(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) EP 3 889 870 A1

(12) **EUROPEAN PATENT APPLICATION**
published in accordance with Art. 153(4) EPC

(43) Date of publication:
06.10.2021 Bulletin 2021/40

(21) Application number: **19905428.9**

(22) Date of filing: **20.11.2019**

(51) Int Cl.:
G06Q 20/40 *(2012.01)*

(86) International application number:
PCT/CN2019/119788

(87) International publication number:
WO 2020/134738 (02.07.2020 Gazette 2020/27)

(84) Designated Contracting States:
AL AT BE BG CH CY CZ DE DK EE ES FI FR GB GR HR HU IE IS IT LI LT LU LV MC MK MT NL NO PL PT RO RS SE SI SK SM TR
Designated Extension States:
BA ME
Designated Validation States:
KH MA MD TN

(30) Priority: **29.12.2018 CN 201811638611**

(71) Applicant: **BIGO TECHNOLOGY PTE. LTD.**
117440 Singapore (SG)

(72) Inventors:
- **HOU, Juncheng**
  Guangzhou, Guangdong 511402 (CN)
- **YANG, Kaixing**
  Guangzhou, Guangdong 511402 (CN)

(74) Representative: **Yang, Shu**
Withers & Rogers LLP
2 London Bridge
London SE1 9RA (GB)

(54) BACKGROUND ARCHITECTURE METHOD AND SYSTEM FOR AGGREGATION PAYMENT, COMPUTER DEVICE, AND STORAGE MEDIUM

(57) A backend architecture method and system for aggregate payment, a computer device, and a storage medium are disclosed. The method includes: acquiring transaction information from a client, wherein the transaction information includes location information of the client; assigning, based on the location information, the transaction information to a transaction node that has jurisdiction over the location information, wherein the transaction node is a transaction center within a specified region in a distributed transaction system; matching, according to the transaction node, a channel gateway in a channel docking layer that pre-establishes a connection relationship with the transaction node, and sending a transaction request in the transaction information to a transaction object over a selected channel gateway such that the transaction object processes the transaction request; and acquiring a processing result from the transaction object, and transmitting information of the processing result to the client. According to the present disclosure, transaction data is processed by distributed transaction nodes, such that data processing for payment is faster. In addition, transaction centers deployed in a distributed fashion are independent of each other, thereby achieving strong data isolation, low node coupling, and strong system expandability.

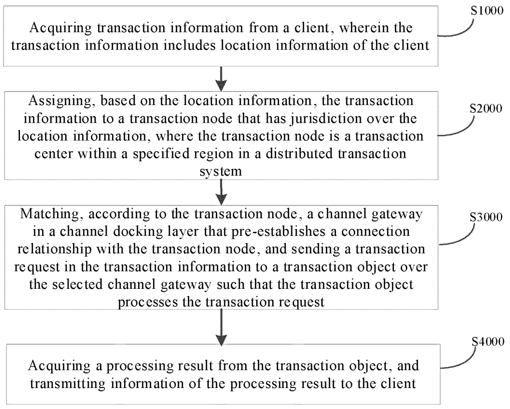
S1000
Acquiring transaction information from a client, wherein the transaction information includes location information of the client
S2000
Assigning, based on the location information, the transaction information to a transaction node that has jurisdiction over the location information, where the transaction node is a transaction center within a specified region in a distributed transaction system
S3000
Matching, according to the transaction node, a channel gateway in a channel docking layer that pre-establishes a connection relationship with the transaction node, and sending a transaction request in the transaction information to a transaction object over the selected channel gateway such that the transaction object processes the transaction request
S4000
Acquiring a processing result from the transaction object, and transmitting information of the processing result to the client

FIG. 1

Printed by Jouve, 75001 PARIS (FR)

Description

TECHNICAL FIELD

[0001] The present disclosure relates to the technical field of computer application, and in particular, relates to a backend architecture method and system for aggregate payment, a computer device, and a storage medium.

BACKGROUND

[0002] With the rapid development of the Internet of things and the Mobile Internet, the electronic payment industry has also been developing rapidly in recent years. Domestically, banks, UnionPay and other clearinghouses, third-party payment companies, and the like provide different types of mobile payment services implemented with different technologies. Moreover, due to the objective demand, the electronic payment technology has been applied in various fields. However, under this context, no unified standard and interface are provided for services from various parties, such that it is difficult to achieve compatibility between various payment tools and platforms. Globally, each country has its own channel providers and access control rules. For merchants with global businesses, the access cost and access efficiency are also greatly affected.

[0003] Only a few third-party payment channels are used in China. Generally, more than 90% of domestic online payment transactions can be covered by accessing WeChat or Alipay for merchants. Therefore, when accessing payment channels, most merchants are directly connected to third-party channels. However, payment processes over various payment channels (such as WeChat and Alipay) differ in detail, such that a plurality of sets of payment processes generally need to be supported at the backend. The domestic Internet industry is on the way to going abroad. There are many different payment channels with different processes overseas. If merchants directly connect to third-party payment channels using the conventional method, the merchants need to handle the complex task of maintaining the plurality of sets of payment processes.

SUMMARY

[0004] The present disclosure provides a backend architecture method and system for aggregate payment, a computer device, and a storage medium, to solve at least one of the aforementioned technical defects. The present disclosure can achieve high isolation, low node coupling, fast access, and high practicability and flexibility in a third-party payment process.

[0005] To achieve the above objective, the present disclosure provides a backend architecture method for aggregate payment. The method includes: acquiring transaction information from a client, wherein the transaction information includes location information of the client; assigning, based on the location information, the transaction information to a transaction node that has jurisdiction over the location information, wherein the transaction node is a transaction center within a specified region in a distributed transaction system; matching, according to the transaction node, a channel gateway in a channel docking layer that pre-establishes a connection relationship with the transaction node, and sending a transaction request in the transaction information to a transaction object over a selected channel gateway such that the transaction object processes the transaction request; and acquiring a processing result from the transaction object, and transmitting information of the processing result to the client.

[0006] According to another aspect, the present disclosure provides a backend architecture system for aggregate payment. The system includes: an access frontend, configured to acquire transaction information from a client, wherein the transaction information includes location information of the client; a relay server, configured to assign, based on the location information, the transaction information to a transaction node that has jurisdiction over the location information, wherein the transaction node is a transaction center within a specified region in a distributed transaction system; and a transaction node, configured to match, according to the transaction node, a channel gateway in a channel docking layer that pre-establishes a connection relationship with the transaction node, send a transaction request in the transaction information to a transaction object over a selected channel gateway such that the transaction object processes the transaction request, acquire a processing result from the transaction object, and transmit information of the processing result to the client.

[0007] According to another aspect, the present disclosure provides a computer device. The computer device includes a memory and a processor. The memory stores at least one computer-readable instruction. The processor, when executing at least one computer-readable instruction, is caused to perform the backend architecture method for aggregate payment as described above.

[0008] According to another aspect, the present disclosure provides a storage medium storing at least one computer-readable instruction. The least one computer-readable instruction, when executed by one or more processors, causes the one or more processors to perform the backend architecture method for aggregate payment as described above.

[0009] The present disclosure achieves the following beneficial effects.

[0010] Based on the location information of the client in the transaction information, distributed processing is performed, such that data processing for payment is faster. Moreover, transaction centers deployed in a distributed fashion are independent of each other and use unified communication interfaces to connect with transaction objects, which shield differences in details of the payment processes of different channels, and provide a universal and unified payment access process for the clients. Therefore, to use all payment channels supported by the payment architecture, a merchant on the client side only needs to maintain one set of processes for interacting with the aggregate payment architecture, thus achieving strong data isolation, low node coupling, strong anti-attack capability, and low service coupling between clients. The aggregate payment architecture is more suitable for distributed deployment, and has high system availability and strong disaster tolerance. The unified scheduling interface makes the system highly expandable and reduces impact of functional iteration on online services.

BRIEF DESCRIPTION OF THE DRAWINGS

[0011] The foregoing and/or additional aspects of the present disclosure and advantages of the present disclosure can become apparent and readily understandable from the following descriptions of the embodiments with reference to the accompanying drawings.

FIG. 1 is a flowchart of a backend architecture method for aggregate payment according to the present disclosure;

FIG. 2 is a flowchart of a method for authenticating transaction information according to the present disclosure;

FIG. 3 is a flowchart of a method for assigning transaction information to a transaction node according to the present disclosure;

FIG. 4 is a flowchart of a transaction requesting method of a transaction node according to the present disclosure;

FIG. 5 is a flowchart of a retry method of an asynchronous task queue according to the present disclosure;

FIG. 6 is an overall flowchart of aggregate payment according to the present disclosure;

FIG. 7 is a schematic diagram of a multilayer structure of aggregate payment according to the present disclosure;

FIG. 8 is a flowchart of receiving information by modules of aggregate payment according to the present disclosure; and

FIG. 9 is a block diagram of a basic structure of a computer device according to the present disclosure.

DETAILED DESCRIPTION

[0012] Specifically, referring to FIG. 1, a backend architecture method for aggregate payment is provided in the present disclosure. Aggregate payment is relative to third-party payment. Unlike third-party payment, aggregate payment does not lie between merchants and banks, but serves as a channel between merchants and third-party payers. Aggregate payment does not involve settlement or transfer of funds, but merely performs the control of information flows and data flows in the payment link, and aggregates payment methods provided by different third-party payment institutions, to provide merchants with a unified interface for payment and settlement. Aggregate payment reduces the access difficulty and cost for merchants, improves the speed and efficiency of operation, and achieves better flexibility and convenience.

[0013] In the present disclosure, the backend architecture method for aggregate payment includes the following processes:

[0014] In S1000, transaction information from a client is acquired, wherein the transaction information includes location information of the client.

[0015] The client is actually a merchant terminal. There are a plurality of clients, and the clients are capable of sending request instructions at the same time for analysis. A request instruction is a related data message that records identity information of a client and request content. The data message is a field combination in an agreed format to facilitate information interaction. In order to distinguish different clients, the data message includes merchant number information that characterizes the identity of the client and service number information that characterizes a service type of the request, and thus the identity information of the client sending the request and the request content can be confirmed.

[0016] The transaction information is related information that is sent by the client and that the client requests to be processed by a third party. The transaction information includes location information of the client.

[0017] In another embodiment, the transaction information further includes location information of a transaction object to which the transaction information is directed. In the present disclosure, the transaction object is a third party information requested by the transaction client to perform transaction processing, and the location information of the transaction object herein refers to location information of the third party. It may be understood that the third party is a payment platform. There are many payment platforms, such as famous banks, WeChat, Alipay and other financial platforms. These third-party platforms have corresponding server location information. The location information includes nationality and a specific region. By acquiring the location information of the transaction object, server location information of the transaction object can be acquired.

[0018] In S2000, the transaction information is assigned, based on the location information, to a transaction node that has jurisdiction over the location information, wherein the transaction node is a transaction center within a specified region in a distributed transaction system.

[0019] The transaction node is a core processing layer performing service processing. Creation of service orders and service processing logics are achieved at the core processing layer. In one embodiment, when the location information is the location information of the client, the transaction node is a transaction center within a specified region in the distributed transaction system. Since the clients may be distributed in various locations around the world, for quick responses to payment requests, a transaction node may be configured in any place around the world. Based on the acquired location information of the client, a transaction node having jurisdiction over the location information is selected for processing.

[0020] In another embodiment, the location information is the location information of the transaction object to which the transaction information is directed, i.e., the location information of the third-party transaction platform. The third-party transaction platform may be deployed at various locations around the world, and the aggregate payment clients of the present disclosure may also be distributed around the world. Therefore, in order to achieve rapid response to the payment request, a plurality of transaction nodes can be configured based on regional locations of the third-party transaction platform. The transaction nodes are independent and isolated from each other. When one of the transaction nodes is abnormal, normal service processing of other transaction nodes is not affected. Further, each transaction node has its own jurisdiction. A new request instruction can be assigned to a transaction node having corresponding jurisdiction for processing, by identifying location information of a transaction object and a service type that correspond to the new request instruction. However, in order to further improve the disaster tolerance of service processing, whether the corresponding transaction node is in an available state when the request instruction is assigned to the corresponding transaction node is determined. When the transaction node is determined to be in an unavailable state, the request instruction is assigned to another available transaction node to ensure the normal processing of the request instruction of the transaction information.

[0021] In S3000, a channel gateway is matched, according to the transaction node, in a channel docking layer that pre-establishes a connection relationship with the transaction node, and a transaction request in the transaction information is sent to a transaction object over the selected channel gateway such that the transaction object processes the transaction request.

[0022] Once the transaction information is assigned to the transaction node, a channel gateway is matched, according to information of the transaction node, in a channel docking layer that has established a connection relationship with the transaction node. A plurality of different channel gateways are stored in the channel docking layer, and different channel gateways are connected to and communicate with different third-party transaction objects.

[0023] The channel gateways are connected to and communicate with transaction nodes by scheduling interfaces. In the present disclosure, the channel gateways can achieve the logic for interacting with the third-party transaction objects, and interaction details between different channels are shielded from an upstream node. In the present disclosure, the scheduling interfaces of the channel gateways are unified, to maintain a consistent transaction process that does not change due to changes of channel objects. Further, each channel in the channel gateway layer is logically independent and does not manipulate order data. Changes in one channel have little impact on other channels and the main process, such that fast access and iteration can be achieved. The unified scheduling interface also facilitate access to a new channel. The new channel can be accessed by simply building a corresponding channel gateway node. Specific differences and details of the interaction with a third-party transaction object are concealed in the channel gateway.

[0024] In the service process of payment, the channel gateways are application interfaces connecting to different third-party payment platforms. Different third-party payment platforms require different channel gateways, thus forming a channel gateway connection library, i.e., the channel docking layer. By identifying transaction objects in request instructions, different channel gateways can be matched from the channel gateway connection library for service processing.

[0025] In an embodiment, each channel gateway interacts with a third-party channel using a unified scheduling interface, interaction details between different channels are shielded from an upstream node, and the connection of each channel forms an independent submodule. Therefore, when relevant service processing needs to be performed, a corresponding interface can be called directly. When a new channel is accessed, it is only necessary to build a channel gateway node that provides a standard and unified interface. Specific differences and details of docking with the third party are concealed in the channel gateway. In this way, an upper core service layer can call a unified interface and maintain a consistent transaction process that does not change with changes in channels; and further, each channel in the channel gateway is logically independent and does not manipulate order data. Changes in one channel have little impact on other channels and the main process, thereby achieving rapid access and iteration.

[0026] In S4000, a processing result from the transaction object is acquired, and information of the processing result is transmitted to the client.

[0027] In the present disclosure, when the transaction request in the transaction information is sent to the corresponding transaction object such that the transaction object processes the transaction request by calling the corresponding communication interface, the transaction result returned by the transaction object is acquired, and information of the transaction result is sent back to the client.

[0028] In the present disclosure, the data reception and assignment in processes S1000 and S2000 are performed in a service access layer. The service access layer includes an access frontend and a relay server. The access frontend receives the transaction information from the client, converts the transaction information into a protocol format recognizable by the backend, and completes tasks such as identity verification, authorization control, and traffic isolation. Afterwards, the relay server performs load balancing and task assignment, and sends, based on a distribution policy, the request information in the transaction information to a specified transaction node for processing. Transaction information monitoring and result information collection are performed by a payment node by calling a communication interface, wherein the payment node belongs to the core service layer. The channel gateways and the unified scheduling interfaces belong to the channel docking layer. The core service layer mainly includes a payment center, payment notification and asynchronous retry, payment monitoring, and the like. The payment center completes the management and recording of payment orders, and calls services of the channel gateways in the next layer. The asynchronous retry is responsible for the replay of asynchronous tasks to ensure the completion of orders. The payment monitoring is responsible for the monitoring and statistics of payment orders. The payment notification is responsible for asynchronous notification to the client in response to the order being completed. The channel docking layer is mainly responsible for connection with services provided by the third-party objects. The connection of each channel forms an independent sub-module, which facilitates channel management and rapid iteration.

[0029] In the present disclosure, distributed processing is performed based on the location information of the transaction object in the transaction information, such that data processing for payment is faster. Moreover, transaction centers deployed in a distributed fashion are independent of each other and use unified communication interfaces to connect with transaction objects, to shield differences in details of the payment processes in different channels. Therefore, a universal and unified payment access process is provided for the clients. To use all payment channels supported by the payment architecture, a merchant on the client side only needs to maintain one set of processes for interacting with the aggregate payment architecture, thus achieving strong data isolation, low node coupling, strong anti-attack capability, and low service coupling between clients. The aggregate payment architecture is more suitable for distributed deployment, and has high system availability and strong disaster tolerance. The unified scheduling interface makes the system highly expandable and reduces the impact of the functional iteration on online services.

[0030] In an embodiment, referring to FIG. 2, in response to acquiring transaction information from a client and before assigning, based on the location information, the transaction information to a transaction node that has jurisdiction over the location information, the method further includes the following processes.

[0031] In S1100, verification information of the transaction information is verified, wherein the verification information is set using an encryption communication protocol.

[0032] In the present disclosure, the transaction information initiated by the client is provided with verification information using an encryption communication protocol, to ensure that the communication is not tampered with by a third party. In response to receiving the transaction information, the access frontend verifies the verification information. During data interaction, the client and the access frontend adopt a uniform information verification method, such as encrypting data using a public key and a private key. In response to receiving the data, the access frontend decrypts the data on the basis of the rule predetermined by the client and the access frontend, thereby ensuring that the data is not easily tampered with in the process of data interaction between the client and the access frontend, and making data transmission more secure.

[0033] Further, as the forefront of service reception, the access frontend can further perform digital signature verification on message data of the received transaction information to ensure that the message of the request is from an authorized merchant and is not tampered with or forged. Then, only the transaction information passing the signature verification can be forwarded subsequently. Upon detecting that the acquired transaction data is tampered with or the acquired transaction data is in other abnormal conditions, the access frontend does not perform subsequent processing, and transmits the detected result information to the client to indicate that the currently transmitted data is not secure.

[0034] In S 1200, identity information of the client sending the transaction information and a service type are acquired by parsing the transaction information that is successfully verified.

[0035] When the signature of the transaction data is verified by the access frontend, data of the current transaction information is considered to be secure. Therefore, the transaction information can be parsed to obtain the identity information of the client sending the transaction information as well as the service type. In this embodiment, the identity information of the client can be extracted from an IP address at which the transaction information is sent and from related fields in the header of a data frame in the information, while the service type can be acquired from a text data frame characterizing content among the transmitted data frame.

[0036] In S1300, matching is performed in an identity authority list based on the parsed identity information and the service type for authentication.

[0037] When the identity information of the client and the service type are parsed out, the client is authenticated based on the information. In the present disclosure, the authentication is mainly for identifying whether the client has the related authority to perform the related requested service. Therefore, in the present disclosure, an identity authority list is also stored in the access frontend. The identity authority list lists identity information and corresponding authority of one or more service types of all qualified clients. The identity information and the service type are one-by-one corresponded in the identity authority list, and therefore whether the client has the authority to initiate a data processing request of the service type is determined.

[0038] Further, when it is determined that the client does not have the relevant authority by authentication, exception information indicating authentication failure is sent to the client as a reminder. For example, in an embodiment, by parsing the transaction information from a client A, it is known that the service types that client A intends to process include service types B and C. In the identity authority list, the service types that can be processed by the client A include the service type C, excluding the service type B. Therefore, requests related to the service type C can be processed subsequently. For the service type B, exception information is sent to the client A to indicate that the client has no authority to process the service type B.

[0039] Further, a rule for determining whether the authentication succeeds or not can also be preset by the access frontend. For example, as described in the above embodiment, the client A has authority for one of the service type B and the service type C, but has no authority for the other service types. In this case, two rules can be set. One is that as long as any one of the service types meets the authority requirement, subsequent operations for the service are allowed to be performed. The other is that as long as the request of any one of the service types does not meet the authority requirement, none of the requests in the transaction information is responded to, and exception information is sent to the client such that the client can modify the transaction information. The specific rule may be set according to the actual situation.

[0040] In S1400, the transaction information passing the authentication is assigned to the transaction node.

[0041] When the transaction information passes the authentication using a predetermined rule, the relay server can assign the transaction information to the corresponding transaction node based on the identified location information.

[0042] In response to receiving the transaction information, the relay server provides a forwarding service for the transaction information, wherein a forwarding rule may be determined by analyzing the specific content of the transaction information, thereby ensuring that a same payment request message can be processed at a same node.

[0043] Further, in an embodiment, referring to FIG. 3, the process of assigning, based on the location information, the transaction information to the transaction node that has jurisdiction over the location information includes the following sub-processes.

[0044] In S2100, an assignment rule is matched in an assignment list based on a service type of the transaction information, such that requests for a same service type from a same client are processed by a same transaction node.

[0045] The assignment list is a mapping table between relevant information and corresponding assignment rules, wherein the relevant information includes identity information of clients sending the transaction information, and service types, and the like. The relay server assigns the transaction information based on the identified service type and location information of the transaction object.

[0046] The transaction object is a third-party payment platform that is identified based on the transaction information and needs to perform service processing, such as a bank end, a WeChat end, or an Alipay end. In an embodiment, the location information is location information of the client. Thus, when the service type is acquired, the transaction information is assigned to a transaction node with jurisdiction based on the location information of the client, such that the transaction information can be processed.

[0047] In another embodiment, the location information is location information of the service type of the transaction. For example, location information of the WeChat end and Alipay end disclosed above is China since WeChat and Alipay are both domestic payment platforms in China. Some of the bank ends may be in foreign countries. For example, Standard Chartered Bank, Citibank, and the like are foreign payment platforms. Transaction nodes are deployed based on server locations of these payment platforms. For example, transaction information, of which payment objects are the five largest domestic banks: Industrial and Commercial Bank of China, Agricultural Bank of China, Bank of China, China Construction Bank, and Bank of Communications, is set to be processed by a transaction node A. Transaction information corresponding to other domestic banks is set to be processed by a transaction node B. Transaction information corresponding to foreign banks is set to be processed by a transaction node C. Transaction information corresponding to non-bank third-party payment platforms, such as WeChat and Alipay, is set to be processed by a transaction node D. Thus, when the transaction object and the corresponding location information is parsed out from the transaction information sent by the client, the transaction information can be assigned based on the location information.

[0048] Further, the above classification is not completely limited. Since the transaction nodes are independent of each other, in order to ensure the normal processing of each piece of transaction information, the transaction information is assigned to nodes based on the transaction objects. In response to a problem occurring in one of the transaction nodes, the transaction information that should be processed by the problematic transaction node can be transferred to other transaction node according to a rule, such that the transaction information can be processed in time. The other transaction node to which the transaction information is transferred according to the rule herein may be a transaction node that processes the same type of transaction information as the original transaction node, or a transaction node configured with the same and available channel gateway.

[0049] Further, by identifying the service type and the identity information of the client, whether a plurality of pieces of transaction information belong to a same service is determined. In response to determining that the plurality of pieces of transaction information belong to the same service, all transaction information belonging to the same service is grouped into one transaction node for transaction processing.

[0050] Further, the relay server routes the transaction information to the corresponding transaction node by hashing (HASH). Hashing can transform an arbitrary-length input into a fixed-length output by using a hashing algorithm, wherein a value of the output is a hash value. In the present disclosure, hashing refers to routing the data message of the transaction information to the corresponding transaction node based on a specific hashing algorithm or rule. In this way, the security of the data during the matching process is ensured.

[0051] In S2200, information of a usage state of a mapped transaction node is acquired according to the assignment rule.

[0052] When a corresponding transaction node is matched by process S2100, the transaction information is not directly assigned to the transaction node. Instead, the information of the usage state of the transaction node is acquired first. The information of the usage state is a current traffic ratio and operation parameters of the transaction node. By acquiring the traffic ratio, whether traffic ratio of the current transaction node exceeds a rating value is determined. When the traffic ratio exceeds the rating value, the current transaction node is oversaturated, which may affect the speed of transaction data processing. By identifying the operation parameters, whether the current transaction node is running normally and is available or not is determined. When the operation parameters are abnormal, the current transaction node is not available and the transaction data needs to be transferred.

[0053] In S2300, whether the usage state of the transaction node meets a first predetermined condition is determined.

[0054] The first predetermined condition is a standard reference condition set based on the usage state information. For example, when the detected usage state is the traffic ratio of the transaction node, the first predetermined condition is set as that the traffic ratio of the transaction node is not higher than 85%. When the detected traffic ratio of the current transaction node is 80%, the usage state of the current transaction node meets the first predetermined condition. Otherwise, it is defined that the current transaction node does not meet the preset condition.

[0055] The description above is only one example of the usage state and the first predetermined condition in the present disclosure, and the usage state may be determined in other fashions in the present disclosure.

[0056] In S2400, the transaction information is assigned to the transaction node in response to the usage state of the transaction node meeting the first predetermined condition.

[0057] When it is determined that the transaction node meets the first preset condition, the transaction information is assigned to the corresponding transaction node based on the identified location information.

[0058] In S2500, the assignment rule is re-matched to re-determine the transaction node in response to the usage state of the transaction node not meeting the first predetermined condition.

[0059] When it is determined that the current usage state of the transaction node does not meet the first predetermined condition, the matching assignment rule is re-matched. The re-match of the assignment rule herein is performed in the same fashion as transferring the transaction information to other transaction node according to a rule in process S2100. The re-matched transaction node may be a transaction node that processes the same type of transaction information as the original transaction node, or a transaction node configured with the same and available channel gateway.

[0060] The way of identifying the usage state is in fact a traffic cutting method, which is implemented by the relay server. Since all types of transaction information messages are routed through the relay server to the corresponding nodes for processing, the configuration of a forwarding rule can be modified at the relay server to achieve traffic cutting of services.

[0061] In response to receiving the transmitted transaction information, the transaction center processes the transaction information. In an embodiment, referring to FIG. 4, the method for processing the transaction request by the transaction object includes following sub-processes.

[0062] In S3100, a unique order number is created based on the transaction request.

[0063] In response to acquiring the transaction information, the transaction center needs to advise a unique order number for the transaction information, to facilitate data processing and data monitoring. Since the identity information of the client, the service type, and the location information of the service object is identified in the foregoing service access layer, an order number can be generated based on the information and the service type according to a rule. Since the client with the same identity information may request the same service object to process a plurality of requests of a same service type, for differentiation, each transaction request in the transaction information may be matched with a unique order number according to a rule to facilitate data traceback.

[0064] In an embodiment, unique order numbers may be automatically generated based on a timeline. For example, starting with 0001, order numbers of transaction requests of transaction information received subsequently are 0002, 0003, and the like in order. Transaction information received at the same time may be randomly sequenced and order numbers may be sequenced for differentiation.

[0065] In another embodiment, the order number can be generated by embedding the identity information of the client, the service type and the transaction object information. For example, the order number is composed of the initials or ID that characterizes the identity information of the client, a serial number of the service type, a serial number of the transaction object, and a sequence number that is set based on the timeline. Orders with same client identity information, same service type, and same transaction object information can be distinguished according to the order number, thereby facilitating data query and traceback.

[0066] Further, in order to ensure the security of the order number, a randomly generated check code can be embedded into the order number to improve the security of the order.

[0067] Further, when the order is generated, the order number and the relevant data parsed out from the transaction information are stored in a database for retrieval at any time.

[0068] In S3200, the transaction request corresponding to the order number is processed in the transaction object by calling a communication interface.

[0069] A plurality of channel gateways are deployed in the transaction node. The scheduling interface between the transaction nodes and channel gateways is unified. Different channel gateways are docked to processing servers of different transaction objects. The channel gateway and the scheduling interface form a communication interface. When the order number is generated for the transaction information, a processing procedure of the corresponding transaction information can be acquired by monitoring the order number. The transaction center sends, through the communication interface, the transaction request in the transaction information to the server of the transaction object such that the transaction object processes the transaction request, and at the same time receives process information, such as a time point at which the transaction information is sent to the server of the transaction object, a process state of the transaction processing, a final result state, a total time spent, and the like.

[0070] In S3300, processing data of the transaction request is written into an asynchronous task queue for buffering.

[0071] In an embodiment, the transaction center writes the relevant processing data of the transaction acquired from the communication interface into the asynchronous task queue for buffering. A plurality of different work nodes are configured in a queue in the asynchronous task queue according to the processes of data processing. In response to receiving relevant data of each step, the information data is stored to a corresponding node, and therefore the current process and state of the processing can be determined.

[0072] In S3400, a retry is performed in response to an exception of a task in the asynchronous task queue during processing.

[0073] In an embodiment, when an exception occurs in any link in the asynchronous queue, the asynchronous task queue automatically sends information to the transaction node to prompt the transaction node to retry the link. Referring to FIG. 5, the process of retry includes the following sub-processes.

[0074] In S3410, a retry count is acquired.

[0075] In an embodiment, in order to save time and control the speed of processing, a maximum count value is set, which is called a second threshold. Therefore, when entering a retry, each time a sequential retry operation is performed, the current count is recorded and compared with the second threshold.

[0076] When the exception is resolved by a retry or retries within the second threshold, subsequent tasks in the asynchronous task queue is sequentially executed.

[0077] In S3420, the asynchronous task queue is deleted in response to the retry count reaching a second threshold, and exception information is fed back to the client.

[0078] When the retry count reaches the second threshold and the exception is still not resolved, information is sent to the transaction node such that the transaction node does not send any more retry tasks to request retries. At the same time, the asynchronous task queue is deleted, and the exception information is fed back to the client. The feedback information is also stored in the database.

[0079] In S3500, the buffered asynchronous task is deleted upon completion of all the tasks in the asynchronous task queue.

[0080] When the data exception is resolved upon a retry or retries, other tasks in the asynchronous task queue are performed until all tasks are completed. Upon completion of all the tasks in the asynchronous task queue, the current transaction information is completely processed. Then, the buffered asynchronous task queue is deleted, to release the data space.

[0081] The foregoing method of the present disclosure forms an aggregate payment system. In a complete payment transaction request, as shown in FIG. 6, the client sends transaction information with a payment request to the aggregate payment system; the aggregate payment system communicates with a third-party transaction object through a communication interface; the third-party transaction object returns ordering data to the aggregate payment system; the aggregate payment system extracts a payment link based on the ordering data and sends the payment link to the client. The client accesses the payment link to complete payment. In response to acquiring a payment result, the third-party transaction object returns the callback payment result to the aggregate payment system. The aggregate payment system generates a payment result notification according to its own rule, and returns the notification to the client to indicate the payment result.

[0082] In an embodiment, as shown in FIG. 7, the aggregate payment system includes a service access layer, a core service layer, and a channel docking layer. The service access layer includes an access frontend and a relay server. The core service layer includes a transaction node. The transaction node includes a database for storing data, an asynchronous module for buffering an asynchronous task queue and performing retries, a payment notifying module for sending relevant data acquired in the transaction node to the client, and a payment monitoring module for monitoring data processing of the entire transaction node. The transaction node is connected, through a unified scheduling interface, to each channel gateway in the channel docking layer, so as to connect to different transaction payment platforms for transaction processing according to different third-party transaction objects.

[0083] Specifically, the backend architecture system for aggregate payment in the present disclosure includes following functions and module relationships:

an access frontend, configured to acquire transaction information from a client, wherein the transaction information includes location information of a transaction object to which the transaction information is directed;
a relay server, configured to assign, based on the location information, the transaction information to a transaction node that has jurisdiction over the location information, wherein the transaction node is a transaction center within a specified region in a distributed transaction system; and
a transaction node, configured to send, based on a communication interface pre-established between the transaction node and the transaction object, a transaction request in the transaction information to the transaction object such that the transaction object processes the transaction request, acquire a processing result from the transaction object, and transmit information of the processing result to the client.

[0084] Optionally, the access frontend further includes:

a verifying module, configured to verify verification information of the transaction information, wherein the verification information is set using an encryption communication protocol;
a data parsing module, configured to acquire identity information of the client sending the transaction information and a service type by parsing the transaction information that is successfully verified;
an authenticating module, configured to match in an identity authority list based on the parsed identity information and the service type for authentication; and assign the transaction node to the transaction information that passing the authentication.

[0085] Optionally, the access frontend further includes:

a traffic recognizing module, configured to acquire a traffic value of transaction information passing the authentication in a first predetermined time period; and
a selecting module, configured to allow assignment of transaction information of which the traffic value is less than or equal to a first predetermined threshold.

[0086] Optionally, the relay server includes:

an assignment rule matching module, configured to match, based on a service type of the transaction information, an assignment rule in an assignment list, such that requests for a same service type from a same client are processed by a same transaction node;
a state acquiring module, configured to acquire, according to the assignment rule, information of a usage state of a mapped transaction node;
a determining module, configured to determine whether the usage state of the transaction node meets a first predetermined condition; and
an assigning module, configured to assign the transaction information to the transaction node in response to the usage state of the transaction node meeting the first predetermined condition.

[0087] Optionally, the relay server further includes a re-matching module, configured to re-match the assignment rule to re-determine the transaction node in response to the usage state of the transaction node not meeting the first predetermined condition.

[0088] Optionally, the transaction node includes:

an order generating module, configured to create a unique order number based on the transaction request;
an calling module, configured to process, in the transaction object, the transaction request corresponding to the order number by calling a communication interface;
an asynchronous module, configured to write processing data of the transaction request into an asynchronous task queue for buffering; and perform a retry in response to an exception of a task in the asynchronous task queue during processing; and
a first deleting module, configured to delete the buffered asynchronous task queue in response to all tasks in the asynchronous task queue being completed.

[0089] Optionally, the transaction node further includes:

a retry counting module, configured to acquire a retry count; and
a second deleting module, configured to delete the asynchronous task queue in response to the retry count reaching a second threshold, and feed back exception information to the client.

[0090] In an embodiment, as shown in FIG. 8, the specific working processes of the modules disclosed above are as follows: the client sends transaction information; the access frontend receives the transaction information, and sends the transaction information to the relay server upon authentication and traffic control; the relay server assigns the transaction information to a corresponding transaction node according to an assignment rule; in response to receiving the related transaction information, the transaction node creates an order number, and stores the related data in a database; at the same time, the transaction node calls a communication interface of a related channel gateway according to relevant information about the transaction object analyzed above; the channel gateway communicates with the third-party transaction object; the transaction object returns ordering data; the channel gateway receives the ordering data and transmits the ordering data to the transaction node; the transaction node extracts a payment link from the ordering data and sends the payment link to the client; the client accesses the payment link to perform a transaction payment directly; in response to determining that the transaction succeeds, the transaction object sends an order callback notification to the access frontend; when the access frontend authenticates relevant data, the relay server assigns, according to an assignment rule, the relevant data to the original transaction node; and the transaction node identifies the validity of the callback notification, and in response to the callback notification being identified as valid, the transaction node instructs the payment notifying module to generate a payment result notification, and constructs a data message to directly return the payment result notification to the corresponding client, to indicate the payment result.

[0091] FIG. 9 shows a block diagram of a basic structure of a computer device according to an embodiment of the present disclosure.

[0092] The computer device includes a processor, a non-transitory storage medium, a memory, and a network interface, wherein the processor, the non-transitory storage medium, the memory, and the network interface are connected via a system bus. The non-transitory storage medium of the computer device stores an operating system, a database, and at least one computer-readable instruction. The database stores a control information sequence. The at least one computer-readable instruction, when executed by the processor, cause the processor to perform a backend architecture method for aggregate payment. The processor of the computer device is configured to provide computing and control capabilities, to support running of the entire computer device. The memory of the computer device may store at least one computer-readable instruction. The at least one computer-readable instruction, when executed by the processor, cause the processor to perform a backend architecture method for aggregate payment. The network interface of the computer device is configured to connect and communicate with a terminal.

[0093] The computer device receives status information of a prompt behavior from an associated client, i.e., whether the associated terminal turns on the prompt function and whether the user turns off the prompt task. By verifying whether the above task conditions are achieved, the computer device sends a corresponding predetermined instruction to the associated terminal such that the associated terminal can perform a corresponding operation according to the predetermined instruction, thereby achieving effective supervision over the associated terminal. At the same time, when the status of the prompt information is different from a predetermined status instruction, the server side controls the associated terminal to continuously ring, to prevent the prompt task of the associated terminal from being automatically terminated after being executed for a period of time.

[0094] The present disclosure further provides a storage medium storing at least one computer-readable instruction. The at least one computer-readable instruction, when executed by one or more processors, causes the one or more processors to perform the backend architecture method for aggregate payment described in any of the foregoing embodiments.

Claims

1. A backend architecture method for aggregate payment, comprising:

   acquiring transaction information from a client, wherein the transaction information comprises location information of the client;
   assigning, based on the location information, the transaction information to a transaction node that has jurisdiction over the location information, wherein the transaction node is a transaction center within a specified region in a distributed transaction system;
   matching, according to the transaction node, a channel gateway in a channel docking layer that pre-establishes a connection relationship with the transaction node, and sending a transaction request in the transaction information to a transaction object over a selected channel gateway such that the transaction object processes the transaction request; and
   acquiring a processing result from the transac-

tion object, and transmitting information of the processing result to the client.

2. The method according to claim 1, wherein before assigning, based on the location information, the transaction information to the transaction node that has jurisdiction over the location information, the method further comprises:

verifying verification information of the transaction information, wherein the verification information is set using an encryption communication protocol;
acquiring identity information of the client sending the transaction information and a service type by parsing the transaction information that is successfully verified;
matching in an identity authority list based on the parsed identity information and the service type for authentication; and
assigning the transaction node to the transaction information passing the authentication.

3. The method according to claim 2, wherein before the process of assigning the transaction node to the transaction information that passing the authentication, the method further comprises:

acquiring a traffic value of transaction information passing the authentication in a first predetermined time period; and
allowing assignment of transaction information of which the traffic value is less than or equal to a first predetermined threshold.

4. The method according to claim 2, wherein assigning, based on the location information, the transaction information to the transaction node that has jurisdiction over the location information comprises:

matching, based on the service type of the transaction information, an assignment rule in an assignment list, such that requests for a same service type from a same client are processed by a same transaction node;
acquiring, according to the assignment rule, information of a usage state of a mapped transaction node;
determining whether the usage state of the transaction node meets a first predetermined condition; and
assigning the transaction information to the transaction node in response to the usage state of the transaction node meeting the first predetermined condition.

5. The method according to claim 4, wherein in response to the usage state of the transaction node not meeting the first predetermined condition, the assignment rule is re-matched to re-determine the transaction node.

6. The method according to claim 1, wherein processing the transaction request by the transaction object comprises:

creating a unique order number based on the transaction request;
processing, in the transaction object, the transaction request corresponding to the order number by calling a communication interface;
writing processing data of the transaction request into an asynchronous task queue for buffering;
performing a retry in response to an exception of a task in the asynchronous task queue during processing; and
deleting the buffered asynchronous task queue in response to all tasks in the asynchronous task queue being completed.

7. The method according to claim 6, further comprising:

acquiring a retry count; and
deleting the asynchronous task queue in response to the retry count reaching a second threshold, and feeding back exception information to the client.

8. A backend architecture system for aggregate payment, comprising:

an access frontend, configured to acquire transaction information from a client, wherein the transaction information comprises location information of the client;
a relay server, configured to assign, based on the location information, the transaction information to a transaction node that has jurisdiction over the location information, wherein the transaction node is a transaction center within a specified region in a distributed transaction system; and
a transaction node, configured to match, according to the transaction node, a channel gateway in a channel docking layer that pre-establishes a connection relationship with the transaction node, send a transaction request in the transaction information to a transaction object over a selected channel gateway such that the transaction object processes the transaction request, acquire a processing result from the transaction object, and transmit information of the processing result to the client.

9. A computer device, comprising a memory and a processor, wherein the memory stores at least one computer-readable instruction, wherein the at least one computer-readable instruction, when executed by the processor, causes the processor to perform the backend architecture method for aggregate payment as defined in any one of claims 1 to 7.

10. A storage medium storing at least one computer-readable instruction, wherein the at least one computer-readable instruction, when executed by one or more processors, causes the one or more processors to perform the backend architecture method for aggregate payment as defined in any one of claims 1 to 7.

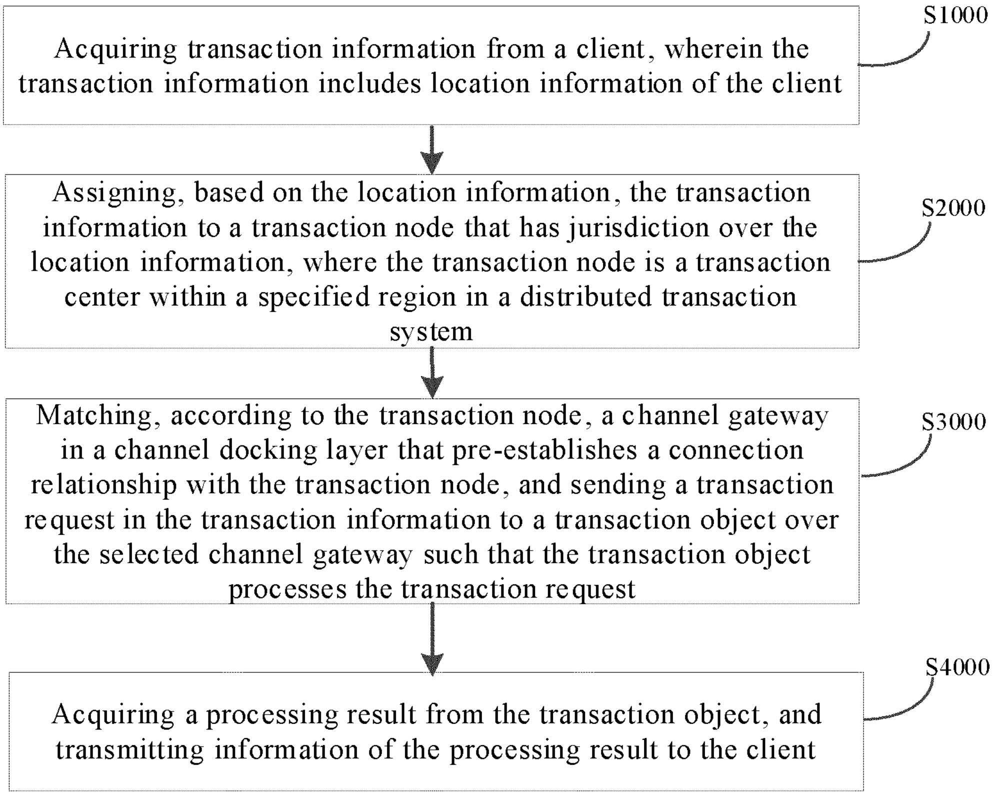
S1000
Acquiring transaction information from a client, wherein the transaction information includes location information of the client
S2000
Assigning, based on the location information, the transaction information to a transaction node that has jurisdiction over the location information, where the transaction node is a transaction center within a specified region in a distributed transaction system
S3000
Matching, according to the transaction node, a channel gateway in a channel docking layer that pre-establishes a connection relationship with the transaction node, and sending a transaction request in the transaction information to a transaction object over the selected channel gateway such that the transaction object processes the transaction request
S4000
Acquiring a processing result from the transaction object, and transmitting information of the processing result to the client

FIG. 1

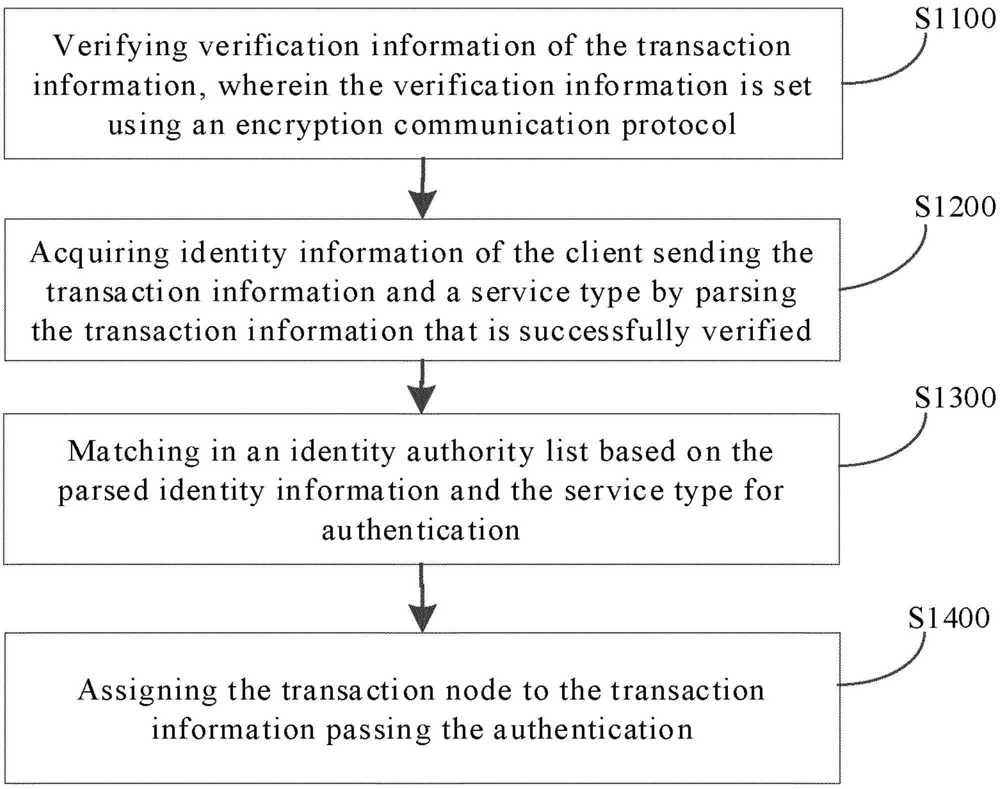
S1100
Verifying verification information of the transaction information, wherein the verification information is set using an encryption communication protocol
S1200
Acquiring identity information of the client sending the transaction information and a service type by parsing the transaction information that is successfully verified
S1300
Matching in an identity authority list based on the parsed identity information and the service type for authentication
S1400
Assigning the transaction node to the transaction information passing the authentication

FIG. 2

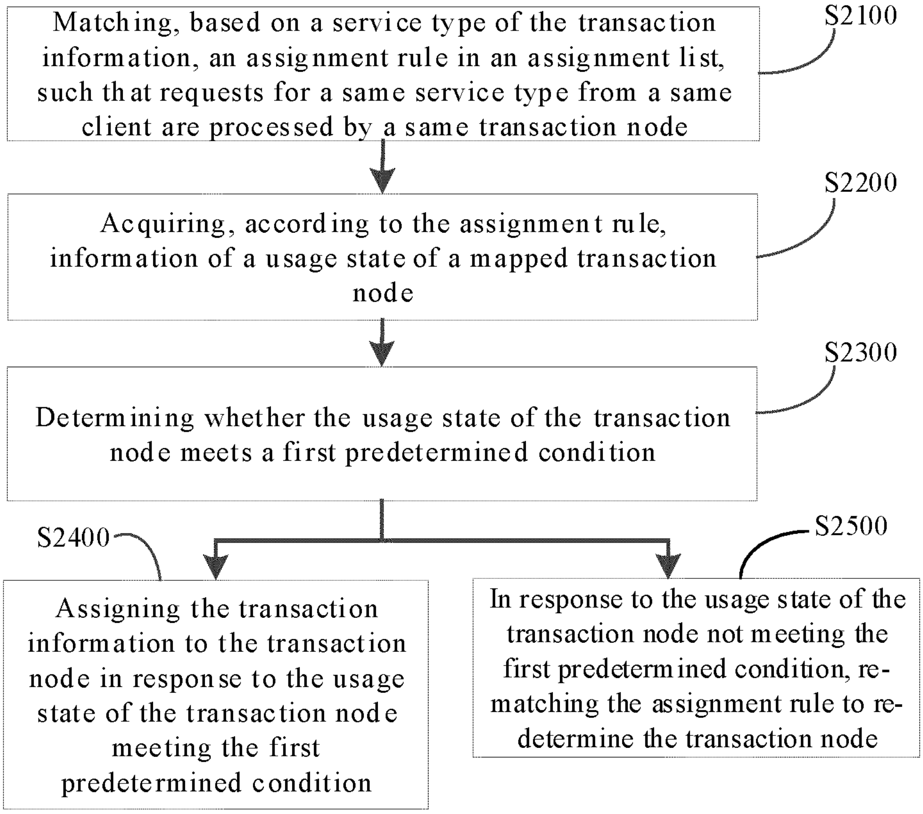
Matching, based on a service type of the transaction information, an assignment rule in an assignment list, such that requests for a same service type from a same client are processed by a same transaction node
S2100
Acquiring, according to the assignment rule, information of a usage state of a mapped transaction node
S2200
Determining whether the usage state of the transaction node meets a first predetermined condition
S2300
S2400
Assigning the transaction information to the transaction node in response to the usage state of the transaction node meeting the first predetermined condition
S2500
In response to the usage state of the transaction node not meeting the first predetermined condition, re-matching the assignment rule to re-determine the transaction node

FIG. 3

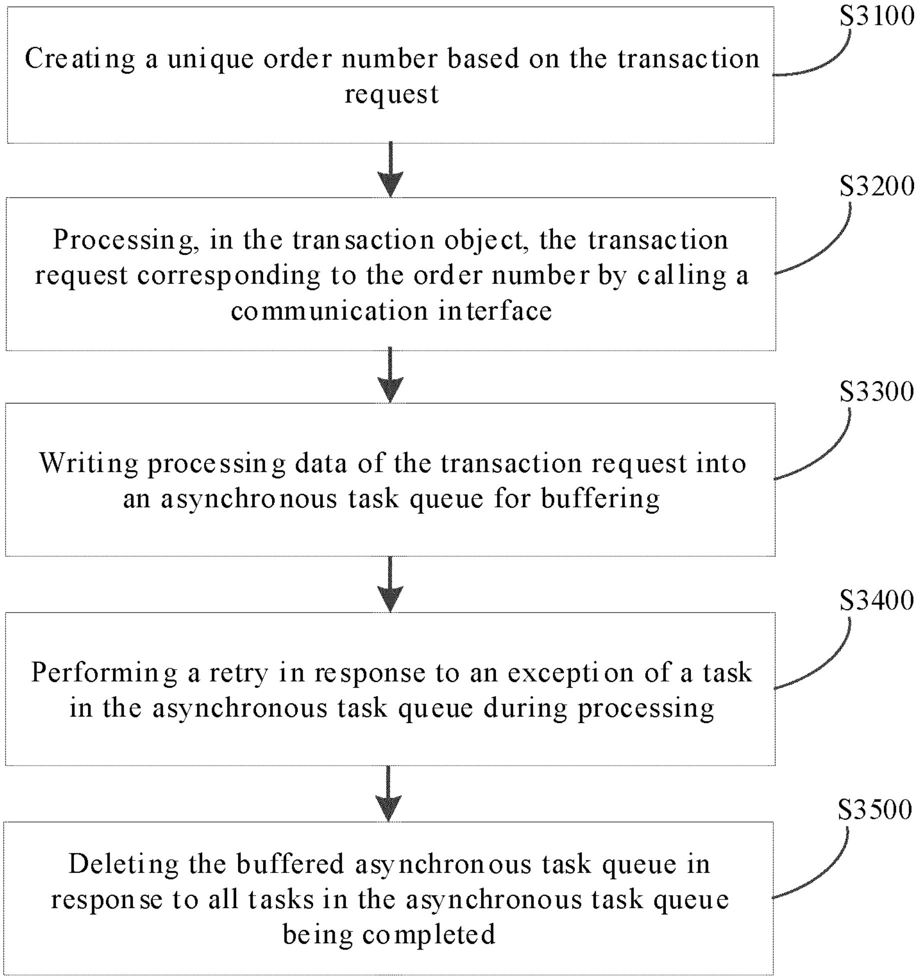
S3100
Creating a unique order number based on the transaction request
S3200
Processing, in the transaction object, the transaction request corresponding to the order number by calling a communication interface
S3300
Writing processing data of the transaction request into an asynchronous task queue for buffering
S3400
Performing a retry in response to an exception of a task in the asynchronous task queue during processing
S3500
Deleting the buffered asynchronous task queue in response to all tasks in the asynchronous task queue being completed

FIG. 4

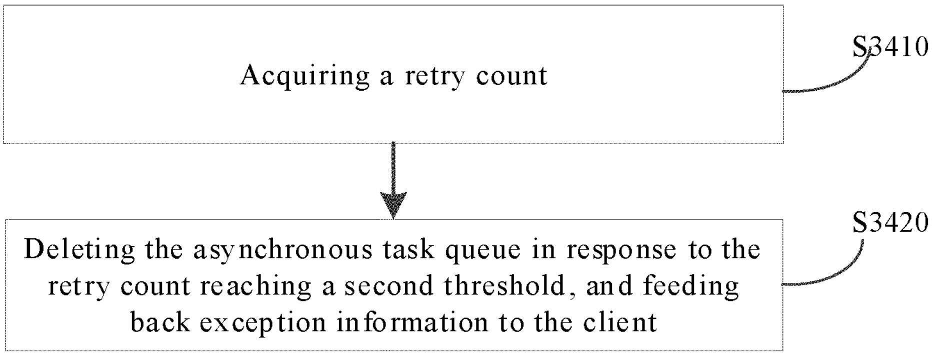
S3410
Acquiring a retry count
S3420
Deleting the asynchronous task queue in response to the retry count reaching a second threshold, and feeding back exception information to the client

FIG. 5

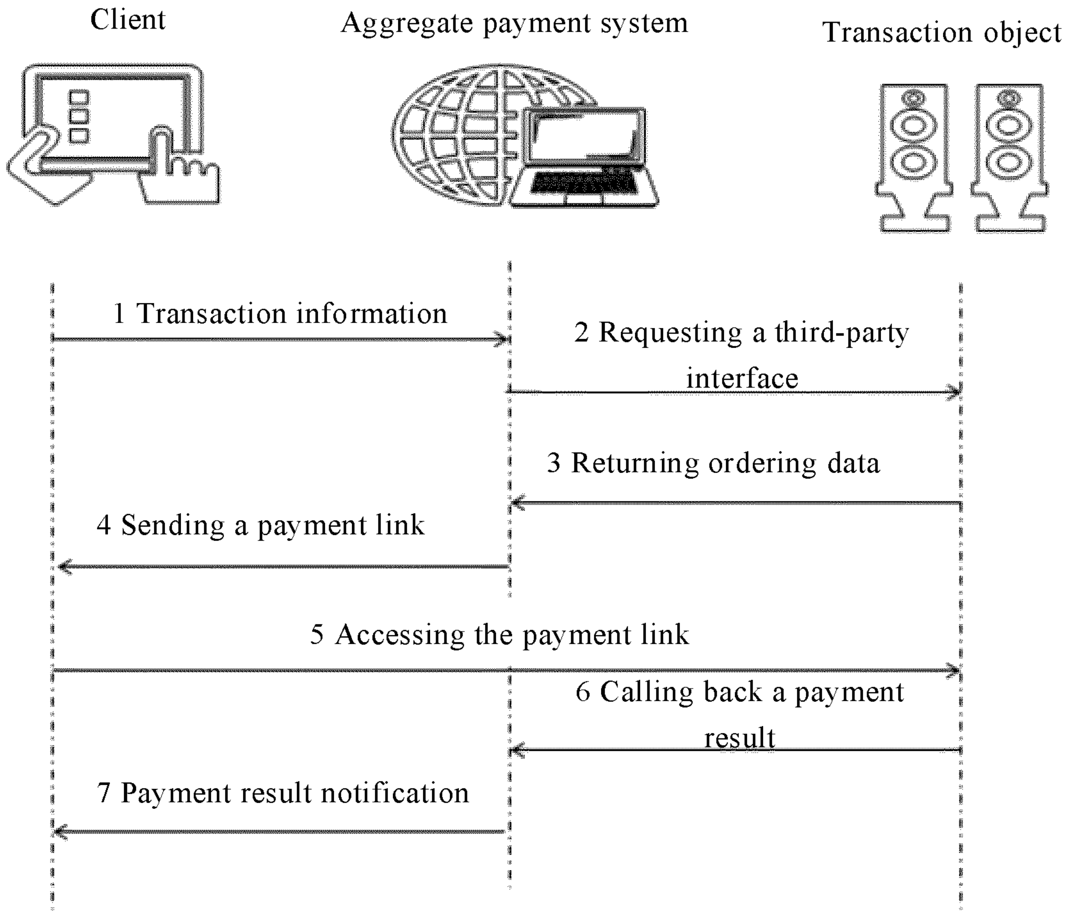
Client
Aggregate payment system
Transaction object
1 Transaction information
2 Requesting a third-party interface
3 Returning ordering data
4 Sending a payment link
5 Accessing the payment link
6 Calling back a payment result
7 Payment result notification

FIG. 6

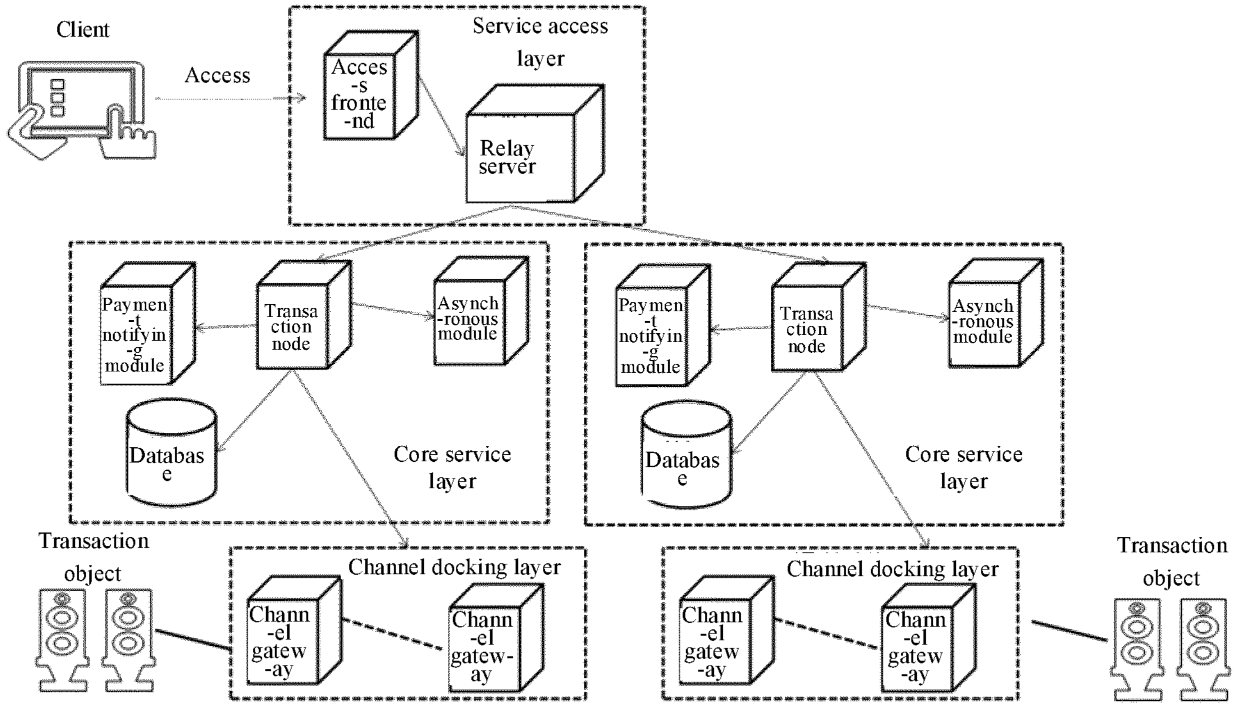
Client
Access
Service access layer
Acces-s fronte-nd
Relay server
Paymen-t notifyin-g module
Transa ction node
Asynch-ronous module
Databas e
Core service layer
Paymen-t notifyin-g module
Transa ction node
Asynch-ronous module
Databas e
Core service layer
Transaction object
Channel docking layer
Chann-el gatew-ay
Chann-el gatew-ay
Channel docking layer
Chann-el gatew-ay
Chann-el gatew-ay
Transaction object

FIG. 7

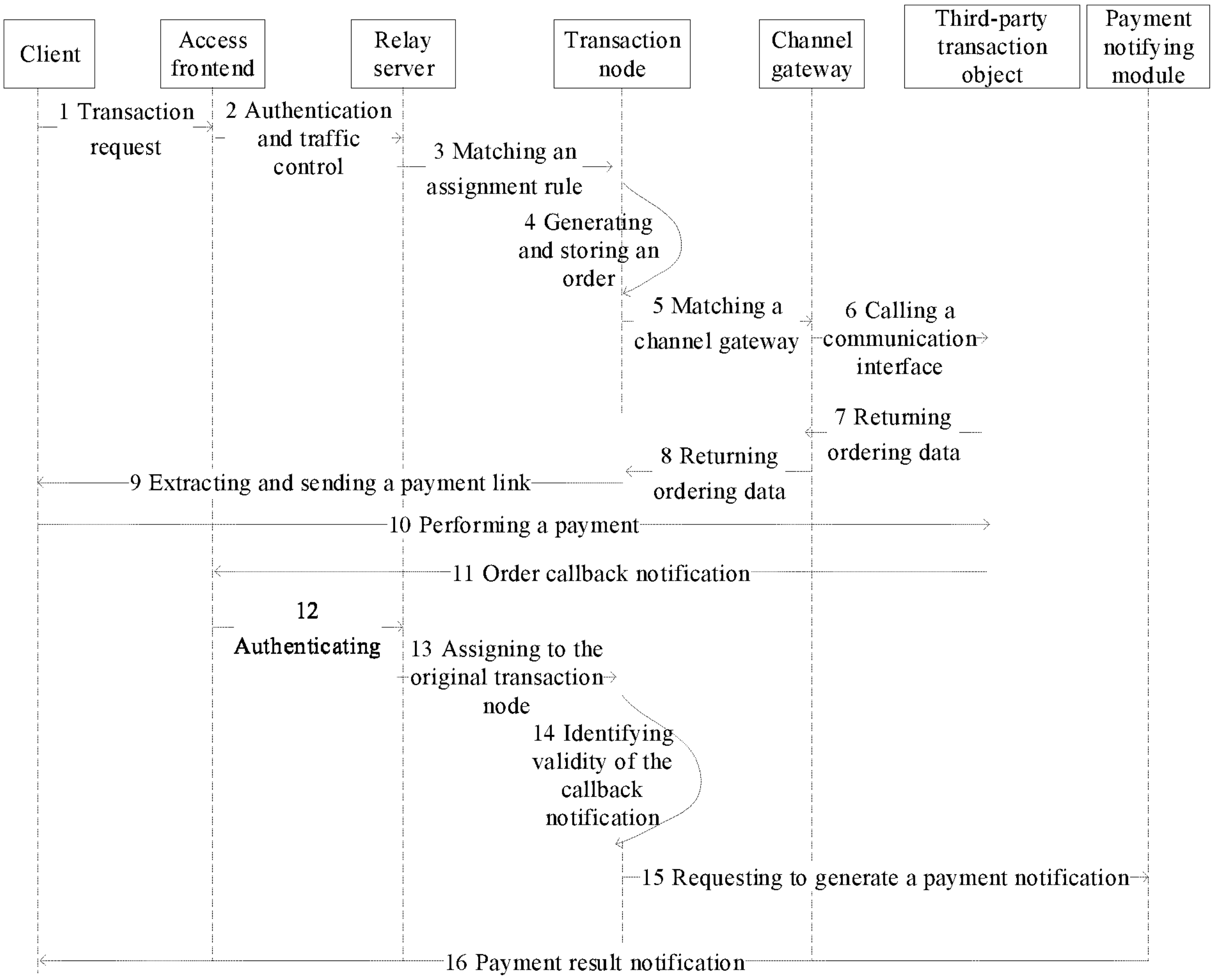
Client
Access frontend
Relay server
Transaction node
Channel gateway
Third-party transaction object
Payment notifying module
1 Transaction request
2 Authentication and traffic control
3 Matching an assignment rule
4 Generating and storing an order
5 Matching a channel gateway
6 Calling a communication interface
7 Returning ordering data
8 Returning ordering data
9 Extracting and sending a payment link
10 Performing a payment
11 Order callback notification
12 Authenticating
13 Assigning to the original transaction node
14 Identifying validity of the callback notification
15 Requesting to generate a payment notification
16 Payment result notification

FIG. 8

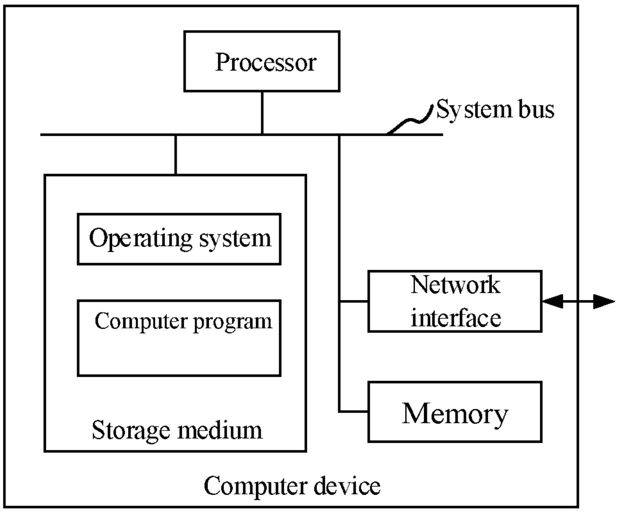
Processor
System bus
Operating system
Computer program
Storage medium
Network interface
Memory
Computer device

FIG. 9

INTERNATIONAL SEARCH REPORT

International application No.
PCT/CN2019/119788

A. CLASSIFICATION OF SUBJECT MATTER

G06Q 20/40(2012.01)i

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

G06Q

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

CNABS, VEN, CNTXT, CNKI, IEEE 聚合支付, 联合, 统一, 后台架构, 交易, 位置, 地理, 地点, 节点, 网关, 渠道, aggregate pay, union, unify, backend architecture, deal, transaction, position, geography, location, node, gateway, channel

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	CN 105474244 A (HEWLETT PACKARD DEVELOPMENT COMPANY LP) 06 April 2016 (2016-04-06) description, paragraphs [0015]-[0042], and figures 1-2B	1-10
Y	CN 106779649 A (SHIJIHEGUANG SCIENCE & TECHNOLOGY DEVELOPMENT (BEIJING) CO., LTD.) 31 May 2017 (2017-05-31) description, paragraphs [0024]-[0034]	1-10
A	CN 103489095 A (BEIJING BAIDU NETCOM SCIENCE AND TECHNOLOGY CO., LTD.) 01 January 2014 (2014-01-01) entire document	1-10
A	CN 105427101 A (CHENGDU LIANYIN INFORMATION TECHNOLOGY CO., LTD.) 23 March 2016 (2016-03-23) entire document	1-10

□ Further documents are listed in the continuation of Box C.   ☑ See patent family annex.

* Special categories of cited documents:

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier application or patent but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art

"&" document member of the same patent family

Date of the actual completion of the international search **07 February 2020**	Date of mailing of the international search report **26 February 2020**
Name and mailing address of the ISA/CN **China National Intellectual Property Administration (ISA/CN)** **No. 6, Xitucheng Road, Jimenqiao Haidian District, Beijing 100088** **China**	Authorized officer
Facsimile No. **(86-10)62019451**	Telephone No.

INTERNATIONAL SEARCH REPORT
Information on patent family members

International application No.
PCT/CN2019/119788

Patent document cited in search report			Publication date (day/month/year)	Patent family member(s)			Publication date (day/month/year)
CN	105474244	A	06 April 2016	EP	3036698	A4	21 June 2017
				WO	2015026323	A1	26 February 2015
				US	2016180299	A1	23 June 2016
				EP	3036698	A1	29 June 2016
CN	106779649	A	31 May 2017	None			
CN	103489095	A	01 January 2014	JP	6473745	B2	20 February 2019
				JP	2016523417	A	08 August 2016
				EP	3057049	A1	17 August 2016
				WO	2015051727	A1	16 April 2015
				US	2016117681	A1	28 April 2016
				EP	3057049	A4	05 April 2017
CN	105427101	A	23 March 2016	None			