US 20070276766A1

(19) **United States**

(12) **Patent Application Publication**

(10) **Pub. No.: US 2007/0276766 A1**

(43) **Pub. Date: Nov. 29, 2007**

Hoyos et al.

(54) **SYSTEM AND METHOD FOR PREVENTING MULTIPLE CHARGES FOR A TRANSACTION IN A PAYMENT SYSTEM**

(76) Inventors: **Carlos Antonio Lorenzo Hoyos**, Morrisville, NC (US); **Marcelo Perazolo**, Cary, NC (US); **Mark E. Peters**, Chapel Hill, NC (US); **Viswanath Srikanth**, Chapel Hill, NC (US); **Andrea Jean Watkins Moryadas**, Brattleboro, VT (US)

Correspondence Address:
IBM CORPORATION - RSW (JVL)
C/O VAN LEEUWEN & VAN LEEUWEN
P.O. BOX 90609
AUSTIN, TX 78709-0609

(21) Appl. No.: **11/456,189**

(22) Filed: **Jul. 8, 2006**

Related U.S. Application Data

(63) Continuation-in-part of application No. 11/420,040, filed on May 24, 2006.

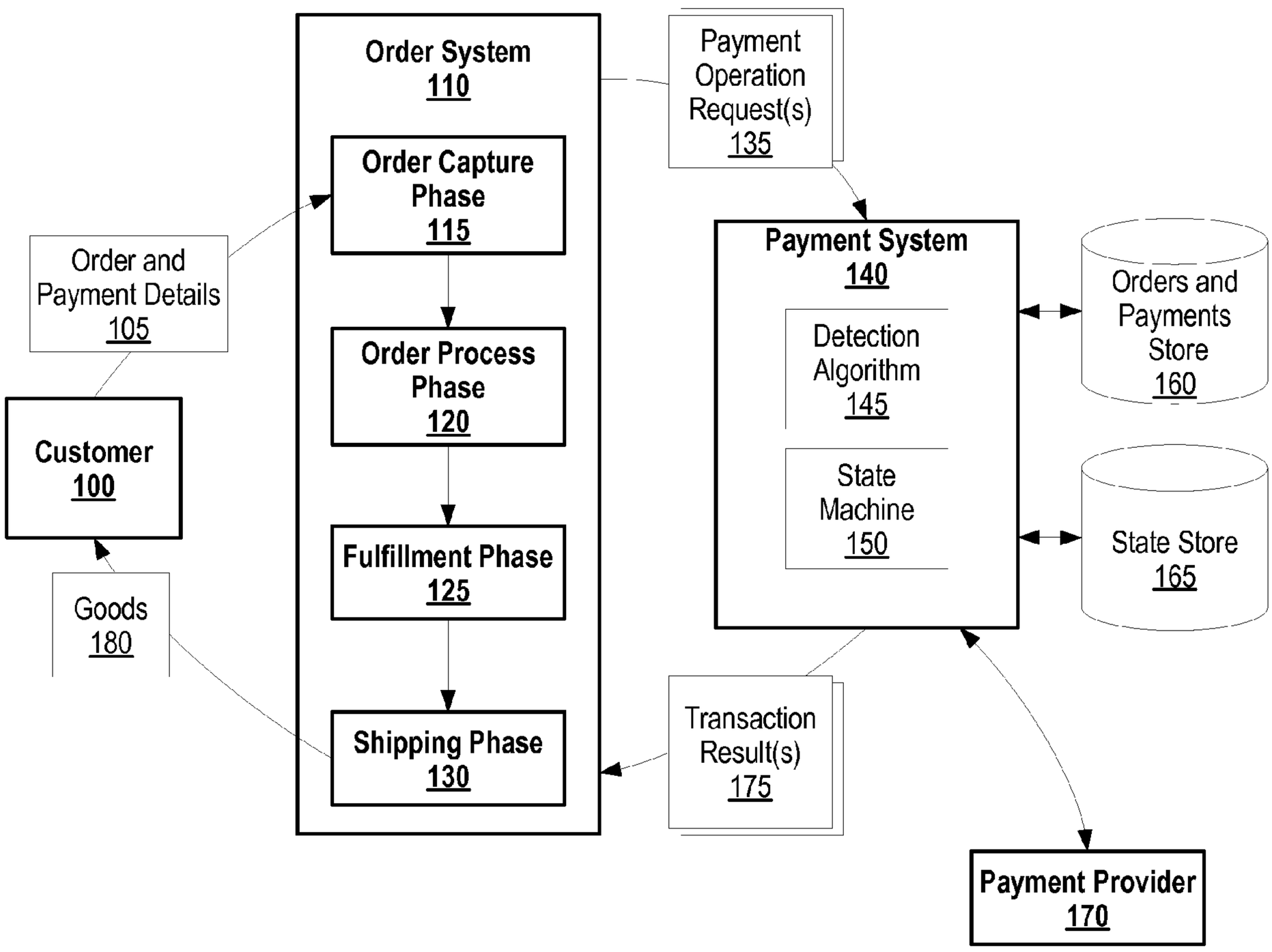
Publication Classification

(51) **Int. Cl.**
G06Q 99/00 (2006.01)

(52) **U.S. Cl.** **705/75**

(57) **ABSTRACT**

A system and method for preventing multiple charges for a transaction in a payment system is presented. A payment system receives a payment operation request from the order system, and determines whether the payment operation is a duplication of a previous payment operation request. If so, the payment system retrieves stored financial transaction results and provides the financial transaction results to the order system. When the payment operation request is not a duplicate, the payment system contacts a payment provider to receive financial transaction results, which is passed to the order system and stored in a persistent data store.



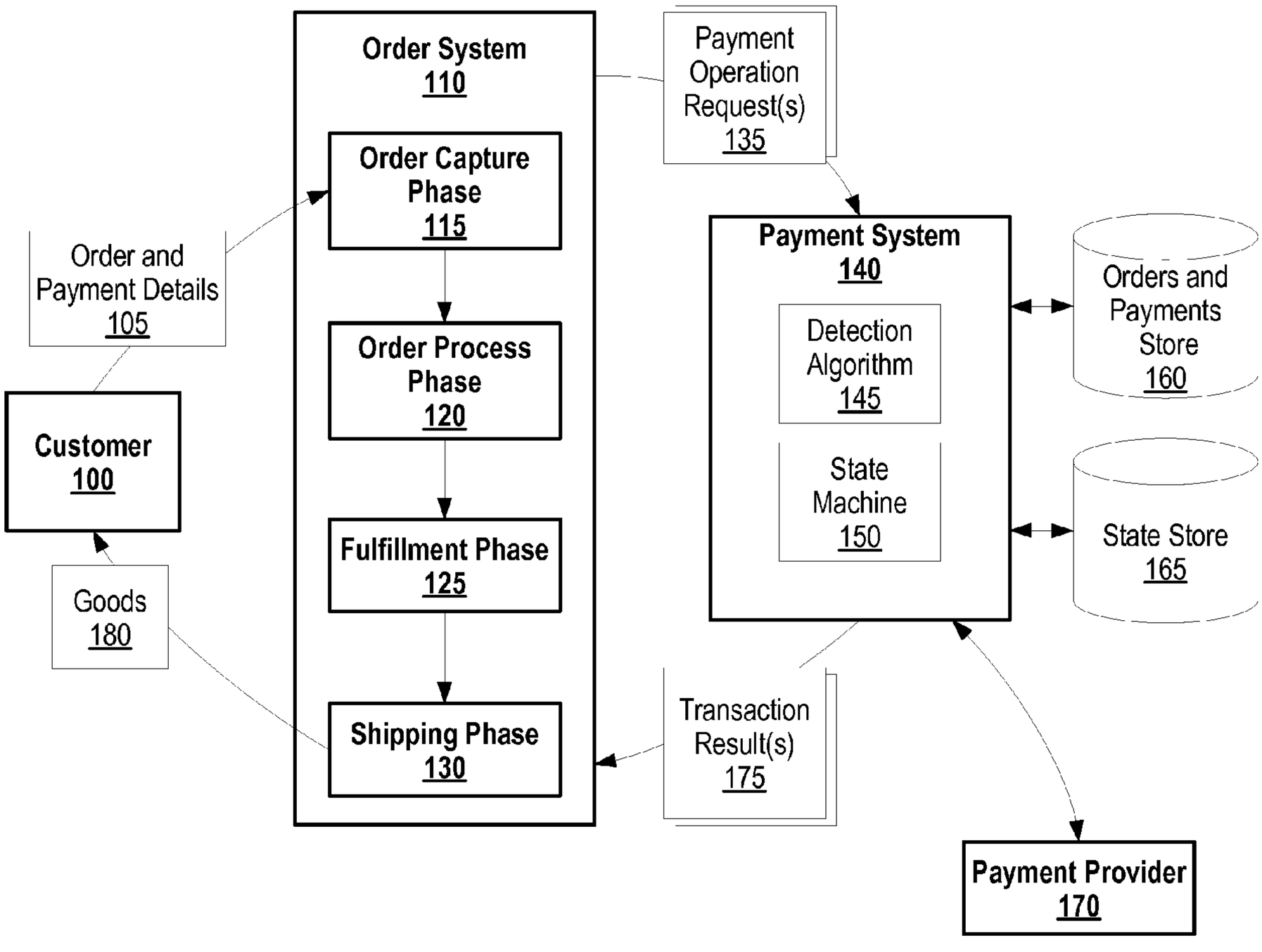

Order System
110
Order Capture Phase
115
Order Process Phase
120
Fulfillment Phase
125
Shipping Phase
130
Payment Operation Request(s)
135
Payment System
140
Detection Algorithm
145
State Machine
150
Orders and Payments Store
160
State Store
165
Order and Payment Details
105
Customer
100
Goods
180
Transaction Result(s)
175
Payment Provider
170

FIG. 1

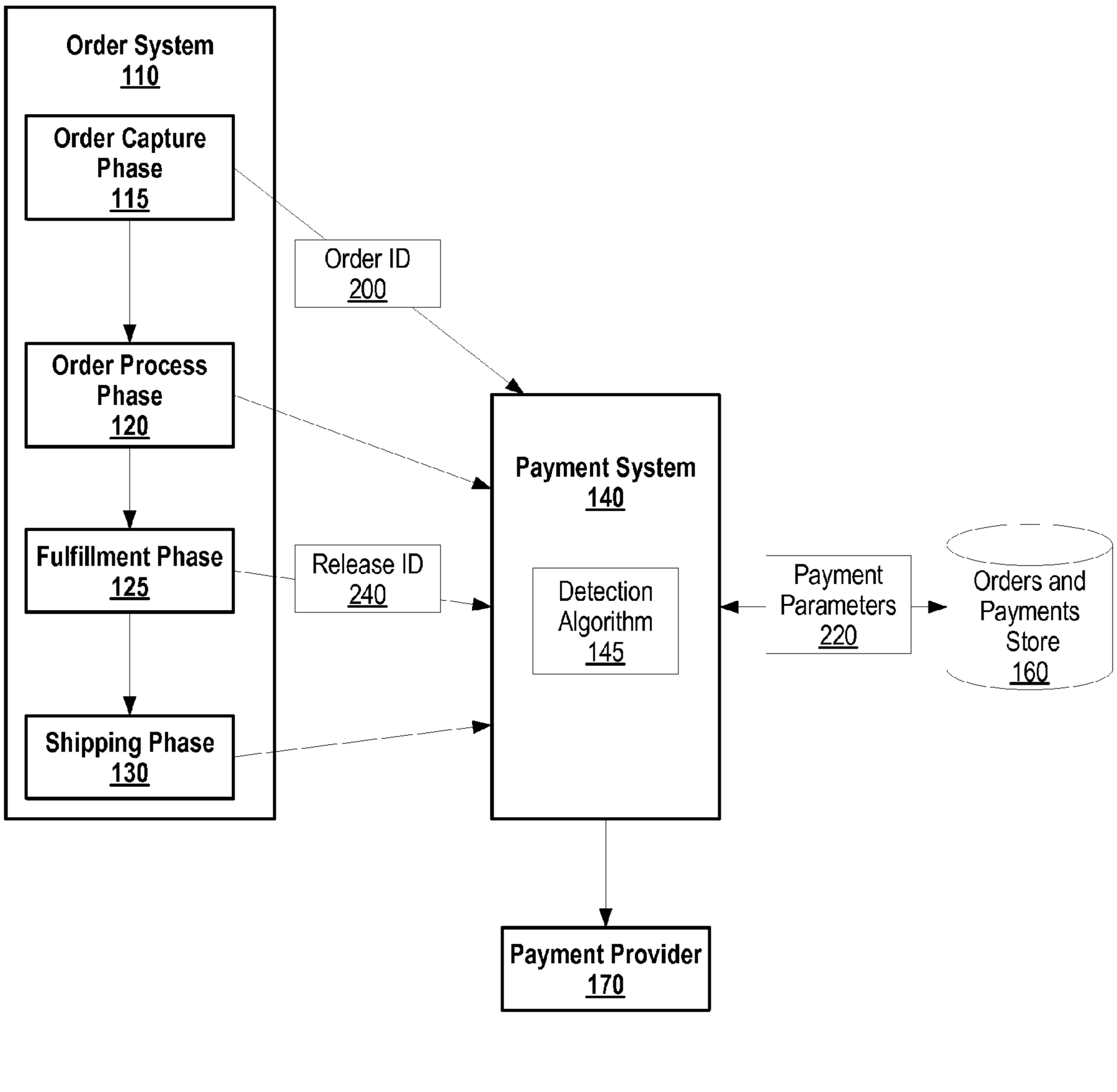

Order System
110
Order Capture Phase
115
Order Process Phase
120
Fulfillment Phase
125
Shipping Phase
130
Order ID
200
Release ID
240
Payment System
140
Detection Algorithm
145
Payment Parameters
220
Orders and Payments Store
160
Payment Provider
170

FIG. 2

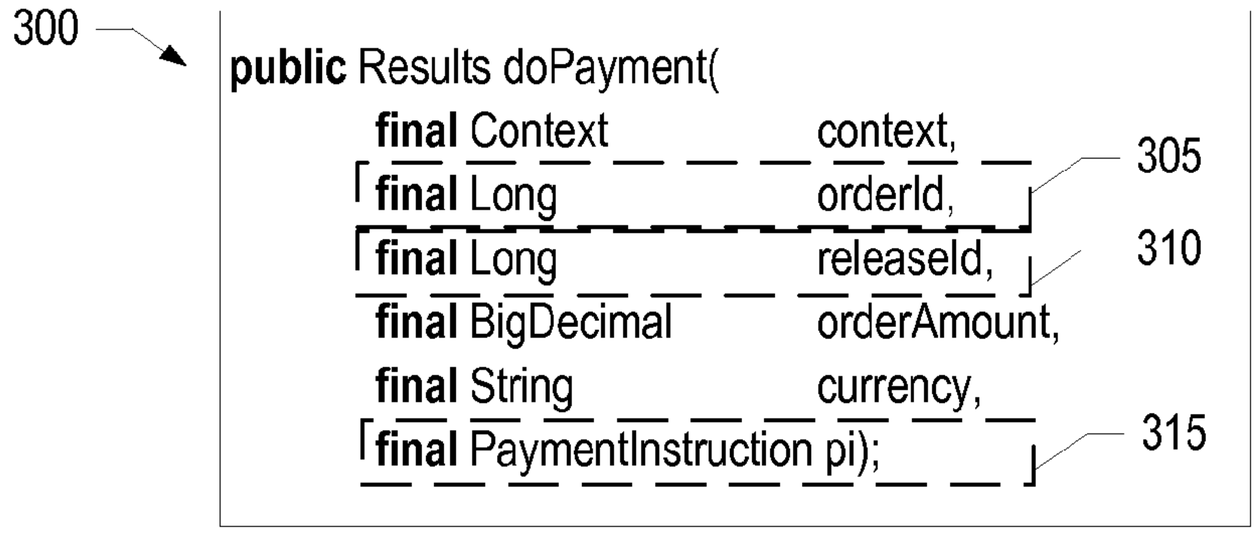


FIG. 3A

320

325 ORDER ID	330 LINE ITEM	335 RELEASE ID	PAYMENT INSTRUCTIONS		
			340 AMOUNT	350 ACCOUNT #	355 ATTRIBUTE
ABCD	1	1111	$60	124895421485	3/08
	2				
	3	1112	$100	165897524655	4/07
	4	--	$210	124895421485	6/08
EFGH	1	1121	$65	6855554655456	12/07

358

FIG. 3B

360

370 AMOUNT	375 ACCOUNT #	380 ATTRIBUTE	385 PHASE	390 ORDER ID	395 RELEASE ID
$38	124895421485	4/09	Process	SFGSGF	7657
$75	165897524655	8/07	Fulfillment	RYRHH	9245
$87	165897524655	4/07	Shipment	GSFGS	7778

398

FIG. 3C

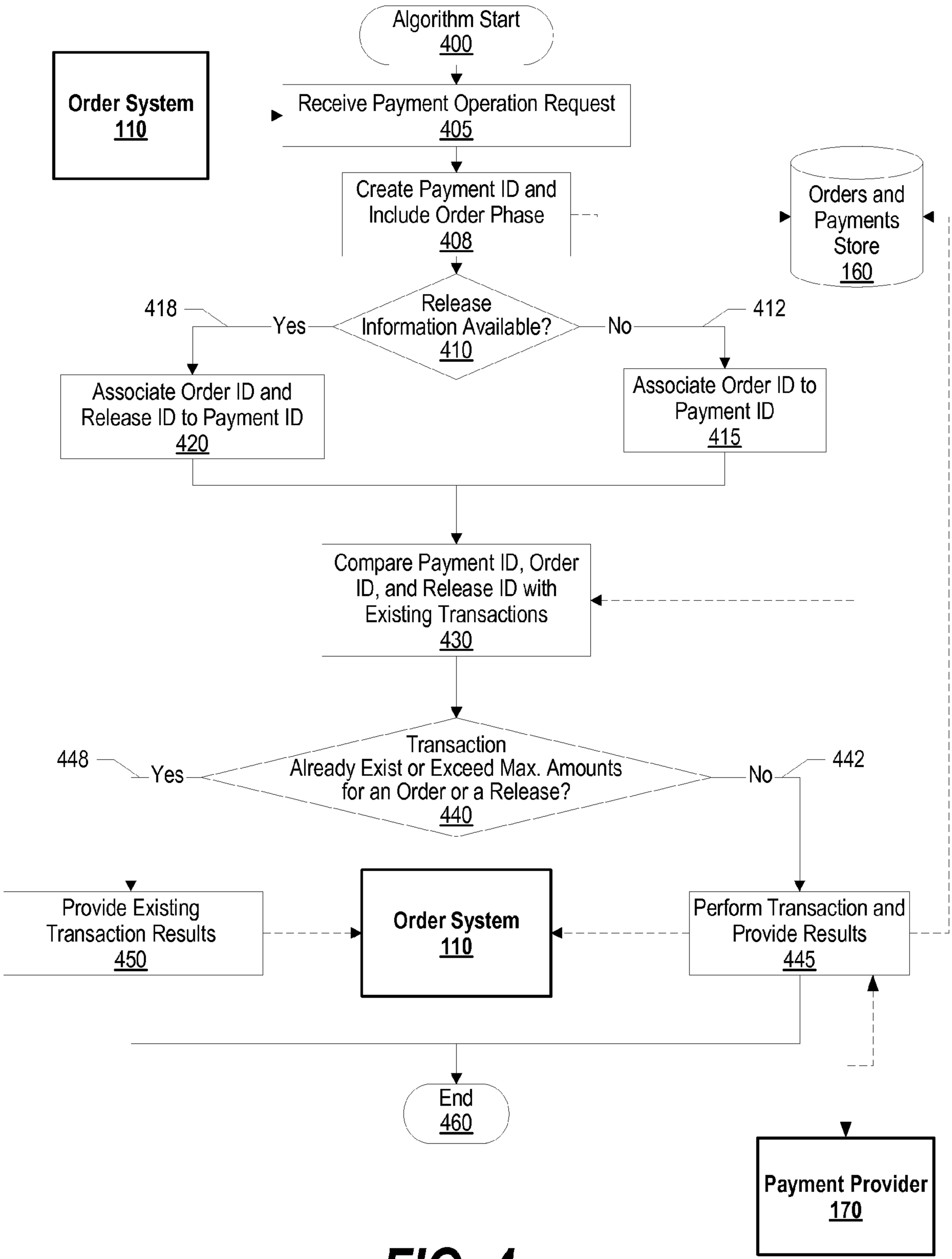

Order System
110
Algorithm Start
400
Receive Payment Operation Request
405
Create Payment ID and Include Order Phase
408
Orders and Payments Store
160
Release Information Available?
410
418
Yes
No
412
Associate Order ID and Release ID to Payment ID
420
Associate Order ID to Payment ID
415
Compare Payment ID, Order ID, and Release ID with Existing Transactions
430
Transaction Already Exist or Exceed Max. Amounts for an Order or a Release?
440
448
Yes
No
442
Provide Existing Transaction Results
450
Order System
110
Perform Transaction and Provide Results
445
End
460
Payment Provider
170

FIG. 4

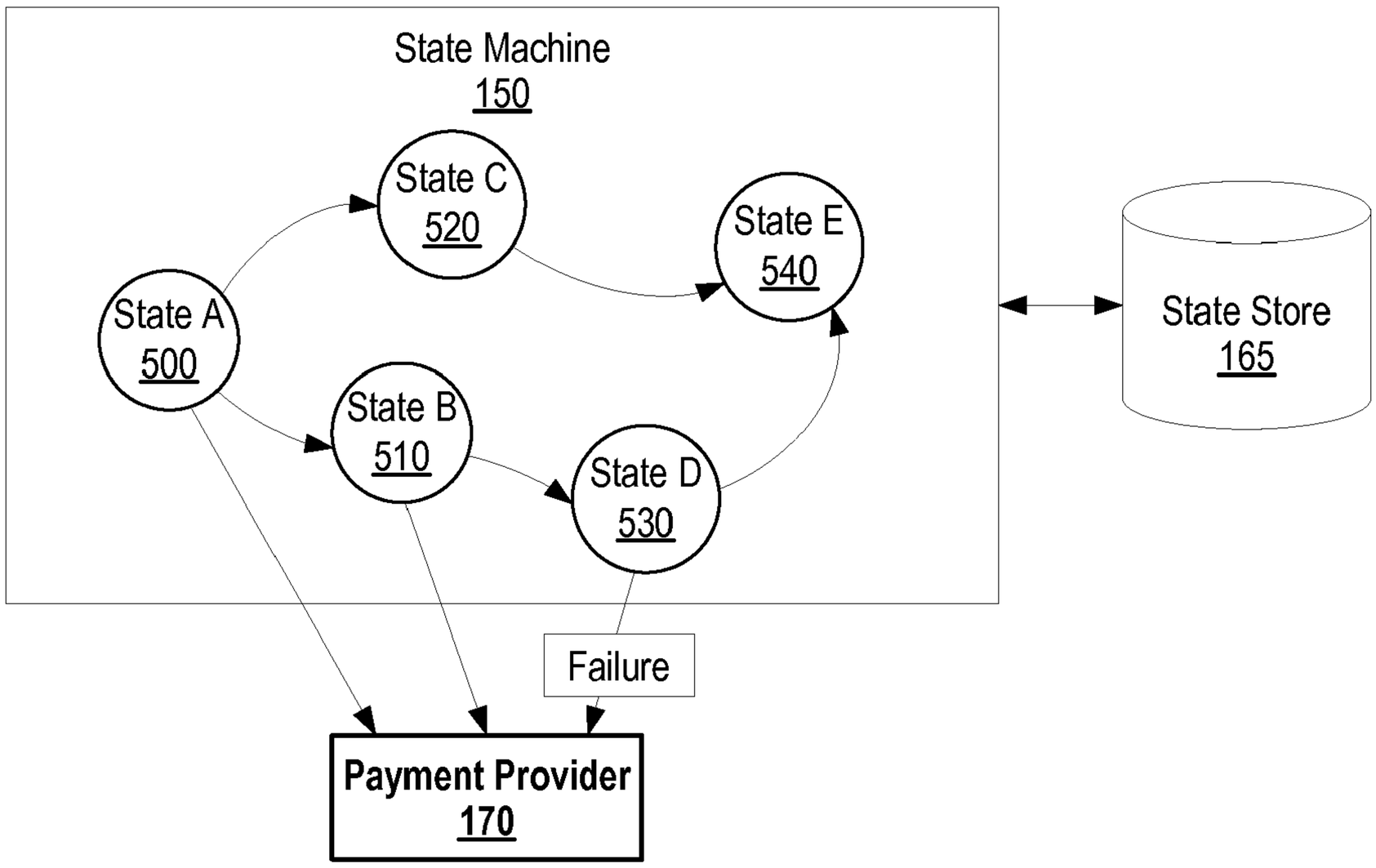

State Machine
150
State C
520
State E
540
State A
500
State B
510
State D
530
State Store
165
Failure
Payment Provider
170

FIG. 5A

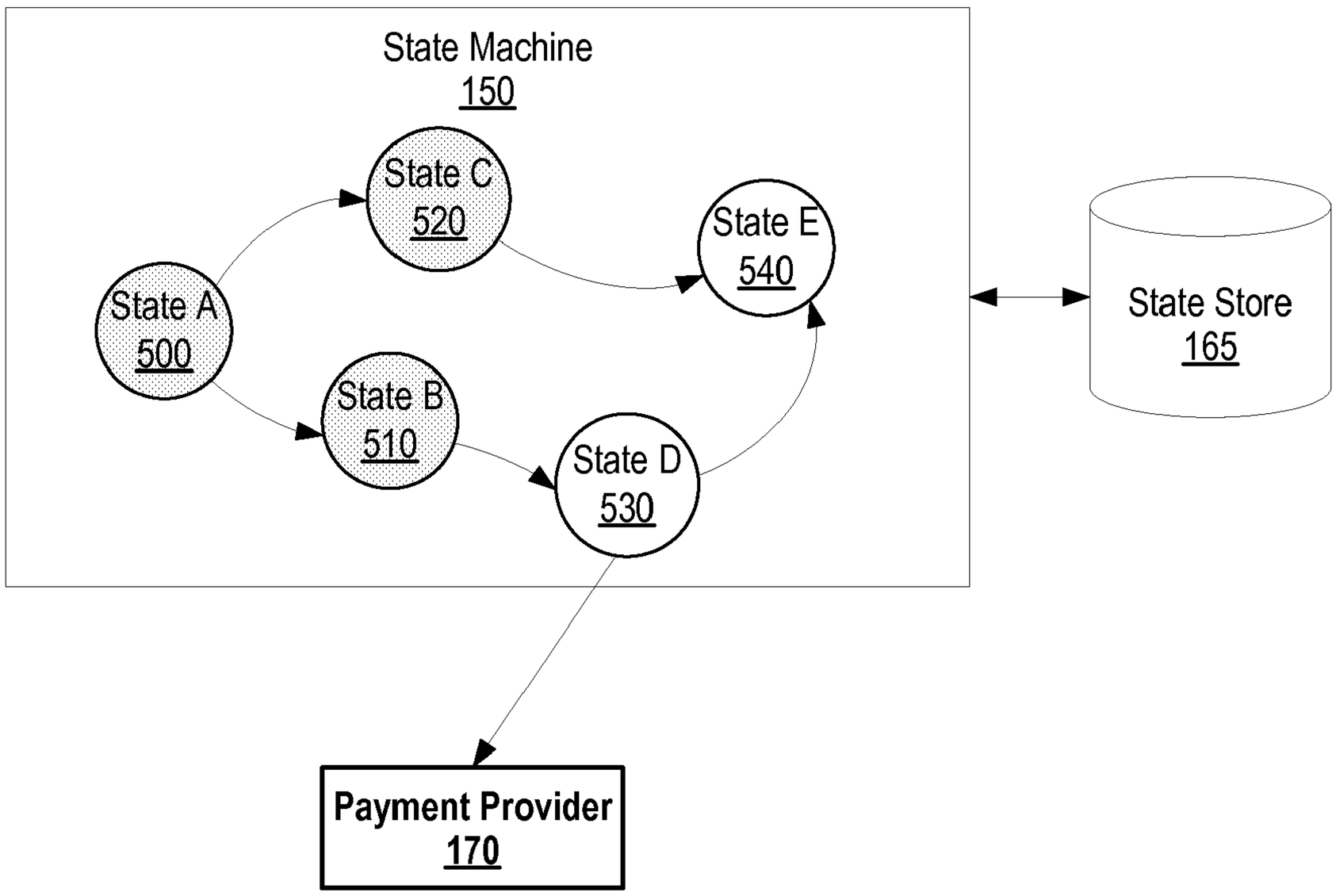

State Machine
150
State C
520
State E
540
State A
500
State B
510
State D
530
State Store
165
Payment Provider
170

FIG. 5B

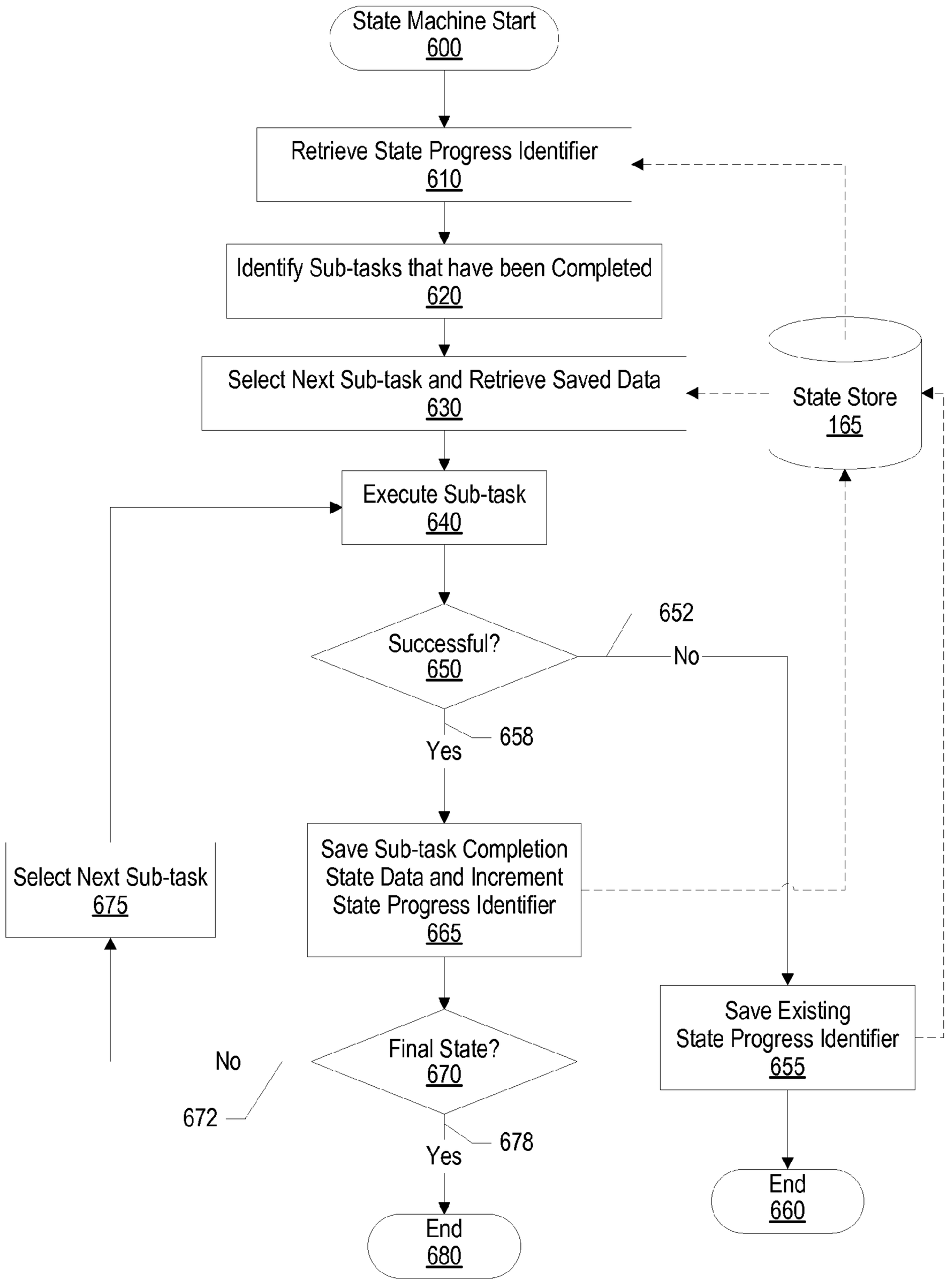

State Machine Start
600
Retrieve State Progress Identifier
610
Identify Sub-tasks that have been Completed
620
Select Next Sub-task and Retrieve Saved Data
630
State Store
165
Execute Sub-task
640
Successful?
650
652
No
Yes
658
Select Next Sub-task
675
Save Sub-task Completion State Data and Increment State Progress Identifier
665
Save Existing State Progress Identifier
655
Final State?
670
No
672
Yes
678
End
680
End
660

FIG. 6

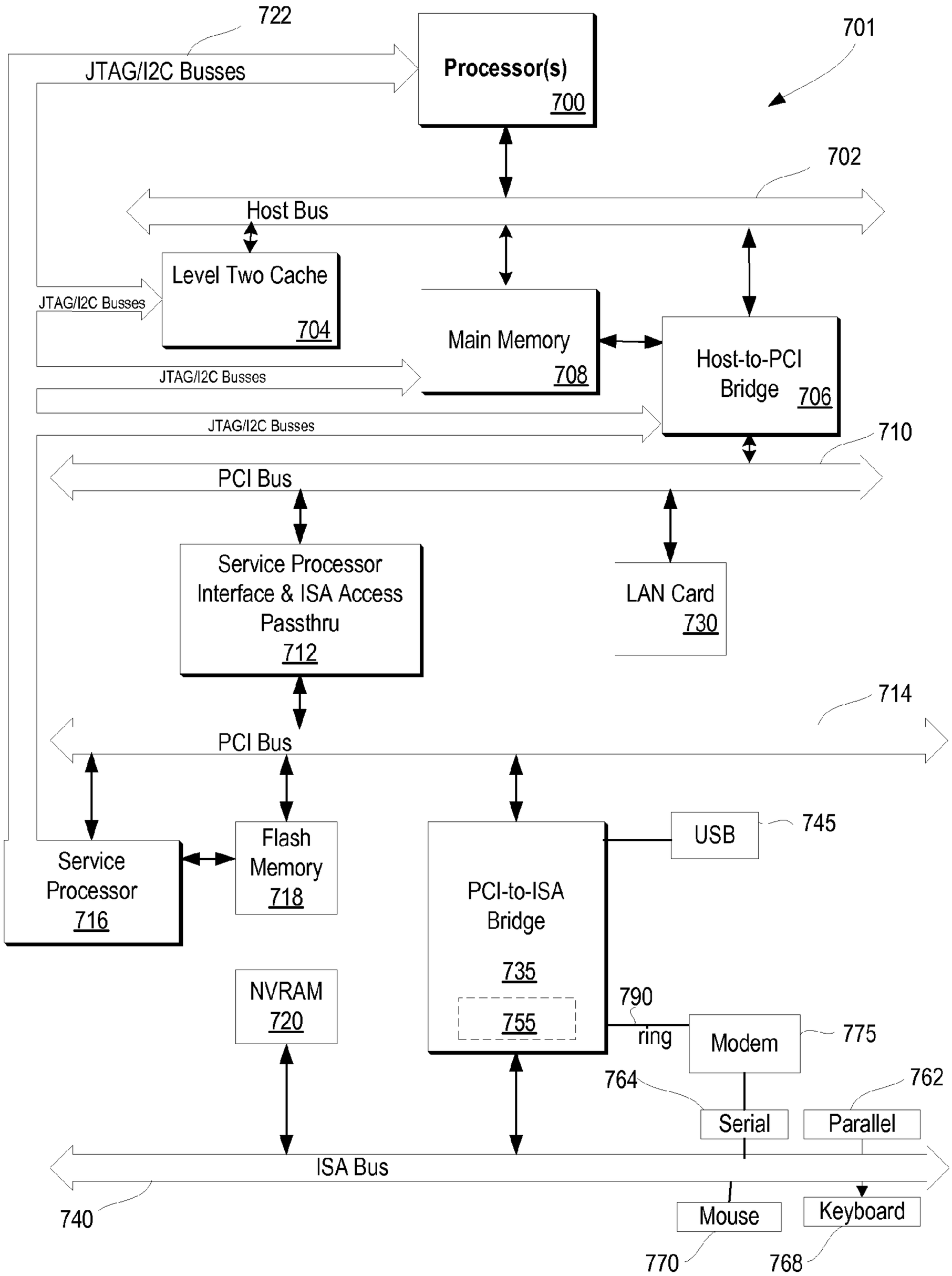

722
JTAG/I2C Busses
Processor(s)
700
701
702
Host Bus
Level Two Cache
704
JTAG/I2C Busses
Main Memory
708
Host-to-PCI
Bridge
706
JTAG/I2C Busses
JTAG/I2C Busses
710
PCI Bus
Service Processor
Interface & ISA Access
Passthru
712
LAN Card
730
714
PCI Bus
Service
Processor
716
Flash
Memory
718
PCI-to-ISA
Bridge
735
755
USB
745
NVRAM
720
790
ring
Modem
775
764
762
Serial
Parallel
ISA Bus
740
Mouse
770
Keyboard
768

FIG. 7

SYSTEM AND METHOD FOR PREVENTING MULTIPLE CHARGES FOR A TRANSACTION IN A PAYMENT SYSTEM

CROSS REFERENCE TO RELATED APPLICATION

[0001] This application is a continuation-in-part of commonly assigned, co-pending U.S. Non-Provisional patent application Ser. No. 11/420,040, entitled "System and Method for State-Based Execution and Recovery in a Payment System," filing date May 24, 2006, Attorney Docket No. RSW920050208US1.

BACKGROUND OF THE INVENTION

[0002] 1. Technical Field

[0003] The present invention relates to a system and method for preventing multiple charges for a transaction in a payment system. More particularly, the present invention relates to a system and method for providing existing financial transaction results to an order system when the order system invokes a duplicate payment operation request.

[0004] 2. Description of the Related Art

[0005] Software-based payment systems today rely upon common techniques to orchestrate financial transactions between external payment providers (e.g., credit card companies). A challenge found with these techniques, however, is that the software-based payment systems may not check for duplicate financial transaction requests. Even though payment systems may implement transaction-based techniques to minimize the possibility of such situations, problems arise when duplicate requests originate external to the payment system, such as from order systems that are connected to the payment system.

[0006] In such cases when a payment operation request is driven from a connected external component, the external component may not provide checks or transactional control for duplicate payment operation requests to the payment system. Hence, the payment system sends duplicate financial transaction requests to the external payment provider. For example, an order system may request a payment system to capture a deposit of $50 twice when the order system's original intent was for the deposit to be captured once.

[0007] Furthermore, a challenge found with double charging is that the external payment provider may not allow the payment system to retract an executed financial transaction request. Meaning, the payment provider may not allow executed financial transaction request "rollbacks" or cancellations. Therefore, even if the payment system discovers a double charge, the payment system has to manually send a new financial transaction request to the payment provider to credit the customer's account.

[0008] While specific solutions may be proposed for solving duplicate payment operation requests that are initiated by an end user (e.g., web-form double-submission, invoice double submission), a challenge found is that a payment system may also receive duplicate payment operation requests from order systems that are not directly initiated by an end user.

[0009] What is needed, therefore, is a system and method that prevents a payment system from performing duplicate financial transaction requests to a payment provider when the payment system receives a duplicate payment operation request from an order system.

SUMMARY

[0010] It has been discovered that the aforementioned challenges are resolved using a system and method for providing existing financial transaction results to an order system when the order system invokes a duplicate payment operation request. A payment system receives a payment operation request from the order system, and determines whether the payment operation is a duplication of a previous payment operation request. If so, the payment system retrieves stored financial transaction results and provides the financial transaction results to the order system. When the payment operation request is not a duplicate, the payment system contacts a payment provider to receive financial transaction results, which is passed to the order system and stored in a persistent data store.

[0011] A customer places an order with an order system by sending order and payment details to the order system. For example, the customer may place an order on a web page for office supplies, in which case the order and payment details may include line item information for the office supplies along with credit card information to pay for the office supplies.

[0012] While processing the customer's order, the order system generates an order identifier, which uniquely identifies the customer's order. The order system also generates a release identifier, which uniquely identifies all or part of the customer's order that plans to ship at the same time to the customer (e.g., a "package"). The order system sends the order identifier, the release identifier, and one or more payment operation requests to the payment system. For example, the payment operation requests may be a payment instruction validation request, a payment instruction storage request, a process payment instruction after allocation request, or a process payment instruction after shipment request.

[0013] Some payment operation requests involve the payment system sending a "financial transaction request" to a payment provider, which may be an external payment provider (e.g., credit card company). When this occurs, the payment provider sends "financial transaction results" back to the payment system. When received, the payment system sends the financial transaction results to the order system, and also stores the financial transaction results in a persistent data store.

[0014] In order to identify duplicate payment operation requests generated by the order system, the payment system uses a detection algorithm. The detection algorithm detects duplicate payment operation requests using the combination of the order identifier, the release identifier (if applicable), and the payment identifier.

[0015] When the detection algorithm detects a duplicate payment operation request, the detection algorithm does not send a financial transaction request to the payment provider. Instead, the detection algorithm instructs the payment provider to retrieve the financial transaction results previously stored in the persistent data store, and send the financial transaction results to the order system. By detecting duplicate payment operation requests, the payment system alleviates duplicate requests to the payment provider that, in turn, reduces cost and eliminates rollback situations.

[0016] The foregoing is a summary and thus contains, by necessity, simplifications, generalizations, and omissions of detail; consequently, those skilled in the art will appreciate that the summary is illustrative only and is not intended to be in any way limiting. Other aspects, inventive features, and advantages of the present invention, as defined solely by the claims, will become apparent in the non-limiting detailed description set forth below.

BRIEF DESCRIPTION OF THE DRAWINGS

[0017] The present invention may be better understood, and its numerous objects, features, and advantages made apparent to those skilled in the art by referencing the accompanying drawings.

[0018] FIG. **1** is a diagram showing a payment system receiving and processing payment operation requests from an order system;

[0019] FIG. **2** is a diagram showing a payment system receiving payment parameters from an order system, and using the payment parameters to identify duplicate payment operation requests;

[0020] FIG. **3**A is a diagram showing interface information for a payment system that permits external components to request payment operations;

[0021] FIG. **3**B is a table showing payment parameters for customer orders;

[0022] FIG. **3**C is a table showing a list of payment identifiers, order identifiers, and release identifiers that correspond to payment operation requests;

[0023] FIG. **4** is a flowchart showing steps taken in determining whether to send a financial transaction request to a payment provider;

[0024] FIG. **5**A is a diagram showing a state machine partially completing a payment operation that includes multiple sub-tasks;

[0025] FIG. **5**B is a diagram showing a state machine completing a partially completed payment operation;

[0026] FIG. **6** is a flowchart showing steps taken in completing a payment operation on a sub-task-by-sub-task basis; and

[0027] FIG. **7** is a block diagram of a computing device capable of implementing the present invention.

DETAILED DESCRIPTION

[0028] The following is intended to provide a detailed description of an example of the invention and should not be taken to be limiting of the invention itself. Rather, any number of variations may fall within the scope of the invention, which is defined in the claims following the description.

[0029] FIG. **1** is a diagram showing a payment system receiving and processing payment operation requests from an order system. Customer **100** places an order with order system **110** by sending order and payment details **105**. For example, customer **100** may place an order on a web page for office supplies, in which case order and payment details **105** may include line item information for the office supplies along with credit card information to pay for the office supplies.

[0030] The embodiment shown in FIG. **1** shows that order system **110** includes four "order phases" to process customer **100**'s order, which are order capture phase **115**, order process phase **120**, fulfillment phase **125**, and shipping phase **130**. Order capture phase **115** receives order and payment details **105** from customer **100**. Order process phase **120** processes the order once customer **100** completes the order (e.g., checks out). Fulfillment phase **125** allocates goods to all or a part of customer **100**'s order and, shipping phase **130** ships the goods (goods **180**) to customer **100**. As one skilled in the art can appreciate, an order system may have more or less order phases that what is shown in FIG. **1**.

[0031] While processing customer **100**'s order, order system **110** generates an order identifier, which uniquely identifies customer **100**'s order. Order system **110** also generates a release identifier, which uniquely identifies all or part of customer **100**'s order that is ready to release for shipment. Order system **110** sends the order identifier, the release identifier, and one or more payment operation requests **135** to payment system **140**. Payment operation requests **135** may include:

[0032] Payment Instruction Validation Request (during order capture phase **115**);

[0033] Payment Instruction Storage Request (during order process phase **120**);

[0034] Process Payment Instruction After Allocation Request (during fulfillment phase **125**); and

[0035] Process Payment Instruction After Shipment Request (during shipping phase **130**).

[0036] Some payment operation requests **135** involve payment system **140** sending a "financial transaction request" to payment provider **170**, which may be an external payment provider (e.g., credit card company). In turn, payment provider **170** sends a "financial transaction result" back to payment system **140**. When received, payment system **140** sends the financial transaction results (transaction results **175**) to order system **110**, and also stores the financial transaction results in orders and payments store **160**.

[0037] In many cases, order system **110** may unintentionally send a duplicate payment operation request **135** to payment system **140**. Payment system **140** uses detection algorithm **145** to circumvent interaction with payment provider **170** when it receives a duplicate payment operation request. Detection algorithm **145** detects duplicate payment operation requests using the order identifier, the release identifier (if applicable), and a payment identifier, which includes payment instructions and an order phase (see FIG. **3**C and corresponding text for further details).

[0038] When detection algorithm **145** detects a duplicate payment operation request, detection algorithm **145** does not send a financial transaction request to payment provider **170**. Instead, detection algorithm **145** instructs payment provider **140** to retrieve the financial transaction results previously stored in orders and payments store **160**, and send the financial transaction results (transaction results **175**) to order system **110**. By detecting duplicate payment operation requests, payment system **140** alleviates duplicate requests to payment provider **170** that, in turn, reduces cost and eliminates rollback situations (see FIGS. **2-4** and corresponding text for further details).

[0039] In addition to receiving duplicate payment operation requests, events may occur at payment system **140** that prevent a payment operation from completing. For example, payment provider **140** may have completed two out of six "sub-tasks" that comprise a particular payment operation, and a failure occurs during the third sub-task. In these

circumstances, payment system **140** uses state machine **150** in order to prevent duplication of the two successfully completed sub-tasks.

[0040] State machine **150** tracks successfully completed sub-tasks, and stores completion state data and a state progress identifier in state store **165**. As a result, when payment system **140** re-initiates a failed payment operation, payment system **140** is able to retrieve the state progress identifier and completion state data from state store **165**, and continue processing the payment operation at the previously failed sub-task point instead of starting at the beginning of the payment operation. Using the example discussed above, payment system **140** re-initiates the payment operation at the third sub-task, which eliminates duplicating the first and second sub-tasks (see FIGS. **5-7** and corresponding text for further details).

[0041] FIG. **2** is a diagram showing a payment system receiving payment parameters from an order system, and using the payment parameters to identify duplicate payment operation requests. As order system **110** receives and processes a customer order, order system **110** generates order identifier **200** and release identifier **240**. During order capture phase **115**, order system **110** generates and sends order identifier **200** to payment system **140**, which identifies a customer's order, such as "#009793." Payment system **140** stores order identifier **200** as a payment parameter (payment parameters **220**) in orders and payments store **160**.

[0042] As order system **110** proceeds through order process phase **120** and reaches fulfillment phase **125**, order system **110** generates and sends release identifier **240** to payment system **140**. Release identifier **240** identifies particular line items of an order that are available and allocated for release. Again, payment system **140** stores release identifier **240** as a payment parameter in orders and payments store **160**. In turn, order system **110** proceeds to shipping phase **130** and ships goods to the customer.

[0043] When payment system **140** receives a payment operation request from order system **110**, payment system **140** retrieves payment parameters **220** from orders and payments store **160**. In addition, payment system **140** generates a payment identifier to correspond with the payment operation request (see FIG. **4** and corresponding text for further details). Detection algorithm **145** uses the payment identifier, the order identifier, and the release identifier (if applicable) to determine whether the payment operation request is a duplication of a previously received payment operation request.

[0044] When the payment operation request is a duplication, payment system **140** retrieves already stored financial transaction results from orders and payments store **160**, and provides the financial transaction results to order system **110**. When the payment operation request is not a duplicate, payment system **140** sends a financial transaction request to payment provider **170**. In turn, payment provider **170** sends financial transaction results to payment system **140**, which payment system **140** sends to order system **110** and also stores in orders and payments store **160** (see FIG. **4** and corresponding text for further details). Order system **110**, order capture phase **115**, order process phase, **120**, fulfillment phase **125**, shipping phase **130**, payment system **140**, detection algorithm **145**, orders and payments store **160**, and payment provider **170** are the same as that shown in FIG. **1**.

[0045] FIG. **3**A is a diagram showing interface information for a payment system that permits external components to request payment operations. An external component, such as order system **110**, uses interface **300** to send a payment operation request to a payment system (e.g., payment system **140**). The invention described herein adds lines **305** and **310** to a typical interface, which include order identifier information and release identifier information, respectively.

[0046] When an order system sends a payment operation request to a payment system, the order system provides an order identifier to the payment system in line **305**, which is a unique system-wide identifier that is assigned to a customer's order when the customer places the order. When the order system releases all or part of an order, the order system assigns a release identifier, which is included in line **310**. The release identifier uniquely identifies a set of products in an order that are shipped together, such as a "package" that includes goods, which are shipped to a customer.

[0047] Line **315** includes payment instruction information for an order, which may include an account number, an account expiration date, an address, and a customer name. The payment system includes the payment instruction information in a payment identifier, in which the payment system uses to identify duplicate payment operation requests (see FIGS. **3**C, **4**, and corresponding text for further details).

[0048] FIG. **3**B is a table showing payment parameters for customer orders. Table **320** includes columns **325** through **355**. Column **325** includes a list of order identifiers associated with customer orders. Column **330** includes a list of line items for particular orders (order identifiers). For example, the first order identifier in table **320** includes four line items. Column **335** includes a list of release identifiers that correspond to one or more line items. The release identifiers are assigned to an order when their associated goods are available for release. Each release identifier may be associated with a particular "package" that is shipped to a customer.

[0049] Columns **340-355** include payment instructions for an order. A customer may provide multiple payment instructions for a single order, such as charging part of an order to one credit card, and charging the remaining part of the order to another credit card. Column **340** includes a list of amounts for particular release identifiers. For example, row **358** includes an amount of $60 for release identifier 1111, which includes line items 1 and 2 of order ABCD.

[0050] Column **350** includes a list of account numbers, such as credit card numbers, to pay for particular line items. And, column **355** includes a list of attributes for the payment instructions, such as expiration dates or card verification numbers.

[0051] FIG. **3**C is a table showing a list of payment identifiers, order identifiers, and release identifiers that correspond to payment operation requests. When a payment system receives a payment operation request, the payment system logs information in table **360**. As a result, the payment system is able to analyze table **360** when it receives subsequent payment operation requests and determine whether the payment operation request is a duplication.

[0052] Table **360** includes columns **370-395**. Columns **370-380** include a list of payment instructions for corresponding orders (see FIG. **3**B and corresponding text for further details). Column **385** includes a list of order phases that the payment system receives a payment operation request. For example, the payment system generated row **398** when it received a payment operation request from an order system when the order system was in the fulfillment phase. Columns **390** and **395** include a list of order identi-

fiers and release identifiers, respectively, that the payment system uses during the detection of duplicate payment operation requests (see FIG. **4** and corresponding text for further details).

[0053] FIG. **4** is a flowchart showing steps taken in determining whether to send a financial transaction request to a payment provider. A payment system uses a detection algorithm to determine when an order system sends a duplicate payment operation request. Processing commences at **400**, whereupon processing receives a payment operation request from order system **110** at step **405**. The payment operation request includes payment parameters, such as an order identifier and a release identifier. At step **408**, processing creates a payment identifier that includes payment instructions as well as an "order phase." The order phase is an order system's phase at which the payment operation is sent. Processing stores the created payment identifier in orders and payments store **165**. Order system **110** and orders and payments store **160** are the same as that shown in FIG. **1**.

[0054] A determination is made as to whether a release identifier is available that corresponds to the payment operation request (decision **410**). For example, the order system may have provided a release identifier that signifies that particular line items are available for release to fulfillment. If a release identifier is not available, decision **410** branches to "No" branch **412** whereupon processing associates the order identifier with the payment identifier at step **415**. On the other hand, if the release identifier is available, decision **410** branches to "Yes" branch **418** whereupon processing associates the payment identifier with the order identifier and the release identifier.

[0055] At step **430**, processing compares the payment identifier, the order identifier, and the release identifier (if applicable) to stored transaction information in orders and payments store **160**. A determination is made as to whether financial transaction results already exist for the particular identifier combination, signifying that the payment operation is a duplicate (decision **440**). Processing also checks for whether the payment operation request exceeds a maximum amount for an order or a release.

[0056] If the identifier combination does not already exist and the payment request does not exceed a maximum amount, decision **440** branches to "No" branch **442** whereupon processing sends a financial transaction request to payment provider **170** that, as a result, provides financial transaction results. These results are then sent to order system **110** and stored in orders and payments store **165** (step **445**). Payment provider **170** is the same as that shown in FIG. **1**.

[0057] On the other hand, if a transaction already exits for the payment request, or the payment request is requesting an amount that exceeds a limit, decision **440** branches to "Yes" branch **448** whereupon processing provides order system **110** with the existing financial transaction results, and does not interact with payment provider **170** (step **450**). Processing ends at **460**.

[0058] FIG. **5**A is a diagram showing a state machine partially completing a payment operation that includes multiple sub-tasks. State machine **150** includes five states that correspond to five sub-tasks, which are state A **500**, state B **510**, state C **520**, state D **530**, and state E **540**. Examples of sub-tasks include:

[0059] Retrieve payment information and save to a persistent data store.

[0060] Retrieve sets of payment configurations and policies that determine what financial transactions to perform in order to process a determined event.

[0061] Establish communications with an external payment provider.

[0062] Perform financial transactions by communicating with the external payment provider.

[0063] Receive financial transaction results from the external payment provider, or querying the external payment provider in order to determine such results.

[0064] Process results and save them into a persistent data store.

[0065] Provide appropriate response and make available as the result of a task.

[0066] FIG. **5**A shows that state machine **150** proceeded through states **500**, **510**, and **520**, in which case interaction with payment provider **170** occurred at states **500** and **510**. At each state, state machine **150** logs the completion of a sub-task and stores completion state data in state store **165** (see FIG. **6** and corresponding text for further details). State store **165** is the same as that shown in FIG. **1**.

[0067] While proceeding to state D **530**, a failure occurred, which prevents state machine **150** from completing a payment operation. Since state machine **150** logged sub-task completion state data up to state C **520**, state machine **150** is able to resume sub-task processing at state D **530** (see FIG. **5**B and corresponding text for further details).

[0068] FIG. **5**B is a diagram showing a state machine completing a partially completed payment operation. State machine **150** previously completed sub-tasks that resulted in the payment operation reaching state C **520**, whose completion state data is stored in state store **165** (see FIG. **5**A and corresponding text for further details). As such, state machine **150** retrieves the completion state data for state C **520** and resumes payment operation processing. Subsequently, state machine **150** completes the payment operation by proceeding through state D **530** and state E **540**. As can be seen, state machine **150** interacts with payment provider **170** at state D **530**, but does not duplicate interaction with payment provider **170** at state A **500** and state B **510** as shown in FIG. **5**A.

[0069] FIG. **6** is a flowchart showing steps taken in completing a payment operation on a sub-task-by-sub-task basis. Processing commences at **600**, whereupon processing retrieves a state progress identifier corresponding to a partially completed payment operation from state store **165** (step **610**). At step **620**, processing uses the state progress identifier to identify subtasks that have already been completed. For example, the state progress identifier may be "4," which signifies that the first four sub-tasks of a payment operation completed successfully. State store **165** is the same as that shown in FIG. **1**.

[0070] At step **630**, processing selects the next sub-task, which is the sub-task following the last completed sub-task, and retrieves completion state data from state store **165**. Using the example described above, processing retrieves the fourth sub-tasks completion state data, and selects the fifth sub-task to execute next. At step **640**, processing executes the next sub-task.

[0071] A determination is made as to whether the sub-task executed successfully (decision **650**). If the sub-task did not

execute successfully, decision **650** branches to "No" branch **652** whereupon processing saves the same state progress identifier in state store **165**, and processing ends at **660**.

[0072] On the other hand, if the sub-task's execution was successful, decision **650** branches to "Yes" branch **658** whereupon processing saves the sub-task's completion state data and increments the state progress identifier in state store **165** (step **665**). A determination is made as to whether processing reached the final state of the payment operation (decision **670**). If processing has not reached the final state of the payment operation, decision **670** branches to "No" branch **672** whereupon processing selects (step **675**) and processes the next sub-task. This looping continues until processing reaches the payment operation's final state, at which point decision **670** branches to "Yes" branch **678** whereupon processing ends at **680**.

[0073] In one embodiment, processing may identify a "best path" to complete a payment operation based upon customer payment policies, such as creating new payment transactions and canceling older payment transactions. In this embodiment, processing may calculate the differences of transaction amounts still pending, and reuse validated, but partially completed, payment instruction transactions in order to provide a better chance that the transaction is successful.

[0074] FIG. **7** illustrates information handling system **701**, which is a simplified example of a computer system capable of performing the computing operations described herein. Computer system **701** includes processor **700** which is coupled to host bus **702**. A level two (L2) cache memory **704** is also coupled to host bus **702**. Host-to-PCI bridge **706** is coupled to main memory **708**, includes cache memory and main memory control functions, and provides bus control to handle transfers among PCI bus **710**, processor **700**, L2 cache **704**, main memory **708**, and host bus **702**. Main memory **708** is coupled to Host-to-PCI bridge **706** as well as host bus **702**. Devices used solely by host processor(s) **700**, such as LAN card **730**, are coupled to PCI bus **710**. Service Processor Interface and ISA Access Pass-through **712** provides an interface between PCI bus **710** and PCI bus **714**. In this manner, PCI bus **714** is insulated from PCI bus **710**. Devices, such as flash memory **718**, are coupled to PCI bus **714**. In one implementation, flash memory **718** includes BIOS code that incorporates the necessary processor executable code for a variety of low-level system functions and system boot functions.

[0075] PCI bus **714** provides an interface for a variety of devices that are shared by host processor(s) **700** and Service Processor **716** including, for example, flash memory **718**. PCI-to-ISA bridge **735** provides bus control to handle transfers between PCI bus **714** and ISA bus **740**, universal serial bus (USB) functionality **745**, power management functionality **755**, and can include other functional elements not shown, such as a real-time clock (RTC), DMA control, interrupt support, and system management bus support. Nonvolatile RAM **720** is attached to ISA Bus **740**. Service Processor **716** includes JTAG and I2C busses **722** for communication with processor(s) **700** during initialization steps. JTAG/I2C busses **722** are also coupled to L2 cache **704**, Host-to-PCI bridge **706**, and main memory **708** providing a communications path between the processor, the Service Processor, the L2 cache, the Host-to-PCI bridge, and the main memory. Service Processor **716** also has access to system power resources for powering down information handling device **701**.

[0076] Peripheral devices and input/output (I/O) devices can be attached to various interfaces (e.g., parallel interface **762**, serial interface **764**, keyboard interface **768**, and mouse interface **770** coupled to ISA bus **740**. Alternatively, many I/O devices can be accommodated by a super I/O controller (not shown) attached to ISA bus **740**.

[0077] In order to attach computer system **701** to another computer system to copy files over a network, LAN card **730** is coupled to PCI bus **710**. Similarly, to connect computer system **701** to an ISP to connect to the Internet using a telephone line connection, modem **775** is connected to serial port **764** and PCI-to-ISA Bridge **735**.

[0078] While FIG. **7** shows one information handling system that employs processor(s) **700**, the information handling system may take many forms. For example, information handling system **701** may take the form of a desktop, server, portable, laptop, notebook, or other form factor computer or data processing system. Information handling system **701** may also take other form factors such as a personal digital assistant (PDA), a gaming device, ATM machine, a portable telephone device, a communication device or other devices that include a processor and memory.

[0079] One of the preferred implementations of the invention is a client application, namely, a set of instructions (program code) in a code module that may, for example, be resident in the random access memory of the computer. Until required by the computer, the set of instructions may be stored in another computer memory, for example, in a hard disk drive, or in a removable memory such as an optical disk (for eventual use in a CD ROM) or floppy disk (for eventual use in a floppy disk drive), or downloaded via the Internet or other computer network. Thus, the present invention may be implemented as a computer program product for use in a computer. In addition, although the various methods described are conveniently implemented in a general purpose computer selectively activated or reconfigured by software, one of ordinary skill in the art would also recognize that such methods may be carried out in hardware, in firmware, or in more specialized apparatus constructed to perform the required method steps.

[0080] While particular embodiments of the present invention have been shown and described, it will be obvious to those skilled in the art that, based upon the teachings herein, that changes and modifications may be made without departing from this invention and its broader aspects. Therefore, the appended claims are to encompass within their scope all such changes and modifications as are within the true spirit and scope of this invention. Furthermore, it is to be understood that the invention is solely defined by the appended claims. It will be understood by those with skill in the art that if a specific number of an introduced claim element is intended, such intent will be explicitly recited in the claim, and in the absence of such recitation no such limitation is present. For non-limiting example, as an aid to understanding, the following appended claims contain usage of the introductory phrases "at least one" and "one or more" to introduce claim elements. However, the use of such phrases should not be construed to imply that the introduction of a claim element by the indefinite articles "a" or "an" limits any particular claim containing such introduced claim element to inventions containing only one such element,

even when the same claim includes the introductory phrases "one or more" or "at least one" and indefinite articles such as "a" or "an"; the same holds true for the use in the claims of definite articles.

What is claimed is:

1. A computer-implemented method comprising:

receiving a payment operation request from an order system, the payment operation request including payment parameters;

prior to sending a financial transaction request to a payment provider, determining, using the payment parameters, whether a payment operation has completed for the payment operation request;

in response to determining that the payment operation has not completed, sending the financial transaction request to the payment provider; and

in response to determining that the payment operation for the payment request has completed, the method further comprising:

retrieving financial transaction results that result from the payment operation that was performed prior to receiving the payment operation request; and

sending the financial transaction results to the order system.

2. The method of claim **1** wherein the payment parameters include an order identifier and a release identifier, the order identifier identifying an order and the release identifier identifying at least a portion of the order that is released for shipment.

3. The method of claim **1** further comprising:

in response to sending the financial transaction request, receiving the financial transaction results from the payment provider, which completes the payment operation.

4. The method of claim **1** further comprising:

wherein the method is performed by a payment system; and

wherein the order system, the payment system, and the payment provider are separate entities.

5. The method of claim **2** wherein the determining further comprises:

in response to receiving the payment operation request, creating a payment identifier that includes an order phase, the order phase corresponding to a phase in the order system that sent the payment operation request;

associating the order identifier and the release identifier to the payment identifier; and

performing the determining using the payment identifier, the order identifier, and the release identifier.

6. The method of claim **5** further comprising:

wherein the order phase is selected from the group consisting of an order capture phase, an order process phase, a fulfillment phase, and a shipping phase; and

wherein the payment operation request is selected from the group consisting of a payment instruction validation request, a payment instruction storage request, a process payment instruction after allocation request, and a process payment instruction after shipment request.

7. A computer program product stored on a computer operable media, the computer operable media containing instructions for execution by a computer, which, when executed by the computer, cause the computer to implement a method for detecting duplicate payment operation requests, the method comprising:

receiving a payment operation request from an order system, the payment operation request including payment parameters;

prior to sending a financial transaction request to a payment provider, determining, using the payment parameters, whether a payment operation has completed for the payment operation request;

in response to determining that the payment operation has not completed, sending the financial transaction request to the payment provider; and

in response to determining that the payment operation for the payment request has completed, the method further comprising:

retrieving financial transaction results that result from the payment operation that was performed prior to receiving the payment operation request; and

sending the financial transaction results to the order system.

8. The computer program product of claim **7** wherein the payment parameters include an order identifier and a release identifier, the order identifier identifying an order and the release identifier identifying at least a portion of the order that is released for shipment.

9. The computer program product of claim **7** wherein the method further comprises:

in response to sending the financial transaction request, receiving the financial transaction results from the payment provider, which completes the payment operation.

10. The computer program product of claim **7** further comprising:

wherein the method is performed by a payment system; and

wherein the order system, the payment system, and the payment provider are separate entities.

11. The computer program product of claim **8** wherein the method further comprises:

in response to receiving the payment operation request, creating a payment identifier that includes an order phase, the order phase corresponding to a phase in the order system that sent the payment operation request;

associating the order identifier and the release identifier to the payment identifier; and

performing the determining using the payment identifier, the order identifier, and the release identifier.

12. The computer program product of claim **11** further comprising:

wherein the order phase is selected from the group consisting of an order capture phase, an order process phase, a fulfillment phase, and a shipping phase; and

wherein the payment operation request is selected from the group consisting of a payment instruction validation request, a payment instruction storage request, a process payment instruction after allocation request, and a process payment instruction after shipment request.

13. An information handling system comprising:

one or more processors;

a memory accessible by the processors;

one or more nonvolatile storage devices accessible by the processors; and

a payment operation request tool for detecting duplicate payment operation requests, the payment operation request tool being effective to:

receive a payment operation request from an order system, the payment operation request including payment parameters;

prior to sending a financial transaction request to a payment provider, determine, using the payment parameters, whether a payment operation has completed for the payment operation request;

in response to determining that the payment operation has not completed, send the financial transaction request to the payment provider; and

in response to determining that the payment operation for the payment request has completed, the method further comprising:

retrieving financial transaction results that result from the payment operation that was performed prior to receiving the payment operation request; and

sending the financial transaction results to the order system.

14. The information handling system of claim **13** wherein the payment parameters include an order identifier and a release identifier, the order identifier identifying an order and the release identifier identifying at least a portion of the order that is released for shipment.

15. The information handling system of claim **13** wherein the payment operation request tool is further effective to:

in response to sending the financial transaction request, receive the financial transaction results from the payment provider, which completes the payment operation.

16. The information handling system of claim **14** wherein the payment operation request tool is further effective to:

in response to receiving the payment operation request, create a payment identifier that includes an order phase, the order phase corresponding to a phase in the order system that sent the payment operation request;

associate the order identifier and the release identifier to the payment identifier; and

perform the determining using the payment identifier, the order identifier, and the release identifier.

17. The information handling system of claim **16** further comprising:

wherein the order phase is selected from the group consisting of an order capture phase, an order process phase, a fulfillment phase, and a shipping phase; and

wherein the payment operation request is selected from the group consisting of a payment instruction validation request, a payment instruction storage request, a process payment instruction after allocation request, and a process payment instruction after shipment request.

* * * * *