(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
25 October 2001 (25.10.2001)

PCT

(10) International Publication Number
WO 01/80096 A1

(51) International Patent Classification^7: G06F 17/30

(21) International Application Number: PCT/US01/12721

(22) International Filing Date: 18 April 2001 (18.04.2001)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
09/550,818 18 April 2000 (18.04.2000) US

(71) Applicant: **INZIGO, INC.** [US/US]; 107 Sunnyvale Court, Somerset, NJ 08873 (US).

(72) Inventors: **KHAN, M., Mobeen**; 107 Sunnyvale Court, Somerset, NJ 08873 (US). **SIDDIQUI, Pervez, R.**; 3460 Peel Street, Montreal, Québec H3A 2M1 (CA).

(74) Agents: **HEIDELBERGER, Louis, M.** et al.; Reed Smith LLP, 2500 One Liberty Place, 1650 Market Street, Philadelphia, PA 19103 (US).

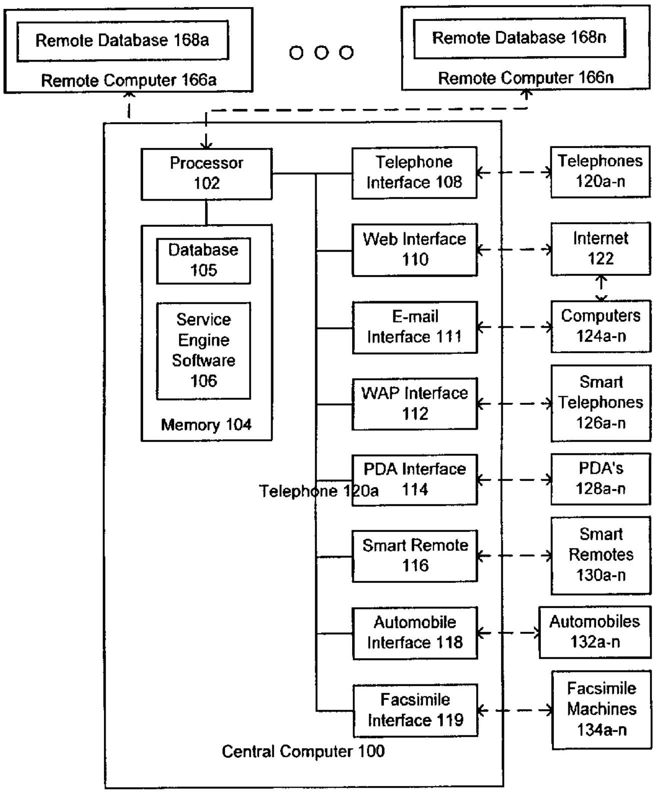
(81) Designated States *(national)*: AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, BZ, CA, CH, CN, CR, CU, CZ, DE, DK, DM, DZ, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, MZ, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, TZ, UA, UG, UZ, VN, YU, ZA, ZW.

(84) Designated States *(regional)*: ARIPO patent (GH, GM, KE, LS, MW, MZ, SD, SL, SZ, TZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE, TR), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG).

Published:
— *with international search report*

[Continued on next page]

(54) Title: SYSTEM AND METHOD FOR FULFILLING A USER'S REQUEST UTILIZING A SERVICE ENGINE



(57) Abstract: A system and method for fulfilling a user's request using a service engine (106), wherein the user is prompted to submit a request through an interface in natural language format (152), the request is evaluated, the user is repeatedly prompted with a question relating to the request if the evaluated request contains insufficient understood data to allow the service engine to fulfill the request until the evaluated request contains sufficient understood data to allow the service engine to fulfill the request, and the request is fulfilled.

— *before the expiration of the time limit for amending the claims and to be republished in the event of receipt of amendments*

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

SYSTEM AND METHOD FOR FULFILLING A USER'S REQUEST UTILIZING A SERVICE ENGINE

Field of the Invention

The present invention relates generally to fulfilling users' requests utilizing a service engine. More particularly, the present invention relates to fulfilling users, requests submitted through a plurality of interfaces utilizing a service engine.

Background

Numerous search engines for searching the Web and particular databases currently exist. Such search engines typically allow only Boolean searches of single databases (or the Web) and make no attempt to tailor searches to the subject matter being searched. Moreover, such search engines do not provide any further assistance to the user beyond returning a list of possibly relevant hits or canned responses. Thus, a user wishing to find the cheapest available flight between two cities on a particular day may search the Web for relevant sites and retrieve a list of sites containing both relevant hits, such as airline and discount reseller sites, and irrelevant hits. However, the user must then search each relevant site separately to uncover the cheapest available fare for the required dates and times. Moreover, the user must access the search engine via the Web (or a proprietary interface), which is not always convenient to the user. What is needed is a service engine capable of fulfilling users, requests submitted in natural language form from a variety of input devices with a high degree of accuracy using a variety of data sources particularly adapted to fulfilling users, needs in particular domains.

It is therefore an object of the present invention to provide a system and method for fulfilling user requests submitted through any one of a plurality of interfaces.

It is a further object of the present invention to provide a system and method for understanding user requests submitted in natural language form.

It is yet a further object of the present invention to provide a system and method for fulfilling user requests other than simple searches.

Summary of the Invention

A system and method for fulfilling a user's request using a service engine, wherein the user is prompted to submit a request through an interface in natural language format, the request is evaluated, the user is repeatedly prompted with a question relating to the request if the evaluated request contains insufficient understood data to allow the service engine to fulfill the request until the evaluated request contains sufficient understood data to allow the service engine to fulfill the request, and the request is fulfilled.

Brief Description of the Drawings

Figure 1A is a block diagram of a system in accordance with a preferred embodiment of the present invention.

Figure 1B is a diagram representing the relationships between software modules in a system in accordance with a preferred embodiment of the present invention.

Figure 2 is a flow diagram of a method in accordance with a preferred embodiment of the present invention.

Detailed Description of the Preferred Embodiments

The following definitions are provided to aid in construing the claims of the present application:

Automobile Communication System. A system embedded in a vehicle to permit communication between the vehicle or a user of the vehicle and a service center for emergency and non-emergency requests. The system may contain a vehicle location sub-system.

Complex Request. Any request other than a simple search request.

Intelligent Service Agent. A program that performs a task automatically, such as searching a database for records corresponding to a query or electronically reserving or purchasing tickets, such as airline tickets, or electronically purchasing a specified item. An intelligent service agent preferably understands specific domains queried by a user and assists in identifying and collecting all information relevant to fulfilling the user's request.

Interface. A combination of hardware and software that allows a user using a device, such as a computer, telephone, personal digital assistant, smart remote control, or automobile communication system to communicate with a service engine.

Natural Language Format. One or more sentences or portions thereof in a language commonly used for communication between humans (such as the English language). The sentences or portions of sentences may, but need not, be grammatically correct.

Request. An instruction to a service engine, such as an instruction to search for information relating to a query, an instruction to place an order, an instruction to make a reservation, or an instruction to perform several tasks, especially several related tasks.

Search Engine. A program that searches a body of information, such as a database or the Internet, for data corresponding to a query. A search engine is a service engine of a specialized and limited form.

Service Engine. A program that fulfills a user's requests, such as searching for data, automatically placing an order, or automatically making a reservation.

Smart Remote Control. A specialized interface that (i) permits a user to submit a query from a remote location, (ii) performs at least limited processing of the query, such as formatting-related processing, and (iii) forwards the query to at least one device for fulfillment. An example of a smart remote control is a television/cable remote control that allows a user to select, record, and search through programming.

Voice Recognition Interface. A telephone line, optionally other hardware, and software capable of receiving and translating spoken words submitted over the telephone line in natural language voice format into text format based on language specific phonetic and grammatical rules.

Web Interface. An Internet connection, optionally other hardware, and software capable of receiving and processing requests submitted over the Web by users utilizing Web browsers. A Web interface typically includes a Web server, an Internet connection such as a dial-up line, a T-1 or equivalent line, a network connection such as an Ethernet network to another computer responsible for parsing the requests, and software for receiving each request from the user, formatting each request, and forwarding each request to the computer responsible for parsing the requests.

Referring to figure 1A, a preferred embodiment of a system in accordance with the present invention is illustrated. Central computer 100, which may be a single computer or a group of computers connected together by a network or other means, includes at least processor 102, memory 104, and at least one interface. Preferably central computer 100 includes a bank of server class computers internally connected by high-speed lines and running Windows NT, Unix, or Linux and connections to communication systems. Memory 104 may be temporary memory, such as random access memory, or permanent storage, such as a hard drive, but is preferably a combination of both. Stored within memory 104 is service engine software 106, illustrated in greater detail in figure 1B.

Preferably central computer 100 includes at least two (and possibly all) of telephone interface 108, Web interface 110, email interface ill, Wireless Application Protocol (hereinafter 11WAP11) interface 112, personal digital assistant (hereinafter "PDA") interface 114, smart remote interface 116, automobile communication system interface 118, and facsimile interface 119. Other interfaces may also be utilized to interface with any communication device convenient to users. Telephone interface 108 may be connected to one or more of telephones 120a through 120n, Web interface 110 may be connected to one or more of computers 124a through 124n via the Internet 122, e-mail interface 111 may be connected to one or more of computers 124a through 124n via the Internet 122, WAP interface 112 may be connected to one or more of smart telephones 126a through 126n, PDA interface 114 may be connected to one or more of PDA's 128a through 128n, smart remote interface 116 may be connected to one or more of smart remote controls 130a through 130n, automobile communication system interface 118 may be connected to one or more of automobile communication systems 132a through 132n, and facsimile interface 119 may be connected to one or more of facsimile machines 133a through 133n. While the interfaces specifically described herein are described in isolation from each other for reasons of clarity, those skilled in the art will recognize that hardware and software components of the interfaces may in some instances be shared among interfaces.

Telephone interface 108 may include a set of termination points to the public or private telephony infrastructure through communication lines and trunks. These lines may terminate into general speech telephony cards, such as the Anteres card or the D41/ESC card provided by the Dialogic Corporation of Parsippany, New Jersey (www.dialogic.com), which may be inserted into ISA or PCI external board slots in the Central Computer. These cards may be used to handle call flow and manage voice calls and may be used to interface with generic speech recognition software such as Speechworks 6 available from the Speechworks corporation of Boston, Massachusetts (www.speechworks.com) or similar products available from Phillips, IBM, Lucent, and Nuance.

Web interface 110 may include a set of communication lines terminating into a bank of servers running standard Web protocols, such as Transmission Control Protocol/Internet Protocol ("TCP/IP"), and Hypertext Transmission Protocol (11HTTP11). These protocols may be used to manage Web connections and transmission of data in html or similar formats between the Central Computer and remote PC's and other devices used for browsing. E-mail interface ill may include a data interface running standard e-mail software, such as Outlook available from the Microsoft Corporation of Redmond, Washington (www.microsoft.com) or Eudora available from the Qualcomm Corporation of San Diego, California (www.qualcomm.com), on a bank of server farms connected through communication lines or trunks. The e-mail software receives and processes requests submitted by e-mail.

WAP interface 112 may include a bank of data servers running the WAP protocol (specified by the World Wide Web consortium) and communication lines connecting to the cellular mobile infrastructure.

PDA interface 114 may include a bank of data servers running PDA communication protocols, such as the Web clipping interface for the Palm VII PDA. The PDA interface is connected to the Internet and wireless infrastructure providers through communication lines.

Smart remote interface 116 may include server software to receive queries from a remote control through an infrared interface according to the Infrared Data Access ("IrDA") protocol.

Automobile communication system interface 118 may include a bank of voice and data servers running standard voice and data communications protocols, such as TCP/IP for data and wireless telephone protocols for voice, to receive queries from automobile systems. Facsimile interface 119 may include a bank of servers to receive facsimiles.

Central computer 100 may optionally be connected to one or more of remote computers 166a through 166n containing remote databases 168a through 168n by a wide area network or other means. Remote databases 168a through 168n may contain domain specific data. For example, remote database 168a might contain travel information (or airline flight information only) while remote database 168b might contain information about music. In addition, one or more local databases 105a through 105n are typically stored in memory 104. The local databases may also contain domain specific data. In addition, or alternatively, local databases 105a through 105n may contain user preferences, databases to aid in the parsing of requests, such as dictionaries and grammars that may (but need not) be domain specific, and databases relating to past requests of users. Remote databases 168a through 168n and local databases 105a through 105n may be relational, object oriented or other databases and each individual database may be of a different type, but preferably Oracle version 8 is used for all of the databases.

Service engine software 106 may be written using the Java programming language using object oriented programming methodologies. Referring to figure 1B, the relationships between several modules of service engine software 106 are illustrated. Telephone interface module 134, Web interface module 136, e-mail interface module 138, WAP interface module 140, PDA interface module 142, smart remote interface module 144, automobile communication system interface module 146, and facsimile interface module 148 form the software portions of telephone interface 108, Web interface 110, e-mail interface 111, WAP interface 112, PDA interface 114, smart remote interface 116, automobile communication system interface 118, and facsimile interface 119 respectively. Each request received from a user is received by the appropriate interface module, which then extracts the information submitted by the user and forwards it to format conversion module 150, which converts the extracted information into a common format for further use by the service engine software.

Format conversion module 150 then forwards the formatted information to natural language processing module 152, which parses the formatted information to determine its meaning, preferably utilizing a plurality of parsing algorithms, as described below. Natural language processing module 152 then forwards the parsed results to post processing module 154, which combines and evaluates the results to determine if they are understood. If the results are not understood, a follow-up question may be generated by response assembly module 162 and output packaging module 164 and returned to format conversion module 152 for dispatch to the user via the appropriate one of interface modules 134 through 148. Post processing module 154 then forwards the results to query generation module 156, which generates an appropriate query in an internally used format for forwarding to an appropriate intelligent service agent, which is preferably a domain specific agent. Intelligent service agents 158a through 158n preferably contain knowledge of variables (especially domain specific variables) and knowledge bases related to those domains and are able to generate follow-up questions for the user when additional information is necessary. The modified query is then passed to smart search module 160, which preferably contains multiple algorithms designed to interface with and search across multiple knowledge bases.

Response assembly module 162 assembles the output of the smart search module into a context specific response based on user preferences and the communications medium used by the user. Output packaging module 164 then packages the response for transmission to the user and forwards it to format converter 152 for format conversion and forwarding to an appropriate interface module.

Referring to figure 2, a method of fulfilling a request using a service engine is illustrated. In step 200, the user is prompted to submit a request through one of a plurality of interfaces. The user may be so prompted with respect to a Web interface by providing a Web page with a text entry control for entering a natural language request and a submit or execute button. Preferably, the user is first prompted (on the same Web page or on another Web page) to supply a username and password through the provision of text entry areas for the username and password and the provision of a login button.

The user may be so prompted with respect to a telephone (voice) interface by providing the user with a telephone number to call to access the system and providing a series of prerecorded message prompts prompting the user to enter a spoken natural language request. Preferably, the user is first prompted (with a pre-recorded message prompt) to supply a username and password either by speaking the username and password or, more preferably, by entering the username and password via an alphanumeric keypad. A biometric voiceprint may instead or in addition be used to help validate the user's identity.

The user may be so prompted with respect to an e-mail interface by providing an e-mail address to the user with directions to send A natural language request to such e-mail address. Preferably, the user is also prompted to supply an encrypted or plain text username and password in the same message.

The user may be so prompted with respect to a WAP interface by storing an address for the service engine on the user's smart wireless telephone and displaying a message or icon prompting the user to construct a query. The user may then construct a query and submit it to the service engine using the cellular data infrastructure (as either a short message service message or as packetized data).

The user may be so prompted with respect to a PDA interface by storing the uniform resource locator ("URL") of the service engine in the PDA and displaying an icon or message prompting the user to construct a query. The user may then type a query into the PDA or use a stored query to send a request to the service engine through a wireless network or otherwise.

The user may be so prompted with respect to a smart remote interface by prompting the user through a voice or data prompt on the remote control. The user may then construct the query using voice or soft keyboard input or may select a stored query.

The user may be so prompted with respect to an automobile communication system interface by providing the user with a call button or a voice interface for connecting to the service engine infrastructure. The user may then enter a query either using a voice interface or a keyboard interface or may optionally select a stored query.

The user may be so prompted with respect to a facsimile interface by providing the user with a facsimile number with directions to send a natural language request to such facsimile number. Preferably, the user is also prompted to supply an encrypted or plain text username and password in the same message.

In step 202, the request is received by the service engine. In the case of a telephone (voice) interface, the request is first evaluated by a recognition engine within the telephone interface. This engine has several associated databases, which allow it to detect phonemes in the request. It then constructs words of the requests and associates probabilities to each computed utterance. It then forwards this list of sentences along with the associated probabilities for further processing by the format conversion module.

For example, John Smithe, a registered user, might call 888-4TRAVEL to access a travel service. A user preference database might contain previously stored entries to the effect that John normally travels on United Airlines and has 65,000 frequent flyer miles accumulated. John might make the following request in response to step 200, "1 want to fly from New York to Seattle next weekend. Find me the cheapest fare available." The recognition engine would then access specialized databases containing specific grammar, dialect, and other speech recognition related information specific to the travel domain and possibly to the specific user. Such user specific databases might include requests previously entered by the user or by different users. Using this information, the recognition engine might produce the following outputs:

> i want to fly from new york to seattle next weekend find me the cheapest fare available (95%)
>
> i want fly from newark to seattle next week and find me the cheapest fare available (5%)

These outputs are then forwarded for format conversion and subsequent evaluation.

In the case of a Web or WAP interface, network connectivity software is used to perform several tasks including management of the logical connection between the user device and the server, managing network protocol (TCP/IP or WAP connectivity), collecting and forwarding user specific information (e.g. through deposited cookies) and identifying the service destination. once the user enters a textual request on his or her browser and presses the submit or execute button, a connection to the server is made. A text extraction sub-module in the Web or WAP interface module identifies the format in use and extracts the text portion of the query from it. It then forwards the request for further processing.

For example, John Smithe, a registered user, might launch a Web browser (similar to a personal computer Web browser) on a personal organizer utilizing WAP to connect to the Web and navigate to a travel web site using a bookmark. John might type the following request into a text box: "I wanna fly from NY to Seattle next weekend, find me cheapest fair" and press a submit button on his screen. The request is passed to the service engine using WAP and Wireless Markup Language (hereinafter "WML"). The text box accepts the query in WML format and strips all headers and tags from the query, converts the query into all lower case letters and then passes on for further processing.

Notwithstanding the above description of several alternative interfaces, those skilled in the art will appreciate that only one or a different number of interfaces or different interfaces may be used in implementing the present invention in lieu of or in addition to those interfaces described above.

In step 204, the format of a request is optionally converted to a common format. Step 204 may be combined with other steps, such as step 202, or dispensed with altogether (such as in the case of the use of only a single interface). In a preferred embodiment, however, plain text queries received by interface modules are forwarded to a format conversion module for conversion into XML format and inter alia the identification of key information.

The format conversion module places XML tags at the beginning and end of each request supplied by the user in the single session and identifies key information associated with the request. The time of the day the request was made, the user (if known), the method the request was received and the service requested (if known) are preferably added to the formatted request. This resulting representation is internal to the Service Engine. The XML request is not parsed at this stage.

For example, the above travel request "I want to fly from New York to Seattle next weekend. Find me the cheapest fare available." might be processed in the following manner. First, the current time and date might be identified. Second, John-Smithe-NZG might be identified as the user. Third, the service request might be identified as relating to the domain TRAVEL-PORTAL-PUBLIC. Fourth, the method might be identified as speech. Fifth, the device type might be identified if known. Sixth, the dialing number might be identified if known. Seventh, the communication provider might be identified if known. Eighth, the priority of the request might be identified. Ninth, multiple interpretations of the query (if applicable) might be identified. Of course, a particular implementation of the present invention might order the above information differently or might include additional, different, or less information.

In a preferred embodiment, the resulting XML query might resemble the following:

```
<XML-INZIGO-INTERNAL-FORMAT>
<TITLE> QUERY REQUEST </TITLE>
<XMLBODY>
<QUERYID> 000214-165042-T-1126JS </QUERY-ID>
<DATE> 02142000 </DATE>
<TIME> 16.50.42 </TIME>
<USER> John-Smithe-NZG </USER>
<SERVICETYPE> TRAVEL-PORTAL-PUBLIC </SERVICETYPE>
<ACCESS> SPEECH </ACCESS>
<DEVICE> CELLPHONE</DEVICE>
<CALLINGID> 732.742.7165</CALLINGID>
<COMMSERV> BELLATL </COMMSERV>
<PRIORITY> LOW </PRIORITY>
<LISTQUERY>
<QUERYINT> i want to fly from new york to seattle next weekend find me
the cheapest fare available <PROBFACT> 0.95 </PROBFACT>
</QUERYINT>
<QUERYINT> i want to fly from newark to seattle next week and find me the
cheapest fare available <PROBFACT> 0.05 </PROBFACT></QUERYINT>
</LISTQUERY>
</XMLBODY>
```

The resulting XML tagged query is logged and then passed on to the natural language processing module.

In step 206, the request is evaluated by at least one parsing algorithm. Preferably, a plurality of algorithms are utilized in parallel in step 206 and the results are then combined and evaluated in step 208. Typically, one or more databases are accessed by each algorithm. These databases may include grammars and dictionaries that may (but need not) be specific to the domain of the request. Databases of user preferences and previously submitted requests of the user and of other users may also be accessed. In a first preferred embodiment, four parallel parsing algorithms are used: case/tree parsing, keyword/phrase parsing, statistical parsing, and domain identity parsing. In particular embodiments of the present invention, a greater or lesser number of parsing algorithms may be used and different algorithms may be utilized in addition to or instead of those here described. The case/tree parsing algorithm uses semantic based parsing methods to process the query. It uses (multiple) domain based, a priori-determined grammar and dictionary databases. The algorithm attempts to match the query with the closest possible outcomes and provide the probability factor to it. Continuing with the above example, once the XML Query is forwarded to this algorithm, each of the <QUERYINT> (query interpretations) is processed independently. The following modifications are made to the XML Query:

```
<LISTQUERY>
<QUERYINT> i want to fly from new york to seattle next weekend find me
the cheapest fare available
<PROBFACT> 0.95 </PROBFACT>
<REQUESTINT> TRV DATASET A464666874TRV <IREQUEST-TNT>
<REQPROB> 1.0 <1REQPROB>
<CONSTANT> air travel <ICONSTANT>
<CONSTANT> new york <ICONSTANT>
<CONSTANT> seattle <ICONSTANT>
<VARIABLE> next weekend <IVARIABLE>
<VARIABLE> lowfare <IVARIABLE>
</QUERYINT>
<QUERYINT> i want to fly from newark to seattle next week and find me the
cheapest fare available
<PROBFACT> 0.05 </PROBFACT>
<REQUESTINT> TRV DATASET A464666874TRV
< IREQUESTINT>
<REQPROB> 1.0 <IREQPROB>
<CONSTANT> air travel <ICONSTANT>
```

```
<CONSTANT> new york <ICONSTANT>
<CONSTANT> seattle <1CONSTANT>
<VARIABLE> next weekend </VARIABLE>
<VARIABLE> lowfare <IVARIABLE>
</QUERYINT>
```

The keyword/phrase parsing algorithm identifies the keywords embedded in each request. It has two functions, first to identify the attributes within the query that are constants or variables and would assist in conducting multiple searches to assemble a response and second to use phrases and keywords from the request to understand the context of the request. it accesses domain specific keyword and phrase databases. The output of this sub-module is addition of keywords to the XML request and estimates on the context of the phrase.

Continuing with the above example, each of the <QUERYINT> fields are processed by the keyword/phrase parsing algorithm. First the algorithm modifies the XML Query by adding the following fields to the two <QUERYINT> variables:

```
<LISTQUERY>
<QUERYINT> i want to fly from new york to seattle next weekend find me
the cheapest fare available
<PROBFACT> 0.95 </PROBFACT>
<CONSTANT> new york <ICONSTANT>
<CONSTANT> seattle <ICONSTANT>
<VARIABLE> next weekend <IVARIABLE>
<VARIABLE> lorwfare <IVARIA13LE>
</QUERYINT>
<QUERYINT> i want to fly from newark to seattle next week and find me the
cheapest fare available
<PROBFACT> 0.05 </PROBFACT>
<CONSTANT> newark </CONSTANT>
<CONSTANT> seattle </CONSTANT>
<VARIABLE> next week</VARIABLE>
```

```
<VARIABLE> lowfare </VARIABLE>
</QUERYINT>
</LISTQUERY>
```

Second, the algorithm identifies the following phrases from the two query interpretations and adds to the XML request as follows.

```
<LISTQUERY>
<QUERYINT> i want to fly from new york to seattle next weekend find me
the cheapest fare available
<PROBFACT> 0.95 </PROBFACT>
<KEYWRD-PHP,ASE> fly </KEYWRb-PHRASE>
<KEYWRD-PHRASE> from new york </KEYWRD-PHPASE>
<KEYWRD-PHP,ASE> to seattle </KEYWRD-PHPASE>
<KEYWRD-PHP,ASE> next weekend </KEYWRD-PHPASE>
<KEYWRD-PHRASE> find cheapest fare
</KEYWRD-PHPASE>
</QUERYINT>
<QUERYINT> i want to fly from newark to seattle next week and find me the
cheapest fare available
<PROBFACT> 0.05 </PROBFACT>
<KEYWRD-PHP,ASE> fly </KEYWRD-PHRASE>
<KEYWRD-PHRASE> from newark </KEYWRD-PHRASE>
<KEYWRD-PHP,ASE> to seattle </KBYWRD-PHRASE>
<KEYWRD-PHRASE> next week
</KEYWRD-PHRASE><KEYWRD-PHRASE> find cheapest fare
</KEYWRD-PHRASE>
</QUERYINT>
</LISTQUERY>
```

The statistical parsing algorithm applies statistical methodologies to arrive at the meaning and context of requests. It uses a knowledge base of past learned data and uses each request to update its learning of specific requests. Continuing with the above example, the XML request may be modified by the statistical parsing algorithm in the following manner.

```
<LISTQUERY>
<QUERYINT> i want to fly from new york to seattle next weekend find me
the cheapest fare available
<PROBFACT> 0.95 </PROBFACT>
<KEYWRD-PHP,ASE> fly </KEYWRD-PHRASE>
<KEYWRD-PHRASE> from new york </KEYWRD-PHRASE>
<KEYWRD-PHRASE> to seattle </KEYWRD-PHRASE>
<KEYWRD-PHRASE> next weekend </KEYWRD-PHRASE>
<KEYWRD-PHRASE> find cheapest fare </KEYWRD-PHRASE>
</QUERYINT>
<QUERYINT> i want to fly from newark to seattle next week and find me the
cheapest fare available
<PROBFACT> 0.05 </PROBFACT>
<KEYWRD-PHRASE> fly </KEYWRD-PHRASE>
<KEYWRD-PHRASE`> from newark </KEYWRD-PHRASE>
<KEYWRD-PHRASE> to seattle </KEYWRD-PHRASE>
<KEYWRD PHRASE> next week </KEYWRD-PHRASE>
<KEYWRD-PHRASE> find cheapest fare </KEYWRD-PHRASE>
</QUERYINT>
</LISTQUERY>
```

The domain identity parsing algorithm performs the task of identifying the service domain that is required to activate to answer the user queries. It processes the request by identifying the keywords and types of phrases in the request and suggests one or several domains with assigned probability factors.

Continuing with the above example, the domain identity parsing algorithm detects the words 11flyll, 11from", "to" and "fare". These keywords match the travel domain and hence the output of the module is to identify the travel domain with a 100% certainty. This output will also be appended to the XML request in the following format:

<LISTDOMAIN>
TRAVEL <PROBFACT> 1.0 </PROBFACT>
</LISTDOMAIN>

In step 208, the output of the parsing algorithms is combined into a single request. One or more combining algorithms are used to select the domain of the request and identify the key information provided in the request to conduct information retrieval or execute tasks. The algorithms are preferably based on probabilistic methods. In its simplest form, this algorithm may be a weighted-average of the probabilities or selection of information based on highest probability results. Any information that has an associated probability below a predetermined value may be discarded at this time. (Many other algorithms may be used instead including algorithms that generate multiple alternative results and algorithms that select results based on voting mechanisms. Alternatively, algorithms utilizing template matching may be used, whereby a user's query is matched to the stored template best matching the query. In embodiments in which only a single parsing algorithm is utilized, step 208 may be omitted.)

Continuing with the above example, in a first preferred embodiment, information may be extracted from the parsing algorithms as follows. Travel is identified as the domain as the probability of the domain is identified as one hundred percent and from the keywords and constants identified from different sub-modules. <QUERYINT> number 2 is rejected (with a five percent probability) may be rejected as having an unduly low probability of occurrence. The information contained in the request is identified as "fly", "new york", 11seattle", "next weekend", "cheapest", and "fare".

In step 210, in a first preferred embodiment, the request is mapped by an intelligent service agent to a template relating to the domain of the request. In other embodiments, however, templates may relate to more than one domain. In certain embodiments, step 210 may be omitted altogether and a simple Boolean or other search may be generated. In the preferred embodiments, however, an XML template relating to the domain of the request (and possibly to certain keywords in the request) is retrieved. Each template preferably contains several types of keywords, including required variables, secondary variables and search variables. The required variables are those necessary to conduct any relevant search for the information request to be processed. The secondary variables are not necessary but extremely helpful in pinpointing the information requested and the search variables identify the requested information. Each of these variables can be represented by a value or by a range of values. If all of the required variables are not available or have low associated confidence levels, the intelligent agent will initiate a dialog with the user to obtain further information in step 212.

Continuing with the above example, the request may consist of the following XML representation:

```
<TITLE> XML TRAVEL REPRESENTATION </TITLE>
<TRAVELTEMP>
<MODE> AIR </MODE>
<LISTREQVAR>
<ORIG> new york </ORIG>
<DEST> seattle </DEST>
<DATE> Feb 00 11 -13 </DATE>
</LISTREQVAR>
<LISTSECVAR>
<CARR> Open </CARR>
<TIME> Open </TIME>
<CLSS> Open </CLSS>
<MEAL> Open </MEAL>
<ARPT> Open </ARPT>
</LISTSECVAR>
</LISTSRCVAR>
<V1> Fare <CRITERIA> Low </Criteria> </Vl>
</TRAVELTEMP>
```

In step 212, the intelligent agent prompts the user to supply any essential information that is not understood or any essential information that has not been supplied by the user. The follow up questions are preferably presented to the user utilizing the same interface that received the original request. Continuing with the above example, if the above request did not specify the date or dates of travel to be "next weekend", the intelligent agent might prompt the user with the question "What date or dates would you like to travel?" and process the user's response.

In addition, secondary variables may be given values (if not earlier supplied by the user) by accessing a user preference database. For example, in the above example, the intelligent agent might identify airport preference to fill the field <ARPT> or carrier preference <CARR> from a user profile in such a database by retrieving records indicating that the user departed on ninety percent of previous recorded flights from LaGuardia Airport in New York and belonged to the American Airlines frequent flier program.

In step 214, any search necessary to fulfill the request is performed. A smart search algorithm is preferably utilized to select and access one or more relevant knowledge bases or other databases (or part or all of the Web or Internet) and to perform a search of each such source of data based on the XML request produced by steps 204 through 212 and also based on the architecture and formats of the several sources of data. Continuing with the above example, the smart search algorithm would select two databases to locate the least expensive domestic airline fares, one listing standard airline fares and one listing wholesale and discount travel operator fares. Depending on the architecture and format of each database, the algorithm generates appropriate searches, connects with remote databases, and retrieves information for the dates requested.

At this point, any other actions necessary to fulfill the user's request may be performed. For example, if the user had requested that the service engine reserve the least expensive seat on the applicable flight, instead of merely identifying it, the intelligent agent might generate an order automatically using previously stored user information, such as a credit card number of the user and the user's seating preferences. This order might then be sent automatically to the applicable airline or other vendor and a confirmation might be included with the results of the search generated in step 214. Similarly, the service engine might purchase goods or services on behalf of a user, request information automatically (such as course catalogs and applications from colleges meeting certain criteria), or perform other tasks automatically.

In step 216, the results generated by the service engine are assembled into a response. The content and format of this response may be based on the interface through which the response will be returned to the user. For example, if the response is to be returned to the user through a WAP interface, only the least expensive fare (or only a few of the least expensive fares) in the above example might be returned in the response, while if the response is to be returned through an email interface, a substantially greater number of fares (perhaps relating to flights departing at different times or with different numbers of connections) might be returned.

In step 218, the assembled response is packaged based on the particular interface through which the response will be transmitted. For example, a response to be returned over a (voice) telephone interface might be packaged into a collection of statements to be played to the user, while a response to be returned over a Web interface might be formatted into a table.

In step 220, the response is sent to the user using the designated interface. Optionally, the response may be sent to the user using more than one interface or may be sent to a different person designated by the user (either in the user's previously stored preferences or in the text of the request itself).

The present invention may be embodied in other specific forms without departing from the spirit or essential attributes of the invention. Accordingly, reference should be made to the appended claims, rather than the foregoing specification, as indicating the scope of the invention.

That which is claimed is:

1. A service engine, comprising:

a processor;

a memory connected to said processor; and

a plurality of interfaces, comprising at least a Web interface and a voice recognition interface,

wherein said processor receives a request through one of said plurality of interfaces;

wherein said processor searches a body of information in response to said request; and

wherein said processor returns the result of the search through one of said plurality of interfaces.

2. The service engine of claim 1, wherein said processor identifies the context of the request; and

wherein the body of information searched in response to the request is chosen by the processor based on the context of the request.

3. The service engine of claim 1, wherein said plurality of interfaces further comprises an interface for receiving requests from and sending responses to a personal digital assistant.

4. The service engine of claim 1, wherein said plurality of interfaces further comprises an interface for receiving requests from and sending responses to an automobile communication system.

5. The service engine of claim 1, wherein said plurality of interfaces further comprises an interface for receiving requests from and sending responses to a smart remote control.

6. The service engine of claim 1, wherein said processor returns the result of the search through the interface through which the request was received.

7. The service engine of claim 1, wherein said processor returns the result of the search through an interface selected based on previously stored user preferences.

8. The service engine of claim 1, wherein said processor returns the result of the search through an interface selected based on.the size of the result.

9. The service engine of claim 1, wherein said processor returns the result of the search through an interface selected based on the type of information in the result.

10. The service engine of claim 1, wherein said processor additionally returns the result of the search through a second interface.

11. The service engine of claim 1, wherein said processor determines the meaning of the request based on previous requests of the originator of the request.

12. The service engine of claim 1, wherein said processor determines the meaning of the request based on previously stored user preferences.

13. The service engine of claim 1, wherein said processor determines the meaning of the request based on predetermined rules.

14. The service engine of claim 1, wherein the body of information searched by the service engine includes the Web.

15. The service engine of claim 1, wherein the body of information searched by the service engine includes the Internet.

16. The service engine of claim 1, wherein the body of information searched by the service engine includes at least one database.

17. The service engine of claim 16, wherein the at least one database comprises a local database.

18. The service engine of claim 16, wherein the at least one database comprises a remote database.

19. The service engine of claim 16, wherein the at least one database comprises a proprietary database.

20. The service engine of claim 16, wherein the at least one database comprises a publicly accessible database.

21. A method of fulfilling a request using a service engine, comprising the steps of:

(a) prompting a user to submit a request through one of a plurality of interfaces;

(b) searching a body of information in response to the submission of a request; and

(c) returning the result of the search through one of the plurality of interfaces.

22. A service engine, comprising:

means for prompting a user to submit a request through one of a plurality of interfaces;

means for searching a body of information in response to the submission of a request; and

means for returning the result of the search through one of the plurality of interfaces.

23. A computer-readable medium tangibly embodying instructions which, when executed by a computer, implement a process comprising the steps of:

(a) prompting a user to submit a request through one of a plurality of interfaces;

(b) searching a body of information in response to the submission of a request; and

(c) returning the result of the search through one of the plurality of interfaces.

24. A method of fulfilling a user's request using a service engine, comprising the steps of:

(a) prompting a user to submit a request through an interface in natural language format;

(b) evaluating the request;

(c) if the evaluated request contains insufficient understood data to allow the service engine to fulfill the request, prompting the user with a question relating to the request;

(d) repeating steps (b) and (c) until the evaluated request contains sufficient understood data to allow the service engine to fulfill the request; and

(e) fulfilling the request.

25. The method of claim 24, further comprising the step of:

(f) converting the request into a common format upon receipt of the request through an interface.

26. The method of claim 24,wherein the request is evaluated using domain rules.

27. The method of claim 24,wherein the request is evaluated using grammatical rules.

28. The method of claim 24,wherein the request is evaluated using statistical rules.

29. The method of claim 24,wherein the request is evaluated based on requests previously submitted by the same user.

30. The method of claim 24,wherein the request is evaluated based on requests previously submitted by other users.

31. The method of claim 24,wherein the request is mapped to at least one previously stored template; and

wherein the understood data is determined to be sufficient if the understood data comprise data relating to all required fields in the at least one previously stored template.

32. The method of claim 24,wherein step (e) comprises performing a search for requested information.

33. The method of claim 32, wherein the search is performed using multiple databases.

34. The method of claim 32, wherein the search is performed based on previously stored user preferences.

35. The method of claim 24,wherein step (e) comprises making a reservation.

36. The method of claim 24,wherein step (e) comprises making a purchase.

37. The method of claim 24,wherein step (e) comprises transmitting data to a person other than the user.

38. A system for fulfilling a user's request using a service engine, comprising:

- processor;

- memory connected to said processor; and at least one interface in communication with said processor, wherein said processor prompts a user to submit a request through one of said at least one interface in natural language format; wherein said processor evaluates the request; wherein said processor repeatedly causes the user to be prompted with a question relating to the request if the evaluated request contains insufficient understood data to allow said processor to cause the request to be fulfilled until the evaluated request contains sufficient understood data to allow aid processor to cause the request to be fulfilled; and wherein said processor causes the request to be fulfilled.

39. A system for fulfilling a user's request using a service engine, comprising:

means for prompting a user to submit a request through an interface in natural language format; means for evaluating the request; means for prompting the user with a question relating to the request if the evaluated request contains insufficient understood data to allow the service engine to fulfill the request until the evaluated request contains sufficient understood data to allow the service engine to fulfill the request; and

means for fulfilling the request.

40. A computer-readable medium tangibly embodying instructions which, when executed by a computer, implement a process comprising the steps of:

(a) prompting a user to submit a request through an interface in natural language format;

(b) evaluating the request;

(c) if the evaluated request contains insufficient understood data to allow the service engine to fulfill the request, prompting the user with a question relating to the request;

(d) repeating steps (b) and (c) until the evaluated request contains sufficient understood data to allow the service engine to fulfill the request; and

(e) fulfilling the request.

41. A method of evaluating a request submitted to a service engine in natural language format, comprising the steps of:

(a) parsing the request using a plurality of parsing algorithms in parallel;

(b) combining the results of step (a) using a combining algorithm; and

(c) determining the meaning of the request based on the results of step (b).

42. The method of claim 41, wherein the plurality of parsing algorithms includes a case/tree parsing algorithm.

43. The method of claim 41, wherein the plurality of parsing algorithms includes a keyword/phrase parsing algorithm.

44. The method of claim 41, wherein the plurality of parsing algorithms includes a statistical parsing algorithm.

45. The method of claim 41, wherein the plurality of parsing algorithms includes a domain identity parsing algorithm.

46. The method of claim 41, wherein the combining algorithm comprises assigning a degree of confidence to the product of each parsing algorithm.

47. The method of claim 46, wherein the combining algorithm selects the parsed product with the highest assigned degree of confidence.

48. The method of claim 41, wherein the combining algorithm comprises selecting data from more than one parsed product and assembling the selected data into a single combined product.

49. The method of claim 41, wherein the combining algorithm comprises selecting data from more than one parsed product and assembling the selected data into a plurality of alternative combined products.

50. The method of claim 41, wherein the combining algorithm comprises determining the extent to which each parsed product is similar to each other parsed product.

51. A system for evaluating a request submitted to a service engine in natural language format, comprising:

processor; and

memory in communication with said processor,

wherein said processor parses the request using a plurality of parsing algorithms in parallel;

wherein said processor combines the results of the parallel parsing using a combining algorithm; and

wherein said processor determines the meaning of the request based on the results of the combination.

52. A system for evaluating a request submitted to a service engine in natural language format, comprising:

means for parsing the request using a plurality of parsing algorithms in parallel;

means for combining the results of the parsing using a combining algorithm; and

means for determining the meaning of the request based on the results of the combination.

53. A computer-readable medium tangibly embodying instructions which, when executed by a computer, implement a process comprising the steps of:

(a) parsing the request using a plurality of parsing algorithms in parallel;

(b) combining the results of step (a) using a combining algorithm; and

(c) determining the meaning of the request based on the results of step (b).

54. A method of evaluating a request submitted to a service engine, comprising the steps of:

(a) parsing the request using at least one database;

(b) determining the meaning of the request based on the results of step (a);

(c) assigning a probability of correctness to the determined meaning;

(d) if the probability of correctness assigned in step (c) is below a predetermined value, prompting the user with a question relating to the request;

(e) repeating steps (a) through (d) until the probability of correctness is at least equal to the predetermined value; and

(f) when the probability of correctness is at least equal to the predetermined value, updating the database with the determined meaning of the request.

55. A system for evaluating a request submitted to a service engine, comprising:

a processor; and

a memory connected to said processor,

wherein said processor parses the request using at least one database;

wherein said processor determines the meaning of the request based on the results of the parsing;

wherein said processor assigns a probability of correctness to the determined meaning;

wherein said processor repeatedly causes the user to be prompted with a question relating to the request if the assigned probability of correctness is below a predetermined value until the probability of correctness is at least equal to the predetermined value; and

wherein said processor updates the database with the determined meaning of the request when the probability of correctness is at least equal to the predetermined value.

56. A system for evaluating a request submitted to a service engine, comprising:

means for parsing the request using at least one database;

means for determining the meaning of the request based on the results of the parsing;

means for assigning a probability of correctness to the determined meaning;

means for repeatedly prompting the user with a question relating to the request if the assigned probability of correctness is below a predetermined value until the probability of correctness is at least equal to the predetermined value; and

means for updating the database with the determined meaning of the request when the probability of correctness is at least equal to the predetermined value.

57. A computer-readable medium tangibly embodying instructions which, when executed by a computer, implement a process comprising the steps of:

(a) parsing the request using at least one database;

(b) determining the meaning of the request based on the results of step (a);

(c) assigning a probability of correctness to the determined meaning;

(d) if the probability of correctness assigned in step (c) is below a predetermined value, prompting the user with a question relating to the request;

(e) repeating steps (a) through (d) until the probability of correctness is at least equal to the predetermined value; and

(f) when the probability of correctness is at least equal to the predetermined value, updating the database with the determined meaning of the request.

58. A method of executing a complex request submitted to a service engine, comprising the steps of:

(a) prompting a user to enter a complex request in natural language format;

(b) evaluating the meaning of the complex request;

(c) determining whether the complex request is complete;

(d) if the complex request is determined in step (c) to be incomplete, prompting the user with a question regarding the complex request;

(e) repeating steps (c) and (d) until the complex request is determined in step (c) to be complete;

(f) if additional data is necessary to fulfill the complex request, searching a body of knowledge; and

(g) executing the complex request using an intelligent service agent.

59. The method of claim 58, wherein the complex request comprises a request for information.

60. The method of claim 58, wherein the complex request comprises a request to make a purchase.

61. The method of claim 58, wherein the complex request comprises a request to make! a reservation.

62. The method of claim 58, wherein the body of knowledge comprises the Web.

63. The method of claim 58, wherein the body of knowledge comprises the Internet.

64. The method of claim 58, wherein the body of knowledge comprises a proprietary database.

65. The method of claim 58, wherein the body of knowledge comprises a publicly accessible database.

66. The method of claim 58, wherein the intelligent service agent is selected based on the domain to which the complex request relates.

67. A system for executing a complex request submitted to a service engine, comprising: processor;

a memory connected to said processor; and

at least one interface in communication with said processor,

wherein said processor causes a user to be prompted to enter a complex request in natural language format;

wherein said processor evaluates the meaning of the complex request;

wherein said processor determines whether the complex request is complete;

wherein said processor repeatedly prompts the user with a question regarding the complex request if the complex request is determined to be incomplete until the complex request is determined to be complete;

wherein said processor searching a body of knowledge if additional data is necessary to fulfill the complex request; and

wherein said processor executes the complex request using an intelligent service agent.

68. A system for executing a complex request submitted to a service engine, comprising:

means for prompting a user to enter a complex request in natural language format;

means for evaluating the meaning of the complex request;

means for determining whether the complex request is complete;

means for repeatedly prompting the user with a question regarding the complex request if the complex request is determined to be incomplete until the complex request is determined to be complete;

means for searching a body of knowledge if additional data is necessary to fulfill the complex request; and

means for executing the complex request using an intelligent service agent.

69. A computer-readable medium tangibly embodying instructions which, when executed by a computer, implement a process comprising the steps of:

(a) prompting a user to enter a complex request in natural language format;

(b) evaluating the meaning of the complex request;

(c) determining whether the complex request is complete;

(d) if the complex request is determined in step (c) to be incomplete, prompting the user with a question regarding the complex request;

(e) repeating steps (c) and (d) until the complex request is determined in step (c) to be complete;

(f) if additional data is necessary to fulfill the complex request, searching a body of knowledge; and

(g) executing the complex request using an intelligent service agent.

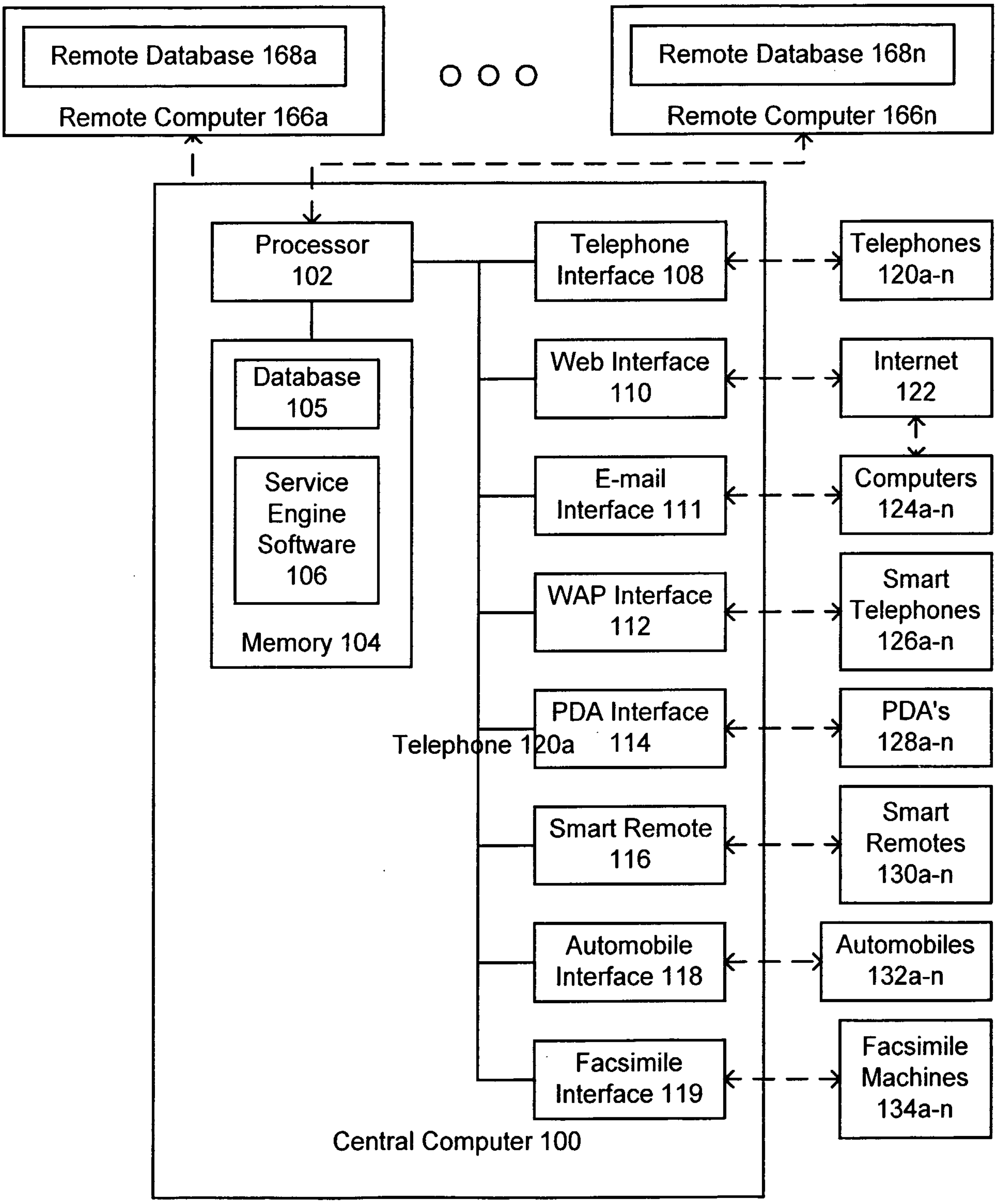

Remote Database 168a
Remote Computer 166a
O O O
Remote Database 168n
Remote Computer 166n
Processor 102
Database 105
Service Engine Software 106
Memory 104
Telephone Interface 108
Web Interface 110
E-mail Interface 111
WAP Interface 112
PDA Interface 114
Telephone 120a
Smart Remote 116
Automobile Interface 118
Facsimile Interface 119
Central Computer 100
Telephones 120a-n
Internet 122
Computers 124a-n
Smart Telephones 126a-n
PDA's 128a-n
Smart Remotes 130a-n
Automobiles 132a-n
Facsimile Machines 134a-n

Figure 1A

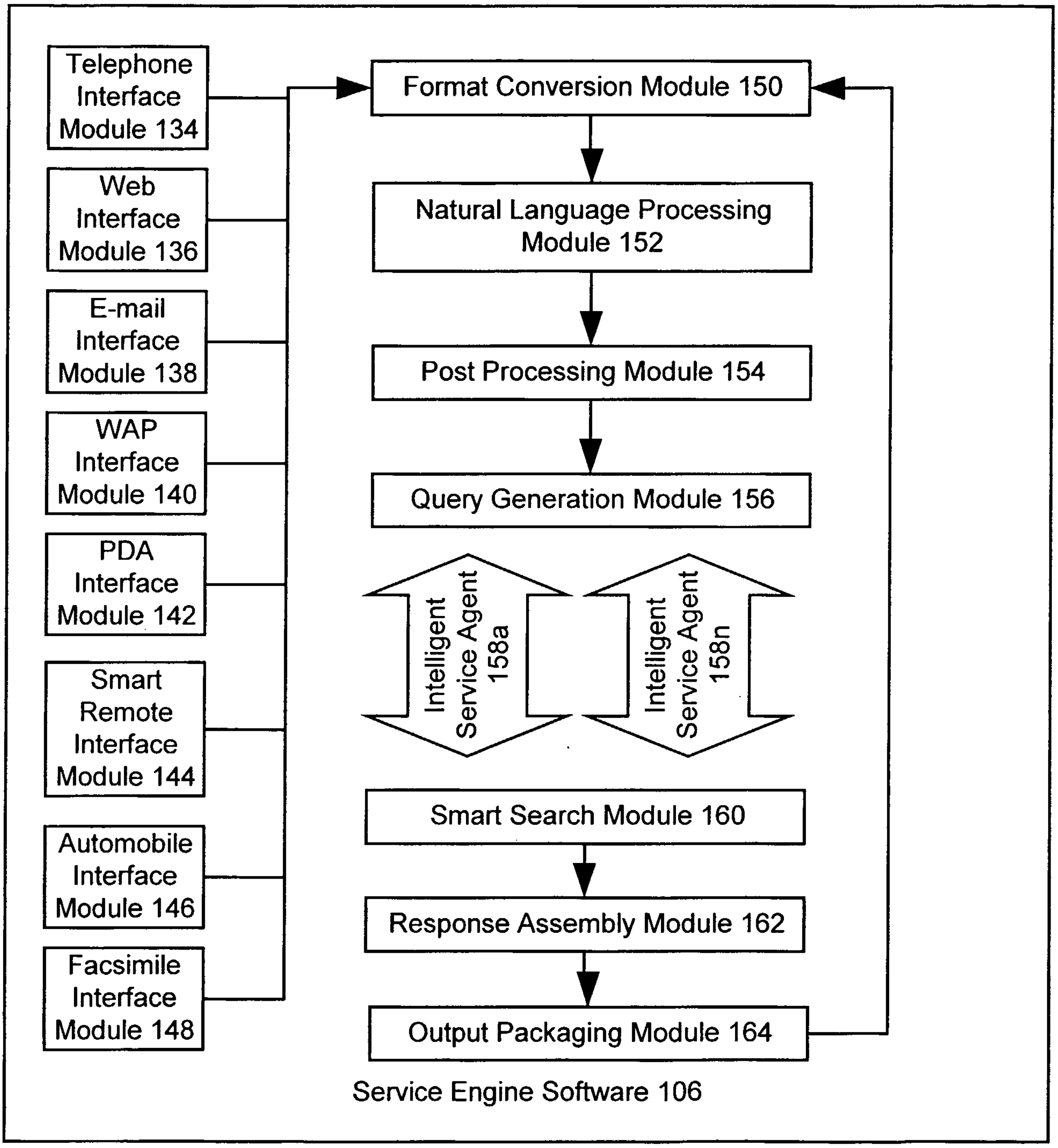

Telephone Interface Module 134
Web Interface Module 136
E-mail Interface Module 138
WAP Interface Module 140
PDA Interface Module 142
Smart Remote Interface Module 144
Automobile Interface Module 146
Facsimile Interface Module 148
Format Conversion Module 150
Natural Language Processing Module 152
Post Processing Module 154
Query Generation Module 156
Intelligent Service Agent 158a
Intelligent Service Agent 158n
Smart Search Module 160
Response Assembly Module 162
Output Packaging Module 164
Service Engine Software 106

Figure 1B

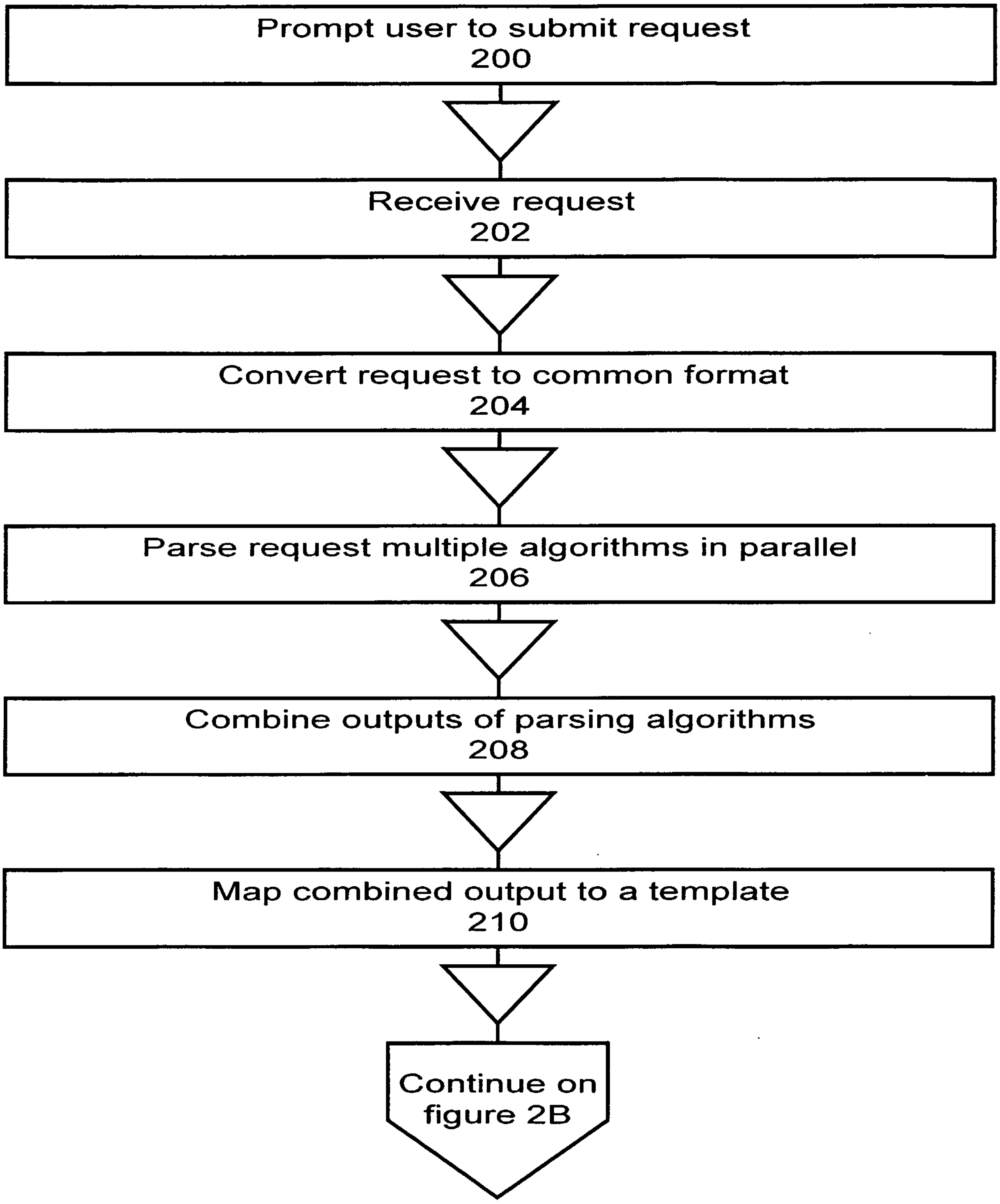

Prompt user to submit request
200
Receive request
202
Convert request to common format
204
Parse request multiple algorithms in parallel
206
Combine outputs of parsing algorithms
208
Map combined output to a template
210
Continue on figure 2B

Figure 2A

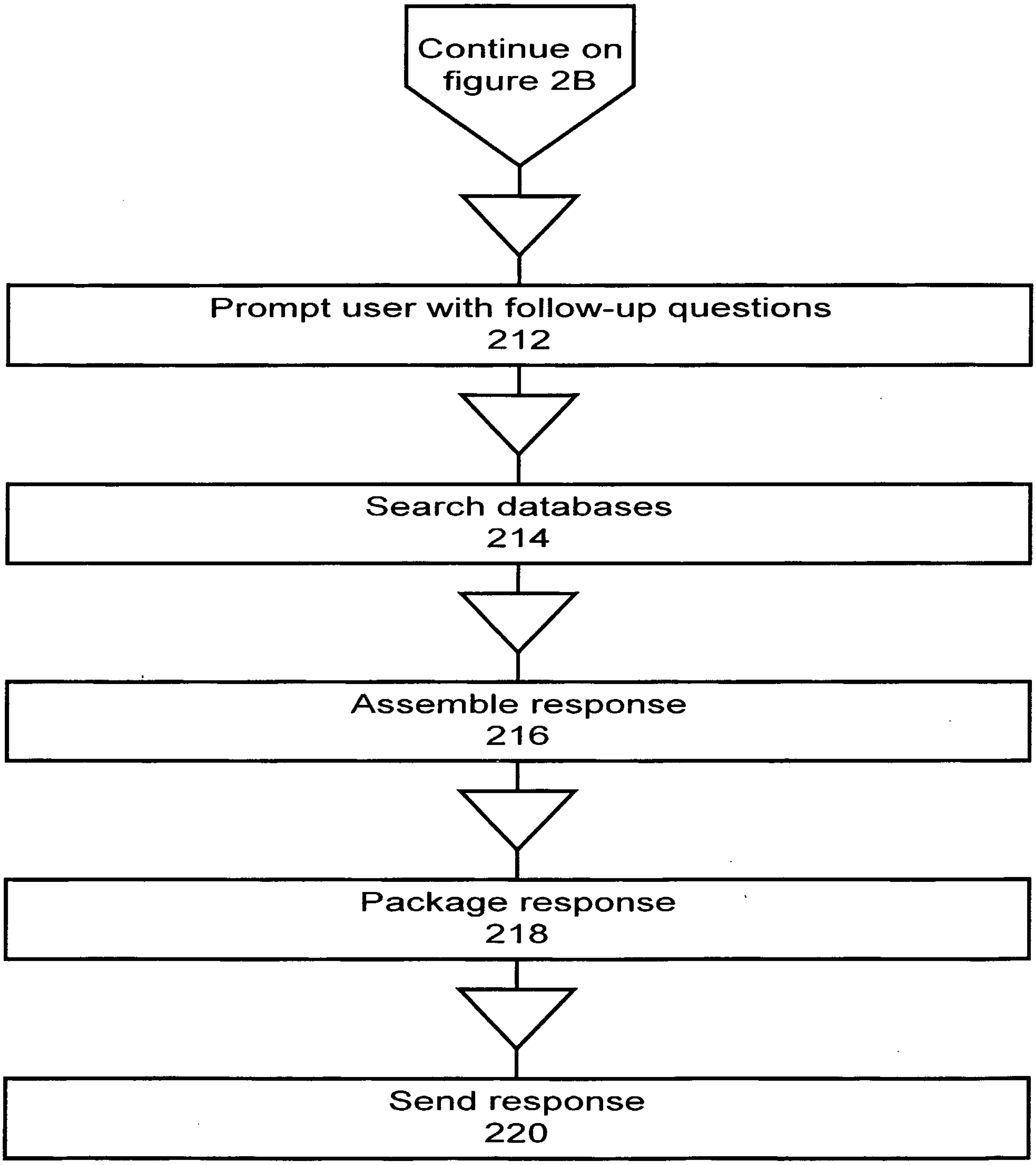

Continue on figure 2B
Prompt user with follow-up questions 212
Search databases 214
Assemble response 216
Package response 218
Send response 220

Figure 2B

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US01/12721

A. CLASSIFICATION OF SUBJECT MATTER

IPC(7) :GO6F:17/30
US CL :707/2, 3, 4; 709/201
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)

U.S. : 707/2, 3, 4; 709/201

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)

WEST, EAST

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	US 6,006,218 A (BREESE et al) 21 December 1999, Abstract, lines 1-26.	1-69
X	US 5,958,003 A (PREINING et al) 28 September 1999, Abstract, paragraphs 1-3.	1-39
Y	US 6,006,251 A (TOYOUCHI et al) 21 December 1999, Abstract, lines 1-19.	40-69

□ 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 document 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

27 JULY 2001

Date of mailing of the international search report

21 AUG 2001

Name and mailing address of the ISA/US
Commissioner of Patents and Trademarks
Box PCT
Washington, D.C. 20231
Facsimile No. (703) 305-3230

Authorized officer
JOHN BREENE James R. Matthews
Telephone No. (703) 305-9790