STATISTICAL COMMISSION and
ECONOMIC COMMISSION FOR EUROPE

Working Paper No. 20
English only

CONFERENCE OF EUROPEAN STATISTICIANS

UN/ECE Work Session on Methodological Issues Involving the Integration of Statistics and Geography
(Neuchâtel, Switzerland, 10-12 April 2000)

Topic (ii): Spatial database management and (geo-)data warehousing

EFFECTIVE COMPILATION AND BROWSING OF GEOGRAPHICAL METADATA USING COMMON INTEROPERABILITY TOOLS

Submitted by the Swiss Federal Institute for Forest, Snow and Landscape Research, Switzerland[1]

Contributed paper

I. INTRODUCTION

1. Today, the broad availability of geographical data puts a user in the comfortable situation of having access to many different data sources. Unfortunately, the available data in most cases is only sparsely described; this renders more difficult the evaluation of the fitness of use of the data for specific requirements. Efforts in standardizing data transfer have led to the requirement that a data set be accompanied by so-called metadata which report on the quality of the transferred data. The Spatial Data Transfer Standard SDTS (National Institute of Standards and Technology, 1994) specifies five major elements consisting of lineage, positional accuracy, attribute accuracy, completeness and logical consistency to comprehensively describe the quality of a data set. The ICA Commission on Spatial Data Quality agreed on two additional elements for quality assessment: semantic accuracy and temporal information (Guptill and Morrison, 1995). However, the quality elements are described on a somewhat conceptual level without specifying concrete measures for individual quality parameters. Therefore, subsequent research has focused on the development of such measures. Whereas much effort has been expanded on the quality elements of positional accuracy, attribute accuracy and completeness (for instance, Tveite and Langaas, 1999, Leung and Yan, 1998), the lineage element received only very little attention. However, it is recognized that particularly lineage tracking and thus the comprehensive description of lineage is one of the key requirements for the operational use of geographic information within a multi-user and multi-organization application (Sargent, 1999).

2. Besides formalizing data quality, categories and attributes to describe the content, condition and other characteristics of data have been researched. Standards such as the "Content Standard for Digital Geospatial Metadata" (CSDGM, Federal Geographic Data Committee, 1997) reflect the work initiated in this field. The CSDGM is one of the most used standards (at least in the US) and provides a common set of terminology and definitions for the documentation of digital geospatial data. It establishes the names of data elements, their definitions and information on the values that must be provided for the data elements. Data elements are identified for information on the identification, data quality, spatial data type and reference, entities and attributes and distribution. Since about 200 different data elements are defined, the standard can be considered as comprehensive. However, this number of different data elements make the interpretation and assessment of a data set for its fitness of use for a specific application a difficult task.

[1] Prepared by Martin Brändli.

3. This paper concentrates on the development and integration of tools that enable fast and effective access to metadata, in particular to lineage, one of the most important data quality components. The tools consist of "de-facto" standard and freely accessible software applications which mainly allow for graphical assessment of data suitability. The main characteristic of the tools is their interoperability which may be defined as the capability to communicate with other software tools and the ability to be used from different platforms and operating systems.

4. The work described here is part of a larger project conducted in collaboration with the Swiss Agency for the Environment, Forests and Landscape. The content of the project is the design and implementation of a database for the storage and management of data of environmental protection areas of mires, riverine forests, semi-arid meadows, etc. (Grünig, 1992). In addition to geometric objects which represent the extent of these areas, base documents such as images, documents containing the criteria for selecting protection areas and governmental documents which build the legal base for the areas must be stored. As well as the database part of the project, access software enabling the exploration of the spatial database has to be implemented. This paper mainly reports on results obtained from the implementation of the access software.

II. REPRESENTATION OF LINEAGE IN THE CSDGM

5. Lineage can be simply defined as "the history of a data set" (Clarke and Clark, 1995, p. 13). History is explained as "the recounting of the life cycle of a data set, from its collection or acquisition, through the many stages of compilations, corrections, conversions, and transformations to the generation of new interpreted products" (Clarke and Clark, 1995, p. 13).

6. The following example (http://water.usgs.gov/GIS/metadata/usgswrd/ag_chem.html) shows which data elements the CSDGM uses to record lineage, and how lineage is usually presented. Of course, data elements necessary for the description of lineage are only a subset of the full metadata set.

Lineage:
  Source_Information:
    Source_Scale_Denominator: 2,000,000
  Process_Step:
    Process_Description: RESELECT COUNTY2M COUNTY.NEW POLY
    Source_Used_Citation_Abbreviation: None
    Process_Date: 19911112
    Process_Time: 1528
    Source_Produced_Citation_Abbreviation: None
  Process_Step:
    Process_Description: LABELERRORS COUNTY.NEW
    Source_Used_Citation_Abbreviation: None
    Process_Date: 19911112
    Process_Time: 1529
    Source_Produced_Citation_Abbreviation: None
  Process_Step:
    Process_Description: LABELERRORS COUNTY.NEW
    Source_Used_Citation_Abbreviation: None
    Process_Date: 19911112
    Process_Time: 1556
    Source_Produced_Citation_Abbreviation: None

Fig. 1: Extract of a metadata set consisting of lineage information. The full lineage description contains additional process steps.

7. The example reveals the textual character of the information, presented in listings which make the comprehensive exploration of lineage a cumbersome task. The description of lineage consists of the specification of source information including the scale of the source, and a number of process steps necessary to produce a final dataset. A user receiving this lineage information for a given dataset is left with many questions, such as:

- What exactly is the data source? How does it look? The specification of a source scale as the only information is not helpful for a user to sufficiently assess a final data set. A reference to the source, the inclusion of the source or a subset of it would significantly improve data suitability assessment.
- What does the process step do? The data supplier in the given example reduces the event description to a single ARC/INFO statement assuming that a potential user of the data set is familiar with the software, knowing the actual command, constraints to input data sets and impacts of specified parameters. The actual specification of the CSDGM allows for a non-standardized and unstructered way of providing process information which means that the structure, content, and extent of the function or tool description is left to the data set supplier. However, a more comprehensive description of a process step should include properties such as the behavior of a function, requirements on input data sets, semantics of and constraints on parameters used, description of output, etc.
- What exactly is the impact of a process step on a given data set? In many cases, it will be difficult to estimate the impact of a process step on a source or derived data set even if process description is complete. Access to intermediate data sets or representative subsets which clearly show the impact would help to get a good feel for a function or tool used for a process step.

8. In summary, the current way of providing lineage information with the CSDGM must be considered unsatisfactory and insufficient. Additionally, the CSDGM is a means to describe metadata for a particular data set. Since data is increasingly stored and managed in databases, lineage description and metadata must be adapted accordingly. The following section briefly describes the database approach chosen within this project, structuring data according to processes and providing the necessary metadata information.

III. PROCESS-ORIENTED DATABASE

9. The database design of the project has to consider different types of data, such as images, text documents and spatial objects or entities. The selection of environmental protection areas normally consists of several steps starting, for instance, with a governmental decision which initiates the selection process. In a next step, experts determine criteria for the selection of areas. Subsequently, base documents (images, maps) are used for digitizing potential protection areas. During subsequent field work, potential areas and their extents are verified, followed by a refinement of the objects. Finally, a legal document published by the government determines the exact position and extent of the protection areas.

10. Since the complete selection process may be divided into clearly identifiable steps, the design of the database was based on this division by introducing a process-oriented data model. Every step of delineating protection areas is considered an individual process which is related to a certain type of data and corresponding metadata. A detailed description of the design and implementation of the database approach, using a relational database management system, is presented in Baumberger and Hägeli (2000). The main components of the data model are process types, which are classes of similar process steps, and process instances, which are the actual occurrences of process steps. Data resulting from a process step are associated to the process instance. The interrelationships between process types are modelled by associating a child process type with its parent types. The linking of the process types essentially defines a directed graph which shows the lineage for a given node of the graph. The model enables the tracking of the history of all entities stored within the database by traversing the graph of processes. An example of a graph of linked process types is shown in fig. 2. Metadata - not restricted to lineage information - may be associated to process types as well as to single processes:

- Process type: Provides metadata for the description of the functions, methods and tools to perform the task specified by the process type.
- Process instance: Provides metadata of two kinds: since an instance of a process type uses its specific parameters for performing a task, the values of the parameters must be specified individually. In addition, the process instance also includes the metadata of the related data set, consisting for instance of accuracy measures.

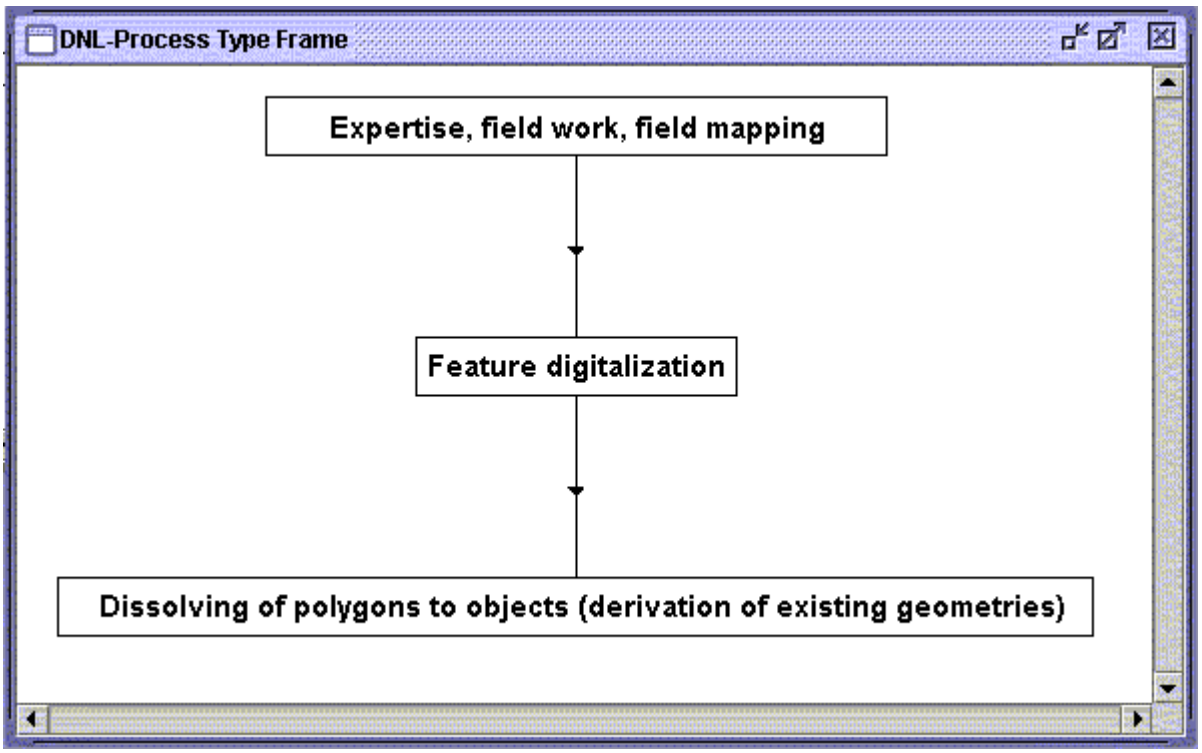

Fig. 2: Example of a directed graph for process type lineage. Visualization of the graph uses the "MetaViewer" software described below.

IV. ACCESS OF LINEAGE INFORMATION

11. Accessing and tracking lineage information of a selected data collection in order to assess its fitness for use is implemented using an interactive method of exploring the database. A user may get a good feel for the content and validity of a data collection by interactively selecting data entities, studying related metadata, and assessing the impact of process steps on subsequent data elements.

12. The implementation of such an interactive system must fulfill a number of requirements:

- Easy access: all data – including spatial entities – are stored in an Oracle database. Spatial data is managed by the Spatial Data Base Engine (SDE) of ESRI (ESRI, 1998). Since only a small part of the expected user community is familiar with SQL and SDE, the user interface for data access has to be built by omitting the use of these access techniques.
- Graphical representation of process type and process graphs: graph drawing algorithms (Di Battista et al., 1994) must be used to display the directed graphs representing data lineage.
- Visualization of different types of data: since the database contains different types of data, the access software must be capable of interpreting them accordingly, i.e. rendering the geometry of spatial objects, images and text documents.
- Platform-independence: The database runs on a server machine. Access must be granted from different personal computers or work stations, running operating systems such as Unix, Windows or MacOS.

13. Since platform-independence is one of the most important requirements, the programming language Java is used to implement access and browsing software for metadata and data contained in the database. In addition, Java provides a rich set of graphical user interface components and numerous different application programming interfaces which enable interoperability of data as well as processing resources to a great extent.

14. The architecture of the currently implemented software, called "MetaViewer", is presented in fig. 3. The central component of the software is a Java-based integration layer which controls access to the database and integrates components for metadata visualization, lineage tracking through a graphical user interface, and rendering of data corresponding to the various data types stored in the database. Taking a closer look at the single components of the architecture reveals that the software for component integration (APIs) and the components themselves - except the DBMS and GIS - consist of freely accessible tools or programming code.

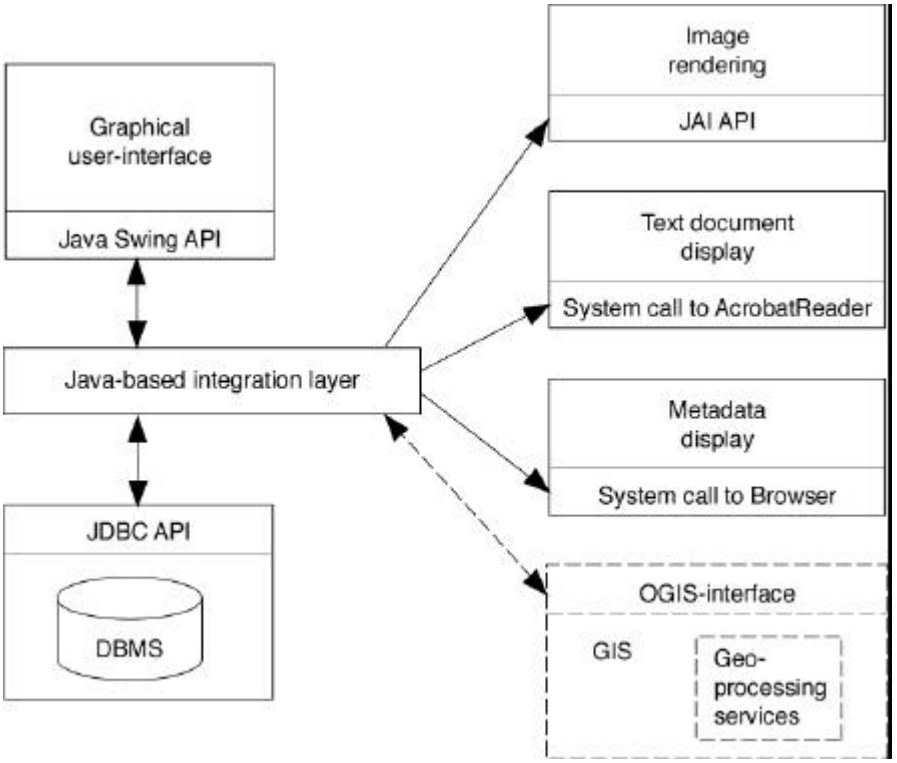

Fig. 3: Architecture of the software implemented for access and visualization of metadata and data contained in the DBMS. Dashed lines denote a component which is not yet integrated.

15. In detail, access to metadata and data and their visualization is implemented as follows:

i) Java-based integration layer: the tasks of the integration layer are to control access to the DBMS by compiling query statements, intermediate storage of retrieved and internally produced data, distributing data to other components according to the data types and user requests, and dynamic compilation of metadata and lineage information. The implementation of these tasks utilizes the basic classes of Java.

ii) Access to the DBMS and retrieval of data and metadata: in order to access and retrieve data from the Oracle database, functionality for establishing a connection to the database and for submitting SQL-statements must be available. This is provided by the Java application programming interface JDBC API which supports basic functionality to SQL databases (White and Hapner, 1998). The JDBC API consists of a set of interfaces to open connections to particular databases,

execute SQL statements, and process the results. Once a connection to the database is established, the integration layer of the "MetaViewer" software is responsible for preparing SQL statements based on user requests, executing the statements, receiving the results via the JDBC interface, and finally interpreting the result sets accordingly.

iii) Graphical user interface: the graphical user interface (GUI) is responsible for user control of the complete access and visualization software. The design and implementation is based on the Java Foundation Classes (JFC) which encompass a group of features to help build particular GUIs. One of these features is Swing, a programming interface which enables the handling of GUI-components such as windows, dialogs, text fields, tables, menus, scrollbars, buttons, and many others. Implementing the Swing-based GUI is straightforward since interface calls to invoke GUI-components are on a high level. The main part of the actual user interface is the handling and display of lineage information. Fig. 2 showed a process type graph displayed within a Swing-based window. The rendering of directed graphs needs specific graph drawing software. Corresponding searches on the web result in a rich set of graph drawing resources (http://www.cs.brown.edu/people/rt/gd.html). One of these resources is Java code of Erlingsson and Krishnamoorthy (1997) including classes and methods for interactively drawing different types of graphs. The code was adapted to display directed acyclic graphs as requested in our application. Fig. 4 shows the graph of process instances belonging to the process types shown in fig. 2. From the display of graphs of specified processes or process types, a single box or several boxes may be selected in order to visualize corresponding data or metadata (an example will be shown below).

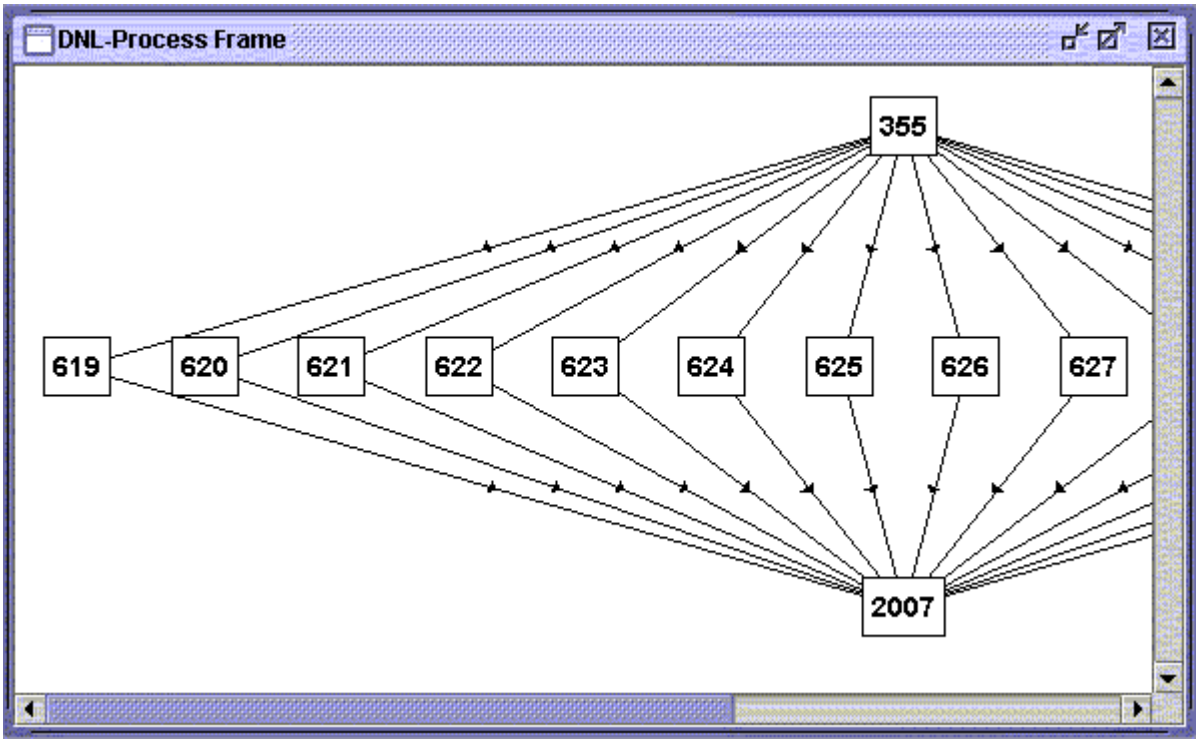

Fig. 4: Example of a directed graph for processes. The graph shows process instances of process types displayed in fig. 2.

iv) Image rendering: the database stores images with data formats such as tiff or gif as binary large objects. The Java application programming interface JAI (Java Advanced Imaging) offers the necessary functions for image processing and display by supporting a broad variety of image formats, data types, and image file types (Sun Microsystems, 1999). For the purpose of this

application, only functions for rendering images are used. The main problem for image rendering is the large size of a great number of stored images and the corresponding upload time from the database to the integration layer. For this reason, a user is first informed about the size of the image and may subsequently decide whether the image should be uploaded and displayed.

v) Text document display: the visualization of text could be implemented with Java Swing Classes by placing text within a specific window. However, text documents are prepared as PDF-files (Portable Document Format, Adobe Systems, 1999) and then stored as binary large objects in the database. Since it is not the aim of the presented software to parse PDF-files, the integration layer establishes a link to a PDF-reader (Adobe AcrobatReader). The text stream received from the database is sent to the reader by a system call which starts the reader application. The reader application in turn displays the text document. This solution cannot be considered a true integration of software components, however, but simple system calls can be implemented independently of a given hardware and operating system. Releasing the constraint of platform-independence would enable the implementation of fully interoperable software components, for instance by using the Component Object Model (COM) of Microsoft and its embedding into Java (Verbowski, 1999).

vi) Metadata display: metadata which are dynamically compiled according to a user request are compiled into a HTML-document (or into an XML-document in the future) and may be visualized by standard browser software. The procedure and remarks are the same as for the display of text. As for text, metadata could also be displayed using Swing components. However, formatting metadata using HTML- or XML-tags and accompanying style sheets and subsequent display in a browser such as Netscape or Explorer greatly reduces programming effort.

vii) GIS component: the dashed lines of the GIS-box in fig. 3 indicate that this part of the software is not yet implemented. Although the display of the geometries of spatial objects is not possible at the moment, corresponding metadata as well as non-spatial attributes of the objects may be visualized. Actually, the visualization of spatial objects and corresponding metadata is separately implemented using GIS-software (Steinmeier, 1999). The planned solution indicated in fig. 3 approaches integration of a GIS component using OGIS interfaces. OGIS is an abbreviation for Open Geodata Interoperability Specifications which reflect work initiated by the Open GIS Consortium (OGC). The OGC is a membership corporation consisting of the most important hardware, software, database and GIS vendors, federal agencies and universities. The goal of the consortium is the promotion of new technical and commercial approaches to interoperable geoprocessing by specifying a software framework for distributed access to geodata and geoprocessing resources (Buehler and McKee, 1998). After four years of work, OGC released the first implementation specifications including the handling of simple spatial features (Open GIS Consortium, 1998a, 1998b, 1998c). However, actual implementations of the standard on top of existing GIS are not yet available since the GIS-vendors only hesitatingly adopt the specifications.

16. One of the important properties of the presented architecture is that it is open for the integration of further components. The current method of acces to the database is mainly metadata-driven. However, metadata-driven exploration is only one paradigm for effective search of databases in order to assess the fitness of use for a given data collection. Flewelling and Egenhofer (1999) propose a subset-supported paradigm which has its origin in the fact that data may be best understood by applying exploratory analysis. This type of analysis needs statistical tools that may be integrated into the proposed architecture. A challenging question is the selection of a subset which is representative of a given collection of stored entities.

V. AN EXAMPLE USING THE ACCESS SOFTWARE

17. This section describes one possibility of a user session for metadata and data access and visualization. First, the user must establish a connection to the database. As outlined above, data of mires, riverine forests, semi-arid meadows, etc. are primarily stored with a process-oriented data model. If a user already knows available process types or individual processes, he or she can directly start lineage tracking by specifying one of these entities after a connection is opened. A second starting point for lineage tracking is access via the selection of a particular inventory to which the data belongs. Processes and process types are aggregated to inventories which store the history of the evolution of a particular type of protection area. A third entry point for database exploration are the different data types available in the database. A user may, for instance, start lineage traversal by selecting a particular image. Once a process type or a single process is selected, the corresponding graphs are displayed. From the graphical representation, metadata and data of the lineage graphs may be selected and displayed, as shown in fig. 5.

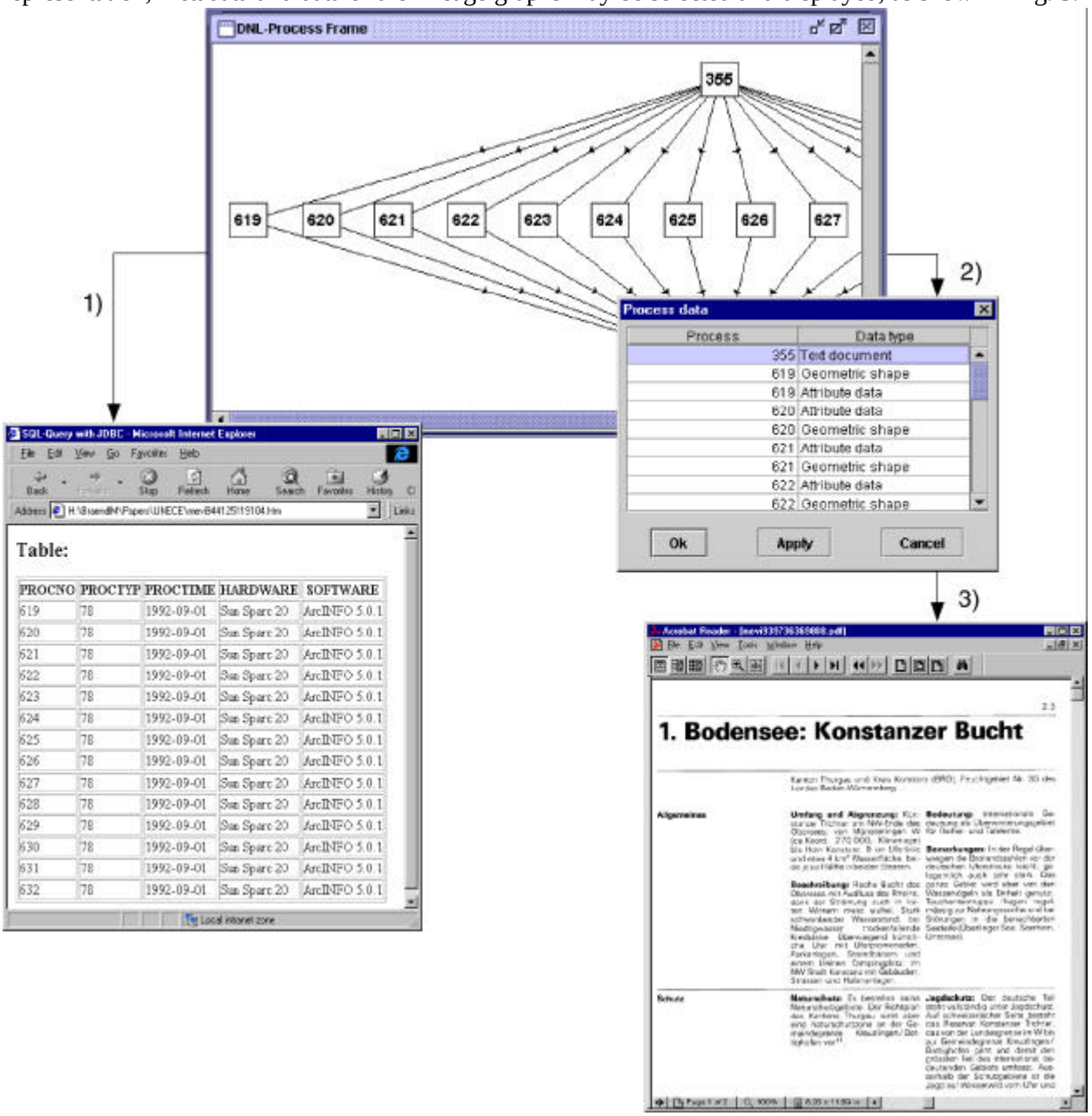

Fig. 5: User session for lineage access. 1) From the graph of processes metadata is displayed in a browser. 2) Data types for the given process boxes are displayed from which items can be selected for rendering. 3) Display of a text document with a pdf-reader.

VI. CONCLUSIONS

18. Experiences with the implementation of the presented architecture for an effective compilation and browsing of metadata using the programming language Java are very promising. This language, with its rich set of GUI-components and additional application programming interfaces, provides an optimal base for the implementation of software components requesting interoperability. However, since Java is an object-oriented language, profound knowledge of and experience with this programming paradigm are indispensable.

19. The project in which the presented work is conducted is still at an early stage. Therefore, the user interface as well as the database design have not yet been tested extensively. In addition, the database is only populated with a small amount of data. However, although users from particular application domains have not yet been addressed, the access software has already demonstrated its potential as a reliable supporting tool for database administrators and data providers. Visually tracking lineage permits the verification of the content of the database for completeness since missing links between successive processes may be identified quickly.

20. Lineage tracking by interpreting listings of currently defined standards is not an effective means for data suitability assessment. The presented approach enables access to metadata in a more comprehensive and flexible way, and in addition allows the user to browse source or derived data as lineage information. In this sense, data themselves become part of metadata.

VII. REFERENCES

Adobe Systems (1999): Adobe PDF. Adobe Systems Inc., San Jose, CA (http://www.adobe.com/products/acrobat/adobepdf.html).

Baumberger, N., and M. Hägeli (2000): Using metadata in multistep preprocessing and longterm monitoring. Submitted Paper, GIS Work Session UN/ECE, Conference of European Statisticians, April 10-12, Neuchatel, Switzerland.

Buehler, K., and L. McKee (1998): The OpenGIS Guide - Introduction to interoperable geoprocessing and the OpenGIS Specification. Third Edition, Open GIS Consortium, Wayland, MA (http://www.opengis.org/techno/guide.htm)

Clarke, D. G., and D. M. Clark (1995): Lineage. In: Guptill, S. C., and J. L. Morrison (eds.): Elements of spatial data quality. Elsevier Science Ltd., Oxford.

Di Battista, G., P. Eades, R. Tamassia, and I. Tollis (1994): Annotated bibliography on graph drawing algorithms. Computational Geometry: Theory and Applications, 4.

Erlingsson, U., and M. Krishnamoorthy (1997): Interactive graph drawing on the World Wide Web. Proceedings 6th International World Wide Web Conference, April 7-11, Santa Clara, CA (http://decweb.ethz.ch/WWW6/Posters/703/igdraw.html).

ESRI (1998): Spatial database engine. ESRI White Paper, Environmental Systems Research Institute, Inc., Redlands, CA (http://www.esri.com/library/whitepapers/sde_lit.html).

Federal Geographic Data Commitee (1997): Content standard for digital geospatial metadata (revised April, 1997). Federal Geographic Data Commitee, Washingon D. C.

Flewelling, D. M., and M. J. Egenhofer (1999): Using digital spatial archives effectively. International Journal of Geographical Information Science, 13 (1).

Grünig, A. (1992): Mires and man: Mire conservation in a densely populated country - the Swiss experience. Excursion Guide and Symposium Proceedings of the 5th Field Symposium of the International Mire Conservation Group (IMCG), Switzerland. Swiss Federal Institute for Forest, Snow and Landscape Research, Birmensdorf, Switzerland.

Guptill, S. C., and J. L. Morrison (1995): Elements of spatial data quality. Elsevier Science Ltd., Oxford.

Leung, Y., and J. Yan (1998): A locational error model for spatial features. International Journal of Geographical Information Science, 12 (6).

National Institute of Standards and Technology (1994): Federal Information Processing Standard Publication 173 (Spatial Data Transfer Standard Part 1, Version 1.1), U.S. Department of Commerce.

Open GIS Consortium (1998a): OpenGIS simple features specification for OLE/COM. Open GIS Consortium, Wayland, MA (http://www.opengis.org/techno/specs.htm).

Open GIS Consortium (1998b): OpenGIS simple features specification for CORBA. Open GIS Consortium, Wayland, MA (http://www.opengis.org/techno/specs.htm).

Open GIS Consortium (1998c): OpenGIS simple features specification for SQL. Open GIS Consortium, Wayland, MA (http://www.opengis.org/techno/specs.htm).

Sargent, P. (1999): Feature identities, descriptors and handles. In: Vckovski, A., K. E. Brassel, and H.-J. Schek (eds): Interoperating geographic information systems. Proceedings Second International Conference, INTEROP'99. Lecture Notes in Computer Science, 1580, Springer, Berlin, Heidelberg.

Steinmeier, C. (1999): Development of a dynamic user-interface to SDE-data for non-GIS specialists. Conference Program, 14th ESRI European User Conference, Munich, 15-17 November.

Sun Microsystems (1999): Java advanced imaging API. White Paper, Version 1.0, Sun Microsystems Inc., Palo Alto, CA (http://java.sun.com/products/java-media/jai/docs/index.html).

Tveite, H., and S. Langaas (1999): An accuracy assessment method for geographical line data sets based on buffering. International Journal of Geographical Information Science, 13 (1).

Verbowski, C. (1999): Using COM objects from Java - A technology overview. Microsoft Corporation, Redmond, WA (http://www.microsoft.com/java/resource/java_com2.htm).

White, S., and M. Hapner (1998): JDBC 2.0 API. Version 1.0. Sun Microsystems Inc. Palo Alto, CA (http://java.sun.com/products/jdk/1.3/docs/guide/jdbc/spec2/jdbc2.0.frame.html).