

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

M&A VENTURES, LLC, REPAY HOLDINGS CORPORATION, and REPAY
HOLDINGS, LLC
Petitioners,

v.

AUTOSCRIBE CORPORATION,
Patent Owner.

IPR2024-01159
U.S. Patent No. 11,620,621

**DECLARATION OF STEPHEN CRAIG MOTT,
UNDER 37 C.F.R. § 1.68 IN SUPPORT OF PETITION FOR
INTER PARTES REVIEW**

TABLE OF CONTENTS

I.	Introduction.....	5
II.	Qualifications and Professional Experience.....	7
III.	Level of Ordinary Skill in the Art	16
IV.	Relevant Legal Standards	17
V.	Overview of the '621 Patent.....	19
A.	Summary of Prosecution of the '621 Patent	25
VI.	Claim Construction.....	30
A.	<i>“providing a non-sensitive electronic data token representing the sensitive financial account information to the merchant server... without providing the non-sensitive electric data token to the payer”</i>	30
1.	Structure of the Claims.....	31
2.	Prosecution History of the '621 Patent	32
3.	Specification of the '621 patent	35
B.	<i>“receiving the non-sensitive electronic data token from the secure server, wherein...the secure server does not provide the non-sensitive electronic data token to the payer”</i>	37
1.	Structure of the Claims.....	38
2.	Prosecution History of the '621 Patent	40
3.	Specification of the '621 patent	42
C.	<i>“generating a uniform resource locator (URL),... the URL comprising either: a dynamic URL generated by the secure server...; or a static URL and a hypertext transport protocol (HTTP) parameter...”</i>	43
1.	Claim Language	44

2.	Specification of the '621 Patent	45
VII.	Identification of how the Claims are Unpatentable	52
A.	Ground 1: Claims 1, 2, 3, 4, 5, 6, 7, 15, 16, 17, 18, 19, 20, 21, 22, 25, 26, and 27 are obvious under 35 U.S.C. § 103 over PayPal Express Checkout.....	53
1.	Reasons to Combine the documents describing express checkout into PayPal Express Checkout	53
2.	Summary of PayPal Express Checkout	65
3.	Claim 1.....	71
4.	Claim 2.....	149
5.	Claim 3.....	153
6.	Claim 4.....	153
7.	Claim 5.....	154
8.	Claim 6.....	156
9.	Claim 7.....	165
10.	Claim 15.....	168
11.	Claim 16.....	177
12.	Claim 17.....	178
13.	Claim 18.....	178
14.	Claim 19.....	178
15.	Claim 20.....	179
16.	Claim 21.....	179
17.	Claim 22.....	182
18.	Claim 25.....	182

19.	Claim 26.....	185
20.	Claim 27.....	186
B.	Ground 2: Claims 8, 9, 10, 11, 12, 13, 14, 23, and 24 obvious over PayPal Express Checkout and Schlessner.....	186
1.	Summary of the Schlessner	186
2.	Reasons to Combine Schlessner with PayPal Express Checkout	189
3.	Claim 8.....	194
4.	Claim 9.....	199
5.	Claim 10.....	200
6.	Claim 11.....	200
7.	Claim 12.....	200
8.	Claim 13.....	200
9.	Claim 14.....	201
10.	Claim 23.....	201
11.	Claim 24.....	205
VIII.	Conclusion	206

I, Stephen C. Mott, do hereby declare as follows:

I. INTRODUCTION

1. I am making this declaration at the request of M&A Ventures, LLC, Repay Holdings Corporation, and Repay Holdings, LLC (“Repay” or “Petitioner”) in the matter of the *Inter Partes* Review of U.S. Patent No. 11,620,621 (“the ’621 patent”) to Pollin *et al.*, which claims priority to June 5, 2012 at the earliest.

2. I am also being reimbursed for reasonable and customary expenses associated with my work and testimony in this investigation. My compensation is not contingent on the outcome of this matter or the specifics of my testimony, and I have no other interest in this case or the parties thereto.

3. I have been asked to provide my opinions regarding whether claims 1-27 (“the Challenged Claims”) of the ’621 patent are unpatentable as they would have been obvious to a person having ordinary skill in the art (“POSITA”) at the time of the alleged invention, in light of the prior art. It is my opinion that all of the limitations of the challenged claims would have been obvious to a POSITA.

4. In the preparation of this declaration, I have studied:

- a. the ’621 patent, Ex.1001;
- b. the prosecution history of the ’621 patent (“’621 File History”),

Ex.1002;

- c. the provisional application to which the '621 patent claims priority (U.S. Provisional Application No. 61/655,482 to Pollin *et al.*), Ex.1005;
- d. documents related to PayPal's Express Checkout product, including PayPal Express Checkout Integration Guide (Ex.1006), PayPal SOAP API Developer Reference (Ex.1007), PayPal Sandbox User Guide (Ex.1008), PayPal Preparing for the New Updated Checkout Experience (Ex.1009), PayPal Express Checkout User Interface Standards (Ex.1010), PayPal Privacy Policy (Ex.1011), PayPal Digital Goods Integration Guide (Ex.1012), PayPal Instant Payment Notification Guide (Ex.1023), PayPal Name-Value Pair API Developer Guide (Ex.1024), and PayPal Merchant Setup and Administration Guide (Ex.1025) (collectively "PayPal Express Checkout");
- e. U.S. Patent Publication No. 2010/0094755 to Kloster ("Kloster"), Ex.1013;
- f. *HTML & Web Design: Tips & Techniques*, Jamsa *et al.* (2002) Ex.1014;
- g. U.S. Patent No. 9,830,596 to Collison *et al.* ("Collison"), Ex.1015;
- h. U.S. Patent Publication No. 2010/0327054 to Hammand ("Hammand"), Ex.1016;
- i. U.S. Patent Publication No. 2012/0303516 to Fellig *et al.* ("Fellig"), Ex.1017;

- j. U.S. Patent Publication No. 2007/0094100 to Hodson *et al.*
(“Hodson”), Ex.1018;
- k. *Dynamic URLs vs. static URLs*, Google Search Central Blog, Google
for Developers (Sept. 22, 2008), Ex.1019;
- l. U.S. Patent Publication No. 20120197807 to Schlesser *et al.*
(“Schleser”), Ex.1020;
- m. PayPal Technical Documentation, Ex.1021; and
- n. PayPal Home Page, Ex.1022.

5. In forming the opinions expressed below, I have considered:

the documents listed above;

the relevant legal standards, including the standard for obviousness,
and any additional authoritative documents as cited in the body of this
declaration; and

my own knowledge and experience based upon my work in the field
of electronic payment systems as described below, as well as the following
materials.

6. Unless otherwise noted, all emphasis in any quoted material has been
added.

II. QUALIFICATIONS AND PROFESSIONAL EXPERIENCE

7. My complete qualifications and professional experience are described

in my *Curriculum Vitae*, a copy of which can be found in Exhibit 1004. The following is a brief summary of my relevant qualifications and professional experience.

8. I have over 40 years of experience working in the electronic payments and security industry. Currently, I am the Principal and President of BetterBuyDesign, LLC, a consulting firm for the electronic payments transaction business headquartered in Stamford, Connecticut. I served in a similar role for BetterBuyDesign's predecessor company, CSI Management Services, Inc.

9. After serving for three years in the U.S. Marine Corps as a technician on the HAWK Missile System, where I graduated from the Army's 8-month Electronics School, I attended the University of California at Berkeley, from which I received a B.A. in Economics and a B.A. in Journalism in 1974. In 1979, I received an M.B.A. from Harvard Business School.

10. After earning my M.B.A., I worked for McKinsey & Co. as an engagement manager, where I managed strategy consulting engagements for various companies, which typically included electronic technology assessments. In 1982, I was hired by McGraw-Hill, one of my former McKinsey clients, to implement a comprehensive corporate strategy for moving physically-based vertical market offerings into electronic format offerings.

11. From 1984-1987, I was the Vice President for Sales and Marketing at

MCI International, Inc., heading sales, marketing and product development for the data communication division. While in this position, I helped design and develop MCI's global, digital network service, which became a flagship offering in MCI's private line telecommunication service offerings to financial services firms. As a common carrier for consumer and business telecommunications services, MCI pioneered alternative international dialing and phone card services. Part of my portfolio was identification and incorporation of new digital technology services, including automated and monitoring wire transfer services and communications protocols enabling PCs as telex and email messaging systems.

12. From 1987-1994, I was President and CEO of Cognitive Systems, Inc., a pioneering artificial intelligence software development firm specializing in automating financial transacting and decision support. My activities included supervising development of a data mining software program utilized in more than 200 innovative data management applications and a network processing and data manipulation program used by U.S. intelligence agencies to detect and track bank payments made by terrorists and drug dealers. The latter program used early-stage artificial intelligence in the form of natural language processing technology in the front-end to read, interpret, and sort textual information to feed an induction engine in the back-end; the induction engine determined which payment messages had a significant potential for fraud based on recognizing patterns developed from the

textual inputs from both free-form and field entries. From 1994-1995, after Ingenia S.A. purchased most of Cognitive Systems' business, I served as President of Ingenia S.A.'s North American operations (Ingenia, Inc.), where I was responsible for the development, sale and installation of software that helped banks comply with federal anti-money laundering regulations (i.e., OFAC, AML, BSA).

13. From 1995-1998, I served as a Senior Vice President of Electronic Commerce/New Ventures for MasterCard International, Inc. During this period, I led efforts to deploy security systems for credit card transactions and to integrate smart card use for both business-to-business and business-to-consumer Internet commerce applications. I also helped launch the SET (Secure Electronic Transaction) protocol for credit card security for online payment transactions. SET was a consolidated effort of MasterCard, Visa, Microsoft, Netscape, IBM, RSA Security and more than a dozen other companies interested in securing online commerce. I developed strategies for the safe use of debit cards, EMV smart cards, and B2B procurement cards by banks, merchants, and consumers online. During this period, I was named as one of the Top 25 "Unsung Heroes of the Internet" by Inter@ctive Week (later eWeek) in December 1997.

14. In early 1998, I left MasterCard to pursue a career in management consulting, speaking, and writing about trends in electronic payments and security. For example, my first client was Citibank, which I advised on its eCiti initiative.

In recognition of my work in the industry, I was named one of the “Top 25 Most Intriguing Minds for the New Economy” by Business 2.0 magazine in July 1998.

15. From 1999-2000, I was hired by priceline.com, serving successively as Chief Strategist, SVP Operations, EVP Business Development, and finally President at Priceline WebHouse Club, leading to the launch of priceline.com’s “name your own price” concept for shopping for grocery and gasoline on the Internet with pickup in physical stores and stations. As part of this project, I designed and helped develop a unique debit card payment processing and settlement system managing online purchases with off-line fulfillment. Besides managing an elaborate front-end fraud-prevention and back-end charge-back handling system for purchases of groceries and gas online (via credit cards), we developed a highly innovative digital marketing capability making and increasing offers to customers in-session. I also helped design a GPS-based transaction system using Palm Pilots and other early-stage mobile devices for ordering gas, food, and hotel rooms while traveling on the Interstate system.

16. In my current capacity as Principal and President BetterBuyDesign, LLC, I have consulted on a wide variety of projects involving payment systems. My first client after leaving priceline.com was RSA security. I helped RSA Security evaluate which security technologies were likely to achieve the best balance between the eCommerce transaction and data security and usability. I also

assisted STAR Network, a unit of First Data Corp. (n/k/a Fiserv), assess and build technologies for chip-based security for mobile and contactless point-of-sale (“POS”) and eCommerce applications for nearly two decades.

17. Since 2005, I have worked on nearly three dozen legal cases—all involving payments or security and computer software. I served as an expert witness for a U.S. Patent Trial and Appeal Board (PTAB) *inter partes* review (IPR) of patents in the electronic payments space. This work sustained a bank’s infringement in cases that later lead to judgments totaling \$400 million. I also consulted on “Opt-Out-Merchant” appeal of a \$5.2 billion interchange suit settlement proposed in 2012 that has helped generate inroads into more ecosystem-friendly policies and rules. As part of my consulting efforts on that case, I provided research on Honor All Cards (HAC) rule impacts, EMV deficiencies, and market constraints from card network programs and rules on eCommerce. I also consulted on matters before the U.S. Tax Court regarding Section 199 deductions for three payments companies. One of these companies was the first to get the IRS to agree to a favorable settlement on this program. Moreover, I authored key expert reports regarding the actual costs of debit card transactions and assessing the history and promulgation of network rules that suppress innovation in support of the Federal Reserve’s mandate to implement the Durbin Amendment legislation aimed at regulating debit card network access and interchange rates. I provided defensive

programs to three large merchants to help them offset PCI Compliance challenges and contracting for new vendors. This worked helped one merchant avoid \$80 million in potential network penalties. I assisted in several payments and security companies in pursuing IPO and capitalization efforts. In this capacity, I evaluated business plans and developed strategies for six emerging technology firms that succeeded in obtaining funding for entry into the payments arena.

18. With respect to the types of work that I have done in electronic payment systems, I have worked in a number of capacities—all entailing my services as evaluator and designer of functionality, as well as assistance with development of technical specifications for emerging industry standards. This work was provided for both clients and in serving as a subject matter expert for a number of industry groups and agencies, including, for example, Bankers Administration Institute (BAI), National Automated Clearing House Association (NACHA), the Financial Services Technology Consortium (FSTC), Financial Services Information Sharing Analysis Center (FS-ISAC), and the Federal Reserve System.

19. Beginning in 2004, I defined and articulated the marketplace for secure real-time debit account payments and worked as a subject matter expert for NACHA on its embryonic product offerings, which included Secure Vault and today's Same-Day ACH/Settlement. I also served on NACHA's Internet Council

for nearly a decade.

20. As another example, between 2008-2010, I co-managed an industry collaborative project for FSTC to define if and how mobile payments could be secured for safe commercial operations across the banking system.

21. In another example, from 2010 to 2021, I also served as a subject matter expert and resource for the Federal Reserve's Mobile Payments Industry Workgroup (MPIW). In this capacity, I co-authored and assisted with white papers on tokenization, card-not-present security, and authentication technologies for electronic and mobile commerce. I also developed the foundation for coordinated and facilitated group session on QR Codes and payment players from China (including Ali Pay, WeChat Pay, China Union Pay), Apple Pay and mobile wallets, and tokenization.

22. Between 2010 and 2012, I was the lead consultant in defining a new mobile payments wallet system for the Merchant Customer Exchange ("MCX"), which was a collaborative effort of dozens of the largest merchants in the U.S. seeking to create secure and efficient payments with mobile devices at both physical POS and online. In this capacity, I defined the marketplace strategy and the technology options, including QR codes, use of the ACH network, loyalty programs, and tokenization.

23. In 2015, I was elected to the Secure Payments Task Force Steering

Committee where I served as chair of the Standards Assessment Team, as a member of the Faster Payments Task Force, and as a founding member of the Faster Payments Council (which succeeded the task forces). I was then elected to the Boards of Directors of the Faster Payment Council, an industry coalition that sought to quicken the pace for the U.S. to achieve secure, real-time payments, where I served two terms.

24. In other roles supporting industry standards for secure electronic transacting, from 2009-2018, I served on the Federal Reserve's Consumer Payments Research Center (CPRC) Advisory Group, which provided detailed views and fielded comprehensive surveys on how consumers used payment products and regarded security protocols.

25. In the telecommunications sector, I researched and authored CTIA's initial white paper on best practices in security and privacy in mobile transacting as an embryonic set of industry standards. I have defended AT&T, Apple, and a very large bank on lawsuits involving SIM card swaps and consumer scams.

26. Over the past two decades, I have been a speaker at up to two dozen industry conferences a year and an author of more than 100 articles in the industry media regarding a wide array of industry developments and topics.

27. While not formally trained in computer science or programming, nearly all of these activities over the course of my career have involved the

development, evaluation, and marketing of computer programs.

III. LEVEL OF ORDINARY SKILL IN THE ART

28. I understand there are multiple factors relevant to determining the level of ordinary skill in the pertinent art, including (1) the levels of education and experience of persons working in the field at the time of the invention; (2) the sophistication of the technology; (3) the types of problems encountered in the field; and (4) the prior art solutions to those problems.

29. It is my understanding that the earliest possible priority date for the '621 patent is June 5, 2012. A POSITA in the field of the '621 patent, as of June 5, 2012, would have been someone having a working knowledge of payment processing systems pertinent to the '621 patent. That person would have a bachelor's degree related to the field of payment processing systems and at least two years working in that field. In this context, a perceived lack of education can be remedied by additional work experience, and vice versa.

30. I meet the definition of a POSITA because of my education and extensive experience in the field of electronic payments. *See supra* ¶¶7-27. For purposes of this Declaration, in general, and unless otherwise noted, my statements and opinions, such as those regarding my experience and the understanding of a POSITA generally (and specifically related to the references I consulted herein), reflect the knowledge that existed in the field as of the alleged priority date of the

'621 patent (i.e., June 5, 2012). Unless otherwise stated, when I provide my understanding and analysis below, it is consistent with the level of a POSITA as of the alleged priority date of the '621 patent.

IV. RELEVANT LEGAL STANDARDS

31. I am not an attorney. In preparing and expressing my opinions and considering the subject matter of the '621 patent, I am relying on certain basic legal principles that counsel have explained to me. These principles are discussed below.

32. I understand that prior art to the '621 patent includes patents and printed publications in the relevant art that predate the priority date of the alleged invention recited in the '621 patent. For the purposes of this declaration, I am applying June 5, 2012 as the priority date of the '621 patent ("priority date").

33. I have been informed that a claimed invention is unpatentable under 35 U.S.C. § 103 if the differences between the invention and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which the subject matter pertains. I have also been informed by counsel that the obviousness analysis takes into account factual inquiries including the level of ordinary skill in the art, the scope and content of the prior art, and the differences between the prior art and the claimed subject matter.

34. I further understand that certain factors may support or rebut the obviousness of a claim. I understand that such secondary considerations include, among other things, commercial success of the patented invention, skepticism of those having ordinary skill in the art at the time of invention, unexpected results of the invention, any long-felt but unsolved need in the art that was satisfied by the alleged invention, the failure of others to make the alleged invention, praise of the alleged invention by those having ordinary skill in the art, and copying of the alleged invention by others in the field. I understand that there must be a nexus—a connection—between any such secondary considerations and the alleged invention recited in the claims. I also understand that contemporaneous and independent invention by others is a secondary consideration tending to show obviousness.

35. I am not aware of any allegations by the named inventor of the '621 patent or any assignee of the '621 patent that any secondary considerations are relevant to the obviousness analysis of any Challenged Claim of the '621 patent.

36. I have been informed by counsel that the Supreme Court has recognized several rationales for combining references or modifying a reference to show obviousness of claimed subject matter. Some of these rationales include the following: (a) combining prior art elements according to known methods to yield predictable results; (b) simple substitution of one known element for another to obtain predictable results; (c) use of a known technique to improve a similar device

(method, or product) in the same way; (d) applying a known technique to a known device (method, or product) ready for improvement to yield predictable results; (e) choosing from a finite number of identified, predictable solutions, with a reasonable expectation of success; and (f) some teaching, suggestion, or motivation in the prior art that would have led one of ordinary skill to modify the prior art reference or to combine prior art reference teachings to arrive at the claimed invention.

V. OVERVIEW OF THE '621 PATENT

37. The '621 patent was originally filed as U.S. Application No. 16/535,424 (“the '424 application”) on August 8, 2019. I have been informed that the '424 application is related to several earlier non-provisional applications, the earliest of which was U.S. Application No. 13/679,545, which was filed on November 16, 2012, and which is now U.S. Patent 9,576,279. I have also been informed that the '424 application claims priority to two provisional applications, the earliest of which was U.S. Provisional Application No. 61/655,482 (Ex.1005), which was filed on June 5, 2012.

38. The '621 patent relates to “systems and methods for obtaining and using account information to process financial payments.” Ex.1001, 1:18-20. The '621 patent explains that “[f]raud in credit card and other financial transactions is a major problem.” Ex.1001, 1:24-25. Thus, the Payment Card Industry Security

Standards Council defined a payment card industry (PCI) standard “to increase controls around cardholder data to reduce credit card fraud.” Ex.1001, 1:35-36. However, “the time, effort, and cost required for merchants and processors to obtain PCI certification to receive and use credit card data are substantial.” Ex.1001, 1:45-48. Thus, “[i]n one approach to managing these costs, credit card processing organizations have used tokenization to prove a link to credit card data for purposes of storage and retrieval.” Ex.1001, 1:49-52.

39. “Tokenization is the process of replacing some piece of sensitive data with a value that is not considered sensitive outside the environment where it is stored and used.” Ex.1001, 1:52-55. By using a “symbolic representation of a financial instrument or instruction that is only meaningful to participants in the processing cycle,” tokenization “safeguards the permissible use of and access to the financial instrument to authorized users.” Ex.1001, 1:55-58. The “tokens are used to reference cardholder data that is stored in a separate database, application or off-site secure facility that is PCI compliant to the appropriate level.” Ex.1001, 1:63-66. “Therefore, the token, which is non-sensitive data, can be stored and used in a wide range of systems in the organization, without bringing those systems within the scope of a higher-level PCI audit and more stringent compliance requirements.” Ex.1001, 1:66-2:3.

40. The ’621 patent acknowledges that “[i]n the payments industry,

tokenization ha[d] become a popular means of bolstering the security of electronic transactions while minimizing the complexity of compliance with industry standards and best practices.” Ex.1001, 1:59-62.

41. The ’621 patent describes a tokenization process in which “merchants are able to process such transactions without transmitting, processing or storing Financial Account data,” thereby “reduc[ing] the potential liability of the merchants for data loss, and avoid[ing] placing their customer tracking and payment processing systems...within the scope of the present Payment Card Industry Data Security Standard (PCI-DSS).” Ex.1001, 5:62-6:2. The “[m]erchant server 204 may store customer identification information, purchase history, and other data that does not include financial account information such as credit card information, without being subject to PCI compliance requirements.” Ex.1001, 7:8-12; *see also* 5:62-6:4 (“[M]erchants are able to process such transactions without transmitting, processing or storing Financial Account data. This...avoids placing their customer tracking and payment processing systems (such as merchant server 204) within the scope of the present Payment Card Industry Data Security Standard (PCI-DSS).”). A “[s]ecure server 202...stores the financial account information received from the customer.” Ex.1001, 7:31-34. The merchant server, secure server, and payer (i.e. customer or buyer) may communicate with each other via the internet. Ex.1001, 6:47-53 (“The secure server 202 and merchant server 204 are

also provided with Internet web server software allowing interactive exchange of information with customer sites 206, 208 and 210 via the World Wide Web.

Computer equipment at customer sites 206, 208 and 210 are preferably provided with standard web browser software to facilitate interaction with secure server 202 and merchant server 204.”). Fig. 2 below illustrates the communication links

between the secure server 202, the merchant server 204, and the customer 208:

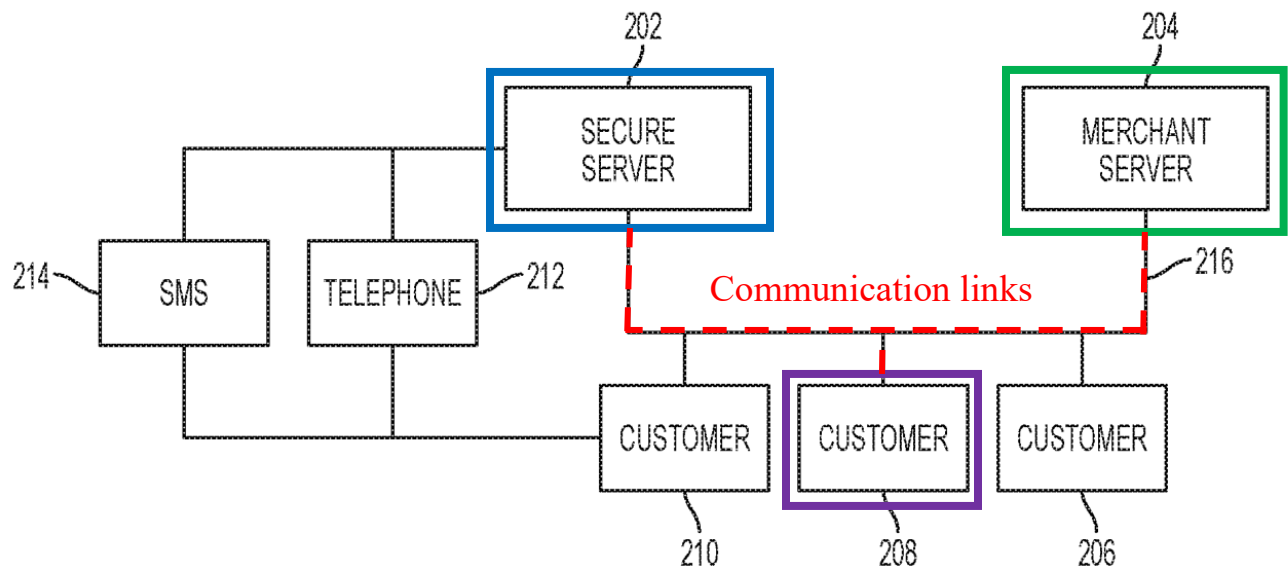


FIG. 2

Ex.1001, Fig. 2 (annotated)

42. The merchant server initiates two separate and distinct functions during the payment process: the financial account registration and the token retrieval function. Ex.1001, 5:57-62 (“A hosted system on a secure server 202

provides Financial Account Registration functions and Token Retrieval functions that can be accessed by merchants and others operating websites, for example using merchant server 204, to process payment transactions.”), 6:63-7:1 (“The financial account registration and token retrieval functions provided in the API can be implemented, in a first example, within the merchant's website hosted on merchant server 204, in a “widget” or frame, using either static or dynamic URL access to secure server 202 (the hosted system that maintains the financial data).”).

43. The payer initiates the payment process by “enter[ing] their identifying information on an Internet website generated by merchant server 204.” Ex.1001, 7:62-64.

44. First, in response to the payer’s request to make a purchase from the merchant, the merchant server initiates the financial account registration function. During the financial account registration function, the merchant server establishes a communication link between the payer and the secure server so that the payer can submit her sensitive financial account information to the secure server to be stored and tokenized. To accomplish this, the merchant server “generates an electronic instruction to a...secure server, to obtain financial account information from the payer.” Ex.1001, 7:66-8:2. The secure server then “communicates with the payer” so that the “payer is preferably able to provide the sensitive financial account information to the [secure server] without this information being visible or audible

to personnel employed by the operator of” the merchant server. Ex.1001, 8:26-29, 8:32-36. To generate a communication link between the payer and the secure server, the “[m]erchant server 204 may provide the customer at site 206, 208 or 210 with a widget or redirected URL that provides an online window for direct, preferably secure communication with the secure server 202 for entry of financial account information.” Ex.1001, 9:55-59.

45. Second, the secure server stores the sensitive financial account information and “a token is generated for the referring payee (e.g. the merchant or operator of the merchant server).” Ex.1001, 8:37-40. “The token is a pointer that allows the payee to reference the financial account information of that payer without possessing the actual financial account information.” Ex.1001, 8:40-43.

46. Third, to process the payment from the payer, the merchant then initiates the token retrieval function. The merchant server “receives the token from the” secure server and “[t]he token is stored in the [merchant server] in association with the electronic enrollment data for the payer.” Ex.1001, 8:44-48. Then, “merchant server 204 submits the token corresponding to the customer account to be debited and information specifying the amount of the payment to the secure server 202.” Ex.1001, 9:12-16. “The operator of the secure server 202 then processes the payment using the financial account pointed to by the token.” Ex.1001, 9:16-19.

A. Summary of Prosecution of the '621 Patent

47. In response to a non-final office action in which the examiner rejected all pending claims under 35 U.S.C. § 101 as being patent ineligible (Ex.1002, 441-474), Patent Owner amended the claims to include several limitations that were ultimately included in the issued claims of the '621 patent. Ex.1002, 270-280. An example of some amendments to pending claim 1 reproduced below:

1. (Currently Amended): A method of processing a payment from a payer to a payee, the method being performed by ~~a secure server~~ one or more secure servers, the method comprising:

providing an application programming interface (API) that:

provides an API endpoint that receives ~~receiving~~ an electronically-generated payment transaction instruction from a merchant server providing a webpage to a payer computing system used by the payer, the payment transaction instruction including at least one data element associated with the payer and specifying the amount of the payment from the payer to the payee;

authenticates the payee;

generates a universal resource locator (URL) for establishing a secure socket layer connection via the internet between the secure server and ~~a computing system used by the payer to receive financial account information provided by the payer~~ the payer computing system, the URL comprising either:

a dynamic URL generated by the secure server for the payer and the payee; or

a static URL and a hypertext transport protocol (HTTP) parameter used by the secure server to identify the payer and the payee;

establishes the secure socket layer connection between the secure server and the payer computing system within a window or frame that is displayed within the webpage provided by the merchant server;

outputs instructions to the payer computing system to render a financial account registration request form, within the window or frame that is displayed within the webpage provided by the merchant server, that provides functionality for the payer to provide financial account information; and

outputs instructions to the payer computing system to encrypt the financial account information provided by the payer and transmit the encrypted financial account information to the secure server, via the secure socket layer connection, without providing the financial account information to the merchant server;

receiving the financial account information provided by the payer via the secure socket layer connection;

storing the financial account information in a secure storage location and performing each software process required to maintain compliance with one or more information security standards;

providing an electronic data token, representing the financial account information provided by the payer, to the merchant server without providing the financial account information to the merchant server and without providing the electronic data token to the payer;

~~storing the financial account information provided by the payer;~~

processing the payment using the financial account information provided by the payer;

and

forwarding at least a portion of the payment to the payee.

Ex.1002, 270-271

48. However, the amendments to the pending claims did not include reference to the financial account or token retrieval functions.

49. After submitting these amendments, Patent Owner requested an interview with the examiner. Ex.1002, 268. The examiner and Patent Owner then communicated back and forth regarding amendments to the claims that would overcome the § 101 rejections via email and phone calls. Ex.1002, 54-238. In these communications, several amendments were proposed by Patent Owner and the examiner. Ex.1002, 54-161. During this process, Patent Owner added limitations regarding the financial account registration and token retrieval functions. Ex.1002, 157-161 (amending the independent claim 1 to recite “a token retrieval function”), 138-140 (amending independent claim 1 to further recite a “financial account registration function” and dividing recited steps of the independent claims as being

performed during the financial account registration function or the token retrieval function), 119-131 (providing a full claim set with amendments similar to independent claim 1 reciting separate and distinct financial account registration and token retrieval functions). Patent Owner and the examiner eventually agreed on the amendments, which were then entered as an examiner's amendment in the Notice of Allowance. Ex.1002, 23-25.

50. Based on the prosecution history, it is my understanding that the financial account registration function and the token retrieval function are separate and distinct functions, with each including a particular set of steps. For example, the financial account registration function includes the steps of “*generating a uniform resource locator (URL), for establishing a secure socket layer connection via the internet between the secure server and the payer computing system, the URL comprising either: a dynamic URL generated by the secure server for the payer and the payee; or a static URL and a hypertext transport protocol (HTTP) parameter used by the secure server to identify the payer and the payee; establishing the secure socket layer connection, in response to an HTTP request received from the merchant server for the generated URL, between the secure server and the payer computing system within a window or frame that is displayed within the webpage provided by the merchant server; outputting instructions to the payer computing system, in response to the HTTP request for the generated URL,*

to render a financial account registration request form, within the window or frame that is displayed within the webpage provided by the merchant server, that provides functionality for the payer to provide sensitive financial account information associated with a financial account; and outputting instructions to the payer computing system, in response to the HTTP request for the generated URL, to encrypt the sensitive financial account information provided by the payer and transmit the encrypted financial account information to the secure server via the secure socket layer connection,” as recited in claim 1. Limitations [1.1.5]-[1.1.5.4].

Additionally, the token retrieval function includes *“providing a non-sensitive electronic data token representing the sensitive financial account information to the merchant server without providing the sensitive financial account information to the merchant server and without providing the non-sensitive electronic data token to the payer; and processing the payment transaction using the sensitive financial account information by generating and transmitting an electronic request requesting the payment amount from the financial account, obtaining the payment amount, and forwarding at least a portion of the payment amount to the payee,”* as recited in claim 1. Limitations [1.4]-[1.4.2]. It is my understanding that these amendments delineating the financial account registration and token retrieval functions were added to the claims in order to overcome the examiner’s § 101 rejections.

VI. CLAIM CONSTRUCTION

51. It is my understanding that in order to properly evaluate the '621 patent, the terms of the claims must first be interpreted. It is my understanding that for the purposes of this *inter partes* review, the claims are to be construed under the so-called *Phillips* standard, under which claim terms are given their ordinary and customary meaning as would be understood by one of ordinary skill in the art in light of the specification and prosecution history, unless the inventor has set forth a special meaning for a term. For the purposes of this declaration, it is my opinion that the claim terms should be given their plain and ordinary meaning, other than the following:

A. “*providing a non-sensitive electronic data token representing the sensitive financial account information to the merchant server... without providing the non-sensitive electric data token to the payer*”

52. Based on my review of the structure of the claims as well as the specification and the prosecution history, it is my opinion that the limitation “*providing a non-sensitive data token to the merchant... without providing the non-sensitive electronic data token to the payer*” should be construed as “*during execution of the token retrieval function, providing a non-sensitive electronic data token representing the sensitive financial account information to the merchant ... without providing the non-sensitive electronic data token to the payer.*” In short, this limitation does not mean that the secure server must never provide the token to

the payer. Instead, the claim language only requires that, *during the token retrieval function*, the secure server does not provide the token to the payer when it provides the token to the merchant. This limitation appears in three claims: limitation [1.4.1] in claim 1, limitation [8.2.1.8.1] in claim 8, and limitation [25.4.1] in claim 25.

The below claim construction analysis focuses on claim 1, but the same analysis applies to claims 8 and 25 as well.

1. Structure of the Claims

53. The claims clearly divide the payment process into two discrete functions: (1) the financial account registration function and (2) the token retrieval function. Limitation [1.1.1] (“*provides financial account registration and token retrieval functions that can be executed to process the payment transaction*”), limitation [1.1.2] (“*provides access to the financial account registration and token retrieval functions to the merchant server*”). Claim 1 recites “*executes the financial account registration function, upon initiation by the merchant server, by*” (limitation [1.1.5]) performing several steps including “*generating a uniform resource locator (URL)...*” (limitation [1.1.5.1]), “*establishing the secure socket layer connection... between the secure server and the payer computing system...*” (limitation [1.1.5.2]), “*outputting instructions to the payer computing system... to render a financial account registration request form...*” (limitation [1.1.5.3]), and “*outputting instructions to the payer computing system...to encrypt the sensitive*

financial account information...” (limitation [1.1.5.4]).

54. However, the limitation “*providing a non-sensitive data token to the merchant...without providing the non-sensitive electronic data token to the payer...*” (limitation [1.4.1]) is performed as a part of “*executing a token retrieval function, upon initiation by the merchant server via the API, by*” (limitation [1.4]). Thus, as specifically delineated in the claim, providing the token “*without providing the non-sensitive electronic data token to the payer*” is performed during the token retrieval function, not during any other part of the process.

2. Prosecution History of the '621 Patent

55. The prosecution history of the '621 patent supports this understanding as well. After responding to the examiner's § 101 rejection of the pending claims, Patent Owner and the examiner communicated extensively back and forth (via emails and phone calls) during which the parties iterated through several versions of the amendments to overcome § 101. Ex.1002, 54-238. During this process, Patent Owner made amendments reciting separate financial account registration and token retrieval functions and grouping several steps of the payment process under one function or the other, as shown below:

1. (Proposed Amendment): A method of processing a payment transaction from a payer to a payee, the method being performed by one or more secure servers, the method comprising: providing, by the one or more secure servers, an application programming interface (API) that:

provides a ~~financial account registration and token retrieval function~~ that can be executed to process the payment transaction;

Commented [JRK1]: [0028] and [0032]

provides access to ~~and execution of the~~ financial account registration and token retrieval function; to a merchant server providing a webpage to a payer computing system used by the payer to ~~generate a payment transaction instruction that includes~~

Commented [JRK2]: moved down

~~receives, from the merchant server via the API, at least one data element~~ associated with the payer and ~~specifies the amount of the payment a payment amount from the~~ payer to the payee;

Commented [JRK3]: If it's important to recite that the secure server "generates a payment transaction instruction", I think it'd be more accurate to recite that step further down.

authenticates the payee; and

~~executes the financial account registration function, upon initiation by the~~ merchant server, by:

Commented [JRK4]: the secure server executes the "financial account registration function" (e.g., FIGS. 4a-4g) upon initiation by the merchant server (see the "request" processed at step 376)

~~generating~~ ~~generates~~ a universal resource locator (URL), for establishing a secure socket layer connection via the internet between the secure server and the payer computing system, the URL comprising either:

Commented [JRK5]: Note that this step is part of the financial account registration function executed by the secure server(s)

a dynamic URL generated by the secure server for the payer and the payee; or

a static URL and a hypertext transport protocol (HTTP) parameter used by the secure server to identify the payer and the payee;

~~establishing~~ ~~establishes~~ the secure socket layer connection, in response to ~~an HTTP request for the generated URL,~~ between the secure server and the payer computing system within a window or frame that is displayed within the webpage provided by the merchant server;

Commented [JRK6]: Note that this step is part of the financial account registration function executed by the secure server(s)

Commented [JRK7]: If it's important to you, I think we can specify in this embodiment that the HTTP request is "received from" (or even, "requested by") the merchant server (see, e.g., the request processed at step 406)

~~outputting~~ ~~outputs~~ instructions to the payer computing system, in response to the HTTP request for the generated URL, to render a financial account registration request

Commented [JRK8]: Note that this step is part of the financial account registration function executed by the secure server(s)

form, within the window or frame that is displayed within the webpage provided by the merchant server, that provides functionality for the payer to provide financial account information; and

~~outputting a payment instruction to the payer computing system, in response to the HTTP request for the generated URL, to encrypt the financial account information provided by the payer and transmit the encrypted financial account information to the secure server, via the secure socket layer connection, without providing the financial account information to the merchant server;~~

~~generating the payment transaction instruction using the token retrieval function of the API;~~

~~in response to generating the payment transaction instruction, receiving the financial account information provided by the payer via the secure socket layer connection;~~

storing the financial account information in a secure storage location and performing each software process required to maintain compliance with one or more information security standards;

~~executing a token retrieval function, upon initiation by the merchant server via the API, by:~~

providing an electronic data token, representing the financial account information provided by the payer, to the merchant server without providing the financial account information to the merchant server and without providing the electronic data token to the payer;

and

~~generating a payment transaction instruction, processing to process the payment transaction, using the financial account information provided by the payer; and~~

forwarding at least a portion of the payment amount to the payee.

Commented [JRK9]: Note that this step is part of the financial account registration function executed by the secure server(s)

Commented [JRK10]: below

Commented [JRK11]: The secure server generates the URL, and provides the form (above) and provides the token and processes the payment (below) "in response to" an API call from the merchant server. But the secure server doesn't receive the financial account information in response to anything other than the payer inputting that information into the form

Commented [JRK12]: secure server(s) executes the token retrieval function (see, e.g., FIG. 4f) upon initiation by the merchant server (see, e.g., the request processed at step 646)

Commented [JRK13]: Per claim 20, the payment transaction is generated by the secure server(s) upon initiation of the token retrieval function by the merchant server via of the API

Ex.1002, 138-139 (annotated)

56. By making amendments reciting a separate financial account registration function and token retrieval function and reciting specific steps that fall under each of those functions, Patent Owner made clear that the limitations under the token retrieval function are separate from the financial account registration function. Because Patent Owner amended the claims to specify that the step of “providing a non-sensitive electronic data token to the merchant... without providing the non-sensitive electronic data token to the payer” (limitation [1.4.1]) falls under “*execut[ing] a token retrieval function*” (limitation [1.4]), this limitation applies the token retrieval function and not to the financial account

registration function (limitation [1.1.5]).

3. Specification of the '621 patent

57. The specification also supports construing limitation [1.4.1] such that the token is provided to the merchant without providing it to the payer during the token retrieval function, but the token can be provided to the payer during other steps of the process. The specification describes the financial account registration and token retrieval functions as being separate and distinct functions:

A hosted system on a secure server 202 provides **Financial Account Registration functions and Token Retrieval functions** that can be accessed by merchants and others operating websites, for example using merchant server 204, to process payment transactions.

Ex.1001, 5:58-62; *see also* 6:63-7:1. The specification further explains:

Secure server 202 returns a token or other account identifier constituting a symbolic representation of the financial account information to merchant server 204 [first part of the token retrieval function] **after** the financial account information has been registered at secure server 202 [financial account registration function]. **Thereafter**, merchant server 204 may use the token to process further payment transactions authorized by the end user or customer at sites 206, 208, 210 in their interaction with the merchant [second part of the token retrieval function].”

Ex.1001, 7:1-8. Thus, the '621 patent clearly distinguishes the financial account registration function in which the financial account information is registered at the

secure server from the token retrieval function in which the secure server provides the token to the merchant and the merchant requests the secure server complete the transaction using the token provided.

58. Moreover, the '621 patent describes the secure server sending the token to the merchant server during the token retrieval function. *See e.g.* Ex.1001, 7:31-34 (“Secure server 202, in a preferred embodiment, stores the financial account information received from the customer (for example from sites 206, 208, or 210) and provides a token to merchant server 204.”). However, the '621 patent does not indicate that the token is sent during any other step in the payment process. Notably, the specification never explicitly states that (1) the token is sent to the merchant server *without* being sent to the payer during the token retrieval function or (2) the token *cannot* be sent to the payer during any other step of the payment process (e.g. during the financial account registration function).

59. In fact, the '621 patent states that the token is “only meaningful to the merchant.” Ex.1001, 10:46-48. The '621 patent itself never explains why it would be necessary to prevent the token from being sent to the payer given that it is *only meaningful to the merchant* and, thus, is not meaningful to any other party including the payer.

60. Additionally, Provisional Application No. 61/655,482 (the '482 application, Ex.1005), which the '621 patent incorporates by reference,

contemplates that the non-sensitive data token *is* provided to the payer outside of execution of the token retrieval function. The '482 application explains that “[t]he hosted system [i.e. secure server] returns a token or other account identifier **to the user** [i.e. payer] and/or the merchant's system **when the financial account information has been registered at the hosted system.**” Ex.1005, 21; *see* 31-32 (the end user (i.e. payer) requests a reference ID from the secure server). Instead, it contemplates that the token may be sent to the payer during the financial account registration function.

61. Therefore, in view of the structure of the claims, the prosecution history, and the disclosure of the '621 patent, it is my opinion that a POSITA would have understood that limitation [1.4.1] requires the token be provided to the merchant without providing the token to the payer only during the token retrieval function and that the token may be sent to the payer during another step in the payment process.

B. “*receiving the non-sensitive electronic data token from the secure server, wherein...the secure server does not provide the non-sensitive electronic data token to the payer*”

62. Based on my review of the structure of the claims as well as the specification and the prosecution history, it is my opinion that the limitation “*receiving the non-sensitive electronic data token from the secure server, wherein...the secure server does not provide the non-sensitive electronic data*

token to the payer” should be construed as “*during execution of the token retrieval function*, receiving the non-sensitive electronic data token from the secure server, wherein...the secure server does not provide the non-sensitive electronic data token to the payer.” In short, this limitation does not mean that the secure server must never provide the token to the payer. Instead, the claim language only requires that, *during the token retrieval function*, the secure server does not provide the token to the payer when the merchant server receives the token. This limitation appears in two claims: limitation [15.8] in claim 15 and limitation [23.2.8] in claim 23. The below claim construction analysis focuses on claim 15, but the same analysis applies to claim 23 as well.

1. Structure of the Claims

63. The claims clearly divide the payment process into two discrete functions: (1) the financial account registration function and (2) the token retrieval function. Claim 15 recites “*initiating a financial account registration function of the API during the authenticated session, the financial account registration function:*” (limitation [15.6]) performing several steps including “*establishing a secure socket layer connection between the secure server and the payer computing system...*” (limitation [15.6.1]), “*outputting instructions to the payer computing system to render a financial account registration request form...*” (limitation [15.6.2]), and “*outputting instructions to the payer computing system to encrypt the*

sensitive financial account information provided by the payer and transmit the encrypted financial account information to the secure server...without providing the sensitive financial account information to the merchant server” (limitation [15.6.3]).

64. However, the limitation “*the secure server does not provide the non-sensitive electronic data token to the payer*” is performed as a part of the “*token retrieval function.*”

65. Limitation [15.7] recites “*initiating a token retrieval function of the API during the authenticated session, **the token retrieval function outputting a non-sensitive electronic data token** representing the sensitive financial account information to the merchant server*” and limitation [15.8] then recites “***receiving the non-sensitive electronic data token from the secure server**, wherein the merchant server does not receive the sensitive financial account information represented by the non-sensitive electronic data token and **the secure server does not provide the non-sensitive electronic data token to the payer.***” See also, limitation [15.10] (“*processing a payment from the payer by generating a payment transaction instruction, using the token retrieval function of the API, that includes a representation of the non-sensitive electronic data token*”). Thus, “*the secure server does not provide the non-sensitive electronic data token to the payer*” in response to the initiation of the “*token retrieval function.*”

66. Moreover, claim 15 recites that, during the “*financial account registration function*,” the encrypted financial account information is transmitted to the secure server “***without providing the sensitive financial account information to the merchant server***” (limitation [15.6.3]). Claim 15 then recites that, during the “*token retrieval function*,” the merchant server receives the non-sensitive data token “***wherein the merchant server does not receive the sensitive financial account information***” (limitation [15.8]). Therefore, the claims specify that the financial account information is not sent to the merchant server ***both*** during the financial account registration function ***and*** during the token retrieval function. In contrast, the claim only requires “*the secure server does not provide the non-sensitive electronic data token to the payer*” during the token retrieval function.

2. Prosecution History of the '621 Patent

67. The prosecution history of the '621 patent supports this understanding as well. As explained above in Section VI.A.2, after responding to the examiner's § 101 rejection of the pending claims, Patent Owner and the examiner communicated extensively back and forth (via emails and phone calls) during which the parties iterated through several versions of the amendments to overcome § 101. Ex.1002, 54-238. During this process, Patent Owner made amendments reciting separate financial account registration and token retrieval functions and grouping several steps of the payment process under one function or the other, as

shown below:

31. (Currently Amended): A method of enrolling a payer by a merchant server operated by or for the benefit of a payee, the method comprising:

- providing a webpage to a payer computing system used by a payer;
- receiving enrollment data identifying a payer the payer;
- providing, to a secure server via an application programming interface (API) provided by the secure server, at least one data element associated with the payer and a payment amount from the payer to the payee;
- displaying a window or frame within the webpage provided to the payer computing system;
- providing authentication credentials to the API to establish an authenticated session, the authentication credentials identifying the payee and authenticating the payee;
- initiating a financial account registration function of the API during the authenticated session, the financial account registration function: generating an electronic instruction for a secure server that includes at least one data element associated with the payer, the electronic instruction requesting the establishment of a secure connection via the internet between the secure server and a computing system used by the payer for the secure server to receive financial account information provided by the payer; outputting the electronic instruction to a communications network for transmittal to the secure server;
- establishing a secure socket layer connection between the secure server and the payer computing system within the window or frame;
- outputting instructions to the payer computing system to render a financial account registration request form, within the window or frame, that provides functionality for the payer to provide financial account information; and
- outputting instructions to the payer computing system to encrypt the financial account information provided by the payer and transmit the encrypted financial account information to the secure server, via the secure socket layer connection, without providing the financial account information to the merchant server.

~~initiating a token retrieval function of the API during the authenticated session, the token retrieval function outputting~~ ~~receiving, from the secure server, an electronic data token representing the financial account information received by the secure server via the secure connection to the merchant server;~~

~~receiving the electronic data token from the secure server, wherein the merchant server does not receive the financial account information represented by the electronic data token and the secure server does not provide the electronic data token to the payer;~~

~~storing the electronic data token in association with the enrollment data identifying the payer; and~~

~~processing a payment from the payer by electronically generating a payment transaction instruction, using the token retrieval function of the API, that includes a representation of the electronic data token and instructs a payment processing system having access to the financial account information represented by the electronic data token.~~

Ex.1002, 125-126

68. By making amendments reciting a separate financial account registration function and token retrieval function and reciting specific steps that fall under each of those functions, Patent Owner made clear that the limitations under the token retrieval function are separate from the financial account registration function. Because Patent Owner amended the claims to specify that the step of “*the secure server does not provide the non-sensitive electronic data token to the payer*” (limitation [15.8]) during the token retrieval function and not the financial account registration function.

3. Specification of the '621 patent

69. For the same reasons discussed in Section VI.A.3, the specification also supports construing limitation [15.8] such that the secure server does not

provide the token to the merchant server in response to the initiation of the token retrieval function.

70. Therefore, in view of the structure of the claims, the prosecution history, and the disclosure of the '621 patent, it is my opinion that a POSITA would have understood that limitation [15.8] requires the secure server does not provide the token to the payer only during the token retrieval function and that the token may be sent to the payer during another step in the payment process.

C. “*generating a uniform resource locator (URL),... the URL comprising either: a dynamic URL generated by the secure server...; or a static URL and a hypertext transport protocol (HTTP) parameter...*”

71. Limitations [1.1.1.5], [1.1.1.5a], and [1.1.1.5b] recite that the step of “*generating*” a URL can be done either dynamically or statically. In my opinion, the generating step should be construed such that when “*generating*” the URL statically, the URL is “*generat[ed]*” prompting the merchant server to combine a predefined static URL with an “*HTTP parameter.*” In other words, a POSITA would not have understood the term “*generating a uniform resource locator (URL)*” to strictly require that a URL be generated by the API on the secure server. Thus, the static URL is “*generat[ed]*” by prompting the merchant server to combine a predefined URL with the “*parameter.*” This limitation appears in three claims: limitations [1.1.5.1]-[1.1.5.1b] in claim 1, limitations [8.2.1.5.1]-[8.2.1.5b] in claim 8, and limitations [25.1.5.1]-[25.1.5.1b] in claim 25. The below claim

construction analysis focuses on claim 1, but the same analysis applies to claims 8 and 25 as well.

1. Claim Language

72. The claims define the act of “generating a URL” as one of two processes: either (1) “*a dynamic URL generated by the secure server*” (limitation [1.5.1.1a]) or (2) “*a static URL*” known to the merchant and a “*hypertext transport protocol (HTTP) parameter used by the secure server to identify the payer and the payee*” (limitation [1.5.1.1b]). For (1) the dynamic URL, the claims explicitly recite that the dynamic URL be “*generated by the secure server.*” On the other hand, for (2) the static URL, the claims do not recite that the static URL is generated by the secure server. There would be no need—the URL is “static” and is not generated by the secure server for each transaction. Instead, for (2) the static URL, the static URL includes “*a hypertext transport protocol (HTTP) parameter [that is] used by the secure server to identify the payer and the payee.*” Instead of the secure server generating a dynamic URL for the specific transaction, a predefined static URL establishes a connection with the secure server and the secure server uses the parameter to identify the payer and payee connected to the specific transaction. Thus, in view of the claims, a POSITA would have understood that, if using a static URL, the static URL would not need to be generated by the secure server to satisfy the claim language.

2. Specification of the '621 Patent

73. The specification also supports this construction because it describes that in the static option, the URL is “*generat[ed]*” by prompting the merchant server to combine a predetermined URL with the HTTP parameter. The '621 patent explains that there are two distinct paths using either: a dynamic URL or a static URL. Ex.1001, 11:48-52 (“[T]he URL interface provided may be implemented either with a static URL in step 506 (designated as 1B) or a dynamic URL in step 504 (designated as 1A).”). This is shown in Fig. 4a below:

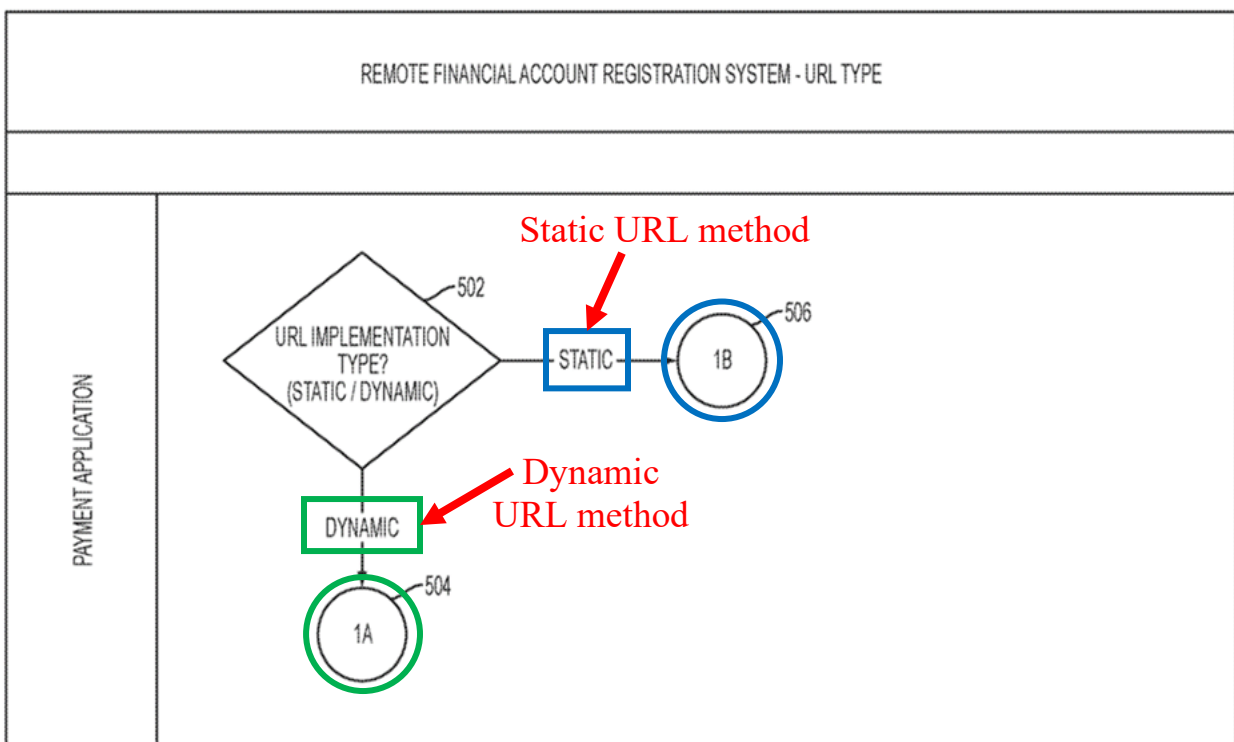


FIG. 4a

Ex.1001, Fig. 4a (annotated)

74. During the dynamic URL method (1A), the end user (payer) starts by “request[ing] registration of a financial account” in step 508 (Ex.1001, 11:55-57), as shown in Fig. 4b below:

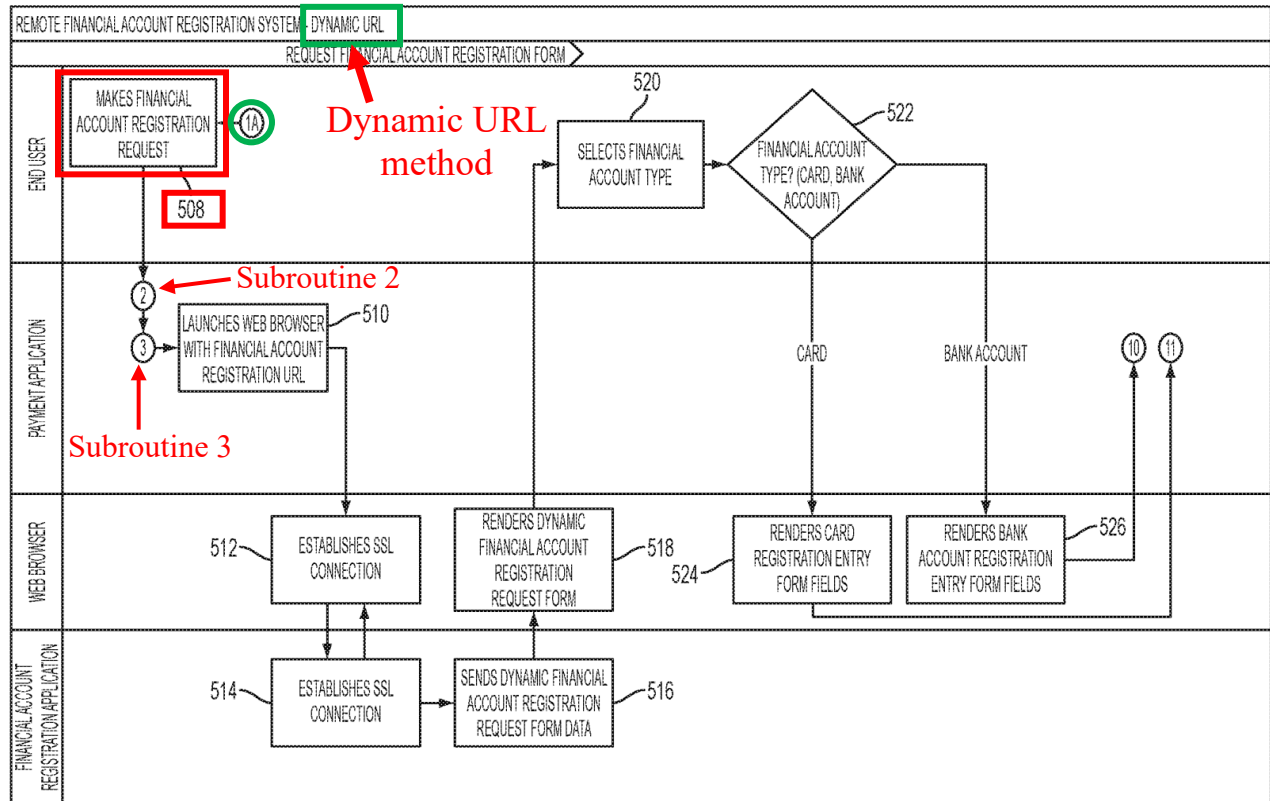


FIG. 4b

Ex.1001, Fig. 4b (annotated)

75. The method then moves to “a process of logging into the payment gateway (designated as subroutine “2”).” Ex.1001, 11:58-59. Neither a dynamic URL or a static URL is generated or used during subroutine 2. *See* Ex.1001, 13:55-14:19, Fig. 4f. The method moves from subroutine 2 to subroutine 3 in which “a financial account registration entry form...is provided” (Ex.1001, 11:60-62) as

shown in Fig. 4g:

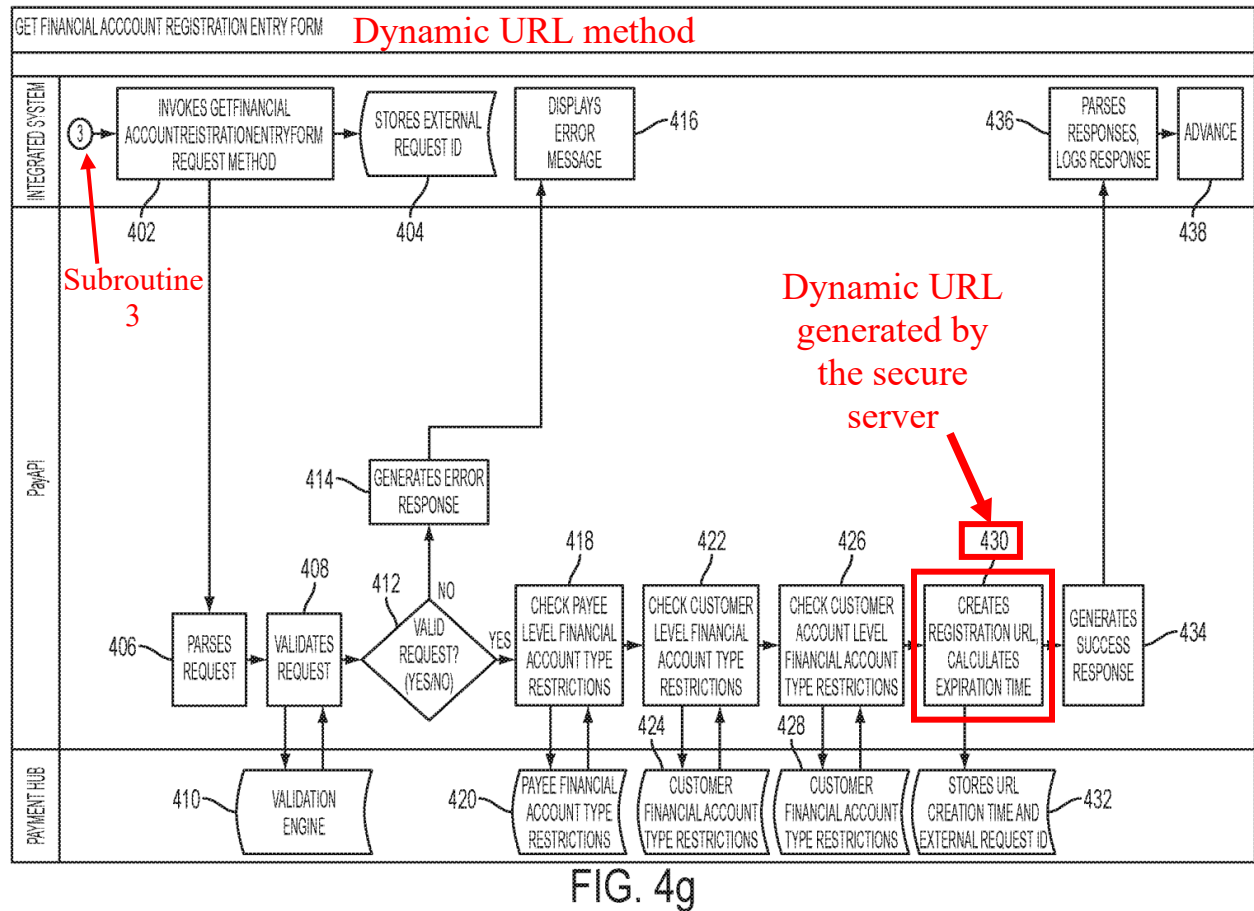


FIG. 4g

Ex.1001, Fig. 4g

76. During subroutine 3, the secure server determines whether the request is valid and checks if there are any restrictions for the payer or payee (steps 406-428). Ex.1001, 14:20-45. “In step 430, the API [of the secure server] creates a registration URL.” Ex.1001, 14:46-47. The “registration URL” is the dynamic URL.

77. After subroutine 3 completes, the method returns to Fig. 4b:

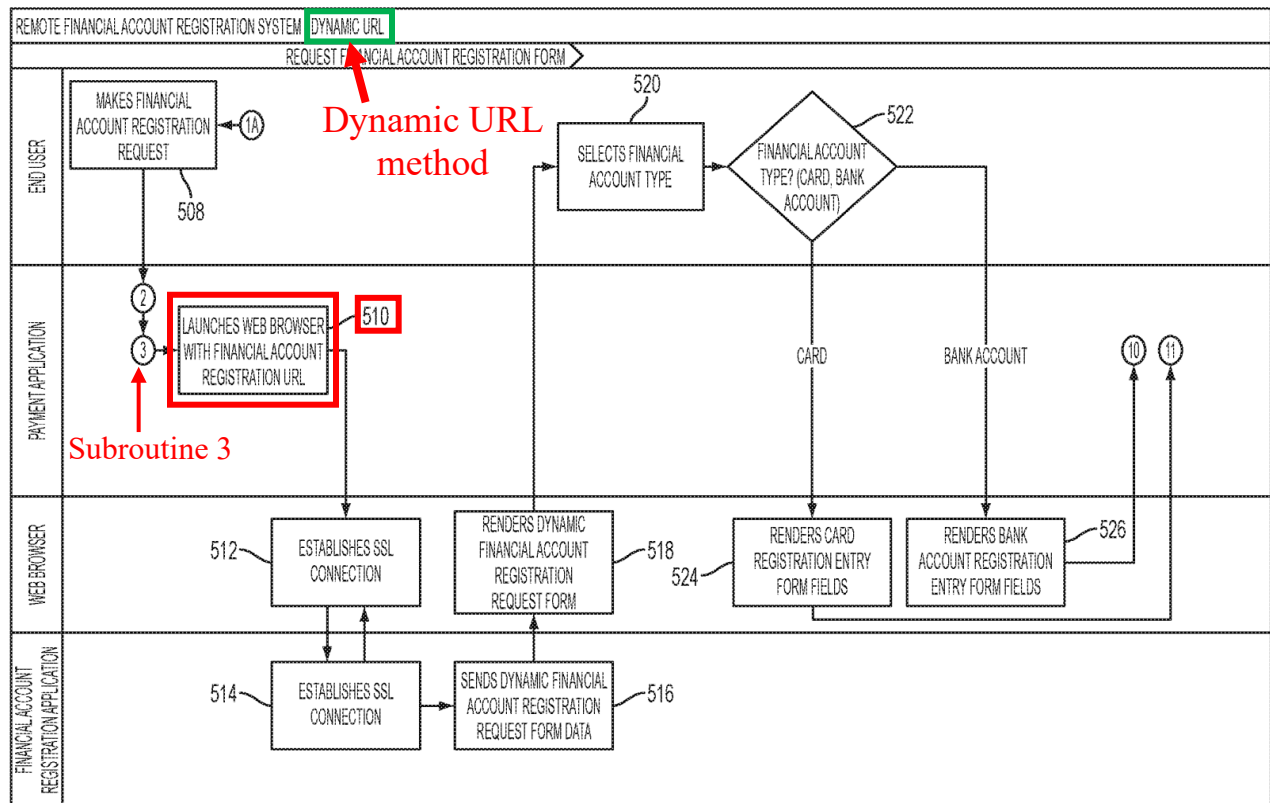


FIG. 4b

Ex.1001, Fig. 4b (annotated)

78. During step 510, “the application [merchant server] launches a browser window for the end user with a specified Financial Account Registration URL.” Ex.1001, 11:62-64. Thus, the dynamic registration URL generated by the secure server in step 430 of subroutine 3 is used by the merchant server to establish the SSL connection between the payer and the secure server.

79. On the contrary, when static URL method (1B) is used, a URL is never generated by the secure server. The static URL method (1B) is shown in Fig. 4d:

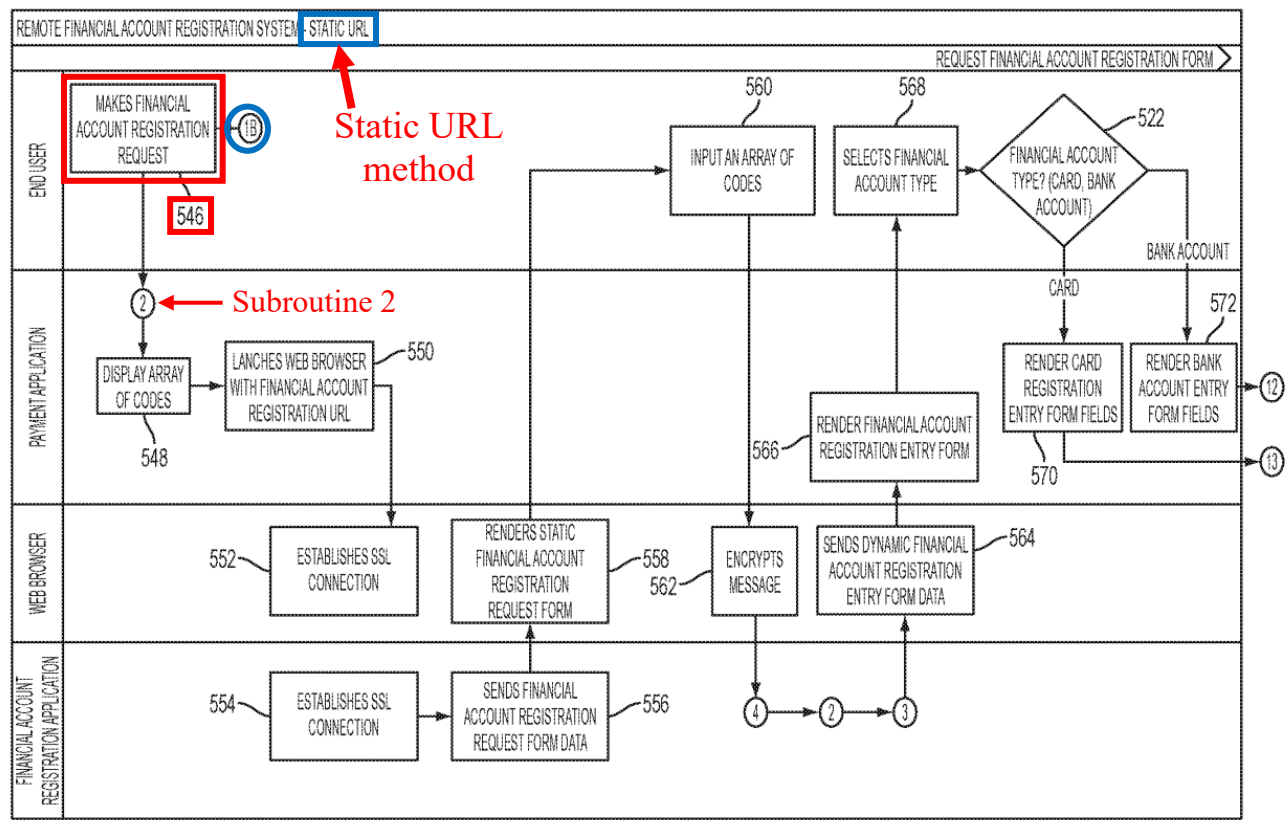


FIG. 4d

Ex.1001, Fig. 4d (annotated)

80. “[T]his process starts at step 546 when the end user requests registration of a financial account.” Ex.1001, 12:53-54. The method then moves on to subroutine 2 “to ensure that the end user is properly identified by the system and the information entered by the end user is associated with the correct user and account.” Ex.1001, 12:55-58. Subroutine 2 is the same as that performed during the dynamic URL method (1A). As explained above, no dynamic URL or static URL is generated or used during subroutine 2.

81. Unlike in the dynamic URL method (1A), in the static URL method (1B) subroutine 3 is **not** performed and the secure server does **not** generate a new URL. Thus, step 430 of subroutine 3 in which the secure server “generates” the registration URL is **not** performed.

82. Instead, the method moves from subroutine 2 to the next steps in which “[t]he payment application then **displays an array of codes in step 548 and launches a web browser with a financial account registration URL [in step] 550**” (Ex.1001, 12:58-60) as shown in Fig. 4d:

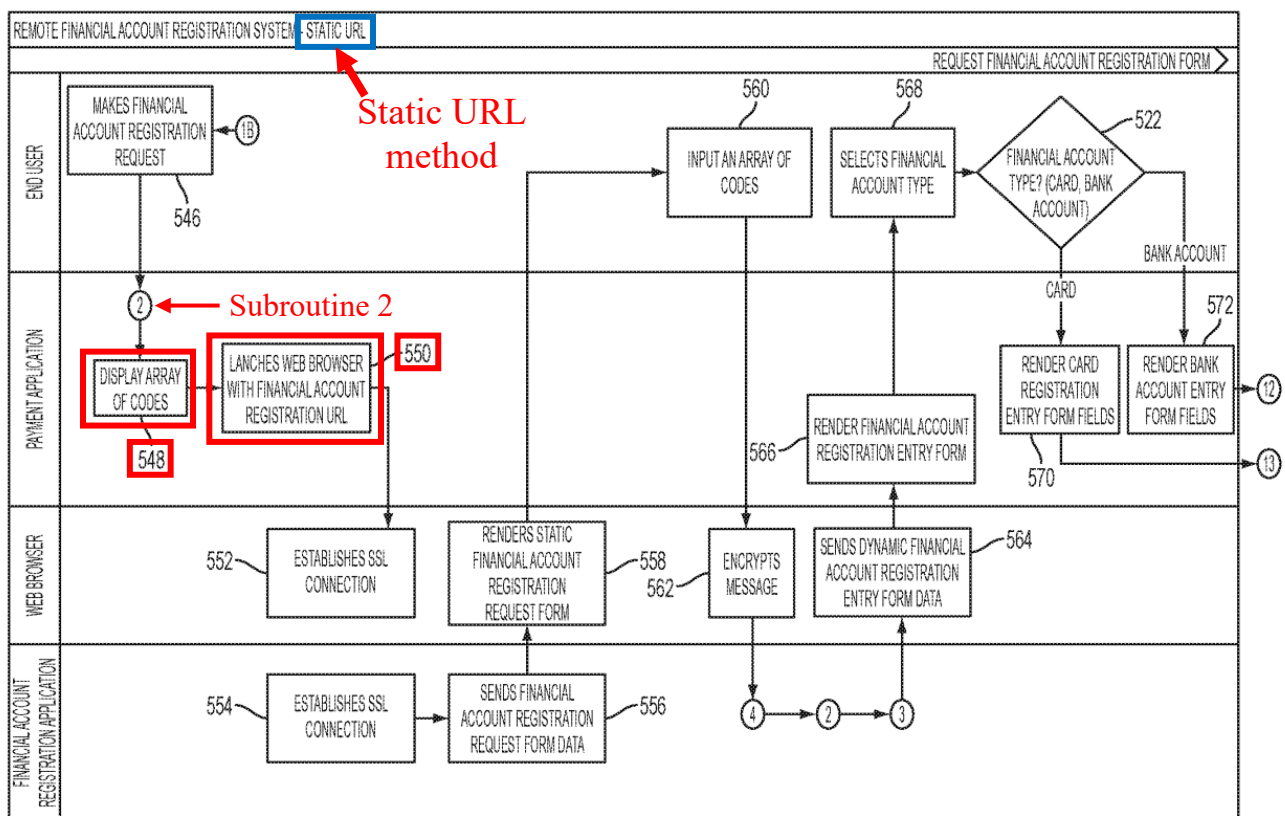


FIG. 4d

Ex.1001, Fig. 4d (annotated)

83. The static URL method (1B) never describes *generating* a new financial account registration URL. Instead, a POSITA would have understood that the payment application (i.e. the merchant server) uses a predefined static URL that is known to the payment application to establish the SSL connection between the end user and the financial account registration application (i.e. the secure server). *See* Ex.1001, 11:28-34.

84. Additionally, the '482 application, which is incorporated by reference in the '621 patent as described above, similarly distinguishes how the dynamic and static URLs are generated. The '482 application explains that the dynamic URL is generated by the secure server and the static URL is not. The '482 application describes the different URL methods below:

The System supports two methods of retrieving a Financial Account Registration Form; as described in the table below.

Workflow Type	Description
Dynamic URL	A temporary URL to a Financial Account Registration Form can be requested via PayAPI; whereby the input parameters in the form request are used to authenticate the registration request, present different variations of the Financial Account Form based on configuration settings and rules in the System, and associate the Financial Account to a Customer.
Static URL	A temporary array of codes can be requested via PayAPI and submitted on a Financial Account Registration site; whereby the codes are used to authenticate the registration request, present different variations of the Financial Account Form based on configuration settings and rules in the System, and associate the Financial Account to a Customer.

Ex.1005, 62 (annotated)

85. As described above, in the dynamic URL method, a “temporary URL” is requested from PayAPI (i.e. secure server). Ex.1005, 62. In the static URL

method, however, “a temporary array of codes” is requested from PayAPI and submitted on a Financial Account Registration site. Ex.1005, 62. Thus, the Financial Account Registration site (i.e. the static URL) is not requested from and, thus, is not generated anew by the secure server.

86. For these reasons, it is my opinion that a POSITA would have understood that, when the API “*generat[es] a... URL*” comprising “*a static URL*,” the static URL itself is “*generat[ed]*” by prompting the merchant server to combine a known static URL with a “*parameter*” “*for establishing the secure socket layer connection.*”

VII. IDENTIFICATION OF HOW THE CLAIMS ARE UNPATENTABLE

87. I have been asked to provide my opinion as to whether the Challenged Claims of the '621 patent would have been obvious in view of the prior art. The discussion below provides a detailed analysis of how the prior art references identified below teach the limitations of the Challenged Claims of the '621 patent.

88. As part of my analysis, I have considered the scope and content of the prior art and any differences between the alleged invention and the prior art. I do so by describing in detail, on an element-by-element basis, for each Challenged Claims of the '621 patent.

89. As a consequence of the following analysis, focused on the well-known online payment innovator (and long-time provider of tokenized

transactions) PayPal, the alleged invention of the Challenged Claims would have been obvious in view of the teachings of the identified prior art references as well as the knowledge of a POSITA.

A. Ground 1: Claims 1, 2, 3, 4, 5, 6, 7, 15, 16, 17, 18, 19, 20, 21, 22, 25, 26, and 27 are obvious under 35 U.S.C. § 103 over PayPal Express Checkout.

1. Reasons to Combine the documents describing express checkout into PayPal Express Checkout

90. PayPal Express Checkout is a product by PayPal in which a merchant can allow a payer (i.e. buyer) to purchase items from the merchant's website using PayPal without the merchant receiving sensitive financial information from the buyer. PayPal Express Checkout is described in a series of printed publications that were publicly available on PayPal's website. These printed publications were published and/or revised between December 2005 and May 2012 and include:

- PayPal Express Checkout Integration Guide ("Integration Guide," Ex.1006),
- PayPal SOAP API Developer Reference ("SOAP Developer Reference," Ex.1007),
- PayPal Sandbox User Guide ("Sandbox User Guide," Ex.1008),
- PayPal Preparing for the New Updated Checkout Experience ("New Updated Checkout Experience," Ex.1009),
- PayPal Express Checkout User Interface Standards ("User Interface Standards," Ex.1010),

- PayPal Privacy Policy (“Privacy Policy,” Ex.1011),
- PayPal Digital Goods Integration Guide (“DG Integration Guide,” Ex.1012),
- PayPal Instant Payment Notification Guide (“IPN Guide,” Ex.1023),
- PayPal Name-Value Pair API Developer Guide (“NVP Developer Guide,” Ex.1024), and
- PayPal Merchant Setup and Administration Guide (“Merchant Setup Guide,” Ex.1025).

Although these are separate documents, they each cover the same product and, thus, are considered a single prior art reference herein. Thus, I will refer to these documents collectively as “PayPal Express Checkout.”

91. PayPal’s Technical Documentation page of PayPal’s Integration Center website (Ex.1021, July 10, 2011) provides documents that explain each of PayPal’s products and how merchants and their developers can integrate each product into their system. Under the Express Checkout product, the Technical Documentation page includes PDF and HTML links to the PayPal Express Checkout Integration Guide (Ex.1006), SOAP API Developer Reference (Ex.1007), Sandbox User Guide (Ex.1008), Preparing for the New Updated Checkout Experience (Ex.1009), Express Checkout User Interface Standards (Ex.1010), Digital Goods Integration Guide (Ex.1012), IPN Guide (Ex.1023), and NVP Developer Guide (Ex.1024) as shown below:

Express Checkout ← Documents relating to PayPal's express checkout product

Express Checkout Integration Guide (PDF, HTML) ← Ex.1006
Learn the basics of how to integrate Express Checkout with your website. See where Express Checkout goes on your site. Learn where your site needs to call the API.

Express Checkout Advanced Features Guide (PDF, HTML)
Learn about advanced features that enhance your integration of Express Checkout, such as customizing PayPal payment pages to match the look of your website and updating shipping charges as buyers make changes on the PayPal review page.

Digital Goods Integration Guide - Express Checkout Edition (PDF, HTML) ← Ex.1012
Learn how to streamline the checkout process for buyers of digital goods. Buyers can conveniently purchase digital goods during checkout with a minimum of clicks without leaving your website or interrupting their online activities.

Preparing for the New Updated PayPal Checkout Experience (PDF) ← Ex.1009
Learn about the new updated PayPal Checkout experience for Express Checkout. This guide provides the programming variables to customize the new updated checkout user interface.

Name-Value Pair API Developer Guide and Reference (PDF, HTML) ← Ex.1024
Learn how to use the Name-Value Pair API. Includes code samples, an API reference, information on SDKs, and web samples.

SOAP API Developer Reference (PDF, HTML) ← Ex.1007
Programmers familiar with SOAP (Simple Object Access Protocol) can learn more about the PayPal SOAP API.

Sandbox User Guide (PDF, HTML) ← Ex.1008
Learn how to test your PayPal-based applications before using them in production with the Sandbox – PayPal's virtual test environment.

Instant Payment Notification Guide (PDF, HTML) ← Ex.1023
Instant Payment Notification (IPN) is a message service that notifies you of events related to PayPal transactions. You can use IPN to automate back-office and administrative functions.

Express Checkout User Interface Standards (PDF) ← Ex.1010
Express Checkout User Interface Standards details the requirements and best practices for placing PayPal graphic images on your website. It also explains how buyers who pay with PayPal will experience checkout on your website, and how you can modify that buyer experience depending on your needs.

Ex.1021, 1-2 (annotated)

92. The Technical Documentation page also includes PDF and HTML links to the Merchant Setup Guide (Ex.1025) as shown below:

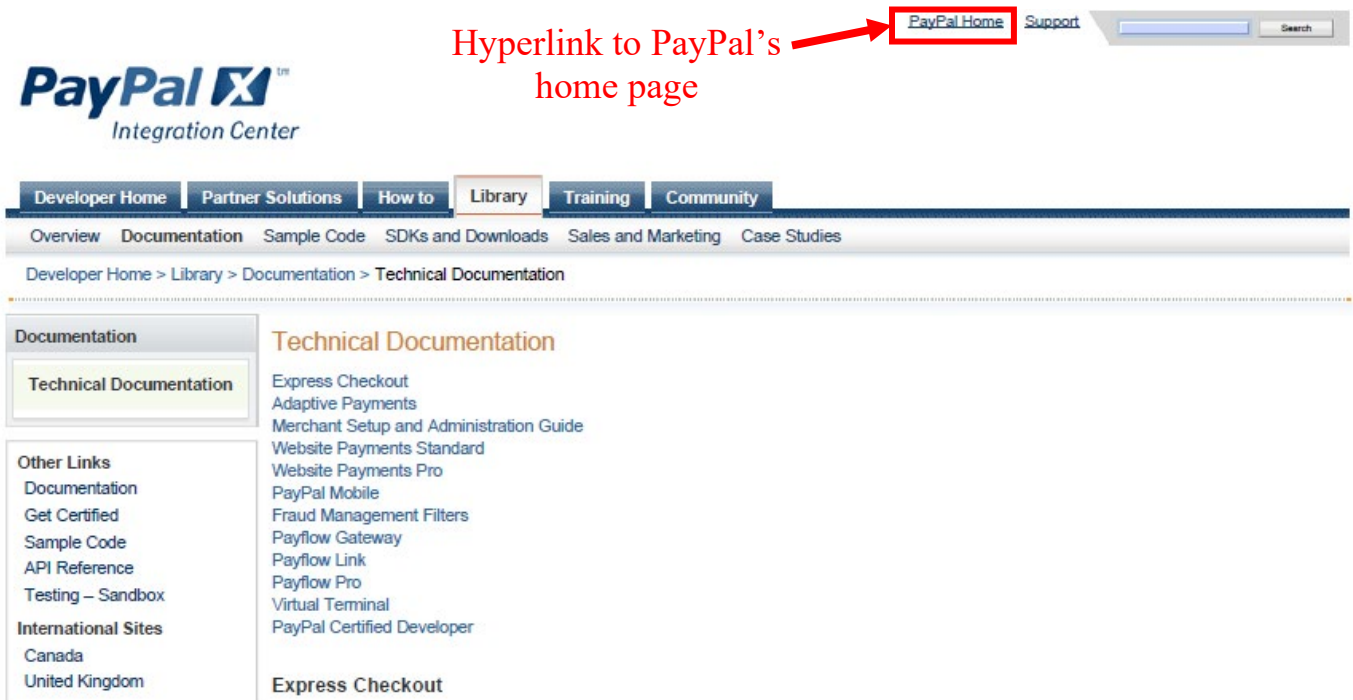
Merchant Setup and Administration Guide

Merchant Setup and Administration Guide (PDF, HTML) ← Ex.1025
Learn how merchants set up and administer their accounts, including setting their credit card statement names, blocking certain kinds of payment, adding users to their accounts, managing multiple currency balances, creating custom payment pages, getting contact numbers from payers, and working with transaction history.

Ex.1021, 2 (annotated)

93. A POSITA would have accessed PayPal's Privacy Policy (Ex.1011)

by first clicking the link to PayPal's home page from the Integration Center, as shown below:



Ex.1021, 1 (annotated)

94. From the home page, the POSITA then would have clicked the link to PayPal's Privacy Policy, as shown below:

[Sign Up](#) | [Log In](#) | [Help](#) | [Security and Protection](#)

PayPal [English](#) ▼

[Home](#) [Personal](#) [Business](#) [Developers](#)

[Get to Know PayPal](#) [Pay Online](#) [Send Money](#) [Get Paid](#) [Products & Services](#)

Account login

Email address

PayPal password

Go to
[My account](#) ▼

[Problem with login?](#)

New to PayPal? [Sign up](#).

WELCOME TO PayPal
The world's most loved way to pay and get paid. [Learn More](#)

Necklace selected.
Privacy protected.

Pay with PayPal

Go for the things you love
with confidence.

94.4 million
people worldwide using PayPal

[Sign Up](#)

Get to Know PayPal

[How PayPal Works](#)

[Getting Started](#)

[Managing Your Account](#)

[Great Ways to Use PayPal](#)

[Top 10 Things to Know about PayPal](#)

Pay Online

Shop and pay online quickly and securely.

[Learn More](#)

Send Money

Send money to just about anyone, anywhere.

[Learn More](#)

Get Paid

Accept online payments for items you sell.

[Learn More](#)

Looking to accept credit cards or set up a merchant account? Visit [Merchant Services](#)

Pay with PayPal

STAY SAFER OUT THERE
PayPal protects your financial info and purchases from checkout to delivery.

All Personal Products & Services

Pay Online	Send Money	Get Paid	Other
PayPal Credit	Send Money Online	Sell on eBay	Student Account
Shop via Your Mobile	International Money Transfer	Accept Credit Cards	PayPal Mobile
	To Your Teen	Request Money	Money Market Fund
	Via Your Mobile	Fundraise	PayPal Debit Card
		Merchant Services	Bill Me Later® a PayPal service

Hyperlink to
PayPal's
Privacy Policy
(Ex.1010)

[About PayPal](#) | [Contact Us](#) | [Fees](#) | [PayPal Developers](#) | [Merchant Services](#) | [Worldwide](#) | [Site Feedback](#) | [Privacy](#) | [PayPal Blog](#) | [PayPal Labs](#) | [Referrals](#) | [Jobs](#) | [Legal Agreements](#) | [Site Map](#) | [eBay](#)

VeriSign Identity Protection

Copyright © 1999-2011 PayPal. All rights reserved.

Ex.1022, 1-2 (annotated)

95. Thus, a POSITA could have accessed information about how PayPal protects personal and billing information via two easy clicks from PayPal's Technical Documentation page on PayPal's Integration Center website. Because these documents are collected under PayPal's "Express Checkout" on PayPal's Technical Documentation page (Ex.1006-1010, 1012, 1023-1025) or easily accessible on PayPal's website from the Technical Documentation page (Ex.1011), a POSITA would have understood that each of these documents describes at least one aspect of PayPal's Express Checkout product. Thus, a POSITA would have understood that because each of these documents describes aspects of the same product—PayPal's Express Checkout—the teachings from each of the documents act as a single reference (Ex.1006-1012, 1023-1025, collectively "PayPal Express Checkout"). Exhibits 1021 and 1022 are documents from the Internet archive, showing that as of at least July 10, 2011 (Ex.1021), the PayPal Express Checkout documentation relied on herein was publicly available. A POSITA looking to implement PayPal's technique would have known to look at PayPal's website under the Technical Documentation section.

96. Further, a POSITA would have found it obvious to combine the teachings of each separate document—in fact, it would have been necessary to combine all of the teachings. Each of the documents are the equivalent of a chapter

in a single manual describing PayPal's Express Checkout product. To gain a full and complete understanding of PayPal's Express Checkout product, a POSITA would have necessarily combined these documents, because only by combining the documents could a POSITA implement the operation and functionality of an existing product on the market. In other words, the combination of these documents is not hypothetical—POSITAs were expected to combine the teachings in order to implement PayPal's Express Checkout product. It would have been desirable and beneficial to implement PayPal's product in the tried-and-true method provided by the producer of that product. Accordingly, combining the teachings of the individual documents is merely a matter of combining prior art elements (the teachings of each individual document) according to known methods to yield predictable results.

97. PayPal Express Checkout is in the same field as the '621 patent. The field of endeavor of the '621 patent is "obtaining and using account information to process financial payments" so that "merchants are able to process such transactions without transmitting, processing or storing Financial Account data." Ex.1001, 1:19-20, 5:63-65. Similarly, the Integration Guide explains that "Express Checkout makes it easier for a buyers [*sic*] to pay online" and "enables [the merchant] to accept PayPal while retaining control of the buyer and the overall checkout flow." Ex.1006, 9.

98. For example, the Integration Guide explains that “PayPal provides a command that [the merchant] use[s] when redirecting [the] buyer’s browser to PayPal,” which “enables [the] buyer to log into PayPal to approve an Express Checkout payment,” thereby obtaining account information from buyers. Ex.1006, 17; *see also* 25-26. The Integration Guide explains that the merchant “[c]ompletes the Express Checkout transaction” with the buyer’s account information using the DoExpressCheckoutPayment API operation without the merchant receiving the buyer’s account information. Ex.1006, 17; *see also* 21-22, 26-27.

99. A POSITA would have been motivated to combine the SOAP Developer Reference with the Integration Guide because the SOAP Developer Reference provides details about the requests and responses for the API operations described in the Integration Guide.¹ *See* Ex.1006, 17 (SetExpressCheckout, GetExpressCheckoutDetails, and DoExpressCheckoutPayment API operations); Ex.1007, 69-133 (requests and responses for SetExpressCheckout, GetExpressCheckoutDetails, and DoExpressCheckoutPayment API operations).

¹ Simple Object Access Protocol (SOAP) is a messaging standard defined by the World Wide Web Consortium (W3C) and its member editors. SOAP uses an XML data format to declare its request and response messages, relying on XML Schema and other technologies to enforce the structure of its payloads.

Moreover, the SOAP Developer Reference is specifically referenced in the Integration Guide and vice versa. *See e.g.* Ex.1006, 44 (“For complete details, see...the *SOAP API Reference*.”)(emphasis in original); 45 (“*SOAP API Reference*”) (emphasis in original); Ex.1007, 10 (“Where to Go for More Information” including a hyperlink to “Express Checkout Integration Guide”).

100. A POSITA would have been motivated to combine the Sandbox User Guide with the Integration Guide because Sandbox User Guide “describes how to test the Express Checkout name-value pair (NVP) API in the Sandbox,” such as the example transaction described in the Integration Guide. *See* Ex.1008, 39-42 (“Testing Express Checkout”); Ex.1006, 23-27 (Testing an Express Checkout Integration” “[The merchant] can test [its] Express Checkout integration in the Sandbox”). Moreover, the Sandbox User Guide specifically references the Integration Guide. Ex.1008, 40 (“For more information about Express Checkout, see the *Express Checkout Integration Guide*.”) (emphasis in original).

101. A POSITA would have been motivated to combine the New Updated Checkout Experience with the Integration Guide because the New Updated Checkout Experience describes PayPal’s interface for the buyer after the buyer has been redirected to PayPal’s website. *See* Ex.1006, 17 (“PayPal provides a command that [the merchant] use[s] when redirecting [the] buyer’s browser to PayPal. This command enables [the] buyer to log into PayPal to approve an

Express Checkout payment.”); Ex.1009, 7-14 (“The PayPal Checkout User Interface” and “The Express Checkout Experience”). Moreover, the New Updated Checkout Experience specifically references the Integration Guide. Ex.1009, 6 (“Where to Go for More Information” including a hyperlink to “Express Checkout Integration Guide”); 10 (“In the Express Checkout Integration Guide, see ‘Customizing Express Checkout.’”). Similarly, a POSITA would be motivated to combine the User Interface Standards with the Integration Guide because the User Interface Standards describes PayPal’s interface for the buyer after the buyer has been redirected to PayPal’s website. *See* Ex.1006, 17 (“PayPal provides a command that [the merchant] use[s] when redirecting [the] buyer’s browser to PayPal. This command enables [the] buyer to log into PayPal to approve an Express Checkout payment.”); Ex.1010, 5 (Express Checkout User Interface Standards “also explains how buyers who pay with PayPal will experience checkout on [the merchant’s] website, and how [the merchant] can modify that buyer experience depending on [its] needs.”).

102. A POSITA would have been motivated to combine the Privacy Policy with the Integration Guide because the Privacy Policy describes how PayPal protects the privacy of information it receives from its products like Express Checkout. Ex.1006, 9 (“PayPal provides the stored information” such as “shipping, building, or payment information.”), 13 (“secure PayPal servers”); Ex.1011, 1

(“This policy describes the ways [PayPal] collect[s], store[s], use[s] and protect[s] [the] personal information. [The buyer] accept[s] this policy when [she] sign[s] up for or use [PayPal’s] products, services or any other features, technologies or functionalities offered by us on [PayPal’s] website or through any other means (collectively the “PayPal Services”).”).

103. A POSITA would have been motivated to combine the Digital Goods (“DG”) Integration Guide with the Integration Guide because the DG Integration Guide explains how PayPal’s Express Checkout works when a buyer purchases digital goods. Ex.1006, 28 (referencing digital goods); Ex.1012, 7 (“*Digital Goods Integration Guide - Express Checkout Edition* describes how to implement PayPal for Digital Goods.”) (emphasis in original), 9 (“Digital Goods for Express Checkout streamlines the checkout process for buyers and keeps buyers on your website.”).

104. A POSITA would have been motivated to combine the IPN Guide with the Integration Guide, because the IPN Guide describes instant payment notifications for PayPal, including specifically for Express Checkout transactions. Ex.1006, 37, 45-46 (discussing IPNs); Ex.1023 7 (“IPN is a message service that PayPal uses to notify you about events such as...Instant payments, including Express Checkout”), 14-15 (example of an IPN message for an Express Checkout transaction). A POSITA would have been motivated to combine the NVP

Developer Guide with the Integration Guide because the NVP Developer Reference provides details about the requests and responses for the API operations described in the Integration Guide. Ex.1006, 61 (“The Name-Value Pair (NVP) API provides parameter-based association between request and response fields of a message and their values.”), 61-70 (“PayPal Name-Value Pair API Basics”); Ex.1024 (“The PayPal NVP API is a simple programmatic interface that allows you, the merchant, to access PayPal’s business functionality to:...Accept PayPal in checkout on [the merchant’s] website using Express Checkout.”), 45-102 (“ExpressCheckout API Operations”). Moreover, the NVP Developer Guide is specifically referenced in the Integration Guide and vice versa. Ex.1006, 44 (“For complete details, see the *Name-Value Pair Developer Guide and Reference*”) (emphasis in original); 45 (“To update the subscriber’s street address, for example, specify all the address fields listed in the *Name-Value Pair Developer Guide and Reference*”) (emphasis in original); Ex.1024, 10 (“Where to Go for More Information...Express Checkout Integration Guide”).

105. A POSITA would have been motivated to combine the Merchant Setup Guide with the Integration Guide because the Merchant Setup Guide “describes how to perform set-up, administration, and other back-office tasks that let [the merchant] manage [its] PayPal account.” Ex.1025, 7; Ex.1006, 7 (The Integration Guide “is for merchants and developers who want to get started

implementing Express Checkout.”). Moreover, the Merchant Setup Guide is specifically referenced in the Integration Guide. Ex.1006, 8 (“Where to Go for More Information...Merchant Setup and Administration Guide”); 9 (“For information about administration tasks [the merchant] can perform...see the Merchant Setup and Administration Guide.”).

2. Summary of PayPal Express Checkout

106. PayPal Express Checkout describes the same tokenization process that is claimed in the ’621 patent. Like the ’621 patent, PayPal Express Checkout “enables [the merchant] to accept PayPal while retaining control of the buyer and the overall checkout flow.” Ex.1006, 9. PayPal Express Checkout “streamlines the checkout process for buyers and keeps them on the merchant’s site after making a purchase.” Ex.1006, 9.²

107. When a buyer decides to make a purchase on the merchant’s site, the buyer “1. Chooses Express Checkout by clicking **Checkout with PayPal**,” “2. Logs into PayPal to authenticate his or her identity,” “3. Reviews the transaction

² Note, PayPal Express Checkout is directed to merchants. Thus, these documents use second-person perspective (“you” and “your”). To ensure clarity, I replaced these instances in the quotes included herein with “the merchant,” “the merchant’s,” or similar to clarify that these statements are directed at the merchant.

on PayPal,” “4. Confirms the order and pays from [the merchant’s] site,” and “5. Receives an order confirmation” (Ex.1006, 10) as shown in the below diagram:

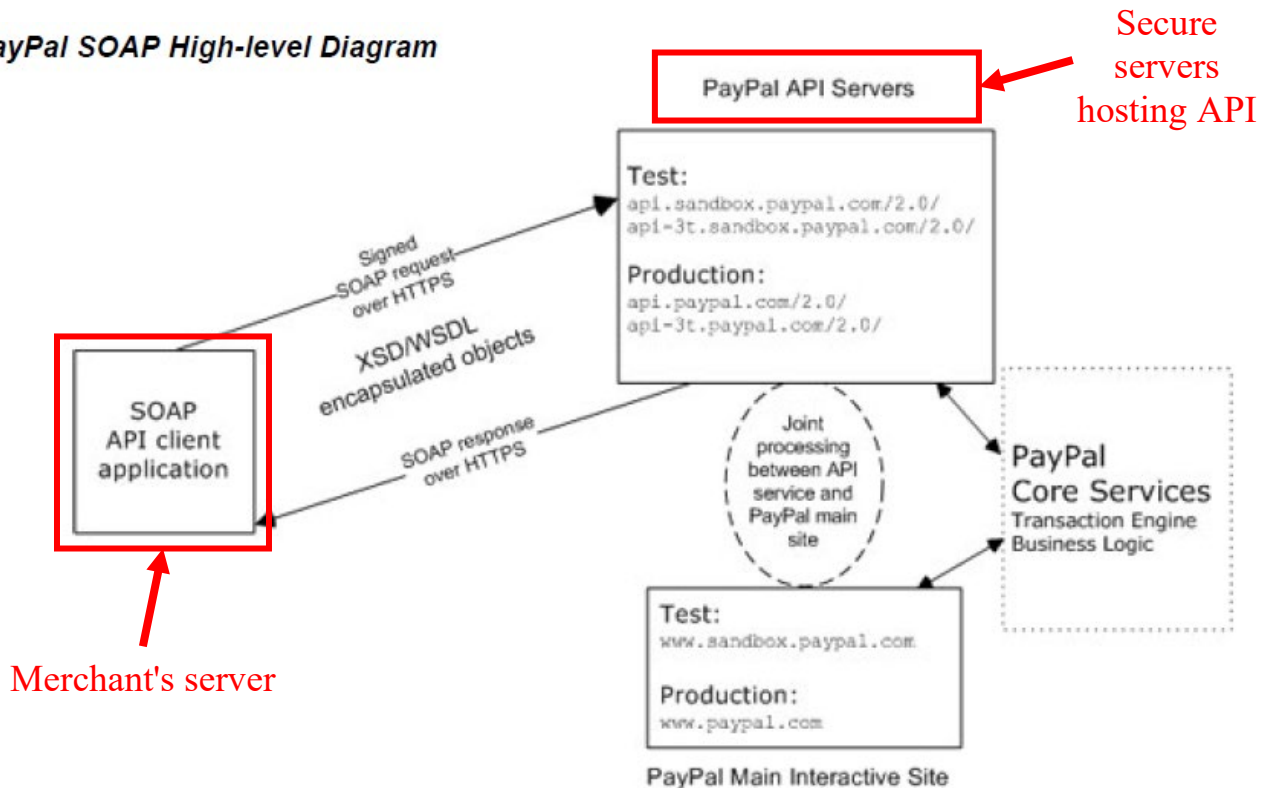


Ex.1006, 10

108. A merchant communicates with PayPal’s secure servers via its SOAP API. *See* 11 (“The PayPal SOAP API provides programmatic access to PayPal features and services.”); 11 (“Like many web services, PayPal SOAP is a combination of client-side and server-side schemas, hardware and software servers, and core services.”); 11-12 (“In an object-oriented processing model, the interface to SOAP requests/responses is an object in [the merchant’s] application’s native programming language. [The merchant’s] third-party SOAP client generates business-object interfaces and network stubs from PayPal-provided WSDL and XSD files that specify the PayPal SOAP message structure, its contents, and the PayPal API service bindings. A business application works with data in the form of object properties to send and receive data by calling object methods. The SOAP client handles the details of building the SOAP request, sending it to the PayPal

service, and converting the response back to an object.”).³This is illustrated in the below diagram:

PayPal SOAP High-level Diagram



Ex.1007, 11 (annotated)

109. “The PayPal API provides three API operations for Express Checkout,

³ Web Service Description Language (WSDL) is an XML based definition language. It is used for describing the functionality of a SOAP based web service. XML Schema Definition (XSD) is a recommendation by the W3C to describe and validate the structure and content of an XML document.

which sets up the transaction, obtains information about the buyer, and handles the payment and completes the transaction,” as shown in the table below:

API Operation	Description
SetExpressCheckout	Sets up the Express Checkout transaction. You can specify information to customize the look and feel of the PayPal site and the information it displays. You must include the following information: • URL to the page on your website that PayPal redirects to after the buyer logs into PayPal and approves the payment successfully. • URL to the page on your website that PayPal redirects to if the buyer cancels. • Total amount of the order or your best estimate of the total. It should be as accurate as possible.
GetExpressCheckout	Obtains information about the buyer from PayPal, including shipping information.
DoExpressCheckoutPayment	Completes the Express Checkout transaction, including the actual total amount of the order.

Ex.1006, 17

110. In order to set up the Express Checkout transaction, the merchant must invoke the SetExpressCheckout API operation. PayPal Express Checkout explains:

To set up an Express Checkout transaction, [the merchant] must invoke the SetExpressCheckout API operation to provide sufficient information to initiate the payment flow and redirect to PayPal if the operation was successful....

When [the merchant] set[s] up an Express Checkout transaction, [the merchant] specif[ies] values in the SetExpressCheckout request and then call[s] the API. The values [the merchant] specif[ies] control the PayPal page flow and the options available to [the merchant] and [its] buyers. [The merchant] should start by setting up a standard Express

Checkout transaction, which can be modified to include additional options.

Ex.1006, 19; *see also* 17. When invoking SetExpressCheckout, the merchant includes several pieces of information, including the amount of the transaction, the return URL, the cancel URL, and the payment action. *See* Ex.1006, 20. After invoking SetExpressCheckout, the merchant “redirect[s] the buyer’s browser to PayPal and execute[s] the _express-checkout command using the token returned in the SetExpressCheckout response.” Ex.1006, 21. The “token is a value assigned by PayPal that associates the execution of API operations and commands with a specific instance of a user experience flow.” Ex.1006, 16.

111. After the buyer sends her payment information to PayPal’s secure server and approves the purchase, the merchant invokes the GetExpressCheckoutDetails API operation, to retrieve the Payer ID. PayPal Express Checkout explains:

To obtain details about an Express Checkout transaction, [the merchant] can invoke the GetExpressCheckoutDetails API operation....

Although [the merchant is] not required to invoke the GetExpressCheckoutDetails API operation, most Express Checkout implementations take this action to obtain information about the buyer. [The merchant] invoke[s] the GetExpressCheckoutDetails API operation from the page specified by return URL, which [the merchant] set in [its] call to the SetExpressCheckout API. Typically, [the

merchant] invoke[s] this operation as soon as the redirect occurs and use[s] the information in the response to populate [its] review page.

To obtain a buyer's shipping address and Payer ID [the merchant]

1. Specif[ies] the token returned by PayPal when it redirects the buyer's browser to [its] site.

PayPal returns the token to use in the token HTTP request parameter when redirecting to the URL [the merchant] specified in [its] call to the SetExpressCheckout API.

TOKEN=tokenValue

2. Execute[s] the GetExpressCheckoutDetails API to obtain information about the buyer.

3. Access[es] the fields in the GetExpressCheckoutDetails API response.

Ex.1006, 21. The Payer ID is a value returned by GetExpressCheckoutDetails and is a “[u]nique PayPal customer account identification number.” Ex.1007, 100; *see* Ex.1006, 21.

112. “To complete an Express Checkout transaction, [the merchant] must invoke the DoExpressCheckoutPayment API operation.” Ex.1006, 22. When invoking DoExpressCheckout, the merchant specifies the token and Payer ID returned by PayPal's secure server in response to the GetExpressCheckoutDetails function, the amount of the order, and the payment action. *See* Ex.1006, 22.

PayPal's secure server then completes the transaction. Ex.1006, 23.

3. Claim 1

[1.0] A method of processing a payment transaction from a payer to a payee, the method being performed by one or more secure servers, the method comprising:

113. First, PayPal Express Checkout discloses a “method of processing a payment transaction from a payer to a payee.” PayPal Express Checkout is “PayPal’s premier checkout solution, which streamlines the checkout process for buyers and keeps them on the merchant’s site after making a purchase.” Ex.1006, 9. “Express Checkout makes it easier for a buyers [sic] to pay online.” Ex.1006, 9. The diagram below illustrates the PayPal Express Checkout process:



Ex.1006, 10

114. During the transaction process, the buyer:

1. Chooses Express Checkout by clicking **Check out with PayPal**
2. Logs into PayPal to authenticate his or her identity
3. Reviews the transaction on PayPal

NOTE: Optionally, (not shown in the diagram), the buyer can then proceed to review the order on [the merchant’s] site. [The merchant]

can also include other checkout steps, including upselling on [its] Review Order page.

4. Confirms the order and pays from [the merchant's] site

5. Receives an order confirmation

Ex.1006, 10 (emphasis in original).

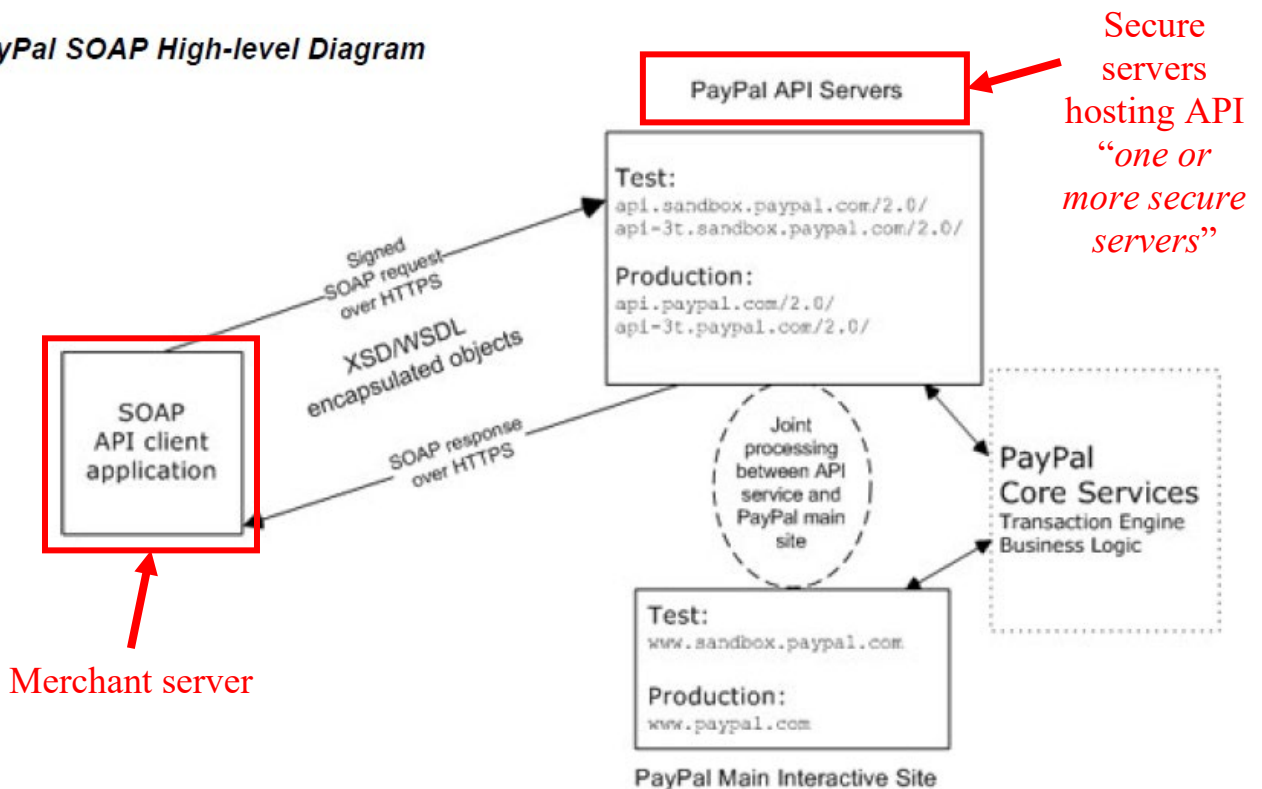
115. “The PayPal API provides three API operations for Express Checkout, which sets up the transaction, obtains information about the buyer, and handles the payment and completes the transaction.” Ex.1006, 17. This includes the DoExpressCheckoutPayment API operation, which “[c]ompletes the Express Checkout transaction, including the actual total of the amount of the order.” Ex.1006, 17; *see also* 21-23 (“To complete an Express Checkout transaction, [the merchant] must invoke the DoExpressCheckoutPayment API operation.”); 26-27 (“Invoke a form on [the merchant's] site that invokes the DoExpressCheckoutPayment API operation on the Sandbox.”).

116. For these reasons, PayPal Express Checkout discloses a method of performing a transaction between the buyer (“*payer*”) and the merchant (“*payee*”).

117. Second, PayPal Express Checkout utilizes a client-server model and is thus “*performed by one or more secure servers.*” PayPal Express Checkout explains that “[t]he PayPal API uses a client-server model in which [the merchant's] website is a client of the PayPal server.” Ex.1006, 61. “The PayPal SOAP API provides programmatic access to PayPal features and services” to the

merchant. Ex.1007, 11. “PayPal SOAP is a combination of client-side and server-side schemas, hardware and software servers, and core services” (Ex.1001, 11), as shown in the diagram below:

PayPal SOAP High-level Diagram



Ex.1007, 11 (annotated)

118. When the merchant sends an API operation request, PayPal’s servers perform the operation and return a response:

To perform a PayPal NVP [Name-Value Pair] API operation, you send an NVP-formatted request to a PayPal NVP server and interpret the response.

In the following diagram, your website generates a request. The request is executed on a PayPal server and the response is returned to [the merchant's] site.

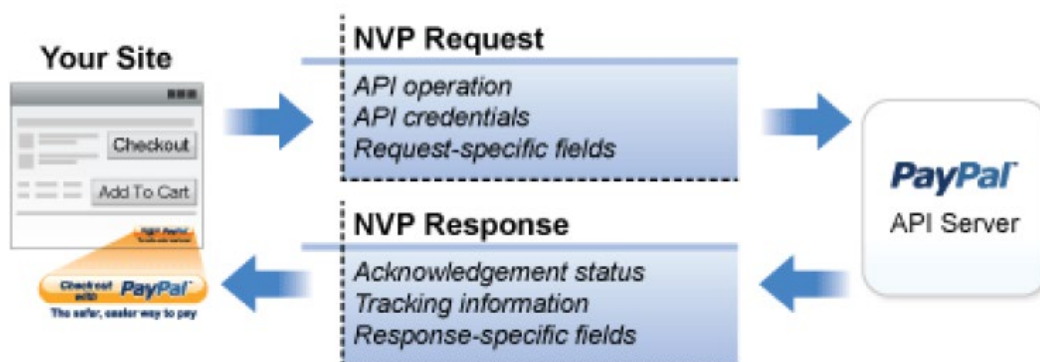
The request identifies

- The name of the API operation to be performed and its version
- Credentials that identify the PayPal account making the request
- Request-specific information that controls the API operation to be performed

A PayPal API server performs the operation and returns a response. The response contains

- An acknowledgement status that indicates whether the operation was a success or failure and whether any warning messages were returned
- Information that can be used by PayPal to track execution of the API operation
- Response-specific information required to fulfill the request

Ex.1006, 62. This is illustrated in the below diagram:



Ex.1006, 62.

Thus, the PayPal API servers perform the method of conducting a transaction.

119. Moreover, PayPal's servers are "*secure*" because:

The PayPal SOAP API service is protected to ensure that only authorized PayPal members use it. There are four levels of security:

1. A required API username (Username field) and API password (Password field)
2. A third required authentication mechanism, which is either one of the following:

Client-side request signing via a PayPal-issued API Certificate

Request authentication via an API Signature included in the request (Signature field)

3. An optional third-party authorization to make the API call on some other account's behalf (the optional Subject field).
4. Secure Sockets Layer (SSL) data transport

A failure of authenticated security at any one of these levels denies access to the PayPal SOAP API service.

Ex.1007, 13; *see also* Ex.1006, 42 ("When redirecting the buyer to PayPal, PayPal recommends that [the merchant] use[s] the HTTPS response 302 'Object Moved' with the URL as the value of the Location header in the HTTPS response. Make sure that **[the merchant] use[s] an SSL-enabled server** to prevent browser warnings about a mix of secure and insecure graphics."). Because communication to PayPal's servers via PayPal SOAP API is protected by requiring authentication of the merchant and SSL data transport, a POSITA would have understood that

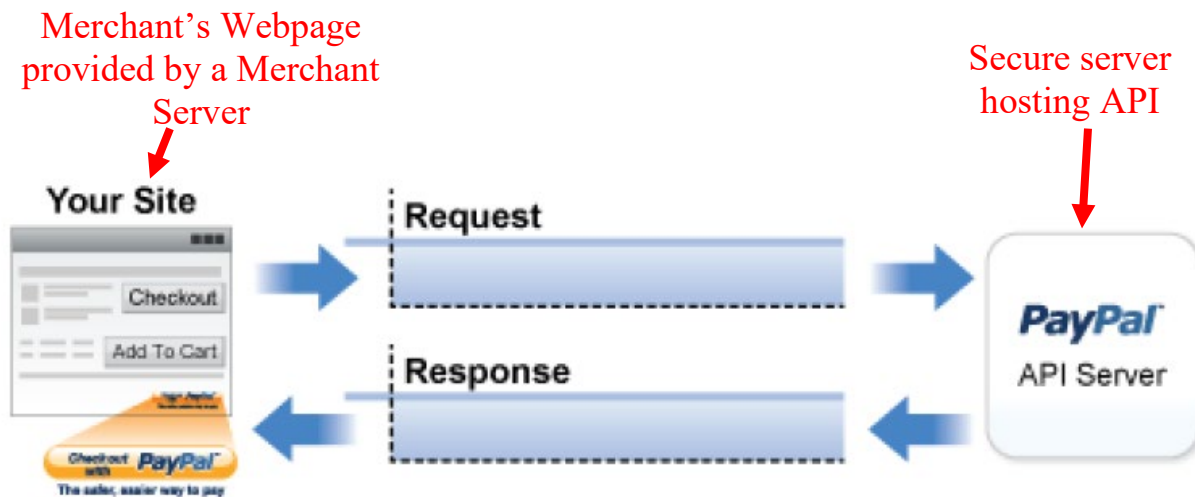
PayPal's servers are "*secure servers.*" Moreover, PayPal Express Checkout refers to PayPal's servers as "secure" throughout. *See e.g.* Ex.1006, 13 ("secure PayPal servers"), Ex.1010, 11 ("PayPal's secure servers").

120. Thus, because PayPal Express Checkout describes a method of conducting a transaction between a buyer and a merchant that is performed by PayPal's secure servers, PayPal Express Checkout renders obvious a "*method of processing a payment transaction from a payer to a payee, the method being performed by one or more secure servers*" as claimed.

[1.1] *providing, by the one or more secure servers to a merchant server providing a webpage to a payer computing system used by the payer, an application programming interface (API) that:*

121. First, PayPal Express Checkout discloses that PayPal's servers act as an API server ("*provid[e] ...an application programming interface (API)*"). PayPal Express Checkout explains that "[t]he PayPal SOAP API provides programmatic access to PayPal features and services." Ex.1007, 11. The PayPal SOAP API allows the merchant to communicate with PayPal's servers. *See* Ex.1006, 61 ("The PayPal API uses a client-server model in which [the merchant's] website is a client of the PayPal server. A page on [the merchant's] website initiates an action on a PayPal API server by sending a request to the server. The PayPal server responds with a confirmation that the requested action was taken or or [*sic*] indicates that an error occurred."); Ex.1007 11-12 ("In an object-oriented processing model, the

interface to SOAP requests/responses is an object in [the merchant's] application's native programming language. [The merchant's] third-party SOAP client generates business-object interfaces and network stubs from PayPal-provided WSDL and XSD files that specify the PayPal SOAP message structure, its contents, and the PayPal API service bindings. A business application works with data in the form of object properties to send and receive data by calling object methods. The SOAP client handles the details of building the SOAP request, sending it to the PayPal service, and converting the response back to an object.”). This is illustrated in the below diagram:



Ex.1006, 61 (annotated)

122. Thus, PayPal's servers provide an API to the merchant.

123. Second, the API is provided to “a merchant server providing a webpage to a payer computing system used by the payer.” PayPal Express

Checkout discloses that the merchant integrates Express Checkout into its website such that the merchant provides its website to the buyer (i.e. payer):

By using these recommended checkout flow and the placement requirements, **[the merchant] optimize[s] [its] website's checkout flow and significantly improve[s] [the] buyer's experience,** increasing conversion. [The] buyers immediately see that [the merchant] ha[s] PayPal and stay[s] on [the] website.

Ex.1010, 7; *see also* Ex.1006, 12 (“To implement Express Checkout, [the merchant] must offer it both as a checkout option and as a payment method. Typically, [the merchant] initiate[s] the Express Checkout flow on [its] shopping cart page and on [its] payment options page. [The merchant] add[s] Express Checkout to [its] existing flow by placing the Checkout with PayPal button on [its] Shopping Cart page and by placing the PayPal mark on [its] Payment Methods page.”); Ex.1006, 15 (“[The merchant] implement[s] Express Checkout flows with Express Checkout buttons, PayPal API operations, PayPal commands, and tokens. The following conceptual diagram identifies the building blocks that [the merchant] use[s] to integrate Express Checkout on [its] website.”)

124. The buyer can access the merchant's website via, for example, a personal computer. *See e.g.* Ex.1006, 55 (Express Checkout can “run[] on personal computers.”).

125. PayPal Express Checkout explains that the merchant's server hosts its

website. Ex.1006, 13-14 (“To inform buyers that PayPal is accepted on [its] website, [the merchant] must place PayPal button and logo images in [its] checkout flow.... PayPal requires that [the merchant] use[s] the Check out with PayPal and the PayPal mark images hosted on secure PayPal servers. When the images are updated, the changes appear automatically in [the merchant’s] application. Do not host copies of the PayPal images locally on **[the merchant’s] servers**. Outdated PayPal images reduces [*sic*] buyer confidence in [the merchant’s] site.”) (emphasis removed and added)⁴, 42 (“When redirecting the buyer to PayPal, PayPal recommends that [the merchant] use[s] the HTTPS response 302 ‘Object Moved’ with the URL as the value of the Location header in the HTTPS response. Make sure that **[the merchant] use[s] an SSL-enabled server** to prevent browser warnings about a mix of secure and insecure graphics.”); Ex.1007, 12 (“An API certificate is suitable **if [the merchant] ha[s] complete control over [its] own web server**...An API signature is suitable for use with Microsoft Windows web servers or other shared web server configurations.”), 13 (“First-Party Access” “A

⁴ If using outdated PayPal images, the buyer may be concerned that clicking on the link may not redirect their browser to PayPal’s website, but may instead redirect their browser to a bad actor’s website. This may reduce the buyer’s confidence in the merchant’s website.

company makes API calls itself from its own server to PayPal's server.

Example: A staff programmer for a merchant's company obtains a PayPal-issued API certificate file and makes API calls for the company from the company's own web server.”)

126. Moreover, it was well-known in the art for a merchant's website to be hosted by a server. Ex.1015, 4:50-57 (“The merchant site is the e-commerce site (e.g., website) of the merchant. The merchant (100) and merchant server (120) in the figures are associated with the merchant site. The merchant site is associated with a client-side (client side) application and a server-side (server side) application. In one preferred embodiment, the merchant site includes Merchant Server (120), and the server-side application executes on the Merchant Server (120).”), 8:50-52 (“The Merchant Servers (120) serve a Stripe.js enabled Payment Form (110) to the Merchant Customer's Browser (210).”); Ex.1018, [0003] (“A website is, in general terms, a server application running on a computer which accepts connections from client programs. Client programs, such as browsers, allow a remote user to access the information stored on the website.”); Ex.1020, [0027] (“Such web servers include the aforementioned merchant computer system 20, also referenced interchangeably herein as the merchant system. The merchant system 20 is likewise understood to be a computer system that includes a central processor, a memory, and input/output devices, though loaded with computer-

executable Software instructions of a web server and those that implement one or more of the disclosed methods for securely processing an online transaction. In further detail, the merchant system 20 includes an HTTP server 24, along with a database 26 that stores, for example, product catalog data. There also is a web application server 27 that executes the functionality of the contemplated methods.”).

127. Thus, PayPal Express Checkout discloses a merchant server providing a webpage to the buyer’s computer system.

128. As a consequence, because PayPal Express Checkout discloses PayPal’s servers provide an API to the merchant server, which hosts a website that a buyer accesses via her personal computer (or other device), PayPal Express Checkout renders obvious “*providing, by the one or more secure servers to a merchant server providing a webpage to a payer computing system used by the payer, an application programming interface (API)*” as claimed.

[1.1.1] provides financial account registration and token retrieval functions that can be executed to process the payment transaction;

129. PayPal Express Checkout provides an Example Transaction using PayPal Sandbox rather than the live PayPal website:

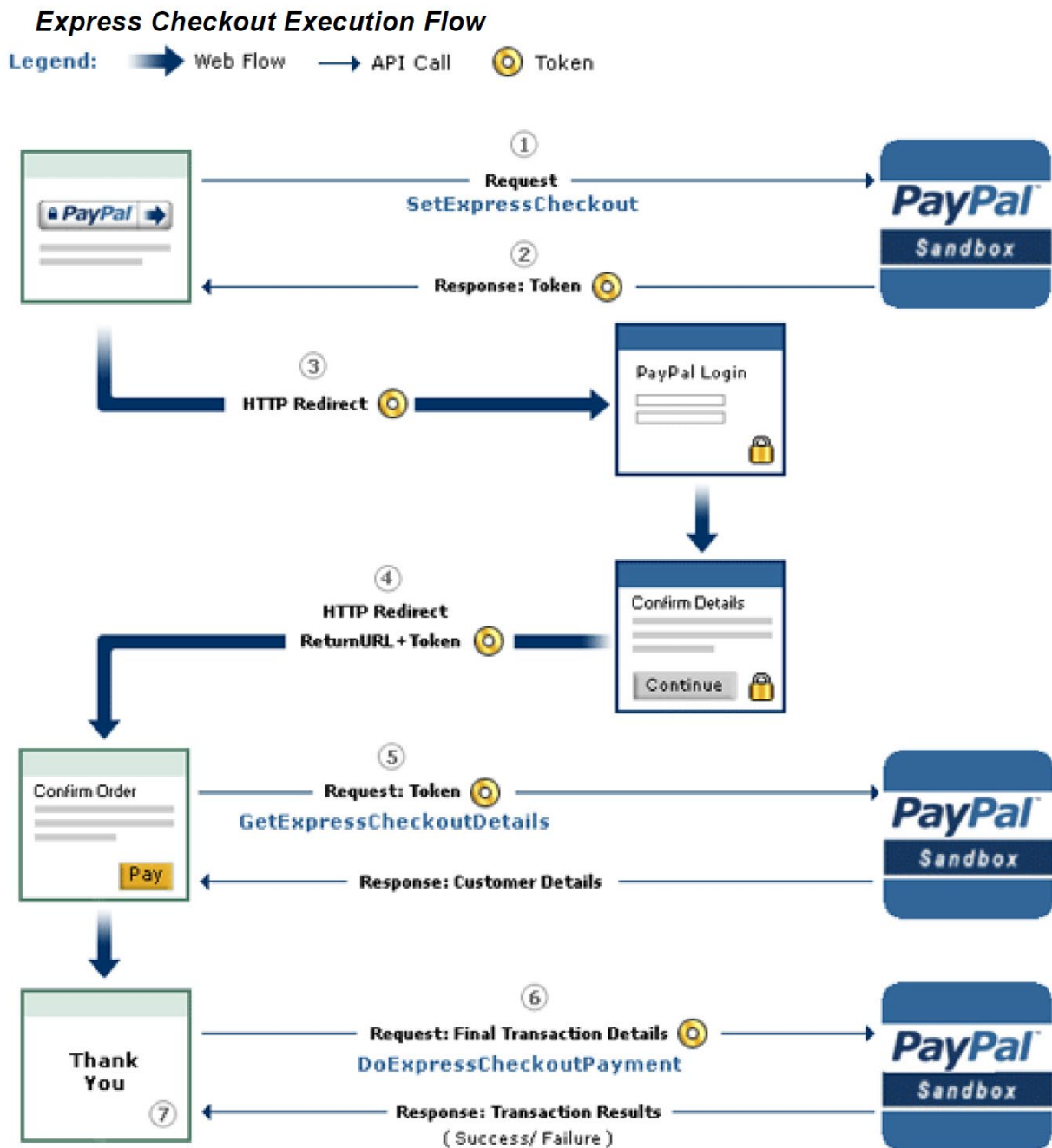
[A merchant] can test [its] Express Checkout integration in the Sandbox.

This example shows how to simulate [the merchant's] web pages using HTTP forms and supplying the values for API operations from these forms. [The merchant] can use this strategy for [its] initial testing; however, for more complete testing, [the merchant] will want to replace these forms with [its] web pages containing actual code.

The following diagram shows the Express Checkout execution flow, which uses the Sandbox as the API server. The pages on the left represent [the merchant's] site.

Ex.1006, 23.

130. The Example Transaction using PayPal's Sandbox is provided below:



Ex.1006, 24

PayPal Sandbox is a tool provided by PayPal Express Checkout that allows merchants to test out various PayPal features and/or products in a self-contained environment before launching them on their website:

The PayPal Sandbox is a self-contained environment within which [a merchant] can prototype and test PayPal features and APIs. **The PayPal Sandbox is an almost identical copy of the live PayPal website.** Its purpose is to give developers a shielded environment for testing and integration purposes and to help avoid problems that might occur while testing PayPal integration solutions on the live site. Before moving any PayPal-based application into production, [the merchant] should test the application in the Sandbox to ensure that it functions as [the merchant] intend[s] and within the guidelines and standards set forth by the PayPal Developer Network (PDN).

Ex.1008, 9. Although the PayPal Sandbox is “an almost identical copy of the live PayPal website,” there are several differences. Ex.1008, 9-11. These differences are described in the table below:

TABLE 1.1 Differences between PayPal Sandbox, and Live PayPal

	PayPal Sandbox	Live PayPal Website and API Service
Type of PayPal Accounts	Depending on the feature you want to develop and test, you need a Personal, Business, or Premier account.	Personal, Business, or Premier account
Site logos in upper left corner	https://www.sandbox.paypal.com 	https://www.paypal.com 

TABLE 1.1 Differences between PayPal Sandbox, and Live PayPal

	PayPal Sandbox	Live PayPal Website and API Service
NVP API Servers	https://api.sandbox.paypal.com/nvp/	For API Certificate security: https://api.paypal.com/nvp/ For API Signature security: https://api-3t.paypal.com/nvp/
SOAP API Servers	https://api.sandbox.paypal.com/2.0/	For API Certificate security: https://api.paypal.com/2.0/ For API Signature security: https://api-3t.paypal.com/2.0/
Business roles	You fill all roles you need to test: merchant, buyer, and seller.	Real-world people fill these roles.
Company and people's names and postal addresses	Completely fictitious. Before you begin working with the Sandbox, create the details for all the business roles you must fulfill. The Sandbox simulates verification of postal addresses and names.	Real companies' and people's names and postal addresses.
Email addresses and email inboxes	The Sandbox has a special-purpose email inbox for your testing, contained in the Sandbox itself.	Real email address and inbox for each business role
Bank account and credit card numbers	The Sandbox creates bank accounts, credit card numbers, and CVV2 numbers you need in order to develop and test; all of which are fictitious and only used within the Sandbox. The Sandbox simulates the verification of these numbers. Transactions do not affect real accounts and actual money is never exchanged.	Actual verification of bank account numbers, credit card numbers, and CVV2 numbers
Social Security Number for Billing Agreements	111- <i>nn-nnnn</i>	Real social security numbers
PayPal transactions	The Sandbox creates all fictitious bank accounts, credit card numbers, and CVV2 numbers you need for development and testing. The Sandbox simulates the verification of these numbers.	Live transactions, cleared by live PayPal processes
Fraud detection	Fraud detection is not enabled for the Sandbox.	Full protection through PayPal's fraud detection
Digital certificates	After you request digital certificates for use with the PayPal Web Services API, the Sandbox automatically generates them. They are available for immediate downloading.	To safeguard your and your customers' security, requests for digital certificates for use with the Live PayPal Web Services API must be verified by PayPal before they are issued. You are notified in email when your request has been approved.

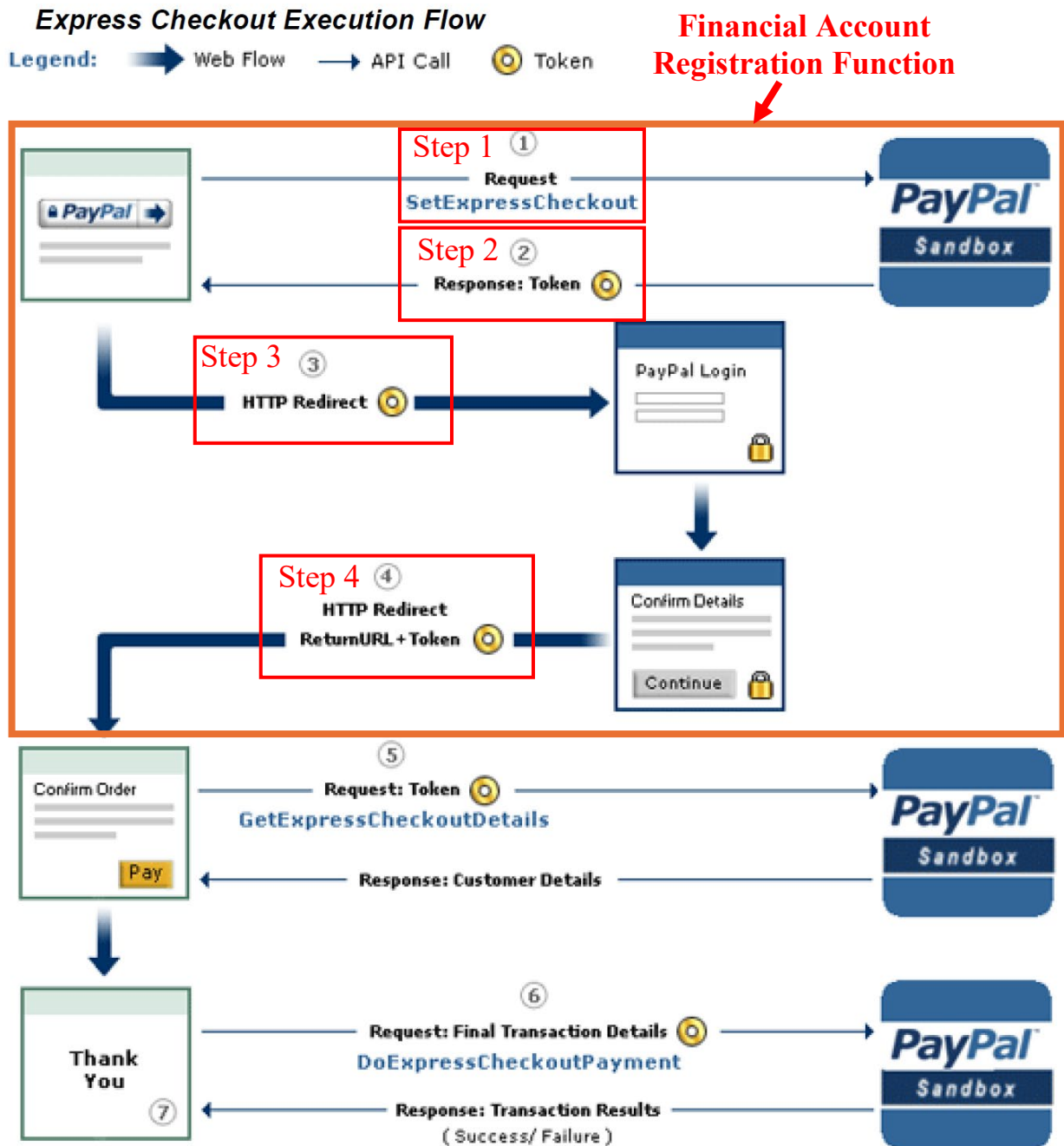
TABLE 1.1 Differences between PayPal Sandbox, and Live PayPal

	PayPal Sandbox	Live PayPal Website and API Service
PayPal Merchant Features supported	All features of the live PayPal website, except closing an account, auction features, monthly statements, shipping preferences, and PayPal Shops.	

Ex.1008, 9-11

131. Thus, the Example Transaction illustrated above is representative of an actual Express Checkout operation conducted between a merchant and a buyer. See Ex.1006, 24.

132. First, PayPal's API provides a "*financial account registration*" function, as shown in Steps 1-4 of the Example Transaction below:



Ex.1006, 24 (annotated)

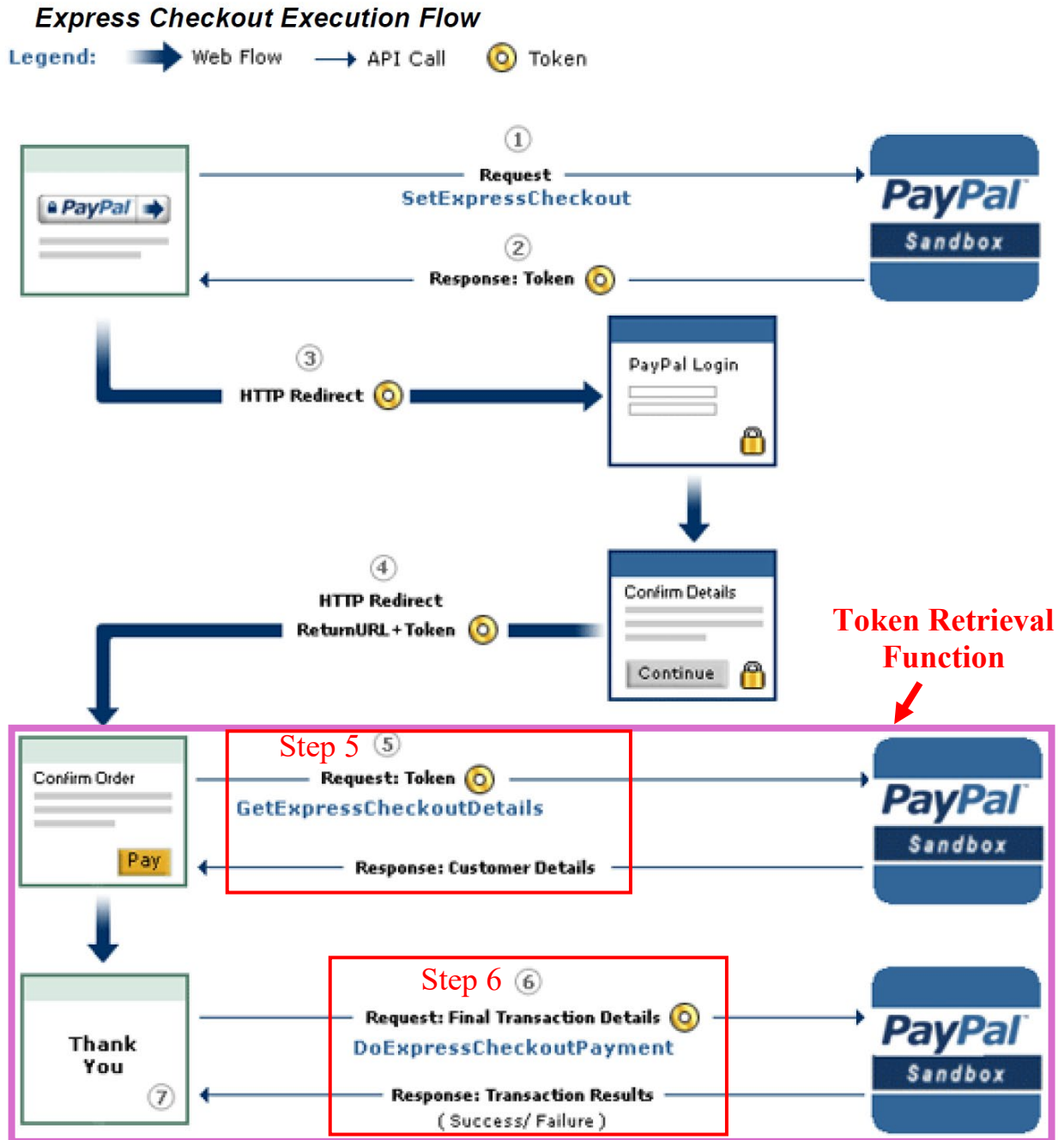
133. Steps 1-4 register the user for the purpose of performing a financial transaction and, thus, teach “a financial account registration function” as claimed.

134. The PayPal API provides the SetExpressCheckout API operation,

which “[s]ets up the Express Checkout transaction.” Ex.1006, 17. Moreover, “PayPal provides a command that [the merchant] use[s] when redirecting [the] buyer’s browser to PayPal.” Ex.1006, 17. “This command enables [the] buyer to log into PayPal to approve an Express Checkout payment.” Ex.1006, 17.

135. When a buyer initiates a transaction on the merchant’s website, the merchant sets up the Express Checkout transaction by “invok[ing] the SetExpressCheckout API operation to provide sufficient information to initiate the payment flow and redirect to PayPal if the operation was successful.” Ex.1006, 19; *see also* 24-25 (Step 1, described in more detail below). SetExpressCheckout returns a token. Ex.1006, 17, 21; *see* 25 (Step 2, described in more detail below). The merchant “redirect[s] the buyer’s browser to PayPal and execute[s] the - express-checkout command using the token returned in the SetExpressCheckout response.” Ex.1006, 21; *see also* 25 (Step 3, described in more detail below). “If the buyer approves the payment, PayPal redirects to the success URL.” Ex.1006, 17; 25-26 (Step 4, described in more detail below). Thus, the financial account registration function is provided by SetExpressCheckout and the _ExpressCheckout command, which are used to set up the Express Checkout transaction and redirect the buyer’s browser to PayPal so that the buyer can send sensitive financial account information to PayPal’s servers without sending it to the merchant’s server.

136. Second, PayPal's API provides a "token retrieval function[]," as shown in Steps 5-6 of the Example Transaction below:



Ex.1006, 24 (annotated)

137. Steps 5-6 utilize a Payer ID (“*token*”) provided by PayPal’s server and, thus, teach a “*token retrieval function*” as claimed.

138. “The PayPal API provides three API operations for Express Checkout,” including the GetExpressCheckoutDetails and DoExpressCheckoutPayment API operations. Ex.1006, 17.

139. After the buyer “approves the payment” and “PayPal redirects to the success URL,” the merchant “invoke[s] the GetExpressCheckoutDetails API operation.” Ex.1006, 17, 21; *see also* 26 (Step 5, described in more detail below). The merchant “take[s] this action to obtain information about the buyer,” including the Payer ID (“*token*”). Ex.1006, 21; *see also* 26 (Step 5, described in more detail below). The Payer ID is a “[u]nique PayPal customer account identification number” that is used to complete the transaction, as described below. Ex.1007, 100; *see* Ex.1006, 17 (“The buyer’s unique identifier (Payer ID)”). The merchant calls GetExpressCheckoutDetails by “[s]pecify[ing] the token returned by PayPal when [PayPal] redirects the buyer’s browser to [the merchant’s] site.” Ex.1006, 21.

140. After receiving the Payer ID, the merchant completes the transaction by “invok[ing] the DoExpressCheckoutPayment API operation.” Ex.1006, 21; *see* Ex.1006, 21-22 (The merchant “may call the GetExpressCheckoutDetails API before calling the DoExpressCheckoutPayment API.”); *see also* 26-27 (Step 6, described in more detail below). The merchant calls DoExpressCheckoutPayment

by “[s]pecify[ing] the token returned by PayPal when it redirects the buyer’s browser back to [the merchant’s] site,” “[s]pecify[ing] the Payer ID,” where “[the merchant] can obtain the Payer ID by calling the GetExpressCheckoutDetails API,” “[s]pecify[ing] the amount of the order,” and “[s]pecify[ing] the payment action.” Ex.1006, 22; *see also* 26-27 (Step 6, described in more detail below).

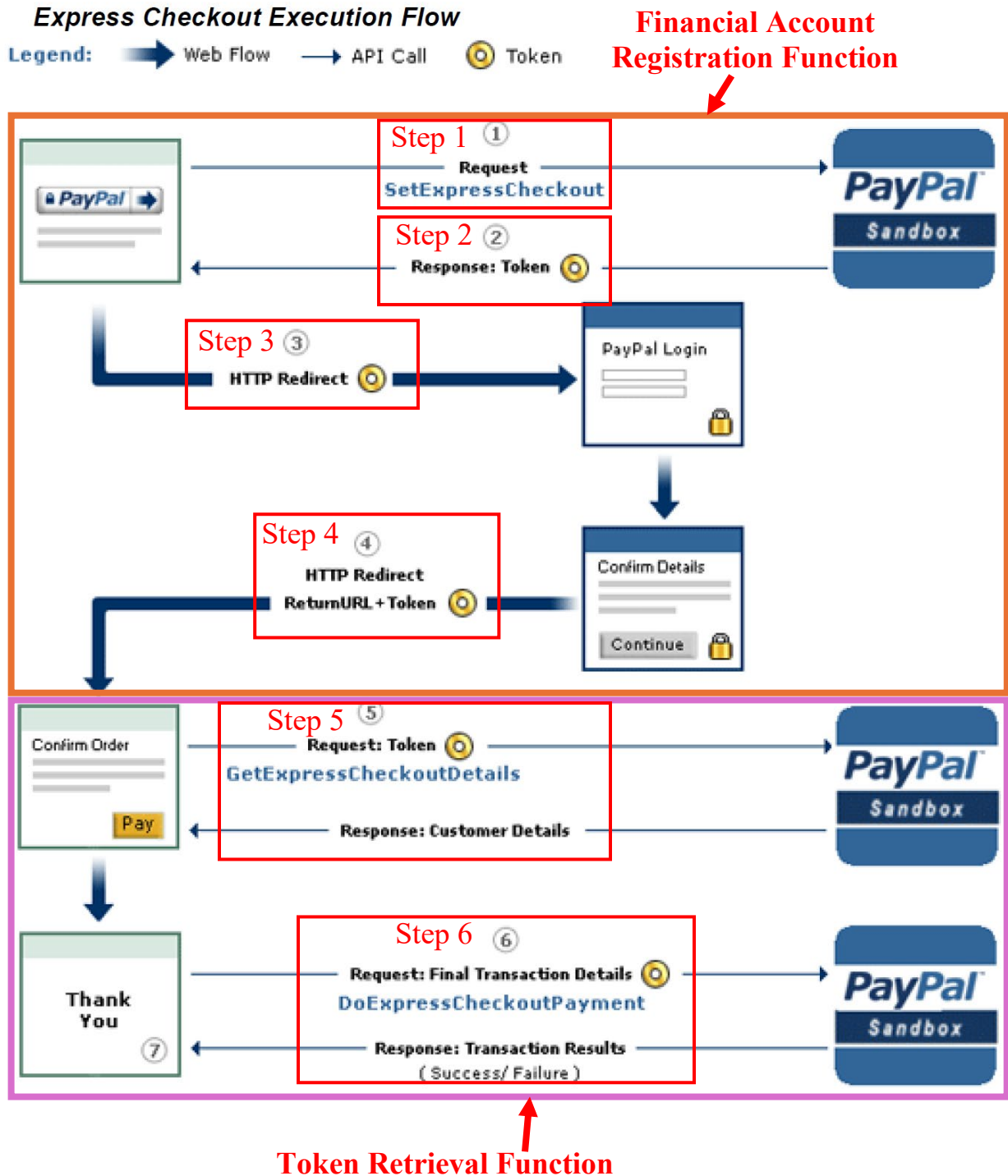
141. Thus, the merchant receives the Payer ID by invoking GetExpressCheckoutDetails and completes the transaction using the Payer ID by invoking DoExpressCheckoutPayment.

142. Thus, because PayPal Express Checkout provides a financial account registration function that includes the SetExpressCheckout API operation and the _ExpressCheckout command and provides a token retrieval function that includes the SetExpressCheckoutDetails and DoExpressCheckoutPayment API operations, PayPal Express Checkout renders obvious “*provides financial account registration and token retrieval functions that can be executed to process the payment transaction*” as claimed.

[1.1.2] provides access to the financial account registration and token retrieval functions to the merchant server;

143. As explained above in limitation [1.1.1], PayPal Express Checkout discloses that PayPal’s API servers provide the SetExpressCheckout API operation and the _ExpressCheckout command (“*financial account registration*” function) and the GetExpressCheckoutDetails and DoExpressCheckoutPayment API

operations (“*token retrieval function*[/]”). The Example Transaction below illustrates these functions:



Ex.1006, 24 (annotated)

144. PayPal’s API provides merchant servers access to these functions:

PayPal provides a command that [the merchant] use[s] when redirecting [the] buyer’s browser to PayPal. This command enables [the] buyer to log into PayPal to approve an Express Checkout payment. When [the merchant] redirect [the] buyer’s browser to PayPal, [the merchant] must specify the _ExpressCheckout command for Express Checkout. [The merchant] also specif[ies] the token that identifies the transaction, which was returned by the SetExpressCheckout API operation.

Ex.1006, 17.

145. Moreover, “[t]he **PayPal API provides three API operations** for Express Checkout,” which include SetExpressCheckout, GetExpressCheckoutDetails, and DoExpressCheckoutPayment. Ex.1006, 17. “The PayPal SOAP API **provides** programmatic access to PayPal features and services,” including the three Express Checkout API operations. Ex.1007, 11; *see also* 12 (“API Calls” “PayPal Application Programming Interface services, by which companies can make payments, search transactions, refund payments, view transaction information, and other business functions.”). The merchant can invoke these API operations to process the transaction:

To set up an Express Checkout transaction, [the merchant] must invoke the SetExpressCheckout API operation to provide sufficient

information to initiate the payment flow and redirect to PayPal if the operation was successful....

When [the merchant] set[s] up an Express Checkout transaction, [the merchant] specif[ies] values in the SetExpressCheckout request and then call[s] the API. The values [the merchant] specif[ies] control the PayPal page flow and the options available to [the merchant] and [the] buyers. [The merchant] should start by setting up a standard Express Checkout transaction, which can be modified to include additional options.

Ex.1006, 19.

To obtain details about an Express Checkout transaction, [the merchant] can invoke the GetExpressCheckoutDetails API operation....

Although [the merchant] [is] not required to invoke the GetExpressCheckoutDetails API operation, most Express Checkout implementations take this action to obtain information about the buyer. [The merchant] invoke[s] the GetExpressCheckoutDetails API operation from the page specified by return URL, which [it] set in [its] call to the SetExpressCheckout API. Typically, [the merchant] invoke[s] this operation as soon as the redirect occurs and use[s] the information in the response to populate [its] review page.

Ex.1006, 21.

To complete an Express Checkout transaction, [the merchant] must invoke the DoExpressCheckoutPayment API operation....

In the simplest case, [the merchant] set[s] the total amount of the order when [it] call[s] the SetExpressCheckout API. However, [the merchant]

can change the amount before calling the DoExpressCheckoutPayment API if [it] did not know the total amount when [it] called the SetExpressCheckout API.

Ex.1006, 21-22.

146. Thus, because PayPal’s API servers provide functions for account registration and token retrieval, PayPal Express Checkout renders obvious “*provides access to the financial account registration and token retrieval functions to the merchant server*” as claimed.

[1.1.3] receives, from the merchant server via the API, at least one data element associated with the payer and a payment amount from the payer to the payee;

147. First, PayPal Express Checkout discloses a token (“*at least one data element*”) that is “*associated with the payer.*” PayPal Express Checkout explains that “Express Checkout uses a token to control access to PayPal and execute Express Checkout API operations.” Ex.1006, 18. “The SetExpressCheckout API operation returns a token, **which is used by other Express Checkout API operations and by the _ExpressCheckout command to identify the transaction.**” Ex.1006, 18. The “token is a value assigned by PayPal **that associates the execution of API operations and commands with a specific instance of a user experience flow.**” Ex.1006, 16. Thus, the token is a “*data element associated with the payer.*”

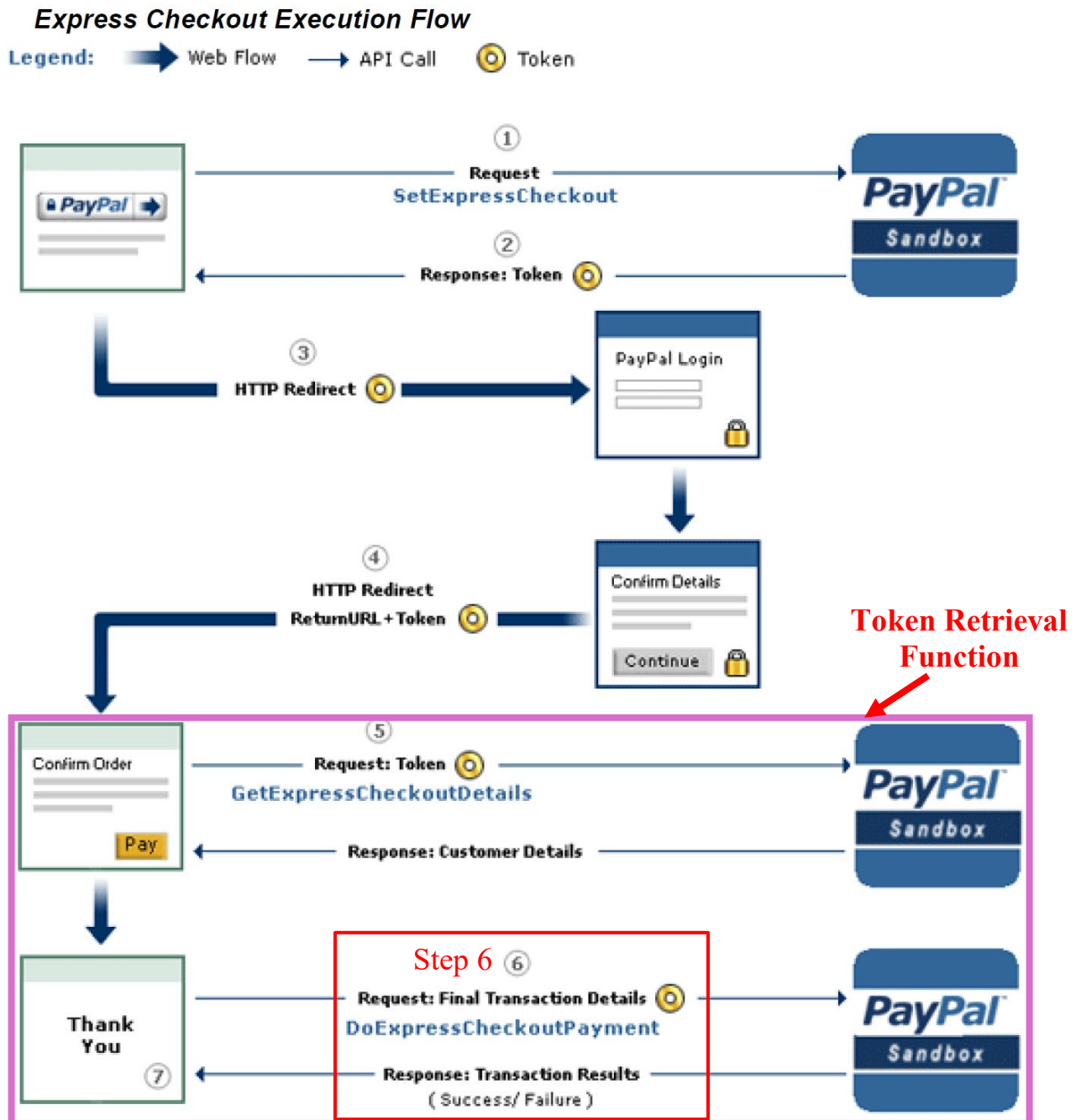
148. Second, PayPal’s servers receive the token (“*data element associated*

with the payer”) and “*a payment amount from the payer to the payee.*” In PayPal Express Checkout, the token is used to invoke the DoExpressCheckoutPayment API operation, which completes the Express Checkout transaction. Ex.1006, 21.

When calling DoExpressCheckoutPayment, the merchant “[s]pecif[ies] **the token** returned by PayPal when it redirects the buyer’s browser back to [its] site,”

“[s]pecif[ies] the Payer ID,” where “[the merchant] can obtain the Payer ID by calling the GetExpressCheckoutDetails API,” “[s]pecif[ies] **the amount of the order,**” and “[s]pecif[ies] the payment action.” Ex.1006, 22.

DoExpressCheckoutPayment is Step 6, which is a part of the token retrieval function, as illustrated in the Example Transaction below:



Ex.1006, 24 (annotated)

149. Thus, because PayPal Express Checkout discloses that the merchant specifies the token identifying the transaction and the amount of the order in the

DoExpressCheckoutPayment request, PayPal Express Checkout renders obvious “receives, from the merchant server via the API, at least one data element associated with the payer and a payment amount from the payer to the payee” as claimed.

[1.1.4] authenticates the payee; and

150. PayPal Express Checkout provides several levels of security to prevent unauthorized access of the PayPal servers, including requiring authentication of the merchant (“payee”):

The PayPal SOAP API service is protected to ensure that only authorized PayPal members use it. There are four levels of security:

1. A required API username (Username field) and API password (Password field)
2. A third required authentication mechanism, which is either one of the following:
 - Client-side request signing via a PayPal-issued API Certificate
 - Request authentication via an API Signature included in the request (Signature field)
3. An optional third-party authorization to make the API call on some other account’s behalf (the optional Subject field).
4. Secure Sockets Layer (SSL) data transport

A failure of authenticated security at any one of these levels denies access to the PayPal SOAP API service.

Ex.1007, 13.

151. Thus, the merchant (“payee”) must provide API credentials to PayPal

to use Express Checkout:

To use the PayPal API, [the merchant] must have API credentials that identify [it] as a PayPal Business account holder who is authorized to perform various API operations. Although [the merchant] can use either an API signature or a certificate for credentials, PayPal recommends [the merchant] use a signature....

An API signature consists of an API username along with an associated API password and signature, all of which are assigned by PayPal. [the merchant] need[s] to include this information whenever [it] execute[s] a PayPal API operation.

[The merchant] must have a PayPal Business account to create a signature.

Ex.1006, 64. PayPal Express Checkout provides exemplary API credentials, as shown in the screenshot below:

[Log Out](#) | [Help](#) | [Security Center](#)

PayPal

[My Account](#) | [Send Money](#) | [Request Money](#) | [Merchant Services](#) | [Auction Tools](#) | [Products and Services](#)

View or Remove API Signature [Back to profile](#)

For **preconfigured shopping carts**: Copy and paste the API username, password, and signature into your shopping cart configuration or administration screen.

For **building custom shopping carts**: Store the following credential information in a secure location with limited access.

Credential	API Signature
API Username	dangarcia_api1.paypal.com
API Password	TQJQTNEXUBMNKCH6
Signature	AFcWxV21C7fd0v3bYYRCpSSRI31ACUgV1hSSbCaoz-crGvgPCTDjuhW
Request Date	Mar 15, 2010 14:20:28 PDT

[About Us](#) | [Contact Us](#) | [Fees](#) | [Legal Agreements](#) | [Privacy](#) | [Site Feedback](#) (+)

Copyright © 1999-2010 PayPal. All rights reserved.

Ex.1006, 65

152. The merchant provides these API credential in each request (i.e. communication) it sends to PayPal's servers:

[The merchant] must specify API credentials in each request to execute a PayPal API operation. When [the merchant] execute[s] a PayPal API operation, [it] use[s] credentials, such as a signature, to authenticate that [it] [is] requesting the API operation....

To enable PayPal to authenticate [its] request[, the merchant:]

1. Specif[ies] the API user name associated with [its] account.

USER=API_username

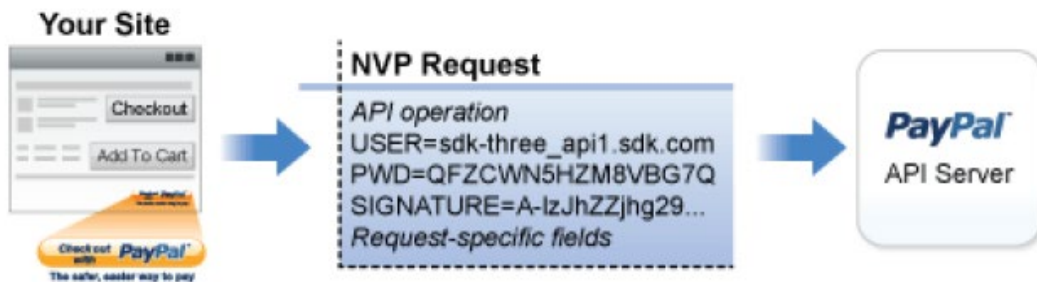
2. Specif[ies] the password associated with the API user name.

PWD=API_password

3. If [the merchant] [is] using an API signature and not an API certificate, specif[ies] the API signature associated with the API username.

SIGNATURE=API_signature

Ex.1006, 66-67; *see also* Ex.1006, 65 (“The Name-Value Pair request format specifies the API operation to perform, **credentials that authorize PayPal to access [the merchant’s] account**, and fields containing additional information to be used in the request.”). This is illustrated in the below diagram:

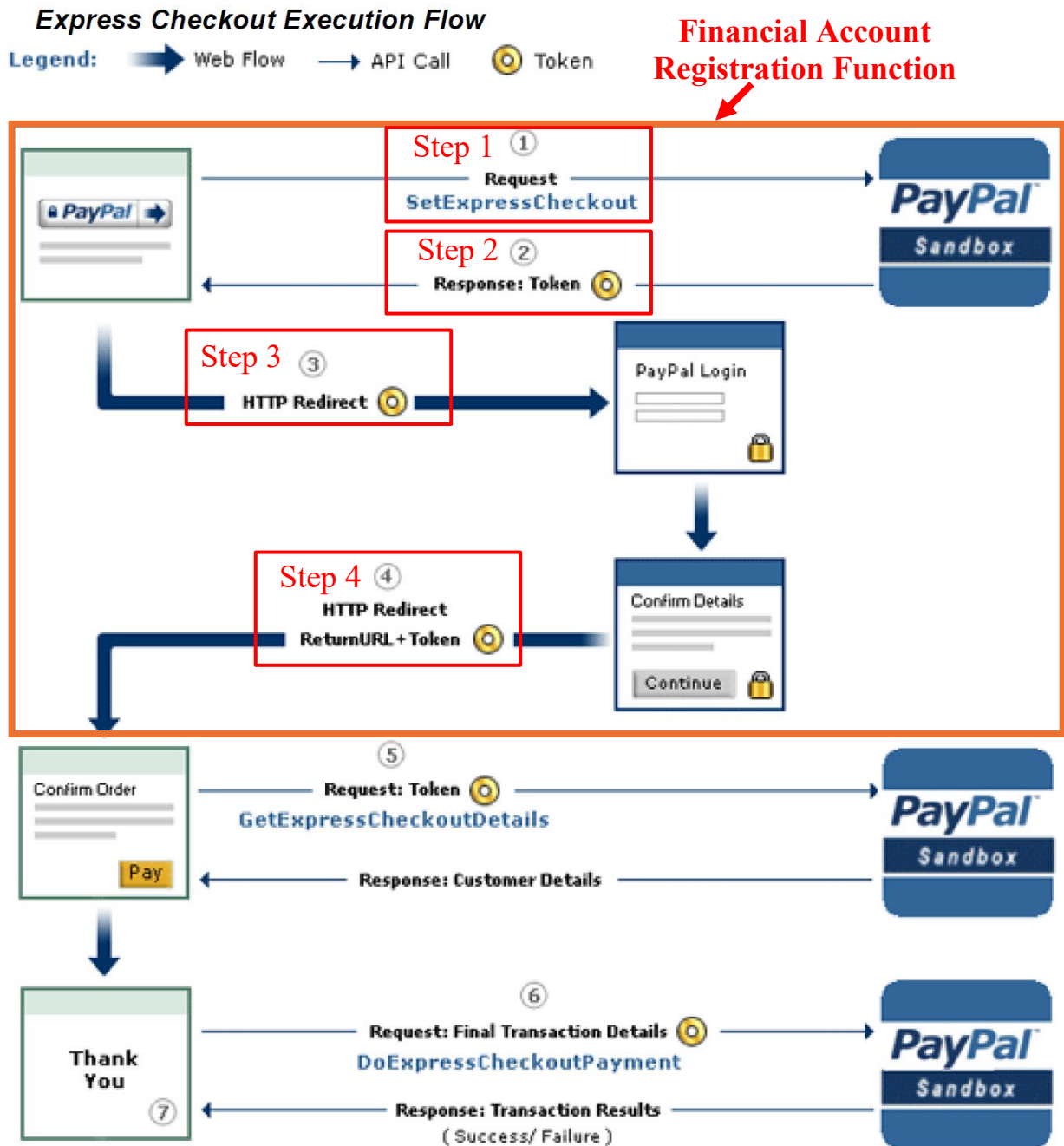


Ex.1006, 67

153. Thus, because PayPal Express Checkout discloses that PayPal’s API server requires authentication of the merchant, PayPal Express Checkout renders obvious “*authenticates the payee*” as claimed.

[1.1.5] executes the financial account registration function, upon initiation by the merchant server, by:

154. As explained above in limitations [1.1.1]-[1.1.2], PayPal Express Checkout discloses that financial account registration function includes the SetExpressCheckout API operation and the _ExpressCheckout command, as shown in the Example Transaction below:



Ex.1006, 24 (annotated)

155. The Express Checkout functions are executed upon being invoked by the merchant server. “To set up an Express Checkout transaction, **[the merchant] must invoke SetExpressCheckout...** to provide sufficient information to initiate

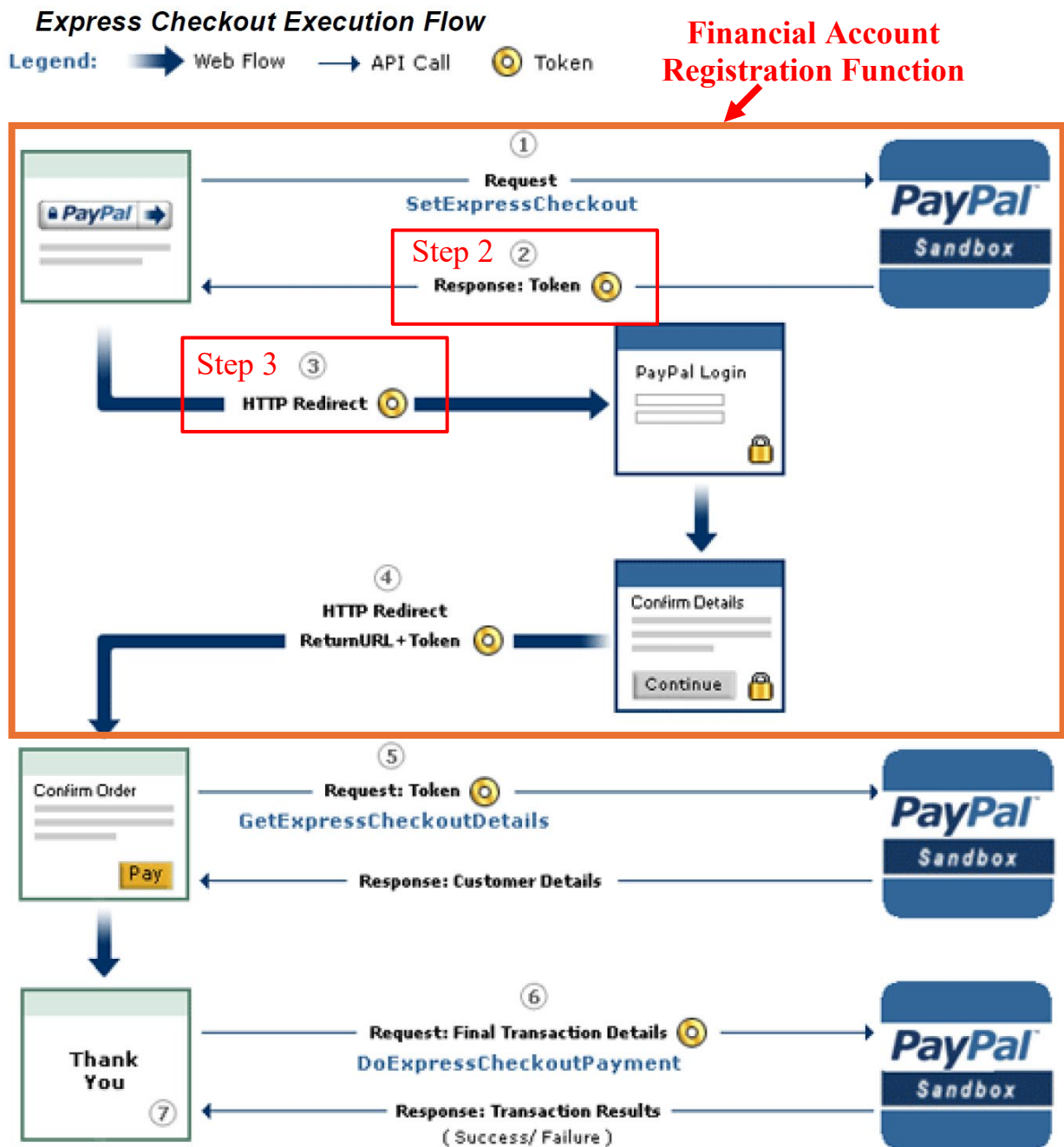
the payment flow and redirect to PayPal if the operation was successful.” Ex.1006, 19. Moreover, “[w]hen [the merchant] redirect[s] [the] buyer’s browser to PayPal, [it] must specify the _ExpressCheckout command for Express Checkout.” Ex.1006, 17.

156. Thus, because the merchant invokes SetExpressCheckout and the _ExpressCheckout command (“*financial account registration function*”), PayPal Express Checkout renders obvious “*executes the financial account registration function, upon initiation by the merchant server*” as claimed.

[1.1.5.1] *generating a uniform resource locator (URL), for establishing a secure socket layer connection via the internet between the secure server and the payer computing system*

157. First, PayPal’s servers cause a URL to be generated by sending the merchant a token that is to be combined with a static, predefined URL (“*generating a uniform resource locator (URL)*”).

158. After the merchant invokes SetExpressCheckout, the server sends the merchant a “token that is used in subsequent steps.” Ex.1006, 25. In response to receiving the token, the merchant redirects the user’s browser using a static URL, the token returned by SetExpressCheckout, and _ExpressCheckout. Ex.1006, 25 (“If the [SetExpressCheckout] operation was successful, [the merchant] use[s] the token and redirect[s] [the] browser to the Sandbox to log in.”). These are illustrated as Steps 2 and 3 in the Example Transaction below:



Ex.1006, 24 (annotated)

159. Specifically, in Step 1 of the Example Transaction, the merchant sets up the transaction by invoking `SetExpressCheckout`. Ex.1006, 24-25. An example `SetExpressCheckout` request is provided below:

```
<form method=post action=https://api-3t.sandbox.paypal.com/nvp>
  <input type=hidden name=USER value=API_username>
  <input type=hidden name=PWD value=API_password>
  <input type=hidden name=SIGNATURE value=API_signature>
  <input type=hidden name=VERSION value=XX.0>
  <input type=hidden name=PAYMENTREQUEST_0_PAYMENTACTION
    value=Authorization>
  <input name=PAYMENTREQUEST_0_AMT value=19.95>
  <input type=hidden name=RETURNURL
    value=http://www.YourReturnURL.com>
  <input type=hidden name=CANCELURL
    value=http://www.YourCancelURL.com>
  <input type=submit name=METHOD value=SetExpressCheckout>
</form>
```

Ex.1006, 25

160. In Step 2 of the Example Transaction, PayPal's servers return a token to the merchant. Ex.1006, 25. An example SetExpressCheckout response is provided below:

```
TIMESTAMP=2007%2d04%2d05T23%3a23%3a07Z
&CORRELATIONID=63cdac0b67b50
&ACK=Success
&VERSION=XX%2e000000
&BUILD=1%2e0006
&TOKEN=EC%2d1NK66318YB717835M
```

← Token

Ex.1006, 25 (annotated)

161. In Step 3 of the Example Transaction, the merchant uses the token returned by SetExpressDetails to redirect the user's browser along with _ExpressCheckout. Ex.1006, 25. An example redirect is provided below:



The diagram shows a URL: `https://www.sandbox.paypal.com/cgi-bin/webscr?cmd=_express-checkout&token=EC-1NK66318YB717835M`. Red arrows point to specific parts of the URL with labels: 'HTTPS request' points to the 'https' protocol; 'URL' points to the entire string; '_ExpressCheckout Command' points to the 'cmd=_express-checkout' parameter; and 'Token' points to the '&token=EC-1NK66318YB717835M' parameter.

Ex.1006, 25 (annotated).

162. As explained above in Section VI.C, a POSITA would have understood “*generating... a URL*” in limitation [1.1.5.1] to include sending the merchant a token to be combined with a predefined static URL. Thus, PayPal’s servers generate a URL by sending the token in Step 2, which prompts the merchant server to combine that token with a static URL (“*generating a uniform resource locator (URL)*”). The static URL is described in further detail below with respect to [1.1.5.1b].

163. Second, the generated URL is used for “*establishing a secure socket layer connection via the internet between the secure server and the payer computing system.*”

164. PayPal Express Checkout explains that communications to and from PayPal’s server use a URL to establish an SSL connection between the sender and PayPal’s servers. See Ex.1006, 67-68 (“All requests to execute PayPal API operations sent via HTTP must be URL encoded. The PayPal NVP API uses the HTTP protocol to send requests and receive responses from a PayPal API server.

[The merchant] must encode all data sent using the HTTP protocol because data that is not encoded could be misinterpreted as part of the HTTP protocol instead of part of the request. Most programming languages provide a way to encode strings in this way. [The merchant] should consistently URL encode the complete API request; otherwise, [it] may find that unanticipated data causes an error. Note: An HTTP form is automatically URL encoded by most browsers.”), 70 (“All responses to HTTP POST operations used by the PayPal NVP API must be decoded. The PayPal NVP API uses the HTTP protocol to send requests and receive responses from a PayPal API server. [The merchant] must decode all data returned using the HTTP protocol so that it can be displayed properly. Most programming languages provide a way to decode strings. Note: Most browsers decode responses to HTTP requests automatically.); Ex.1007, 13 (“The PayPal SOAP API service is protected to ensure that only authorized PayPal members use it,” including “Secure Sockets Layer (SSL) data transport.”). Thus, a POSITA would have understood that the connection between the buyer’s browser and the PayPal servers would use a URL to establish an SSL connection therebetween.

165. Moreover, it was well-known in the art that HTTPS requests establish an SSL connection between the requesting browser and the destination server over the Internet. For example, HTML & Web Design explains that “[t]o safeguard sensitive information (such as usernames and passwords), ask the ISP hosting your

site to set up the Secure Socket Layer (SSL) protocol.” Ex.1014, 565. “After the ISP installs the security certificate for your site on the Web server, the Web server and Web browser will encrypt the HTTP messages before sending them across the Internet.” Ex.1014, 565. Compared to a URL using HTTP, “[t]he only difference you will notice is that your secure URLs will start with https:// instead of the standard (nonsecure) http://.” Ex.1014, 565.

166. Thus, because PayPal Express Checkout discloses that PayPal’s servers send the merchant server a token which is combined with a predefined static URL that is used to establish a secure connection between the buyer and PayPal’s servers, PayPal Express Checkout renders obvious “*generating a uniform resource locator (URL), for establishing a secure socket layer connection via the internet between the secure server and the payer computing system, the URL*” as claimed.

[1.1.5.1a] *the URL comprising either: a dynamic URL generated by the secure server for the payer and the payee;*

167. The claim language recites two alternative types of URLs—static or dynamic. As I explain below in limitation [1.1.5.1b], PayPal Express Checkout teaches a static URL as claimed and, thus, meets the claim. Nevertheless, a POSITA would have understood that, instead of PayPal’s servers generating a token identifying the transaction and prompting the merchant server to combine the token with a static URL, PayPal’s servers could generate a dynamic URL. The

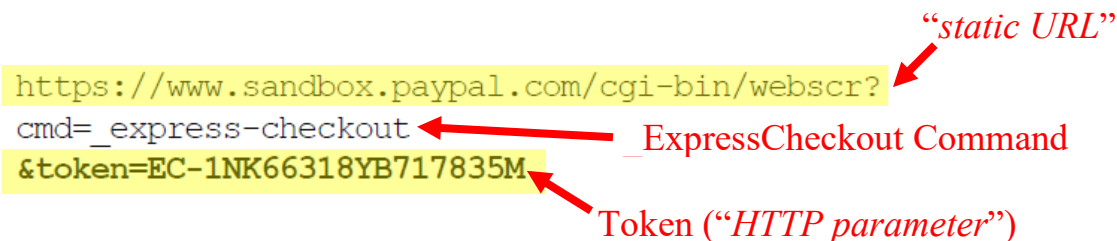
merchant may then redirect the buyer's browser using the dynamic URL.

168. It was well-known in the art to use dynamic URLs in place of static URLs. *See* Ex.1019, 1 (explaining the difference between static and dynamic URLs). For example, Hammad describes a processor that sends a network address to another processor, where “[t]he network address may be **static or dynamic**, the latter of which may be obtained through API function calls to the computer’s network services module.” Ex.1016, [0058]. Similarly, Hodson explains that “the user may be redirected to a **predetermined or dynamic URL** on the first website server 31.” Ex.1018, [0046]. Moreover, Fellig describes a system server generates an HTML link including “**a dynamic URL destination generated by the system server** for each Merchant profile generated.” Ex.1017, [0036]. Thus, a POSITA would have understood how to modify PayPal Express Checkout so that PayPal’s server generates a dynamic URL rather than generating the token that is combined with a static URL.

169. Thus, because a POSITA would have understood that the static URL and token (“*HTTP parameter*”) may be replaced with a dynamic URL, PayPal Express Checkout renders obvious “*the URL comprising either: a dynamic URL generated by the secure server for the payer and the payee*” as claimed.

[1.1.5.1b] *the URL comprising either:...or a static URL and a hypertext transport protocol (HTTP) parameter used by the secure server to identify the payer and the payee;*

170. As explained above in limitation [1.1.5.1], the merchant redirects the buyer's browser using a URL ("*static URL*"), the `_ExpressCheckout` command, and the token ("*HTTP parameter*") returned by `SetExpressCheckout`. See Ex.1006, 17 ("When [the merchant] redirect[s] [the] buyer's browser to PayPal, [the merchant] must specify the `_ExpressCheckout` command for Express Checkout. [The merchant] also specif[ies] the token that identifies the transaction, which was returned by the `SetExpressCheckout` API operation."); see also 24-25 (Step 3). In the Example Transaction, the merchant "use[s] the token and redirect[s] [the merchant's] browser to the Sandbox to log in, as follows:"



The image shows a URL from Ex.1006, 25 with three red arrows pointing to specific parts of it. The first arrow points to the domain and path `https://www.sandbox.paypal.com/cgi-bin/webscr?` and is labeled "*static URL*". The second arrow points to the query parameter `cmd=_express-checkout` and is labeled "`_ExpressCheckout` Command". The third arrow points to the query parameter `&token=EC-1NK66318YB717835M` and is labeled "Token ("*HTTP parameter*")".

Ex.1006, 25 (annotated).

171. As explained above in limitation [1.1.3], the token "is used by other Express Checkout API operations and by the `_ExpressCheckout` command to identify the transaction," and, thus, identifies the buyer and the merchant ("*identiff[ies] the payer and payee*"). Ex.1006, 18; see also 16 ("A token is a value assigned by PayPal that associates the execution of API operations and commands with a specific instance of a user experience flow."). Because in Step 3, the token

is used as part of the HTTP request along with the static URL, the token is used as an “*HTTP parameter*” as claimed.

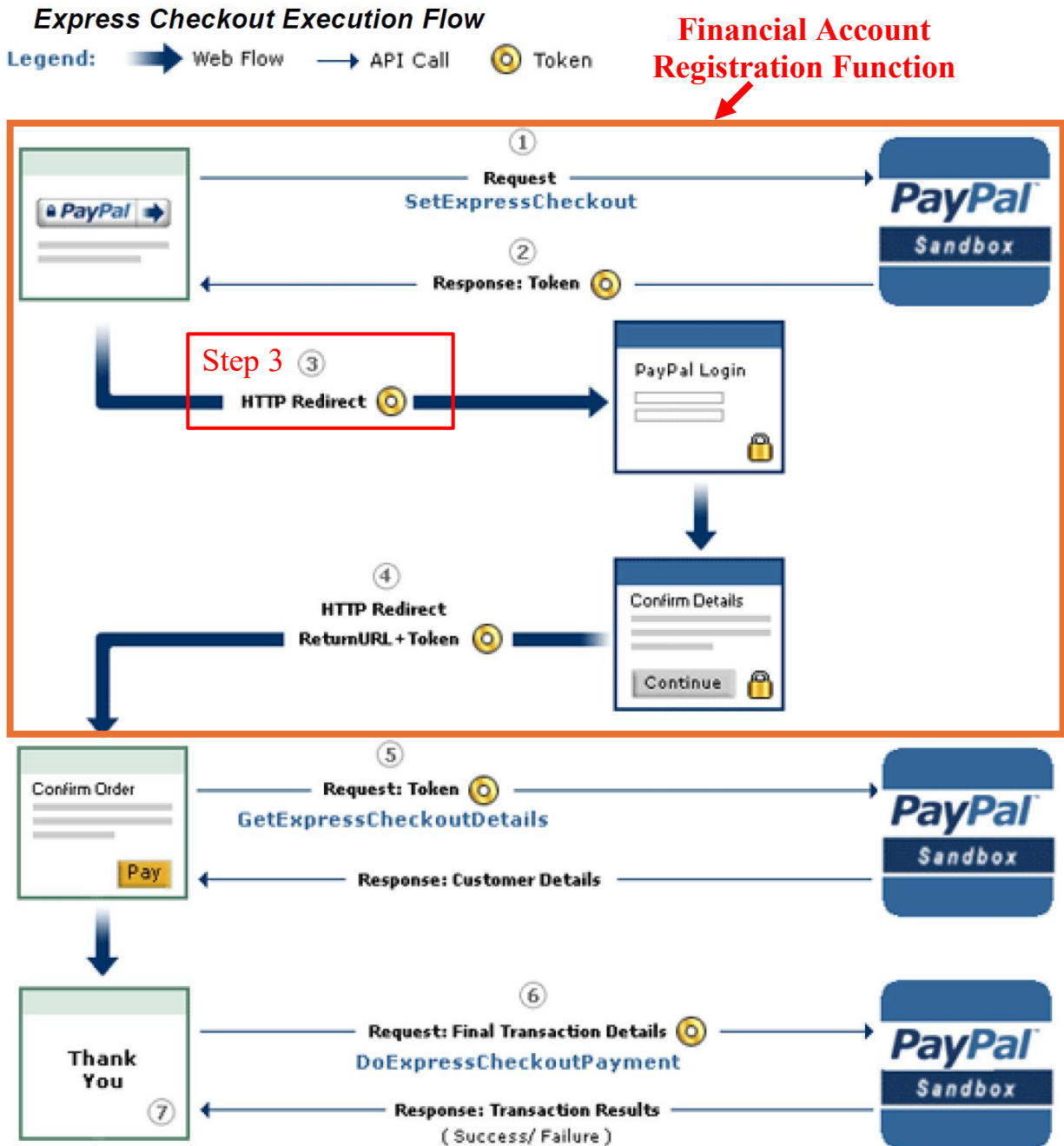
172. Moreover, as explained above in Section VI.C, a POSITA would have understood “*generating a... URL*” in limitation [1.1.5.1] to include sending the merchant a token to be combined with a predefined static URL.

173. Thus, because PayPal Express Checkout discloses combining a static URL with the token identifying the transaction to redirect the buyer’s browser to PayPal’s website, PayPal Express Checkout renders obvious “*the URL comprising either:...or a static URL and a hypertext transport protocol (HTTP) parameter used by the secure server to identify the payer and the payee*” as claimed.

[1.1.5.2] *establishing the secure socket layer connection, in response to an HTTP request received from the merchant server for the generated URL, between the secure server and the payer computing system within a window or frame that is displayed within the webpage provided by the merchant server;*

174. First, PayPal Express Checkout discloses “*establishing the secure socket layer connection...between the secure server and the payer computing system*”).” As explained above in limitations [1.1.5.1]-[1.1.5.1b], the merchant server redirects the buyer’s browser to PayPal’s website using a URL including an HTTPS request (e.g. <https://www.sandbox.paypal.com/cgi-bin/webscr?>), the `_ExpressCheckout` command, and the token, thereby establishing an SSL connection between the buyer’s browser (which is accessed using the buyer’s personal computer or other device (“*payer computing system*”)) and PayPal’s

servers (“secure server”). This is illustrated in Step 3 of the Example Transaction:



Ex.1006, 24 (annotated)

175. **Second**, as explained in limitation [1.5.1.1], the SSL connection is established “in response to an HTTP request received from the merchant server for

the generated URL.” In other words, the HTTP request as defined by the static URL and token triggers the establishment of the SSL connection. An example redirect is illustrated below:

`https://www.sandbox.paypal.com/cgi-bin/webscr?` “static URL”
`cmd=_express-checkout` _ExpressCheckout Command
`&token=EC-1NK66318YB717835M` Token (“HTTP parameter”)

Ex.1006, 25 (annotated)

176. Third, PayPal Express Checkout discloses that the SSL connection is provided “*within a window or frame that is displayed within the webpage provided by the merchant server.*” For example, when a user purchases digital goods, PayPal’s website is displayed in a mini-browser or lightbox⁵ that overlay’s the merchant’s website:

To the buyer, the digital goods payment process appears to be built into [the merchant’s] website. The digital goods payment process occurs in a lightbox that temporarily overlays [the merchant’s] website.

[The] buyer’s digital goods payment experience depends upon two factors:

⁵ A lightbox is a window that overlays a webpage and blocks some of the content on the webpage.

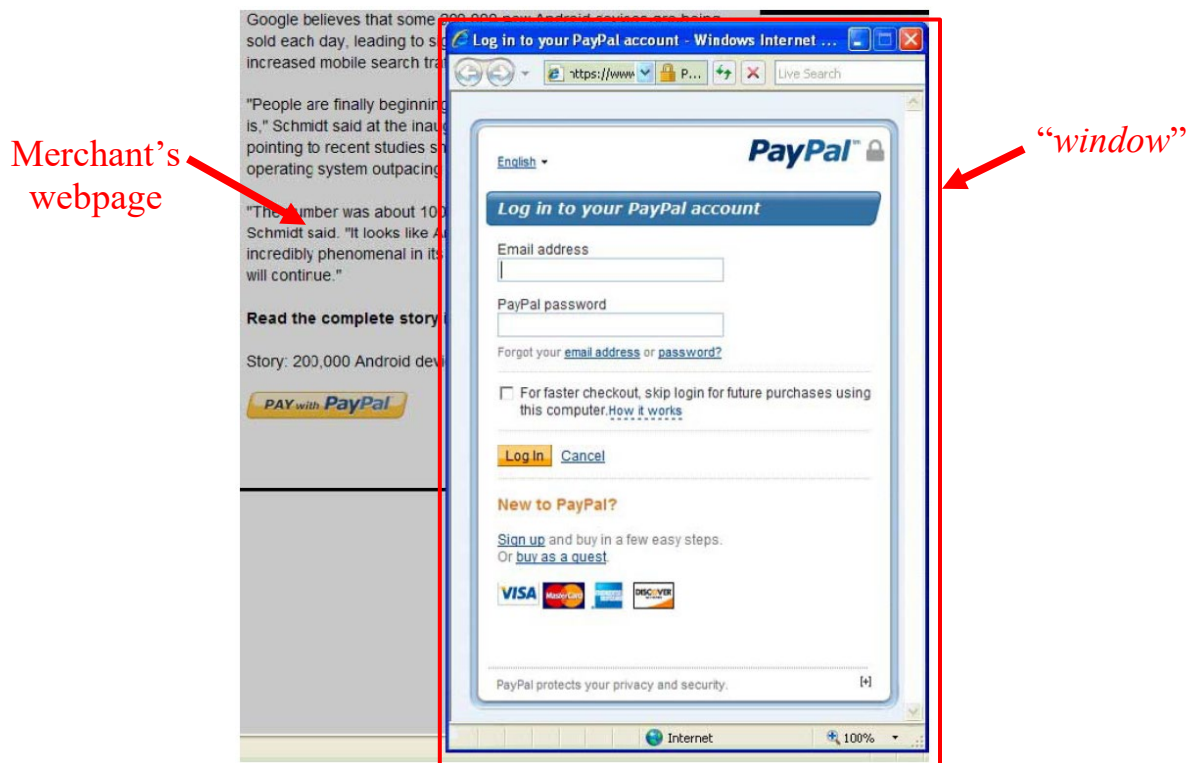
1. The value [the merchant] set[s] for the experience type variable on [its] web page

[The merchant] determine[s] how the digital goods flow appears to [its] buyers by setting (or not setting) the experience type variable (expType) on [its] web page...

2. Whether the buyer is remembered from a previous transaction

[The] buyers partially determine the digital goods flow by choosing whether to check the “remember me” box on the login page. (Checking the box creates a “remember me cookie” on the buyer’s machine that lets them skip the login page on subsequent visits.)

Ex.1012, 10-11. For example, when “[t]he expType variable is set to ‘instant’ and the buyer is new: The DG flow **opens the Log in to your PayPal account page in a mini-browser**, with a mask over the underlying page” (Ex.1012, 11), as shown in the screen shot below:

**Ex.1012, 12 (annotated)**

177. Moreover, a POSITA would have found the alternatively claimed “frame” to be obvious as well. It was well-known in the art to use an inline frame (“iframe”) to establish a connection between a buyer’s browser and a secure server. For example, Kloster explains that an inline frame can be directly hosted on the merchant’s website and can communicate directly with a secure payment system server. Ex.1013, [0016] (“The payment system server 102 provides the business entity 104 with one or more inline frames 122 that can be directly hosted on the website 120. **Each inline frame 122 is configured to communicate directly with the payment system server 102.** For example, the customer may enter

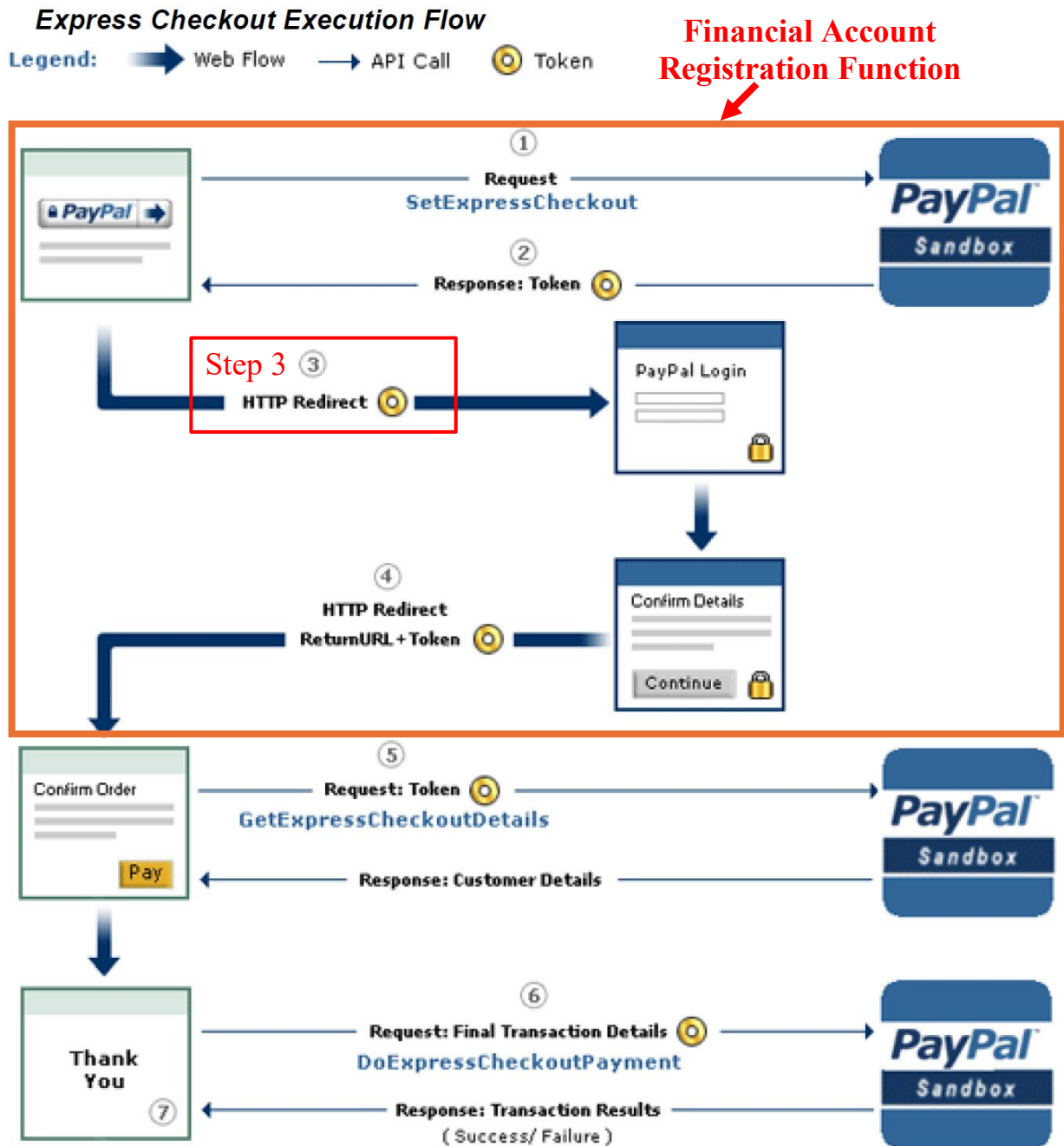
confidential information regarding an account and then submit the information.”). Similarly, Collison describes the merchant serving a payment form to a customer’s browser using an iframe. Ex.1015, 8:54-57 (“Stripe.js establishes a secure API tunnel to Stripe API Server (bapi) (304) of Stripe (300). When the Merchant's Payment Form (110) with Stripe.js first loads, it creates an iframe with URL <https://api.stripe.com/js/v1/apitunnel.html>.”); 9:24-26 (“Once created, messages are sent/received to this iframe using `xd.postMessage`. `xd.postMessage` provides a browser-agnostic method for communicating with an iframe.”). Therefore, a POSITA would have understood that Express Checkout could be modified such that the buyer’s browser can be redirected to PayPal’s website via an inline frame (“*frame*”) displayed within the merchant’s website.

178. Thus, because PayPal Express Checkout describes redirecting the buyer’s browser to the PayPal website using an HTTPS request within a mini-browser and a POSITA would have understood that an iframe could have been used instead of a mini-browser, PayPal Express Checkout renders obvious “*establishing the secure socket layer connection, in response to an HTTP request received from the merchant server for the generated URL, between the secure server and the payer computing system within a window or frame that is displayed within the webpage provided by the merchant server*” as claimed.

[1.1.5.3] *outputting instructions to the payer computing system, in response to the HTTP request for the generated URL, to render a financial account*

registration request form, within the window or frame that is displayed within the webpage provided by the merchant server, that provides functionality for the payer to provide sensitive financial account information associated with a financial account; and

179. First, PayPal Express Checkout discloses “*outputting instructions to the payer computing system, in response to the HTTP request for the generated URL.*” As explained above in [1.1.5.1], the merchant redirects the buyer’s browser to PayPal’s website using a URL including an HTTPS request, the `_ExpressCheckout` command, and the token, thereby establishing an SSL connection between the buyer’s browser and PayPal’s servers. This is illustrated in Step 3 of the Example Transaction:



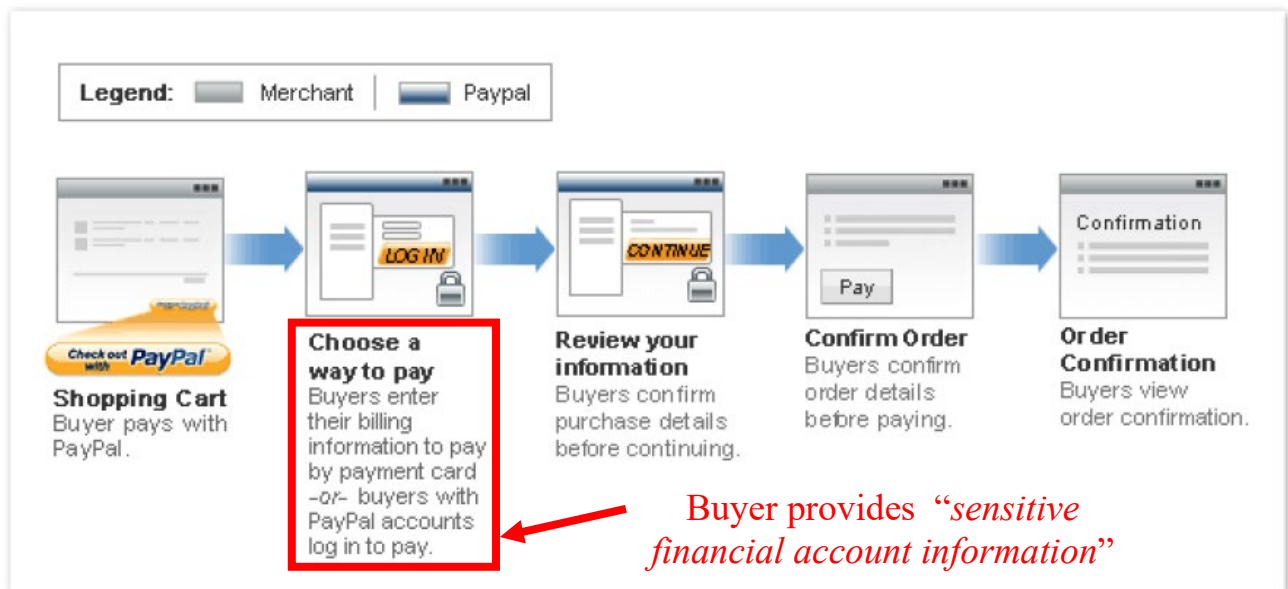
Ex.1006, 24 (annotated)

180. Specifically, the buyer's browser is directed to PayPal's log in page. See Ex.1006, 17 ("PayPal provides a command that [the merchant] use[s] when redirecting [its] buyer's browser to PayPal. This command enables [the] buyer to

log into PayPal to approve an Express Checkout payment.”), 25 (“If the [SetExpressCheckout API operation] was successful, use the token and redirect [the] browser to the Sandbox to log in.”).

181. When using Express Checkout, the buyer “[l]og[s] in to PayPal or choose[s] other ways to pay.” Ex.1009, 12. As shown in the below diagram, buyers can choose a way to pay where they either “enter their billing information to pay by payment card” or log in to their PayPal accounts:

The Express Checkout Experience



Ex.1009, 12 (annotated)

182. Thus, in response to the HTTP request of the redirect URL (“in response to the HTTP request for the generated URL”), PayPal’s servers “output[] instructions” to render the PayPal log in page on the buyer’s browser (displayed on

“the payer computing system”).

183. Second, PayPal Express Checkout discloses that the instructions *“render a financial account registration request form...that provides functionality for the payer to provide sensitive financial account information associated with a financial account.”* PayPal’s log in page provides a data entry form that allows buyers to input information, including their financial/billing information (*“sensitive financial account information”*):

The data entry area lets buyers specify information about their payments. Different forms slide into view as buyers progress through steps of the checkout process. Some forms are divided into sections, which buyers expand and collapse to complete the form.

Ex.1009, 10-11; *see also* Ex.1010, 20. An example data entry form is shown below:

Example Data Entry Form in the PayPal Checkout User Interface

The screenshot displays the PayPal checkout user interface. It features a form titled "Choose a way to pay" in orange text. Below the title, there are two main sections. The first section, "Pay with my PayPal account", is expanded and contains fields for "Email" and "Password", a "LOG IN" button, and a link for "Forgot email address or password?". The second section, "Don't have a PayPal account?", is collapsed and contains the text "Pay with your debit or credit card as a PayPal guest" and a link to "Cancel and return to <Merchant name>". Red arrows point to the form title, the expanded section, and the collapsed section.

Ex.1009, 10

184. These data entry forms include: “Choose a way to pay – Buyers enter their PayPal credentials to log in and pay. Other buyers **enter their billing information** and other contact information to pay with debit or credit cards.”

Ex.1009, 11. The data entry forms include fields for the buyer to provide financial/billing information, such as credit card information. *See* Ex.1009, 8; *see also* Ex.1010, 22-23. Below is an example data entry form:

Your order summary

Description	Amount
Item description 1	\$79.00
Item # 123	Item Price: \$79.00
Qty: 1	
Item description 2	\$89.00
Item # 123	Item Price: \$89.00
Qty: 1	
Item total	\$168.00
Tax	\$15.54
Shipping and handling	\$18.50
Total	\$202.04 USD

Choose a way to pay

Pay with my PayPal account

Don't have a PayPal account?
Pay with your debit or credit card as a guest

Country: United States

Credit or debit card number

Expiry Date: mm / yy

CVC

What's this?

First name

Last name

Billing Address Line 1

Billing Address Line 2 (Optional)

Zip code

City

State: Select

e.g. 955-555-5555

Phone

Email Address

This transaction will appear on your statement as merchant name

CONTINUE

Secure checkout by PayPal

Cancel and return to Merchant name

PayPal protects your privacy and security. For more information, read our [Use Agreement](#) and [Privacy Policy](#).

Cart review area

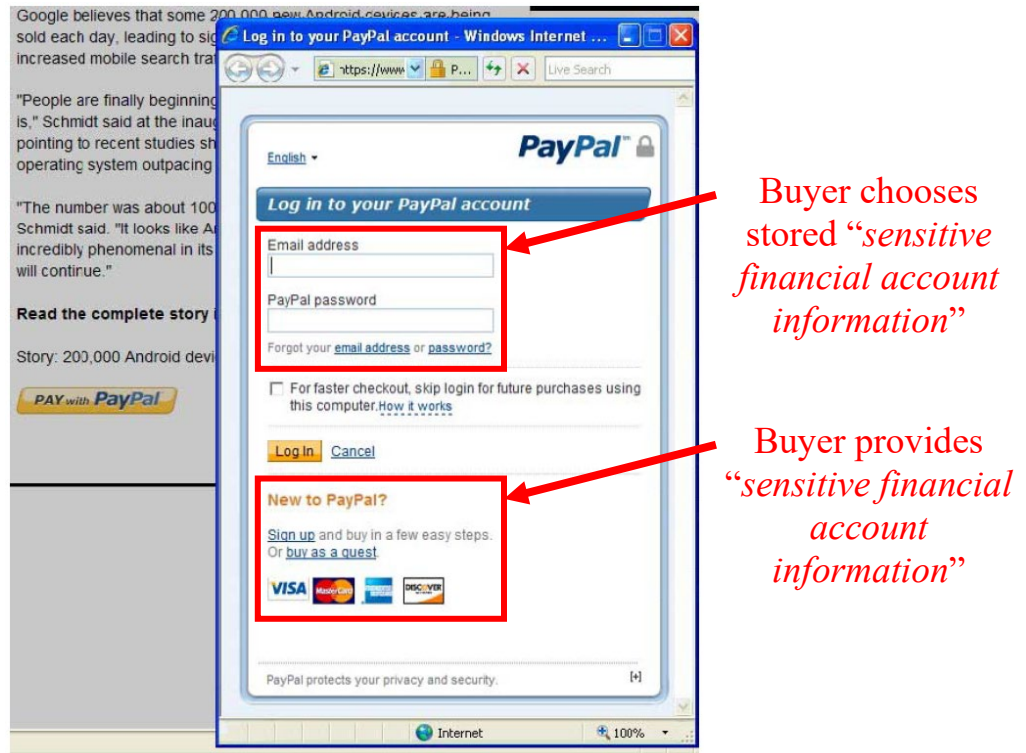
Data entry area

Buyer provides “sensitive financial account information”

Ex.1009, 8 (annotated)

185. Thus, the data entry forms (*“financial account registration request form”*) allow the buyer to input her financial/billing information (*“provides functionality for the payer to provide sensitive financial account information associated with a financial account”*).

186. Third, PayPal Express Checkout discloses that the data entry form may be displayed *“within the window or frame that is displayed within the webpage provided by the merchant server.”* As explained above in limitation [1.1.5.2], the buyer’s browser is redirected to PayPal’s website within a mini-browser overlaid on the merchant’s website. For example, when purchasing digital goods, “[t]he DG flow opens the Log in to your PayPal account page **in a mini-browser**, with a mask over the underlying page. Ex.1012, 12 (emphasis removed and added). The screenshot below illustrates the mini-browser overlaid on the merchant’s webpage:



Ex.1012, 12 (annotated).

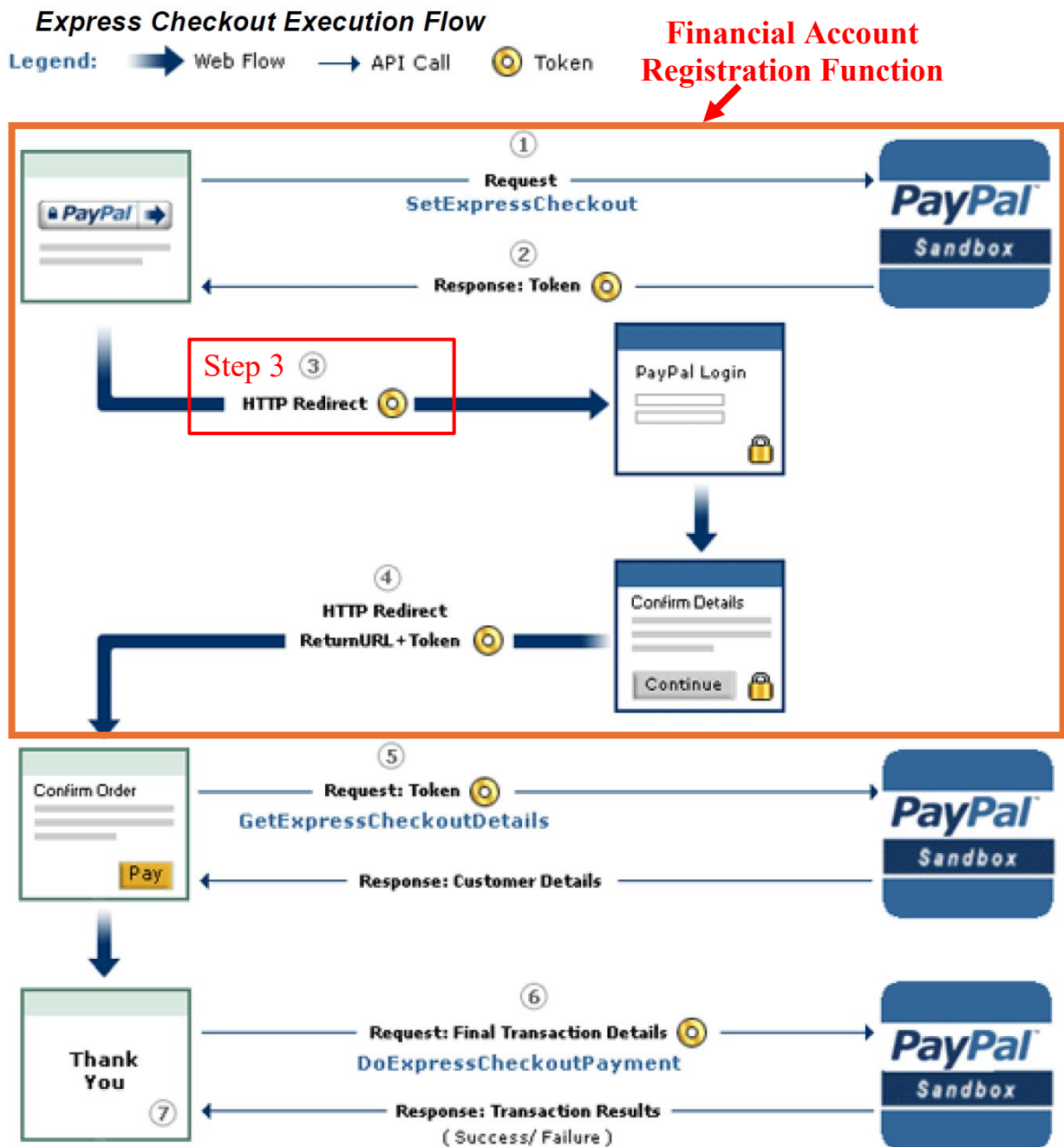
187. From the log in page, “the buyer can log in” to her PayPal account or “[b]uyers without a PayPal account can sign up for one, or proceed as a guest and complete their purchase with a credit card.” Ex.1012, 12. In other words, the buyer can pay using previously stored sensitive financial account information from her PayPal account or can provide new sensitive financial account information (e.g. credit card information). Thus, the data entry forms (“*financial account registration request form*”) can be rendered within a mini-browser (“*within the window or frame that is displayed within the webpage provided by the merchant server*”).

188. Thus, because PayPal Express Checkout discloses redirecting the

buyer's browser to PayPal's log in page using an HTTP request and PayPal's log in page provides data entry forms to the buyer's browser that allow the buyer to input her financial/billing information via a mini-browser, PayPal Express Checkout renders obvious "*outputting instructions to the payer computing system, in response to the HTTP request for the generated URL, to render a financial account registration request form, within the window or frame that is displayed within the webpage provided by the merchant server, that provides functionality for the payer to provide sensitive financial account information associated with a financial account*" as claimed.

[1.1.5.4] *outputting instructions to the payer computing system, in response to the HTTP request for the generated URL, to encrypt the sensitive financial account information provided by the payer and transmit the encrypted financial account information to the secure server via the secure socket layer connection;*

189. First, as explained above in limitation [1.1.5.3], PayPal Express Checkout discloses "*outputting instructions to the payer computing system, in response to the HTTP request for the generated URL.*" This is illustrated in Step 3 of the Example Transaction:



Ex.1006, 24 (annotated)

190. Moreover, as discussed in [1.1.5.3] a POSITA would have understood that the HTTPS request outputs instructions to the buyer's browser (displayed on the "payer computing system") that establishes an SSL connection between the

buyer's browser and PayPal's servers.

191. Second, the instructions “*encrypt the sensitive financial account information provided by the payer.*” A POSITA would have understood that communications sent via the SSL connection are encrypted by the sending browser, then decrypted at the receiving server. Ex.1014, 565 (“To safeguard sensitive information (such as usernames and passwords), ask the ISP hosting your site to set up the Secure Socket Layer (SSL) protocol. The ISP must first obtain a security certificate for your Web site from a certificate authority.... After the ISP installs the security certificate for your site on the Web server, **the Web server and Web browser will encrypt the HTTP messages before sending them across the Internet.** Having the Web browser and server encrypt their transmissions protects your form results (and other message content) from unauthorized access, because a hacker viewing an encrypted HTTP message body sees only an unintelligible jumble of characters, numbers, and special symbols.”) (emphasis removed and added). Moreover, a POSITA would have understood that the communications sent via the SSL connection would be encrypted by PayPal's servers as well.

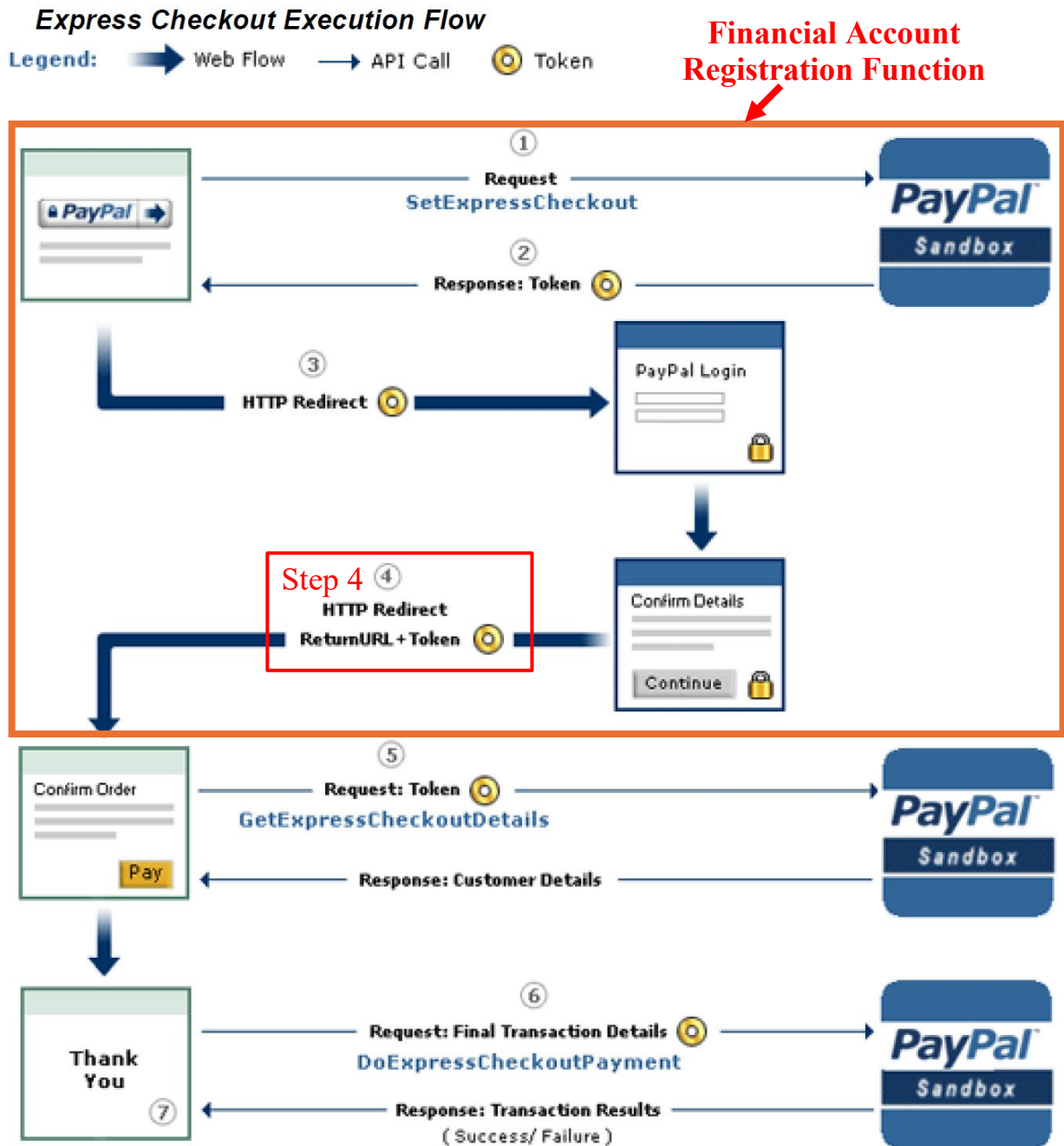
192. Third, the instructions “*transmit the encrypted financial account information to the secure server via the secure socket layer connection.*” After the buyer inputs her billing/financial information (“*sensitive financial account*

information”) into the data entry forms (“*financial account registration form*”), the buyer submits the information. *See* Ex.1006, 10 (“In the Express Checkout flow, the buyer” “Reviews the transaction on PayPal” and “Confirms the order and pays from [the merchant’s] site.”); Ex.1009, 12; Ex.1010, 23 (“When the customer clicks the Pay Now button, they return to [the merchant’s] website.”) (emphasis removed). Thus, the billing/financial information is sent to PayPal’s servers (“*the secure server*”) via the SSL connection (see limitation [1.2] below). As explained above, a POSITA would have understood that the message containing the billing/financial information (“*financial account information*”) would have been encrypted before sending over the SSL connection.

193. Thus, because PayPal Express Checkout discloses redirecting a user’s browser using an HTTPS request and allowing the buyer to submit her financial/billing information to PayPal’s servers and it was well-known in the art that this establishes an SSL connection that encrypts/decrypts communications between the connected browser and servers, PayPal Express Checkout renders obvious “*outputting instructions to the payer computing system, in response to the HTTP request for the generated URL, to encrypt the sensitive financial account information provided by the payer and transmit the encrypted financial account information to the secure server via the secure socket layer connection*” as claimed.

[1.2] *receiving the sensitive financial account information provided by the payer via the secure socket layer connection;*

194. As explained above in limitation [1.1.5.3], PayPal Express Checkout discloses that PayPal's servers provide data entry forms to the buyer's browser that allow the buyer to provide her financial/billing information (e.g. credit card information). This is illustrated in Step 4 of the Example Transaction:



Ex.1006, 24 (annotated)

195. PayPal Express Checkout explains that the buyer may input her financial/billing information and submit the form to PayPal's servers:

In the Express Checkout experience, buyers:

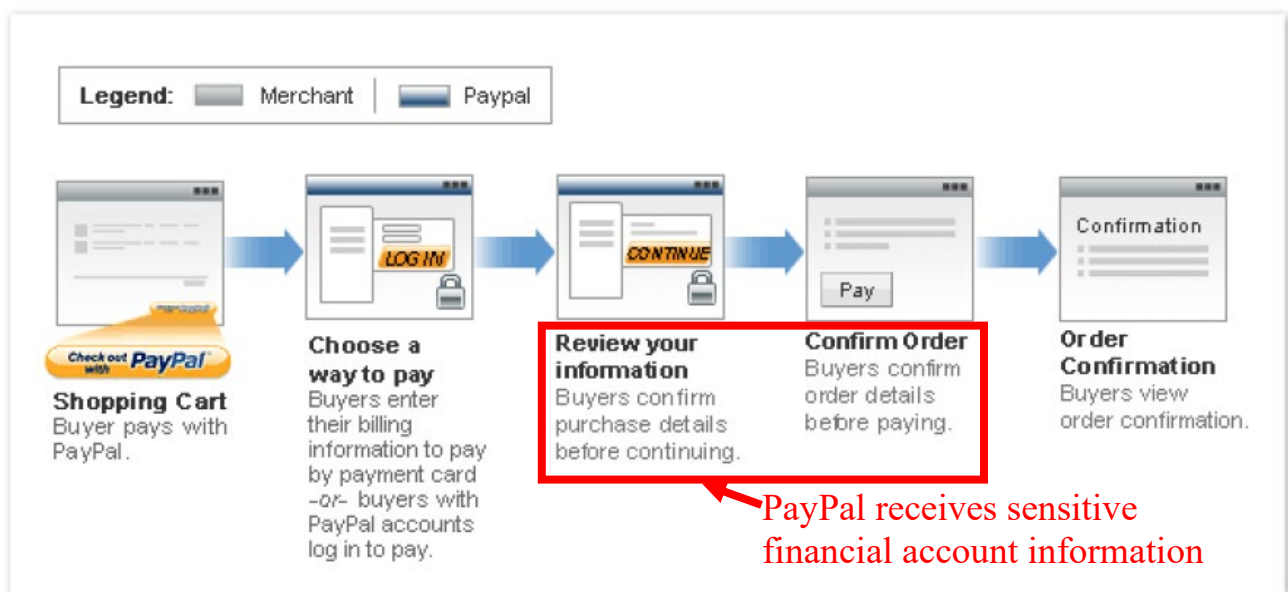
1. Choose Express Checkout on [the merchant's] shopping cart page
2. Log in to PayPal or choose other ways to pay
3. Review their payments on PayPal
4. Confirm their orders on [the merchant's] website before paying
5. View [the merchant's] order confirmation

Ex.1009, 12; *see* Ex.1006, 10 (“In the Express Checkout flow, the buyer:...3.

Reviews the transaction on PayPal... 4. Confirms the order and pays from [the merchant's] site.”); Ex.1010, 12 (“Buyers without a PayPal account can sign up for one, or proceed as a guest and complete their purchase with a credit card.”);

Ex.1012, 12-13 (“After successfully logging in, the digital flow presents the You are about to pay page in a mini-browser... If the buyer chooses Pay, the Thank you for using PayPal page appears in the mini-browser”) (emphasis removed). This process is illustrated in the below diagram:

The Express Checkout Experience



Ex.1009, 12 (annotated)

196. Thus, “[b]uyers enter and review information about their payments” via the data entry forms provided by PayPal, then submit the information to PayPal’s servers. Ex.1009, 11; Ex.1010, 22-23.

197. Thus, because PayPal Express Checkout discloses that the buyer enters and submits her payment information to PayPal’s server via data entry forms provided on PayPal’s website, PayPal Express Checkout renders obvious “*receiving the sensitive financial account information provided by the payer via the secure socket layer connection*” as claimed.

[1.3] *storing the sensitive financial account information in a secure storage location and performing each software process required to maintain compliance with one or more information security standards;*

198. **First**, PayPal Express Checkout discloses “*storing the sensitive financial account information.*” PayPal Express Checkout explains that a buyer’s shipping, billing, or payment information may be stored on their PayPal account. Ex.1006, 9 (“However, the buyer does not enter shipping, billing, or payment information, because PayPal provides the stored information. This simplifies and expedites the checkout process.”). When purchasing digital goods using Express Checkout, “[b]uyers without an account can sign up for one.” Ex.1009, 12.

199. To sign up for an account, the buyer provides shipping, billing, and/or payment information that are stored by PayPal in association with the buyer’s

account:

If [the buyer] open[s] an account or use[s] the PayPal Services, [PayPal] may collect the following types of information:

- Contact information – [the buyer’s] name, address, phone, email, Skype ID and other similar information.
- Financial information - the full bank account numbers and/or credit card numbers that [the buyer] link[s] to [her] PayPal account or give[s] [PayPal] when [she] use[s] the PayPal Services.

Ex.1011, 1-2. Thus, PayPal’s servers store financial/billing information received from a buyer.

200. Second, PayPal Express Checkout discloses that the financial/billing information is stored in “*a secure storage location.*” PayPal stores and protects information submitted by the buyer. For example, “[PayPal] use[s] computer safeguards such as firewalls and data encryption, [PayPal] enforce[s] physical access controls to [its] buildings and files, and [PayPal] authorize[s] access to personal information only for those employees who require it to fulfill their job responsibilities.” Ex.1011, 2.

201. Moreover, as explained above in limitation [1.1.4], PayPal only allows authenticated users to access PayPal’s servers and includes four levels of security to prevent unauthorized access. Ex.1007, 13 (“The PayPal SOAP API service is protected to ensure that only authorized PayPal members use it. There

are four levels of security: 1. A required API username (Username field) and API password (Password field) 2. A third required authentication mechanism, which is either one of the following: Client-side request signing via a PayPal-issued API Certificate [or] Request authentication via an API Signature included in the request (Signature field) 3. An optional third-party authorization to make the API call on some other account's behalf (the optional Subject field). 4. Secure Sockets Layer (SSL) data transport[.] A failure of authenticated security at any one of these levels denies access to the PayPal SOAP API service.”).

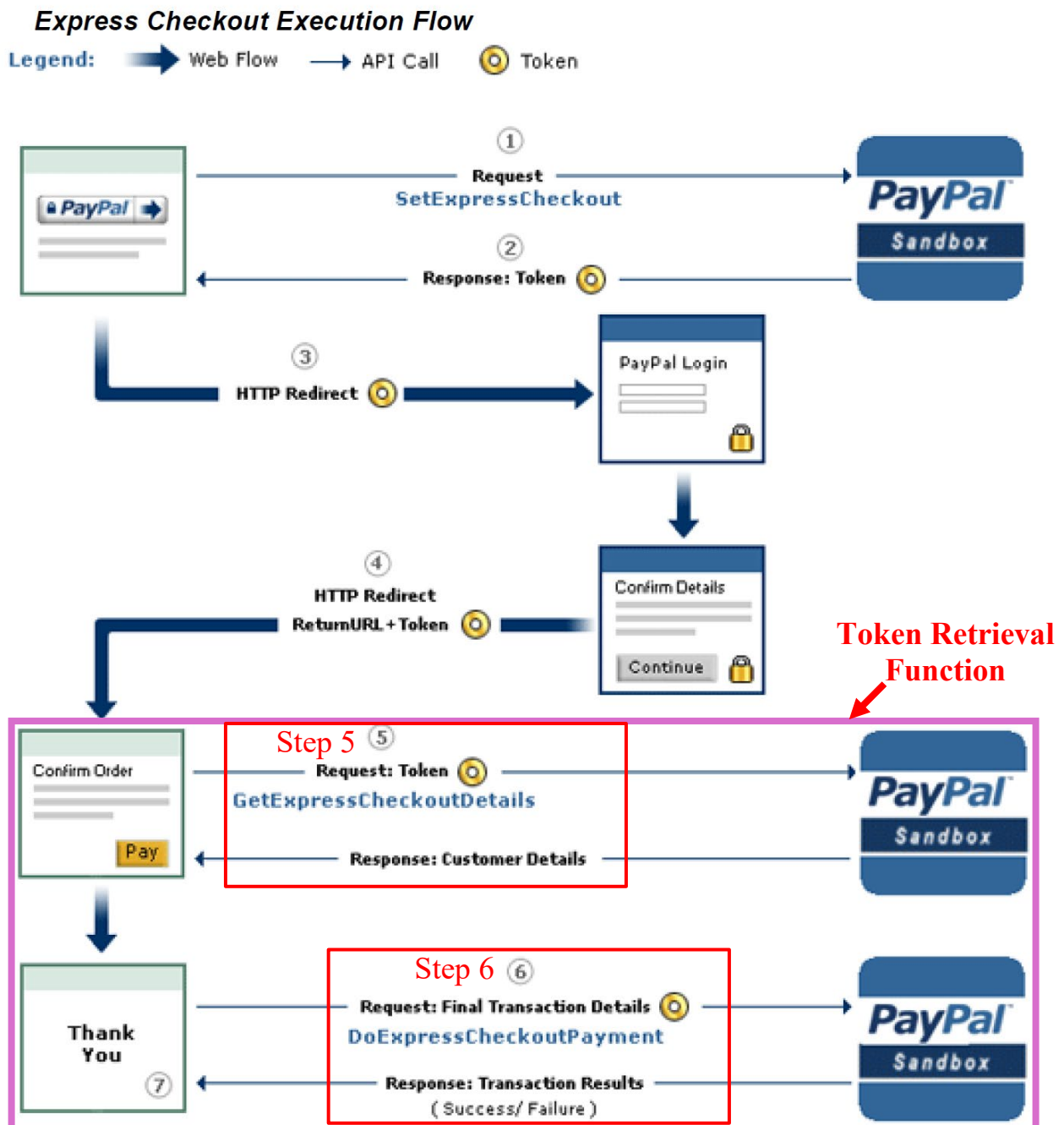
202. Third, PayPal explains that it “*perform[s] each software process required to maintain compliance with one or more information security standards.*” For example, “[PayPal] store[s] and process[es] [the buyer’s] personal information on [PayPal’s] computers in the US and elsewhere in the world where PayPal facilities are located, and [PayPal] protect[s] it by maintaining physical, electronic and procedural safeguards **in compliance with applicable US federal and state regulations.**” Ex.1011, 2. A POSITA would have considered the applicable US federal and state regulations to satisfy “*one or more information security standards.*”

203. Thus, because PayPal Express Checkout discloses signing up for a PayPal account with shipping, billing and/or payment information and storing and protecting this information according to well-known security protocols, PayPal

Express Checkout renders obvious “*storing the sensitive financial account information in a secure storage location and performing each software process required to maintain compliance with one or more information security standards*” as claimed.

[1.4] *executing a token retrieval function, upon initiation by the merchant server via the API, by:*

204. As explained above in limitations [1.1.1]-[1.1.2], PayPal Express Checkout discloses that PayPal’s servers provide GetExpressCheckoutDetails and DoExpressCheckoutPayment (“*token retrieval function*”) to the merchant server. These are Steps 5 and 6 illustrated in the Example Transaction below:



Ex.1006, 24 (annotated)

205. The merchant invokes `GetExpressCheckoutDetails` to retrieve the Payer ID (“token”) from PayPal’s servers. Ex.1006, 21 (“To obtain details about an

Express Checkout transaction, [the merchant] can invoke the GetExpressCheckoutDetails API operation.”), 26. The merchant then invokes DoExpressCheckoutPayment specifying the Payer ID returned by GetExpressCheckoutDetails to complete the transaction. *See* Ex.1006, 21-23 (“To complete an Express Checkout transaction, [the merchant] must invoke the DoExpressCheckoutPayment API operation.”), 26-27.

206. Thus, because PayPal Express Checkout discloses that the merchant initiates the retrieval and use of the PayerID to complete the transaction by invoking GetExpressCheckoutDetails and DoExpressCheckoutPayment, PayPal Express Checkout renders obvious “*executing a token retrieval function, upon initiation by the merchant server via the APP*” as claimed.

[1.4.1] *providing a non-sensitive electronic data token representing the sensitive financial account information to the merchant server without providing the sensitive financial account information to the merchant server and without providing the non-sensitive electronic data token to the payer; and*

207. First, PayPal Express Checkout discloses “*providing a non-sensitive electronic data token representing the sensitive financial account information to the merchant server.*” As explained above in limitations [1.1.1]-[1.1.2] and [1.4], the merchant invokes GetExpressCheckoutDetails to retrieve the Payer ID (“*non-sensitive electronic data token*”). Ex.1006, 21 (“To obtain details about an Express Checkout transaction, you can invoke the GetExpressCheckoutDetails API operation.”). The merchant “obtain[s] a buyer’s shipping address and Payer ID”

with GetExpressCheckoutDetails by:

1. Specify[ing] the token returned by PayPal when it redirects the buyer's browser to [the merchant's] site.

PayPal returns the token to use in the token HTTP request parameter when redirecting to the URL [the merchant] specified in [its] call to the SetExpressCheckout API.

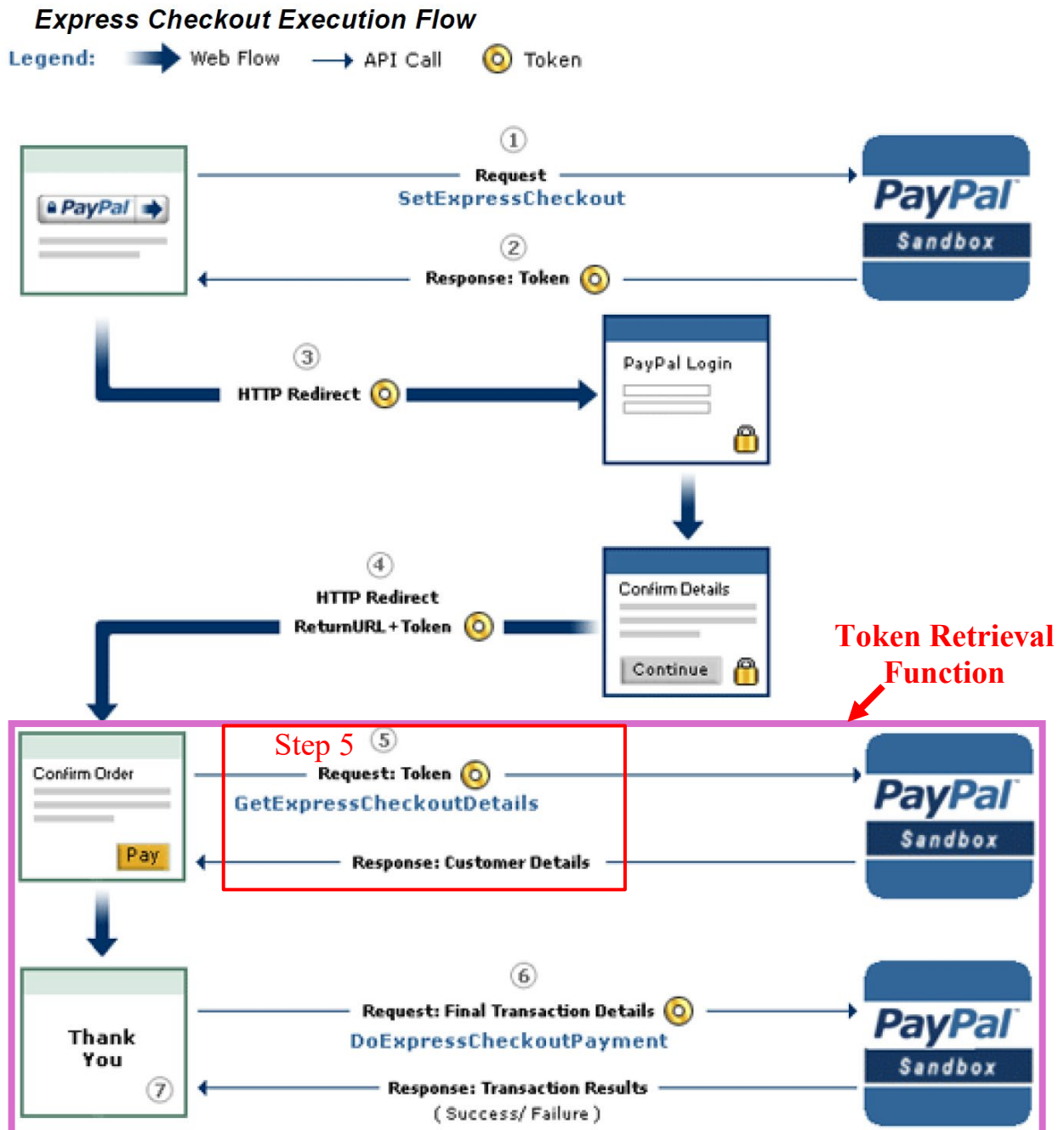
TOKEN=tokenValue

2. Execut[ing] the GetExpressCheckoutDetails API to obtain information about the buyer.

3. Access[ing] the fields in the GetExpressCheckoutDetails API response.

Ex.1006, 21.

208. GetExpressCheckoutDetails is Step 5 of the Example Transaction:



Ex.1006, 24 (annotated)

209. An example `GetExpressCheckoutDetails` request is provided below:

```

<form method=post action=https://api-3t.sandbox.paypal.com/nvp
  <input type=hidden name=USER value=API_username>
  <input type=hidden name=PWD value=API_password>
  <input type=hidden name=SIGNATURE value=API_signature>
  <input type=hidden name=VERSION value=XX.0>
  <input name=TOKEN value=EC-1NK66318YB717835M>
  <input type=submit name=METHOD value=GetExpressCheckoutDetails>
</form>

```

Token

Ex.1006, 26 (annotated)

210. An example response from PayPal’s servers, including the Payer ID (“token”) is provided below:

```

TIMESTAMP=2007%2d04%2d05T23%3a44%3a11Z
&CORRELATIONID=6b174e9bac3b3
&ACK=Success
&VERSION=XX%2e000000
&BUILD=1%2e0006
&TOKEN=EC%2d1NK66318YB717835M
&EMAIL=YourSandboxBuyerAccountEmail
&PAYERID=7AKUSARZ7SAT8
&PAYERSTATUS=verified
&FIRSTNAME=...
&LASTNAME=...
&COUNTRYCODE=US
&BUSINESS=...
&PAYMENTREQUEST_0_SHIPTONAME=...
&PAYMENTREQUEST_0_SHIPTOSTREET=...
&PAYMENTREQUEST_0_SHIPTOCITY=...
&PAYMENTREQUEST_0_SHIPTOSTATE=CA
&PAYMENTREQUEST_0_SHIPTOCOUNTRYCODE=US
&PAYMENTREQUEST_0_SHIPTOCOUNTRYNAME=United%20States
&PAYMENTREQUEST_0_SHIPTOZIP=94666
&PAYMENTREQUEST_0_ADDRESSID=...
&PAYMENTREQUEST_0_ADDRESSSTATUS=Confirmed

```

Payer ID (“token”)

Ex.1006, 26 (annotated)

211. The Payer ID is a “[u]nique PayPal customer account identification

number,” such as 7AKUSARZ7SAT8, as shown in the example GetExpressCheckoutDetails response above. Ex.1007, 100. Thus, because the Payer ID identifies the customer’s account, including her financial/billing information, the Payer ID “*represent[s] the sensitive financial account information.*”

212. Second, PayPal Express Checkout discloses that the Payer ID is provided to the merchant server “*without providing the sensitive financial account information to the merchant server.*” In response to the GetExpressCheckoutDetails request from the merchant server, PayPal’s servers only send the merchant server the Payer ID (“*token*”), the buyer’s email, and shipping information. The Payer ID is a “[u]nique PayPal customer account identification number,” such as, for example, 7AKUSARZ7SAT8. Ex.1007, 100. The Payer ID is an alphanumeric string that does not include the buyer’s financial/billing information (“*sensitive financial account information*”). The buyer’s email and shipping information likewise do not include the financial/billing information. Thus, when PayPal’s servers return the Payer ID in response to GetExpressCheckoutDetails, the financial/billing information is not provided to the merchant server. *See* Ex.1011, 3 (“Regardless, [PayPal] will not disclose [the buyer’s] credit card number or bank account number to anyone [the buyer] ha[s] paid or who has paid [the buyer] through PayPal”).

213. Third, PayPal Express Checkout discloses that the Payer ID is also provided to the merchant server “*without providing the non-sensitive electronic data token to the payer.*” As explained above in Section VI.A., a POSITA would have understood that limitation [1.4.1] merely requires the secure server provide the “*non-sensitive data token to the merchant server...without providing the non-sensitive data token to the payer*” **during the token retrieval function.** A POSITA would have understood that this limitation applies only to the token retrieval function and does not apply to other steps in the transaction process. In PayPal Express Checkout, the merchant “invoke[s] a form **on [its] site** that calls the GetExpressCheckoutDetails API operation” (part of the “*token retrieval function*”) and “the GetExpressCheckoutDetails API returns information about the payer,” including the Payer ID. Ex.1006, 26. The Payer ID is sent to the merchant server **not** the buyer (“*payer*”) during this step.

214. A POSITA would have understood that this limitation does not apply to any other step in the transaction process. Thus, regardless of whether PayPal provides the PayerID (“*non-sensitive data token*”) to the payer at some point other than during the token retrieval function, PayPal meets this limitation because it does not provide the PayerID to the payer during execution of the token retrieval function. Again, as mentioned above at VI.A, even the specification of the ’621 patent (parts incorporated by reference) allows for the non-sensitive data token to

be provided to the payer outside of the token retrieval function. See Ex.1005, 21, 31-32.

215. Thus, because PayPal Express Checkout discloses that a merchant can retrieve the Payer ID using the GetExpressCheckoutDetails API operation without the payment information being sent to the merchant and without the Payer ID being sent to the buyer during execution of the token retrieval function, PayPal Express Checkout renders obvious “*providing a non-sensitive electronic data token representing the sensitive financial account information to the merchant server without providing the sensitive financial account information to the merchant server and without providing the non-sensitive electronic data token to the payer*” as claimed.

[1.4.2] *processing the payment transaction using the sensitive financial account information by generating and transmitting an electronic request requesting the payment amount from the financial account, obtaining the payment amount, and forwarding at least a portion of the payment amount to the payee.*

216. First, PayPal Express Checkout discloses “*processing the payment transaction using the sensitive financial account information.*” As explained above in limitations [1.1.1]-[1.1.2] and [1.4], the merchant invokes DoExpressCheckoutPayment with the Payer ID (“*non-sensitive electronic data token*”) and a “*payment amount*” to complete the transaction. Ex.1006, 21-23 (“To complete an Express Checkout transaction, [the merchant] must invoke the DoExpressCheckoutPayment API operation.”). A merchant “execute[s] an Express

Checkout transaction” using DoExpressCheckoutPayment by:

1. Specify[ing] the token returned by PayPal when it redirects the buyer’s browser to [the merchant’s] site.

PayPal returns the token to use in the token HTTP request parameter when redirecting to the URL [the merchant] specified in [its] call to the SetExpressCheckout API.

TOKEN=tokenValue

2. **Specify[ing] the Payer ID** returned by PayPal when it redirects the buyer’s browser to [the merchant’s] site.

PayPal returns the Payer ID to use in the token HTTP request parameter when redirecting to the URL [the merchant] specified in [its] call to the SetExpressCheckout API. **Optionally, [the merchant] can obtain the Payer ID by calling the GetExpressCheckoutDetails API.**

PAYERID=id

3. **Specify[ing] the amount of the order** including shipping, handling, and tax; include the currency if it is not in US dollars....

PAYMENTREQUEST_0_AMT=amount

PAYMENTREQUEST_0_CURRENCYCODE=currencyID

4. Specify the payment action.

Although the default payment action is a Sale, it is a best practice to explicitly specify the payment action as one of the following values:

PAYMENTREQUEST_0_PAYMENTACTION=Sale

PAYMENTREQUEST_0_PAYMENTACTION=Authorization

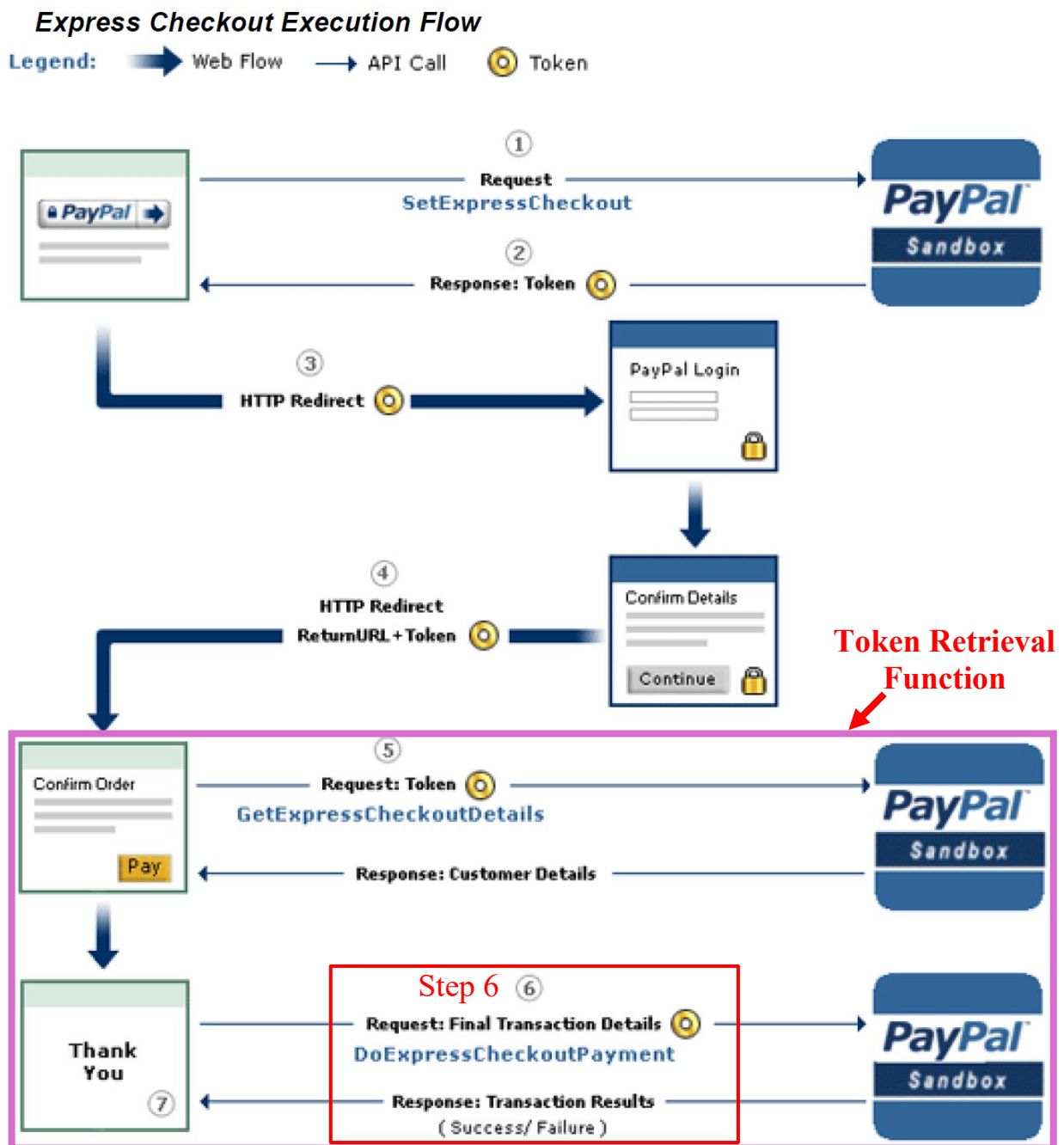
PAYMENTREQUEST_0_PAYMENTACTION=Order

5. Execut[ing] the DoExpressCheckoutPayment API to complete the Express Checkout transaction.

6. Examining the values returned by the API if the transaction completed successfully.

Ex.1006, 22-23.

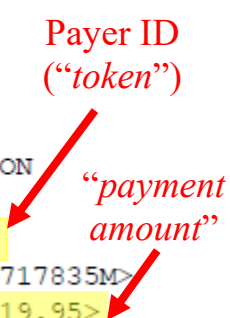
217. DoExpressCheckoutPayment is Step 6 in the Example Transaction:



Ex.1006, 24 (annotated)

218. An example DoExpressCheckout request is provided below:

```
<form method=post action=https://api-3t.sandbox.paypal.com/nvp>
  <input type=hidden name=USER value=API_username>
  <input type=hidden name=PWD value=API_password>
  <input type=hidden name=SIGNATURE value=API_signature>
  <input type=hidden name=VERSION value=XX.0>
  <input type=hidden name=PAYMENTREQUEST_0_PAYMENTACTION
    value=Authorization>
  <input type=hidden name=PAYERID value=7AKUSARZ7SAT8>
  <input type=hidden name=TOKEN value= EC%2d1NK66318YB717835M>
  <input type=hidden name=PAYMENTREQUEST_0_AMT value= 19.95>
  <input type=submit name=METHOD value=DoExpressCheckoutPayment>
</form>
```



Ex.1006, 17 (annotated).

219. As shown above, the merchant provides the Payer ID and the payment amount to indicate to PayPal’s servers that they should transfer the payment amount to the merchant using the financial/billing information (“*sensitive financial/billing information*”) associated with the Payer ID. An example DoExpressCheckoutPayment response is provided below:

```

TIMESTAMP=2007%2d04%2d05T23%3a30%3a16Z
&CORRELATIONID=333fb808bb23
ACK=Success
&VERSION=XX%2e000000
&BUILD=1%2e0006
&TOKEN=EC%2d1NK66318YB717835M
&PAYMENTREQUEST_0_TRANSACTIONID=043144440L487742J
&PAYMENTREQUEST_0_TRANSACTIONTYPE=expresscheckout
&PAYMENTREQUEST_0_PAYMENTTYPE=instant
&PAYMENTREQUEST_0_ORDERTIME=2007%2d04%2d05T23%3a30%3a14Z
&PAYMENTREQUEST_0_AMT=19%2e95
&PAYMENTREQUEST_0_CURRENCYCODE=USD
&PAYMENTREQUEST_0_TAXAMT=0%2e00
&PAYMENTREQUEST_0_PAYMENTSTATUS=Pending
&PAYMENTREQUEST_0_PENDINGREASON=authorization
&PAYMENTREQUEST_0_REASONCODE=None

```

Indicating transaction was successful

“payment amount”

Ex.1006, 27 (annotated).

220. “If the operation was successful, the response should include ACK set to Success.” Ex.1006, 27; *see also* Ex.1007, “The status of the payment:...Completed: The payment has been completed, and the funds have been added successfully to [the merchant’s] account balance.”); Ex.1024, 97 (same). Thus, the response from DoExpressCheckoutPayment indicates that the transaction was processed successfully.

221. Second, processing the payment transaction is done by “*generating and transmitting an electronic request requesting the payment amount from the financial account, obtaining the payment amount, and forwarding at least a portion of the payment amount to the payee.*” PayPal describes completing a transaction. Ex.1023, 7 (Instant Payment Notification “notif[ies] [the merchant]

about events such as...[i]nstant payments, including Express Checkout,...and direct credit card payments.”); Ex.1007, 128 (“The status of the payment: ...Completed: The payment has been completed, and the funds have been added successfully to [the merchant’s] account.”); Ex.1024, 97 (same). For example, the merchant’s PayPal account balance is credited when PayPal completes a credit card transaction from the buyer. See Ex.1025, 45 (“Monthly account statements list the totals of the payments received.”), 62 (“When [the merchant] receive[s] [a]...credit card transaction, the Balance Impact column indicates a credit.”). Accordingly, because PayPal receives the funds from the payer’s bank account, a POSITA would have understood that PayPal sent a request to the payer’s bank to obtain those funds (“*generating and transmitting an electronic request requesting the payment amount from the financial account, obtaining the payment amount, and forwarding at least a portion of the payment amount to the payee*”).

222. It was well-known in the art to send an electronic request to the financial institution associated with the buyer’s financial/billing information to obtain the payment amount and forward it to the merchant. For example, Kloster explains that the “a portion of the monies held for the end user is transferred to the business entity utilizing the retrieved sensitive payment data via a backend payment gateway, 218.” Ex.1013, [0024]; *see* [0014] (“The payment system server 102 is connected to a backend payment gateway 114 for processing transactions.”).

Similarly, Collison also explains that a secure server can mediate the transaction by communicating with the card issuer or bank (depending on the method of payment). *See* Ex.1015, 3:39-49 (“Stripe (300) settles the charge on behalf of the Merchant (100) with the Processor (400) or directly with the Card Network (500). 8. The Card Network (500) causes the funds to be paid by the Card Issuing Bank (600) to Stripe (300) or to Stripe’s Acquiring Bank (700). 9. Stripe (300) causes the settled funds to be sent to the Merchant (100) (or to the Merchant’s Bank (800)), net of any applicable fees....10. The Card Issuing Bank (600) collects the paid funds from the Customer (200).”); 23:2-17 (“All charges flagged for settlement are submitted in batches to the Processor (400) and/or Card Network (500) directly. The Card Network (500) then causes the funds to be paid by the Card Issuing Bank (600) to Stripe (300) or to Stripe's Acquiring Bank (700). Stripe (300) then causes the settled funds to be sent to the Merchant (100) (or to the Merchant's Bank (800)), net of any applicable fees. The Card Issuing Bank (600) collects the paid funds from the Customer (200). Stripe (300) settles the charge on behalf of the Merchant (100) with the Processor (400) or directly with the Card Network (500). 8. The Card Network (500) causes the funds to be paid by the Card Issuing Bank (600) to Stripe (300) or to Stripe’s Acquiring Bank (700). 9. Stripe (300) causes the settled funds to be sent to the Merchant (100) (or to the Merchant’s Bank (800)), net of any applicable fees....10. The Card Issuing Bank

(600) collects the paid funds from the Customer (200).”).

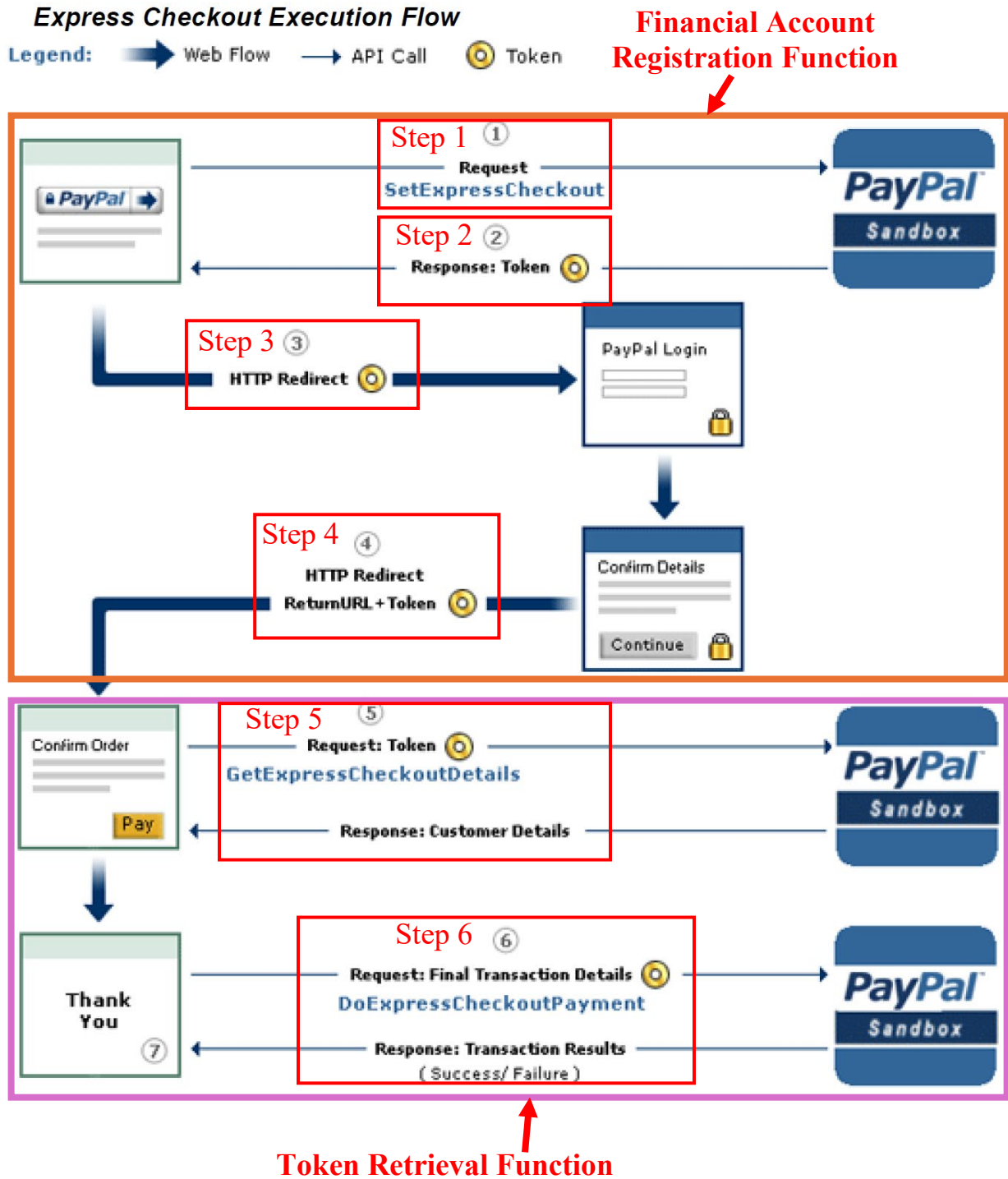
223. Thus, because PayPal Express Checkout discloses that a merchant completes the transaction by invoking the DoExpressCheckoutPayment so that PayPal mediates the transaction and a POSITA would have understood that this involves sending a request to a payment gateway to obtain the payment amount and transfer it to the merchant, PayPal Express Checkout renders obvious *“processing the payment transaction using the sensitive financial account information by generating and transmitting an electronic request requesting the payment amount from the financial account, obtaining the payment amount, and forwarding at least a portion of the payment amount to the payee”* as claimed.

4. Claim 2

[2.1] *The method of claim 1, further comprising: receiving a second payment transaction instruction from the merchant server, the second payment transaction instruction including a representation of the non-sensitive electronic data token and specifying a second payment amount;*

224. First, it would have been obvious to a POSITA that PayPal’s servers would have received *“a second payment transaction instruction from the merchant server.”* As explained in claim 1 above, PayPal Express Checkout discloses that a merchant collects a first payment from a buyer by invoking SetExpressCheckout to set up the Express Checkout transaction (Steps 1-2), redirecting the buyer’s browser to PayPal’s website so that the buyer can submit her financial/billing information to PayPal’s servers (Steps 3-4), invoking GetExpressCheckoutDetails

to retrieve the Payer ID (Step 5), then invoking DoExpressCheckoutPayment to complete the transaction (Step 6), as illustrated in the Example Transaction below:



Ex.1006, 24 (annotated).

225. A POSITA would have understood that a buyer could make a second purchase on the merchant's website, which initiates the Example Transaction a second time. During this second process, the merchant sends a DoExpressCheckoutPayment request (*"a second payment transaction instruction from the merchant server"*) to PayPal's servers to request PayPal's servers transfer the second payment from the buyer to the merchant, as explained above in limitation [1.4.2].

226. Second, PayPal Express Checkout discloses that the DoExpressCheckoutPayment request *"includ[es] a representation of the non-sensitive electronic data token and specif[ies] a second payment amount."* When invoking DoExpressCheckoutPayment, the merchant specifies the Payer ID (*"non-sensitive electronic data token"*) and the payment amount (*"second payment amount"*) to complete the transaction. Ex.1006, 21-23, 26-27. As explained above in limitation [1.4.1], the merchant retrieves the Payer ID (*"non-sensitive electronic data token"*) by sending a GetExpressCheckoutDetails request to PayPal's servers. Because the Payer ID is a *"non-sensitive electronic data token,"* it is itself a *"representation of the non-sensitive electronic data token."*

227. Thus, because a POSITA would have understood that a second payment could be collected by repeating an Express Checkout transaction for the

same buyer, PayPal Express Checkout renders obvious “*receiving a second payment transaction instruction from the merchant server, the second payment transaction instruction including a representation of the non-sensitive electronic data token and specifying a second payment amount*” as claimed.

[2.2] *processing the second payment by obtaining the second payment amount using the sensitive financial account information represented by the non-sensitive electronic data token; and forwarding at least a portion of the second payment amount to the payee.*

228. As discussed in [1.4.2], in response to receiving the DoExpressCheckoutPayment request, PayPal’s servers “*process[] the payment transaction using the sensitive financial account information*” by “*obtaining the payment amount, and forwarding at least a portion of the payment amount to the payee.*”

229. As discussed in [2.1], a POSITA would have understood that the Payer ID would have been used to complete the second payment. The Payer ID is a “[u]nique PayPal customer account identification number.” Ex.1007, 100; *see also* Ex.1006, 17 (“The buyer’s unique identifier (Payer ID)”). A POSITA would have understood that, if the same buyer makes a second purchase on the merchant’s website using PayPal, the same unique PayPal customer account identification number (Payer ID) will be used to identify the buyer that was used for the first purchase. Thus, it would have been obvious to a POSITA that PayPal’s servers would use the financial account information represented by the Payer ID to

complete the second payment.

230. Thus, because a POSITA would have understood that the second payment would be processed in the same way as the first payment, PayPal Express Checkout renders obvious “*processing the second payment by obtaining the second payment amount using the sensitive financial account information represented by the non-sensitive electronic data token; and forwarding at least a portion of the second payment amount to the payee*” as claimed.

5. Claim 3

[3.1] *The method of claim 2, wherein the representation of the non-sensitive electronic data token is the non-sensitive electronic data token.*

231. As explained above in limitation [2.1], a POSITA would have understood that the second payment is collected using the Payer ID (“non-sensitive electronic data token”) by performing a second Express Checkout transaction.

Thus, because a POSITA would have understood that a second payment would have been collected using the Payer ID associated with the buyer, PayPal Express Checkout renders obvious “*wherein the representation of the non-sensitive electronic data token is the non-sensitive electronic data token*” as claimed.

6. Claim 4

[4.1] *The method of claim 1, wherein the non-sensitive electronic data token constitutes a symbolic representation of the sensitive financial account information.*

232. As explained above in limitation [1.4.1], the Payer ID represents the

buyer's financial/billing information. The Payer ID is a "[u]nique PayPal customer account identification number" such as 7AKUSARZ7SAT8. Ex.1007, 100; Ex.1006, 26. Thus, because the Payer ID is a customer account identification number that represents that financial/billing information, the Payer ID ("*non-sensitive electronic data token*") is a "*symbolic representation*" of the financial/billing information ("*financial account information*"), rendering this limitation obvious.

7. Claim 5

[5.1] *The method of claim 1, wherein the at least one data element associated with the payer is a customer identification number generated by the merchant server to identify the payer.*

233. As explained above in limitation [1.1.3], the PayPal's servers receive the token ("*at least one data element associated with the payer*") and a payment amount from the merchant when the merchant invokes DoExpressCheckoutPayment. Ex.1006, 21 ("To execute an Express Checkout transaction...Specify the token returned by PayPal when it redirects the buyer's browser to [the merchant's] site...Specify the amount of the order."). The token "associates the execution of API operations and commands with a specific instance of a user experience flow." Ex.1006, 16. The merchant may set the value of the token when it calls the SetExpressCheckout API operation to set up the transaction: "If [the merchant] set[s] the token in the SetExpressCheckout request,

the value of the token in the response is identical to the value in the request.”

Ex.1007, 91.

234. A POSITA would have understood that the merchant can set any value as the token. For example, Kloster explains that “the payment system server 102 may include memory 116 for storing the sensitive payment data 112, the payment data token 110, and possibly other information regarding customers in an electronic database 118.” Ex.1013, [0015]. “[A] unique customer identifier created by the merchant could also be stored for the customer.” Ex.1013, [0015]. Thus, the business entity may send the unique customer identifier to the payment system server so that it can be stored with the payment data, payment data token, and any other information stored for the customer.

235. Thus, a POSITA would have found it obvious that the merchant would set the token value to be a customer identification number (a unique customer identifier) generated by the merchant so that the transaction can be tied both to a particular user experience flow and to the customer identification number set by the merchant and used in its system to identify the buyer.

236. Thus, because PayPal Express Checkout discloses that the token value may be set by the merchant when it invokes the SetExpressCheckout API operation and a POSITA would have understood that its customer identification number could be used as the token value, PayPal Express Checkout and Kloster

render obvious “wherein the at least one data element associated with the payer is a customer identification number generated by the merchant server to identify the payer” as claimed.

8. Claim 6

[6.1] *The method of claim 1, wherein the secure socket layer connection is synchronous or asynchronous with a process, performed by the merchant server, to receive enrollment data provided by the payer.*

237. First, PayPal Express Checkout discloses “a process performed by the merchant, to receive enrollment data provided by the payer.”

GetExpressCheckoutDetails allows a merchant to “obtain[] information about the buyer from PayPal, including shipping information.” Ex.1006, 17; *see also* 21

(“Although [the merchant] is not required to invoke the

GetExpressCheckoutDetails API operation, most Express Checkout

implementations take this action to obtain information about the buyer.”). Thus,

although in a basic Express Checkout transaction GetExpressCheckoutDetails only returns the Payer ID and shipping address, any “populated fields are returned in the response.” Ex.1006, 21.

238. PayPal Express Checkout provides dozens of fields that can be returned by GetExpressCheckoutDetails to the merchant. *See* Ex.1007, 92-108 (“GetExpressCheckoutDetails Response Message”). For example, GetExpressCheckoutDetails may return “[i]nformation about the payer”

(“PayerInfo”), the “[p]ayer’s contact telephone number” (“ContactPhone”), and “[t]he buyer’s email address” (“BuyerMarketingEmail”), as shown in the return fields below:

PayerInfo	ebl:PayerInfoType Information about the payer.
-----------	---

Ex.1007, 98

ContactPhone	xs:string Payer’s contact telephone number. NOTE: PayPal returns a contact telephone number only if your Merchant account profile settings require that the buyer enter one. Character length and limitations: Field mask is XXX-XXX-XXXX (for US numbers) or +XXX XXXXXXXXXX (for international numbers)
--------------	---

Ex.1007, 99

BuyerMarketingEmail	ebl:EmailAddressType The buyer email address opted in by the buyer on the PayPal Review page. Limitations: 127 single-byte characters
---------------------	---

Ex.1007, 100. Additionally, GetExpressCheckoutDetails may return the payer’s name (“PayerNameTypeFields”) and shipping address (“AddressType Fields”), as shown in the return fields below:

PayerNameType Fields

Field	Description
Salutation	xs:string Payer's salutation. Character length and limitations: 20 single-byte characters.
FirstName	ebl:PersonNameType Payer's first name. Character length and limitations: 25 single-byte characters.
MiddleName	ebl:NameUser Payer's middle name. Character length and limitations: 25 single-byte characters.
LastName	ebl:NameType Payer's last name. Character length and limitations: 25 single-byte characters.
Suffix	ebl:SuffixType Payer's suffix. Character length and limitations: 12 single-byte characters.

Ex.1007, 101**AddressType Fields**

Field	Description
Name	xs:string Person's name associated with this shipping address. Character length and limitations: 32 single-byte characters.

Field	Description
Street1	xs:string First street address. Character length and limitations: 100 single-byte characters.
Street2	xs:string Second street address. Character length and limitations: 100 single-byte characters.
CityName	xs:string Name of city. Character length and limitations: 40 single-byte characters.
StateOrProvince	xs:string State or province. Character length and limitations: 40 single-byte characters.
PostalCode	xs:string U.S. zip code or other country-specific postal code. Character length and limitations: 20 single-byte characters.
Country	ebl:CountryCodeType Country code. Character limit: 2 single-byte characters.
Phone	xs:string Phone number. Character length and limit: 20 single-byte characters.
AddressStatus	ebl:AddressStatusTypeCode Status of street address on file with PayPal. Valid values are: • none • Confirmed • Unconfirmed

Ex.1007, 101-102

239. For a digital goods payment, GetExpressCheckoutDetails returns various information about the payer, including the “[e]mail address of the payer” (“Payer”), the “[f]irst and last name of the payer” (“PayerName”), and the “[p]ayer’s shipping address information” (“Address”), as shown in the return fields below:

PayerInfoType Fields for Digital Goods

Field	Description
Payer	ebl:EmailAddressType Email address of payer. Character length and limitations: 127 single-byte characters
PayerID	ebl:UserIDType Unique PayPal customer account identification number. Character length and limitations: 13 single-byte alphanumeric characters
PayerName	ebl:PersonNameType First and last name of payer.
PayerCountry	ebl:CountryCodeType Payer's country of residence in the form of ISO standard 3166 two-character country codes. Character length and limitations: Two single-byte characters
PayerBusiness	xs:string Payer's business name. Character length and limitations: 127 single-byte characters
Address	xs:string Payer's shipping address information.

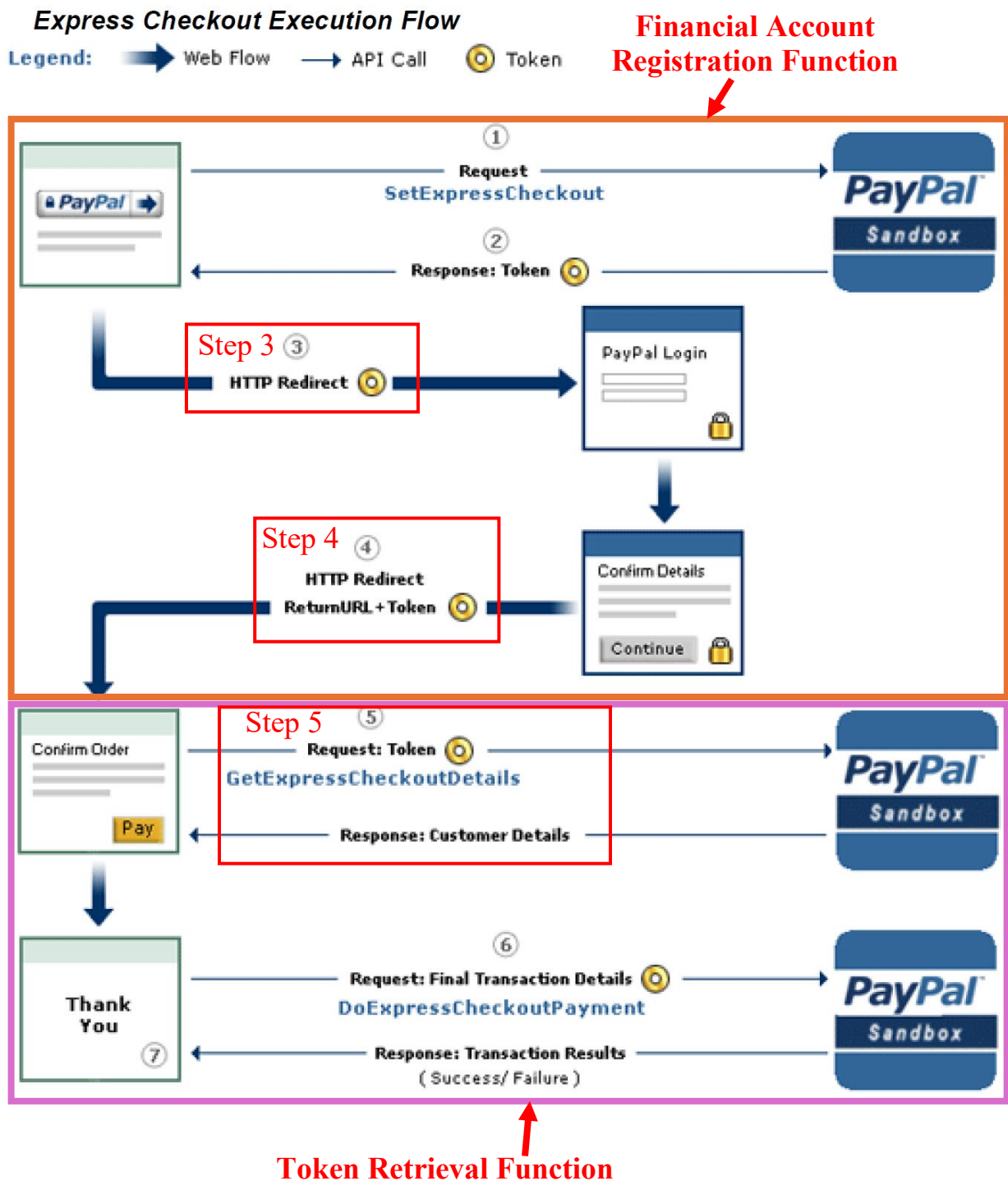
Ex.1007, 100-101

240. As indicated by the '621 patent, enrollment data can include any data that can be used to enroll a buyer in an account with the merchant or the secure server (e.g. PayPal). *See* Ex.1001, 2:64-3:3 (“The token is an index to the stored financial account information for that customer that is stored by the merchant (or a server or service controlled by the merchant or merchant's agent) in connection with the customer record and is used to identify the customer account to be debited to process a payment authorized by the customer.”), 3:9-15 (“By retaining the

token corresponding to the customer account to be debited, the merchant retains control of the customer's enrollment information and the ability to process payments whenever authorized (using financial account information provided by the customer) but does not hold the underlying financial account information or have security responsibility for that information.”), 7:53-55 (“The method starts with step 302, in which enrollment data identifying a payer is stored in a first computing system.”). Thus, any of the information described above that the merchant receives from GetExpressCheckoutDetails (e.g. the buyer’s phone number, name, email, and/or shipping address) may be enrollment data that a merchant can use to create a customer account for the buyer, including the buyer’s phone number, name, email, and/or shipping address. *See* Ex.1006, 69 (The merchant “should log basic information about each PayPal API operation [it] execute[s]. All responses to PayPal API operations contain information that may be useful for debugging purposes. [The merchant] should log the Correlation ID, which identifies the API operation to PayPal, and response-specific information, such as the transaction ID, which [the merchant] can use to review a transaction on the PayPal website or through the API. **[The merchant] can log other information that may be useful**, such as the timestamp. **[The merchant] could implement a scheme that logs the entire request and response** in a ‘verbose’ mode.”).

241. Therefore, the merchant invoking GetExpressCheckoutDetails is a “*process, performed by the merchant server, to receive enrollment data provided by the payer.*”

242. Second, PayPal Express Checkout discloses that “*the secure socket layer [SSL] connection is synchronous or asynchronous*” with the enrollment process performed by the merchant. PayPal Express Checkout explains that the merchant “invoke[s] the GetExpressCheckoutDetails API operation from the page specified by return URL, which [the merchant] set in [its] call to the SetExpressCheckout API.” Ex.1006, 21. The merchant typically invokes GetExpressCheckoutDetails “as soon as the redirect occurs and use[s] the information in the response to populate [its] review page,” in Step 5 of the Example Transaction. Ex.1006, 21. This is illustrated as Step 5 in the Example Transaction:



Ex.1006, 24 (annotated)

243. As discussed in [1.5.1.1], when the merchant redirects the buyer's

browser to PayPal's website, an SSL connection is established between the buyer's browser and PayPal's servers. *See* Ex.1006, 25 ("If the [SetExpressCheckout] operation was successful, use the token and redirect [the] browser to the Sandbox to log in" as the buyer.). After the buyer confirms her financial/billing information, PayPal's servers redirect the buyer's browser back to the merchant's website in Step 4 of the Example Transaction using the return URL the merchant specified in SetExpressCheckout. Ex.1006, 25 ("When [the merchant testing Express Checkout] confirm[s], the Sandbox redirects [the] browser to the return URL [the merchant] specified when invoking the SetExpressCheckout API operation.>"). "The return URL is the page to which PayPal redirects [the] buyer's browser after the buyer logs into PayPal and approves the payment." Ex.1006, 20.

244. Thus, if the merchant calls GetExpressCheckoutDetails as soon as the second redirect occurs, the process of receiving enrollment data may be asynchronous with the SSL connection.

245. Therefore, because PayPal Express Checkout discloses the merchant receives enrollment data from the payer using GetExpressCheckoutDetails asynchronously with the SSL connection, PayPal Express Checkout renders obvious "*wherein the secure socket layer connection is synchronous or asynchronous with a process, performed by the merchant server, to receive enrollment data provided by the payer*" as claimed.

9. Claim 7

[7.1] *The method of claim 1, wherein the sensitive financial account information includes at least one of a credit card number or bank account number.*

246. First, PayPal Express Checkout discloses that the “*financial account information*” includes “*a credit card number.*” As explained above in limitations [1.1.5.3]-[1.1.5.4], PayPal’s servers provide forms for a buyer to input her billing and/or payment information so that it can be sent to PayPal’s servers. When using Express Checkout, the buyer “log[s] in to PayPal or choose[s] other ways to pay.” Ex.1009, 12. Thus, PayPal’s servers provide a “Choose a way to pay” form where “[b]uyers enter their PayPal credentials to log in and pay” or “[o]ther buyers enter their billing information and other contact information to pay with debit or **credit cards.**” Ex.1009, 11; *see* Ex.1010, 22-23; Ex.1012, 12 (“Buyers without a PayPal account can sign up for one, or proceed as a guest and complete their purchase with a credit card.”). For example, the below screenshot illustrates an exemplary data entry form that allows a buyer to input her credit card number:

Your order summary

Description	Amount
Item description 1	\$79.00
Item # 123	Item Price: \$79.00
Qty: 1	
Item description 2	\$89.00
Item # 123	Item Price: \$89.00
Qty: 1	
Item total	\$168.00
Tax	\$15.54
Shipping and handling	\$18.50
Total	\$202.04 USD

Choose a way to pay

[Pay with my PayPal account](#)

Don't have a PayPal account?

Pay with your debit or credit card as a PayPal guest

Country:

Credit or debit card number:

Expiration Date: /

CVC:

First name:

Last name:

Billing Address Line 1:

Billing Address Line 2 (Optional):

Zip code:

City:

State:

e.g. 555-555-5555

Phone:

Email Address:

This transaction will appear on your statement as: *Merchant name*

[CONTINUE](#)

Secure checkout by **PayPal**

[Cancel and return to <Merchant name>](#)

PayPal protects your privacy and security. For more information, read our [Use Agreement](#) and [Privacy Policy](#).

Cart review area

Data entry area

“credit card number”

Ex.1009, 8 (annotated)

247. Second, PayPal Express Checkout discloses that the “*financial account information*” includes “*a bank account number.*” For example, when “the buyer selects PayPal from [the merchant’s] Payments [*sic*] Methods page,” the below data entry form may be displayed on the buyer’s browser:

WebsiteName.com

Review your payment

If the information below is correct, click **Pay Now** to complete your payment.

[View PayPal policies](#) and your payment source rights.

Description	Amount
Item Total:	\$10.00
Item Total:	\$10.00
Total:	\$10.00 USD

Payment Method: Instant Transfer: Chase Checking (Confirmed) XXXX-XXXX-XXXX-1743
 PayPal will use Visa XXXX-XXXX-XXXX-2183 to fund this transaction if your bank does not have enough funds.
 If funds are added to your PayPal balance before this transaction completes, the additional balance may be used for this transaction. [Learn More](#)
[Change](#)

Ship to: 33 Main Street
 Mountain View, CA 94043
 United States
[Change](#)

Contact information: janesmith@yahoo.com
 408-555-1212
[Change](#)

Pay Now

Cancel and return to [WebsiteName.com](#)

1 PayPal displays order amount.

2 When the customer clicks the **Pay Now** button, they return to your website. Even if you do not display your Review page, you still must call the DoExpressCheckoutPayment API request to complete the PayPal payment.

“bank account number”

Ex.1010, 23 (annotated)

248. Thus, in this example, the buyer has input her “*bank account number*” as her financial/billing information (“*financial account information*”).

249. Therefore, because PayPal Express Checkout discloses using a credit card or bank account number to complete the transaction, PayPal Express Checkout renders obvious “*wherein the sensitive financial account information*

includes at least one of a credit card number or bank account number” as claimed.

10. Claim 15

[15.0] *A method of enrolling a payer by a merchant server operated by or for the benefit of a payee, the method comprising:*

250. As explained above in limitation [1.1], the “merchant server” hosts its website and communicates with PayPal’s server via PayPal’s SOAP API to complete a transaction between the merchant and a buyer.

251. PayPal Express Checkout discloses that the merchant performs various processes that allow a buyer to provide information, including information that can be used for enrollment by either PayPal or the merchant. PayPal Express Checkout explains that SetExpressCheckout “[s]ets up the Express Checkout transaction.” Ex.1006, 17; *see also* 19 (“To set up an Express Checkout transaction, you must invoke the SetExpressCheckout API operation to provide sufficient information to initiate the payment flow and redirect to PayPal if the operation was successful.”). The merchant “specif[ies] values in the SetExpressCheckout request,” which “control the PayPal page flow and the options available to [the merchant] and [its] buyers.” Ex.1006, 19. The merchant then redirects the buyer’s browser to PayPal’s website using the _ExpressCheckout command and the token returned by SetExpressCheckout so that the buyer can provide her information to PayPal. Ex.1006, 17 (“When [the merchant] redirect[s] [the] buyer’s browser to PayPal, [the merchant] must specify the

_ExpressCheckout command for Express Checkout. [The merchant] also specif[ies] the token that identifies the transaction, which was returned by the SetExpressCheckout API operation.”). The merchant then calls GetExpressCheckoutDetails to retrieve the Payer ID and other information about the payer. Ex.1006, 21 (“To obtain details about an Express Checkout transaction, [the merchant] can invoke the GetExpressCheckoutDetails API operation.... Although [the merchant] [is] not required to invoke the GetExpressCheckoutDetails API operation, most Express Checkout implementations take this action to obtain information about the buyer.”).

252. As explained above in limitation [6.1], the merchant can retrieve the Payer ID and as well as “*enrollment data*” about the buyer, including the buyer’s phone number, name, email, and address.

253. PayPal Express Checkout also explains that the merchant “can log other information that may be useful” including “implement[ing] a scheme that logs the entire request and response.” Ex.1006, 69. For example, the merchant may receive an instant payment notification (“IPN”) “when [it] invoke[s] an API operation, such as DoExpressCheckoutPayment.” Ex.1023, 11. When the merchant receives the IPN, the merchant may “store information in a customer relationship management (CRM) or accounting system.” Ex.1023, 11. Thus, PayPal contemplates storing information, such as a response including enrollment

data, in a CRM system for the buyer. A merchant would understand that this would enroll the buyer in the merchant's system with information received about the buyer during the Express Checkout transaction, including all of the information returned by GetExpressCheckoutDetails.

254. Thus, because PayPal Express Checkout discloses that the merchant retrieves information about the buyer that can be used for enrollment and can save information about the buyer in its system, PayPal Express Checkout renders obvious a “*method of enrolling a payer by a merchant server operated by or for the benefit of a payee*” as claimed.

[15.1] *providing a webpage to a payer computing system used by a payer;*

255. This limitation is substantially similar to limitation [1.1], and for the same reasons as discussed above, the prior art renders this limitation obvious.

[15.2] *receiving enrollment data identifying the payer;*

256. As explained in limitation [6.1] above, PayPal Express Checkout describes receiving enrollment data. For example, the merchant may invoke the GetExpressCheckoutDetails API operation to retrieve various information about the buyer from PayPal. *See* Ex.1006, 17, 21. For example, this information may include the buyer's phone number, email, name, or address. Ex.1007, 99, 100, 101, 102. All of this exemplary information identifies the buyer.

257. Thus, because PayPal Express Checkout discloses a merchant

invoking GetExpressCheckoutDetails to retrieve information identifying the buyer, PayPal Express Checkout renders obvious “*receiving enrollment data identifying the payer*” as claimed.

[15.3] *providing, to a secure server via an application programming interface (API) provided by the secure server, at least one data element associated with the payer and a payment amount from the payer to the payee;*

258. This limitation is substantially similar to limitation [1.1.3], and for the same reasons as discussed above, the prior art renders this limitation obvious.

[15.4] *displaying a window or frame within the webpage provided to the payer computing system;*

259. This limitation is substantially similar to limitation [1.1.5.2], and for the same reasons as discussed above, the prior art renders this limitation obvious.

[15.5] *providing authentication credentials to the API to establish an authenticated session, the authentication credentials identifying the payee and authenticating the payee;*

260. As explained above in limitation [1.1.4], PayPal Express Checkout discloses authenticating the merchant. For example, PayPal Express Checkout explains that “[t]o use the PayPal API, [the merchant] must have API credentials that **identify [the merchant]** as a PayPal Business account holder **who is authorized to perform various API operations.**” Ex.1006, 64; *see also* 66-67 (“To enable PayPal to authenticate [the merchant’s] request”: “1. Specify the API user name associated with [the merchant’s] account” and “2. Specify the password associated with the API user name”); Ex.1007, 13 (“The PayPal SOAP API service

is protected to ensure that only authorized PayPal members use it. There are four levels of security” including a “required API username (Username field) and API password (Password field).”).

261. Thus, because PayPal Express Checkout discloses providing API credentials that identify and authenticate the merchant so that the merchant can perform various API operations, PayPal Express Checkout renders obvious “*providing authentication credentials to the API to establish an authenticated session, the authentication credentials identifying the payee and authenticating the payee*” as claimed.

[15.6] *initiating a financial account registration function of the API during the authenticated session, the financial account registration function:*

262. As explained in limitation [1.1.5], PayPal Express Checkout discloses that the merchant initiates the financial account registration function by invoking SetExpressCheckout and redirecting the buyer’s browser with _ExpressCheckout. The merchant “must specify API credentials in each request to execute a PayPal API operation.” Ex.1006, 66; *see also* 66 (“When [the merchant] execute[s] a PayPal API operation, [it] use[s] credentials, such as a signature, to authenticate that [it] [is] requesting the API operation.”). Thus, when invoking SetExpressCheckout, the merchant provides its credentials to authenticate that it is making the request and thereby establishes the “*authenticated session*” for the Express Checkout transaction.

263. Therefore, because PayPal Express Checkout discloses that the merchant provides authentication credentials when invoking SetExpressCheckout, PayPal Express Checkout renders obvious *“initiating a financial account registration function of the API during the authenticated session, the financial account registration function”* as claimed.

[15.6.1] *establishing a secure socket layer connection between the secure server and the payer computing system within the window or frame;*

264. This limitation is substantially similar to limitation [1.1.5.2], and for the same reasons as discussed above, the prior art renders this limitation obvious.

[15.6.2] *outputting instructions to the payer computing system to render a financial account registration request form, within the window or frame, that provides functionality for the payer to provide sensitive financial account information associated with a financial account; and*

265. This limitation is substantially similar to limitation [1.1.5.3], and for the same reasons as discussed above, the prior art renders this limitation obvious.

[15.6.3] *outputting instructions to the payer computing system to encrypt the sensitive financial account information provided by the payer and transmit the encrypted financial account information to the secure server, via the secure socket layer connection, without providing the sensitive financial account information to the merchant server;*

266. As explained in limitation [1.1.5.4], PayPal Express Checkout discloses *“outputting instructions to the payer computing system to encrypt the sensitive financial account information provided by the payer and transmit the encrypted financial account information to the secure server, via the secure socket*

layer connection.”

267. Moreover, PayPal Express Checkout explains that after redirecting the buyer’s browser to PayPal, “[i]f the buyer approves the payment, PayPal redirects to the success URL with the following information:” “[t]he token that was included in the redirect to PayPal” and “[t]he buyer’s unique identifier (Payer ID).”

Ex.1006, 17. Because neither the token nor the Payer ID includes payment information, no payment information is provided to the merchant during the financial account registration function (or at any point during the Express Checkout process) (“*without providing the sensitive financial account information to the merchant server*”).

268. Thus, because PayPal Express Checkout discloses establishing an SSL connection that encrypts and decrypts all messages between the buyer’s browser and PayPal, including payment information provided by the buyer, without sending the payment information to the merchant, PayPal Express Checkout renders obvious “*outputting instructions to the payer computing system to encrypt the sensitive financial account information provided by the payer and transmit the encrypted financial account information to the secure server, via the secure socket layer connection, without providing the sensitive financial account information to the merchant server*” as claimed.

[15.7] *initiating a token retrieval function of the API during the authenticated session, the token retrieval function outputting a non-sensitive electronic data*

token representing the sensitive financial account information to the merchant server;

269. As explained in limitation [1.4], PayPal Express Checkout discloses that the merchant initiates the “*token retrieval function*” by invoking GetExpressCheckoutDetails. Moreover, as explained in limitation [1.4.1], PayPal Express Checkout discloses that PayPal outputs the Payer ID (“*non-sensitive electronic data token representing the sensitive financial account information*”) to the merchant in response to GetExpressCheckoutDetails.

270. The merchant “must specify API credentials in each request to execute a PayPal API operation.” Ex.1006, 66; *see also* 66 (“When [the merchant] execute[s] a PayPal API operation, [it] use[s] credentials, such as a signature, to authenticate that [it] [is] requesting the API operation.”). Thus, the merchant provides its credentials when invoking GetExpressCheckoutDetails to authenticate that it is making the request. Therefore, the merchant initiates the token retrieval function during the “*authenticated session*.”

271. Thus, because PayPal Express Checkout discloses invoking GetExpressCheckoutDetails using its API credentials to retrieve the Payer ID, PayPal Express Checkout renders obvious “*initiating a token retrieval function of the API during the authenticated session, the token retrieval function outputting a non-sensitive electronic data token representing the sensitive financial account information to the merchant server*” as claimed.

[15.8] *receiving the non-sensitive electronic data token from the secure server, wherein the merchant server does not receive the sensitive financial account information represented by the non-sensitive electronic data token and the secure server does not provide the non-sensitive electronic data token to the payer;*

272. This limitation is substantially similar to limitation [1.4.1], and for the same reasons as discussed above, the prior art renders this limitation obvious.

[15.9] *storing the non-sensitive electronic data token in association with the enrollment data identifying the payer; and*

273. As explained above in limitation [6.1], the merchant can retrieve the Payer ID (“token”) and “enrollment data identifying the payer” including the buyer’s phone number, name, email, and address from PayPal’s servers by invoking GetExpressCheckoutDetails. *See* Ex.1007, 92-108 (“GetExpressCheckoutDetails Response Message”).

274. The merchant “can log other information that may be useful” including “implement[ing] a scheme that logs the entire request and response.” Ex.1006, 69. Thus, if the GetExpressCheckoutDetails response includes both the Payer ID and information about the buyer, when the merchant logs the response, it will store the Payer ID (“token”) along with the information about the buyer (“enrollment data”).

275. Moreover, PayPal Express Checkout explains that the merchant may receive an instant payment notification (“IPN”) “when [it] invoke[s] an API operation, such as DoExpressCheckoutPayment.” Ex.1023, 11. When the

merchant receives the IPN, the merchant may “store information in a customer relationship management (CRM) or accounting system.” Ex.1023, 11. Thus, PayPal Express Checkout explains that information may be stored in an account for the buyer. A POSITA would have understood that the merchant would store the Payer ID, which is used to complete the transaction, and the information about the buyer, which may be used to fulfill the order, together within the buyer’s CRM account.

276. Therefore, because PayPal Express Checkout discloses storing the Payer ID and information about the buyer returned by GetExpressCheckoutDetails in a customer account, PayPal Express Checkout renders obvious “*storing the non-sensitive electronic data token in association with the enrollment data identifying the payer*” as claimed.

[15.10] *processing a payment from the payer by generating a payment transaction instruction, using the token retrieval function of the API, that includes a representation of the non-sensitive electronic data token and instructs the secure server to generate and transmit an electronic request requesting the payment amount from the financial account, obtain the payment amount, and forward at least a portion of the payment amount.*

277. This limitation is substantially similar to limitation [1.4.2], and for the same reasons as discussed above, the prior art renders this limitation obvious.

11. Claim 16

[16.0] *The method of claim 15, wherein the representation of the non-sensitive electronic data token is the non-sensitive electronic data token.*

278. As explained in limitation [1.4.2], the merchant uses the Payer ID in the DoExpressCheckoutPayment API operation to complete the transaction. The Payer ID is the “*non-sensitive electronic data token*.” Thus, for these reasons, PayPal Express Checkout renders obvious “*wherein the representation of the non-sensitive electronic data token is the non-sensitive electronic data token*” as claimed.

12. Claim 17

[17.1] *The method of claim 15, wherein the non-sensitive electronic data token constitutes a symbolic representation of the sensitive financial account information.*

279. This limitation is substantially similar to limitation [4.1], and for the same reasons as discussed above, the prior art renders this limitation obvious.

13. Claim 18

[18.1] *The method of claim 15, wherein the at least one data element associated with the payer is a customer identifier generated by the merchant server to identify the payer.*

280. This limitation is substantially similar to limitation [5.1], and for the same reasons as discussed above, the prior art renders this limitation obvious.

14. Claim 19

[19.1] *The method of claim 15, wherein the secure socket layer connection provides access to the secure server using a static or dynamic uniform resource locator (URL).*

281. This limitation is substantially similar to limitations [1.5.1.1a]-

[1.1.5.1b], and for the same reasons as discussed above, the prior art renders this limitation obvious.

15. Claim 20

[20.1] *The method of claim 15, wherein the secure socket layer connection is synchronous or asynchronous with a process, performed by the merchant server, to receive enrollment data from the payer.*

282. This limitation is substantially similar to limitation [6.1], and for the same reasons as discussed above, the prior art renders this limitation obvious.

16. Claim 21

[21.1] *The method of claim 15, wherein the enrollment data identifying the payer is received from the payer over a network.*

283. PayPal Express Checkout discloses this limitation. As explained in limitation [1.1.5.1], PayPal Express Checkout discloses that the merchant redirects the buyer's browser to PayPal's website using a URL including an HTTPS request, `_ExpressCheckout`, and the token, thereby establishing an SSL connection between the buyer's browser and PayPal's servers. See Ex.1006, 17 ("PayPal provides a command that [the merchant] use[s] when redirecting [its] buyer's browser to PayPal. This command enables [the] buyer to log into PayPal to approve an Express Checkout payment."). As explained in limitation [6.1], the merchant can retrieve information about the buyer ("*enrollment data*") such as her phone number, email, name or address by invoking `GetExpressCheckoutDetails`.

284. As explained in limitations [1.1.5.3] and [1.2], the buyer provides and

submits her financial/billing information via a data entry form. *See* Ex.1009, 8, 10-12 (“Choose a way to pay – Buyers enter their PayPal credentials to log in and pay. Other buyers **enter their billing information** and other contact information to pay with debit or credit cards.”); Ex.1012, 11-12 (From the Log in to your PayPal account page, “the buyer can log in” to her PayPal account or “[b]uyers without a PayPal account can sign up for one, or proceed as a guest and complete their purchase with a credit card.”) (emphasis removed). An example data entry form is shown below:

Your order summary

Description	Amount
Item description 1	
Item # 123	\$79.00
Item Price: \$79.00	
Qty: 1	
Item description 2	
Item # 123	\$89.00
Item Price: \$89.00	
Qty: 1	
Item total	\$168.00
Tax	\$15.54
Shipping and handling	\$18.50
Total	\$202.04 USD

Choose a way to pay

Pay with my PayPal account

Don't have a PayPal account?

Pay with your debit or credit card as a PayPal guest

Country: United States

Credit or debit card number

Expiration Date: mm / yy

CSC

Why is this?

First name

Last name

Billing Address Line 1

Billing Address Line 2 (Optional)

Zip code

City

State: Select

Phone: 800-555-5555

Email Address

This transaction will appear on your statement as: merchant name

CONTINUE

Secure checkout by PayPal

Cancel and return to: <merchant name>

PayPal protects your privacy and security. For more information, read our [Use Agreement](#) and [Privacy Policy](#).

Cart review area

Data entry area

“sensitive financial account information”

Name

Address

Phone

Email

Ex.1009, 8 (annotated)

285. This example form allows the buyer to provide her phone number, email, name, or address.

286. Thus, because PayPal Express Checkout discloses that the buyer’s browser sends the provided information via a data entry form to PayPal via the SSL connection over the internet, PayPal Express Checkout renders obvious “wherein the enrollment data identifying the payer is received from the payer over a network” as claimed.

17. Claim 22

[22.1] *The method of claim 15, wherein the sensitive financial account information includes at least one of a credit card number or bank account number.*

287. This limitation is substantially similar to limitation [7.1], and for the same reasons as discussed above, the prior art renders this limitation obvious.

18. Claim 25

[25.0] *A method of processing a payment transaction from a payer to a payee, the method being performed by one or more secure servers, the method comprising:*

288. This limitation is substantially similar to limitation [1.0], and for the same reasons as discussed above, the prior art renders this limitation obvious.

[25.1] *providing, by the one or more secure servers to a merchant server providing a webpage to a payer computing system used by the payer, an application programming interface (API) that:*

289. This limitation is substantially similar to limitation [1.1], and for the same reasons as discussed above, the prior art renders this limitation obvious.

[25.1.1] *provides financial account registration and token retrieval functions that can be executed to process the payment transaction;*

290. This limitation is substantially similar to limitation [1.1.1], and for the same reasons as discussed above, the prior art renders this limitation obvious.

[25.1.2] *provides access to the financial account registration and token retrieval functions to the merchant server;*

291. This limitation is substantially similar to limitation [1.1.2], and for the same reasons as discussed above, the prior art renders this limitation obvious.

[25.1.3] receives, from the merchant server via the API, at least one data element associated with the payer and a payment amount from the payer to the payee;

292. This limitation is substantially similar to limitation [1.1.3], and for the same reasons as discussed above, the prior art renders this limitation obvious.

[25.1.4] authenticates the payee; and

293. This limitation is substantially similar to limitation [1.1.4], and for the same reasons as discussed above, the prior art renders this limitation obvious.

[25.1.5] executes the financial account registration function, upon initiation by the merchant server, by:

294. This limitation is substantially similar to limitation [1.1.5], and for the same reasons as discussed above, the prior art renders this limitation obvious.

[25.1.5.1] generating a uniform resource locator (URL), for establishing a secure socket layer connection via the internet between the secure server and the payer computing system, the URL comprising either:

295. This limitation is substantially similar to limitation [1.1.5.1], and for the same reasons as discussed above, the prior art renders this limitation obvious.

[25.1.5.1a]-[25.1.5.1b] a dynamic URL generated by the secure server for the payer and the payee; or a static URL and a hypertext transport protocol (HTTP) parameter used by the secure server to identify the payer and the payee;

296. These limitations are substantially similar to limitations [1.1.5.1a]-[1.1.5.1b], and for the same reasons as discussed above, the prior art renders this limitation obvious.

[25.1.5.2] establishing the secure socket layer connection, in response to an HTTP request received for the generated URL, between the secure server and the

payer computing system;

297. This limitation is substantially similar to limitation [1.1.5.2], and for the same reasons as discussed above, the prior art renders this limitation obvious.

[25.1.5.3] outputting instructions to the payer computing system, in response to the HTTP request for the generated URL, to render a financial account registration request form that provides functionality for the payer to provide sensitive financial account information associated with a financial account; and

298. This limitation is substantially similar to limitation [1.1.5.3], and for the same reasons as discussed above, the prior art renders this limitation obvious.

[25.1.5.4] outputting instructions to the payer computing system, in response to the HTTP request for the generated URL, to encrypt the financial account information provided by the payer and transmit the encrypted financial account information to the secure server, via the secure socket layer connection;

299. This limitation is substantially similar to limitation [1.1.5.4], and for the same reasons as discussed above, the prior art renders this limitation obvious.

[25.2] receiving the sensitive financial account information provided by the payer via the secure socket layer connection;

300. This limitation is substantially similar to limitation [1.2], and for the same reasons as discussed above, the prior art renders this limitation obvious.

[25.3] storing the sensitive financial account information in a secure storage location and performing each software process required to maintain compliance with one or more information security standards;

301. This limitation is substantially similar to limitation [1.3], and for the same reasons as discussed above, the prior art renders this limitation obvious.

[25.4] executing a token retrieval function, upon initiation by the merchant

server via the API, by:

302. This limitation is substantially similar to limitation [1.4], and for the same reasons as discussed above, the prior art renders this limitation obvious.

[25.4.1] *providing a non-sensitive electronic data token representing the sensitive financial account information to the merchant server without providing the sensitive financial account information to the merchant server and without providing the non-sensitive electronic data token to the payer; and*

303. This limitation is substantially similar to limitation [1.4.1], and for the same reasons as discussed above, the prior art renders this limitation obvious.

[25.4.2] *processing the payment transaction using the sensitive financial account information by generating and transmitting an electronic request requesting the payment amount from the financial account, obtaining the payment amount, and forwarding at least a portion of the payment amount to the payee.*

304. This limitation is substantially similar to limitation [1.4.2], and for the same reasons as discussed above, the prior art renders this limitation obvious.

19. Claim 26

[26.1] *The method of claim 25, wherein the URL is a redirect URL that redirects the payer computing system to the secure server.*

305. As explained in limitations [1.1.1]-[1.1.2] and [1.1.5.1], the merchant redirects the buyer's browser to PayPal's website using the URL. "PayPal provides a command that [the merchant] use[s] when redirecting [the] buyer's browser to PayPal." Ex.1006, 17. Thus, after setting up the transaction using SetExpressCheckout, the merchant "redirect[s] the buyer's browser to PayPal and execute[s] the _express-checkout command using the token returned in the

SetExpressCheckout response.” Ex.1006, 21. For example, the merchant may redirect the buyer’s browser using the following link:

https://www.sandbox.paypal.com/webscr?cmd=_express-checkout&token=tokenValue.

Ex.1006, 21.

306. Thus, because PayPal Express Checkout discloses redirecting the buyer’s browser to PayPal’s browser using the URL, PayPal Express Checkout renders obvious “*wherein the URL is a redirect URL that redirects the payer computing system to the secure server*” as claimed.

20. Claim 27

[27.1] *The method of claim 25, wherein the non-sensitive electronic data token constitutes a symbolic representation of the sensitive financial account information.*

307. This limitation is substantially similar to limitation [4.1], and for the same reasons as discussed above, the prior art renders this limitation obvious.

B. Ground 2: Claims 8, 9, 10, 11, 12, 13, 14, 23, and 24 obvious over PayPal Express Checkout and Schlesser.

1. Summary of the Schlesser

308. Schlesser “relates generally to electronic commerce, and more particularly to secure online transaction processing.” Ex.1020, [0004].

309. Schlesser discloses a system comprising a customer computer system 14, a merchant computer system 20, and a payment processor system 28, as shown

in Fig. 1 below:

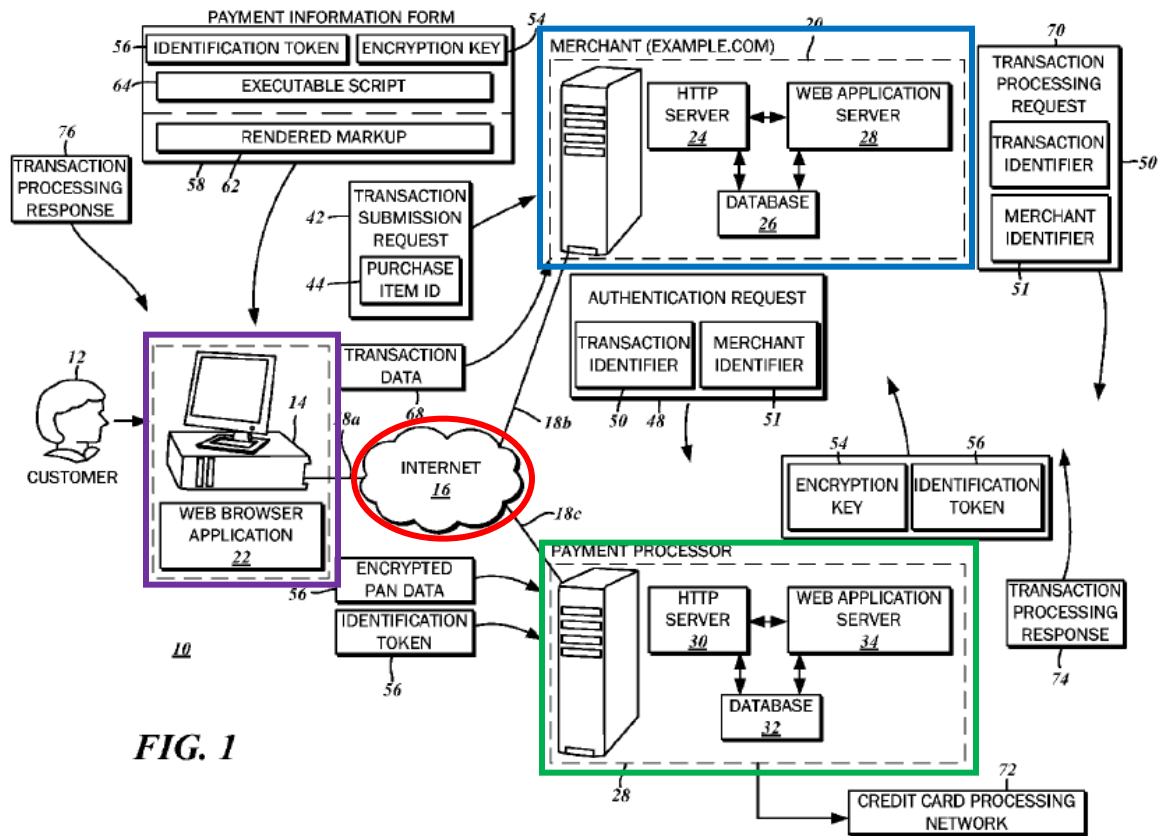


FIG. 1

Ex.1020, Fig. 1

310. Schlessner explains that the merchant computer system and the payment processor system may each include a processor and a memory and each system can communicate via the internet:

Such web servers include the aforementioned merchant computer system 20, also referenced interchangeably herein as the merchant system. **The merchant system 20 is likewise understood to be a computer system that includes a central processor, a memory, and input/output devices, though loaded with computer-executable software instructions of a web server and those that implement one**

or more of the disclosed methods for securely processing an online transaction. In further detail, the merchant system 20 includes an HTTP server 24, along with a database 26 that stores, for example, product catalog data. There also is a web application server 27 that executes the functionality of the contemplated methods. The division of the components of the merchant system 20, that is, the HTTP server 24, the database 26, and the web application server 27, is for the purposes of illustrating its different functional blocks, and is not intended to be limiting. It will be recognized by those having ordinary skill in the art that there are numerous ways in which a merchant system 20 can be configured using a different commercial, off-the-shelf hardware and software components.

Ex.1020, [0027].

Like the merchant system 20, the payment processor system 28 includes a computer system with a central processor, a memory, and input/output devices. The payment processor system 28 is loaded with computer-executable software instructions of a web server, as well as those that implement the methods for securely processing an online transaction.

Ex.1020, [0028]. The customer computer system and the merchant computer system communicate over the internet. Ex.1020, [0024] (“It is understood that the customer computer system 14 can establish a bi-directional data communications link with the merchant computer system 20 over the Internet 16 and the respective Internet links 18 a, 18 b.”). The payment processor system also communicates with the customer computer system and the merchant computer system over the

internet. Ex.1020, [0028] (“In this regard, an Internet link 18 c connects the payment processor system 28 to the Internet 16, and to the customer system 14. Again, there is an HTTPS server 30 that cooperates with a database 32 and a web application server 34 that performs various functions in accordance with the contemplated methods of the present disclosure. The payment processor system 28 is understood to communicate with both the customer system 14 and the merchant system 20.”).

311. The payment processor computer system can handle part of a transaction between the customer computer system and the merchant computer system. Ex.1020, [0028] (“A basic e-commerce transaction typically involves just the customer system 14 and the merchant system 20, though according to the various embodiments of the present disclosure, certain aspects of the transaction are handled by a payment processor computer system 28, also interchangeably referenced as the payment processor system.”).

2. Reasons to Combine Schlesser with PayPal Express Checkout

312. A POSITA would have been motivated to combine the teachings of Schlesser with those of PayPal Express Checkout. As an initial matter, a POSITA when considering the '621 patent, would have looked to the teachings of Schlesser. The field of endeavor of the '621 patent is “obtaining and using account information to process financial payments” so that “merchants are able to process

such transactions without transmitting, processing or storing Financial Account data.” Ex.1001, 1:19-20, 5:63-65. Schlessner is analogous art to the ’621 patent because it is in the field of “securely processing an online transaction between a customer and a merchant” such that “the encrypted PAN data 66, or ciphertext, and the previously received session token 56 are directly transmitted to the payment processor system 28 over the secure data transfer link established with the customer system 14” and “non-sensitive transaction data 68” is transmitted to the merchant system 20. Ex.1020, [0010], [0038], [0040].

313. A POSITA would have been specifically motivated to apply Schlessner’s merchant computer system and payment processor system to PayPal Express Checkout’s methods and systems for conducting an online transaction between a merchant and a buyer using PayPal’s secure servers. PayPal Express Checkout explains that “[l]ike many web services, PayPal SOAP is a combination of client-side and server-side schemas, **hardware and software servers**, and core services.” Ex.1007, 11. Thus, a POSITA would have been motivated to combine hardware servers or systems with those known in the art for handling online transactions between merchants and buyers.

314. First, Schlessner describes a payment processor system that includes a central processor and a memory that is “loaded with computer-executable software instructions of a web server.” Ex.1020, [0028]. Schlessner’s payment processor

system communicates with a merchant computer system and a customer computer system via the internet to handle a transaction between the merchant and customer. Ex.1020, [0028] (“In this regard, an Internet link 18 c connects the payment processor system 28 to the Internet 16, and to the customer system 14. Again, there is an HTTPS server 30 that cooperates with a database 32 and a web application server 34 that performs various functions in accordance with the contemplated methods of the present disclosure. The payment processor system 28 is understood to communicate with both the customer system 14 and the merchant system 20.”). Like PayPal Express Checkout, Schlessner explains that the customer inputs information such as the personal account number (PAN) data into a payment information form rendered on the web browser application of the customer computer system and the PAN data is “directly transmitted to the payment processor system 28 over the secure data transfer link established with the customer system 14.” Ex.1020, [0038]; *see* [0034] (“[T]he payment information form 58 is transmitted to the customer system 14 over a secure data transfer link established with the merchant system 20, and rendered by the web browser application 22. In this regard, the payment information form 58 is understood to have a browser-rendered markup segment 62 comprised of the various text and form input boxes.”); [0036] (“The method continues with a step 204 of receiving personal account number (PAN) data that is entered into the various fields 60a-d of

the payment information form 58 by the customer 12. This data is understood to be sensitive and subject to various technological and legal protective measures to prevent its dissemination and abuse. In addition to PAN data, the payment information form 58 may request non-sensitive data pertinent to merchant system 20 for the transaction, and the step 204 also involves the receiving of such data.”); [0043] (“One contemplated step is processing the payment for completion of the transaction between the customer system 14 and the merchant system 20 using the provided PAN data 66. In one embodiment, the PAN data 66 is a credit card number, and so the payment processor System 28 contacts a credit card processing network 72, which debits the account associated with the customer 12 in an amount specified in the transaction processing request 70.”). Thus, a POSITA would have been motivated to implement PayPal’s servers with the hardware of Schlesser’s payment processor system to perform PayPal Express Checkout’s method of handling a transaction between a merchant and a buyer. A POSITA would have had a reasonable expectation of success. Schlesser’s payment processor system performs similar processes to the PayPal’s servers and, thus, a POSITA would have expected that the hardware components of Schlesser’s payment processor system could be used in PayPal’s servers.

315. Second, Schlesser’s merchant computer system includes a central processing unit and a memory “loaded with computer-executable software

instructions of a web server.” Ex.1020, [0027]. Like PayPal Express Checkout, the customer chooses products and/or services to purchase on the merchant’s website implemented on the merchant computer system, provides a payment information form to the customer, the merchant computer system receives non-sensitive data without receiving the PAN data, and the merchant computer system sends a transaction processing request to the payment processor system to complete the transaction. Ex.1020, [0029] (“As will be recognized, the customer 12, using the web browser application 22, navigates a conventional web-based product catalog or shopping system implemented on the merchant system 20.”); [0034] (“[T]he payment information form 58 is transmitted to the customer system 14 over a secure data transfer link established with the merchant system 20, and rendered by the web browser application 22.”); [0040] (“[T]he method continues with a step 210 of transmitting non-sensitive transaction data 68 to the merchant system 20.”); [0041] (“[T]he method executed by the merchant system 20 continues with a step 310 of transmitting, to the payment processor system 28, a transaction processing request 70.”). Thus, a POSITA would have been motivated to modify PayPal Express Checkout to replace the merchant’s server with the hardware of Schlessner’s merchant computer system to perform PayPal Express Checkout’s method of handling a transaction between the merchant and a buyer. A POSITA would have had a reasonable expectation of success. Schlessner’s merchant

computer system performs similar processes as PayPal Express Checkout's merchant's server and, thus, a POSITA would have expected that the hardware components of Schlesser's merchant computer system could be used in the merchant server for completing an Express Checkout transaction according to PayPal Express Checkout.

3. Claim 8

[8.0] *A secure server that processes a payment transaction from a payer to a payee, the secure server comprising:*

316. This limitation is substantially similar to limitation [1.0], and for the same reasons as discussed above, the prior art renders this limitation obvious.

[8.1] *one or more processors; and*

317. PayPal Express Checkout and Schlesser render this limitation obvious.

318. As explained in limitation [1.0], PayPal Express Checkout discloses that PayPal's servers handle a transaction between a merchant and a buyer.

Ex.1007, 11 ("PayPal SOAP is a combination of client-side and server-side schemas, hardware and software servers, and core services.").

319. Schlesser explains that a "payment processor system 28 includes a computer system with **a central processor**." Ex.1020, [0028]. As explained above at VI.C.2, a POSITA implementing PayPal Express Checkout's method would have found it obvious, and indeed would have been motivated, to use the hardware

of Schlesser's payment processing system, including the central processor, in PayPal's servers.

320. Thus, because Schlesser discloses a payment processing system with a central processor, PayPal Express Checkout and Schlesser render obvious "*one or more processors*" as claimed.

[8.2] *memory storing instructions that, when executed by the one or more processors, cause the secure server to:*

321. PayPal Express Checkout and Schlesser render this limitation obvious.

322. As explained in limitation [1.0], PayPal Express Checkout discloses that PayPal's servers handle a transaction between a merchant and a buyer. Ex.1007, 11 ("PayPal SOAP is a combination of client-side and server-side schemas, hardware and software servers, and core services.").

323. Schlesser explains that a "payment processor system 28 includes...**a memory**" and "is loaded with **computer-executable software instructions** of a web server, as well as those **that implement the methods for securely processing an online transaction.**" Ex.1020, [0028]. As explained above at VI.C.2, a POSITA implementing PayPal Express Checkout's method would have found it obvious, and indeed would have been motivated, to use the hardware of Schlesser's payment processing system, including the memory with computer-executable instructions, in PayPal's servers.

324. Thus, because Schlesser discloses a payment processing system with a memory loaded with computer-executable instructions for securely processing an online transaction, PayPal Express Checkout and Schlesser render obvious “memory storing instructions that, when executed by the one or more processors” as claimed.

[8.2.1] *provide an application programming interface (API) to a merchant server providing a webpage to a payer computing system used by the payer, the API configured to:*

325. This limitation is substantially similar to limitation [1.1], and for the same reasons as discussed above, the prior art renders this limitation obvious.

[8.2.1.1] *provide financial account registration and token retrieval functions that can be executed to process the payment transaction;*

326. This limitation is substantially similar to limitation [1.1.1], and for the same reasons as discussed above, the prior art renders this limitation obvious.

[8.2.1.2] *provide access to the financial account registration and token retrieval functions to the merchant server;*

327. This limitation is substantially similar to limitation [1.1.2], and for the same reasons as discussed above, the prior art renders this limitation obvious.

[8.2.1.3] *receive, from the merchant server via the API, at least one data element associated with the payer and a payment amount from the payer to the payee;*

328. This limitation is substantially similar to limitation [1.1.3], and for the same reasons as discussed above, the prior art renders this limitation obvious.

[8.2.1.4] *authenticate the payee;*

329. This limitation is substantially similar to limitation [1.1.4], and for the same reasons as discussed above, the prior art renders this limitation obvious.

[8.2.1.5] *execute the financial account registration function, upon initiation by the merchant server, by:*

330. This limitation is substantially similar to limitation [1.1.5], and for the same reasons as discussed above, the prior art renders this limitation obvious.

[8.2.1.5.1] *generating a uniform resource locator (URL) to establish a secure socket layer connection via the internet between the secure server and the payer computing system, the URL comprising either:*

331. This limitation is substantially similar to limitation [1.1.5.1], and for the same reasons as discussed above, the prior art renders this limitation obvious.

[8.2.1.5.1a]-[8.2.1.5.1b] *a dynamic URL generated by the secure server for the payer and the payee; or a static URL and a hypertext transport protocol (HTTP) parameter used by the secure server to identify the payer and the payee;*

332. These limitations are substantially similar to limitations [1.1.5.1a] and [1.1.5.1b], and for the same reasons as discussed above, the prior art renders this limitation obvious.

[8.2.1.5.2] *establishing the secure socket layer connection, in response to an HTTP request received from the merchant server for the generated URL, between the secure server and the payer computing system within a window or frame that is displayed within the webpage provided by the merchant server;*

333. This limitation is substantially similar to limitation [1.1.5.2], and for the same reasons as discussed above, the prior art renders this limitation obvious.

[8.2.1.5.3] *outputting instructions to the payer computing system, in response to the HTTP request for the generated URL, to render a financial account*

registration request form, within the window or frame that is displayed within the webpage provided by the merchant server, that provides functionality for the payer to provide sensitive financial account information associated with a financial account; and

334. This limitation is substantially similar to limitation [1.1.5.3], and for the same reasons as discussed above, the prior art renders this limitation obvious.

[8.2.1.5.4] outputting instructions to the payer computing system, in response to the HTTP request for the generated URL, to encrypt the sensitive financial account information provided by the payer and transmit the encrypted financial account information to the secure server via the secure socket layer connection;

335. This limitation is substantially similar to limitation [1.1.5.4], and for the same reasons as discussed above, the prior art renders this limitation obvious.

[8.2.1.6] receive the sensitive financial account information provided by the payer via the secure socket layer connection;

336. This limitation is substantially similar to limitation [1.2], and for the same reasons as discussed above, the prior art renders this limitation obvious.

[8.2.1.7] store the sensitive financial account information in a secure storage location and perform each software process required to maintain compliance with one or more information security standards;

337. This limitation is substantially similar to limitation [1.3], and for the same reasons as discussed above, the prior art renders this limitation obvious.

[8.2.1.8] execute a token retrieval function, upon initiation by the merchant server by the API, by:

338. This limitation is substantially similar to limitation [1.4], and for the same reasons as discussed above, the prior art renders this limitation obvious.

[8.2.1.8.1] *providing a non-sensitive electronic data token representing the sensitive financial account information to the merchant server without providing the sensitive financial account information to the merchant server and without providing the non-sensitive electronic data token to the payer; and*

339. This limitation is substantially similar to limitation [1.4.1], and for the same reasons as discussed above, the prior art renders this limitation obvious.

[8.2.1.8.2] *processing the payment transaction using sensitive the financial account information by generating and transmitting an electronic request requesting the payment amount from the financial account, obtaining the payment amount, and forwarding at least a portion of the payment amount to the payee.*

340. This limitation is substantially similar to limitation [1.4.2], and for the same reasons as discussed above, the prior art renders this limitation obvious.

4. Claim 9

[9.1] *The secure server of claim 8, wherein the instructions further cause the secure computing system to: receive a second payment transaction instruction from the merchant server, wherein the second payment transaction instruction includes a representation of the non-sensitive electronic data token and specifies a second payment amount;*

341. This limitation is substantially similar to limitation [2.1], and for the same reasons as discussed above, the prior art renders this limitation obvious.

[9.2] *process the second payment by obtaining the second payment amount using the sensitive financial account information represented by the non-sensitive electronic data token; and*

342. This limitation is substantially similar to limitation [2.2], and for the same reasons as discussed above, the prior art renders this limitation obvious.

[9.3] *forward at least a portion of the second payment amount to the payee.*

343. This limitation is substantially similar to limitation [2.3], and for the same reasons as discussed above, the prior art renders this limitation obvious.

5. Claim 10

[10.1] *The secure server of claim 9, wherein the representation of the non-sensitive electronic data token is the non-sensitive electronic data token.*

344. This limitation is substantially similar to limitation [3.1], and for the same reasons as discussed above, the prior art renders this limitation obvious.

6. Claim 11

[11.1] *The secure server of claim 8, wherein the non-sensitive electronic data token constitutes a symbolic representation of the sensitive financial account information.*

345. This limitation is substantially similar to limitation [4.1], and for the same reasons as discussed above, the prior art renders this limitation obvious.

7. Claim 12

[12.1] *The secure server of claim 8, wherein the at least one data element associated with the payer is a customer identification number generated by the merchant server to identify the payer.*

346. This limitation is substantially similar to limitation [5.1], and for the same reasons as discussed above, the prior art renders this limitation obvious.

8. Claim 13

[13.1] *The secure server of claim 8, wherein the secure socket layer connection is synchronous or asynchronous with a process, performed by the merchant server, to receive enrollment data from the payer.*

347. This limitation is substantially similar to limitation [6.1], and for the

same reasons as discussed above, the prior art renders this limitation obvious.

9. Claim 14

[14.1] *The secure server of claim 8, wherein the sensitive financial account information includes at least one of a credit card number or bank account number.*

348. This limitation is substantially similar to limitation [7.1], and for the same reasons as discussed above, the prior art renders this limitation obvious.

10. Claim 23

[23.0] *A merchant server, operated by or for the benefit of a payee, comprising:*

349. This limitation is substantially similar to limitation [15.0], and for the same reasons as discussed above, the prior art renders this limitation obvious.

[23.1] *one or more computer processors; and*

350. PayPal Express Checkout and Schlessner render this limitation obvious.

351. As explained in limitation [15.0], a POSITA would have found it obvious over PayPal Express Checkout that the merchant implements its website and interacts with PayPal's servers using a merchant server.

352. Schlessner explains that a "merchant system 20 is likewise understood to be a computer system that includes **a central processor.**" Ex.1020, [0027]. As explained above at VI.C.2, a POSITA implementing PayPal Express Checkout's method would have found it obvious, and indeed would have been motivated, to

use the hardware of Schlessner's merchant computer system, including the central processor, as the merchant server.

353. Thus, because Schlessner discloses a merchant computer system with a central processor, PayPal Express Checkout and Schlessner render obvious "*one or more computer processors*" as claimed.

[23.2] *non-transitory computer readable storage media storing instructions that, when executed by the one or more computer processors, cause the merchant server to:*

354. PayPal Express Checkout and Schlessner render this limitation obvious.

355. Schlessner explains that "[t]he merchant system 20 is likewise understood to be a computer system that includes...**a memory**" and is "**loaded with computer-executable software instructions** of a web server and those **that implement one or more of the disclosed methods for securely processing an online transaction.**" Ex.1020, [0027]. As explained above at VI.C.2, a POSITA implementing PayPal Express Checkout's method would have found it obvious, and indeed would have been motivated, to use the hardware of Schlessner's merchant computer system, including the memory with computer-executable instructions, in the merchant's servers. A POSITA would have understood that a memory is a "*non-transitory computer readable storage media.*"

356. Thus, because Schlessner discloses a merchant computer system with a

memory loaded with computer-executable instructions for securely processing an online transaction, PayPal Express Checkout and Schlesser render obvious “*non-transitory computer readable storage media storing instructions that, when executed by the one or more computer processors*” as claimed.

[23.2.1] *provide a webpage to a payer computing system used by a payer;*

357. This limitation is substantially similar to limitation [15.1], and for the same reasons as discussed above, the prior art renders this limitation obvious.

[23.2.2] *receive enrollment data identifying the payer;*

358. This limitation is substantially similar to limitation [15.2], and for the same reasons as discussed above, the prior art renders this limitation obvious.

[23.2.3] *provide, to a secure server via an application programming interface (API) provided by the secure server, at least one data element associated with the payer and a payment amount from the payer to the payee;*

359. This limitation is substantially similar to limitation [15.3], and for the same reasons as discussed above, the prior art renders this limitation obvious.

[23.2.4] *display a window or frame within the webpage provided to the payer computing system;*

360. This limitation is substantially similar to limitation [15.4], and for the same reasons as discussed above, the prior art renders this limitation obvious.

[23.2.5] *provide authentication credentials to the API to establish an authenticated session, the authentication credentials identifying the payee and authenticating the payee;*

361. This limitation is substantially similar to limitation [15.5], and for the

same reasons as discussed above, the prior art renders this limitation obvious.

[23.2.6] *initiate a financial account registration function of the API during the authenticated session, the financial account registration function:*

362. This limitation is substantially similar to limitation [15.6], and for the same reasons as discussed above, the prior art renders this limitation obvious.

[23.2.6.1] *establishing a secure socket layer connection between the secure server and the payer computing system within the window or frame;*

363. This limitation is substantially similar to limitation [15.6.1], and for the same reasons as discussed above, the prior art renders this limitation obvious.

[23.2.6.2] *outputting instructions to the payer computing system to render a financial account registration request form, within the window or frame, that provides functionality for the payer to provide sensitive financial account information associated with a financial account; and*

364. This limitation is substantially similar to limitation [15.6.2], and for the same reasons as discussed above, the prior art renders this limitation obvious.

[23.2.6.3] *outputting instructions to the payer computing system to encrypt the sensitive financial account information provided by the payer and transmit the encrypted financial account information to the secure server via the secure socket layer connection;*

365. This limitation is substantially similar to limitation [15.6.3], and for the same reasons as discussed above, the prior art renders this limitation obvious.

[23.2.7] *initiate a token retrieval function of the API during the authenticated session, the token retrieval function outputting a non-sensitive electronic data token representing the sensitive financial account information to the merchant server;*

366. This limitation is substantially similar to limitation [15.7], and for the

same reasons as discussed above, the prior art renders this limitation obvious.

[23.2.8] *receive the non-sensitive electronic data token from the secure server, wherein the merchant server does not receive the sensitive financial account information represented by the non-sensitive electronic data token and the secure server does not provide the non-sensitive electronic data token to the payer;*

367. This limitation is substantially similar to limitation [15.8], and for the same reasons as discussed above, the prior art renders this limitation obvious.

[23.2.9] *store the non-sensitive electronic data token in association with the enrollment data identifying the payer; and*

368. This limitation is substantially similar to limitation [15.9], and for the same reasons as discussed above, the prior art renders this limitation obvious.

[23.2.10] *process a payment from the payer by generating a payment transaction instruction, using the token retrieval function of the API, that includes a representation of the non-sensitive electronic data token and instructs the secure server to generate and transmit an electronic request requesting the payment amount from the financial account, obtain the payment amount, and forward at least a portion of the payment amount.*

369. This limitation is substantially similar to limitation [15.10], and for the same reasons as discussed above, the prior art renders this limitation obvious.

11. Claim 24

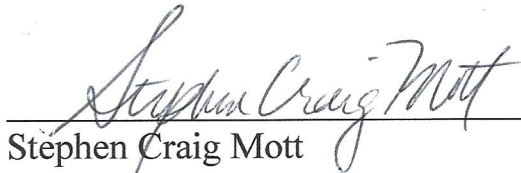
[24.1] *The merchant server of claim 23, wherein the non-sensitive electronic data token constitutes a symbolic representation of the sensitive financial account information.*

370. This limitation is substantially similar to limitation [3.1], and for the same reasons as discussed above, the prior art renders this limitation obvious.

VIII. CONCLUSION

371. This declaration and my opinions herein are made to the best of my knowledge and understanding, and based on the material available to me, at the time of signing this declaration. I declare that all statements made herein on 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 or Title 18 of the United States Code.

Date: 18 JULY 2024



Stephen Craig Mott