Environ Earth Sci (2015) 74:7101–7115
DOI 10.1007/s12665-015-4737-4

THEMATIC ISSUE

Describing data formats of geographical models

Di Hu[1,2,3] · Shaosong Ma[4] · Fei Guo[1,2] · Guonian Lu[1,3] · Junzhi Liu[1,2,3]

Received: 16 March 2015 / Accepted: 17 August 2015 / Published online: 2 September 2015

Abstract The variations that characterise the design and implementation of geographical models create significant difficulties in the sharing of models. One such difficulty concerns the fact that geographical models and data are highly coupled, and various heterogeneous data formats exist. This results in a laborious and unending data format conversion process during model sharing. In this paper, data format is defined to include three components: data type, structure and layout. A new method is proposed to clearly describe data formats of geographical models in a universal manner in an attempt to solve the problem identified above. First, the characteristics of geographical models, data and their formats were analysed. A concept model of data formats was then proposed, focusing on data location, data types and separators. A novel data format description language called the Data Format Markup Language (DFML) was designed accordingly. DFML uses XML markup elements to describe format information. The primary markup elements and syntax rules of DFML are introduced, and the use of DFML to describe data formats is explained step by step. Finally, case studies were performed to test this approach with two types of data formats: (1) data formats of a watershed model: Soil and Water Assessment Tool (SWAT), and (2) typical GIS data formats, namely, the ESRI ASCII Raster and shapefile formats. The results of these case studies demonstrated that DFML is easy to understand and use and can adequately describe data formats of geographical models.

Keywords Geographical model · Web services · Model sharing · Data format · Format description

✉ Di Hu
hud316@njnu.edu.cn

[1] Key Laboratory of Virtual Geographic Environment (Nanjing Normal University), Ministry of Education, Nanjing 210023, China

[2] Jiangsu Center for Collaborative Innovation in Geographical Information Resource Development and Application, Nanjing 210023, China

[3] State Key Laboratory Cultivation Base of Geographical Environment Evolution (Jiangsu Province), Nanjing 210023, China

[4] Water Resource and Hydropower Corp, NARI Group Corp, Nanjing 211106, China

Introduction

Geographical models are fundamental to modern geography (Lu 2011). They describe important elements of geographic entities and phenomena in the real world and their relationships (Crosier et al. 2003). Geographical models encapsulate a wealth of practical and theoretical scientific knowledge in an easy-to-use form (Yue 2001; Goodchild 2005; Voinov and Shugart 2013). Sharing geographical models can enable the full use of numerous models distributed in cyberspace to reduce modeling costs and time on the creation of new models (Lu 2011; Lin et al. 2013a). Moreover, the sharing of geographical models is of crucial importance for improving the spatial analysis ability of GIS (Goodchild 2005; Torrens 2009; Granell et al. 2010) and is a key issue in the modeling and simulation component of Virtual Geographic Environments (VGEs) (Lin et al. 2013b).

Geographers have long investigated the sharing of geographical models. Service-oriented systems have become a preferred practice for the sharing of geographical models compared to the isolated and monolithic modeling systems (Zhang et al. 2007; Ames et al. 2012; Nativi et al.

2013). Service-oriented frameworks and architectures have been investigated with the concept of model service and model sharing (Papajorgji et al. 2004; Yu et al. 2006; Wen et al. 2006; Tang et al. 2008). More recently, approaches have been proposed for open sharing of geographical models compliant with Web Processing Service (WPS) standard (Feng et al. 2011; Wu et al. 2012; Dubois et al. 2013; Castronova et al. 2013). Model Web, first defined by Geller and Turner (2007), is an open-ended system of interoperable geographical models and databases, with machine and end-user internet access via web services. Nativi et al. (2013) introduced the Group on Earth Observation (GEO) Model Web initiative and discussed the basic principles and technical challenges of implementing a Model Web. By employing state of the art of cloud computing, Li et al. (2014) presented methodologies for designing and implementing Model as a Service (MaaS).

The heterogeneity of data formats is an important issue in the sharing of geographical models (Crosier et al. 2003; Feng et al. 2009, 2011; Li et al. 2014) because geographical models and data are highly coupled (Voinov and Cerco 2010; Granell et al. 2010). Users must prepare data strictly according to the data format requirements of geographical models (Feng et al. 2009; Yue et al. 2015). Model services designed and implemented above use eXtensible Markup Language (XML) and Geography Markup Language (GML), which is also an XML language, as required data formats. Therefore, when using these web services to share models, data format conversion is a necessary process. Model service developers should write programmes to convert the native formats of geographical models to the XML formats required by model services. The model service end-users must convert native formats to XML formats. In addition, data preparation tools are designed to support native data formats, not XML formats. However, because native data formats of geographical models are heterogeneous and numerous, the data format conversion process would be laborious and unavailing. Managing these heterogeneous data formats remains a challenge for designing a widespread model service (Zhu et al. 2010; Goodall et al. 2011; Shi 2015).

Although it is impossible to develop a conversion method for all data formats of geographical models, it is plausible to describe all data formats in a universal approach. The Data Format Description Language (DFDL), proposed by the Open Grid Forum, is a beneficial attempt to develop an extensible standard for describing data formats (Alan et al. 2011). However, DFDL does not directly describe data format information, and the data format information described by DFDL is attached to data items. Furthermore, DFDL is derived from the computer domain. Although DFDL is somewhat technical and complex, it is suitable for use by computer experts that are not modelers.

The aim of this paper was to design a data format description method to solve the heterogeneous data format problem in the geographical model sharing. The paper first analyses the characteristics of geographical model data and their formats. Compared to DFDL and other languages, which are essentially data-oriented, a description model of data formats is designed based on data location, data types and separators. Then, the markup elements and usage of the new data format description language, Data Format Markup Language (DFML), are discussed in detail. Finally, the method is demonstrated with two case studies of Soil and Water Assessment Tool (SWAT) input/output data formats and the ESRI ASCII Raster and shapefile formats. Although the focus of this paper is to describe the data formats of geographical models, the concept and approach are general enough to apply to other disciplines and fields.

Basis for data format modeling

Geographical model, data and format

Geographical models are created based on long-term studies in various sub-disciplines and fields of geography (Goodchild 2005). Each geographical model is a processing unit producing outputs with specific inputs. The handling of model input and output (I/O) can be very complex. It is an error-prone and time-intensive step (Bhatt et al. 2014) because the input types and formats are heterogeneous for different models (Li et al. 2014; Lu 2011). Generally, model publishers provide documentation to describe the input/output formats. To use a model effectively, the input/output documentation as well as the model software (code) and data are needed (Fig. 1).

The data, which are fundamental for running a model (Argent 2004), can originate from different disciplines and fields and usually include various contents and indicators with complex and heterogeneous formats (Denzer 2005; Granell et al. 2007; Kunkel et al. 2013). Specifically, first, geographical model data are gathered from all sub-disciplines of geography and other non-geography disciplines. Second, in terms of the data content, model data often

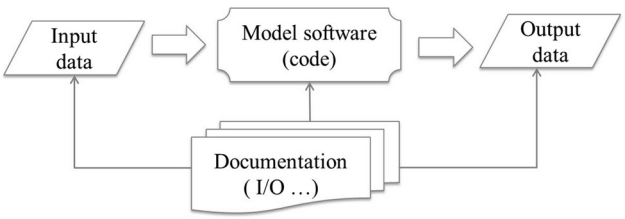

Fig. 1 Components of a geographical model

consist of data specification, data body and supplementary specifications. Data specification is not an essential component in all model data. Third, data indicators are derived from one or many sub-disciplines of geography, e.g. temperature and humidity in meteorology and agrotype in soil geography. Fourth, data are structured in specific manners, including table structure and dendritic structure. Finally, data are presented in various formats, including standard formats and user-defined formats. Model users are advised to prepare data strictly according to the format requirements of the model to ensure that the model runs correctly (Li et al. 2010).

Characteristics of data formats of geographical models

The data of geographical models originate from various disciplines, and the content of the data may be recorded in standard formats of a discipline or in user-defined formats. Therefore, various data formats can be found in geographical models. Files are the primary representation of data. Traditionally, the TXT file format is thought to be a data format, as are the Excel file, NetCDF file, ESRI ASCII Raster file and shapefile formats. Data formats in this sense only describe a general framework of a data format without specific details. Since the understanding of data format is unclear and inconformity, we propose that a data format consists of three components: data type, structure and layout. The data type component indicates data types of data items. The structure component indicates the number of parts in the data and the relationships among the parts. The layout component indicates the location of a data item and the representation of a specific data type (e.g. the real data type with two decimal places).

In terms of TXT, Word and Excel files, the data type, structure and layout of their contents can be significantly different. Each of these file types is not a data format. The structure of a NetCDF file is described by a CDL file. Different structures result in NetCDF files in different data formats. According to the geometric data expressed by the file, a shapefile can be categorised as a point shapefile, polyline shapefile and polygon shapefile. These file types are different in terms of data type, structure and layout. Thus, they are different data formats. The ESRI ASCII Raster file clearly defines the data meaning, types and separators in each line. Therefore, it can be considered as a data format.

The formats used for geographical model data are diversified and complicated and include various characteristics related to geographic data.

1. The types of data formats, such as Word file, Excel file, shapefile, Database, text file and binary file, are diverse.
2. Data types are simple and are primarily integer or real values. Quantitative data in geographical models are usually represented by real values, whereas qualitative data are described by integer values and characters. Three data types of data values are used for geographic data: integer, real and character. Time-series data use specific character combinations or integer values to represent date and time information.
3. Data structures are diverse. Data of geographical models can usually be represented using files, including text file and binary file. Text files are usually organised by lines, whereas binary files are organised by bytes. Generally, the content of a file consists of data specification and data body. The former is placed in the file header and contains basic data characteristics, such as the column name, line/column count and unit of measurement. The latter is the actual data used for model calculations and is usually placed after the data specification.
4. Data layouts are diverse, and data items are separated by separators. For data in a text file format, certain characters, such as blank spaces, tabs and commas, are used to separate data items. Each of these characters is referred to as a "separator" in this paper. The combination of data items and separators determines the data layouts.

Conceptual modeling of data formats

Data format modeling objectives and principles

Several data format description languages, such as DFDL, have been developed for different purposes. Because the basis of DFDL is not a format, users should first construct the schema of the data before describing the data format. As a result, DFDL is complex and difficult for modelers to use. Therefore, it is important to be mindful of the objectives of the language, to clearly describe the data formats of geographical models and address the needs of users.

Various data format modeling principles are listed below:

1. *Limited objectives* The format model attempts to support the description of data formats of geographical models in a universal manner, especially frequently used text file formats that are not standard or well-known formats. There is likely no need to describe the standard and well-known formats, which are familiar to the users.
2. *Direct description of the data format* The format model should be independent and not attached to the data.

Entities and their attributes should directly describe the data format information.

3. *Simple but unabridged* A complex format model will heavily degrade user acceptability. The entity's quantity and structure of the format model should be limited. Unabridged means that the format model is able to express most aspects of data formats and provide limited scalability for certain special situations.
4. *Human and computer readable* The description results will be used by both humans and computers. Users could directly use the format model to describe the data formats of their geographical models, and the results should be able to be easily read by computers to facilitate automatic processing.
5. *Easy to understand and easy to use* Most users are not proficient in computer technology, and even more, they have no desire to learn a new technology. It is important that users who have little computer technology experience can easily and quickly use the data format model.

Concept model of data formats

In contrast to the use of existing data format description languages, such as DFDL, a method for directly, clearly and completely describing data formats remains in the exploratory stage. It is essential to design a model that can adequately abstract data format information. The model should focus on the initial requirement, the requirement of model sharing and the special requirements of geographical modeling, including technological and non-technological factors. As stated above, a data format consists of the data type, structure and layout. The location of a data item is of great importance in a data format, and the separator is as important as the data item. The dislocation of a single character or a byte can lead to a misunderstanding of the entire dataset. This study designed a new data format description model centred on data location and oriented for data types and separators (Fig. 2).

The model consists of five types of entities: DataFormat entity, Location entity, DataType entity, Separator entity and Group entity. These entities are abstracted from the analysis of input/output documentations of geographical models. The entities represent the concepts recognised by the users when they describe a data format with text and tables.

The DataFormat entity is designed to represent the basic information of a data format. The DataType entity is a class of entities, including itself and its subtypes, and is designed to describe data type information. The Separator entity is also a class of entities and is designed to describe separator information. The Location entity describes the location information of data. The Group entity describes the structure information of the data. The DataType, Separator and Group entities are associated with the Location entity, and the combination of these forms the DataFormat entity. The Group entity is able to nest and invoke other Group entities, and the DataFormat entity can invoke other DataFormat entities.

The DataType entity consists of eight entity subtypes: String entity, Integer entity, Real entity, Boolean entity,

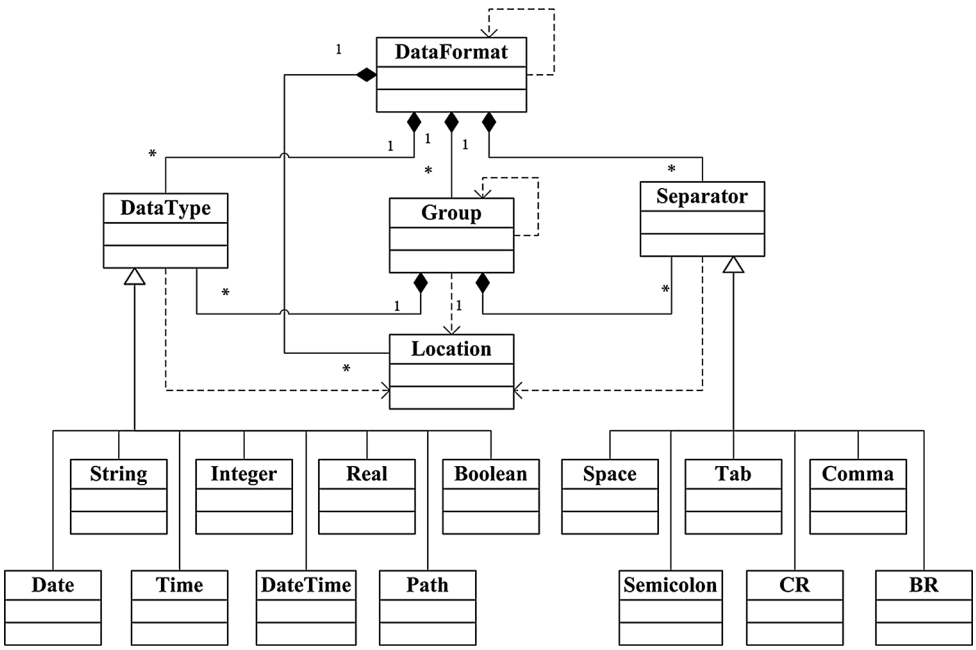

Fig. 2 The concept model of data formats

Date entity, Time entity, DateTime entity and Path entity. Specifically, the String entity describes the character string type, the Integer entity represents the integer type, the Real entity indicates the real type, the Boolean entity describes the Boolean type, the Date entity represents the date type, the Time entity describes the time type, the DateTime entity indicates the date and time type and the Path entity indicates the path type. The Separator entity has six entity subtypes: Space entity, Tab entity, Comma entity, Semicolon entity, CR entity and BR entity. Specifically, the Space entity indicates the space separator, the Tab entity indicates the tab separator, the Comma entity describes the comma separator, the Semicolon entity describes the semicolon separator, the CR entity indicates the line break separator and the BR entity indicates the blank line separator.

Data format description language: DFML

DFML elements

In this study, DFML was designed to describe data formats in a universal manner. DFML is a simple language based on XML, using markup elements to describe data format information. According to the data format concept model, a total of six types of elements are designed in DFML: root element (dataformat), import element (import), location element (location), data type element, separator element and group element.

Specifically, the root element is designed to describe basic information of a data format with several primary attributes, such as name, namespace, version and mode. The import element can be used to import elements defined in other data format markup documents. In this way, the subject document will be able to invoke elements defined in other documents. The import element has a required attribute, i.e. a link. The location element is designed to describe the location of a data item or a range of data locations with two primary attributes: name and value. The data type element primarily consists of a comprehensive data type element (datatype) and several basic data type elements (string, integer, real, boolean, date, time, datetime and path) with several primary attributes, such as name, type, value, format, default, domain, number, separator and location. The attributes of type and value only apply to comprehensive data type element. The comprehensive data type element is designed to describe the type and format information for any data item and provides essential extendibility for data types. The basic data type elements can be used to describe the type and format information of data items of specific types. The separator element can also be divided into comprehensive separator element (separator) and basic separator elements (space, tab, comma, semicolon, cr and br) with primary attributes, such as name, type, value, number and location. The attributes of type and value only apply to comprehensive separator element. The comprehensive separator element is designed to describe the separator type and format information of any data items, whereas the basic separator element can be used to describe the separator type and format information of data items of specific types. The group element is used to combine several elements into a group, and in this way, the data structure information can be represented with three primary attributes: name, location and number. In many cases, an additional introduction is recommended so that users can better understand the data format as well as the data item. To obtain this information, each element has a description attribute. The key attributes of DFML elements are illustrated in Table 1.

DFML is based on XML and follows the syntax rules of XML. To precisely describe data formats, DFML should follow several other essential syntax rules:

1. The mode attribute of the dataformat element has the value range {"char", "byte"}. mode = "char" indicates the character mode, and the locations of data items are determined using characters. This is used to describe the formats of text files. mode = "byte" indicates the byte mode, and the locations of data items are determined using bytes. This is used to describe the formats of binary files.
2. The import element should be the direct sub-element of the dataformat element. In this way, humans and computers can easy to identify the elements that are not defined in this document, and the readability of DFML documents is enhanced. The value of the link attribute can be a local location or network location.
3. The value attribute of the location element must be determined in accordance with the mode attribute in the dataformat element. When the value of the mode attribute is "char", the value of the location element's value attribute will be the character positions of the starting row, starting column, ending row and ending column. When the value of the mode attribute is "byte", the value of the location element's value attribute will be the positions of the starting byte and ending byte. Particularly, the number "−1" that appears in the value attribute of the location element indicates the last line, column or byte.
4. The group element is allowed to nest group elements, data type elements and separator elements. The outermost group element must be the direct sub-element of the dataformat element. When describing a data format, the data format is divided into several parts, and each part is then described using group

Table 1 Description of the key attributes of DFML elements

Element	Attribute	Description
Dataformat	Name	The name of the described data format
	Mode	Represents text file format and binary file format using character mode and byte mode
Import	Link	The location of the imported data format markup document
Location	Name	The name of the location
		A named location can be quoted in the context
	Value	The value of the location's starting position and ending position
Datatype	Name	The name of a new data type, only used for expandability
		The named data type can be quoted in the context
	Type	The specific data type of the datatype element
		The value range includes the 8 basic data types and other types for expandability
	Format	The presentation format of a special data type
	Number	The number of repeating datatype elements
	Location	The location of a data type element
Separator	Name	The name of the separator element
		A named separator element can be quoted in the context
	Type	The specific separator of the separator element. The value range includes the 6 basic separator types and other separators for expandability
	Number	The number of repeating separator elements
	Location	The location of a separator element
Group	Name	The name of the group element
		A named group element can be quoted in the context
	Location	The location of the group element
	Number	The number of repeating group elements

elements. If there are no clear boundaries between different parts, the formats of text files will generally be subdivided by rows, and the formats of binary files will be subdivided by bytes. Alternatively, without subdividing, the data formats can also be described by data type and separator elements. However, there will be no hierarchical structure in the generated data format markup document.

Using DFML

DFML is designed to clearly describe data formats of geographical models to enable model sharing in a universal manner. Model providers, including the modelers and the model encapsulators, also called model service developers, who convert geographical models into web services, can use DFML to describe the data formats of their models. Modelers can use DFML to describe the data formats of their existing models and new models. Although various data formats are used by geographical models, model users can quickly understand DFML because all data formats are described in the same manner. Model encapsulators can use DFML to describe the data formats of models, generate DFML documents and embed DFML documents into Web Services Description Language (WSDL) documents. In this way, model service users will obtain the data format information from the DFML document and prepare the data for the model service. Therefore, model encapsulators do not need to perform data format conversion between native data formats of geographical models and XML formats.

When using DFML to describe the data format of a geographical model, data format information should first be obtained from the data format specifications or the data. DFML is then used to describe the data format information. Finally, a data format markup document can be generated. The workflow for describing a data format is presented below (Fig. 3).

1. Create a new data format markup document.
2. Describe the basic information of the data format. First, add a root element (dataformat) and then set the attributes of the name, namespace and mode to describe the name, identifier and mode information of the data format, respectively.
3. Divide the data format into several parts according to the structure information and determine the name and location range of each part.
4. Describe the basic information of each part. First, use group elements to describe the framework of each part and then set the name and location attributes of the

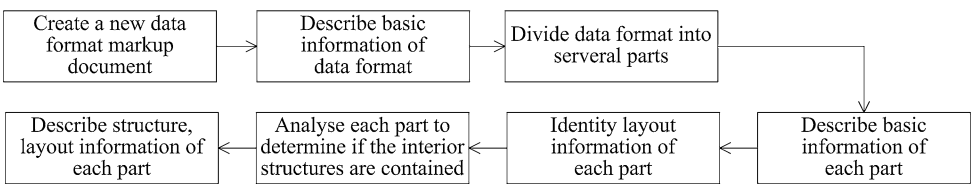

Fig. 3 DFML data format description workflow

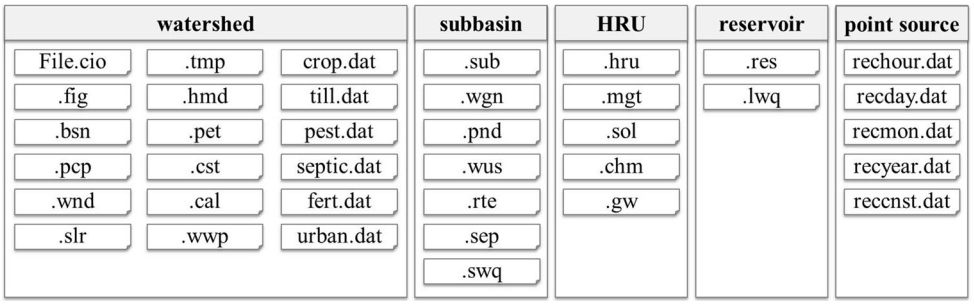

Fig. 4 Types of SWAT input files

Table 2 Brief description of the variables in the temperature file

Variable name	Definition	Optional/ required
TITLE	The first line of the temperature file is reserved for comments	Optional
LATITUDE	Latitude of temperature recording gauge location	Optional
LONGITUDE	Longitude of temperature recording gauge location	Optional
ELEVATION	Elevation of temperature recording gauge (m)	Optional
YEAR	Year (4-digit)	Required
DATE	Julian date	Required
MAX TEMP	Daily maximum temperature (°C)	Required
MIN TEMP	Daily minimum temperature (°C)	Required

group elements to describe the name and location range of each part, respectively.

5. Identify the layout information of each part. Identify the data type, separator type, start location, end location and presentation format of the data type of each data item.
6. Analyse each part to determine if the interior structures are contained. Analyse the layout information of each part to determine if regular repetitive structures are contained. If so, categorise the structures as subparts of the part; these are known as the interior structures of the part.
7. Describe the structure and layout information of each part. According to the structure and layout information obtained in steps 5 and 6, describe data items successively using group, data type and separator elements. The representation format of the data type is described by the format attribute of the data type element. When a part contains an interior structure, describe it using the group element nested in the superior group element.

Case studies

Case 1: SWAT input/output data formats

SWAT has emerged as one of the most widely used watershed scale water quality models and has been extensively applied to a broad range of hydrologic and environmental problems (Gassman et al. 2014). SWAT is a comprehensive model that requires a diverse amount of inputs. The input files of SWAT can be divided into five categories, with a total of 37 types of files (Fig. 4). The variety and number of input files may overwhelm novice users; thus, SWAT input/output data formats are suitable for illustrating the descriptive ability of DFML.

In the SWAT IO documentation, the authors explain each type of file using two tables, which is clearer than using text. The first table provides a brief description of variables, and the second table provides the format of the variables listed in the first table (Arnold et al. 2012). The temperature input data of SWAT were chosen as an example to illustrate the application of DFML. The other types of input data can be described by DFML in a similar way. The files used to store the temperature data are summarised in Tables 2 and 3.

Clearly, this data format can be divided into two parts. The data specification part begins at line 1 and ends at line 4, and the data body part begins at line 5 and continues to the end of the file. The first line is reserved for comments, and the next 3 lines are the latitude, longitude and elevation of the temperature recording gauge. Line 5 through the end of file lists the daily maximum and minimum temperatures as one record for each line.

The data format markup document of this data format is created as described below.

```
<dataformat name="Data format of temperature file for SWAT"
 namespace="com.vge.swat.input" mode="char">
    <group location="1 1, 4 -1">
        <string location="1 1, 1 -1" description="TITLE"></string> <cr></cr>
        <real location="2 8, 2 17" description="LATITUDE"></real> <cr></cr>
        <real location="3 8, 3 17" description="LONGITUDE"></real> <cr></cr>
        <integer location="4 8, 4 17" description="ELEVATION"
format =”i10”></integer ><cr></cr>
    </group>
    <group location="5 1, -1 -1">
        <group number="unknown">
            <integer location=”0 1, 0 4” description=”YEAR” format =”i4”></integer>
            <integer location=”0 5, 0 7” description =”DATE” format = “i3”></integer>
            <real location=”0 8, 0 12” description =”MAX TEMP” format=”f5.1”></real>
            <real location=”0 13, 0 17” description =”MIN TEMP” format=”f5.1”></real>
            <cr></cr>
        </group>
    </group>
</dataformat>
```

In this example, the basic information of the data format is described using the dataformat element and its attributes. The value of the name attribute is “Data format of temperature file for SWAT”, and the value of the mode attribute is “char”. Two group elements are used to describe the two parts. The value of the location attribute of the first group element is set as “1 1, 4 -1”. For the second group element, the value of the location attribute is set as “5 1, -1 -1”. Within this part, each line has the same layout, with the same data type and arrangement. Therefore, the structure of this part is repetitive. Specifically, this part has an interior structure. This structure is described by a nested group element, inside which records are described successively with two integer and two real data type elements.

Case 2: ESRI ASCII Raster and shapefile formats

Geographical information systems (GISs) have been used for data processing and post-simulation data analysis and visualisation to support geographical models (Goodchild 2005). ESRI ASCII Raster and shapefile formats are the two most commonly used GIS data formats and are widely used in geographical models as input/output data formats.

Table 3 The data format of the temperature file with one record

Variable name	Line #	Position	Format	F90 Format
TITLE	1	Unrestricted	Character	Unrestricted
LATITUDE	2	Space 8–17	Free	
LONGITUDE	3	Space 8–17	Free	
ELEVATION	4	Space 8–17	Integer	i10
YEAR	5-END	Space 1–4	Integer	i4
DATE	5-END	Space 5–7	Integer	i3
MAX TEMP	5-END	Space 8–12	Decimal (xxx.x)	f5.1
MIN TEMP	5-END	Space 13–17	Decimal (xxx.x)	f5.1

Fig. 5 ESRI ASCII Raster format

ncols	ncol				/* number of columns in the grid*/
nrows	nrow				/* number of rows in the grid*/
xllcorner	X				/* lower left x coordinate of grid*/
yllcorner	Y				/* lower bottom y coordinate of grid*/
cellsize	Size				/* grid cell size */
NODATA_VALUE			NODATA		/* value of an empty grid cell */
Z_{11}	Z_{12}	Z_{13}	…	Z_{1ncols}	/* values of row 1 */
Z_{21}	Z_{22}	Z_{23}	…	Z_{2ncols}	/* values of row 1 */
.					
.					
.					
Z_{nrow1}	Z_{nrwos2}	Z_{nrows3}	……	$Z_{nrowsncols}$	/* values of last row */

The ASCII Raster file is the most important raster data format, and shapefile is the most important vector data format. If DFML can be used to describe these two types of formats, then it can be used to describe other GIS data formats.

The ESRI ASCII Raster format file begins with header information that defines the properties of the raster, such as the cell size, the number of rows and columns and the coordinates of the origin of the raster. The header information is followed by cell value information specified in space-delimited row-major order, with each row separated by a carriage return (ESRI 2009). The specification of this format is shown in (Fig. 5).

This data format also contains two parts. Lines 1–6 comprise the file header, and each line consists of a string and an integer or real data item. The file body comprises line 7 to the end of file, and each line consists of several real-type data items. Data items are separated by one or more spaces. The ESRI ASCII Raster format is described below using DFML.

```
<dataformat name="ESRI ASCII Raster Format" namespace="com.vge.esri" mode="char">
    <group description="file header">
        <group location="1 1, 1 -1">
            <string value="ncols" description="Number of columns in the grid"></string>
            <space location="1 0, 1 14"></space>
            <integer></integer> <cr> </cr>
        </group>
        <group location="2 1, 2 -1">
            <string value="nrows" description="Number of rows in the grid"></string>
            <space location="2 0, 2 14"></space>
            <integer></integer> <cr></cr>
        </group>
        <group location="3 1, 3 -1">
            <string value="xllcorner" description="Lower left x coordinate of
grid"></string>
            <space location="3 0, 3 14"></space>
            <integer></integer> <cr></cr>
        </group>
        <group location="4 1, 4 -1">
            <string value="yllcorner" description=" Lower bottom y coordinate of grid
"></string>
            <space location="4 0, 4 14"></space>
            <integer></integer> <cr></cr>
        </group>
        <group location="5 1, 5 -1">
            <string value="cellsize" description="Grid cell size"></string>
            <space location="5 0, 5 14"></space>
            <integer></integer> <cr></cr>
        </group>
        <group location="6 1, 6 -1">
            <string value="NODATA_value" description="Value of an empty grid
```

```
cell"></string>
                <space location="6 0, 6 14"></space>
                <integer></integer> <cr></cr>
            </group>
        </group>
        <group location="7 1, -1 -1" description="Values of grid">
            <group number="dataformat/group@location='2 1, 2 -1'/interger@value">
                <real number="dataformat/group@location='1 1, 1 -1'/interger@value"
separator=","></real>
                <cr></cr>
            </group>
        </group>
</dataformat >
```

A shapefile consists of three files, i.e. a main file, an index file and a dBASE table. The main file is used as an example in this case. The main file (.shp) contains a fixed-length file header followed by variable-length records. Each variable-length record consists of a fixed-length record header followed by variable-length record contents (ESRI 1998). The organisation of the main file is illustrated in (Fig. 6).

The main file header is 100-bytes long. The fields of the file header, with their byte position, value, type and byte order, are shown in Table 4. In this table, the position is shown with respect to the start of the file.

The header for each record stores the record number and content length of the record. The record header has a fixed

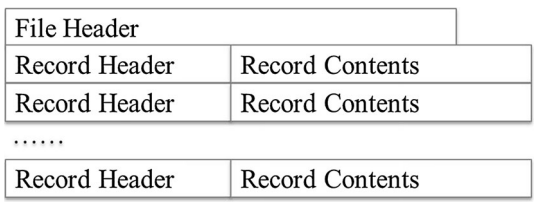

Fig. 6 Organisation of the main file

Table 4 Description of the main file header

Position	Field	Value	Type	Byte order
Byte 0	File code	9994	Integer	Big
Byte 4	Unused	0	Integer	Big
Byte 8	Unused	0	Integer	Big
Byte 12	Unused	0	Integer	Big
Byte 16	Unused	0	Integer	Big
Byte 20	Unused	0	Integer	Big
Byte 24	File length	File length	Integer	Big
Byte 28	Version	1000	Integer	Little
Byte 32	Shape type	Shape type	Integer	Little
Byte 36	Bounding box	Xmin	Double	Little
Byte 44	Bounding box	Ymin	Double	Little
Byte 52	Bounding Box	Xmax	Double	Little
Byte 60	Bounding box	Ymax	Double	Little
Byte 68[a]	Bounding box	Zmin	Double	Little
Byte 76[a]	Bounding box	Zmax	Double	Little
Byte 84[a]	Bounding box	Mmin	Double	Little
Byte 92[a]	Bounding box	Mmax	Double	Little

[a] Unused, with value 0.0, if not Measured or Z type

Table 5 Description of the main file record header

Position	Field	Value	Type	Byte order
Byte 0	Record number	Record number	Integer	Big
Byte 4	Content length	Content length	Integer	Big

Table 6 Point record contents

Position	Field	Value	Type	Number	Byte order
Byte 0	Shape type	1	Integer	1	Little
Byte 4	X	X	Double	1	Little
Byte 12	Y	Y	Double	1	Little

length of 8 bytes. The fields in the file header, with their byte position, value, type and byte order, are shown in Table 5. In this table, the position is shown with respect to the start of the record.

The record contents are dependent on the shape type. Different shape types result in different data types, structures and layouts of the record contents. We use the point shape type as an example in this case. The shape and its mapping used to record contents on the disk are presented in Table 6. The position is shown with respect to the start of the record contents.

The ESRI shapefile format is slightly complex but is also easily described using DFML. The data format markup document for the format is created as shown below.

```
<dataformat name="ESRI Shapefile Format" namespace="com.vge.esri" mode="byte"
description=″Location is with respect to byte 0″>
    <group name=″File header″ location="0, 99">
        <integer location=″0, 3″ value=″9994″ byteOrder=″bigEndian″ description=″File
Code″></integer>
        <group location=″4, 23″>
            <integer value=″0″ byteOrder=″bigEndian″ description=″Unused″
number=″5″></integer>
        </group>
        <integer location=″24, 27″ byteOrder=″bigEndian″ description=″File
Length″></integer>
        <integer location=″28, 31″ value=″1000″ byteOrder=″littleEndian″
description=″Version″></integer>
        <integer location=″32, 35″ byteOrder=″littleEndian″ description=″Shape
Type″></integer>
        <real location=″36, 43″ byteOrder=″littleEndian″ description=″Bounding Box,
Xmin″></real>
        <real location=″44, 51″ byteOrder=″littleEndian″ description=″Bounding Box,
Ymin″></real>
        <real location=″52, 59″ byteOrder=″littleEndian″ description=″Bounding Box,
Xmax″></real>
```

```
            <real location="60, 67" byteOrder="littleEndian" description="Bounding Box,
Ymax"></real>
            <real location="68, 75" value="0.0" byteOrder="littleEndian"
description="Bounding Box, Zmin"></real>
            <real location="76, 83" value="0.0" byteOrder="littleEndian"
description="Bounding Box, Zmax"></real>
            <real location="84, 91" value="0.0" byteOrder="littleEndian"
description="Bounding Box, Mmin"></real>
            <real location="92, 99" value="0.0" byteOrder="littleEndian"
description="Bounding Box, Mmax"></real>
        </group>
        <group name="record" number="unknown">
            <group name="Record header" description="Location is with respect to the start of
the record">
                <integer location="0, 3" byteOrder="bigEndian" description="Record
Number"></integer>
                <integer location="4, 7" byteOrder="bigEndian" description="Content
Length"></integer>
            </group>
            <group name="Record content; Location is with respect to the start of the record
contents">
                <integer location="0, 3" value="1" byteOrder="littleEndian"
description="Shape Type, Point"></integer>
                <real location="4, 11" byteOrder="littleEndian" description="X"></real>
                <real location="12, 19" byteOrder="littleEndian" description="Y"></real>
            </group>
        </group>
    </dataformat>
```

In these case studies, three typical data formats of geographical models were successfully described by DFML. The first format was a user-defined format, and the two other formats were standard formats. The first two formats were text file formats, and the third format was a binary file format. The descriptive ability of DFML has been well demonstrated. Therefore, the data format specification of geographical models can be written by DFML instead of the traditional documents and tables. When sharing geographical models through web services, model service developers and end-users can clearly understand and exchange data format information of the model service with DFML, both of which no longer need to perform data format conversion. In addition, DFML uses only five types of elements (dataformat, datatype, separator, group and location) to describe data formats. Modelers without considerable computer expertise can easily learn and use DFML.

Conclusions and future work

This paper addresses the problem whereby various heterogeneous data formats lead to laborious and unending data format conversion during geographical model sharing. The characteristics of geographical models, data and their formats are initially analysed. To clearly describe the data formats of geographical models in a universal manner, this study proposes a data format description model based on data location, data types and separators and provides the design of a new data format description language: Data Format Markup Language (DFML). DFML can describe data format information, such as data type, structure and layout, in detail using markup elements. DFML has an adequate descriptive ability for data formats and overcomes the complexity problems of existing data format description languages.

This paper facilitates the unified description of the data formats of geographical models. It fills up the lack of attention to format information in VGEs, especially in data environment and modeling environment. It is the foundation for building unified interface of geographical models and description of geographical model services, which provides a new way to share and use geographical models in VGEs more conveniently. That will greatly increase modelers' enthusiasm to share geographical models to jointly construct VGEs, and accordingly add richness and usability to contemporary VGE's research community. In the future, DFML will be applied to model service description. We will develop a DFML Editor to help users automatically or semi-automatically describe data formats of geographical models. Looking ahead, we hope DFML to be a data interface specification among the four components of VGEs. It is a more efficient and friendly way to cope with the various heterogeneous data formats in VGEs.

Acknowledgments The authors thank two anonymous reviewers for their helpful comments. The work described in this article was supported by the National Nature Science Foundation of China (Grant nos. 41301414; 41306078), Program of Natural Science Research of Jiangsu Higher Education Institutions of China (Grant no. 13KJB170008), NSF of Jiangsu Province of China (Grant no. BK20130904) and the Priority Academic Program Development of Jiangsu Higher Education Institutions (Grant no. 164320H116).

References

Alan PW, Beckerle MJ, Hanson SM (2011) Data Format Description Language (DFDL) v1.0 Specification. https://www.ogf.org/documents/GFD.174.pdf. Accessed 12 March 2015

Ames DP, Horsburgh JS, Cao Y, Kadlec J, Whiteeaker T, Valentine D (2012) HydroDesktop: web services-based software for hydrologic data discovery download, visualization, and analysis. Environ Model Softw 37:146–156

Argent RM (2004) An overview of model integration for environmental applications: components, frameworks and semantics. Environ Model Softw 19(3):219–234

Arnold JG, Kiniry JR, Srinivasan R, Williams JR, Haney EB, Neitsch SL (2012) Input/output documentation version 2012. http://swat.tamu.edu/documentation/2012-io/. Accessed 12 March 2015

Bhatt G, Kumar M, Duffy CJ (2014) A tightly coupled GIS and distributed hydrologic modeling framework. Environ Model Softw 62:70–84

Castronova AM, Goodall JL, Elag MM (2013) Models as web services using the open geospatial consortium (OGC) Web Processing Service (WPS) standard. Environ Model Softw 41:72–83

Crosier SJ, Goodchild MF, Hill LL, Smith TR (2003) Developing an infrastructure for sharing environmental models. Environ Plann B 30(4):487–501

Denzer R (2005) Generic integration of environmental decision support systems-state-of-the-art. Environ Model Softw 20(10):1217–1223

Dubois M, Schulz M, Skøien J, Bastin L, Peedell S (2013) eHabitat, a multi-purpose Web Processing Service for ecological modeling. Environ Model Softw 41:123–133

ESRI (1998) ESRI Shapefile Technical Description. http://www.esri.com/library/whitepapers/pdfs/shapefile.pdf. Accessed 12 March 2015

ESRI (2009). ESRI ASCII Grid Format. http://resources.esri.com/help/9.3/arcgisengine/java/GP_ToolRef/spatial_analyst_tools/esri_ascii_raster_format.htm. Accessed 12 March 2015

Feng M, Liu SG, Euliss NH, Yin F (2009) Distributed geospatial model sharing based on open interoperability standards. J Remote Sens 13(6):1060–1066

Feng M, Liu SG, Euliss NH, Young C, Mushet DM (2011) Prototyping an online wetland ecosystem services model using open model sharing standards. Environ Model Softw 26(4):458–468

Gassman PW, Sadeghi AM, Srinivasan R (2014) Applications of the SWAT model special section: overview and insights. J Environ Qual 43(1):1–8

Geller GN, Turner W (2007) The model web: a concept for ecological forecasting. In: IEEE International Geoscience and Remote Sensing Symposium. Barcelona, Spain, July 2007

Goodall JL, Robinson BF, Castronova AM (2011) Modeling water resource systems using a service-oriented computing paradigm. Environ Model Softw 26(5):573–582

Goodchild MF (2005) GIS and modeling overview. In: Maguire DJ, Batty M, Goodchild MF (eds) GIS, spatial analysis, and modeling. ESRI Press, Redlands, pp 1–18

Granell C, Díaz L, Gould M (2007) Managing earth observation data with distributed geoprocessing services. In: Proceedings of the International Geoscience and Remote Sensing Symposium (IGARSS 2007). IEEE CS, pp 4777–4780

Granell C, Díaz L, Gould M (2010) Service-oriented applications for environmental models: reusable geospatial services. Environ Model Softw 25(2):182–198

Kunkel R, Sorg J, Eckardt R, Kolditz O, Rink K, Vereecken H (2013) TEODOOR: a distributed geodata infrastructure for terrestrial observation data. Environ Earth Sci 69(2):507–521

Li S, Kang JW, Wang ZH (2010) Customization of input data for integrated application of SWAT model. Geogr Geo Inf Sci 26(4):16–20 **(in Chinese)**

Li ZL, Yang CW, Huang QY, Liu K, Sun M, Xia JZ (2014) Building model as a service to support geosciences. Comput Environ Urban. doi:10.1016/j.compenvurbsys.2014.06.004

Lin H, Chen M, Lu GN (2013a) Virtual geographical environment: a workspace for computer-aided geographic experiments. Ann Assoc Am Geogr 103(3):465–482

Lin H, Chen M, Lu GN, Zhu Q, Gong JH, You X, Wen YN, Xu BL, Hu MY (2013b) Virtual geographic environments (VGEs): a new generation of geographical analysis tool. Earth Sci Rev 126:74–84

Lu GN (2011) Geographic analysis-oriented virtual geographic environment: framework, structure and functions. Sci China (D) 54(5):733–743

Nativi S, Mazzetti P, Geller GN (2013) Environmental model access and interoperability: the GEO Model Web initiative. Environ Model Softw 39:214–228

Papajorgji P, Beck JL, Braga JL (2004) An architecture for developing service-oriented and component-based environment models. Ecol Model 179(1):61–76

Shi S (2015) Design and development of an online geoinformation service delivery of geospatial models in the United Kingdom. Environ Earth Sci. doi:10.1007/s12665-015-4243-8

Tang ZS, Rao SB, Xie Z, Zhao HR, Wang HW (2008) The research of GIS model sharing platform based on Web 2.0. Sci Surv Mapp 33(4):181–183 **(in Chinese)**

Torrens PM (2009) Process models and next-generation geographic information technology. Gis Best Prac Essays Geogr Gis 31(2):63–75

Voinov A, Cerco C (2010) Model integration and the role of data. Environ Model Softw 25(8):965–969

Voinov A, Shugart HH (2013) 'Integronsters', integral and integrated modeling. Environ Model Softw 39:149–158

Wen YL, Lu GN, Yang H, Cao D, Chen M (2006) A service-oriented framework of distributed geographic model integration. J Remote Sens 10(2):160–168 **(in Chinese)**

Wu N, He HL, Zhang L, Ren XL, Zhou YC, Yu GR, Wang XF (2012) Designing and implementing an online carbon cycle model service platform based on OGC Web Processing Service. J Geo Inf Sci 14(3):320–326

Yu HL, Wu L, Liu Y, Li DJ, Liu LP (2006) A study integration between GIS and GIS-based model based on web services. Acta Geodaetica et Cartographica Sinica 35(2):153–159 **(in Chinese)**

Yue TX (2001) Standardized documentation of models for resources and environment and their integration with GIS. Acta Geographica Sinica 56(1):107–112 **(in Chinese)**

Yue SS, Wen YN, Chen M, Lu GN, Hu D, Zhang F (2015) A data description model for reusing, sharing and integrating geo-analysis models. Environ Earth Sci. doi:10.1007/s12665-015-4270-5

Zhang JQ, Gong JH, Lin H, Wang G, Huang JL, Zhu J, Xu BL, Jack T (2007) Design and development of distributed virtual geographical environment system based on web services. Inf Sci 177(19):3968–3980

Zhu SJ, Nan ZT, Chen H, Liu Y (2010) Online hydrological model service using web service. Remote Sens Technol Appl 25(6):853–859 **(in Chinese)**