2014

ISSN : 0974 - 7435

Volume 10 Issue 24

BioTechnology

An Indian Journal

FULL PAPER

BTAIJ, 10(24), 2014 [16240-16255]

Design and implementation of context driven SoftMan knowledge communication framework

Wu Dan-feng[1,2,3*], Zeng Guangping[1,3], Cui Weihui[4], Xu Xiao-wei[1,3], Wang Kang[1,3]

[1]School of Computer & Communication Engineering, University of Science & Technology Beijing, Beijing 100083, (CHINA)

[2]School of Software, Liaoning Technical University, Huludao Liaoning 125105, (CHINA)

[3]Beijing Key Laboratory of Knowledge Engineering for Materials Science, Beijing 100083, (CHINA)

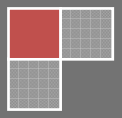
[4]Langfang Yanjing Career Technical College, Langfang Hebei 065200, (CHINA)

ABSTRACT

In recent years, with the deepening research of SoftMan, the traditional SoftMan communication mode has begun to appear insufficient on the ability of expression and communication efficiency. Aiming at the problems, based on the study of previous SoftMan system and its communication theory as well as the SoftMan Knowledge model and context awareness mechanism, the paper draws lessons from the mature language specification of Agent communication, proposes the CSMKC (Context driven SoftMan Knowledge Communication) framework, designs and implements its message layer, Knowledge layer and scene layer detailedly, achieves the communication of knowledge level between different SoftMans as well as the maintenance of situational context basically. Experimental results show that when the system is to complete a task, the higher situation dependence, the more obvious advantages of CSMKC in communication consumption.

KEYWORDS

Context driven; SoftMan; Knowledge communication.



INTRODUCTION

In the SoftMan System[1-2], each SoftMan is intelligent, self-controlled and authropomorphic computational entity. Each of them plays their respective roles, but their computing power is limited. In order to make the SoftMan population come together to complete complex tasks[3-6] coordinately and corporately[7-11], it is necessary to ensure the good communication between them, which is based on the design of communication framework[12-15] and language[16-17]. In the field of software, in terms of theoretical structure and implementation, SoftMan is a sublimation of Agent, and scholars home and abroad have made deep study about it. Kone defined it as a framework model, which shields the heterogeneous layer for the interaction between Agents[18]; TILAB and Parma University jointly developed a Java-based Agent development framework JADE, which implements the FIPA-ACL perfectly and contains all the necessary components based on FIPA platform[19]; Singh considered ACL as an information exchange mechanism within the Agent community[20]; Zhang Shijie et al applied KQML on the implementation of the CAPP system[21]; Huang Jianqing et al combined CORBA with KQML and implemented the multi-Agent communication[22]. On the whole, domestic research on ACL mainly focus on communication between Agents using KQML and FIPA ACL, lacking of theory and system research on ACL.

The existing SoftMan communication language is the message-based language. It has realized the basic communication between SoftMans, but with the increase of complexity of problem and introduced the Vague set theory's simulation of thinking process and fuzzy concept, behavior description and reasoning process based on knowledge model into the research of SoftMan, the anthropomorphic degree of SoftMan system has been improved, while the requirement of communication has reached to the knowledge level. Obviously, the communication language based on message doesn't perform perfectly on the support and description of SoftMan cooperation. So, the study of knowledge SoftMan communication language is imperative. In addition, the communication of human achieves the communication processing with the support of pragmatic layer, while the communication of SoftMan encodes and decodes on the syntax layer. We want to know whether the SoftMan could act like human beings, not only carrying messages in the communication, but also transmitting communication scenario and other auxiliary information, whether the receiver could decode and fuse the receiving multi-source heterogeneous information when it receives multi-source information including tone and expression. Therefore, we proposed the Context driven SoftMan Knowledge Communication (CSMKC).

DESIGN OF CONTEXT DRIVEN SOFTMAN COMMUNICATION FRAMEWORK

Building the framework of CSMKC is intended to build a buffer layer between the SoftMan and the lower transmission layer, and shield the heterogeneity from transmission layer. The framework of CSMKC uses hierarchical structure, with the uppermost Scene Layer and Knowledge Layer, which are responsible for describing and processing the scene and content respectively. From down to up, the framework also contains Communication Layer, Message Layer and Content Layer, and the structure of framework is shown is Figure 1.

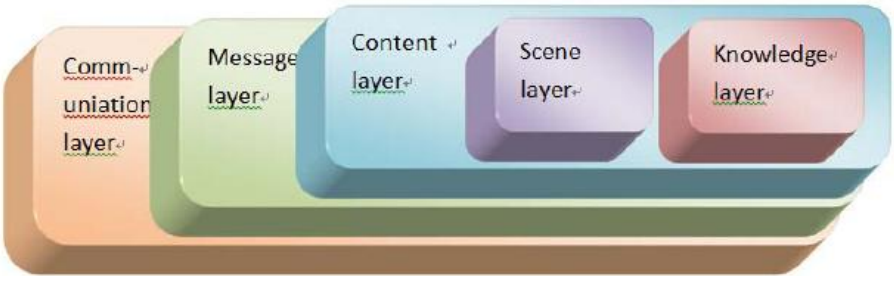


Figure 1: Framework of CSMKC

The communication layer

The communication layer encodes the message features on a lower layer through the communication parameters, such as the sender and recipient address, the communication id, et

The message layer

The message layer encodes some message that is sent to one application by another application, and describes the message style, the generated action expected and the desired results by defining the message parameters. This layer is the core, and it provides the optional feature parameters such as content language and scene content, which can provide support for the scene and knowledge communication.

The content layer

The content layer is described by the language defined by the message layer, and maintains the content of real communication.

The knowledge layer

The knowledge layer mostly does CRUD on the knowledge database. The traditional communication language of SoftMan only supports the message-level communication, which could only describe the purpose of communication plainly. In order to provide better support to the top-level SoftMan and to introduce knowledge into the SoftMan communication mechanism, the knowledge layer is dedicated to interactive in the level of knowledge.

The Scene Layer

This layer mainly describes the required operations of the current scenario sessions, which makes the communication language of SoftMan could support the scenario context.

It's clear that the main purpose of CSMKC is to support the knowledge communication, at the same time, it could simulate the process of session in the human social communication. Therefore, the overall framework of CSMKC must meet the following two requirements:

(1) It should increase supporting and presentation capabilities of knowledge in the syntax level, which could realize the support of collaboration and fuzzy reasoning to the top-level SoftMan.

(2) It should also add the content of maintenance and exchange to the scenario session, which could better simulate the the process of human communication.

DESIGN OF MESSAGE LAYER

As an intermediate layer of SoftMan communication system, the message layer can provide message-level support for the content layer, knowledge layer and the scene layer in the reliable communication platform which has been established by the communication layer. The good ability of description and functional expression of message layer can simplify the difficulties of code resolving of the upper layer to a great extent, and play a decisive role on the ultimate realization of the upper level. This section will focus on the design and description of the message instructions and its semantics.

Based on the function and effective object, the communication instructions of the content driven SoftMan can be divided into four categories: basic communication instructions, mission communication instructions, knowledge-base communication instructions and scenario maintenance communication instructions.

The basic communication instruction

The basic communication instruction is used to guarantee the basic communication among SoftMans, such as inquiring, replying and rejection.

TABLE 1: The basic communication instruction

The definition of communication instruction	The meaning of communication instruction
GENERAL_ACT_AVILIABLE	To inquire if the status is available
GENERAL_ACT_REPLY	To reply the information of SoftMan
GENERAL_ACT_DENY	To reject the SoftMan's inquiry ofcommunication
GENERAL_ACT_EOS	To inform the communication is over
GENERAL_ACT_DISCARD	To discard the communication
GENERAL_ACT_WAIT	To inform the other one waiting for a certain length of time
GENERAL_ACT_NOTICE	To acknowledge each other the current status of the SoftMan

The mission communication instruction

The objective of the mission communication instruction is to accomplish the management function and service function of the entire SoftMan community or a single SoftMan.

TABLE 2: The mission communication instruction

The definition of communication instruction	The meaning of communication instruction
TASK_ACT_ASK_NODE	To ask one node of the community
TASK_ACT_REPLY_NODE	To reply the information of one node in the community
TASK_ACT_ASK_SM	To ask Function SoftMan
TASK_ACT_REPLY_SM	To reply the information of the function SoftMan
TASK_ACT_ASK_ALL_SM	The ask all the SoftMan in the community
TASK_ACT_REPLY_ALL_SM	To reply all the SoftMan in the community
TASK_ACT_REG_NODE	To register a node in the community of SoftMan
TASK_ACT_UNREG_NODE	To cancel a node in the community of SoftMan
TASK_ACT_ASK_API	To register a Function SoftMans
TASK_ACT_REPLY_API	To reply the status of registered Function SoftMan
TASK_ACT_ASK_ALL_API	To cancel a Function SoftMan
TASK_ACT_REPLY_ALL_API	To reply the status of cancelled function SoftMan

The knowledge-base communication instruction

The knowledge-base communication instruction is mainly used to do the operations of CRUD on the knowledge base of SoftMan, and some of which will also do modify queries on the scenario information.

TABLE 3: The knowledge-base communication instruction

The definition of communication instruction	The meaning of communication instruction
KBASE_ACT_EXIST	To search if the content exists in the knowledge-base
KBASE_ACT_INSERT	To insert content into the receiver's knowledge-base
KBASE_ACT_UNINSERT	To cancel the inserting action
KBASE_ACT_DELETE	To delete content of the receiver's knowledge-base
KBASE_ACT_UNDELETE	To cancel the deleting action to the knowledge-base
KBASE_ACT_MODIFY	To modify a piece of content in the knowledge-base
KBASE_ACT_SEARCH_ATRR	To search content by a certain attribute in the knowledge-base
KBASE_ACT_SEARCH	To search content in the receiver's knowledge-base
KBASE_ACT_TOPN	To ask node satisfied the TOP N in the knowledge-base

The scenario maintenance communication instruction

This kind of communication instruction is specially designed to maintain the scenario information.

TABLE 4: The scenario maintenance communication instruction

The definition of communication instruction	The meaning of communication instruction
CONTEXT_ACT_EXIST	To search if the content exists in the scenario-base
CONTEXT_ACT_INSERT	To insert content into the receiver's scenario-base
CONTEXT_ACT_UNINSERT	To cancel the inserting action
CONTEXT_ACT_DELETE	To delete content of the receiver's scenario-base
CONTEXT_ACT_UNDELETE	To cancel the deleting action
CONTEXT_ACT_MODIFY	To modify a piece of content in the scenario-base
CONTEXT_ACT_SEARCH_ATRR	To search a certain attribute in the scenario-base
CONTEXT_ACT_SEARCH	To search content in the receiver's scenario -base
CONTEXT_ACT_TOPN	To ask node satisfied the TOP N in the scenario -base
CONTEXT _ACT_CREATE	To inform the partners of the creation of the communication
CONTEXT _ACT_EXIST	To inform the partners to exit the communication
CONTEXT _ACT_EXIST_WAIT	To inform the SoftMan exiting the communication to wait
CONTEXT _ACT_JOIN	SoftMan ask to join the communication
CONTEXT _ACT_JOIN_DENY	To refuse the SoftMan to join the communication
CONTEXT _ACT_JOIN_ACCEPT	To accept the SoftMan to join the communication
CONTEXT _ACT_SYNC	To synchronous the information of communication
CONTEXT_ACT_SUBSCRIBE	To subscribe to the scenario of partners according to the concern

DESIGN OF KNOWLEDGE LAYER

The information expressed in the knowledge layer is refined, processed and structured content, which is directly controlled by the instructions operating the knowledge. This section will focus on the expression of communication language in SoftMan.

Definition 1: The knowledge and information in the system of SoftMan can be defined as a triple: <Data, Type, Relation>

(1) Data is the data part of the knowledge, and it consists of the type of data, the content of data and the length of data, which can be defined as Data ::=<DataType, Content, Length>.

(2) Type is the type part of the knowledge, different from the type of data, it represents the logical type of the node. Depending on their difference of the content and function of the storage, the knowledge nodes are divided into resource knowledge node and attribute knowledge node.

(3) Through the membership degree and the false membership degree, Relation described the one node's fuzzy association with the other knowledge nodes, which is difined as Realtion::= U<Other, tValue, fValue>, and tValue is the membership degree, fValue is the false membership degree.

We apply the knowledge defined above in the function SoftMan in the SoftMan system, and the knowledge model is shown in Figure 2.

In the whole SoftMan system, the Function SoftMan is not only the owner and defender of all the functions, but also is the performer of the tasks. Throughout the operation, the communication among the Function SoftMan is an important part of the communication in the SoftMan system. We divide the knowledge maintained by the Function SoftMan into two parts, knowledge base and behavior. The knowledge base includes: Task, the tasks of Function SoftMan, Context, the information of the context, Message, the message of the SoftMan. And the behavior includes migration activities Migrate, life-cycle maintenance activities LifeCycle, resource maintenance activities ResouceCtrl, communication activities Communication, service activities Service and extended service activities Extend. The solid node represents the node reserved for the system, the name for node is a reserved word, and the hollow box represents a node defined by users. The solid line represents the determined association of which the membership degree is 1 and the false membership degree is 0, while the dashed line represents the non-

determined association of which the membership degree is less than 1 and the false membership degree is bigger than 0.

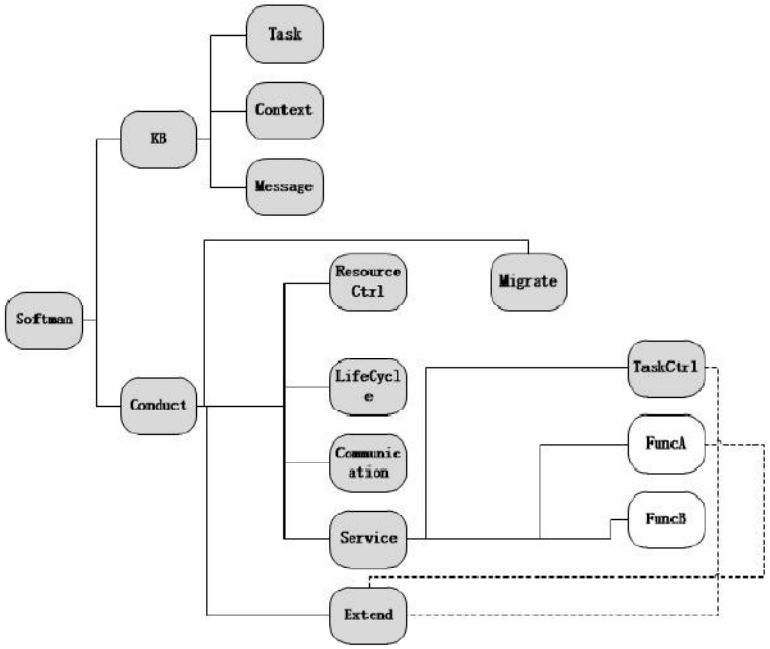


Figure 2: The knowledge model of function SoftMan

DESIGN OF SCENE LAYER

The scene layer is responsible for maintaining the context of the current seesion, and the content is related to the scene library, which completes the semantic operation about the scene with instructions.

The model of session context

The session context is the data recording the information of the communication partners or the communication itself in the process of communication among SoftMans, and each communication node maintains a scene library, at the same time, all the parties involved in each session will maintain their own version of the scene in themselves' scene libraries, and bind with dialogid, which is convenient for the query and parametric expression of the subsequent communication.

The model of session context mainly includes node resource information, historical knowledge information and the session metadata information, as shown in Figure 3.

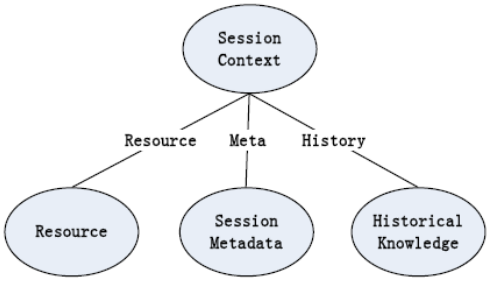


Figure 3: The model of session context

(a) The node resource information

The context information is the SoftMan's maintenance record of the resource information with which the SoftMan communicates. In the beginning, the session setup initialization and subscribe based

on the focus of the current SoftMan in the following communication. The opposite SoftMan will store the subscribed information in the history scene, and in each communication, the SoftMan access the changes of the subscribed information and transmit. Throughout the communication, the SoftMan keeps on modifying and maintaining its resource information.

(b) The session history information

The content of each communication will become common knowledge of the two sides of communication and be recorded, a common session history information such as scenario subscription information and parameter definition information. The scenario subscription information support the SoftMan to subscribe the scenario information in which itself is interested in, and the modification of the subscribed scene will be transmitted to the destination SoftMan in the next communication, so as to update and synchronize the scene information. The parameter definition information supports the SoftMan to define parameters mutually recognized by multi parties in the process of operation, which can reduce the cost of communication and improve the effect of unit information.

(c) The session metadata information

The session metadata information is used to describe the current session information, and to record the SoftMan participating in the session, the start time of the session, the expiration time of the session and the other basic information. The data among the session metadata may be updated constantly rewritten during the operation, such as the communication counter and the expiration time of the session.

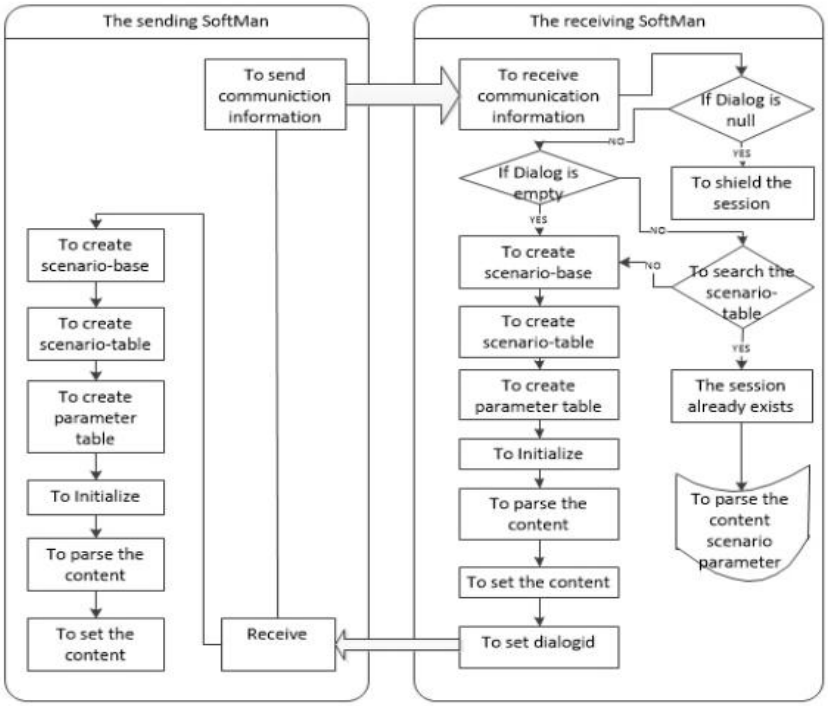
The maintenance of session scenario

The session context is the data recording the information of the communication partners or the communication itself in the process of communication among SoftMan, and each communication node maintains a scene library, and bind with dialogid, which is convenient for the query and parametric expression of the subsequent communication. The session context maintains through dialogid, and is uniquely explained by dialogid.

(a) The establishment of session

When the communication activity happens among SoftMans, the recipient will check the dialogid information through the dialog field of the communication. If the dialog information exists and can be found in the dialog table, it means the dialog of the current communication has been established. Otherwise, we need to establish a new dialog and insert a new table entry into the dialog table, the key of which is the current dialog, and value is the session information, the data structure is as described above, and wherein the metadata has been initialized, and set expiration time. Then the recipient needs to send dialogid and the current session information to the sender to acknowledge that the session has been established. According to the session information received, the SoftMan will create its own version of session, and add the entry in its dialog table, and the session information is also recorded in the dailogid table. Then in the communication, the SoftMan, two sides of the communication, will add this dialogid information to the dialog field of the message and identify which session the communication belongs to through the dialog id. If it belongs to the current session, the communication can use all the recorded node resource information and history knowledge information of the current session, as shown in Figure 4.

For some simple communication process, it only requires once or a few times of communication, which means the cost of establishment and maintenance of sessions is too high, so the benefits of the sessions are completely offset. In this case, the initiator of the communication shall inform the recipient of the session mask operation, the initiator could add DIALOG_ID_NULL in the dialog to describe the demand, then the receiver will skip the process of the session establishment.



Figue 4: The establishing process of session

(b) The close of the session

When one complete all the information, and there will no other information to be sent for a long time, then the session should be closed. First of all, the SoftMan checks the dialogid table, and sends close demand to the SofMan included in the current session, and wait to close. Once receiving the demand of session close, the other SoftMan will check the current message queue, and check if there is any message which is needed to send to the SoftMan requires to close or not, if yes, the SoftMan reply with a close waiting message, and estimate a shot waiting time, otherwise it will modify the session information and query the participants of the session, if no other participant, quit the session. When the waiting time expired, the side of requesting close will destroy the current session and exit. The state transmission of the session close is shown in Figure 5.

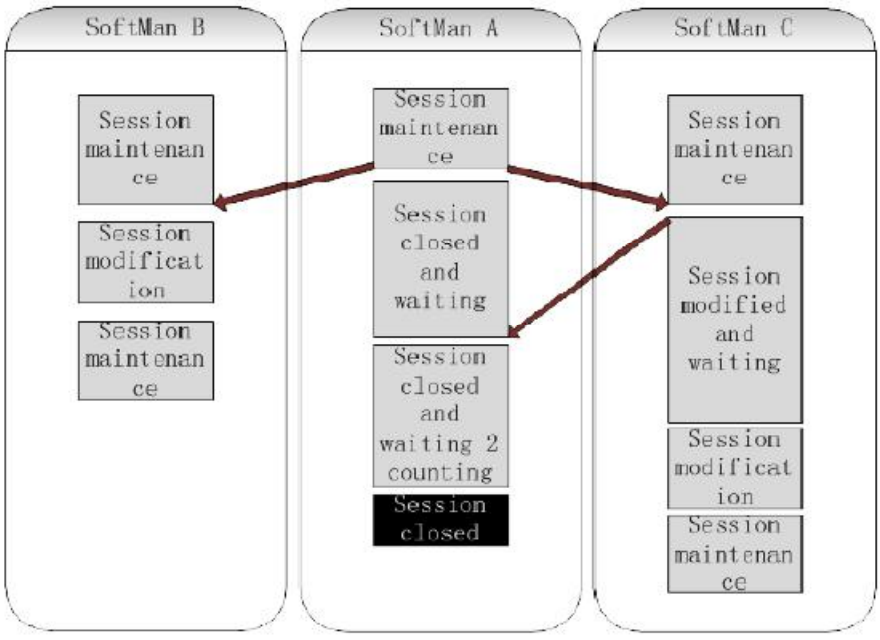


Figure 5: The status transition diagram of session closing

(c) Session expired

The SoftMan participating the session may die out or migrate in the near further, or there is no other communication requirement for the current communication participants for a long time, or the SoftMan in the communication may end the session without sending session close demand due to some cases. In these situations, the abandoned sessions will occupy the valuable storage resources of SoftMan for a long time. In this paper, we set expiring time for the sessions to avoid these situations, set a timer for the session, when the timer expires, the session will be set to be expired, and notify the participant. Each time the communication belongs to one session happens, the time will be reset.

(d) Parameterized communication

The historical knowledge information is stored in the form of key-value, and provide efficient interface of querying and modifying. All the SoftMan included in the session will maintain an information table of the current history information. When there is new communication requirements, the sender SoftMan will lookup the arguments supported by checking the current information table, and use the defined pubic knowledge to send the argument to replace the actual information. After receiving the message, the receiver SoftMan will analyze the received message, and translates the parameterized message to real information to avoid repeatedly sending messages. Establishing a logical communication link between the communication parties will improve the efficiency of communication.

The process of maintenance and synchronization for the historical knowledge information: we assume that SoftMan A and B are in the same session, and A requests B to calculate a binary add, then A or B may record the result as the history information. Assuming A records the result and names the variable as service+stamp, now, in order to make the variable become public knowledge of A and B, A sends new defined parameters to all the participants of the session through a response command or the command of CONTEXT_ACT_INSERT described by the context. At the same time, A needs to maintain the mapping from the current result to the parameter, which is convenient to be reused. When the SoftMan request for the functions of the other SoftMan joined in the session, it will check the mapping and the history scenes to try to make the communication partly parameterized, then it needs to query the scene library.

(e) Add a new node

The scene layer support new nodes to join the current session. When a new node joins an existing session, firstly, the SoftMan modifies the meta-information of the current scenario, and adds the information of the participants, then gives the dialogid to the new joined node, and deliveries a copy of context information to it. The command of CONTEXT _ACT_SYNC instructs the current session scenario in the field of newly added SoftMan, and adds it to scenario library, inserts to dialog table.

IMPLEMENTATION OF CONTEXT DRIVEN SOFTMAN COMMUNICATION

This section introduce the key point of implementation of Context driven SoftMan Communication from the four aspects, the implementation of message layer, the algorithm implementation of instruction parsing, the implementation of knowledge layer and the implementation of scene layer.

The implementation of message layer

In order to completely separate the content and the structure, we use XML to describe the message of CSMKCL, so as to make it easy for human cognition, while parsing XML for program is not difficult. In addition, XML has good interoperability and expansibility, which provides a good support for the communication layer of SoftMan. It’s described as follows:

In the label of scl_msg, it is the definition of the communication message of SoftMan, and it typically includes five parameters, as described below:

```
<?xml version="1.0" encoding="utf-8"?>
<smcl-root>
<scl_msg>
<sender>
<type></type>
<node></node>
<session></session>
<id></id>
</sender>
<reveiver>
<type></type>
<node></node>
<session></session>
<id></id>
</reveiver>
<dialog></ dialog >
<primitive></primitive>
<content></content>
<context></context>
</scl_msg>
</smcl-root>
```

(a) Sender, receiver

These two parameters illustrate the actual sender and recipient of primitive, different from the port address of the communication layer, the sender and recipient here are the description of data structure of senior layer, both belonging to one SoftMan, whose information is described by the parameter of type, node, session and id. Type described the type of the SoftMan, session is the community identity among SoftMans, node means which node the SoftMan belongs to, and id is the unique identity of the subordinate community of SoftMan. Node with id and uniquely identify a SoftMan.

(2) Dialog

The field of the global dialogid of the current session may have three values: normally the globally unique dialogid, the value of DIALOG_ID_EMPTY means that the current communication doesn't belong to any exisiting fields, and the value of DIALOG_ID_NULL means that the communication shields the session.

(3) Primitive

The communication command of SoftMan describes the behavior and type of SoftMan's communication.

(4) Content

This parameter specifies the object of the command directly have effect on. For example, if the command is ASK-NODE, the content describes the address of the community.

(5) Context

It describes the scenario and content. One piece of command could only have effect on content, and also only on scenario, or both on content and scenario.

Implementation of knowledge layer

The knowledge layer will eventually loads the knowledge model maintained by SoftMan into the process of communication in the way of data stream. To achieve this process, it is required to manipulate the encoding process of the knowledge model. The following introduces processing mechanism from the views of data and encoding.

(a) Data representation of knowledge layer

The knowledge model of SoftMan mainly includes the data structures as following:

```
enum SMKMType{smkm_no_type, smkm_int_type, smkm_float_type, smkm_str_type,
smkm_struct_type, smkm_list_type};
enum SMKMLogicType{smkm_logic_res_type, smkm_logic_attr_type};
typedef enum SMKType smkm_type;
typedef enum SMKMLogicType smkm_logic_type;
struct smkmTerm{
smkm_type smkmType;
smkm_logic_type logicType;
double wMs;
double wUnMs;
void *p;
size_t len;
};
struct smkmStructType{
struct smmTerm **args;
};
```

(A) SMKLogicType

From the design of the knowledge layer, the node of knowledge model of SoftMan includes two types, resource and attribute. The nodes belonging to the type of resource or attribute is called logic type of SoftMan, and is represented by enumeration of SMKLogicType. smkm_str_type represents a type of resource, and smkm_logic_attr_type represents a type of attribute.

(B) SMKMType

In addition to the logic type of node previously mentioned, the data represented by the node still requires data type, which is represented by SMKMType. It includes 6 values, smkm_no_type means no data, smkm_int_type, smkm_float_type, smkm_str_type represent int, float and char data relatively, smkm_struct_type indicates the type of structure, and smkm_list_type represents the type of link.

(C) smkmTerm

The structure of smkmTerm represents the node of SoftMan's knowledge model, whose members is smkmType, representing the data type, logicType, representing the logical type, len, representing the length of data and the pointer p, recording the position of knowledge data. This paper uses the wMs and wUnMs to express the fuzzy relationship with their father nodes, corresponding to the tvalue and fvalue previously mentioned in the knowledge model.

(D) smkmStructType

It is used to maintain the knowledge model of SoftMan.

The data structure above provides a flexible assembly method for the knowledge expression of SoftMan, and it has good scalability and easy to modify and maintain.

(b) Mechanism of encoding process

The knowledge data finally performs in the format of string in the communication. In order to deliver the complicated data model to the other side communication SoftMan in the format of string without errors, in addition to the self-description of carrying data with the good semi-structured character of XML, we still need a good encoding mechanism, as shown in Figure 6.

Implementation of scenario library

The implementation of scenario library includes three main data structure jointly to maintain the building, maintaining, querying and exiting of the scenario library.

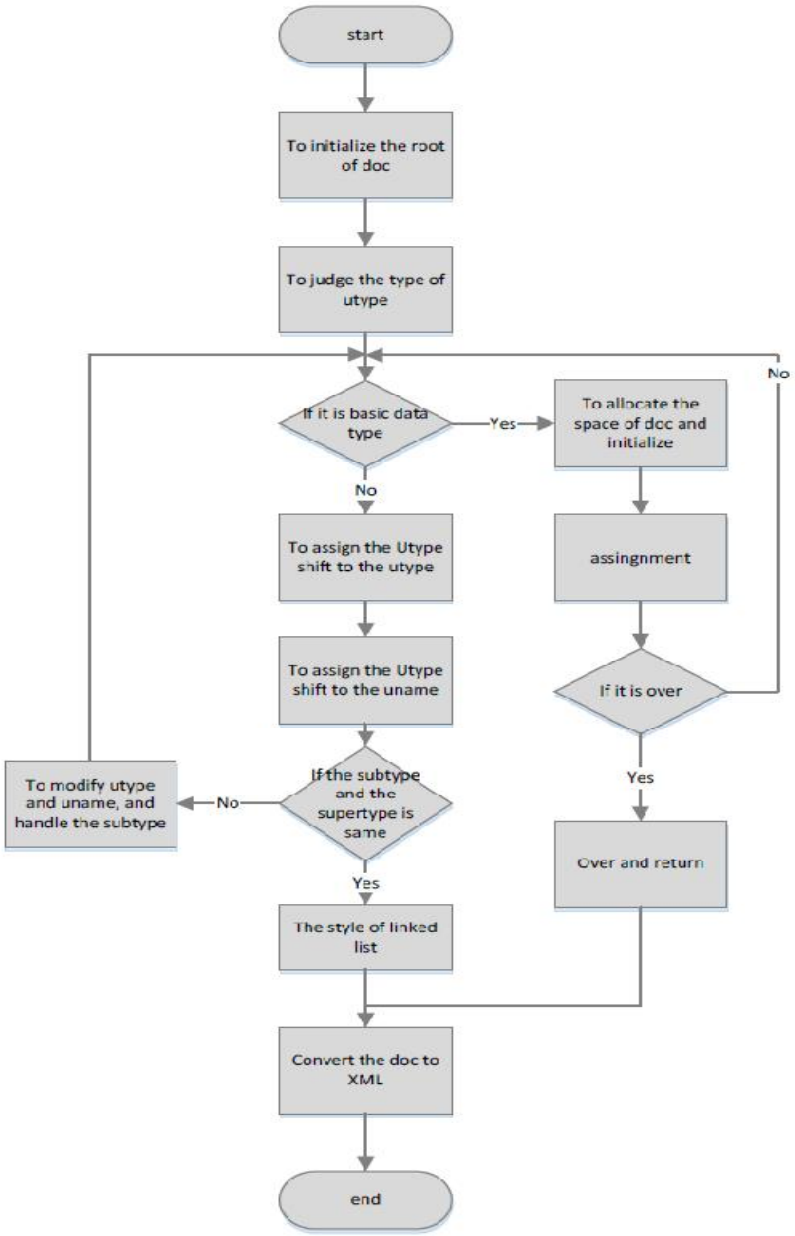


Figure 6: Mechanism of encoding process

(a) The scenario library

Maintaining resource scenario, history scenario and scenario metadata through different paths, one SoftMan can maintain multiple libraries, while one library can have many SoftMan participants.

(b) The scenario table

The scenario table is designed to record the corresponding relationship among individual SoftMan, scenario id and scene library, in which scenario id is the primary key.

(c) The parameter table

The parameter table is design for scenario subscription and maintenance of history scenarios, which records the pubic knowledge of the scenario parties. The parameters existing in the parameter table will not repeat over a communication link.

The implementation of scenarios is supported by the operation module of scenario, ctxlib, which defined all the functions of operations on scenario, and most of the functions are designed around the three core data structures mentioned above, scenario library, scenario table and parameter table. The main functions of this operation module is described in IDL as follows:

```
module ctxlib{
typedef sequence<Context> context_lib;
typedef sequence<ContextLib> ctx_table;
typedef sequence<Parameter> para_table;
interface Behavior{
void parse_context(string context, context_lib * lib, para_table *ptable);
int query_ctx_table_byid(int id, ctx_table* table);
int query_ctx_table_bysm(int sm_type, ctx_table* table);
int query_para_table_bykey(string key, char *path, para_table * ptable);
int create_para_table(para_table ** ptable);
int para_table_append(para_table *ptable,char* name, char* path);
int query_ctxlib_bypath(string path, string value, context_lib * ctxlib);
int ctx_cteate(int id, int sm_type, ctx_table** table);
int ctx_lib_initial(context_lib **newlib, int sm_type);
int ctx_lib_append(context_lib *newlib, char* path, char* value);
int para_replace(smcl_msg *msg, context_lib *ctxlib, para_table *para_t);
int para_define_int(string name, int value, ctx_table* ctxt, para_table * t);
int para_define_charp(string name, string value, para_table * t);
int ctx_aware(smcl_msg *msg, ctx_table* table, para_table *para_t);
};
};
```

The operation of parse_context is used to resolve all the scenarios received, and decides the following operation according to the different types of scenarios. query_ctx_table_byid and query_ctx_table_bysm are two different methods of query onf scenario table. query_para_table_bykey queries on the parameter table by the value of key. create_para_table is used to create parameter table. para_table_append is used to add parameters into the existed parameter table. ctx_cteate creates and initializes all the data structures related to the scenario, which accepts a specified formal parameter id, if not exists, it was -1. ctx_lib_initial and ctx_lib_append are used to initialize and add items on the scenario library. The operation of para_replace is the core method of parameter communication, and resolves the received message according to the scenario information of the current SoftMan. ctx_aware is the first step of operation for all the content SoftMan when they receive messages, which is called content awareness, and can handle according to different situations.

The context-aware algorithm is expressed as follows:

```
ctx_aware()
input：smcl_msg *msg, ctx_table* table, para_table *para_t
output：end state
begin
dialogid = msg->dialogid;
if（dialogid==DIALOG_ID_NULL）
return OK;
PrimitiveSync（dialogid, information）
if（dialogid==DIALOG_ID_EMPTY）
cur_ctx_table_index = ctx_create（-1, sm_type,&table）;
create_para_table（&para_t）;
add_all_ctxlib_into_context（）;
else
cur_ctx_table_index = query_ctx_table_byid（dialogid, table）;
if（cur_ctx_table_index==NO_EXIST）
cur_ctx_table_index = ctx_create（dialogid, sm_type,&table）;
create_para_table（&para_t）;
parse_context（msg->context, table->ctx_lib[cur_ctx_table_index],
para_t）;
else
return ERROR;
end if
end if
para_replace（msg, table->ctx_lib[cur_ctx_table_index], para_t）;
return OK;
end
```

COMPARATIVE EXPERIMENTS

The traditional Softman communication language is based on message, and the communication mode is byte stream socket communication based on the TCP protocol. While upon the context layer of that, the Context driven SoftMan Knowledge Communication Language adds the knowledge layer which is responsible for negotiation and the scenario layer which is responsible for maintaining the scenario context. On the one hand, the maintenance of knowledge layer and scenario layer will increase the cost of communication, on the other hand, the establishment of scenario layer will provide additional information to improve the density of traffic and reduce the cost of communication. Next we will compare the communication cost of the two kinds of communication mechanisms.

In the experiments, we use two PCs to run the tests, whose processor adopts Inter Core Duo, CPU Clock Speed is 2.1GHz, and memory is 2G DDR2. We simulate the working process of SoftMan in real environment by the communication of the two PCs. The operation system is unified in the SoftMan system, using Linux operating system. The two computers are connected to the Ethernet, whose delay is about 100us.

Then, in both low scenario-dependent communication and high scenario-dependent communication cases, we use CSMKCL and traditional communication language to make comparative experiments of average communication consumption in per unit time We assume that the network is stable and reliable in the experiments. The cost of communication is approximately equivalent to the amount of communication data, and the amount of communication data is convenient for procedure to determine. In the LAN environment, the communication consumption of the comparative experiments is shown in Figure 7 and Figure 8.

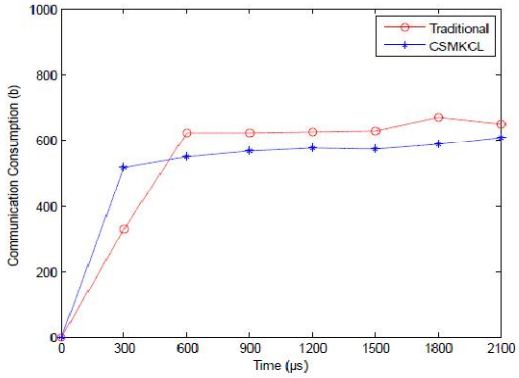


Figure 7: The comparative experiments of CSMKCL and traditional communication language in low scenario-dependent communication case

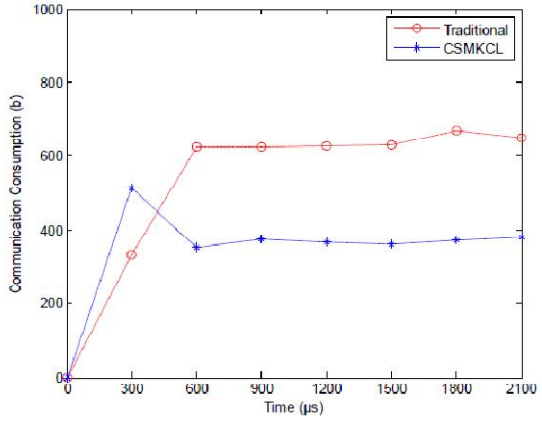


Figure 8: The comparative experiments of CSMKCL and traditional communication language in high scenario-dependent communication case

Through Figure 7 and Figure 8, we can see the consumption of CSMKCL is significantly higher than that of the traditional communication language as expected, because at this stage, it spends a lot of consumption on the establishment and synchronization of the scenario context. Furthermore, in the low scenario-dependent subsequent communication, the gap between the two communication mechanisms is not large, but when the subsequent is highly scenario-dependent, the Context driven SoftMan Knowledge Communication Language is significantly better than the traditional one.

In order to reflect the gap between the two mechanisms, we take the extremely communication requirements to compare, while most of the actual communication requirements are between the two extremes. In the SoftMan system, most basic communication scenarios such as registration and cancellation of SoftMan are low scenario-dependent communication, while the completion of a function is much closer to the high scenario-dependent one. And the more time the communication takes, the advantages of the Context driven SoftMan Knowledge Communication Language could be better reflected.

CONCLUSION

Unlike distributed systems and Multi-Agent systems, individuals in SoftMan system have anthropomorphic emotion, so the communication between them has distinguished specific needs against communication between traditional Agents. Traditional SoftMan communication language has already satisfied the basic communication needs on the basis of messages. In this paper, in order to provide effective support for the fuzzy reasoning between upper layer SoftMans and improve the success rate of communication, the Content Layer has been subdivided into Knowledge Layer focusing on the operation of the base of knowledge. In order to maintain the context scenario information in the process of communication and improve the quality and density of communication, we introduce the Scene Layer. And we discuss the design and implementation of the framework in detail from the view of Message Layer, Knowledge Layer and Scene Layer.

REFERENCES

[1] Zeng Guang Ping, Tu Xu Yan, Wang Hong Bo; SoftMan Research & Application. BeiJing: Science Press **(2007)**.

[2] Wang Hong Bo, Zeng Guang Ping, Tu Xu Yan; Research on SoftMan systems and its application, CAAI Transactions on Intelligent Systems, **1(1)**, 24-28 **(2006)**.

[3] Tao Xueli, Zheng Yanbin; A level task allocation method for multiple-agents [J], Journal of Computational Information Systems, **9(2)**, 813-820 January 15 **(2013)**.

[4] He Yong, Fei Shengwei, Hu Yang; Study on the multi-task intelligent synchronous control system[J], Journal of Computational Information Systems, **9(12)**, 4629-4638 June 15 **(2013)**.

[5] Wang Lu, Wang Zhiliang; Collection path ant colony optimization for multi-agent static task allocation [J], Journal of Information and Computational Science, **9(18)**, 5689-5696 December 19 **(2012)**.

[6] Lei Ming , Zhou Shaolei, Zhang Yi, Zhang Wenguang; Consensus based non-close formation control of multi-agent with communication noises [J], Journal of Information and Computational Science, **9(18)**, 5595-5602 December 19 **(2012)**.

[7] Wang Hong Bo, Zeng Guang Ping, Wang Zong Jie, Tu Xu Yan; SoftMan Group Intelligent Autonomous Coordination Model and Its Application, Acta Automatica Sinica, **33(9)**, 924-929 **(2007)**.

[8] F.Raji, B.T.Ladani; Anonymity and security for autonomous mobile agents [J], IET Information Security, **4(4)**, 397-410 **(2010)**.

[9] Wang Lu, Wang Zhi Liang, Yang Yi; Framework of multi-agent system and consultation mechanism[J], Application Research of Computers, **29(3)**, 852-855 **(2012)**.

[10] Ma Zhonggui, Zhou Xianwei, Zeng Guangping, Tu Xuyan; Research on common coordination model for SoftMan society[C], Proceedings - Third International Conference on Natural Computation, ICNC 2007, **5**, 356-360 **(2007)**.

[11] Zhang Fu-sheng, Wang Hong-bo, Zeng Guang-ping, Yang Yang, Tu Xuyan; Multi-softman coordination model and its application in metallurgical construction project management[C], Proceedings - 2009 International Conference on New Trends in Information and Service Science, NISS 2009, 963-968 **(2009)**.

[12] Zhang Qingchuan, Zeng Guangping, Chen Yunbo, Liu Jinfu; Research and implementation of SoftMan's communication language[C], Proceedings - 2010 6th International Conference on Natural Computation, ICNC 2010, **6**, 2816-2819 **(2010)**.

[13] Ge Yuanzheng, Liu Liang, Qiu Xiaogang; A framework of multilayer social networks for communication behavior with agent-based modeling, Simulation-Transactions of the Society for Modeling and Simulation International, **89(7)**, 810-828, Jul **(2013)**.

[14] K.Chopra Amit, Artikis Alexander, Bentahar Jamal; Research Directions in Agent Communication, ACM Transactions on Intelligent Systems and Technology, **4(2)**, Mar **(2013)**.

[15] Wen Guanghui, Duan Zhisheng, Yu Wenwu; Consensus in multi-agent systems with communication constraints, International Journal of Robust and Nonlinear Control, **22(2)**, 170-182 Jan 25 **(2012)**.

[16] Hu Cuiyun, Mao Xinjun, Li Mengjun; Organization-based agent-oriented programming: model, mechanisms, and language, Frontiers of Computer Science, **8(1)**, 33-51 Feb **(2014)**.

[17] Ciobanu Gabriel, Juravle Calin; Flexible software architecture and language for mobile agents, Concurrency and Computation-Practice & Experience, **24(6)**, 559-571 Apr 25 **(2012)**.

[18] M.T.Kone, T.Nakajima; An architecture for a QoS-based mobile agent system[C]//Real-Time Computing Systems and Applications, 1998. Proceedings, Fifth International Conference on, IEEE, 145-148 **(1998)**.

[19] Magid Nikraz, Giovanni Caire, Parisa A.Bahri; A Methodology for the Analysis and Design of Multi-agent Systems using JADE, International Journal of ComPuter Systems Science and Engineering, May **(2006)**.

[20] M.P.Singh; Agent communication languages: rethinking the PrinciPles [J], ComPuter, **31(12)**, 40-47 **(1998)**.

[21] Zhang ShiJie, Jing Shuang; The application of KQML in the Multi-Agent CAPP System for Ship Hull[J], Computer Integrated Manufacturing Systems, **7(4) (2001)**.

[22] Huang Jian Qing, Xu Lingce; A CORBA-based Agent Communication Mechanism[J], Computer Engineering and Application, **38(5)**, 164-165 **(2002)**.