Knowledge Interchange in Task-Oriented Architecture: For Space Robot Application

Cen Yu1,2, a, WeiJia Zhou2,b

1University of Chinese Academy of Sciences, Beijing 100049, CHN
2Shenyang Institute of Automation Chinese Academy of Sciences, State key Laboratory of Robotics, Nanta Street 114#, Shenyang, Liaoning Province, China, 110016E

a yucen@sia.cn, b zwj@sia.cn

Keywords: Knowledge Interchange, Multi-Robot System, Task-Oriented Architecture, Space Robot, KIF Language, OKBC.

Abstract. Behaviors of multi-robot system based on task-oriented architecture are intuitional according to the flow of task processing that is obvious to plan and monitor. This paper tables a novel task-oriented architecture for space robot application, which consists of task description, task completion analysis, task compromise. For this architecture, author designed a knowledge interchange mechanism base on KIF (Knowledge Interchange Format) and OKBC (Open Knowledge Base Connectivity). Using this knowledge interchange mechanism, knowledge bases designed by different languages comprehend information transmitted form each other.

Introduction

Now the space robot application researches have focused on the in-space operation robots and planetary surface exploration robots. In-space operation robots assist shuttle docking and scientific experiment in the satellites and space stations, such as ERA (European Robotic Arm), Canadarm, JEMRMS (Japan Experiment Module Remote Manipulator System). Planetary surface exploration robots rove and sample on the planetary surface, such as Sojourner.

Space robotics technology assessment report (2002, by NASA) [1] classified the tasks of in-space operation into in-space assembly, in-space inspection, in-space maintenance, in-space human EVA (Extra-vehicular activity) assistance. And the tasks of planetary surface exploration were classified into surface mobility, surface instrument deployment and sample manipulation, surface science perception planning and execution, surface human EVA assistance. Space robots must be multifunctional since the varied tasks and the high cost of launch.

Designing a complex robot to accomplish varied tasks is high cost and difficult to achieve. A number of simple robots can replace it with reasonable cost and lower difficulty. The redundant of quantity also will improve the robustness of the system. Multi-robot system has attracted more and more attention in the space robot researches. Several principle prototypes were developed, SuperBot [2], PloyBot [3].

From simple system composed of 2 or 3 robots to complex system composed of hundreds robots the communication and the coordination will be more difficult to carry out. It has been a hotspot on the research of robots system architecture [4] ~ [6] of how the multi-robot system to accomplish varied tasks in complex environment.

Architecture is the description of component modules, the relationship and interaction between modules. Researchers have presented effective multi-robot architectures with simulation and experiment [7]. For example, ALLIANCE (L.E.Paker)[8], KAMARA (Lueth & Laengle)[9], STEAM (Tambe)[10], GOPHER (Caloud)[11], Three Levels Architecture (Noreils)[12].

According to the behavior mode of system, these researches form two directions:

1. Multi-agent negotiation. Such as ALLIANCE, KAMARA, STEAM. System behavior is obscure for observers.
(2) Task-oriented. Such as GOPHER and Three Levels Architecture. Task description, task completion analysis and system fault is obvious. Task-oriented architecture meets the requirement of space robot application.

Robots, human and AI systems form a hierarchical organization to decompose, plan and feedback the given tasks. The knowledge interchanged in the organization consists of facts, rules and targets [13]. The knowledge interchange mechanism is the foundation of the architecture.

The remainder of this paper is organized as follows: section 2 presents the task-oriented architecture for space robot application. Section 3 describes the knowledge interchange mechanism based on KIF and OKBC. Section 4 illustrates how the mechanism works with an example.

**Task-Oriented Architecture**

Task-oriented architecture is designed according to the procedure of tasks. External demand is described by the form of abstract task. The set of abstract tasks could be divided into a number of levels. Abstract task is planned into lower level tasks until it can be implemented by robot. The task that can be autonomously implemented by robot is called specific task. Specific task is described with predefined functions and varied parameters. In contrast, the task need for the participation of operator is called remote-control task.

As shown in Fig. 1, abstract tasks are planned into lower level abstract tasks. Finally, we have specific tasks for robots and remote-control tasks for operators.

![Figure 1. Tasks in task-oriented architecture](image)

In task-oriented architecture, the layers to process abstract tasks are defined as task planning layers. Task planning layers consist of human and AI systems. The layers to process specific tasks and remote-control tasks are defined as robot control layer, which consist of robots. Operator executes remote-control tasks through the human operator interface.

Task planning layers deal with abstract tasks in 3 ways: task description, task completion analysis, task compromise. Robot control layer executes specific tasks and remote-control tasks, feeds back and evaluates the health status of robots. Global knowledge base makes effective interaction that both sides can understand each other's information.

The schematic diagram of task-oriented architecture is shown in Fig. 2.
In essence, Multi-robot system is a distributed system. Knowledge interchange is a prerequisite for the task-oriented architecture. Section 3 designed a knowledge interchange mechanism based on KIF and OKBC that realized the interaction between individual knowledge bases.

### Knowledge Interaction Mechanism

The knowledge about the world and tasks is represented by models in robot system [13]. Table 1 classifies models into detail and feature. Detail models are used in robot control layer for robot control. The knowledge delivered in global knowledge base is represented by feature models.

Table 1. Two kind of model

<table>
<thead>
<tr>
<th>Content</th>
<th>Detail Model</th>
<th>Feature Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>Completeness</td>
<td>Details of the structured description</td>
<td>System feature for reasoning</td>
</tr>
<tr>
<td>Example</td>
<td>Complete</td>
<td>Incomplete</td>
</tr>
<tr>
<td>Scenarios</td>
<td>Path optimization</td>
<td>Decision making and monitoring</td>
</tr>
<tr>
<td></td>
<td>Map</td>
<td>Can the robot pass this area? (Y/N)</td>
</tr>
</tbody>
</table>

Early work on enabling technology for knowledge sharing established that three components are needed to allow knowledge to be shared between two knowledge base [14]:

- A common set of definitions of terminology - conceptual model;
- A common language in which to express knowledge - formal language;
- A common protocol in which to communicate knowledge – communication protocol.

Ontology is the most typical conceptual model that is an explicit specification of a conceptualization (Gruber, 1993) [15]. The ontology provides vocabulary (or names) for referring to the terms in that subject area, and the logical statements that describe what the terms are, how they are related to each other, and how they can or cannot be related to each other. It also provides rules for combining terms and relations to define extensions to the vocabulary, as well as the problem semantics independent of reader and context. [16].
Definition 1: Domain space \(<D, W>\), where \(D\) is a domain and \(W\) is a set of maximal states of affairs of such domain (also called possible worlds). A conceptual relation \(\rho^n\) of arity \(n\) on \(<D, W>\) as a total function \(\rho^n : W \rightarrow 2^{D^n}\) from \(W\) into the set of all \(n\)-ary (ordinary) relations on \(D\). A conceptualization for \(D\) can be now defined as an ordered triple \(C = <D, W, \Re>\), where \(\Re\) is a set of conceptual relations on the domain space \(<D, W>\). [17]

Knowledge representation language refers things in the world, and the proposition about the world. According to the range of quantification, logical language can be classified into first-order predicate logic and higher-order predicate logic.

Definition 2: Logical language \(L\), with a vocabulary \(V\). An ontological commitment \(K\) for \(L\), \(K = <C, \Im>\), where \(C = <D, W, \Re>\) is a conceptualization and \(\Im : V \rightarrow D \cup \Re\) is a function assigning elements of \(D\) to constant symbols of \(V\), and elements of \(\Re\) to predicate symbols of \(V\).

Now several first-order predicate logic languages were developed to construct the knowledge base, such as Emycin, Prolog etc. It’s possible that the knowledge bases in a system are developed by different languages. Then we need a middle language to achieve the interchange between knowledge bases, just like KIF (Knowledge Interchange Format).

KIF is a kind of knowledge representation language that contains no command to access and operate knowledge bases. OKBC (Open Knowledge Base Connectivity) is a protocol developed by the Knowledge Systems Lab of Stanford University for knowledge base accessing. It provides common interface to access knowledge base. [19]

**Knowledge Interchange Model.** In the task planning layers, the human and AI systems are called planners. Every planner has a knowledge base. \(kb^1, kb^2\) and so on refer to the knowledge bases in layer \(n\). Knowledge interchange model defined the relations of knowledge bases, as shown in Fig. 3.

Knowledge is transformed into ontology described by middle language, and delivered through communication protocol. Specifically, the ontology of higher layer knowledge base is abstracted by lower layer knowledge bases. The interchange between layers needs abstract process. As shown in Fig. 4.
Knowledge interchange process will be discussed in more detail at later stage.

**Ontology described by KIF.** A class is a set of entities. Each of the entities in a class is said to be an instance of the class. A class can be an instance of a class. Entities that are not classes are referred to as individuals. The domain D of discourse consists of individuals and classes. Entities have properties that can take different values. The set of possible property values consist of possible worlds W.

The unary relation class is true if and only if its argument is a class and the unary relation individuals true if and only if its argument is an individual. The class membership relation (called instance-of) that holds between an instance and a class is a binary relation that maps entities to classes. The properties refer to relation $\rho^n$ between n entities ($n \geq 1$).

Frame is a primitive object that represents an entity in the domain of discourse. Formally, a frame corresponds to a KIF constant. A frame that represents a class is called a class frame, and a frame that represents an individual is called an individual frame.

A frame has associated with it a set of slots and each slot of a frame has associated with it a set of entities called slot values. Formally, a slot is a binary relation, and each value V of a slot S of a frame F represents the assertion that the relation S holds for the entity represented by F and the entity represented by V. (S F V)

A slot of a frame has associated with it a set of facets, and each facet of a slot of a frame has associated with it a set of entities called facet values. Formally, a facet is a ternary relation, and each value V of facet Fa of slot S of frame Fr represents the assertion that the relation Fa holds for the relation S, the entity represented by Fr, and the entity represented by V. (Fa S Fr V)

Frames and the rules about frames can be stored and delivered by script written in KIF language. As shown in Fig. 5.

Protocol based on OKBC. OKBC assumes a client-server architecture for application development. To access a knowledge base, an application loads the OKBC client library for the appropriate programming language. The client library defines methods for all of the OKBC operations with that language.

To access an OKBC server, a client application typically undertakes the following steps.

Firstly, the application must establish a connection to an OKBC server (using establish-connection).

Secondly, the application may find out the set of KRSs (Knowledge Retrieval System) that the server supports on that connection (using get-kb-types).

Thirdly, the application can get information about the KBs (Knowledge Base) of a given type that can be opened on the connection (using openable-kbs).

Finally, the application can either open a specific KB (using open-kb) or create a new one (using create-kb). The application may now query and manipulate the KB by executing OKBC operations. For example, it can look for specific frames (using get-frames-matching), find the root classes of the KB (using get-kb-roots, create new frames (using create-frame), and save changes (using save-kb).
Abstract procedure. Abstract is the mapping from a set of level n-1 entities to level n entity. In abstract, the set of level n-1 entity is components of the level n entity. Abstract function describes that mapping relation.

\[ Frame^n = f^n_{com}(Frame_1^{n-1}, Frame_2^{n-1}, \ldots, Frame_k^{n-1}) \]

\[ Frame^n \in kb^n, \ Frame_1^{n-1}, Frame_2^{n-1}, \ldots, Frame_k^{n-1} \in kb^{n-1} \]

\( f^n_{com} \) is the abstract function mapping a set of level n-1 entities to a level n entity. \( kb^n \) is a level n knowledge base, and \( Frame^n \) is the frame of entity in \( kb^n \). As shown in Fig. 6.

Figure 6. The abstract between levels

Abstract function will have different connotation according to the context. If the reasoning mechanism quantifies the abstract function, then we have to deal with second-order predicate logic that is very difficult to design. In this paper, we use the first-order predicate logic and the interchanges between layers take the steps as follows:

Level 1: refresh frames according to the feedback from robots as the Fig. 7;

Figure 7. Level 1 Abstract

Level n: refresh the abstract functions according to the level n-1 frames if necessary. Then level n frames will be refreshed by using the abstract functions. As shown in Fig. 8.

Figure 8. Level n Abstract (n>1)

We can use knowledge of related field to develop first-order predicate logic machine for the reasoning of abstract functions. For some simple application, finite-state machine is enough.
Application Case

A representative application of manipulator in the space station is chosen, just like the JEMRMS (Japan Experiment Module Remote Manipulator System). Robot system consists of a railway, a heavy-arm and a light-arm. Light-arm is mounted on the end of heavy-arm, and the heavy-arm is mounted on the railway to extend its operating range. Heavy-arm can operate heavy payloads and the light-arm can operate light payloads precisely.

The organization of system is divided into 3 levels: 2 task planning layers and 1 robot control layer. The entities in each layer and the abstract relationships are described in Fig. 9.

Based on the knowledge interchange mechanism, task-oriented architecture can achieve 3 procedures about given tasks.

(1) Task description
Task received by robot system will be decomposed to the tasks for railway, heavy-arm and light-arm. If the heavy-arm or the light-arm cannot perform task automatically, then operator will interact with them through human operator interface and the task will be described in the form of remote-control task. Finally, the joints and end-effectors will receive the command just like target position, target speed, target torque.

(2) Task completion analysis
Task planning layers refresh the frames of the knowledge bases according to the abstract procedure mentioned above. Then we can analyze the progress of tasks.

(3) Describe task compromise
As the same way of task completion analysis, frames of the knowledge bases will be refreshed for planners to make a compromise as necessary.

When the configuration of heavy-arm of light-arm need to be changed or some faults are detected, abstract functions will be refreshed to maintain the knowledge interchange mechanism. There are several effective methods for automatic configuration recognition, kinematics and dynamics of reconfigurable modular robots [21]. These are useful for designer to develop the reasoning mechanism of abstract functions.

Conclusions
This paper presents a task-oriented architecture for space robot application, and the knowledge interchange mechanism for the knowledge interchange in this architecture. In the knowledge interchange mechanism, conceptual model is defined with ontology using KIF language. Communication protocol is defined with OKBC. The reasoning mechanism of abstract functions realizes the understandability of information crossing layers. To deal with abstract functions individually prevents the reasoning mechanism from higher-order predicate logic that is difficult to implement.
References

[5] Yadgar O.Kraus S. Ortiz C Hierarchical organizations for real time large-scale task and team environments, 2002