TELKOMNIKA, Vol.11, No. 4, April 2013, pp. 1857~1867
ISSN: 2302-4046

Knowledge Interchange in Task-Oriented Architecture for Space Robot Application

Cen Yu*[1], WeiJia Zhou[2]
[1]University of Chinese Academy of Sciences, Beijing 100049, CHN
[2]Shenyang Institute of Automation Chinese Academy of Sciences, State key Laboratory of RoboticsNanta Street 114#, Shenyang, Liaoning Province, China, 110016E, 086-24-23970937
*Corresponding author, e-mail: yucen@sia.cn

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.

Keywords*: knowledge interchange, multi-robot system, task-oriented architecture, space robot*

1. 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.

Received December 28, 2012; Revised February 11, 2013; Accepted February 22, 2013

(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.

Knowledge is theoretically expressed independently of the ways it will be used. It should be easier to maintain the system through modifications, deletions and additions to the knowledge base. Different exploitation mechanisms can be applied to the same knowledge base. A knowledge-based system is able to provide some form of justifications of the results it produces and explanations on the problem solving process it has followed to obtain them [14].

A knowledge-based system (KBS) can be considered an extension of a deductive database in that it permits function symbols as part of the theory [15]. The architecture of knowledge-based system attempts to explicitly separate the expression of knowledge from its exploitation mechanisms. Classical Programming languages do not support this separation. Any program certainly incorporates knowledge, but this knowledge is always mixed up with control. The separation of knowledge from its exploitation mechanisms requires a knowledge representation language.

Different knowledge models have been studied for knowledge-based systems. The most popular on remains the rule-based model, for which different implementations have been proposed [14]. The knowledge model can conveniently describe knowledge by text form, but text form is disadvantageous to classification and logical reasoning [16]. The traditional means has inherent weakness in the process of knowledge representation and knowledge sharing. For solving this problem, knowledge ontology technology is used in the field of knowledge engineering. It is an ideal approach to construct knowledge case based on ontology in order to store and represent domain knowledge [17].

As the view of ontology, the knowledge includes three parts: the logic, ontology and computing. The logic is the function that can deduce new logical representation from the present knowledge by logical computing. The computing is a deducing process. The ontology is a common, shared and formal description for important concepts in a specific domain, which includes a specification of the terms used (terminology) and agreements that allow to determine their meaning, along with the possible inter-relationships between these terms, standing for concepts [18].

The remainder of this paper is organized as follows: section 2 presents the task-oriented architecture for space robot application. Section 3 describes the related works. Section 4 describes the knowledge interchange mechanism based on KIF and OKBC and gives an example. In section 5, author gives a conclusion.

2. The Proposed Method

Architecture is the fundamental concepts or properties of a system in its environment embodied in its elements, relationships, and in the principles of its design and evolution.

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. Three kinds of tasks in task-oriented architecture are shown in Table 1.

Table 1.Three Kinds of Tasks in Task-oriented Architecture

Classification	Abstract task	Specific task	Remote-control task
Content	Abstract description that ignore some details	Specific working mode and parameters	Operator's task
Execution	Comprehend, plan, execute	Predefined routine	Follow the input

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

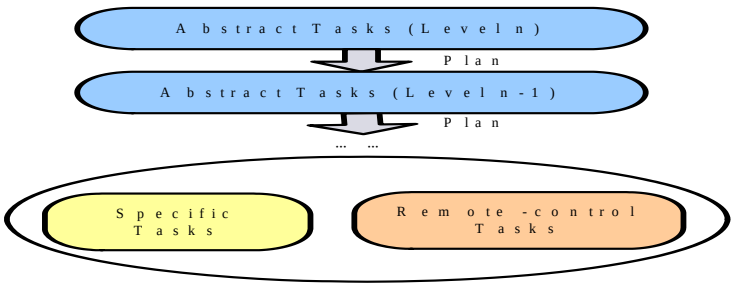

Figure 1. Tasks in Task-oriented Architecture

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 Figure 2.

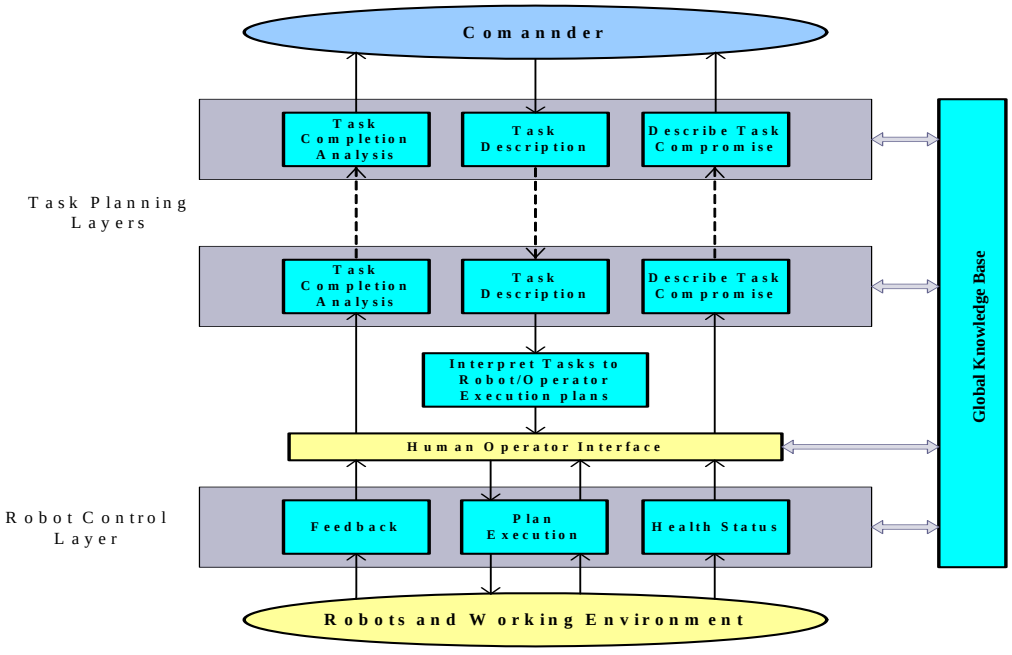

Figure 2. Task-oriented Architecture

Task planning layers consist of planners with different duties. Planner is human or AI system. A planner deals with task demanded by higher layer planner (commander for the highest layer), gives a plan to accomplish the task, assigns plan to lower layer planner (robot for the lowest layer), and gathers information about the plan execution.

Robot control layer consist of robots with predefined interface and working mode. Sometimes, robot remotely downloads codes to change working mode. Lowest layer planners should know the interface and working mode of robots they control.

Planners and robots form a hierarchy tree structure. They interchange knowledge with their parent node and children nodes.

In essence, Multi-robot system is a distributed system. Planers and robots have local knowledge bases with different terminologies, and the knowledge bases are possibly developed by different knowledge languages. Local knowledge bases are assigned to corresponding layers in global knowledge base. Task information is interchanged between adjacent layers, and cooperation information is interchanged in layers. As shown in Figure 3.

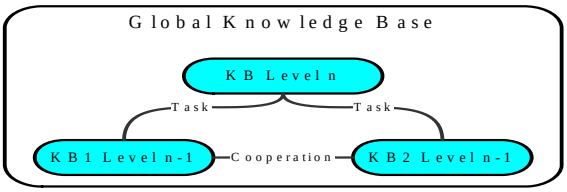

Figure 3. Knowledge Interchange in Global Knowledge Base

Knowledge interchange in global knowledge base defines format and meaning of the information. It 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.

3. Research Method

The knowledge about the world and tasks is represented by models in robot system [13]. Table 2 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 2. Two Kind of Model

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

Early work on enabling technology for knowledge sharing established that three components are needed to allow knowledge to be shared between two knowledge base [19] :

- 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) [20]. 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 [21].

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 ρ^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> [22].

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 $\Im : V \to D \bigcup \Re$ is a function assigning elements of D to constant symbols of V, and elements of $\Re$ to predicate symbols of V [23].

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 computer-oriented language for the interchange of knowledge among disparate programs. It has declarative semantics (i.e. the meaning of expressions in the representation can be understood without appeal to an interpreter for manipulating those expressions); it is logically comprehensive (i.e. it provides for the expression of arbitrary sentences in the fires-order predicate calculus); it provides for the representation of knowledge about the representation of knowledge; it provides for the representation of nonmonotonic reasoning rules; and it provides for the definition of objects, functions, and relations [24]. As a pure specification language, KIF does not include commands for knowledge base query or manipulation. 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 [25].

4. Results and Analysis

4.1. Knowledge Interchange Model

In the task planning layers, the human and AI systems are called planners. Every planner has a knowledge base. kb_1^n , kb_2^n and so on refer to the knowledge bases in layer n. Knowledge interchange model defined the relations of knowledge bases, as shown in Figure 4.

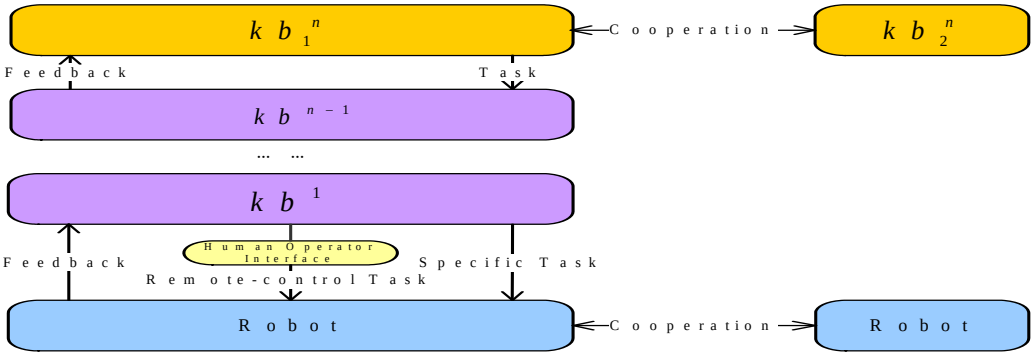

Figure 4.Knowledge Interchange Model

In same layer, knowledge bases have consistent terminologies. Knowledge is transformed into ontology described by middle language, and delivered through communication protocol. Specifically, terminologies of knowledge bases in different layers are inconsistent. The ontology of higher layer knowledge base is abstracted by lower layer knowledge bases. They use different terms or same term with different meanings. The interchange between layers needs abstract process. As shown in Figure 5.

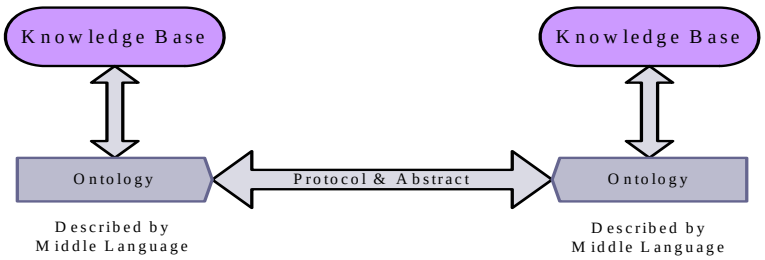

Figure 5. Knowledge Interchange Process

Knowledge interchange takes the steps as follows:

(1) Ontology translation. Knowledge expressed by local language of source knowledge base (KB) is translated into middle language. Each kind of local language needs a translation program.

(2) Communication. Ontology described by middle language is delivered to target KB. Communication protocol defines the KB operations, such as establishing a connection, setting or getting values.

(3) Abstraction. When a level n-1 KB communicates with a level n KB, they use different terminology or same terminology with different meanings. Abstraction is the set of logic relations between terminologies of adjacent levels. Feedbacks from level n-1 KBs is computed to change level n KB.

(4) Ontology translation. After communication and abstraction (if necessary), ontology expressed by middle language is translated into local language of target KB. Then target KB can comprehend the knowledge from source KB. As before, we need a translation program to translate middle language into local language of target KB.

Knowledge interchange process will be discussed in more detail at later stage.

4.2. Ontology Described by KIF

An ontology is composed of the following entities [26]:

- A set of concepts;
- A set of relations;
- A set of axioms;

Frame ontology contains a complete axiomatization of classes and instances, slots and slot constraints, class and relation specialization, relation inverses, relation composition, and class partitions. We can use it to describe conceptualization C=<D, W, $\Re$ >.

Domain space <*D, W*>: A class is a set of individuals. Each of the individuals 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.

Thing is the root of class hierarchy for a KB, meaning that Thing is the superclass of every class in every KB. Class is the class of all classes. That is, every entity that is a class is an instance of Class. Individual is the class of all entities that are not classes. That is, every entity that is not a class is an instance of Individual. Domain space is shown in Figure 6.

Set of conceptual relations $\Re$: 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 ρ^n between n entities (n ≥ 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)

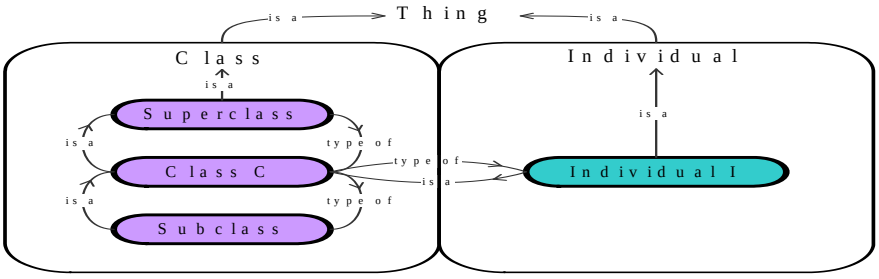

Figure 6. Domain Space

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)

Ary-n conceptual relation can be represented by frame as shown in Figure 7.

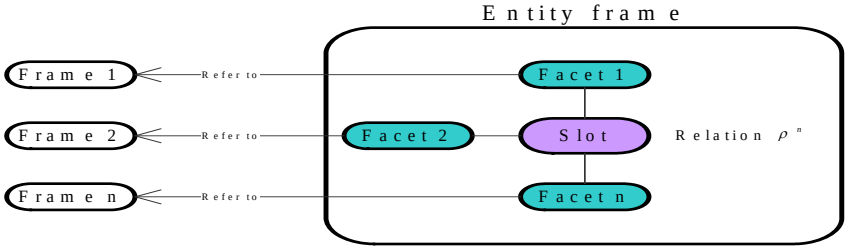

Figure 7. Ary-n Conceptual Relation Represented by Frame

Using KIF language, we can describe definitions, relations and axioms as follows:

- Concept definitions

(Class ?C) means that C is a class, and (Class @C) means that C is a sequence of classes.

A variable that begins with ? is called an individual variable.

A variable that begins with an @ is called a sequence variable.

- Relations

(Type-of ?C ?I) means that class C is the type of instance I.

(Slot-value ?S ?F ?V) means that V is the value of slot S in frame F, and so on.

- Axioms

Sentence operators are = | /= | not | and | or | => | <= | <=> | forall | exists.

An axiom is just like:

```
( <=>   ( subclass-of  ?Csub  ?Csuper)
        ( forall  ?I  (  => (instance-of  ?I  ?Csub)
                            (instance-of  ?I  ?Csuper) ) ) )
```

Knowledge about frames can be stored and delivered by script written in KIF language. As shown in Figure 8.

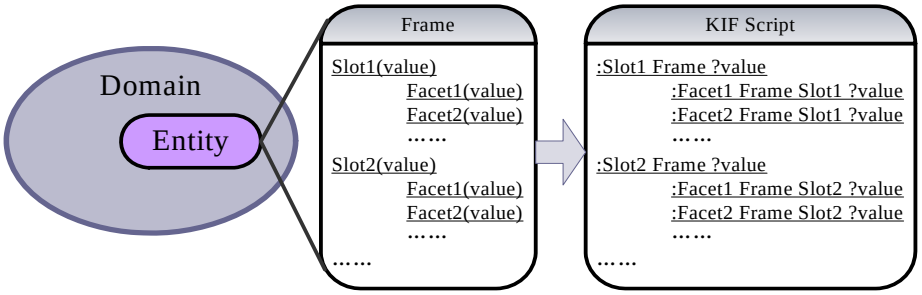

Figure 8. Entity Frame

4.3. Protocol based on OKBC

OKBC specifies a knowledge model of knowledge representation systems (KRSs) with KBs, classes, individuals, slots, and facets. It also specifies a set of operations based on this model (e.g., find a frame matching a name, enumerate the slots of a frame, and delete a frame). An application uses these operations to access and modify knowledge stored in an OKBC-compliant KRS [25].

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.

(1) The application must establish a connection to an OKBC server (using establish-connection).

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

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

(4) 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).

4.4. 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.

$$Frame^{n} = f_{com}^{n}\left(Frame_{1}^{n-1}, Frame_{2}^{n-1}, \ldots\ldots Frame_{k}^{n-1}\right)$$

$$Frame^{n} \in kb^{n},\ Frame_{1}^{n-1}, Frame_{2}^{n-1}, \ldots\ldots Frame_{k}^{n-1} \in kb^{n-1}$$

f_{com}^{n} 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 Figure 9.

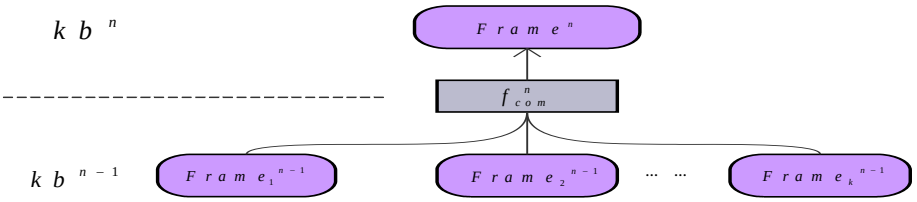

Figure 9. The Abstract between Levels

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 Figure 10.

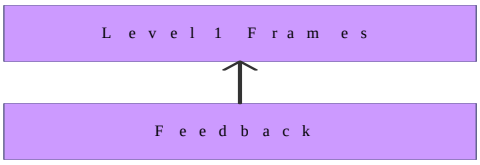

Figure 10.Level 1 Abstract

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 Figure 11.

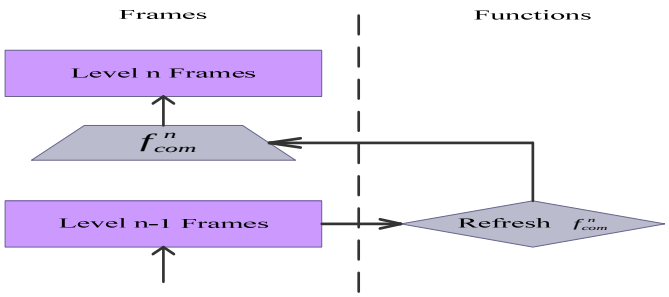

Figure 11. Level n Abstract (n>1)

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.

4.5. 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 Figure 12.

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.

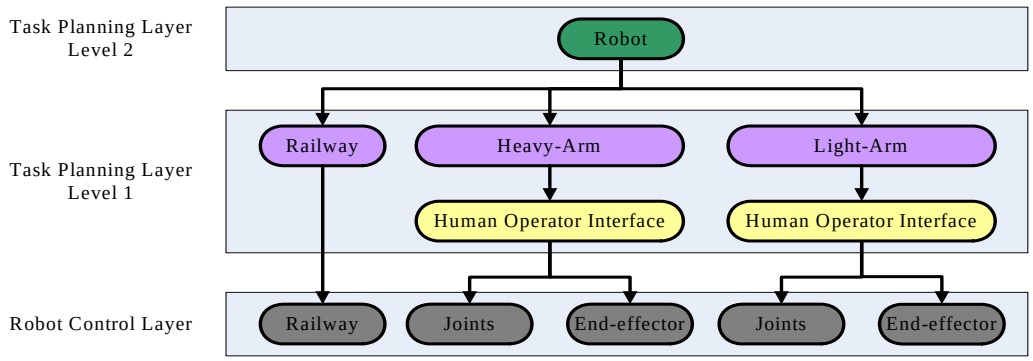

Figure 12.The Manipulator System in Space Station

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 [27]. These are useful for designer to develop the reasoning mechanism of abstract functions.

5. Conclusion

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

[1] Pedersen L, Kortenkamp D, Wettergreen D, Nourbakhsh I, Smith T. Space robotics technology assessment report. *National Aeronautics and Space Administration.* 2002.

[2] Salemi B, Moll M, Wei-Min Shen. SUPERBOT: A Deployable, Multi-Functional, and Modular Self-Reconfigurable Robotic System. *Intelligent Robots and Systems*. 2006; 3636-3641.

[3] Yim M, Roufas K, Duff D, Ying Zhang, Eldershaw C, Homans S. Modular Reconfigurable Robots in Space Applications. *Autonomous Robots.* 2003; 14: 225–237.

[4] Farinelli A, Scerri P, Tambe M. Building Large-scale Robot Systems: Distributed Role Assignment in Dynamic, Uncertain Domains. *In AAMAS'03 Workshop on Resources, role and task allocation in multiagent systems*. 2003.

[5] Yadgar O, Kraus S, Ortiz C. Hierarchical Organizations for Real Time Large-scale Task and Team Environments. *AAMAS '02 Proceedings of the first international joint conference on Autonomous agents and multiagent systems*. 2002; 3: 1147-1148.

[6] Konolige K, Fox D, Ortiz C. Centibots: Very Large Scale Distributed Robotic Teams. *IFIP 18th World Computer Congress Topical Sessions.* 2004; 22–27.

[7] Bernarine D. *TraderBots: A New Paradigm for Robust and Efficient Multirobot Coordination in Dynamic Environments*. Carnegie Mellon University Robotics Institute. 2004.

[8] Parker L E. ALLIANCE: An Architecture for Fault Tolerant Multi-Robot Cooperation. *IEEE Transactions on Robotics and Automation*. 1998; 14(2): 220-240.

[9] Laengle T, Lueth T C, Rembold U, Woern H. A Distributed Control Architecture for Autonomous Mobile Robots-Implementation of the Karlsruhe Multi-agent Robot Architecture (KAMARA). *Advanced Robotics, International Journal of the Robotics Society of Japan*. 1998; 4(12): 411-432.
[10] Tambe M. Towards Flexible Teamwork. *Journal of Artificial Intelligence Research*. 1997; 7:83-124.
[11] Caloud P, Choi W, Latombe JC, Pape CL, Yim M. Indoor Automation with many Mobile Robots. *Intelligent Robots and Systems '90*. 1990; 1: 67-72.
[12] Noreils FR. Toward a Robot Architecture Integrating Cooperation between Mobile Robots: Application to Indoor Environment. *The International Journal of Robotics Research February*. 1993; 12(01): 79-98.
[13] Nilsson NJ. Artificial Intelligence: a New Synthesis. 1 edition. Morgan Kaufmann Publishers. 1998.
[14] Rechenmann F. Declarative and Procedural Object-based Knowledge Modeling. *Systems, Man and Cybernetics, 'Systems Engineering in the Service of Humans', Confer*. 1993; 98.
[15] Baral C, Kraus S, Minker J. Combining Multiple Knowledge Bases. *Knowledge and Data Engineering*. 1991; 3(2): 208-220.
[16] Pham DT, Gourash NS. Knowledge-based Configuration Design. *Industrial Informatics, INDIN 2003 Proceedings, IEEE International Conference*. 2003; 248-254.
[17] Chenjian Hao. Research on Knowledge Model for Ontology-Based Knowledge Base. *Business Computing and Global Informatization (BCGIN)*. 2011; 397-399.
[18] Gruber TR. A Translation Approach to Portable Ontologies. *Knowledge Acquisition*. 1993; 5(2): 199-220.
[19] Waterson A, Preece A. Verifying Ontological Commitment in Knowledge-based Systems. *Knowledge-Based Systems*. 1999; 12: 45-54.
[20] Gruber TR. *A Translation Approach to Portable Ontology Specification*. Stanford University. Report number: Logic-92-1. 1993.
[21] Devedzie V. A Survey of Modem Knowledge Modeling Techniques. *Expert Systems with Applications*. 1999; 17: 275-294.
[22] Guarino N. Formal Ontology and Information Systems. *In: Proc of the 1st Int'l Conf on Formal Ontology in Information Systems*. 1998; 3 -15.
[23] Genesereth MR, Nilsson NJ. Logical Foundations of Artificial Intelligence. San Mateo: Morgan Kaufmann Publishers. 1987.
[24] Genesereth MR, Fikes R. *Knowledge Interchange Format, Version 3.0 Reference Manual*. Stanford Logic Group. Report number: Logic-92-1. 1992.
[25] Chaudhri VK, Farquhar A, Fikes R, Karp PD, Rice JP. *Open Knowledge Base Connectivity 2.0.3*. Knowledge Systems Laboratory, Stanford. 1998.
[26] Maillot N, Theonnat M, Bouchere A. Towards Ontology-based Cognitive Vision. *Machine Vision and Applications*. 2004; 16(1): 33-40.
[27] I-Ming Chen, Guilin Yang. Inverse Kinematics for Modular Reconfigurable Robots. *International Conference on Robotics & Automation*. 1998; 2: 1647-1652.