3D Space Framework for the Multi-facet Accessing of Database Records

Makoto Ohigashi and Yuzuru Tanaka

Meme Media Laboratory, Hokkaido University,
N.13 W.8, Kita-ku Sapporo 060-8628, Japan
{ohigashi,tanaka}@meme.hokudai.ac.jp

Abstract. This paper proposes a framework for the construction of a 3D information access space that supports users to intuitively access large amounts of information. To cope with a large set of database records, we need a dynamic method for organizing and accessing records through multiple different views, such as topological, temporal, categorical, hierarchical and alphabetical views. In the proposed information space architecture, each record is visualized together with its related views, called facets. Each facet is provided as a 3D window-like component that displays a relevant information space on its surface and works as an entrance gate to the space. Through facet components, we can catch a glimpse of more detailed or related information spaces. We can also access relevant records by diving into arbitrarily chosen one of these spaces. Users can dynamically edit these multiple views in order to change the navigation and visualization functions by directly selecting and manipulating facet components.

1 Introduction

Recently, interactive information visualization of a large set of data or records is one of the most perspective applications of interactive 3D graphics. Simultaneously, the growth of the storage technologies derives a large set of data or records in databases. To cope with such volumes of information, users need efficient methods for organizing and accessing database records through various types of views. Conventional 3D visualization systems provide efficient methods for accessing a large amount of information[1, 2]. However, these systems provide *static* information spaces tightly coupled with the data structure, such as hierarchical data on the file systems[3–5] or nested-relations in an OODB[6, 7]. To support efficient access to large amounts of information, it is necessary to change the navigating functions dynamically such a way R.S.Wurman described[8].

CastingNet[9] is a hypermedia system for organizing and accessing semi-structured data such as those over the WWW. This system represents each data item as a frame with a list of attributes, and a relationship among frames as an axis associated with one attribute of frames. Users can combine some axes to organize a large set of frames. This approach allows users to define an axis using only one attribute. However, in order to obtain the required information from a large information repository, it is necessary to narrow down a large set of information by restricting the multiple attributes of data or records.

This paper proposes a framework for generating a 3D information space that supports users to organize and access information through multiple views. Our approach enables users to change their navigation and visualization functions dynamically in order to obtain their required information. In the proposed information space architecture, each database record is visualized together with multiple different views of the record, called facets. A facet works as a basis for organizing and accessing database records related to an arbitrary number of attributes. It allows users to organize and access database records through various views, including topological, temporal, categorical, hierarchical and alphabetical views, as R.S. Wurman said[8].

This paper also proposes a component-based framework for automatic creation of a proposed information access space. We construct a multi-facet information access space as a composition with 3D interactive components. It allows users to access relevant information by diving into another space repeatedly through arbitrarily chosen one of the facets. Furthermore, users can dynamically change their navigation and visualization functions by directly selecting and manipulating facet components.

The remainder of the paper is organized as follows. In chapter 2, we describe the concept of our proposed information access space, called MFIS. Chapter 3 gives the details of the implementation architecture and the automatic creation mechanism of an MFIS. Chapter 4 illustrates an example spatial navigation in an MFIS and an experimental evaluation. Finally, we make some concluding remarks.

2 The Concept

In order to obtain the required information from a large information repository, we need to explore information through multiple different views, including categorical, topological and temporal views. This paper proposes a 3D information access space that supports such explorations by associating multiple views with each database record. These views are called *facets* of each record. The information space, in which each record is visualized together with multiple facets, is called a Multi-Facet Information access Space, or an MFIS for short.

In this chapter, we illustrate the concept of a multi-facet of a record and a new information access method through such multiple facets. Next, we suggest the mappings of such an information access process to the users' navigation process in virtual spaces.

2.1 Multi-facet Information Access

Fig. 1 shows the concept of a multi-facet information access. In order to simplify the following explanations, let us use a historical database relation R, which

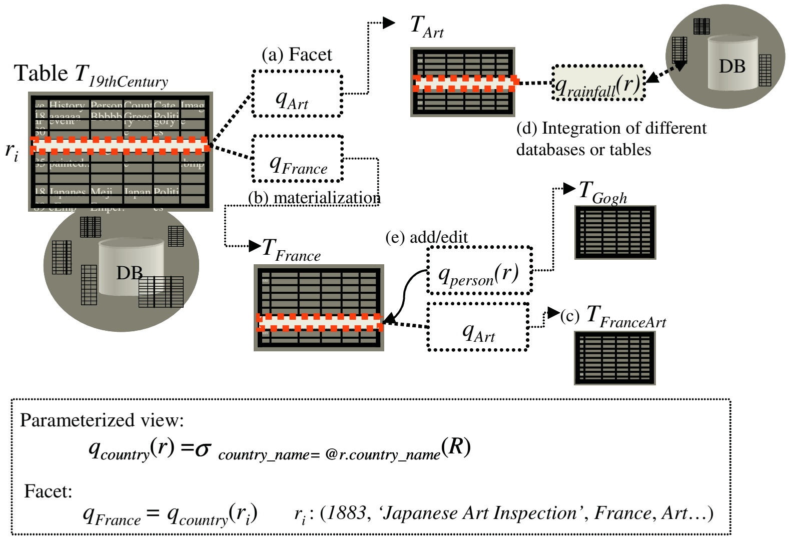


Fig. 1. The Multi-Facet Information Access.

contains attributes (*year, title, country_name, category, person_name, ...*). A relational table $T_{19thCentury}$ in Fig. 1 is the result of a query $q_{19thCentury}(= \sigma_{1800<year\leq 1900}(R))$. The following discussion, which used the specific relation R, is easily applicable to general relations. Furthermore, our architecture can be easily extended to cope with more than one relation as described in later subsection.

Multi-facet of a Record. In relational data model, we can consider an arbitrary query as a virtual relation, or a virtual table, called a database view[10]. It means that users can define an arbitrary new viewpoint as a database view. Suppose that users may browse the relation R through different views, including temporal(such as "19th century", "1870's" or "1998"), topological("Europe", "Germany" or "Berlin"), alphabetical(e.g. concerned with "Vincent van Gogh") and categorical(scientific, political, economical or artistic, etc.) views. We can define these views as parameterized queries, i.e. parameterized views. For instance, a topological view is defined as a following record-parameterized view,

$$q_{country}(r) = \sigma_{Country(r)}(R)$$
$$Country(r) : country\_name = @\mathbf{r.country\_name}$$

where **@r.country_name** part is the value of *country_name* attribute of a record r. When this view is instantiated by a record value, it becomes a ***facet*** of this record, e.g. the $q_{country}(r)$ is instantiated by the record r_i(*1883, 'Japanese Art Inspection', France, Art, ...*) to become a France facet q_{France}(see Fig. 1(a)). Other records similarly instantiate this view to work as a Japan facet or Germany facet etc.

We can define more complicated facet using two or more attributes. The following parameterized view q_{decade} gives a temporal and categorical view of focusing on, for example, the economical events in the decade around "the Great Depression(1837)".

$$\begin{aligned} &q_{decade}(r) = \sigma_{Decade(r)}(R) \\ &Decade(r) : category = @\mathbf{r.category} \\ &\qquad \wedge @\mathbf{r.year} - 5 \leq year \leq @\mathbf{r.year} + 5 \end{aligned}$$

Associating various types of parameterized views with a record, users can browse multiple different facets of the database record.

Facets Using Different Databases or Relations. A data model that gives multiple viewpoints to a single table is commonly referred to as a star schema model[11]. A star schema model contains a centralized table known as a fact table and its highly normalized tables known as dimension tables. However, this model has some operational limitations due to the distinction of these two tables. Our approach allows users to use different tables in the same or different databases for the definition of a facet because it is given as an arbitrary database view(see Fig. 1(d)). Suppose that a relational table S is a weather database relation, we can define the following facet combining a historical table R and a weather table S,

$$\begin{aligned} &q_{rainfall}(r) = \Pi_{S.month, S.rainfall}(\sigma_{Rainfall(r)}(R \bowtie S)) \\ &Rainfall(r) : S.country = @\mathbf{r.country} \wedge S.year = @\mathbf{r.year}. \end{aligned}$$

This facet defines a view that represents local monthly precipitation related to a historical event r. Furthermore, we can define another facet using these or other relation tables, and associate it with one of the record in the table T_{Art}.

Information Access Through Facets. In general information retrieval systems, users can obtain the required information through the recursive querying process, i.e. modifying existing queries and retrieving their results repeatedly. Turning now to the facets, an existing query is modified by the instantiation of a parameterized view, and its result is retrieved by the materialization of the instantiated view(see Fig. 1(b)). In such a process, we consider two different retrieving methods according to the users' intentions. (i) One is a method to retrieve the relevant information by specifying a new condition or integrating different relations as described in the above subsection. (ii) The other is a method to narrow down the current view by restricting the condition of the current focusing view. Consider now the case of the facet q_{France} in Fig. 1, we can define the following two types of facets,

(i) $q_{France} = \sigma_{Country(r_i)}(R)$

(ii) $q_{France} = \sigma_{Country(r_i)}(q_{19thCentury}) = \sigma_{Country(r_i) \wedge 1800 < year \leq 1900}(R)$

The facet (i) q_{France} selects historical events in France *of all date* from the relational table R. The other facet (ii) q_{France} selects historical events in France *in the 19th century.* The former facet is called an *extensional* facet and the latter is called a *restrictive* facet. Using the restrictive facets recursively, users can narrow down large tables. Through the two *restrictive* facets q_{France} and q_{Art} as shown in Fig. 1(c), we can obtain the relational table $T_{FranceArt}$ that contains the Artistic events in France in the 19th century.

Customization of the Facets. Associating an arbitrary parameterized view with a record, user can browse database through a new viewpoint. For example, let us consider the case that a user needs to associate a new alphabetical view 'person' with one record in the table T_{France} as shown in Fig. 1(e). First, users define a parameterized view $q_{person}(r)(= \sigma_{person\_name=@\mathbf{r.person\_name}}(R))$. When the user associates it with the record $(\dots,$ *Gogh,* $\dots)$, it makes a new facet q_{Gogh} of the record. Then, the facet retrieves a table T_{Gogh} containing records concerned with the whole Gogh's life in a database R.

2.2 Mapping of Database Elements to 3D Objects

The purpose of our work is to associate the database exploration process with a spatial navigation process in virtual spaces. In order to create such an information access space, we need a method to map database elements, such as relational tables, records and facets, to component objects in virtual spaces. Table 1 shows our mapping framework between database elements and component objects in virtual spaces. Fig. 2 shows the virtual space generated from the result of the mapping of the database elements in Fig. 1.

A database table is mapped into an individual virtual space, e.g. the relational table $T_{19thCentury}$ in Fig. 1 is mapped to the $19thCentury$-Space in Fig. 2. Each record in a relational table is mapped into a record object in a virtual space, e.g. records in the relational table T_{France} are mapped to record objects in $France$-Space. A record object is a 3D visual object that holds the value of a corresponding record as a list of attribute values. A facet of a record is mapped to a facet object of a record object, e.g. the facets q_{France} and q_{Art} of the record r_i are mapped to the facet objects f_{France} and f_{Art} of the record object r_i. The result table of the materialized facet is also mapped to a virtual space. A facet object displays the mapped space on its surface like a 3D window and works as an entrance gate to it, e.g. the facet object f_{France} displays the $France$-Space and enables us to enter this space.

Table 1. Primitive database constructs and their corresponding contents in a virtual space.

database constructs	virtual space elements
relational table/view	virtual space
record in a table	record object in a virtual space
facet of a record	facet object of a record object

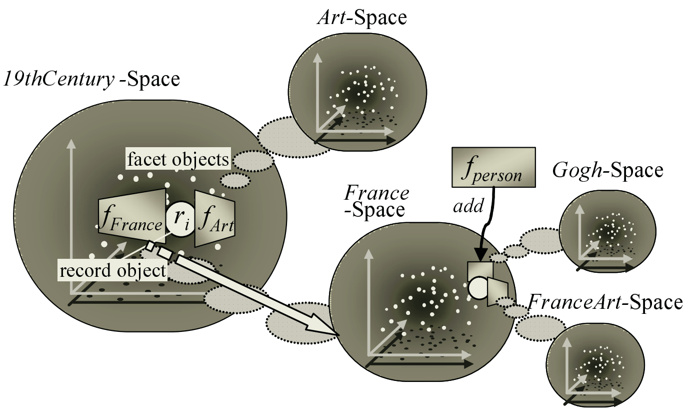


Fig. 2. The Multi-Facet Information Access Space.

These mappings result in the information access space as shown in Fig. 2, where users can explore database records through virtual space navigations. In this space, users can first browse the relational table $T_{19thCentury}$ by traveling around in the $19thCentury$-Space. Selecting one of the facet objects surrounding a focused record object, we can browse the relevant records of the focused record by diving into another information space. In Fig. 2, the user selects the facet object f_{France} surrounding the record object r_i and dives into the $France$-Space through this facet object. Let us denote this navigating function as follows,

$$19thCentury \xrightarrow{f_{country}} France.$$

The Addition of an *extensional* facet object f_{person} to one of the records in the $France$-Space enables users to access a new information space $Gogh$-Space. Let us also denote this navigating function as follows,

$$France \xrightarrow{\downarrow \tilde{f}_{person}} Gogh.$$

The proposed Information Space Architecture allows users to access a large information repository through the recursive spatial navigation. Furthermore, such navigation may retrieve the related information from different tables int the same or different databases in such a way as described in section 2.1.

3 Component-Based Framework

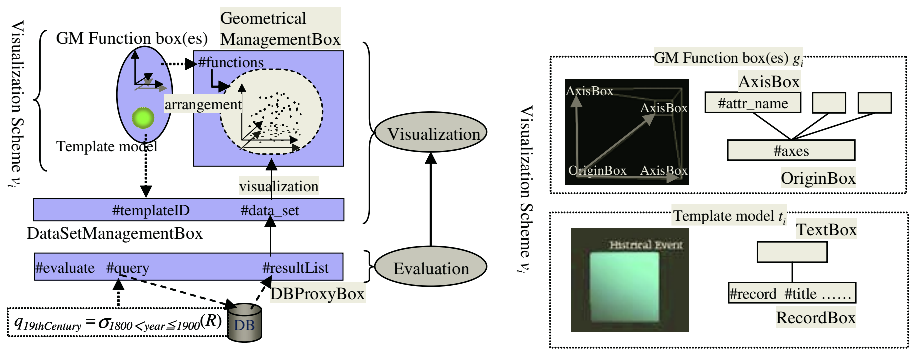
This chapter describes the details of our component-based framework for automatic creation of an MFIS. We use the IntelligentBox system[12] as the basis of construction of an MFIS. IntelligentBox system is a component-based visual software development system for interactive 3D graphic applications. This system represents objects as interactive 3D visual components, called boxes. It provides a uniform framework for the simultaneous definition of geometrical compound structures among boxes and their functional compositions. Each box has its own state value stored in variables, called slots. Let us denote a slot name as #slot_name. We can define a compound function between two boxes through selecting one slot from each of them and connecting the selected two slots. Developing the MFIS based on the IntelligentBox system, we can provide facet objects as interactive 3D visual components.

In order to construct an MFIS, we need two fundamental functions. One is the function of 3D visualization of database records. It generates a 3D visualization space from a database table or a virtual table. The other function provides a facet object as an interactive component. It enables users to associate arbitrary views with each record through direct manipulations.

3.1 3D Visualization of Database Records

A 3D visualization function of the retrieval records is realized by a Database Record Reification Box (DRRB)[13]. A DRRB is a composite box which visualizes database records as interactive components using user-defined visualization scheme. A DRRB consists of three basic components, including (i) a query evaluation, (ii) the visual representation of each record and (iii) the arrangement of a set of records, as shown in the Fig. 3(a).

A DBProxyBox works as an interface between a database management system(DBMS) and the IntelligentBox system. Users can use a set of database functions through a DBProxyBox. A DataSetManagementBox(DSMBox) virtually reifies each database record as an interactive component. A DSMBox stores a template box for visualizing the retrieved records. A GeometricalManagementBox (GMBox) geometrically arranges a set of records in a 3D virtual space. A GMBox holds a set of GM function boxes which specify a rule of records arrangement by composing a set of primitive geometrical functions, such as a coordinate origin box and coordinate axis boxes.



(a) The component-based architecture (b) An example visualization scheme

Fig. 3. A Database Record Reification Box(DRRB).

Fig. 3(b) shows an example visualization scheme. A visualization scheme needs two types of visual mappings, an intrinsic mapping and an extrinsic mapping[14]. A visualization scheme consists of a template model and GM function boxes. A template model defines an intrinsic mapping which specifies how to represent each record. A template model consists of a RecordBox and a set of attribute representation boxes. We can use an arbitrary box provided by the IntelligentBox system as an attribute representation box. In Fig. 3(b), the template model t_i includes a TextBox that represents the value of the attribute slot #title of the RecordBox. GM function boxes define an extrinsic mapping which specifies how to arrange a set of records geometrically. Each of the GM function boxes is a primitive geometrical function box such as an OriginBox and an AxisBox. In Fig. 3(b), the GM function box g_i consists of an OriginBox and three Axis boxes. The #attr_name slot of each AxisBox specifies the corresponding name of attribute.

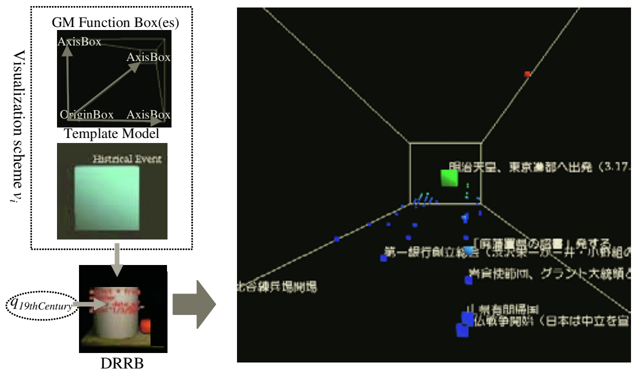


Fig. 4. An example visualization space of the table $T_{19thCentury}$.

Fig. 4 shows an example visualization space automatically generated by the DRRB which uses the visualization scheme v_i in Fig 3(b). In this example, a query $q_{19thCentury}$ is set in the #query slot of the DBProxyBox. The DBProxyBox retrieves the result table $T_{19thCentury}$ from a DBMS. The DSMBox and the GMBox generate the $19thCentury$-Space from the table $T_{19thCentury}$. Each record object represents the value of the attribute #title, because it is generated according to the template model t_i. The GM function box g_i arranges a set of record objects in the 3D space. In this case, the *category* attribute is mapped to the x-axis box, the *importance* attribute to the y-axis box, and the *year* attribute to the z-axis box. This space allows users to explore the history events in the 19th century.

3.2 Facet Object

In order to associate multiple different viewpoints with each of the record objects in the above space, a facet object needs a mechanism to be combined

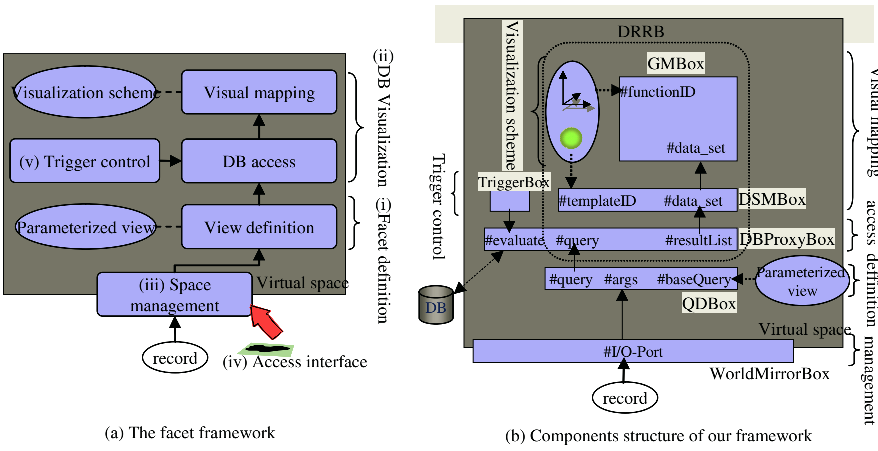


Fig. 5. The facet-object framework and its component-based implementation architecture.

with an arbitrary record object. Fig. 5 shows a facet-object framework and its component-based implementation architecture. We give the following five functions to a facet component.

(i) a definition of a facet,
(ii) the mapping of a facet to a virtual space,
(iii) a representation of a virtual space,
(iv) an access interface to a virtual space,
(v) a trigger control of a space creation.

The function (i) enables users to associate different views with each of the database records. Each view is defined as a parameterized view. When a facet object is connected to a record object, a facet object specifies a facet from a given parameterized view. For example, when a parameterized view $q_{country}(r)$ is connected with a record (*1878, 'The Treaty of Berlin', Germany, Politics, ...*), it becomes a facet $q_{Germany}$. This function is provided by a QueryDefinitionBox(QDBox). A QDBox holds a parameterized view in the #baseQuery slot. When it receives a record value in its #args slot, it defines a facet from the parameterized view, and it stores the facet in the #query slot.

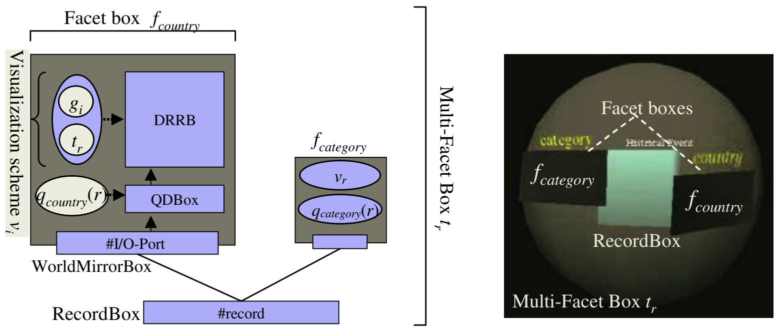
The function (ii) generates a visualization space from a facet. This function consists of two primitive functions, a query evaluation and a visualization of the retrieved records using a pre-registered visualization scheme. These functions are implemented as the functions of DRRB described in the previous section.

The function (iii) displays an arbitrary virtual space like a 3D window, in order to represent multiple facet objects together with a record object. Many preceding researches proposed such a function[15–17]. However, these systems provide no mechanism to define a functional linkage between a virtual space object and another objects. This means that it is impossible to combine a facet object with a record object. Therefore, we need a WorldMirrorBox[18] as the basis of a facet object. A WorldMirrorBox allows us to embed a 3D space in another 3D space environment. Users can see the contents of the embedded space through this 3D window-like WorldMirrorBox. Furthermore, we need another function that sends a record value to the #args slot of the QDBox in its embedded space. So, we extend the function of WorldMirrorBox to be able to send a record value to the inside objects through its #I/O-Port slot.

The function (iv) is provided by the World Mirror. The WorldMirrorBox allows users to dive into the embedded space. When a user has a collision with a WorldMirrorBox, the user can dive into the embedded space. It allows uses to navigate through the different spaces.

The function (v) is implemented as a TriggerBox. Since the MFIS structure is hierarchically nested, we need to control the timing of each space creation. A TriggerBox is connected with the #evaluate slot of the DBProxyBox. The DRRB generates a visualization space only when a TriggerBox issues a signal. A TriggerBox issues a signal only when a user enters to the space that includes a facet component. Each space is generated dynamically according to the user's database explorations process.

We realize the facet function by constructing these components as the structure as illustrated in Fig. 5(b).



(a) Components structure (b) Visual structure

Fig. 6. A multi-facet box structure used as a template model for generating an MFIS and an example visual design.

3.3 Multi-facet Component

A multi-facet component consists of a record object and an arbitrary number of facet objects. The structure of a multi-facet component is shown in Fig. 6. Visualizing all the retrieval records as multi-facet components, we can construct an MFIS. The base component of a multi-facet component is a RecordBox. All the facet objects are connected to the #record slot of the RecordBox. When a facet object receives a record value, it assigns the value to a parameterized view stored in the #baseQuery slot of the QDBox. In Fig. 6, the facet object

$f_{country}$ holds a parameterized view $q_{country}(r)$, and the facet object $f_{category}$ holds another parameterized view $q_{category}(r)$. Using this multi-facet component t_r as the template model in each facet object recursively, we can incrementally create an infinitely nested MFIS.

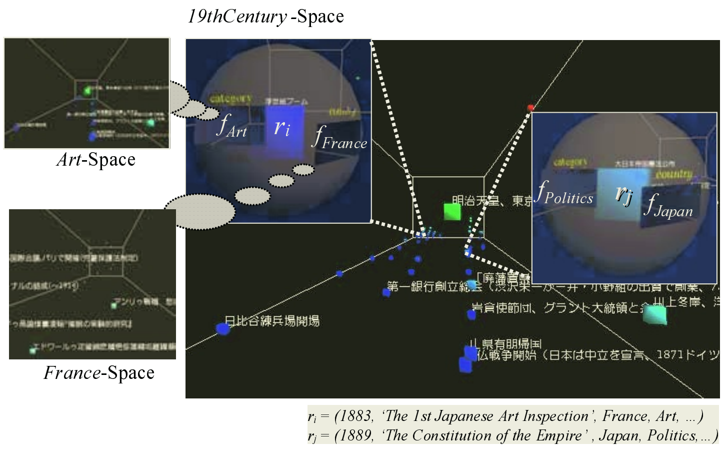
3.4 Automatic Creation Process

Fig. 7 shows an example MFIS automatically generated by the DRRB in Fig. 3 which uses the multi-facet component in Fig. 6 as its template model. The DBProxyBox holds the query $q_{19thCentury}$. When a trigger signal is sent to the #evaluate slot of the DBProxyBox, the DRRB makes a copy of the multi-facet component for each of the retrieval records, and geometrically distributes a set of visualized records. The RecordBox of each multi-facet component receives the corresponding record value. When the record value is sent to all the connected facet components, each facet component defines an appropriate facet and materializes it, and it generates the result visualization space. As shown in Fig. 7, the facet component $f_{country}$ connected with the record object r_i (*1883, 'The 1st Japanese Art Inspection', France, Art, . . .*) specifies a facet q_{France} from the parameterized view $q_{country}(r)$. Then, it generates a *France*-Space.

4 Spatial Navigation

4.1 Navigation Through Different Spaces

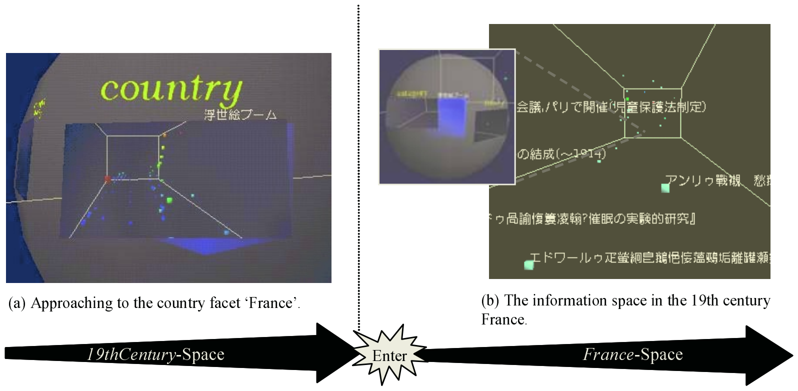
The MFIS incrementally generates the required information spaces according to the users' access behaviors. Fig. 8(a) shows a representation of the facet component f_{France} when the user is approaching to the record r_i in Fig. 7. It displays the *France*-Space on its surface, so the user can catch a glimpse of that space. As shown in Fig. 8(b), all the facets in the *France*-Space are materialized when the user enters into the *France*-Space through the facet component f_{France}. By



19thCentury -Space
Art-Space
France-Space
f_{Art}
r_i
f_{France}
$f_{Politics}$
r_j
f_{Japan}
category
country
明治天皇、東京
第一銀行創立総会
岩倉使節団、グラント大統領と会
日比谷練兵場開場
山県有朋帰国
普仏戦争開始（日本は中立を宣言、1871ドイツ
r_i = (1883, 'The 1st Japanese Art Inspection', France, Art, ...)
r_j = (1889, 'The Constitution of the Empire', Japan, Politics, ...)

Fig. 7. An example MFIS automatically generated by our framework.

traversing over the different spaces, users can narrow down a large set of records or retrieve the relevant records from different tables in the same or different databases.



(a) Approaching to the country facet 'France'.

(b) The information space in the 19th century France.

Fig. 8. The process of entering into the *France*-Space.

4.2 Modification of Navigation Functions

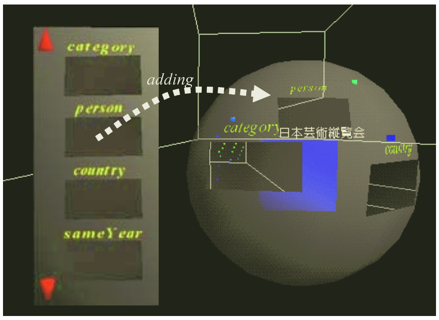
As shown in the preceding, we can explore database records by traversing access different spaces using the pre-registered multiple facets. However, in order to obtain the required information, it is also necessary for users to modify the views dynamically on their navigation process. Our approach provides each facet component as an interactive component. It means that users can dynamically edit navigation functions through direct manipulations of facet components.

Fig. 9 shows an example modification of a facet component. The left panel in Fig. 9(a) shows a list of the pre-registered facet objects. Fig. 9(b) shows a change of the components structure of a multi-facet component when a facet is added. Suppose that a user needs to add a new viewpoint 'person' to one of the records. When the user selects a facet component f_{person} from the list panel and connects it to the RecordBox by drag-and-drop, the facet component instantiates the parameterized view $q_{person}(r)$ by the record value. If the $person\_name$ attribute value is '*Gogh*', it defines a facet q_{Gogh}. The TriggerBox of the facet component f_{Gogh} issues a trigger signal. Then, the facet component generates a *Gogh*-Space and displays that space on its surface. It means that it becomes to be possible for the user to access a new visualization space *Gogh*-Space.

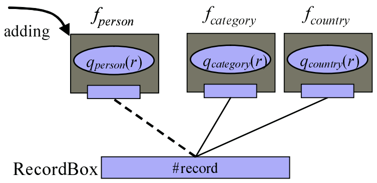
In this way, our approach enables users to change their viewpoints dynamically in order to access different information spaces.

4.3 Example Navigation

Fig. 10 illustrates an example navigation in an MFIS generated from the historical table R described in Section 2.1. The $19thCentury$-Space in Fig.10(a) is



(a) Addition of the person facet to the record object.



(b) The change of the components structure.

Fig. 9. The addition of a new facet to a record object.

the result space of the query $q_{19thCentury}$. The navigation process in this figure is one of the navigation paths to find the following objective α,

α : *all the records concerned with a painter*
who has made a copy of the UKIYOE in France.

We assume that we start our navigation from the $19thCentury$-Space. As described in Section 2.2, we can denote a candidate navigation path as follows,

$$19thCentury \xrightarrow[0.161]{f_{country}} France \xrightarrow[0.341]{f_{category}} Art \xrightarrow[0.107]{\downarrow \tilde{f}_{person}} Gogh,$$

where the number under the first arrow shows the ratio of the entrances to $France$-Space in the total records in the $19thCentury$-Space. Such a ratio indicates how easy to enter the next space. The notation $\downarrow \tilde{f}_{person}$ means the addition of an extensional facet object f_{person} to a record.

In the $19thCentury$-Space shown in Fig. 10(a), each record object has two types of facets, $q_{country}(r)$ and $q_{category}(r)$. Through the restrictive facet object f_{France} of the record *"The Napoleonic Code"*, we can enter the $France$-Space as shown in Fig. 10(b). Note that the 16.1% records in the $19thCentury$-Space have facet object f_{France}, which means that we can enter the $France$-Space through any of them. This navigation leads to the narrow down of the historical events to those *in France* in the 19th century.

In a similar way, entering the $FranceArt$-Space through one of the restrictive facet objects f_{Art} covering the 34.1% records in the $France$-Space, we can narrow down the historical events to the 399 historical events concerned with *France Art* in the 19th century(see Fig. 10(c)). In this space, we can see the records *"The 1st Japanese Art Inspection"* or *"The Exhibition of UKIYOE"*, which suggest that the Japanese arts were in fashion around that time in France. The record *"The copy of Japanese UKIYOE"* is found in the neighborhood of them. As shown in Fig. 10(c), adding an *extensional* facet object f_{person} to the record, the facet generates the *Gogh*-Space as shown in Fig. 10(d). In the *Gogh*-Space, we can see Gogh's life. We may see that he was born in the Netherlands and immigrated to France where the Japanese arts are in fashion at that time.

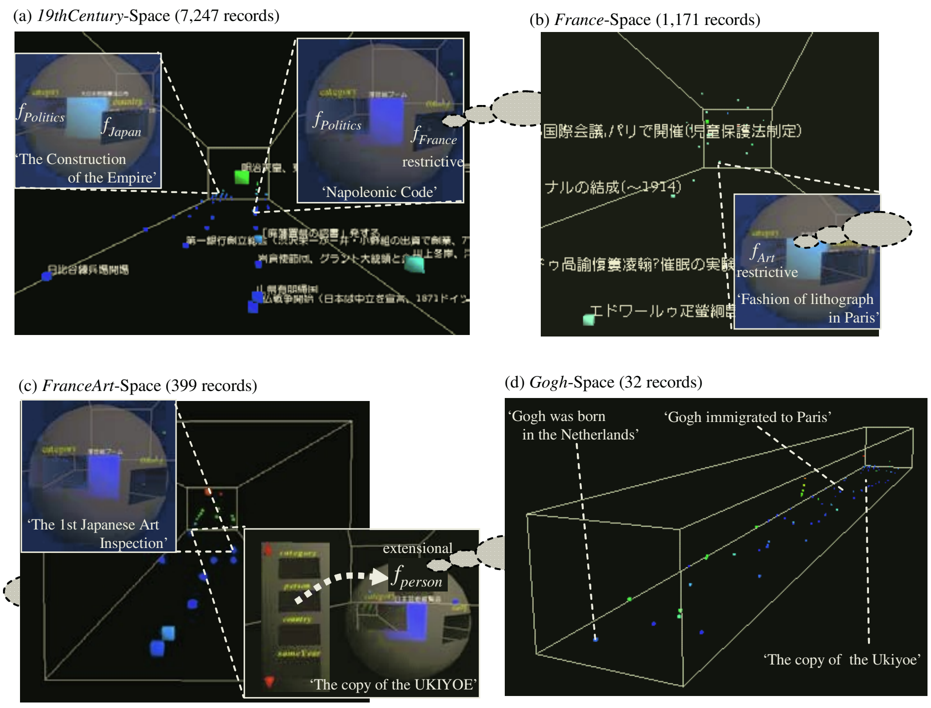

(a) 19thCentury-Space (7,247 records)
$f_{Politics}$
f_{Japan}
'The Construction of the Empire'
$f_{Politics}$
f_{France}
restrictive
'Napoleonic Code'
(b) France-Space (1,171 records)
f_{Art}
restrictive
'Fashion of lithograph in Paris'
(c) FranceArt-Space (399 records)
'The 1st Japanese Art Inspection'
extensional
f_{person}
'The copy of the UKIYOE'
(d) Gogh-Space (32 records)
'Gogh was born in the Netherlands'
'Gogh immigrated to Paris'
'The copy of the Ukiyoe'

Fig. 10. An example navigation of a historical database in the MFIS.

4.4 Experiments

This section describes the experiment to show the effectiveness of the information access in which the user who is not familiar with databases issues the complicated queries described in chapter 2 using the interfaces realized in chapter 3. In order to address the advantage of the MFIS approach over the conventional query specifications, it may be appropriate to report how long time users spent to reach to the destination space. First, we gave the objective α, *"Find all the records concerned with a painter who has made a copy of the UKIYOE in France."*, to 10 users, and recorded their access behaviors starting from the 19*thCentury*-Space to the destination *Gogh*-Space. Before the experiments, they learned the way to move freely in a virtual space. And the 4 facets, $f_{country}$, $f_{category}$, f_{person} and f_{decade}, are given to them.

Table 2 shows times they spent to reach the *Gogh*-Space using the four different navigation paths, P_1, P_2, P_3 and P_4. These paths are given below the table. The average time denotes how much time they spent in one space. The path P_1 introduced in Fig. 10 needs the average time of almost 61 seconds. The P_4 takes almost 62 seconds in similar process to the P_1. The P_2 takes the longest total time of about 75 seconds, because it required 5 steps to reach the destination. However, it takes the least time of about 15 seconds per one space. On the other hand, the P_3 needs only 3 steps, but it takes the longest average time of almost 23 seconds. The reason why P_3 takes so much time per

Table 2. The navigation time for each type of paths.

Navigation Path	Number of Spaces	Average Time(s/space)	Total Time(s)
P_1	4	15.27	61.10
P_2	5	14.97	74.85
P_3	3	22.85	68.56
P_4	4	15.53	62.11

$$P_1 : 19thCentury \xrightarrow[0.161]{f_{country}} France \xrightarrow[0.341]{f_{category}} Art \xrightarrow[0.107]{\downarrow\tilde{f}_{person}} Gogh$$
$$P_2 : 19thCentury \xrightarrow[0.161]{f_{country}} France \xrightarrow[0.341]{f_{category}} Art \xrightarrow[0.190]{\downarrow f_{decade}} Decade \xrightarrow[0.160]{\downarrow\tilde{f}_{person}} Gogh$$
$$P_3 : 19thCentury \xrightarrow[0.200]{f_{category}} Art \xrightarrow[0.022]{\downarrow\tilde{f}_{person}} Gogh$$
$$P_4 : 19thCentury \xrightarrow[0.200]{f_{category}} Art \xrightarrow[0.274]{f_{country}} France \xrightarrow[0.107]{\downarrow\tilde{f}_{person}} Gogh$$

one space is that it forces the user to find the 2.2% records related to ‘Gogh’ in the *Art*-Space.

Even if a user may take any of these paths, it takes only 66.65 seconds on the average to navigate to the target in an MFIS. However, the SQL statement for the same navigation as the path P_2 has the following complexity,

```
q_France :
    Select $attr\_list$
     From $q_{19thCentury}$
   Where $country\_name = subquery_{country}$

$subquery_{country}$ :
    Select $country\_name$
     From $q_{19thCentury}$
   Where $title =$ ‘Napoleonic Code’

q_Art :
    Select $attr\_list$
     From $q_{France}$
   Where $category = subquery_{category}$

$subquery_{category}$ :
    Select $category$
     From $q_{France}$
   Where $title =$ ‘The fashion of Lithograph in Paris’

q_Decade :
    Select $attr\_list$
     From $q_{Art}$
   Where $year \geq subquery_{year} - 5$ And $year \leq subquery_{year} + 5$
```

$subquery_{year}$:

```
  Select $year$
   From $q_{Art}$
 Where $title$ = ‘The 1st Japanese Art Inspection’
```

$\mathbf{q_{Gogh}}$:

```
  Select $attr\_list$
   From $R$
 Where $person\_name = subquery_{person}$
```

$subquery_{person}$:

```
  Select $person\_name$
   From $R$
 Where $title$ = ‘The copy of the UKIYOE’.
```

This query needs to specify conditions such as

$$title = \text{‘The 1st Japanese Art Inspection’}$$

and

$$title = \text{‘The copy of the UKIYOE’}.$$

However, it is difficult to specify these conditions correctly in advance without reading any attribute values of relevant records. Our approach allows users to specify these predicates by focusing on one of the records or selecting one of the facets in a virtual space repeatedly. It means that users can issue the above complicated queries only by traversing over the virtual spaces.

In this experiment, the user issues a set of queries to narrow down large tables. However, users can also issue more complicated queries, such as a query which joins different tables in the same or different databases, using the same interface in our proposal space.

The result of our experiment shows that our approach provides much more rapid and intuitive access to such target records than conventional query specifications.

5 Conclusion

In this paper, we have proposed an MFIS and its component-based implementation framework. In oder to support efficient explorations of a large amount of information, it is required to define and dynamically change the viewpoints for organizing and accessing them as R.S.Wurman said. First, we have proposed a framework for generating a 3D information space that enables users to organize and access database records through various different views. In this space, users can dynamically change their navigation functions to obtain the required information from a large information repository. This paper has also proposed a component-based framework for the automatic creation of a proposed information access space. The proposed framework allows users to explore a large amount of information by diving from a record in one information space to another space related to the record repeatedly. Moreover, it enables users to manipulate and select one of the facet components to change the navigation and the visualization functions dynamically. The result of our experiment shows the advantage of our approach over the conventional query specifications.

References

1. Card, S.K., Robertson, G.G., Mackinlay, J.D.: The information visualizer, an information workspace. In: Proc. ACM Conf. on Human Factors in Computing Systems(CHI'91). (1991) 181–188
2. Card, S.K., Mackinlay, J.D., Shneiderman, B.: Reading in Information Visualization Using Vision to Think. Morgan Kaufmann (1999)
3. Rekimoto, J., Green, M.: The Information Cube: Using Transparencty in 3D Information Visualization. In: Proc. Third Annual Workshop on Information Technologies & Systems (WITS'93). (1993) 125–132
4. Robertson, G.G., Mackinlay, J.D., Card, S.K.: Cone trees: Animated 3D visualizations of hierarchical information. In: Proc. ACM Conf. on Human Factors in Computing Systems(CHI'91). (1991) 189–194
5. Silicon Graphics, Inc.: FSN: File System Navigator. online manual edn. (1992)
6. Boyle, J., Leishman, S., Fothergill, J., Gray, P.: WIMPS to 3D: design of a visual language for a database. Technical report, Aberdeen University (1994)
7. Massari, A., Saladini, L., Hemmja, M., Sisinni, F.: Virgilio : A non-immersive VR system to browse multimedia databases. In: Proc. of IEEE Intel. Conf. on Multimedia Computing and Systems. (1997)
8. Wurman, R.S.: Information Anxiety. Doubleday (1989)
9. Masuda, Y., Ishitobi, Y., Ueda, M.: Frame-axis model for automatic information organizing and spatial navigation. In: Proc. ACM European Conf. on Hypermedia technology. (1994) 146–157
10. C.J.Date: An Introduction to Database Systems. Addison-Wesley (1990) Fifth Edition.
11. Kimball, R.: The Data Warehouse Toolkit: Practical Techniques for Building Dimensional Data Warehouses. John Wiley & Sons (1996)
12. Okada, Y., Tanaka, Y.: IntelligentBox: A constructive visual software developement system for interactive 3D graphic apprications. In: Proc. of Computer Animation'95. (1995) 114–125
13. Ohigashi, M., Tanaka, Y.: A framework for the virtual reification of database records. IPSJ **42** (2001) 80–91
14. Benedikt, M.: Cyberspace: Some Proposals in Cyberspace: First Steps. MIT Press (1991)
15. Elvins, T.T., Nadeau, D.R., Kirsh, D.: Worldlets - 3D thumbnails for wayfinding in virtual environments. In: Proc. ACM User Interface Software and Technology Symposium-CUIST'97. (1997)
16. Stoakley, R., Conway, M.J., Pausch, R.: Virtual reality on a WIM: Interactive worlds in miniature. In: Proc. Conf. ACM SIGCHI. (1995)
17. Viega, J., Conway, M.J., Williams, G., Pausch, R.: 3D Magic Lenses. In: Proc. ACM UIST'96. (1996) 51–58
18. Itoh, M., Tanaka, Y.: WorldMirror and WorldBottle: Components for embedding multiple spaces in a 3D virtual environment. Jounal of IPSJ **42** (2001) 2403–2414