SYSTEMS AND METHODS FOR VALUE UNIT CONVERSION, UTILIZATION, MEDIATION, AND FOR CREATING A TRANSACTION PRODUCT APPLICATION

CROSS REFERENCE TO RELATED APPLICATIONS

[0001] This application claims priority to US patent application 15/630,853, titled "SYSTEM AND METHOD FOR VALUE UNIT CONVERSION MEDIATED BY MULTIPLE EXCHANGE QUERYING AND EVALUATION" which was filed on June 22, 2017. This application also claims priority to US patent application 15/623,386, titled "SYSTEM AND METHOD FOR VALUE UNIT CONVERSION AND UTILIZATION", which was filed on June 15, 2017. This application also claims priority to US patent application 15/608,974, titled "SYSTEM AND METHOD FOR CREATING A TRANSACTION PRODUCT APPLICATION PROGRAMMING INTERFACE THAT INTERACTS WITH BOTH A TRANSACTION NETWORK AND COMMUNICATIONS NETWORK", which was filed on May 30, 2017, the entire specifications of which are each hereby incorporated by reference.

BACKGROUND

Field of the invention

[0002] The disclosure as detailed herein is in the technical field of transactional systems. More specifically, the present disclosure relates to the technical field of end user configured application programming interfaces. Even more specifically, the present disclosure relates to the technical field of exchange systems for transactions.

Description of related art

[0003] Currently, banks and other payment product issuers have financial instruments and transaction products that they offer to their customers. These include personal or business checking, savings and borrowing accounts with different functionalities as well as terms and conditions, payment products and functionalities such as electronic and wire transfers, online bill payments, debit and credit cards,

person-to-person transfers and others. These financial instruments and transaction products are valuable tools banks, issuers and other companies offer to their customers.

[0004] When a consumer initiates a transaction using a payment product, the acquirer receiving the transaction creates a transaction message which includes details of the transaction requirements and the payment product which identify the issuer and the customer account of the product. The payment transaction message is transmitted via a switch into the transaction network where it is routed to the issuer processor of the particular issuer and particular account associated with the transaction. Once the transaction gets to the issuer processor, the issuer processor receives the payment transaction details and examines the account associated with the message to see if the transaction requirements can be met, then authorizes the transaction if the requirements can be met. A response message is then sent from the issuer processor back to the switch, which then passes the response message to the acquirer which would approve or decline the transaction at the point of sale.

[0005] Currently, there are few means for customizing the interaction that a consumer or company might desire on a per transaction basis. One of the existing means, for example, is that rules can be established by the issuer that set limits per day on the amount of transaction for a particular payment product. However, these rules that would regulate the types of features that may exist for a transaction are typically hard coded into software at particular issuing institutions. This hard-coded software is then often used to provide financial products to end users in order to be competitive on the market.

[0006] What is needed is a configurable plurality of digital wallets comprising an exchange processor that interacts with one or more value unit exchanges in order to create value unit conversion rates for value units across multiple value unit exchanges (as integrations). This allows the nominal usage of multiple types of value units such as digital currencies (e.g. bitcoin), fiat currency, consumer rewards (e.g. loyalty points) and other types of value units to be used on a per-transaction basis by an end user.

SUMMARY OF THE INVENTION

[0007] This invention is a platform that allows banks, issuers and other companies to create financial instruments and transaction products using a control console. Further, the control console accepts transaction product input as a means to configure a transaction product that can be offered to end users. The platform has a product engine that accepts these parameters and builds an application programming interface transaction product, that allows an end user to manage and configure aspects of the transaction product and value unit transactions at the product level and on a per-transaction basis (once an end user application is built for the product).

[0008] Further, this invention discloses one type of transaction product called a value unit conversion and utilization product, which allows the designation and manipulation of multiple value unit wallets that allow an end user to designate which types of value units in a transaction product account may be implemented on a per-transaction basis such that a person may perform a transaction in one value unit and settle it using another. The creation of the transaction product specifically may decrease the fees associated with value unit exchange.

[0009] In addition, the value unit conversion and utilization product may have multiple interfaces where a person may select alternate sequences of value unit withdrawal such that an end user may prioritize the different types of value units that are withdrawn from an account during a transaction.

[0010] Further, a preferred embodiment of the invention discloses an exchange processor that interacts with one or more value unit exchanges in order to create value unit conversion rates for value units across multiple value unit exchanges (as integrations). This allows the nominal usage of multiple types of value unit exchanges such as digital currencies (e.g. bitcoin), fiat currency, consumer rewards (e.g. loyalty points) and other types of value units to be used on a per-transaction basis by an end user.

DESCRIPTION OF FIGURES

[0011] Figure 1 is a perspective view, which shows an exemplary hardware architecture of a computing device used in an embodiment of the invention.

[0012] Figure 2 is a perspective view, which shows an exemplary logical architecture for a client device, according to an embodiment of the invention.

[0013] Figure 3 is a perspective view, which shows an exemplary architectural arrangement of clients, servers, and external services, according to an embodiment of the invention.

[0014] Figure 4 is a perspective view, which shows an embodiment of a hardware architecture of a computing device connected to a network used in various embodiments of the invention.

[0015] Figure 5 is a perspective view, which shows a system diagram for creating a transaction product.

[0016] Figure 6 is a perspective view, which shows the data and organization configured by the control console that allows the creation of a transaction product.

[0017] Figure 7 is a perspective view, which shows the creation and organization of wallets for processing from an acquirer on a per transaction basis.

[0018] Figure 8 is a perspective view, which shows implementation of the transaction product during a transaction.

[0019] Figure 9A-9B is a diagram of overall use of the system.

[0020] Figure 10 is a diagram of setting up a transaction product partner.

[0021] Figure 11A-11B is a diagram of setting up a transaction product.

[0022] Figure 12 is a diagram of configuring the value unit conversion and utilization product.

[0023] Figure 13 is a diagram of using the value unit conversion and utilization product.

[0024] Figure 14 is a diagram of computing the total required amount.

[0025] Figure 15 is a diagram of examination of the current wallet.

[0026] Figure 16 is a diagram of determining the value unit conversion rate.

[0027] Figure 17 is a diagram of determining the method for establishing a value unit conversion rate.

[0028] Figure 18 is a diagram of interacting with two integrations when there are no integrations with both the wallet value unit and the target amount value unit.

[0029] Figure 19 is a diagram of getting a quote value unit conversion rate from a single interaction.

[0030] Figure 20 is a diagram of getting a quote value unit conversion rate from two or more integrations that have both wallet value unit and target value unit.

DETAILED DESCRIPTION

[0031] One or more different inventions may be described in the present application. Further, for one or more of the inventions described herein, numerous alternative embodiments may be described; it should be appreciated that these are presented for illustrative purposes only and are not limiting of the inventions contained herein or the claims presented herein in any way. One or more of the inventions may be widely applicable to numerous embodiments, as may be readily apparent from the disclosure. In general, embodiments are described in sufficient detail to enable those skilled in the art to practice one or more of the inventions, and it should be appreciated that other embodiments may be utilized and that structural, logical, software, electrical and other changes may be made without departing from the scope of the particular inventions. Accordingly, one skilled in the art will recognize that one

or more of the inventions may be practiced with various modifications and alterations.

[0032] Particular features of one or more of the inventions described herein may be described with reference to one or more particular embodiments or figures that form a part of the present disclosure, and in which are shown, by way of illustration, specific embodiments of one or more of the inventions. It should be appreciated, however, that such features are not limited to usage in the one or more particular embodiments or figures with reference to which they are described. The present disclosure is neither a literal description of all embodiments of one or more of the inventions nor a listing of features of one or more of the inventions that must be present in all embodiments.

[0033] Headings of sections provided in this patent application and the title of this patent application are for convenience only and are not to be taken as limiting the disclosure in any way.

[0034] Devices that are in communication with each other need not be in continuous communication with each other, unless expressly specified otherwise. In addition, devices that are in communication with each other may communicate directly or indirectly through one or more communication means or intermediaries, logical or physical.

[0035] A description of an embodiment with several components in communication with each other does not imply that all such components are required. To the contrary, variety of optional components may be described to illustrate a wide variety of possible embodiments of one or more of the inventions and in order to more fully illustrate one or more aspects of the inventions. Similarly, although process steps, method steps, algorithms or the like may be described in a sequential order, such processes, methods, and algorithms may generally be configured to work in alternate orders, unless specifically stated to the contrary. In other words, any sequence or order of steps that may be described in this patent application does not, in and of itself, indicate a requirement that the steps be performed in that order. The steps of described processes may be performed in any order practical. Further, some steps

may be performed simultaneously despite being described or implied as occurring non-simultaneously (e.g., because one step is described after the other step).

[0036] Moreover, the illustration of a process by its depiction in a drawing does not imply that the illustrated process is exclusive of other variations and modifications thereto, does not imply that the illustrated process or any of its steps are necessary to one or more of the invention(s), and does not imply that the illustrated process is preferred. Also, steps are generally described once per embodiment, but this does not mean they must occur once, or that they may only occur once each time a process, method, or algorithm is carried out or executed. Some steps may be omitted in some embodiments or some occurrences, or some steps may be executed more than once in a given embodiment or occurrence.

[0037] When a single device or article is described herein, it will be readily apparent that more than one device or article may be used in place of a single device or article. Similarly, where more than one device or article is described herein, it will be readily apparent that a single device or article may be used in place of the more than one device or article. The functionality or the features of a device may be alternatively embodied by one or more other devices that are not explicitly described as having such functionality or features. Thus, other embodiments of one or more of the inventions need not include the device itself. Techniques and mechanisms described or referenced herein will sometimes be described in singular form for clarity. However, it should be appreciated that particular embodiments may include multiple iterations of a technique or multiple instantiations of a mechanism unless noted otherwise.

[0038] Process descriptions or blocks in figures should be understood as representing modules, segments, or portions of code which include one or more executable instructions for implementing specific logical functions or steps in the process. Alternate implementations are included within the scope of embodiments of the present invention in which, for example, functions may be executed out of order from that shown or discussed, including substantially concurrently or in reverse order,

depending on the functionality involved, as would be understood by those having ordinary skill in the art.

[0039] Software/hardware hybrid implementations of at least some of the embodiments disclosed herein may be implemented on a programmable network-resident machine (which should be understood to include intermittently connected network-aware machines) selectively activated or reconfigured by a computer program stored in memory. Such network devices may have multiple network interfaces that may be configured or designed to utilize different types of network communication protocols. A general architecture for some of these machines may be described herein in order to illustrate one or more exemplary means by which a given unit of functionality may be implemented.

[0040] According to specific embodiments, at least some of the features or functionalities of the various embodiments disclosed herein may be implemented on one or more general-purpose computers associated with one or more networks, such as for example an end-user computer system, client computer, network server or other server system, mobile computing device (e.g., tablet computing device, mobile phone, smartphone, laptop, or other appropriate computing device), consumer electronic device, music player, or any other suitable electronic device, router, switch, or other suitable device, or any combination thereof. In at least some embodiments, at least some of the features or functionalities of the various embodiments disclosed herein may be implemented in one or more virtualized computing environments (e.g., network computing clouds, virtual machines hosted on one or more physical computing machines, or other appropriate virtual environments).

[0041] Referring now to **Figure 1**, which shows an exemplary hardware architecture of a computing device used in an embodiment of the invention. Computing device **101** comprises an electronic device capable of executing software- or hardware-based instructions according to one or more programs stored in memory. In some embodiments, examples of computing device **101** may include: desktop computers, carputers, game consoles, laptops, notebooks, palmtop, tablet, smartphones, smartbooks, or a server system utilizing CPU **102**, local memory **103** and/or remote

memory **105**, and interface 106. CPU **102** comprises a unit responsible for implementing specific functions associated with the functions of specifically configured computing device or machine. The central processing unit is an acronym which stands for CPU **102**. In some embodiments, examples of CPU **102** may include: system-on-a-chip (SOC) type hardware, Qualcomm SNAPDRAGON™, or Samsung EXYNOS™ CPU.

[0042] Local memory **103** comprises one or more physical devices used to store programs (sequences of instructions) or data (e g. program state information) on a temporary or permanent basis for use in a computer or other digital electronic device, which may be configured to couple to the system in many different configurations. In some embodiments, examples of local memory **103** may include: non-volatile random-access memory (RAM), read-only memory (ROM), or one or more levels of cached memory. Processor **104** comprises a component that performs the instructions and tasks involved in computer processing. In some embodiments, examples of processor **104** may include: Intel processor, ARM processor, Qualcomm processor, AMD processor, application-specific integrated circuits (ASICs), electrically erasable programmable read-only memories (EEPROMs), field-programmable gate arrays (FPGAs), mobile processor, microprocessor, microcontroller, microcomputer, programmable logic controller, or programmable circuit.

[0043] Remote memory **105** comprises a service that provides users with a system for the backup, storage, and recovery of data. Interface **106** comprises a mechanism to control the sending and receiving of data packets over a computer network or support peripherals used with the computing device **101**. In some embodiments, examples of interface **106** may include: network interface cards (NICs), ethernet interfaces, frame relay interfaces, cable interfaces, DSL interfaces, token ring interfaces, graphics interfaces, universal serial bus (USB) interfaces, Serial port interfaces, Ethernet interfaces, FIREWIRE™ interfaces, THUNDERBOLT™ interfaces, PCI interfaces, parallel interfaces, radio frequency (RF) interfaces, BLUETOOTH™ interfaces, near-field communications interfaces, 802.11 (WiFi) interfaces, frame relay interfaces, TCP/IP interfaces, ISDN interfaces, fast Ethernet interfaces, Gigabit Ethernet interfaces, Serial ATA (SATA) or external SATA (ESATA) interfaces, high-

definition multimedia interface (HDMI), digital visual interface (DVI), analog or digital audio interfaces, asynchronous transfer mode (ATM) interfaces, high-speed serial interface (HSSI) interfaces, Point of Sale (POS) interfaces, or fiber data distributed interfaces (FDDIs).

[0044] Communications network **107** comprises a communications network that allows computers to exchange data using known protocols. In some embodiments, examples of communications network **107** may include: personal area network, wireless personal area network, near-me area network, local area network, wireless local area network, wireless mesh network, wireless metropolitan area network, wireless wide area network, cellular network, home area network, storage area network, campus area network, backbone area network, metropolitan area network, wide area network, enterprise private network, virtual private network, intranet, extranet, Internetwork, Internet, near field communications, mobile telephone network, CDMA network, GSM cellular networks, or WiFi network.

[0045] Referring now to **Figure 2**, which shows an exemplary logical architecture for a client device, according to an embodiment of the invention. Client application **201** comprises a computing device **101** capable of obtaining information and applications from a server. Shared service **202** comprises web-enabled services or functionality related to a computing device **101**. Operating systems **203** comprises system software that manages computer hardware and software resources and provides common services for computer programs. In some embodiments, examples of operating systems **203** may include: Microsoft's WINDOWS™, Apple's Mac OS/X, iOS operating systems, Linux operating system, or Google's ANDROID™ operating system. Input devices **204** comprises device of any type suitable for receiving user input. In some embodiments, examples of input devices **204** may include: keyboard, touchscreen, microphone, mouse, touchpad, or trackball.

[0046] Memory **205** comprises mechanism designed to store program instructions, state information, and the like for performing various operations described herein, may be storage devices **207**, in some embodiments. In some embodiments, examples of memory **205** may include: read-only memory (ROM), read-only memory (ROM)

devices, memristor memory, random access memory (RAM), or RAM hardware modules.

[0047] Output devices **206** comprises device of any type suitable for outputting computing device **101** related information. In some embodiments, examples of output devices **206** may include: screens for visual output, speakers, or printers. Storage devices **207** comprises mechanism designed to store information which in some embodiments may be memory **205**. In some embodiments, examples of storage devices **207** may include: magnetic media, hard disks, floppy disks, magnetic tape, optical media, CD-ROM disks, magneto-optical media, optical disks, flash memory, solid state drives (SSD), "hybrid SSD" storage drives, swappable flash memory modules, thumb drives, thumb drives, removable optical storage discs, or electrical storage device.

[0048] Referring now to **Figure 3**, which shows an exemplary architectural arrangement of clients, servers, and external services, according to an embodiment of the invention. Server **301** comprises a computing device **101** configured to handle requests received from one or more client **302** over a communications network **107**. Client **302** comprises one or more computing device **101** with program instructions for implementing client-side portions of the present system which in some embodiments, may be connected to a communications network **107**.

[0049] Database **303** comprises programming instructions to provide an organized collection of data designed to allow the definition, creation, querying, update, and administration of databases. In some embodiments, examples of database **303** may include: relational database system, NoSQL system, Hadoop system, Cassandra system, Google BigTable, column-oriented databases, in-memory databases, or clustered databases.

[0050] External service **304** comprises web-enabled services or functionality related to or installed on a computing device **101** itself which may be deployed on one or more of a particular enterprise's or user's premises. Configuration system **305** comprises a system common to information technology (IT) and web functions that implements configurations or management system. Security system **306** comprises a

system common to information technology (IT) and web functions that implements security related functions for the system. Distributed computing network **307** comprises any number of client **302** and/or server **301** operably connected to a communications network **107** for the purposes of implementing the system.

[0051] Referring now to **Figure 4**, which shows an embodiment of a hardware architecture of a computing device connected to a network used in various embodiments of the invention. Real time clock **401** comprises a computing device **101** clock (most often in the form of an integrated circuit) that keeps track of the current time. Nonvolatile memory **402** comprises computer memory that can retrieve stored information even after having been power cycled (turned off and back on). Power supply **403** comprises an electronic device that supplies electric energy to an electrical load. Input output units **404** comprises devices used by a human (or other system) to communicate with a computer. NIC **405** comprises a computer hardware component that connects a computer to a computer network.

[0052] Referring now to **Figure 5**, which shows a system diagram for creating a transaction product. Transaction platform configuration system **501** comprises a distributed computing network **307** system that is used to create a transaction product **510**. One goal of transaction platform configuration system **501** is to allow one or more persons to design and build one or more transaction product **510**. Transaction platform configuration system **501** preferably comprises control console **503**, product engine **507**, transaction instrument product data **504**, transaction product **510**, integration **515**, transaction product account **514**, and finally, end user application system **512**.

[0053] To start, a transaction product partner configures the transaction instrument product data **504** within the control console **503**. Transaction product partner **502** comprises an entity that specifies the input to the transaction platform configuration system **501** so that transaction product **510** is created. In some embodiments, transaction product partner **502** may include: a bank, a company, or a payment processor.

[0054] Control console **503** comprises an interface on a computing device **101** within a transaction platform configuration system **501** that accepts the transaction product data **505** that is used by the product engine **507** to create a transaction product **510**.

[0055] Transaction instrument product data **504** comprises data or data objects that are received by the product engine **507** that allow the creation of one or more transaction product **510**. Transaction instrument product data **504** functions to store one or more data parameters or objects for a customized implementation of a transaction product **510**. Transaction instrument product data **504** that may be entered into the control console preferably comprises transaction product data **505** and partner data **506**.

[0056] Transaction product data **505** comprises the overall data or data objects that is used to create a transaction product **510**. One goal of transaction product data **505** is to allow configuration and creation of multiple types of transaction product **510** dependent on the input data, that can be created by product engine **507**.

[0057] Transaction product data **505** preferably comprises settlement date data **601**, partner fee data **602**, program name data **606**, card profile data **605**, security level data **612**, default language data **611**, customer interaction data **609**, partner risk data **614**, end user risk data **607**, mail user group data **613**, fee list data **603**, value unit data **608**, mail template data **604**, partner location data **610**, and finally, memo template data **622**.

[0058] Partner data **506** comprises data or data objects that configure the transaction product parameters when the transaction product **510** may be being used typically done by a transaction product partner **502**. Partner data **506** preferably comprises partner name data **619**, partner settlement time data **617**, partner main contact data **618**, partner language data **620**, partner value unit list data **621**, and finally, partner administrator data **623**.

[0059] Once control console receives the transaction instrument product data, a product engine produces the transaction product. Product engine **507** comprises one

or more modules on a computing device **101** operably connected to distributed computing network **307** that process transaction product data **505** and create a transaction product **510**. One goal of product engine **507** may be to allow the creation of multiple transaction product **510** from different types of transaction product data **505**. Product engine **507** preferably comprises integration manager **509** and transaction product creator **508**.

[0060] In one embodiment, the transaction product may be created by a transaction product creator **508**. Transaction product creator **508** comprises one or more modules that produce the end user application system **512**. Transaction product creator **508** has many purposes which are as follows: First, the purpose of transaction product creator **508** may be to allow the creation of end user application system **512** within the transaction product **510**. Next, it serves to receives transaction instrument product data **504**. Next, transaction product creator **508** serves to create a user interface for one or more end-user for operating and/or managing a transaction product **510**.

[0061] In one embodiment, an integration manager aids the product engine for configuring integrations for the transaction product. Integration manager **509** comprises one or more modules that operably connects to one or more integration **515** for use within the output transaction product **510**. One goal of integration manager **509** may be to allow outside data integrations into the transaction product **510** for use, such as from a third-party service provider.

[0062] In one embodiment, the transaction product may be a resultant sub-system created by the product engine and includes an API for interaction with an end user application system. Transaction product **510** comprises an output of the transaction platform configuration system **501** as a sub-system that includes application programming interface (API) that may be configured to be accessed and managed by one or more transaction product partner **502** and one or more end user **516** for creating applications that allow control over value unit resources on a per transaction basis. In some embodiments, a transaction product **510** preferably comprises value unit conversion and utilization product **711**.

[0063] Integration system **511** comprises one or more modules that allow the configuration of the transaction product **510** to be operably connected to one or more external services such as an integration **515**.

[0064] End user application system **512** comprises an end user based computing device for interaction with API for integration with preferably an interface that allows use of the transaction product **510**. End user application system **512** comprises end user application device **513** and integration system **511**.

[0065] End user application device **513** comprises one or more computing device **101** that may be used to interact with a transaction product **510** operably connected to a communications network **107**. End user application device **513** preferably comprises end user application display **701**.

[0066] In some embodiments, a transaction product account **514** comprises a holding account for one or more wallet **703** into which value unit can be stored. Integration **515** comprises an external service that connects to the transaction product **510** to perform a transaction. In some embodiments, examples of integration **515** may include: payment gateways and networks, bitcoin exchanges, metal exchanges, value unit exchanges, loyalty point managers, or foreign exchange providers. Transaction network **517** comprises a type of communications network **107** that allows for transactions to be performed.

[0067] End user **516** comprises the person who uses the system. Acquirer **518** comprises the entity that processes payments on behalf of a merchant. The acquirer allows merchants to accept payment products from different issuers. Switch **519** comprises the entity that routes transaction messages to the correct networks or entities. Payment transaction message **520** comprises a message including details regarding the transaction and the payment product used to satisfy the transaction, and whether authorization or clearing may be requested or if the transaction was authorized or declined. Acquiring amount **521** comprises the amount of the charge for a transaction that may be received by an acquirer **518** as part of the payment transaction message **520**. Authorization message **522** comprises a message indicating that a transaction was authorized.

[0068] Referring now to **Figure 6**, which shows the data and organization configured by the control console that allows the creation of a transaction product. Settlement date data **601** comprises data or data objects that configure the specific transaction product **510** for the settlement date. One goal of settlement date data **601** may be to allow the specific configuration of the time in which the settlements for transactions should occur. Partner fee data **602** comprises data or data objects that characterize a per transaction fee that may be levied by transaction product partner **502**. One goal of partner fee data **602** may be to configure the transaction product **510** to designate the transaction product partner **502** fee. Fee list data **603** comprises data or data objects that specifies one or more fee levied to an end user **516** of the transaction product **510**. One goal of fee list data **603** may be to allow the transaction product partner **502** to ascribe one or more fees to particular end user **516** or transactions.

[0069] Mail template data **604** comprises template data, data object or message that may be used to enable a transaction product partner **502** to notify an end user **516** of an event. Mail template data **604** allows communication from transaction product partner **502** to end user **516** to be styled and customized. Card profile data **605** comprises data or data objects that configure the specific transaction product **510** for the design of any physical cards issued as part of the transaction product **510** and establishes the first 6 digits of any physical or virtual card. Program name data **606** comprises data or data objects that configures the specific name of the transaction product **510** that may be being used.

[0070] End user risk data **607** comprises data or parameters that are set that configure the transaction product **510** to limit a customer's spending or withdrawal on specified metrics. In some embodiments, examples of end user risk data **607** may include: limiting the one time, one day or seven day customer P2P amount, limiting the one time, one day or seven day ATM withdrawal amount, limiting one time, one day or seven day purchase amount, limiting the one time, one day or seven day adjustment amount, limiting the one time, one day or seven day refund count, limiting the one time, one day or seven day refund amount, limiting the number of pin failures, listing blocked countries, listing blocked merchant categories, or defining

transaction speed (for example, in mph, calculated based on time lapsed and distance traveled between two in-person transactions using the same transaction product **510** by the end user **516**, where if exceeded, a flag might be set as too fast). One goal of end user risk data **607** may be to allow the specific metrics to be set for risk associated with an end user's transactions.

[0071] Value unit data **608** comprises data or data objects that configure the specific types of value unit when the transaction product **510** may be used. One goal of value unit data **608** may be to have a list of value units that a transaction product **510** can operate on. Customer interaction data **609** comprises data or data objects that configure the specific interactions that customers can have with the transaction product **510**. Partner location data **610** comprises data or data objects that contain a location where the transaction product partner **502** can distribute, sell, or service a transaction product **510**. One goal of partner location data **610** may be to allow the configuration of the transaction product **510** to be configured to a location of the transaction product partner **502**.

[0072] Default language data **611** comprises data or data objects that configure the default language to be used in the transaction product **510**. One goal of default language data **611** may be to allow the transaction product **510** to be configured into different languages for potential use in countries of varying language. Security level data **612** comprises data or data objects that may be a grouping of one or more end user risk data **607** parameters that can be designated as nominal security level. For example, 3 risk parameters can be grouped into security level 1. Then, an additional 5 risk parameters can be designated as level 2. Security level data **612** functions to both 1) allow the transaction product **510** to have varying groups of end user risk data **607** designated as security levels and to 2) allow customer to be assigned to different security levels.

[0073] Mail user group data **613** comprises data or data objects that are customers or users of the transaction product **510** for communication such as marketing, safety, security, etc. One goal of mail user group data **613** may be to allow emails to be sent to an end-user of the transaction product **510** with specific templating or styles.

Partner risk data **614** comprises data or data objects used to control the transactional volume of one or more transaction product partner locations used by the transaction product partner **502** to service one or more of the transaction product **510**. One goal of partner risk data **614** may be to allow the configuration for setting parameters. Partner risk data **614** comprises partner location risk data **615** and partner user risk data **616**. In some embodiments, an example of partner location risk data **615** could be limiting the one time, one day or seven-day customer load amount per location or perhaps limiting the one time, one day or seven day customer withdrawal amount per location and the like.

[0074] Partner user risk data **616** comprises data or data objects that are used to control the transactional volume of partner users at transaction product partner locations used by the transaction product partner **502** to service one or more of the transaction product **510**. In some embodiments, an example of partner user risk data **616** could be limiting the one time, one day or seven-day customer withdrawal amount per partner user or perhaps limiting the one time, one day or seven day customer load amount per partner user and the like. Partner settlement time data **617** comprises data or data objects that configure the transaction partner settlement time when the transaction product **510** may be used. One goal of partner settlement time data **617** may be to allow the specific configuration of the settlement time (of day) when aggregating transactions and in which the settlements for transactions should occur.

[0075] Partner main contact data **618** comprises data or data objects that configure the transaction product partner **502** main person's contact information. Partner name data **619** comprises data or data objects that configure the partner data **506** name information for the transaction product **510**. Partner language data **620** comprises data or data objects that configure the partner data **506** for the languages to display when using the transaction product **510**. Partner value unit list data **621** comprises data, array, data objects or data object arrays that configure the transaction product partner **502** value units that they can support that are to be part of the transaction product **510**. Memo template data **622** comprises template data, data object or message that may be used to provide an end user **516** with specific information

regarding each transaction. One goal of memo template data **622** may be to allow transaction detail communication to end user **516** to be styled and customized. Partner administrator data **623** comprises data or data objects that allow the establishment of profiles and roles for people to administer the transaction product **510**. In some embodiments, examples of partner administrator data **623** may include: risk parameters, security levels, role configurations, or permissions configurations. One goal of partner administrator data **623** may be to allow the configuration of users of the transaction product **510**.

[0076] Referring now to **Figure 7**, which shows the creation and organization of wallets for processing from an acquirer on a per transaction basis. End user application display **701** comprises a display for interaction with value unit conversion and utilization product **711** platform. End user application display **701** comprises wallet container **702** and risk management interface **705**. Wallet container **702** comprises the display where the wallet **703** order can be adjusted to determine the primary wallet sequence data **804**. Wallet container **702** preferably comprises one or more wallet **703**.

[0077] Wallet **703** comprises a graphical module that may have an interface, data, data object or processor functions that hold an amount of value unit. Wallet **703** has an alternative embodiment herein termed a “cross-boundary wallet”. The cross-boundary wallet comprises a wallet that does not interact with a transaction product account but rather interacts with a cross-boundary account through an external call.

[0078] Wallet **703** preferably comprises secondary wallet sequence selector toggle **704**. The secondary wallet sequence selector toggle **704** comprises an end-user interface selector that allows an end user to toggle the status of a wallet **703** between being part of the secondary wallet sequence or not.

[0079] Risk management interface **705** comprises a means for transaction product partner **502** or end user **516** to manage their own risk based on end user risk data **607**. Value unit conversion manager **706** comprises a module that implements algorithms and/or stores data for value unit conversion. In one embodiment, value unit conversion manager **706** preferably comprises wallet processor **710**, transaction

evaluator **708**, exchange processor **709**, and finally, value unit registry **707**. Value unit registry **707** comprises parameters that configure value unit exchange.

[0080] Transaction evaluator **708** comprises one or more modules that manages the data, algorithms and functions of the transaction product **510** on a per transaction basis. In one embodiment, transaction evaluator **708** preferably comprises acquiring amount data **808**, transaction determiner **809**, applicable fees data **807**, fee selector **806**, transaction calculator **805**, and finally, total required amount **810**.

[0081] Exchange processor **709** comprises one or more modules that implements algorithms and/or stores data to mediate exchanges of value unit between one or more wallet container **702**. Exchange processor **709** preferably comprises wallet to target value unit rate **820**, target cross-trade value unit rate **821**, wallet cross-trade value unit rate **815**, cross-trade value unit **818**, cross-trade manager **817**, wallet value unit **816**, target value unit wallet **803**, target value unit **819**, and finally, required amount running total.

[0082] Wallet processor **710** comprises one or more modules that implement algorithms/store data to support wallet-based functions of the transaction product **510**. Wallet processor **710** preferably comprises secondary wallet sequence selector **811**, primary wallet sequence data **804**, secondary wallet sequence data **802**, wallet type data **813**, wallet process algorithm **812**, and finally, hold clearer **814**.

[0083] Value unit conversion and utilization product **711** comprises an embodiment of transaction product **510** that allows an end user **516** to manage multiple value unit types and conduct one or more transaction. One goal of value unit conversion and utilization product **711** may be to allow a person to have one or more wallet **703** of different value units operably connected to one or more exchanges for denoting the manner within which value unit can be exchanged, withdrawn, used for purchases or otherwise utilized. Value unit conversion and utilization product **711** preferably comprises value unit conversion manager **706**.

[0084] Referring now to **Figure 8**, which shows implementation of the transaction product during a transaction. Current wallet sequence **801** comprises the instance

data sequence that may be being iterated over. For example, the primary wallet sequence data **804** or secondary wallet sequence data **802**. Secondary wallet sequence data **802** comprises a sequence of one or more wallet **703** that may be selected from the primary wallet sequence data **804** and inherits the order of the primary wallet sequence data **804**.

[0085] Target value unit wallet **803** comprises a currency type that may be required from the payment transaction message **520** to fulfill. Primary wallet sequence data **804** comprises a sequence of one or more wallet **703** set in a particular order. One goal of primary wallet sequence data **804** may be to allow end user **516** of the transaction product **510** to set the order in which wallet value unit **816** should be applied to transactions.

[0086] Transaction calculator **805** comprises a module that sums the applicable fees data **807** and the acquiring amount data **808** to get the total required amount **810**. Fee selector **806** comprises a module that determines the additional fees to be added to the amount gathered from the payment transaction message **520**. One goal of fee selector **806** may be to have a means to query configured data to determine the fees to be added when determining the total required amount **810**.

[0087] Applicable fees data **807** comprises a module that holds the sum of the fees gathered from the fee selector **806**. Acquiring amount data **808** comprises the value unit amount gathered from the payment transaction message **520**. Transaction determiner **809** comprises module that gets one or more data from the payment transaction message **520** for use within the transaction evaluator **708**. One goal of transaction determiner **809** may be to allow processing of the payment transaction message **520** to gather relevant data to implement use of the value unit conversion and utilization product **711**.

[0088] Total required amount **810** comprises data or data object that stores the sum of the applicable fees data **807** and the acquiring amount data **808**.

[0089] Secondary wallet sequence selector **811** comprises a module that adds or removes a wallet **703** from the secondary wallet sequence data **802**. Wallet process

algorithm **812** comprises logics that determine and calculate if sufficient value units are available to satisfy a transaction, including ability to determine the order in which value units in wallet **703** within a value unit conversion and utilization product **711** should be applied, calculating and storing data regarding the required amount running total, placing authorization holds on value unit amounts. Wallet type data **813** indicates if a wallet **703** may be part of the secondary wallet sequence. Hold clearer **814** comprises one or more modules that clears a hold off of an authorization transaction. One goal of hold clearer **814** may be to allow one to clear the account of demarcated value units that were designated during an authorization transaction.

[0090] Wallet cross-trade value unit rate **815** comprises the value unit conversion rate between the target value unit **819** and the cross-trade value unit **818**. Wallet value unit **816** comprises a value unit type that may be available in the wallet container **702** in the value unit conversion and utilization product **711**. Cross-trade manager **817** comprises a module for identifying the different integration **515** that can have a cross-trade value unit **818** for generating a wallet to target value unit rate **820**.

[0091] Cross-trade value unit **818** comprises a common value unit between two or more integration **515**, and in some embodiments, allows the conversion from a first value unit type to a second value unit through this as an intermediary value unit. Target value unit **819** comprises a value unit type that may be required from the payment transaction message **520** to fulfill. Wallet to target value unit rate **820** comprises a value unit conversion rate derived from the wallet cross-trade value unit rate **815** and the target cross-trade value unit rate **821** through the intermediate cross-trade value unit **818**. Target cross-trade value unit rate **821** comprises a value unit conversion rate between the target value unit **819** and the cross-trade value unit **818**.

[0092] Referring now to **Figure 9A-9B**, which shows overall use of the system. In a first step, control console **503** within a transaction platform configuration system **501** may be displayed on a computing device **101** (Step **901**). Next, partner data **506** may be received from the control console **503** by the transaction product creator **508**

(Step **902**). Step **902** may be further detailed below in a related method (**1000** - 'setting up a transaction product partner').

[0093] Next, transaction product data **505** may be received from the control console **503** by the transaction product creator **508** (Step **903**). Step **903** may be further detailed below in a related method (**1100** - 'setting up a transaction product'). Next, transaction product **510** and transaction product partner **502** are configured to relate to one another as transaction instrument product data **504** (Step **904**). Next, end user application system **512** receives configuration for one or more transaction product account **514** through one or more computing device **101** (Step **905**). Next, integration manager **509** may be configured with product engine **507** and transaction product creator **508** to create transaction product **510** (Step **906**).

[0094] Next, end user application system **512** may be operably connected to communications network **107** for access to one or more end user application system 512 (Step **907**). Next, one or more end user application display **701** are created to be displayed on one or more end user application device **513** (Step **908**).

[0095] Next, end user **516** implements one or more transaction at an acquirer **518**, and the related payment transaction message **520** are received by a transaction product **510**, which then implements its algorithms (Step **909**). If the transaction product **510** is a value unit conversion and utilization product **711** (Step **910**), then transaction evaluator **708** of the transaction determiner **809** identifies the type of transaction from the payment transaction message **520** (Step **911**).

[0096] If the transaction determiner **809** determines that the transaction type is a clearing transaction (Step **912**), then transaction evaluator **708** examines to see if clearing transaction has prior authorization (Step **913**).

[0097] If it does not have a prior authorization (Step **914**), then transaction may be evaluated by the value unit conversion manager **706** (Step **915**). Step **915** is further detailed below in a related method (**1300** - 'using the value unit conversion and utilization product').

[0098] From Step **913**, if it does have a prior authorization (Step **916**), then hold clearer **814** module clears any holds associated with one or more wallet **703** (Step **917**). Next, refer to Step **915**.

[0099] From Step **911**, if the transaction determiner **809** determines the transaction type is an authorization transaction (Step **918**), then refer to Step **915**.

[0100] From Step **908**, if the end user application system **512** of the transaction product **510** is a value unit conversion and utilization product **711** and the end user **516** wants to customize the transaction product **510** (Step **919**), then value unit conversion and utilization product **711** may be configured for a transaction product account **514** (Step 920). Step **920** is further detailed below in a related method (**1200** - 'configuring the value unit conversion and utilization product'). Next, refer to Step **909**.

[0101] Referring now to **Figure 10**, which shows setting up a transaction product partner. In a first step, partner name data **619** may be received from the control console **503** and configured as part of the partner data **506** (Step **1001**). Next, partner language data **620** may be received from the control console **503** and configured as part of the partner data 506 (Step **1002**). Next, partner value unit list data **621** may be received from the control console **503** and may be configured as part of the partner data **506** (Step **1003**).

[0102] Next, partner settlement time data **617** may be received from the control console **503** and may be configured as part of the partner data **506** (Step **1004**). Next, partner main contact data **618** may be received from the control console **503** and may be configured as part of the partner data **506** (Step **1005**). Next, partner administrator data **623** may be received from the control console **503** and may be configured as part of the partner data **506** (Step **1006**).

[0103] From Step **1003**, if the transaction partner wants to be a cross boundary wallet provider (Step **1007**), then one or more portions of the partner value unit list data **621** may be designated as pertaining to a cross boundary account for a value unit (Step **1008**). Next, refer to Step **1004**.

[0104] Referring now to **Figure 11A-11B**, which shows setting up a transaction product. In a first step, value unit data **608** may be received from control console **503** and configured as part of the transaction product data **505** (Step **1101**). Next, default language data **611** may be received from the control console **503** and configured as part of the transaction product data **505** (Step **1102**). Next, partner location data **610** may be received from the control console **503** and configured as part of the transaction product data **505** (Step **1103**). Next, partner risk data **614** may be received from the control console **503** and configured as part of the transaction product data **505** (Step **1104**).

[0105] Next, partner fee data **602** may be received from the control console **503** and configured as part of the transaction product data **505** (Step **1105**). Next, customer interaction data **609** may be received from the control console **503** and configured as part of the transaction product data **505** (Step **1106**). Next, card profile data **605** may be received from the control console **503** and configured as part of the transaction product data **505** (Step **1107**). Next, program name data **606** may be received from the control console **503** and configured as part of the transaction product data **505** (Step **1108**).

[0106] Next, fee list data **603** may be received from the control console **503** and configured as part of the transaction product data **505** (Step **1109**). Next, mail user group data **613** may be received from the control console **503** and configured as part of the transaction product data **505** (Step **1110**). Next, settlement date data **601** may be received from the control console **503** and configured as part of the transaction product data **505** (Step **1111**). Next, security level data **612** may be received from the control console **503** and configured as part of the transaction product data **505** (Step **1112**).

[0107] Next, security level data **612** may be received from the control console **503** and configured as part of the transaction product data **505** as grouping of end user risk data **607** (Step **1113**). Next, end user risk data **607** may be received from the control console **503** and configured as part of the transaction product data **505** (Step **1114**). Next, mail template data **604** may be received from the control console **503**

and configured as part of the transaction product data **505** (Step **1115**). Next, memo template data **622** may be received from the control console **503** and configured as part of the transaction product data **505** (Step **1116**).

[0108] Referring now to **Figure 12**, which shows configuring the value unit conversion and utilization product. In a first step, default primary wallet sequence data **804** may be displayed on end user application display **701** and relevant value unit data **608** may be displayed (Step **1201**). Next, one or more wallet **703** in wallet container **702** in the end user application display **701** may be configured for wallet type data **813** by an end user **516** interaction (Step **1202**).

[0109] If one or more wallet **703** is added or removed as a secondary wallet sequence data **802** member within the end user application display **701** (Step **1203**), then secondary wallet sequence selector **811** adds or removes the wallet member selected (Step **1204**). Next, wallet processor **710** configures data within the value unit conversion and utilization product **711** and wallet type data **813** data may be stored (Step **1205**).

[0110] From Step **1202**, if primary wallet sequence data **804** of the wallet **703** inside the wallet container **702** is rearranged within the end user application display **701** (Step **1206**), then new primary wallet sequence data **804** may be updated by the wallet processor **710** as configured to that end user **516** (Step **1207**). Next, refer to Step **1205**.

[0111] Referring now to **Figure 13**, which shows using the value unit conversion and utilization product. In a first step, transaction determiner **809** processes the acquiring amount data **808** (Step **1301**). Next, transaction determiner **809** identifies the target value unit **819** of the acquiring amount data **808** (Step **1302**). Next, transaction calculator **805** computes the total required amount **810** and sets the required amount running total as the total required amount **810** (Step **1303**). Step **1303** is further detailed below in a related method (**1400** - 'computing the total required amount'). Next, the wallet processor 710 begins examination of wallet **703**, starting with the first wallet **703** in the secondary wallet sequence as the current wallet **703** (Step

1304). Step **1304** is further detailed below in a related method (**1500** - 'examination of the current wallet').

[0112] If transaction needs are not satisfied (Step **1305**), then the transaction evaluator sends a transaction declined message to the switch **519** (Step **1306**).

[0113] From Step **1304**, if transaction needs are satisfied (Step **1307**), and if the transaction is an authorization transaction (Step **1308**), then the wallet processor **710** places authorization holds on value unit amounts in wallet **703** which together, sum to the total required amount **810** (Step **1309**). Next, the transaction evaluator **708** sends an authorization message **522** to the switch **519** (Step **1310**).

[0114] From Step **1307**, if transaction is a clearing transaction (Step **1311**), then the exchange processor **709** executes the value unit conversions required to fulfill the total required amount **810** using the then applicable wallet to target value unit rate **820** from the integration **515** (Step **1312**). Next, the transaction evaluator **708** sends a message to the switch **519** accepting the clearing transaction (Step **1313**).

[0115] Referring now to **Figure 14**, which shows computing the total required amount. In a first step, fee selector **806** queries the fee list data **603** to determine the extra fee added to the end user (Step **1401**). Next, fee selector **806** may query the partner fee data **602** to determine the extra amount added to the end user if necessary (Step **1402**). Next, transaction calculator **805** sums the applicable fees data **807** and the acquiring amount data **808** to a total required amount **810** (Step **1403**).

[0116] Referring now to **Figure 15**, which shows examination of the current wallet. In a first step, wallet processor **710** examines the value unit and available amount of the current wallet **703** (Step **1501**). If the current wallet **703** is a cross boundary wallet, wallet processor **710** will examine the value unit and available amount of the current wallet **703** in the cross-boundary wallet provider using applicable programming interface (Step **1502**), then exchange processor **709** determines the applicable wallet to target value unit rate **820** (Step **1503**). Step **1503** is further detailed below in a related method (**1600** - 'determining the value unit conversion rate').

[0117] Next, transaction evaluator **708** uses the applicable wallet to target value unit rate **820** to convert the available value unit amount in the wallet **703** into the target value unit **819** of the total required amount **810**, and subtracts such amount from the required amount running total (Step **1504**).

[0118] If the resulting required amount running total is zero or less, the transaction needs are satisfied (Step **1505**). Then refer to Step **1307**.

[0119] From Step **1504**, if the resulting required amount running total is greater than zero, the transaction needs are not satisfied (Step **1506**). If there are other wallet **703** within the wallet sequence being examined available for examination, then the next wallet **703** in the wallet sequence being examined becomes the current wallet **703** (Step **1507**), then refer to Step **1501**.

[0120] From Step **1506**, if there are no other wallet **703** within the wallet sequence being examined available for examination (Step **1508**), and if the target value unit wallet **803** has not been examined, then the target value unit wallet **803** becomes the current wallet **703** (Step **1509**). Then refer to Step **1501**.

[0121] From Step **1508**, if the target value unit wallet **803** has already been examined (Step **1510**), and if the primary wallet sequence has not been examined, the first wallet **703** in the primary wallet sequence becomes the current wallet **703** (Step **1511**). Then refer to Step **1501**. From Step **1510**, if the primary wallet sequence has already been examined (Step **1512**). Then refer to step **1305**.

[0122] Referring now to **Figure 16**, which shows determining the value unit conversion rate. In a first step, exchange processor **709** queries the exchange table (Step **1601**). Next, the exchange processor **709** returns the value unit conversion rate (Step **1602**).

[0123] From Step **1601**, if the exchange table needs a periodic update (Step **1603**), then exchange table updater determines one or more value unit conversion rates from one or more value unit (Step **1604**). Step **1603** is further detailed below in a related method (**1700** – 'determining the method for establishing a value unit conversion rate'). Next, exchange processor **709** updates the exchange table (Step **1605**).

[0124] Referring now to **Figure 17**, which shows determining the method for establishing a value unit conversion rate. In a first step, exchange processor **709** determines the target value unit **819** (Step **1701**). Next, exchange processor **709** determines the wallet value unit **816** (Step **1702**).

[0125] If the system wants to get a value unit conversion rate quote from an integration **515** (Step **1703**), then it gets a quote value unit conversion rate from a single integration **515** (Step **1704**). Step **1704** is further detailed below in a related method (**1900** - 'getting a quote value unit conversion rate from a single interaction').

[0126] From Step **1702**, if the system wants to get a value unit conversion rate quote from two or more integration **515** (Step **1705**), then it gets a quote conversion rate from two or more integration **515** that have both wallet value unit **816** and target value unit **819** (Step **1706**). Step **1706** is further detailed below in a related method (**2000** - 'getting a quote value unit conversion rate from two or more integrations that have both wallet value unit and target value unit').

[0127] From Step **1702**, if there is no integration **515** that can match both the wallet value unit **816** and target value unit **819** (Step **1707**), then refer to step **2001**. Step **1707** is further detailed below in a related method (**1800** - 'interacting with two integrations when there are no integrations with both the wallet value unit and the target amount value unit').

[0128] Referring now to **Figure 18**, which shows interacting with two integrations when there are no integrations with both the wallet value unit and the target amount value unit. In a first step, cross-trade manager **817** identifies one or more integration **515** that has cross-trade value unit **818** and also wallet value unit **816** (Step **1801**). Next, cross-trade manager **817** identifies one or more integration **515** that has cross-trade value unit **818** and also target value unit **819** (Step **1802**). Next, wallet cross-trade value unit rate **815** may be received from one or more integration **515** (Step **1803**).

[0129] Next, target cross-trade value unit rate **821** may be received from one or more integration **515** (Step **1804**). Next, multiply the wallet cross-trade value unit

rate **815** by the target cross-trade value unit rate **821** in order to get the wallet to target value unit rate **820** (Step **1805**). Next, calculate the wallet value unit **816** amount needed based on the wallet to target value unit rate **820** to satisfy the transaction needs in the target value unit **819** (Step **1806**).

[0130] If this is a clearing transaction (Step **1807**), then submit the transaction to integration **515** to perform the value unit exchange or to be batch traded later (Step **1808**).

[0131] From Step **1806**, if this is an authorization transaction (Step **1809**), then add an authorization hold to the required wallet value unit **816** amount (Step **1810**).

[0132] Referring now to **Figure 19**, which shows getting a quote value unit conversion rate from a single interaction. In a first step, identify an integration **515** that has wallet value unit **816** and the target value unit **819** (Step **1901**). Next, system gets the wallet to target value unit rate **820** (Step **1902**).

[0133] Referring now to **Figure 20**, which shows getting a quote value unit conversion rate from two or more integrations that have both wallet value unit and target value unit. In a first step, identify two or more integration **515** that has wallet value unit **816** and target value unit **819** (Step **2001**). Next, choose the integration **515** with the lowest wallet to target value unit rate **820** (Step **2002**).

[0134] The following elements and/or terms: cross boundary wallet provider, authorization transaction, clearing transaction, required amount running total, exchange table, exchange table updater, value unit, transaction, cross boundary account, physical ports, independent processor, interface memory, busses, program instructions, system server, mouse, keyboard, graphical user interface, cross boundary wallet, and data object, are important for the working functionality, but do not appear in the drawings and are shown below.

[0135] Cross boundary wallet provider comprises an entity that holds an account for a wallet 703 that has URL, username, password, exposes that allows account management including authorization holds, withdrawals, transfers, and the like. Authorization transaction comprises a transaction transmitted from the acquirer **518**

through the transaction network **517**, requesting approval to authorize a transaction initiated by an end user **516**. Clearing transaction comprises a transaction in which actual settlement of payment may be required. Required amount running total comprises a running tally amount in the target value unit **819** used by the transaction evaluator **708** during the examination of the current wallet to determine if the transaction needs can be satisfied. Exchange table comprises data on different value unit and their exchange rates. Exchange table updater comprises a means to update the data in an exchange table.

[0136] Value unit comprises a unit of tradable value that can be calculated to have at least one exchange rate. Transaction comprises an event that effects a change in one or more value unit. In some embodiments, an example of transaction may include conversion, trade, transfer, payment, withdrawal and the like. Cross boundary account comprises the associated account for cross boundary wallet provider.

[0137] Physical ports comprise a specialized outlet on a piece of equipment to which a plug or cable connects. Independent processor comprises a processor **104** that allows communication with appropriate media. In some embodiments, an example of independent processor could be audio processor or perhaps video processor, and the like. In some embodiments, examples of interface memory may include: volatile and/or non-volatile memory (e.g., RAM), DDR, DDR2, or GDDR. Busses comprises a communication system that transfers data between components inside a computer, or between computers.

[0138] Program instructions comprises a mechanism for control execution of, or comprise of an operating system, and/or one or more applications. In some embodiments, examples of program instructions may include: object code, code produced by a compiler, machine code, code produced by an assembler or a linker, byte code, or code executed using an interpreter. System server comprises a computing device **101**. Mouse comprises a pointing device that detects two-dimensional motion relative to a surface. Keyboard comprises a text entry interface to type text and numbers into a word processor, text editor or other programs. Graphical user interface comprises a type of user interface that allows users to interact with

electronic devices through graphical icons and visual indicators such as secondary notation, instead of text-based user interfaces, typed command labels or text navigation.

[0139] Cross-boundary wallet comprises a wallet that does not interact with a transaction product account but rather interacts with a cross boundary account through an external call.

[0140] Data object comprises a variable, a data structure, a function, or a method, and as such, is a location in memory having a value and possibly referenced by an identifier. In some embodiments, an example of data object could be a particular instance of a class where the object can be a combination of variables, functions, and data structures or perhaps a table or column, or an association between data and a database entity (such as relating a person's age to a specific person) and the like.

What is claimed is:

1. An electronic nested-wallet sequencing system comprising:

a network-connected transaction product management computer comprising a processor, a memory, and programming instructions, the programming instructions, when executed by the processor, cause the processor to:

receive a plurality of connections from a plurality of user devices;

receive a plurality of connections from a plurality of external services;

receive a wallet container comprising a plurality of transaction product accounts, each transaction product account associated with a value unit type and a quantity of value units of a plurality of types of value units, the wallet container associated with a first user device of a plurality of user devices, the wallet container comprising a predefined wallet sequence for determining an order of usage the plurality of transaction product accounts;

send the plurality of transaction product accounts consistent with the predefined wallet sequence and the quantity of value units associated with each of the plurality of transaction product accounts to a graphical user interface associated with the first user device, the graphical user interface operable to rearrange the predefined wallet sequence, the graphical user interface further operable to enable each of the plurality of transaction product accounts, to be toggled for prioritization via user input from the first user device;

receive a primary wallet sequence from the first user device, the primary wallet sequence comprising a rearrangement of at least one of the plurality of transaction product accounts comprised within the predefined wallet sequence;

receive, from a first external service, a first transaction of a plurality of transactions;

determine a transaction value unit type associated with the first transaction;

match the transaction value unit type with the value unit type associated with at least one of the plurality of transaction product accounts comprised within the predefined wallet sequence;

receive, from the first user device, a toggle for one or more transaction product accounts of the plurality of transaction product accounts;

create a secondary wallet sequence comprising the toggled one or more transaction product accounts, the secondary wallet sequence determining an order of usage of the toggled one or more transaction product accounts, the order of usage of the secondary wallet sequence's toggled one or more transaction product accounts consistent with the primary wallet sequence, wherein the primary wallet sequence and the secondary wallet sequence is transformed into at least one application programming interface (API) used to create multiple utilization products for the first user device;

create a utilization product defining a prioritized order of usage for processing the first transaction, the prioritized order of usage of the utilization product for processing the first transaction determined by a sequence of: the matched at least one of the plurality of transaction product accounts followed by the toggled one or more transaction product accounts from the secondary wallet sequence followed by remaining transaction product accounts from the primary wallet sequence, wherein each of the multiple utilization products are generated based on a respective transaction value unit type for each of the plurality of transactions;

display the utilization product on the graphical user interface of the first user device;

determine a target value unit type associated with the first transaction;

determine a transaction amount corresponding to the target value unit type based on the first transaction;

activate the utilization product of the multiple utilization products based on the determined target value unit type;

start a running total based on the transaction amount;

iteratively evaluate each transaction product account sequentially within the utilization product based on the order of usage of the utilization product for processing the first transaction;

update the running total by subtracting the transaction amount from a converted quantity of value units associated with each transaction product account, wherein the converted quantity of value units corresponds to available target value units and other value units that are converted to the target value units when available target value units have been exhausted;

transfer the converted quantity of value units to a holding transaction product account associated with the first transaction when the running total reaches zero;

upon the iterative evaluation of all transaction product accounts comprised within the utilization product:

send a transaction declined message to the first external service when the running total is greater than zero;

otherwise,

receive a password from the first user device to release the converted quantity of value units from the holding account to a payee account;

send an authorization message to the first external service;

send the transaction declined message or the authorization message to the graphical user interface of the first user device.