基于构件的地理工作流框架:一个方法学的探讨*

刘 瑜[1], 高 勇[1,2+], 王映辉[3,4], 邬 伦[1], 王立福[3]

[1](北京大学 遥感与地理信息系统研究所,北京 100871)

[2](北京大学 信息科学中心,北京 100871)

[3](北京大学 计算机科学技术系,北京 100871)

[4](陕西师范大学 计算机学院,陕西 西安 710062)

A Component-Based Geo-Workflow Framework: A Discussion on Methodological Issues

LIU Yu[1], GAO Yong[1,2+], WANG Ying-Hui[3,4], WU Lun[1], WANG Li-Fu[3]

[1](Institute of Remote Sensing and Geographic Information Systems, Peking University, Beijing 100871, China)

[2](Center for Information Science, Peking University, Beijing 100871, China)

[3](Department of Computer Science and Technology, Peking University, Beijing 100871, China)

[4](School of Computer Science, Shanxi Normal University, Xi'an 710062, China)

+ Corresponding author: Phn: +86-10-62751186, Fax: +86-10-62751187, E-mail: gaoyong@pku.edu.cn, http://geosoft.pku.edu.cn

Received 2004-03-25; Accepted 2004-10-09

Liu Y, Gao Y, Wang YH, Wu L, Wang LF. A component-based geo-workflow framework: A discussion on methodological issues. *Journal of Software*, 2005,16(8):1395−1406. DOI: 10.1360/jos161395

Abstract: Software framework (SF) brings forth lots of conveniences for software reuse in specific domains. It is well-known that a framework is much harder to be developed than a common reusable component. Based on the conception of component-based software framework (CBSF) that provides black-box reuse approach, a development process for geo-workflow domain is in discussion. A geo-workflow application is one sort of workflow management systems applied in the geographic information domain, and can be implemented through reusing component-based geo-workflow framework (CBGWF). From the methodological point of view, the following steps are necessary to develop a CBGWF: domain analysis, domain design, framework design and implementation. The outputs of them are domain models with identified variabilities, domain-specific software architecture (DSSA), and products of software framework, respectively. In the context of geo-workflow, domain variabilities, which are

* Supported by the National Natural Science Foundation of China under Grant No.40352002 (国家自然科学基金); the National High-Tech Research and Development Plan of China under Grant No. D0120-40201042 (国家高技术研究发展计划(863))

LIU Yu was born in 1971. He is an associate professor at the Institute of Remote Sensing and Geographic Information Systems, Peking University. His research areas are GIS theory and software engineering. **GAO Yong** was born in 1974. He is a post-doctor at the Center for Information Science, Peking University. His research areas are GIS theory, GI service and geo-workflow. **WANG Ying-Hui** was born in 1967. He is a post-doctor at the Department of Computer Science and Technology, Peking University. His research area is software engineering. **WU Lun** was born in 1964. He is a professor and doctoral supervisor at the Institute of Remote Sensing and Geographic Information Systems, Peking University. His researches areas are GIS theory and application. **WANG Li-Fu** was born in 1945. He is a professor and doctoral supervisor at the Department of Computer Science and Technology, Peking University. His research areas are software engineering and information security.

identified and classified first, involve variabilities of geospatial data types, variabilities of geospatial data management, variabilities of geospatial operations, variabilities of processes, and variabilities of geospatial data representation. These variabilities can be organized in a tree view. Then, using HMB style, DSSA and architecture of CBGWF are designed, where the former can be regarded as a template of the latter. Finally, the framework component development based on EJB component model and the reuse of CBGWF are described in brief. Compared with other geo-workflow systems, the main advantage consists in its flexibility caused by the extensibility of CBGWF.

Key words: component-based software framework; geo-workflow; domain variabilities

摘 要: 软件框架为特定领域的软件复用带来了便利.众所周知,软件框架开发的难度要大于开发一个普通可复用构件.采用支持黑盒复用的基于构件的软件框架概念,探讨了地理工作流框架的开发过程.一个地理工作流应用是地理信息领域的工作流管理系统,它可以通过复用地理工作流框架实现.从方法学的角度看,为了开发地理工作流框架,需要进行以下活动:领域分析、领域设计、框架设计和实现.其输出分别是识别了领域变化性的领域模型、领域特定的软件体系结构(DSSA)、软件框架产品.在地理工作流的上下文中,首先对领域变化性进行了识别和分类,包括空间数据类型、空间数据管理、空间操作、过程和空间数据表现 5 个方面的变化性,它们可以组织成树状视图.然后,为了处理上述变化性,设计了 DSSA 和框架的软件体系结构,其中前者是后者的模板,而框架体系结构采用 HMB 风格.最后,对地理工作流框架构件的开发以及框架复用进行了简单描述.与其他地理工作流系统相比,其优势在于由于框架扩展能力带来的灵活性.

关键词: 基于构件的软件框架;地理工作流;领域变化性
中图法分类号: TP391 文献标识码: A

1 Introduction

A geographical information system (GIS) is a computer-based information system that enables capture, modeling, manipulation, retrieval, analysis and presentation of geographically referenced data[1]. With effective tools solving complex and usually ill- or semi-structured geospatial problems, it provides a framework to support decisions for the intelligent use and management of the Earth's resources and environment. At present, GIS is widely applied in socio-economic, environmental, and management applications, such as urban planning, environmental monitoring and hazard evaluation, etc. Most of such applications consist of complex geospatial processes that require variable models and long-term design efforts under the cooperation of many people and departments. The potential utility of software systems that integrates specific process models with GIS technology is well recognized[2]. Although much work has been accomplished[3−5], there is still a long way to go to integrate them seamlessly.

As the most recent technology for process support, a workflow management system (WFMS) aims at modeling and controlling the execution of processes in both business applications and scientific applications. When incorporating workflow technology with GIS, the geo-workflow system can be established to manage complex geospatial processes. Although some geospatial-process-support systems had been built up years before, GOOSE[6], Geo-Opera[7] and SPMS[8] as examples, the term of "geo-workflow" was first put forward by Weske[9] in WASA projects in 1997. Then a geospatial decision support system entirely based on workflow, WOODSS[10], was accomplished in 1999. WOODSS can interact directly with GIS, sparing environmental planners and decision makers the burden of low-level programming. Most of the afore-mentioned projects focused on the integration of GIS with modeling/workflow systems, and some software architectures were given[7,9]. However, they are either built on commercial workflow systems or particular to some very domain. In the former case the geo-workflow

systems are short of specialties of geospatial processes, while in the latter case they are too specialized to be reused. Because of the characteristics of composing and reusability, component-based software development (CBSD)[11,12] brings about lots of conveniences to implement a WFMS.

Software framework (SF) is a sort of large-scaled components, which makes an important asset during the process of CBSD. As argued in Ref.[13], the advantages fetched to WFMS by SF include:

(1) Because of the “inverse control” characteristic of software frameworks, the workflow can be predefined and controlled by event sending and response;

(2) After being registered, legacy systems can be easily integrated into WFMS. They are called back using event dispatch mechanism.

Besides these two points, if a WFMS is developed on the basis of SF, the domain-specific commonalties (such as workflow definition and enactment) will be implemented inside the SF, while the variabilities are supported by hot-spots of SF. This feature makes the reuse of workflow possible.

In this paper, design and implementation of component-based geo-workflow framework (CBGWF) are in discussion. After introducing geo-workflow and component-based software framework briefly in Section 2, the architecture and design of CBGWF are presented based on domain analysis and domain design of geo-workflow in Section 3, and the design and implementation of some core components using EJB (enterprise java bean) component model are also discussed to handle domain variabilities. In the last section we reach the conclusions.

2 Geo-Workflow and CBSF

2.1 Geo-Workflow

Workflows are the computerized facilitation or the automation of a business process in whole or part. Workflow management system is a system that completely defines, manages and executes “workflows” through the execution of software whose order of execution is driven by a computer representation of the workflow logic[14]. When adopted in geographical information scientific applications, workflow evolves into geo-workflow. In general, geo-workflow is a workflow that supports geospatial process modeling and execution under the constraints of geospatial semantics. As a kind of scientific workflows, it has the common characteristics as partialness, dynamic modification, reusability, learning from mistakes, tracing and automatic documentation[15]. More especially, geospatial process is its core element, which is composed of a set of complex geospatial operators and geospatial analysis models, so geo-workflow is particularly characterized by the following statements as well:

(1) The process activity is a rather complicated computing unit with variable parameters, and some geospatial activities can be executed automatically with rare manual interaction.

(2) Geospatial data, which play important role in geo-workflow, drive the geospatial process by combining all activities. In implementation, it is represented by geospatial data type, and usually complex and massive.

(3) All elements of a geospatial process, including activities, data, parameters, process sequence, transition conditions and result data, are spatially referenced and constrained by geospatial semantics.

In order to support geo-workflow better, the applications should integrate WFMS with GIS seamlessly. A sound solution to the realization of such an application is component-based software framework (CBSF), in which geospatial operators or models are designed as a set of components and invoked by the framework itself. This measure provides an efficient way to combine WFMS and GIS.

2.2 Component-Based software framework

Software framework is a kind of reusable software products composed of both design and code[16], which offers an efficient approach to large-scaled software reuse in a specific application domain. As shown in Fig.1, a

framework-based application can be divided into two parts: framework itself and application-specific increments (ASIs). The ASIs, implementing domain variabilities, are “plugged” into the reused framework during the application development process.

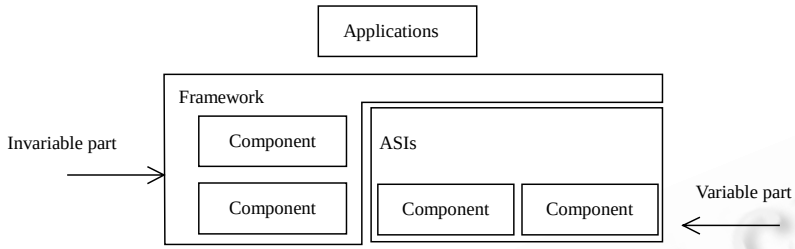


Fig.1 Software framework and domain variabilities

In the 1990s, lots of object-oriented frameworks have been developed and researched in industry and academia for various domains, including graphical user interfaces, graph editors, business applications, network servers, just to mention but a few. Recently, some component-based software frameworks supporting black-box reuse have been designed and implemented. According to the meta-model of CBSF shown in Fig.2, a component-based software framework consists of components, patterns, hot spots and constraints, in which components are basic constructing units of CBSF and they are designed to embody domain commonalties; patterns define the collaborative relationship of components; hot spots provide a mechanism to manage domain variabilities; while constraints contain some assertions or properties on the domain.

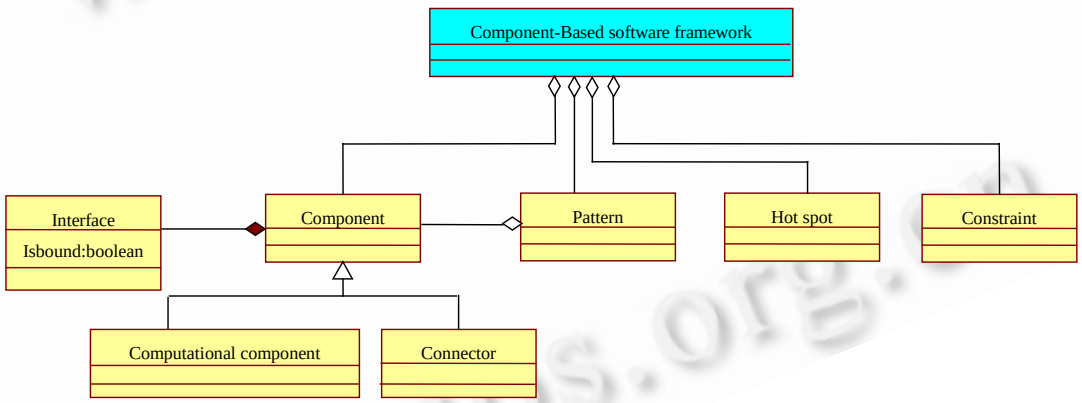


Fig.2 Meta-Model of component-based software framework[17]

Compared with other types of reusable components, including object-oriented SF, the major advantage of CBSFs is that they support domain reuse at design level and make software reuse much easier. In the context of CBSF, a component means an executable unit that conforms with a component model (such as COM/DCOM, CORBA and EJB, etc.), which is an opaque implementation of functionality in the form of binary codes. These properties guarantee the reuse manners of CBSF be black-boxed, i.e. a CBSF can be reused through interfaces invocation and composition.

The most difficult part of CBSF design is hot spot design, and the difficulty lies in the fact that the hot spots must be designed flexible enough to support all variability types and should not become the obstacle of framework evolvement. In Ref.[18], a series of hot spot types are summarized and described in pattern language to aid the CBSF design. They are socket-plug, abstract component, template, sharable data, façade, glue, script, proxy, message broker and wrapper.

3 Design of CBGWF

In order to design a CBSF, the following process should be followed. The first step is domain analysis including commonality analysis and variability analysis. The output of this phase is domain models. Based on the domain models, the second step is domain design to build the domain specific software architecture (DSSA). In the last step of framework design and implementation, since a SF is the instance of a DSSA, the basic rule of CBSF development is to realize the commonalties inside the framework while to manage variabilities through hot spots.

3.1 Domain analysis

The objective of domain analysis is to identify and model the domain variabilities. The essential of geo-workflow is to adopt workflow techniques in geographical information applications, so the domain variability mainly comes from these two aspects, i.e. workflow management and geospatial operations provided by GIS.

Common functions of geo-workflow include the following two parts:

(1) Commonalties sourcing from GIS

The process objects of each task unit are geo-referenced data, and the task units of workflow provide general geospatial functions. These functions are constrained by geospatial semantics, geospatial reference systems and data qualities. These constraints can be represented through pre-conditions, post-conditions and invariants of activities in workflow.

(2) Commonalties sourcing from workflow management

The commonalties sourcing from workflow management consist of workflow definition, user monitoring and interaction, workflow enactment engine, etc.

When variabilities (commonalty can be regarded as a special type of variability) are taken into account, many scholars have tried to classify them[19]. In general, the domain variabilities can be classified into five categories, i.e. data, functions, processes, organization and representation (user interface)[20]. According to this categorization method, the following five types of domain variabilities can be identified.

(1) Variabilities of geospatial data types

In different geo-workflow applications, variant geospatial data types are processed. Generally speaking, the data types in GIS include vector, raster, TIN (triangulated irregular network) and network, which support field model and feature model respectively. Vector data can be further classified into point, line, polygon and complex data type. Each geospatial data type fits for corresponding operations, for example, network data can be applied to nearest path search analysis.

(2) Variabilities of geospatial data management

The geospatial data can be managed in different ways. At present, there are two kinds of dominating data storing approaches, i.e. files-system-based approach and extended relational DBMS-based approach. The former is more flexible and efficient, such as ArcGIS® coverage files; while the latter is easier to implement a set of unified access interfaces, which includes Oracle® Spatial and IBM DB2® Extender, etc.

(3) Variabilities of geospatial operations

A task unit is composed by a set of geospatial operating primitives, such as buffer, overlay, interpolation, etc. They are the basic functions provided by GIS software, and can be classified into 6 categories, i.e. search operation, location analysis, terrain analysis, distribution/ neighborhood analysis, geospatial analysis and measurement operation[21]. Different workflow instances contain different sets of geospatial operations, and the geo-workflow framework should be designed to handle this type of domain variabilities.

(4) Variabilities of processes

Obviously, different geo-workflow applications have various processes. As listed in Ref.[22], 21 common

elements named workflow patterns are summarized. These patterns can be classified into 6 classes, which are basic control patterns, advanced branching and synchronization patterns, structural patterns, patterns involving multiple instances, state-based patterns and cancellation patterns. In order to handle process variabilities, the framework should be flexible enough. A reasonable solution to this goal is implementing each pattern inside the framework and composing them to form a definite workflow instance.

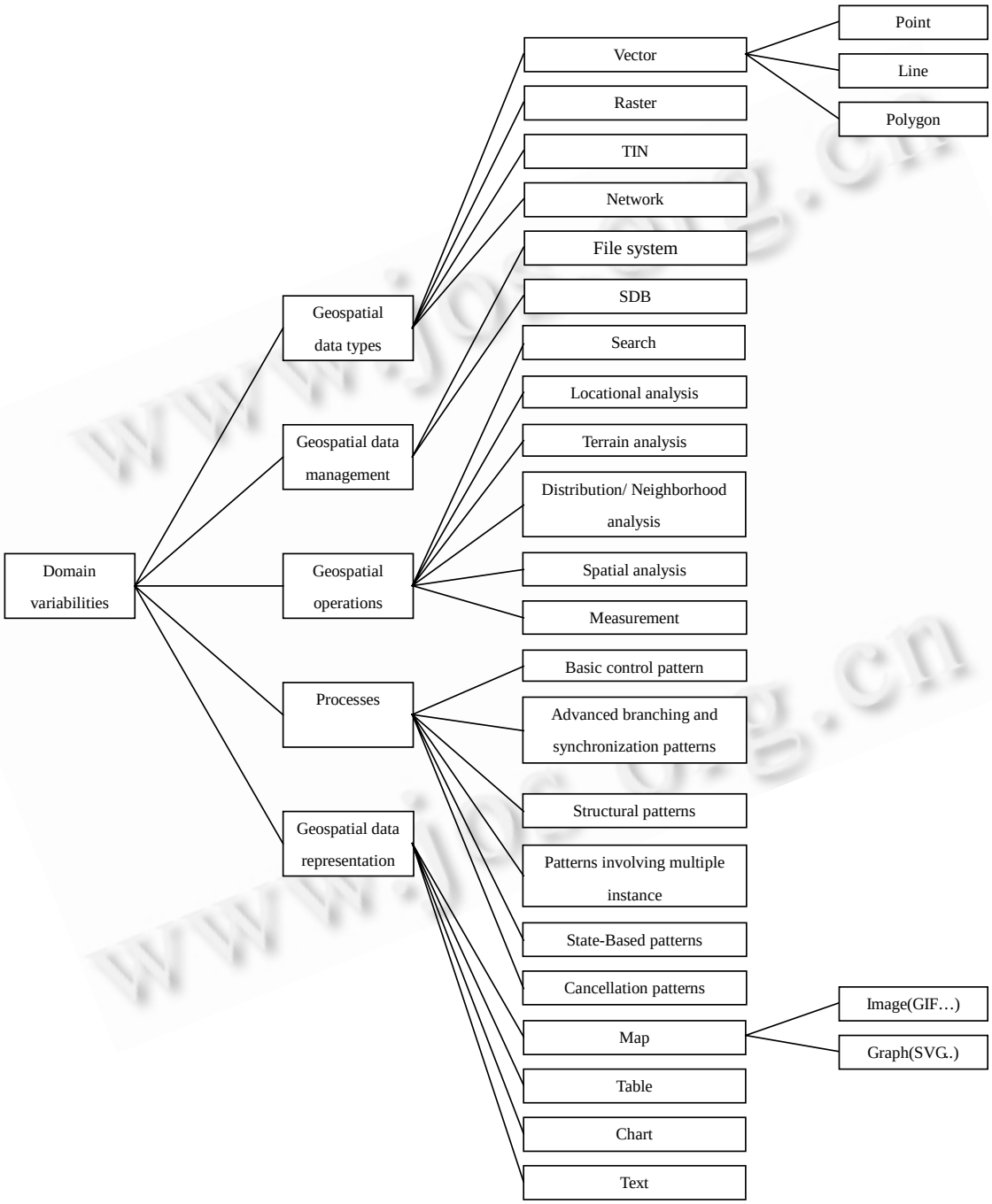


Fig.3 Tree organizations of domain variabilities

(5) Variabilities of geospatial data representation

In different geo-workflow applications, the representation of data might also be different. The representation mode of geospatial data includes image (such as GIF or JPEG files for Web applications), vector graph (such as SVG documents), chart, table and text (such as XML-based documents), etc. They are different views of the data. The framework should support corresponding functions to demonstrate geospatial data in specific representing manners.

As a summary, the above variabilities can be organized in a tree view (Fig.3). Each node in the tree represents one type of domain variabilities in geo-workflow.

3.2 Domain design and DSSA

The second step of domain engineering is domain design, the output of which is domain specific software architecture. DSSA is the reference software architecture of all applications in a given domain, which represents the common features of the application architectures. When a framework is to be developed, DSSA embodies a higher-level abstract of framework design. As a rule, a DSSA is composed of domain components and the connections between them. In order to represent domain variabilities, the components are modified with two stereotypes in DSSA,《mandatory》and《optional》. The mandatory components are domain-specific and accomplish domain commonalities, while the optional components are application-specific and support variabilities.

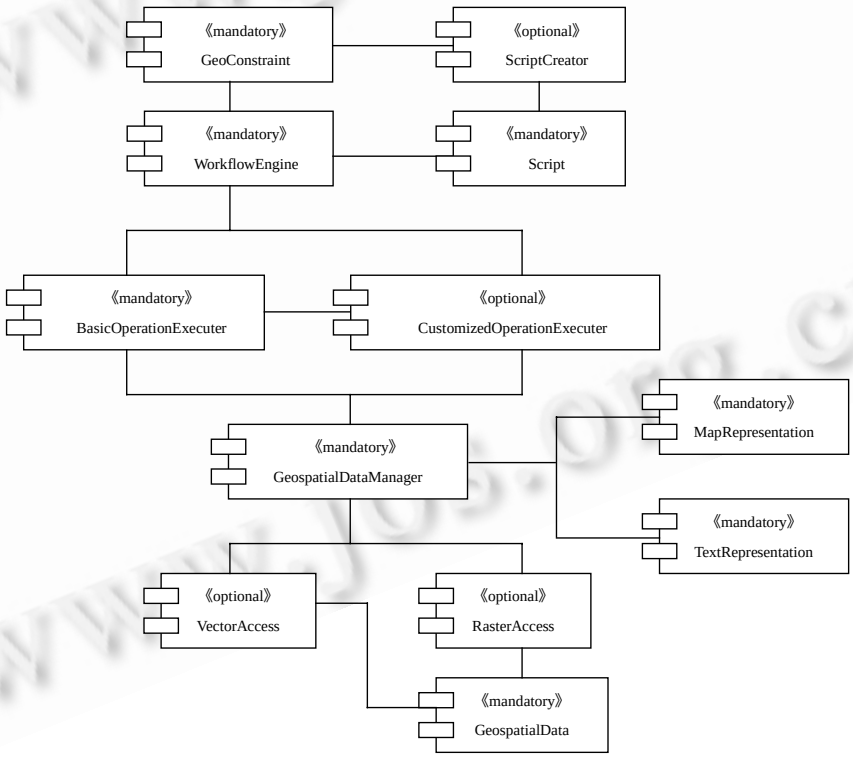


Fig.4 DSSA of geo-workflow

As a result of domain design, the DSSA's skeleton is presented in Fig.4. Obviously, the《mandatory》component of *WorkflowEngine* is the core of the whole DSSA. It schedules and runs the customized workflow. Moreover, invoked by *WorkflowEngine*, the components of *BasicOperationExecuter* and *CustomizedOperation-Executer* are corresponding to each activity unit in the workflow. The difference between them is that the former is generic while the latter is more specific to a concrete application. In order to process geospatial data, a

GeospatialDataManager component is required to manage all possible geospatial data access interfaces, no matter whether the interface is implemented inside or outside of the system. At last, for the convenience of application development, two《mandatory》components, *MapRepresentation* and *TextRepresentation* are involved in the DSSA. They are in charge of representing geospatial data in difference views to handle the variabilities of geospatial data representation.

The components in DSSA of geo-workflow with functions description and corresponding variabilities are listed in Table 1.

Table 1 Components list in DSSA of geo-workflow

Component name	Functions	Corresponding variabilities
WorkflowEngine	WorkflowEngine executes a customized workflow according with specific geo-constraints and forms a workflow instance.	N/A
Script	Script describes a workflow created by the users' workflow applications.	Processes
GeoConstraint	GeoConstraint defines the geo-constraints, and provides the access interfaces to them. The geo-constraints include a set of constraints on data quality, geo-reference and temporal features of data.	Processes
ScriptCreator	ScriptCreator helps the users to create a workflow script. Different applications may implement variant forms of components, such as a creator with GUI or a simple text editor.	N/A
BasicOperationExecuter	BasicOperationExecuter executes the general geo-operations, including buffer, overlay, interpolation, etc. as task units of workflow.	N/A
CustomizedOperationExecuter	CustomizedOperationExecuter executes the application-specific geo-operations and they must comply with the predefined interface specifications.	Geospatial operation
GeospatialDataManager	GeospatialDataManager manages the geospatial dataset manipulated in a workflow, and the dataset is organized in layers.	Geospatial data management
MapRepresentation	MapRepresentation defines interfaces to create a map view of geospatial data. Different applications may need different concrete represent components.	Geospatial data representation
TextRepresentation	TextRepresentation defines interfaces to create a text view of geospatial data.	Geospatial data representation
VectorAccess	VectorAccess provides access interfaces for vector geospatial data. Different applications would implement corresponding access component.	Geospatial data management
RastorAccess	RastorAccess provides access interfaces for raster geospatial data.	Geospatial data management
GeospatialData	GeospatialData defines the generic geospatial data types, such as point, line, polygon, etc.	Geospatial data type

3.3 Architecture design of CBGWF

The DSSA of geo-workflow in Fig.4 is a template of CBGWF's architecture. During architecture design phase, the concrete components and the relationships between them should be "frozen". The key activity in architecture design of CBSF is choosing proper patterns, including large-scaled architecture patterns and small-scaled design patterns[23], to make the architecture flexible enough to manage domain variabilities.

With further consideration of the DSSA of geo-workflow, the component of *WorkflowEngine* should be able to connect《optional》components without amount limitation, such as *CustomizedOperationExecuter* and *MapRepresentation*. The implementations of optional components are application-specific, but they should follow the predefined interfaces description. To achieve the above purpose, two patterns, i.e. HMB (hierarchical message bus) architecture pattern[24] and proxy design pattern, are employed to design architecture of CBGWF. The skeleton of CBGWF is shown in Fig.5. According to HMB pattern, the *WorkflowEngine* is designed to be a "message bus" that can send, receive and transform messages. Meanwhile, each task unit is designed as a functional component connected with the message bus.

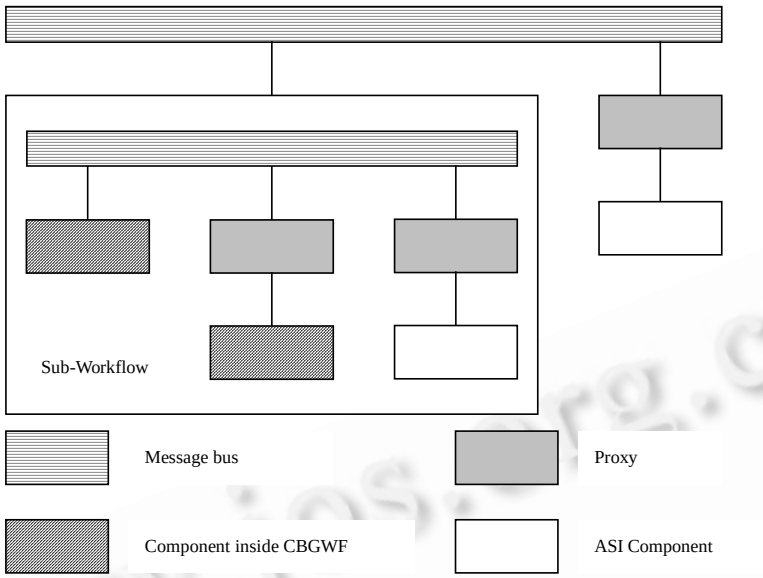


Fig.5 Architecture of CBGWF using HMP & proxy patterns

Employing the rule of event-condition-action (ECA)[25], when a component finishes its task, a specific message is sent to the message bus. After the message bus has received the message, it will process the message and send it to a relevant component to execute the next task. Through this mechanism the workflow can be performed. Because of the hierarchical characters of CBGWF's architecture, the sub-workflow can be easily supported.

In a geo-workflow application, the customized components in ASI part should be composed during run-time, so some "proxy" components are needed to bridge them with the message bus. The main functions of proxy include two aspects: (1) transmitting messages; (2) defining interfaces specification for the customized components. The proxy components in CBGWF include *VectorAccess*, *RasterAccess*, *MapRepresentation*, etc.

3.4 Components design and implementation in CBGWF

It is pointed out in Ref.[26] that *framework=components+patterns*. As described in 3.3, patterns are structure-related, and a pattern prescribes local collaborations of framework components. At the same time, framework components are usually function-related, and they implement the domain commonalties. In the following parts, the accomplishment of some core components in CBGWF is discussed with the support of EJB component platform.

(1) Workflow engine and script description

During CBGWF implementation, *WorkflowEngine* component in the form of message bus is developed using message-driven bean, which ensures the engine sending and receiving messages. The relationship between *WorkflowEngine* and each component of task unit is loose-coupled. According to ECA rule and the 6 classes of workflow patterns, the workflow script is recorded using XML document and can be interpreted by the engine. In this way, the component of *Script* can be regarded as a data component.

(2) Geospatial data access and management

In EJB component model, entity beans are designed to access structural data; however, most data are nonstructural and should be accessed as variable-length binary blocks in databases. So during the implementation phase, session bean is utilized as geospatial data access component. When the framework is instantiated, the

customized data access component should follow the interfaces specification defined by *VectorAccess* and *RasterAccess*, and the following two methods should be implemented:

Raster: Value GetValueAt ([in] int mColNumber, [in] int mRowNumber);

Vector: Collection GetFeatures ([in] String Condition);

In order to manage the geospatial data, they are referenced by URI (uniform resource identifier) strings in CBGWF, for example, the URI of plainfile://c:/data/contour.lin describes a file path of data stored in plain text. The component of *SpatialDataManager* manages a set of such URIs, so it can load the data by invoking the access functions provided by the corresponding components. For the convenience of data process, a class of *GeoLayer*, which is implemented in *GeospatialData* component, is adopted. A *GeoLayer* object encapsulates the URI of data and can access the detail of data, such as a point feature. Besides *GeoLayer*, *GeospatialData* component contains definition and implementation of all common geospatial data types, such as point, line, polygon, etc.

(3) Component of task unit

In geo-workflow framework, a session bean is responsible for the task unit. The framework instantiation requires the consistency of the component interfaces, but the number and type of parameters of each geographical activity are always variable. So during implementation, a parameter-list is used to maintain the consistency. The interface for operators of the activity component in the framework is designed as:

flagSuccess DoOperation ([in] VARIANT pParaList, [out] GeoLayer layerResult);

However, the lack of type checking is one shortcoming of this definition. What's more, VARIANT type is not supported in Java language, so it must be defined based on string. This interface specification makes each component of task unit be called by *WorkflowEngine* easily.

3.5 Reuse of CBGWF

When a CBGWF has been finished, in order to develop a geo-workflow application, we should use appropriate instantiating approaches. Considering the domain variabilities depicted in Fig.3, different instantiating approaches are listed in Table 2.

Table 2 Instantiating approaches for corresponding variabilities

Domain variabilities	Instantiating approaches
Geospatial Data Type	N/A (because all data types are defined inside the framework)
Geospatial data management	Develop new components according with *VectorAccess* or *RasterAccess*, and *GeospatialData* can be called
Spatial operation	Develop new components according with *CustomizedOperationExecuter*
Process	Customize the workflow script
Geospatial data representation	Develop new components according with *MapRepresentation* or *TextRepresentation*

3.6 A reuse case of CBGWF

Through reusing CBGWF, a geo-workflow application can be easily built. As a sample case, the objective of it is to support the decision of agriculture planning. In the process of computer-aided agriculture planning, a set of data layers with different types should be overlaid and analyzed. The data set includes soil type, elevation grade, slope and slope aspect. The last three items described the terrain of the studied area and they can be generated from DEM (digital elevation model) data in grid format. Finally, the DEM data should be created through a proper interpolation method.

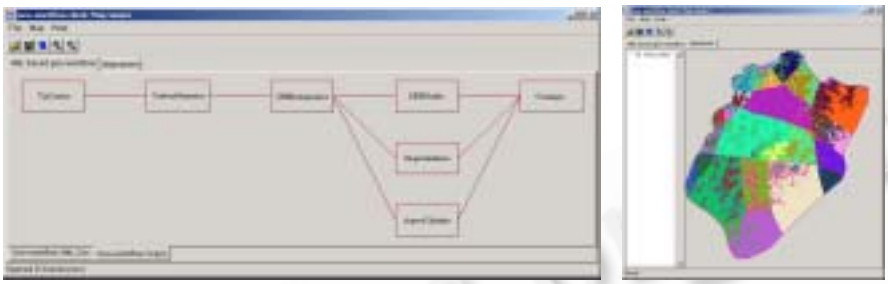
In order to build such an application, the following extensions are required:

(1) Determining the data storage approach and developing corresponding data access component;

(2) Defining the workflow using XML file;

(3) Accomplishing some necessary components to execute each workflow activity unit which is out of inclusion of CBGWF.

For this sample case, the data are managed using Oracle® spatial, so the data access components are implemented in the form of EJB component by calling OCI (Oracle call interface). At the same time, an algorithm for generating grid DEM from discrete points is programmed as a session bean. Fig.6 demonstrates the graphic representation of workflow and the analysis result. Compared with other geo-workflow systems, such as Geo-Opera mentioned in Ref.[7], the main advantage consists in its flexibility caused by the extensibility of CBGWF.



(a) (b)

Fig.6 User interface of the sample application instantiate instantiating CBGWF

4 Conclusions

Geo-workflow system has great advantages over geospatial data process. As a sort of important reusable asset, software framework can make the development of geo-workflow applications much easier. Although the CBSF support black-box reuse approach, it is hard to be developed. In this paper, focusing on geo-workflow domain, the process and methodology of component-based geo-workflow framework are discussed. The whole process includes four phases, i.e. domain analysis, domain design, framework architecture design, framework components design and implementation. During this practical process, the following conclusions can be obtained.

Firstly, the domain variabilities are key issues in requirement analysis phase. The variabilities can be identified through 5 aspects, i.e. *D* (data), *F* (functions), *P* (processes), *O* (organization) and *R* (representation). In this paper, due to the characteristics of geo-workflow, only *D*, *F*, *P* and *R* are considered.

Secondly, for the purpose of managing variabilities, a group of《optional》components would be involved in DSSA, such as *RasterAccess* in this paper.

Moreover, appropriate SA styles should be chosen to satisfy the demand of connecting《optional》components in framework design phase. As for CBGWF, HMB style is a reasonable solution.

In the long term, we will address the general process model for component-based framework, which can guide software reuse in a specific domain.

References:

[1] Worboys WF. GIS: A Computing Perspective. London/Bristol: Taylor & Francis, 1995.

[2] Stefanakis E, Vazirgiannis M, Sellis T. Incorporating fuzzy logic methodologies into GIS operations. In: Proc. of the Int'l Conf. on Geographical Information Systems in Urban, Regional and Environmental Planning, 1996. 61−68.

[3] Bennet DA. A framework for the integration of geographical information systems and model base management. Int'l Journal of Geographical Information Science, 1997,11(4):337−357.

[4] Lilburne L. The integration challenge. In: Pascoe RT, Sutherland NC, Gorman P, eds. Proc. of the 8th Annual Colloquium of the Spatial Information Research Centre. Dunedin: Spatial Information Research Center, 1996. 85−94.

[5] Ungerer MJ, Goodchild MF. Integrating spatial data analysis and GIS: A new implementation using the component object model (COM). Inte'l Journal of Geographical Information Science, 2002,16(1):41−53.

[6] Alonso G, Abbadi A. Cooperative modeling in applied geographic research. Int'l Journal of Intelligent and Cooperative Information Systems, 1994,3(1):83−102.

[7] Alonso G, Hagen G. Geo-Opera: Workflow concepts for spatial processes. In: Scholl M, Voisard A, eds. Advances in Spatial Databases (Proc. of the SSD'97). Berlin: Springer-Verlag, 1997. 238−257.

[8] Marr AJ, Pascoe RT, Benwell GL, Mann S. Development of a generic system for modeling spatial processes. Computer, Environmental and Urban Systems, 1998,1(22):57−69.

[9] Weske M, Vossen G, Mederos CB. Workflow management in geoprocessing applications. In: Laurini R, Makki K, Pissinou N, eds. Proc. of the 6th Int'l Symp. on Advances in Geographic Information Systems (ACMGIS'98). Washington DC: ACM Press, 1998. 88−93.

[10] Seffino LA, Medeiros CB, Rocha JV, Yi B. WOODSS −− a spatial decision support system based on workflows. Decision Support Systems, 1999(27):105−123.

[11] Brown AW, Wallnau KC. The current state of CBSE. IEEE Software, 1998,15(5): 37−46.

[12] Aoyama M. New age of software development: How component-based software engineering changes the way of software development? In: Proc. of the 1998 Int'l CBSE. Kyoto, 1998. 25−26.

[13] Kappel G, Rausch-Schott S, Reich S, Retschitzegger W. Hypermedia document and workflow management based on active object-oriented databases. In: Nunamaker JF, Sprague RH, eds. Proc. of the 30th Hawaii Int'l Conf. on System Science IEEE, Hawaii: Maui, 1997. 377−386.

[14] Workflow Management Coalition. The workflow reference model. WFMC-TC-1003. Workflow Management Coalition, 1995.

[15] Meidanis J, Vossen G, Weske M. Using workflow management in DNA sequencing. In: Proc. of the 1st Int'l Conf. on Cooperative Information Systems. Brussels, Washington DC: IEEE Computer Society Press, 1996. 114−123.

[16] Bosch J. Design patterns & frameworks: On the issue of language support. In: Proc. of the LSDF'97. London: Springer-Verlag, 1997. 133−136.

[17] Liu Y, Zhang SK, Wang LF, Yang FQ. Component-Based software frameworks and role extension form. Journal of Software, 2003, 14(8):1364−1470 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/14/1364.htm

[18] Liu Y. Research on component-based software framework: JBSF, and related techniques [Ph.D. Thesis]. Beijing: Peking University, 2003 (in Chinese with English abstract).

[19] Svahnberg M, Bosch J. Issues concerning variability in software product lines. In: Linden F, ed. Proc. of the 3rd Int'l Workshop on Software Architectures for Product Families. Berlin: Springer-Verlag, 2000. 146−157.

[20] Zhang SK, Hu WH, Chen ZL, Wang LF. Study on classification and control mechanism of domain variability. Acta Electronica Sinica, 2004,32(3):446−451 (in Chinese with English abstract).

[21] Marr AJ, Pascoe RT, Benwell GL. Interoperable GIS and spatial process modeling. In: Proc. of the 2nd Int'l Conf. on GeoComputation. University of Otago New Zealand, 1997.

[22] van der Aalst W, Barros AP, ter Hofstede A, Kiepuszewski B. Advanced workflow patterns. In: Etzion O, Scheuermann P, eds. Proc. of the 7th Int'l Conf. on Cooperative Information Systems. London: Springer-Verlag, 2000. 18−29.

[23] Buschmann F, Meumier R, Rohnert H, Sommerlad P, Stal M. Pattern-Oriented software architecture −− a system of patterns. John Wiley and Sons, 1996.

[24] Zhang SK, Wang LF, Yang FQ. Hierarchical message bus based software architecture style. Science in China (Series F), 2002, 45(2):111−120.

[25] Bae J, Bae H, Kang SH, Kim Y. Automatic control of workflow processes using ECA rules. IEEE Trans. on Knowledge and Data Engineering, 2004,16(8):1010−1023.

[26] Johnson R. Frameworks = Patterns + Components. Communications of the ACM, 1997,40(10):39−42.

附中文参考文献:

[17] 刘瑜,张世琨,王立福,杨芙清,基于构件的软件框架与角色扩展形态研究.软件学报,2003,14(8):1364−1370. http://www.jos.org.cn/1000-9825/14/1364.htm

[18] 刘瑜.基于构件的软件框架 JBSF 及相关技术研究[博士学位论文].北京:北京大学,2003.

[20] 张世琨,胡文蕙,陈兆良,王立福.领域变化性分类与控制机制研究.电子学报,2004,32(3):446−451.