

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

TESLA, INC.,
Petitioner

IPR2025-00341
U.S. Patent No. 7,181,743

**PETITION FOR *INTER PARTES* REVIEW
UNDER 35 U.S.C. § 312 AND 37 C.F.R. § 42.104**

TABLE OF CONTENTS

Petitioner’s Exhibit List	4
I. Introduction.....	6
II. Grounds for standing	7
III. Note.....	7
IV. Technology background	7
V. The ’743 patent.....	7
VI. Prosecution history of the ’743 patent.....	14
VII. Priority date of the ’743 patent	15
VIII. Level of ordinary skill in the art	17
IX. Claim construction.....	17
X. Relief requested and the reasons for the requested relief.....	17
XI. Identification of how the claims are unpatentable.....	17
A. Challenged claims	17
B. Statutory grounds for challenges.....	18
C. Ground 1: Claims 10, 11, 14, 16, and 18 are obvious under 35 U.S.C. § 103 over Welch alone and in combination with Ravindran.	20
D. Ground 2: Claims 10, 11, 14, 16, and 18 are obvious under 35 U.S.C. § 103 over Welch, Ravindran, and Hosokawa.	79
XII. Welch and Ravindran are Printed Publications	96
XIII. Discretionary denial is inappropriate.....	97

A.	No basis for § 325(d) denial	97
B.	No basis for <i>Fintiv</i> denial	98
C.	No basis for <i>General Plastic</i> denial	100
XIV.	Conclusion	101
XV.	Mandatory notices	102
A.	Real party-in-interest.....	102
B.	Related matters	102
C.	Lead and back-up counsel and service information.....	103
	Certificate of Word Count	104
	Certificate of Service	105

PETITIONER’S EXHIBIT LIST

Ex.1001	U.S. Patent No. 7,181,743 to Beale et al. (“’743 patent”)
Ex.1002	Prosecution File History of U.S. Patent No. 7,181,743
Ex.1003	Declaration of Erez Zadok under 37 C.F.R. § 1.68
Ex.1004	<i>Curriculum Vitae</i> of Erez Zadok
Ex.1005	“ <i>Distributed, Scalable, Dependable Real-Time Systems: Middleware Services and Applications.</i> ” Lonnie R. Welch et al., Proceedings 13th International Parallel Processing Symposium and 10th Symposium on Parallel and Distributed Processing, April 1999 (“Welch”)
Ex.1006	“ <i>Specification and Modeling of Dynamic, Distributed Real-time Systems.</i> ” Lonnie R. Welch et al. IEEE Real-Time Systems Symposium, 72-81, IEEE Computer Society Press, Dec. 1998 (“Ravindran”)
Ex.1007	U.S. Patent No. 6,088,727 to Hosokawa et al. (“Hosokawa”)
Ex.1008	U.S. Patent No. 6,269,396 to Shah et al. (“Shah”)
Ex.1009	U.S. Provisional Patent No. 60/207,891 (“’891 provisional”)
Ex.1010	IV’s Complaint, <i>Intellectual Ventures II, LLC v. Tesla, Inc.</i> , No. 6:24-cv-188-ADA (WDTX)
Ex.1011	Proposed Scheduling Order, <i>Intellectual Ventures II, LLC v. Tesla, Inc.</i> , No. 6:24-cv-188-ADA (WDTX)
Ex.1012	Statistics on District Court Timing
Ex.1013	Interim Procedure for Discretionary Denials in AIA Parallel District Court Litigation, June 21, 2022
Ex.1014	IV’s Preliminary Infringement Contentions, <i>Intellectual Ventures II, LLC v. Tesla, Inc.</i> , No. 6:24-cv-188-ADA (WDTX)

Ex.1015	Declaration of Sylvia Hall-Ellis, Ph.D.
Ex.1016	U.S. Patent Publication No. 2002/0064126 to Bhaatal (“Bhattal”)
Ex.1017	U.S. Patent No. 6,041,306 to Du (“Du”)
Ex.1018	U.S. Patent No. 6,742,020 to Dimitroff (“Dimitroff”)
Ex.1019	Microsoft Computer Dictionary, 3rd ed., 1997, excerpts
Ex.1020	Microsoft Computer Dictionary, 4th ed., 1999, excerpts

I. INTRODUCTION

U.S. Patent No. 7,181,743 (“the ’743 patent,” Ex.1001) was granted in error because a subset of the named inventors (along with others) publicly disclosed the claimed subject matter in a printed publication—a whitepaper—more than a year before filing. During prosecution, the whitepaper was not disclosed to or otherwise considered by the Office.

The ’743 patent attempts to protect the concept of a “resource manager for a distributed environment,” which creates copies of a scalable application across a network. The resource manager starts, stops, or moves the application copies in response to performance information. Ex.1001, Abstract. The whitepaper (“Welch,” Ex.1005) describes this same “resource manager” concept using the same terminology. Ex.1005, 2-4 (describing “moving or replicating an application program” based on Quality of Service (QoS) performance).

Accordingly, pursuant to 35 U.S.C. §§ 311, 314(a), and 37 C.F.R. § 42.100, Tesla, Inc. (“Petitioner”) respectfully requests that the Board review and cancel as unpatentable under (pre-AIA) 35 U.S.C. §103(a) claims 10, 11, 14, 16, and 18 (the “Challenged Claims”) of the ’743 patent.

II. GROUNDS FOR STANDING

Petitioner certifies that the '743 patent is eligible for IPR and that Petitioner is not barred or estopped from requesting IPR challenging the patent claims. 37 C.F.R. § 42.104(a).

III. NOTE

Petitioner cites to exhibits' original page numbers, unless noted otherwise. **Emphasis** in quoted material has been added. Claim terms are presented in *italics*.

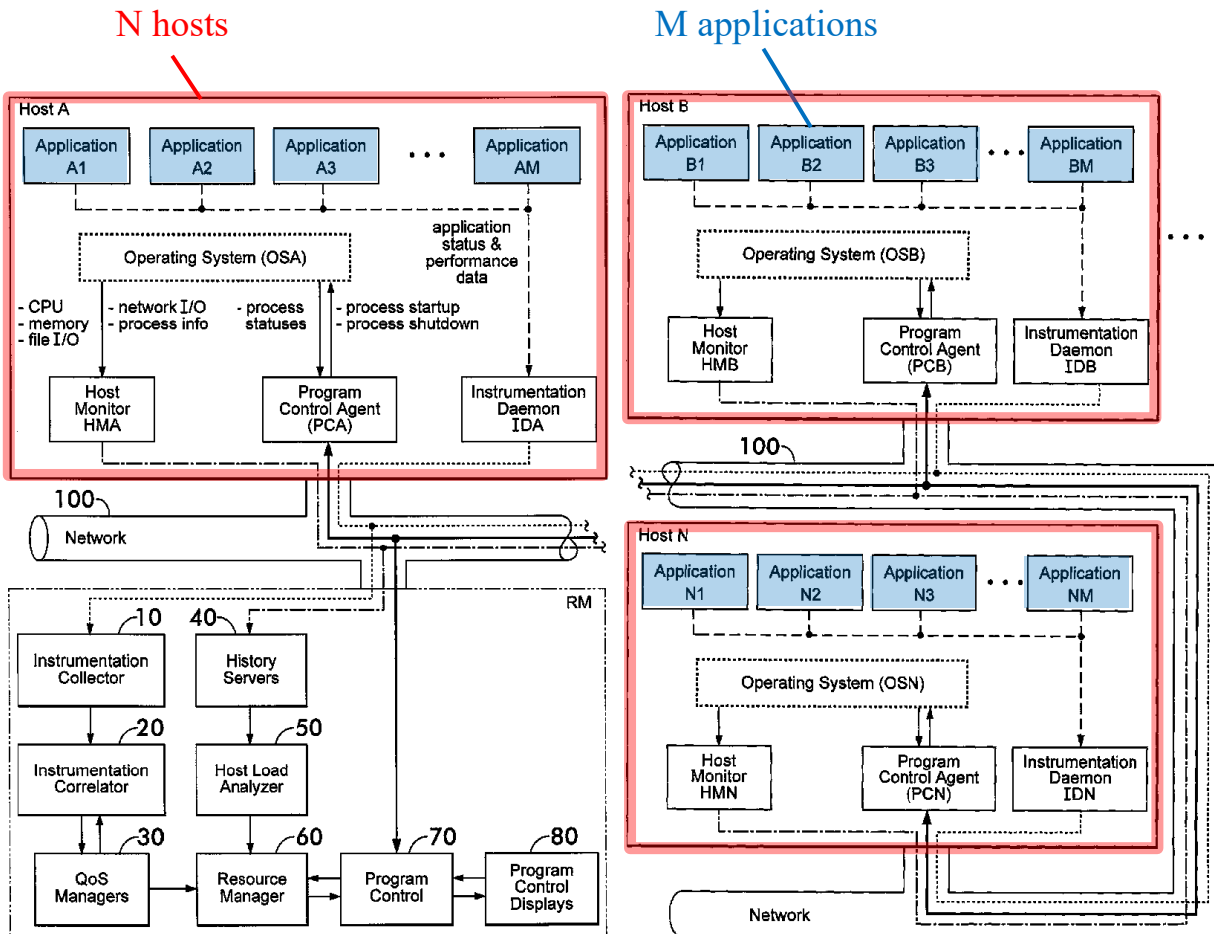
IV. TECHNOLOGY BACKGROUND

Dr. Zadok's expert declaration provides a brief background on distributed and scalable computing systems, as they were known in the late 1990s. *See* Ex.1003, ¶¶39-79.

V. THE '743 PATENT

The '743 patent relates to “a distributed environment comprised of hosts instantiating copies of a scalable application.” Ex.1001, 2:46-47. A scalable application is one that uses multiple copies (i.e., instances) of an application that scale up or down to maintain a desired performance. *See, e.g.*, Ex.1001, 18:12-58. “[T]he present invention is illustrated in FIGS. 1A, 1B, which includes a plurality of Host computers A, B, ..., N operatively connected to one another and Resource Management hardware RM via a Network 100.” Ex.1001, 5:31-36. “[I]t will be appreciated that hosts A-N each can instantiate applications 1-M.” Ex.1001, 5:45-

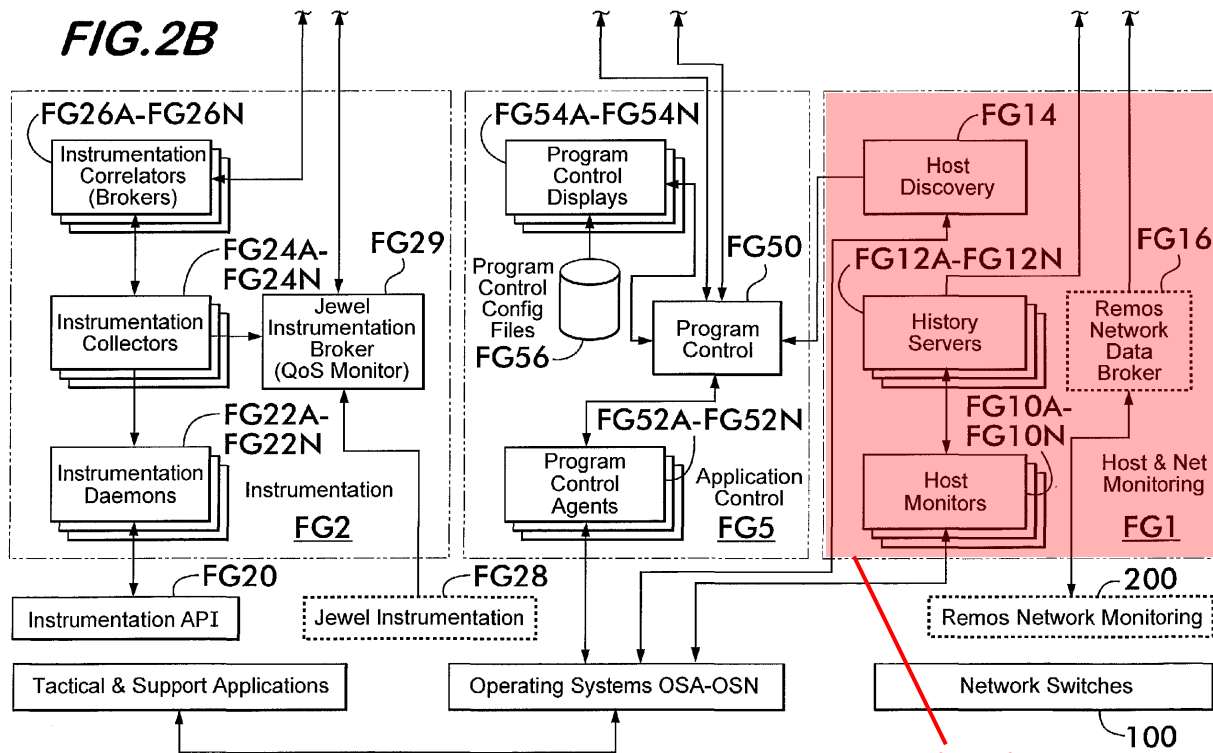
49. The N hosts (red) and M applications (blue) are shown below in Figs. 1A and 1B.



Ex.1001, Figs. 1A, 1B (annotated); Ex.1003, ¶80.

The distributed environment, shown above, is managed by “a resource management device generating signals which start up, shutdown or move a selected one of the copies responsive to [] information regarding performance of all copies of the scalable application and [] information regarding performance of the hosts.” Ex.1001, 2:47-52.

The '743 patent describes management of the distributed computing system in terms of “functions” and “function groups” (described in FG# format). Ex.1001, 7:15-8:61. For example, the '743 patent describes a function group for monitoring the hosts and the network: “FG1—Host and Network Monitoring. This function group consists of software which monitors the host and network resources within the distributed environment.” Ex.1001, 7:29-31. FG1 “collects extensive run-time information on host and network configuration, statuses, and performance.” Ex.1001, 7:31-32. FG1 is recited in claim 10, for example, as “*a first function group which monitors the host and network resources.*” Ex.1001, claim 10 (limitation [10.1] below). The first function group is illustrated in Fig. 2B below.

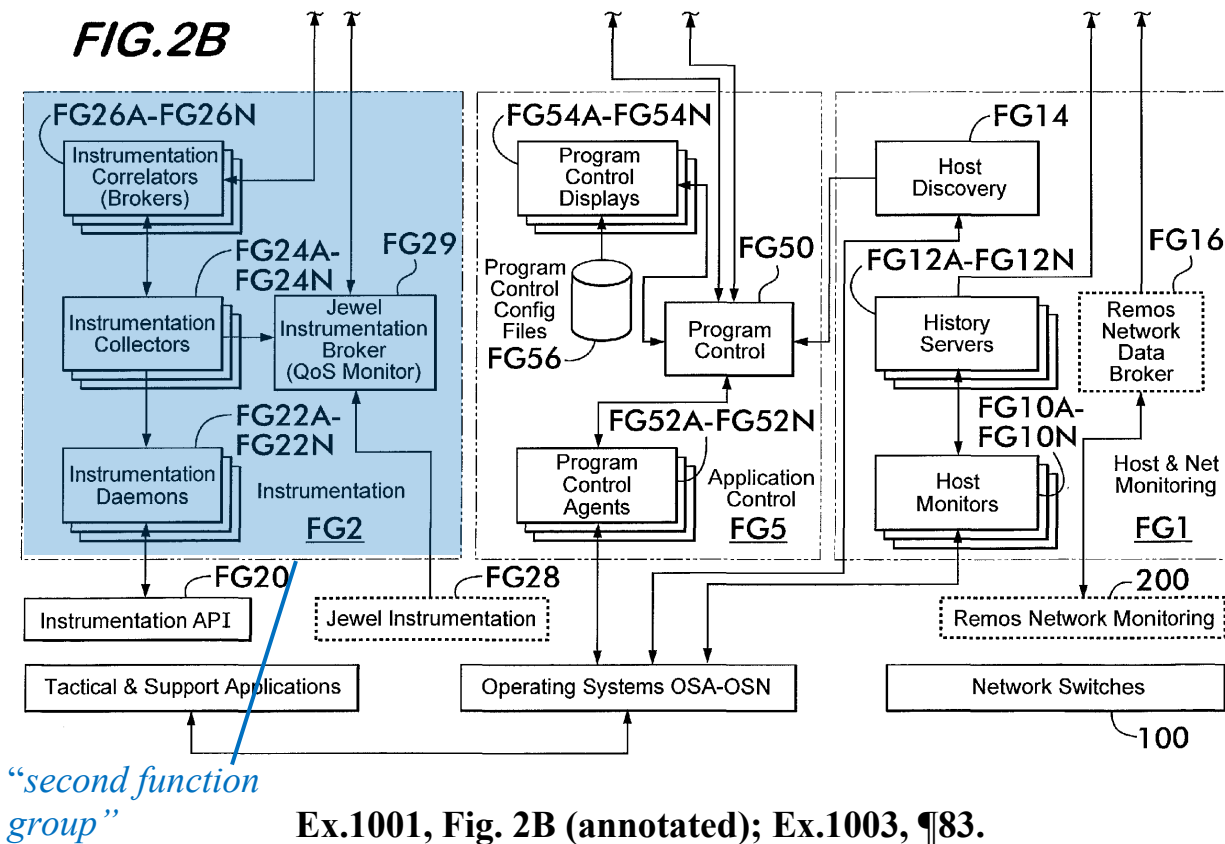


Ex.1001, Fig. 2B (annotated); Ex.1003, ¶82.

“first function group”

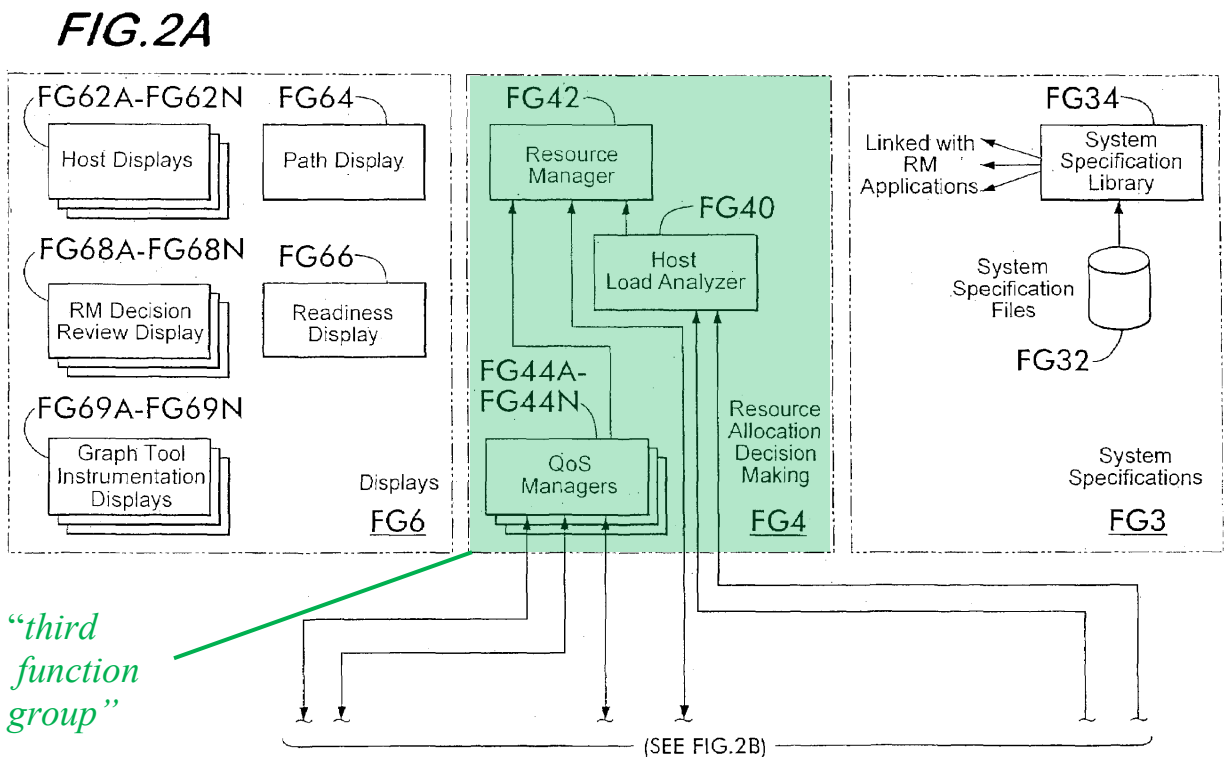
The '743 patent also describes a second function group for monitoring the applications that are running on the hosts: “FG2—Application-Level Instrumentation. The instrumentation function group provides general-purpose application event reporting and event correlation capabilities.” Ex.1001, 7:40-42. “This application data is forwarded to other components of the instrumentation subsystem which collect data from applications on hosts throughout the distributed environment.” Ex.1001, 7:47-49. The second function group is recited in claim 10, for example, as “a second function group which provides application computer program event reporting and event correlation capabilities.” Ex.1001, claim 10

(limitation [10.2] below). The second function group is illustrated in Fig. 2B below.



The '743 patent also describes a third function group for making decisions and allocating resources: “FG4—Resource Allocation Decision-Making. This subsystem provides the reasoning and decision-making capabilities of the Resource Management architecture.” Ex.1001, 8:5-7. “The components of this subsystem use information from other subsystems in order to determine the health and state of the distributed environment and the options that are available for attempting to recover from faults or unacceptable performance.” Ex.1001, 8:7-9. For example,

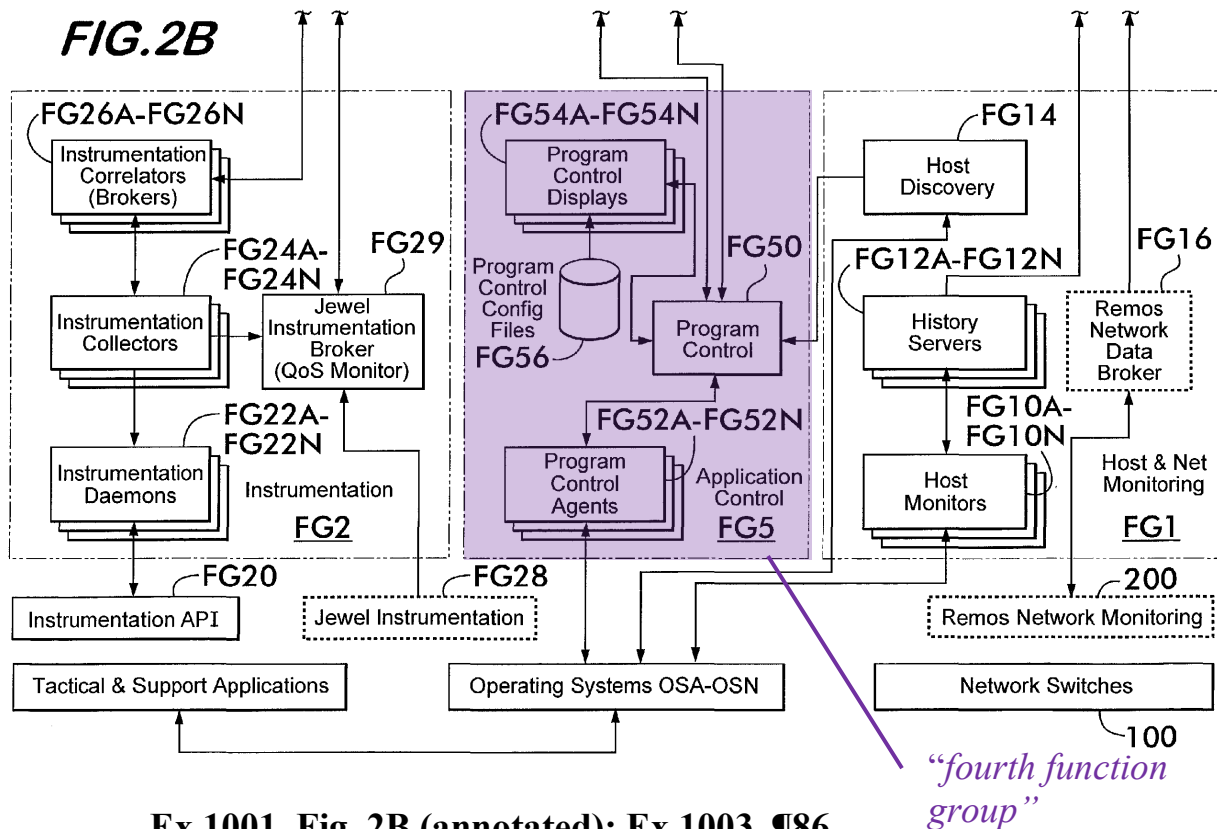
the third function group may decide “1) where new applications should be started; 2) whether and where failed applications should be restarted; 3) based on application inter-dependencies, whether and where additional applications should [] be started prior to starting a particular application,” etc. Ex.1001, 8:14-23. This function group is recited in claim 10 as “a third function group which provides reasoning and decision making capabilities for the resource managed system.” Ex.1001, claim 10 (limitation [10.3] below). The third function group is illustrated in Fig. 2A below.



Ex.1001, Fig. 3 (annotated); Ex.1003, ¶85.

Finally, the '743 patent describes a fourth function group for enacting the

decisions made by the third function group: “FG5—Application (Resource) Control. This subsystem provides application control (i.e., Program Control) capabilities which permit starting, stopping, and configuring applications on each of the hosts in the distributed environment.” Ex.1001, 8:31-35. This function group is recited in claim 10 as “a fourth function group which provides program control capabilities permitting starting, stopping, and configuring of selected ones of the *M* managed characteristic application computer programs on respective ones of the *N* hosts in the resource managed system.” Ex.1001, claim 10 (limitation [10.7] below). The fourth function group is illustrated in Fig. 2B below.



Ex.1001, Fig. 2B (annotated); Ex.1003, ¶86.

The claims of the '743 patent merely recite basic concepts of managing a distributed computing system using the functionality described above. In particular, the claims recite the basic concepts of 1) monitoring the hosts and network, 2) monitoring the applications, 3) making decisions based on the monitoring, and 4) enacting those decisions. *See, e.g.*, Ex.1001, claim 10; Ex.1003, ¶87.

However, as explained in this Petition, the functionality for managing a distributed system was not new. Indeed, some of the inventors of the '743 patent, along with others, already disclosed these concepts in various papers before the '743 patent was filed. Ex.1003, ¶¶80-88.

VI. PROSECUTION HISTORY OF THE '743 PATENT

The '743 patent was filed as U.S. Application No. 09/864,825 on May 24, 2001. It claims priority to U.S. provisional application No. 60/207,891 (“'891 provisional”) (Ex.1009), filed on May 25, 2000. *See* Ex.1001, title page.

After several rounds of prosecution, the Office allowed some claims, including what is now issued claim 10. Ex.1002, 121, 123-33, 164, 182-192, 208-222. The Office did not give any specific reasons for allowance. Ex.1002, 14-17. The most recent amendment to issued claim 10 related to “the QoS requirements dictating parameters regarding service quality of the M management characteristic application programs” and “that the managed characteristic application computer

programs are moved, shutdown, and started in accordance with satisfaction of the QoS requirements.” Ex.1002, 150-51.

However, for the reasons explained in this Petition, these concepts were already known. Ex.1003, ¶¶89-93.

VII. PRIORITY DATE OF THE ’743 PATENT

For the ’743 patent to be entitled to the priority date of the ’891 provisional, the ’891 provisional (Ex.1009) must provide written description support for the subject matter claimed in the ’743 patent. *See New Railhead Mfg., L.L.C. v. Vermeer Mfg. Co.*, 298 F.3d 1290, 1294 (Fed. Cir. 2002) (to be entitled to the filing date of an earlier application, the earlier application must meet the written description requirement).

To satisfy the written description requirement of pre-AIA 35 U.S.C. §112 ¶ 1, 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 claimed invention. *See Ariad Pharm., Inc. v. Eli Lilly & Co.*, 598 F.3d 1336, 1340, 94 USPQ2d 1161, 1167 (Fed. Cir. 2010) (en banc). An applicant shows possession of the claimed invention by describing the claimed invention with **all of its limitations**. *See Lockwood v. Amer. Airlines, Inc.*, 107 F.3d 1565, 1572, 41 USPQ2d 1961, 1966 (Fed. Cir. 1997). Moreover, it is not sufficient that the specification merely lists each claimed element. Rather, the specification must

support the particular claimed combination of elements. *See Hyatt v. Dudas*, 492 F.3d 1365, 1371, 83 USPQ2d 1373, 1376-1377 (Fed. Cir. 2007).

Here, the '891 provisional does not provide written description support for the specific combination of elements recited in the claims. Claim 10, for example, organizes various functions of the distributed computing management system into specific “functions” and “function groups.” Ex.1001, claim 10. However, the '891 provisional does not **describe** these specific function groups. Indeed, the term “function group” and the concept of function groups are entirely absent from the provisional application. *See* Ex.1009; Ex.1003, ¶¶94-96.

For example, the '891 provisional application does not describe “*a third function group which provides reasoning and decision making capabilities for the resource managed system.*” *See* Ex.1009. Moreover, while claim 10 recites that the “*third function*” includes a “*first function*,” a “*second function*,” and a “*third function*,” the '891 provisional does not describe these three specific functions as being within a “*third function group*.” Ex.1001, claim 10. In other words, no such arrangement of functions exists within the provisional application. *See* Ex.1009; Ex.1003, ¶97.

Accordingly, the Challenged Claims of the '743 patent are entitled to a priority date no earlier than the May 24, 2001 filing date of the '743 patent. Ex.1003, ¶98.

VIII. LEVEL OF ORDINARY SKILL IN THE ART

A person of ordinary skill in the art (“POSITA”) in May of 2001 would have been someone knowledgeable and familiar with the areas of networking and distributed computing that are pertinent to the ’743 patent. That person would have a bachelor’s degree in computer science, computer engineering, electrical engineering, or equivalent training, and approximately 2-3 years of professional experience relating to distributed computing. Lack of professional experience can be remedied by additional education, and vice versa. Ex.1003, ¶¶32-34.

IX. CLAIM CONSTRUCTION

Petitioner submits that, for the purposes of this proceeding and the grounds presented herein, no claim term requires express construction. *Nidec Motor Corp. v. Zhongshan Broad Ocean Motor Co.*, 868 F.3d 1013, 1017 (Fed. Cir. 2017). Ex.1003, ¶99.

X. RELIEF REQUESTED AND THE REASONS FOR THE REQUESTED RELIEF

Petitioner asks that the Board institute an IPR trial and find the Challenged Claims unpatentable in view of the analysis below.

XI. IDENTIFICATION OF HOW THE CLAIMS ARE UNPATENTABLE

A. Challenged claims

Petitioner challenges claims 10, 11, 14, 16, and 18.

B. Statutory grounds for challenges

Grounds	Claims	Pre-AIA Basis
#1	10, 11, 14, 16, and 18	35 U.S.C. § 103 over Welch alone and in combination with Ravindran
#2	10, 11, 14, 16, and 18	35 U.S.C. § 103 over Welch, Ravindran, and Hosokawa

Welch (Ex.1005): “Distributed, Scalable, Dependable Real-Time Systems: Middleware Services and Applications.” Welch is a whitepaper that was published in the Proceedings 13th International Parallel Processing Symposium and 10th Symposium on Parallel and Distributed Processing. The proceedings compilation has a copyright date of 1999, and includes papers presented from Apr. 12-19, 1999, at the symposium. Welch was publicly available as of at least Dec 16, 1999. *See* Section XII below (citing Dr. Hall-Ellis Declaration (Ex.1015)). Welch is thus prior art under pre-AIA 35 U.S.C. §102(b) because it was presented publicly and published more than one year before the non-provisional filing date (May 24, 2001) of the ’743 patent.

To the extent Patent Owner attempts to rely on the provisional filing date (May 25, 2000) of the ’743 patent, Welch is also prior art under pre-AIA § 102(a). Welch was authored by a subset of the inventors of the ’743 patent along with other non-inventors. Because the list of authors is different than the set of ’743 inventors (a different inventive entity), Welch is prior art under pre-AIA § 102(a).

See In re Katz, 687 F.2d 450, 215 USPQ 14 (CCPA 1982).

Ravindran (Ex.1006): “Specification and analysis of dynamic, distributed real-time systems,” in Proceedings of the 19th IEEE Real-Time Systems Symposium, 72-81, IEEE Computer Society Press, 1998” (“Ravindran”).

Ravindran is prior art under pre-AIA §102(b) because it was published and publicly accessible more than one year before the earliest priority date of the ’743 patent. In addition, Ravindran, like Welch, is prior art under pre-AIA §102(a) because it was publicly available as of at least December 10, 1998 (Ex.1015) and was authored by a subset of the inventors of the ’743 patent along with other non-inventors.

While the lead author of Ex.1006 is the same as Ex.1005 (Lonnie Welch), Petitioner refers to Ex.1006 by the second-named author (Binoy Ravindran) to distinguish from Ex.1005.

Hosokawa: U.S. Patent No. 6,088,727 (“Hosokawa”) was filed October 17, 1997, and issued on July 11, 2000. Hosokawa is prior art under pre-AIA § 102(a) and (e) for the non-provisional filing date (May 24, 2001) of the ’743 patent.

Petitioner’s § 103 obviousness grounds rely on the combined teachings of the references and not on a physical incorporation of elements. *See In re Mouttet*, 686 F.3d 1322, 1332 (Fed. Cir. 2012); Ex.1003, ¶102.

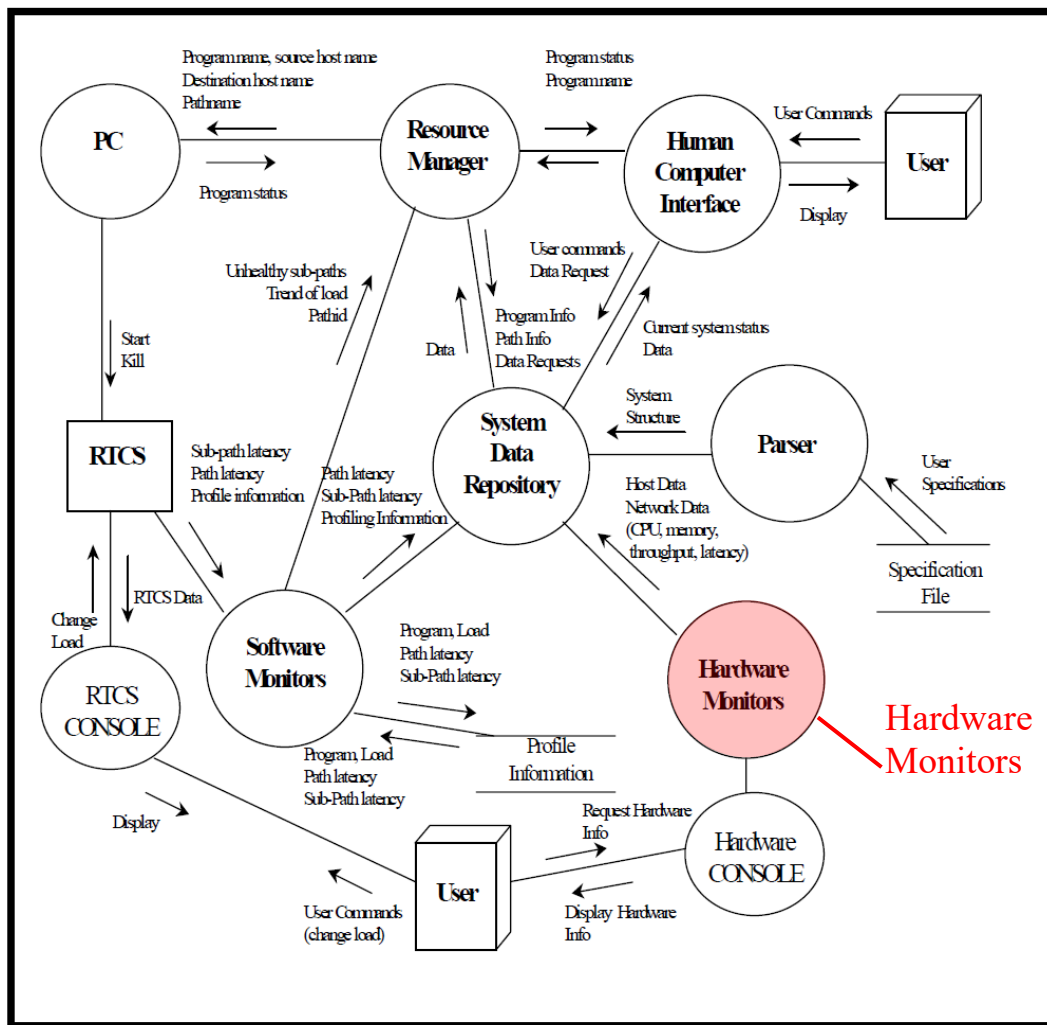
C. Ground 1: Claims 10, 11, 14, 16, and 18 are obvious under 35 U.S.C. § 103 over Welch alone and in combination with Ravindran.

1. Welch

Welch describes the same distributed system management software that is described and claimed in the '743 patent. Ex.1005, 1-5. In particular, Welch describes “distributed real-time systems” which are “scalable” and designed to “meet the QoS requirements of each application they manage.” Ex.1005, 1.

Welch is directed to QoS management software that manages resources for a distributed real-time system. Ex.1005, 1. Welch describes “distributed hardware” (e.g., a plurality of hosts) that run several instances of “Real-Time Control System (RTCS) application programs on the specified hosts.” Ex.1005, 2-3. These RTCS applications are scalable according to demand. A resource manager is used to “move a program to a different host or LAN, shed a program, or replicate a program) to improve QoS.” Ex.1005, 3. Welch illustrates both the “logical architecture” and the “physical architecture” of the QoS management software in Figs. 1 and 2 below.

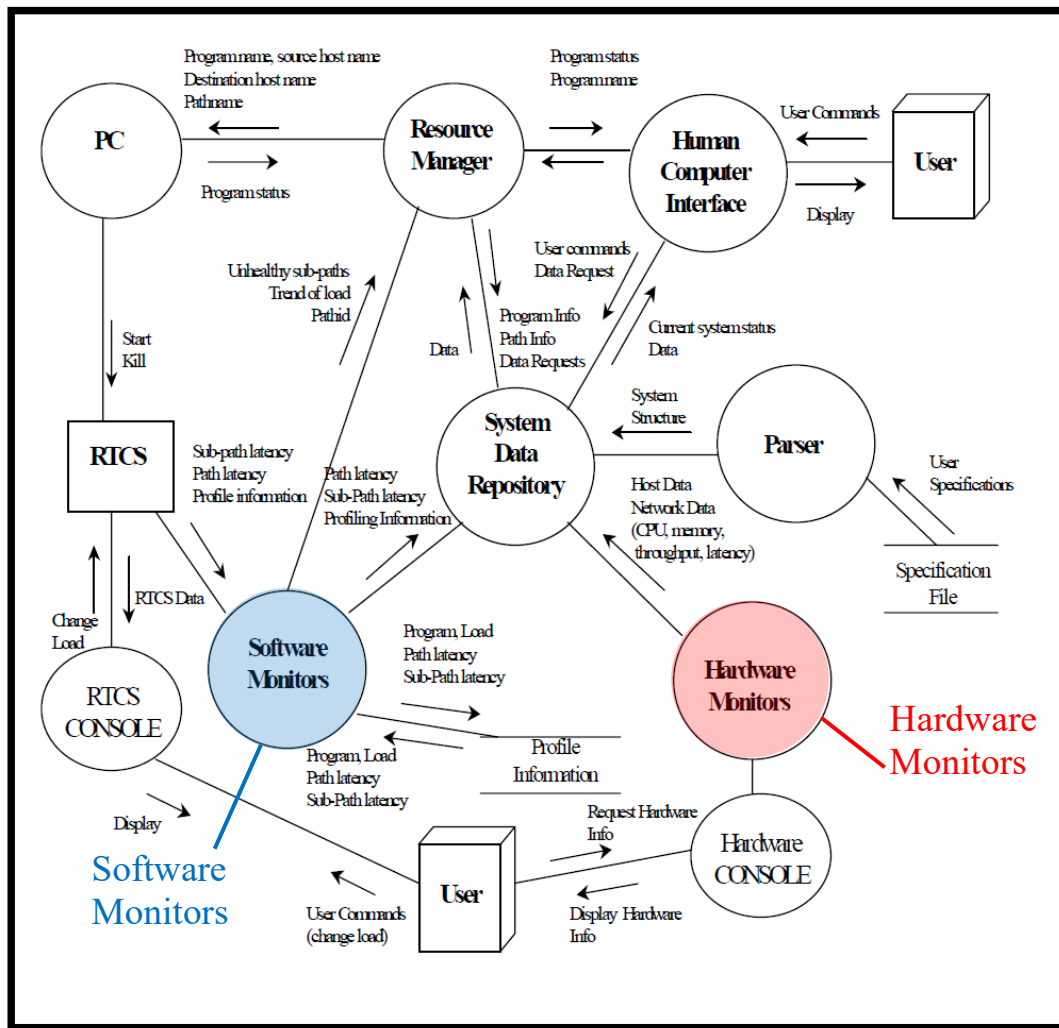
Welch discloses the same functionality corresponding to the “function groups” that are recited in the Challenged Claims. Ex.1005, 2-3. For example, Welch describes a first function group in the form of hardware monitors that monitor host and network resources. “**Hardware (HW) Monitors** continuously observe a resource’s load index and pass this information to the Hardware Data Repository (step 2 of Figure 3), which in turn passes such information to the System Data Repository (step 3 of Figure 3).” Ex.1005, 3. As illustrated in Fig. 2 below, the hardware monitors collect host data related to “CPU” and “memory” and network data related to “throughout” and “latency” (“*host and network resources*”). Ex.1005, Fig. 2.



Ex.1005, Fig. 2 (annotated); Ex.1003, ¶106.

Welch also discloses a second function group in the form of software monitors. Ex.1005, 3. “The reallocations made by the resource manager make use of information provided by the hardware and software monitors.” Ex.1005, 3. As shown in Fig. 2 below, the RTCS applications provide information to the software monitors, including “Sub-path latency,” “Path latency,” and “Profile information.”

The software monitors then report information to the Resource Manager as
“Unhealthy sub-paths,” “Trend of load,” and “Pathid.” Ex.1005, Fig. 2.

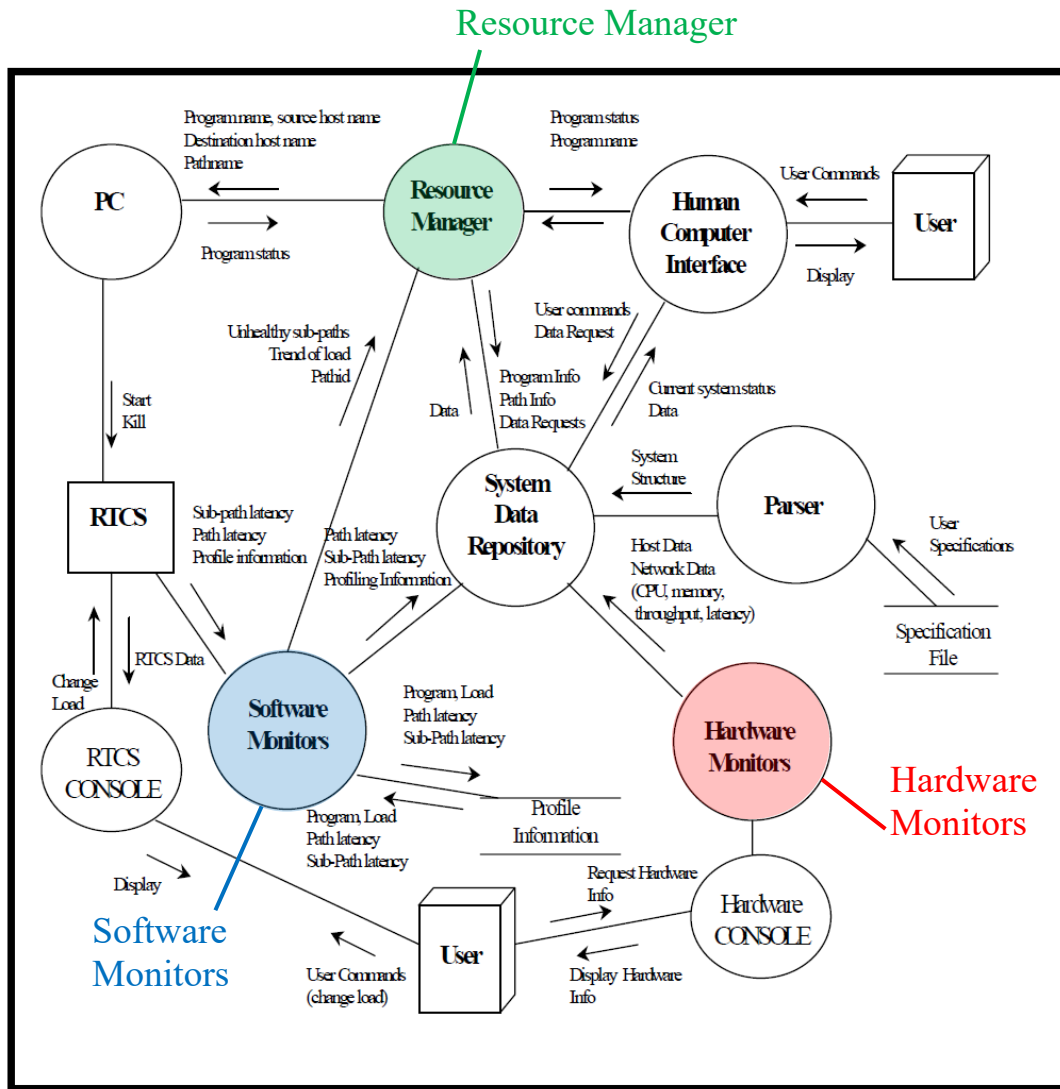


Ex.1005, Fig. 2 (annotated); Ex.1003, ¶107.

Welch further discloses a third function group in the form of a resource manager. Ex.1005, 3. “The core component of the middleware is the resource manager. It is activated when programs die and when time-constrained control paths miss their deadlines. In response to these events, it takes appropriate

measures to improve the quality of service delivered to the applications.” Ex.1005,

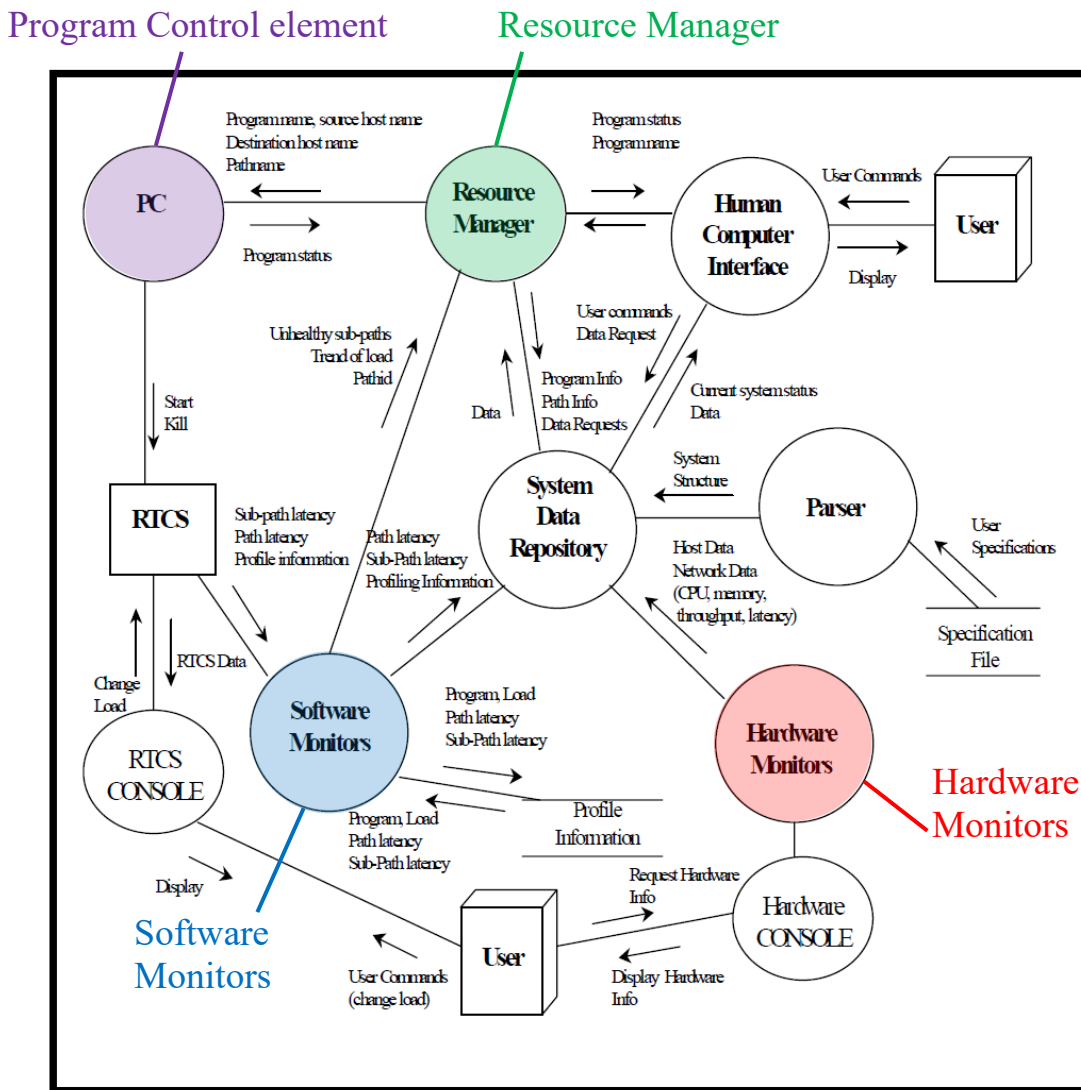
3. The resource manager is shown in Fig. 2 below.



Ex.1005, Fig. 2 (annotated); Ex.1003, ¶108.

Welch also discloses a fourth function group in the form of a program control element. Ex.1005, 2-3. “The program control (PC) component consists of a central control program and a set of startup daemons. When resource manager

needs to start a program on a particular host, it informs the control program, which notifies the startup daemon on that host.” Ex.1005, 3. The program control element is illustrated in Fig. 2 below.



Ex.1005, Fig. 2 (annotated); Ex.1003, ¶109.

Accordingly, Welch’s authors—of which some are inventors of the ’743 patent—had previously publicly disclosed the elements in the Challenged Claims.

Welch is analogous art to the '743 patent because it is directed to the same field of endeavor (distributed computing) as the '743 patent. *Compare* Ex.1001, Abstract (“A resource manager for a distributed environment”) *with* Ex.1005, title (“Distributed, Scalable, Dependable Real-Time Systems: Middleware Services and Applications”). In addition, Welch is reasonably pertinent to a problem addressed by the inventors of the '743 patent because Welch describes the same resource management techniques for improving efficiency and scalability that are purportedly solved by the '743 patent. *See* Ex.1005, 1. The '743 patent likewise purports to address inefficient use of resources. *See* Ex.1001, 2:15-20 (seeking to “provide a system that is both flexible and adaptive (or at least easily adapted) to changes in system configuration, performance bottlenecks, survivability requirements, scalability, etc.”); Ex.1003, ¶¶103-12.

2. Ravindran

Welch references other whitepapers (e.g., Ravindran) that were authored by some of the same authors (and some of the inventors of the '743 patent). These whitepapers describe various aspects of Welch’s distributed system in further detail.

For example, Welch refers to a “path-based real-time subsystem (see [11]).” Ravindran (Ex.1006) is Paper 11 in Welch. Ravindran, for example, also describes

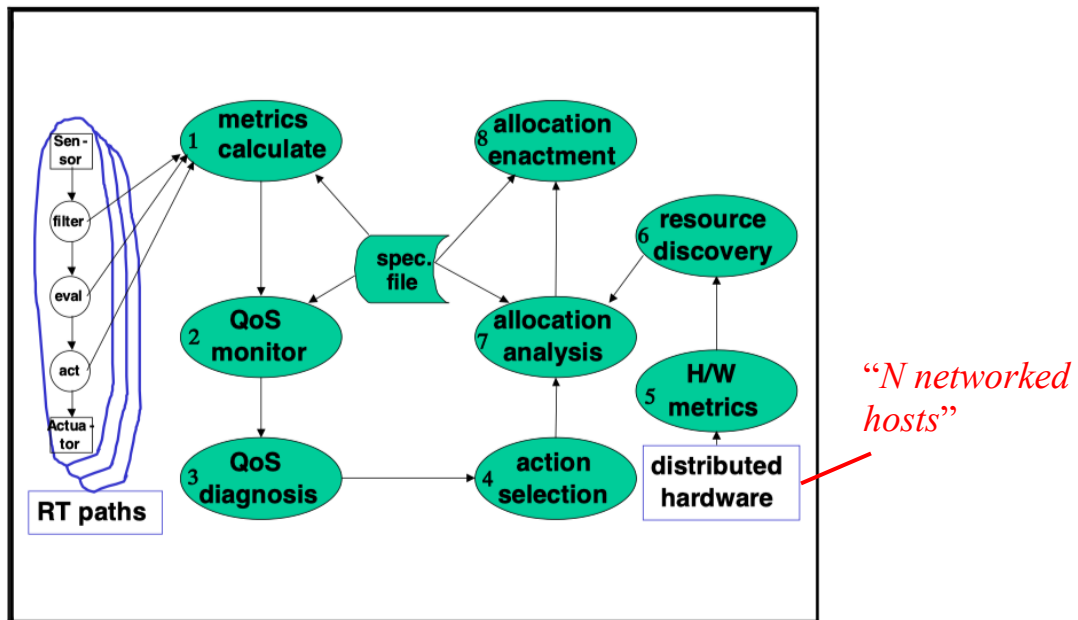
hardware and software monitors for monitoring the performance of applications and hosts. Ex.1006, 7; Ex.1003, ¶¶113-14.

3. Claim 10.

[10.0] *Software stored on at least one host for converting N networked hosts into a resource managed system instantiating M managed characteristic application computer programs, each managed characteristic application computer program managed by one of the N networked hosts, the software comprising:*

To the extent the preamble is limiting, Welch discloses or at least renders obvious [10.0]. Ex.1003, ¶117.

First, Welch describes QoS management software (“*Software*”) that manages the resources of a distributed system (“*software...for converting N networked hosts into a resource managed system*”). Welch “presents middleware services that support such dynamic real-time systems through adaptive resource management.” Ex.1005, 1. Welch’s middleware is embodied as “QoS management software.” Ex.1005, 2-3. Welch’s Fig. 1, below, illustrates the “logical architecture” of the QoS management software (“*Software*”). Ex.1005, 2; Ex.1003, ¶¶118-22.

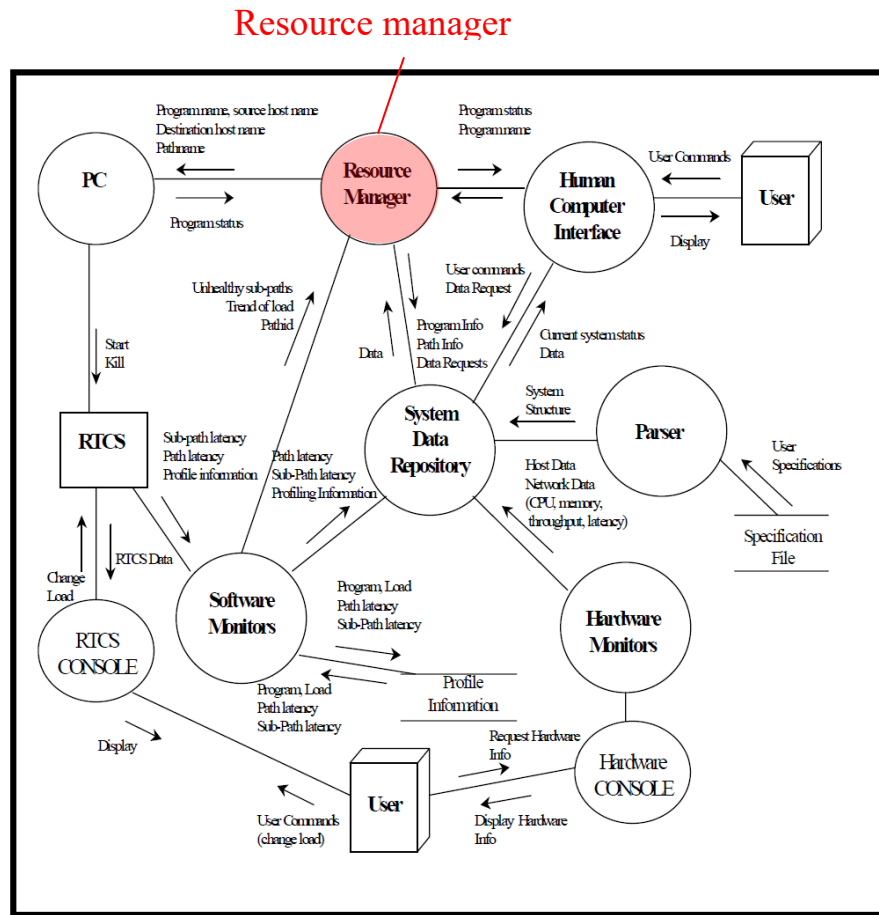


Ex.1005, Fig. 1 (annotated); Ex.1003, ***.

As illustrated in Fig. 1, one logical function of the QoS management software is to receive hardware metrics from “distributed hardware.” Ex.1005, Fig. 1. The distributed hardware represents “*N networked hosts*” because the QoS management software “recommend[s] actions (e.g., move[s] a program to a **different host or LAN** ...)” which indicates that the “distributed hardware” includes multiple hosts interconnected by at least one LAN (Large Area Network). Ex.1005, 2. Welch explains that “the allocation analysis component ... consults resource discovery **for host and LAN** load index metrics.” Ex.1005, 2. Welch further explains that “[t]he Resource Man[a]ger needs to decide on which **host(s) and LAN(s)** the unhealthy sub-path needs to be replicated or moved. This decision is made by choosing the **host(s) and LAN(s)** with the smallest load indices.”

Ex.1005, 3. Accordingly, Welch’s QoS management software manages “*N networked hosts*” because the software manages applications across multiple hosts and LANs of distributed hardware. Ex.1003, ¶¶123-25.

The QoS software’s resource management of the distributed hardware (“*N networked hosts*”) makes that distributed hardware a resource managed system (“*software ... for converting N networked hosts into a resource managed system*”). Indeed, Welch explains that “the core component of the middleware [QoS management software] is **the resource manager**.” Ex.1005, 2. Welch illustrates the “resource manager” in Fig. 2, below, which provides additional detail as to the “physical architecture” of the QoS management software. Ex.1005, Fig. 2; Ex.1003, ¶¶126-27.



Ex.1005, Fig. 2 (annotated); Ex.1003, ¶128.

Accordingly, the QoS management software (“*Software*”) includes a resource manager that “takes appropriate measures to improve the quality of service delivered to the applications.” Ex.1005, 2. Welch explains that “[t]he reallocations made by the resource manager make use of information provided by the hardware and software monitors, as well as from a specification file that describes QoS requirements and the structures of the software system and the hardware system.” Ex.1005, 2. The resource manager of the QoS management software thus makes the distributed system a resource managed system

(“software...for converting N networked hosts into a resource managed system”).

Ex.1003, ¶129.

Second, the QoS management software (“Software”) runs on multiple heterogeneous host platforms and is thus “*stored on at least one host.*” Welch explains that “**The software runs on** a heterogeneous network configuration that includes Myrinet, ATM, FDDI, and ethernet, on **multiple heterogeneous host platforms.**” Ex.1005, 4. Accordingly, in Welch, the host platform(s) storing and running the QoS management software is “*at least one host*” as claimed. Ex.1003, ¶¶130-31.

Third, the resource manager of the QoS management software starts up real-time control programs (“*instantiating M managed characteristic application computer programs*”). Ex.1005, 2-3. “When resource manager needs to start a program on a particular host, it informs the control program, which notifies the startup daemon on that host.” Ex.1005, 3. “The Resource Manager informs the Program Control to start the Real-Time Control System (RTCS) application programs on the specified hosts.” Ex.1005, 3. The RTCS application programs are “*instantiate[ed]*” because they are each instances of a scalable or replicable program. Ex.1005, 4 (“Scale time (T_s) is the time taken by the Program Manager and Startup Daemons to actually **start a new copy of the program.**”); Ex.1003, ¶133. Welch explains that these programs can be replicated. “If a path becomes

unhealthy (misses its deadline), the QoS Monitors detect such a condition, diagnose the cause of poor health, and suggest an action (such as moving or **replicating an application program**) to the Resource Manager (step 8 of Figure 3).” Ex.1005, 3; Ex.1003, ¶¶132-34.

Furthermore, the RTCS application programs are “*managed characteristic application computer programs*” because the resource manager manages at least their characteristics of scalability, survivability, fault tolerance or priority. Ex.1005, 2-4. This is consistent with how the ’743 patent describes a “managed characteristic.” Ex.1001, 54:20-21 (“managed characteristic may be one of scalability, survivability, fault tolerance or priority”). Welch explains that “[w]e performed experiments to determine the responsiveness of our QoS and resource management middleware for **survivability and scalability** services.” Ex.1005, 4. Indeed, the goal of Welch’s system is that “resources should be utilized in an efficient manner, and **scalability** must be provided to address the ever-increasing complexity of scenarios that confront such systems.” Ex.1005, 1. “Experiments show that the services provide bounded response times, **scalable services**, and low intrusiveness.” Ex.1005, 4. Accordingly, Welch’s real-time control system (RTCS) application programs are “*each managed characteristic application computer programs*” as claimed. Ex.1003, ¶¶135-36.

Fourth, the RTCS programs run on distributed hardware (*“each managed characteristic application computer program managed by one of the N networked hosts”*). Ex.1005, 3. Welch explains that “[t]he Resource Manager informs the Program Control to start the **Real-Time Control System (RTCS) application programs on the specified hosts.**” Ex.1005, 3. The resource manager manages applications by starting them on one of the networked hosts: “When resource manager needs to start **a program on a particular host**, it informs the control program, which notifies the startup daemon **on that host.**” Ex.1005, 3; Ex.1003, ¶¶137-38.

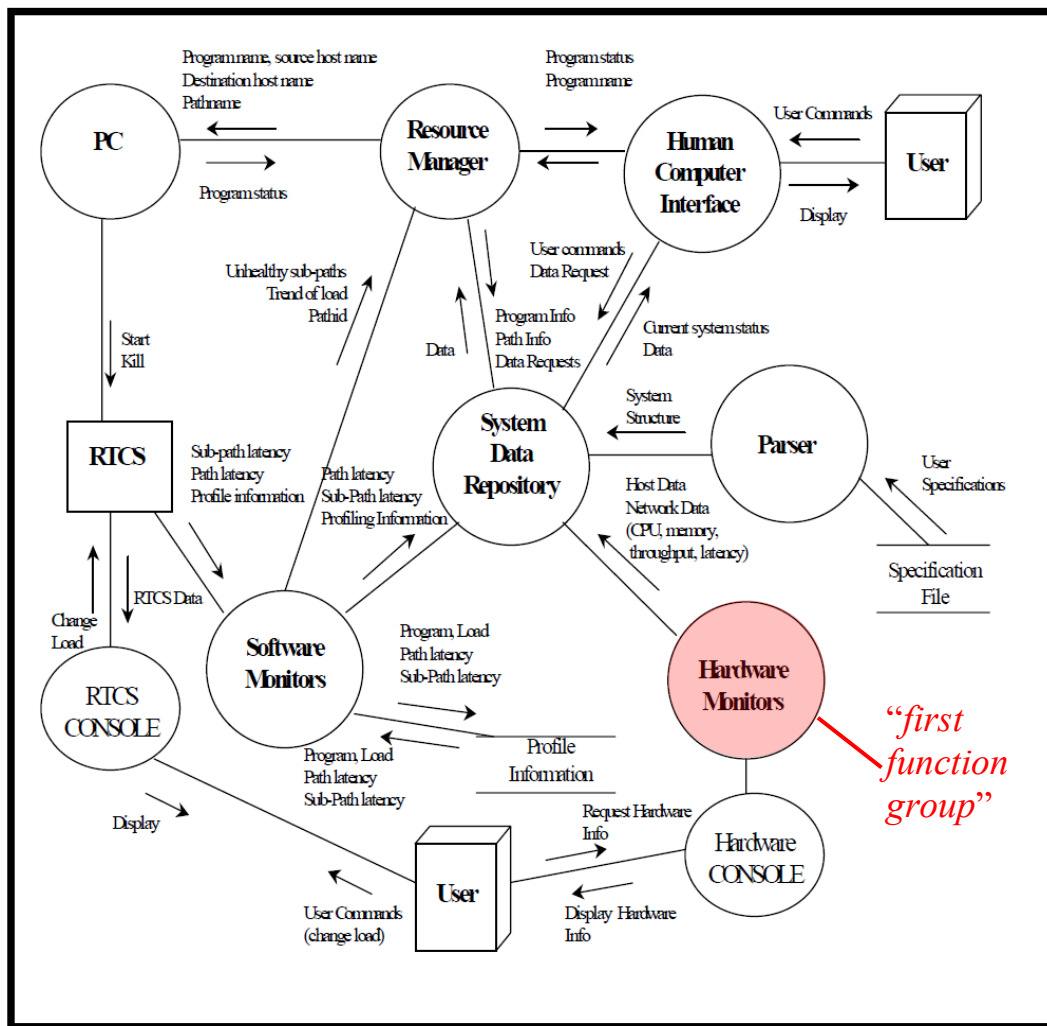
Thus, Welch describes QoS management software for managing real-time control system applications on distributed hardware to provide startup, replicating, and other actions related to providing for scalability and survivability, which discloses or renders obvious limitation [10.0]. Ex.1003, ¶¶117-140.

[10.1] a first function group which monitors the host and network resources;

Welch alone and in combination with Ravindran renders obvious limitation [10.1]. Ex.1003, ¶141.

Welch’s QoS management software (*“the software”*) includes “hardware monitors” (*“a first function group”*) that monitor host and network resources. Ex.1005, 2-3. “**Hardware (HW) Monitors** continuously observe a resource’s load index and pass this information to the Hardware Data Repository (step 2 of Figure

3), which in turn passes such information to the System Data Repository (step 3 of Figure 3).” Ex.1005, 3. Welch illustrates the hardware monitors for both host resources (one function) and network resources (another function) (“*first function group*”) in Fig. 2 below. Ex.1003, ¶142-43.



Ex.1005, Fig. 2 (annotated); Ex.1003, ¶143.

The hardware monitors (“*first function group*”) are also illustrated in Fig. 3 below. Ex.1003, ¶144.

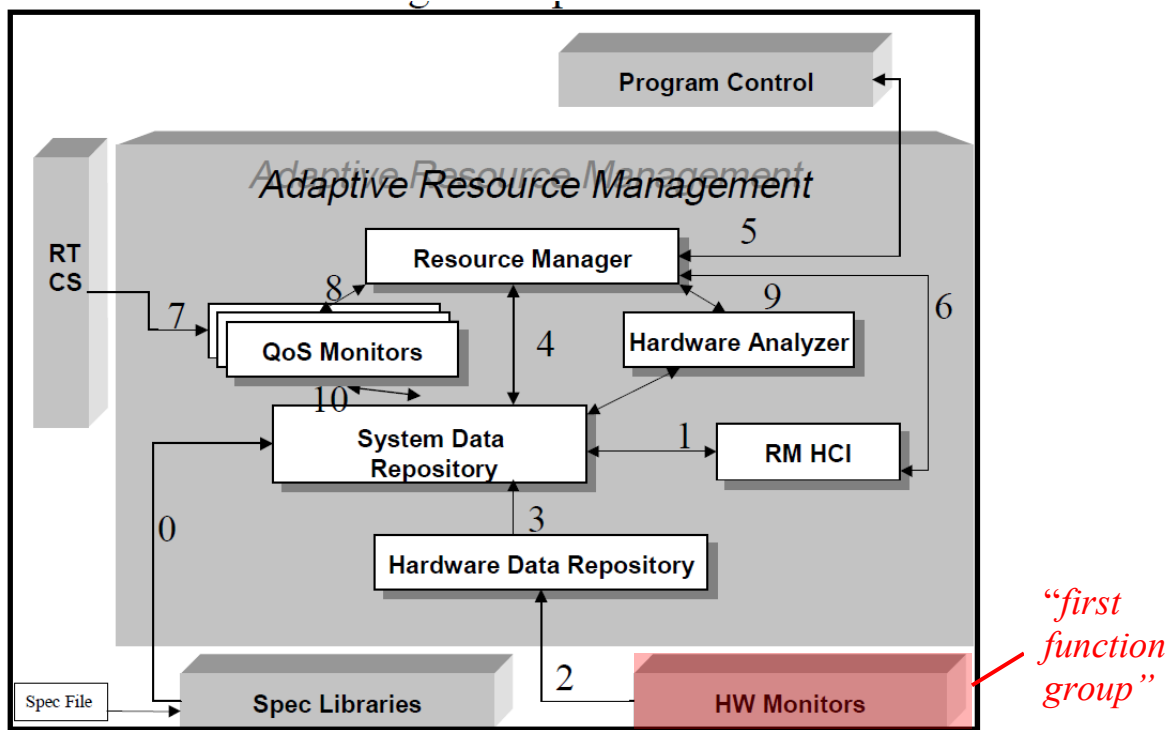
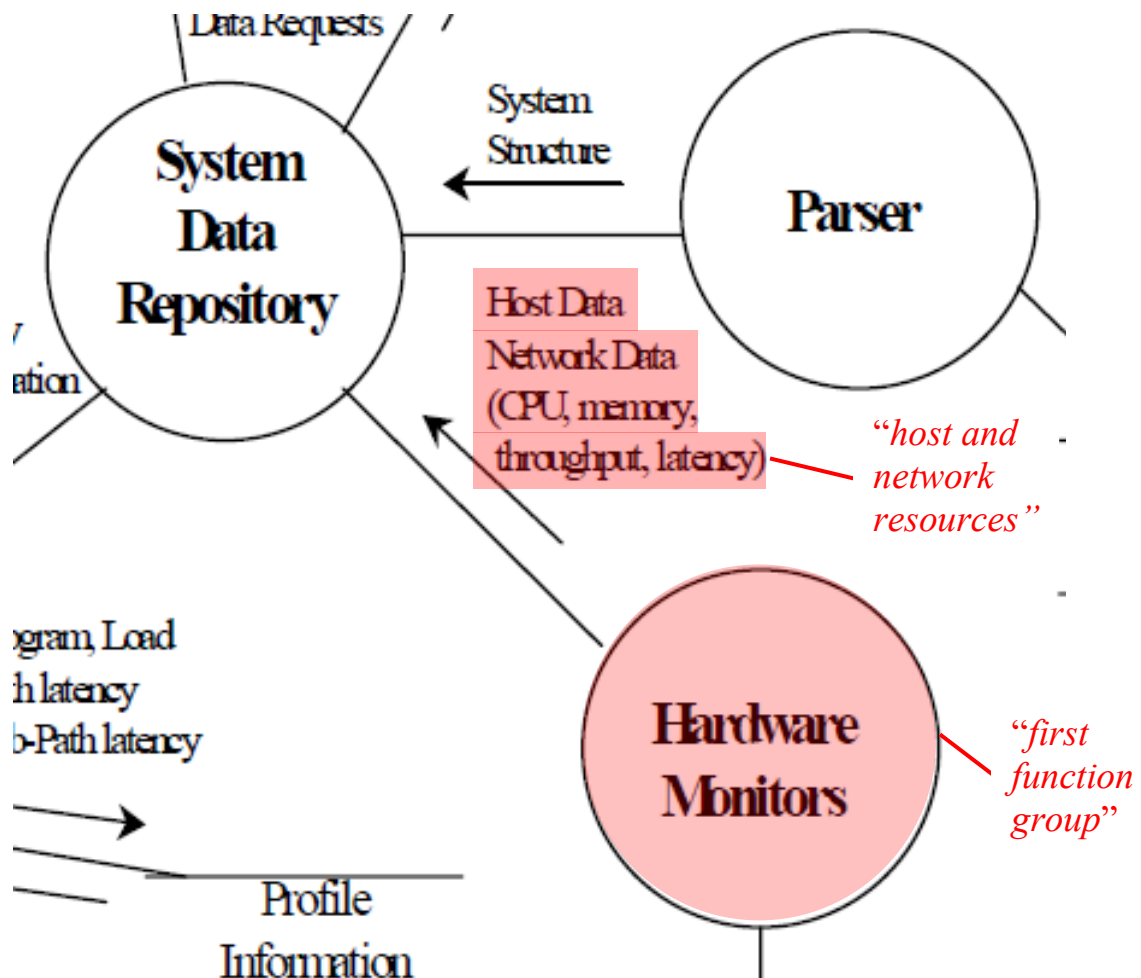


Figure 3. Adaptive resource management scenarios. Ex.1005, Fig. 3 (annotated); Ex.1003, ¶144.

As can be seen from a close-up view of Fig. 2 below, the hardware monitors collect and send to the repository “Host Data” related to “CPU” and “memory” and “Network Data” related to “throughput” and “latency” (“*host and network resources*”). Ex.1005, 2, Fig. 2; Ex.1003, ¶145.



Ex.1005, Fig. 2 (partial, annotated); Ex.1003, ¶145.

The host data and network data represent “*host and network resources*” because Welch refers to their collection as “network resource discovery” and “host resource discovery.” Ex.1005, 4. In particular, Welch describes the process of testing the QoS software system. Ex.1005, 4. In that description, Welch refers to obtaining resource discovery of network-level metrics and host level metrics from the host monitors. “Network Resource Discovery (t32) is the time interval required

to **obtain network-level metrics** from the network controller. Host Resource Discovery (t33) is the time interval required to **obtain host-level metrics** from all eligible host monitors.” Ex.1005, 4; Ex.1003, ¶146.

Some of Welch’s authors published other papers that describe the hardware monitors in detail. For example, Ravindran (Ex.1006) describes hardware monitors as follows: **Hardware (HW) Monitors** (which run on each host) periodically **calculate load indices for hosts and LANS**, and pass this information to the Hardware Data Repository (HDR).” Ex.1006, 7; Ex.1003, ¶147.

A POSITA would have found it obvious for Welch’s hardware monitors to be implemented as described in Ravindran. Ex.1003, ¶115. Indeed, Ravindran shares many of the same authors as Welch and Welch explicitly cites Ravindran. It states that “[o]ur approach is based on the dynamic path paradigm. A path-based real-time subsystem (see [**Ravindran**]) typically consists of a detection & assessment path, an action initiation path and an action guidance path.” Ex.1005, 1. Accordingly, implementing Welch’s hardware monitors as they are described in Ravindran provides the predictable benefit of implementing the hardware monitors as the Welch authors intended. Ex.1003, ¶115.

The combination of Welch and Ravindran is merely the combination of prior art elements (Welch’s and Ravindran’s hardware monitors) according to known methods (for collecting host and LAN metrics) to yield predictable results

(providing the resource manager with real-time statuses). *See KSR Int’l v. Teleflex Inc.*, 550 U.S. 398, 416 (2007). Welch explicitly teaches, suggests, and/or motivates the combination by identifying Ravindran as related to its disclosed system. Ex.1005, 1; *see In re Kahn*, 441 F.3d 977, 986 (Fed. Cir. 2006) (obviousness may be established by combining or modifying prior art elements when there is a teaching, suggestion, or motivation to do so). A POSITA would have had a reasonable expectation of success in combining Welch and Ravindran because both relate to the same type of path-based system, and Welch expressly references the conventional implementation details in Ravindran. *See* Ex.1005, 1, 5 (Ravindran is document [11] referenced in Welch); Ex.1006, 1-7; Ex.1003, ¶116.

Accordingly, the system described by Welch and Ravindran includes hardware monitors, which include one sub-function of monitoring host resources, and another sub-function of monitoring network resources. Ex.1005, 2-3. Together, these sub-functions form “*a first function group*” as claimed. Ex.1003, ¶148.

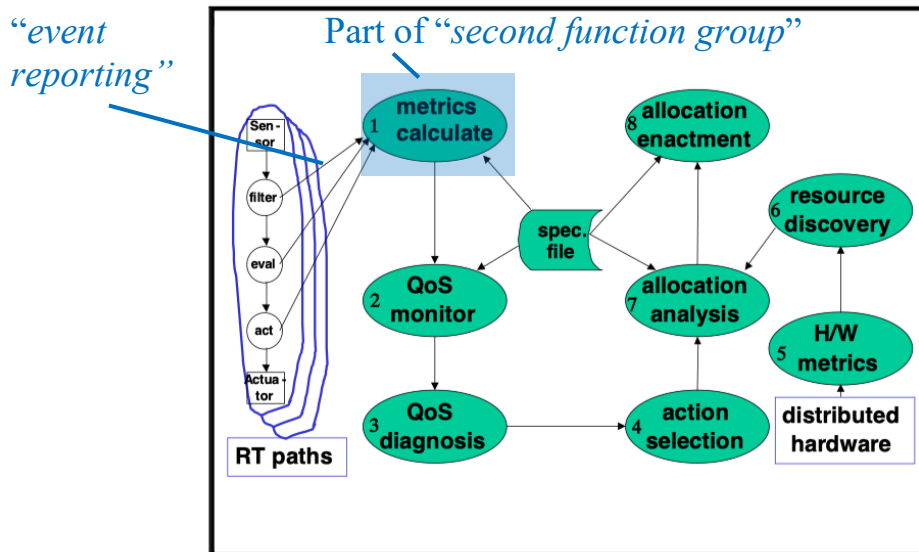
Thus, because Welch describes QoS software that includes hardware monitors for monitoring host and network resources, Welch’s QoS software system including hardware monitors, as taught by Ravindran, renders obvious limitation [10.1]. Ex.1003, ¶¶141-50.

[10.2] *a second function group which provides application computer program event reporting and event correlation capabilities;*

Welch discloses or at least renders obvious limitation [10.2]. Ex.1003, ¶151.

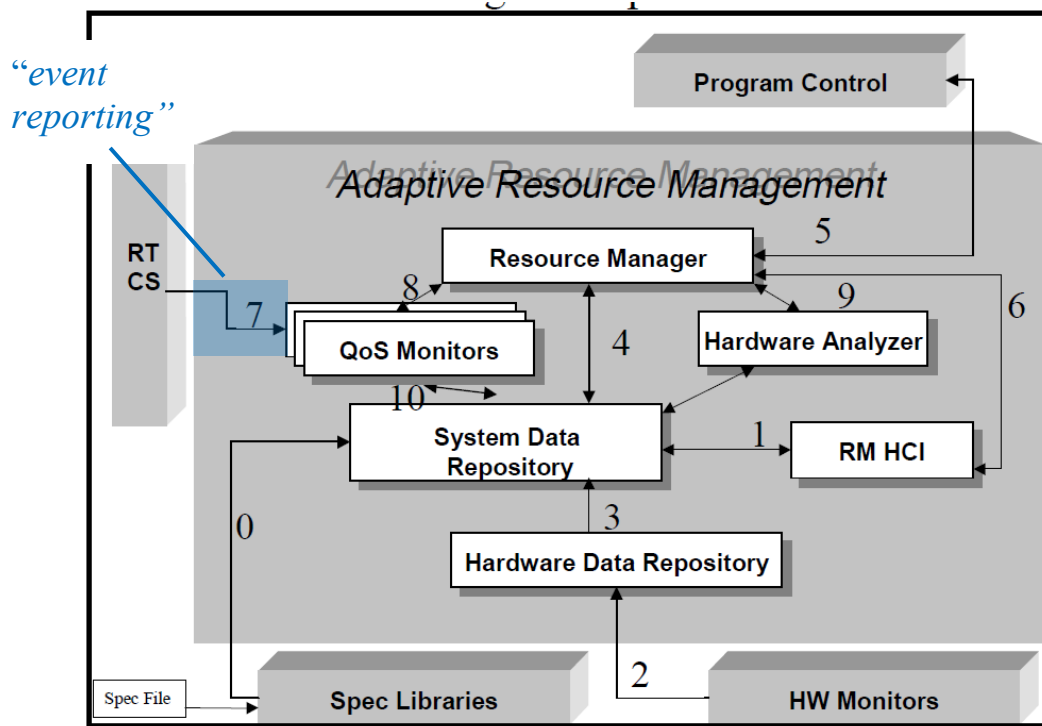
Welch's QoS management software includes software monitors and a metrics calculation component that together teach a "*second function group*," as shown in Fig. 1, below. Ex.1005, 2-3. The part of the QoS management software corresponding to the "*second function group*" includes the functionality of providing "*application computer program event reporting and event correlation capabilities*." Ex.1005, 2; Ex.1003, ¶152.

Welch's QoS management software provides "*application computer program event reporting*" because the "application programs of real-time control paths send timestamped events to the metrics calculation component, which calculates path-level QoS metrics and **sends them to the QoS monitor.**" Ex.1005, 2; Ex.1003, ¶153.



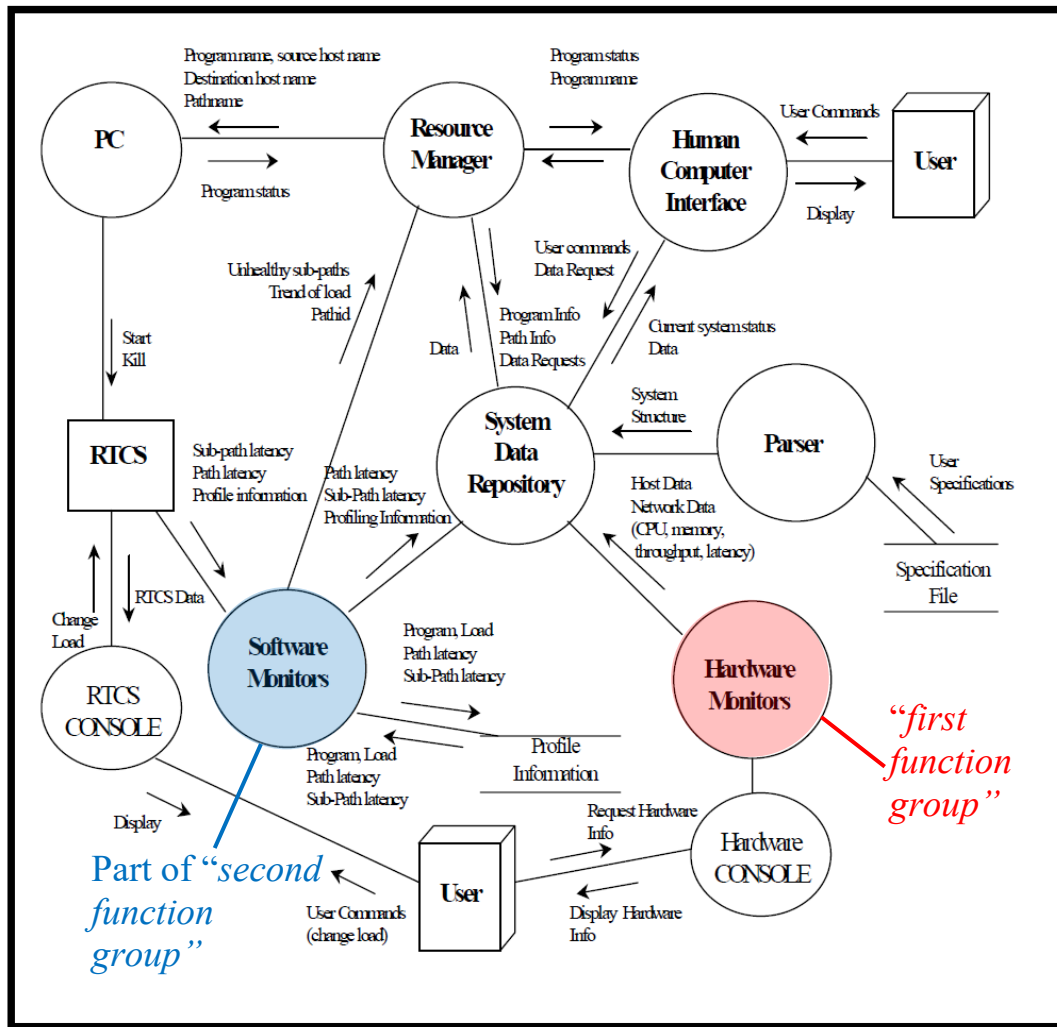
Ex.1005, Fig. 1 (annotated); Ex.1003, ¶152.

This event reporting is also described and illustrated in Fig. 3. “The RTCS continuously sends the application profile (event time stamps or program/path latencies) information to QoS Monitors (step 7 of Figure 3).” Ex.1005, 3; Ex.1003, ¶154.



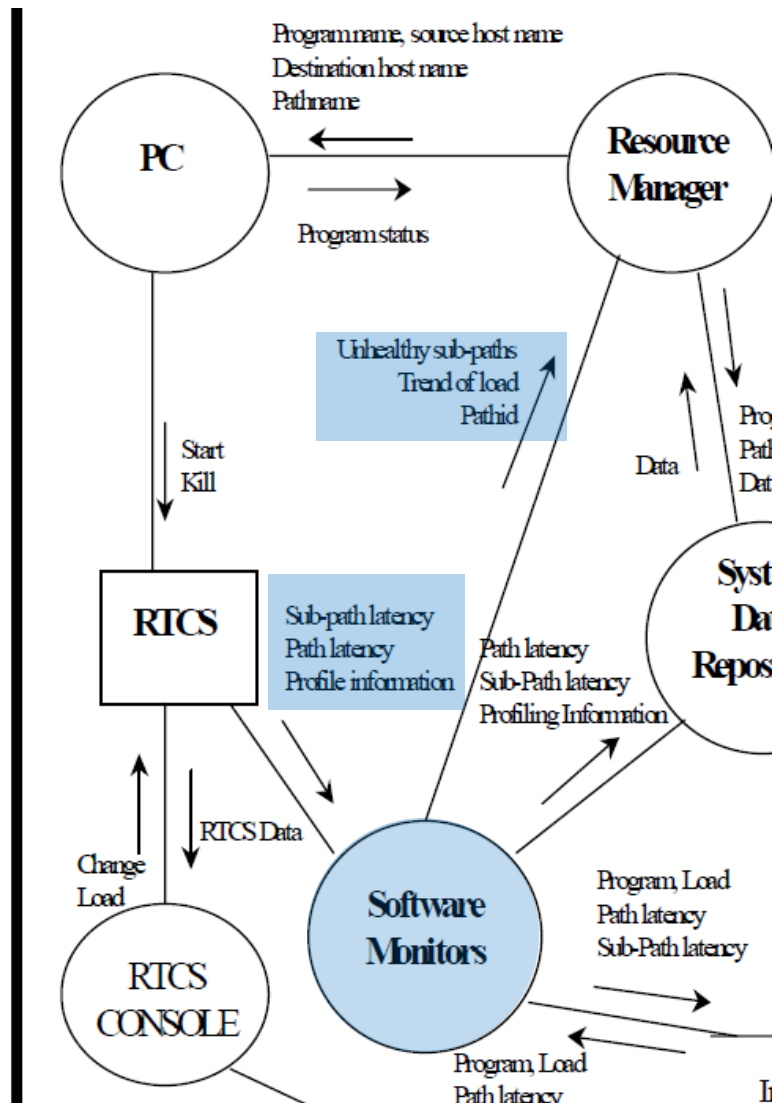
Ex.1005, Fig. 3 (annotated); Ex.1003, ¶154.

Welch explains that the time-stamped events, path latencies, and other pieces of information are provided from the applications to the QoS monitors through “Software Monitors,” as shown in Fig. 2 below. Ex.1005, 2; Ex.1003, ¶155.



Ex.1005, Fig. 2 (annotated); Ex.1003, ¶155.

As shown below, in a close-up of Fig. 2, the RTCS applications provide information to the software monitors, including “Sub-path latency,” “Path latency,” and “Profile information.” Ex.1005, Fig. 2. The software monitors then provide “*application computer program event reporting*” because they report information to the Resource Manager as “Unhealthy sub-paths,” “Trend of load,” and “Pathid.” Ex.1005, Fig. 2.



Ex.1005 Fig. 2 (partial, annotated); Ex.1003, ¶156.

Accordingly, Welch's QoS management software includes software monitors that receive time-stamped events, path latencies, profiles information, and other pieces of information for reporting to a QoS monitor. Ex.1005, 2-3; Ex.1003, ¶157.

Welch's metrics calculation component (part of the "second function

group”) provides “*event correlation capabilities*” because the QoS management software “implementation includes the following **capabilities**:...(2) track **correlation** and filtering algorithms.” Ex.1003, 4. Moreover, given that Welch’s “applications are dependent (communicate with each other),” the performance information of those applications includes “*event correlation capabilities*.” Ex.1005, 2; Ex.1003, ¶¶158-59. Welch’s software monitors and metrics calculation component aggregate the information received from the real time control systems (RTCS) through the software monitors: “The application programs of real-time control paths send timestamped events to the metrics calculation component, which calculates path-level QoS metrics and sends them to the QoS monitor.” Ex.1005, 2; *see also* Ex.1005, 3 (“The reallocations made by the resource manager make use of information provided by the hardware and **software monitors**.”). This is similar to how correlation capabilities are described in the ’743 patent. *See* Ex.1001, 7:51-54 (“The system also provides grammar-driven capabilities for correlating, combining, and reformatting application data into higher-level metrics (composite events) for use by displays or other Resource Management components.”). Ex.1003, ¶160.

Accordingly, Welch describes software monitors that receive and report events, which is one sub-function. Welch also describes a metric calculation component, which is another sub-function that provides aggregated data from the

software applications to the QoS monitor. These two sub-functions work together to collect and compile data related to the RTCS applications and thus together render obvious “*a second function group.*” Ex.1005, 2-3; Ex.1003, ¶¶161-62.

Thus, because Welch’s QoS system includes software monitors and a metrics calculation component that report events and provide aggregated information to a QoS monitor, Welch renders obvious limitation [10.2]. Ex.1003, ¶¶151-64.

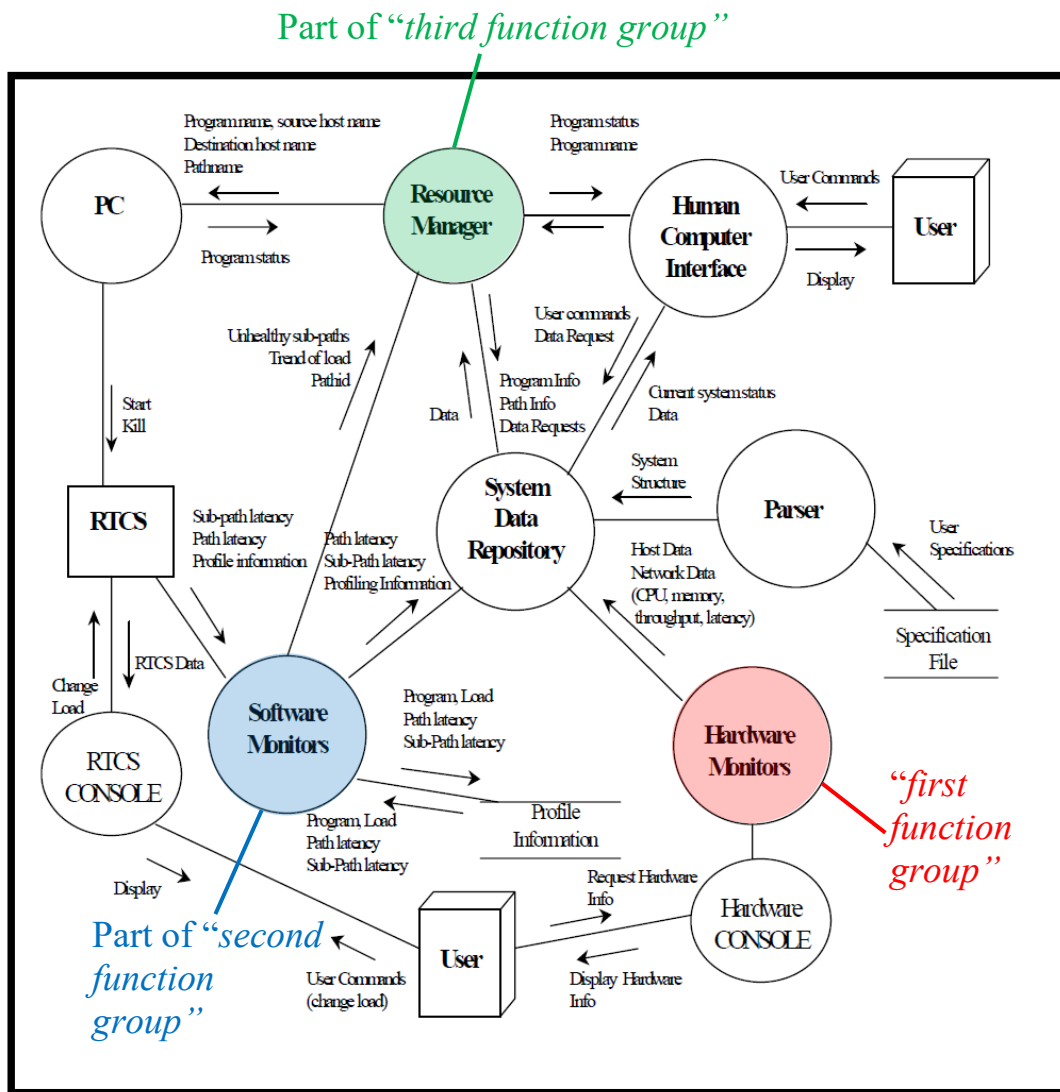
[10.3] *a third function group which provides reasoning and decision making capabilities for the resource managed system,*

Welch discloses or at least renders obvious limitation [10.3]. Ex.1003, ¶165.

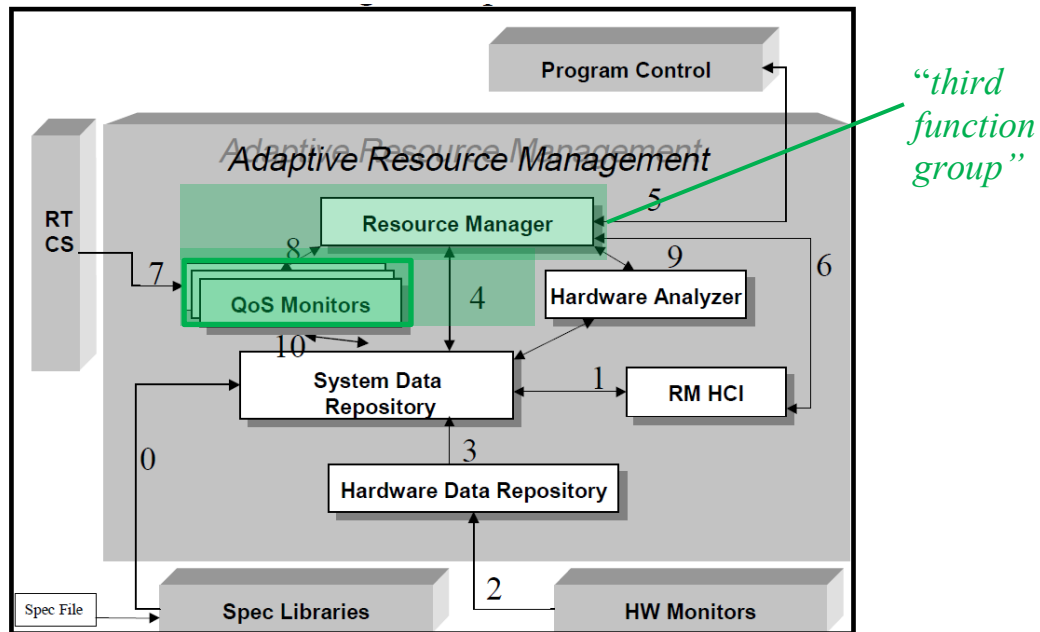
First, the “resource manager” and QoS monitors of Welch’s QoS management software perform functions related to managing host and application resources to ensure QoS requirements are made, and thus together teach “*a third function group,*” as shown in Fig. 2, below. Ex.1003, ¶166. Welch’s resource manager (“*third function group*”) includes functionality that “*provides reasoning and decision making capabilities for the resource managed system.*” Ex.1005, 2-3. As an example, the resource manager in Welch’s QoS management software provides “*reasoning*” and “*decision-making capabilities*” in conformance to QoS requirements. Ex.1005, 2-3; Ex.1003, ¶¶166-67.

Second, Welch’s resource manager includes *reasoning...capabilities*

because it includes an “[a]ction selection [that] ranks the recommended actions, identifies redundant actions, and forwards the results to the allocation analysis component.” Ex.1005, 2; Ex.1003, ¶168.



Ex.1005, Fig. 2 (annotated); Ex.1003, ¶167.



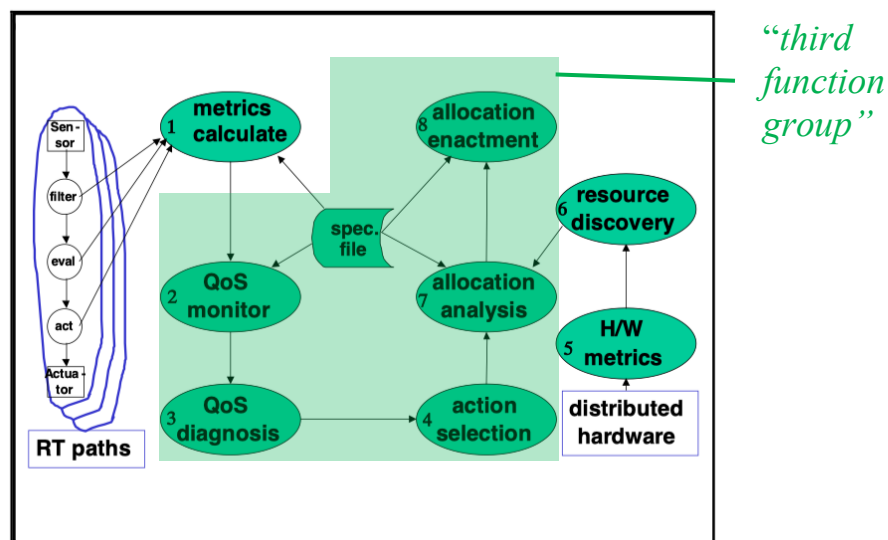
Ex.1005, Fig. 1 (annotated); Ex.1003, ¶167.

The resource manager also includes “*decision making capabilities*” because it “consults resource discovery for host and LAN load index metrics, **determines** a good way to allocate the hardware resources in order to perform the actions, and requests the actions be performed by the allocation enactment component.”

Ex.1005, 2. Additionally, the resource manager decides which actions to take to improve QoS. Ex.1005, 2-3. The resource manager “is activated when programs die and when time-constrained control paths miss their deadlines. **In response to these events, [the resource manager] takes appropriate measures to improve the quality of service delivered to the applications.**” Ex.1005, 2. “The Resource Manger **needs to decide** on which host(s) and LAN(s) the unhealthy sub-path needs to be replicated or moved. This **decision** is made by choosing the host(s) and

LAN(s) with the smallest load indices.” Ex.1005, 3. The resource manager is illustrated in Fig. 2 below. Ex.1003, ¶¶169-71.

Welch illustrates specific functions that are performed by the Resource Manager in Fig. 3. These specific functions will be discussed with respect to limitations [10.4]-[10.6] below. Ex.1003, ¶172.



Ex.1005, Fig. 1 (annotated); Ex.1003, ¶172.

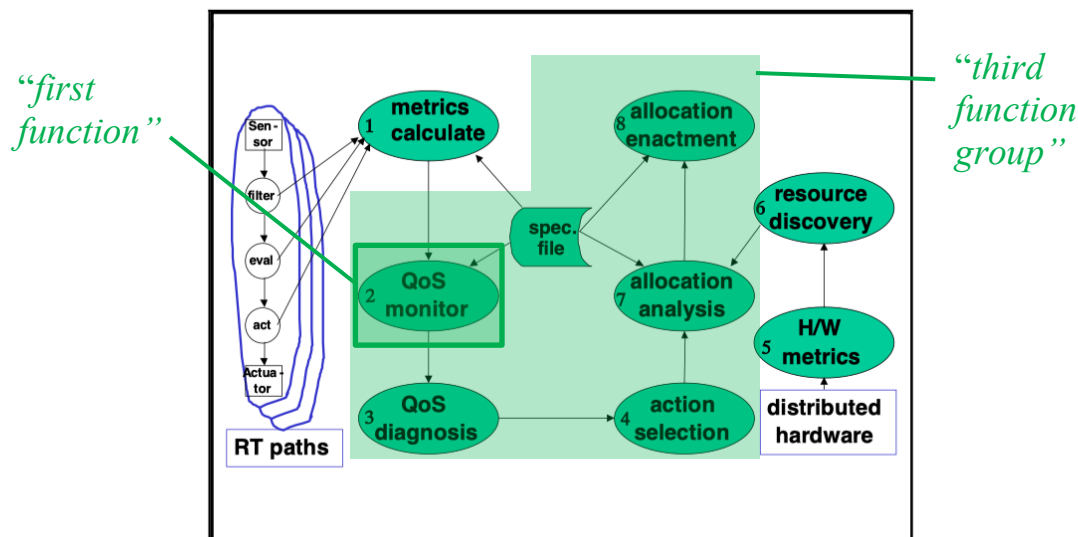
Thus, because Welch’s QoS management software includes a resource manager that ranks recommended actions and determines a good way to allocate hardware resources, Welch renders obvious limitation [10.3]. Ex.1003, ¶¶165-74.

[10.4] wherein the third function group comprises: a first function which determines a state and health of the N hosts, a network operatively coupling the N hosts to one another and the M managed characteristic application computer programs in the distributed environment;

Welch discloses or at least renders obvious limitation [10.4]. Ex.1003, ¶175.

Welch’s resource manager (“third function group”) includes a QoS monitor

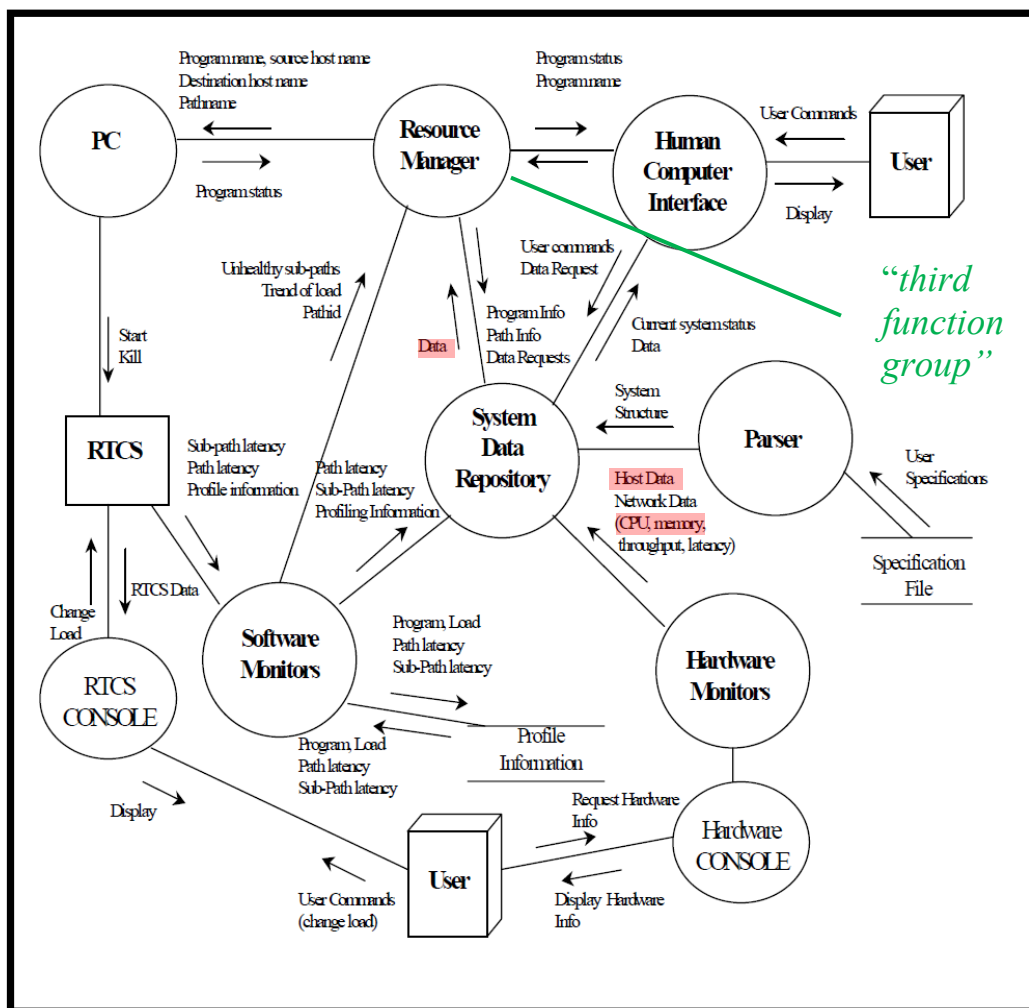
(“*first function*”) that determines the “*state and health*” of the hosts, network, and applications (“*state and health of the N hosts*”). Ex.1003, ¶176. Specifically, Welch discloses that “[t]he application programs of real-time control paths send timestamped events to the metrics calculation component, which calculates path-level QoS metrics and sends them to the QoS monitor. The monitor checks for conformance of observed QoS to required QoS, and notifies the QoS diagnosis component when a QoS violation occurs.” Ex.1005, 2. The QoS monitor is shown below in Fig. 2. Ex.1003, ¶177.



Ex.1005, Fig. 1 (annotated); Ex.1003, ¶177.

First, the QoS monitor (“*first function*”) of “*the third function group*,” as shown in Fig. 1, above, “*determines a state and health of the N hosts.*” Ex.1003, ¶178. Consistent with the discussion above at [10.1], Welch discloses that the resource manager receives reports from the hardware monitors for various metrics

such as the status of the CPU or memory (“*state*” of a host). Ex.1005, 3. “The reallocations made by the resource manager make use of information provided by the hardware and software monitors.” Ex.1005, 3. As shown in Fig. 2 below, the resource manager (“*third function group*”) receives data from the repository, which includes the CPU and memory data received from the hardware monitors. Ex.1005, Fig. 2.



Ex.1005, Fig. 2 (annotated); Ex.1003, ¶179.

FG40...advantageously receives operating **system-level statuses** and statistics for each host A-N from the History Server(s) FG12A-FG12N, respectively. This information can be employed **for calculating CPU**, network, **memory** ...

Ex.1001, 35:12-18; Ex.1003, ¶182.

The '743 patent further provides a list of types of information that represent the state and health of the hosts: “The functions associated with this functional group employ information (listed below) to (1) determine the state and health of the distributed environment (hosts, networks, and applications).” Ex.1001, 17:9-39. This information includes host loads.

System Specifications:

- Host configuration and capabilities
- Application capabilities
 - Survivability
 - Scalability
- Potential hosts to run on
- Application startup and shutdown dependencies
- Application and path performance requirements

Program Control:

- Application statuses
- Detected application faults
- Detected host failures
- Detection of new host
- Operator initiated requests
 - Resolution of application startup or shutdown dependencies
 - Selection of application-to-host mappings

History Servers:

- Host statuses, configuration, **and loads**
- Network link statuses and loads

Remos Network Data Broker:

- Network link statuses and loads

Instrumentation Subsystem:

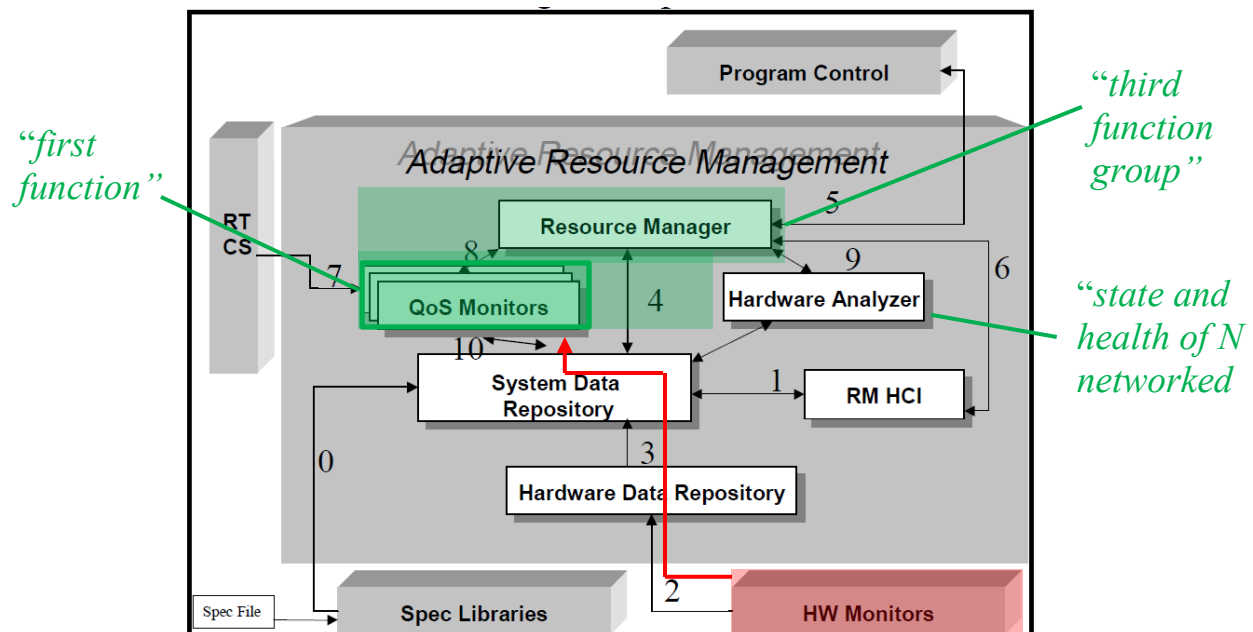
- Application performance information

Readiness Display:

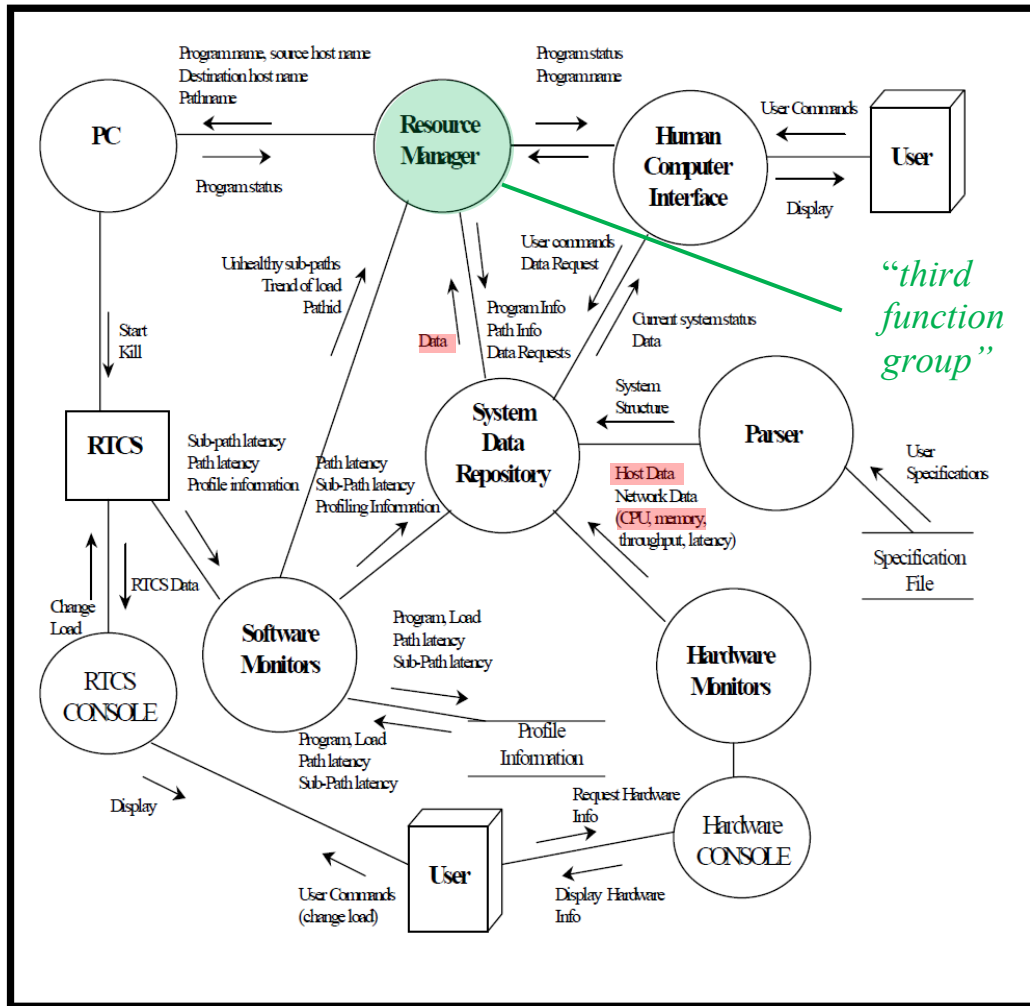
- Run-time changes to application system priorities

Accordingly, just like the '743 patent, which describes the health and state of a host as the CPU status, memory status, and host load, Welch describes that the QoS monitor (*“first function”* of the *“third function group”*) receives similar data and thus *“determines a state and health of the N hosts.”* Ex.1003, ¶¶183-84.

Second, Welch’s QoS monitor (*“first function”*) *“determines a state and health of ... a network operatively coupling the N hosts to one another.”* As explained above at [10.1], the hardware monitors report “network data,” which includes throughput (*“state”*) and latency (*“health”*). This information is ultimately received by the QoS monitor within the resource manager (*“third function group”*). Ex.1005, 3; Ex.1003, ¶185.



Ex.1005, Fig. 1 (annotated); Ex.1003, ¶185.



Ex.1005, Fig. 2 (annotated); Ex.1003, ¶186.

Third, the QoS monitors (“*first function*”) of Welch’s resource manager (“*third function group*”) receive status information from the real-time applications (“*determines a state and health of ... the M managed characteristic application computer programs in the distributed environment*”). Ex.1005, 2-3; Ex.1003, ¶187. In particular, the resource manager receives information from the software monitors as to the status and health of RTCS applications, as described above at

[1.2]. “The reallocations made by the resource manager make use of information provided by the hardware and **software monitors**, as well as from a specification file that describes QoS requirements and the structures of the software system and the hardware system.” Ex.1005, 2. From the data received from the software monitors, the QoS monitor (“*first function*”) of the resource manager determines the state and health of the RTCS applications. Ex.1005, 2. Ex.1003, ¶188.

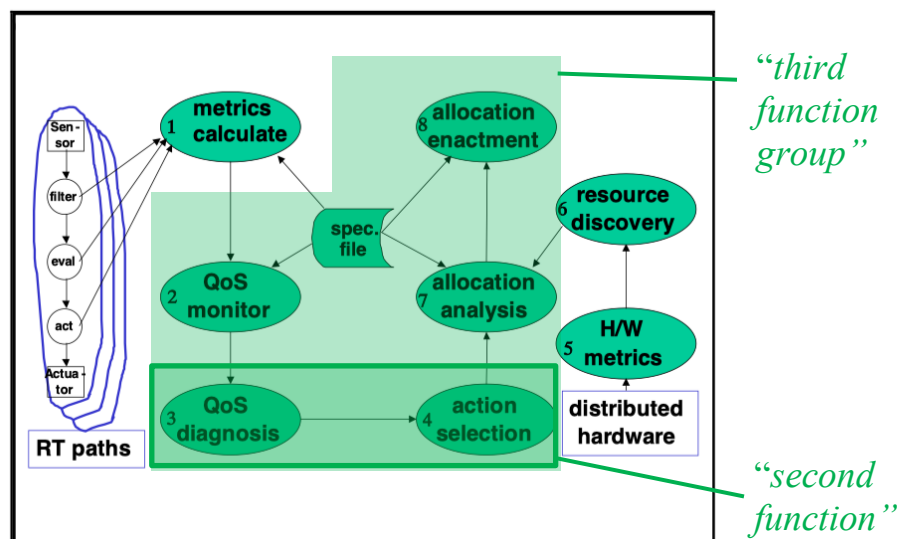
The resource manager determines the state of the applications through time stamp and path latency information. Ex.1005, 2-3. “The RTCS continuously sends the application profile (time stamps or program/path latencies) information to QoS Monitors (step 7 of Figure 3).” Ex.1005, 3. The resource manager determines whether the paths (i.e., applications) are healthy or not as well. “**If a path becomes unhealthy (misses its deadline), the QoS Monitors detect such a condition,** diagnose the cause of poor health, and suggest an action (such as moving or replicating an application program) to the Resource Manager (step 8 of Figure 3).” Ex.1005, 3; Ex.1003, ¶¶189-90.

Thus, because Welch describes a QoS monitor that determines the status and health of the hosts, network, and real-time applications, Welch renders obvious limitation [10.4]. Ex.1003, ¶¶175-92.

[10.5] *a second function which determines required allocation and reallocation actions needed to maintain a plurality of Quality of Service (QoS) requirements established for the M managed characteristic application computer programs, the QoS requirements dictating parameters regarding service quality of the M management characteristic application programs; and*

Welch discloses or at least renders obvious limitation [10.5]. Ex.1003, ¶193.

First, Welch’s resource manager (“*third function group*”) includes “QoS diagnosis” and “action selection” components (a “*second function*”) that allocate and reallocate resources to meet QoS requirements (“*determines required allocation and reallocation actions needed to maintain a plurality of Quality of Service (QoS) requirements established for the M managed characteristic application computer programs*”). Ex.1005, 2-3; Ex.1003, ¶194.



Ex.1005, Fig. 1 (annotated); Ex.1003, ¶194.

Welch’s resource manager “*determines required allocation and reallocation actions*” because it includes an “allocation enactment component.” Ex.1005, 2.

And, “[t]he **reallocations** made by the resource manager make use of information provided by the hardware and software monitors, as well as from a specification file that describes **QoS requirements and the structures of the software system and the hardware system.**” Ex.1005, 2; Ex.1003, ¶¶195-96.

Welch’s QoS management software determines the actions “*needed to maintain a plurality of Quality of Service (QoS) requirements*” because it includes a QoS diagnosis component. Ex.1005, 2. “The application programs of real-time control paths send timestamped events to the metrics calculation component, which calculates path-level QoS metrics and sends them to the **QoS monitor.**” Ex.1005, 2. “**The monitor** checks for conformance of observed **QoS to required QoS**, and notifies the QoS diagnosis component when a QoS violation occurs.” Ex.1005, 2; *see also* Ex.1005, 1 (“dynamic QoS-based resource managers strive to meet the **QoS requirements** of each application they manage.”); Ex.1003, ¶197.

The QoS diagnoses are used by the action selection component. Ex.1005, 2-3. “QoS diagnosis notifies the action selection component of the cause(s) of poor QoS and recommends actions (e.g., move a program to a different host or LAN, shed a program, or replicate a program) to improve QoS.” Ex.1005, 2. “Action selection ranks the recommended actions, identifies redundant actions, and forwards the results to the allocation analysis component.” Ex.1005, 2.

Accordingly, allocation and reallocation actions described above are done to

maintain QoS “*for the M managed characteristic application computer programs*” by performing the allocation-reallocation actions in accordance with those requirements. Ex.1003, ¶198.

Second, Welch discloses that the “*QoS requirements dictat[e] parameters regarding service quality of the M management characteristic application programs.*” Ex.1005, 2-3. Welch describes using QoS objectives related to timeliness and dependability (“*parameters*”). Welch explains that “real-time systems have rigorous, multidimensional Quality-of-Service (QoS) objectives. They must behave in a dependable manner, respond to threats in a timely fashion and provide continuous availability within hostile environments.” Ex.1005, 1. Thus, dependability, timeliness, and availability are examples of a plurality of QoS requirements regarding the quality of the RTCS applications. Ex.1003, ¶199.

Again, “[t]he **reallocations** made by the resource manager make use of information provided by the hardware and software monitors, as well as from a specification file that describes **QoS requirements and the structures of the software system and the hardware system.**” Ex.1005, 2. “If a path becomes unhealthy (misses its deadline), the QoS Monitors detect such a condition, diagnose the cause of poor health, and suggest an action (such as moving or replicating an application program) to the Resource Manager (step 8 of Figure 3).” Ex.1005, 3. Accordingly, the functions of Welch’s software that allocate and

reallocate resources to match QoS requirements are thus a “*second function*” as claimed. Ex.1003, ¶200.

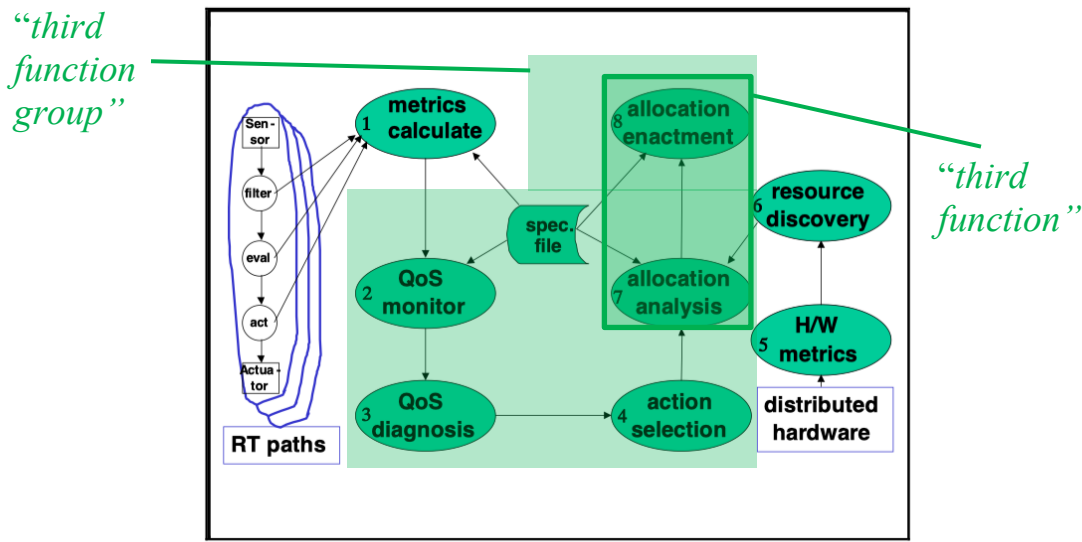
Thus, because Welch’s resource manager includes a QoS diagnosis and action selection components that make allocation and reallocation decisions to maintain QoS requirements, Welch renders obvious limitation [10.5]. Ex.1003, ¶¶193-202.

[10.6] *a third function which generates automatic control signal requests corresponding to the actions dictated by the QoS requirements, such that the managed characteristic application computer programs are moved, shutdown, and started in accordance with satisfaction of the QoS requirements; and*

Welch discloses or at least renders obvious limitation [10.6]. Ex.1003, ¶203.

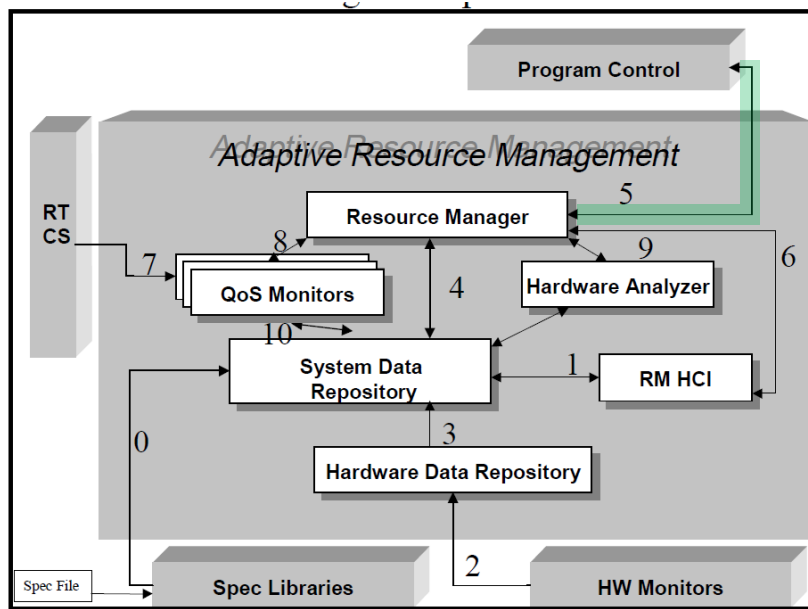
First, as shown in Fig. 1, below, Welch’s resource manager (“*third function group*”) includes “allocation analysis” and “allocation enactment” components (“*third function*”) that automatically notify a Program Control to make changes to the applications in response to QoS violations (“*generates automatic control signal requests corresponding to the actions dictated by the QoS requirements.*”).

Ex.1003, ¶204.



Ex.1005, Fig. 1 (annotated); Ex.1003, ¶204.

Welch’s allocation enactment component “consults resource discovery for host and LAN load index metrics, determines a good way to allocate the hardware resources in order to perform the actions, and **requests the actions be performed by the allocation enactment component.**” Ex.1005, 2. “Once a host is selected, the Resource Manger [sic] notifies the Program Control to make the change (step 5 of Figure 3) and updates the RM HCI accordingly (step 6 of Figure 3).” Ex.1005, 3; Ex.1003, ¶205.



Ex.1005, Fig. 3 (annotated); Ex.1003, ¶205.

Second, Welch teaches that the actions performed at the direction of the allocation enactment component include replicating, moving, terminating, and starting programs (“*such that the managed characteristic application computer programs are moved, shutdown, and started in accordance with satisfaction of the QoS requirements.*”). “QoS diagnosis notifies the action selection component of the cause(s) of poor QoS and recommends actions (e.g., move a program to a different host or LAN, shed a program, or replicate a program) to improve QoS.” Ex.1005, 2. “The Resource Manger needs to decide on which host(s) and LAN(s) the unhealthy sub-path needs to be replicated or moved. Ex.1005, 3. “Once a host is selected, the Resource Manger notifies the Program Control to make the change (step 5 of Figure 3).” Ex.1005, 3.

Additionally, “[t]he Resource Manager informs the Program Control to start

the Real-Time Control System (RTCS) application programs on the specified hosts (step 5 of Figure 3).” Ex.1005, 3. “The Program Control **starts the programs** and informs the Resource Manger [sic] accordingly (step 5 of Figure 3).” Ex.1005, 3. “When resource manager needs to start a program on a particular host, it informs the control program, which notifies the startup daemon on that host.” Ex.1005, 3; Ex.1003, ¶206. Accordingly, Welch’s allocation enactment component of the resource manager moves, replicates, shed, and starts programs (“*the managed characteristic application computer programs are moved, shutdown, and started*”). Moreover, these actions are performed “to improve QoS.” (“*in accordance with satisfaction of the QoS requirements*”). Ex.1005, 2; Ex.1003, ¶¶208-11.

Thus, because Welch’s resource manager uses a program control to move, shed, or startup copies of the RTCS applications to maintain QoS requirements, Welch renders obvious limitation [10.6]. Ex.1003, ¶¶203-13.

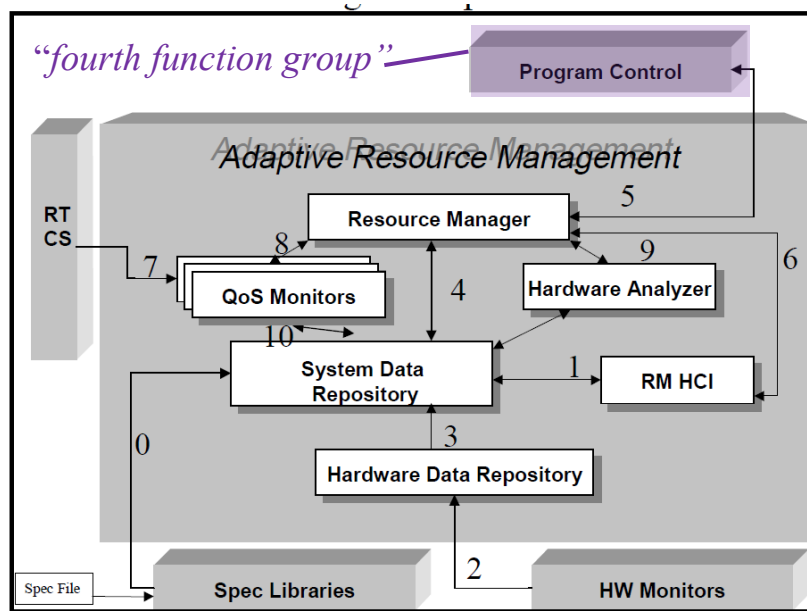
[10.7] a fourth function group which provides program control capabilities permitting starting, stopping, and configuring of selected ones of the M managed characteristic application computer programs on respective ones of the N hosts in the resource managed system,

Welch discloses or at least renders obvious limitation [10.7]. Ex.1003, ¶214.

Welch describes a program control component (“*fourth function group*”) that “*provides program control capabilities permitting starting, stopping, and configuring of*” RTCS applications as explained below. As described above at

[10.1], Welch describes a number of RTCS applications that run on distributed hardware of a resource managed system (“*M managed characteristic application computer programs on respective ones of the N hosts in the resource managed system.*”); Ex.1005, 3; Ex.1003, ¶¶215-16.

Welch’s program control is illustrated in Fig. 3 below.



Ex.1005, Fig. 3 (annotated); Ex.1003, ¶217.

The program control can “start” and “configure” a program on a particular host. Ex.1005, 3. “The program control (PC) component consists of a central control program and a set of startup daemons. When resource manager needs to **start** a program on a particular host, it informs the control program, which notifies the startup daemon on that host.” Ex.1005, 3. “The Resource Manager needs to decide on which host(s) and LAN(s) the unhealthy sub-path needs to be **replicated**

or moved.” Ex.1005, 3. “The HCI provides information to the user regarding the system configuration, application status ...” Ex.1005, 3. The software can also “shed a program” (*stop*) an application. Ex.1005, 2; Ex.1003, ¶218.

Welch’s Program Control thus starts, stops, and configures “*selected ones of the M managed characteristic application computer programs on respective ones of the N hosts in the resource managed system.*” Ex.1005, 2-3. This is because “[t]he Resource Manger finds the host(s) and LAN(s) with the smallest load” and “then re-starts the terminated program on that host **by informing the Program Control** (step 5 of Figure 3) and RM HCI (step 6 of Figure 3) accordingly.” Ex.1005, 3. The applications are started, stopped, or configured on “*respective*” hosts in that they are started, stopped or configured on the “specified hosts.” Ex.1005, 3; Ex.1003, ¶219.

Thus, because Welch’s program control is responsible for starting, stopping, moving, and configuring the programs that run on the plurality of hosts, Welch renders obvious limitation [10.7]. Ex.1003, ¶¶214-21.

[10.8] *where M and N are positive integers and where M may be equal to, greater than, or less than N.*

Welch renders obvious limitation [10.8]. Ex.1003, ¶222.

A POSITA would have recognized that the language “where M may be equal to, greater than, or less than N” merely states that M and N are independent

variables. Ex.1003, ¶¶223-24. A POSITA would have recognized that the N number of hosts in Welch (*see* [10.0]) and the M number of applications (*see* [10.0]) in Welch are positive¹, independent variables because Welch describes that applications can be shed, started, replicated and moved, which dictates that there is not a 1:1 correspondence between hosts and applications. *See supra* [10.0]; Ex.1005, 1 (“Traditional load balancing systems assume *independent* jobs.”); Ex.1005, 2-3. Indeed, it would be impractical to install or remove a new physical system each time a new application is started or shutdown. Ex.1003, ¶226. Accordingly, because Welch describes adding, shedding, moving, and replicating applications on distributed hardware (independent of the number of hosts), a POSITA would have understood that in Welch, “*M and N are positive integers and where M may be equal to, greater than, or less than N*” as claimed.

Thus, because Welch describes systems in which the number of applications is independent from the number of hosts, Welch renders obvious limitation [10.8]. Ex.1003, ¶¶222-28.

Additionally, a POSITA would have found it obvious for Welch’s RTCS applications to scale independently of the number of heterogeneous hosts for at least the reasons described below with respect to Ground 2. Ex.1003, ¶¶225-26.

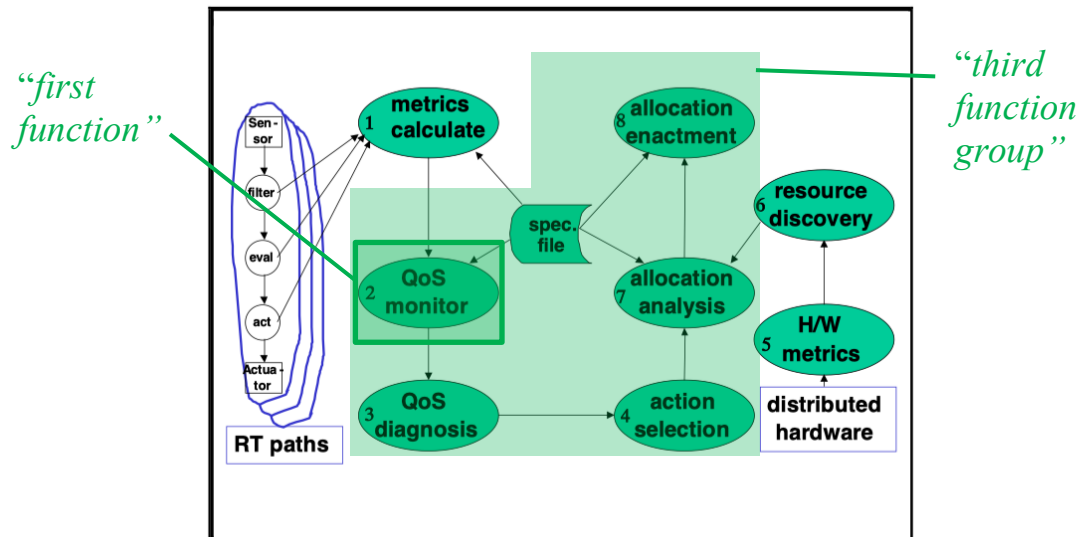
¹ A negative number of hosts or applications is nonsensical. Ex.1003, ¶223.

4. Claim 11.

[11.0] *The software as recited in claim 10, wherein the first function receives system specification information comprising host configuration and capabilities.*

First, as described above at [10.4], Welch teaches a QoS monitor (“*first function*”). Ex.1003, ¶¶229-30.

Second, the QoS monitor (“*first function*”) “*receives specific information about host configuration and capabilities.*” Ex.1005, 2-3. With reference to Fig. 1, below, Welch’s QoS monitor (“*first function*”) receives QoS metrics: “The application programs of real-time control paths send timestamped events to the metrics calculation component, which calculates path-level QoS metrics and sends them to the QoS monitor. The monitor checks for conformance of observed QoS to required QoS, and notifies the QoS diagnosis component when a QoS violation occurs.” Ex.1005, 2; Ex.1003, ¶231.



Ex.1005, Fig. 1 (annotated); Ex.1003, ¶231.

The QoS metrics include “*specific information about host configuration and capabilities*” in the form of the status of the CPU or memory (“*configuration and capabilities*” of a host). Welch explains that the Resource Manager receives data from the hardware monitors: “The reallocations made by **the resource manager make use of information provided by the hardware** and software **monitors.**”

Ex.1005, 3. As shown in Fig. 2 below, the Resource Manager (“*third function group*”) receives data from the repository, which includes the CPU and memory data received from the hardware monitors. Ex.1005, Fig. 2; Ex.1003, ¶¶232-34.

5. Claim 14.

[14.0] *The software as recited in claim 10, wherein the first function receives program control information comprising application status and detected application faults for each of the M managed characteristic application computer programs, and detected failures regarding the N hosts.*

Welch discloses or at least renders obvious limitation [14.0]. Ex.1003, ¶237.

First, as described above at [10.4], Welch teaches a first function that “determines a state and health of the N hosts...and the M managed characteristic application computer programs in the distributed environment.” Ex.1005, 2-3; Ex.1003, ¶238.

Second, the process of determining the state and health of the M applications includes “receiv[ing] program control information comprising application status.” Ex.1005, 2-3. Welch’s resource manager receives information from the software monitors as to the health of RTCS applications, which are described as “paths.” Ex.1005, 2-3. From that data, the QoS monitor (“*first function*”) of the resource manager determines the state and health of the RTCS applications: “The reallocations made by the resource manager make use of information provided by the hardware and software monitors, as well as from a specification file that describes QoS requirements and the structures of the software system and the hardware system.” Ex.1005, 2. The resource manager determines the state of the applications through time stamp and path latency

information (“*application status*”): “The RTCS continuously sends the application profile (time stamps or program/path latencies) information to QoS Monitors.”

Ex.1005, 3; Ex.1003, ¶¶239-40.

Third, the process of determining the state and health of the M applications includes detecting application faults, which teaches “*receiv[ing] program control information comprising...detected application faults for each of the M managed characteristic application computer programs.*” Ex.1005, 2-3. “If a sub-path (RTCS program) is terminated due to some hardware/software failure, Program Control detects such a condition and informs the Resource Manager accordingly (step 5 of Figure 3).” Ex.1005, 3. “If a path becomes unhealthy (misses its deadline), the QoS Monitors detect such a condition, diagnose the cause of poor health, and suggest an action (such as moving or replicating an application program) to the Resource Manager (step 8 of Figure 3). Ex.1005, 3; Ex.1003, ¶241.

Welch’s resource manager also receives information as to application failures: “If a sub-path (RTCS program) is terminated due to some hardware/software failure, Program Control detects such a condition and informs the Resource Manager accordingly (step 5 of Figure 3).” Ex.1005, 3. “The Resource Manger finds the host(s) and LAN(s) with the smallest load indices by querying the Host analyzer (step 9 of Figure 3).” It then re-starts the terminated program on that host by informing the Program Control.” Ex.1005, 3; Ex.1003,

¶242.

Fourth, the process of determining the state and health of the N hosts includes detecting hardware failures *receiv[ing] program control information comprising...detected failures regarding the N hosts.*” If a sub-path (RTCS program) is terminated due to some **hardware**/software **failure**, Program Control detects such a condition and informs the Resource Manager accordingly (step 5 of Figure 3).” Ex.1005, 3; Ex.1003, ¶243.

Thus, because the QoS monitor of Welch’s resource manager receives information as to application and host statuses, faults (e.g., missing a deadline), and failures (e.g., terminations), Welch renders obvious limitation [14.0]. Ex.1003, ¶¶237-45.

6. Claim 16.

[16.0] *The software as recited in claim 10, wherein the first function receives application performance data representing each one of the M managed characteristic application computer programs.*

Welch discloses or at least renders obvious limitation [16.0]. Ex.1003, ¶246.

First, as described above at [10.4], Welch teaches a first function that “*determines a state and health of the N hosts, a network operatively coupling the N hosts to one another and the M managed characteristic application computer programs in the distributed environment.*” Ex.1003, ¶247.

Second, the process of determining the state and health of the N hosts

includes “*receiv[ing] application performance data representing each one of the M managed characteristic application computer programs.*” Welch explains that the “RTCS [applications] continuously sends the application profile (time stamps or program/path latencies) information to QoS Monitors (step 7 of Figure 3).”

Ex.1005, 3. The program/path latencies are “performance data” as claimed. And Welch further states that “dynamic QoS-based resource managers strive to meet the QoS requirements of each application they manage.” Ex.1005, 1; Ex.1003, ¶248.

Thus, because Welch’s QoS monitors receive performance data in the form of program/path latencies, Welch renders obvious limitation [16.0]. Ex.1003, ¶¶246-50.

7. Claim 18.

[18.0] *The software as recited in claim 10, wherein the second function which determines the required allocation and reallocation actions established for the M managed characteristic application computer programs by; and*

For the reasons explained above at [10.5], Welch teaches a “*second function which determines the required allocation and reallocation actions established for the M managed characteristic application computer programs.*” Ex.1003, ¶¶251-52.

[18.1] *responding to application and host failures by determining if and what recovery actions should be taken;*

First, the action selection and QoS diagnosis components of Welch's resource manager ("*second function*") responds to a QoS violation ("*responding to application and host failures*") by moving, shedding, or replicating a program ("*by determining if and what recovery actions should be taken*"). Ex.1005, 2-3. "The monitor checks for conformance of observed QoS to required QoS, and notifies the QoS diagnosis component when a QoS violation occurs. QoS diagnosis notifies the action selection component of the cause(s) of poor QoS and recommends actions (e.g., move a program to a different host or LAN, shed a program, or replicate a program) to improve QoS." Ex.1005, 2. QoS violations may include host or application failures: "If a path becomes unhealthy (misses its deadline), the QoS Monitors detect such a condition, diagnose the cause of poor health, and suggest an action (such as moving or replicating an application program) to the Resource Manager (step 8 of Figure 3)." Ex.1005, Fig. 3.

Thus, because the action selection and QoS diagnosis components of Welch's resource manager respond to a QoS violation by moving, shedding, or replicating a program, Welch renders obvious limitation [18.1]. Ex.1003, ¶¶253-56.

[18.2] *determining if and where to place new copies of managed characteristic application computer programs or which managed characteristic application computer programs should be shutdown when QoS Manager functions indicate that scale up or scale down actions should be taken based on measured application performance and QoS specifications established for the M managed characteristic application computer programs;*

Welch renders both alternatives in limitation [18.2] obvious. Limitation [18.2] recites “*determining if and where to place new copies of managed characteristic application computer programs... or [determining] which managed characteristic application computer programs should be shutdown....*” Ex.1003, ¶¶257-58,

First, Welch teaches the first option of “*determining if and where to place new copies of managed characteristic application computer programs.*” Ex.1005, 2-3. “The Resource Manager informs the Program Control to start the Real-Time Control System (RTCS) application programs on the specified hosts (step 5 of Figure 3).” Ex.1005, 3. “The Resource Manger needs to decide on which host(s) and LAN(s) the unhealthy sub-path needs to be replicated or moved. This decision is made by choosing the host(s) and LAN(s) with the smallest load indices.” Ex.1005, 3; Ex.1003, ¶259.

Second, Welch teaches the alternative option: “*determining...which managed characteristic application computer programs should be shutdown when QoS Manager functions indicate that scale up or scale down actions should be*

taken based on measured application performance and QoS specifications established for the M managed characteristic application computer programs.”

Ex.1005, 2-3; Ex.1003, ¶260. “The monitor checks for conformance of observed QoS to required QoS, and notifies the QoS diagnosis component when a QoS violation occurs. QoS diagnosis notifies the action selection component of the cause(s) of poor QoS and recommends actions (e.g., move a program to a different host or LAN, shed a program, or replicate a program) to improve QoS.” Ex.1005, 3; Ex.1003, ¶260.

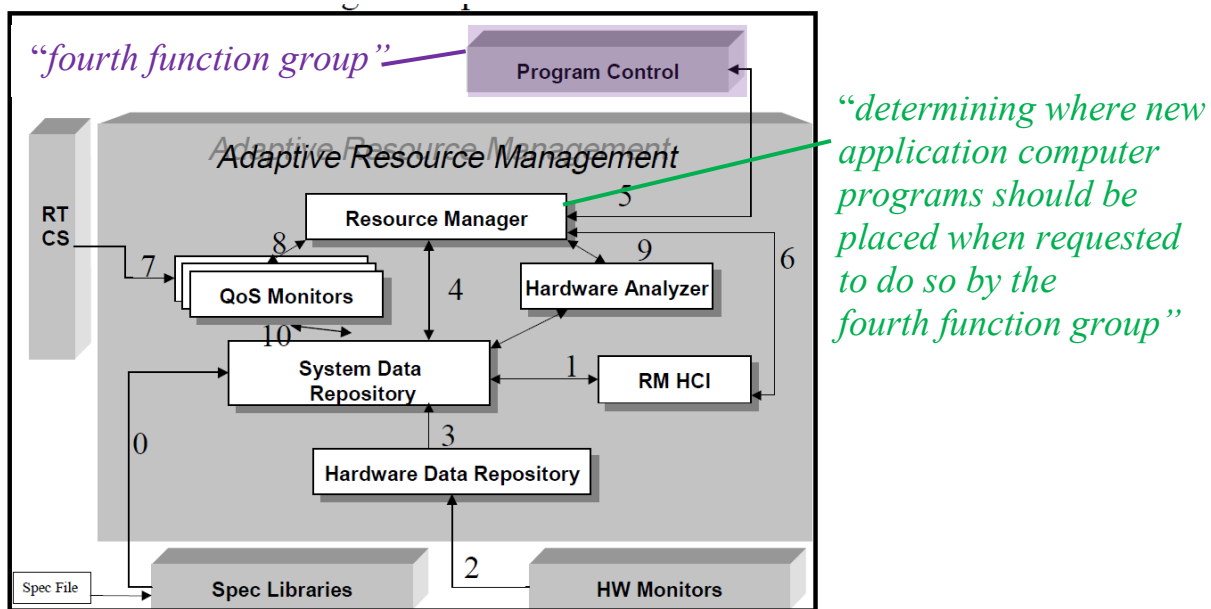
Thus, because Welch’s resource manager decides which applications to replicate, and where to place them, Welch renders obvious limitation [18.2].
Ex.1003, ¶¶257-63.

[18.3] *determining where new application computer programs should be placed when requested to do so by the fourth function group; and*

Welch’s resource manager determines where to place an application (*“determining where new application computer programs should be placed”*).

Ex.1005, 3. “The Resource Manger finds the host(s) and LAN(s) with the smallest load indices by querying the Host analyzer (step 9 of Figure 3).” Ex.1005, 3. “It then re-starts the terminated program on that host by informing the Program Control (step 5 of Figure 3) and RM HCI (step 6 of Figure 3) accordingly.”
Ex.1005, 3. This is done in response to being informed of a failure by the program

control (“when requested to do so by the fourth function group”). “If a sub-path (RTCS program) is terminated due to some hardware/software failure, Program Control detects such a condition and informs the Resource Manager accordingly (step 5 of Figure 3).” Ex.1005, 2-3; Ex.1003, ¶264.



Ex.1005, Fig. 3 (annotated); Ex.1003, ¶264.

Thus, because Welch’s program control detects a failure and the resource manager responsively determines where to restart the terminated program, Welch renders obvious limitation [18.3]. Ex.1003, ¶¶264-66.

[18.4] *determining which and how many application computer programs should run based on application system priorities.*

Welch’s resource manager decides which and how many applications should run based in part on a specification file that describes QoS specifications (“application system priorities”). “The System Data Repository loads the user spec

file “via the Spec Libraries, which consist of a compiler and data structures to store the compiled real-time system and **application QoS specifications** (step 0 of Figure 3).” Ex.1005, 3. “The System Data Repository sends the system information to Resource Management Human Computer Interface (RM HCI) for display purposes (step 1 of Figure 3).” Ex.1005, 3. “Hardware (HW) Monitors continuously observe a resource’s load index and pass this information to the Hardware Data Repository (step 2 of Figure 3), which in turn passes such information to the System Data Repository (step 3 of Figure 3).” Ex.1005, 3. “The Resource Manager receives the initial startup information from the System Data Repository, as specified by the spec file (step 4 of Figure 3).” Ex.1005, 3. “The Resource Manager informs the Program Control to start the Real-Time Control System (RTCS) application programs on the specified hosts (step 5 of Figure 3).” Ex.1005, 3. “The Program Control starts the programs and informs the Resource Manger accordingly (step 5 of Figure 3).” Ex.1005, 3; Ex.1003, ¶¶267-68.

Welch’s system also determines the placement of applications “*based on application system priorities*” in that it considers the load index. Ex.1005, 3. “The Resource Manger needs to decide on which host(s) and LAN(s) the unhealthy sub-path needs to be replicated or moved. This decision is made by choosing the host(s) and LAN(s) **with the smallest load indices**.” Ex.1005, 3. “The Hardware Analyzer ranks the hosts and LANs in ascending order of their load index and

passes this information to the Resource Manager (step 9 of Figure 3).” Ex.1005, 3; Ex.1003, ¶269.

Thus, because Welch’s resource manager places applications based on a spec file and load indices, Welch renders obvious limitation [18.4]. Ex.1003, ¶¶267-71.

D. Ground 2: Claims 10, 11, 14, 16, and 18 are obvious under 35 U.S.C. § 103 over Welch, Ravindran, and Hosokawa.

Hosokawa provides additional evidence as to the obviousness of limitation 10.8 and provides examples in which $M > N$, $M = N$, or $M < N$. In addition, to the extent that the “*at least one host*” on which the “*Software*” is stored is one of the “*N networked hosts*” managed by the software, it would have been obvious for Welch’s QoS management software to be stored on one of the “*N networked hosts*” as taught by Hosokawa. Ex.1003, ¶¶272-73.

1. Hosokawa

Like the ’743 patent, Hosokawa relates to a “cluster controlling system [that] transfers packages which have been operating on one computer to another computer when a fault or failure has occurred by monitoring and controlling the packages in the entire system.” Ex.1007, Abstract. Hosokawa describes a system in which the manager resides on one of the hosts being managed. “The manager 105 is one of the packages installed in the cluster system.” Ex.1007, 7:15-17. “If a fault

or failure occurs in the manager or in the computer running the manager, the manager is re-started on another computer by a cluster daemon.” Ex.1007, Abstract. Hosokawa’s manager is illustrated as being run on one of the computer nodes in Fig. 1 below.

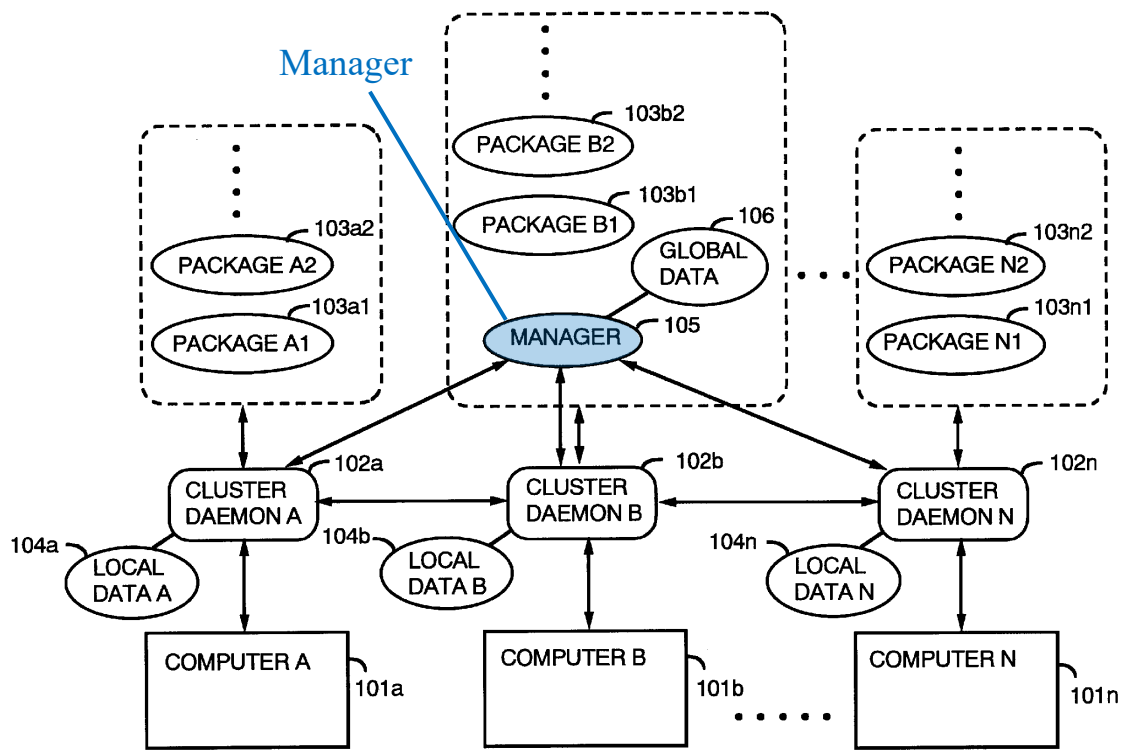


FIG. 1 Ex.1007, Fig. 26a (annotated); Ex.1003, ¶274.

Hosokawa illustrates an example in which the number of applications (referred to as software packages in Hosokawa²) is the same as the number of hosts

² “In this specification, package program is a general term referring to application programs and to service programs.” Ex.1007, 7:3-5.

(referred to as nodes in Hosokawa). In other words, Hosokawa provides an example in which M is equal to N, as shown in Fig. 26A below.

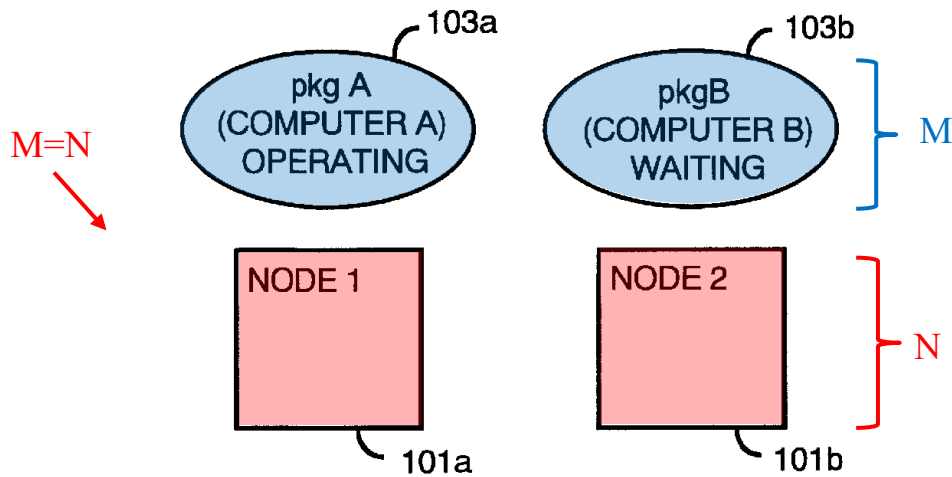


FIG. 26A

Ex.1007, Fig. 26a (annotated); Ex.1003, ¶277.

Additionally, Hosokawa illustrates an example where there are fewer applications M than hosts N ($M < N$). Such example is shown in Fig. 37A below.

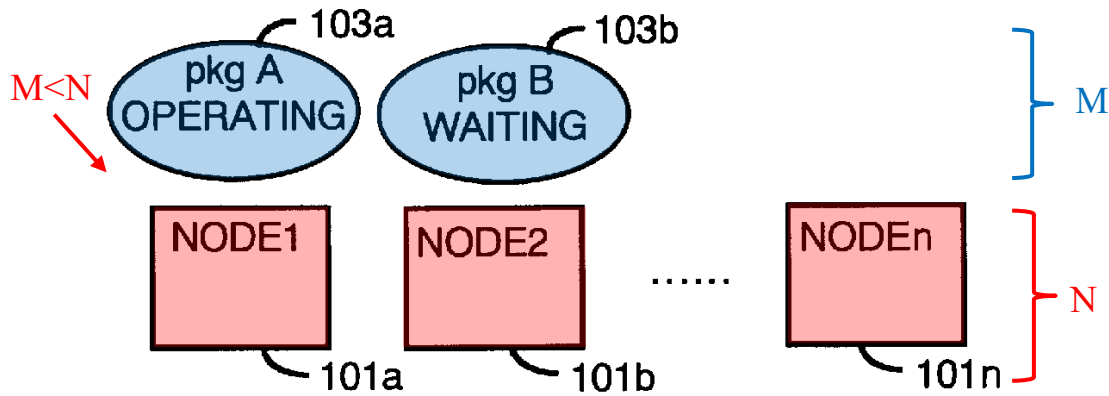


FIG. 37A

Ex.1007, Fig. 37a (annotated); Ex.1003, ¶278.

Hosokawa further illustrates an example where there are more applications M than hosts N ($M > N$). Such example is shown in Fig. 46A below.

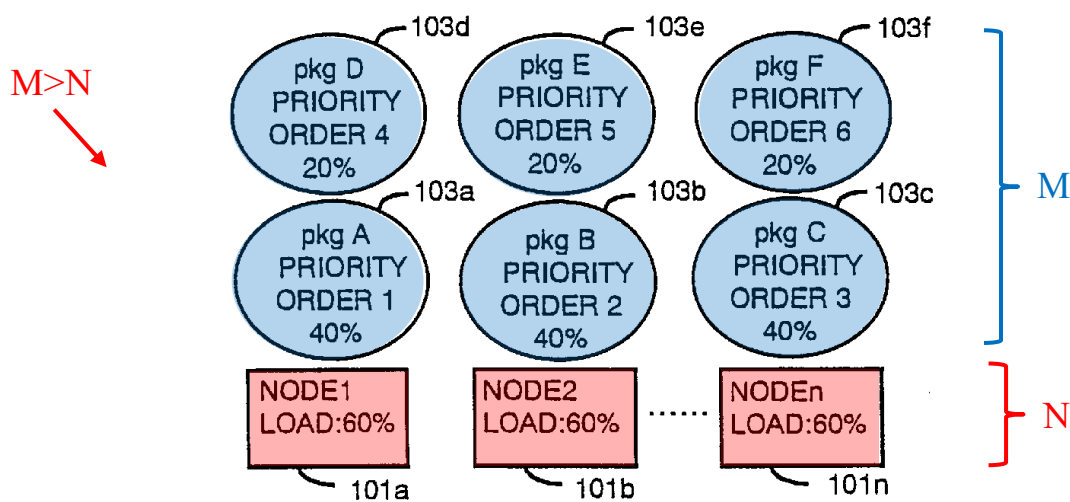


FIG. 46A

Ex.1007, Fig. 26a (annotated); Ex.1003, ¶279.

Accordingly, Hosokawa shows that it was known that in distributed systems, the number of applications can be independent from the number of hosts. Ex.1003, ¶280.

Hosokawa is analogous art to the '743 patent because it is directed to the same field of endeavor (distributed computing) as the '743 patent. *Compare* Ex.1001, Abstract (“A resource manager for a distributed environment”) *with* Ex.1007, Title (“Cluster Controlling System Operating on a Plurality of Computers in a Cluster System.”). In addition, Hosokawa describes the same application survivability techniques that are reasonably pertinent to a problem purportedly solved by the '743 patent. *See* Ex.1007, Abstract (The system “transfers packages which have been operating on one computer to another computer when a fault or failure has occurred.”). The '743 patent likewise purports to address survivability. *See* Ex.1001, 2:15-20 (seeking to “provide a system that is both flexible and adaptive (or at least easily adapted) to changes in system configuration, performance bottlenecks, survivability requirements, scalability, etc.”). Ex.1003, ¶¶274-82.

2. Claim 10

[10.0] Software stored on at least one host for converting N networked hosts into a resource managed system instantiating M managed characteristic application computer programs, each managed characteristic application computer program managed by one of the N networked hosts, the software comprising:

Welch and Hosokawa render limitation [10.0] obvious. Ex.1003, ¶300.

First, for the reasons described above at [10.0] in Ground 1, Welch teaches “software...for converting N networked hosts into a resource managed system instantiating M managed characteristic application computer programs, each managed characteristic application computer program managed by one of the N networked hosts.” Ex.1003, ¶301.

Second, it would have been obvious for Welch’s software to be “stored on at least one host” where that at least one host is one of the managed hosts, as taught by Hosokawa. In particular, Welch explains that its “**software runs on** a heterogeneous network configuration that includes Myrinet, ATM, FDDI, and ethernet, on **multiple heterogeneous host platforms**.” Ex.1005, 4. The “multiple heterogeneous host platforms” are the “ N networked hosts” managed by the QoS management software. Ex.1003, ¶302.

As explained above at XI.D.1, Hosokawa describes a distributed system in which “[t]he manager 105 is one of the packages installed in the cluster system.” Ex.1007, 7:15-17. “If a fault or failure occurs in the manager or in the computer

running the manager, the manager is re-started on another computer by a cluster daemon.” Ex.1007, Abstract. Hosokawa’s manager is illustrated as being run on one of the computer nodes in Fig. 1 below. Ex.1003, ¶303.

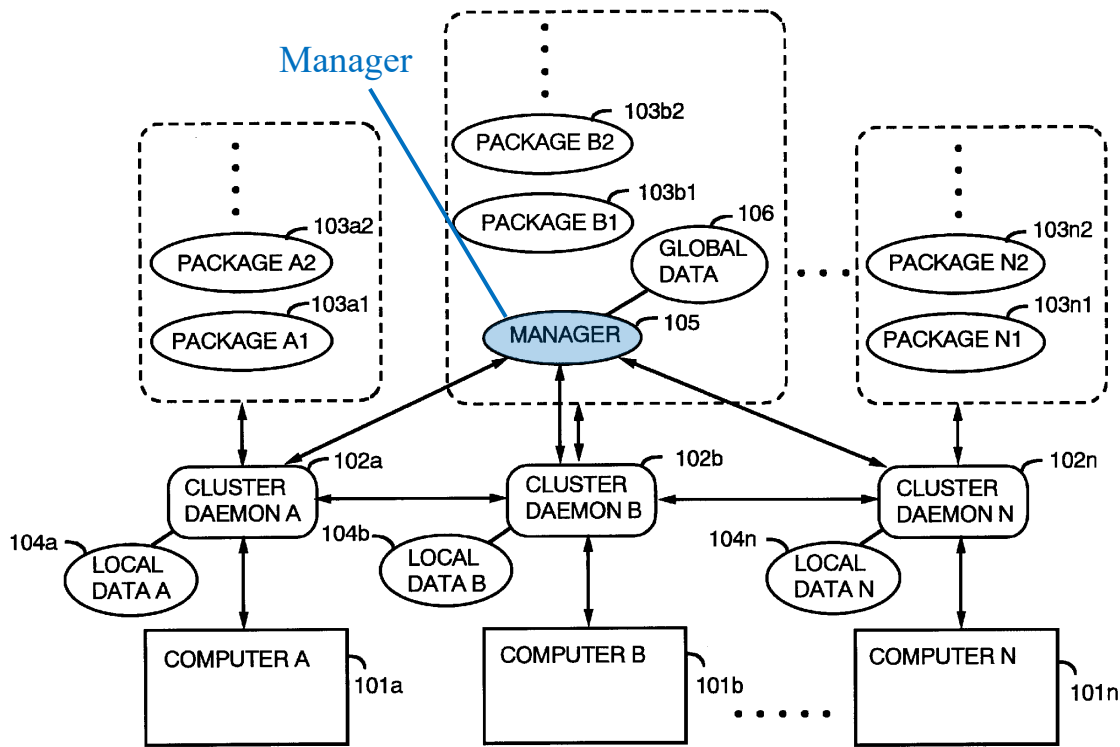


FIG. 1 Ex.1007, Fig. 26a (annotated); Ex.1003, ¶303.

Accordingly, to the extent limitation [10.0] requires that the “at least one host” be one of the “N networked hosts,” it would have been obvious for Welch’s QoS management software to run on one of the managed hosts as taught by Hosokawa. Ex.1003, ¶304.

Thus, Welch describes QoS management software for managing real-time control system applications on distributed hardware to provide startup, replicating,

and other actions related to providing for scalability and survivability, which may run on one of the managed hosts as taught by Hosokawa. Welch and Hosokawa thus render obvious limitation [10.0]. Ex.1003, ¶¶300-306.

a) Reasons to Combine Welch and Hosokawa

A POSITA also would have found it obvious for Welch's QoS management software to run on one of the hosts that it manages, as taught by Hosokawa. Ex.1003, ¶¶283-85. These are conventional implementation details that were well known in the art, as shown by Hosokawa. Welch describes using QoS management software (sometimes referred to as middleware) that manages the scaling and moving of applications across multiple hosts. Ex.1005, 2-3. As Welch explains, a "core component of the middleware is the resource manager. It is activated when programs die and when time-constrained control paths miss their deadlines. In response to these events, it takes appropriate measures to improve the quality of service delivered to the applications." Ex.1005, 2. For example, "[i]f a path becomes unhealthy (misses its deadline), the QoS Monitors detect such a condition, diagnose the cause of poor health, and suggest an action (such as moving or replicating an application program)." Ex.1005, 3. "This decision is made by choosing the host(s) and LAN(s) with the smallest load indices." Ex.1005, 3; Ex.1003, ¶294.

Hosokawa describes a similar management software that “mov[es] package programs to other computers when a fault or failure occurs on one computer in the cluster.” Ex.1007, 2:23-26. In particular, the “manager 105 communicates with cluster daemons on the respective computers to monitor and control the entire cluster system.” Ex.1007, 7:10-12. The manager then “transfers packages which have been operating on one computer to another computer when a fault or failure has occurred by monitoring and controlling the packages in the entire system.” Ex.1007, Abstract; Ex.1003, ¶295.

Hosokawa provides an example of a known implementation technique in which the software manager executes on one of the managed hosts. In particular, “[t]he manager 105 is one of the packages installed in the cluster system.” Ex.1007, 7:15-17. This provides the benefit of recovering the manager when there is a fault or failure. Ex.1007, 7:17-20. In an example, the manager is illustrated as being executed on Computer B in Fig. 1 below.

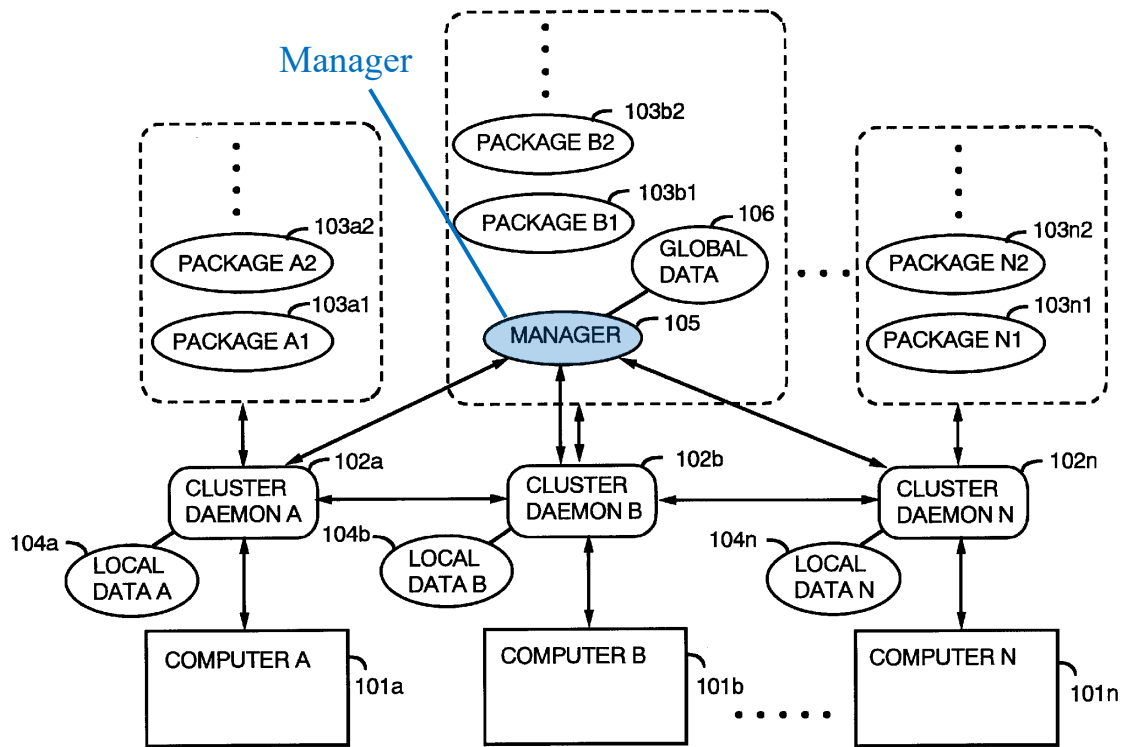


FIG. 1 Ex.1007, Fig. 26a (annotated); Ex.1003, ¶296.

A POSITA would have thus found it similarly beneficial for Welch’s QoS management software to be executed on one of several managed hosts. Ex.1003, ¶297. That way, if the host running the QoS management software fails, the QoS management software can be restarted on another host, as taught by Hosokawa. Ex.1007, 7:17-20. This is consistent with Welch’s goals of “QoS and resource management middleware for survivability and scalability services.” Ex.1005, 4. And Welch’s system is already configured to restart faulty applications on other hosts. *See* Ex.1005, 3. Indeed, the combination is nothing more than “the

predictable use of prior art elements according to their established functions.” *See KSR*, 550 U.S. at 417.

Accordingly, the combination of Welch and Hosokawa represents combining prior art elements (Welch and Hosokawa’s distributed software managers) according to known methods (with the software manager executing on one of the managed hosts) to yield predictable results (allowing for survivability of the software manager). Additionally, the combination is merely applying a known technique (Hosokawa’s management software stored on the managed hosts) to a known device (Welch’s distributed computing system) ready for improvement to yield predictable results (survivability of the management software). Ex.1003, ¶298.

A POSITA would have had a reasonable expectation of success in combining the teachings of Hosokawa with those of Welch because both Welch and Hosokawa describe similar distributed computing systems designed for survivability. *See* Ex.1005, 2-3; Ex.1007, Fig.1, Abstract (“If a fault or failure occurs in the manager or in the computer running the manager, the manager is re-started on another computer by a cluster daemon.”). Accordingly, a POSITA would have expected Hosokawa’s survivable management software technique to work just as well in Welch’s distributed computing system. Ex.1003, ¶299.

[10.1]-[10.7]

See [10.1]-[10.7], Ground 1. Ex.1003, ¶307.

[10.8] *where M and N are positive integers and where M may be equal to, greater than, or less than N.*

Welch and Hosokawa render limitation [10.8] obvious. Ex.1003, ¶308.

A POSITA would have found it obvious that the number of hosts N and the number of applications M in Welch are independent variables. As described above at [10.8], Ground 1, Welch describes distributed hardware that includes a plurality of heterogeneous hosts and increasing and decreasing the number of applications on a fixed number of hosts. Ex.1003, ¶¶309-10.

Further, [10.8] is taught by Hosokawa. Hosokawa explicitly discloses an example in which the number of hosts (N) is independent from the number of applications (M) running on those hosts, including where M is greater than, lesser than, or equal to N. First, Hosokawa discloses that the number of applications (referred to as software packages in Hosokawa³) can be the same as the number of hosts (referred to as nodes in Hosokawa). In other words, Hosokawa provides an

³ “In this specification, package program is a general term referring to application programs and to service programs.” Ex.1007, 7:3-5.

example in which M is equal to N, as shown in Fig. 26A below. Ex.1007, 16:3-21;
Ex.1003, ¶¶311-12.

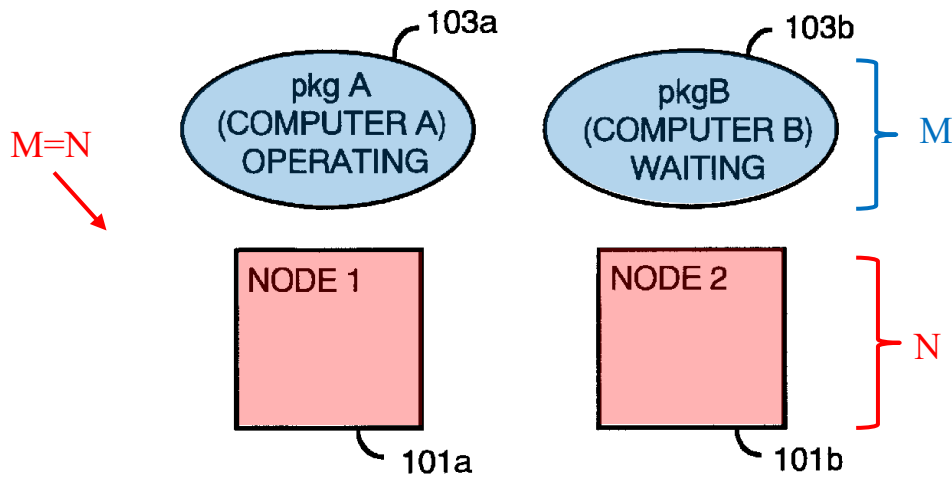


FIG. 26A

Ex.1007, Fig. 26a (annotated); Ex.1003, ¶312.

Additionally, Hosokawa discloses that there can be fewer applications M than hosts N ($M < N$). Such example is shown in Fig. 37A below. Ex.1007, 22:25-39.

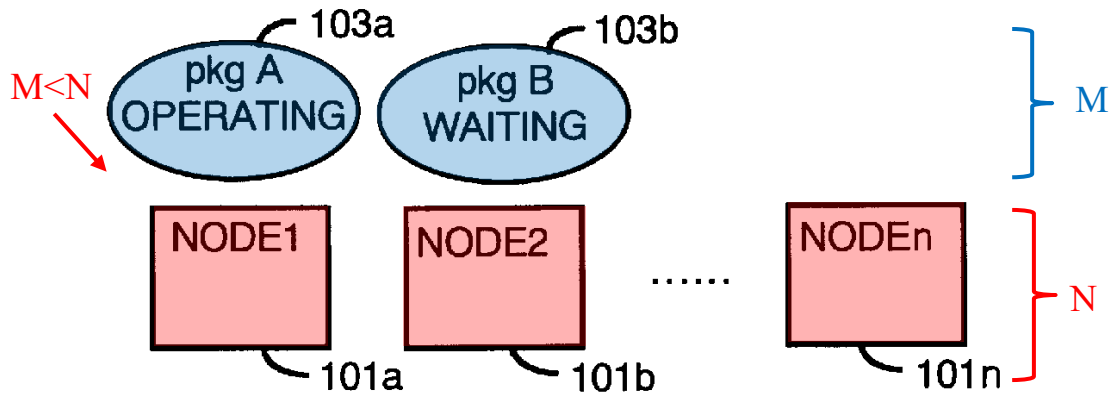


FIG. 37A

Ex.1007, Fig. 37a (annotated); Ex.1003, ¶313.

Hosokawa further discloses that there can be more applications M than hosts N ($M > N$). Such example is shown in Fig. 46A below. Ex.1007, 28:65-29:20.

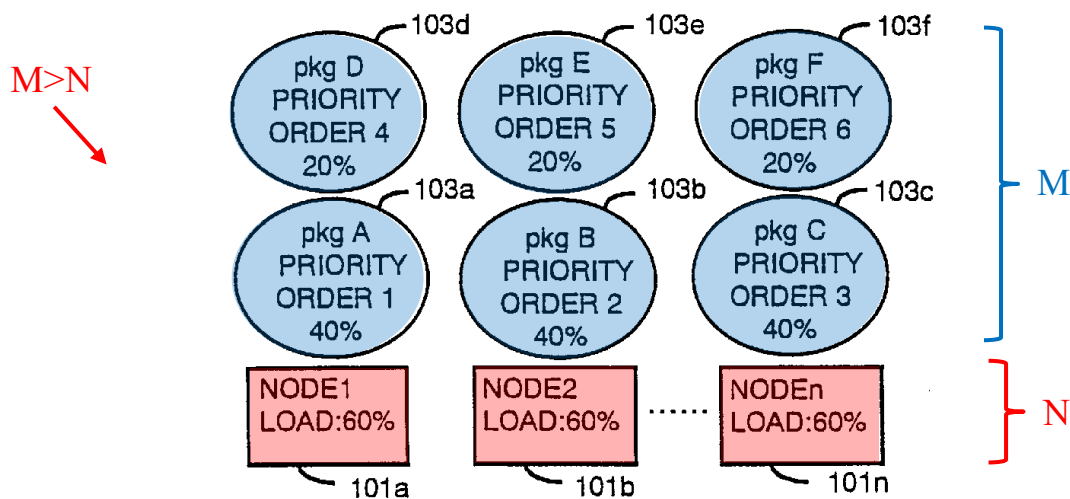


FIG. 46A

Ex.1007, Fig. 26a (annotated); Ex.1003, ¶314.

Accordingly, Hosokawa provides both the general teaching that it was known that in distributed systems, the number of applications can be independent

from the number of hosts, and the specific example recited in [10.8]. Ex.1003, ¶315. Thus, Welch alone and in combination with Hosogawa render obvious “where M and N are positive integers and where M may be equal to, greater than, or less than N ” as claimed. Ex.1003, ¶¶308-17.

b) Reasons to Combine Welch and Hosokawa

A POSITA would have found it obvious to combine the teachings of Hosokawa with the teachings of Welch. Ex.1003, ¶¶283-85. Welch describes a distributed system that includes a plurality of hosts and LANs. Ex.1005, 2-3. Indeed, the QoS management software is to “recommend actions (e.g., move a program to a **different host or LAN**,” which indicates that the “distributed hardware” includes multiple hosts. Ex.1005, 2. These multiple hosts support a plurality of RTCS applications. Ex.1005, 2-3. “The Resource Manager informs the Program Control to start the Real-Time Control System (RTCS) application programs on the specified hosts (step 5 of Figure 3).” Ex.1005, 3. These applications are scaled up or down to meet QoS requirements: Welch’s system “recommends actions (e.g., move a program to a different host or LAN, shed a program, or replicate a program) to improve QoS.” Ex.1005, 2; Ex.1003, ¶¶286-87.

Hosokawa describes a similar distributed system that involves multiple hosts. Ex.1007, Abstract. Hosokawa relates to a “cluster controlling system [that] transfers packages which have been operating on one computer to another

computer when a fault or failure has occurred by monitoring and controlling the packages in the entire system.” Ex.1007, Abstract. Hosokawa’s system includes “a plurality of computers A-N ... included in a cluster.” Ex.1007, 6:65-66. “**The respective computers are executing** cluster daemon A-N (102a-102n), and the respective cluster daemons start-up respective **package programs** A1-N2.” Ex.1007, 6:66-7:2.

A POSITA would have found it beneficial for Welch’s scalable system to have the number of applications be independent from the number of hosts, including for the applications to have the capability of being less than, more than, or equal to the number of hosts. Ex.1003, ¶289. That way, as applications are replicated, or shed, host systems do not have to be correspondingly commissioned or decommissioned. And when application demand is low, the applications can be shed so as to conserve computing resources, and then scale up as demand grows. Ex.1003, ¶289. The number of applications being equal to, less than, or greater than the number of hosts thus allows Welch to meet its explicit design goals: “Furthermore, resources should be utilized in an efficient manner, and scalability must be provided to address the ever-increasing complexity of scenarios that confront such systems, even though the worst case scenarios of the environment may be unknown.” Ex.1005, 1; Ex.1003, ¶289-90. Given this explicit teaching, suggestion, and motivation to be able to have a system with variable number of

applications on a fixed number of hosts, a POSITA would have understood that in Welch, M may be greater than, less than, or equal to N. *KSR*, 550 U.S. at 418.

Accordingly, the combination of Welch and Hosokawa represents combining prior art elements (Welch and Hosokawa's distributed system) according to known methods (independently scaling applications) to yield predictable results (allowing for scalability to quickly respond to demand for resources). *KSR*, 550 U.S. at 418. Additionally, the combination is merely applying a known technique (number of applications independent from number of hosts) to a known device (distributed computing systems) ready for improvement to yield predictable results (efficient utilization of resources). *KSR*, 550 U.S. at 417. Ex.1003, ¶291.

A POSITA would have had a reasonable expectation of success in combining the teachings of Hosokawa with those of Welch because both Welch and Hosokawa describe similar distributed computing systems suitable for scalability. *See* Ex.1005, Title, 1 ("scalability must be provided."); Ex.1007, 2:15-4:55 (describing a cluster controlling system for monitoring and controlling resources and starting up software packages on a plurality of computers."). Accordingly, a POSITA would have expected that Hosokawa's independent number of applications to work just as well in Welch's distributed computing system. Ex.1003, ¶292. In addition, independently scaling in distributed systems

was known. *See e.g.*, Ex.1009 (describing a “software system designed to support the development and execution of distributed, scalable, fault resilient telecommunications applications 12.”). Ex.1003, ¶293.

3. Claims 11, 14, 16, and 18

See claims 11, 14, 16, and 18, Ground 1. Ex.1003, ¶318.

XII. WELCH AND RAVINDRAN ARE PRINTED PUBLICATIONS

Petitioner submits the Declaration of Dr. Hall-Ellis (Ex.1015) to show that Welch (Ex.1005) and Ravindran (Ex.1006) were publicly accessible before the filing date of the ’743 patent (May 24, 2001) and the earliest priority date of the ’743 patent (May 25, 2000). Specifically, Welch was publicly accessible by December 16, 1999, and Ravindran was publicly accessible by December 10, 1998. Ex.1015, ¶¶52, 59. Dr. Hall-Ellis’s Declaration is accompanied by Attachments 1A-1D and 2A-2C, which include the evidence on which Dr. Hall-Ellis relies in making her Declaration.

“Public accessibility” is “the touchstone in determining whether a reference constitutes a printed publication.” *In re Hall*, 781 F.2d 897, 899 (Fed. Cir. 1986) (cleaned up). A reference is publicly accessible if it was “made available to the extent that persons interested and ordinarily skilled in the subject matter or art, exercising reasonable diligence, can locate it.” *Acceleration Bay, LLC v. Activision Blizzard, Inc.*, 908 F.3d 765, 772 (Fed. Cir. 2018) (cleaned up).

Dr. Hall-Ellis' Declaration (Ex.1015) shows that Welch was publicly accessible to POSITAs by December 16, 1999, which is more than one year before the filing date of the '743 patent and five months before the earliest priority date of the '743 patent. Specifically, Welch was publicly available, cataloged, and indexed at the Karl F. Wendt Engineering Library at the University of Wisconsin – Madison and the Linda Hall Library of Science, Engineering & Technology by December 16, 1999. Ex.1015, ¶¶48-52 (citing Attachments 1A-1D).

In addition, Dr. Hall-Ellis' Declaration shows that Ravindran was publicly accessible to POSITAs by December 10, 1998, which is more than one year before the earliest priority date of the '743 patent. Specifically, Ravindran was publicly available, cataloged, and indexed at the Library Hall Library of Science by December 10, 1998. Ex.1015, ¶¶55-60 (citing Attachments 2A-2C).

XIII. DISCRETIONARY DENIAL IS INAPPROPRIATE

A. No basis for § 325(d) denial

Denial under § 325(d) is not warranted because the challenges presented in this petition are neither cumulative nor redundant to the prosecution of the '743 patent. The Examiner did not consider Welch, Ravindran, or Hosokawa, or the combination of Welch, Ravindran, and Hosokawa as presented in this Petition. Accordingly, the Board should not exercise its discretion to deny this petition under § 325(d).

B. No basis for *Fintiv* denial

The Board balances six factors in considering denial under § 314. *Apple Inc. v. Fintiv, Inc.*, IPR2020-00019, Paper 11 (Mar. 20, 2020) (precedential). These factors strongly favor institution.

1. No evidence regarding a stay

No motion to stay has been filed, so the Board should not infer the outcome of such a motion. *Sand Revolution II, LLC v. Continental Intermodal Group – Trucking LLC*, IPR2019-01393, Paper 24 at 7 (June 16, 2020) (informative). Thus, this factor is neutral on discretionary denial.

2. Parallel proceeding trial date

This factor weighs against discretionary denial because the projected trial date—based on median time-to-trial statistics—is in January of 2027⁴, which is after the projected date for the Board to issue a Final Written Decision in June of 2026. While trial is currently proposed⁵ for May 14, 2026 (Ex.1011, 4), the Board recognizes “that scheduled trial dates are unreliable and often change.” *See* Director’s Memorandum on Discretionary Denials (Ex.1013), 8. Because the

⁴ The current average time-to-trial for the district court hearing the co-pending litigation is 33.1 months. *See* Ex.1012.

⁵ The district court has not entered an order scheduling a trial date.

projected trial date is approximately seven months *after* the Board’s expected date for issuing a Final Written Decision, this factor weighs against discretionary denial.

3. Investment in the parallel proceeding

The investment in the co-pending litigation has been minimal. As mentioned above, a claim construction hearing has not yet occurred, fact discovery is not scheduled until April 11, 2025, and expert discovery is not scheduled until November 20, 2025. Ex.1018, 3-6; *see PEAG LLC v. Varta Microbattery GmbH*, IPR2020-01214, Paper 8 at 17 (Jan. 6, 2021). This lack of investment favors institution.

Moreover, Petitioner only learned which claims were being asserted in August 2024. *See* Ex.1014 (infringement contentions cover pleading). Since then, Petitioner has worked expeditiously to file this petition within four months. And, as of this filing, Patent Owner has not yet served its final infringement contentions. Under *Fintiv*, Petitioner’s prompt filing “weigh[s] against exercising the authority to deny institution.” *Fintiv*, Paper 11 at 11.

4. Overlapping issues with the parallel proceeding

The co-pending litigation is in its early stages. Petitioner served its preliminary invalidity contentions on October 24, 2024 and supplemental and supplemental invalidity contentions on November 27, 2024. However, Petitioner

will not serve its final invalidity contentions until May 22, 2025, near when the Board would issue an institution decision in this case. Ex.1011, 3. The extent of overlap is thus speculative at this point in time and therefore weighs against discretionary denial.

5. Petitioner is a defendant

Petitioner is a defendant in the co-pending litigation. That is true of most Petitioners in IPR proceedings, making this factor neutral. *See HP Inc. v. Slingshot Printing LLC*, IPR2020-01084, Paper 13 at 9 (Jan. 14, 2021).

6. Other circumstances

“[T]he PTAB will not deny institution of an IPR or PGR under *Fintiv* (i) when a petition presents compelling evidence of unpatentability.” Ex.1013, 9. Here, the petition—along with Dr. Zadok’s expert testimony—plainly shows that the ’743 patent claims no more than what others had previously disclosed.

Because the *Fintiv* factors are either neutral or weigh against discretionary denial, institution should not be denied on discretionary factors.

C. No basis for *General Plastic* denial

The ’743 patent has not been challenged in any prior IPR petition, so *General Plastic* discretionary institution does not apply to this Petition. *See General Plastic Indus. Co., Ltd. v. Canon Kabushiki Kaisha*, IPR2016-01357, Paper 19 at 16 (Sept. 6, 2016) (Section II.B.4.i. precedential).

XIV. CONCLUSION

Petitioner has established a reasonable likelihood that the Challenged Claims are unpatentable.

Respectfully submitted,

Dated: December 23, 2024
HAYNES AND BOONE, LLP
2801 N. Harwood Street
Suite 2300
Dallas, TX 75201
Customer No. 27683

/Jonathan R. Bowser/
Jonathan R. Bowser
Registration No. 54,574
Lead Counsel for Petitioner

XV. MANDATORY NOTICES

A. Real party-in-interest

Pursuant to 37 C.F.R. § 42.8(b)(1), Petitioner certifies that the real party-in-interest is Tesla, Inc.

B. Related matters

Pursuant to 37 C.F.R. § 42.8(b)(2), to the best knowledge of the Petitioner, the '743 patent⁶ is or was involved in the following cases:

Case Heading	Number	Court	Filed
<i>Intellectual Ventures II LLC v. Tesla, Inc</i>	<i>WDTX-1-24-cv-00390</i> (terminated)	WDTX	Apr. 11, 2024
<i>Intellectual Ventures II LLC v. Tesla, Inc.</i>	<i>WDTX-6-24-cv-00188</i> (pending)	WDTX	Apr. 12, 2024
<i>Intellectual Ventures II LLC v. Tesla, Inc.</i>	<i>WDTX-1-24-cv-00884</i> (transferred)	WDTX	Apr. 12, 2024

⁶ In its Complaint for Patent Infringement, Intellectual Ventures II LLC identifies itself as the exclusive licensee of the '743 patent. Ex.1010, 2 (¶4), 18 (¶40).

C. Lead and back-up counsel and service information

Lead Counsel

Jonathan R. Bowser
Haynes and Boone, LLP
2801 N. Harwood Street
Suite 2300
Dallas, TX 75201

Phone: (202) 654-4503
Fax: (214) 200-0853
jon.bowser.ipr@haynesboone.com
USPTO Reg. No. 54,574

Back-up Counsel

Calmann J. Clements
Haynes and Boone, LLP
2801 N. Harwood Street
Suite 2300
Dallas, TX 75201

Phone: 972-739-8638
Fax: (214) 200-0853
calmann.clements.ipr@haynesboone.com
USPTO Reg. No. 66,910

Scott T. Jarratt
Haynes and Boone, LLP
2801 N. Harwood Street
Suite 2300
Dallas, TX 75201

Phone: (972) 739-8663
Fax: (214) 200-0853
scott.jarratt.ipr@haynesboone.com
USPTO Reg. No. 70,297

Ashraf Fawzy
Tesla, Inc.
1 Tesla Road
Austin, TX 78725

Phone: (202) 905-9221
afawzy@tesla.com
USPTO Reg. No. 67,914

Please address all correspondence in this proceeding to lead and back-up counsel. Petitioner consents to service in this proceeding by email at the addresses above.

CERTIFICATE OF WORD COUNT

Pursuant to 37 C.F.R. § 42.24(d), Petitioner hereby certifies, in accordance with and reliance on the word count provided by the word-processing system used to prepare this Petition, that the number of words in this paper is 13,983. Pursuant to 37 C.F.R. § 42.24(d), this word count excludes the table of contents, table of authorities, mandatory notices under § 42.8, certificate of service, certificate of word count, appendix of exhibits, and any claim listing.

Dated: December 23, 2024

/Jonathan R. Bowser/
Jonathan R. Bowser
Lead Counsel for Petitioner
Registration No. 54,574

CERTIFICATE OF SERVICE

The undersigned certifies that, in accordance with 37 C.F.R. § 42.6(e) and 37 C.F.R. § 42.105, service was made on Patent Owner as detailed below.

Date of service December 23, 2024

Manner of service FEDERAL EXPRESS

Documents served Petition for *Inter Partes* Review Under 35 U.S.C. § 312 and 37 C.F.R. § 42.104 of U.S. 7,181,743;
Petitioner's Power of Attorney;
Petitioner's Exhibit List;
Exhibits 1001-1020.

Persons served Naval Surface Warfare Center
Office of Counsel Code 00L
17632 Dahlgren Road, Suite 158
Dahlgren, VA 22448-5110

Courtesy service on litigation counsel for
Intellectual Ventures II LLC:
Jonathan K. Waldrop
Kasowitz Benson Torress LLP
333 Twin Dolphin Drive, Suite 200
Redwood Shores, California 94065

/Jonathan R. Bowser/
Jonathan R. Bowser
Lead Counsel for Petitioner
Registration No. 54,574