Development Technique for Dynamic Node Management of Visual Modeler

C. R. Yoon and K. O. Kim
Spatial Imagery Information Team, ETRI
161 Kajeong-dong Yuseong-gu, Daejeon 305-350, Korea
{cryoon, kokim}@ksrs.or.kr

Abstract: Spatial image processing software requires various user interactions to make a plan, prepare necessary data such as images, vectors, ancillary data and user-defined data, execute functions according to pre-defined procedures, analyze and store the results. In this manner, overall processes are controlled by user interactions. In this paper, we propose visual modeler which has the automated spatial image processing technique to minimize user interactions and re-use repeatable procedure. The proposed visual modeler is designed to use inter-operable components proposed by OpenGIS consortium as well as conventional COM components.
Keywords: Visual Modeler, COM

1. Introduction

We can find out the advances in the part of spatial image processing systems which provide powerful functionalities as well as systemic solutions. Because the spatial image manipulation is complex and time consuming, these processing systems should be able to lead to the goal by flexible and easy user interfaces to use.

In this paper, we propose visual modeler to help to make and execute processing models via visual programming user interaction. The proposed visual modeler recognizes pre-defined node components, edit visual models by using node components and run the edited models. In addition, man-made node components are added in the node pool in anytime.

The additional functionalities of the visual modeler are to manage the node pool which is composed of single or group node, to control data flow of visual model, store the edited models in the storage to use later. These additional functionalities can be co-operated with major functionalities.

In this paper, we describe the principal design of visual modeler, specification of the node component and the technique to control Data Flow of visual modeler.

2. Visual Modeler Design

The visual modeler proposed in this paper has several node components 1) to read and write versatile spatial image formats such as CEOS, HDF, GeoTIFF, etc., 2) to do spatial analysis such as convolution filtering, statistical filtering, adaptive filtering and texture filtering, and 3) to do spectral transformation such as principal component transformation, vegetation index transformation and color space transformation. In addition, geometric processing such as 2D geometric correction and image mosaic are supported, too. Fig. 1 shows the overview design of visual modeler.

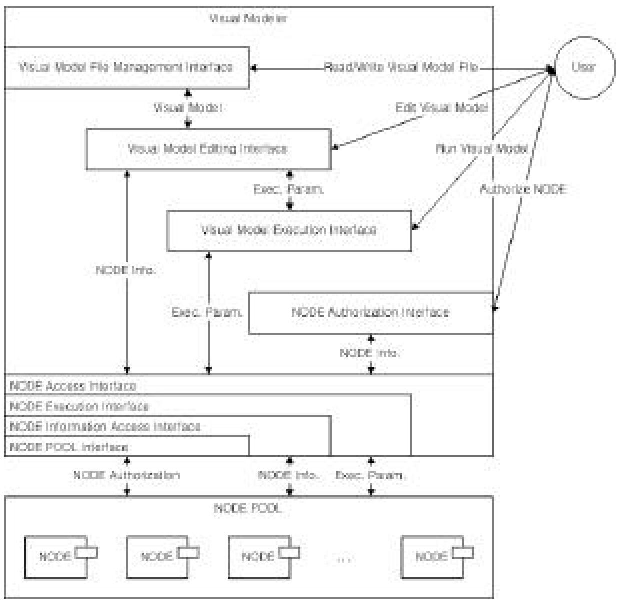

Fig. 1. Overview of visual modeler.

As shown in Fig. 1, visual modeler communicates with node pool by using node access interface to authorize node components, get node information and execute node in visual model. The node access interface is composed of node pool interface to manage node pool, node information access interface to get node information such as node GUI and node execution parameter and node execution interface to run node component. Furthermore, visual modeler provides client-side interfaces, which are 1) visual model file management interface to read and write visual model file, 2) visual model editing interface to create and modify visual model via visual programming GUI, 3) visual model execution interface to run user-defined visual model and 4) node authorization interface to add and remove node components in node pool. These client-side interfaces communicate each other. Visual model file management interface and visual model editing interface exchange visual model object to read, change and write it. Visual model editing interface and visual model execution interface give and take node parameters to run visual model. Node access interface gives node information to visual model editing interface and node parameter to visual model execution interface.

Basically, visual modeler should be able to create, edit, execute and write visual model composed of various nodes. Node component proposed in visual modeler is encapsulated into ActiveX control with its own GUI and functionalities. In this paper, we designed node component by using OpenGIS components, which conform to grid coverage implementation specification recommended by OpenGIS consortium. The architecture of OpenGIS grid coverage encapsulated into node component is composed of three component packages, which are coverage, grid coverage and grid coverage processor as shown in Fig. 2.

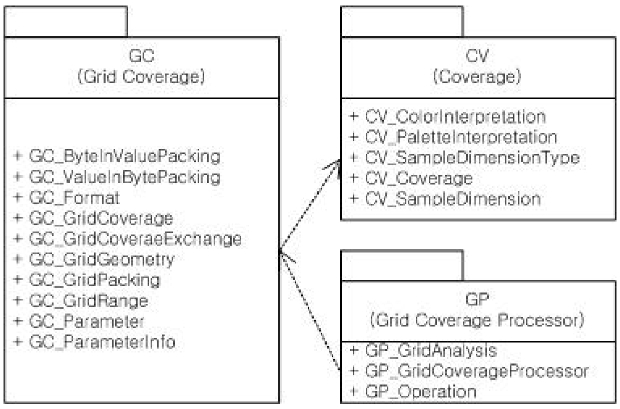

Fig. 2. Architecture of OpenGIS Grid Coverage package.

The grid coverage specification can accommodate grid coverages with the following characteristics: variable number of bits per grid value, multi-bands, multi-dimensions, various band ordering, variable number of no data values, various color models, reduced resolution data sets, grid coordinate systems, tiling data sets and various color palettes. The CV (coverage) interface package is abstract interface package providing access to an OpenGIS coverage. The essential property of coverage interface package is to be able to generate a value for any point within its domain however coverage is represented internally. A coverage may be represented by a set of polygons which exhaustively tile a plane, a grid of values, a mathematical function or combination of these. The GC (grid coverage) interface package would represent the basic implementation which provides access to grid coverage data. A GC interface package provides the ability to update grid values. A basic read-only implementation would be fairly easy to implement. The GP (grid coverage processor) interface package is composed of optional interfaces and provides operations for different ways of accessing the grid coverage values as well as image processing functionality. The list of available processing operations is implementation dependent. The GP interface package has a discovery mechanism to determine the available processing operations. Node components in visual modeler encapsulate CV, GC and GP interface package to add specific GUI and to interpret user-defined visual model. Node components are classified into three specific nodes, which are data node to provide data such as image, scalar, vector, matrix, string and table, processor node to read, process, analyze and write data, and annotation node to decorate visual model. Each node has unique interface to access them easily. But, their parameters are different according to their purposes. Node component has the node access interface described in Fig. 1. Each interface does its own role with client request.

Node execution interface is important because it decides execution sequence of visual model, runs the each node of visual model, transfers the result of each node processing into next node. Visual modeler uses discovery and pipe-lined mechanism to process specific operation in node component. Discovery and pipe-lined mechanism lets the client to have no copies of image data during processing. In this way, node component object requests the data block in previous node component object and then the requested node component object processes its implicit operation to return the result data through pipe-line. It is recommended that client uses data node object if a processor node should execute operations with user-defined or pre-calculated specific input parameter. The input and output image parameters of all node components in visual modeler are defined into OpenGIS GC interface package. If necessary, other type data of data node are engaged in input parameters. So, discovery and pipe-lined processing mechanism of visual modeler is basically identical to that recommended by OpenGIS consortium. Fig. 3 shows discovery and pipe-lined mechanism to threshold an image.

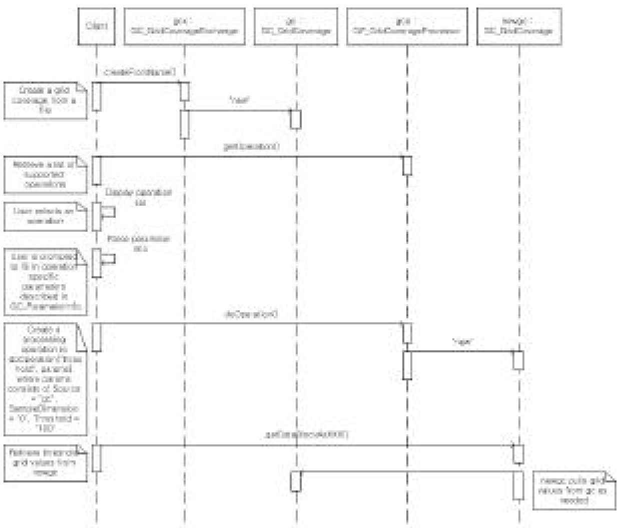

Fig. 3. Discovery and pipe-lined processing mechanism.

As shown in Fig. 3, client requires the creation a GC object from a grid coverage exchange object and invoking a processing operation after discovering the available operations in GP object. Grid values are retrieved from the modified GC object and the GP object fetches grid values from the source GC object as required. Node component embraces all sequences of this mechanism to produce specific result so client creates visual model to achieve its goal without detailed knowledge on this mechanism.

Node components in visual modeler are dynamically attached and detached into node pool by node authorization interface. Node components should be developed in standard node implementation specification to be recognized and executed by same ways. The developed node components should be registered to be used in visual model later. Visual modeler recognizes node component in node pool as ready-to-use node component and update its own usable node list. Node component has the below characteristics at a minimum:

- Identify(..) : announces its identification such as CLS_ID, Prog_ID and/or private identification information
- Parameters(..) : informs input and output parameter information such as number of parameters, parameter types, and/or parameter values
- ShowGUI(..) : activates specific GUI to obtain parameter values if necessary
- Run(..) : executes internal functionalities using input parameters and return result as output parameter

Node component should be able to announce its identification to be recognized and registered by visual modeler. Identification information in node component is transferred to visual modeler and then node pool database to be used later. Parameters of node component are used to execute visual model efficiently. These parameters are decided in editing or executing visual model and stored in memory or physical storage if necessary. To decide parameter value, visual modeler should be able to give a way to enter the parameter value in node object. Node GUI is not essential characteristics but visual modeler should provide other means to decide parameter value. The execution order of visual model is determined when execution event is invoked and execution code of node object returns their result after running. Fig.4 shows the screen capture of developed visual modeler.

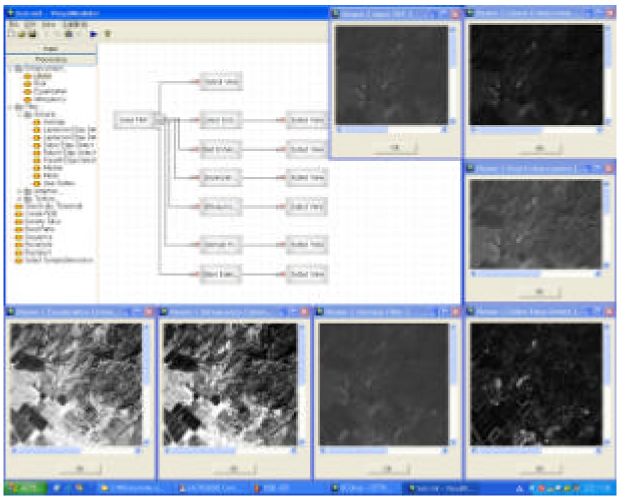

Fig. 4. Screen capture of developed visual modeler.

As shown in Fig. 4, we classified all nodes into 3 top-level categories, which are input node, process node and output node. The input node category is composed of several spatial image file format data provider nodes. The process node category has enhancer node, filter node and data preparation node. The output node category has view node and several file writing node. The node categories are dependent on implementing visual modeler client.

3. Conclusions

In this paper, we propose visual modeler to help to make and execute processing models via visual programming user interaction. The proposed visual modeler recognizes pre-defined node components, edit visual models by using node components and run the edited models. In addition, man-made node components are added in the node pool in anytime.

The proposed visual modeler helps to establish, run and analyze many complex satellite imagery processing procedures by using the visual programming method. Visual modeler has two major merits. One is that users can obtain and reuse complex satellite imagery processing procedures very easily. The other is that users can minimize the processing cost by executing pre-defined batch-job procedures.

Acknowledgement

The Radio Research Laboratory in Ministry of Information and Communication is acknowledged for supporting this research through a grant “Development of High Resolution Satellite Imagery Processing Technology.”

References

[1] Kim, K. S., 2001. The Implementation of Grid Coverage Processing Component in the OpenGIS’s Grid Coverage Specification, *Proc. ISRS’2001*, Korea, pp.225-228.

[2] Kim, H. K. and K. O. Kim, 2002. The Implementation of a Pipe-lined Grid Coverage and Grid Coverage Processor, *Proc. ISRS’2002*, Korea, pp.70-73.

[3] Yoon, C. R., K. O. Kim and Y. K. Yang, 2002. Component-based Architecture Design for Fast Interoperability in Satellite Image Processing, *Proc. ISRS’2002*, Korea, pp.45-48.

[4] Yoon, C. R., J. H. Seo and K. O. Kim, 2002. Development of High Resolution Satellite Image Processing System by using CBD, *Proc. ISRS’2002*, Korea, pp.49-52.

[5] **URL:** OpenGIS Consortium. OpenGIS Specification Documents. Available at: http://www.opengis.org/.