HISTORICAL METHODS, Fall 2004, Volume 37, Number 4

Using Java and XML in Interdisciplinary Research

A New Data-gathering Tool for Historians Working with EuroClimHist

URS DIETRICH-FELBER
Institute of History
University of Bern

Abstract. Searching, gathering, organizing, and retrieving data are basic tasks for historians. As long as historians work by themselves, decisions concerning data format, data exchange, computer platform, and the like remain secondary. Teams of historians often use relational databases for centralized data storage. However, fundamental risks are implicit when one uses databases. Among technical considerations, the process of transformation between the source and the database is a deciding factor. For those who gather data, the effective use of such possibilities as data exchange, compatibility, and simplicity of survey and the reuse of the data in other contexts and platforms becomes increasingly important. In contrast, the user's needs include the possibility of data verification and use of the data for more than one question. Many relational databases have considerable shortcomings because the stored data lose the visual characteristics, the syntax, and the semantics of the original source. The EuroClimHist database environment is a part of the NCCR Climate project. It uses a data tool, written in Java for the gathering of documentary data, which generates extensible markup language (XML) files for data exchange.

Keywords: climate change, data exchange, documentary data, EuroClimHist, Java, XML

Searching, gathering, organizing, and retrieving data are basic tasks for historians. Small projects involving only a few sources can be organized in an old-fashioned way without any information technology (IT) support. Larger projects, comprising a number of sources, a quantitative approach, or projects with more than one person involved, very often use IT tools. A simple way to organize data is to store them in a common file format, such as Microsoft Word, on a hard disk. The disadvantages, however, are obvious. Data exchange is limited to the owners of the program used; more important, no specific languages—structured query language (SQL),[1] for instance—enable data retrieval.

Ideally, one should gather, code, and then transform the information into a database, generally into a relational database. By so doing, one can query the data very quickly, but the source is cut into pieces; essentially, when one is viewing the source, the syntax and the semantics are excluded. The following two examples are typical of methods used in the past to organize documentary data.

The first example, the former EuroClimHist system, was built to gather documentary data for the reconstruction of climate as a database for historical climatology. Historical climatology attempts to reconstruct weather and climate as well as natural disasters for the last millennium; that is, before the creation of national meteorological networks (about 1860). Historians gathered data from throughout Europe, including Russia, and the data included all the symbols for the various languages.

The data were processed in two steps. In the first step, the researcher performed a partial transcription of the source (documentary data) in American standard code for information interchange (ASCII)[2] text files by using a four-element record structure. The data owner used a numerical codebook to code the data. The codebook included instructions on coding a date or period of time and a place, as well as guidelines for coding weather observations. The fourth element of this structure enclosed a fragment of the original source text, including all special characters of the various languages, coded in LaTeX, a platform-independent typesetting system that allows any symbol, such as those used in the Chinese or Arabic languages, to be typed through the use of ASCII code sequences. Four files were created in this process: the data file with the described record structure, containing the transcribed source, and three separate files containing information about new observations, place codes used, and metadata of the source, respectively.

In the second step, after obtaining the data via e-mail or floppy disk, the person responsible inspected the four files and, in the course of a costly process using several different programs, transformed each of the files into the desired flat files. The output of this process was a number of ASCII files stored on a VMS[3] server.

The second example, developed by Stephan Hagnauer and Niklaus Bartlome (1998) and based on the annual financial reports of various Bernese governmental districts, used both a quantitative and a qualitative approach. Thousands of entries providing information about expenses in the public sector were included. The historians processed the data in the following four steps.

First, the researcher made a complete transcription of the source through the use of Microsoft Word. Although this was a time-consuming step, it gave the owner the chance to use the source in other contexts. Second, using a Microsoft Word Macro package, the person responsible read the text and assigned words, sentences, or sections to a list of entries. This step also included searching for a place name or an entry, as well as assigning each a specific code. Third, the researcher exported the data in ASKSAM,[4] a plain database system that allows one to store structured and unstructured elements of data without programming skills. When the data were imported into ASKSAM, the marked elements of the Word file were split section by section and stored in the database—an intermediate step for transforming the Word files into the relational database. Fourth, the researcher exported the data from ASKSAM to a relational database system such as Microsoft Access, which provides many more possibilities for joining, querying, and reporting than does ASKSAM or Word.

Initial Evaluation

The previous EuroClimHist approach was very problematic in conjunction with new data. The user needed a large codebook with a complex structure for coding the three basic entries of date, place, and observation; furthermore, a completely different syntax using LaTeX codes for special characters in the source text had to be implemented. As a result, the initial outlays were too large. It so happens that the user had no clue that potential mistakes were being made during the coding process: tests that control for such possibilities must be conducted before any other kind of processing. In addition, the structure of the file was not specifically outlined. For example, line feeds were allowed everywhere, but a line feed also separated records. Finally, the source was fragmented and unavailable as a complete transcription. The main advantages to this approach, however, were the ASCII file format and the use of LaTeX, which guaranteed the independence of the platform.

Hagnauer and Bartlome's (1998) solution is much easier to handle, but it requires more steps to transform the data from the source to the database. In addition, as opposed to the first example, the complete source is available, thus enabling its use in other contexts. The main problems are the file format, the use of three different programs—Word, ASKSAM, and Access, for example—and the Macro packages. The files are not platform independent, and the maintenance of the program is costly and time consuming because the process of updating from one Word release to another can mean that all the Macro programs must be upgraded. Furthermore, the user must have the same Word release used by the programmers; whenever the programmers update the package, the user must also update his or her version. For a comparison of the two solutions, see table 1 (which also shows the data workflow).[5]

A Short Introduction

The overall aims of the National Center of Competence in Research on Climate (NCCR Climate)[6] relate to the understanding and prediction of climate variation, the

TABLE 1. Comparison Between the Two Data Tools

Description	EuroClimHist	Bernese public finances
Data tool is platform independent	Yes	No
File format is platform independent	Yes	No
Tool is based on open standards	No	No
Tool is basically built to handle more than one problem and can be adapted	No	No
Allows the transcription of sources in any language without coding the special characters with code sequences	No	Yes (no in older versions of Microsoft Word)
Allows syntax checks before further processing of the data	No	No
Tool is easy to use or has a GUI	No	Yes

Source. For Bernese public finances, see Hagnauer and Bartlome (1998).
Note. GUI = graphical user interface.

assessment of its impact on the natural environment, society, and the economy, and the consideration of possible mitigating options. These aims are timely, challenging, and significant. Understanding the climate system and changes therein was accorded highest priority in a recent initiative of the International Council of Scientific Unions.

NCCR Climate is a major integrated, interdisciplinary initiative. It pursues coordinated research on the variability of the climate system and its potential for change, and it also encompasses the reconstruction of past climate and the study of key physical, chemical, and ecological processes. It makes a concerted effort to develop procedures for seasonal forecasting and for the forecasting of extreme weather events. This research is conducted in close conjunction with the evaluation of the associated risks to governmental and business sectors, as well as to society in general. To this end, NCCR Climate is structured into four complementary and interdependent work packages: (1) past climate—variability, trends, and extreme events; (2) future climate—processes and forecasting; (3) impacts of climate variability and change; and (4) risk assessment, risk hedging, and socioeconomic response.

Integral to NCCR Climate is the deployment of a diverse spectrum of approaches, methods, and disciplinary knowledge, and this integration can only be secured in the context of multi-institutional and interdisciplinary collaboration.

The overall goals of NCCR Climate are as follows:

- Acquiring a better understanding of climate system processes, variability, and predictability, as well as the complex interrelationships of climate, economic, and societal-driving factors
- Adapting and refining scientific tools and acquired knowledge for Switzerland, considering specific characteristics in physical, chemical, biological, geographical, economic, and societal-forcing factors
- Transferring and applying the knowledge to assess the future cost and risks of expected climate change and providing a basis for adaptation strategies
- Investigating new financial and economic tools to hedge against the increased probability of extreme events

Work Package 1: Past Climate—Variability, Trends, and Extreme Events

A quantitative reconstruction of variability during times when anthropogenic influences on the climate system were either absent, different, or reduced is basic to the understanding of the processes that govern natural variability. The goal of Work Package 1 (past climate—variability, trends, and extreme events) is to obtain high-resolution records for an extended time period, including annual or seasonal resolution of proxy data commensurate with the physical variables (e.g., temperature, precipitation, and greenhouse and other trace-gas concentrations), and to combine them with the records of observation of the past 140 years. A well-constrained time scale is a prerequisite to making it possible to synchronize the various records. The major archives used are ice cores, tree rings, and lake sediments. To quantify the impacts on natural climate variability, particularly the biotic response (vegetation cover), high-resolution data from specific areas that react sensitively to climate change (lakes and high-latitude or high-altitude tree lines) are required. Further data, especially on extreme events, can be derived from historical archives and documentary data when dense coverage in the Alpine region permits the construction of regional patterns. Work Package 1 addresses the following major questions: When, where, and how did historic extreme events occur? Is there a pattern of preconditioning? Which extreme events and related natural disasters had a significant impact on past ecosystems and societies in the natural climate before 1900?

As a part of Work Package 1, the subproject "paleoclimate variability and extreme events" (Palvarex) attempts, among other things, to investigate the following tasks: (a) estimating the time resolution necessary to precisely diagnose extreme events in the central European and Alpine areas and to determine their temporal distribution; (b) determining the correlations of extreme events with distinct climate modes or regimes or through the use of data from natural archives; (c) assessing the impact of extreme climate events on past ecosystems and societies in a natural climate (i.e., before 1900); and (d) studying how the most extreme events documented within the last millennium relate to the "once-in-a-century" events documented in the twentieth century.

To answer those questions, documentary data are necessary, because only such data allow the reconstruction of extreme events from the past. According to Christian Pfister, Rudolf Brázdil, and Mariano Barriendos (2002), documentary data are classified as follows: *Narrative* information is the dominant mode of information in chronicles, letters, weather diaries, and the like. *Numerical* information is the dominant mode of information in early instrumental observations and in certain kinds of documentary proxy data (e.g., date on which the harvest was opened; volume of grape harvest), occasionally included in narrative reports (e.g., "it did not rain for 40 days"). *Pictorial* information consists of dated visual representations of natural objects or human artifacts in the past that may be used for climate reconstructions (e.g., paintings representing the position of glacier tongues in a given year). *Epigraphic* or *archaeological* information exists when some handmade artifacts outside the archives are used as indicators for climate reconstructions (e.g., marks on buildings indicating the level of specific historic floods).

To manage such documentary data in the Palvarex subproject, researchers used the data from the former EuroClimHist database system as a starting point. Those data formed the basis of a completely new database system, also called EuroClimHist. However, for the tasks described herein, much more documentary data are needed. To obtain the necessary quantity of documentary data, a new network of partners must be established throughout Europe. Researchers created the new data tool to make it possible for all partners to gather new data in an easy manner.

The Data-gathering Tool

To simplify the process of transcribing and coding documentary data, researchers developed a data-gathering tool with the following features: (a) it runs on any platform; (b) it is easy to use; (c) it has a graphical user interface (GUI); (d) it has a file format based on Unicode,[7] not on ASCII, thus allowing the use of any known symbol without coding it; (e) it has an easy exchange of files because it has an open, well-known format: extensible markup language (XML); (f) it offers many possibilities for further processing; (g) it makes use of open standards; (h) it is adaptable for use with other kinds of historical data projects; (i) it includes a basic syntax check; (j) it allows the use of the source in other contexts; and (k) it does not require a codebook.

The tool is implemented in Java 1.4.2,[8] which is a platform-independent, object-oriented language developed by SUN Microsystems and available without charge. Programs written in this language run on PC, Apple-Macintosh, Linux, and Unix. The user interface allows keyboard commands and the use of the mouse. After starting the tool, the user obtains a window with four tabs and three menus.

The first tab (see fig. 1) includes entries about the individual who is transcribing the source; that is, the individual's given name, surname, publications, and so on. The second tab (see fig. 2) contains a number of entries pertaining to the critical evaluation of sources, including the author, dates or years of birth and death, place where the original source was obtained by the historian or scientist, and so on.

Figure 3 shows tab 3, which is separated into two major areas. The area on the left is a table with six columns. Column 1 contains the entries for the date or the time range; column 2 is a Boolean value that indicates whether or not a date may be considered to be certain. Columns 3, 4, and 5 indicate the place, the observation, and a fragment of the source text, respectively. The column entitled "pc" (not

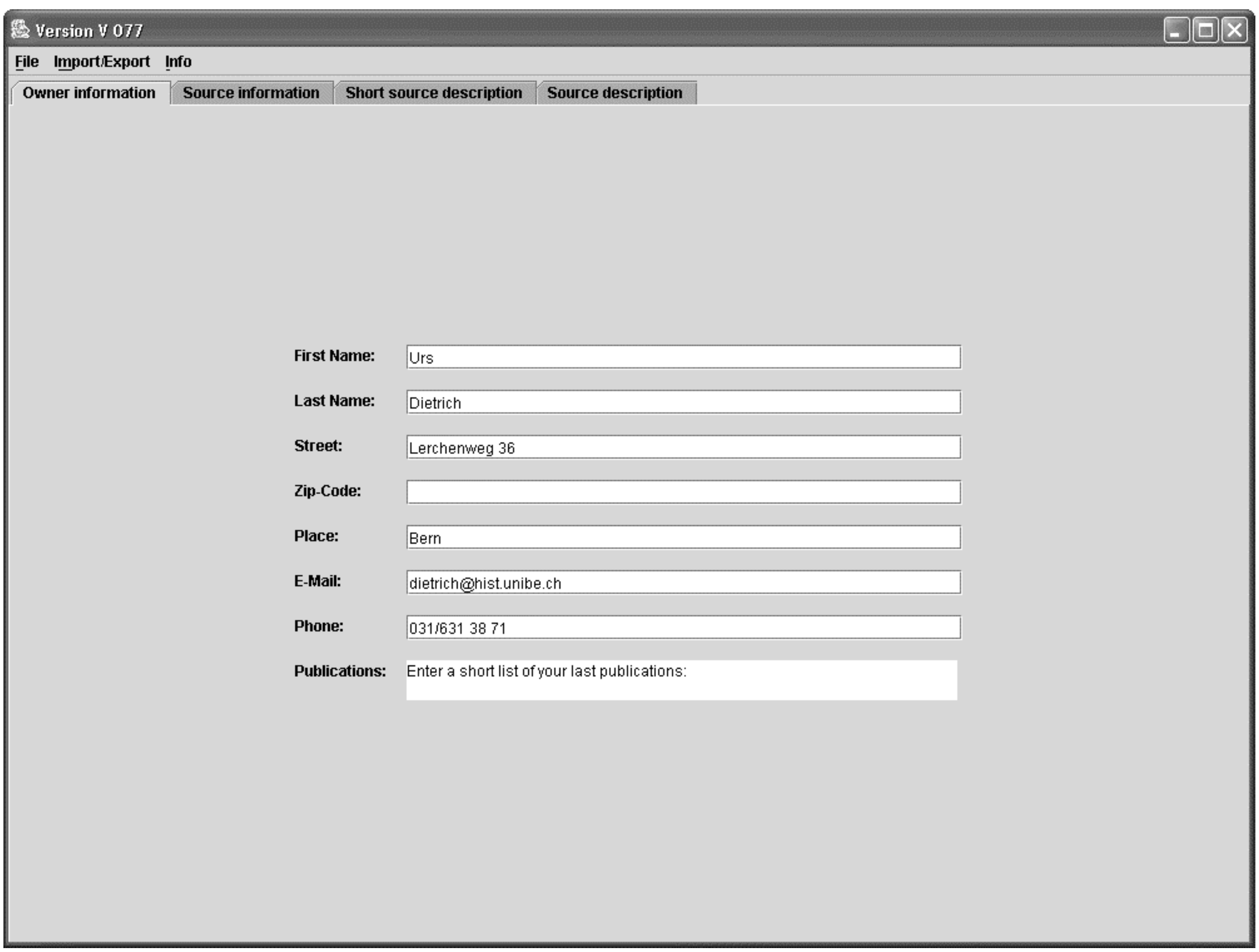

FIGURE 1. Screenshot of tab 1 containing information about owner of source.

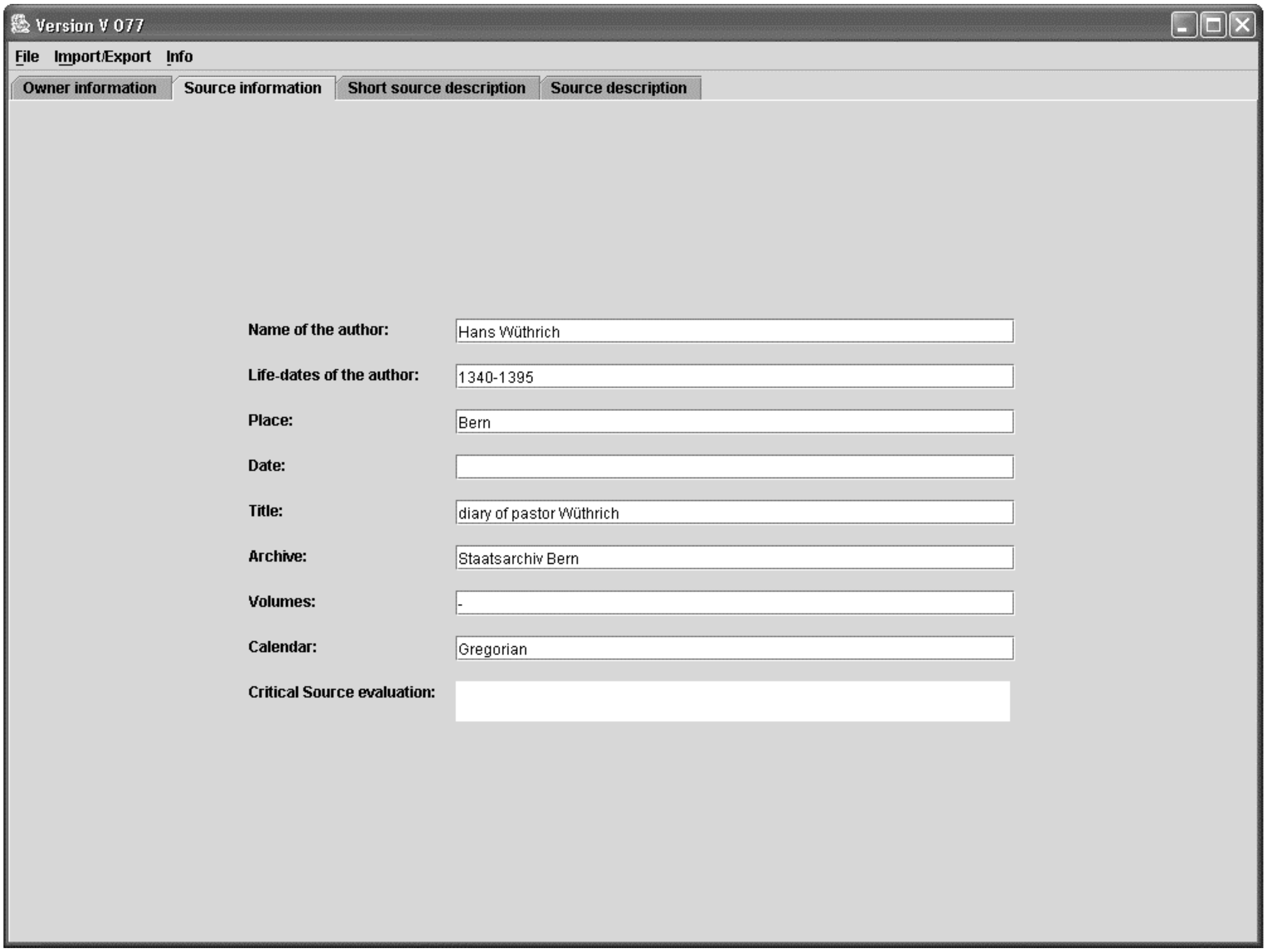

FIGURE 2. Screenshot of tab 2 dealing with source information and critical evaluation of source.

shown in fig. 3) is an optional entry and allows personal comments to be added concerning the source.

The area on the right of tab 3 holds two trees—a place tree and an observation tree—that cannot be changed by the user. Coding an entry means that the user fills in the date, firm, and pc columns and then merely clicks on the desired leaves of the place and observation trees. The entry will be overtaken in the active row. The user can add new records by clicking the "Add new record" button at the bottom of the window. The database administrator must be provided with notification of places or observations that are not available but nonetheless desired. The user can change the size of the two major areas by using the vertical line called SplitPane.

The workplace on tab 4 (see fig. 4) is also separated into two areas. The left side is a text component; the right side contains the well-known trees and a list box. In contrast with tab 3, the editor field can host the entire transcribed source. Inasmuch as a completely transcribed source is much more useful for a historian, tab 4 is preferred for coding the source. Tab 3 is solely available because some sources (e.g., weather diaries) contain only lists with no additional text. Furthermore, it is necessary to have a tabulated view because some sources are too long and a historian cannot always take the time to transcribe the entire text.

Working in the preferred mode, the historian transcribes the entire source directly into the text field on tab 4, or uses an editor such as WordPad to do so. If the historian uses an external editor, he or she can import the transcribed source with the Import/Export menu. This routine allows the import of unformatted text.

The next step in the process is to identify logical units in the transcribed text and to increase those units with standardized metadata. Such a logical unit in this context contains a date or a description of a date, respectively, an observation (or several), and place descriptions. The historian marks the data one by one as a record by selecting the "record" item from the list box after such a unit has been highlighted. The marked passage will be enlarged with a beginning "<s_record>" and an ending "</s_record>" known as XML tags (McLaughlin 2001).

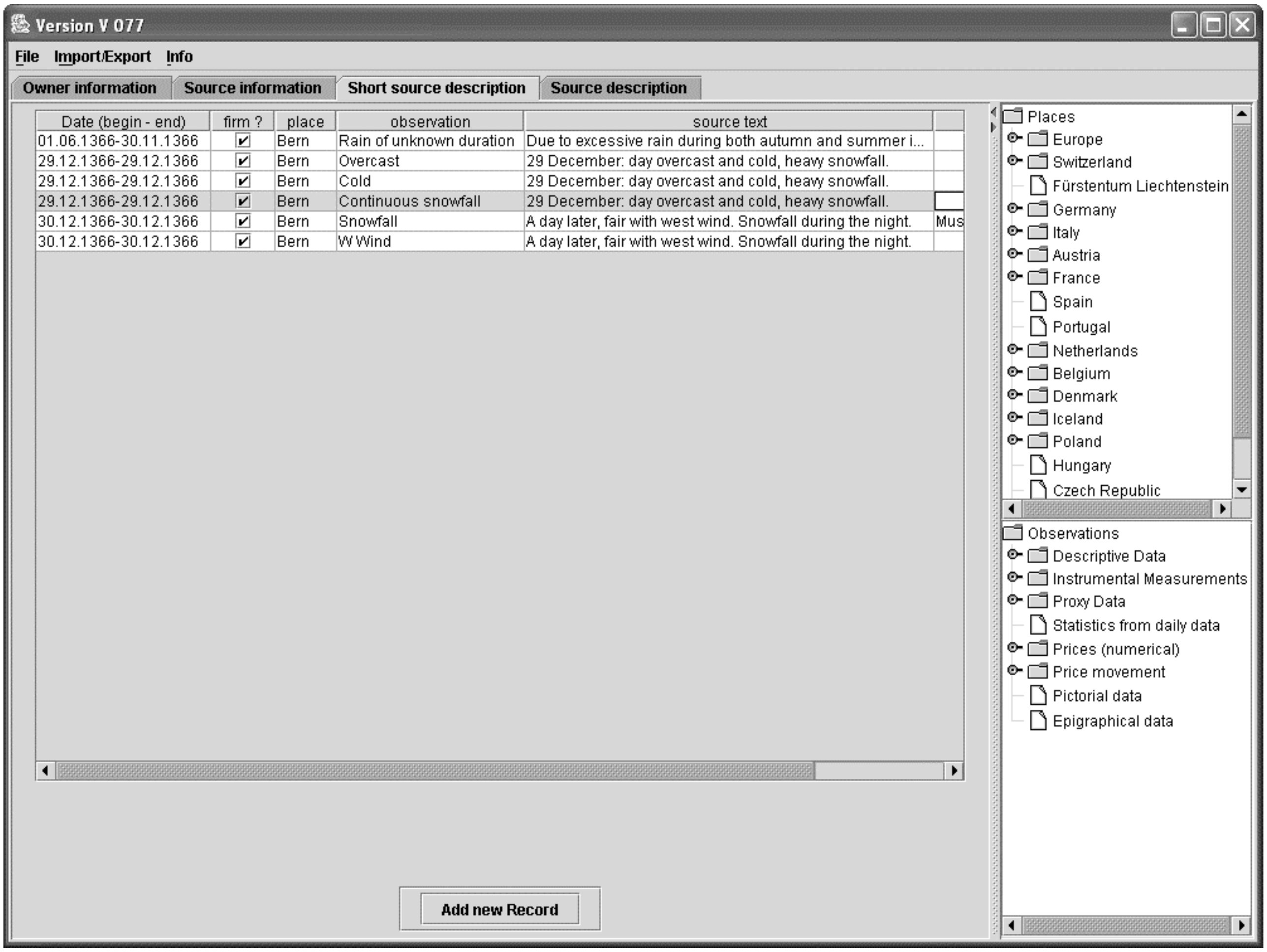

FIGURE 3. Tab 3 and its two major areas, data sheet on left and two trees for places and observations on right. Button for adding new records appears at bottom of window.

Example—Step 1

The following lines show a purely fictional source with just four phrases.

> Due to excessive rain during both autumn and summer in Bern, the year of 1366 was very difficult for farming. 29 December: day overcast and cold, heavy snowfall. A day later, fair with west wind. Snowfall during the night.

The first phrase provides information about the place where the observer lives and the year of the observation. In the second phrase, another day is described, but it appears to be in the same year at the same place with only differing observations. The third phrase, an observation without a place description, has a descriptive value with respect to the date, although it is only in the context of the second phrase. The fourth phrase is the resumption of phrase 3, albeit without specific reference to the date. Therefore, three logical units are separated by date information. After the records are identified, marked, and enlarged as already described, the source appears as follows:

> <s_record>Due to excessive rain during both autumn and summer in Bern, the year of 1366 was very difficult for farming.</s_record><s_record>29 December: day overcast and cold, heavy snowfall.</s_record><s_record>A day later, fair with west wind. Snowfall during the night.</s_record>

Example—Step 2

Now, the user identifies elements such as date, observation, or place fragments. When assigning an identification to a date, the user first highlights the date. Thereafter, he or she selects the item "date" from the list box above the two trees. The source identification then looks like the following (line feeds have no meaning):

> <s_record>Due to excessive rain during both autumn and summer in Bern, the year of <d><sd>1366</sd><nd></nd></d> was very difficult for farming.</s_record>
>
> <s_record><d><sd>29 December</sd><nd></nd></d>: day overcast and cold, heavy snowfall.</s_record>

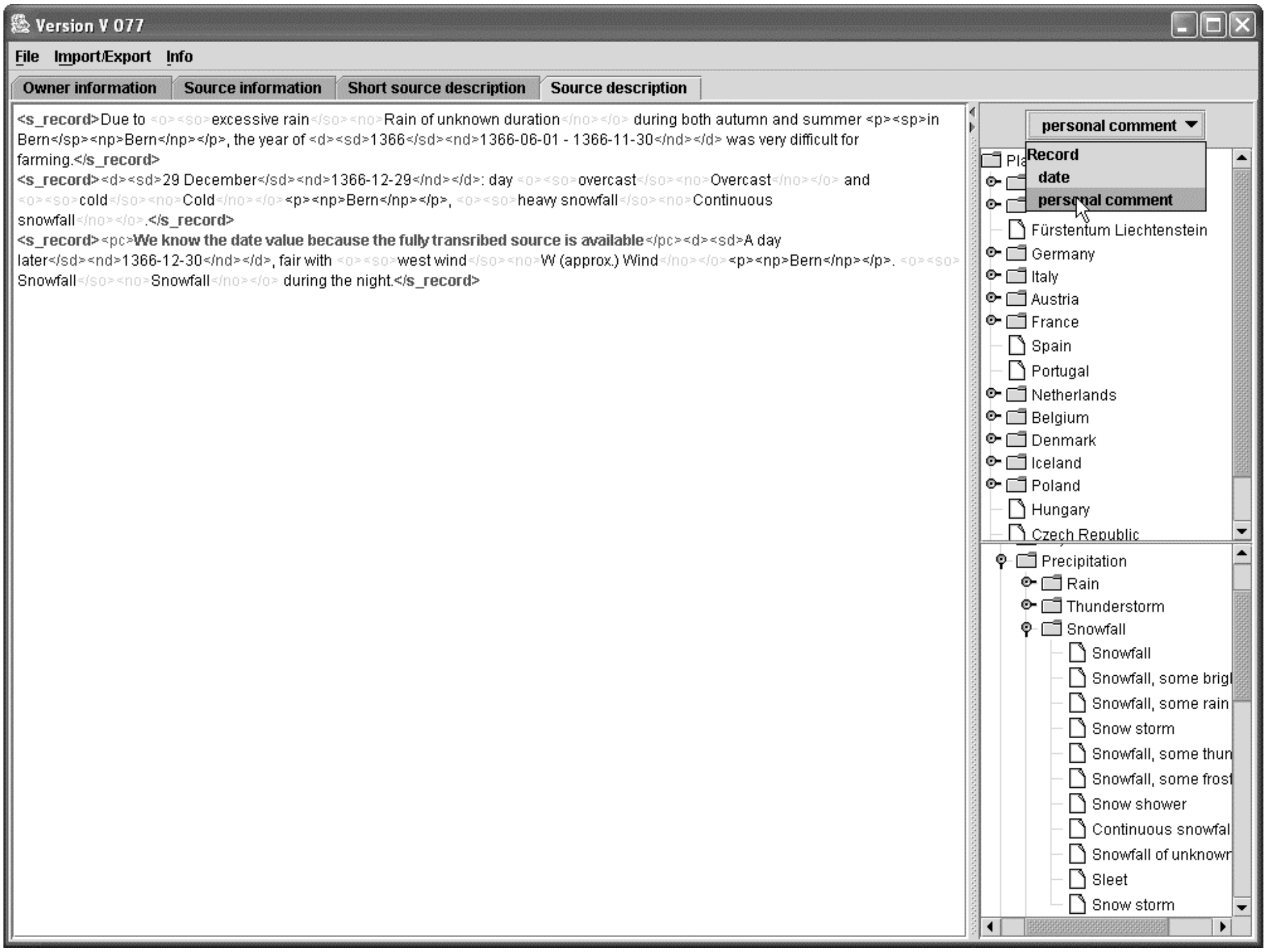

FIGURE 4. Tab 4 with editor field, two trees, SplitPane, and list box at top of area on right. Even though source text is written in German or Latin, for instance, all normalized places and observations taken from trees are in English.

<s_record><d><sd>A day later</sd><nd></nd></d>, fair with west wind. Snowfall during the night.</s_record>

A "d" stands for "date," and the two elements <d> and </d> are the outer brackets for the entire statement concerning the date of the source. The first of the two inner pairs of brackets, <sd> </sd>, identifies the date as a fragment of the source text as it appears in the source. The inner tag <sd> stands for source date. The second record contains the information concerning the specific day, "29 December," and the third record, "a day later." The tag <nd>—the abbreviation of normalized date—remains empty thus far and refers to the date value as written in the present dating system. The historian now has to fill in these empty elements with a date or a date range in Gregorian style (see fig. 5).

<s_record>Due to excessive rain during both autumn and summer in Bern, the year of <d><sd>1366</sd><nd>1366/06/01-1366/11/30</nd></d> was very difficult for farming.</s_record>

<s_record><d><sd>29 December</sd><nd>1366/12/29</nd></d>: day overcast and cold, heavy snowfall.</s_record>

<s_record><d><sd>A day later</sd><nd>1366/12/30</nd></d>, fair with west wind. Snowfall during the night.</s_record>

Particular mention is made of the fact that the first record contains a date range because the detailed observations took place during summer and autumn (between July and November) of the specifically stated year.

Example—Step 3

The historian will find that assigning identification and enhancing elements of place or observation is easier. The historian highlights the description of the place, if known, and clicks on the appropriate node of the corresponding tree. The value "in Bern" then looks like "<p><sp>in Bern</sp><np>Bern</np></p>." The tags <p> and </p>

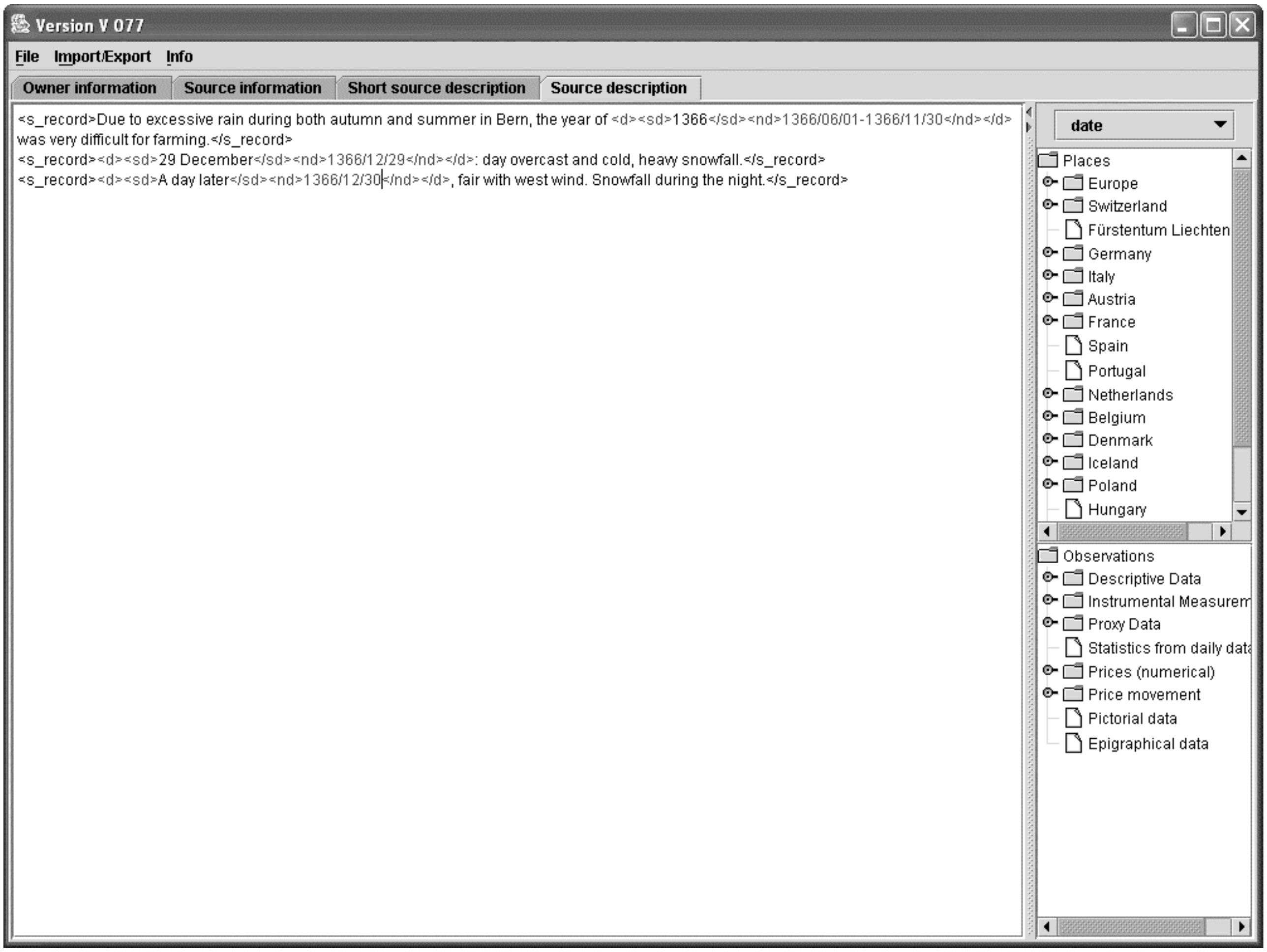

FIGURE 5. Source after assigning an identification to records and establishing date values.

encapsulate a place, <sp> and </sp> give the source place as written in the original source text, and the pair <np> and </np> stands for "normalized place." If no place description is stated, such as in records 2 and 3, the historian simply sets the cursor between the two words and selects the proper item from the place tree. Figure 6 shows the source with the final record structure, as enhanced with information concerning date and place. Red is the color for elements concerning dates, blue is the color for elements concerning places, and yellow is the color for elements concerning observations. Note that the "brackets" <sp> and </sp> are missing in records 2 and 3 because no specific places are stated.

To mark text passages concerning observations, the user follows the same procedure as for elements of place. Instead of using the terms <p> and </p>, he or she encapsulates observations with <o> and </o>, respectively, with <so> </so> for displaying source observation values, and <no> </no> for "normalized observation" values. As already stated, one can use the tag <pc> </pc> (personal comment) at every position inside the tags <s_record> </s_record>. The user just types in the personal comment, highlights it, and chooses the item "personal comment" from the list box (see fig. 7).

This process is very similar to working with a paper version of the source in which some passages are highlighted. As opposed to using the highlighted portions, the researcher enhances the text with XML tags and additional information, such as homogenized place values and information concerning observation.

After saving the file as an XML file (e.g., test.xml), the owner can export the file in two different modes. The mode "export the source text" from the menu bar "import/export" exports the complete source text from tab 4 into a text file without any XML tags and without any additional information. The output is exactly the same as it would be if the owner transcribed it in an editor such as WordPad. The other export option, "export for the database," is more useful and creates five different text files. Four of those files contain the very same information as

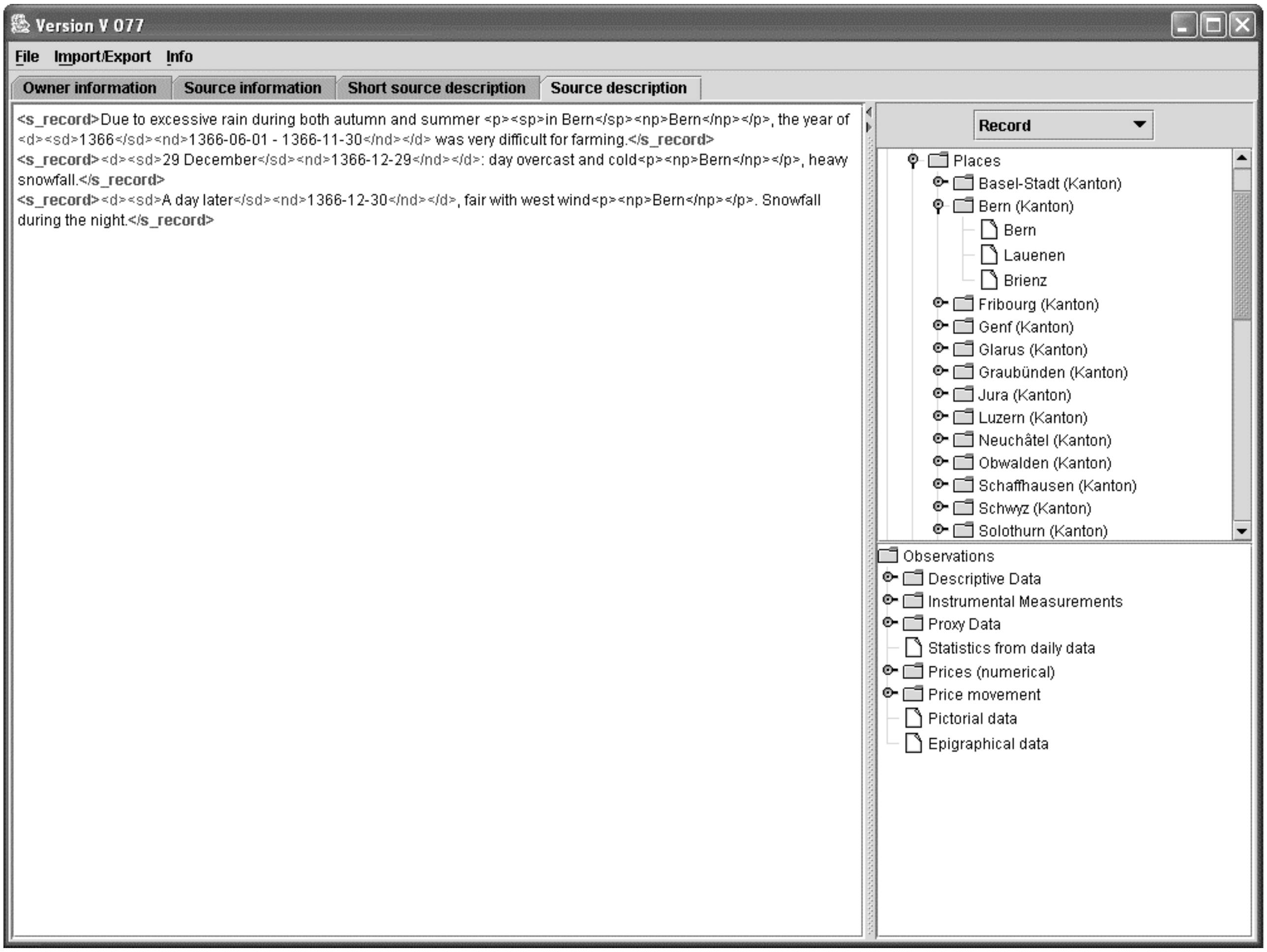

FIGURE 6. Source enhanced with XML tags for date and place values.

one of the individual tabs—a file with the owner information, a file with the source information, and so on. File 5 contains the source text from tab 4, cleaned from the XML tags. All files use tabulators as delimiters, thereby allowing the import in applications such as SPSS, Excel, or, in the present case with an additional routine, into the new EuroClimHist database.

From the Archive into the Database

When the database administrator receives an XML file from an owner, the data are processed in two steps. Step 1 simply tests if the structure of the data file is correct. For this test, the user opens the file with Internet Explorer or another tool that allows syntax checks against a document type definition (DTD). Such a DTD is like a construction plan and specifies the structure and the allowed elements of the XML data files. When one of these data files is opened with Internet Explorer, the program itself searches for the DTD and compares this construction plan with the present file. The following section shows the simple DTD of the EuroClimHist data files:

<!ELEMENT EuroClimHist (personal_information, source_information, number_of_lines, source_table, source_complete?)>
<!ELEMENT personal_information (last_name, first_name, street, Zip_Code, place, email*, phone*, publications*)>
<!ELEMENT last_name (#PCDATA)>
<!ELEMENT first_name (#PCDATA)>
<!ELEMENT street (#PCDATA)>
<!ELEMENT Zip_Code (#PCDATA)>
<!ELEMENT place (#PCDATA)>
<!ELEMENT email (#PCDATA)>
<!ELEMENT phone (#PCDATA)>
<!ELEMENT publications (#PCDATA)>

<!ELEMENT source_information (author, lifetime?, place_of_observation?, time_of_observation?, title, archive?, number_of_volumes?, calendar, critical_eval_of_source)>
<!ELEMENT author (#PCDATA)>
<!ATTLIST author secured (secured | unsecured | unknown) "unknown">
<!ELEMENT lifetime (#PCDATA)>

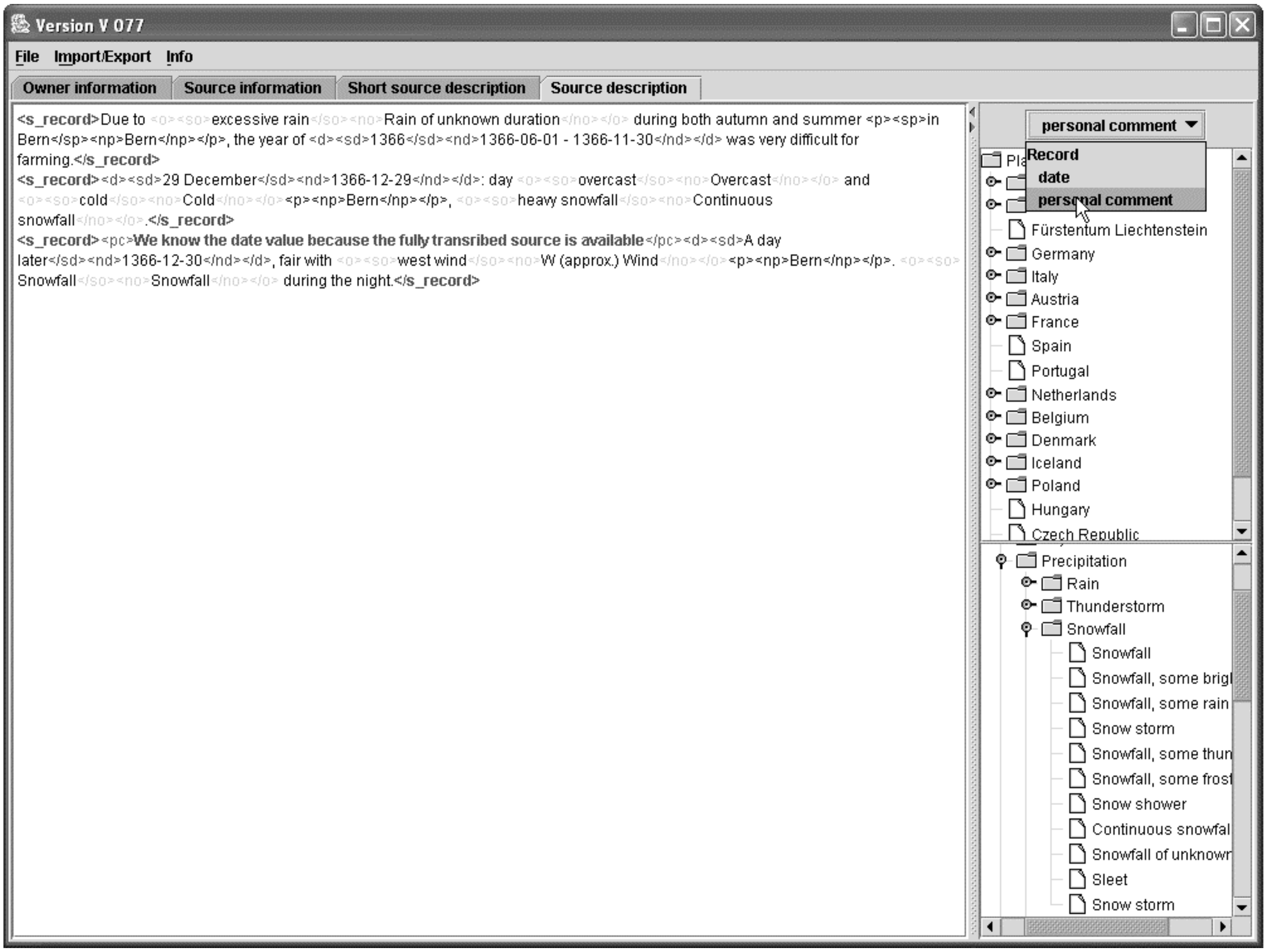

FIGURE 7. Example with fully prepared source.

```
<!ELEMENT place_of_observation (#PCDATA)>
<!ELEMENT time_of_observation (#PCDATA)>
<!ELEMENT title (#PCDATA)>
<!ELEMENT archive (#PCDATA)>
<!ELEMENT number_of_volumes (#PCDATA)>
<!ELEMENT calendar (#PCDATA)>
<!ELEMENT critical_eval_of_source (#PCDATA)>

<!ELEMENT number_of_lines (#PCDATA)>

<!ELEMENT source_table (t_record)*>
<!ELEMENT t_record (t_date, t_firm, t_place, t_observation,
t_source, t_pc)*>
<!ELEMENT t_date (#PCDATA)>
<!ELEMENT t_firm (#PCDATA)>
<!ELEMENT t_place (#PCDATA)>
<!ELEMENT t_observation (#PCDATA)>
<!ELEMENT t_source (#PCDATA)>
<!ELEMENT t_pc (#PCDATA)>

<!ELEMENT source_complete (#PCDATA|s_record)*>
<!ELEMENT s_record (#PCDATA|d|p|o|pc)*>
<!ELEMENT p (sp?, np)*>
<!ELEMENT np (#PCDATA)>
<!ELEMENT sp (#PCDATA)>
<!ELEMENT o (so?, no)*>
<!ELEMENT no (#PCDATA)>
<!ELEMENT so (#PCDATA)>
<!ELEMENT d (sd?, nd)*>
<!ELEMENT nd (#PCDATA)>
<!ELEMENT sd (#PCDATA)>
<!ELEMENT pc (#PCDATA)>
```

Figure 8 shows the example of a screenshot of the Internet Explorer and the opened test file. Line 1 specifies the character set; in this case, it is a Unicode encoding (UTF-16, which is a 16-bit encoding). Line 3 is an XML comment and shows the last point in time that the file was saved. The following sections—that is, <personal_information>, <source_information>, <source_table>, and <source_complete> (the last section is not shown in the figure)—correspond to the four tabs of the data tool. A plus sign on the left side of the screen allows an expansion of the lower nodes of the tree, and by clicking on the minus sign the researcher makes the underlying nodes disappear. (Fig. 8 shows the tree fully expanded; i.e., minus signs, not plus signs, are shown.) The

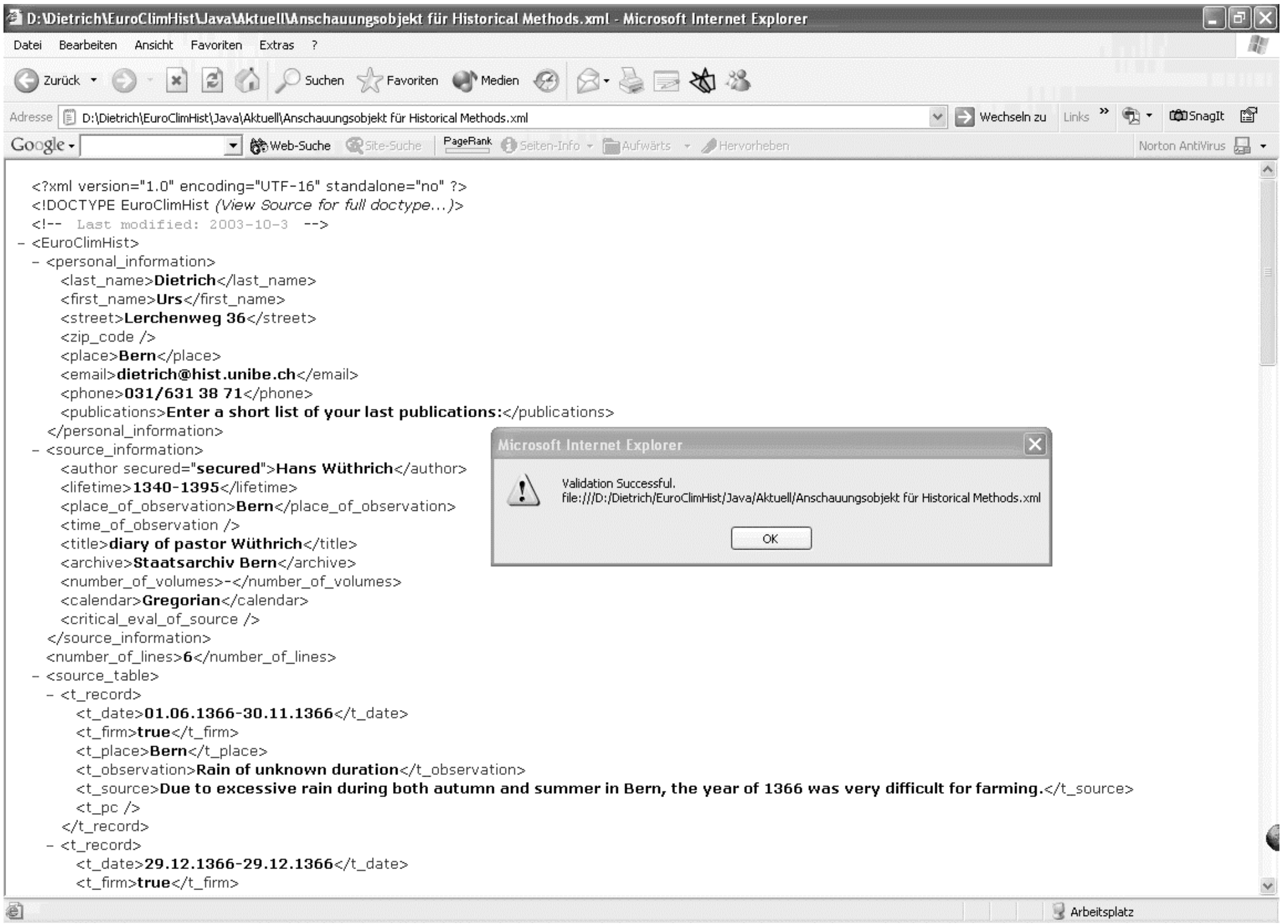

FIGURE 8. Internet Explorer with opened XML test file and pop-up window showing "validation successful" message.

message box in the center of the screen confirms the successful validation of the file against the EuroClimHist DTD. If the validation is successful, the database administrator is thus assured that the syntax of the prepared source is correct and the data can be imported into the database.

Conclusions

The part of the Palvarex data flow shown herein solves all the problems of the other solutions previously discussed. Java-based programs run on any platform; the user's selection of computer system is irrelevant. Storing the data in XML files allows the historian to use any symbol or special character from any language, because XML supports Unicode. With a DTD as a set of rules in the background of the data-gathering process, the tool validates the structure of a prepared document. Data exchange and additional use in other contexts are not a problem because many technologies for processing and transforming XML files are available. The time needed to learn how to use the data tool is short; the GUI and the common use of the mouse make the program intuitive. Finally, use of the data tool can be made in other contexts even if programming skills in Java are less than optimal. For example, in the field of the history of medicine, a project researcher needs data concerning specific diseases, the number of the deceased, and the places of their death. Only minor changes are necessary for the adaptation: (a) the DTD must be changed; (b) possible additions or changes may be made in the place tree; and (c) the observation tree must be exchanged for a new "diseases" tree. Programming skills are necessary only for (d) the adaptation of the items in the list box—for example, "date" can be changed to "deceased"; and (e) for the adaptation of the XML tags within the Java program and the output routine.

NOTES

1. SQL is a standardized language that allows users to define the data in a database and manipulate that data.

2. Because computers understand only numerals, ASCII is a code for changing letters to numerals. ASCII is used to describe files stored in clear text format. ASCII text is the simplest form of text because no single document includes any kind of format, and the text is neither platform nor application specific. It uses a 7-bit code to represent characters with

numerals (0 to 127), although ASCII extensions using 8-bit codes may be used to represent international characters not included in the standard ASCII scheme.

3. VMS is an operating system made by Digital Equipment Corporation (DEC), which runs on the ALPHA processor. Compaq, recently merged with Hewlett Packard (HP), bought out DEC and is now the seller and developer of OpenVMS and the ALPHA processor.

4. This is an easy-to-use system for storing simple, structured, or random data and finding it again without programming. It is often used for unstructured data. See http://www.asksam.com/products.asp.

5. See http://www.nccr-climate.unibe.ch/download/NCCR_Climate_FINAL_Proposal.pdf.

6. The NCCR is a research instrument of the Swiss National Science Foundation.

7. Fundamentally, computers deal only with numbers. They store letters and other characters by assigning a number for each one. Before Unicode was invented, there were hundreds of different coding systems for assigning those numbers. No single encoding contained enough characters. The European Union alone, for example, requires several different crosslingual on-demand information extraction (Codie) systems to cover all its languages. Even for a single language such as English, no single system was adequate for all the letters, punctuation, and technical symbols in common use. Unicode is the successor of ASCII; it provides a unique number for every character, regardless of the platform, program, or language.

8. See http://java.sun.com/ for further information about Java, including location of software development kits, documentation, tutorials, code samples, and the like.

REFERENCES

Hagnauer, S., and N. Bartlome. 1998. Vom Quellentext zur Datenbank—ein Konzept zur integrierten Verarbeitung quantitativer und qualitativer Daten. *Geschichte und Informatik* 9:141–58.

McLaughlin, B. 2001. *Java & XML*. Cambridge: O'Reilly.

Pfister, C., R. Brázdil, and M. Barriendos. 2002. Reconstructing past climate and natural disasters in Europe using documentary evidence. Documentary data on weather and climate (http://www.pages.unibe.ch/products/newsletters/nl2002_3.pdf).