

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

SHOPIFY INC.
Petitioner

v.

DKR CONSULTING LLC
Patent Owner

IPR2025-00132
U.S. Patent No. 11,157,995

**DECLARATION OF TODD MOWRY, PH.D.
U.S. PATENT NO. 11,157,995
CHALLENGING CLAIMS 1–14**

TABLE OF CONTENTS

	Page
I. INTRODUCTION	1
II. QUALIFICATIONS AND PROFESSIONAL EXPERIENCE	1
III. MATERIALS CONSIDERED	5
IV. LEGAL PRINCIPLES.....	6
A. Claim Construction	6
B. Anticipation	7
C. Obviousness.....	7
V. LEVEL OF ORDINARY SKILL	11
VI. TECHNICAL BACKGROUND	11
A. Electronic Commerce	11
B. Electronic Commerce in Social Networks	12
C. Widgets for Product Listings.....	13
VII. OVERVIEW OF THE '995 PATENT	14
A. Alleged Invention	14
B. Prosecution History	15
VIII. OVERVIEW OF THE PRIOR ART REFERENCES	16
A. Johnston.....	16
1. Johnston is Entitled to its Provisional Filing Date.....	21
B. Wiser.....	24
C. Dierks	26
D. Chang.....	29
IX. SPECIFIC GROUNDS FOR UNPATENTABILITY	31
A. Ground 1: Claims 1–14 are Obvious over Johnston and Dierks.....	31
1. Independent Claim 1	31
2. Dependent Claim 2	63

3.	Dependent Claim 3	65
4.	Dependent Claim 4	66
5.	Dependent Claim 5	67
6.	Independent Claim 6	68
7.	Dependent Claim 7	71
8.	Dependent Claim 8	71
9.	Dependent Claim 9	73
10.	Dependent Claim 10	75
11.	Independent Claim 11	76
12.	Dependent Claim 12	80
13.	Dependent Claim 13	82
14.	Dependent Claim 14	82
B.	Ground 2: Claims 1–14 are Obvious over Wiser, Dierks, and Chang	83
1.	Independent Claim 1	83
2.	Dependent Claim 2	118
3.	Dependent Claim 3	120
4.	Dependent Claim 4	120
5.	Dependent Claim 5	120
6.	Independent Claim 6	121
7.	Dependent Claim 7	123
8.	Dependent Claim 8	124
9.	Dependent Claim 9	125
10.	Dependent Claim 10	125
11.	Independent Claim 11	127
12.	Dependent Claim 12	129
13.	Dependent Claim 13	130
14.	Dependent Claim 14	130
X.	CONCLUSION.....	131
XI.	AVAILABILITY FOR CROSS-EXAMINATION	131
XII.	RIGHT TO SUPPLEMENT	131
XIII.	JURAT	132

I, Todd Mowry, declare as follows:

I. INTRODUCTION

1. My name is Todd Mowry.

2. I have been retained as an expert in this proceeding by counsel for Shopify Inc. (“Shopify”) to provide my expert conclusions regarding the validity of claims 1–14 of U.S. Patent No. 11,157,995 (the “’995 patent”). For the reasons set forth below, it is my conclusion that claims 1–14 of the ’995 patent are invalid.

II. QUALIFICATIONS AND PROFESSIONAL EXPERIENCE

3. My qualifications are stated more fully in my curriculum vitae, which is attached as Exhibit A. Below is a summary of my education, work experience, and other qualifications.

4. I am a Professor in the Department of Computer Science at Carnegie Mellon University. I also have a courtesy appointment in the Department of Electrical and Computer Engineering. I have served on the faculty of Carnegie Mellon University for 27 years starting in 1997 through the present (2024).

5. I also served on the faculty of the University of Toronto for four years between 1993 and 1997, in the Department of Electrical and Computer Engineering and a courtesy appointment in the Department of Computer Science. Prior to that appointment, I served as a Graduate Research Assistant in the Department of Electrical Engineering at Stanford University for four years between 1989 and 1993.

6. As a faculty member, I have taught and continue to teach courses and directed research in computer systems and software, operating systems, distributed and network systems, parallel processing, and compiler optimization.

7. I received a B.S. degree in Electrical Engineering with Highest Distinction from the University of Virginia in May 1988. I received an M.S. in Electrical Engineering from Stanford University in June 1989, and a Ph.D. in Electrical Engineering from Stanford University in March 1994.

8. I have worked in the computer industry in various capacities. I was a part-time Computer Architect and then Computer Architecture Consultant at Silicon Graphics, Inc., in Mountain View, California (formerly MIPS Computer Systems in Sunnyvale, California) from 1989 to 1993 and 1993 to 1996, respectively. I was a Visiting Scientist at IBM in Toronto from 1996 to 2004. During that same time period (1996 to 2004), I was also a Member of the Technical Advisory Board of SandCraft, Inc. in Santa Clara, California. I was the Director of the Intel Research Pittsburgh Lab at Intel Corporation in Pittsburgh, Pennsylvania from 2004 to 2007.

9. I have authored 21 journal articles and 71 conference papers. I am also an inventor on five patents.

10. I have published a number of papers in the top research conferences in the fields of operating systems and data storage (including a paper that won the Best Paper Award at the USENIX 2nd Symposium on Operating Systems Design and

Implementation (OSDI '96)). *See, e.g.,* Olatunji Ruwase, Michael A. Kozuch, Phillip B. Gibbons, and Todd C. Mowry, *Guardrail: High Fidelity Correctness Checking of Device Drivers for Safeguarding I/O Operations*, in Proceedings of the Nineteenth International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS) (2014); Andrew Pavlo, Gustavo Angulo, Joy Arulraj, Haibin Lin, Jiexi Lin, Lin Ma, Prashanth Menon, Todd Mowry, Matthew Perron, Ian Quah, Siddharth Santurkar, Anthony Tomasic, Skye Toor, Dana Van Aken, Ziqi Wang, Yingjun Wu, Ran Xian, and Tieying Zhang, *Self-Driving Database Management Systems*, in Proceedings of the 2017 Conference on Innovative Data Systems Research (CIDR) (2017); and Amit Manjhi, Charles Garrod, Bruce M. Maggs, Todd C. Mowry, Anthony Tomasic, *Holistic Query Transformations for Dynamic Web Applications*, in 2009 IEEE 25th International Conference on Data Engineering (ICDE) (2009).

11. I have also been an active member of the Parallel Data Lab at Carnegie Mellon University since 1997, which is often described as “academia’s premiere storage systems research center.”

12. I am the recipient of several honors and awards: the Arthur Samuel Thesis Award (awarded by the Stanford Computer Science department to the top two Ph.D. theses in a given year), several IBM Faculty Development Awards (1996, 1997, 1998, 2000, 2001, 2002, and 2003), several Best Paper Awards (the Second

Symposium on Operating Systems Design and Implementation in 1996; the 20th International Conference on Data Engineering (ICDE) in 2004), the Alfred P. Sloan Research Fellowship (awarded to researchers in recognition of distinguished performance and a unique potential to make substantial contributions to their field), the Most Thought-Provoking Idea Award in 2004 (awarded by the Architectural Support for Programming Languages and Operating Systems (ASPLOS)), the TR100 Award in 1999 (awarded by MIT's Technology Review magazine to the top 100 most promising young innovators in science and technology), and I became an ACM Fellow in 2016.

13. I am a member of the Institute of Electrical and Electronics Engineers (IEEE) and the Association of Computing Machinery (ACM). I was the Editor-in-Chief of ACM Transactions on Computer Systems (the premier journal for computer systems research) from 2013–2018. I was an Associate Editor for the journal prior to that, since 2001. I was the Program Chair of the International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS) in 2010. I was the Co-Program Chair of the International Conference on Parallel Architectures and Compilation Techniques (PACT) in 2001. I have been on the programming committee in various years for ASPLOS, the International Symposium on Computer Architecture (ISCA), the International Symposium on

Microarchitectures, and the Workshop on Architectural and System Support for Improving Format.

14. Overall, I have over 30 years of experience in the field of computer science and, specifically, computer architecture, operating systems, and document storage and retrieval.

15. In light of the foregoing, I consider myself to be an expert in the field of computer science, and believe that I am qualified to provide an opinion as to what a person of ordinary skill in the art would have understood, known, or concluded regarding the subject matter of the '995 patent at the time of its alleged invention.

16. I am being compensated for my time at my ordinary hourly rate of \$650. My compensation is not dependent on the outcome of these proceedings or the content of my opinions. To the best of my knowledge, I have no financial interest in Shopify. I have no financial interest in the '995 patent.

III. MATERIALS CONSIDERED

17. In preparing this declaration, I have reviewed the claims, specification, and file history of the '995 patent.

18. I have reviewed and understand the references and exhibits cited in this declaration, including the following:

Exhibit	Description
1001	U.S. Patent No. 11,157,995
1003	U.S. Patent No. 6,868,403 (“Wiser”)
1004	U.S. Patent Application Publication No. 2008/0183593 (“Dierks”)
1005	U.S. Patent Application Publication No. 2010/0114739 (“Johnston”)
1006	U.S. Patent Application Publication No. 2010/0250382 (“Babaria”)
1008	U.S. Patent Application Publication No. 2008/0255962 (“Chang”)
1009	’995 Patent File History
1010	U.S. Provisional Application No. 61/093,917 (“Johnston Provisional”)
1011	U.S. Patent No. 9,324,098 (“Agrawal”)
1012	U.S. Patent No. 7,565,308 (“Bollay”)
1013	U.S. Patent Application Publication No. 2008/0228595 (“Hill”)
1022	U.S. Patent No. 9,203,971 (“Narain”)

IV. LEGAL PRINCIPLES

19. I am not an attorney. For purposes of this declaration, I have been informed by Shopify’s counsel about certain aspects of the law that are relevant to my analysis and opinions, as stated in the following paragraphs. I have applied these legal principles in arriving at my conclusions expressed in this declaration.

A. Claim Construction

20. I understand that “claim construction” is the process of determining a patent claim’s meaning. I have been informed that claim construction is a matter of law and that in this proceeding, claim construction will be determined by the Patent Trial and Appeal Board (“the Board”). I understand that, during IPR, claims are construed according to the “*Phillips* standard.” I understand that the Board need only construe the claims when necessary to resolve the underlying controversy. I also understand that, given the close correlation between the asserted prior art and the challenged claims of the ’995 patent, the Board need not construe any terms of the challenged claims to resolve the underlying controversy, as any reasonable interpretation of those terms consistent with their plain meaning (as would have been understood by a person of ordinary skill in the art (“POSITA”) at the time of the invention, having taken into consideration the language of the claims, the specification, and the prosecution history of record) reads on the prior art.

B. Anticipation

21. I have been informed and understand that a claim is invalid if it was anticipated or not novel. I also have been informed and understand that a claim is anticipated if each and every element as set forth in the claim is found, either expressly or inherently described, in a single prior art reference.

C. Obviousness

22. I have been informed and understand that a patent claim is invalid if it would have been obvious to a POSITA at the time of the alleged invention. This means that, even if all of the requirements of a claim are not found in a single prior art reference, the claim is not patentable if the differences between the subject matter in the prior art and the subject matter in the claim would have been obvious to a POSITA at the time of the alleged invention.

23. I have been informed and understand that a determination of whether a claim was obvious should be based upon several factors, including, among others: (1) the level of ordinary skill in the art at the time of the alleged invention; (2) the scope and content of the prior art; (3) the differences, if any, that existed between the claimed invention and the prior art; and (4) any objective considerations of nonobviousness.

24. I have been informed and understand that when determining any differences between the invention covered by the patent claims and the prior art, one should not look at the individual differences between the patent claims and the prior art in isolation, but rather consider the claimed invention as a whole and determine whether or not it would have been obvious in light of all of the prior art.

25. I have been informed and understand that a single reference can render a patent claim obvious if any differences between that reference and the claims would have been obvious to a POSITA. Alternatively, the teachings of two or more

references may be combined in the same way as disclosed in the claims. I have been informed and understand that when deciding whether to combine the various items described in the prior art, the relevant question is whether the prior art combination would have been obvious to a person with ordinary skill in the art at the time of the invention. I have been further informed that it can be important to identify a teaching, suggestion, or motivation, in either the prior art or the knowledge of persons skilled in the art, that would have prompted a person of ordinary skill in the relevant field to combine the elements in the prior art in the way the claimed invention does.

26. In determining whether a combination based on either a single reference or multiple references would have been obvious, it is appropriate to consider, among other factors:

- whether the teachings of the prior art references disclose known concepts combined in familiar ways, and when combined, would yield predictable results;
- whether a POSITA would implement a predictable variation, and would see the benefit of doing so;
- whether the claimed elements represent one of a limited number of known design choices, and would have a reasonable expectation of success by a POSITA;

- whether a POSITA would have recognized a reason to combine known elements in the manner described in the claim;
- whether there is some teaching or suggestion in the prior art to make the modification or combination of elements claimed in the patent; and
- whether the innovation applies a known technique that had been used to improve a similar device or method in a similar way.

27. I have been informed and understand that a POSITA has ordinary creativity, and is not an automaton. I have been informed and understand that in considering obviousness, it is important not to determine obviousness using the benefit of hindsight derived from the patent being considered. I have been informed and understand that a patent composed of several elements is not proved obvious merely by demonstrating that each of its elements was independently known in the prior art.

28. I have been informed and understand that to show a motivation to combine, one is not required to show how the references would be physically combined to render a claim obvious. The analysis is not focused on whether the features of one reference can be physically incorporated into the structure of another reference, but rather whether a POSITA would have been motivated to combine the teachings of the references to achieve the claimed invention. For example, a POSITA would understand how predictable variations of known concepts could be

combined and would recognize that where multiple tools were available in a toolbox, any could be chosen.

V. LEVEL OF ORDINARY SKILL

29. I have considered the '995 patent from the perspective of a POSITA at the time of the alleged invention. A POSITA for the '995 patent would have had at least the equivalent of a bachelor's degree in computer science, computer engineering, or a related subject and two or more years of experience in a related field of distributed systems, transaction processing, and web-based user interfaces. Less work experience may be compensated by a higher level of education, such as a master's degree, and vice versa. At the time of the claimed invention, I would have qualified as a POSITA under this standard.

VI. TECHNICAL BACKGROUND

A. Electronic Commerce

30. Electronic commerce—the trading of goods and services over the Internet—was well-known at least more than a decade before the priority date of the '995 patent. For example, Wiser, which claims priority to a provisional application filed in 1998, explained that “[t]he rapid development of the Internet and the World Wide Web has primarily focused on these technologies as vehicles for online commerce for the distribution of their products.” Exhibit 1003, 1:15–17; *see also* Exhibit 1006, ¶ 4 (“Merchants, service providers, and/or private individuals, may

offer items for sale on a web page appearing on a website accessible by a browser via the internet.”); Exhibit 1011, 1:15–16 (“Consumers routinely shop for and purchase products and services from merchant web sites.”); Exhibit 1012, 1:15–17 (“Internet electronic commerce allows businesses to conduct sales transactions over the Internet between customers (buyers) and vendors supplying goods and services.”); Exhibit 1013, ¶ 4 (“Electronic commerce sites allowing users to post and sell their own items (or items on which they receive a commission) have also become popular.”). Indeed, the ’995 patent itself acknowledges that “[e]lectronic commerce on the Internet has become commonplace.” Exhibit 1001, 1:25–26.

B. Electronic Commerce in Social Networks

31. Performing electronic commerce in social networks was also well-known before the priority date of the ’995 patent. For example, USP Application Publication No. 2008/0228595 to Hill, filed in March 2007, disclosed a “system for supporting electronic commerce in social networks.” Exhibit 1013, Title; *see also id.*, ¶ 8 (“The present invention provides users the ability to add commerce capability (e.g., buy and sell goods and services) and syndicate product information to numerous social network third-party websites.”); *see also* Exhibit 1010 (Johnston-Provisional), specification at 7 (“The seller can be selling their content on Facebook, Bebo, and MySpace in a few minutes. ... The Storefront can then be placed on social networking sites such as Facebook, MySpace, Bebo and can also be placed on the

artist's website if they so wish.”). Similarly, Chang, filed in 2007, disclosed a distributed commerce system for distributing an “item page” that includes an item for sale to “social networking servers reflecting a plurality of social networking sites.” Exhibit 1008, Abstract, ¶ 41; *see also id.*, ¶ 41 (“These social networking sites may have trade names including FACEBOOK™, MYSPACE™, LINKED-IN™, or some other suitable social networking site.”).

C. Widgets for Product Listings

32. Given the proliferation of electronic commerce over the Internet, including social networks, it became common to use “widgets” or widget code to allow sellers to more easily and freely advertise (i.e., “list”) their products for sale. A “widget” is simply a portable piece of code that can be installed and executed within an environment using the same or a different programming language. *See, e.g.*, Exhibit 1008, ¶ 31 (“In some example embodiments, a widget may be a portable piece of code that can be installed and executed within any separate eXtensible Markup Language (XML) and/or Hyper Text Markup Language (HTML) based web page. Widgets may be implemented using technology and programming techniques that includes Dynamic HTML (DHTML), Asynchronous JavaScript and XML (AJAX), FLASH™, a Java applet or some other suitable technology.”). For example, Chang disclosed widget servers that generate widget code for item/web pages related to items for sale that could be distributed to social networking sites.

See, e.g., id., ¶¶ 37, 41, Figure 1. Notably, systems were known to gather pertinent information from sellers and generate such widget code. For instance, Dierks in 2008 disclosed a user interface that “presents a user with an easy, step-by-step process for entering relevant information and obtaining ... code” that could be “cut-and-pasted into the user’s web page.” Exhibit 1004, ¶¶ 71, 74. Likewise, Johnston disclosed a server that provides user interfaces that allow sellers to generate “Embed code” and publish their product listings to the sellers’ social networking sites and “any HTML web site.” Exhibit 1005, Figures 3A–3G, ¶¶ 225–231. The widget code in the prior art comports with the ’995 patent’s description. *See* Exhibit 1001, 1:62–65 (“A web widget is portable software that can be installed and executed within a hypertext-markup-language web page by an end user that does not require additional compilation.”).

VII. OVERVIEW OF THE ’995 PATENT

A. Alleged Invention

33. Titled “[s]ystem and method for generating and distributing embeddable electronic commerce stores,” the ’995 patent generally relates to systems for the “generating an embeddable electronic commerce store.” Exhibit 1001, Abstract. The claims of the ’995 patent describe a system with a server that is configured to generate and embed a product listing into a web site and into a social media network, and control a transaction processing service to process a payment

for a purchase of a product associated with the product listing. *See* Exhibit 1001, claims 1, 6, and 11.

34. As I will demonstrate in the grounds below, a server generating and embedding product listings into a web site and into a social media network and processing payments for a purchase was well-known in the art before the '995 patent's priority date.

B. Prosecution History

35. The '995 patent issued from application No. 17/104,224, filed on November 25, 2020, as a continuation to application No. 16/512,018, filed on July 15, 2019, now the '678 patent. Both applications claim priority to application No. 12/852,090, filed on August 6, 2010.

36. The applicant was unsuccessful in overcoming the Examiner's first round of rejections. *See* Exhibit 1009 at 207–217. Therefore, the applicant heavily amended the claims and requested for continued examination. *See id.* at 445–463. The Examiner issued a second non-final Office Action, which was followed by further claim amendments by both the applicant and the Examiner, resulting in a Notice of Allowance on September 17, 2021. *See id.* at 551–562, 571–583.

37. In the Notice of Allowance, the Examiner simply stated that the closest prior art on record “teaches product configuration and content distribution system and method” but “does not disclose or render obvious the claims.” *Id.* at 580. As I

will demonstrate in the grounds below, the prior art discloses or renders obvious the claims.

VIII. OVERVIEW OF THE PRIOR ART REFERENCES

A. Johnston

38. U.S. Patent Application Publication No. 2010/0114739 (“Johnston”; Exhibit 1005) relates to techniques that allow “a seller to establish at a hosted service a store front that is distributable across a plurality of third party hosts.” Exhibit 1005, Abstract. In particular, Johnston discloses, and illustrates in Figure 1A (reproduced below), a system including a **platform server 110** that communicates, over a network 104', with devices that display a plurality of **Storefronts 102a–102c** that allows buyers to purchase content, products, and services from sellers. *See, e.g., id.*, ¶ 114. Johnston’s system includes a **storage 106** “to store and retrieve content by the **platform 110**.” *Id.*, ¶ 118.

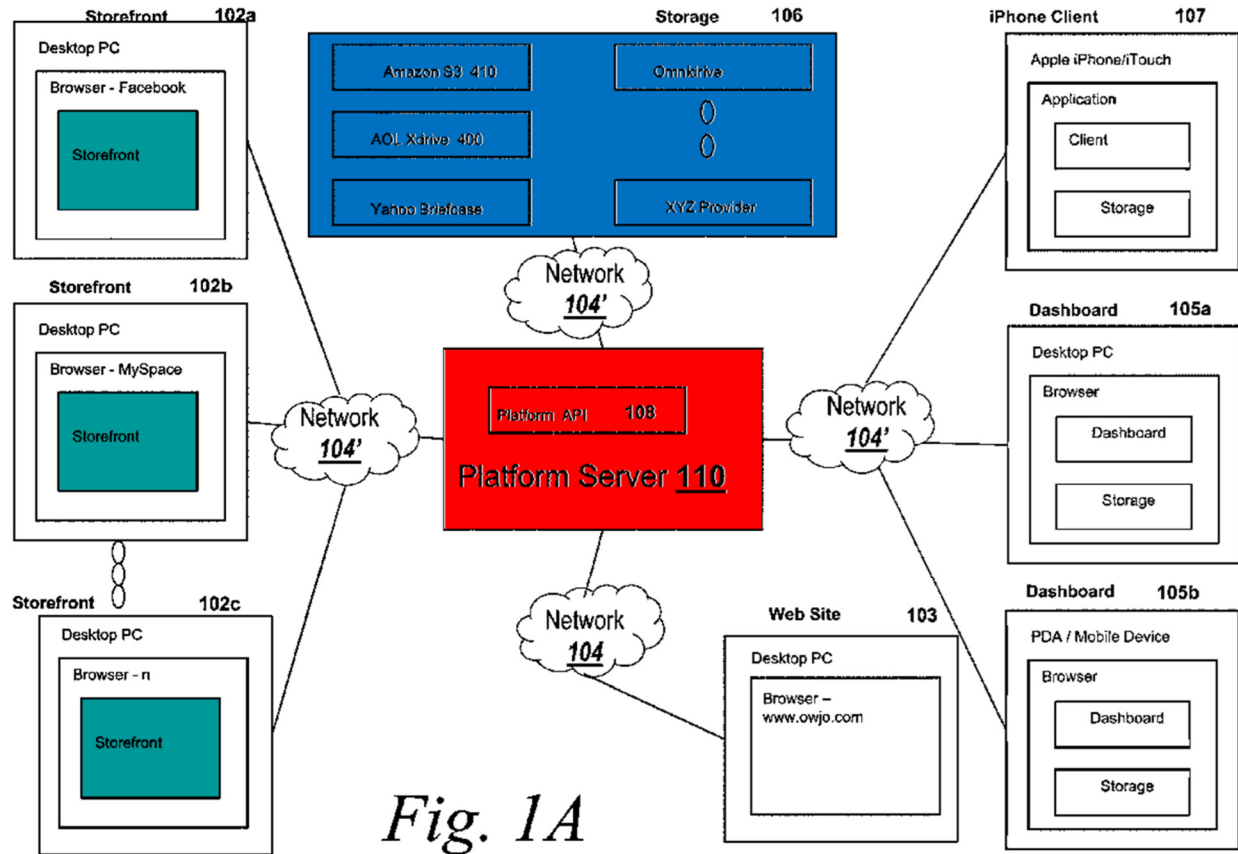


Fig. 1A

Exhibit 1005, Figure 1A (annotated)

39. Johnston’s **platform server 110** includes a billing module 158, which includes a **micro payment service 145**. *Id.*, ¶ 184, Figure 1C (reproduced below). **Micro payment service 145** allows “sellers and buyers to pay for products at low values or for small amounts of monies without incurring credit card or other third-party transaction fees.” *Id.*, ¶ 185.

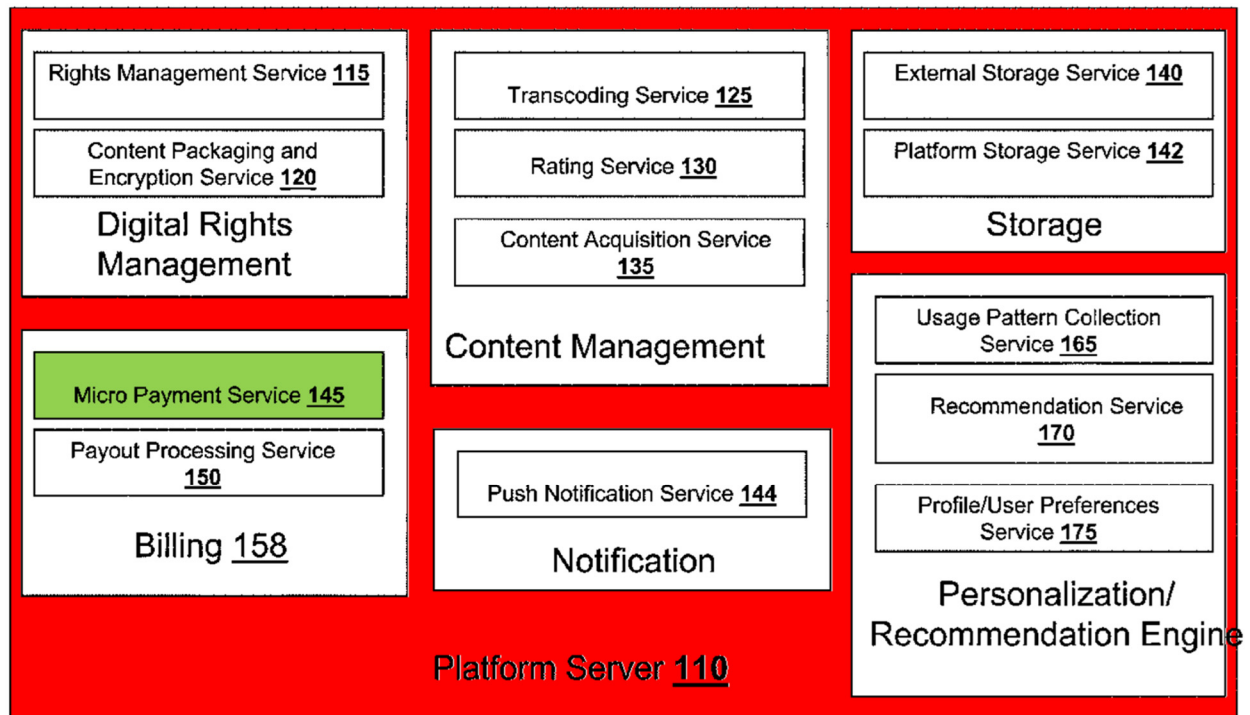


Exhibit 1005, Figure 1C (annotated)

40. Further, **platform server 110** provides user interfaces that allow sellers to create Storefronts and product listings to sell music, videos, images, podcasts, and vodcasts. *See id.*, Figures 3A–3F, ¶¶ 225–230. **Platform server 110** allows the sellers to generate an “Embed code” and publish their Storefronts to the sellers’ social networking sites and “any HTML web site.” *Id.*, ¶ 231, Figure 3G.

41. Referring to Figures 4A–4C, Johnston describes an exemplary **Storefront** running within a seller’s social networking profile. As can be seen in Figure 4B, the **Storefront** includes a **product listing**.

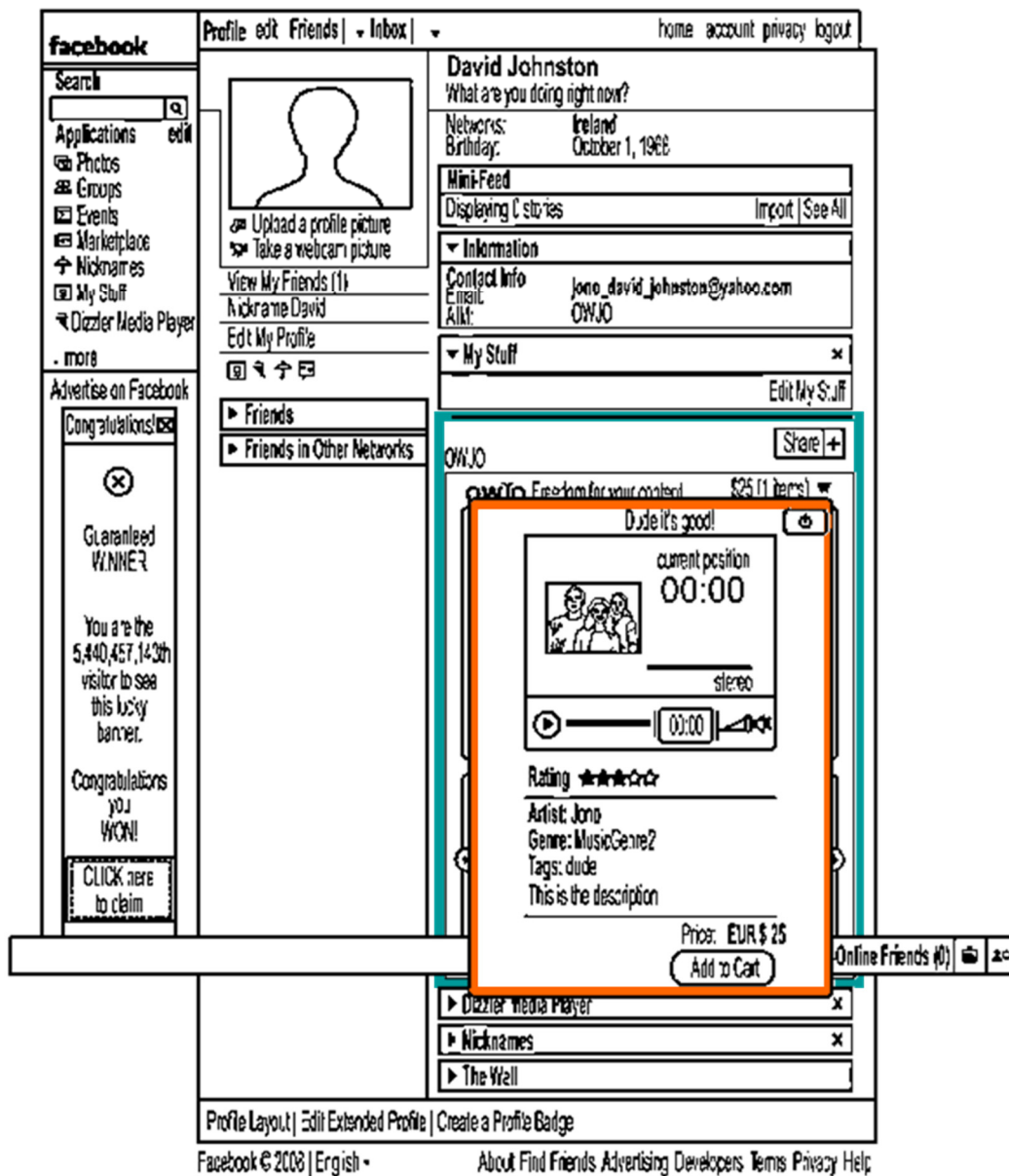


Exhibit 1005, Figure 4B (annotated)

42. Johnston’s **product listing** includes a **thumbnail image**, **information**, and a **preview** related to a music track, and an “**Add to Cart**” button. *See id.*, ¶¶ 253, 260, 261, 291.

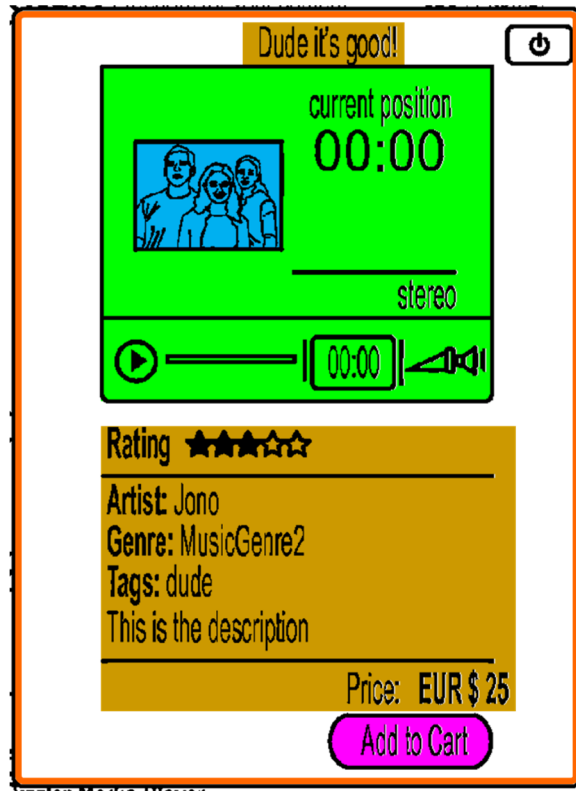


Exhibit 1005, Figure 4B (excerpted and annotated)

43. Johnston also discloses an exemplary “shopping cart” interface in the **Storefront** in Figure 4C (reproduced below) that includes a “**Check Out**” button and allows a buyer to initiate a checkout process to “purchase items.” See Exhibit 1005, ¶ 254.

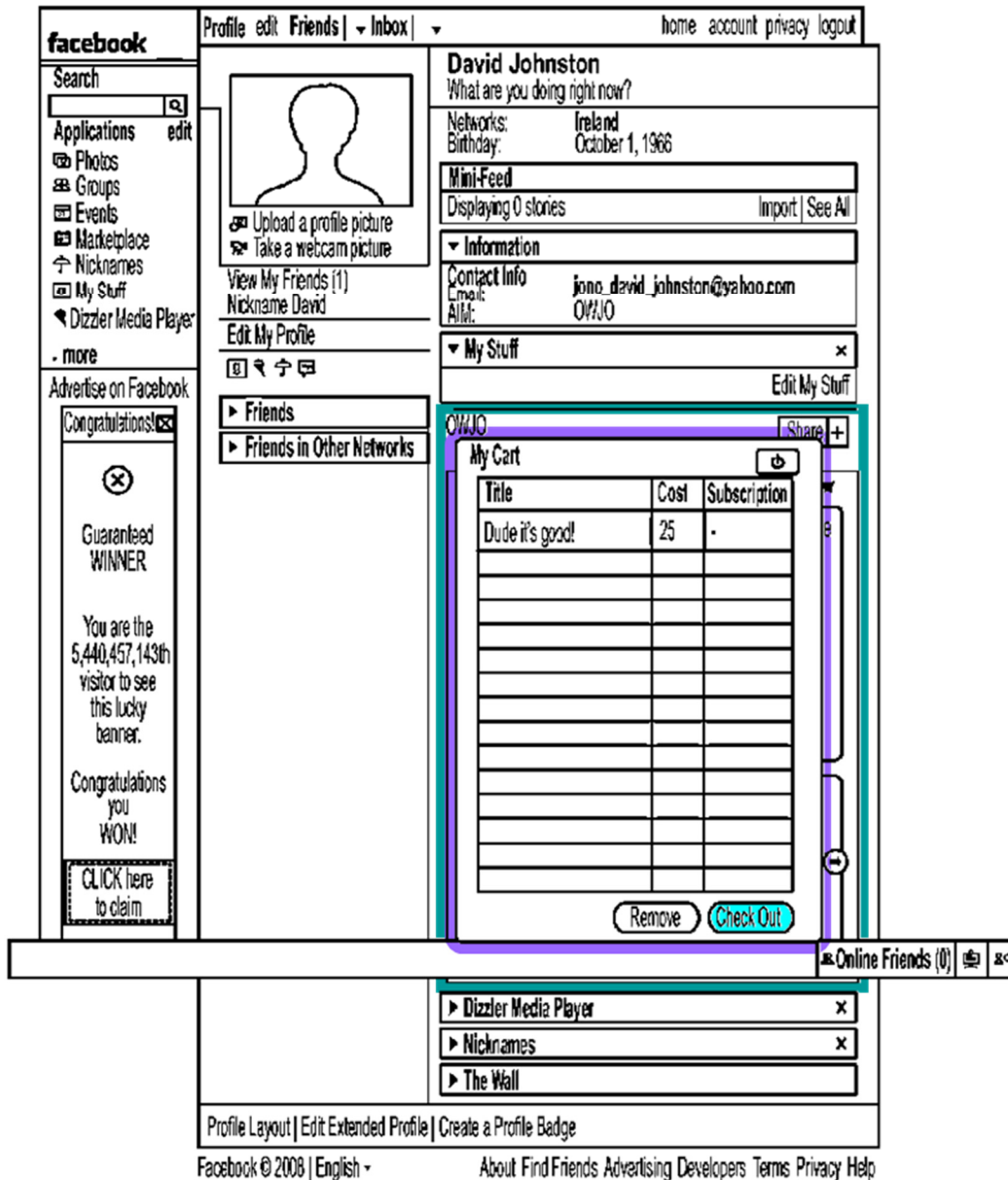


Exhibit 1005, Figure 4C (annotated)

1. Johnston is Entitled to its Provisional Filing Date

44. Johnston is entitled to the September 3, 2008 filing date of provisional application No. 61/093,917 (“Johnston-Provisional”) to which it claims priority.

45. I have been informed that, for purposes of §102(e), a published patent application is entitled to use the filing date of a beneficially claimed provisional application as an effective prior art date, provided that 1) the subject matter relied upon in the application is described in the provisional application; and 2) at least one of the claims in the application is supported by the written description of the provisional application.

46. As the following exemplary citations show, the subject matter relied upon in Johnston is described in Johnston-Provisional.

Johnston (Exhibit 1005)	Johnston-Provisional (Exhibit 1010)
FIGS. 1A–1C, 3A–3G, 4A–4E, 8B	FIGS. 1A–1C, 3A–3G, 4A–4E, 8B
¶ 21	pp. 4–5 ¹
¶¶ 41–42	p. 7
¶¶ 114–116	pp. 11–12
¶ 118	pp. 12–13
¶ 138	p. 15
¶ 161	p. 19
¶ 176	p. 19
¶¶ 184–186	p. 24
¶ 211	p. 31

¹ Page cites refer to the specification.

¶¶ 225–231	pp. 35–38
¶¶ 252–257	pp. 39–40
¶¶ 260–261	p. 41
¶ 286	p. 46
¶ 291	p. 39
¶ 309	p. 31

47. In addition, the exemplary citations below demonstrate that at least claim 1 of Johnston is supported by the written description of Johnston-Provisional.

Johnston (Exhibit 1005), claim 1	Johnston-Provisional (Exhibit 1010)
1. A method of self-service by a seller to establish at a hosted service a store front that is distributable across a plurality of third party hosts, the method comprising:	FIGS. 1A, 3H; pp. 11–12, 38–39 ²
a) receiving, by a hosted service operating on one or more servers, specification of one or more items to be displayed for sale in a store front for a seller;	FIGS. 1A, 3H; pp. 11–12, 38–39
b) generating, by the hosted service responsive to the specification, a portable widget for the store front identifying the one or more items for sale;	FIGS. 1A, 3H; pp. 11–12, 38
c) receiving, by the hosted service from the seller, identification of one or more third party hosts to which to publish the portable widget of the store front; and	FIGS. 1A, 3H; pp. 11–12, 39

² Page cites refer to the specification.

d) initiating, by the hosted service, publishing of the portable widget of the store front to the one or more third party hosts.	FIGS. 1A, 3H; pp. 11–12, 39
--	--------------------------------

48. Accordingly, for purposes of 35 U.S.C. §102(e), Johnston is entitled to an effective prior art date of September 3, 2008 filing date of Johnston-Provisional.

B. Wiser

49. Like the '995 patent, U.S. Patent No. 6,868,403 (“Wiser”; Exhibit 1003) “relates generally to the field of online commerce, and more particularly, to system and methods for the online distribution of digital media data over public communication networks.” Exhibit 1003, 1:8–11. Wiser discloses “[a] computer implemented online music distribution system [that] provides for the secure delivery of audio data and related media, including text and images, over a public communications network.” *Id.*, Abstract; *see also id.*, 5:46–48.

50. Wiser illustrates, in Figure 1 (reproduced below), an embodiment of its online music distribution system—system 100—that “includes a music distribution center 124 which operates with any number of client systems 126,” allowing artists to sell media content stored in a **master media file system 120**. *Id.*, 5:49–50.

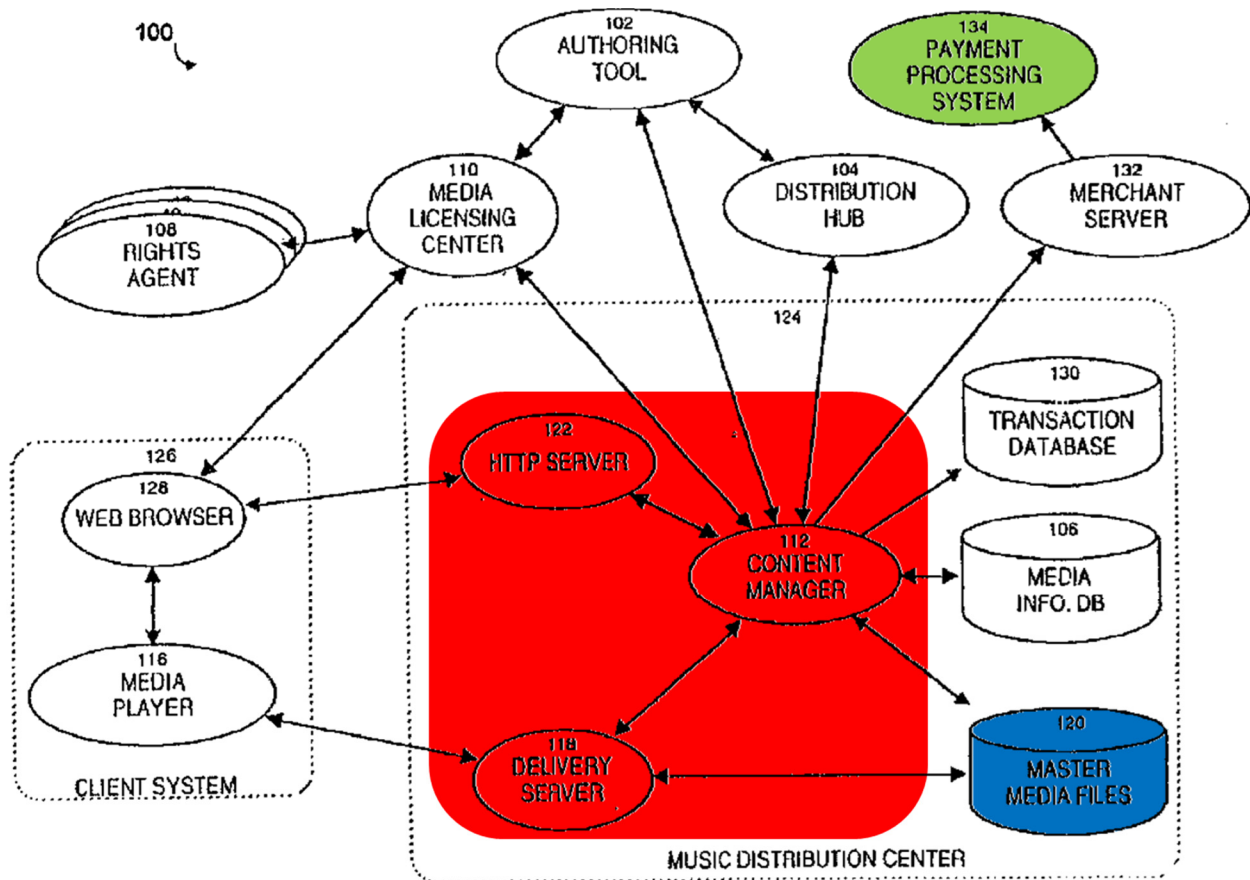


Exhibit 1003, Figure 1 (annotated)

51. “The music distribution center 124 includes a content manager 112, and at least one delivery server 118, an HTTP (HyperText Transfer Protocol) server 122,” collectively a “**server**.” Exhibit 1003, 5:51–54. The **content manager 112** is in communication with **payment processing system 134** and handles “the administration and management of all purchases and other transactions.” *Id.*, 5:56–61, 6:34–36.

52. Wiser further discloses a “**preview listing**” displayed on a **web page** in a web browser—for example, web browser 128 of client system 126—that allows a

user to purchase content listed for sale by an artist by “clicking on a ‘Buy it’ button.” See Exhibit 1003, 16:30–36, Figure 8 (reproduced below). Similar to the product listing that ’995 patent illustrates in Figure 10 (reproduced below alongside Wiser’s Figure 8), Wiser’s **preview listing** includes the “**Song Title**” and “**Artist**,” a **preview of the product** for sale, and a “**Place Order**” button.

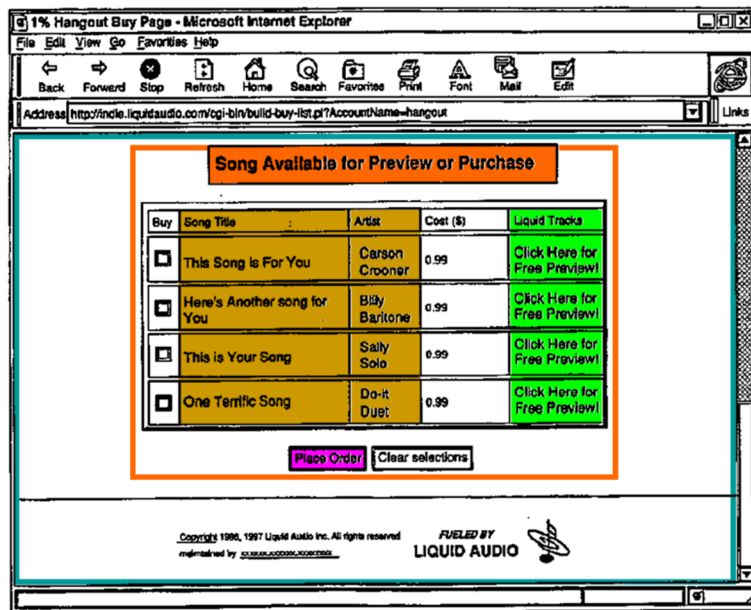


Exhibit 1003, Figure 8 (annotated)

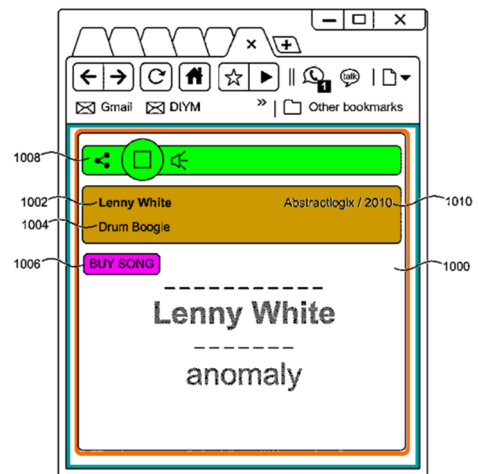


Exhibit 1001, Figure 10
(annotated)

C. Dierks

53. U.S. Patent Application Publication No. 2008/0183593 (“Dierks”; Exhibit 1004) discloses a “computer-implemented method [that] includes generating on a user terminal a merchant web site.” Exhibit 1004, Abstract. Similar to the ’995 patent, Johnston, and Wiser, Dierks illustrates in Figure 1B (reproduced below) a web browser displaying a **web page** with a **product listing** that includes a **preview**

of the product for sale and a “purchasing button,” which Dierks illustrates as an “Add to Cart” button 105. See Exhibit 1004, ¶¶ 36, 71, 73.

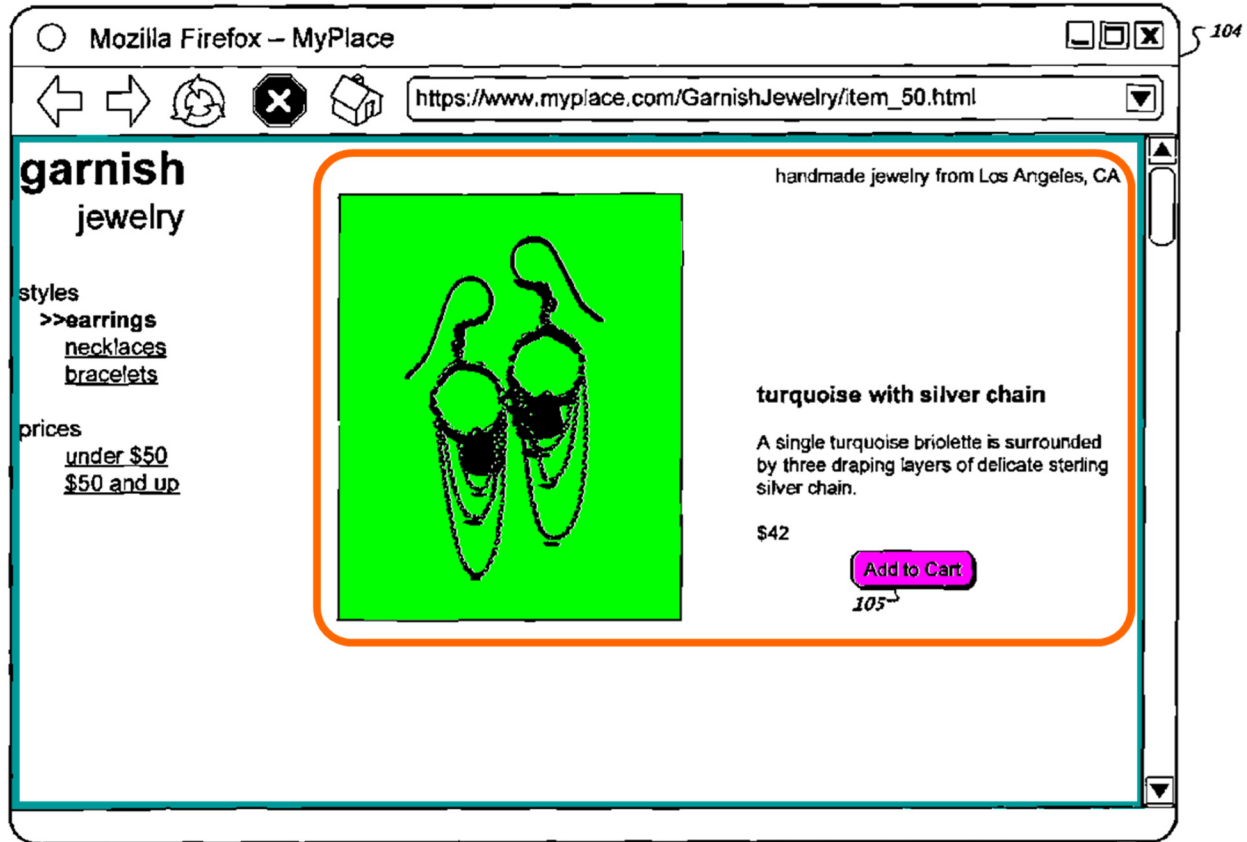


Exhibit 1004, Figure 1B (annotated)

54. Dierks explains that, when a user selects the “Add to Cart” button 105, “a shopping cart object shown as a graphical element such as an iframe 116” is displayed in the web page as shown in Figure 1C (reproduced below). *Id.* at ¶ 43. The shopping cart includes a “Proceed to Checkout” button 118 and allows a purchaser to initiate a checkout process and make a payment. *Id.* at ¶ 44.

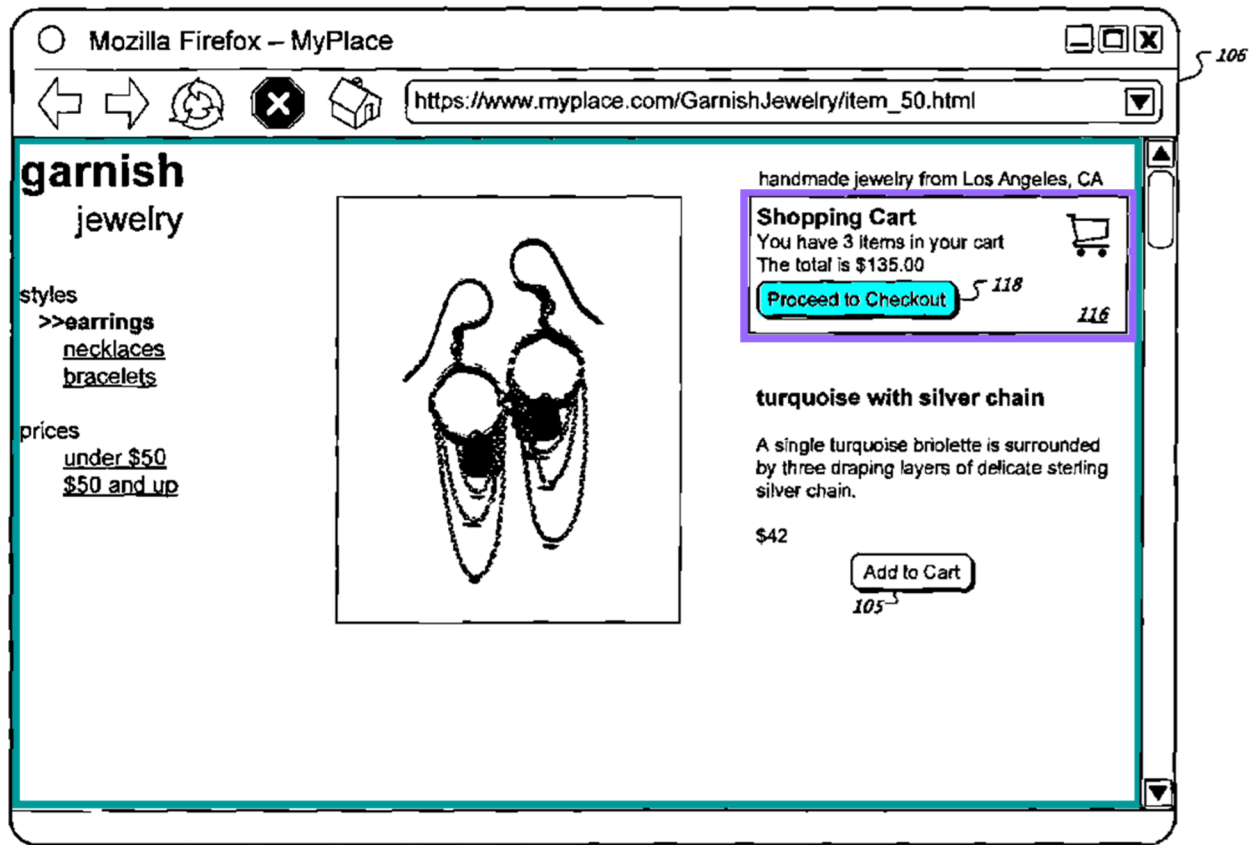


Exhibit 1004, Figure 1C (annotated)

55. Dierks discloses and illustrates in Figure 3 (reproduced below) an e-commerce system 300 that includes an **HTML generator 312** that “is configured to receive parameters regarding items to be sold by vendors and to generate code” that can be embedded in a web page to display a **product listing** as shown above. Exhibit 1004, ¶¶ 60, 62, Figure 3; *see also id.*, ¶¶ 71, 73, 74, Figure 4.

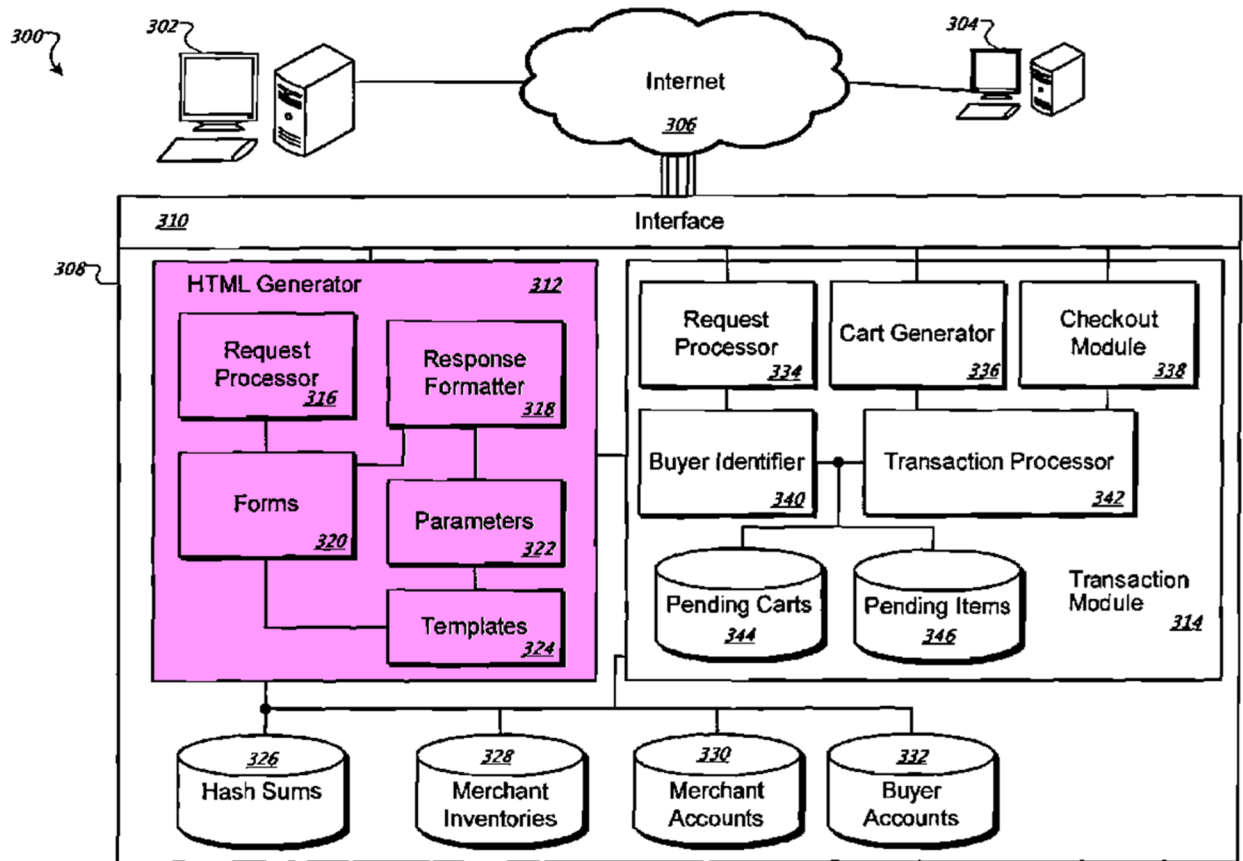


Exhibit 1004, Figure 3 (annotated)

D. Chang

56. U.S. Patent Application Publication No. 2008/0255962 (“Chang”; Exhibit 1008) relates to “the use of widgets to implement a distributed commerce and viral marketing regime.” Exhibit 1008, ¶ 2. In particular, Chang discloses “generating an item page that includes widget code and details related to an item for sale.” *Id.*, Abstract; *see also id.*, ¶ 37 (“a web page (e.g., an item page)”). Chang discloses widget and commerce servers which operate to generate item pages integrating widget code, which are then accessed by users via various devices over

the Internet, as Chang illustrates in Figure 1 (reproduced below). *See id.*, ¶¶ 36–37, 41.

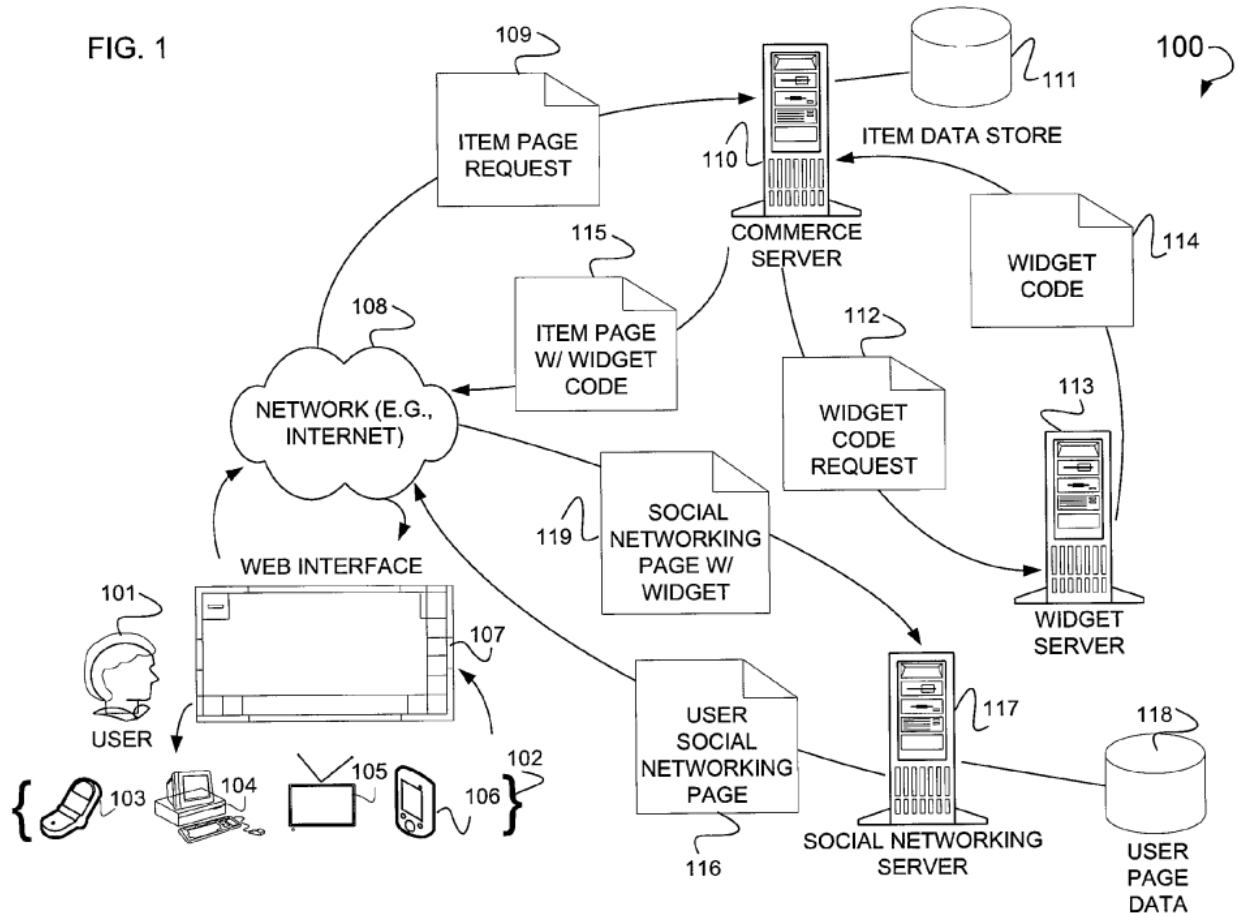


Exhibit 1008, Figure 1

57. Chang describes distributing widget code to “a plurality of social networking servers reflecting a plurality of social networking sites.” Exhibit 1008, ¶ 41, Figure 2. In one embodiment, Chang discloses including a share button in the widget code. *Id.*, ¶ 43. Chang explains that “[t]his share button, when executed using an input device, may generate a prompt asking the user 201 where they would

like to embed the widget code (e.g., what web page associated with what web site and servers associated with the web site).” *Id.*

IX. SPECIFIC GROUNDS FOR UNPATENTABILITY

58. In my opinion, for the reasons noted below, claims 1–14 of the ’995 patent would have been obvious in light of the prior art.

A. Ground 1: Claims 1–14 are Obvious over Johnston and Dierks

1. Independent Claim 1

a) 1[preamble]: “A system for generating an electronic commerce store, comprising:”

59. In my opinion, even if the preamble is limiting, Johnston discloses it.

60. *First*, for example, Johnston discloses, and illustrates in Figure 1A, “an environment of a platform providing **systems and methods for monetizing content, products and services, and for handling buyer-side and sell-side.**” Exhibit 1005, ¶ 114. Johnston’s system includes a platform server 110 that communicates, over networks 104 and 104’, with devices that display a plurality of **Storefronts 102a–102c** that allows buyers to electronically purchase content, products, and services from sellers. *See, e.g., id.*, ¶ 114, Figure 1A. Thus, each **Storefront** is “*an electronic commerce store.*”³

³ Terms that appear in the challenged claims are italicized throughout.

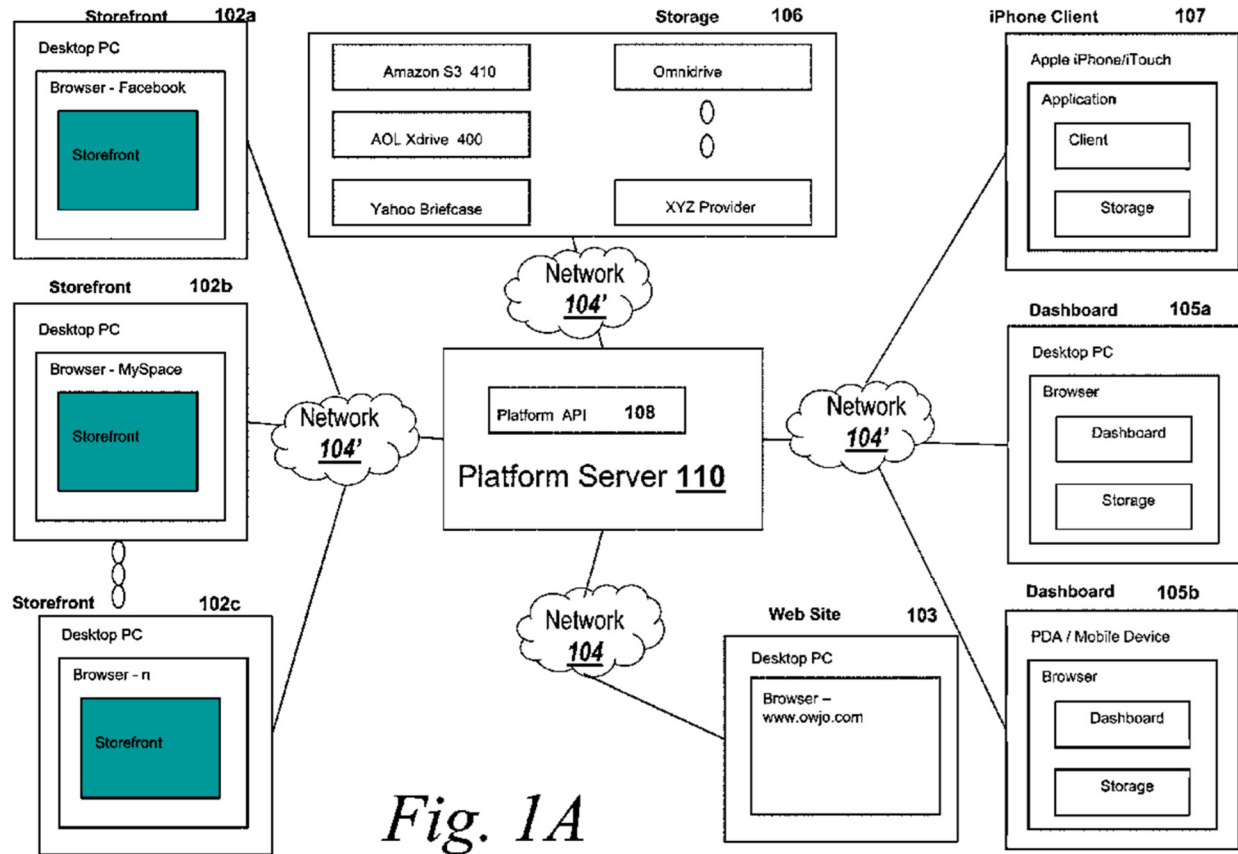


Exhibit 1005, Figure 1A (annotated)

61. **Second**, Johnston discloses that its system generates the **Storefronts**. Johnston explains that “the seller or artist establishes, via the OWJO web site 103, a Storefront and makes available content for purchase.” *Id.*, ¶ 211. The web site 103 “is the web presentation layer to the platform Server 110” and provides access to “the functionality and services of the platform 110 via any type and form of user interface.” *Id.*, ¶ 138; *see also id.*, ¶ 225 (“In FIG. 3A, the seller is depicted creating an OWJO Storefront via a user interface of the OWJO platform.”), Figure 3A. For example, in Figure 3B, the user interface allows the seller to enter information

associated with music to be listed on the seller's Storefront. *See id.*, ¶ 226. The seller can do the same for videos, images, podcasts, and vodcasts. *See id.*, ¶¶ 227–230, Figures 3C–3F.

The screenshot displays the OWJO 'Upload Content' interface. At the top, a navigation bar includes 'Welcome CPA (Log out...)', 'Home', 'About us', and 'Site map'. Below this is a secondary bar with 'Upload', 'Sales', 'Purchases', 'Top Up', 'Summary', 'Publish', 'Settings', and 'Admin'. The main content area is titled 'Upload Content' and features tabs for 'music', 'videos', 'images', 'podcasts', and 'vodcasts', with 'music' currently selected. The 'Music file Settings' section contains instructions for uploading a music track. It includes input fields for 'Name of track:', 'Artist:', and 'Tags:', and a dropdown menu for 'Genre:' set to 'Pop'. Below these is a 'Preview track:' section with a text box for a sample of the music track. Further down are sections for 'Music file:' and 'Thumbnail:', each with a 'Browse...' button. At the bottom, there is a 'Set your price' section with a currency dropdown set to 'EUR' and a price input field showing '0'. An 'upload' button is located at the bottom right. On the left side of the interface, there is a 'My Stuff' section showing a list of items with star ratings and a 'Featured' section showing a list of items with star ratings and a '1/2' indicator.

Exhibit 1005, Figure 3B

62. Thus, Johnston provides a system for generating a **Storefront** (i.e., “*A system for generating an electronic commerce store*”). Therefore, Johnston discloses the preamble of claim 1.

b) 1[a]: “an application server, the application server controlling a transaction processing service;”

63. Johnston discloses element 1[a].

64. ***First***, Johnston’s system includes **platform server 110** that “may comprise a single server, a server farm or a cluster of one or more **application servers** or nodes.” Exhibit 1005, ¶ 115. **Platform server 110** corresponds to “*an application server*.”

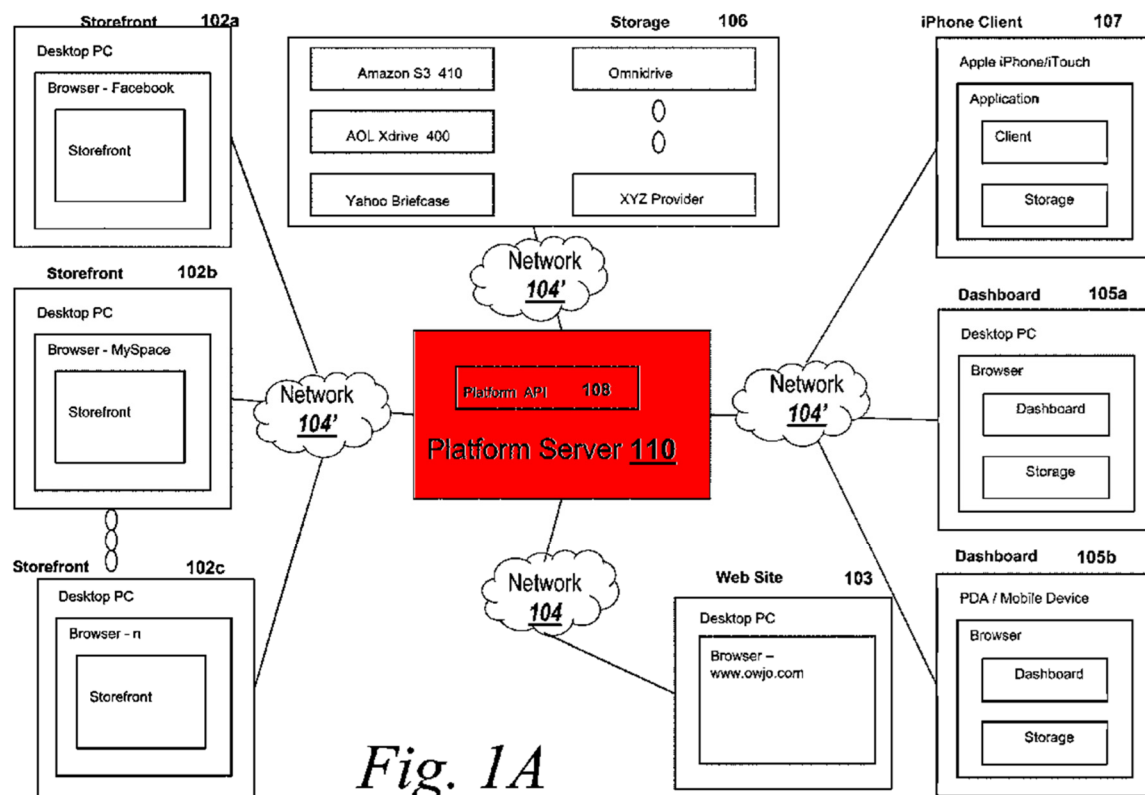


Exhibit 1005, Figure 1A (annotated)

65. *Second*, as Johnston illustrates in Figure 1C, **platform server 110** includes a billing module 158, which includes a **micro payment service 145**. Exhibit 1005, ¶ 184. **Micro payment service 145** is “a transaction processing service.” Johnston explains that **micro payment service 145** allows “sellers and buyers to pay for products at low values or for small amounts of monies without incurring credit card or other third-party transaction fees.” *Id.*, ¶ 185.

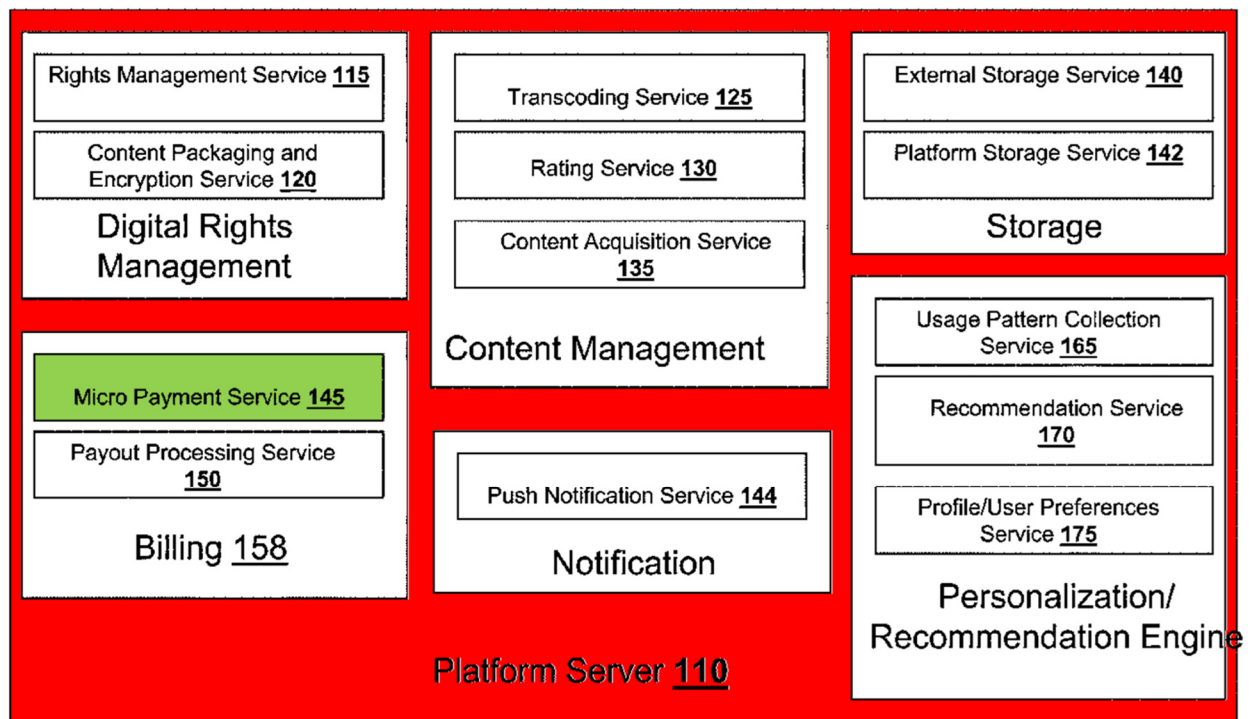


Exhibit 1005, Figure 1C (annotated)

66. *Third*, Johnston discloses that the third-party transaction fees can be avoided “by processing a transaction via the OWJO platform and **controlling** the transaction independent of a third party payment processor.” *Id.*, ¶ 185. In other words, **platform server 110** is “controlling” **micro payment service 145**.

67. Therefore, Johnston discloses element 1[a].

c) 1[b]: “a storage resource communicatively coupled to the application server, the storage resource storing an image and metadata related to a product;”

68. Johnston discloses element 1[b].

69. *First*, Johnston’s system includes a **storage 106**—“a set of one or more storage services or locations **to store and retrieve content by the platform 110.**” Exhibit 1005, ¶ 118. **Storage 106** is “*communicatively coupled to*” **platform server 110** (i.e., “*the application server*”) via network 104’. *See id.*, Figure 1A; *cf.* Exhibit 1001, Figure 1A (illustrating storage resource 124 coupled to application server 106 via a network including file management server 114).

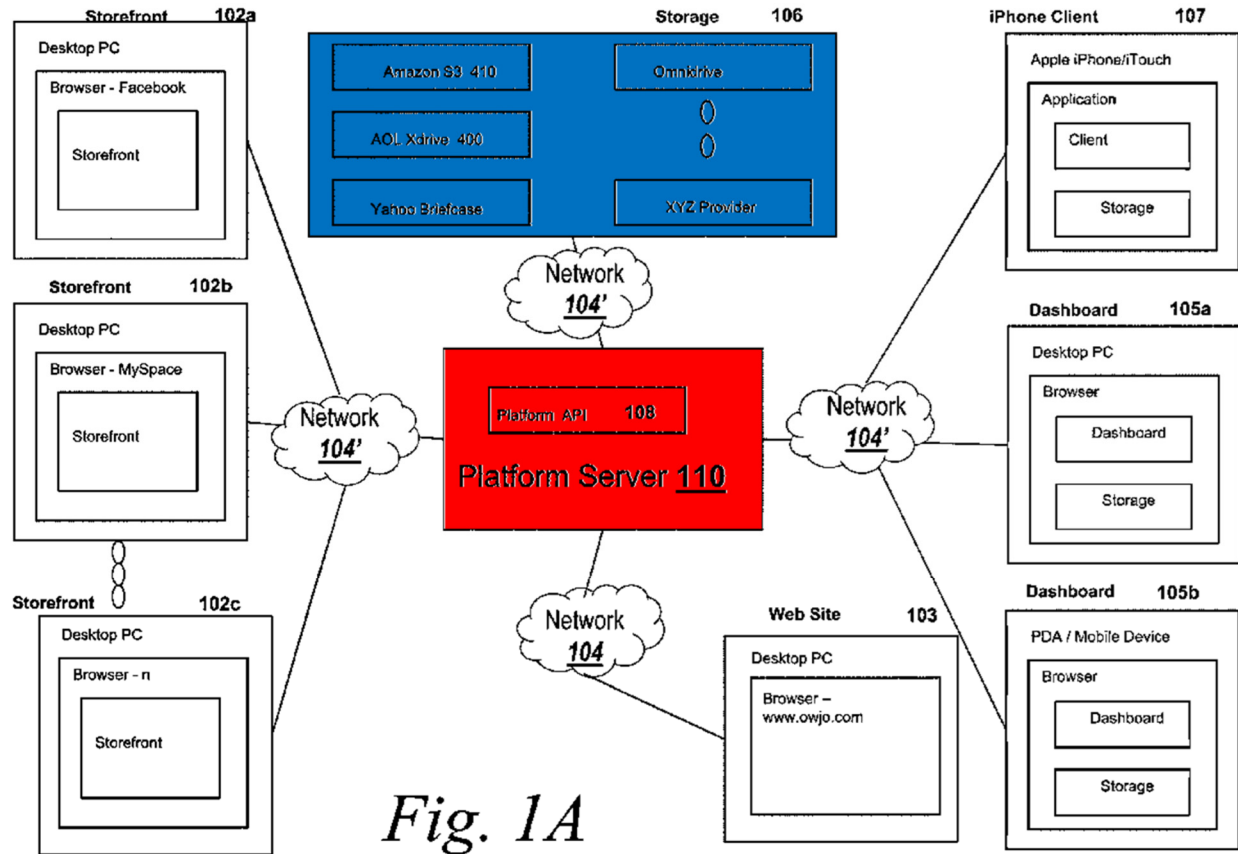


Exhibit 1005, Figure 1A (annotated)

70. *Second*, **storage 106** is “a storage resource.” According to the ’995 patent, “cloud-based storage resource [is] provided by the Amazon Web Services group at Amazon.com, Inc.” Exhibit 1001, 5:6–8. Similarly, Johnston explains that “storage 106 includes any type and form of storage drives, mediums and units” and that, in some embodiments, **storage 106** uses storage services like “Amazon S3, AOL Xdrive, Microsoft SkyDrive, Google Documents, Yahoo Briefcase and Omnidrive.” Exhibit 1005, ¶ 118.

71. **Third**, Johnston’s **storage 106** stores content for sellers. *See* Exhibit 1005, ¶ 118 (“In one embodiment, the Storage is a highly scalable, reliable, secure and fast storage that is used by one or more of the OWJO Server, OWJO Dashboard and OWJO Storefront to store and retrieve content for buyers and sellers.”); *see also id.*, ¶ 41 (“an artist can upload content ... that the artist wishes to sell”), ¶ 114 (“Content may include any type or form of digital content or content recordable on media, such as music, digital books, application tools or software, and video clips.”). Johnston further discloses that “a digital right, including information associated with the digital right, can be encoded along with content **metadata** into ... an MP3 file.” *Id.*, ¶ 309. Thus, Johnston discloses storing a music file along with its “*metadata*” in **storage 106**.

72. **Fourth**, Johnston discloses that a seller can provide “a thumbnail **image** that can be displayed in the OWJO Storefront” and that the “thumbnail can... be uploaded from the machine that is executing the web browser they are using or loaded from a Uniform Resource Identifier anywhere on the internet.” Exhibit 1005, ¶ 226, Figure 3B; *see also id.*, ¶¶ 227–230, Figures 3C–3F. Because **storage 106** can be an online storage service, a POSITA would have understood that the seller could have stored the thumbnail image in **storage 106** before loading it for display in the Storefront. *See, e.g., id.*, ¶ 118 (“In one embodiment, the **Storage** is a highly scalable, reliable, secure and fast storage that is used by one or more of the OWJO

Server, OWJO Dashboard and OWJO Storefront **to store** and retrieve **content for buyers and sellers**. ... In some embodiments, OWJO Storage interfaces to, communicates with or otherwise uses any of the following storage services or solutions: Amazon S3, AOL Xdrive, Microsoft SkyDrive, Google Documents, Yahoo Briefcase and Omnidrive.”).

73. Moreover, Johnston makes clear that “[p]roducts may be physical products or be in some intangible form.” *Id.*, ¶ 5. Thus, digital content such as a music file stored on **storage 106** for sale is “*a product*.” This is consistent with how the ’995 patent describes “product”: “each uploaded content file represents an item for promotion and distribution” and “[e]ach ‘item’ is made available as a ‘product’ for marketing purposes.” Exhibit 1001, 11:25–27.

74. To the extent Patent Owner argues that “*a product*” is a physical item, Johnston discloses that “physical items include products such as CDs and vinyl records” and that, in some embodiments, these products may be associated with digital content. Exhibit 1005, ¶ 291. A POSITA would have understood that **storage 106** may store digital content associated with a physical item—for example, an MP3 version or portion of a song that is on a CD or vinyl record. Thus, in such embodiments, **storage 106** stores an image and metadata related to a physical item.

75. Accordingly, Johnston’s **storage 106** is a “storage resource storing an image and metadata related to a product.”

76. Therefore, Johnston discloses element 1[b].

d) 1[c]: “a widget builder resource executing on the application server,”

77. Johnston discloses element 1[c].

78. **First**, Johnston discloses that “[t]he **Storefront** is a **widget** or component, such as a Macromedia Flash component, that can be embedded on any website or on the artist’s own social networking profile.” Exhibit 1005, ¶ 21.

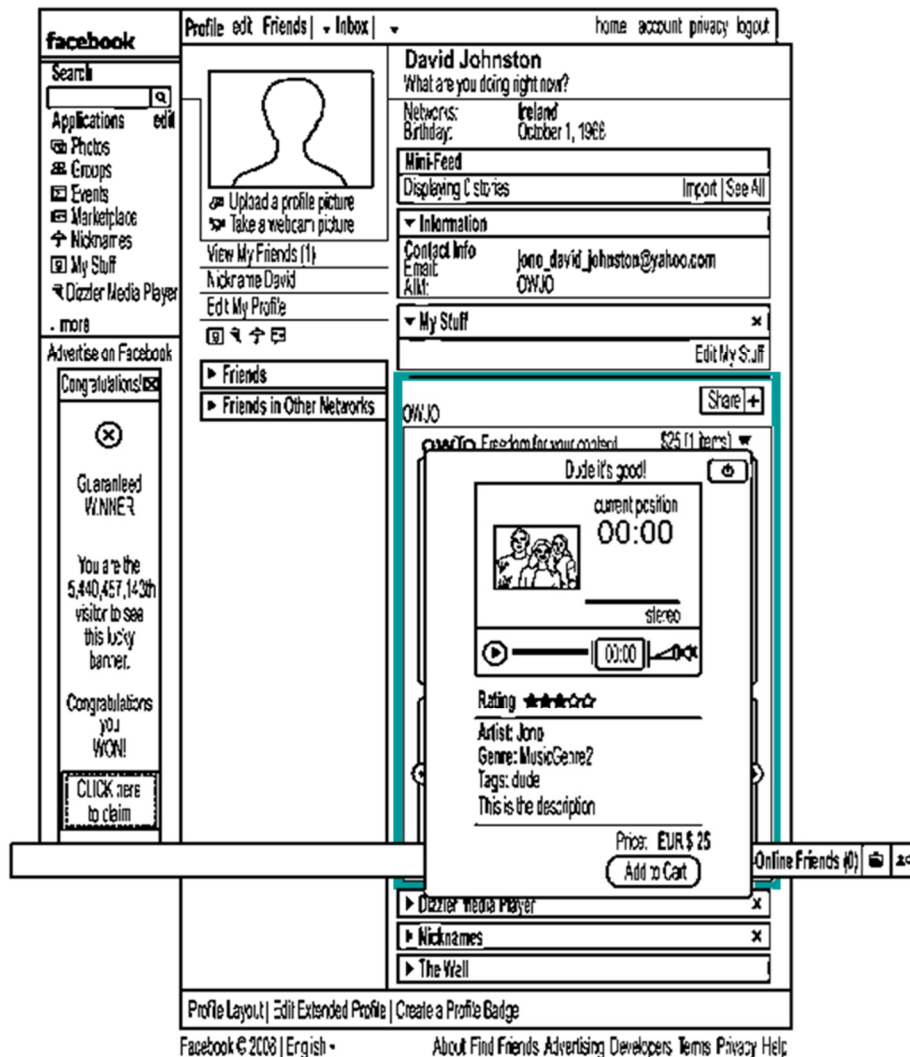


Exhibit 1005, Figure 4B (annotated)

79. **Second**, referring to Figures 3A–3G, Johnston discloses a seller “creating an OWJO Storefront via a user interface of the OWJO platform.” Exhibit 1005, ¶ 225. For example, in Figure 3B, the user interface allows the seller to enter information associated with music to be listed in a product listing on the seller’s Storefront and upload files (e.g., “a preview of the music track” and “the actual music track that is for sale”). *Id.*, ¶ 226, Figure 3B. The seller can do similarly for videos, images, podcasts, and vodcasts. *See id.*, ¶¶ 227–230, Figures 3C–3F.

The screenshot shows a web browser window with the OWJO logo and navigation links. The main content area is titled 'Upload Content' and has tabs for music, videos, images, podcasts, and vodcasts. The 'music' tab is selected, showing 'Music file Settings'. The settings include fields for Name of track, Artist, Tags, and Genre (set to Pop). There are sections for 'Preview track' and 'Music file' with 'Browse...' buttons. A 'Thumbnail' section also has a 'Browse...' button. At the bottom, there is a 'Set your price' section with a currency dropdown (set to EUR) and a price input field. An 'upload' button is at the bottom right.

Welcome CPA (Log out...)

Home About us Site map

OWJO
Freedom for your content

Upload Sales Purchases Top Up Summary Publish Settings Admin

Upload Content

music videos images podcasts vodcasts

Music file Settings

Use this form to upload a Music track to OWJO. This music track will be available on your OWJO Widget.

Name of track:

Artist:

Tags:

Genre:

Preview track:

Please upload a sample of the music track. This will be available to everyone using the OWJO Widget. The file must be in one of the following formats (MP3...) and must not exceed 10MB in size.

Browse...

Music file:

Please upload music file. This will be only be made available when it is purchased. The file must be in one of the following formats (MP3...) and must not exceed 10MB in size.

Thumbnail:

Please provide a thumbnail file that we will display on the OWJO Widget.

Browse...

Set your price

upload

Exhibit 1005, Figure 3B

80. In Figure 3G, Johnston further discloses that the user interface allows the seller to publish the seller's **Storefront** widget to the seller's social networking sites and generates an "Embed code" that allows the seller to embed the **Storefront** widget in "any HTML web site." *Id.*, ¶ 231.

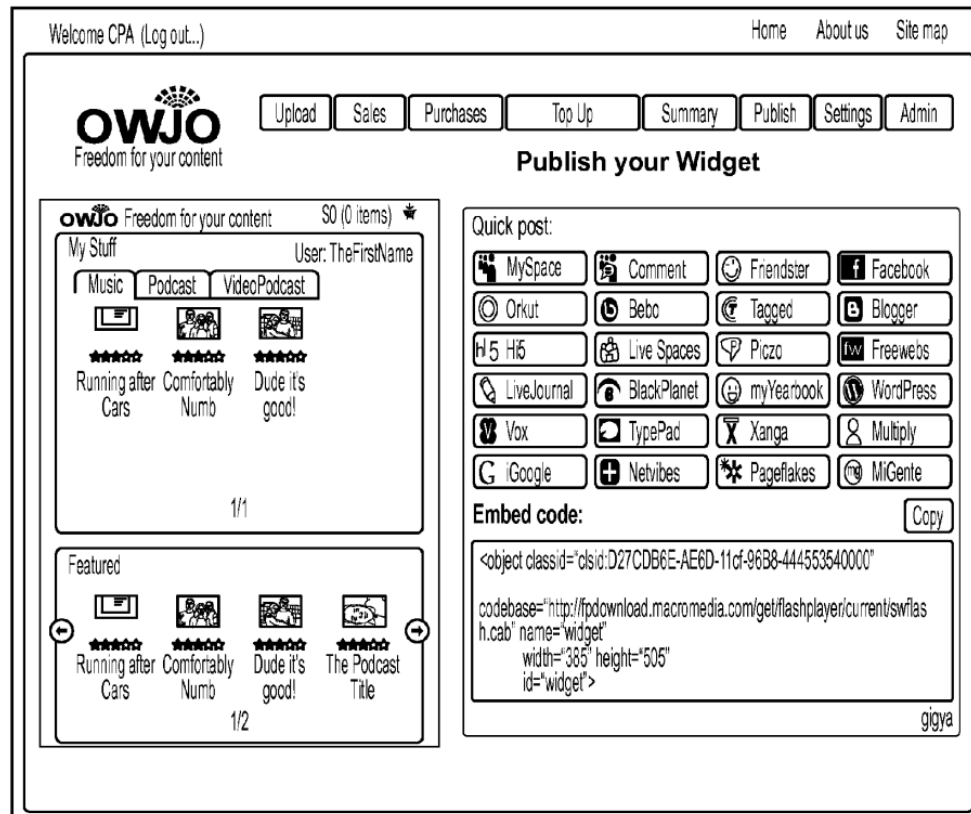


Exhibit 1005, Figure 3G

81. Thus, Johnston's Figures 3A–3G user interface constitutes "*a widget builder resource.*"

82. **Third**, the Figures 3A–3G user interface is executed on the platform server 110 (i.e., "*the application server*"). Johnston explains that "the seller or artist establishes, via the OWJO web site 103, a Storefront and makes available content

for purchase.” Exhibit 1005, ¶ 211. Johnston further describes that the web site 103 “is the web presentation layer to the platform Server 110” and that it provides access to “the functionality and services of the platform 110 via any type and form of user interface.” *Id.*, ¶ 138. The functionality and services of platform server 110 include providing the Figures 3A–3G user interface.

83. Therefore, Johnston discloses element 1[c].

*e) 1[d]: “the widget builder resource configured to:
receive the image and metadata from the storage resource;
and”*

84. Johnston discloses element 1[d]. As I discussed above for element 1[b], **storage 106** (i.e., “*the storage resource*”) stores a thumbnail image (i.e., “*the image*”) and content metadata (i.e., “*metadata*”) related to a music file for sale by a seller. Like the thumbnail image, Johnston discloses that the “actual music track ... can either be uploaded from the machine that is executing the web browser they are using or loaded from a Uniform Resource Identifier anywhere on the internet.” Exhibit 1005, ¶ 226, Figure 3B. Thus, also like the thumbnail image, a POSITA would have understood that a seller could have stored the actual music track to be sold in **storage 106**. *See, e.g., id.*, ¶ 118 (“In one embodiment, the **Storage** is a highly scalable, reliable, secure and fast storage that is used by one or more of the OWJO Server, OWJO Dashboard and OWJO Storefront **to store** and retrieve **content for buyers and sellers**. ... In some embodiments, OWJO Storage interfaces

to, communicates with or otherwise uses any of the following storage services or solutions: Amazon S3, AOL Xdrive, Microsoft SkyDrive, Google Documents, Yahoo Briefcase and Omnidrive.”). As also discussed above for element 1[b], a music track for sale is stored along with its “*metadata*” in **storage 106**. Thus, when the seller loads a thumbnail image and a music file via the Figure 3B user interface, “*the widget builder resource is configured to receive the image and metadata from the storage resource.*”

The screenshot displays the OWJO website's 'Upload Content' page. The header includes a welcome message 'Welcome CPA (Log out...)' and navigation links for 'Home', 'About us', and 'Site map'. Below the header, there are buttons for 'Upload', 'Sales', 'Purchases', 'Top Up', 'Summary', 'Publish', 'Settings', and 'Admin'. The main content area is divided into two sections. On the left, under 'My Stuff', there are tabs for 'Music', 'Podcast', and 'VideoPodcast'. The 'Music' tab is active, showing a list of uploaded tracks: 'Running after Cars', 'Comfortably Numb', and 'Dude it's good!'. Each track has a star rating and a thumbnail. Below this is a 'Featured' section with a similar list of tracks. On the right, the 'Upload Content' section has tabs for 'music', 'videos', 'images', 'podcasts', and 'votcasts'. The 'music' tab is selected, leading to the 'Music file Settings' form. This form includes fields for 'Name of track', 'Artist', 'Tags', and 'Genre'. Below these is a 'Preview track' section with instructions to upload a sample of the music track. Further down is a 'Music file' section with instructions to upload the music file. At the bottom, there is a 'Thumbnail' section with instructions to provide a thumbnail file. A 'Set your price' section is also present, with a dropdown menu for currency (set to 'EUR') and a text input for the price (set to '0'). An 'upload' button is located at the bottom right of the form.

Exhibit 1005, Figure 3B

85. Therefore, Johnston discloses element 1[d].

f) 1[e]: “[the widget builder resource configured to:] generate a product listing, wherein the product listing includes the image, the metadata, and a clickable link, wherein the clickable link is a buy button and includes a preview of the product, and”

86. Johnston discloses element 1[e].

87. As I explained above for element 1[c], Johnston’s Figures 3A–3G user interface (i.e., “*the widget builder resource*”) allows a seller to generate a **Storefront**. Johnston describes that a “**storefront widget** may comprise an arrangement of elements for a user interface for the display, selling and buying of any content, products and/or services.” Exhibit 1005, ¶ 42; *see also id.*, ¶ 44, claim 1 (“a portable widget for the store front identifying the one or more items for sale”). In other words, the **Storefront** includes a **product listing**, as shown in Johnston’s Figure 4B, for example. Thus, the widget builder resource generates a product listing when generating the Storefront.

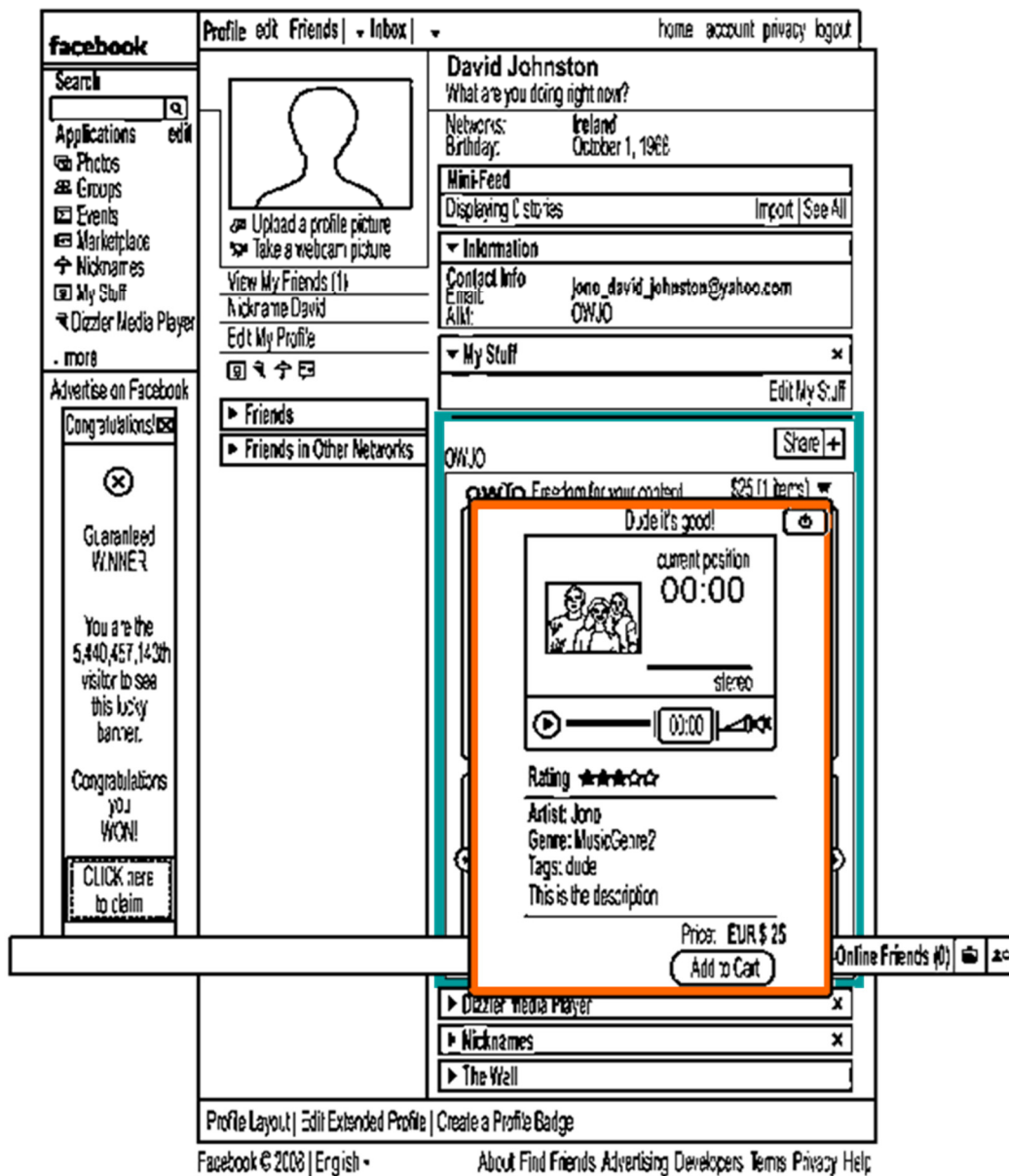


Exhibit 1005, Figure 4B (annotated)

88. The **product listing** includes a **thumbnail image** (i.e., “the image”), **information**, and a **preview** related to a music track, and an “**Add to Cart**” button.

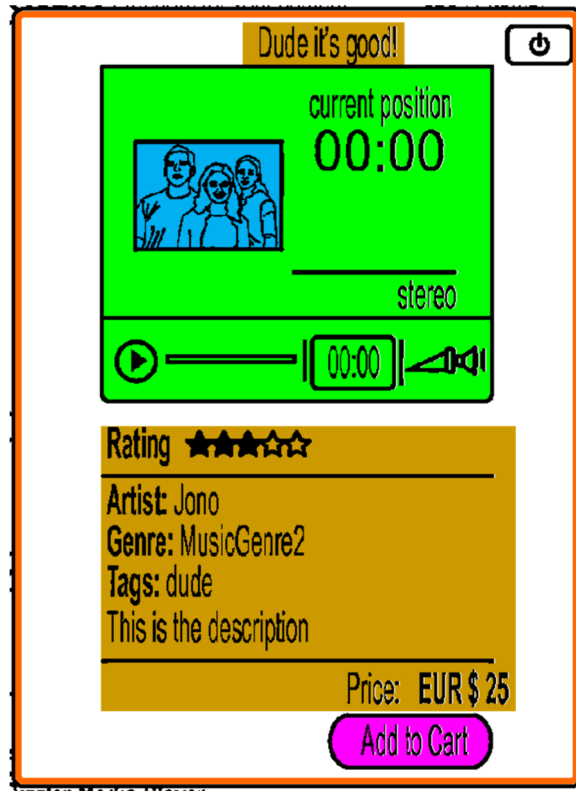


Exhibit 1005, Figure 4B (excerpted and annotated)

89. Johnston discloses that, “[i]n FIG. 4B, the buyer has selected a music track to **preview**.” Exhibit 1005, ¶ 253; *see also id.*, ¶ 260 (“The **previews** may include any type or form of trailer, teaser, portion of the content, commentary, endorsements, and/or reviews related to the content and/or artist.”), ¶ 291 (“The customer can preview both physical items (i.e., products) and digital items (e.g., digital content), as well as services (e.g., subscription services).”), Figure 9A. Thus, the **preview** is “a preview of the product.”

90. Johnston also discloses that “the buyer can decide to **purchase** content by **clicking** on the ‘**Add to Cart**’ button on FIG. 4B,” in which case, the item

displayed in the product listing in the Storefront is added to the user's shopping cart. Exhibit 1005, ¶ 261. A POSITA would have understood that the “Add to Cart” button is “a buy button” and “a clickable link.” See, e.g., Exhibit 1022, 5:26–28 (“The web page of the reseller includes a hyperlink, which is indicated by a graphic display, such a button, highlighted text, or the like.”).

91. Furthermore, although Johnston describes an embodiment where a seller may enter or specify the “Name of track,” “Artist,” “Tags,” and “Genre” for a music track in the Figure 3B interface, Johnston also discloses storing content metadata (i.e., “the metadata”) along with a music file to be sold. Exhibit 1005, ¶ 309. Accordingly, a POSITA would have understood that one or more of the pieces of **information** could also be part of the content metadata such that “the product listing includes... the metadata.” See, e.g., Exhibit 1003, 6:61–63 (“Media descriptive data 204... include descriptive text, such as title. ...”).

92. Therefore, Johnston discloses element 1[e].

g) 1[f]: “wherein the application server is configured to embed the product listing into a web site and into a social media network, and”

93. Johnston discloses element 1[f]. As I discussed for element 1[c], Johnston's user interface for **platform server 110** allows a seller to publish (i.e., “embed”) the seller's **Storefront** widget including a **product listing** into the seller's

social networking sites and “any HTML web site.” See Exhibit 1005, ¶ 231, Figure 3G (“Embed code”).

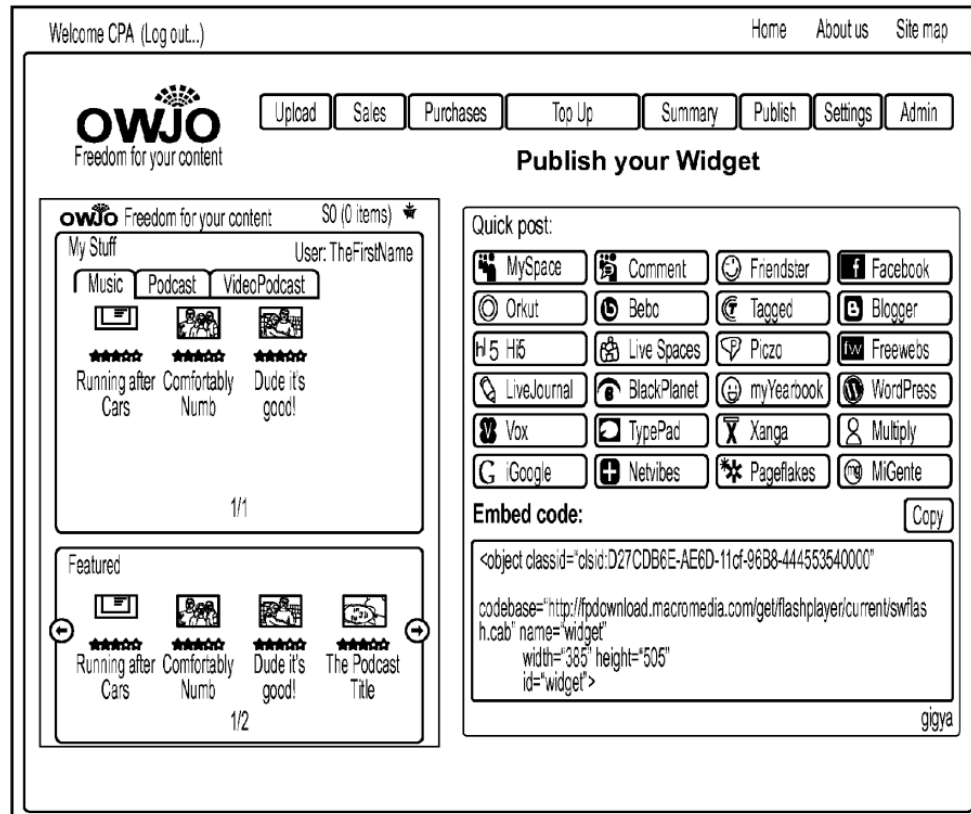


Exhibit 1005, Figure 3G

94. Referring to Figure 1A, Johnston discloses **Storefronts 102a–102c** running in web browsers of desktop computers, respectively within a “Facebook profile,” a “MySpace profile,” and “any website or social networking profile of a contributing artist.” *Id.*, ¶ 116; *see also id.* (“Amazon, Ebay, Google Product Search, Bizrate, Dealnews, Lulu, Twitter, LinkedIn and web-blogs are further examples of websites that can support a Storefront.”).

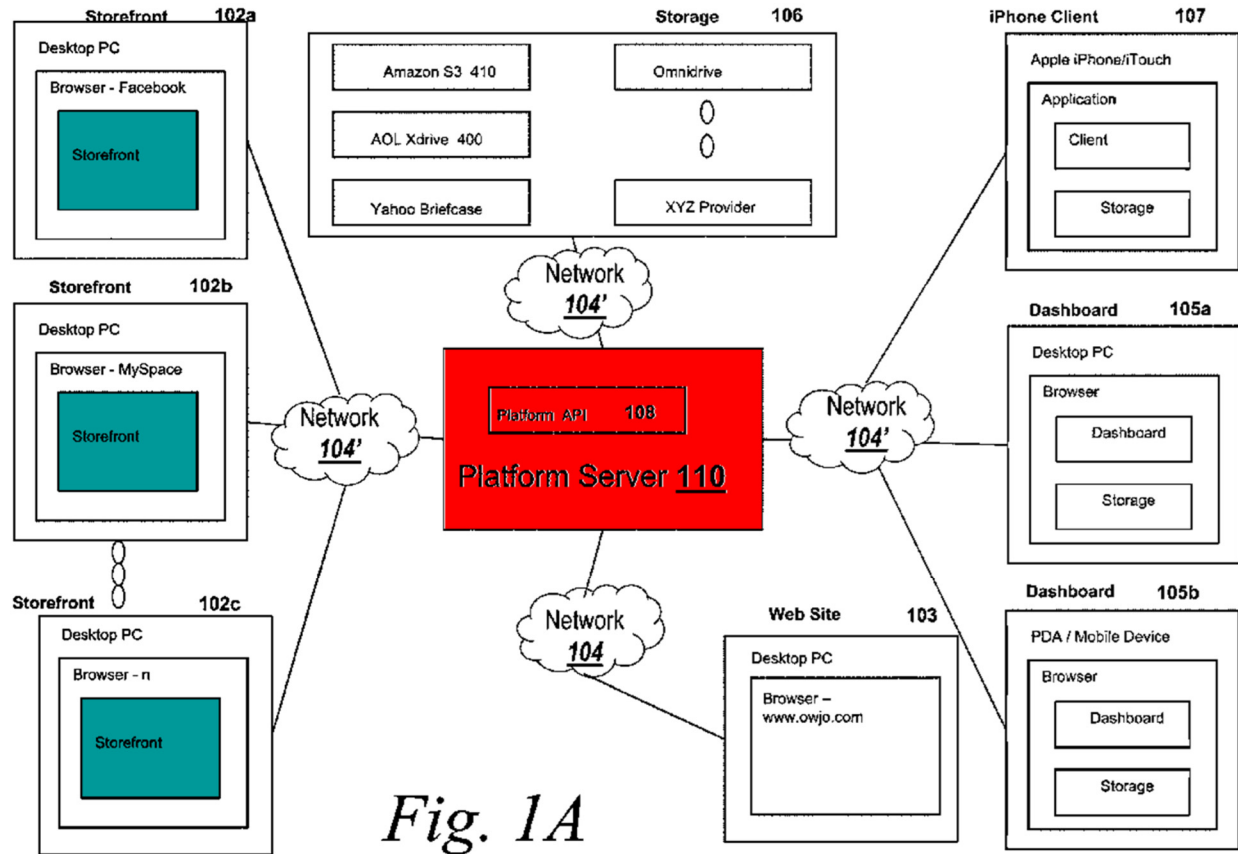


Exhibit 1005, Figure 1A (annotated)

95. For example, Johnston illustrates a more detailed **Storefront** widget including a **product listing** running within a seller's Facebook profile webpage in Figures 4A–4C (Figure 4B below). *See id.*, ¶¶ 252–254. The Facebook profile webpage is “a web site” in “a social media network.”

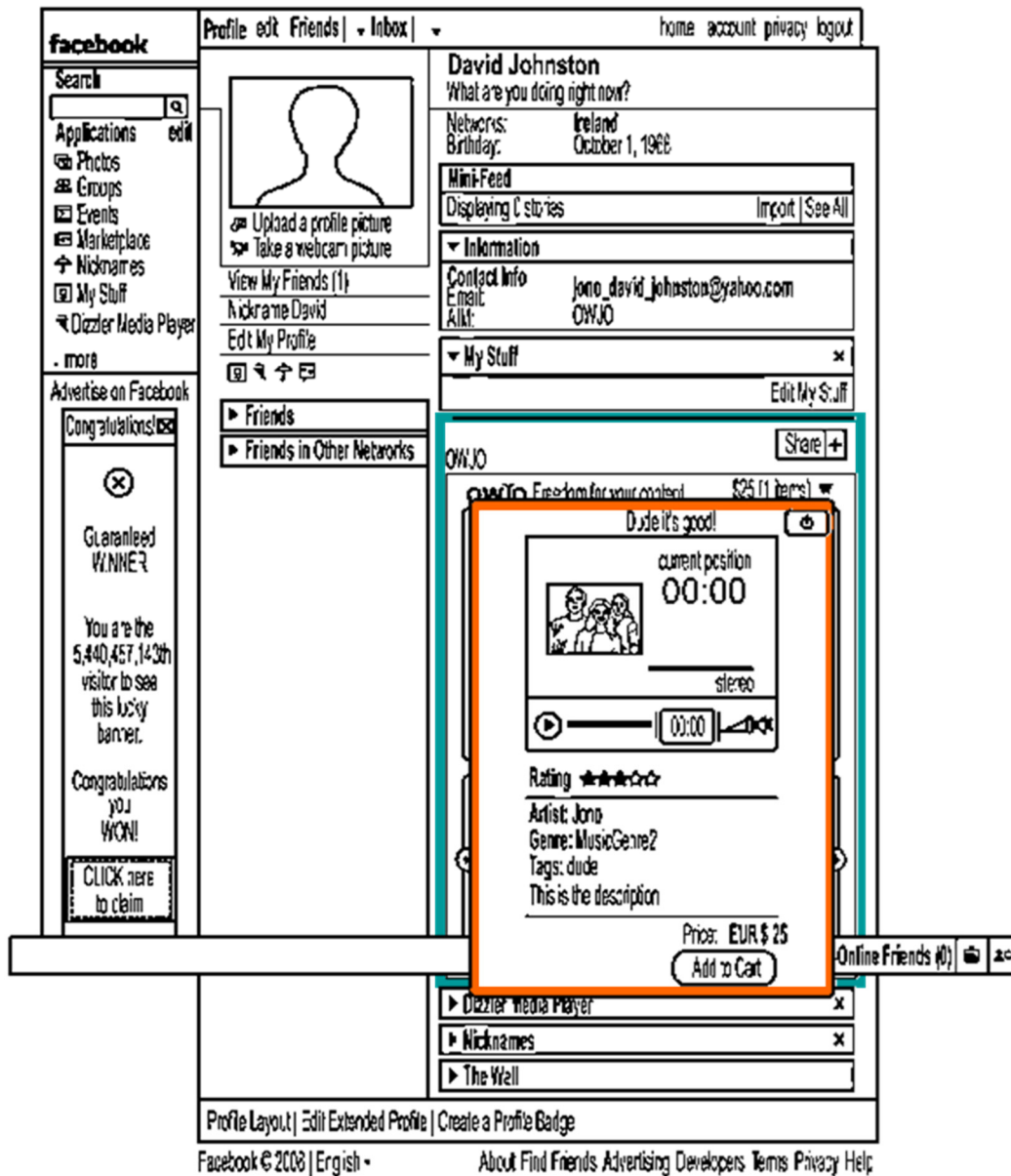


Exhibit 1005, Figure 4B (annotated)

96. Therefore, Johnston discloses element 1[f].

h) 1[g]: “wherein when the application server receives an indication that the clickable link has been activated by a client device in Internet communication with the web site, the application server provides a checkout screen to the client device,”

97. Johnston in combination with Dierks renders obvious element 1[g].

(1) Johnston

98. **First**, for example, Johnston discloses that “the buyer can decide to purchase content by clicking on the ‘**Add to Cart**’ button on FIG. 4B,” in which case the item displayed in the **product listing** in the **Storefront** is added to the user’s shopping cart. Exhibit 1005, ¶ 261. As I discussed for element 1[e], the “**Add to Cart**” button corresponds to “*the clickable link.*” Because the **Storefront** and **product listing** are displayed to the buyer on a website in a social media network, distinct from but in communication over network 104’ with platform server 110, a POSITA would have understood that the buyer would click on the “**Add to Cart**” button on “*a client device in Internet communication with the web site.*” See *id.*, ¶ 114, Figure 1A.

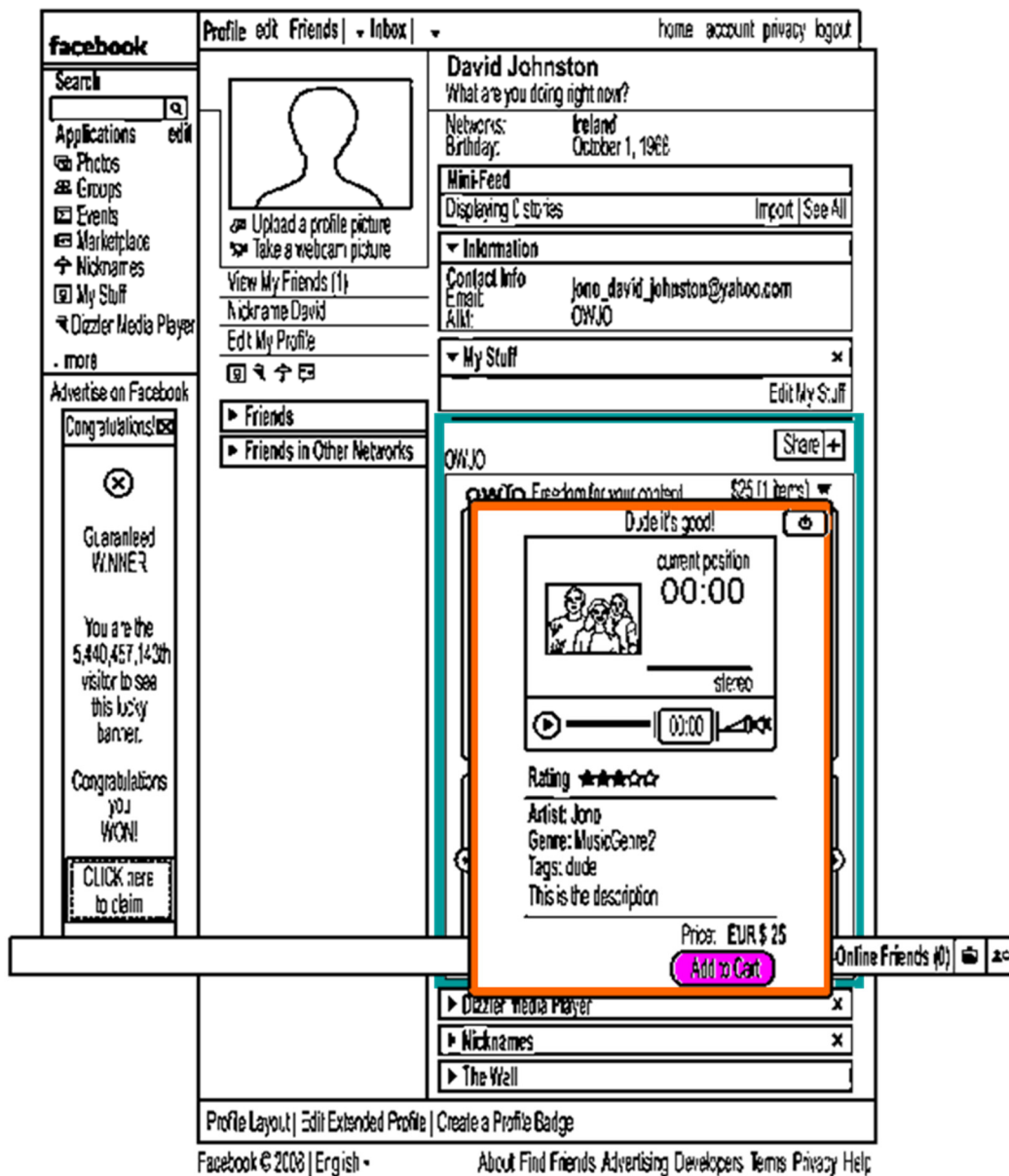


Exhibit 1005, Figure 4B (annotated)

99. A POSITA would have also understood that Johnston's platform server 110 (i.e., "the application server") would receive an indication that the buyer clicked on the "Add to Cart" button to allow platform server 110 to keep track of item(s) added to the user's shopping cart and subsequently complete the checkout process.

100. **Second**, Johnston discloses an exemplary “shopping cart” interface in the **Storefront** in Figure 4C that includes a “Check Out” button and allows a buyer to initiate a checkout process to “purchase items.” See Exhibit 1005, ¶ 254. Johnston illustrates the shopping cart with an item to be purchased along with its cost. Cf. Exhibit 1001, 17:55–56 (“The checkout screen of this web widget 1300 includes the total cost of the content file. ...”). Accordingly, the shopping cart interface corresponds to “a checkout screen.”

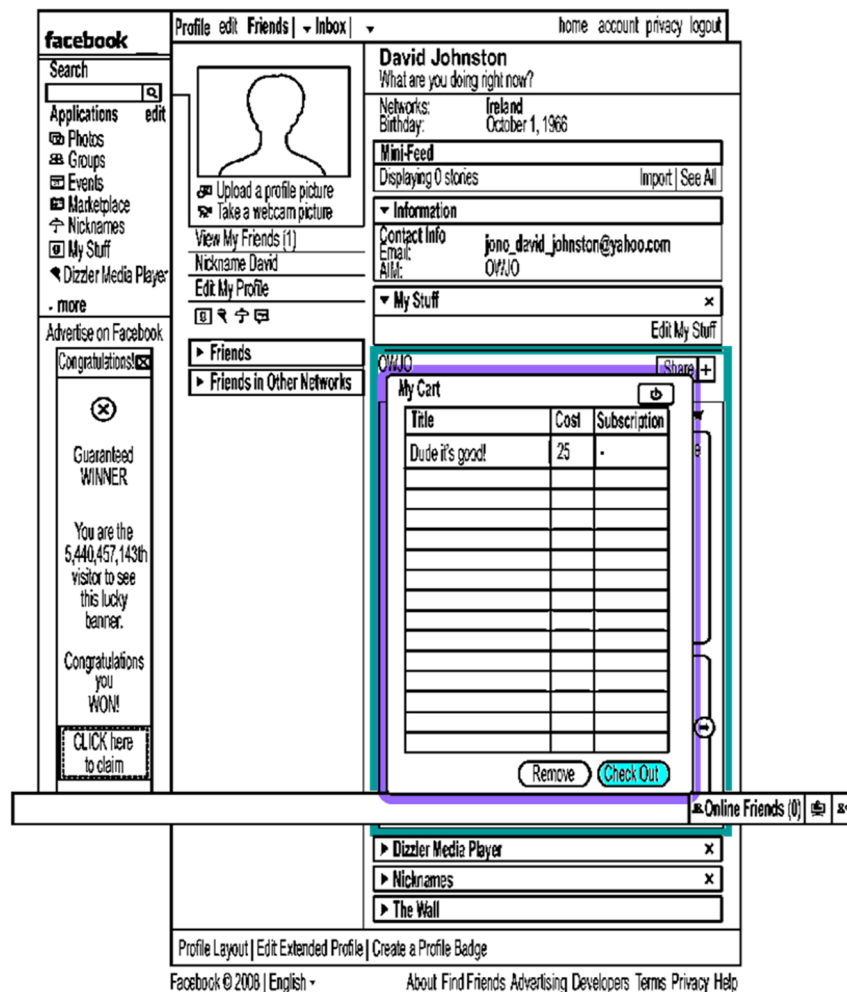


Exhibit 1005, Figure 4C (annotated)

101. In some embodiments, Johnston’s system may prompt the buyer to sign in as “a returning customer” or “create a new OWJO account,” as shown in Figure 4D. EX1005, ¶¶ 255–256. A POSITA would have understood that the interfaces in Figure 4D—also with a “Checkout” button—are part of the checkout process and would not be displayed if the buyer had already signed in.

Home About Us Site Map

Purchase Content

My Basket

Name	Description	Subscription Length	Price
Name	Description	Price	
Dude it's good!	This is the description	25.00	remove...
Total:		25.00	

You are not logged in. To continue with your purchase please enter your login and password. Please select "I am a new customer" tab if you don't have an OWJO account yet.

☐ I am a returning customer ☐ I am a new customer

User Name:

Password:

Checkout

Home About Us Site Map

Purchase Content

My Basket

Name	Description	Subscription Length	Price
Name	Description	Price	
Dude it's good!	This is the description	25.00	remove...
Total:		25.00	

You are not logged in. To continue with your purchase please enter your login and password. Please select "I am a new customer" tab if you don't have an OWJO account yet.

☒ I am a returning customer ☐ I am a new customer

User Name:

New Password:

Confirm Password:

Security Code:
A 2 9 9 1 6

Enter Code:

Checkout

Exhibit 1005, Figure 4D

102. *Third*, because the **Storefront** is displayed on a website in a desktop browser or a distributed application, distinct from platform server 110, a POSITA

would have understood that Johnston’s **shopping cart** (i.e., “*checkout screen*”) is provided to “*the client device*.”

103. **Fourth**, a POSITA would have understood that, because the **shopping cart** is displayed in the **Storefront**, which is distributed by platform server 110 (i.e., “*the application server*”) (see element 1[e]), the platform server 110 also provides the **shopping cart**.

104. **Fifth**, Johnston discloses that a “buyer can view the contents of the buyer’s shopping cart at any time by clicking the shopping cart icon.” Exhibit 1005, ¶ 261. However, a POSITA would have been motivated by Dierks’s teachings—as I will discuss next—to configure Johnston’s platform server 110 to display the **shopping cart** when the buyer clicks on the “**Add to Cart**” button.

(2) Dierks

105. Like Johnston, Dierks discloses a **product listing** with an “**Add to Cart**” button 105. Exhibit 1004, Figure 1B, ¶¶ 36, 71, 74.

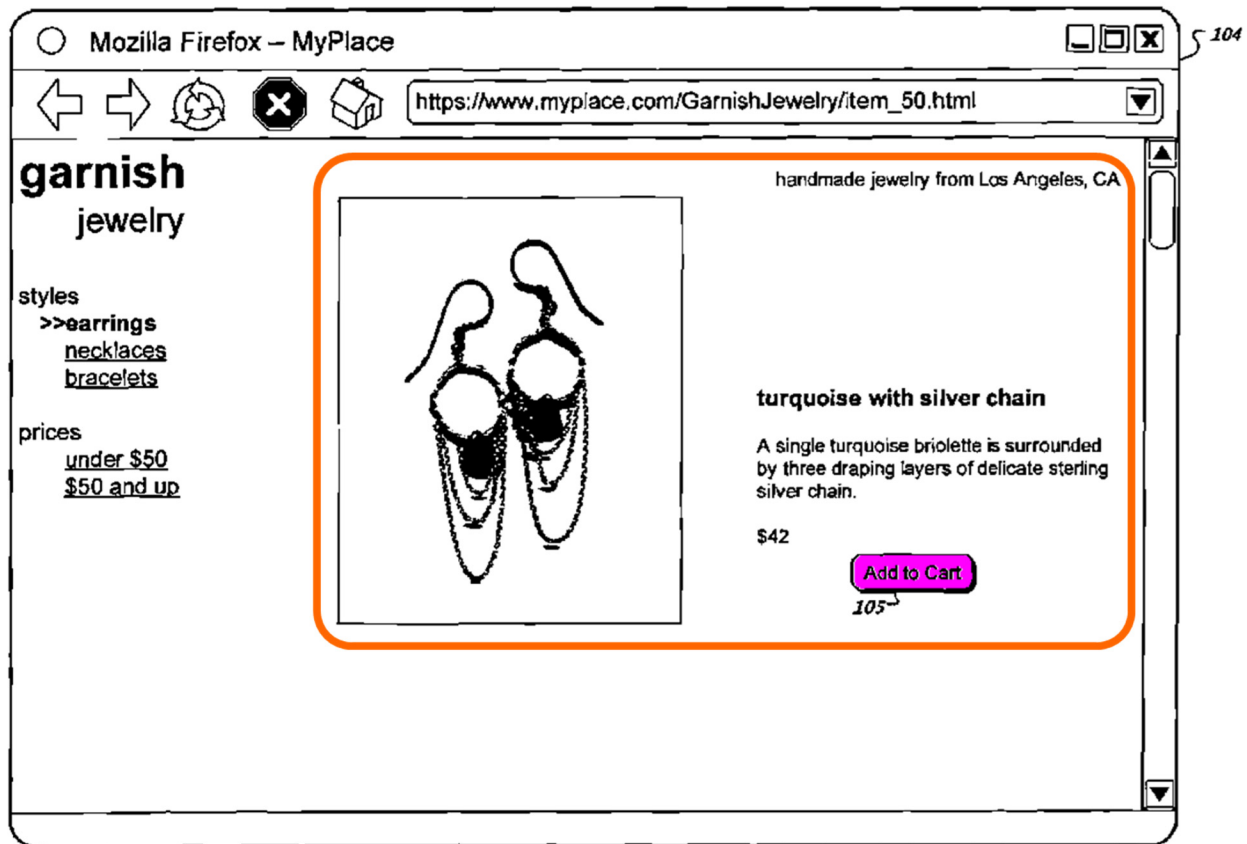


Exhibit 1004, Figure 1B (annotated)

106. Dierks explains that, when a user selects the “**Add to Cart**” button 105, “a **shopping cart** object shown as a graphical element such as an iframe 116” is displayed in the web page as shown in Figure 1C (reproduced below). *Id.* at ¶ 43. Like in Johnston, the **shopping cart** includes a “**Proceed to Checkout**” button 118 and allows a purchaser to initiate a checkout process and make a payment. *Id.* at ¶ 44.

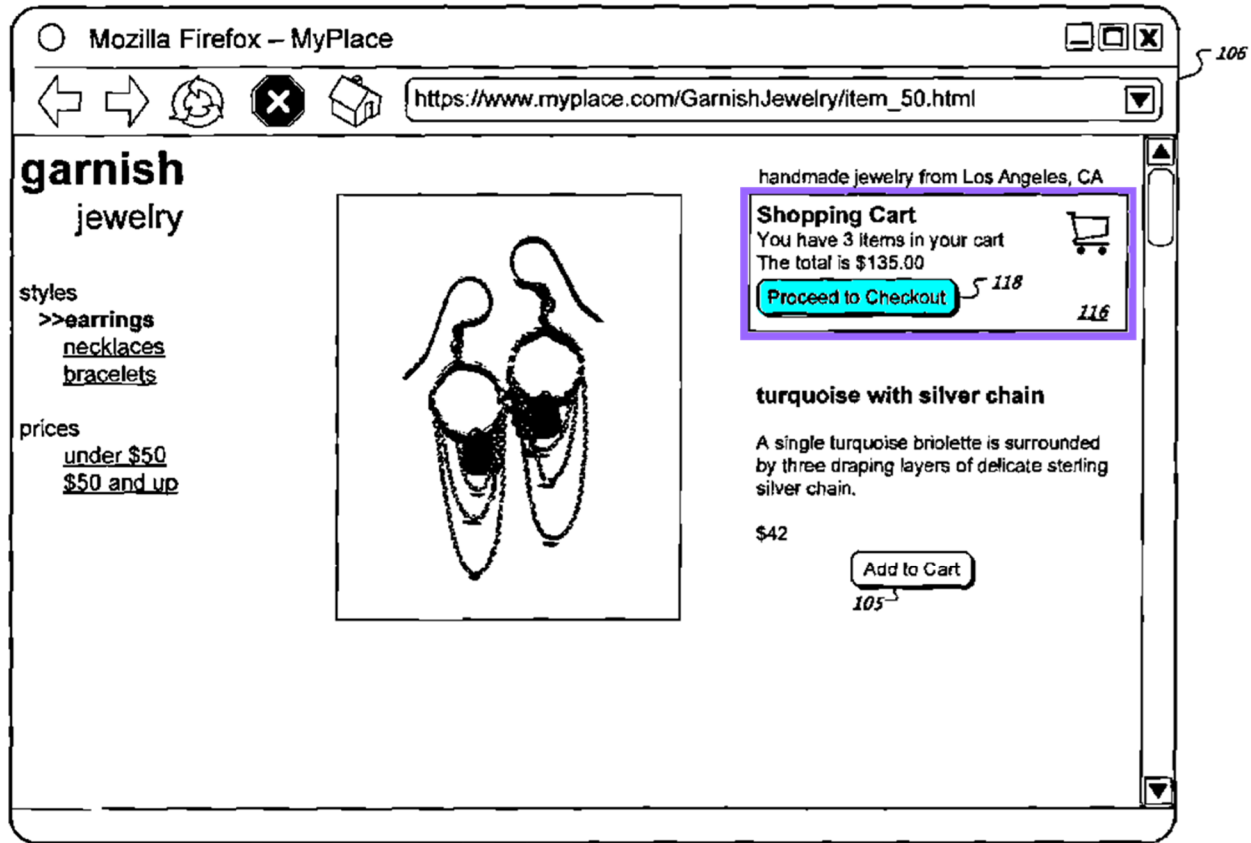


Exhibit 1004, Figure 1C (annotated)

107. A POSITA would have been motivated to configure Johnston's platform server 110 to display the **shopping cart** when the buyer clicks on the "**Add to Cart**" button, as taught by Dierks. Doing so would save the buyer from having to click on the shopping cart icon to check out, thereby streamlining the checkout process for the buyer, requiring fewer clicks and fewer buttons since the shopping cart icon could be removed. Such a configuration would also provide the buyer with instant confirmation that the product has been added to the shopping cart.

(3) Motivation to Combine Johnston and Dierks

108. A POSITA would have been motivated, according to and informed by Dierks's teachings, to configure Johnston's platform server 110 to display the shopping cart when the "Add to Cart" button is clicked.

109. **First**, Johnston and Dierks are from the same field of endeavor. Both references are directed to electronic commerce. Johnston discloses "systems and methods for providing a platform for the delivery, management and payment of content between sellers and buyers," in particular "systems and methods of self-service by a seller to establish a store front at one or more hosted services." Exhibit 1005, ¶2. Dierks discloses a "computer-implemented method [for] generating on a user terminal a merchant web site associated with a merchant with code for a merchant web page." Exhibit 1004, Abstract. Thus, both references are directed to electronic commerce. Accordingly, a POSITA would have understood that Dierks's teachings were directly applicable to Johnston's system.

110. **Second**, implementing Johnston's platform server 110 to display the shopping cart when the "Add to Cart" button is clicked, as taught by Dierks, would have yielded expected, predictable results. Johnston discloses a product listing with an "Add to Cart" button in a Storefront, and a shopping cart that is displayed when a shopping cart icon in the Storefront is clicked. *See* Exhibit 1005, Figure 4B, ¶ 261; *see also* Section IX.A.1.h)(1). Dierks discloses, in a web page, a product listing with an "Add to Cart" button 105, which, when clicked, displays a shopping cart 116.

See Exhibit 1004, Figures 1B, ¶¶ 36, 71, 74; *see also* Section IX.A.1.h)(2). A POSITA would have combined the elements taught by Johnston and Dierks—namely, an “Add to Cart” button and shopping cart in a Storefront as taught by Johnston and an “Add to Cart” button that displays a shopping cart in a web page as taught by Dierks—using known and routine computer programming principles to configure Johnston’s platform server 110 to display the shopping cart when the “Add to Cart” button is clicked. This result would have been readily predictable and recognized by a POSITA.

111. *Third*, configuring Johnston’s platform server 110 to display the shopping cart when the “Add to Cart” button is clicked, as taught by Dierks, would have been readily implemented by configuring code at Johnston’s platform server 110 to display the shopping cart in the Storefront when it receives an indication that the “Add to Cart” has been clicked. A POSITA could have used basic computer programming principles to configure Johnston’s platform server 110 to display the shopping cart when the “Add to Cart” button is clicked. This implementation would have been well within the knowledge and skillset of a POSITA.

(4) Reasonable Expectation of Success in Combining Johnston and Dierks

112. A POSITA would also have had a reasonable expectation of success. The combination of Johnston and Dierks represents a straightforward configuration

of Johnston's platform server 110 to display the shopping cart when the "Add to Cart" button is clicked, which a POSITA would have been familiar with and been able to implement. The reasons a POSITA would have expected success parallel those that provide motivation for this combination—including because both Johnston and Dierks are directed to electronic commerce and both disclose similar product listings. *See, e.g.*, Exhibit 1005, Figure 4B, ¶ 261; Exhibit 1004, Figures 1B, ¶¶ 36, 71, 74; *see also* Section IX.A.1.h)(3).

113. Accordingly, Johnston in view of Dierks renders obvious element 1[g] and a POSITA would have been motivated to combine the teachings of these references and would have had a reasonable expectation of success. For these same reasons, a POSITA would have been motivated to combine the teachings of these references for all subsequently discussed claims and limitations in this ground.

i) 1[h]: "wherein the transaction processing service processes a payment transaction for a purchase of the product initiated from the checkout screen using the client device."

114. Johnston discloses element 1[h].

115. **First**, as I discussed for element 1[g], a buyer adds an item (e.g., music track titled "Dude it's good!"; "*the product*") displayed in the **product listing** to a shopping cart by clicking on the "**Add to Cart**" button. Exhibit 1005, ¶ 261, Figure 4B.

116. **Second**, as I discussed for element 1[g], Johnston’s **shopping cart** (i.e., “*the checkout screen*” with the “Dude it’s good!” music track) is provided on “*the client device*” and allows a buyer to “purchase items.” Exhibit 1005, ¶ 254. Thereafter, when the buyer clicks on the “**Check Out**” button (and signs in if not already), Johnston’s system prompts the buyer to select a payment method. *See id.*, ¶¶ 254–257. Thus, the “*purchase of the product [is] initiated from the checkout screen using the client device.*”

117. **Third**, Johnston discloses an exemplary interface of the Storefront, illustrated in Figure 4E, that provides the buyer with payment options to buy credits, which may then be used to purchase an item. Exhibit 1005, ¶ 257. As I discussed for element 1[a], Johnston’s micro payment service 145 (i.e., “*the transaction processing service*”) allows “buyers to pay for products at low values or for small amounts of monies without incurring credit card or other third-party transaction fees.” *Id.*, ¶ 185. Johnston explains that “this is possible ... through the use of OWJO Credits.” *Id.*, ¶ 186. Thus, using credits, Johnston’s micro payment service 145 “*processes a payment*” for an item in the buyer’s shopping cart.

Welcome, rachel_bye

Checkout

Amount to be billed: **25.00**

OWJO Account Balance: **0.00**

PayPal Top Up by Paypal

€5 OWJO Credits

Buy Now

€10 OWJO Credits

Buy Now

€20 OWJO Credits

Buy Now

€30 OWJO Credits

Buy Now

 Top Up by Mobile

 **Ireland:**

Top up by 2.00

Text OWJO01 to 50102

1. Service Charge

Top up by 4.00

Text OWJO01 to 50102

2. Service Charge

 **UK:**

Top up by 2.00

Text OWJO01 to 50102

1. Service Charge

Top up by 4.00

Text OWJO01 to 50102

2. Service Charge

Redeem Code:

Redeem

Exhibit 1005, Figure 4E

118. Therefore, Johnston discloses element 1[h].

119. For the above reasons, Johnston in view of Dierks renders obvious claim 1.

2. Dependent Claim 2

a) “The system of claim 1, wherein the metadata includes at least one of a title, a description, or a price.”

120. Johnston in view of Dierks renders obvious claim 1, as I discussed above. Johnston also renders obvious the additional limitation of claim 2. As I discussed for element 1[b], Johnston discloses storing content metadata (i.e., “*the metadata*”) along with a music file in storage 106. *See* Exhibit 1005, ¶ 309. And, as I discussed for element 1[e], a POSITA would have understood that one or more of the pieces of **information** displayed in the **product listing**—“Dude it’s good!” (i.e., “*a title*”), “This is the description” (i.e., “*a description*”), “EUR \$ 25” (i.e., “*a price*”)—could be included in the content metadata.

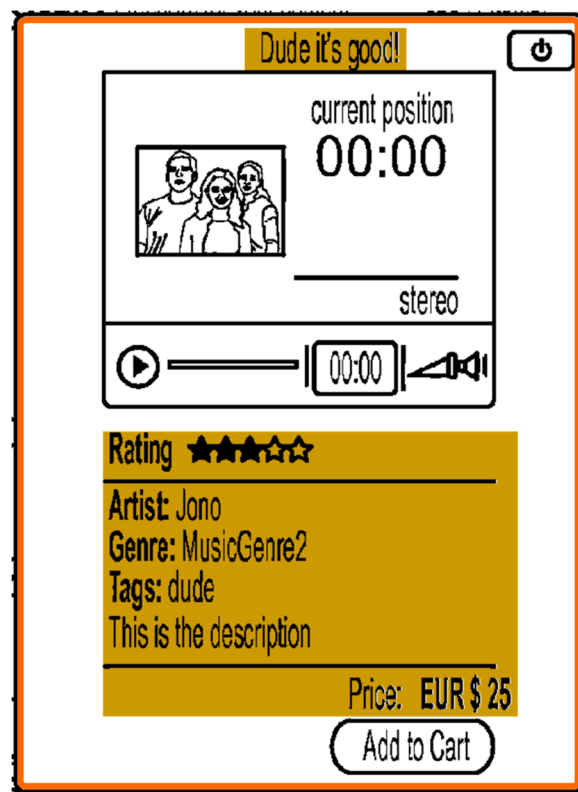


Exhibit 1005, Figure 4B (excerpted and annotated)

121. Thus, Johnston in view of Dierks renders claim 2 obvious.

3. Dependent Claim 3

a) “The system of claim 1, wherein the web site is a social media web site.”

122. Johnston in view of Dierks renders obvious claim 1, as I discussed above. Johnston discloses the additional limitation of claim 3. As I discussed for element 1[f], Johnston discloses embedding a **Storefront** widget including a **product listing** into a seller’s Facebook profile webpage (i.e., “the website”). See Exhibit 1005, ¶¶ 252–254, Figures 4A–4C; *see also id.*, ¶ 231, Figure 3G. A Facebook webpage is “a social media web site.”

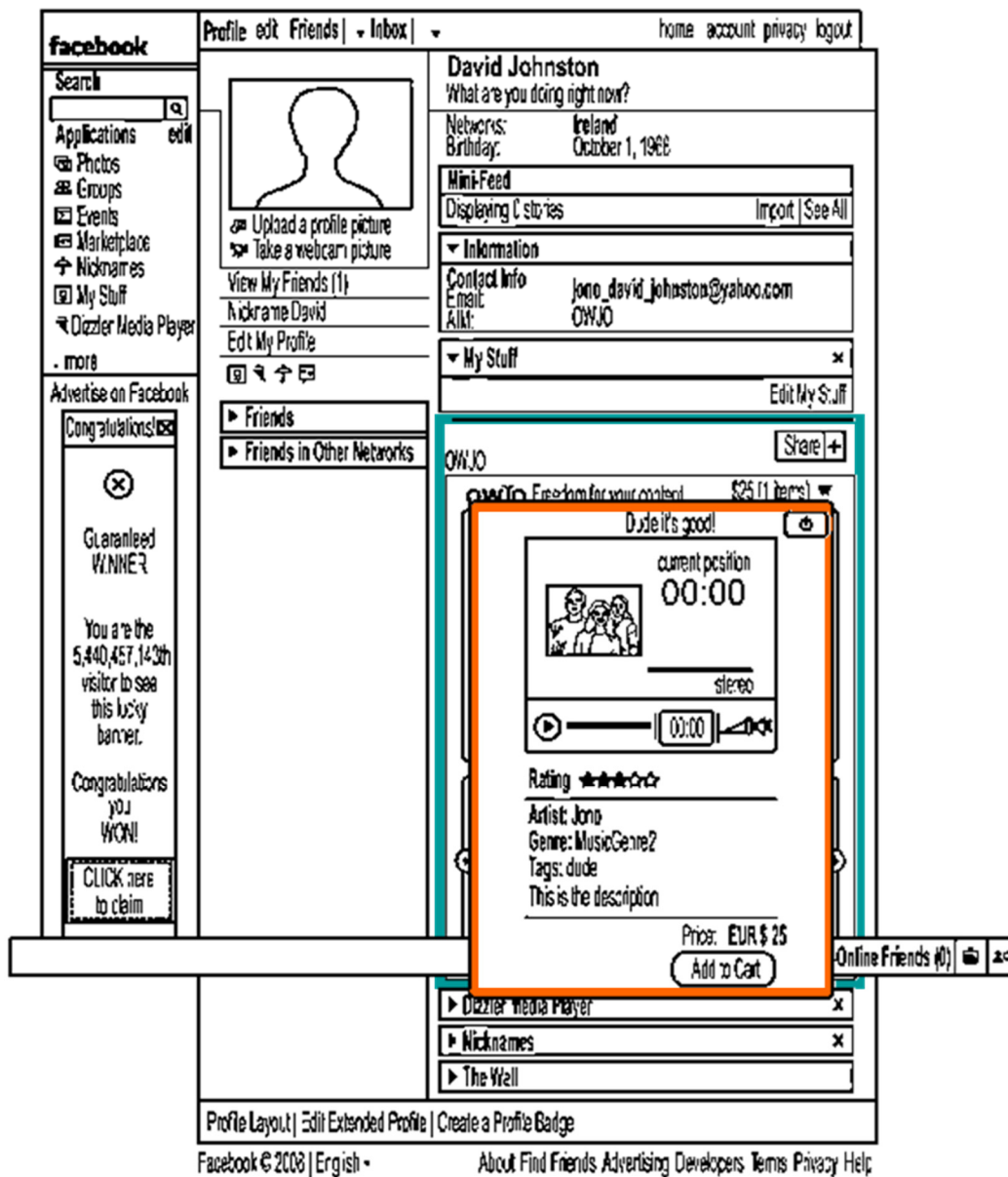


Exhibit 1005, Figure 4B (annotated)

123. Thus, Johnston in view of Dierks renders claim 3 obvious.

4. Dependent Claim 4

a) *“The system of claim 1, wherein the transaction processing service accepts a credit card.”*

124. Johnston in view of Dierks renders obvious claim 1, as I discussed above. Johnston also renders obvious the additional limitation of claim 4. Johnston discloses that **micro payment service 145** (i.e., “*the transaction processing service*”) “provides a means for sellers and buyers to pay for products at low values or for small amounts of monies without incurring credit card or other third-party transaction fees.” Exhibit 1005, ¶ 185. Johnston further explains that “this is possible ... through the use of OWJO Credits” and that “OWJO credits in a buyer’s account may be replenished automatically (e.g., via **credit card** to maintain a minimum threshold in the account), or as directed by the buyer.” *Id.*, ¶¶ 186, 258. Thus, Johnston’s **micro payment service 145** “*accepts a credit card.*”

125. Thus, Johnston in view of Dierks renders claim 4 obvious.

5. Dependent Claim 5

a) “The system of claim 1, further comprising, distributing the product listing to a social media network.”

126. Johnston in view of Dierks renders obvious claim 1, as I discussed above. Johnston discloses the additional limitation of claim 5 for at least the same reasons discussed for element 1[e]. Moreover, referring to Figure 8B, Johnston discloses that “[a] seller can decide to promote the seller’s OWJO **Storefront** to some or all of their contacts on Facebook or MySpace by clicking on the ‘**share**’ link.” Exhibit 1005, ¶ 286. As Johnston illustrates in Figure 8B, when a seller **shares** the seller’s Storefront to the seller’s contacts on Facebook or MySpace, the

platform server 110 distributes the **Storefront** including the product listing to the social media network (e.g., “Facebook” or “MySpace”) of the seller’s contacts (on the right of Figure 8B).

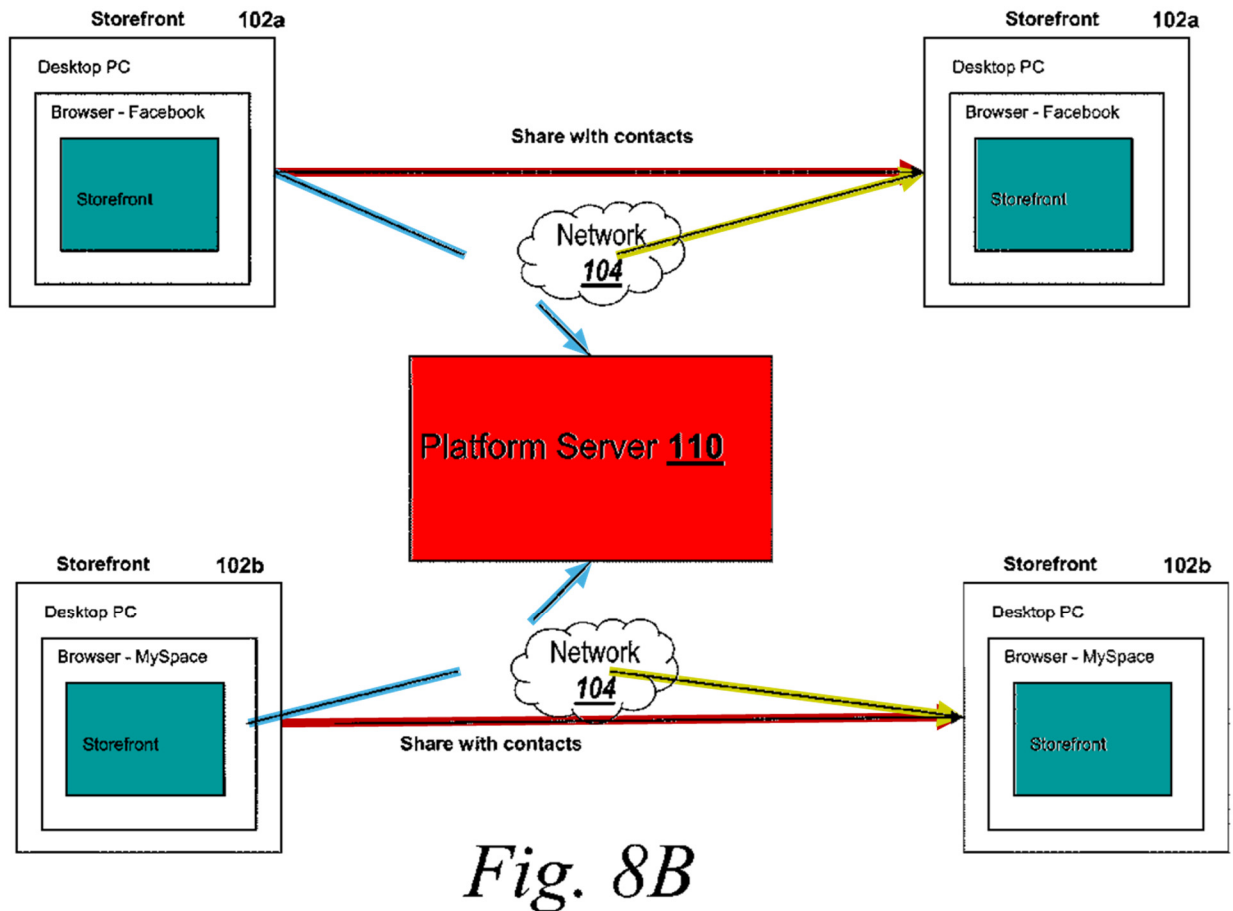


Exhibit 1005, Figure 8B (annotated)

127. Thus, Johnston in view of Dierks renders claim 5 obvious.

6. Independent Claim 6

a) 6[preamble]: “A system for generating an electronic commerce store, comprising:”

128. In my opinion, even if the preamble is limiting, Johnston discloses it for at least the same reasons discussed for element 1[preamble].

b) 6[a]: “an application server, the application server controlling a transaction processing service;”

129. Johnston discloses element 6[a] for at least the same reasons discussed for element 1[a].

c) 6[b]: “a storage resource communicatively coupled to the application server, the storage resource storing an image and metadata related to a product;”

130. Johnston discloses element 6[b] for at least the same reasons discussed for element 1[b].

d) 6[c]: “a widget builder resource executing on the application server,”

131. Johnston discloses element 6[c] for at least the same reasons discussed for element 1[c].

e) 6[d]: “the widget builder resource is programmed to: (i) receive the image and metadata from the storage resource;”

132. Johnston discloses element 6[d] for at least the same reasons discussed for element 1[d].

f) 6[e]: “[the widget builder resource is programmed to:] (ii) generate a product listing, wherein the product listing includes the image, the metadata, and a clickable link, and wherein the clickable link is a buy button and includes a preview of the product, and”

133. Johnston discloses element 6[e] for at least the same reasons discussed for element 1[e].

g) 6[f]: “wherein the application server is configured to embed the product listing into a web site not operated by the application server and into a social media network, and”

134. Johnston discloses element 6[f] for at least the same reasons discussed for element 1[f]. Element 6[f] further requires that the web site is “*not operated by the application server.*” Johnston discloses that the web site is “*not operated by the server.*” As I discussed for element 1[f], Johnston discloses embedding a **Storefront** widget including a **product listing** into a seller’s Facebook profile webpage (i.e., “*the website*”). See Exhibit 1005, ¶¶ 252–254, Figures 4A–4C; see also *id.*, ¶ 231, Figure 3G. A Facebook webpage is “*a web site not operated by the application server.*”

135. Therefore, Johnston discloses element 6[f].

h) 6[g]: “wherein when the application server receives an indication that the clickable link has been activated by a client device in Internet communication with the web site, the application server provides a checkout screen to the client device,”

136. Johnston in view of Dierks renders obvious element 6[g] for at least the same reasons discussed for element 1[g].

i) 6[h]: “wherein the transaction processing service processes a payment transaction for a purchase of the product initiated from the checkout screen using the client device.”

137. Johnston discloses element 6[h] for at least the same reasons discussed for element 1[h].

138. Therefore, Johnston in view of Dierks renders claim 6 obvious.

7. Dependent Claim 7

a) “The system of claim 6, wherein the application server is capable of compiling analytics related to the product listing.”

139. Johnston in view of Dierks renders obvious claim 6, as I discussed above. Johnston discloses the additional limitation of claim 7. For example, in describing the features of platform server 110 (i.e., “the application server”) in Figure 1B, Johnston discloses that “[t]he OWJO Platform may collect statistics on sales activity across some or all the OWJO sellers, the type of content and where that content has been sold.” Exhibit 1005, ¶¶ 161, 176. Thus, platform server 110 “is capable of compiling analytics related to the product listing.”

140. Thus, Johnston in view of Dierks renders claim 7 obvious.

8. Dependent Claim 8

a) “The system of claim 6, wherein the product listing further includes a share link.”

141. Johnston in view of Dierks renders obvious claim 6, as I discussed above. Johnston also renders obvious the additional limitation of claim 8. In Johnston’s Figures 4A–4C, the **Storefront** includes a “**Share**” link. Johnston explains that “[a] seller can decide to promote the seller’s OWJO **Storefront** to some

or all of their contacts on Facebook or MySpace by clicking on the ‘**share**’ link.” Exhibit 1005, ¶ 286. A POSITA would have found it obvious to include a similar **share** link in every **product listing**. It was known before the ’995 patent’s priority date to include a share link in a product listing as disclosed, for example, by Chang. *See, e.g.*, Exhibit 1008, ¶43. A POSITA would have been motivated to do so to allow the seller to promote specific product listings from the seller’s Storefront rather than the entire Storefront. Implementing Johnston’s Storefront such that every product listing includes a share link would have been well within the knowledge and skillset of a POSITA and would have been effected using basic computer programming principles.

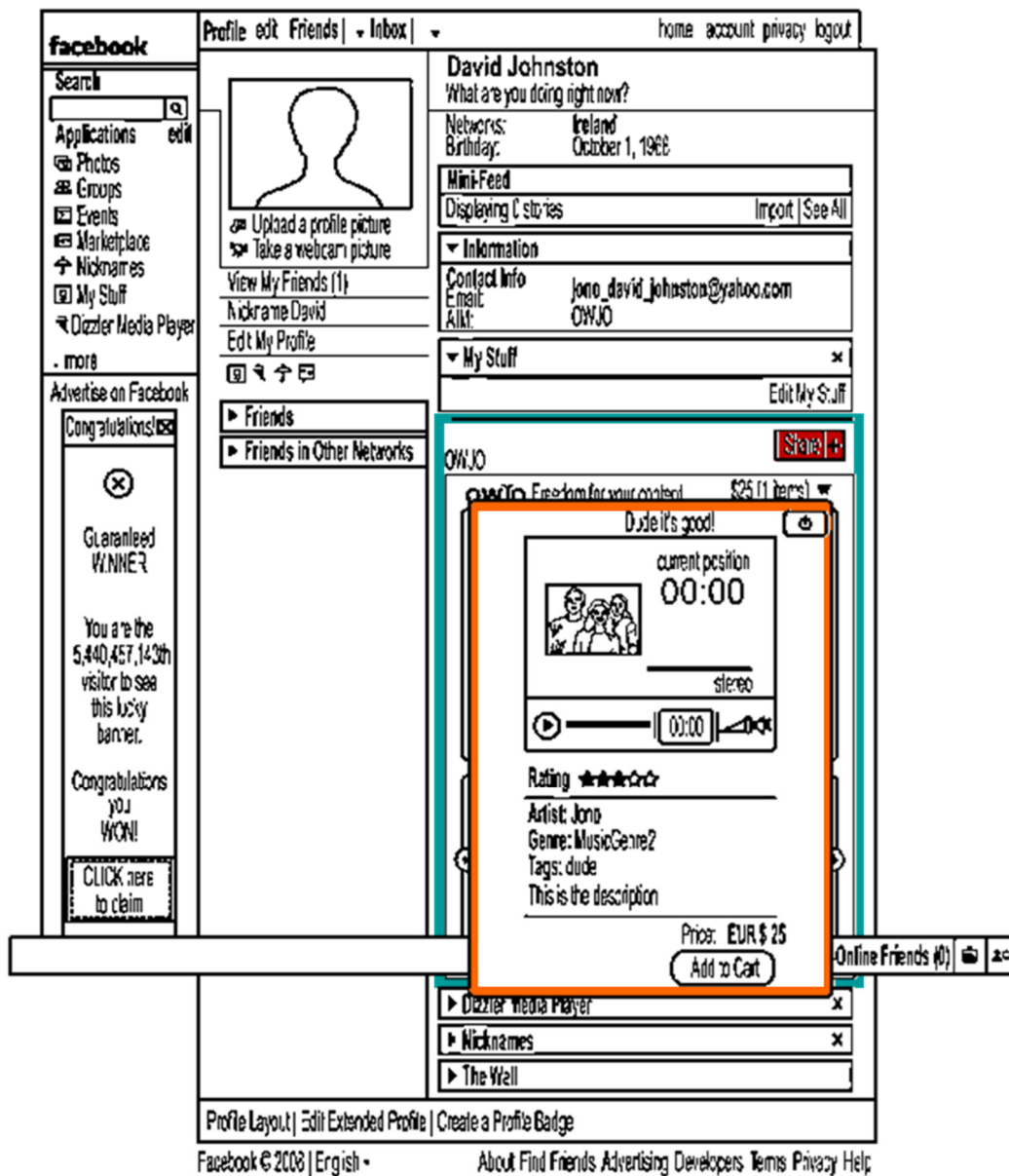


Exhibit 1005, Figure 4B (annotated)

142. Thus, Johnston in view of Dierks renders claim 8 obvious.

9. Dependent Claim 9

a) *“The system of claim 6, wherein the web site is a social media network site.”*

143. Johnston in view of Dierks renders obvious claim 6, as I discussed above. Johnston discloses the additional limitation of claim 9. As I discussed for element 1[f], Johnston discloses embedding a **Storefront** widget including a **product listing** into a seller's Facebook profile webpage (i.e., "the website"). See Exhibit 1005, ¶¶ 252–254, Figures 4A–4C; see also *id.*, ¶ 231, Figure 3G. Facebook is a well-known social media network. Thus, a Facebook profile webpage is "*a social media network site.*"

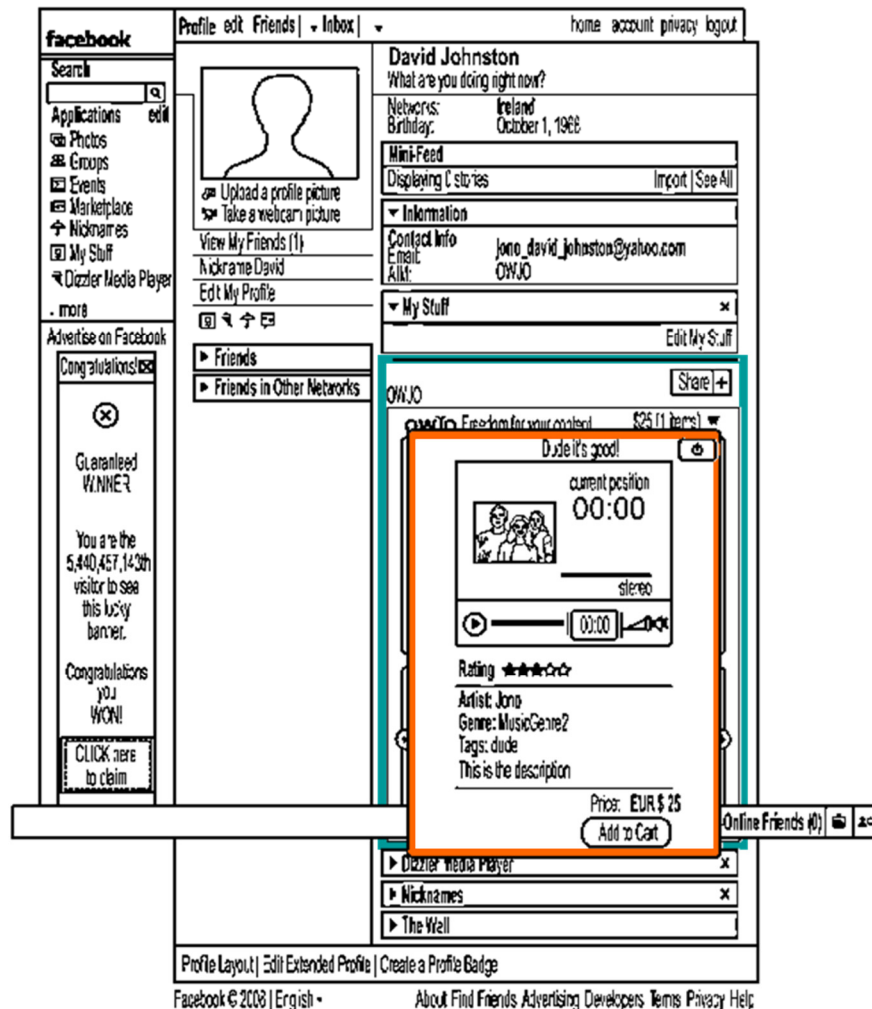


Exhibit 1005, Figure 4B (annotated)

144. Thus, Johnston in view of Dierks renders claim 9 obvious.

10. Dependent Claim 10

a) “The system of claim 6, wherein the transaction processing service is a proprietary service controlled by the application server.”

145. Johnston in view of Dierks renders obvious claim 6, as I discussed above. Johnston discloses the additional limitation of claim 10. As I discussed for element 1[a], Johnston’s **platform server 110** (i.e., “the application server”) controls **micro payment service 145** (i.e., “the transaction processing service”). In Figure 1C, Johnston illustrates **micro payment service 145** as part of the billing module 158, which is part of **platform server 110**. Johnston explains that **micro payment service 145** “provides a means for sellers and buyers to pay for products at low values or for small amounts of monies without incurring credit card or other third-party transaction fees.” Exhibit 1005, ¶ 185. Johnston further explains that “this is possible ... through the use of OWJO Credits” and that, in some embodiments, “transacting with OWJO Credits may incur a one-time fee or periodic fees (e.g., indirectly through OWJO membership fees), a low fee or indirect service charges (e.g., via OWJO revenue-sharing on each transaction[]).” *Id.*, ¶ 186. Johnston’s **micro payment service 145** is managed internally by OWJO and is thus “a proprietary service.” Cf. Exhibit 1001, 18:13–16 (“the transaction processing service 112 is implemented with a wholly internal, proprietary service that is not

generally accessible over the Internet or other public computer communications networks”).

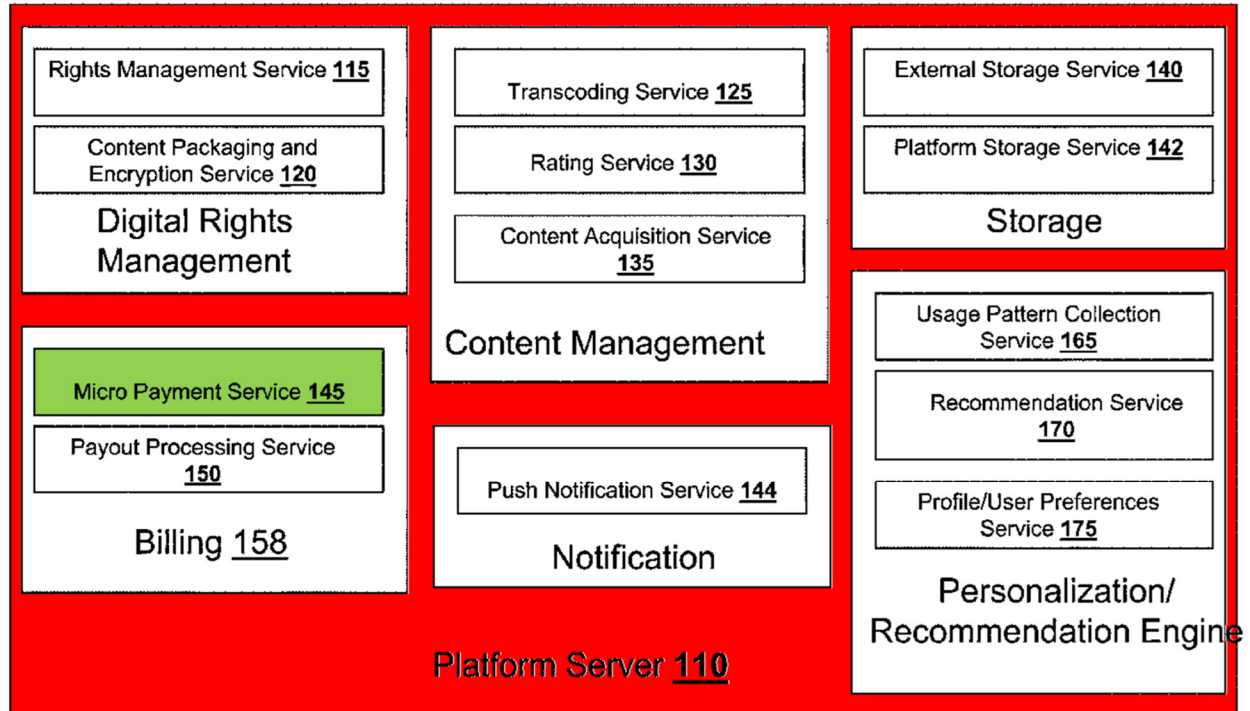


Exhibit 1005, Figure 1C (annotated)

146. Thus, Johnston in view of Dierks renders claim 10 obvious.

11. Independent Claim 11

a) 11[preamble]: “A system for generating an electronic commerce store, comprising:”

147. In my opinion, even if the preamble is limiting, Johnston discloses it for at least the same reasons discussed for element 1[preamble].

b) 11[a]: “an application server, the application server controlling a transaction processing service;”

148. Johnston discloses element 11[a] for at least the same reasons discussed for element 1[a].

c) 11[b]: “a storage resource communicatively coupled to the application server, the storage resource storing an image and metadata related to a product;”

149. Johnston discloses element 11[b] for at least the same reasons discussed for element 1[b].

d) 11[c]: “a widget builder resource executing on the application server,”

150. Johnston discloses element 11[c] for at least the same reasons discussed for element 1[c].

e) 11[d]: “the widget builder resource configured to: receive the image and metadata from the storage resource;”

151. Johnston discloses element 11[d] for at least the same reasons discussed for element 1[d].

f) 11[e]: “[the widget builder resource configured to:] generate a product listing that includes the image, the metadata, and a clickable link, wherein the clickable link is a buy button and includes a preview of the product, and”

152. Johnston discloses element 11[e] for at least the same reasons discussed for element 1[e].

g) 11[f]: “wherein the application server is configured to embed the product listing into a social media network and into a web site, and”

153. Johnston discloses element 11[f] for at least the same reasons discussed for element 1[f].

h) 11[g]: “wherein when the application server receives an indication that the clickable link has been activated by a client device in Internet communication with the social media network, the application server provides a checkout screen to the client device,”

154. Johnston in view of Dierks renders obvious element 11[g] for at least the same reasons discussed for element 1[g]. Element 11[g] recites “*a client device in Internet communication with the social media network*” rather than “*with the website*.” As I discussed for element 1[e], Johnston’s “Add to Cart” button (i.e., “*the clickable link*”) is included in a **product listing** in a **Storefront**. And as I discussed for element 1[f], the **Storefront** along with the **product listing** is embedded in a seller’s Facebook profile webpage, which corresponds to “*the website*” in “*the social media network*.” Thus, when the client device is in communication with the website, it is also in communication with the social media network.

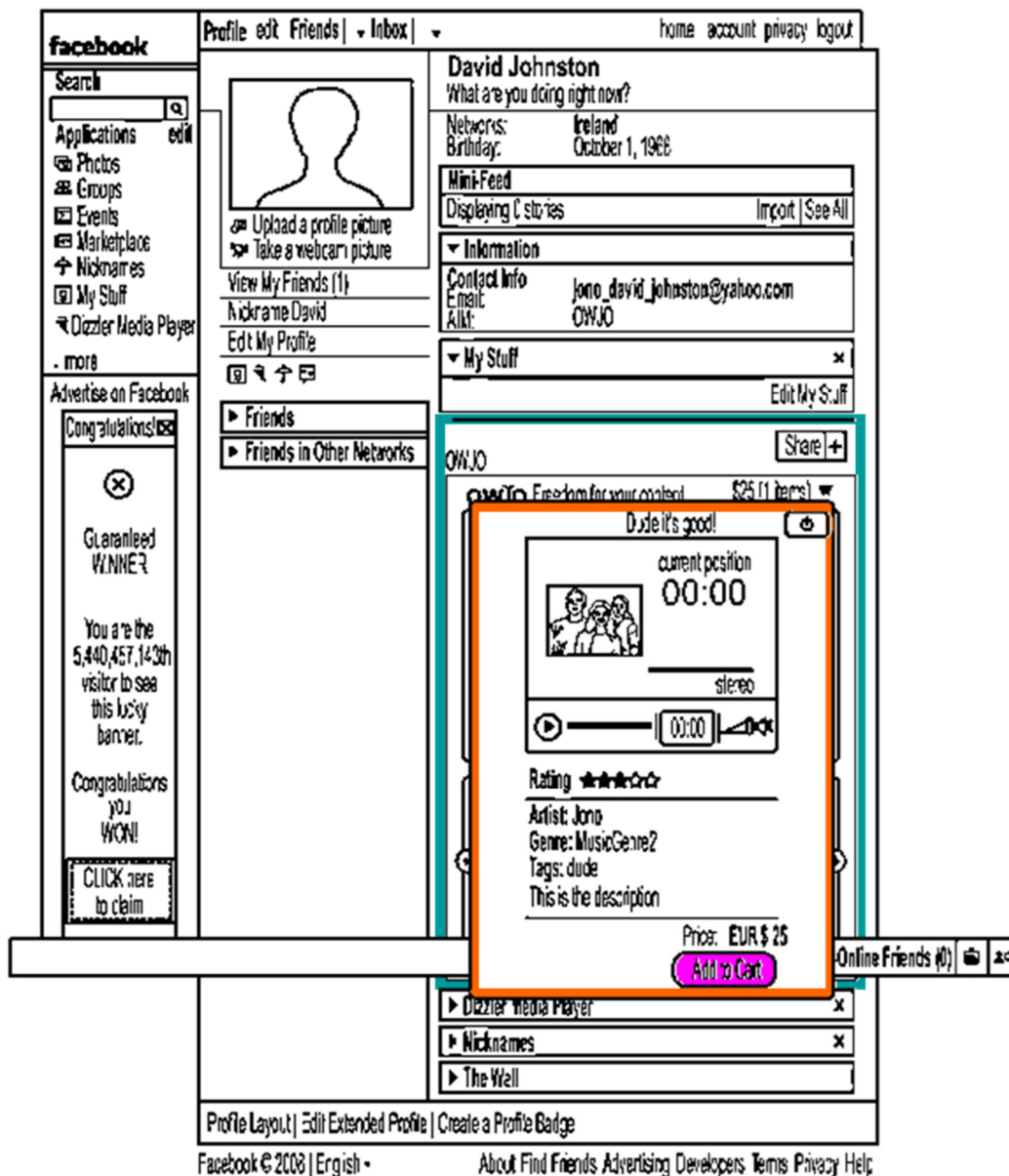


Exhibit 1005, Figure 4B (annotated)

155. Therefore, Johnston in view of Dierks renders obvious element 11[g].

i) **11[h]:** “wherein the transaction processing service is configured to process a payment transaction for a purchase of the product initiated from the checkout screen using the client device.”

156. Johnston discloses element 11[h] for at least the same reasons discussed for element 1[h].

157. Therefore, Johnston in view of Dierks renders obvious claim 11.

12. Dependent Claim 12

a) *“The system of claim 11, wherein the social media network is capable of displaying multiple product listings.”*

158. Johnston in view of Dierks renders obvious claim 11, as I discussed above. Johnston discloses the additional limitation of claim 12. In Figure 4B, Johnston illustrates an exemplary **product listing** for a music track named “Dude it’s good!” in the seller’s **Storefront** in the seller’s profile webpage in Facebook.

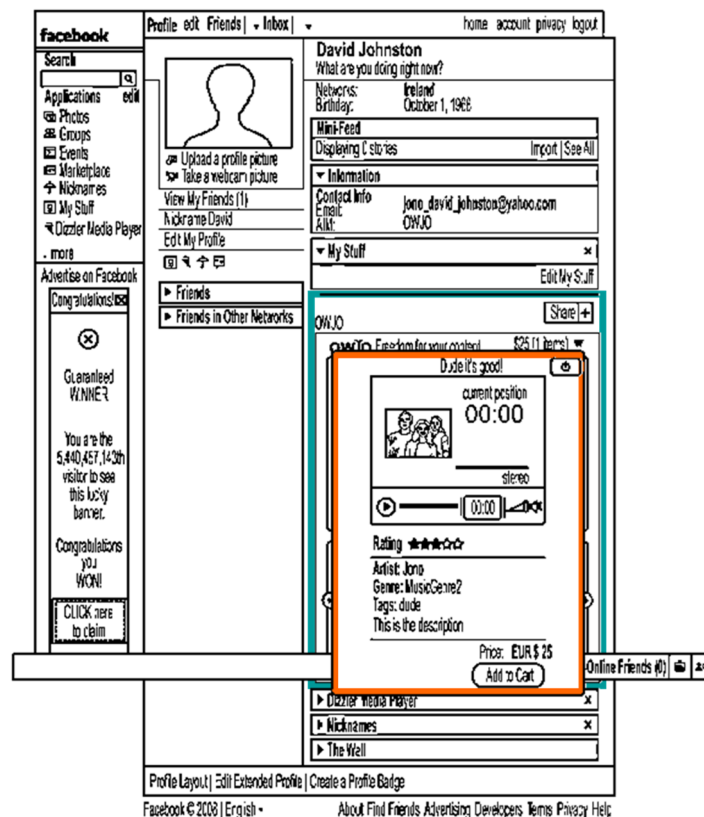


Exhibit 1005, Figure 4B (annotated)

159. This **product listing** is for one of three music tracks that Johnston illustrates in Figure 4A. A POSITA would have understood that clicking on each of the other two music tracks—“Running after Cars” and “Comfortably Numb”—would display a product listing similar to the one in Figure 4B. That is, Facebook (i.e., “the social media network”), which hosts and displays the seller’s profile webpage, is “capable of displaying multiple product listings.”

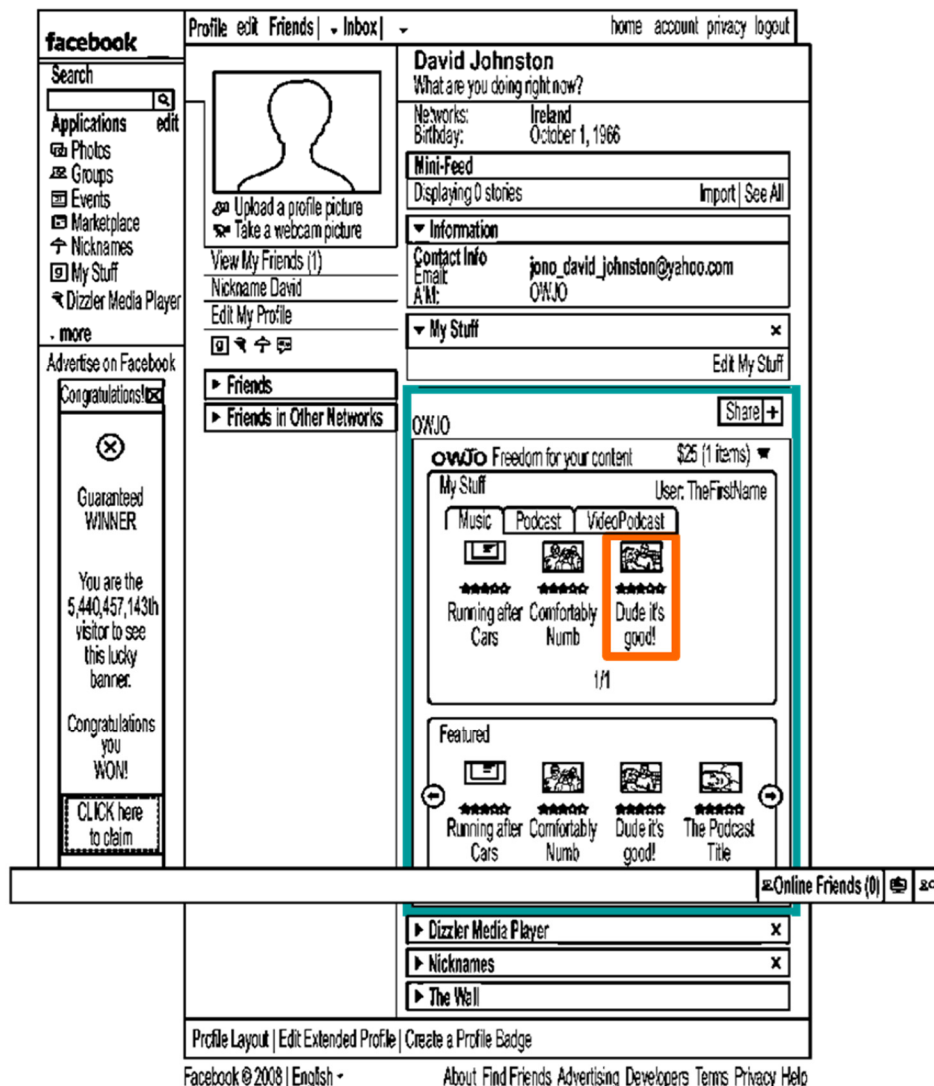


Exhibit 1005, Figure 4A (annotated)

160. Alternatively, Johnston's Figures 4A and 4B correspond to the Facebook profile webpage of one user—"David Johnston." A POSITA would have understood that Facebook hosts the profile webpages of multiple users who could embed Storefronts with product listings into their profiles. Under this scenario, Facebook is also "*capable of displaying multiple product listings.*"

161. Thus, Johnston in view of Dierks renders claim 12 obvious.

13. Dependent Claim 13

a) "The system of claim 11, wherein the product listing includes at least one of a title, a description, or a price."

162. Johnston in view of Dierks renders obvious claim 11, as I discussed above. Johnston discloses the additional limitation of claim 13 for at least the same reasons discussed for claim 2.

163. Thus, Johnston in view of Dierks renders claim 13 obvious.

14. Dependent Claim 14

a) "The system of claim 11, wherein the transaction processing service is a proprietary service operated by the application server."

164. Johnston in view of Dierks renders obvious claim 11, as I discussed above. Johnston discloses the additional limitation of claim 14 for at least the same reasons discussed for claim 10.

165. Thus, Johnston in view of Dierks renders claim 14 obvious.

B. Ground 2: Claims 1–14 are Obvious over Wiser, Dierks, and Chang

1. Independent Claim 1

a) 1[preamble]: “A system for generating an electronic commerce store, comprising:”

166. In my opinion, even if the preamble is limiting, Wiser discloses it. For example, Wiser discloses “[a] computer implemented online music distribution **system** [that] provides for the secure delivery of audio data and related media, including text and images, over a public communications network.” Exhibit 1003, Abstract; *see also id.*, 5:46–48 (“a system for the secure distribution of music and related media over a public telecommunications network, such as the Internet”). Wiser relates to “**a complete system for the online distribution of digital media**, such as digital audio, would provide online support for both the purchase and delivery phases.” *Id.*, 1:34–37. Wiser illustrates, in Figure 1, an embodiment of its online music distribution system—system 100.

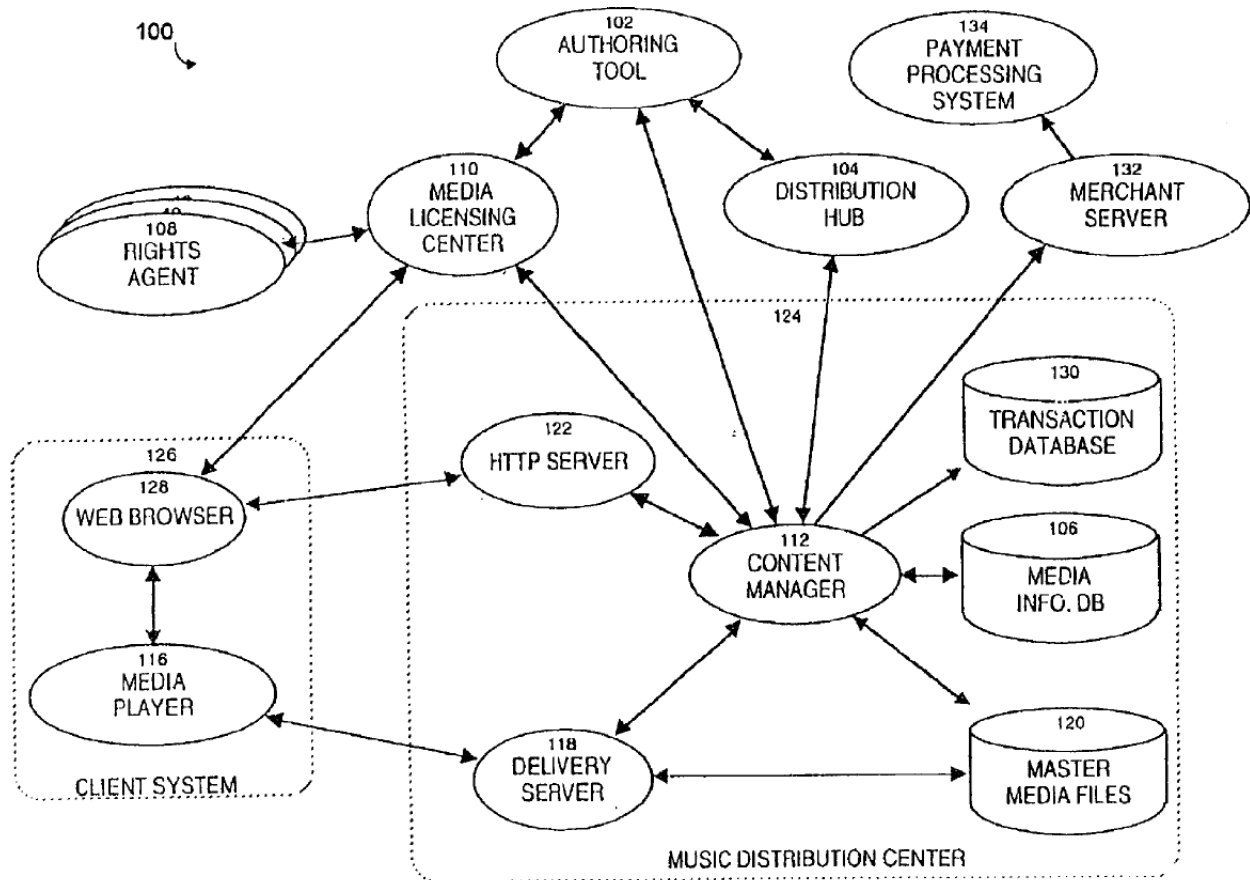


Exhibit 1003, Figure 1

167. Wiser further discloses that “[t]he online music distribution system also provides for previewing of audio data prior to purchase.” *Id.*, Abstract. Wiser shows an exemplary **web page** provided by its system in Figure 8 that allows a purchaser to preview and/or purchase a media data file. *See id.*, 14:39–42; 16:30–33. Wiser’s Figure 8 **web page** is “an electronic commerce store” and, based on Wiser’s teachings, a POSITA would have understood that the **web page** is generated by the online music distribution system. *See also* my discussion in Section IX.B.1.f) below.

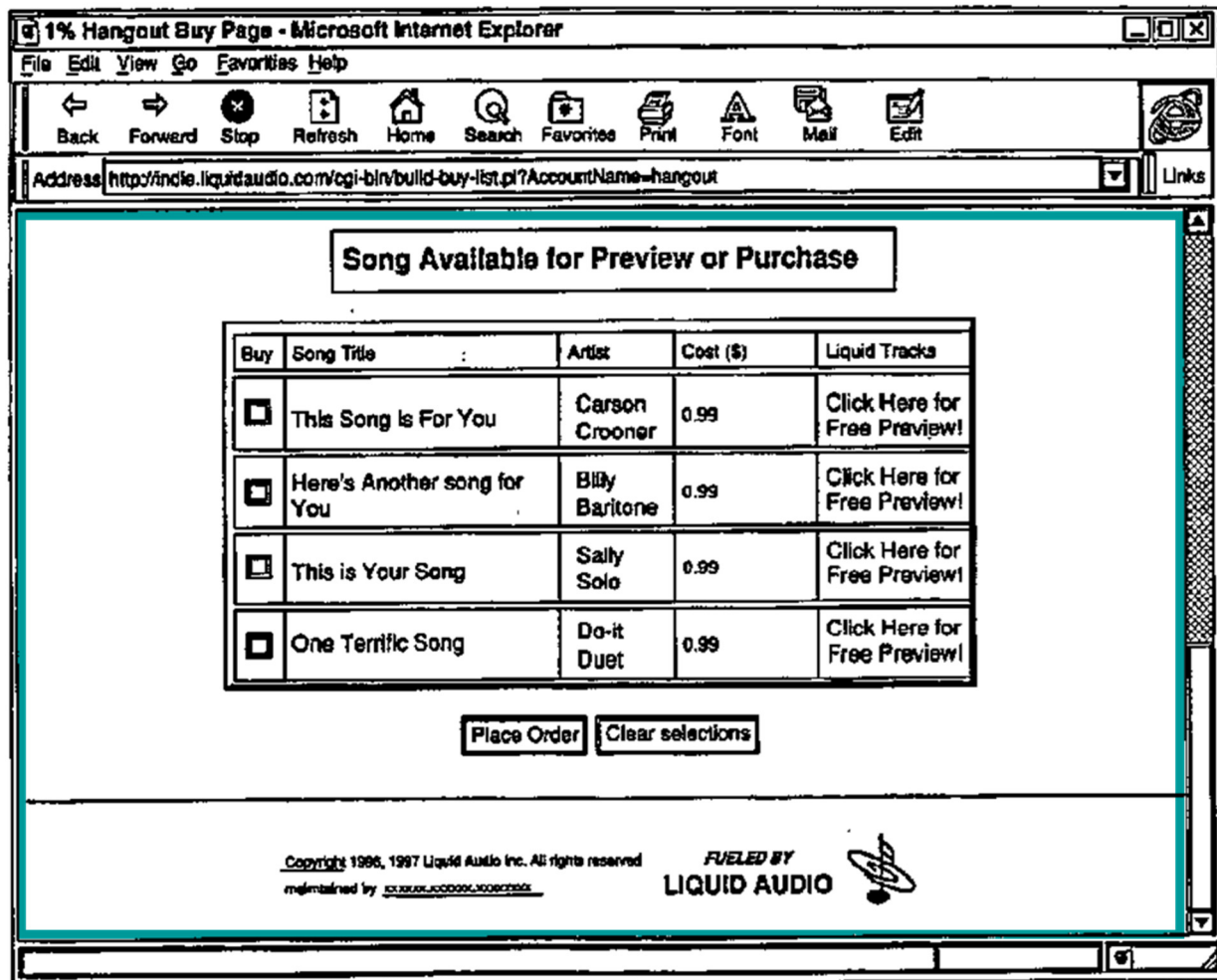


Exhibit 1003, Figure 8 (annotated)

168. Therefore, Wiser discloses the preamble of claim 1.

b) 1[a]: “an application server, the application server controlling a transaction processing service;”

169. Wiser discloses element 1[a].

170. **First**, Wiser discloses, in system 100, a music distribution center 124 that “includes a content manager 112, and at least one delivery server 118, an HTTP (HyperText Transfer Protocol) server 122.” Exhibit 1003, 5:51–54. The

combination of **content manager 112, delivery server 118, and HTTP server 122** collectively corresponds to a server. It was well-known before the priority date of the '995 patent that a server could be a cluster of servers and computing devices running computer programs for client systems. *See, e.g.*, Exhibit 1005, ¶ 115 (“The Server 110 ... may comprise a single server, a server farm or a cluster of one or more application servers or nodes.”); Exhibit 1006, ¶ 62 (“The merchant server or the third-party server may comprise a computer, server, or combination or network of computing devices such as a servers, computers, or processors, which in combination perform the operations described herein.”).

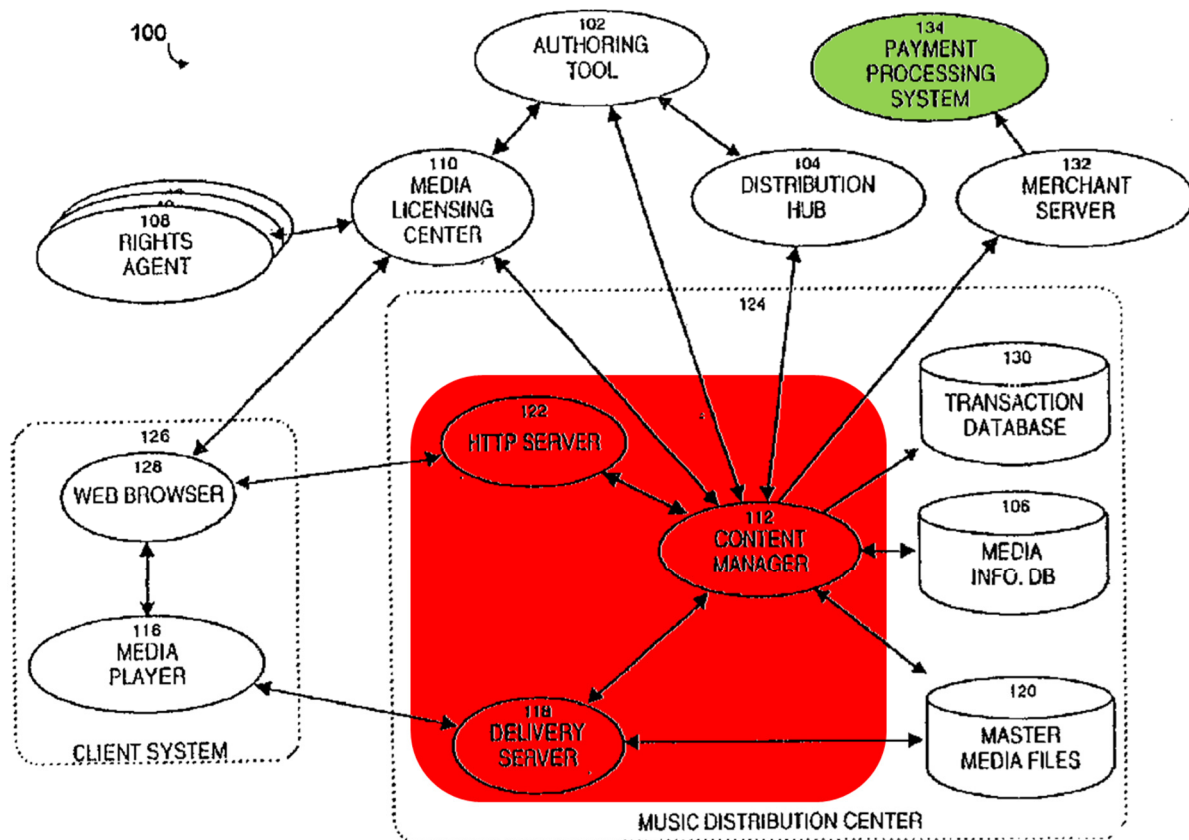


Exhibit 1003, Figure 1 (annotated)

171. *Second*, Wiser explains that the **content manager 112** handles the administration and management of (i.e., is “*configured to control*”) all purchases and other transactions. EX1003, 6:34–36. As Wiser illustrates in Figure 1, the **content manager 112** is in communication with merchant server 132, which in turn interfaces with **payment processing system 134**. *See id.*, 5:60–61. Wiser’s **payment processing system 134** corresponds to “*a transaction processing service.*” *See id.*, 11:12–15 (“The payment processing systems 134 are conventional payment authorization systems, such as credit card authorization systems or debit card payment authorization systems.”); *cf.* Exhibit 1001, 5:47–50 (“transaction processing service 112 for the provision of credit card, debit card or bank account information and completion of the purchase transaction”). Wiser’s **content manager 112** is like the ’995 patent’s application server 106 that “acts as a centralized compute processing and transaction management resource.” Exhibit 1001, 3:47–49.

172. Accordingly, the combination of **content manager 112, delivery server 118, and HTTP server 122** in Wiser is “*an application server configured to control a transaction processing service,*” as recited by element 1[a].

c) 1[b]: “a storage resource communicatively coupled to the application server, the storage resource storing an image and metadata related to a product;”

173. Wiser discloses element 1[b].

174. *First*, Wiser discloses a **master media file system 120** in communication with **content manager 112 and delivery server 118** (i.e., “the application server”). Exhibit 1003, Figure 1; *see also id.*, 14:56–57, 19:31–32.

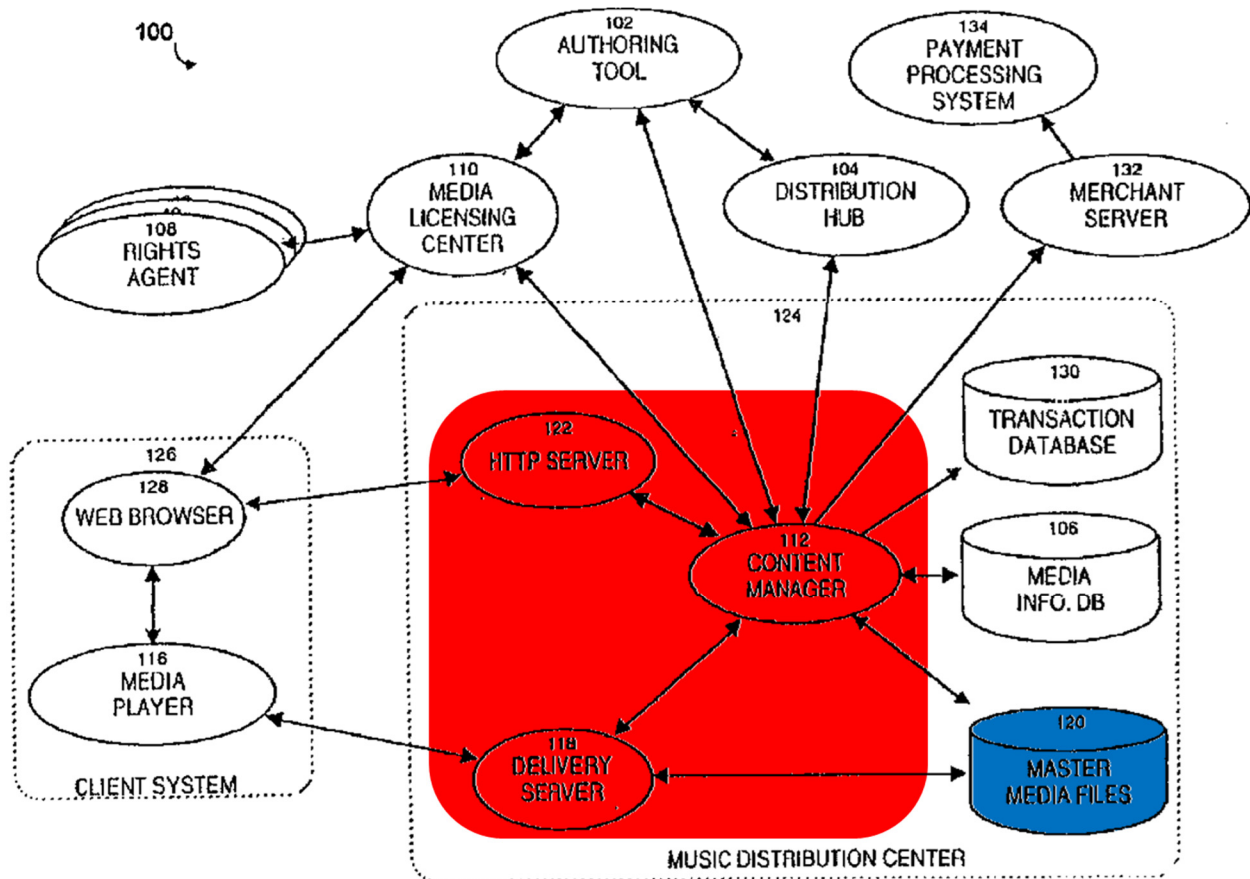


Exhibit 1003, Figure 1 (annotated)

175. *Second*, the **master media file system 120** stores at least one media data file 200. *Id.*, 6:52–53 (“The media data files 200 are stored in the master media file system 120.”), 10:51–56 (“In order to obtain media data files 200 for distribution, the authoring tools 102 are used by individual artists to create the audio data and associated media data in the media data files 200 to be delivered over the

network to the content manager 112 for storage in the master media data file system 120.”). Thus, the **master media file system 120** is a “*storage resource*.” Moreover, the media data file 200 is “*a product*.” Cf. Exhibit 1001, 11:25–27 (“each uploaded content file represents an item for promotion and distribution”; “[e]ach ‘item’ is made available as a ‘product’ for marketing purposes”); *id.*, 6:37–38 (“the items include individual songs, song releases, or entire song catalogues”).

176. *Third*, as Wiser illustrates in Figure 2, the **media data file 200** includes **media descriptive data 204**. Wiser explains that **media descriptive data 204** include “descriptive text, such as title, artist, lyrics, and liner notes” (i.e., “*metadata*”) and “promotional art image data, and cover art image data” (i.e., “*an image*”). Exhibit 1003, 6:61–64. Thus, the **master media file system 120** stores “*an image and metadata related to a product*.”

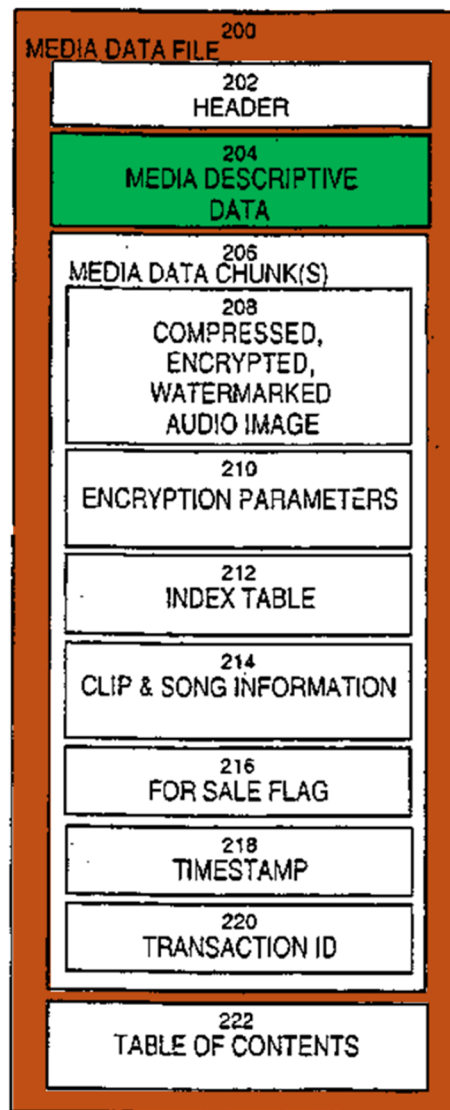


Exhibit 1003, Figure 2 (annotated)

177. Therefore, Wiser discloses element 1[b].

d) 1[c]: “a widget builder resource executing on the application server,”

178. Wiser in combination with Dierks renders obvious element 1[c].

(1) Wiser

179. Wiser discloses an exemplary “**preview listing**” in a **web page** in Figure 8, reproduced below alongside the ’995 patent’s Figure 10. Exhibit 1003, 16:30–33. Like the ’995 patent’s product listing, Wiser’s **preview listing** is displayed on or embedded into the **web page** displayed in a web browser—for example, web browser 128 of client system 126. *See id.* (“the user will be viewing in the Web browser 128 some form of menu, catalogue, index or other listing of music and media available for purchase, and may be similar in form to the preview listing of FIG. 8”).

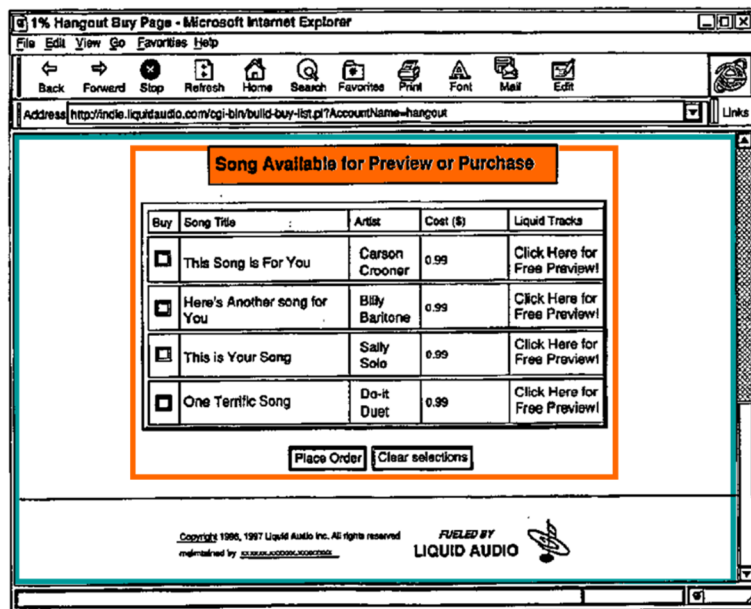


Exhibit 1003, Figure 8 (annotated)

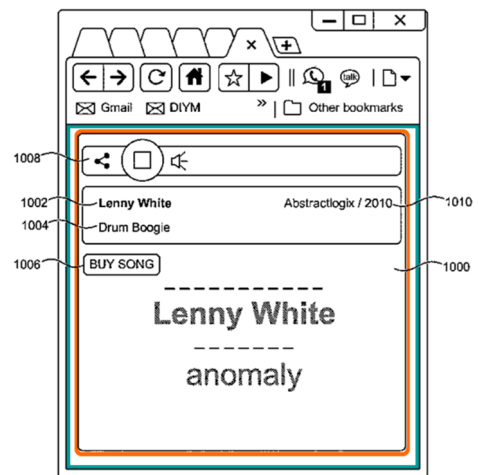


Exhibit 1001, Figure 10
(annotated)

180. Wiser is silent as to how its **preview listing** is generated. However, a POSITA would have found it obvious to implement Wiser’s **content manager 112**, **delivery server 118**, and/or **HTTP server 122** (i.e., “the application server”) to

include and execute a widget builder resource or the like to generate Wiser's **preview listing**. Wiser makes clear that the "rapid development of the Internet and the World Wide Web" has enabled online commerce. Exhibit 1003, 1:15–33. Wiser is particularly concerned with both the online purchase and delivery phases of distributing music created by a multitude of artists. *See id.*, 1:34–37 ("In contrast to conventional online purchase-off-line distribution systems, a complete system for the online distribution of digital media, such as digital audio, would provide online support for both the purchase and delivery phases."), 10:51–56 ("In order to obtain media data files 200 for distribution, the authoring tools 102 are used by individual artists to create the audio data and associated media data in the media data files 200 to be delivered over the network to the content manager 112 for storage in the master media data file system 120."). A POSITA would have known that it would be inefficient and impractical to write unique code to generate a preview listing for every media data file 200 or for a group of media data files by every artist. Instead, in view of Wiser's objectives, a POSITA would have found it obvious to use a template or a tool (i.e., "*a widget builder resource*") that can use information associated with every media data file 200 or group of media data files by an artist to readily generate a preview listing that has a predetermined format and structure. Doing so was well-known in the art prior to the priority date of the '995 patent, for example, as taught by Dierks and as I will discuss next.

(2) Dierks

181. Dierks discloses and illustrates in Figure 3 an e-commerce system 300 that includes an **HTML generator 312** that “is configured to receive parameters regarding items to be sold by vendors and to **generate code, such as HTML or XHTML, for the vendors to include in their web pages, to generate controls for making ordering and selection of the items easier for visitors to the sites.**” Exhibit 1004, ¶¶ 60, 62, Figure 3; *cf.* Exhibit 1001, 8:40–50 (“The buy buttons 310 are generated by content owners using the widget builder resource component 300 and linked to web widgets. The buy buttons 310 can be customized and distributed to online resources on the Internet, on mobile communication networks or on other computer networks for use in promoting the various groupings of multimedia content available from a content owner. Once a buy button 310 is clicked or selected by a consumer, it will activate its linked web widget and take the consumer to a checkout screen in the web widget.”).

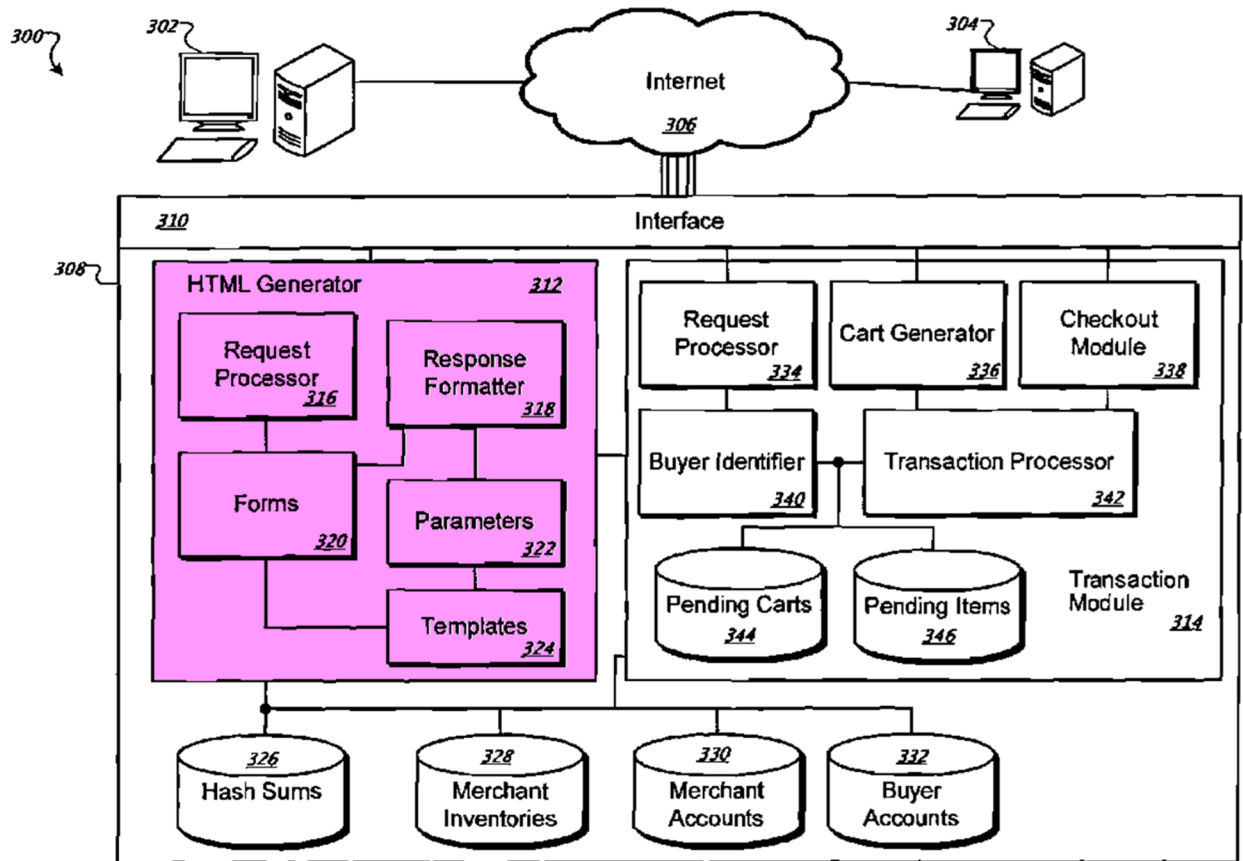


Exhibit 1004, Figure 3 (annotated)

182. Dierks's **HTML generator 312** includes a forms module 320 that serves a form that allows a requester to “provide information about an item the requester hopes to offer for sale, such as a description and price for the item.” EX1004, ¶ 63. The supplied information is stored in the parameters module 322. *Id.*, ¶ 64. Dierks explains that the **HTML generator 312** “may access generic HTML code from templates module 324” and “[b]y combining the template information with the supplied parameters in a conventional manner, the **HTML**

generator 312 may produce mark-up code for the user that the user can employ to provide payment processing services on their web site.” *Id.*

183. Dierks goes on to disclose an exemplary interface, illustrated in Figure 4, that “presents a user with an easy, step-by-step process for entering relevant information and obtaining the code.” Exhibit 1004, ¶ 71. Dierks explains that radio buttons 402 and 404 allow the user to choose the type of “purchasing button.” *Id.*, ¶ 71. Information about the item to be sold (e.g., name, description, price, etc.) provided by the user “may be coded along with the button, so as to be displayed on the vendor’s web page with the button.” *Id.*, ¶ 73; *see also id.*, ¶ 76 (“The payment processing system may then receive item information from the seller (box 506). Such information may include titles of items (e.g., ‘home made lefse’), item descriptions (e.g., ‘a scrumptious Scandinavian potato concoction like your Scandinavian grandmother made in Oslo’), and a price (e.g., ‘\$3 per package—10 sheets per package’). Other information about items for sale may also be provided.”), ¶ 77 (“At box 508, encoded item information is calculated, and at box 510, an item signature is created. Such actions may result in a code such as a hash code being created for the submitted parameters, so that the hash code can be checked later when shoppers submitting information from the seller’s web site try to make purchases.”). For example, the generated code is displayed in area 418 and may be “cut-and-pasted into the user’s web page.” *Id.*, ¶ 74.

Mozilla Firefox - MyCheckout 400

https://www.mycheckout.com/Merchant/8675309/CreateItemCode/

1. Choose a button type.

☒ Basic Button 402 ☐ Button with Options 404

Select this button type to offer different item options, such as sizes or colors.

2. Choose a button style.

My web page has a 406 background.

3. Describe your item.

Item Name: 408 Price: 410 USD

Item Description: 412

414

Insert this code for shopping cart:

```
<script type="text/javascript" src="https://www.mycheckout.com/shoppingcart.js?sellerId=8675309"></script><span id="mycheckout_status_bar"></span>
```

416

Insert this code for add to cart button:

```
<form action=https://www.mycheckout.com/Merchant/8675309/add onsubmit="return false;" method="post"><input type="hidden" name="item" value="PD94bVWgdmVyc2lvbj0iMS4wliBlbmNvZGluZz0iVVRGLTgiPz4NCjxJdGVtIHhtbG5zPSJodHRwOi8vY2hiY2tvdXQuZ29vZ2x1LmNvbS9zY2hibWEvMil+DQogIDxldWFudGI0eT4xPC9xdWFudGI0eT4NCiAgPHVuaXQ
```

418

Exhibit 1004, Figure 4

184. Dierks's Figure 1B, for example, illustrates a web browser displaying a **web page** with a **product listing**.

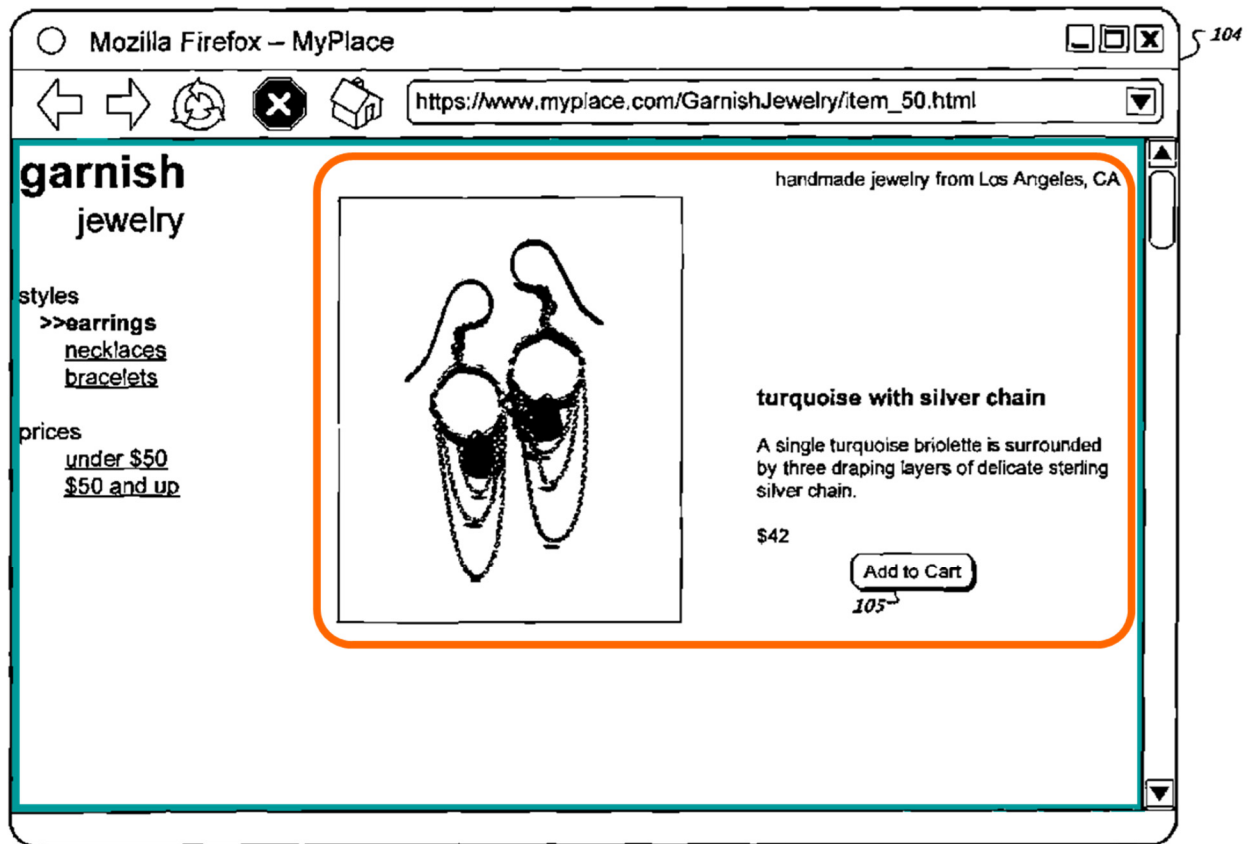


Exhibit 1004, Figure 1B (annotated)

185. Based on the above, Dierks's **HTML generator 312** is a "widget builder resource."

(3) *Motivation to Combine Wiser and Dierks*

186. A POSITA would have been motivated, according to and informed by Dierks's teachings, to implement Wiser's application server (i.e., content manager 112, delivery server 118, and/or HTTP server 122) to include and execute an HTML generator as taught by Dierks to generate Wiser's preview listing.

187. **First**, Wiser and Dierks are from the same field of endeavor. Both references are directed to electronic commerce. Wiser discloses a "computer

implemented online music distribution system provides for the secure delivery of audio data and related media, including text and images, over a public communications network.” Exhibit 1003, Abstract. Dierks discloses a “computer-implemented method [for] generating on a user terminal a merchant web site associated with a merchant with code for a merchant web page.” Exhibit 1004, Abstract. Thus, both references are directed to electronic commerce. Accordingly, a POSITA would have understood that Dierks’s teachings were directly applicable to Wiser’s system.

188. ***Second***, implementing Wiser’s application server to include and execute an HTML generator as taught by Dierks to generate Wiser’s preview listing would have yielded expected, predictable results. Wiser discloses a preview listing on a web site that includes information associated with an item for sale, price of the item, and a “Place Order” or “Buy It” button. *See* Exhibit 1003, Figure 8, 16:30–36; *see also* Section IX.B.1.d)(1). Dierks discloses an HTML generator that—based on information provided by a user about an item for sale, including its price and description—generates code for a product listing, including a “purchasing button,” that the user can cut-and-paste (i.e., embed) in the user’s web site. *See* Exhibit 1004, Figures 1B, 3, and 4, and ¶¶ 62–64, 71, 73, 74; *see also* Section IX.B.1.d)(2). A POSITA would have combined the elements taught by Wiser and Dierks—namely, an application server that provides a preview listing in a web site displayed in a web

browser as taught by Wiser and an HTML generator that generates code for a product listing, similar to Wiser's, as taught by Dierks—using known and routine computer programming principles to implement Wiser's application server to include and execute an HTML generator to generate Wiser's product listing. This result would have been readily predictable and recognized by a POSITA.

189. **Third**, a POSITA would have been motivated to combine the teachings of Wiser and Dierks because Dierks teaches the implementation details for generating a product listing that Wiser omits. Combining the two teachings would have entailed the simple substitution of one known element (an application server that displays a preview listing in Wiser) for another (a system with an HTML generator that generates code for a product listing as taught by Dierks) to obtain predictable results. This substitution would have been readily achievable by a POSITA via known and routine computer programming principles to implement Wiser's application server and would have provided the predictable result of including and executing an HTML generator to generate the code needed to display a product listing on a web site.

190. **Fourth**, a POSITA would have been motivated to combine the teachings of Wiser and Dierks because including and executing an HTML generator in an application server to generate a product listing would have been obvious to try. There were only two possible known and predictable choices for generating a

product listing in an application server—(1) write unique code to display a listing for every piece or group of music for sale and (2) have a tool such as Dierks’s HTML generator that gathers unique information associated with a piece or group of music for sale but uses mostly the same code to generate a product listing. A POSITA could have pursued these two potential solutions with a reasonable expectation of success because the analysis would have simply involved trying each one of these two solutions. Thus, it would have been obvious to a POSITA to have selected the inclusion and execution of an HTML generator as taught by Dierks in Wiser’s system.

191. *Fifth*, including and executing Dierks’s HTML generator in Wiser’s application server would have been readily implemented by simply augmenting Wiser’s application server to include and execute an HTML generator that generates code to list music for sale on a web site. A POSITA could have used basic computer programming principles to implement Wiser’s application server to include and execute Dierks’s HTML generator. This implementation would have been well within the knowledge and skillset of a POSITA.

(4) Reasonable Expectation of Success in Combining Wiser and Dierks

192. A POSITA would also have had a reasonable expectation of success. The combination of Wiser and Dierks represents a straightforward implementation

of an HTML generator in Wiser’s application server to generate code for a product listing, which a POSITA would have been familiar with and been able to implement. The reasons a POSITA would have expected success parallel those that provide motivation for this combination—including because both Wiser and Dierks are directed to electronic commerce and both disclose similar product listings. *See, e.g.*, Exhibit 1003, Abstract, Figure 8; Exhibit 1004, Abstract, Figure 1B; *see also* Section IX.B.1.d)(3).

193. Accordingly, Wiser in view of Dierks renders obvious element 1[c] and a POSITA would have been motivated to combine the teachings of these references and would have had a reasonable expectation of success. For these same reasons, a POSITA would have been motivated to combine the teachings of these references for all subsequently discussed claims and limitations in this ground.

***e) 1[d]: “the widget builder resource configured to:
receive the image and metadata from the storage resource;
and”***

194. Wiser in combination with Dierks renders obvious element 1[d].

195. As I discussed for element 1[c], Wiser in combination with Dierks renders obvious “*a widget builder resource executing on the application server.*” In Section IX.B.1.d)(2), I explained how Dierks’s HTML generator 312 (i.e., “*the widget builder resource*”) is configured to generate mark-up code for vendors to include in web pages based on information received about items to be sold. *See*

Exhibit 1004, ¶¶ 62, 73, 76, 77. Dierks gives an example of the generated code in area 418 of its Figure 4. *See id.*, ¶¶ 71, 74; *see also id.* ¶ 38.

Mozilla Firefox - MyCheckout 400

https://www.mycheckout.com/Merchant/8675309/CreateItemCode/

1. Choose a button type.

☒ Basic Button 402 ☐ Button with Options 404

402 Add to Cart

404 Select this button type to offer different item options, such as sizes or colors.

\$21.00 – T-Shirt (small) Add to Cart

2. Choose a button style.

My web page has a White 406 background.

3. Describe your item.

Item Name: turquoise earrings 408 Price: 50.00 410 USD

Item Description: A single turquoise briolette is surrounded by three draping layers of delicate sterling silver chain. 412

414 Create Button Code

Insert this code for shopping cart:

```
<script type="text/javascript" src="https://www.mycheckout.com/shoppingcart.js?sellerId=8675309"></script><span id="mycheckout_status_bar"></span>
```

 416

Insert this code for add to cart button:

```
<form action="https://www.mycheckout.com/Merchant/8675309/add" onsubmit="return false;" method="post"><input type="hidden" name="item" value="PD94bVWgdmVyc2lvbj0iMS4wliBlbmNvZGluZz0iVVRGLTgiPz4NCjxJdGVtIHhtbG5zPSJodHRwOi8vY2hiY2tvdXQuZ29vZ2xlLnNvbS9zY2hibWEvMil+DQogIDxudWFudGloT4xPC9xdWFudGloT4NCiAgPHVuaXQ
```

 418

Exhibit 1004, Figure 4

196. The generated code in area 418 includes a hash code (i.e., the string of characters equated to “value”). *See id.*, ¶ 39 (“This hash code may have been generated when the underlying mark-up code for the ‘add to cart’ button 105 was generated, and may represent a hash function applied to that mark-up code.”). A POSITA would have understood that the hash code corresponds to information corresponding to the item to be sold—for example, its price, description, etc.—that

may be displayed on a vendor's web page and that also allows the system to ascertain that the correct item is being bought at the correct price when processing a transaction. *See id.*, ¶ 73 (“Such entries may be coded along with the button, so as to be displayed on the vendor's web page with the button, and may also be stored by a payment processing system for use in processing transactions or as checks on information submitted when a purchaser selects the button.”), ¶ 39 (“The hash code may then be checked by the payment server 114 in a conventional manner to ensure that the mark-up code has not been changed since it was provided by the system, such as by a purchaser attempting to lower the price for a good so that they can avoid paying full price.”); *see also id.*, ¶ 77. A POSITA would have understood that the hash sums database 326 would typically contain not only the hash code values, but also the information (or a pointer to the information) that was used to generate the hash code. *See id.*, Figure 3. Hence, when the hash sums database 326 is queried based upon a hash code value, a matching record would provide the payment processing system with relevant information regarding the item for sale. The more information of the item the hash code corresponds to, the harder it is for a purchaser to circumvent the system.

197. As I discussed for element 1[b], Wiser discloses that **master media file system 120** (i.e., “*the storage resource*”) is in communication with **content manager 112 and delivery server 118** (i.e., “*the application server*”) and stores

media data file 200, which includes **media descriptive data 204**. **Media descriptive data 204** include “descriptive text, such as title, artist, lyrics, and liner notes” (i.e., “*metadata*”) and “promotional art image data, and cover art image data” (i.e., “*the image*”). Exhibit 1003, 6:61–64. In an aspect of its invention, Wiser discloses that “cover art” is available for preview. *Id.*, 3:63–65 (“In addition, descriptive information, such as cover art, lyrics, credits and the like, is also available for previewing.”). Wiser’s preview listing in Figure 8 provides links to preview audio but Figure 8 itself is part of the overall preview process (e.g., by providing the name of the song, artist, and price). *See id.*, 14:39–42 (“Previewing begins with the user viewing a Web page in the Web browser 128 that has a link to a preview of a desired media data file 200. FIG. 8 illustrates an exemplary Web page for selecting a preview.”). Thus, for Wiser’s system as configured by Dierks’s teachings (the “Wiser-Dierks system”) to generate a preview listing like in Wiser’s Figure 8 that further displays a cover art for preview, a POSITA would have found it obvious to configure the HTML generator to receive and use not only Wiser’s “descriptive text, such as title, artist, lyrics, and liner notes” (i.e., “*metadata*”), as taught by Dierks, but also Wiser’s “promotional art image data, and cover art image data” (i.e., “*the image*”) from **media descriptive data 204** stored in **master media file system 120** (i.e., “*the storage resource*”). The hash code in the code generated by the HTML generator would thus correspond to both “*the metadata*” and “*the image*” to include

in the product listing. Such a configuration would simplify the process of creating a product listing for the seller and require less input from the seller. Accordingly, a POSITA would have configured the widget builder resource to “*receive the image and metadata from the storage resource.*”

198. Therefore, Wiser in combination with Dierks renders obvious element 1[d].

f) 1[e]: “[the widget builder resource configured to:] generate a product listing, wherein the product listing includes the image, the metadata, and a clickable link, wherein the clickable link is a buy button and includes a preview of the product, and”

199. Wiser in combination with Dierks renders obvious element 1[e].

200. **First**, as I discussed for element 1[c], a POSITA would have been motivated to combine and would have had a reasonable expectation of success in combining the teachings of Wiser and Dierks to implement Wiser’s application server to include and execute an HTML generator (i.e., “*the widget builder resource*”) as taught by Dierks to generate Wiser’s preview listing.

201. **Second**, Wiser’s **preview listing** is a “*a product listing.*” Wiser discloses that “the user will be viewing in the Web browser 128 some form of menu, catalogue, index or other **listing of music and media available for purchase**, and may be similar in form to the **preview listing** of Figure 8.” Exhibit 1003, 16:30–33; cf. Exhibit 1001, 11:25–27 (“each uploaded content file represents an item for

promotion and distribution”; “[e]ach ‘item’ is made available as a ‘product’ for marketing purposes”); *id.*, 6:37–38 (“the items include individual songs, song releases, or entire song catalogues”).

202. *Third*, similar to the ’995 patent’s product listing in Figure 10 (reproduced below alongside Wiser’s Figure 8), Wiser’s **preview listing** includes the “**Song Title**” and “**Artist**,” a “*preview of the product*,” and a “**Place Order**” button. Exhibit 1003, Figure 8.

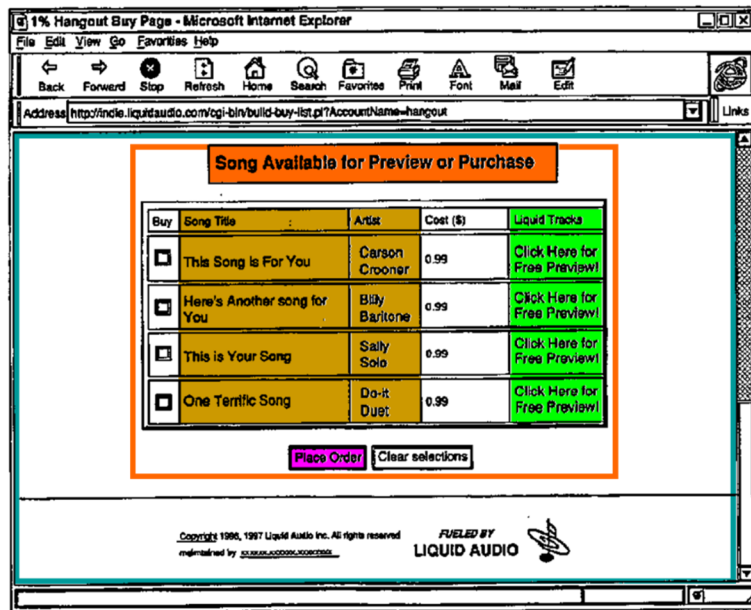


Exhibit 1003, Figure 8 (annotated)

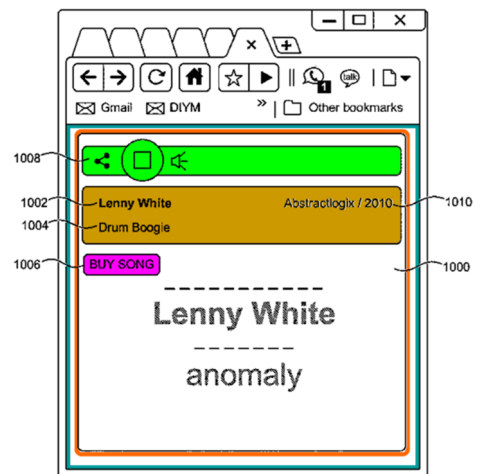


Exhibit 1001, Figure 10
(annotated)

203. As I explained for element 1[b], a media data file 200 stored in master media file system 120 includes media descriptive data 204, which in turn include “descriptive text, such as **title**, **artist**, lyrics, and liner notes” (i.e., “*the metadata*”).

Exhibit 1003, 6:62–63. Thus, the “**Song Title**” and “**Artist**” in Wiser’s **preview listing** corresponds to “*the metadata.*”

204. Wiser makes clear that “FIG. 8 illustrates an exemplary Web page for selecting a **preview.**” *See* Exhibit 1003, 14:41–42. Wiser explains that “[p]reviewing begins with the user viewing a Web page in the Web browser 128 that has a link to **a preview of a desired media data file 200.**” *Id.*, 14:39–41. Accordingly, Wiser’s **preview listing** includes “*a preview of the product.*”

205. Further, Wiser’s “**Place Order**” button is “*a buy button.*” Wiser discloses that “[f]rom the user’s Web browser 128 a purchase request for a specific song is sent 902 to the HTTP server 122, for example by the user clicking on a “**Buy It**” button.” Exhibit 1003, 16:33–36. A POSITA would have understood that the “**Place Order**” button is also “*a clickable link.*” *See, e.g.,* Exhibit 1022, 5:26–28 (“The web page of the reseller includes a hyperlink, which is indicated by a graphic display, such a button, highlighted text, or the like.”).

206. **Fourth**, as I discussed for element 1[d], Wiser discloses making cover art (i.e., “*the image*”) available for preview (*see* Exhibit 1003, 14:39–42) and a POSITA would have configured the widget builder resource in the Wiser-Dierks system to receive the image from the storage resource. Wiser is silent regarding where and when the cover art would be displayed. However, displaying the cover art or a thumbnail of the cover art in Wiser’s **preview listing** in Figure 8, just like

Dierks displays an image of the product for sale in its Figure 1B, would have been an obvious option to a POSITA. A POSITA would have been motivated to do so because displaying the image of the product for sale would make the product readily recognizable to the buyer. Further, it was well-known to display the cover image for a music track in a product listing. *See, e.g.*, Exhibit 1005, Figure 4B; *see also* Section IX.A.1.f). Implementing the widget builder resource in the Wiser-Dierks system to display the cover art or a thumbnail of the cover art in Wiser's **preview listing** would have been well within the knowledge and skillset of a POSITA, and would have been effected using basic computer programming principles. *See also* Sections IX.B.1.d)(3) and IX.B.1.d)(4).

207. Therefore, Wiser in combination with Dierks renders obvious element 1[e].

g) 1[f]: “wherein the application server is configured to embed the product listing into a web site and into a social media network, and”

208. Wiser in combination with Dierks and Chang renders obvious element 1[f].

209. As I discussed above, a POSITA would have been motivated to combine and would have had a reasonable expectation of success in combining the teachings of Wiser and Dierks to implement Wiser's application server to include and execute an HTML generator as taught by Dierks to generate Wiser's preview

listing. Dierks makes clear that the code generated by the HTML generator can be “cut-and-pasted into the user’s web page.” Exhibit 1004, ¶ 74. In other words, “*the application server is configured to embed the product listing into a web site.*”

210. Wisner and Dierks are silent as to whether the web page hosting their respective product listing belongs to a social media network or whether their product listing gets embedded into a social media network. However, prior to the priority date of the ’995 patent, a POSITA would have found it obvious to embed a product listing generated by the Wisner-Dierks system into a social media network. For example, Chang discloses systems and methods that “facilitate distributed commerce and/or viral marketing through the use of widgets embedded in web pages [that] may be associated with social networking web sites or web sites transacting commerce.” Exhibit 1008, ¶ 30. Chang’s “widgets may be used to sell good[s] or services” (i.e., “*product listing*”). Chang explains that, upon a user 101’s request, a commerce server 110 (i.e., “*application server*”) “may distribute the widget code included within the item page with widget code 201 to a plurality of social networking servers.” *Id.* ¶ 41. The social networking servers correspond to “a plurality of social networking sites” with “trade names including FACEBOOK™, MYSPACE™, LINKED-IN™, or some other suitable social networking site.” *Id.* Embedding the Wisner-Dierks system’s product listing into a social media network as taught by Chang would have expanded the pool of buyers.

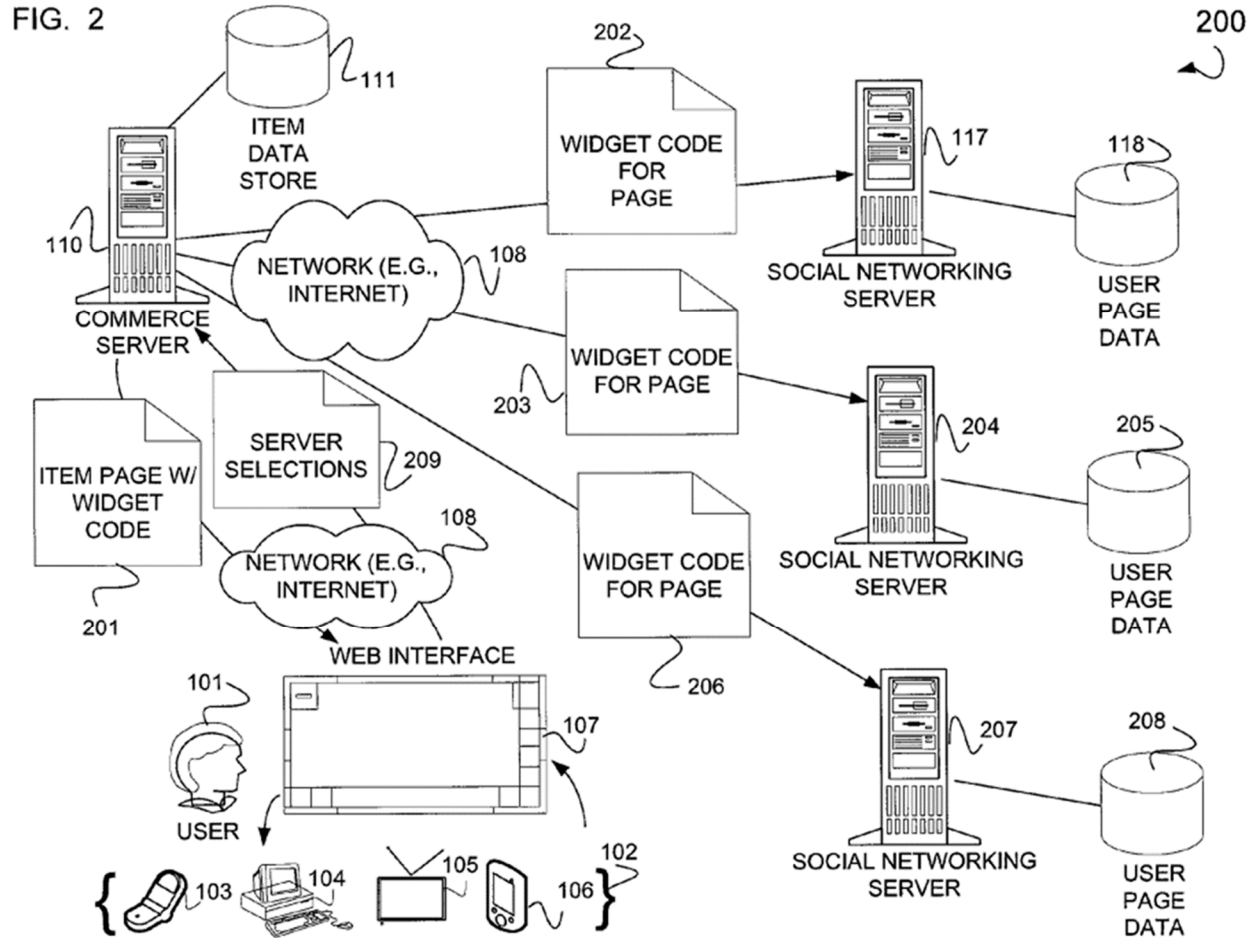


Exhibit 1008, Figure 2

(1) Motivation to Combine Wiser, Dierks, and Chang

211. A POSITA would have motivated, according to and informed by Chang’s teachings, to configure the Wiser-Dierks system to embed the Wiser-Dierks system’s product listing into a social media network.

212. **First**, Wiser, Dierks, and Chang are from the same field of endeavor. These references are directed to electronic commerce. Wiser discloses a “computer implemented online music distribution system provides for the secure delivery of audio data and related media, including text and images, over a public

communications network.” Exhibit 1003, Abstract. Dierks discloses a “computer-implemented method [for] generating on a user terminal a merchant web site associated with a merchant with code for a merchant web page.” Exhibit 1004, Abstract. Chang discloses “the use of widgets to implement a distributed commerce and viral marketing regime.” Exhibit 1008, ¶ 2. Thus, all three references are directed to electronic commerce. Accordingly, a POSITA would have understood that Chang’s teachings were directly applicable to the Wiser-Dierks system.

213. **Second**, implementing the Wiser-Dierks system to embed a product listing into a social media network as taught by Chang would have yielded expected, predictable results. The Wiser-Dierks system generates a product listing. *See* Section IX.B.1.f). Chang discloses embedding a widget code for a product listing into a social media network. Exhibit 1008, Abstract, ¶ 41, Figure 2. A POSITA would have combined the elements of the Wiser-Dierks system and the elements taught by Chang—namely, the Wiser-Dierks system’s product listing and Chang’s embedding of a product listing into a social media network—using known and routine computer programming principles to implement the Wiser-Dierks system to embed its product listing into a social media network. This result would have been readily predictable and recognized by a POSITA.

214. **Third**, configuring the Wiser-Dierks system to distribute a product listing to a social media network as taught by Chang would have been readily

implemented by simply augmenting the code, which distributes the Wiser-Dierks system's product listing to a web site, to also distribute to a site or application of a social media network. A POSITA could have performed this implementation using basic computer programming principles. This implementation would have been well within the knowledge and skillset of a POSITA.

(2) Reasonable Expectation of Success in Combining Wiser, Dierks, and Chang

215. A POSITA would also have had a reasonable expectation of success. The combination of the Wiser-Dierks system and Chang represents a straightforward implementation of distributing a product listing to additional sites or applications of a social media network. The reasons a POSITA would have expected success parallel those that provide motivation for this combination—including because Wiser, Dierks, and Chang are directed to electronic commerce and all relate to product listings. *See, e.g.*, Exhibit 1003, Abstract, Figure 8; Exhibit 1004, Abstract, Figure 1B; Exhibit 1008, Abstract, ¶ 2; *see also* Section IX.B.1.g)(1).

216. Accordingly, Wiser in view of Dierks and Chang renders obvious element 1[f] and a POSITA would have been motivated to combine the teachings of these references and would have had a reasonable expectation of success. For these same reasons, a POSITA would have been motivated to combine the teachings of these references for all subsequently discussed claims and limitations in this ground.

h) 1[g]: “wherein when the application server receives an indication that the clickable link has been activated by a client device in Internet communication with the web site, the application server provides a checkout screen to the client device,”

217. Wiser discloses or, in combination with Dierks, renders obvious element 1[g]. For example, Wiser discloses that “[f]rom the user’s Web browser 128 a purchase request for a specific song is sent 902 to the **HTTP server 122**, for example by the user clicking on a ‘Buy It’ button.” Exhibit 1003, 16:33–36. The web browser 128 belongs to a **client system 126** (i.e., “a client device”). See *id.*, 5:61–62 (“Client systems 126 include a media player 116 and a Web browser 128.”), 6:1-5, Figure 1 (illustrating client system 126 including web browser 128 and media player 116). Wiser’s Figure 9a shows that the **HTTP server 122** (i.e., “the application server”) receives the purchase request (resulting from the user clicking on a “Buy It” button) from the **web browser 128 of client system 126** at step 902 (arrow highlighted in yellow below) (i.e., “an indication that the clickable link has been activated by a client device in Internet communication with the web site”).

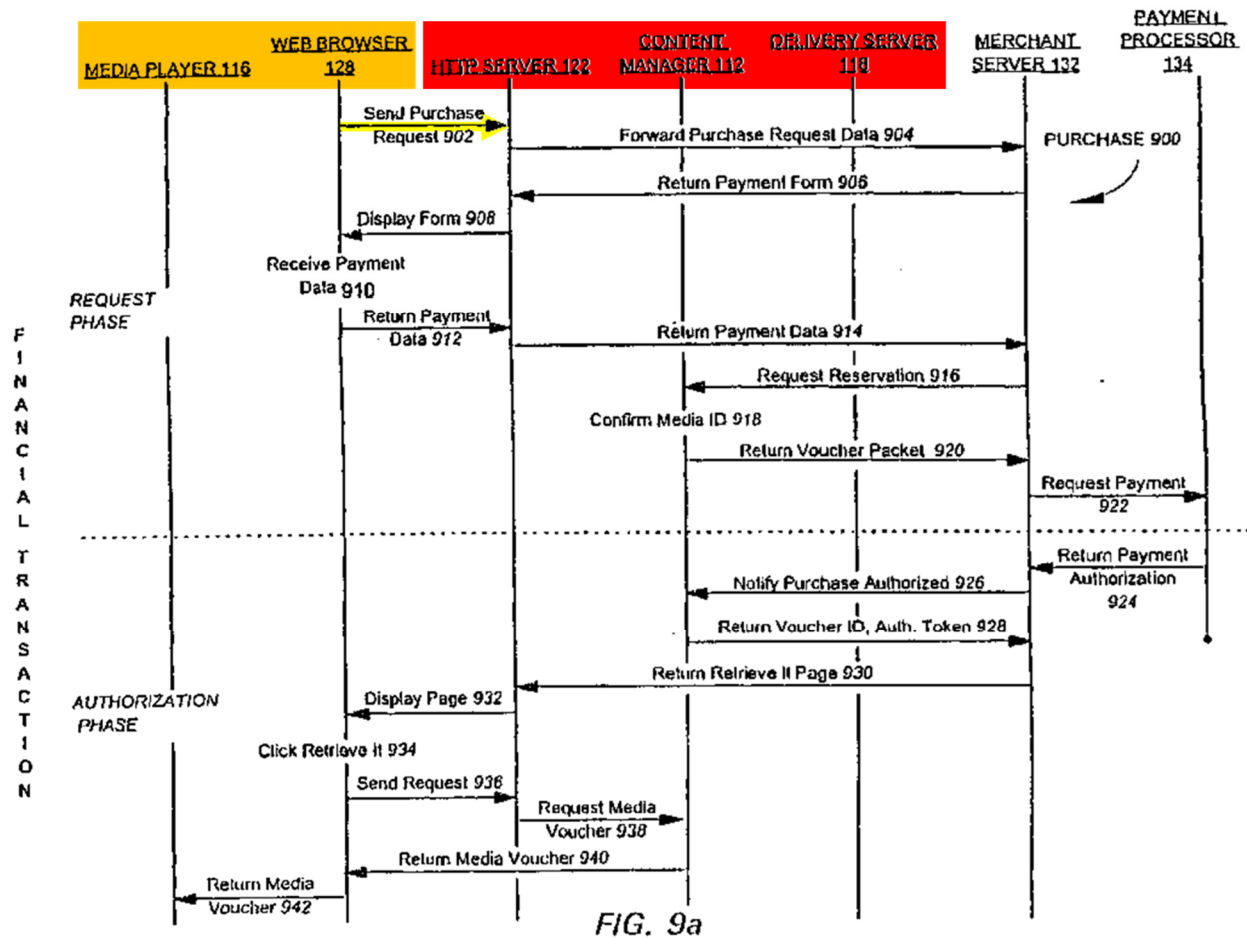


Exhibit 1003, Figure 9a (annotated)

218. Wiser further discloses that, after the **HTTP server 122** forwards the purchase request to the merchant server 132 at step 904, it receives a payment request form from the merchant server 132 at step 906. Exhibit 1003, 16:43–50. Subsequently, the **HTTP server 122** provides the payment request form for display at the **web browser 128 of client system 126** (i.e., “the client device”) at step 908 (arrow highlighted in yellow below). *Id.*, 16:50–51. Wiser further explains that “[t]he user completes 910 the form, which preferably requests the user’s name, credit

card number, and expiration date.” *Id.*, 16:52–54. Thus, Wiser’s payment request form corresponds to “a checkout screen.”

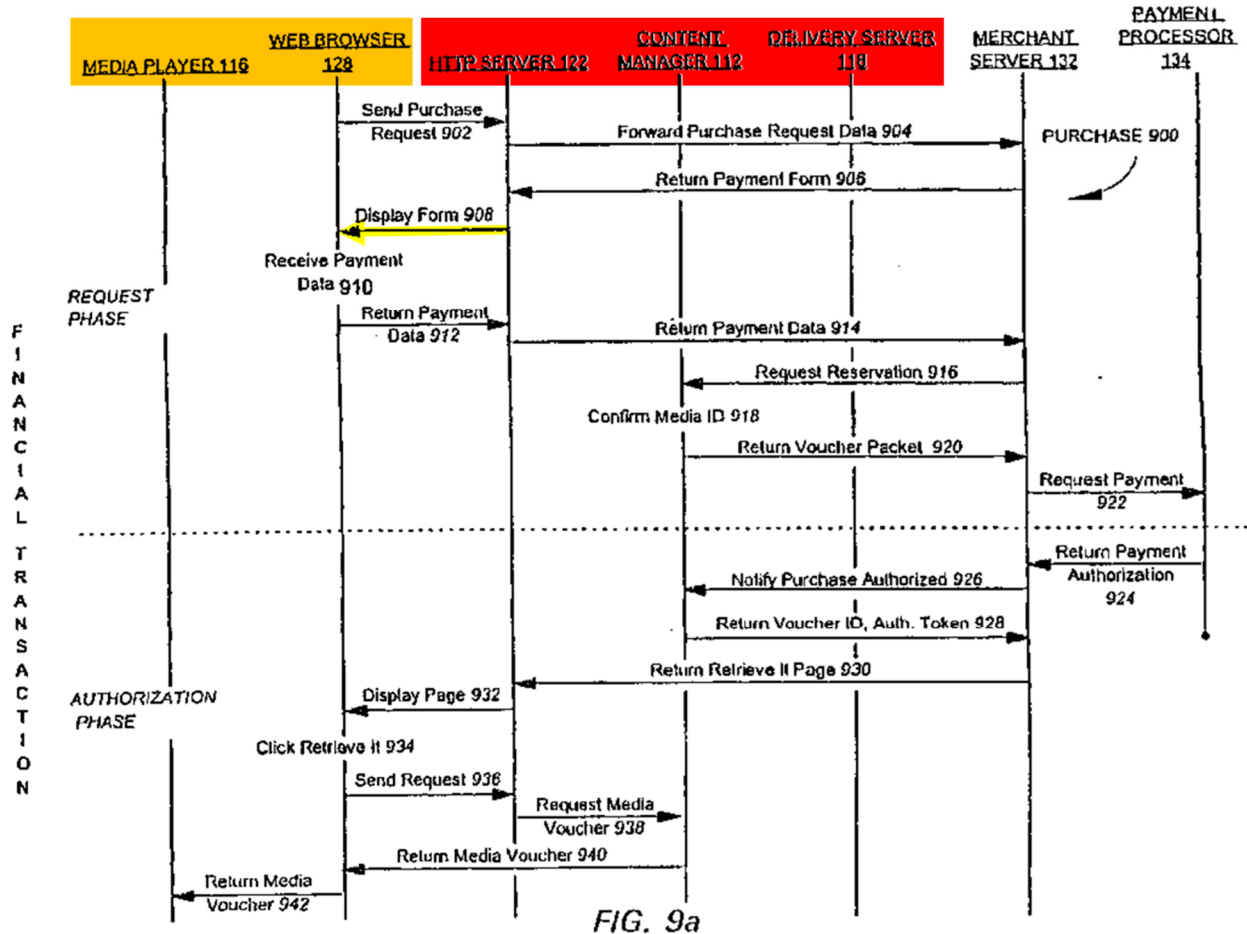


Exhibit 1003, Figure 9a (annotated)

219. Dierks illustrates an exemplary checkout screen in a web browser in Figure 1E. As can be seen, Dierks’s checkout screen includes fields for a user to enter the user’s name, credit card number, and expiration date, just like in Wiser’s payment request form. A POSITA would have found it obvious to implement Wiser’s disclosed payment request form to be displayed in a similar fashion to

Dierks's checkout screen. The POSITA would have been motivated to combine the teachings of Wiser and Dierks for the reasons I discussed in Section IX.B.1.d)(3) and would have had a reasonable expectation of success in combining the teachings of Wiser and Dierks as I explained in Section IX.B.1.d)(4).

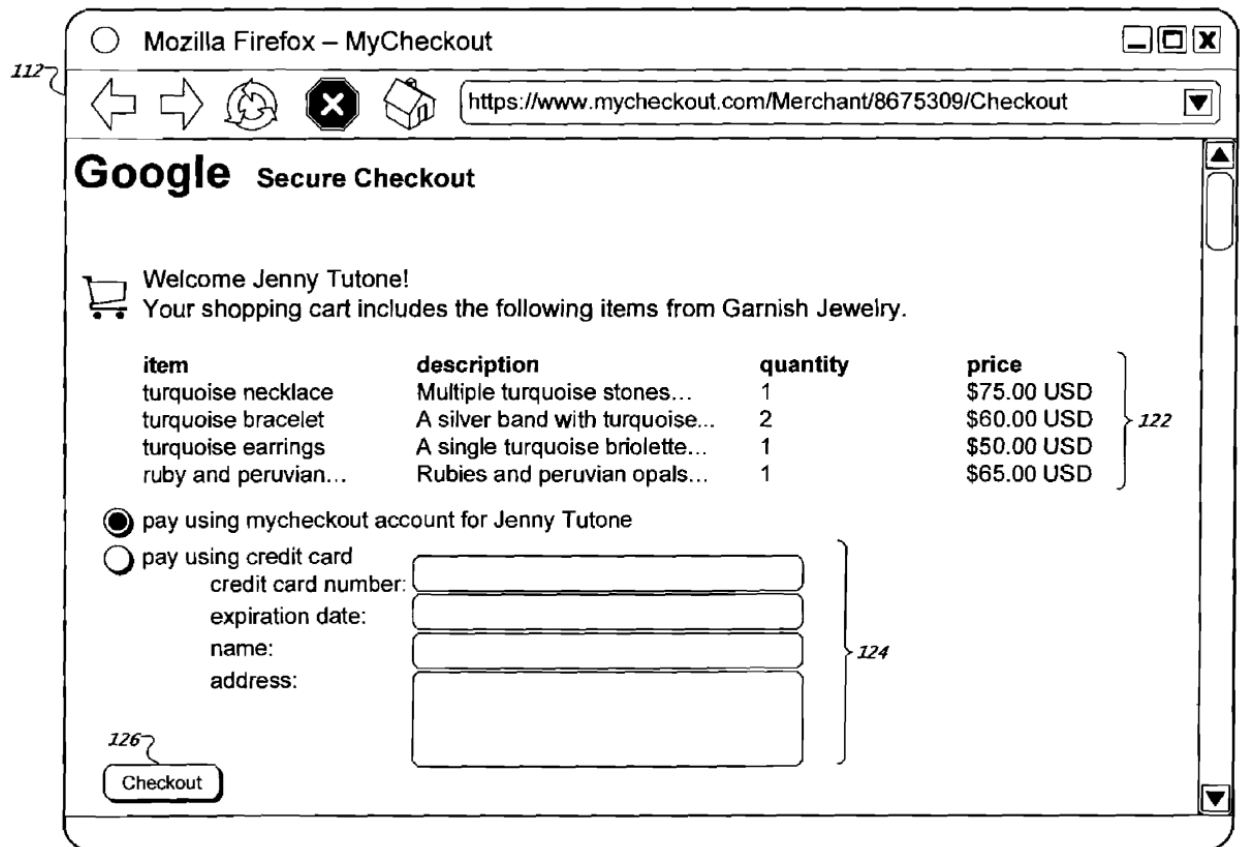


Exhibit 1004, Figure 1E

220. Therefore, Wiser discloses or, in combination with Dierks, renders obvious element 1[g].

i) 1[h]: “wherein the transaction processing service processes a payment transaction for a purchase of the product initiated from the checkout screen using the client device.”

221. Wiser discloses element 1[h].

222. **First**, Wiser discloses that data entered by the user in the payment request form are “transmitted back 912 to the **HTTP server 122** which passes 914 it to the merchant server 132.” Exhibit 1003, 16:52–61. Thereafter, “the merchant server 132 connects to a payment processor gateway 134 to request payment 922 by verifying the availability of funds and receiving 924 payment authorization.” *Id.*, 18:3–6; *see also id.*, 11:12–15 (“The payment processing systems 134 are conventional payment authorization systems, such as credit card authorization systems or debit card payment authorization systems.”). In other words, the **payment processing system 134** (i.e., “*the transaction processing service*”) “*processes a payment for a purchase of the product.*”

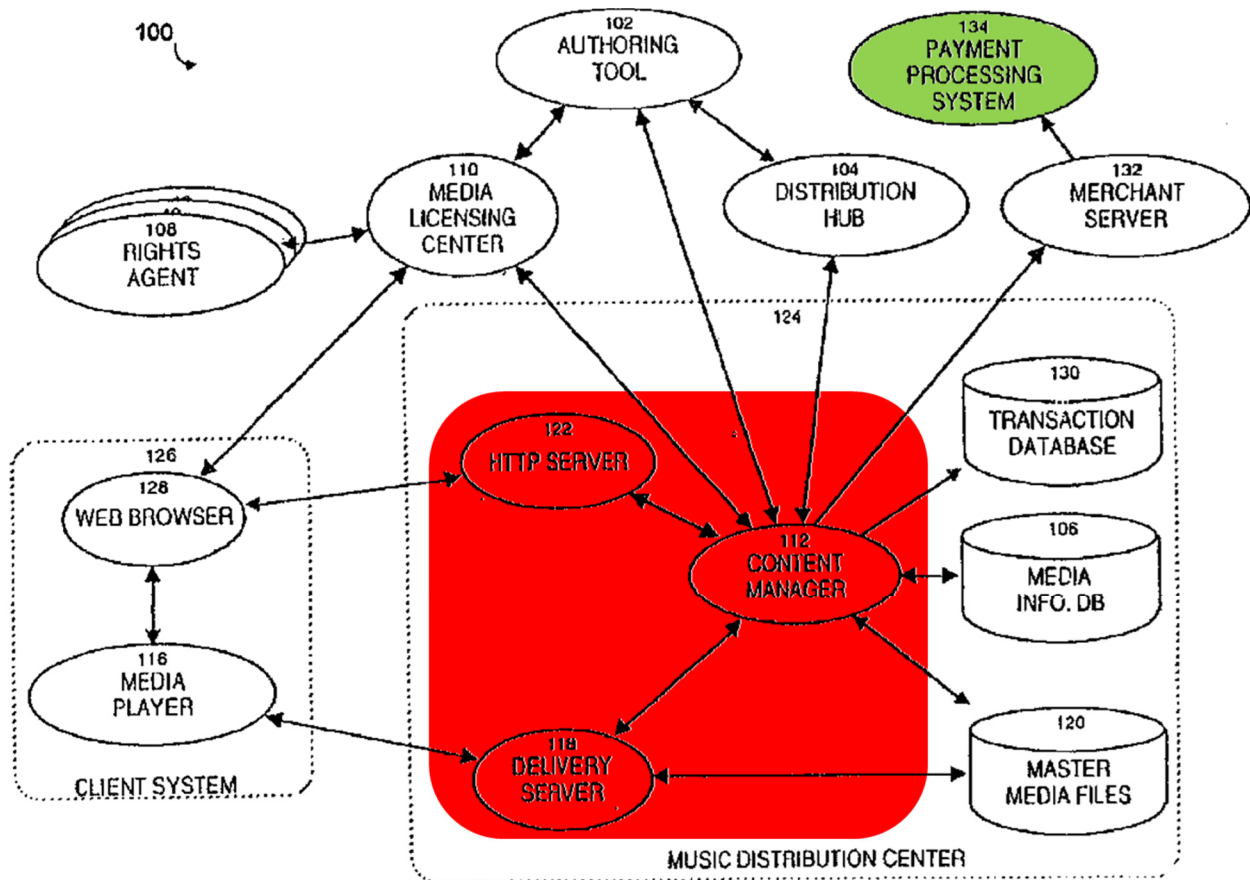


Exhibit 1003, Figure 1 (annotated)

223. *Second*, the “purchase [is] initiated from the checkout screen using the client device” when the user submits the data requested (e.g., user’s name, credit card number, and expiration date) in the payment request form displayed on the **web browser 128 of client system 126**, as I discussed for element 1[g].

224. Therefore, Wiser discloses element 1[h].

225. For the above reasons, Wiser in view of Dierks and Chang renders obvious claim 1.

2. Dependent Claim 2

a) *“The system of claim 1, wherein the metadata includes at least one of a title, a description, or a price.”*

226. Wiser in view of Dierks and Chang renders obvious claim 1, as I discussed above. Wiser discloses the additional limitation of claim 2. For example, Wiser discloses that **media descriptive data 204** “include **descriptive text**, such as **title**, artist, lyrics, and liner notes” (i.e., “*the metadata*”). Exhibit 1003, 6:62–63; *see also* Section IX.B.1.c).

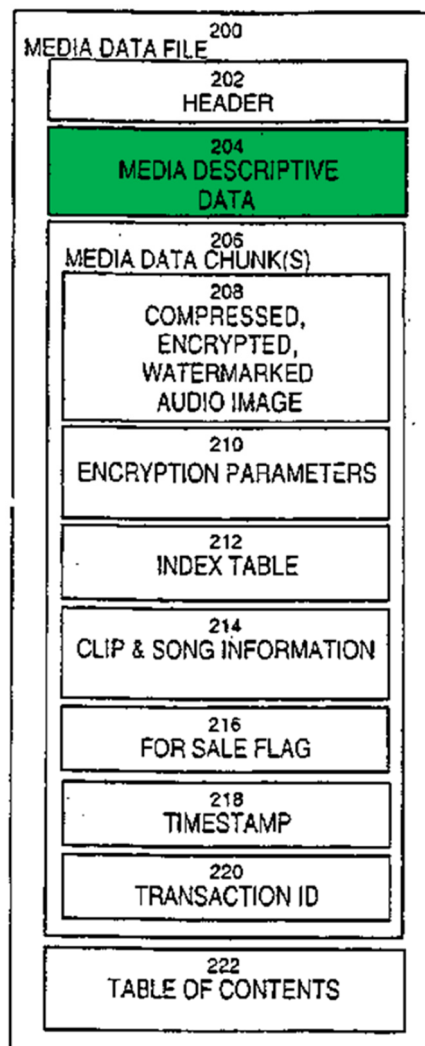


Exhibit 1003, Figure 2 (annotated)

227. Thus, Wiser in view of Dierks and Chang renders claim 2 obvious.

3. Dependent Claim 3

a) “The system of claim 1, wherein the web site is a social media web site.”

228. Wiser in view of Dierks and Chang renders obvious claim 1, as I discussed above. Wiser in combination with Dierks and Chang also renders obvious the additional limitation of claim 3 for at least the same reasons discussed for element 1[f].

229. Thus, Wiser in view of Dierks and Chang renders claim 3 obvious.

4. Dependent Claim 4

a) “The system of claim 1, wherein the transaction processing service accepts a credit card.”

230. Wiser in view of Dierks and Chang renders obvious claim 1, as I discussed above. Wiser discloses the additional limitation of claim 4. For example, Wiser discloses that “[t]he **payment processing systems 134** are conventional payment authorization systems, such as **credit card** authorization systems or debit card payment authorization systems.” Exhibit 1003, 11:12–15. Thus, Wiser’s **payment processing systems 134** “accepts a credit card.”

231. Thus, Wiser in view of Dierks and Chang renders claim 4 obvious.

5. Dependent Claim 5

a) “The system of claim 1, further comprising, distributing the product listing to a social media network.”

232. Wiser in view of Dierks and Chang renders obvious claim 1, as I discussed above. The combination of Wiser, Dierks, and Chang also renders obvious the additional limitation of claim 5 for at least the same reasons discussed for element 1[f].

233. Thus, Wiser in view of Dierks and Chang renders claim 5 obvious.

6. Independent Claim 6

a) 6[preamble]: “A system for generating an electronic commerce store, comprising:”

234. In my opinion, even if the preamble is limiting, Wiser discloses it for at least the same reasons discussed for element 1[preamble].

b) 6[a]: “an application server, the application server controlling a transaction processing service;”

235. Wiser discloses element 6[a] for at least the same reasons discussed for element 1[a].

c) 6[b]: “a storage resource communicatively coupled to the application server, the storage resource storing an image and metadata related to a product;”

236. Wiser discloses element 6[b] for at least the same reasons discussed for element 1[b].

d) 6[c]: “a widget builder resource executing on the application server,”

237. Wiser in combination with Dierks renders obvious element 6[c] for at least the same reasons discussed for element 1[c].

e) 6[d]: “the widget builder resource is programmed to: (i) receive the image and metadata from the storage resource;”

238. Wiser in combination with Dierks renders obvious element 6[d] for at least the same reasons discussed for element 1[d].

f) 6[e]: “[the widget builder resource is programmed to:] (ii) generate a product listing, wherein the product listing includes the image, the metadata, and a clickable link, and wherein the clickable link is a buy button and includes a preview of the product, and”

239. Wiser in combination with Dierks renders obvious element 6[e] for at least the same reasons discussed for element 1[e].

g) 6[f]: “wherein the application server is configured to embed the product listing into a web site not operated by the application server and into a social media network, and”

240. Wiser in combination with Dierks and Chang renders obvious element 6[f] for at least the same reasons discussed for element 1[f]. Chang makes clear its widget may be embedded in web pages “associated with social networking web sites” that include “FACEBOOK™, MYSPACE™, LINKED-IN™, or some other suitable social networking site.” Exhibit 1008, ¶¶ 30, 41. Any of these social networking web sites is “a web site not operated by the application server.”

241. Therefore, Wiser in combination with Dierks and Chang renders obvious discloses element 6[f].

h) 6[g]: “wherein when the application server receives an indication that the clickable link has been activated by a client device in Internet communication with the web site, the

application server provides a checkout screen to the client device,”

242. Wiser discloses or, in combination with Dierks, renders obvious element 6[g] for at least the same reasons discussed for element 1[g].

i) 6[h]: “wherein the transaction processing service processes a payment transaction for a purchase of the product initiated from the checkout screen using the client device.”

243. Wiser discloses element 6[h] for at least the same reasons discussed for element 1[h].

244. Therefore, Wiser in view of Dierks and Chang renders claim 6 obvious.

7. Dependent Claim 7

a) “The system of claim 6, wherein the application server is capable of compiling analytics related to the product listing.”

245. Wiser in view of Dierks and Chang renders obvious claim 6, as I discussed above. Wiser discloses the additional limitation of claim 7. For example, Wiser discloses that content manager 112 (i.e., “the application server”) includes an administration module 1006, which defines “the number of songs available for purchase” and “manages and tracks performance statistics, such as overall volume, throughput, and the like.” Exhibit 1003, 20:49–51, 22:32–39, Figure 10. Thus, Wiser’s content manager 112 “is capable of compiling analytics related to the product listing.”

246. Thus, Wiser in view of Dierks and Chang renders claim 7 obvious.

8. Dependent Claim 8

a) “The system of claim 6, wherein the product listing further includes a share link.”

247. Wiser in view of Dierks and Chang renders obvious claim 6, as I discussed above. Wiser in view of Dierks and Chang also renders obvious the additional limitation of claim 8.

248. As I discussed for element 1[f], a POSITA would have been motivated to combine and would have had a reasonable expectation of success in combining the teachings of Wiser, Dierks, and Chang to embed the Wiser-Dierks system’s product listing into a social media network. Wiser and Dierks do not explicitly disclose that the product listing “*includes a share link.*” However, a POSITA would have been motivated by Chang’s teachings to further configure the Wiser-Dierks system such that the product listing “*includes a share link.*”

249. Chang discloses including a share button in the widget code. Exhibit 1008, ¶ 43. Chang explains that “[t]his share button, when executed using an input device, may generate a prompt asking the user 201 where they would like to embed the widget code (e.g., what web page associated with what web site and servers associated with the web site).” *Id.* Including a share button as taught by Chang in the Wiser-Dierks system’s product listing would, for example, have allowed sellers to readily share their product listings, thereby expanding their market. *Id.* The POSITA would have been motivated to combine the teachings of Wiser, Dierks, and

Chang for the reasons I discussed in Section IX.B.1.g)(1) and would have had a reasonable expectation of success in combining the teachings of Wiser, Dierks, and Chang as I explained in Section IX.B.1.g)(2).

250. Thus, Wiser in view of Dierks and Chang renders claim 8 obvious.

9. Dependent Claim 9

a) “The system of claim 6, wherein the web site is a social media network site.”

251. Wiser in view of Dierks and Chang renders obvious claim 6, as I discussed above. Wiser in view of Dierks also renders obvious the additional limitation of claim 9 for at least the same reasons discussed for element 1[f].

252. Thus, Wiser in view of Dierks and Chang renders claim 9 obvious.

10. Dependent Claim 10

a) “The system of claim 6, wherein the transaction processing service is a proprietary service controlled by the application server.”

253. Wiser in view of Dierks and Chang renders obvious claim 6, as I discussed above. Wiser discloses the additional limitation of claim 10. As I discussed above, Wiser’s **content manager 112** (i.e., “the application server”) is configured to control **payment processing system 134** (i.e., “a transaction processing service”). Wiser discloses that “[t]he **payment processing systems 134** are conventional payment authorization systems, such as credit card authorization systems or debit card payment authorization systems.” Exhibit 1003, 11:12–15.

Because conventional credit card authorization systems or debit card payment authorization systems are typically managed internally by banking institutions that issue credit or debit cards, a POSITA would have understood **payment processing systems 134** to be “a proprietary service.” Cf. Exhibit 1001, 18:13–16 (“the transaction processing service 112 is implemented with a wholly internal, proprietary service that is not generally accessible over the Internet or other public computer communications networks”).

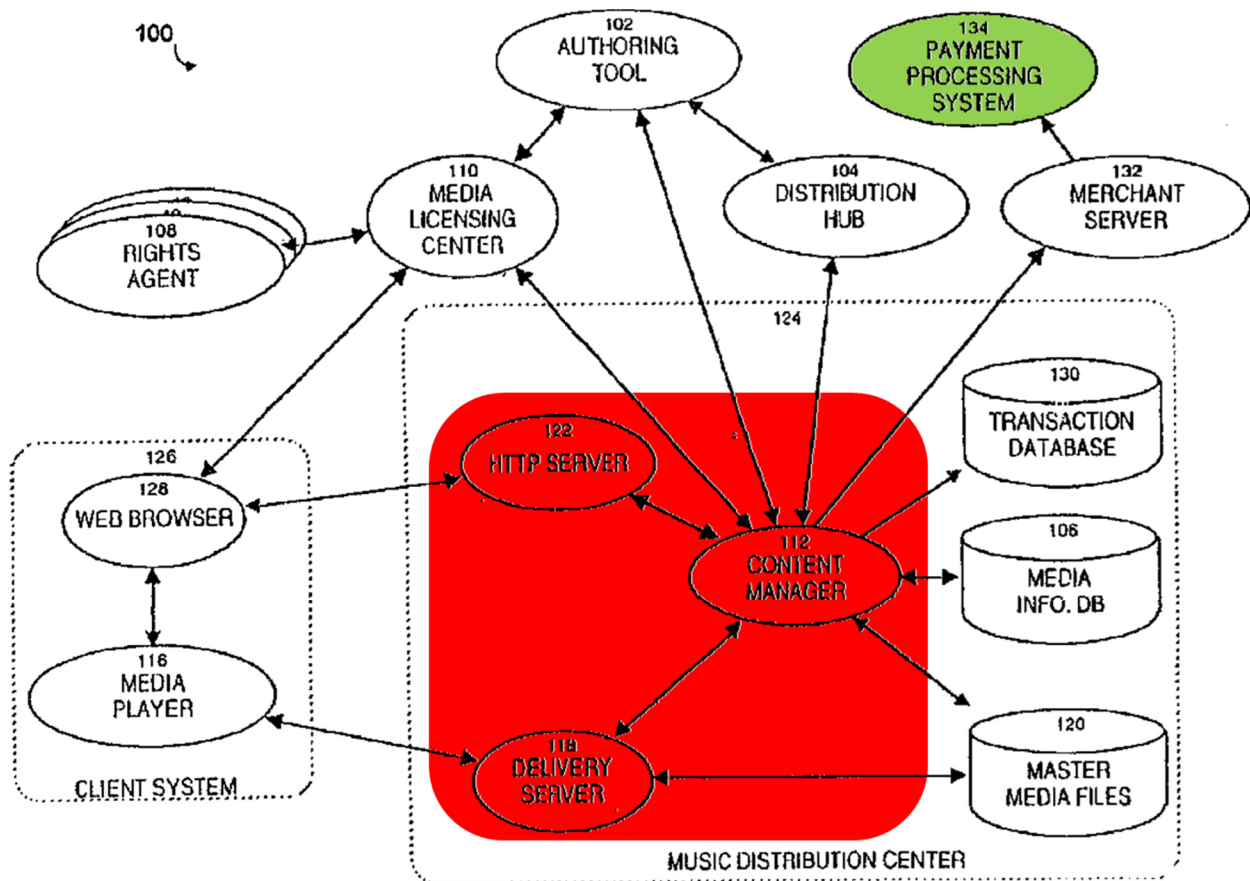


Exhibit 1003, Figure 1 (annotated)

254. Thus, Wiser in view of Dierks and Chang renders claim 10 obvious.

11. Independent Claim 11

a) 11[preamble]: “A system for generating an electronic commerce store, comprising:”

255. In my opinion, even if the preamble is limiting, Wiser discloses it for at least the same reasons discussed for element 1[preamble].

b) 11[a]: “an application server, the application server controlling a transaction processing service;”

256. Wiser discloses element 11[a] for at least the same reasons discussed for element 1[a].

c) 11[b]: “a storage resource communicatively coupled to the application server, the storage resource storing an image and metadata related to a product;”

257. Wiser discloses element 11[b] for at least the same reasons discussed for element 1[b].

d) 11[c]: “a widget builder resource executing on the application server,”

258. Wiser in combination with Dierks renders obvious element 11[c] for at least the same reasons discussed for element 1[c].

e) 11[d]: “the widget builder resource configured to: receive the image and metadata from the storage resource;”

259. Wiser in combination with Dierks renders obvious element 11[d] for at least the same reasons discussed for element 1[d].

f) 11[e]: “[the widget builder resource configured to:] generate a product listing that includes the image, the

metadata, and a clickable link, wherein the clickable link is a buy button and includes a preview of the product, and”

260. Wiser in combination with Dierks renders obvious element 11[e] for at least the same reasons discussed for element 1[e].

g) 11[f]: “wherein the application server is configured to embed the product listing into a social media network and into a web site, and”

261. Wiser in combination with Dierks and Chang renders obvious element 11[f] for at least the same reasons discussed for element 1[f].

h) 11[g]: “wherein when the application server receives an indication that the clickable link has been activated by a client device in Internet communication with the social media network, the application server provides a checkout screen to the client device,”

262. Wiser discloses or, in combination with Dierks, renders obvious element 11[g] for at least the same reasons discussed for element 1[g]. Element 11[g] recites “a client device in Internet communication with the social media network” rather than “with the website.” Chang discloses embedding a product listing in web sites that “may be associated with social networking web sites.” Thus, in the combination of Wiser, Dierks, and Chang, when the client device is in communication with the website, it is also in communication with the social media network.

263. Therefore, Wiser discloses or, in combination with Dierks and Chang, renders obvious element 11[g].

i) 11[h]: “wherein the transaction processing service is configured to process a payment transaction for a purchase of the product initiated from the checkout screen using the client device.”

264. Wiser discloses element 11[h] for at least the same reasons discussed for element 1[h].

265. Therefore, Wiser in view of Dierks and Chang renders obvious claim 11.

12. Dependent Claim 12

a) “The system of claim 11, wherein the social media network is capable of displaying multiple product listings.”

266. Wiser in view of Dierks and Chang renders obvious claim 11, as I discussed above. The combination of Wiser, Dierks, and Chang further renders obvious the additional limitation of claim 12.

267. As I discussed for the element 1[f] (to which I refer for element 11[f]), the Wiser-Dierks-Chang system is configured to embed a product listing in a social media network such as “FACEBOOK™, MYSPACE™, LINKED-IN™, or some other suitable social networking site.” Exhibit 1008, ¶ 41.

268. Wiser teaches multiple products. For instance, Wiser discloses that its invention is directed to “online distribution of digital media” and defines digital media as being “computer software, video, and audio.” Exhibit 1003, 1:8–11, 1:27–

32 (“Even digital media, such as computer software, video, and audio is purchased in this manner...”).

269. Thus, in the Wiser-Dierks-Chang system, a POSITA would have found it obvious to provide, in a social media network, multiple product listings that include at least audio and video. Accordingly, in the Wiser-Dierks-Chang system, *“the social media network is capable of displaying multiple product listings.”*

270. Thus, Wiser in view of Dierks and Chang renders claim 12 obvious.

13. Dependent Claim 13

a) “The system of claim 11, wherein the product listing includes at least one of a title, a description, or a price.”

271. Wiser in view of Dierks and Chang renders obvious claim 11, as I discussed above. Wiser discloses the additional limitation of claim 13 for at least the same reasons discussed for claim 2. Claim 2 recites *“the metadata”* instead of *“the product listing.”* As I discussed for element 1[e], the product listing includes the metadata, which “include **descriptive text**, such as **title**, artist, lyrics, and liner notes.” Exhibit 1003, 6:62–63.

272. Thus, Wiser in view of Dierks and Chang renders claim 13 obvious.

14. Dependent Claim 14

a) “The system of claim 11, wherein the transaction processing service is a proprietary service operated by the application server.”

273. Wiser in view of Dierks and Chang renders obvious claim 11, as I discussed above. Wiser discloses the additional limitation of claim 14 for at least the same reasons discussed for claim 10.

274. Thus, Wiser in view of Dierks and Chang renders claim 14 obvious.

X. CONCLUSION

275. Based on the foregoing, it is my opinion that claims 1–14 of the '995 patent are invalid.

XI. AVAILABILITY FOR CROSS-EXAMINATION

276. In signing this declaration, I recognize that the declaration will be filed as evidence in a contested case before the Patent Trial and Appeal Board of the United States Patent and Trademark Office. I also recognize that I may be subject to cross-examination in the case and that cross-examination will take place within the United States. If cross-examination is required of me, I will appear for cross-examination within the United States during the time allotted for cross-examination.

XII. RIGHT TO SUPPLEMENT

277. I reserve the right to supplement my opinions in the future to respond to any arguments that the Patent Owner raises and to take into account new information as it becomes available to me.

XIII. JURAT

278. I 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 Section 1001 of Title 18 of the United States Code.

Dated: November 21, 2024



Dr. Todd Mowry

EXHIBIT A

Todd C. Mowry

CURRICULUM VITAE

November, 2024

Contact Information

Computer Science Department
Carnegie Mellon University
5000 Forbes Avenue
Pittsburgh, PA 15213

Phone: (412) 268-3725
Fax: (412) 268-5576
Email: tcm@cs.cmu.edu
URL: <http://www.cs.cmu.edu/~tcm>

Education

Ph.D. in Electrical Engineering, Stanford University, March 1994.

- Thesis: “Tolerating Latency Through Software-Controlled Data Prefetching.”
- Supervisors: Anoop Gupta and Monica Lam.

M.S. in Electrical Engineering, Stanford University, June 1989.

B.S. in Electrical Engineering with Highest Distinction, University of Virginia, May 1988.

Academic Appointments

7/08–Present: Professor, Computer Science Department (with a courtesy appointment in the Department of Electrical and Computer Engineering), Carnegie Mellon University.

8/09–6/10: Associate Department Head for Faculty, Computer Science Department

6/15–Present: Director of the Ph.D. program, Computer Science Department

7/97–6/08: Associate Professor, Computer Science Department (with a courtesy appointment in the Department of Electrical and Computer Engineering), Carnegie Mellon University.

12/93–6/97: Assistant Professor, Department of Electrical and Computer Engineering (with a courtesy appointment in the Department of Computer Science), University of Toronto.

9/89–11/93: Graduate Research Assistant, Stanford University.

Honors and Awards

- ASPLOS Influential Paper Award, 2022 (for “Design and Evaluation of a Compiler Algorithm for Prefetching,” from ASPLOS 1992).
- Best Paper Runner-Up Award, VLDB conference, 2018.
- ACM Fellow, for contributions to software prefetching and thread-level speculation, 2016.
- SCS Doctoral Dissertation Award co-won by Angela Demke Brown (Ph.D. advisee), 2005.
- Most Thought-Provoking Idea Award (for Claytronics), the Wild and Crazy Idea Session IV held at ASPLOS-XI, 2004.
- Best Paper Award, the 20th International Conference on Data Engineering (ICDE), 2004.
- SCS Doctoral Dissertation Award co-won by J. Gregory Steffan (Ph.D. advisee), 2003.
- Runner-Up for Best Paper Award, the ACM SIGMOD Conference, 2001.
- Alfred P. Sloan Research Fellow, 1999–2001.
- TR100 Award (awarded by MIT’s Technology Review magazine to the top 100 most promising young innovators in science and technology), 1999.
- Best Paper Award, the Second Symposium on Operating Systems Design and Implementation, 1996.

Doctoral Thesis Supervision

Student	Thesis Title (or Topic)	Graduation Date
Chi-Keung Luk	Optimizing the Cache Performance of Non-Numeric Applications. <i>(Nominated for the ACM Thesis Award by the University of Toronto Department of Computer Science.)</i>	January, 2000
J. Gregory Steffan	Hardware Support for Thread-Level Speculation. <i>(Co-winner of the SCS Doctoral Dissertation Award.)</i>	April, 2003
Antonia Zhai	Compiler Optimization of Value Communication for Thread-Level Speculation.	January, 2005
Angela Demke Brown	Explicit Compiler-based Memory Management for Out-of-core Applications. <i>(Co-winner of the SCS Doctoral Dissertation Award.)</i>	May, 2005
Christopher Colohan	Applying Thread-Level Speculation to Database Transactions.	November, 2005
Shimin Chen	Redesigning Database Systems in Light of CPU Cache Prefetching.	December, 2005
Amit Manjhi	Increasing the Scalability of Dynamic Web Applications	March, 2008
Olatunji Ruwase	Improving Device Driver Reliability Through Decoupled Dynamic Binary Analysis	May, 2013
Michelle Goodstein	Dataflow Analysis-Based Dynamic Parallel Monitoring	August, 2014
Vivek Seshadri	Simple DRAM and Virtual Memory Abstractions to Enable Highly Efficient Memory Subsystems	May, 2016
Gennady Pekhimenko	Practical Data Compression for Modern Memory Hierarchies	July, 2016
Chris Fallin	Finding and Exploiting Parallelism with Data-Structure-Aware Static and Dynamic Analysis	February, 2019
Prashanth Menon	On Building Robustness into Compilation-Based Main-Memory Database Query Engines	May, 2021
Pratik Fegade	Auto-batching Techniques for Dynamic Deep Learning Computations	January, 2023
Ziqi Wang	Building a More Efficient Cache Hierarchy by Supporting Multiversioning and Leveraging Application-Level Information	January, 2023
Sam Arch	Unifying Declarative and Non-Declarative Code in Relational Database Management Systems	June, 2027 <i>(expected)</i>
Hongyi Jin	Compiler Optimizations for Machine Learning Applications	June, 2027 <i>(expected)</i>
Ruihang Lai	Compiler Optimizations for Machine Learning Applications	June, 2027 <i>(expected)</i>
Patrick Coppock	OS Support for Machine Learning Applications	June, 2027 <i>(expected)</i>
Eliot Solomon	OS Support for Machine Learning Applications	June, 2029 <i>(expected)</i>

Publications

Book Chapters

1. Amir Yazdanbakhsh, Gennady Pekhimenko, Hadi Esmaeilzadeh, Onur Mutlu, and Todd C. Mowry. Towards Breaking the Memory Bandwidth Wall Using Approximate Value Prediction. In *Approximate Circuits: Methodologies and CAD*, Springer International Publishing, 2019, pages 417–441.

Refereed Journal Articles

1. Mishra, Deepanjali, Konstantinos Kanellopoulos, Ashish Panwar, Akshitha Sriraman, Vivek Seshadri, Onur Mutlu, and Todd C. Mowry. Address Scaling: Architectural Support for Fine-Grained Thread-Safe Metadata Management. In *IEEE Computer Architecture Letters*, 23(01), January 2024.
2. Prashanth Menon, Amadou Ngom, Lin Ma, Todd C. Mowry, and Andrew Pavlo. Permutable Compiled Queries: Dynamically Adapting Compiled Queries without Recompiling. In *Proceedings of the VLDB Endowment*, 14(2):101-113, October 2020.
3. Amir Yazdanbakhsh, Gennady Pekhimenko, Bradley Thwaites, Hadi Esmaeilzadeh, Onur Mutlu, and Todd C. Mowry. RFVP: Rollback-Free Value Prediction with Safe-to-Approximate Loads. In *ACM Transactions on Architecture and Code Optimization (TACO)*, 12(4):62, February 2016.
4. Amir Yazdanbakhsh, Gennady Pekhimenko, Bradley Thwaites, Hadi Esmaeilzadeh, Onur Mutlu, and Todd C. Mowry. Mitigating the Memory Bottleneck with Approximate Load Value Prediction. In *IEEE Design and Test*, 33(1), January 2016.
5. Vivek Seshadri, Samihan Yedkar, Hongyi Xin, Onur Mutlu, Phillip B. Gibbons, Michael A. Kozuch, and Todd C. Mowry. Mitigating Prefetcher-Caused Pollution Using Informed Caching Policies for Prefetched Blocks. In *ACM Transactions on Architecture and Code Optimization (TACO)*, 11(4), January 2015.
6. Shimin Chen, Phillip B. Gibbons, Michael Kozuch, and Todd C. Mowry. Log-Based Architectures: Using Multicore to Help Software Behave Correctly. In *Operating Systems Review*, 45(1), January 2011.
7. Shimin Chen, Michael Kozuch, Phillip B. Gibbons, Michael Ryan, Theodoros Strigkos, Todd C. Mowry, Olatunji Ruwase, Evangelos Vlachos, Babak Falsafi, and Vijaya Ramachandran. Flexible Hardware Acceleration for Instruction-Grain Lifeguards. In *IEEE Micro (Top Picks from 2008 Computer Architecture Conferences)*, 29(1), January 2009.
8. Seth C. Goldstein, Todd C. Mowry, Jason D. Campbell, Michael P. Ashley-Rollman, Michael De Rosa, Stanislav Funiak, James F. Hoburg, Mustafa E. Karagozler, Brian Kirby, Peter Lee, Padmanabhan Pillai, J. Robert Reid, Daniel D. Stancil, and Michael P. Weller. Beyond Audio and Video: Using Claytronics to Enable Pario. In *AI Magazine*, 30(2), March 2009.
9. Antonia Zhai, J. Gregory Steffan, Christopher B. Colohan, and Todd C. Mowry. Compiler and Hardware Support for Reducing the Synchronization of Speculative Threads. In *ACM Transactions on Architecture and Code Optimization (TACO)*, 5(1), May 2008.
10. Christopher B. Colohan, Anastassia Ailamaki, J. Gregory Steffan, and Todd C. Mowry. Incrementally parallelizing database transactions with thread-level speculation. In *ACM Transactions on Computer Systems (TOCS)*, 26(1), February 2008.
11. Shimin Chen, Anastassia Ailamaki, Phillip B. Gibbons, and Todd C. Mowry. Improving Hash Join Performance through Prefetching. In *ACM Transactions on Database Systems*, 32(3):1-32, September 2007.

12. Christopher B. Colohan, Anastasia Ailamaki, J. Gregory Steffan, and Todd C. Mowry. CMP Support for Large and Dependent Speculative Threads. In *IEEE Transactions on Parallel and Distributed Systems*, 18(8):1041-1054, August 2007.
13. J. Gregory Steffan, Christopher B. Colohan, Antonia Zhai and Todd C. Mowry. The STAMPede Approach to Thread-Level Speculation. In *ACM Transactions on Computer Systems*, 23(3):253-300, August 2005.
14. Seth Copen Goldstein, Jason Campbell and Todd C. Mowry. Programmable Matter. In *IEEE Computer*, 38(6):99-101, June 2005.
15. Angela Demke Brown, Todd C. Mowry and Orran Krieger. Compiler-Based I/O Prefetching for Out-of-Core Applications. In *ACM Transactions on Computer Systems*, 19(2):111-170, May 2001.
16. Chi-Keung Luk and Todd C. Mowry. Architectural and Compiler Support for Effective Instruction Prefetching: A Cooperative Approach. In *ACM Transactions on Computer Systems*, 19(1):71-109, February 2001.
17. Todd C. Mowry and Chi-Keung Luk. Understanding Why Correlation Profiling Improves the Predictability of Data Cache Misses in Nonnumeric Applications. In *IEEE Transactions on Computers*, 49(4), April 2000.
18. Chi-Keung Luk and Todd C. Mowry. Automatic Compiler-Inserted Prefetching for Pointer-Based Applications. In *IEEE Transactions on Computers*, 48(2), February 1999.
19. Mark Horowitz, Margaret Martonosi, Todd C. Mowry, and Michael D. Smith. Informing Memory Operations: Memory Performance Feedback Mechanisms and their Applications. In *ACM Transactions on Computer Systems*, 16(2):170-205, May 1998.
20. Todd C. Mowry. Tolerating Latency in Multiprocessors through Compiler-Inserted Prefetching. In *ACM Transactions on Computer Systems*, 16(1):55-92, February 1998.
21. Todd Mowry and Anoop Gupta. Tolerating Latency Through Software-Controlled Prefetching in Shared-Memory Multiprocessors. In *Journal of Parallel and Distributed Computing*, 12(2):87-106, 1991.

Refereed Conference Papers

1. Sam Arch, Yuchen Liu, Todd C. Mowry, Jignesh Patel, and Andy Pavlo. The Key to Effective UDF Optimization: Before Inlining, First Perform Outlining. To appear in *Proceedings of VLDB 18*, 2025.
2. Fegade, Pratik, Tianqi Chen, Phillip Gibbons, and Todd Mowry. ACROBAT: Optimizing Auto-batching of Dynamic Deep Learning at Compile Time. In *Proceedings of the Sixth Conference on Machine Learning and Systems (MLSys)*, May 2024.
3. Kai Franz, Samuel Arch, Denis Hirn, Torsten Grust, Todd C. Mowry, and Andrew Pavlo. Dear User-Defined Functions, Inlining Isn't Working Out So Great For Us. Let's Try Batching To Make Our Relationship Work. Sincerely, SQL. To appear in *Proceedings of the 2024 Conference on Innovative Data Systems Research (CIDR)*, January 2024.
4. Wang, Ziqi, Kaiyang Zhao, Pei Li, Andrew Jacob, Michael Kozuch, Todd Mowry, and Dimitrios Skarlatos. Memento: Architectural Support for Ephemeral Memory Management in Serverless Environments. In *Proceedings of the 56th Annual IEEE/ACM International Symposium on Microarchitecture (Micro)*, October 2023.
5. Siyuan Chen, Pratik Pramod Fegade, Phillip Gibbons, Todd C. Mowry, and Tianqi Chen. ED-Batch: Efficient Automatic Batching of Dynamic Neural Networks via Learned Finite State Machines. In *Proceedings of the Fortieth International Conference on Machine Learning (ICML 2023)*, July 2023.

6. Graham Gobieski, Souradip Ghosh, Marijn Heule, Todd C. Mowry, Tony Nowatzki, Nathan Beckmann, and Brandon Lucia. Riptide: A programmable, energy-minimal dataflow compiler and architecture. In *Proceedings of the 2022 55th IEEE/ACM International Symposium on Microarchitecture (MICRO)*, October 2022.
7. Pratik Fegade, Tianqi Chen, Phillip Gibbons, and Todd C. Mowry. The CoRa Tensor Compiler: Compilation for Ragged Tensors with Minimal Padding. In *Proceedings of the Fifth Conference on Machine Learning and Systems (MLSys)*, August 2022.
8. Amadou Ngom, Prashanth Menon, Matthew Butrovich, Lin Ma, Wan Shen Lim, Todd C. Mowry, and Andrew Pavlo. Filter Representation in Vectorized Query Execution. In *Proceedings of the 17th International Workshop on Data Management on New Hardware (DaMoN)*, June 2021.
9. Ziqi Wang, Chul-Hwan Choo, Michael A. Kozuch, Todd C. Mowry, Gennady Pekhimenko, Vivek Seshadri, and Dimitrios Skarlatos. NVOOverlay: Enabling Efficient and Scalable High-Frequency Snapshotting to NVM. In *Proceedings of the 48th Annual International Symposium on Computer Architecture (ISCA)*, June 2021.
10. Pratik Fegade, Tianqi Chen, Phillip Gibbons, and Todd C. Mowry. Cortex: A Compiler for Recursive Deep Learning Models. In *Proceedings of the Fourth Conference on Machine Learning and Systems (MLSys)*, April 2021.
11. Daming D. Chen, Phillip B. Gibbons, and Todd C. Mowry. TardisTM: Incremental Repair for Transactional Memory. In *Proceedings of the Eleventh International Workshop on Programming Models and Applications for Multicores and Manycores (PMAM '20)*, February 2020.
12. Ziqi Wang, Michael A. Kozuch, Todd C. Mowry, and Vivek Seshadri. Multiversioned Page Overlays: Enabling Faster Serializable Hardware Transactional Memory. In *Proceedings of the 28th International Conference on Parallel Architectures and Compilation Techniques (PACT)*, September 2019.
13. Vivek Seshadri, Donghyuk Lee, Thomas Mullins, Hasan Hassan, Amirali Boroumand, Jeremie Kim, Michael A. Kozuch, Onur Mutlu, Phillip B. Gibbons, and Todd C. Mowry. Ambit: in-memory accelerator for bulk bitwise operations using commodity DRAM technology. In *Proceedings of the 50th Annual International Symposium on Microarchitecture (MICRO-50)*, October 2017.
14. Prashanth Menon, Andrew Pavlo, and Todd C. Mowry. Relaxed Operator Fusion for In-Memory Databases: Making Compilation, Vectorization, and Prefetching Work Together At Last. In *Proceedings of VLDB 11(1): 1-13*, September 2017.
15. Andrew Pavlo, Gustavo Angulo, Joy Arulraj, Haibin Lin, Jiexi Lin, Lin Ma, Prashanth Menon, Todd Mowry, Matthew Perron, Ian Quah, Siddharth Santurkar, Anthony Tomasic, Skye Toor, Dana Van Aken, Ziqi Wang, Yingjun Wu, Ran Xian, and Tieying Zhang. Self-Driving Database Management Systems” In *Proceedings of the 2017 Conference on Innovative Data Systems Research (CIDR)*, January 2017.
16. Gennady Pekhimenko, Evgeny Bolotin, Nandita Vijaykumar, Onur Mutlu, Todd C. Mowry, and Stephen W. Keckler. A Case for Toggle-Aware Compression for GPU Systems. In *Proceedings of the 22nd International Symposium on High-Performance Computer Architecture (HPCA)*, March 2016.
17. Michelle L. Goodstein, Phillip B. Gibbons, Michael A. Kozuch, and Todd C. Mowry. Tracking and Reducing Uncertainty in Dataflow Analysis-Based Dynamic Parallel Monitoring. In *Proceedings of the 2015 International Conference on Parallel Architectures and Compilation Techniques (PACT '15)*, October 2015.
18. Nandita Vijaykumar, Gennady Pekhimenko, Adwait Jog, Abhishek Bhowmick, Rachata Ausavarungnirun, Chita Das, Mahmut Kandemir, Todd C. Mowry, and Onur Mutlu. A Case for Core-Assisted Bottleneck Acceleration in GPUs: Enabling Efficient Data Compression. In *Proceed-*

- ings of the 42nd Annual International Symposium on Computer Architecture (ISCA), June 2015.
19. Vivek Seshadri, Gennady Pekhimenko, Olatunji Ruwase, Onur Mutlu, Phillip B. Gibbons, Michael A. Kozuch, Todd C. Mowry, and Trishul Chilimbi. Page Overlays: An Enhanced Virtual Memory Framework to Enable Fine-grained Memory Management. In *Proceedings of the 42nd Annual International Symposium on Computer Architecture (ISCA)*, June 2015.
 20. Gennady Pekhimenko, Tyler Huberty, Rui Cai, Onur Mutlu, Phillip P. Gibbons, Michael A. Kozuch, and Todd C. Mowry. Exploiting Compressed Block Size as an Indicator of Future Reuse. In *Proceedings of the 21st International Symposium on High-Performance Computer Architecture (HPCA)*, February 2015.
 21. Vivek Seshadri, Abhishek Bhowmick, Onur Mutlu, Phillip B. Gibbons, Michael A. Kozuch, and Todd C. Mowry. The Dirty-Block Index. In *Proceedings of the 41st Annual International Symposium on Computer Architecture (ISCA)*, June 2014.
 22. Olatunji Ruwase, Michael A. Kozuch, Phillip B. Gibbons, and Todd C. Mowry. Guardrail: High Fidelity Correctness Checking of Device Drivers for Safeguarding I/O Operations. In *Proceedings of the Nineteenth International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS 2014)*, March 2014.
 23. Gennady Pekhimenko, Vivek Seshadri, Yoongu Kim, Hongyi Xin, Onur Mutlu, Phillip B. Gibbons, Michael A. Kozuch, and Todd C. Mowry. Linearly Compressed Pages: A Low-Complexity, Low-Latency Main Memory Compression Framework. In *Proceedings of the 46th Annual International Symposium on Microarchitecture (MICRO-46)*, December 2013.
 24. Vivek Seshadri, Yoongu Kim, Chris Fallin, Donghyuk Lee, Rachata Ausavarungnirun, Gennady Pekhimenko, Yixin Luo, Onur Mutlu, Phillip B. Gibbons, Michael A. Kozuch, and Todd C. Mowry. RowClone: Fast and Energy-Efficient In-DRAM Bulk Data Copy and Initialization. In *Proceedings of the 46th Annual International Symposium on Microarchitecture (MICRO-46)*, December 2013.
 25. Michelle L. Goodstein, Shimin Chen, Phillip B. Gibbons, Michael A. Kozuch, and Todd C. Mowry. Chrysalis Analysis: Incorporating Synchronization Arcs in Dataflow-Analysis-based Parallel Monitoring. In *Proceedings of the 21st International Conference on Parallel Architectures and Compilation Techniques (PACT-2012)*, September 2012.
 26. Vivek Seshadri, Onur Mutlu, Todd C. Mowry, and Michael A. Kozuch. The Evicted-Address Filter: A Unified Mechanism to Address Both Cache Pollution and Thrashing. In *Proceedings of the 21st International Conference on Parallel Architectures and Compilation Techniques (PACT-2012)*, September 2012.
 27. Gennady Pekhimenko, Vivek Seshadri, Onur Mutlu, Todd C. Mowry, Phillip B. Gibbons, and Michael A. Kozuch. Base-Delta-Immediate Compression: A Practical Data Compression Mechanism for On-Chip Caches. In *Proceedings of the 21st International Conference on Parallel Architectures and Compilation Techniques (PACT-2012)*, September 2012.
 28. Olatunji Ruwase, Shimin Chen, Phillip Gibbons, and Todd C. Mowry. Decoupled Lifeguards: Enabling Path Optimizations for Dynamic Correctness Checking Tools. In *Proceedings of the ACM SIGPLAN 2010 Conference on Programming Language Design and Implementation (PLDI)*, June 2010.
 29. F. Ryan Johnson, Radu Stoica, Anastasia Ailamaki, and Todd C. Mowry. Decoupling Contention Management from Scheduling. In *Proceedings of the Fifteenth International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS 2010)*, March 2010.
 30. Michelle Goodstein, Evangelos Vlachos, Shimin Chen, Phillip B. Gibbons, Michael Kozuch, and Todd C. Mowry. Butterfly Analysis: Adapting Dataflow Analysis to Dynamic Parallel Monitoring. In *Proceedings of the Fifteenth International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS 2010)*, March 2010.

31. Evangelos Vlachos, Michelle Goodstein, Michael Kozuch, Shimin Chen, Babak Falsafi, Phillip B. Gibbons, and Todd C. Mowry. ParaLog: Enabling and Accelerating Online Parallel Monitoring of Multithreaded Applications. In *Proceedings of the Fifteenth International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS 2010)*, March 2010.
32. Amit Manjhi, Charles Garrod, Bruce M. Maggs, Todd C. Mowry, Anthony Tomasic. Holistic Query Transformations for Dynamic Web Applications. In *Proceedings of the 2009 IEEE 25th International Conference on Data Engineering (ICDE)*, March-April 2009.
33. Daniel J. Dewey, Michael P. Ashley-Rollman, Michael DeRosa, Seth Copen Goldstein, Todd C. Mowry, Siddhartha S. Srinivasa, Padmanabhan Pillai, and Jason Campbell. Generalizing metamodules to simplify planning in modular robotic systems. In *Proceedings of the IEEE/RSJ 2008 International Conference on Intelligent Robots and Systems (IROS)*, September 2008.
34. Charles Garrod, Amit Manjhi, Anastasia Ailamaki, Bruce Maggs, Todd Mowry, Christopher Olston, and Anthony Tomasic. Scalable Query Result Caching for Web Applications. In *Proceedings of the 34th International Conference on Very Large Databases (VLDB)*, August 2008.
35. Lei Li, Wenjie Fu, Fan Guo, Todd C. Mowry, and Christos Faloutsos. Cut-and-stitch: efficient parallel learning of linear dynamical systems on SMPs. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, August 2008.
36. Shimin Chen, Michael Kozuch, Theodoros Strigkos, Babak Falsafi, Phillip B. Gibbons, Todd C. Mowry, Michael Ryan, Olatunji Ruwase, and Evangelos Vlachos. Flexible Hardware Acceleration for Instruction-Grain Program Monitoring. In *Proceedings of the 35th Annual International Symposium on Computer Architecture (ISCA)*, June 2008.
37. Olatunji Ruwase, Phillip B. Gibbons, Todd C. Mowry, Vijaya Ramachandran, Shimin Chen, Michael Kozuch, and Michael Ryan. Parallelizing dynamic information flow tracking. In *Proceedings of the 20th Annual ACM Symposium on Parallel Algorithms and Architectures (SPAA)*, June 2008.
38. Brian Kirby, Burak Aksak, James Hoburg, Todd C. Mowry, and Padmanabhan Pillai. A Modular Robotic System Using Magnetic Force Effectors. In *Proceedings of the IEEE/RSJ 2007 International Conference on Intelligent Robots and Systems (IROS)*, October 2007.
39. Michael Ashley-Rollman, Seth Goldstein, Peter Lee, Todd C. Mowry, and Padmanabhan Pillai. Meld: A Declarative Approach to Programming Ensembles. In *Proceedings of the IEEE/RSJ 2007 International Conference on Intelligent Robots and Systems (IROS)*, October 2007.
40. Amit Manjhi, Phillip B. Gibbons, Anastassia Ailamaki, Charles Garrod, Bruce M. Maggs, Todd C. Mowry, Christopher Olston, Anthony Tomasic, and Haifeng Yu. Invalidation Clues for Database Scalability Services. In *Proceedings of the 2007 IEEE 23rd International Conference on Data Engineering (ICDE)*, pages 316-325, April 2007.
41. Michael De Rosa, Peter Lee, Seth Goldstein, Jason Campbell, Padmanabhan Pillai, and Todd C. Mowry. Distributed Watchpoints: Debugging Large Multi-Robot Systems. In *Proceedings of the 2007 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3723-3729, April 2007.
42. Benjamin D. Rister, Jason Campbell, Padmanabhan Pillai, and Todd C. Mowry. Integrated Debugging of Large Modular Robot Ensembles. In *Proceedings of the 2007 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2227-2234, April 2007.
43. Shimin Chen, Phillip Gibbons, Michael Kozuch, Vasileios Liaskovitis, Anastassia Ailamaki, Guy Blelloch, Babak Falsafi, Limor Fix, Nikos Hardavellas, Todd C. Mowry and Chris Wilkerson. Scheduling Threads for Constructive Cache Sharing on CMPs. In *Proceedings of 18th ACM Symposium on Parallelism in Algorithms and Architectures (SPAA)*, pages 105-115, June 2007.

44. Shimin Chen, Babak Falsafi, Phillip B. Gibbons, Michael Kozuch, Todd C. Mowry, Radu Teodorescu, Anastassia Ailamaki, Limor Fix, Gregory R. Ganger, Bin Lin, and Steven W. Schlosser. Log-Based Architectures for General-Purpose Monitoring of Deployed Code. In *Proceedings of the Workshop on Architectural and System Support for Improving Software Dependability (ASID), held with ASPLOS XII*, October 2006.
45. Christopher B. Colohan, Anastassia Ailamaki, J. Gregory Steffan, and Todd C. Mowry. Tolerating Dependences Between Large Speculative Threads Via Sub-Threads. In *Proceedings of the 33rd Annual International Symposium on Computer Architecture (ISCA)*, pages 216-226, June 2006.
46. Amit Manjhi, Anastassia Ailamaki, Bruce M. Maggs, Todd C. Mowry, Christopher Olston, and Anthony Tomasic. Simultaneous Scalability and Security for Data-Intensive Web Applications. In *Proceedings of the 2006 ACM SIGMOD International Conference on Management of Data*, pages 241-252, June 2006.
47. Vasileios Liaskovitis, Shimin Chen, Phillip B. Gibbons, Anastassia Ailamaki, Guy E. Blelloch, Babak Falsafi, Limor Fix, Nikos Hardavellas, Michael Kozuch, Todd C. Mowry, and Chris Wilkerson. Parallel Depth First vs. Work Stealing Schedulers on CMP Architectures. In *Proceedings of 18th ACM Symposium on Parallelism in Algorithms and Architectures (SPAA)*, August 2006.
48. Brian Kirby, Jason Campbell, Burak Aksak, Padmanabhan Pillai, James F. Hoburg, Todd C. Mowry, and Seth Copen Goldstein. Catoms: Moving Robots without Moving Parts. In *Proceedings of the Twentieth National Conference on Artificial Intelligence (AAAI)*, pages 1730-1731, July 2005.
49. Christopher Olston, Amit Manjhi, Charles Garrod, Anastassia Ailamaki, Bruce M. Maggs, and Todd C. Mowry. A Scalability Service for Dynamic Web Applications. In *Proceedings of the Second Biennial Conference on Innovative Data Systems Research (CIDR)*, pages 56-69, January 2005.
50. Christopher B. Colohan, Anastassia Ailamaki, J. Gregory Steffan, and Todd C. Mowry. Optimistic Intra-Transaction Parallelism on Chip Multiprocessors. In *Proceedings of the 31st International Conference on Very Large Data Bases (VLDB)*, pages 73-84, September 2005.
51. Shimin Chen, Anastassia Ailamaki, Phillip B. Gibbons, and Todd C. Mowry. Inspector Joins. In *Proceedings of the 31st International Conference on Very Large Data Bases (VLDB)*, pages 817-828, September 2005.
52. Antonia Zhai, Christopher B. Colohan, J. Gregory Steffan, and Todd C. Mowry. Compiler Optimization of Memory-Resident Value Communication Between Speculative Threads. In *Proceedings of the 2007 International Symposium on Code Generation and Optimization (CGO)*, pages 39-52, March 2004.
53. Shimin Chen, Anastassia Ailamaki, Phillip B. Gibbons and Todd C. Mowry. Improving Hash Join Performance through Prefetching. In *Proceedings of the 20th International Conference on Data Engineering (ICDE)*, pages 116-127, April 2004. (*This paper received the Best Paper Award.*)
54. Antonia Zhai, Christopher B. Colohan, J. Gregory Steffan and Todd C. Mowry. Compiler Optimization of Scalar Value Communication Between Speculative Threads. In *Proceedings of the Tenth International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS-X)*, October 2002.
55. Shimin Chen, Philip B. Gibbons, Todd C. Mowry and Gary Valentin. Fractal Prefetching B+-Trees: Optimizing Both Cache and Disk Performance. In *Proceedings of the 2002 ACM SIGMOD International Conference on Management of Data*, June 2002.
56. J. Gregory Steffan, Christopher B. Colohan, Antonia Zhai and Todd C. Mowry. Improving Value Communication for Thread-Level Speculation. In *Proceedings of the Eighth International Symposium on High-Performance Computer Architecture (HPCA)*, February 2002.

57. Shimin Chen, Philip B. Gibbons and Todd C. Mowry. Improving Index Performance through Prefetching. In *Proceedings of the 2001 ACM SIGMOD International Conference on Management of Data*, pages 235-246, May 2001. (*This paper received the Runner-Up for Best Paper Award.*)
58. Angela Demke Brown and Todd C. Mowry. Taming the Memory Hogs: Using Compiler-Inserted Releases to Manage Physical Memory Intelligently. In *Proceedings of the Fourth Symposium on Operating Systems Design and Implementation (OSDI 2000)*, pages 31-44, October 2000.
59. J. Gregory Steffan, Christopher B. Colohan, Antonia Zhai and Todd C. Mowry. A Scalable Approach to Thread- Level Speculation. In *Proceedings of the 27th Annual International Symposium on Computer Architecture*, pages 1-12, June 2000.
60. Todd C. Mowry and Sherwyn R. Ramkisson. Software-Controlled Multithreading Using Informing Memory Operations. In *Proceedings of the Sixth International Symposium on High-Performance Computer Architecture*, January, 2000.
61. Chi-Keung Luk and Todd C. Mowry. Memory Forwarding: Enabling Aggressive Layout Optimizations by Guaranteeing the Safety of Data Relocation. In *Proceedings of the 26th Annual International Symposium on Computer Architecture (ISCA)*, pages 88-99, May 1999.
62. Chi-Keung Luk and Todd C. Mowry. Cooperative Prefetching: Compiler and Hardware Support for Effective Instruction Prefetching in Modern Microprocessors. In *Proceedings of the 31st Annual International Symposium on Microarchitecture*, pages 182-193, December 1998.
63. J. Gregory Steffan and Todd C. Mowry. The Potential for Using Thread-Level Data Speculation to Facilitate Automatic Parallelization. In *Proceedings of the Fourth International Symposium on High-Performance Computer Architecture*, pages 2-13, February, 1998.
64. Charles Chan, Adley Lo, and Todd C. Mowry. Comparative Evaluation of Latency Tolerance Techniques for Software Distributed Shared Memory. In *Proceedings of the Fourth International Symposium on High-Performance Computer Architecture*, pages 300-311, February, 1998.
65. Todd C. Mowry and Chi-Keung Luk. Predicting Data Cache Misses in Non-Numeric Applications Through Correlation Profiling. In *Proceedings of the 30th Annual International Symposium on Microarchitecture*, pages 314-320, December 1997.
66. Todd C. Mowry, Angela K. Demke and Orran Krieger. Automatic Compiler-Inserted I/O Prefetching for Out-of-Core Applications. In *Proceedings of the Second Symposium on Operating Systems Design and Implementation (OSDI '96)*, pages 3-17, October 1996. (*This paper received the Best Paper Award.*)
67. Chi-Keung Luk and Todd C. Mowry. Compiler-Based Prefetching for Recursive Data Structures. In *Proceedings of the Seventh International Conference on Architectural Support for Programming Languages and Operating Systems*, pages 222-233, October 1996.
68. Edouard Bugnion, Jennifer M. Anderson, Todd C. Mowry, Mendel Rosenblum and Monica S. Lam. Compiler- Directed Page Coloring for Multiprocessors. In *Proceedings of the Seventh International Conference on Architectural Support for Programming Languages and Operating Systems*, pages 244-255, October 1996.
69. Mark Horowitz, Margaret Martonosi, Todd C. Mowry, and Michael D. Smith. Informing Memory Operations: Providing Memory Performance Feedback in Modern Processors. In *Proceedings of the 23rd Annual International Symposium on Computer Architecture*, pages 260-270, May 1996.
70. Todd C. Mowry, Monica S. Lam and Anoop Gupta. Design and Evaluation of a Compiler Algorithm for Prefetching. In *Proceedings of the Fifth International Conference on Architectural Support for Programming Languages and Operating Systems*, pages 62-73, October 1992.

71. Anoop Gupta, John Hennessy, Kourosh Gharachorloo, Todd Mowry, and Wolf-Dietrich Weber. Comparative Evaluation of Latency Reducing and Tolerating Techniques. In *Proceedings of the 18th Annual International Symposium on Computer Architecture*, pages 254-263, May 1991.
72. Anoop Gupta, Wolf-Dietrich Weber, and Todd Mowry. Reducing Memory and Traffic Requirements for Scalable Directory-Based Cache Coherence Schemes. In *Proceedings of International Conference on Parallel Processing*, pages 312-321, August 1990.

Technical Reports

- Evangelos Vlachos, Michelle Goodstein, Michael Kozuch, Shimin Chen, Babak Falsafi, Phillip B. Gibbons, Todd C. Mowry, and Olatunji Ruwase. Parallel LBA: Conherence-based Parallel Monitoring of Multithreaded Applications. Carnegie Mellon University Technical Report CMU-CS-09-108, March 2009.
- Michelle Goodstein, Evangelos Vlachos, Shimin Chen, Phillip Gibbons, Michael Kozuch, and Todd C. Mowry. The Butterfly Model: Theoretical Foundations. Carnegie Mellon University Technical Report CMU-CS-08-170, February 2009.
- Amit Manjhi, Anastassia Ailamaki, Bruce M. Maggs, Todd C. Mowry, Christopher Olston, and Anthony Tomasic. Simultaneous Scalability and Security for Data-Intensive Web Applications. Carnegie Mellon University Technical Report CMU-CS-06-116, March 2006.
- Charles Garrod, Amit Manjhi, Anastassia Ailamaki, Phillip Gibbons, Bruce Maggs, Todd Mowry, Christopher Olston, and Anthony Tomasic. Scalable Consistency Management for Web Database Caches. Carnegie Mellon University Technical Report CMU-CS-06-128, July 2006.
- Amit Manjhi, Phillip B. Gibbons, Anastassia Ailamaki, Charles Garrod, Bruce M. Maggs, Todd C. Mowry, Christopher Olston, Anthony Tomasic, and Haifeng Yu. Invalidation Clues for Database Scalability Services. Carnegie Mellon University Technical Report CMU-CS-06-139, July 2006.
- Christopher B. Colohan, Anastassia Ailamaki, J. Gregory Steffan, and Todd C. Mowry. Supporting Large Speculative Threads for Databases and Beyond. Carnegie Mellon University Technical Report CMU-CS-05-109, July 2005.
- Christopher B. Colohan, Anastassia Ailamaki, J. Gregory Steffan, and Todd C. Mowry. Optimistic Intra-Transaction Parallelism on Chip Multiprocessors. Carnegie Mellon University Technical Report CMU-CS-05-118, March 2005.
- Shimin Chen, Anastassia Ailamaki, Phillip B. Gibbons, and Todd C. Mowry. Improving Hash Join Performance through Prefetching. Carnegie Mellon University Technical Report CMU-CS-03-157, October 2003.
- Shimin Chen, Phillip B. Gibbons, Todd C. Mowry, and Gary Valentin. Fractal Prefetching B+-Trees: Optimizing Both Cache and Disk Performance. Carnegie Mellon University Technical Report CMU-CS-02-115, March 2002.
- Spiros Papadimitriou and Todd C. Mowry. Exploring Thread-Level Speculation in Software: The Effects of Memory Access Tracking Granularity. Carnegie Mellon University Technical Report CMU-CS-01-145, July 2001.
- Shimin Chen, Phillip B. Gibbons and Todd C. Mowry. Improving Index Performance through Prefetching. Carnegie Mellon University Technical Report CMU-CS-00-177, December 2000.
- J. Gregory Steffan, Christopher B. Colohan and Todd C. Mowry. Extending Cache Coherence to Support Thread-Level Data Speculation on a Single Chip and Beyond. Carnegie Mellon University Technical Report CMU-CS-98-171, December 1998.
- Todd C. Mowry and Sherwyn R. Ramkisson. Software-Controlled Multithreading Using Informing Memory Operations. Carnegie Mellon University Technical Report CMU-CS-98-169, October 1998.

- Chi-Keung Luk and Todd C. Mowry. Compiler and Hardware Support for Automatic Instruction Prefetching: A Cooperative Approach. Carnegie Mellon University Technical Report CMU-CS-98-140, June 1998.
- J. Gregory Steffan, Christopher B. Colohan and Todd C. Mowry. Architectural Support for Thread-Level Data Speculation. Carnegie Mellon University Technical Report CMU-CS-97-188, November 1997.
- Todd C. Mowry and Chi-Keung Luk. Predicting Data Cache Misses in Non-Numeric Applications Through Correlation Profiling. Carnegie Mellon University Technical Report CMU-CS-97-175, September 1997.
- J. Gregory Steffan and Todd C. Mowry. The Potential for Thread-Level Data Speculation in Tightly-Coupled Multiprocessors. University of Toronto Technical Report CSRI-TR-350, February 1997.
- Chi-Keung Luk and Todd C. Mowry. Predicting Data Cache Misses in Non-Numeric Applications Through Correlation Profiling. University of Toronto Technical Report CSRI-TR-359, February, 1997.
- Mark Horowitz, Margaret Martonosi, Todd C. Mowry, and Michael D. Smith. Informing Loads: Enabling Software to Observe and React to Memory Behavior. Stanford University Technical Report CSL-TR-95-673, July 1995.
- Todd C. Mowry. Tolerating Latency Through Software-Controlled Data Prefetching. Technical Report CSL-TR-94-628, Stanford University, June 1994.

Patents Held

- Todd C. Mowry. U.S. Patent 7,127,586: Prefetching hints. Issued October, 2006.
- Shimin Chen, Phillip B. Gibbons, and Todd C. Mowry. U.S. Patent 6,772,179: System and method for improving index performance through prefetching. Issued August, 2004.
- Todd C. Mowry. U.S. Patent 6,240,488: Prefetching hints. Issued May, 2001.
- Todd C. Mowry. U.S. Patent 5,732,242: Consistently specifying way destinations through prefetching hints. Issued March, 1998.
- Todd C. Mowry and Earl A. Killian. U.S. Patent 5,696,958: Method and apparatus for reducing delays following the execution of a branch instruction in an instruction pipeline. Issued December, 1997.

Distinguished Lectures

- 9/25/13: *“The Log-Based Architectures Project: Exploiting Multicore to Help Parallel Software Behave Correctly”*, University of Wisconsin-Madison, Department of Computer Science, Distinguished Lecture Series.
- 2/7/08: *“Pario: the Next Step Beyond Audio and Video”*, University of Toronto, Department of Electrical and Computer Engineering Distinguished Lecture Series.

Professional Activities

Professional Societies

- Member of the Institute of Electrical and Electronics Engineers (IEEE)
- Member of the Association of Computing Machinery (ACM)

Industrial Employment

5/04–6/07: Intel Corporation, Pittsburgh, Pennsylvania: Director of the Intel Research Pittsburgh Lab.

6/89–11/93: Silicon Graphics, Inc., Mountain View, California (formerly MIPS Computer Systems, Sunnyvale California): Computer Architect (part-time).

Consulting

7/07–2/11: Intel Corporation, Pittsburgh, Pennsylvania: Research Advisor.

8/96–4/04: SandCraft, Inc., Santa Clara, California: Member of the Technical Advisory Board.

1/96–4/04: IBM, Toronto: Visiting Scientist.

12/93–12/96: Silicon Graphics, Inc., Mountain View, California: Computer Architecture Consultant.

University Committee Work and Other Service Activities

Committee Work

- Faculty Hiring Committees:
 - Computer Science Department: 2008, 2010.
 - Computer Systems Area: 2000, 2001, 2002, 2003, 2004.
- Chair of the Computer Science Department Head Selection Committee, 2007.
- CSD Ph.D. Admissions Committee, 2001, 2002, 2003.
- School of Computer Science Council, 2001-2004.
- Carnegie Mellon University Faculty Senate (Presidential Appointee), 2000-2001.
- Computer Systems Area Advocate, 2000-2004.
- Doctoral Review Committee, 1998-2004.
- ACM Thesis Award Nomination Committee, 1999.

Journal, Conference and Workshop Organization

- Editor-in-Chief, ACM Transactions on Computer Systems, 2013-2018.
- Associate Editor, ACM Transactions on Computer Systems, 2001-2013.
- Program Chair of the International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS), 2011.
- Sponsorship Chair, International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS), 2010.
- Member of the ISCA program committee, 1998, 2000, and 2011.
- Member of the ASPLOS program committee, 2000, 2004.
- Member of the ASPLOS external review committee, 2012.
- Co-Program Chair of the International Conference on Parallel Architectures and Compilation Techniques (PACT), 2001.
- Member of the International Symposium on Microarchitecture program committee, 1998.
- Member of the Workshop on Architectural and System Support for Improving Software Dependability (ASID) program committee, 2005.
- Member of the Workshop on Memory System Performance (MSP) program committee (in conjunction with PLDI), 2002 and 2004.
- Member of the IBM CASCON program committee, 1995, 1996, 1997.
- Member of the First SUIF Workshop Program Committee, 1996.
- Organized the CASCON 97 workshop on Compiler Optimization, 1997.
- Initiated and organized the 1st Toronto / Rochester Workshop on Shared-Memory Multiprocessors, 1994.