WORK ORDER TASK PROCESSING METHOD, COMPUTER SYSTEM, AND STORAGE MEDIUM

TECHNICAL FIELD

[0001] One or more embodiments of this specification relate to the large model field, and in particular, to a work order task processing method, a computer system, and a storage medium.

BACKGROUND

[0002] With continuous development of Internet technologies, many large Internet service platforms (such as a payment platform, a shopping platform, a social platform, and various comprehensive platforms) gradually emerge, and various types of platform services are provided for a user. To ensure that the service platform can operate normally to provide a better service for the user, a work order system is usually used for platform maintenance. For example, the service platform can provide an input interface for the user, and the user can input, through the input interface, a problem encountered when the service platform is used, to generate a work order task used to resolve the problem of the platform. Currently, a work order task processing solution is needed.

SUMMARY

[0003] One or more embodiments of this specification provide a work order task processing method, a computer system, and a storage medium.

[0004] According to a first aspect, a work order task processing method is provided, including: performing the following target operations for at least one time, where the target operations include: obtaining current task information corresponding to a work order task; determining at least one task processing interface based on the current task information by using a target large model and a target graph, where the target graph includes a plurality of first nodes that represent candidate task processing interfaces; invoking the task processing interface to process the work order task, to obtain a current result; and continuing to perform the target operations for a next time if it is

determined, based on the current result, that processing of the work order task is not completed; or outputting a target result based on the current result if it is determined, based on the current result, that processing of the work order task is completed.

[0005] According to a second aspect, a computer system is provided, including a storage, a processor, and a computer program that is stored in the storage and that is capable of running on the processor. The processor performs the following target operations for at least one time when executing the program, and implements the following steps by performing the target operations: obtaining current task information corresponding to a work order task; determining at least one task processing interface based on the current task information by using a target large model and a target graph, where the target graph includes a plurality of first nodes that represent candidate task processing interfaces; invoking the task processing interface to process the work order task, to obtain a current result; and continuing to perform the target operations for a next time if it is determined, based on the current result, that processing of the work order task is not completed; or outputting a target result based on the current result if it is determined, based on the current result, that processing of the work order task is completed.

[0006] According to a third aspect, a computer-readable storage medium is provided. The storage medium stores a computer program, and when the computer program is executed by a processor, the method according to the first aspect is implemented.

[0007] The technical solutions provided in the embodiments of this specification can include the following beneficial effects.

[0008] According to the work order task processing method provided in the embodiments of this specification, the following target operations are performed for at least one time: obtaining current task information corresponding to a work order task; determining at least one task processing interface based on the current task information by using a target large model and a target graph, where the target graph includes a plurality of first nodes that represent candidate task processing interfaces; invoking the task processing interface to process the work order task, to obtain a current result; and continuing to perform the target operations for a next time if it is determined, based on the current result, that processing of the work order task is not completed; or outputting a target result based on the current result if it is determined, based on the current result, that processing of the work order task is completed. In this way, processing of the work order task can be efficiently and accurately completed, and an error rate is reduced.

[0009] It should be understood that the above-mentioned general description and the following detailed description are merely exemplary and explanatory, and are not intended to limit this application.

BRIEF DESCRIPTION OF DRAWINGS

[0010] To describe the technical solutions in embodiments of this specification more clearly, the following briefly describes the accompanying drawings needed for describing the embodiments. Clearly, the accompanying drawings in the following descriptions show merely some embodiments recorded in this specification, and a person of ordinary skill in the art can still derive other drawings from these accompanying drawings without creative efforts.

[0011] FIG. 1 is a schematic architectural diagram of a processing scenario of a work order task according to an example embodiment of this specification;

[0012] FIG. 2 is a flowchart of a work order task processing method according to an example embodiment of this specification;

[0013] FIG. 3 is a schematic diagram of a graph according to an example embodiment of this specification; and

[0014] FIG. 4 is a schematic diagram of a computer system according to an example embodiment of this specification.

DESCRIPTION OF EMBODIMENTS

[0015] To make a person skilled in the art better understand the technical solutions of this specification, the following clearly and completely describes the technical solutions in the embodiments of this specification with reference to the accompanying drawings in the embodiments of this specification. Clearly, the described embodiments are merely some rather than all of the embodiments of this specification. All other embodiments obtained by a person of ordinary skill in the art based on the embodiment of this specification without creative efforts shall fall within the protection scope of this specification.

[0016] With continuous development of Internet technologies, many large Internet service platforms (such as a payment platform, a shopping platform, a social platform, and various comprehensive platforms) gradually emerge, and various types of platform services are provided

for a user. To ensure that the service platform can operate normally to provide a better service for the user, a work order system is usually used for platform maintenance. For example, the service platform can provide an input interface for the user, and the user can input, through the input interface, a problem encountered when the service platform is used, to generate a work order task used to resolve the problem of the platform.

[0017] In a related technology, an operation and maintenance person usually processes the work order task manually, to solve the problem of the service platform. Alternatively, the platform can invoke a pre-compiled rule, to resolve the problem of the service platform. However, human resources need to be consumed when the operation and maintenance person processes the work order task manually. In addition, efficiency of manually processing the work order task is usually relatively low, and it is prone to an error. A problem of processing the work order task by invoking the pre-compiled rule lies in that it is difficult to compile a full and accurate rule set, maintenance costs of the rule set are relatively high, and the rule set may be incapable of covering all cases, leading to limited accuracy and integrity of an invoking result.

[0018] Embodiments of this specification provide a work order task processing solution, and the following target operations are performed for at least one time: obtaining current task information corresponding to a work order task; determining at least one task processing interface based on the current task information by using a target large model and a target graph, where the target graph includes a plurality of first nodes that represent candidate task processing interfaces; invoking the task processing interface to process the work order task, to obtain a current result; and continuing to perform the target operations for a next time if it is determined, based on the current result, that processing of the work order task is not completed; or outputting a target result based on the current result if it is determined, based on the current result, that processing of the work order task is completed. In this way, processing of the work order task can be efficiently and accurately completed, and an error rate is reduced.

[0019] FIG. 1 is a schematic architectural diagram of a processing scenario of a work order task according to an example embodiment of this specification.

[0020] As shown in FIG. 1, a specific application scenario can be as follows: After buying a product on a shopping platform, a user requests a refund, and after receiving the refund, finds that a bank arrival amount is inconsistent with an actual payment amount. The user can find a problem input interface on a personal account page of the shopping platform, and input, through the

problem input interface, problem content Q described in a natural language. Alternatively, a shop customer service can input, through a problem input interface on a shop management page of the shopping platform, problem content Q described in a natural language. For example, the problem content Q can be "A transaction order number is 123456789. A bank arrival amount is inconsistent with an actual payment amount after a refund is received. A reason needs to be found". After the problem content is input and submitted, the shopping platform generates a new work order task W.

[0021] First, current task information A1 of the work order task W can be obtained. The current task information A1 can include the problem content Q, a prompt for a large model, and a description of a task processing interface. The prompt for a large model can be used to indicate a role played by the large model or text content of a thing that needs to be done by the large model. The prompt for a large model can be content entered together when the user enters the problem content Q, or can be preset in a problem content input template. The description of a task processing interface can be a description of use of each task processing interface, etc.

[0022] Then, the first round of operation can be performed. Specifically, the current task information A1 can be input into a large model M1, to obtain a result output by the large model M1. The result can include task purpose information B1 that describes a task intent. In this application scenario, the task purpose information B1 that describes the task intent can be "Query a bank payment history of the transaction". The current task information A1 and the task purpose information B1 can be input into a large model M2, to obtain a result output by the large model M2. An input parameter type and an input parameter value of the task processing interface can be obtained from a result output by a second large model. If the result does not include the input parameter value, the input parameter value can be obtained based on the current task information A1 and the task purpose information B1 by using a preset regular expression. The input parameter type of the task processing interface can be a type of a parameter that needs to be input into the task processing interface when the task processing interface is used. The input parameter type can be used as a key k. In this case, the input parameter value is a value v of the key k. In this application scenario, the input parameter type can be "Transaction order number", and the input parameter value can be "123456789".

[0023] Then, a pre-constructed graph T can be obtained. The graph T can include a plurality of nodes, some nodes represent task processing interfaces, and some nodes represent parameter types. There is a connecting edge between nodes, the connecting edge is a directed edge, and the

connecting edge represents an input/output relationship between a task processing interface and a parameter type. For example, a node 1 represents a task processing interface a, a node 2 represents a parameter type b, and a node 3 represents a parameter type c. There is a connecting edge 21 between the node 1 and the node 2, the connecting edge 21 points to the node 1 by using an arrow, and the connecting edge 21 indicates that an input parameter type of the task processing interface a can be the parameter type b. There is a connecting edge 23 between the node 2 and the node 3, the connecting edge 23 points to the node 3 by using an arrow, and the connecting edge 23 indicates that an output parameter type of the task processing interface a can be the parameter type c. A plurality of task processing interfaces and respective input and output parameter types can be collected in advance; and then the corresponding graph T is constructed based on the plurality of task processing interfaces and respective input and output parameter types, and the graph T is stored. When the task processing interface is determined, the graph T can be searched for a target path from an input parameter to an output parameter based on the input parameter type and the input parameter value of the task processing interface, and the task purpose information B1.

[0024] At least one task processing interface indicated by a node included in the target path in the graph T can be determined, and the at least one task processing interface is sequentially invoked based on an input parameter value, to obtain a current result. In this application scenario, the task purpose information B1 is "Query a bank payment history of the transaction", the task processing interface can include an interface K1 and an interface K2, an input parameter type of the interface K1 can be "Transaction order number", an input parameter value of the interface K1 can be "123456789", an output parameter type of the interface K1 can be "Bank payment number", and an output parameter value of the interface K1 can be "kajdajkd". An input parameter type of the interface K2 can be "Bank payment number", an input parameter value of the interface K2 can be "kajdajkd", an output parameter type of the interface K2 can be "Payment amount" and "Refund amount", output parameter values of the interface K2 can be "n" and "n", and the output parameter values "n" and "n" are a current result.

[0025] Then, the current result and the current task information A1 can be input into a large model M3. The large model M3 can output, based on the current result and the current task information A1, a result that is used to indicate whether processing of the work order task is completed currently. In this application scenario, the result can be "Processing of the work order task is not completed". Therefore, the second round of operation further needs to be performed.

Specifically, the output parameter values "n" and "n" of the first round of operation and the problem content Q can be used as current task information A2 of the second round of operation. Then, the current task information A2 is input into the large model M1, to obtain a result output by the large model M1. The result can include task purpose information B2 that describes a task intent. In this application scenario, the task purpose information B2 that describes the task intent can be "Query an actual payment amount and details of the transaction".

[0026] The current task information A2 and the task purpose information B2 can be input into the large model M2, to obtain a result output by the large model M2. The input parameter type and the input parameter value of the task processing interface are determined based on the result output by the second large model. The graph T is searched for the target path based on the input parameter type and the input parameter value of the task processing interface, and the task purpose information B2. The at least one task processing interface indicated by the node included in the target path in the graph T is determined, and the at least one task processing interface is sequentially invoked based on the input parameter value, to obtain a result of the second round of operation. In this application scenario, the input parameter type can be "Transaction order number", and the input parameter value can be "123456789". The task purpose information B2 is "Query an actual payment amount and details of the transaction", the task processing interface can include an interface K3, an input parameter type of the interface K3 can be "Transaction order number", an input parameter value of the interface K3 can be "123456789", an output parameter type of the interface K3 can be "Actual payment amount and details of the transaction", and an output parameter value of the interface K3 can be "Cash n and red packet y", namely, the result of the second round.

[0027] Finally, the result of the second round and the current task information A2 can be input into the large model M3. The large model M3 can output, based on the result of the second round and the current task information A2, a result that is used to indicate whether processing of the work order task is completed currently. In this application scenario, the result can be "Processing of the work order task is completed". Therefore, the third round of operation does not need to be performed. The problem content Q, the result "n" and "n" of the first round, and the result "Cash n and red packet y" of the second round can be analyzed based on a preset rule set or based on another large model, to obtain a conclusion, and the conclusion is returned to the user. The conclusion can be text content in a natural language, for example, can be "An actual payment

amount includes a cash n and a red packet y, the cash part n is paid by a bank account and is fully returned to the bank account now, and the red packet part y is returned to a transaction account".

[0028] The following describes the solutions provided in this specification in detail with reference to specific embodiments.

[0029] FIG. 2 is a flowchart of a work order task processing method according to an example embodiment. The method can be performed by a service platform, and the service platform can be implemented as any device, server, or device cluster with a computing and processing capability. The method includes the following steps.

[0030] Step 201: Obtain current task information corresponding to a work order task.

[0031] In this embodiment, the involved service platform can include but is not limited to a shopping platform, a payment platform, a social platform, various other comprehensive platforms, etc. When a user makes a transaction on the service platform or after a user makes a transaction, if a problem is encountered, the user can enter, through a problem input interface provided by the service platform, problem content described in a natural language, to query or resolve a problem. After the problem content is entered and submitted, the service platform generates a new work order task.

[0032] When a work order task is processed, target operations need to be performed for at least one time. If the target operations are performed for the first time, task content entered by the user for the work order task can be obtained, and the task content is used as the current task information. If the target operations are not performed for the first time, task content entered by the user for the work order task and a result obtained by performing the target operations last time can be obtained, and the task content and the result are used as the current task information.

[0033] Step 202: Determine at least one task processing interface based on the current task information by using a target large model and a target graph.

[0034] Specifically, parameter information can be first determined based on the current task information by using the target large model. The parameter information can include task purpose information, an input parameter type, and an input parameter value of the task processing interface. The target large model can be a general large language model. A specific category aspect of the target large model is not limited in this embodiment. The task processing interface can be an interface tool used to process a task. The task processing interface corresponds to an input parameter and an output parameter. The task processing interface can be invoked to process the

input parameter value, to obtain the output parameter value.

[0035] In an implementation, the parameter information can be determined based on the current task information by using one target large model. In another implementation, the parameter information can alternatively be determined based on the current task information by using a plurality of target large models. For example, the target large model can include a first large model and a second large model. The current task information can be input into the first large model, to obtain task purpose information that describes a task intent. Then, the current task information and the task purpose information are input into the second large model, and the input parameter type and the input parameter value are determined based on a result output by the second large model. For example, the input parameter type and the input parameter value can be obtained from the result output by the second large model. If the result output by the second large model does not include the input parameter value, the input parameter value can be further obtained based on the current task information, the input parameter type, and the task purpose information by using a preset regular expression.

[0036] Then, the at least one task processing interface can be determined based on the parameter information and the target graph. The target graph can be a graph pre-constructed based on a plurality of task processing interfaces and an input parameter type and an output parameter type of each task processing interface that are collected. The target graph includes a plurality of nodes, some nodes represent task processing interfaces, and some nodes represent parameter types. A node that represents a task processing interface can serve as a first node, and a node that represents a parameter type can serve as a second node. There is a connecting edge between nodes, the connecting edge is a directed edge, and the connecting edge represents an input/output relationship between a task processing interface and a parameter type.

[0037] As shown in FIG. 3, all of nodes 101 to 105 are first nodes that represent task processing interfaces, and all of nodes 106 to 111 are second nodes that represent parameter types. The node 106 represents an input parameter type of the node 101 and a node 102, a node 104 and a node 108 represent output parameter types of the node 101, a node 107 represents an output parameter type of the node 102, the node 107 represents an input parameter type of a node 103, the node 104 represents an output parameter type of the node 103, etc.

[0038] Specifically, the target graph can be searched for a target path from an input parameter to an output parameter based on the parameter information, and the at least one task processing

interface is determined based on the target path. For example, the target graph can be searched for the shortest path from the input parameter to the output parameter based on a Dijkstra algorithm and the parameter information, and the shortest path serves as the target path. Then, a task processing interface included in the target path serves as a task processing interface used to process the work order task.

[0039] It should be noted that if a new candidate task processing interface needs to be added, the target graph can be updated. The target graph can be updated in the following manner: A to-be-added candidate task processing interface is first obtained. An input parameter type and an output parameter type that correspond to the to-be-added candidate task processing interface are determined. The node in the target graph can be traversed based on the input parameter type and the output parameter type that correspond to the to-be-added candidate task processing interface, to determine a second node associated with the to-be-added candidate task processing interface. A parameter type indicated by the second node associated with the to-be-added candidate task processing interface is the same as the input parameter type or the output parameter type that correspond to the to-be-added candidate task processing interface. A new first node is generated in the target graph for the to-be-added candidate task processing interface, and a connecting edge is generated between the new first node and the associated second node.

[0040] Step 203: Invoke the task processing interface to process the work order task, to obtain a current result.

[0041] In this embodiment, if it is determined that there is one task processing interface, the task processing interface can be invoked to process the input parameter value included in the parameter information. To be specific, the input parameter value included in the parameter information is directly input into the task processing interface, to obtain an output parameter value output by the task processing interface and use the output parameter value as a current result. If it is determined that there are a plurality of task processing interfaces, the plurality of task processing interfaces can be sequentially invoked based on a sequence of task processing interfaces included in the target path. To be specific, an input parameter value is input into the first task processing interface, and an output parameter value of the first task processing interface is input into a next task processing interface. By analogy, an output parameter value output by a last task processing interface is finally obtained and used as a current result.

[0042] Step 204: Continue to perform the target operations for a next time if it is determined,

based on the current result, that processing of the work order task is not completed; or output a target result based on the current result if it is determined, based on the current result, that processing of the work order task is completed.

[0043] In this embodiment, after the current result is obtained, whether processing of the work order task is completed can be determined based on the current result. For example, whether processing of the work order task is completed can be determined based on a preset rule. Alternatively, the current result and the current task information can be input into a third large model, whether processing of the work order task is completed currently is determined based on a result output by the third large model, and the third large model can directly output information used to indicate whether processing of the work order task is completed.

[0044] The target operations continue to be performed for the next time if it is determined, based on the current result, that processing of the work order task is not completed. To be specific, the current result and the task content are used as current task information of the target operations performed for the next time, and a task processing interface is determined based on the current task information of the target operations performed for the next time and by using the target large model and the target graph. The target result is output based on the current result if it is determined, based on the current result, that processing of the work order task is completed. The target result can be determined based on the preset rule and the current result, or the target result can be determined based on the current result by using a large model. It can be understood that the target result can be determined in any proper manner. A specific manner of determining the target result is not limited in this embodiment.

[0045] According to the work order task processing method provided in this embodiment of this specification, the following target operations are performed for at least one time: obtaining current task information corresponding to a work order task; determining at least one task processing interface based on the current task information by using a target large model and a target graph, where the target graph includes a plurality of first nodes that represent candidate task processing interfaces; invoking the task processing interface to process the work order task, to obtain a current result; and continuing to perform the target operations for a next time if it is determined, based on the current result, that processing of the work order task is not completed; or outputting a target result based on the current result if it is determined, based on the current result, that processing of the work order task is completed. In this way, processing of the work

order task can be efficiently and accurately completed, and an error rate is reduced.

[0046] It should be noted that although operations of the method in the embodiments of this specification are described in a particular order in the above-mentioned embodiments, it is not required or implied that these operations need to be performed in the particular order or that all the shown operations need to be performed to achieve a desired result. In the contrary, an execution sequence of steps depicted in the flowchart can be changed. Additionally or alternatively, some steps can be omitted, a plurality of steps can be combined into one step for execution, and/or one step can be decomposed into a plurality of steps for execution.

[0047] The specification also provides a computer-readable non-transitory storage medium, where the storage medium stores a computer program, which, when executed by a processor, can be used to perform one or more steps of one or more methods described or illustrated herein or provides functionality described or illustrated herein. Herein, the computer-readable non-transitory storage medium or media may include one or more semiconductor-based or other integrated circuits (ICs) (such, as for example, field-programmable gate arrays (FPGAs) or application-specific ICs (ASICs)), hard disk drives (HDDs), hybrid hard drives (HHDs), optical discs, optical disc drives (ODDs), magneto-optical discs, magneto-optical drives, floppy diskettes, floppy disk drives (FDDs), magnetic tapes, solid-state drives (SSDs), RAM-drives, SECURE DIGITAL cards or drives, any other suitable computer-readable non-transitory storage media, or any suitable combination of two or more of these, where appropriate. A computer-readable non-transitory storage medium may be volatile, non-volatile, or a combination of volatile and non-volatile, where appropriate.

[0048] The specification also provides a computer system. FIG. 4 illustrates an example computer system 1100. In particular embodiments, one or more computer systems 1100 perform one or more steps of one or more methods described or illustrated herein. In particular embodiments, one or more computer systems 1100 provide functionality described or illustrated herein. In particular embodiments, software running on one or more computer systems 1100 performs one or more steps of one or more methods described or illustrated herein or provides functionality described or illustrated herein. Particular embodiments include one or more portions of one or more computer systems 1100. Herein, reference to a computer system may encompass a computing device, and vice versa, where appropriate. Moreover, reference to a computer system may encompass one or more computer systems, where appropriate.

[0049] This disclosure contemplates any suitable number of computer systems 1100. This disclosure contemplates computer system 1100 taking any suitable physical form. As example and not by way of limitation, computer system 1100 may be an embedded computer system, a system-on-chip (SOC), a single-board computer system (SBC) (such as, for example, a computer-on-module (COM) or system-on-module (SOM)), a desktop computer system, a laptop or notebook computer system, an interactive kiosk, a mainframe, a mesh of computer systems, a mobile telephone, a personal digital assistant (PDA), a server, a tablet computer system, or a combination of two or more of these. Where appropriate, computer system 1100 may include one or more computer systems 1100; be unitary or distributed; span multiple locations; span multiple machines; span multiple data centers; or reside in a cloud, which may include one or more cloud components in one or more networks. Where appropriate, one or more computer systems 1100 may perform without substantial spatial or temporal limitation one or more steps of one or more methods described or illustrated herein. As an example and not by way of limitation, one or more computer systems 1100 may perform in real time or in batch mode one or more steps of one or more methods described or illustrated herein. One or more computer systems 1100 may perform at different times or at different locations one or more steps of one or more methods described or illustrated herein, where appropriate.

[0050] In particular embodiments, computer system 1100 includes a processor 1102, memory 1104, storage 1106, an input/output (I/O) interface 1108, a communication interface 1110, and a bus 1112. Although this disclosure describes and illustrates a particular computer system having a particular number of particular components in a particular arrangement, this disclosure contemplates any suitable computer system having any suitable number of any suitable components in any suitable arrangement.

[0051] In particular embodiments, processor 1102 includes hardware for executing instructions, such as those making up a computer program. As an example and not by way of limitation, to execute instructions, processor 1102 may retrieve (or fetch) the instructions from an internal register, an internal cache, memory 1104, or storage 1106; decode and execute them; and then write one or more results to an internal register, an internal cache, memory 1104, or storage 1106. In particular embodiments, processor 1102 may include one or more internal caches for data, instructions, or addresses. This disclosure contemplates processor 1102 including any suitable number of any suitable internal caches, where appropriate. As an example and not by way of

limitation, processor 1102 may include one or more instruction caches, one or more data caches, and one or more translation lookaside buffers (TLBs). Instructions in the instruction caches may be copies of instructions in memory 1104 or storage 1106, and the instruction caches may speed up retrieval of those instructions by processor 1102. Data in the data caches may be copies of data of 80 in memory 1104 or storage 1106 for instructions executing at processor 1102 to operate on; the results of previous instructions executed at processor 1102 for access by subsequent instructions executing at processor 1102 or for writing to memory 1104 or storage 1106; or other suitable data. The data caches may speed up read or write operations by processor 1102. The TLBs may speed up virtual-address translation for processor 1102. In particular embodiments, processor 1102 may include one or more internal registers for data, instructions, or addresses. This disclosure contemplates processor 1102 including any suitable number of any suitable internal registers, where appropriate. Where appropriate, processor 1102 may include one or more arithmetic logic units (ALUs); be a multi-core processor; or include one or more processors 1102. Although this disclosure describes and illustrates a particular processor, this disclosure contemplates any suitable processor.

[0052] In particular embodiments, memory 1104 includes main memory for storing instructions for processor 1102 to execute or data for processor 1102 to operate on. As an example and not by way of limitation, computer system 1100 may load instructions from storage 1106 or another source (such as, for example, another computer system 1100) to memory 1104. Processor 1102 may then load the instructions from memory 1104 to an internal register or internal cache. To execute the instructions, processor 1102 may retrieve the instructions from the internal register or internal cache and decode them. During or after execution of the instructions, processor 1102 may write one or more results (which may be intermediate or final results) to the internal register or internal cache. Processor 1102 may then write one or more of those results to memory 1104. In particular embodiments, processor 1102 executes only instructions in one or more internal registers or internal caches or in memory 1104 (as opposed to storage 1106 or elsewhere) and operates only on data in one or more internal registers or internal caches or in memory 1104 (as opposed to storage 1106 or elsewhere). One or more memory buses (which may each include an address bus and a data bus) may couple processor 1102 to memory 1104. Bus 1112 may include one or more memory buses, as described below. In particular embodiments, one or more memory management units (MMUs) reside between processor 1102 and memory 1104 and facilitate accesses to memory

1104 requested by processor 1102. In particular embodiments, memory 1104 includes random access memory (RAM). This RAM may be volatile memory, where appropriate. Where appropriate, this RAM may be dynamic RAM (DRAM) or static RAM (SRAM). Moreover, where appropriate, this RAM may be single-ported or multi-ported RAM. This disclosure contemplates any suitable RAM. Memory 1104 may include one or more memories 1104, where appropriate. Although this disclosure describes and illustrates particular memory, this disclosure contemplates any suitable memory.

[0053] In particular embodiments, storage 1106 includes mass storage for data or instructions. As an example and not by way of limitation, storage 1106 may include a hard disk drive (HDD), a floppy disk drive, flash memory, an optical disc, a magneto-optical disc, magnetic tape, or a Universal Serial Bus (USB) drive or a combination of two or more of these. Storage 1106 may include removable or non-removable (or fixed) media, where appropriate. Storage 1106 may be internal or external to computer system 1100, where appropriate. In particular embodiments, storage 1106 is non-volatile, solid-state memory. In particular embodiments, storage 1106 includes read-only memory (ROM). Where appropriate, this ROM may be mask-programmed ROM, programmable ROM (PROM), erasable PROM (EPROM), electrically erasable PROM (EEPROM), electrically alterable ROM (EAROM), or flash memory or a combination of two or more of these. This disclosure contemplates mass storage 1106 taking any suitable physical form. Storage 1106 may include one or more storage control units facilitating communication between processor 1102 and storage 1106, where appropriate. Where appropriate, storage 1106 may include one or more storages 1106. Although this disclosure describes and illustrates particular storage, this disclosure contemplates any suitable storage.

[0054] In particular embodiments, I/O interface 1108 includes hardware, software, or both, providing one or more interfaces for communication between computer system 1100 and one or more I/O devices. Computer system 1100 may include one or more of these I/O devices, where appropriate. One or more of these I/O devices may enable communication between a person and computer system 1100. As an example and not by way of limitation, an I/O device may include a keyboard, keypad, microphone, monitor, mouse, printer, scanner, speaker, still camera, stylus, tablet, touch screen, trackball, video camera, another suitable I/O device or a combination of two or more of these. An I/O device may include one or more sensors. This disclosure contemplates any suitable I/O devices and any suitable I/O interfaces 1108 for them. Where appropriate, I/O

interface 1108 may include one or more device or software drivers enabling processor 1102 to drive one or more of these I/O devices. I/O interface 1108 may include one or more I/O interfaces 1108, where appropriate. Although this disclosure describes and illustrates a particular I/O interface, this disclosure contemplates any suitable I/O interface.

[0055] In particular embodiments, communication interface 1110 includes hardware, software, or both providing one or more interfaces for communication (such as, for example, packet-based communication) between computer system 1100 and one or more other computer systems 1100 or one or more networks. As an example and not by way of limitation, communication interface 1110 may include a network interface controller (NIC) or network adapter for communicating with an Ethernet or other wire-based network or a wireless NIC (WNIC) or wireless adapter for communicating with a wireless network, such as a WI-FI network. This disclosure contemplates any suitable network and any suitable communication interface 1110 for it. As an example and not by way of limitation, computer system 1100 may communicate with an ad hoc network, a personal area network (PAN), a local area network (LAN), a wide area network (WAN), a metropolitan area network (MAN), or one or more portions of the Internet or a combination of two or more of these. One or more portions of one or more of these networks may be wired or wireless. As an example, computer system 1100 may communicate with a wireless PAN (WPAN) (such as, for example, a BLUETOOTH WPAN), a WI-FI network, a WI-MAX network, a cellular telephone network (such as, for example, a Global System for Mobile Communications (GSM) network), or other suitable wireless network or a combination of two or more of these. Computer system 1100 may include any suitable communication interface 1110 for any of these networks, where appropriate. Communication interface 1110 may include one or more communication interfaces 1110, where appropriate. Although this disclosure describes and illustrates a particular communication interface, this disclosure contemplates any suitable communication interface.

[0056] In particular embodiments, bus 1112 includes hardware, software, or both coupling components of computer system 1100 to each other. As an example and not by way of limitation, bus 1112 may include an Accelerated Graphics Port (AGP) or other graphics bus, an Enhanced Industry Standard Architecture (EISA) bus, a front-side bus (FSB), a HYPERTRANSPORT (HT) interconnect, an Industry Standard Architecture (ISA) bus, an INFINIBAND interconnect, a low-pin-count (LPC) bus, a memory bus, a Micro Channel Architecture (MCA) bus, a Peripheral Component Interconnect (PCI) bus, a PCI-Express (PCIe) bus, a serial advanced technology

attachment (SATA) bus, a Video Electronics Standards Association local (VLB) bus, or another suitable bus or a combination of two or more of these. Bus 1112 may include one or more buses 1112, where appropriate. Although this disclosure describes and illustrates a particular bus, this disclosure contemplates any suitable bus or interconnect.

[0057] It should be further noted that, the terms "include", "comprise", or any other variant thereof are intended to cover a non-exclusive inclusion, so that a process, a method, a product or a device that includes a list of elements not only includes those elements but also includes other elements which are not expressly listed, or further includes elements inherent to such process, method, product or device. Without more constraints, an element preceded by "includes a …" does not preclude the existence of additional identical elements in the process, method, product or device that includes the element.

[0058] The embodiments of this specification are described in a progressive way. For same or similar parts in the embodiments, references can be made to each other. Each embodiment focuses on a difference from another embodiment. Particularly, some system embodiments are briefly described because they are basically similar to some method embodiments. For related parts, references can be made to related descriptions in some method embodiments.

[0059] The above-mentioned descriptions are merely embodiments of this specification, and are not intended to limit this specification. A person skilled in the art can make various changes and changes to this specification. Any modification, equivalent replacement, improvement, etc. made without departing from the spirit and principle of this specification shall fall within the scope of the claims in this application.

CLAIMS

What is claimed is:

1. A work order task processing method, comprising the following target operations for at least one time, wherein the target operations comprise:

obtaining current task information corresponding to a work order task;

determining at least one task processing interface based on the current task information by using a target large model and a target graph, wherein the target graph comprises a plurality of first nodes that represent candidate task processing interfaces;

invoking the task processing interface to process the work order task, to obtain a current result; and

continuing to perform the target operations for a next time if it is determined, based on the current result, that processing of the work order task is not completed; or outputting a target result based on the current result if it is determined, based on the current result, that processing of the work order task is completed.

2. The method according to claim 1, wherein the obtaining current task information corresponding to a work order task comprises:

if the target operations are performed for the first time, obtaining task content entered for the work order task, and using the task content as the current task information; and

if the target operations are not performed for the first time, obtaining the task content and a result obtained by performing the target operations last time, and using the task content and the result as the current task information.

3. The method according to claim 1, wherein the determining at least one task processing interface based on the current task information by using a target large model and a target graph comprises:

determining parameter information based on the current task information by using the target large model, wherein the parameter information comprises task purpose information, and an input parameter type and an input parameter value of the task processing interface; and

determining the at least one task processing interface based on the parameter information and the target graph.

4. The method according to claim 3, wherein the target large model comprises a first large

model and a second large model; and the determining parameter information based on the current task information by using the target large model comprises:

inputting the current task information into the first large model, to obtain task purpose information that describes a task intent; and

inputting the current task information and the task purpose information into the second large model, and determining the input parameter type and the input parameter value based on a result output by the second large model.

5. The method according to claim 4, wherein the determining the input parameter type and the input parameter value based on a result output by the second large model comprises:

obtaining the input parameter type and the input parameter value from the result output by the second large model; and

if the result output by the second large model does not comprise the input parameter value, obtaining the input parameter value based on the current task information, the input parameter type, and the task purpose information by using a preset regular expression.

6. The method according to claim 3, wherein the determining the at least one task processing interface based on the parameter information and the target graph comprises:

searching the target graph for a target path from an input parameter to an output parameter based on the parameter information; and

determining the at least one task processing interface based on the target path.

7. The method according to claim 1, wherein after the current result is obtained, the method further comprises:

inputting the current result and the current task information into a third large model; and

determining, based on a result output by the third large model, whether processing of the work order task is completed currently.

8. The method according to claim 1, wherein the target graph further comprises a plurality of second nodes representing parameter types, there is a connecting edge between nodes in the target graph, the connecting edge is a directed edge, and the connecting edge represents an input/output relationship between a candidate task processing interface and a parameter type.

9. The method according to claim 8, wherein the target graph is updated in the following manner:

obtaining a to-be-added candidate task processing interface;

determining an input parameter type and an output parameter type that correspond to the to-be-added candidate task processing interface;

traversing the node in the target graph based on the input parameter type and the output parameter type that correspond to the to-be-added candidate task processing interface, and determining a second node associated with the to-be-added candidate task processing interface; and

generating a new first node in the target graph for the to-be-added candidate task processing interface, and generating a connecting edge between the new first node and the associated second node.

10. A computer system, comprising a storage, a processor, and a computer program that is stored in the storage and that is capable of running on the processor, wherein the processor performs the following target operations for at least one time when executing the program, and implements the following steps by performing the target operations:

obtaining current task information corresponding to a work order task;

determining at least one task processing interface based on the current task information by using a target large model and a target graph, wherein the target graph comprises a plurality of first nodes that represent candidate task processing interfaces;

invoking the task processing interface to process the work order task, to obtain a current result; and

continuing to perform the target operations for a next time if it is determined, based on the current result, that processing of the work order task is not completed; or outputting a target result based on the current result if it is determined, based on the current result, that processing of the work order task is completed.

11. The computer system according to claim 10, wherein the obtaining current task information corresponding to a work order task comprises:

if the target operations are performed for the first time, obtaining task content entered for the work order task, and using the task content as the current task information; and

if the target operations are not performed for the first time, obtaining the task content and a result obtained by performing the target operations last time, and using the task content and the result as the current task information.

12. The computer system according to claim 10, wherein the determining at least one task

processing interface based on the current task information by using a target large model and a target graph comprises:

determining parameter information based on the current task information by using the target large model, wherein the parameter information comprises task purpose information, and an input parameter type and an input parameter value of the task processing interface; and

determining the at least one task processing interface based on the parameter information and the target graph.

13. The computer system according to claim 12, wherein the target large model comprises a first large model and a second large model; and the determining parameter information based on the current task information by using the target large model comprises:

inputting the current task information into the first large model, to obtain task purpose information that describes a task intent; and

inputting the current task information and the task purpose information into the second large model, and determining the input parameter type and the input parameter value based on a result output by the second large model.

14. The computer system according to claim 13, wherein the determining the input parameter type and the input parameter value based on a result output by the second large model comprises:

obtaining the input parameter type and the input parameter value from the result output by the second large model; and

if the result output by the second large model does not comprise the input parameter value, obtaining the input parameter value based on the current task information, the input parameter type, and the task purpose information by using a preset regular expression.

15. The computer system according to claim 12, wherein the determining the at least one task processing interface based on the parameter information and the target graph comprises:

searching the target graph for a target path from an input parameter to an output parameter based on the parameter information; and

determining the at least one task processing interface based on the target path.

16. The computer system according to claim 10, wherein after the current result is obtained, the processor further implements the following steps when executing the program:

inputting the current result and the current task information into a third large model; and

determining, based on a result output by the third large model, whether processing of the work

order task is completed currently.

17. The computer system according to claim 10, wherein the target graph further comprises a plurality of second nodes representing parameter types, there is a connecting edge between nodes in the target graph, the connecting edge is a directed edge, and the connecting edge represents an input/output relationship between a candidate task processing interface and a parameter type.

18. The computer system according to claim 17, wherein the target graph is updated in the following manner:

obtaining a to-be-added candidate task processing interface;

determining an input parameter type and an output parameter type that correspond to the to-be-added candidate task processing interface;

traversing the node in the target graph based on the input parameter type and the output parameter type that correspond to the to-be-added candidate task processing interface, and determining a second node associated with the to-be-added candidate task processing interface; and

generating a new first node in the target graph for the to-be-added candidate task processing interface, and generating a connecting edge between the new first node and the associated second node.

19. A computer-readable storage medium, wherein the computer-readable storage medium stores a computer program, and when the computer program is executed on a computer, the computer is enabled to perform the method according to any one of claims 1 to 9.