

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

CARVANA, LLC,
Petitioner

v.

INTERNATIONAL BUSINESS MACHINES CORP.
Patent Owner

Case IPR2025-00564

U.S. Patent No. 7,702,719

DECLARATION OF DR. HENRY H. HOUH
ON BEHALF OF PATENT OWNER

TABLE OF CONTENTS

	<u>Page</u>
I. INTRODUCTION	1
II. BASIS OF OPINIONS	1
III. MATERIALS REVIEWED	2
IV. EDUCATION AND EXPERIENCE.....	3
V. LEGAL STANDARDS	9
A. Claim Construction	10
B. Anticipation	10
C. Obviousness.....	11
VI. LEVEL OF ORDINARY SKILL IN THE ART	14
VII. TECHNOLOGICAL BACKGROUND	15
A. Overview of the '719 Patent	15
B. Prosecution History of the '719 Patent.....	21
VIII. CLAIM CONSTRUCTION	23
IX. PETITIONER'S ASSERTED PRIOR ART.....	24
A. Roy (Ex. 1004)	24
X. PETITIONER'S ASSERTED GROUNDS	31
XI. GROUND 1: PETITIONER DOES NOT SHOW THAT ROY DISCLOSES THE CHALLENGED CLAIMS 1, 13, 25, AND 26.....	32
A. Roy Does Not Disclose "Configuring The Server To . . . Execute View-Generating . . . Logic Associated With The Application" As Recited In Limitations 1a, 13a, 25a, 26a (Claims 1, 13, 25, 26).....	32

1.	Roy Does Not Disclose “Configuring The Server To . . . Execute View-Generating . . . Logic Associated With The Application”	32
B.	Roy Does Not Disclose “Wherein The Client And The Server Both Locally Maintain At Least A Portion Of The Model And Execute The View-Generating And Controller Logic Associated Therewith” As Recited In Limitations 1d, 13d, 25d, 26d (Claims 1, 13, 25, 26).....	38
1.	Roy Does Not Disclose That The Client Executes Alleged “View-Generating Logic” That Must Be The Same As The Alleged “View Generating Logic” Executed On The Server, Or A Subset Thereof, As Recited In Limitation (d).....	39
2.	Roy Does Not Disclose That The Client Executes Alleged “Controller Logic” That Must Be The Same As The Alleged “Controller Logic” Executed On The Server, Or A Subset Thereof, As Recited In Limitation (d).....	47
XII.	GROUND 2: PETITIONER DOES NOT SHOW THAT ROY WOULD RENDER OBVIOUS THE CHALLENGED CLAIMS 1, 13, 25, AND 26.....	53
A.	Roy Does Not Disclose, Teach, Or Suggest “Configuring The Server To ... Execute View-Generating ... Logic Associated With The Application” As Recited In Limitations 1a, 13a, 25a, 26a (Claims 1, 13, 25, 26).....	53
B.	Roy Does Not Disclose, Teach, Or Suggest “Wherein The Client And The Server Both Locally Maintain At Least A Portion Of The Model And Execute The View-Generating And Controller Logic Associated Therewith” As Recited In Limitations 1d, 13d, 25d, 26d (Claims 1, 13, 25, 26)	54
1.	Roy Does Not Disclose, Teach, Or Suggest That The Client Executes Alleged “View-Generating Logic” That Must Be The Same As The Alleged “View Generating Logic” Executed On The Server, Or A Subset Thereof, As Recited In Limitation (d)	54

XIII. GROUNDS 3-4: PETITIONER DOES NOT SHOW THAT THE COMBINATIONS OF ROY WITH O'REILLY AND/OR FENNELL WOULD RENDER OBVIOUS THE CHALLENGED CLAIMS.	57
XIV. DECLARATION IN LIEU OF OATH	58

CLAIMS

[Claim 1, 1-PRE] A method for use in a client/server system of reducing interactions between a client and server in association with an application being accessed by the client at the server, the method comprising the steps of:

[1a] configuring the server to store a model associated with the application and to execute view-generating and controller logic associated with the application; and

[1b] configuring the client to store at least a subset of the model associated with the application and to execute at least a subset of the view-generating and controller logic associated with the application;

[1c] wherein one or more portions of the application are performed at the client without the client having to interact with the server, and

[1d] further wherein the client and the server both locally maintain at least a portion of the model and execute the view-generating and controller logic associated therewith.

[Claim 2] The method of claim 1, wherein the client and server communicate over a HyperText Transport Protocol network.

[Claim 3] The method of claim 1, wherein the client performs the one or more portions of the application in accordance with browser software running thereon.

[Claim 4] The method of claim 3, wherein the configuring step further comprises the step of partitioning a screen area associated with the browser software into frames.

[Claim 5] The method of claim 4, wherein the at least a subset of the model, the view-generating and the controller logic associated with the application are associated with at least one frame and one or more views for display in accordance with the application are associated with at least another frame.

[Claim 6] The method of claim 5, wherein the at least one view frame is a visible frame.

[Claim 7] The method of claim 5, wherein the at least one frame associated with the at least a subset of the model, the view-generating logic and the controller logic is not a visible frame.

[Claim 8] The method of claim 4, wherein the configuring step further comprises forming at least one frame with which application-independent view-generating logic and controller logic is associated.

[Claim 9] The method of claim 8, wherein the at least one application-independent view-generating logic and controller logic frame further has an application-independent model associated therewith.

[Claim 10] The method of claim 8, wherein the at least one application-independent view-generating logic and controller logic frame serves as an

application programming interface for developing views to be displayed in accordance with the application.

[Claim 11] The method of claim 10, wherein the views are implemented in accordance with the HyperText Markup Language and the application programming interface is implemented in accordance with the JavaScript language.

[Claim 12] The method of claim 1, wherein the at least a subset of the model, the view-generating and the controller logic associated with the application are downloaded from the server to the client upon demand.

[Claim 13, 13-PRE] A network-based system, comprising:

[13a] a server having at least one processor operative to: (i) store a model associated with an application associated with the server; and (ii) execute view-generating and controller logic associated with the application; and

[13b] a client, coupled to the server via a network, having at least one processor operative to: (i) store at least a subset of the model associated with the application; and (ii) execute at least a subset of the view-generating and controller logic associated with the application;

[13c] wherein one or more portions of the application are performed at the client without the client having to interact with the server such that interactions between the client and server are reduced, and

[13d] further wherein the client and the server both locally maintain at least a portion of the model and execute the view-generating and controller logic associated therewith.

[Claim 14] The system of claim 13, wherein the network is a HyperText Transport Protocol network.

[Claim 15] The system of claim 13, wherein the client processor performs the one or more portions of the application in accordance with browser software running thereon.

[Claim 16] The system of claim 15, wherein the client processor is further operative to partition a screen area associated with the browser software into frames.

[Claim 17] The system of claim 16, wherein the at least a subset of the model, the view-generating and the controller logic associated with the application are associated with at least one frame and one or more views for display in accordance with the application are associated with at least another frame.

[Claim 18] The system of claim 17, wherein the at least one view frame is a visible frame.

[Claim 19] The system of claim 17, wherein the at least one frame associated with the at least a subset of the model, the view-generating logic and the controller logic is not a visible frame.

[Claim 20] The system of claim 16, wherein the client processor is further operative to form at least one frame with which application-independent view-generating logic and controller logic is associated.

[Claim 21] The system of claim 20, wherein the at least one application-independent view-generating logic and controller logic frame further has an application-independent model associated therewith.

[Claim 22] The system of claim 20, wherein the at least one application-independent view-generating logic and controller logic frame serves as an application programming interface for developing views to be displayed in accordance with the application.

[Claim 23] The system of claim 22, wherein the views are implemented in accordance with the HyperText Markup Language and the application programming interface is implemented in accordance with the JavaScript language.

[Claim 24] The system of claim 13, wherein the at least a subset of the model, the view-generating and the controller logic associated with the application are downloaded from the server to the client upon demand.

[Claim 25, 25-PRE] An article of manufacture for use in a client/server system for reducing interactions between a client and server in association with an application being accessed by the client at the server, comprising machine readable media containing one or more programs which when executed implement the steps of:

[25a] configuring the server to store a model associated with the application and to execute view-generating and controller logic associated with the application; and

[25b] configuring the client to store at least a subset of the model associated with the application and to execute at least a subset of the view-generating and controller logic associated with the application;

[25c] wherein one or more portions of the application are performed at the client without the client having to interact with the server, and

[25d] further wherein the client and the server both locally maintain at least a portion of the model and execute the view-generating and controller logic associated therewith.

[Claim 26, 26-PRE] A method for use in a client/server system of reducing interactions between a client and server in association with an application being accessed by the client at the server, the method comprising the steps of:

[26a] configuring the server to: (i) store a model associated with the application; (ii) execute view-generating logic associated with the application; and (iii) execute controller logic associated with the application; and

[26b] configuring the client to: (i) store at least a subset of the model associated with the application; (ii) execute at least a subset of the view-

generating logic associated with the application; and (iii) execute at least a subset of the controller logic associated with the application;

[26c] wherein one or more portions of the application are performed at the client without the client having to interact with the server, and

[26d] further wherein the client and the server both locally maintain at least a portion of the model and execute the view-generating and controller logic associated therewith;

[26e] further wherein, in accordance with such a dual model-view-controller arrangement, a model comprises application data, rules, and algorithms affecting the data, a view comprises a screen or window representation of a subset of the model that the application chooses to display, and a controller comprises the logic that processes user requests, and causes the model to be changed and the view to be refreshed.

I. INTRODUCTION

I, Henry H. Houh, Ph.D., declare as follows:

1. I have personal knowledge of all the matters about which I testify in this declaration.

2. I have been retained as an expert in this case on behalf of International Business Machines Corporation (“IBM” or “Patent Owner”) to provide expert opinions and testimony regarding U.S. Patent No. 7,702,719 (the “’719 Patent”).

3. I am being compensated for my work done in connection with this matter at my standard rate plus expenses. My compensation does not depend on the outcome of this case.

II. BASIS OF OPINIONS

4. I have reviewed and considered the documents and other materials listed below in Section III in light of my specialized knowledge provided in my education, training, research, and experience, as summarized in Section IV and described in detail in my CV, which is provided as Exhibit 2030. My analysis of those materials, combined with the specialized knowledge that I have obtained over the course of my education and career, form the bases of my opinions in this declaration.

III. MATERIALS REVIEWED

5. In forming the opinions set forth in this declaration, I have relied upon my experience and knowledge in the relevant fields of art, as well as my review of certain documents in this matter, including:

- U.S. Patent No. 7,702,719;
- The prosecution history of U.S. Patent No. 7,702,719;
- Carvana, LLC's' Petition for *Inter Partes* Review of U.S. Patent No. 7,702,719 and all materials cited therein;
- The exhibits listed below; and
- All other materials cited herein.

EX #	DESCRIPTION
2005	<i>IBM v. Zynga</i> , 1:22-cv-00590, Def Final Invalidity Contentions (December 12, 2023) (excerpted with redactions) (bearing Bates number range beginning with IBM-CARVANA00022845)
2006	<i>IBM v. Zynga</i> , 1:22-cv-00590, Def Exhibit C-13 Roy Invalidity Chart (December 12, 2023) (with redactions) (bearing Bates number range beginning with IBM-CARVANA00411693)
2007	<i>IBM v. Zynga</i> , 1:22-cv-00590, Expert Report of Dr. John K. Bennett (February 2, 2024) (excerpted with redactions) (including pages bearing Bates number range beginning with IBM-CARVANA00238717) ¹
2008	<i>IBM v. Zynga</i> , 1:22-cv-00590, Trial Transcript (September 12, 2024) (excerpted with redactions)

¹ Ex. 2007 includes excerpts of the Bates-stamped version of the Expert Report of Dr. John K. Bennett as well as the corresponding pages from a version of the same report without Bates stamps.

EX #	DESCRIPTION
2009	<i>IBM v. Zynga</i> , 1:22-cv-00590, DI 0529 Jury Verdict (September 13, 2024) (with redactions)
2010	<i>IBM v. Zynga</i> , 1:22-cv-00590, DI 0539 Partial Judgment following Jury Verdict (September 24, 2024)
2020	<i>Carvana v. IBM</i> , 7:23-cv-08616, Carvana Supplemental Invalidity Contentions (July 10, 2024) (excerpted)
2021	<i>Carvana v. IBM</i> , 7:23-cv-08616, Carvana Exhibit D-13 Roy Invalidity Chart (December 12, 2023)
2022	Annotated Carvana Exhibit D-13 Roy Invalidity Chart
2024	<i>Carvana v. IBM</i> , 7:23-cv-08616, Claim Construction Briefing (Combined) (excerpted)
2025	<i>Carvana v. IBM</i> , 7:23-cv-08616, Markman Hearing Transcript (November 20, 2024) (excerpted)
2032	Krasner, G. & Pope, S., <i>A Cookbook for Using the Model-View-Controller User Interface Paradigm in Smalltalk-80</i> , Journal of Object-Oriented Programming, August/September 1988 (“Krasner-Pope”)

IV. EDUCATION AND EXPERIENCE

6. As demonstrated in my professional experience described in my curriculum vitae (provided as Exhibit 2030) and the summary of my qualifications below, I have significant scientific and technological experience in computer science, electrical engineering, and web sites. I have had first-hand knowledge of the state of the prior art of and terminology for computer science and electrical engineering since at least 1998.

7. I received a Ph.D. in Electrical Engineering and Computer Science from the Massachusetts Institute of Technology (“MIT”) in 1998. I also received a Master of Science degree in Electrical Engineering and Computer Science in 1991,

a Bachelor of Science degree in Electrical Engineering and Computer Science in 1989, and a Bachelor of Science Degree in Physics in 1990, all from MIT.

8. I am currently self-employed as an independent technical consultant. Until 2022, I was also president of Einstein's Workshop, a company that provides supplemental science, technology, engineering, and mathematics ("STEM") education to children of all ages.

9. I first worked in the area of telecommunications in 1987 when I worked as a summer intern at AT&T Bell Laboratories as part of a five-year dual degree program at MIT. I continued to work at AT&T Bell Laboratories as part of this MIT program. While I was at MIT, I was a teaching assistant ("TA") in the Electrical Engineering and Computer Science Department's core Computer Architectures course. I first was a TA as a senior, performing a role typically reserved for graduate students. I later became head TA. As a TA, I helped write homework assignments, lab assignments, and exams. I also taught in the recitation sections.

10. Later, as part of my doctoral research at MIT from 1991-1998, I was a research assistant in the Telemedia Network Systems ("TNS") group at the Laboratory for Computer Science. The TNS group built a high-speed gigabit network and created applications that ran over the network. Example applications included ones for remote video capture, processing, and display of video on computer terminals. In addition to working on the design of core network

components, designing and building the high-speed links, and designing and writing the device drivers for the interface cards, I also set up the group's web server.

11. With the TNS group I researched and installed web servers and web caches. I also helped to build the web pages that initiated these video sessions via a web interface. Vice-President Al Gore visited our group in 1996 and received a demonstration of—and remotely drove—a radio controlled toy car with a wireless video camera mounted on it; the video was encoded by TNS-designed hardware, streamed over the TNS-designed network and displayed using TNS-designed software.

12. Our videos could be displayed on both the servers that originated the video stream, and on a remote client. The server and client could be running on different operating systems, and, as there was no native support for streaming video display in a web browser at the time, playback on the client required client software compatible with the X Windows protocols. There were several different software packages for the Microsoft Windows platform, and these were different than the software on our Unix servers.

13. One of my fellow graduate students and I presented a paper on our video system at the first ever World Wide Web Conference, which was held in Geneva, Switzerland in May 1994. In addition, we also presented a paper on database-backed web pages our group had implemented, which we called “Active

Pages” at the time. The pages were driven by back-end software that used programmatic templated views, a data model of the source data, and business logic to drive the page display and functionality.

14. I defended and submitted my Ph.D. thesis, titled “Designing Networks for Tomorrow’s Traffic,” in January 1998. As part of my thesis research, I analyzed local-area and wide-area flows to show a more efficient method for routing packets in a network, based on traffic patterns at the time.

15. While I was in graduate school, I started a company that provided web site design services and targeted advertising opportunities. We also sold targeted advertising placements. As the web became popular first at colleges and universities, we focused initially on building college recruiting web sites and sold advertisements to companies wishing to target students to recruit at specific colleges. We developed technology that utilized the web user’s college affiliation to customize advertisements targeted to students at that college.

16. From 1997 to 1999, I was a Senior Scientist and Engineer at NBX Corporation, a start-up that made business telephone systems for streaming packetized audio over data networks instead of using traditional telephone lines. NBX was later acquired by 3Com Corporation, and the phone system created by NBX is still used today by numerous businesses. As part of my work at NBX, I designed the core audio reconstruction algorithms for the telephones, as well as the

packet transmission algorithms. I also designed and validated the core packet transport protocol used by the phone system. The protocol was used for all signaling in the phone system, including for the setup of conference calls. The NBX system also featured a computer interface for initiating phone calls, which could also initiate conference calls. The NBX system also supported TAPI, the Telephony Application Programming Interface, thus allowing other computer programs to integrate with our system telephony features. We obtained U.S. Patent No. 6,697,963, entitled “Telecommunication method for ensuring on-time delivery of packets containing time-sensitive data,” as a result of part of this work.

17. From 1999 to 2004, I was employed by Empirix or its predecessor company, Teradyne. Empirix was a leader in test tools for telecommunications protocols and software systems, providing functional testing tools as well as load testing tools. I worked on potential new products, including on a test tool product that was focused on testing Enterprise Java Beans, a Java servlet-based technology that have many similarities to portlets, including that they are both packaged in WAR files. From 2000 to 2001, I conceived and built a test platform for testing Voice-over-IP (VoIP). The first application on this new test platform was a cloud emulator for simulating the effects of transmitting VoIP over a busy network. Starting in 2001, I was architect for the next generation of web testing product by Empirix known as e-Test Suite. e-Test Suite is now owned by Oracle Corporation. e-Test provided

functional and load testing for web sites. e-Test emulated a user's interaction with a web site and provided web developers with a method of creating various scripts and providing both functional testing (e.g., did the web site provide the correct response) and load testing (e.g., could the web site handle 5000 users on its web site simultaneously). Among Empirix's customers was H&R Block, who used e-Test Suite to test the tax filing functionality of their web site as well as to test whether the web site could handle a large expected load prior to the filing deadline. During this time, I was also invited to present a guest lecture on software testing (which I presented in several semesters) in an upper-level software project class at MIT, and in which I also mentored a student project group on their Java based web application for their final course project.

18. I have also continued to develop web sites for various business projects, as well as setting up web sites on a volunteer basis for various groups that I am associated with. I have also worked on projects that involved clustering data as well as analyzing text documents to generate vectors that are fed into clustering algorithms.

19. Most of my projects, both academic and commercial, involved client-server architectures. Furthermore, many of my web-based projects utilized concepts that were present in MVC architectures, from the mid-1990's through the 2000's, including implementing custom back end database-backed web pages, and use of the

Zope, Plone, and Ruby on Rails frameworks. For example, our group developed some of the earliest database-backed web pages in the early-to-mid 1990's. Zope and Plone were not strictly MVC, but had overlap with MVC by the use of models and views. Ruby on Rails ("RoR") is an MVC framework, and I developed and managed several public website projects utilizing RoR. To this day, my go-to back end for web development is RoR.

20. I have been awarded several United States patents, and I have several patent applications pending, including the following examples:

- U.S. Patent No. 7,877,736, "Computer language interpretation and optimization for server testing"
- U.S. Patent No. 9,697,230, "Methods and apparatus for dynamic presentation of advertising, factual, and informational content using enhanced metadata in search-driven media applications"
- U.S. Patent No. 9,697,231, "Methods and apparatus for providing virtual media channels based on media search"; and
- U.S. Patent No. 7,975,296, "Automated security threat testing of web pages".

V. LEGAL STANDARDS

21. IBM's attorneys have explained to me the legal standards that apply in this case. My understanding of those standards is described below. I am not an

attorney, and I do not have formal training in the law regarding patents. I have used my understanding of the following legal principles set forth in this section in reaching my opinions.

A. Claim Construction

22. I have been informed and I understand that claim construction is a matter of law and the construction of a claim term will be determined by the Board. In this declaration, I have been informed and I understand that the claim terms should be interpreted under “the Phillips standard.” I have been informed and I understand that under the Phillips standard, claim terms are given their plain and ordinary meaning as understood by a person of ordinary skill in the art at the time of the invention in light of the claim language and the patent specification. I have been informed and I understand that the claim terms, as construed, encompass the embodiments disclosed in the specification. I have been informed and I understand that the patentee can serve as their own lexicographer. As such, if the specification provides a definition of a claim term, I understand that I should interpret that claim term in light of the particular definition provided in the specification.

B. Anticipation

23. I understand that a claim is invalid as anticipated if every element of a claim, as properly construed, is disclosed either explicitly or inherently in a single prior art reference as arranged in the claim.

24. I understand that for a claim element or limitation to be inherently disclosed, the reference must necessarily include the claim element or limitation that is not explicitly disclosed in the prior art reference. I further understand that the mere possibility or probability that an unstated claim element or limitation is present in the prior art reference is not sufficient to support inherent disclosure.

C. Obviousness

25. I understand that a claim is invalid for obviousness if the differences between the invention and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person of ordinary skill in the art (“POSITA”) to which the subject matter pertains. Similarly, I understand that a claim is invalid for obviousness if two or more prior art references in combination disclose, expressly or inherently, every claim limitation so as to render the claim, as a whole, obvious to a POSITA at the time the invention was made.

26. I understand that obviousness is ultimately a question of law based on a determination of a number of factual issues. Those factual issues include: (1) the scope and content of the prior art, (2) the differences between the prior art and the claimed invention as a whole, (3) the level of ordinary skill in the art at the time the invention was made, and (4) objective secondary considerations that reflect the

contemporaneous response to the invention at the time, such as commercial success, long-felt need, and failure of others.

27. I understand that the obviousness inquiry takes place at the time of the invention. Therefore, care must be taken to avoid the impermissible use of hindsight in an obviousness analysis. I understand that it is improper to use the invention as a plan or template for hindsight reconstruction of bits and pieces of the prior art to form the invention. For example, the inventive contribution of a patent may lie in defining a problem in a new way. By merely presenting someone of skill in the art with the identical problem and telling him or her to make the patented invention, it often becomes virtually certain that the artisan will succeed in making the invention. However, the obviousness inquiry must show by clear and convincing evidence that a POSITA at the time of the invention would have recognized the specific problem recognized by the inventor and found it obvious to perform the inventor's methods to solve that problem.

28. It is my further understanding that a patent claim composed of several elements is not proved obvious merely by demonstrating that each of its elements was, independently, known in the prior art. There must be an apparent reason why it would have been obvious to modify the prior art to arrive at the claimed invention. Counsel for IBM has informed me that the patent challenger has the burden to show that a person of ordinary skill in the relevant field had a reason to combine the

elements in the manner claimed when asserting obviousness in view of a combination of references.

29. I also understand that conclusory statements are insufficient to support the legal conclusion of obviousness. Instead, I understand that the Petitioner must articulate a basis on which it concludes that it would have been obvious to make the claimed invention.

30. I understand that there is no suggestion or motivation to make a modification to a prior art reference if the proposed modification would render the prior art invention unsatisfactory for its intended purpose. I also understand that an obviousness allegation cannot be supported by a combination of references that would require a substantial reconstruction and redesign of the elements shown in the primary reference as well as a change in the basic principle under which the primary reference was designed to operate. I further understand that a reference qualifies as prior art for an obviousness determination only when it is analogous to the claimed invention, meaning that it is in the field of the inventor's endeavor or if a person of ordinary skill would reasonably have consulted the reference and applied its teachings in seeking a solution to the problem that the inventor was attempting to solve.

VI. LEVEL OF ORDINARY SKILL IN THE ART

31. I have been informed by IBM's attorneys that obviousness is considered from the perspective of a person of ordinary skill in the art at the time of the invention.

32. I understand that several factors are considered in determining the level of ordinary skill in the art, including the educational level of active workers in the field, the types of problems encountered in the art, the nature of prior art solutions to those problems, prior art patents and publications, the activities of others, the sophistication of the technology involved, and the rapidity of innovations in the field.

33. A POSITA at the time the '719 Patent was filed would have had two years of computer programming and network experience, including UI (user interface) design and being knowledgeable with MVC (Model-view-controller) concepts.

34. I understand that Petitioner alleges that a person of ordinary skill in the art ("POSITA") "would have had at least a bachelor's degree in computer science, computer engineering, or a related field, with at least two years of professional experience in the fields of networking and network-based systems or applications, such as client-server and web-based systems, or an equivalent level of skill, knowledge, and experience. Additional relevant work experience would substitute

for education, and additional relevant educational experience would substitute for work experience.” Pet. at 11.

35. I met or exceeded both Petitioner’s and my definition of a POSITA at the time of the ’719 Patent’s priority date, which is February 8, 2000.

36. The differences between my definition of a POSITA and Petitioner’s definition of POSITA do not materially impact my opinions submitted in this declaration.

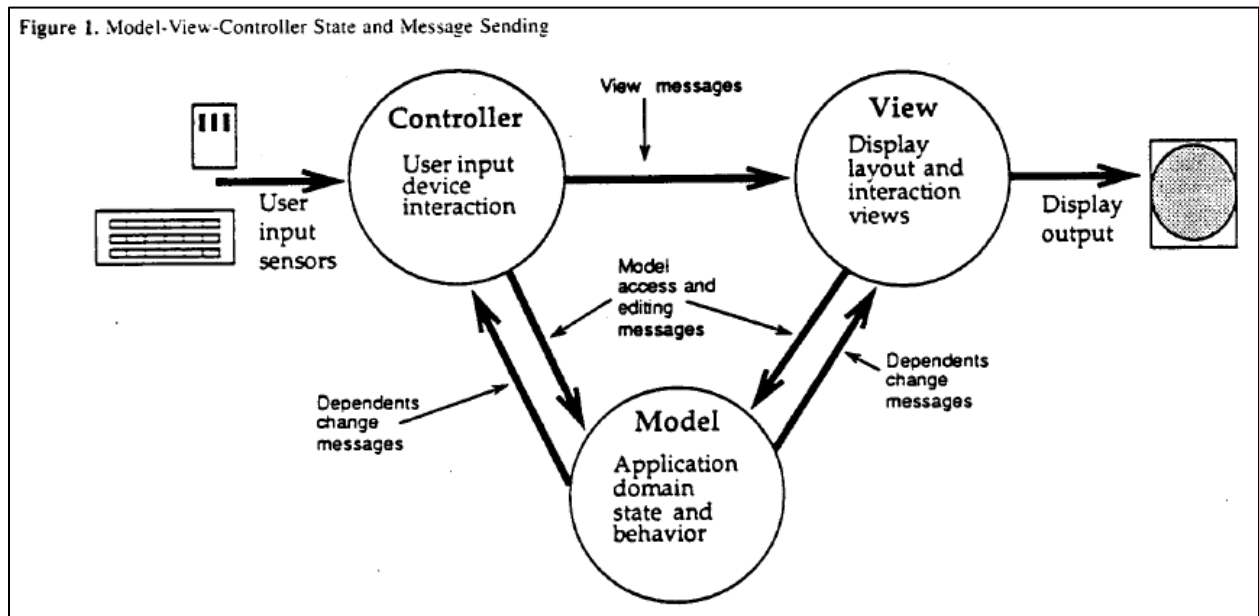
VII. TECHNOLOGICAL BACKGROUND

A. Overview of the ’719 Patent

37. In the 1970s, an Internet users’ client devices connected, via a network, to large mainframe systems (i.e., servers) to execute network-based client/server applications. Ex. 1001 at 1:17-33. At that time, the users’ client devices acted only as a display. That display was used to view the master application data, which was hosted on the server. Ex. 1001 at 1:17-22.

38. In early network-based client/server systems, developers commonly used a Model-View-Controller (“MVC”) architecture. This programming paradigm was introduced in a 1988 paper titled “A Cookbook for Using the Model-View-Controller User Interface Paradigm in Smalltalk-80” by G. E. Krasner and S. T. Pope (“Krasner-Pope”). Ex. 2032; Ex. 1001 at 1:35-41 (incorporating Krasner-Pope by reference). According to Krasner-Pope, the MVC architecture organizes software

into three components: the model manages the application's state, the view presents that state to the user, and the controller handles user input that may change the state or update the display. Ex. 2032 at 1-2; Ex. 1001 at 1:41-47.

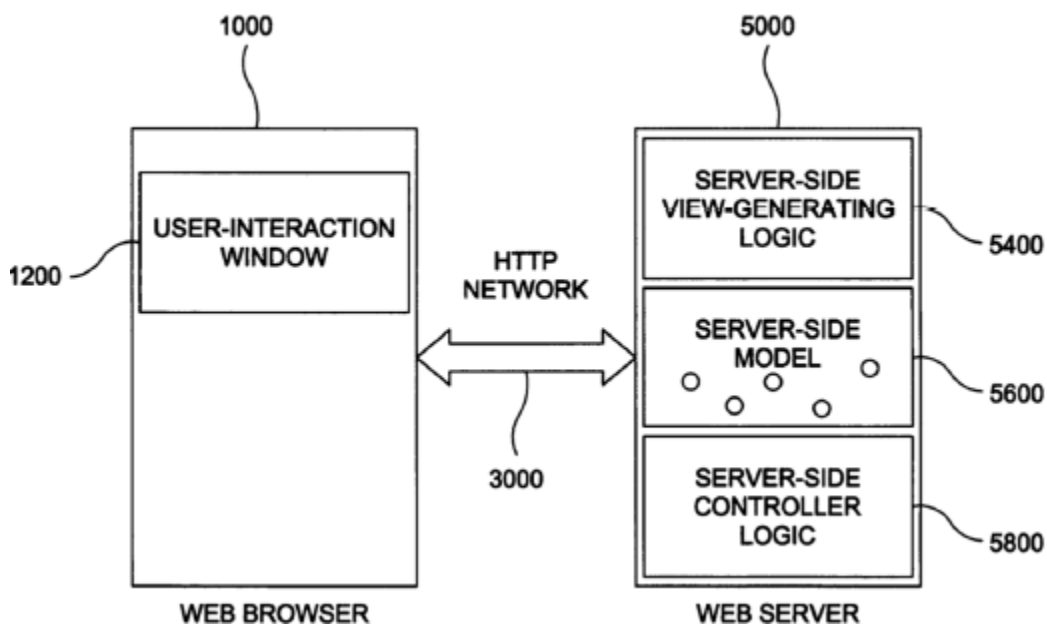


Ex. 2032 at 2.

39. Prior to the invention described in the '719 Patent, network-based applications were usually built using a “fat-client” model, where most of the application’s processing occurred on the client side. Ex. 1001 at 1:34-2:5. This approach was favored in early network applications because it offered greater flexibility and performance. However, it also created significant challenges, particularly in how the applications were distributed and maintained. Ex. 1001 at 1:52-59.

40. With the rise of the World Wide Web, “thin-client” applications—designed to execute primarily on the server—gained popularity. Ex. 1001 at 2:6-14. In contrast to fat-client applications, a thin-client application runs only the browser display logic on the client device, specifically “the application-independent display logic.” Ex. 1001 at 2:6-14. As a result, “[e]ach time the user interacts with the View, the remote server is notified to update the View and/or Model.” Ex. 1001 at 2:6-14.

41. The '719 Patent describes an application employing “a conventional thin-client architecture.” Ex. 1001 at 5:7-9, Fig. 2. In this setup, the client—referred to as the “web browser”—only includes a “user-interaction window.” Meanwhile, the server—referred to as the “web server”—stores the “server-side model” and executes both the “server-side view-generating logic” and the “server-side controller logic.” Ex. 1001, Fig. 2.



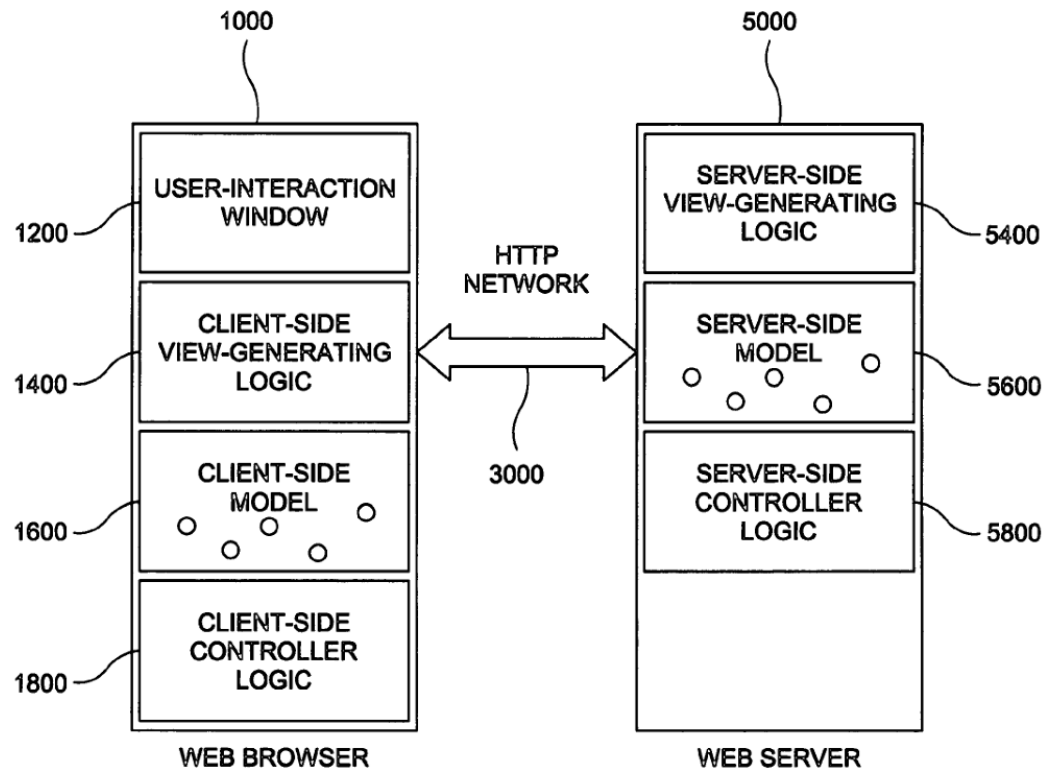
42. Despite their advantages, thin-client applications had drawbacks. In particular, they often suffered from poor performance and excessive latency. This was because the client communicated with the server every time the user made a change to the display, resulting in a great deal of interactions between the client and the server. Ex. 1001 at 2:25-31.

43. To overcome the limitations of both thin-client and fat-client models, the inventors of the '719 Patent invented a novel dual-MVC architecture, where “the client and server both maintain MVC components.” Ex. 1001 at 5:24-26. In this architecture, the server stores the full model and runs the view-generating and controller logic for the application, while the client maintains a subset of that model and executes a corresponding subset of the same view-generating and controller logic that executes on the server. Ex. 1001 at 5:14-26, claim 1.

44. The dual-MVC architecture resolves the latency issues in thin-client applications. The '719 Patent explains that, by storing a subset of the MVC components locally on the client, the client can locally execute a subset of the server's view-generating and controller logic, resulting in fewer interactions between the client and the server. Ex. 1001 at 2:39-49; *see also* Ex. 1001 at 3:14-25, 5:28-30. It also solved the problems associated with fat-client applications. Unlike the fat-client model, the dual-MVC approach “does not require any application

installation or persistence on the client.” Ex. 1001 at 2:47-49. Instead, “the MVC of the application is preferably downloaded from the server on demand by the user of the client device, avoiding the install process, and is preferably not saved permanently to the disk of the client.” Ex. 1001 at 3:1-13.

45. The ’719 Patent illustrates the dual-MVC architecture according to one embodiment of the invention. Ex. 1001 at 4:16-17. In this setup, the “web browser” (i.e., the client) includes a “user-interaction window,” stores the “client-side model,” and executes both the “client-side view-generating logic” and the “client-side controller logic.” Ex. 1001, Fig. 3. Similarly, the “web server” (i.e., the server) stores the “server-side model” and executes both the “server-side view-generating logic” and the “server-side controller logic.” Ex. 1001, Fig. 3.



Ex. 1001, Fig. 3.

46. The dual-MVC architecture decreases the number of interactions required for the client and server to communicate because “the client machine maintains part of the Model.” Ex. 1001 at 3:14-16. This allows the client to respond to user interactions locally, avoiding the performance delays caused by constant server communication: “When part of the Model is maintained on the client, the client has the ability to respond to user interaction without paying the performance penalty of continually interacting with the server.” Ex. 1001 at 3:22-25. Those performance improvements are possible because “both the client and server execute the same model and view-generating and controller logic associated with the

application (albeit, with the client executing at least a subset thereof).” Ex. 1002 at 186 (underline in original).²

47. The parties’ agreed-upon construction reflects the core inventive concept that both the client and the server execute the same model, the same view-generating logic, and the same controller logic, or some subset thereof. Specifically, for claims 1, 13, 25, and 26, the parties agree that the “‘view-generating logic’ and ‘controller logic’ executed on client must be the same ‘view-generating logic’ and ‘controller logic’ executed on the server, or some subset thereof.” Pet. at 12.

B. Prosecution History of the ’719 Patent

48. I have reviewed the file history of the ’719 Patent. Ex. 1002. I summarize only the relevant information that I have reviewed pertaining to my opinions submitted in this declaration.

49. I understand that Exhibit 1002 is a true and correct copy of the file history of the ’719 Patent. The file history has what appears to be the application number of the application (i.e., 09/500,208) and the February 8, 2000, filing date. Ex. 1002 at 1. The application number and the filing date are consistent with the information shown on the cover of the ’719 Patent. Ex. 1001 at Cover. The application was filed with 25 total claims. Ex. 1002 at 17-21. Claims 1, 13, and 25 were independent. Ex. 1002 at 17-21.

² Emphases and annotations added unless otherwise noted.

50. The prosecution history of the '719 Patent further confirms that the view-generating logic and controller logic executed on the client must be the same as the view-generating logic and controller logic executed on the server, or at least a subset of that same logic. The examiner rejected the pending claims of the '719 Patent based on U.S. Patent No. 5,768, 510 ("Gish"), which described the client and server executing different view-generating and controller logic. Ex. 1002 at 184-86; Ex. 1007 (Gish). In response, IBM clarified that "in the claimed approach, both the client and the server execute the same model and view-generating and controller logic associated with the application (albeit, with the client executing at least a subset thereof)." Ex. 1002 at 186 (underline in original).

51. IBM amended the independent claims of the '719 Patent to make clear that, in the claimed invention, both the client and the server execute the same view-generating logic and the same controller logic associated with the application:

1. (Currently Amended) A method for use in a client/server system of reducing interactions between a client and server in association with an application being accessed by the client at the server, the method comprising the steps of:

configuring the server to store a model associated with the application and to execute view-generating and controller logic associated with the application; and

configuring the client to store at least a subset of the model associated with the application and to execute at least a subset of the view-generating and controller logic associated with the application;

wherein one or more portions of the application are performed at the client without the client having to interact with the server, and further wherein the client and the server both locally execute the respective model and view-generating and controller logic ~~resident thereon~~ associated therewith.

Ex. 1002 at 293.

52. As explained below, the Roy prior art reference does not teach or suggest the inventive aspect of configuring both the server and the client to execute the same view-generating and controller logic, or a subset of that logic.

VIII. CLAIM CONSTRUCTION

53. I note that Petitioner and Dr. Bennett acknowledge that “the parties in the Related Litigation disagree on the construction of [certain] claim terms,” but contend that “the ’719 Patent claims are invalid under either set of the proposed constructions.” Pet. at 13; *see also* Ex. 1003, ¶ 18. I further understand that, for the remaining claim elements, Dr. Bennett has “given the term or phrase its plain and ordinary meaning as would be understood by a person of ordinary skill in the art at the time the ’719 Patent was filed.” Ex. 1003, ¶ 19.

54. I have evaluated the ’719 Patent claims under the meanings proposed by each party in the Related Litigation, and, in my view, my opinions regarding the validity of the ’719 Patent do not change regardless of which party’s construction is applied.

IX. PETITIONER’S ASSERTED PRIOR ART

55. I understand that Petitioner challenges the validity of the ’719 Patent’s independent claims (Claims 1, 13, 25, and 26) based only on the Roy reference. *See infra* Section X (Grounds 1 and 2). Accordingly, and for the purposes of this declaration, I do not address Petitioner’s secondary prior art references U.S. Patent No. 5,695,400 (“Fennell”) and JavaScript: The Definitive Guide (“O’Reilly”).

A. Roy (Ex. 1004)

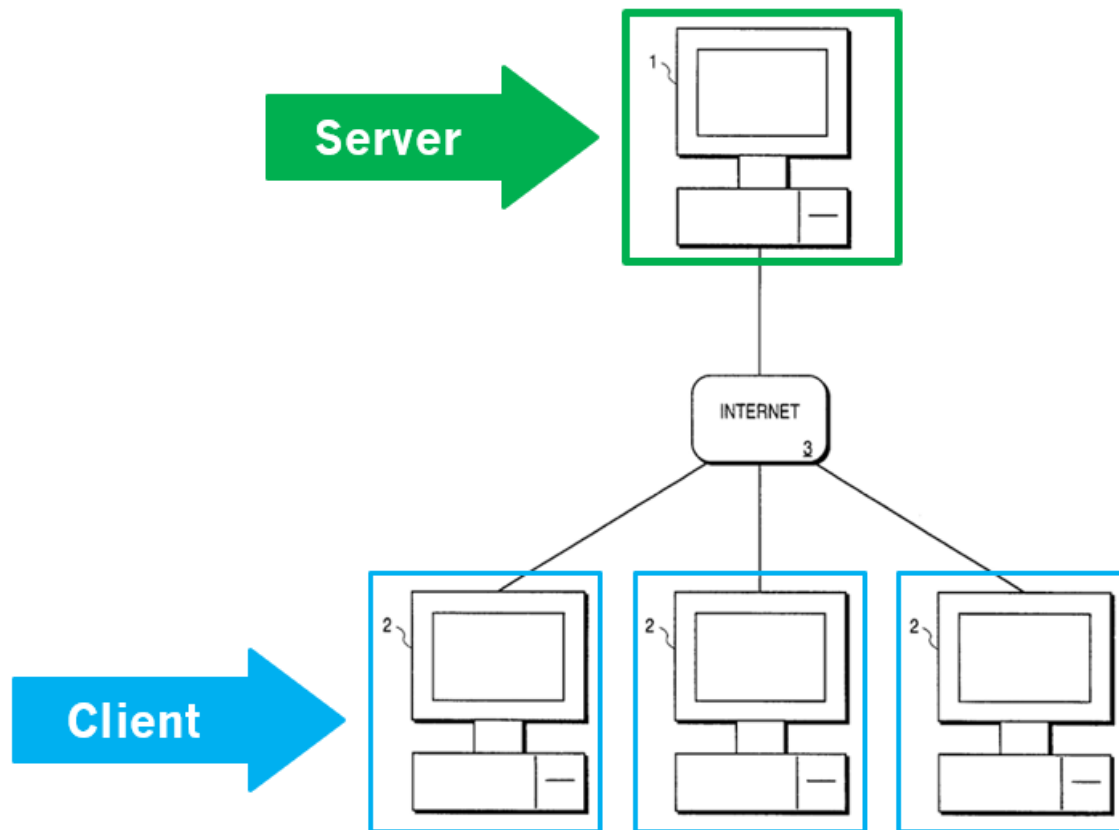
56. U.S. Patent No. 5,838, 909 (“Roy”) is directed towards “[a] method and apparatus for reducing network latency during execution of a multi-player game across a computer network.” Ex. 1004 at Abstract.

57. Roy discloses that a “central server computer 1 is coupled via the Internet 3 to a number of client computer systems 2.” Ex. 1004 at 3:48-51. Roy describes that “every machine that participates in the game runs software that is referred to herein as an ‘event server.’” Ex. 1004 at 4:37-39. Among these, one is designated as the “master event server,” while the others function as “slave event servers.” Ex. 1004 at 4:39-44.

58. Additionally, Roy states that, in the preferred embodiment, the “master database is located on the same machine as the master event server” and that “[s]lave copies of the master database (or portions thereof) are stored on each machine other than the machine in which the master event server is located.” Ex. 1004 at 4:44-48.

59. Roy further states that “by providing slave copies of the database in every other machine involved in the game, the need to send rendering commands across the network is reduced or eliminated, which reduces bandwidth consumption.” Ex. 1004 at 4:51-55.

60. An example of the computer systems participating in the multi-player game disclosed in Roy is shown below in Figure 1.



Ex. 1004, Fig. 1 (annotated).

61. At the start of the online multi-player game—such as a simulated American football game—Players A and B operate computer systems 60 and 62,

respectively. Ex. 1004 at 5:17-36. “Initially, the master database 71 and the master event server 66 are located in a central server computer system 61.” Ex. 1004 at 5:23-25. Each player’s machine—60 for Player A and 62 for Player B—contains a slave event server (65 and 67, respectively) and a slave copy of the master database (68 and 72, respectively). Ex. 1004 at 5:25-29. Player A’s and B’s displays (70 and 74, respectively) are updated using rendering commands (69 and 73, respectively) issued from their local slave database copies (68 and 72, respectively), rather than from the master database 71. This design, depicted in Figure 5A, is intended to reduce bandwidth consumption. Ex. 1004 at 5:29-36.

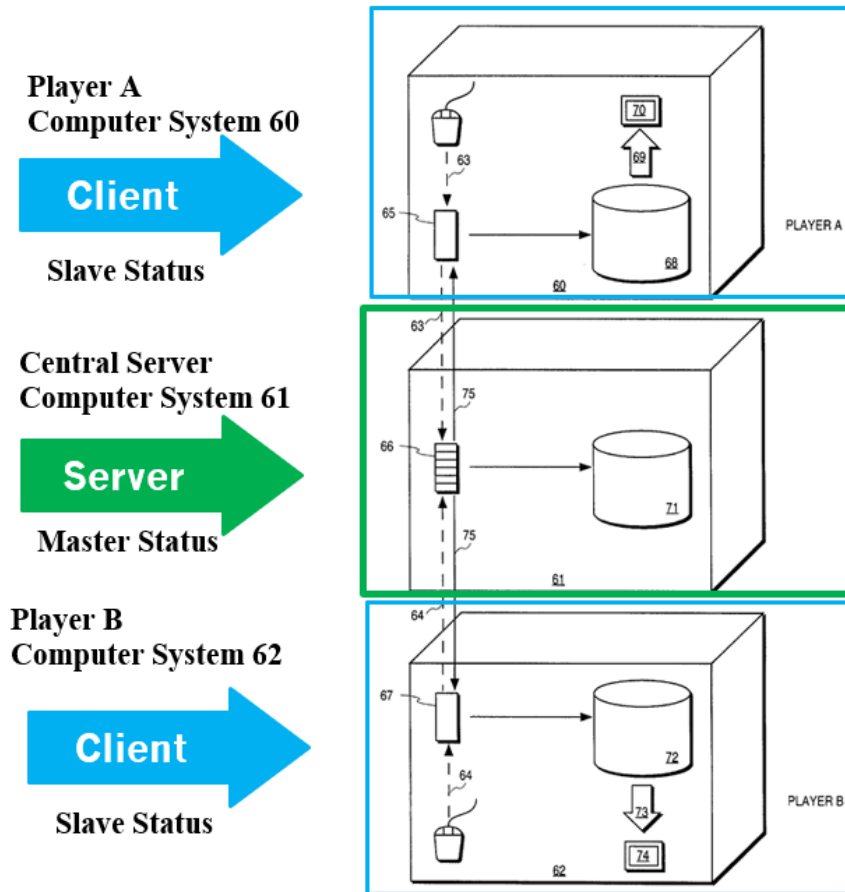


FIG. 5A

Ex. 1004, Fig. 5A (annotated).

62. As the game progresses, Roy describes that “a determination is made of which computer system participating in the game is used by the player having the role which requires the least latency.” Ex. 1004 at 5:65-6:1. Based on that determination, the system migrates the status of the “master” event server and “master” database to the player’s computer (i.e., the player’s client computer system) with the lowest latency requirement. Ex. 1004 at 6:1-2.

63. Roy describes a migration event in which Player B's computer system 62 was identified as requiring the lowest latency. Ex. 1004 at 6:2-14. The migration event causes "the status of event server 66 [to be] changed from master event server to slave event server, and the status of event server 67 [to be] changed from slave event server to master event server," and "the master database 71 is migrated to computer system 62." Ex. 1004 at 6:6-11. As Player A's computer system 60 did not have the most stringent latency requirement, its slave event server 65 maintains its slave status. Ex. 1004 at 6:11-12. Figure 5B illustrates the architecture after such a migration, in which Player B's computer system 62 was identified as needing the lowest latency.

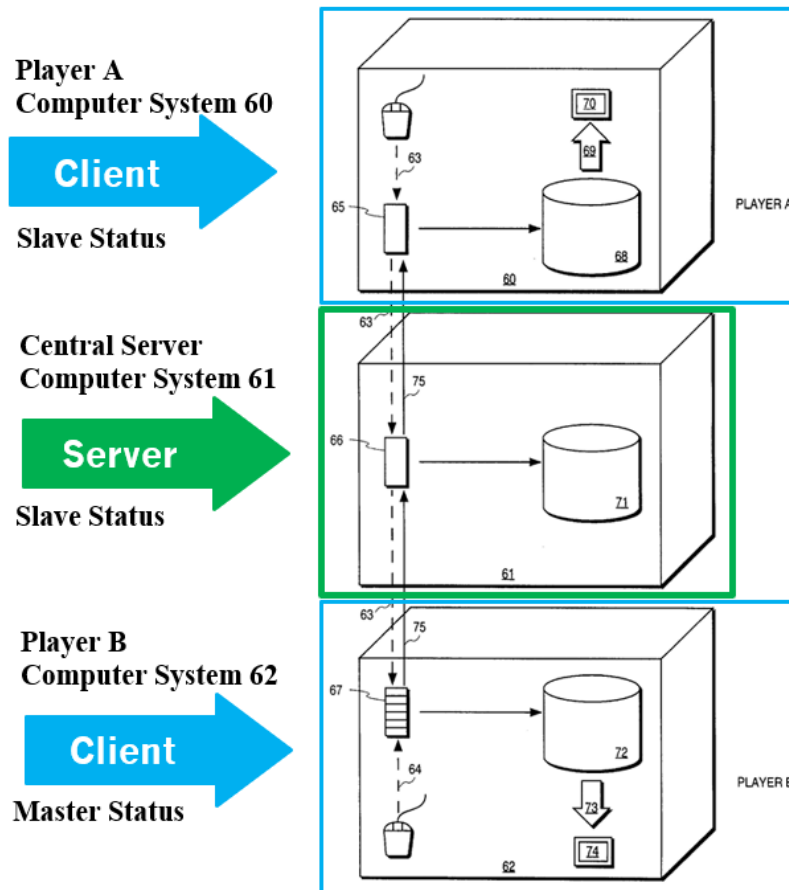


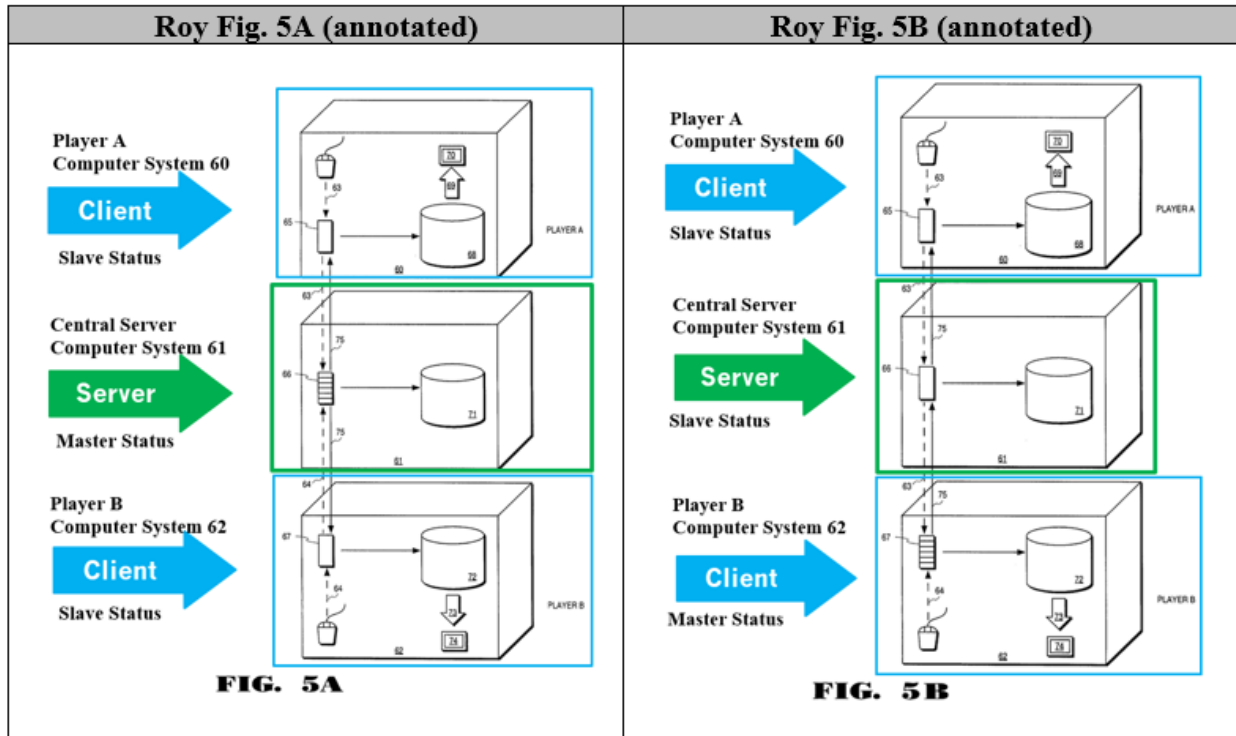
FIG. 5B

Ex. 1004, Fig. 5B (annotated).

64. Roy states that, after a migration event, “it may not be desirable from a latency standpoint (or for other reasons) to require events originating from computer system 60 to pass through the central server computer system 61.” Ex. 1004 at 8:52-56. In other words, after the master event server and master database are transferred to Player B’s client system, the two client systems—A and B—can communicate directly in a peer-to-peer configuration, rather than through a centralized client-server model involving server computer system 61.

65. I understand that Petitioner and Dr. Bennett contend that Player B's computer system 62 is a "server." Pet. at 16 (citing Exhibit 1003, ¶¶ 84-85). Roy, however, continues to describe system 62 as a client device—even after the migration event occurs.

66. For example, claim 52 states: "The program storage medium of claim 50, wherein the method steps further comprise the step of using the *server computer* to forward a slave event received from a *second one of the client computers* to the master event server migrated to the *first one of the client computers*." Ex. 1004 at claim 52. Similarly, claim 53 recites: "The program storage medium of claim 50, wherein the method steps further comprise the step of using the *server computer* to forward a master event received from the master event server migrated to the *first one of the client computers* to a *second one of the client computers*." Ex. 1004 at claim 53. As shown by comparing Roy's Figure 5A with Figure 5B, Player B's computer system 62 remains a client device even after the migration event.



Compare Ex. 1004, Fig. 5B (annotated) with Ex. 1004, Fig. 1 (annotated). After a migration event, the central server computer system 61 (marked in green) remains the “server,” and Player A’s and B’s computer systems 60 and 62, respectively, (marked in blue) remain the “clients.”

X. PETITIONER’S ASSERTED GROUNDS

67. Petitioner asserts the following grounds in its petition. Pet. at 3.

Grounds	References	Challenged Claims
1	Roy	1, 13, 25, 26
2	Roy	1, 13, 25, 26
3	Roy, Fennell, and O’Reilly	2, 4-12, 14, 16-24
4	Roy and Fennell	3, 15

**XI. GROUND 1: PETITIONER DOES NOT SHOW THAT ROY
DISCLOSES THE CHALLENGED CLAIMS 1, 13, 25, AND 26**

68. Petitioner asserts anticipation of independent claims 1, 13, 25, and 26 based on Roy. Petitioner and Dr. Bennett fail to show that Roy discloses all the recited limitations of those independent claims because Roy fails to disclose: (1) “configuring the *server* to . . . execute view-generating . . . logic associated with the application” (Limitations 1a, 13a, 25a, and 26a) (which I will refer to as Limitation a); and (2) “the client and the server both locally maintain at least a portion of the model and execute the view-generating and controller logic associated therewith” (Limitations 1d, 13d, 25d, and 26d) (which I will refer to as Limitation d).

**A. Roy Does Not Disclose “Configuring The Server To . . .
Execute View-Generating . . . Logic Associated With The
Application” As Recited In Limitations 1a, 13a, 25a, 26a
(Claims 1, 13, 25, 26)**

**1. Roy Does Not Disclose “Configuring The Server To .
. . . Execute View-Generating . . . Logic Associated
With The Application”**

69. Limitation a recites “configuring the server to store a model associated with the application and to execute view-generating and controller logic associated with the application.”

70. Petitioner asserts that this limitation is disclosed by Roy’s disclosures regarding Roy’s “computer system 62 (used by Player B),” Ex. 1004 at 5:27-28, because “Player B in Figure 6 demonstrates the server side of Roy’s dual-MVC architecture.” Pet. at 18; *see also* Pet. at 35 (“Server computer 62 executes view-

generating logic when it creates rendering commands 73 used to update the display 74 for Player B.”) (citing Ex. 1003 ¶¶ 144-46).

71. The ’719 Patent states that Player B’s computer system 62 is a client device. For example, Roy’s specification contains express explanations that the “network configuration in which the present invention may be implemented” comprises of a “central *server* computer 1 [that] is coupled via the Internet 3 to a number of *client computer systems* 2.” Ex. 1004 at 3:50-51.

72. In Roy’s embodiments—such as those shown in Figures 5A, 5B, and 6—the “central server computer 1” corresponds to “computer system 61,” and the “client computer systems 2” correspond to computer systems 60 and 62, which are operated by Players A and B. *See* Pet. at 17 n.8 (“[T]he embodiment of Roy described in Fig. 5A uses *computer 61* ‘initially’ as a ‘*central server*.’”) (citing Ex. 1003 ¶ 85 n.54); *see also* Ex. 1004 at Figs. 5A-6; Pet. at 16-18. Roy repeatedly refers to the disclosed system architecture as being composed of “a server computer and a plurality of client computers.” Ex. 1004 at claims 50, 52-55.

73. Petitioner’s own expert, Dr. Bennett, agrees that Player B’s computer system 62 is not a server. I understand that Dr. Bennett has been retained in this IPR to essentially restate opinions he previously gave—unsuccessfully—on behalf of Zynga in *Int’l Bus. Mach. Corp. v. Zynga Inc.*, C.A. No. 1:22-cv-00590-GBW (D.

Del.). During that litigation, Dr. Bennett stated during cross-examination that Roy does not describe computer system 62 as a server:

Q. And figure 6, that's from Roy, the Roy patent?

A. It is.

Q. And over here in the callout, you've labeled this Box 62 as "server." Do you see that?

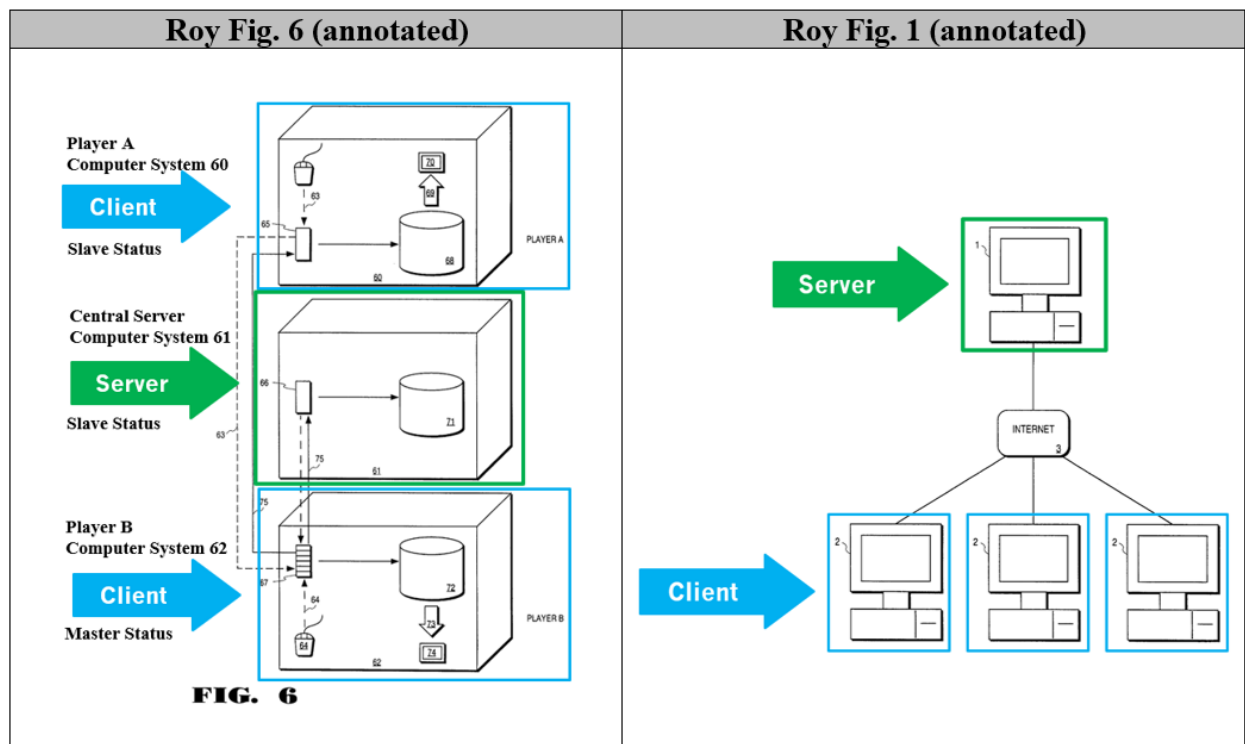
A. Correct.

...

Q. Roy doesn't say that 62 is a server, it doesn't say that, right?

A. No, not specifically.

Ex. 2008 at 1035:9-20.



Compare Ex. 1004, Fig. 6 (annotated) *with* Ex. 1004, Fig. 1 (annotated).

74. Petitioner also appears to agree that Roy’s central computer 61 is the “server”, not Roy’s client computer 62. Pet. at 17 n.8. Despite this, Petitioner states that “[i]n Roy, the computer system containing the master event server and master database is the ‘server’, [and] the other computers are clients.” Pet. at 16 (citing Ex. 1003 ¶¶ 84-85). In particular, Petitioner asserts that once “the master event server and master database (i.e., the ‘server’) have migrated to the most active user [i.e., Player B’s computer system 62 in Figures 5B and 6], ... computer 62 is now the master server.” Pet. at 17 n.8.

75. This argument—that Player B’s computer system 62 transforms from a client to a “server”—is not supported by Roy for two reasons.

76. First, Roy does not disclose that Player B’s computer system 62 transforms into a “server” after a component in that system receives software or data for the master event server and the master database.

77. Roy never refers to Player B’s computer system 62 as a server. Ex. 2008 at 1035:9-20. Instead, Roy describes that the “network configuration in which the present invention may be implemented” as consisting of a “central server computer 1 [that] is coupled via the Internet 3 to a number of client computer systems 2.” Ex. 1004 at 3:48-51.

78. Because Roy does not provide any evidence that computer system 62—which contains the master event server and master database—is the “server” recited in Limitation a, Petitioner instead relies on the declaration of its expert, Dr. Bennett. However, Dr. Bennett simply repeats the same conclusory assertion without offering any new evidence or analysis. *Compare* Pet. at 16 (“In Roy, the computer system containing the master event server and master database is the ‘server.’”) with Ex. 1003 ¶ 85 (“In Roy, the computer system containing the master event server and master database are [sic] considered the ‘server.’”).

79. Second, Roy does not disclose that a migration event converts Player B’s client system 62 into a server. As previously explained, Roy describes the migration event as a mechanism to reduce latency by changing the status of the event server and database on both Player B’s computer system 62 and the central server system 61:

Consequently, master event server 66 is migrated to computer system 62. More specifically, the status of event server 66 is changed from master event server to slave event server, and the status of event server 67 is changed from slave event server to master event server. Similarly, the master database 71 is migrated to computer system 62.

Ex. 1004 at 6:5-11

80. After “the master event server 66 is migrated” Ex. 1004 at 6:5-6 to a client device—such as Player B’s computer system 62—the central server computer

system 61 continues to function as the “server computer,” and Player B’s computer system 62 continues to function as a “client computer.” The migration event merely changes the status of certain software components, namely the event server and database, on each machine. Ex. 1004 at 6:5–11.

81. Roy’s claims also disclose that Player B’s computer system 62 remains a client and central server computer system 61 remains a server after a migration event. *See* Ex. 1004 at claims 50, 52-53. As an example, Roy’s independent claim 50 recites “migrating the master event server from the server computer to the first one of the client computers.” Ex. 1004 at claim 50. Similarly, dependent claims 52 and 53 state that the computer receiving the master event server remains a “client computer,” and that the central system remains a “server computer,” even after the migration:

52. “The program storage medium of claim 50, wherein the method steps further comprise the step of ***using the server computer to forward a slave event*** received from a second one of the client computers ***to the master event server migrated to the first one of the client computers.***”

Ex. 1004 at claim 52 (emphases added).

53. The program storage medium of claim 50, wherein the method steps further comprise the step of ***using the server computer to forward a master event received from the master event server migrated to the first one of the client computers*** to a second one of the client computers.

Ex. 1004 at claim 53 (emphases added); *see also, e.g.*, Ex. 1004 at claims 54-55.

82. In light of Roy’s disclosures that the computer containing the master event server remains a client, Petitioner’s argument that “[i]n Roy, the computer system containing the master event server and master database is the ‘server’, [and] the other computers are clients,” Pet. at 16, is inconsistent with Roy’s teachings. Petitioner thus fails to demonstrate that Roy discloses “configuring the server to store a model associated with the application and to execute view-generating and controller logic associated with the application.”

B. Roy Does Not Disclose “Wherein The Client And The Server Both Locally Maintain At Least A Portion Of The Model And Execute The View-Generating And Controller Logic Associated Therewith” As Recited In Limitations 1d, 13d, 25d, 26d (Claims 1, 13, 25, 26)

83. The term “wherein the client and the server both locally maintain at least a portion of the model and execute the view-generating and controller logic associated therewith” is found in every independent claim of the ’719 Patent. *See* Ex. 1001 at claims 1, 13, 25, 26. *See also* Pet. at 12 (identifying “Relevant Claims in the ’719 Patent”).

84. As discussed above in Section VIII, I understand that the parties agree that the term “the client and the server both locally . . . execute the view-generating and controller logic” should be construed as “‘view-generating logic’ and ‘controller logic’ executed on client *must be the same* ‘view-generating logic’ and ‘controller logic’ executed on the server, or some subset thereof.” Pet. at 12 (emphasis added).

As explained below, in light of this agreed-upon construction, Petitioner has failed to show that Roy discloses Limitation (d).

1. Roy Does Not Disclose That The Client Executes Alleged “View-Generating Logic” That Must Be The Same As The Alleged “View Generating Logic” Executed On The Server, Or A Subset Thereof, As Recited In Limitation (d)

85. In its Ground 1 analysis, Petitioner argues that Roy’s “rendering commands” disclose Limitation (d)’s “view-generating logic” element. Pet. at 43 (“[T]he client and server ‘execute the view-generating . . . logic’ (e.g., rendering commands 69 updating display 70 and rendering commands 73 updating display 74.”).)

86. In particular, Petitioner argues that the disclosed “rendering commands 69” issued in Player A’s computer system 60 (marked in the figure below in blue) is the recited “client . . . execut[ing] the view-generating . . . logic” element, and the disclosed “rendering commands 73” issued in Player B’s computer system 62 (marked in the figure below in green) is the recited “server execut[ing] the view-generating . . . logic” element:

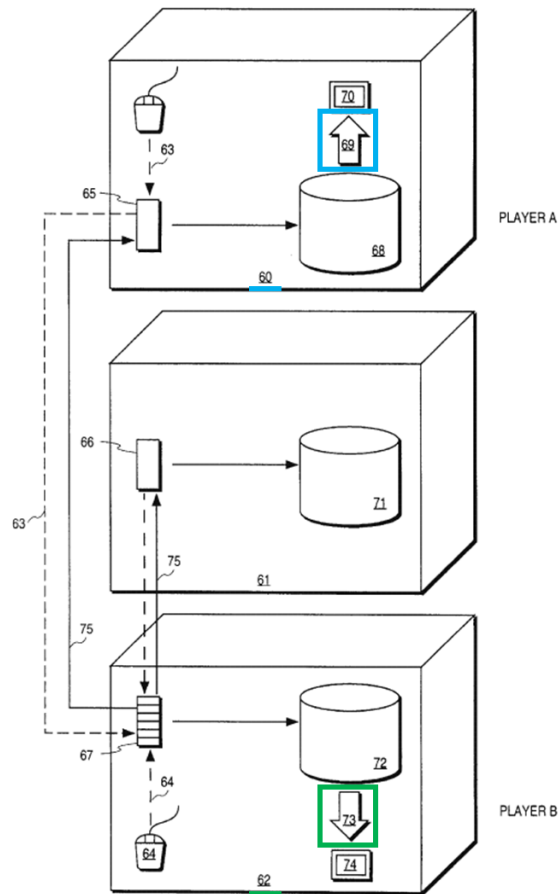


FIG. 6

Ex. 1004 at Fig. 6 (annotations added); Pet. at 43. *See also* Pet. at 16-17 (“Roy likewise describes: . . . (2) **view-generating logic** in client-side **rendering commands 69** and **display 70** and server-side **rendering commands 73** and **display 74**”) (emphases added, coloring in original), 39 (“Roy discloses ‘configuring the client to execute at least a subset of the **view-generating [] logic** associated with the application.’ As an example, Roy discloses **display 70** of client **computer 60** is updated by **rendering commands 69** . . .”).

87. Petitioner’s analysis fails to show that Roy discloses this limitation under the agreed-upon construction: “‘view-generating logic’ . . . executed on client ***must be the same*** ‘view-generating logic’ . . . executed on the server, or some subset thereof.” Pet. at 12 (emphasis added).

88. Petitioner’s analysis relies on the doctrine of inherency. Specifically, Petitioner refers back to its earlier arguments, seemingly its Limitation (b) arguments in particular, and argues that “[a] POSITA would understand that the client-side and server-side view-generating logic . . . are the same, as discussed above.” Pet. at 43. In the referenced argument from Petitioner’s Limitation (b) analysis, Petitioner claims that the alleged view-generating logic (the “rendering commands”) would “***inherently*** be the same on each computer (and therefore the same whether on the server or client) for efficiency, economic, and common sense reasons.” Pet. at 40 (emphasis added).

89. Petitioner further claims that “[i]t would be ***particularly inherent*** to a POSITA because in Roy’s application, each user sees the same world and it would not make sense to use different logic to generate views of the same world for each computer, especially given Roy’s teachings that client and server computers have the same architecture.” Pet. at 40.

90. Petitioner’s inherency argument is deficient for the following reasons explained below.

91. First, Petitioner does not show that Roy meets the standard for inherency, which, as I understand it, requires that the prior art reference ***must necessarily*** include the undisclosed limitation. In fact, Petitioner does not even attempt to argue, let alone show, that Roy teaches that the alleged “view-generating logic” executed on the alleged client (rendering commands 69) ***must necessarily*** be the same as the alleged “view-generating logic” executed on the alleged server (rendering commands 73) or some subset thereof. *See* Pet. at 40, 43.

92. Petitioner claims that “it would not make sense to use different logic” at the alleged client (“computer 60”) and the alleged server (“computer 62”). Pet. at 35, 40. However, Petitioner does not assert that the alleged client ***must necessarily*** use the same view-generating logic, or a subset thereof, as the alleged server. *See*, Pet. at 35, 40.

93. Second, Petitioner ***cannot*** demonstrate that Roy ***inherently*** discloses using rendering commands 69 that are either the same as rendering commands 73 or a subset thereof. For example, Petitioner’s claim that “it would not make sense to use different logic” at the alleged client computer 60 and the alleged server computer 62 is based on Petitioner’s assertion that “in Roy’s application, each user sees the same world.” Pet. at 12, 40.

94. However, this assertion that each user sees the exact same “representation of a subset of the model” is simply incorrect, as can be seen from

Roy's disclosed embodiments. Pet. at 12 (agreed-upon construction of "view generating . . . logic associated with the application" as "program code that generates a screen or window representation of a subset of the model that the application chooses to display"), 40.

95. As one example, Roy discloses an embodiment where users play "a simulated (American) football game in which one player plays offense and a second player plays defense at any given point in time." Ex. 1004 at 3:23-25.

96. In such an embodiment, the player on "offense" would have a point of view that involves interaction options for controlling, for example, "movement of the players, including the speed and direction." Ex. 1004 at 3:23-29. Accordingly, this "offense" player would execute "offense"-oriented program code that would generate the visual representation of this "offense"-oriented point of view of the game world along with the "offense"-oriented interaction options.

97. Meanwhile, by contrast, the player on defense would have a *different* point of view that involves interaction options for controlling, for example, selection of "a predefined defensive pattern." Ex. 1004 at 3:23-33. Accordingly, this "defense" player would execute *different* "defense"-oriented program code that would generate the visual representation of this "defense"-oriented point of view of the game world along with the "defense"-oriented interaction options.

98. In other words, a POSITA would understand from Roy's disclosed embodiment that *different* players will play *different* roles in the game (e.g., offense versus defense) and are therefore expected to see *different* representations of the game world based on those roles.

99. As another example, Roy discloses an embodiment where users play "a role-playing fantasy game" in which players can play either a "superhero" role or a "peripheral" role. Ex. 1004 at 3:35-38.

100. In such an embodiment, the player in the "superhero" role would have a point of view that involves interaction options that correspond to the "very active" "superhero" role. Ex. 1004 at 3:35-39. So this "superhero" player would execute "superhero"-oriented program code that would generate the visual representation of this "superhero"-oriented point of view of the game world along with the superhero-oriented interaction options.

101. Meanwhile, by contrast, the player in the "peripheral" role would have a *different* point of view that involves interaction options for controlling, for example, "setting game conditions and selecting obstacles." Ex. 1004 at 3:35-47. Accordingly, the "peripheral" player would execute *different* "peripheral"-oriented program code that would generate the visual representation of this "peripheral"-oriented point of view of the game world along with the "peripheral"-oriented control options.

102. In other words, a POSITA would also understand from this disclosed embodiment that *different* players will play *different* roles in the game (e.g., superhero versus peripheral) and are therefore expected to see *different* representations of the game world based on those roles.

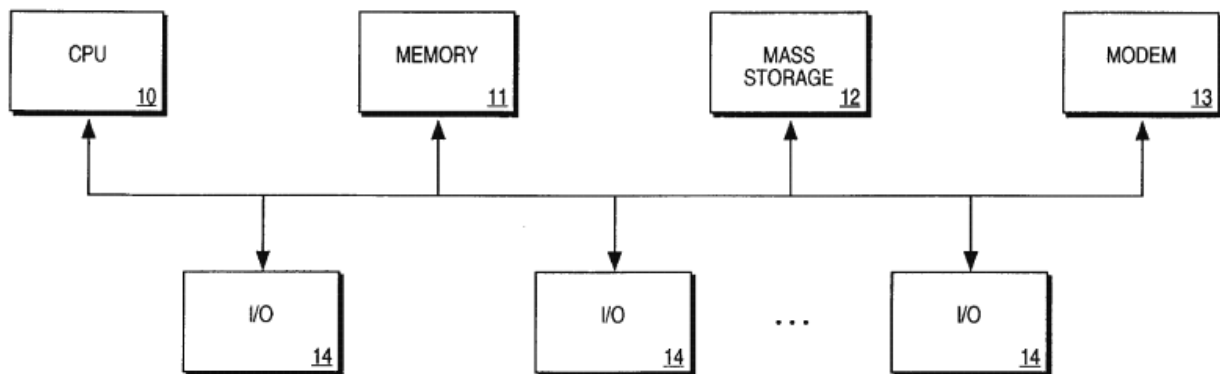
103. Furthermore, Petitioner appears to agree that rendering commands 69 would not necessarily be either the same as the rendering commands 73, or a subset thereof. In a similar fashion, Petitioner's expert, Dr. Bennett, also appears to agree that Roy does not necessarily disclose the missing limitation.

104. Both Petitioner and Dr. Bennett try to support the assertion that "Roy's view-generating logic would inherently be the same" by claiming that "any POSITA would, whenever technically feasible, seek to avoid the expense and inefficiency of developing different logic and program code." Ex. 1003, ¶ 155. *See also* Pet. at 40 (citing Ex.1003¶¶154-155), 45.

105. In making that claim, both Petitioner and Dr. Bennett appear to agree that it is not necessary for the view-generating logic to be the same in the client device and the server. For example, as both Petitioner and Dr. Bennett appear to recognize, a POSITA could, and would, choose to *not* execute the same view-generating logic, or a subset thereof, on both the client and the server whenever it is *not* "technically feasible."

106. Furthermore, neither Petitioner nor Dr. Bennett demonstrates that it would *necessarily* be “technically feasible” in Roy’s disclosed system to execute the same view-generating logic, or a subset thereof, on both the client and the server. Pet. at 40, 43, 45.

107. Petitioner appears to cite Figure 2 from Roy in support of its inherency argument. See Pet. at 40 (citing Ex.1004, Fig. 2, 3:57-59). Roy explains that “FIG. 2 illustrates an example of an architecture which may be used for any of the client computer systems 2 or the server computer system 1.” Ex. 1004 at 3:57-59. As shown below, Figure 2 is a diagram that lists various basic computer hardware components:



Ex. 1004 at Fig. 2.

108. Neither Figure 2 nor Roy’s description of Figure 2 discloses what view-generating logic is used on the client computer system or the server computer system.

109. Therefore, for the reasons provided above, Petitioner fails to demonstrate that Roy discloses that the alleged “view-generating logic” executed on the client must be the same as, or some subset thereof, the alleged “view-generating logic” executed on the server, as recited by Limitation (d) under the agreed-upon construction.

2. Roy Does Not Disclose That The Client Executes Alleged “Controller Logic” That Must Be The Same As The Alleged “Controller Logic” Executed On The Server, Or A Subset Thereof, As Recited In Limitation (d)

110. In its Ground 1 analysis, Petitioner argues that Roy’s “event server” disclose Limitation (d)’s “controller logic” element. Pet. at 43 (“the client and server ‘execute the . . . controller logic’ (e.g., event server 65 processing events 63 and event server 67 processing events 64).”).

111. In particular, Petitioner argues that the disclosed “event server 65” in Player A’s computer system 60 (marked in the figure below in blue) is the recited “client . . . ‘execut[ing] the . . . controller logic” element, and the disclosed “event server 67” issued in Player B’s computer system 62 (marked in the figure below in green) is the recited “server execut[ing] the view-generating . . . logic” element:

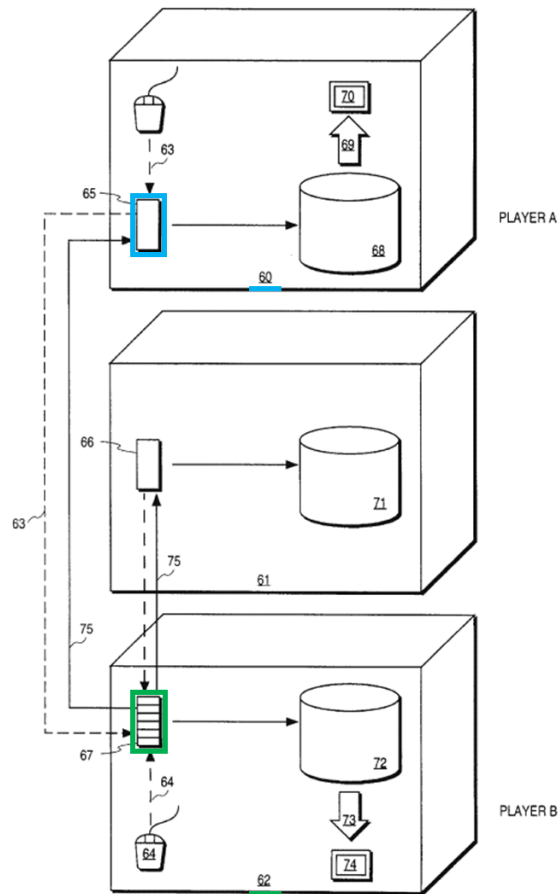


FIG. 6

Ex. 1004 at Fig. 6 (annotations added); Pet. at 43. *See also* Pet. at 16-17 (“Roy likewise describes: . . . (3) **controller logic** in client-side generated events 63 and **event server 65** and server-side generated events 64 and **event server 67**.”) (emphases added, color in original), 36, 40-41.

112. Petitioner’s analysis fails to show that Roy discloses this limitation under the agreed-upon construction: “‘controller logic’ executed on client **must be the same** . . . ‘controller logic’ executed on the server, or some subset thereof.” Pet. at 12 (emphasis added).

113. Petitioner’s analysis relies on the doctrine of inherency. Specifically, Petitioner refers back to its earlier arguments, seemingly its Limitation (b) arguments in particular, and argues that “[a] POSITA would understand that the client-side and server-side . . . controller logic are the same, as discussed above.” Pet. at 43. In the referenced argument from Petitioner’s Limitation (b) analysis, Petitioner claims that the alleged controller logic (the “event servers”) would “*inherently* be the same on each computer (and therefore the same whether on the server or client) for efficiency, economic, and common sense reasons.” Pet. at 41 (emphasis added).

114. Petitioner further claims that because “every participating computer has an event server,” “the same controller logic can be executed on any computer.” Pet. at 41. Petitioner also claims that “[a] POSITA would understand that each computer processes ‘purely local events’ in the same way, such as local events for setting display options.” Pet. at 41. Petitioner’s inherency argument for “controller logic” is deficient for reasons substantially similar to Petitioner’s inherency argument for “view-generating logic,” as explained below.

115. First, Petitioner does not show that Roy meets the standard for inherency, which, as I understand it, requires that the prior art reference *must necessarily* include the undisclosed limitation. In fact, Petitioner does not even attempt to argue, let alone show, that Roy teaches that the alleged “controller logic” executed on the alleged client (event server 65) *must necessarily* be the same as the

alleged “controller logic” executed on the alleged server (event server 67) or some subset thereof. *See* Pet. at 41, 43.

116. Petitioner claims that “every participating computer has an event server . . . [t]hus, the same controller logic can be executed on any computer.” Pet. at 41. Although Petitioner makes claims about what logic a computer “can” possibly execute, Petitioner does not assert that the alleged client *must necessarily* execute the same controller logic, or a subset thereof, as the alleged server.

117. Second, Petitioner *cannot* demonstrate that Roy *inherently* discloses using event server 65 that is either the same as event server 67 or a subset thereof. In particular, Petitioner’s argument that “POSITA would understand that the controller logic would inherently be the same on each computer” at the alleged client computer 60 and the alleged server computer 62 is based on Petitioner’s assertion that “each computer processes ‘purely local events’ in the same way, such as local events for setting display options.” Pet., 12, 41.

118. This assertion that the alleged client and the alleged server would execute the same controller logic simply because they both process “local events,” allegedly in the same way, is incorrect.

119. For example, Roy’s invention is directed to implementing a game based on the presumption that players will always have *different* roles and *different* resulting latency requirements. Roy explains that “[i]n particular, the present

invention is suited to implementing a game . . . in which one player of the game has a role with a latency requirement that is significantly more restrictive than the other players' roles." Ex. 1004 at 3:18-22. *See also* Ex. 1004 at 5:65-6:14.

120. Indeed, as explained above, Roy discloses multiple embodiments where the *different* players have significantly *different* methods of interacting with the game that correspond to those players' *different* roles. *See supra* Section XI.B.1. These embodiments include "a simulated (American) football game" with players in "offense" or "defense" roles, as well as "a role-playing fantasy game" with players can in "superhero" or "peripheral" roles. Ex. 1004 at 3:24-35; 3:35-47. As explained above, a POSITA would understand that in such embodiments, the *different* players' computer systems would have to use *different* controller logic in order to process the *different* events resulting from the players' *different* roles and interactions. *See supra* Section XI.B.1.

121. Furthermore, Petitioner appears to agree that [event server 65](#) would **not** *necessarily* be either the same as the [event server 67](#), or a subset thereof. In a similar fashion, Petitioner's expert, Dr. Bennett, also appears to agree that Roy does **not** *necessarily* disclose the missing limitation.

122. Both Petitioner and Dr. Bennett try to support the assertion that "the controller logic would inherently be the same on each computer" by claiming that a "POSITA would, whenever technically feasible, seek to avoid the expense and

inefficiency of developing different logic and program code.” Ex. 1003, ¶ 158. *See also* Pet. at 41 (citing Ex.1003¶¶158), 45. This is the exact same claim that was made with respect to executing the same “view-generating logic.” *See* Ex. 1003, ¶ 155; Pet. at 40 (citing Ex.1003¶¶154-155), 45.

123. In again making that claim, both Petitioner and Dr. Bennett appear to agree that it is ***not necessary*** for the controller logic to be the same in the client device and the server. For example, as both Petitioner and Dr. Bennett appear to recognize, a POSITA could, and would, choose to ***not*** execute the same controller logic, or a subset thereof, on both the client and the server whenever it is ***not*** “technically feasible.”

124. Furthermore, neither Petitioner nor Dr. Bennett demonstrates, that it would ***necessarily*** be “technically feasible” in Roy’s disclosed system to execute the same controller logic, or a subset thereof, on both the client and the server. Pet. at 41, 43, 45.

125. Therefore, for the reasons provided above, Petitioner fails to demonstrate that Roy discloses that the alleged “controller logic” executed on the client must be the same as, or some subset thereof, the alleged “view-generating logic” executed on the server, as recited by Limitation (d) under the agreed-upon construction.

XII. GROUND 2: PETITIONER DOES NOT SHOW THAT ROY WOULD RENDER OBVIOUS THE CHALLENGED CLAIMS 1, 13, 25, AND 26

126. With respect to Ground 2, Petitioner argues that Roy renders obvious independent claims 1, 13, 25, and 26. *See* Pet. at 44. However, Petitioner’s analysis in support of Ground 2 focuses only on Limitation (d). *See* Pet. at 45 (“Roy’s ‘view-generating logic’ and ‘controller logic’ are the same”). For the other ’719 limitations, Petitioner relies on its Ground 1 arguments. *See* Pet. at 44.

127. As explained in more detail below, Petitioner’s Ground 2 argument fails for two main reasons. First, Petitioner fails to demonstrate that Roy’s disclosed client computer system is a “server” as recited in Limitation (a). Second, Petitioner fails to demonstrate that Roy discloses, teaches, or suggests that the client executes “view-generating logic” and “controller logic” that must be the same as the “view-generating logic” and “controller logic” executed on the server, or a portion thereof.

A. Roy Does Not Disclose, Teach, Or Suggest “Configuring The Server To ... Execute View-Generating ... Logic Associated With The Application” As Recited In Limitations 1a, 13a, 25a, 26a (Claims 1, 13, 25, 26)

128. In its Ground 2 analysis, Petitioner does not provide a separate analysis for Limitation (a)—or any other ’719 claim limitation apart from Limitation (d). *See* Pet. at 44-45.

129. Instead, Petitioner refers back to “the reasons stated above with respect to Ground 1” and argues that “[i]t is well settled that a disclosure that anticipates under § 102 also renders the claim invalid under § 103.” Pet. at 44 n.17.

130. Petitioner does not provide any additional analysis for Limitation (a) in its Ground 2 analysis, and therefore fails to show that Roy discloses Limitation (a) for the same reasons I explained above in Section XI.A.

B. Roy Does Not Disclose, Teach, Or Suggest “Wherein The Client And The Server Both Locally Maintain At Least A Portion Of The Model And Execute The View-Generating And Controller Logic Associated Therewith” As Recited In Limitations 1d, 13d, 25d, 26d (Claims 1, 13, 25, 26)

131. With respect to Petitioner’s Ground 2 analysis for Limitation (d), Petitioner makes one additional argument that that Roy *renders obvious* Limitation (d). *See* Pet. at 45. Petitioner otherwise makes the same flawed inherency argument it made in its Ground 1 analysis for Limitation (d). *See* Pet. at 40-41, 43, 45.

132. As explained below, both Petitioner’s inherency argument as well as its additional argument for Limitation (d) are insufficient to show obviousness.

1. Roy Does Not Disclose, Teach, Or Suggest That The Client Executes Alleged “View-Generating Logic” That Must Be The Same As The Alleged “View Generating Logic” Executed On The Server, Or A Subset Thereof, As Recited In Limitation (d)

133. In its Ground 2 analysis, Petitioner addresses the “view-generating logic” element and the “controller logic” element of Limitation (d) together. *See* Pet.

at 45. Because Petitioner makes the same argument for both the “view-generating logic” element and the “controller logic” element, the deficiencies I explain below are equally applicable to both of these Limitation (d) elements.

134. Petitioner’s additional Ground 2 argument consists of a single sentence claiming that “[a] POSITA would inherently understand that Roy’s ‘view-generating logic’ and ‘controller logic’ are the same whether on the master/server or slave/client system.” Pet. at 45.

135. In support of this claim, Petitioner cites to largely the same allegations Petitioner advanced in its Ground 1 analysis. Pet. at 45 (“POSITA would seek to employ code reuse and *avoid expense and inefficiency* of developing different logic and program code to perform the same task when technically feasible, as *a matter of common sense* and good engineering practice”) (emphases added). *See also* Pet. at 40 (“[A] POSITA would understand that Roy’s view-generating logic would inherently be the same on each computer . . . for *efficiency, economic, and common sense* reasons.”) (emphases added). These unsupported allegations, and Petitioner’s inherency argument, fail for the same reasons I explained above with respect to Ground 1 in Section XI.B.

136. Insofar as Petitioner is now also arguing that it would have been obvious to *modify* Roy such that the alleged “view-generating logic” executed on

the client “must be the same as” the alleged “view-generating logic” executed on the server, or some subset thereof, Petitioner’s argument fails for the reasons below.

137. First, Petitioner fails to identify or explain how or why a POSITA would have modified Roy such that the same “view-generating logic” and “controller logic,” or a subset thereof, would be executed on both the client and the server. *See* Pet. at 45.

138. Second, Petitioner’s obviousness allegations appear to rely solely on Petitioner’s inherency arguments. *See* Pet. at 45. For example, Petitioner claims that “POSITA would seek to employ code reuse and avoid the expense and inefficiency of developing different logic and program code to perform the same task when technically feasible, as a matter of common sense and good engineering practice.” Pet. at 45 (citing Ex.1003, ¶¶154-155, 167). Petitioner’s arguments fail for at least the same reasons explained above and in Section XI.B.

139. For the reasons provided above and in Section XI.B, Petitioner fails to demonstrate that Roy renders obvious that the alleged “view-generating logic” and “controller logic” executed on the client must be the same as, or some subset thereof, the alleged “view-generating logic” and “controller logic” executed on the server, as recited by Limitation (d).

XIII. GROUNDS 3-4: PETITIONER DOES NOT SHOW THAT THE COMBINATIONS OF ROY WITH O'REILLY AND/OR FENNEL WOULD RENDER OBVIOUS THE CHALLENGED CLAIMS.

140. Petitioner's Ground 3 analysis is solely directed towards dependent claims 2, 4-12, 14, and 16-24. Pet. at 3. Petitioner's Ground 4 analysis is solely directed towards dependent claims 3 and 15. Pet. at 3. These dependent claims all incorporate by reference the limitations of underlying independent claims 1 and 13. *See* Ex. 1004 at claims 2-12, 14-24.

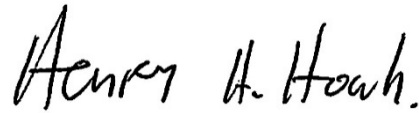
141. The newly cited references, O'Reilly and Fennell, do not address any of the deficiencies in Grounds 1 and 2, from which Grounds 3 and 4 depend, nor does Petitioner use either reference to show any claimed limitation of independent claims 1 and 13. *See* Pet. at 46-77 (addressing solely dependent claim limitations). Accordingly, all deficiencies identified with respect to Grounds 1 and 2 above apply equally to Grounds 3 and 4.

142. Because Petitioner does not rely on either O'Reilly and/or Fennell to cure any of the deficiencies of Ground 1 or Ground 2 discussed above in Sections XI and XII, for those same reasons Petitioner fails to show that Ground 3 renders obvious dependent claims 2, 4-12, 14, and 16-24, and that Ground 4 renders obvious dependent claims 3 and 15.

XIV. DECLARATION IN LIEU OF OATH

143. I hereby declare that all statements made herein of my own knowledge are true and that all statements made on information and belief are believed to be true; and further that these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under § 1001 of Title 18 of the United States Code. If called to testify as to the truth of the matters stated herein, I could and would testify competently.

I declare under penalty of perjury that the foregoing is true and correct.

A handwritten signature in black ink, reading "Henry H. Houh." with a period at the end.

Henry Houh, Ph.D.

Date: June 10, 2025