Visual exploration of temporal object databases*

Chaouki Daassi[†‡], Marlon Dumas[†], Marie-Christine Fauvet[†],
Laurence Nigay[‡] and Pierre-Claude Scholl[†]

[†] LSR-IMAG
BP 72
38402 St-Martin d'Hères (France)
{ *Firstname.Lastname* }@imag.fr

[‡] CLIPS-IMAG
BP 53
38041 Grenoble Cedex 9 (France)
{ *Firstname.Lastname* }@imag.fr

Abstract

Two complementary families of users' tasks may be identified during database visualization: data browsing and data analysis. On the one hand, data browsing involves extensively exploring a subset of the database using navigational interaction techniques. Classical object database browsers provide means for navigating within a collection of objects and amongst objects by way of their relationships. In temporal object databases, these techniques are not sufficient to adequately support time-related tasks, such as studying a snapshot of a collection of objects at a given instant, or detecting changes within temporal attributes and relationships. Visual data analysis on the other hand, is dedicated to the extraction of valuable knowledge by exploiting the human visual perception capabilities. In temporal databases, examples of data analysis tasks include observing the layout of a history, detecting regularities and trends, and comparing the evolution of the values taken by two or more histories. In this paper, we identify several users' tasks related to temporal database exploration, and we propose three novel visualization techniques addressing them. The first of them is dedicated to temporal object browsing, while the two others are oriented towards the analysis of quantitative histories. All three techniques are shown to satisfy several ergonomic properties.

Keywords: temporal database, object database, data browsing and analysis, visualization technique.

1 Introduction

Effectively assisting the exploitation of large databases by users with various profiles and tasks, is an open challenge for both the domain of Human Computer Interaction and that of Databases. One way of addressing this issue is to design visual exploration techniques which maximize both the amount and the readability of the information perceivable by the user, and which satisfy the basic ergonomic properties of interactive software [GC96].

Three main tasks performed by users with different expertises may be identified in the context of a database: schema definition, query specification, and analysis of query outputs. Correspondingly, three

*Published in proceedings of BDA'00, October 2000, Blois (France)

kinds of visual interfaces have been designed and are provided by most commercial DBMS: schema browsers and editors, visual query languages, and data visualization tools (see [CCLB97] for a survey).

Among data visualization tools we distinguish:

- Data browsers [MDT88, PBL+92, DGJS95, DAA+95, CHMW96] which rely on simple interactional metaphors such as tables and forms.
- Visual data analysis tools, which rely on more sophisticated interactional metaphors that are dedicated to knowledge extraction for a particular application domain [Kei97, CK98, SC98, FNPS99].

In this paper, we propose a data browser and two visual data analysis tools for temporal object databases, i.e. databases which keep track of the evolution of object states and relationships. These tools are designed according to a list of users tasks and several well-known ergonomic properties.

1.1 Temporal object databases

A temporal object database is defined as a database whose schema includes *temporal classes and properties*. A class is temporal if at least one of its properties is temporal. A property is temporal, if its value denotes the evolution of an object characteristic over a period of time. The value of a temporal property is called a history. For the sake of simplicity, we define a history as a collection of instant-timestamped or interval-timestamped data items, although there are many other ways of representing a history [FDS99].

```
typedef struct { Date lb; Date ub; } Interval;
typedef struct { Supervisor value; Interval timestamp; } TimestampedSupervisor;
typedef struct { set<Worker> value; Interval timestamp; } TimestampedWorkerSet;
typedef struct { long value; Date timestamp; } TimestampedProduction;
typedef struct { float value; Interval timestamp; } TimestampedWage;
typedef struct { AssemblyLine value; Interval timestamp; } TimestampedAssemblyLine;
typedef struct { set<AssemblyLine> value; Interval timestamp; } TimestampedAssemblyLineSet;
class AssemblyLine (extent AssemblyLine, key lineNumber) {
    attribute string lineNumber;
    attribute set<TimestampedSupervisor> supervisor;
    attribute set<TimestampedWorkerSet> workers;
    attribute set<TimestampedProduction> production;
}
class Employee (extent Employees, key name) {
    attribute string name; attribute set<TimestampeWage> wage;
}
class Worker extends Employee (extent Workers) {
    attribute set<TimestampedAssemblyLine> worksIn;
}
class Supervisor extends Employee (extent Supervisors) {
    attribute set<TimestampedAssemblyLineSet> supervises;
}
```

Figure 1: ODMG schema of a temporal database.

Figure 1 presents the schema of the example temporal database used throughout the paper. This database tracks the evolution of the productivity parameters, supervisors and workers of each assembly

line of a factory. In addition, it tracks the evolution of the salary of each factory employee (whether a worker or a supervisor), as well as the history of the assembly line to which each of them is assigned.

All classes in this schema are temporal. The history of property AssemblyLine::production is represented as an instant-timestamped collection of objects, while that of property Employee::wage is represented using intervals. All histories are observed at the granularity of the day.

1.2 Addressed users' tasks

Within a temporal database, time is orthogonal to the other components of an object. Therefore data visualization tools in this setting, must support time-dependent navigational tasks. In this paper, we specifically address two of them :

- Exploring the snapshots of a set of related temporal objects at a fixed point in time.
- Analyzing quantitative histories (i.e. histories of numbers) with the goal of detecting regularities and trends within their evolution.

To support the first task, we propose an interactive visualization technique called *pointwise temporal object browsing*. The basic idea of this technique is to display a snapshot of a temporal object at a given instant. The user interacts with this representation, either by navigating through object relationships as in classical object browsers (e.g. PESTO [CHMW96]), or by modifying the reference instant. The resulting interface stresses simultaneity by focusing on one instant at a time. It also supports the exploration of value changes, by providing interaction devices for "jumping" to the next/previous instant where a change occurs in a given visualized path expression.

To support the second task, we develop two techniques for mining regularities, concentrations and trends within a given history. The first of them, namely Concentric Circles, consists of a set of circles with different radii and a common center. Each circle represents one particular period of the visualized history. Data items are represented through graphical objects whose size convey their values. This technique allows the user to identify periodic patterns and to retrieve correlations between synchronous values of two quantitative histories. The second technique, namely Agenda, is a matrix: each line represents the evolution of a history over a particular period. This technique mainly allows the user to identify hot spots, though it may also be used to observe the layout of a history and to detect periodic patterns.

1.3 Ergonomic properties

An ergonomic property is a feature of a human-computer interface which is chosen to be the subject of analysis and evaluation. Ergonomic properties are neither necessarily good nor bad features; they simply serve to define an absolute design space by pruning off several design possibilities.

The visualization techniques that we propose, are designed so as to fulfill the ergonomic properties described below. Most of these properties are extensively discussed in [GC96].

- Representation multiplicity: the interface must be flexible in the rendering of state elements. For instance, an interface to a system tracking the evolution of the temperature of a room over time should support alternative representations of this evolution (e.g. as a thermometer and a time-slider if the exact value at each instant is important, or as a graph if it is important to detect trends).
- Input/output re-use: it should be possible to articulate an input or output expression by referring to previous ones. Cut, paste and copy commands are typical examples of I/O re-use.
- Non-preemptiveness: whenever it is possible, the user should be free in deciding what is the next action to perform.
- Observability: the system should allow users to inspect all information relevant to their tasks.
- Insistence: the relevant information should be the only one available at the interface, even if it is not necessarily the case that the user will notice it. Additionally, the interface should ensure that critical information is not only available, but is actually perceived by the user.
- Honesty: the interface should guarantee that the user correctly interprets the graphical objects and symbols.
- Screen stability: the transition between one state of the interface and the next one should be smooth, and the differences between the old and the new states should be easily perceivable.
- Spatial continuity: gaps between related graphical objects should be avoided, unless they are either significant (i.e. they convey a meaning), or necessary to perceive the boarders of each object.

The strict adherence to the above properties is one of the main originalities of our work with respect to related ones.

1.4 Paper structure

The rest of the paper is structured as follows. In section 2, we present the temporal pointwise browsing technique. Next, in section 3, we describe the concentric circles and the agenda techniques. Throughout both of these sections, we discuss how the proposed techniques address the users' tasks and fulfill the ergonomic properties described above. Lastly, we compare our three techniques to existing ones in section 4, before drawing our conclusions in section 5.

2 Pointwise temporal object browsing

2.1 Overview

The pointwise browser interface (see figure 2) is made up of two parts: a *time-line window* and a tree of *snapshot windows*. A snapshot window displays either a non-temporal object[1] or a snapshot of a temporal object at a given instant. The instant with respect to which the object snapshots are determined

[1] An object is *temporal* if it owns at least one temporal property and *non-temporal* otherwise.

is the same for all the windows in the tree, and is subsequently called the *reference instant*. The reference instant is constrained to reside within a given interval called the *temporal browsing range*.

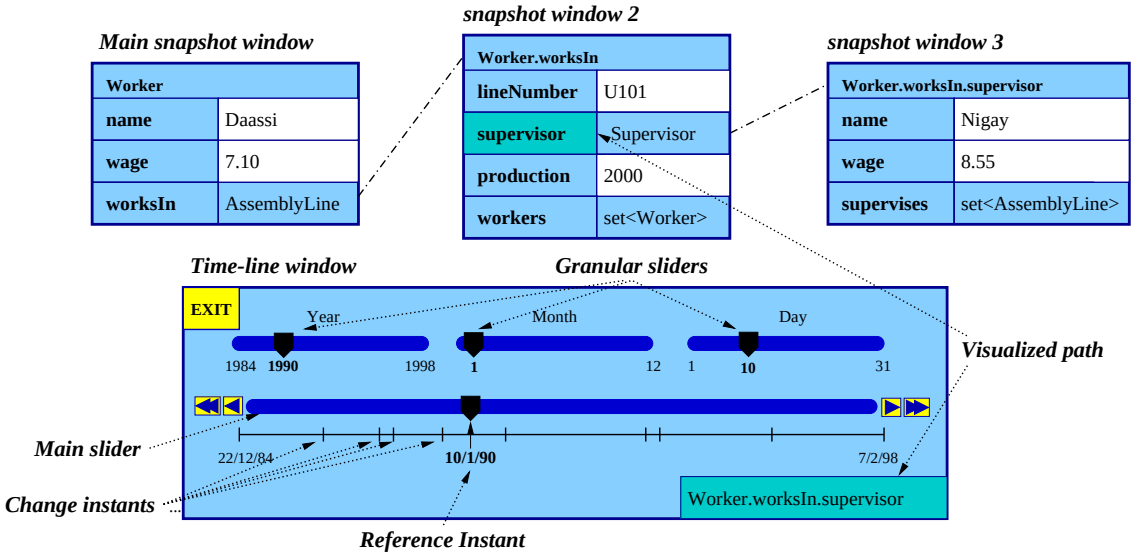


Figure 2: The pointwise temporal object browser. The schema of the underlying database is the one given in figure 1. In particular, properties Worker::wage and Worker::worksIn are temporal.

2.1.1 The time-line window

The role of the time-line window is to fix the reference instant. At the beginning of a session, the reference instant is at the middle of the temporal browsing range. Its position varies thereafter according to the user interactions with the sliders and buttons composing the time-line window. In its simplest form, the time-line window is composed of a slider (called the *main slider*) and four buttons placed at the ends of this slider. Two of the buttons (labeled by simple arrows), allow the user to move the reference instant forward or backward by one unit. The other pair of buttons (labeled with double-arrows), allow to move the reference instant to the next/previous instant where the value of a given navigation path (called the *visualized path*) changes. The instants at which the value of the visualized path changes are called *change instants*. Change instants are visually represented as vertical marks lying within a horizontal line just beneath the main slider.

In figure 2, the change instants are those when the supervisor of worker “Daassi” changes, whether this change is due to the fact that this worker is assigned to a new assembly line and that this assembly line has a different supervisor than the previous one, or to the fact that the supervisor of the assembly line to which this worker is assigned changes.

In addition to the main slider, the time-line window may additionally contain several *granular sliders*, which allow the user to move the reference instant with different “steps” according to a given calendar. For instance, if the reference instant is a date, and that the user specifies the calendar Year/Month/Day

(as in figure 2), three granular sliders appear in the time-line window: the first one allows one to move the reference instant with a step of a year, the second one with a step of a month, and the third one with a step of a day (within the limits of a given month).

2.1.2 The snapshot windows

Snapshot windows are structured as forms containing one line per property of the visualized object or object snapshot[2]. Each line is composed of two buttons: the left one labeled with the name of the property, and the right one labeled with its value at the reference instant[3]. The value of a non-temporal property is always the same regardless of the reference instant. The value of a temporal property at a given instant is equal to the value of its history at that instant, which is itself defined as follows:

- The value at instant I, of a history represented as an instant-timestamped collection of objects, is equal to the object within this collection whose timestamp is equal to I. If no such object exists, the history's value is null.
- The value at instant I, of a history represented as an interval-timestamped collection of objects, is equal to the object within this collection whose timestamp contains instant I. If no such object exists, the history's value is null.

All buttons within a snapshot window are clickable, except those which denote literal values (i.e. integers, reals, string, and characters). For instance, in figure 2 all the buttons within the snapshot windows are clickable, except the white-colored ones.

At the beginning of a session, there is a single snapshot window. Other snapshot windows are incrementally added to the tree according to the user interactions with the clickable buttons denoting object references which appear within existing forms. The object displayed by a given snapshot window other than the main one, is equal to the object referenced by the button from which this window was opened. For instance, the configuration shown in figure 2 is obtained by displaying the worker named "Daassi" and successively clicking on the buttons labeled AssemblyLine and Supervisor.

The user may also click on the buttons labeled with property names (i.e. the buttons on the left column of a form). The semantics of this interaction is that the selected property becomes the visualized path expression and the set of "change instants" attached to the time-line window are updated accordingly. For instance, clicking on the button labeled production on window 2 of figure 2, sets the visualized path to be Worker.worksIn.production instead of Worker.worksIn.supervisor. The vertical marks drawn on the line just below the main slider, and the label appearing in the low-right corner of the time-line window are then modified accordingly.

[2]For the time being, we restrict our examples to snapshot windows displaying single objects or object snapshots. We will discuss afterwards how collections are accommodated.

[3]If the value of a property is not printable (i.e. its type is not integer, string, etc.), the name of its class is used as its label.

Whenever the user modifies the reference instant, the new reference instant is notified to the main snapshot window. Upon receiving this notification, the main window computes the snapshot at the new reference instant, of the object that it displays, and updates its appearance so as to reflect this new snapshot. During this process, if the value of a temporal property changes, the new value is transmitted to its dependent window if any. Finally, the main window propagates the notification of the new reference instant to all its dependent windows, and the above process is carried out recursively.

Figure 3 shows the result of modifying the reference instant over the example of figure 2.

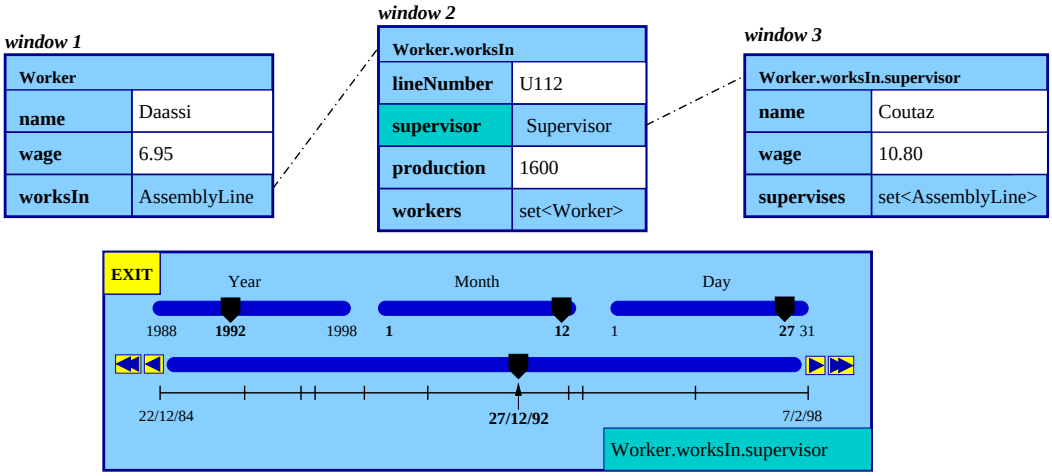


Figure 3: Modification of the reference instant upon the configuration given in figure 2.

2.1.3 The temporal browsing range

As stated before, the reference instant ranges through an interval called the temporal browsing range. In order to fulfill the observability ergonomic property (see section 1.3), this range is taken to be the smallest interval containing all the instants when at least one of the temporal properties of the object displayed by the main snapshot window (subsequentely called the *main object*) is defined. For example, if the main object is a worker W, such that:

W.salary = set(struct(timestamp: [1..4], value: 10.0), struct(timestamp: [6..9], value : 12.0)) and
W.worksIn = set(struct(timestamp: [2..4], value: X), struct(timestamp: [6..8], value : Y)).

(where X and Y are two assembly lines), then the temporal browsing range is taken to be interval [1..9].

Notice that the above definition entails that the main object is temporal, since otherwise, the browsing range would be empty.

2.2 Pointwise browsing in the presence of null-valued properties

Heretofore, we have implicitly assumed that all properties displayed within snapshot windows have non-null values. However, null-valued properties within a snapshot window may arise in two cases:

- The value of the property at the reference instant was actually set to “null” through an update (this can occur whether the property is temporal or not).
- The history of a temporal property is not defined at the reference instant, in which case we consider that its value is null. This situation can occur in the middle of a pointwise browsing session, since the temporal browsing range may include instants in which some of the temporal properties of the main object are defined while others are not.

As in O_2Look [PBL+92], we visually denote a null value through a filled rectangle. However, this does not solve all the problems arising from nulls. Indeed, suppose that the worker displayed in figure 3 is not assigned to any assembly line on 1/5/97 (i.e. there is no element in its history whose timestamp contains this date). If the reference instant is set to this date, the value of property worksIn becomes null, and something has to be done with its dependent forms (i.e. windows 2 and 3 in figure 3).

In our approach, if further a modification of the reference instant, one of the properties displayed by a snapshot window becomes null, and if this property has a snapshot window attached to it, then all the windows in the sub-tree stemming from this property become *inactive*. Inactivity of a snapshot window can be visually rendered in at least two ways:

- Hide the window (and redisplay it when it becomes active again).
- Modify the appearance of some elements within the window, e.g. by graying out the labels denoting property names and erasing the labels denoting property values. In this case, labels should be restored as the window becomes active again. Figure 4 illustrates this approach.

We believe that the second approach is to be preferred, since hiding and redisplaying windows violates the screen stability ergonomic property (see section 1.3).

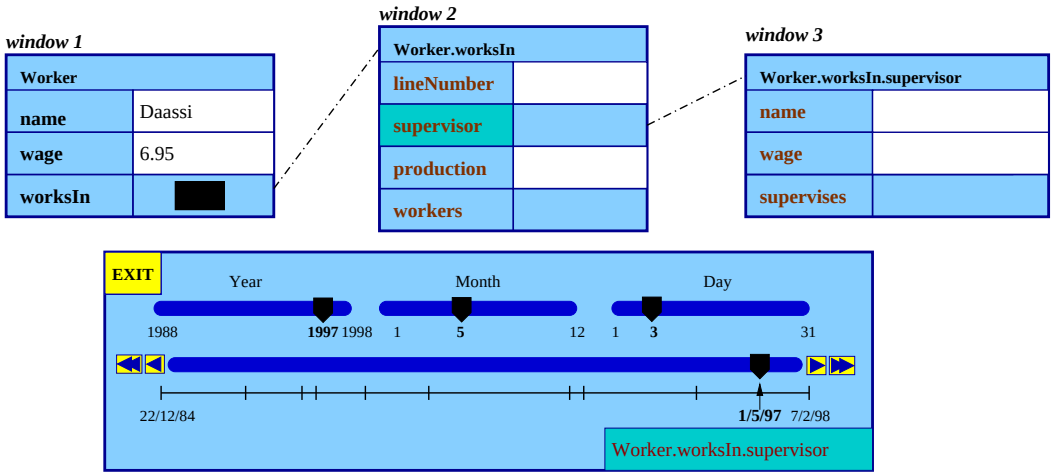
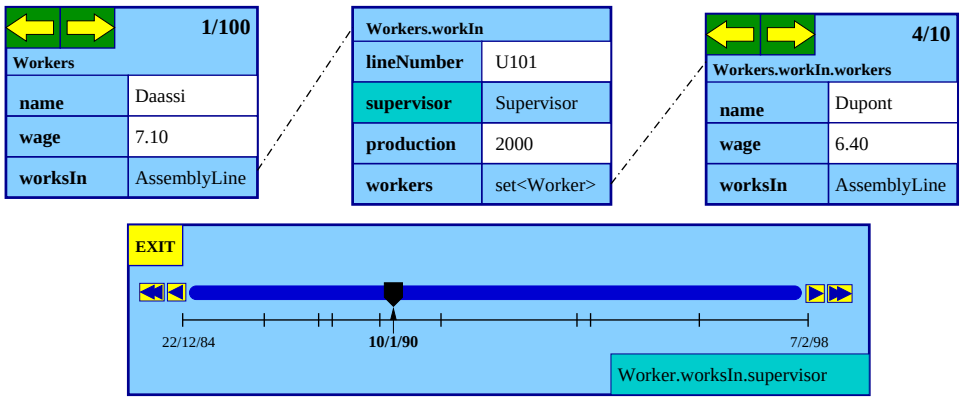


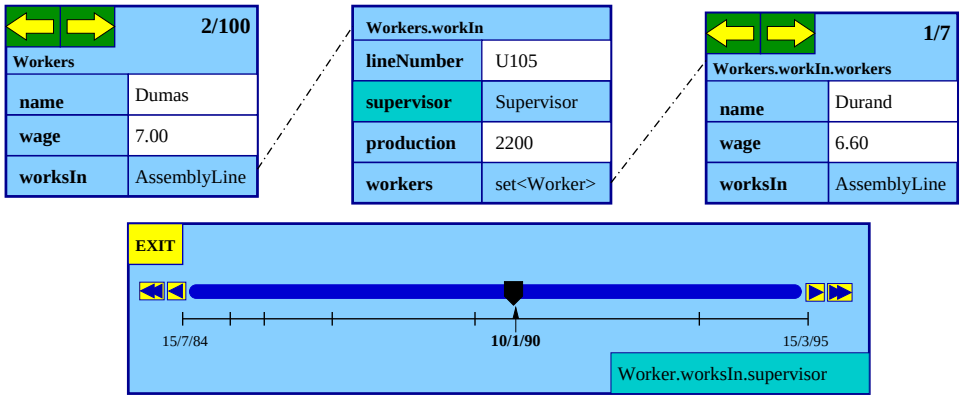
Figure 4: Modification of the reference instant upon the configuration given in figure 3. Windows 2 and 3 become inactive as a result of this interaction.

2.3 Pointwisely browsing collections of temporal objects

To accommodate collections, we augment the pointwise browser with the concept of *synchronous navigation* as defined in object browsers such as PESTO [CHMW96]. Basically, a collection of temporal objects is displayed in the same way as a single one, except that the corresponding snapshot window contains a couple of arrow-labeled buttons on top of it. This window displays the snapshot at the reference instant, of one of the objects within the collection. Clicking on either of the arrows allows one to switch to the next or the previous object in the collection (see figure 5).



(a) Configuration 1



(b) Configuration 2

Figure 5: Pointwisely browsing a collection of temporal objects. Configuration 2 is the result of clicking on the right arrow of the main snapshot window in configuration 1.

As before, the temporal browsing range is defined with respect to the object visualized by the main snapshot window. Therefore, when this object changes, the browsing range is recomputed. This is the reason why the time-line is redrawn when transitioning from configuration 1 to configuration 2 in figure 5. Notice also that during this transition, the change instants are also recomputed. Under some circumstances this computation involves a relatively large amount of data. For instance, consider the example of figure 5 and suppose that the visualized path expression is Workers.worksIn.supervisor.wage

(which means that the path Workers.worksIn.supervisor is displayed), computing the change instants then involves the following histories:

- The history of the employee's assembly lines.
- The history of the supervisors of each line where the visualized employee has ever worked.
- The history of the wages of each supervisor appearing within any of the histories referenced in the previous item.

In order to ensure an acceptable response time when transitioning from one object to another, a good history join algorithm should be used, and the involved histories must be well clustered on disk. Studying these two issues is therefore an interesting perspective to the work reported here.

To conclude, we can state that the pointwise temporal object browser allows one to orthogonally navigate through the following three dimensions :

- Through the objects composing a collection using the arrow-labeled buttons on top of each snapshot window denoting a collection.
- Through object relationships using the clickable buttons within the right columns of a snapshot window, whether this snapshot window denotes a single object or a collection of objects.
- Through the time dimension using the components of the time-line window.

With respect to our example database, the pointwise browser addresses users' tasks such as:

- Analyze data about the supervisor and workers of each assembly line at a given date.
- Compare at different dates, a given worker's wage with respect to that of the supervisor of the assembly line to which he is assigned.
- Find out whether the composition of a given assembly line (equipment plus workers) considerably changes when its supervisor does.

2.4 Implementation notes

We have prototyped the pointwise browsing technique using TEMPOS [FDS99] : an extension of the ODMG standard featuring a set of types modeling temporal values and histories, in addition to an extension of OQL. TEMPOS is implemented on top of the O_2 DBMS as a library of classes and a preprocessor.

Following a widespread approach in the area of interactive software, the prototype is structured into three components: a *functional kernel* responsible of the data and metadata extraction, a *presentation module* which generates the user interfaces, and a *controller* which bridges the above two components.

In the current version of the prototype, the generation of the presentation module is carried out using XForms[4], a toolkit based on Xlib providing a rich set of graphic objects such as texts, buttons, sliders, etc. The choice of this toolkit was exclusively guided by its simplicity, efficiency, and adequacy regarding the design of form-structured windows.

[4] http://world.std.com/~xforms

3 Techniques for quantitative histories visualization

In this section we focus on the visualization of quantitative histories, i.e. histories whose values are numbers. Such histories may either correspond to the value of a temporal property (e.g. the history of the daily productions of a given assembly line), or they may be the output of a temporal query possibly involving data aggregations (e.g. the history of the monthly productions of an assembly line, or the daily total production of all the assembly lines).

Concretely, we describe two novel interactive visualization techniques for this kind of histories, respectively called Agendas and Concentric Circles. Throughout this description, we emphasize the benefits of these techniques according to a list of users' needs, and we justify several design choices according to the ergonomic properties described in section 1.3.

From the user's viewpoint, these techniques are mainly oriented towards retrieving regularities, such as periodical events, value evolution trends, and concentrations of similar values. These tasks were identified during the requirement analysis phase of an application involving a survey on the activities and displacements of individuals in a ski resort over several days [FCD+99]. However, their scope goes far beyond this kind of applications, since time-varying quantitative data are involved in many application domains such as finance, economy and clinical data management, where detecting trends at an early stage leads to more effective decisions.

3.1 Agendas

The interface of the Agenda technique consists of a matrix of colored cells. All the cells have the same color but different intensities so as to fulfill the observability property. In order to fulfill the honesty property, the color's intensity of a cell reflects the value of the visualized history at a given instant: higher intensities correspond to higher values. The exact value of a cell may be displayed by placing the mouse cursor on top of it, as shown in figure 6.

Cells are grouped into rows according to some partitioning of the temporal domain of the visualized history. This partitioning is determined by a granularity. For example, if the granularity of the visualized history is the Month, and that the partitioning is carried out with respect to the granularity Year, then each line consists of twelve cells : one per month in a year. The agenda technique can also be applied when the involved granularities are not mutually regular[5] (e.g. Month and Day). In this case, the visualization area is actually not a perfect matrix, some lines being longer than others.

In any case, two granularities are involved in an agenda: the granularity of the visualized history, and the granularity specified by the user. Accordingly, two bar sliders are included: one for navigating through the columns, and the other for navigating through the rows. In figure 6 , the bar slider associated to the columns (i.e. the horizontal slider) is completely filled, since all the months of the year are present

[5]Two granularities are *mutually regular* if the conversion function between them is entirely described by an integer

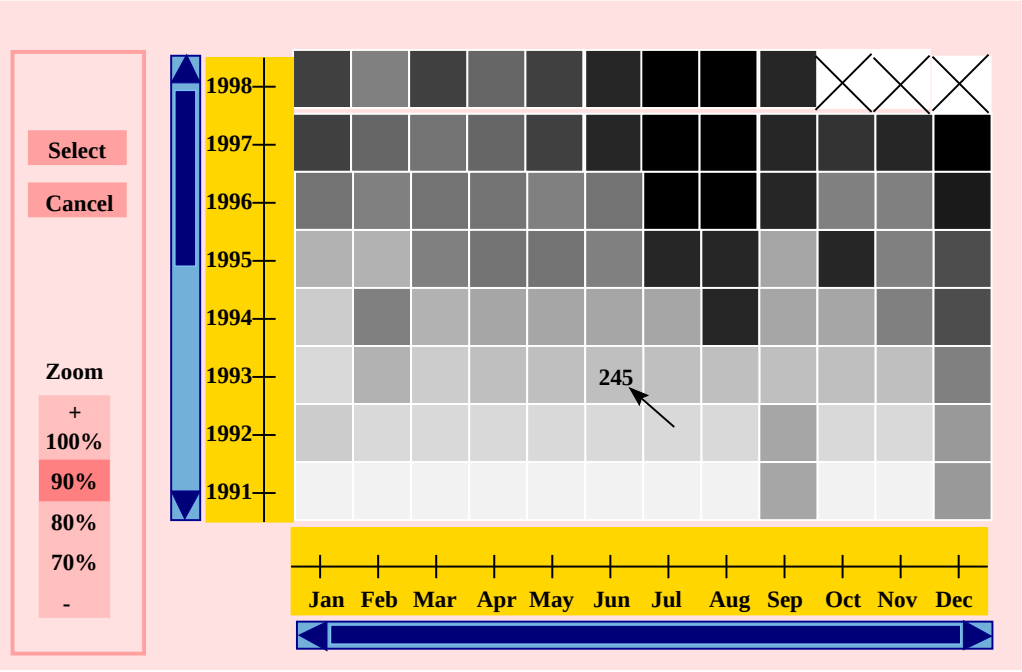


Figure 6: Visualization of the monthly productions of an assembly line using the agenda technique.

in the screen. On the contrary, the bar slider associated to the rows (i.e. the vertical slider) is not completely filled from the bottom, meaning that the temporal domain of the visualized history extends below 1991, but not beyond 1998. Actually, one can deduce from figure 6 that the temporal domain of the visualized history ends on September 98, since the cells corresponding to October, November and December of this year are marked with an X, meaning that they denote null values.

Notice that the time sliders involved in the Agenda technique are slightly different from the one involved in the pointwise browser. This is because in the pointwise browser, the role of the time slider is to fix a particular point in time (the reference instant), whereas in the agenda technique, the time sliders are used to fix an interval of time (the time window that is displayed on the screen).

In addition to navigation through time, the agenda interface supports two kinds of user interactions :

- The user may select a subspace of the visualized data, in order to visualize it in another agenda, or using any other technique. To do so, he must click on the button labeled Select within the control panel, and draw an arbitrary contour in the matrix. Unlike many existing interfaces offering similar subspace selection facilities, this contour needs not be a rectangle, but may take any shape. Once the contour drawn, a dialog box opens to assist the user in specifying the technique that should be employed to visualize the selected cells (the I/O re-use property is thus fulfilled).
- The user may choose a scale factor by clicking the corresponding value in the zoom panel. The smaller the rate is, the smaller and thus numerous are the squares of the agenda.

The agenda technique is mainly oriented towards three user tasks:

- Observing data layout and identifying general trends: in the assembly line's production history displayed in figure 6, it can be seen that the production volume is higher in recent years.
- Retrieving periodical patterns: for example, it can be seen from the agenda in figure 6, that the productions of the assembly line in December are usually higher than those of the other months, and also that the production volume attained in July tends to increase with time.
- Detecting hotspots of similar values: figure 6 shows that there is a hotspot of very high values around July and August of the last three years.

It is interesting to note that in the Agenda technique, the user perceives the data as a continuous space (i.e. the spatial continuity property is satisfied). Indeed, the gaps between cells are all the same size, and just big enough to act as boarders between cells. In this way, the user does not have the feeling of a discontinuous space as with the "superposed histograms" representation depicted in figure 7. Thanks to this property, the selection of a data subspace is also easier and more intuitive to perform.

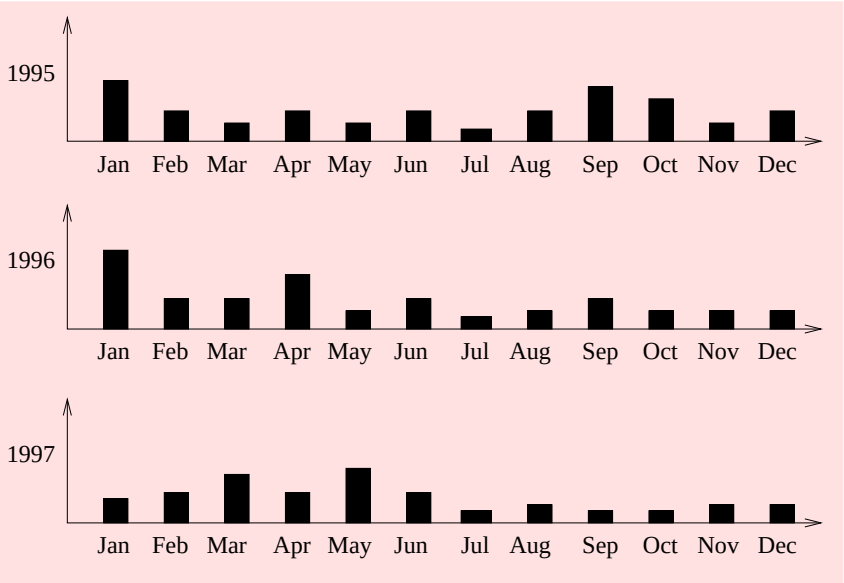


Figure 7: Superposed histograms

3.2 Concentric circles

We describe here our Concentric Circles Technique (CCT) and compare it with the only other technique for visualizing serial periodic data that we know, that is, the spiral technique depicted in figure 8.

The Concentric Circles Technique (figures 9 and 10) is designed to visualize one or two quantitative histories. Its interface essentially consists of a set of concentric circles, each one denoting the evolution of the visualized history(ies) during a fixed-length period of time. History values are denoted as rectangles

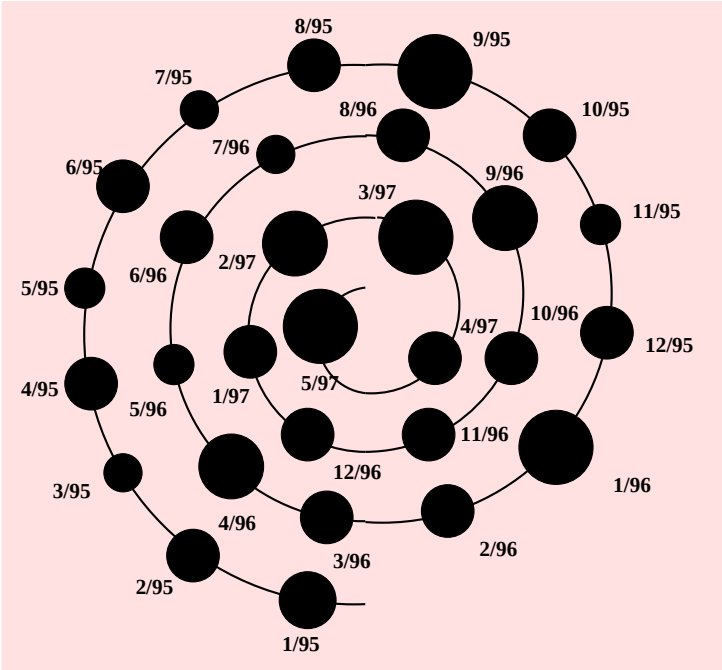


Figure 8: A visualization of the monthly productions of an assembly line using the Spiral Technique [CK98].

adjacent to the circles. The height of each rectangle and its color's intensity is proportional to the quantity that it denotes (observability and insistence property). Circles are arranged so as to reflect the time order associated with the data (honesty property).

At least four users' needs are addressed by the CCT: identify general trends, navigate through a data space, identify correlations within the evolution of two histories, and detailedly compare a history's values over two periods of time.

Observe data layout and identify general trends. With the spiral representation users must scour the data space along the spiral to observe data evolution, while with the CCT, the user can directly observe the evolution of data over a given period (e.g. a year or a month) as well as from one period to the next. For instance, it can be seen from figure 9 that the production volume's attained in January tend to be higher than the rest of the year, and that every year, this volume tends to be low in June and December.

Navigate through a (possibly large) data space. The output of a temporal query may constitute a huge space to be explored along the time axis. In the spiral representation, the time axis is continuous and has a fixed origin. The size of the spiral is consequently proportional to the number of periods. It is clear that with a large number of periods (e.g. a hundred), the spiral representation cannot be applied, due to limitation of screen space. We face the same problem in the CCT, since the number of circles corresponds to the number of periods. To address this issue, the CCT's interface provides a time-slider identical to those used in the Agenda technique. The displayed circles correspond to the periods falling

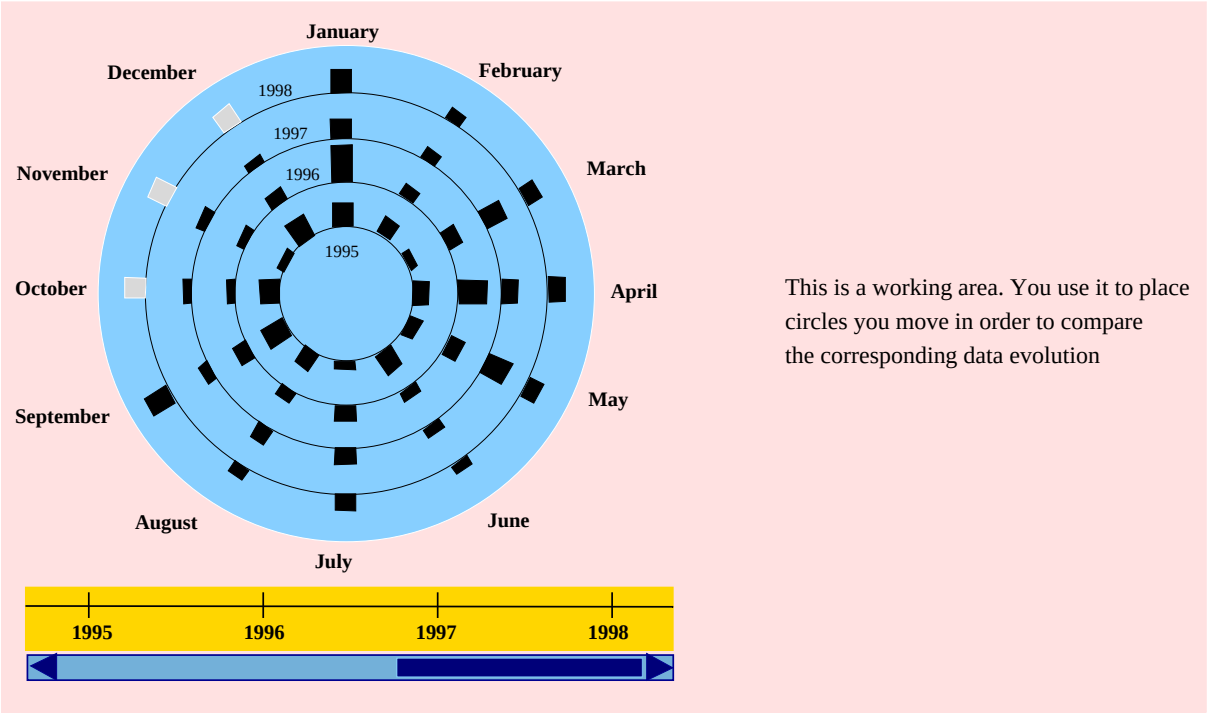


Figure 9: A visualization of the monthly productions of an assembly line using the Concentric Circles Technique (CCT).

within the time interval defined by the slider. Two graphical representations of the time axis are thus available: the set of circles, and the time-slider (representation-multiplicity property).

In the spiral technique, data items are represented as filled circles proportional to their values. This representation reaches its limits when there is a large number of elements per circle. Meanwhile, in the CCT, data items are represented as polygons whose widths are computed according to the number of elements per circle. The honesty property is thus satisfied.

Correlate the evolution of two histories. Users frequently wish to rapidly and interactively retrieve correlations between two variables according to their evolution over time. In order to address this task, the CCT supports the visualization of two quantitative histories at a time. One of the histories is represented exactly as before, while the other is represented similarly, but with rectangles of a different color and oriented in an opposite direction. For instance, in figure 10, the rectangles denoting the values of one of the histories are black-colored and oriented away from the center, and the values of the second history are white-colored and oriented towards the center. It is straightforward to see from this figure, that the evolution of the production volumes of the two assembly lines in December, tend to evolve inversely (for one assembly line they increase while they decrease for the other one).

Histories usually have different temporal domains [FDS99]. To correlate two histories, the two corresponding temporal domains must have at least one instant in common. The resulting time axis is the union of the temporal domains of the two histories. Missing data values are represented by grayed

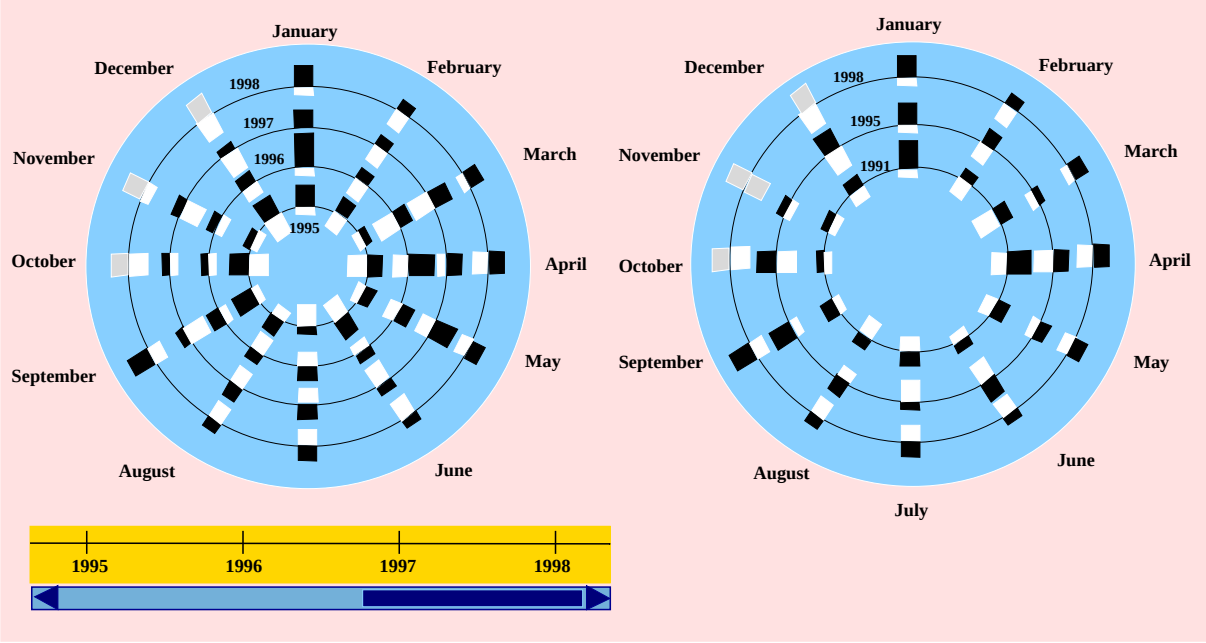


Figure 10: Visualization of the monthly productions of two assembly lines using the Concentric Circles Technique (CCT).

rectangles, to distinguish them from data values equal to zero (honesty property).

Compare the evolution of one or two histories over two periods of time possibly located far way from each other. From the visualization in figure 9, as well as that to the left of figure 10, it is difficult to compare two data items positioned far from each other on the screen. This is why the CCT interface provides two visualization areas. The first one, located on the left in figure 10 and named the *reference area*, contains circles ordered in time. The second one, located on the right in figure 10 and named the *working area*, enables the user to place selected circles in an arbitrary order. At any time, the user can copy a circle from the left area to the right one by dragging it with the mouse. For example in figure 10, the user has copied three periods in an arbitrary order (non-preemptiveness property) : first 1991, then 1998 and finally 1995 .

Notice that the above "copy" facility cannot be straightforwardly integrated into the spiral technique, since the time-line constitutes a single graphical object.

Putting circles close to each other, facilitates the comparison of data values belonging to different periods. Indeed the ratio between a value, and the height of the rectangle used to denote it, depends on the radius of the circle over which it appears. By placing two circles next to each other (radii are similar), we reduce this effect and the user can compare their values more easily (honesty property). For example, by observing the area on the right in figure 10, one can detect that the productions of the two assembly lines exhibit similar values in 1991 and 1998 (except for the last three months of the year).

As a concluding remark, it is interesting to note that CCT is applicable to non-temporal data. For

example, it can be used to represent the economic development of a set of countries through some fixed variables, including foreign investment, trade balance, GDP, etc. A circle then corresponds to a country while a radius denotes one of the observed variables (or vice-versa).

3.3 Implementation notes

We have developed the Concentric Circles and the Agenda techniques on the basis of reusable Java software components (JavaBeans). The time-slider for instance, is implemented as a JavaBean parameterized by its orientation, size, etc. The implementation required about 7500 lines of source code for the CCT and the Agenda techniques, and 1600 lines of source code for the time-slider. As for the pointwise browser, we adopted an architecture which separates the data extraction from the interface generation, so that the implementations of the techniques on themselves are independent of the underlying DBMS.

4 Related works

4.1 Temporal data browsing

Data visualization has received little attention within the temporal database research domain[6]. A notable exception to this remark is [PT94], which specifies a 3D interface for browsing temporal relational databases. In this approach, a temporal relation is represented as a sequence of time-indexed planes, each one displaying a table denoting a snapshot of the relation. Although the interface is not detailedly described, it seems that the interaction devices are limited to two scroll-bars: one for browsing through the records of a relation snapshot, and the other for navigating across the time dimension.

In the setting of information visualization, many techniques for graphically displaying and browsing temporal data have been considered [Ber81, Tuf84]. Most of these techniques are oriented towards quantitative time series, i.e. periodical series of numerical data items. [PMR+96] adapts some concepts developed in these works to design an interface for visualizing legal and medical personal records involving non-quantitative temporal data such as histories of texts and of complex objects.

The pointwise temporal object browser is an extension of data browsers such as KIVIEW [MDT88], O_2Look [PBL+92], ODEVIEW [DGJS95], SUPER [DAA+95], PESTO [CHMW96], and to some extent DTL's DataSpot [DEGP98]. In particular, the technique used to navigate through collections is based on the concept of synchronous navigation proposed in KIVIEW, and refined in ODEVIEW and PESTO. Nevertheless, the pointwise browser considerably differs from all the above ones, since it treats time as a dimension per se. Moreover, the pointwise browser takes into account the impacts of null-valued properties during synchronous navigation, whereas this issue is completely neglected in the above proposals.

[6] On the contrary, several visual query languages for temporal databases have been proposed (e.g. [FSC97]). We do not discuss them since the functionalities addressed by these proposals are beyond the scope of the present paper.

4.2 Quantitative histories visualization

We distinguish two categories of quantitative histories visualization techniques depending on the representation mode of elementary data items. In the first category [CK98, PMR+96, SC98], data values are mapped into graphical objects such as polygons, circles, etc., drawn in a 2D, 2D 1/2, or 3D space. Data values are easily distinguishable from each other, so the user can easily compare two elements. This representation mode is particularly used in exploratory data analysis and visual data mining tools (e.g. EXPLOR[7], 3DVDM[8], and several statistical packages[9]). However, this representation mode reaches its limits when displaying a huge amount of data. Consequently, these techniques must either be augmented with sliders, or the data space must be deformed [MRC91, SB92, NV98]. The second category is made up of the pixel-oriented techniques [Kei97, ABKS99, AEEK99, FNPS99], in which each data value is mapped into a pixel colored with some intensity. In this way, a huge amount of data can be represented in a limited screen area. The disadvantage of this technique is that users can not compare two elements, but rather have a global representation of the data space.

We can conclude that depending on the user needs, one mode of data representation is more or less adequate. Since our aim is to provide intuitive visualization tools which allow to detect correlations, trends, and periodic patterns, the techniques that we have developed belong to the first category.

5 Conclusion and future works

In this paper we have proposed three visualization techniques for two families of users' tasks arising during the interactive exploitation of temporal database query outputs: data browsing and data analysis.

On the one hand, we have proposed a new interaction technique for exploring temporal object databases. This technique orthogonally supports three types of navigation: (i) navigation through time, (ii) navigation via object relationships, and (iii) navigation within the elements of a collection.

On the other hand, we have presented two novel interactive techniques dedicated to the analysis of quantitative histories. These techniques address users' tasks such as retrieving regularities (e.g. periodical events), detecting value evolution trends, and retrieving concentrations of similar values. We believe that these proposals fall within the interactional data mining domain and are therefore complementary to computational data mining techniques. In other words, our tools are intended to be used in conjunction with mathematical and statistical algorithms operating over temporal databases.

In order to ensure their usability, all the proposed techniques have been designed according to a set of established ergonomic properties. In addition, we are currently carrying out an experimental evaluation of these techniques, both to measure their ability to assist the users in fulfilling the aforementioned users' tasks, and to validate and refine our design choices. Indeed, we believe that user testing is essential, and

[7] http://www.unige.ch/ses/sococ/eda/explor/expsoft.html
[8] http://www.cs.auc.dk/3DVDM/
[9] http://www.links2go.com/topic/Statistical_Software

that it should be carried out at early stages in the system life cycle, as part of an iterative design process.

Data contained in a temporal database may either reflect the evolution of the modeled reality (valid-time), or the time when the data were inserted/deleted into/from the database (transaction-time). In the former case, the database is said to be *historical*, while in the latter case, it is said to be *rollback* [JE98]. In some applications, it is necessary to timestamp data both with respect to the modeled reality and the database evolution, leading to *bi-temporal databases*. Up to now, we have limited our study to temporal databases managing a single time dimension (either valid-time or transaction-time). Extending it to bi-temporal databases constitutes a challenging research avenue.

As a second research avenue, we intend to extend our application scope to the area of spatio-temporal databases. On the long term, our aim is to design novel techniques for interactive spatio-temporal data mining. To this end, we need to identify the constraints and characteristics of the data that arise from the association of the time and space, and to establish a list of users' tasks arising during their manipulation.

References

[ABKS99] M. Ankerst, M. M. Breunig, H.-P. Kriegel, and J. Sander. OPTICS: Ordering Points To Identify the Clustering Structure. In *ACM SIGMOD Int. Conf. on Management of Data*, 1999.

[AEEK99] M. Ankerst, C. Elsen, M. Ester, and H.-P Kriegel. Visual classification: An interactive approach to decision tree construction. In *ACM SIGKDD Int. Conference on Knowledge Discovery and Data Mining (KDD'99)*, 1999.

[Ber81] J. Bertin. *Graphics and Graphic Information Processing*. Walter de Gruyter & Co, Berlin, 1981.

[CCLB97] T. Catarci, M.-F. Costabile, S. Levialdi, and C. Battini. Visual query systems for databases: a survey. *Journal of Visual Languages and Computing*, 8(2):215–260, June 1997.

[CHMW96] M. Carey, L. Haas, V. Maganty, and J. Williams. PESTO : an integrated query/browser for object databases. In *Proc. of the Int. Conference on Very Large Databases (VLDB)*, Mumbai, India, August 1996.

[CK98] J.-V. Carlis and J.-A. Konston. Interactive visualization of serial periodic data. In *Proc. of the ACM Conference on User Interface Software and Technology (UIST)*, San Francisco, CA, 1998.

[DAA+95] Y. Dennebouy, M. Andersson, A. Auddino, Y. Dupont, E. Fontana, M. Gentile, and S. Spaccapietra. SUPER: visual interfaces for object + relationship data models. *Journal of visual languages and computing*, 6(1):27 – 52, 1995.

[DEGP98] S. Dar, G. Entin, S. Geva, and E. Palmon. DTL's DataSpot: Database exploration as easy as browsing the web .. In *Proc. of the ACM SIGMOD Int. Conference on Management of Data*, Seattle, WA (USA), June 1998.

[DGJS95] S. Dar, N.H. Gehani, H.V. Jagadish, and J. Srinivasan. Queries in an object-oriented graphical interface. *Journal of visual languages and computing*, 6(1):27 – 52, 1995.

[FCD+99] M.-C. Fauvet, S. Chardonnel, M. Dumas, P.-C. Scholl, and P. Dumolard. Applying temporal databases to geographical data analysis. In *Proc. of the DEXA Workshop on spatio-temporal data models and languages*. IEEE Computer Society, September 1999.

[FDS99] M.-C. Fauvet, M. Dumas, and P.-C. Scholl. A representation independent temporal extension of ODMG's Object Query Language. In *actes des Journées Bases de Données Avancées*, Bordeaux, France, Octobre 1999. 20 pages.

[FNPS99] A. Fredrikson, C. North, C. Plaisant, and B. Shneiderman. Temporal, geographical and categorical aggregations viewed through coordinated displays: A case study with highway incident data. Technical Report 99-31, HCIL, University of Maryland, December 1999.

[FSC97] S. Fernandes, U. Schiel, and T. Catarci. Visual query operators for temporal databases. In *Proc. of the 4th Int. Workshop on Temporal Representation and Reasoning (TIME)*, May 1997.

[GC96] C. Gram and G. Cockton. *Design Principles for Interactive Software*. St Edmundsbury Press, 1996.

[JE98] C.-S. Jensen and C.-E. Dyreson (Eds). The consensus glossary of temporal database concepts – February 1998 version. Springer Verlag, LNCS 1399, 1998.

[Kei97] D.-A. Keim. Visual techniques for exploring databases, invited tutorial. In *Int. Conference on Knowledge Discovery in Databases (KDD'97)*, Newport Beach, CA, 1997.

[MDT88] A. Motro, A. D'Atri, and L. Tarantino. KIVIEW: An object oriented browser. In *Proc. of the Int. Conference on Expert Database Systems*, Vienna, Virginia (USA), April 1988. Benjamin Cummings.

[MRC91] J.-D. Mackinlay, G. Robertson, and S.-K Card. The perspective wall: Detail and context smoothly integrated. 1991.

[NV98] L. Nigay and F. Vernier. Design method of interaction techniques for large information spaces. In *Proc. of the Int. Conference on Advanced Visual Interfaces (AVI)*, L'Aquila, Italy, May 1998. ACM Press.

[PBL+92] D. Plateau, P. Borras, D. Leveque, J.C. Mamou, and D. Tallot. Building user interfaces with Looks. In F. Bancilhon, C. Delobel, and P. Kanellakis, editors, *The story of O_2*. Morgan Kaufmann, 1992.

[PMR+96] C. Plaisant, B. Milash, A. Rose, S. Widoff, and B. Schneiderman. LifeLines: Visualizing Personal Histories. In *proc. of the ACM CHI conference*, Vancouver, Canada, April 1996.

[PT94] P. Papapanagiotou and B. Theodoulidis. ERT/vql: A visual environment for querying and manipulating temporal database applications. Technical Report TR-94-5, Timelab, UMIST, 1994.

[SB92] M. Sarkar and M.-H. Brown. Graphical fisheye view of graphs. In *Proc. of Computer Human Interaction Conference (CHI'92)*, May 3-7 1992.

[SC98] Y. Shahar and C. Cheng. Intelligent visualization and exploration of time-oriented clinical data. Technical Report TR SM1-98-0732, Stanford University, 1998.

[Tuf84] E. Tufte. *The visual display of quantitative information*. Graphics Press, 1984.