other services. Steen explains that excessive requests can make the system less efficient by demanding repeated requests: "This assumption is necessary to avoid having to query the management service each time a subnode needs to communicate with its parent or children, which would turn the management service into a potential communication bottleneck."⁶⁸²

As Mr. Greene details, hashing tables makes it easier to find a piece of data when needed.⁶⁸³ Thus, Mr. Greene explains that a POSITA would have been motivated to hash the partitioned location servers taught by Steen. Doing so would have allowed any component on the network to "know" where a desired piece of location information was located.⁶⁸⁴ As a result of this knowledge, as Mr. Greene explains, it would have been obvious to a POSITA for Steen's location information to include a portion of a hash table distributed over its location servers.⁶⁸⁵

Further, as shown above in SNQ 1, Skagerwall's teachings render obvious claim 24's limitations. As shown below in annotated Figure 2, Skagerwall further provides examples of a

⁶⁸⁰ Steen at 108.

⁶⁸¹ See Greene Decl. ¶ 456.

⁶⁸² Steen at 108.

⁶⁸³ See Greene Decl. ¶ 457.

⁶⁸⁴ See *id.*

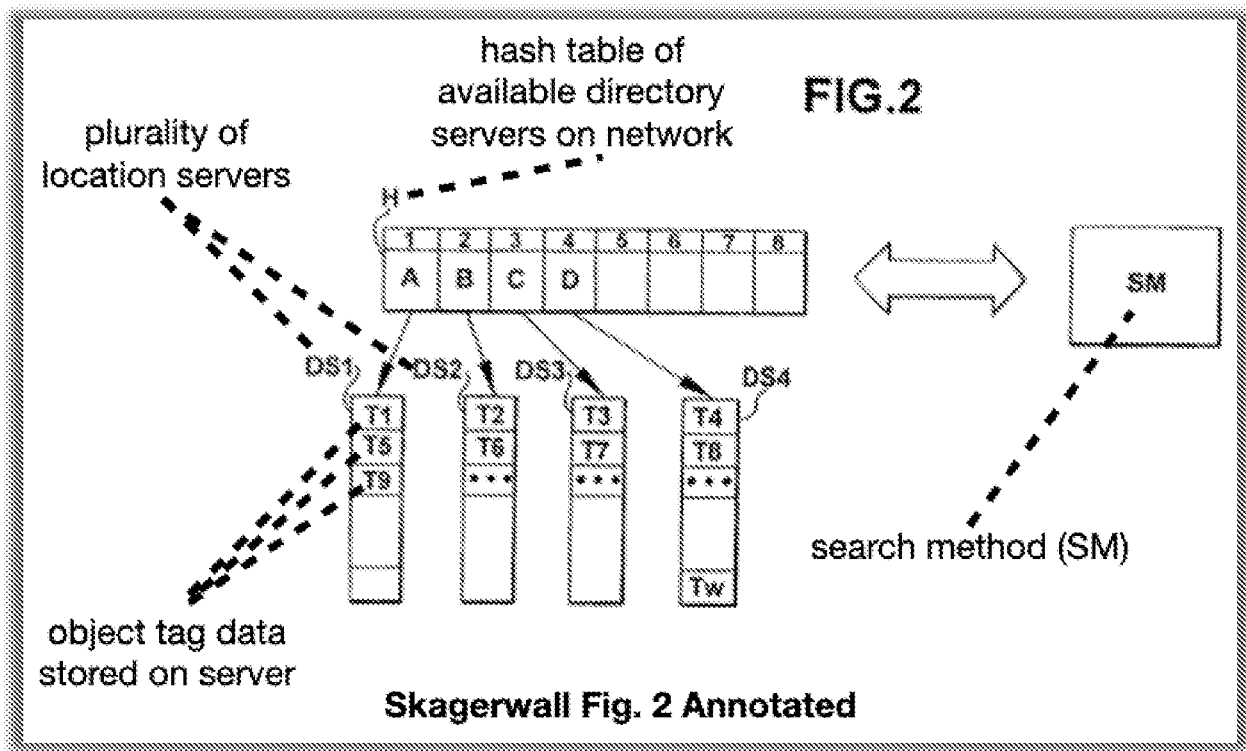
⁶⁸⁵ See *id.*

hashtable showing the available directory servers and their contents.⁶⁸⁶ Based on Skagerwall's teachings, a POSITA would have appreciated that the directory servers store the hash table and that the hash table contains the list of available directory servers with their respective contents. Further, a POSITA would have understood that the access server (i.e., client) uses portions of the hash table to identify which directory server to send its data request based on the information in the redirect message. As Mr. Greene explains, a POSITA would have concluded that sending the entire hash table or storing the entire hash table at the client would be too cumbersome.⁶⁸⁷ As a result, a POSITA would have found it obvious to break the hash table into portions, to distribute those portions to the directory server, and to send only the relevant portion to the client. Further, a POSITA would have understood that this portion includes at least the portion describing the object tag requested by the client. As a result, based on Skagerwall's teachings, a POSITA would have been motivated to have Steen's location information include a portion of a hash table distributed over the plurality of location servers.⁶⁸⁸

⁶⁸⁶ Skagerwall at 10:39-49.

⁶⁸⁷ See Greene Decl. ¶ 458.

⁶⁸⁸ See *id.*



As explained regarding claim element 18.h, Steen, Skagerwall, and Wolff render obvious sending a redirect message with information necessary to identify the directory server containing the object data for a given object tag. As Mr. Greene explains, the mapping information or the relevant portion of the hash table, would be necessary for the access server (i.e., client) to identify the correct directory server and send the request to it.

Thus, as Mr. Greene explains, Steen, Skagerwall, and Wolff render obvious claim 24.

G. Secondary Considerations

This Request demonstrates that the Challenged Claims of the '640 Patent are unpatentable as obvious in view of the prior art references. In the related district court matter, Patent Owner has argued that the purported existence of copying, long-felt-but unsolved-need, failure of others, commercial success and industry praise, and skepticism of skilled artisans weighs against

a finding of obviousness.⁶⁸⁹ But these alleged secondary indicia of non-obviousness aren't sufficient to overcome this Request's showing of obviousness for at least two reasons.

First, the Federal Circuit has made it clear that a nexus must exist between an indicium of non-obviousness and the claimed invention.⁶⁹⁰ For example, *In re Kao* explains that, “[w]here the offered secondary consideration actually results from something other than what is both claimed and novel in the claim, there is no nexus to the merits of the claimed invention.”⁶⁹¹ As another example, *Fox Factory* specifies that, “[a]lthough we do not require the patentee to prove perfect correspondence to meet the coextensiveness requirement, what *we do require is that the patentee demonstrate that the product is essentially the claimed invention.*”⁶⁹² In the related district court case, Patent Owner has presented evidence relating only generally to distributed systems. But Patent Owner has failed to present the required objective evidence of non-obviousness having *a nexus* to the claimed invention or that the product-at-issue is *essentially the claimed invention*. And in Reexamination Control No. 90/019,034 for the related '978 Patent, the Reexamination Specialist found that there was no objective evidence of a nexus between the scope of the claims and Patent Owner's evidence of secondary considerations of non-obviousness.⁶⁹³

Second, Patent Owner has never shown that any sold commercial product actually practices the claimed invention. Without such a showing, Patent Owner's purported evidence toward copying, commercial success, industry praise, and skepticism isn't relevant to the non-obvious-

⁶⁸⁹ See Ex. 22 (Final Validity Contentions) at 4-5.

⁶⁹⁰ *In re Kao*, 639 F.3d 1057, 1068 (Fed. Cir. 2011); *Tokai Corp. v. Easton Enters., Inc.*, 632 F.3d 1358, 1369 (Fed. Cir. 2011); *Ormco Corp. v. Align Tech., Inc.*, 463 F.3d 1299, 1312 (2006).

⁶⁹¹ 639 F.3d at 1068.

⁶⁹² *Fox Factory, Inc. v. SRAM LLC*, 944 F.3d 1366, 1374 (Fed. Cir. 2019).

⁶⁹³ 10/27/2022 Final Office Action in Reexamination Control No. 90/019,034 at 21.

ness analysis. Indeed, the Federal Circuit has pronounced that “to determine whether a nexus exists, our cases require consideration of whether the marketed product embodies the claimed features.”⁶⁹⁴

As explained below, Patent Owner hasn’t produced evidence that a commercial product embodies the claimed invention or that there is a nexus between the claimed invention and the purported secondary indicia of non-obviousness:

Copying: To establish copying, the MPEP states that a patentee must show that others in the field copied the claimed invention after failing to design a product based on the prior art.⁶⁹⁵ But Patent Owner has previously made only nebulous assertions about the similarities between the claimed invention and other commercial products. More specifically, Patent Owner has only claimed that its non-adjudicated infringement allegations in the related district court matter are evidence of copying, but provided no further evidence.

Long-Felt Need and Failure of Others: To show this secondary consideration, the MPEP states that a patentee must present evidence of (a) a persistent need recognized by POSITAs, (b) others not satisfying the purported need before the claimed invention, or (c) the claimed invention actually satisfying the long-felt need.⁶⁹⁶ But Patent Owner has asserted that others (e.g., U.S. Patent No. 5,542,087 to Neimat) discuss the general need for hashing schemes and load balancing in distributed systems without providing evidence how the claimed invention actually satisfies any purported-long felt need.⁶⁹⁷

⁶⁹⁴ *Bosch Automotive Service Solutions, LLC v. Matal*, 878 F.3d 1027, 1036-37 (Fed. Cir. 2017).

⁶⁹⁵ See MPEP 716.06.

⁶⁹⁶ See *id.* 716.04.

⁶⁹⁷ 08/04/2022 Patent Owner’s Reply to Office Action in Reexamination Control No. 90/019,036 at 47.

Similarly, Patent Owner has asserted that others (e.g., *Distributed Names Servers: Naming and Caching in Large Distributed Computing Environments* to Terry) discussed the failure of others to develop technology that would enable highly scalable information databases and the unlikelihood of the claimed invention.⁶⁹⁸ But Patent Owner hasn't provided any evidence establishing what the claimed invention actually achieves to overcome any purported failure of others. And as explained above in Section III.C.5, OracleNamesAdminGuide establishes that its system enables highly scalable information databases overcoming any purported failures by others that Patent Owner has asserted.

Further, Patent Owner has asserted that a 2004 article suggests “distributed hash tables” allow for identifying “files scattered around the Internet.”⁶⁹⁹ The '640 Patent, however, specifies that it employs the “well-known function” hashpjw for distributing its hash table over the locations servers (e.g., NDTP servers):

The appropriate NDTP server is selected from the table in the NDTP_RDR_RSP message by applying a **well-known function** to the identifier string and using the function result as an index into the NDTP server table.

The well-known function preferably applied is the hash pjw function presented by Aho, Sethi and Ullman in their text *Compilers, Principles, Techniques and Tools*:

```
uint32_t
hash (uint8_t *s, uint32_t slen, uint32_t size)
{
    uint32_t g;
    uint32_t i;
        uint32_t h = 0;
    uint8_t c;
    for (i = 0; i < slen; i++) {
        c = s[i];
```

⁶⁹⁸ *Id.* at 48-51.

⁶⁹⁹ *Id.* at 47.

```
        h = (h << 4) + c;  
        g = (h & 0xf0000000);  
        if (g) {  
            h ^= g >> 24;  
            h = g;  
        }  
    }  
    return h % size;  
}.700
```

Thus, the ‘640 Patent acknowledges that the authors Aho, Sethi, and Ullman already solved any purported long-felt need for the claimed hashing scheme.

Commercial Success and Industry Praise: Patent Owner hasn’t met its burden of proof in establishing a nexus between the claimed invention and any evidence of commercial success. In Reexamination Control No. 90/019,034 for the related ’978 Patent, the Reexamination Specialist stated that “it is important to point out the product family is facilitated by numerous patents and numerous independent claims within said patents. Thus, without any actual evidence of a nexus between the commercial success and the claimed invention ... it would be impossible to prove that [the challenged claims] of this specific patent and not one of the other patents is the reason for commercial success.”⁷⁰¹ The Specialist also stated that Patent Owner must show that commercial success is directly derived from the claimed invention, but has failed to do so:

PO has merely alleged that the claimed invention has “substantial overlap” with the accused products. As clearly stated, the PO must prove that the success is “directly derived” and not merely related to the claimed invention. These allegations can not serve as proof that the claimed elements meet the requirements for showing evidence of commercial success.⁷⁰²

⁷⁰⁰ See ‘640 Patent at 14:64-15:21.

⁷⁰¹ 10/27/2022 Final Office Action in Reexamination Control No. 90/019,034 at 22.

⁷⁰² *Id.* at 22-23 (citing *In re Mageli*, 470 F.2d 1380 (C.C.P.A. 1973); *In re Noznick*, 478 F.2d 1260 (C.C.P.A. 1973)).

Further, Patent Owner has failed to establish that any purported success isn't the result of promotion or advertising, shift in advertising, consumption by purchasers normally tied to sellers, or other business events extraneous to the merits of the claimed invention, as the MPEP requires.⁷⁰³ And Patent Owner has failed to present any evidence of persons praising the merits of the claimed invention rather than praising general concepts like distributed data storage. And Patent Owner hasn't presented any evidence that it has licensed the claimed invention to any third party.

Skepticism of Skilled Artisans: The MPEP establishes that, to show skepticism, a patentee must present evidence of actual skepticism of the claimed invention.⁷⁰⁴ But Patent Owner has previously made only nebulous hearsay assertions about audience members' reception of a presentation at Yale University.

As such, the purported existence of copying, long-felt need, failure of others, commercial success, industry praise, and skepticism doesn't overcome the conclusion of obviousness.⁷⁰⁵

⁷⁰³ See MPEP 716.03.

⁷⁰⁴ See MPEP 716.05.

⁷⁰⁵ See Greene Decl. ¶ 461.

IV. CONCLUSION

Skagerwall in view of Wolff, OracleNamesAdminGuide in view of OracleUnleashed and/or Steen, and Steen in view of Skagerwall and/or Steen raise a substantial new question of patentability for independent claims 17 and 18, and for dependent claim 24, which depends from claim 18. As such, the Requester respectfully requests reexamination of claims 17, 18, and 24.

Dated: February 16, 2023

Respectfully submitted,

/Andrew Landers Ramos/
Andrew Landers Ramos (Reg. No. 76,865)

FISCH SIGLER LLP
5301 Wisconsin Avenue NW
Suite 400
Washington, DC 20015
202.362.3522
Andrew.Ramos@fischllp.com
PostGrant@fischllp.com