Scene-Graph Design for Knowledge Memory Space

Boonprasert Surakratanasakul [1] and Kazuhiko Hamamoto [2]

[1] Graduate School of Science and Technology, Tokai University, Japan
[2] School of Information and Telecommunication Engineering, Tokai University, Japan

1 Introduction

On knowledge management strategies, knowledge engineer needs to explore a lot of past knowledge information according to a current situation and then pile the new ideas upon. The difficultly is to find an appropriate way for manage and maintain huge knowledge's information in a schematic way on an understanding view. Knowledge memory system on virtual space is a scenic representation of huge information that is organized spatially and temporally [1]. It enables knowledge engineer to visually grasp and explore the information space for accommodate new information at an appropriate place based on manner understandable and knowledge management concept.

To develop a knowledge memory space, we create a knowledge schema by with CommonKADS methodology [2], is called "Knowledge Landscape" schema [3]. The purpose of the schema is to explicate in detail the types and structures of the knowledge used in performing task and concentrate on the conceptual structure of knowledge. For simulation the virtual memory space, we design a scene graph from the schema, to be an imaginary guideline before implementation. The benefit from manage knowledge on the virtual space are simple and well-known by learning from experiences in a controlled spiral that understandable for humans. In addition, it makes an important vehicle role for communication with experts and users during both development and system execution.

2 State of the Art

CommonKADS methodology originates from the need to build industry-quality knowledge systems on a large scale, in a structured, controllable, and repeatable way. It has a predefined set of models that focusing and providing a comprehensive view for developing knowledge system. The one important model is the knowledge model, that we choose to apply for develop the virtual space.

2.1 Knowledge Landscape Schema

Knowledge landscape is a technique that helps to clarify the structure of a knowledge-intensive task and provide a specification of data and knowledge structures. It has an essence structure that similar to analysis models in software engineering. So we apply an UML extension mechanism [4] to develop schema for these system.

In knowledge landscape concept, it consists of three parts each capturing a related group of knowledge structure called "Knowledge category": Domain knowledge, Inference knowledge, and Task knowledge. Domain knowledge specifies the domain specific knowledge and information types. Its modeling implies capturing the static structure of information and knowledge types. Inference knowledge describes the basic inference steps that want to make using domain knowledge. It describes how these static structures can be used to carry out a reasoning process. Task knowledge describes what goals and application pursues, and how these goals can be realized through decomposition into subtasks and ultimately inferences. Fig. 1 shows the knowledge landscape schema that applied from the CommonKADS methodology.

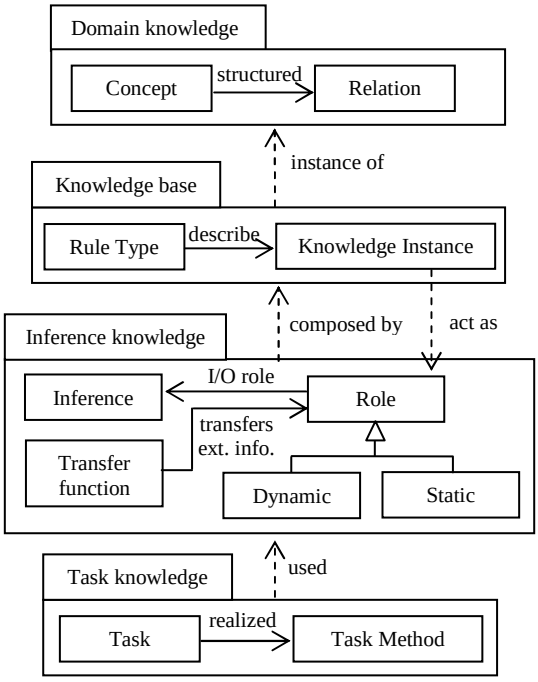

Figure 1. Landscape of Knowledge Schema.

2.2 World Tool Kit™'s Scene graph

World Tool Kit™ (WTK) is a virtual reality library written in C/C++. It manages the details of reading sensor input, rendering scene geometry, and loading databases. An application developer only needs to worry about manipulating the simulation and changing the scene graph based on user inputs. In addition, it also allows the developer to edit the scene graph by hand if that level of control is needed [5].

Generally, a common definition of a graph is a data structure composed of nodes and arcs. A node is a data element, and arc is a relationship between data elements [6]. To render the scene graph, WTK provides function for creating nodes and placing them at specific positions in the scene graph. The developer might be concerned the scene graph in common rules:

• The scene graph is rendered automatically into the window as the simulation runs.

• Different scene graphs may have common sub-trees. This means that the same geometry can be referenced by more than one scene graph.

• Each scene graph has a single root node.

• Traversal begins at the root node of the scene graph.

• The renderer traverses the tree from top to bottom and left to right.

• Depending on the type of node, WTK will do different things. Nodes in WTK can be grouped into three distinct types: Geometry node, Attribute node, and Procedural node.

Geometry nodes contain the representation of visible entities by draw the specified set of polygons. Attribute nodes used to affect the way geometry nodes are rendered by modify the current state, which determines the appearance of subsequent geometry. Procedural nodes used to control the way a scene graph is put together by process the children of this node, depending on the type of traversal directed by the node.

For managing the state of the scene graph, Separator and Transform Separator nodes are used to manage the state of the scene graph by isolating the effects of the attribute nodes.

3 Knowledge Memory Space's Scene-graph

From the knowledge landscape schema, we design the scene graph shown in Fig 2.

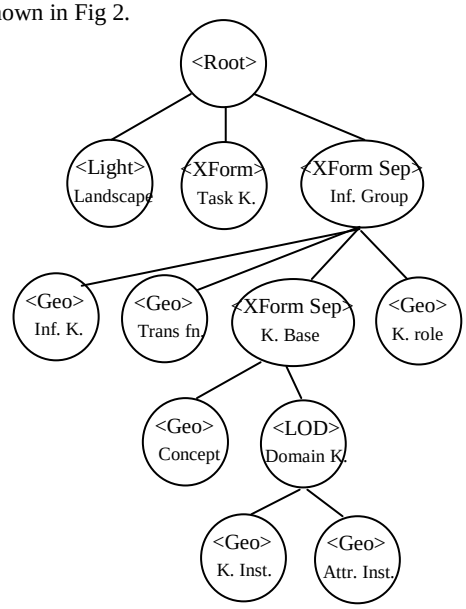

Figure 2. Scene Graph of Knowledge Memory Space.

From the scene graph, it starts at the root node. The root node has relationships with three nodes. Light node specifies a light for the landscape virtual space. Transform node sets position and orientation information for develop Task method into Task knowledge. The transform separator manages the sub-graph of Inference knowledge level. In inference level, it consists of geometry node of Inference knowledge, Transfer function, and Knowledge role. The transform separator in this level separates the Domain knowledge level by scope of the knowledge base. In this level, it has concept node and level of detail (LOD) node that describes the knowledge instance and attribute instance in geometry node.

For imaginary outstanding, we show a draft of simple simulation scene in Fig 3. This scene has two layers: functional view layer and logical view layer. The functional view describes the knowledge function with the Inference and Task knowledge. The logical view describes the knowledge entity and clarifies the concept of knowledge. Both of layers correlate by knowledge instance (in logical view) and knowledge role (in functional view).

4 Conclusion

Knowledge memory space is a memory system that has an objective view enables a user to edit contents on surface by using geographical arrangement and topological connection. It provides an overview of large data content and facilitate for knowledge sharing by people and increasing their connectivity as a lightweight activity.

To develop the virtual memory space, the scene graph is an important tool for constructed render the simulation. It completely specifies the state of information on knowledge node, for examples; the location, orientation, and properties of knowledge. We develop the scene graph from the knowledge landscape schema, which applies from the CommonKADS knowledge model.

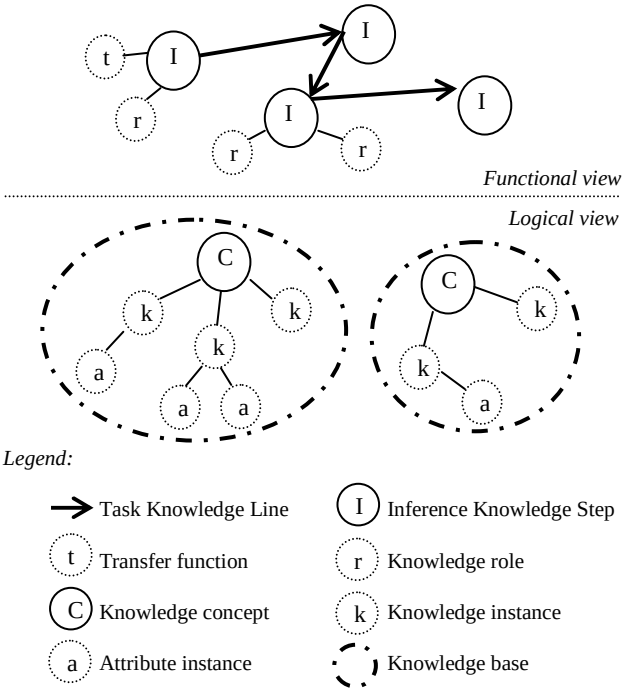

Figure 3. Simulation scene of Knowledge Memory Space

References

[1] Kubota H., Nomura S., Sumi Y., and Nishida T., "Sustainable Memory System using Global and Conical Spaces", Journal of Universal Computer Science, vol.13, no.2, 2007, pp. 135-148.

[2] Guus Schreiber, Hans Akkermans, Anjo Anjewierden, Robert de Hoog, Nigel Shadbolt, Walter Van de Velde, and Bob Wielinga, "Knowledge Engineering and Management: The CommonKADS Methodology", A Bradford Book, The MIT Press, 1999.

[3] S. Boonprasert, K. Hamamoto, "CommonKADS's Knowledge Model using UML Architectural View and Extension Mechanism", 7th International Conference on Advanced Information Management and Service, Jeju Korea, November 2011.

[4] Booch G., Rumbaugh J., Jaconson I., "The Unified Modeling Language User Guide", Massachusetts: Addison Wesley, 1999.

[5] Leno Franzen, Hans Kessock, and Dave Hinkle, "WorldToolKit Reference Manual", Engineering Animation inc., April 1999.

[6] Dennis J Bouvier, "Getting Started with the Java 3D API", Sun Microsystems, 1999.