A Formal Approach for Modeling Interactive Visual Interfaces

Emanuele Covino and Giovanni Pani
Dipartimento di Informatica
Bari, Italy
Email: covino@di.uniba.it, pani@di.uniba.it

***Abstract*—We provide a mathematical model that supports the formal description of visual interfaces' behaviour. The formalism is based on type-inference notation, in which each variable is defined in the domain of the interface basic widgets, and each transition from a given state of the interface to the following one is represented by the application of an inference rule. When a sequence of actions is made by the user, the behaviour of the corresponding interface is totally defined by the set of inference rules. This formalism allows the designer to formally verify the properties of the interface.**

I. Introduction

The design of user driven interfaces is a part of the more general Human-Computer Interaction field, and the related problems are well recognized and described. In particular, the approaches based on design-tools have several limitation as regards the possibility of ensuring the proper functioning, testing and verification of interfaces design and behaviour in an event-driven environment. For most computer controlled systems an effective design process requires a very early conceptual and architectural validation prior to the implementation in order to avoid re-design cycles. All relevant system's characteristics have to be checked during the verification phase in order to have a high design quality. Unfortunately, most designer usually haven't the appropriate mathematical knowledge required for applying formal verification techniques in the software design process. On the other hand, consistency between the mathematical model of the system and the original one cannot be assured. For this reasons, the introduction of a unified formalism to describe the structure and the behaviour of a complex system - as a generic interface - has central importance. As a result, the interface should be understandable, reusable and open to evolution. A formal approach must also offer the tools for the evaluation of the interface's properties, such as absence of deadlock, predictability and availability of commands, re-initiability. Moreover, the formalism should offer features such as design verification, graphical representation, simulation and dynamic analysis. Formal notations that describe both syntactic and semantics of interfaces have been often used; for instance, diagrammatic notations, such as state transition diagrams, Petri nets, or traditional flow diagrams; textual notation, such as grammars or event algebras; matrix notation; model checking. The reader can find an extensive list of these approachs in [3], [4], [5], [6], [7], [8], [10], [11], [12], [15], [16], [17], and a survey in [9]. All these notations have been developed with the purpose to describe the behaviour of the system, keeping a certain level of abstraction from the related programming. A formal description should allow us to verify whether or not the real system satisfies the requirements.

In this paper, a new formal specification of visual interfaces is presented, which is based on type-inference notation; each element of the interface is defined in the domain of the basic widgets, and each transition from a given view (i.e. state of the interface) to the following one is represented by the application of an inference rule. The designer can specify the behaviour of the interface by defining the set of inference rules. We believe that this approach, when used to describe a specific interactive system, provides a simpler understanding of the dynamics of the interface, and that it is appropriate for the description and analysis of more generic principles of interactive systems; for instance, observability or reachability of a given state of an interface can be easily detected by analyzing the properties of the related inference tree. Moreover, it provides the designer with a easy-to-understand description of the interaction, with a mathematically precise description of transformation rules, and with an efficient back-annotation of mathematical analysis results.

As pointed out by an anonymous referee, this paper represents a preliminary approach to the definition of a new visual interface's specification language; more usability tests and comparison with other specification languages are needed, and they will appear in the prosecution of this paper.

The paper is organized as follows: in Section II and III we give a quick overview of two query preview systems, the EOSDIS (see [2], [13] and [14]) and the DaeQP (see [1]), respectively; these systems are used for the visual analysis of databases and we use them as subjects of our formalization. In Section IV we provide the static description and the dynamic behaviour of the interface, according to our notation. In Section V we discuss some properties that could be verified with our formalism and further work.

II. Interacting with the EOSDIS query preview system

In [2], [13] and [14], Doan, Plaisant et al. proposed a two-phase approach to the dynamic query formulation by volume preview; the two phases were the query preview and the query refinement. They did it in order to overcome the obstacles facing users in a querying process: network performance, data volume, and data complexity. In the query preview phase, the user formulate an initial query by selecting desired attribute values, which are shown graphically on previews bars; after this phase, the query refinement phase takes place. In this section, we provide a short overview of the result presented in [2], focusing our attention on the first phase, in order to understand how the system interacts with the user, and how to formally specify (in the following sections) its behaviour. Figure and examples comes from [2].

The Earth Observing System Data and Information System (EOSDIS) is a data and information system, developed by NASA under the Mission to Planet Earth (MTPE) Program. It handles data from NASA's Earth science research satellites and field measurement programs, providing data archive, distribution, and information management services.

In [2], a dynamic query interface to the EOSDIS was introduced, consisting of a query preview panel (see figure 1) and of a query refinement panel. There are three different selectable attributes in the query preview panel: the spatial coverage, the parameters, and the temporal coverage. The spatial coverage is defined by continents, oceans, and a map of rectangular areas. The parameters are classified into nine groups depending on the types of the data sets they represent (e.g. Atmospheric Composition, Atmospheric Dynamics, Biosphere, etc). The temporal coverage is measured in terms of years. When the query preview panel starts off, it displays the number of data sets for each parameter, region, and year, in form of the attribute preview bars or pie charts (see figure 1). The query preview bar, on the bottom of the panel, displays graphically the total number of the selected data sets in the left section of the bar, and the excessive region (above the recommended level) in the right section. An initial query in the panel is made by selecting the parameter group of interest; this results in (1) the display of all the available parameters in that group and its corresponding preview bars, and (2) the updating of the attribute preview bars for each continent and year in order to display the corresponding number of data sets that contain one or more parameters of the selected parameter group. For example, if the user is interested in the temperature of U.S. coastal waters, he selects the ”By Continents” option from the Geographical Selection's menu in figure 1; then, he selects the ”Atmospheric Dynamics” parameter in order to get the desired data (he knows that the parameter ”Sea Surface Temp”, which is used to study the temperature of coastal waters, is in both the ”Atmospheric Dynamics” and ”Ocean Dynamic” parameter groups; and the pie chart of the parameter groups shows that there are more data sets in the ”Atmospheric Dynamics” than in the ”Ocean Dynamic”). The result of the parameter group selection is illustrated in the following figure 2.

The user now selects the parameter ”Sea Surface Temp”; this results in the change of the attribute preview bars representing each continent and year (see figure 3). The updated preview bars represent the number of data sets in which appears the parameter ”Sea Surface Temp”. The user has to choose a specific year; the attribute preview bars associated to each year help the user to know the number of the data set's hits he might get if he selects that year. For example, preview bars related to the years 1984 and 1985 indicate that there are no data sets in those years (see figure 3). They also reveals that the majority of data sets on the ”Sea Surface Temp” occurs in the year 1992. Thus, the user selects 1992. The total number of the selected data sets is reduced to 276 (see figure 4), and the user can continue to reduce the volume of the relevant data sets to 91 by selecting ”North America” (see figure 5). Finally, this initial query can be submitted to the data acquisition's archive centers for the extraction of metadata of the selected data sets.

In [2] it is also introduced a query refinement panel that supports dynamic queries over a local database that stores the metadata of the data sets extracted from the query preview panel. The metadata contains the information of all the attributes of the data sets such as the parameter, sensor, platform, project, data archive centers, processing data level, time, location which are also visually represented in the interface. The main function of the query refinement panel is to support further refinement for the data sets in the first step. Note that, to our purpose, both the query preview panel and the query refinement panel support multiple selection of the attributes and going back and forth between the two phases, to refine the best data set. We will define our formalism in order to represent this situation.

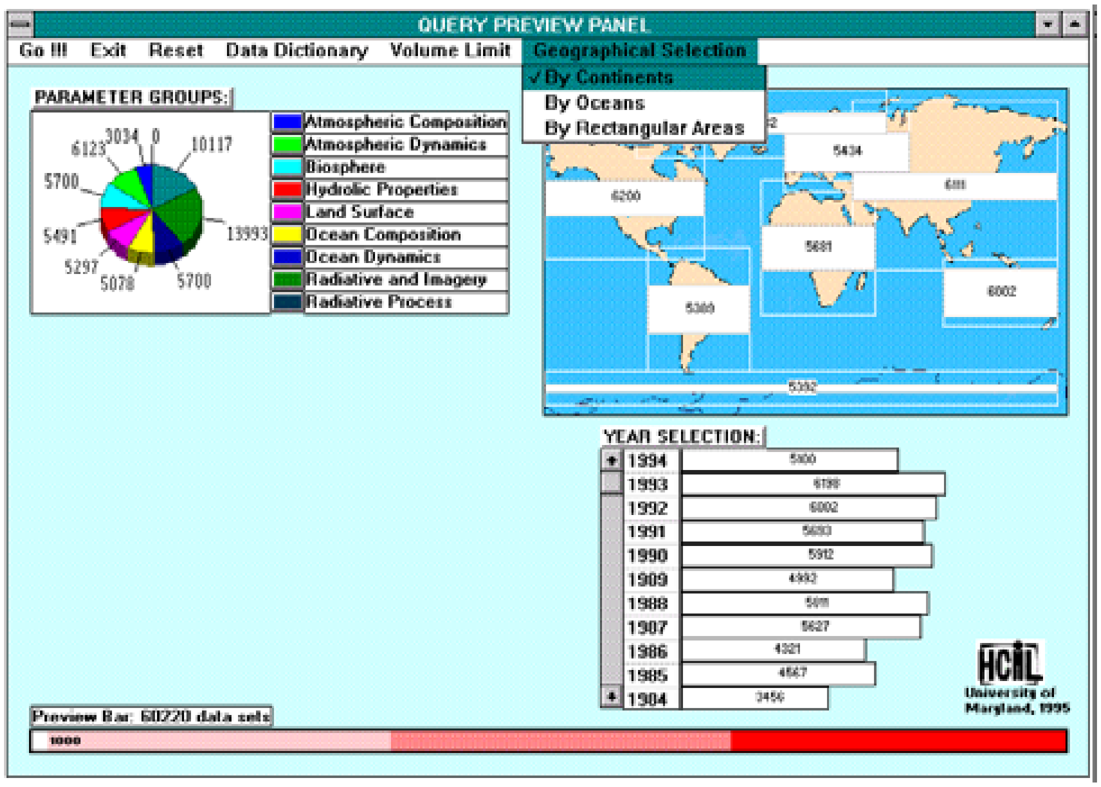


Fig. 1. Initial configuration of the query preview panel

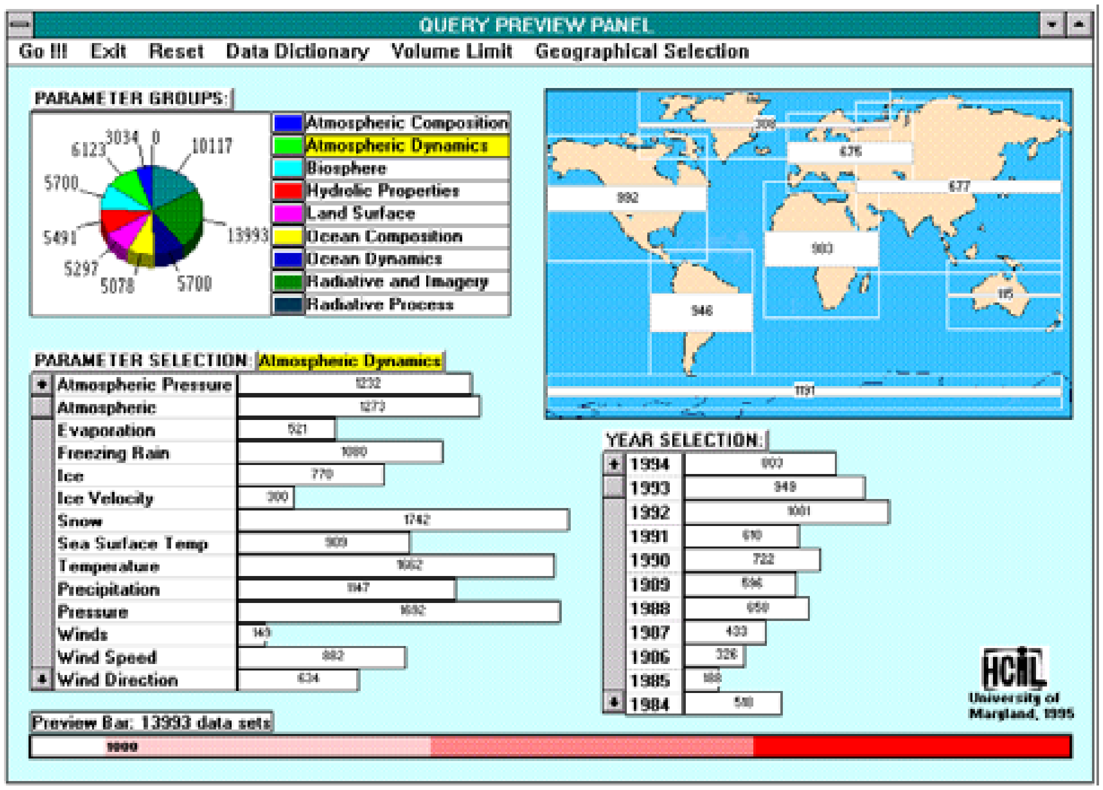
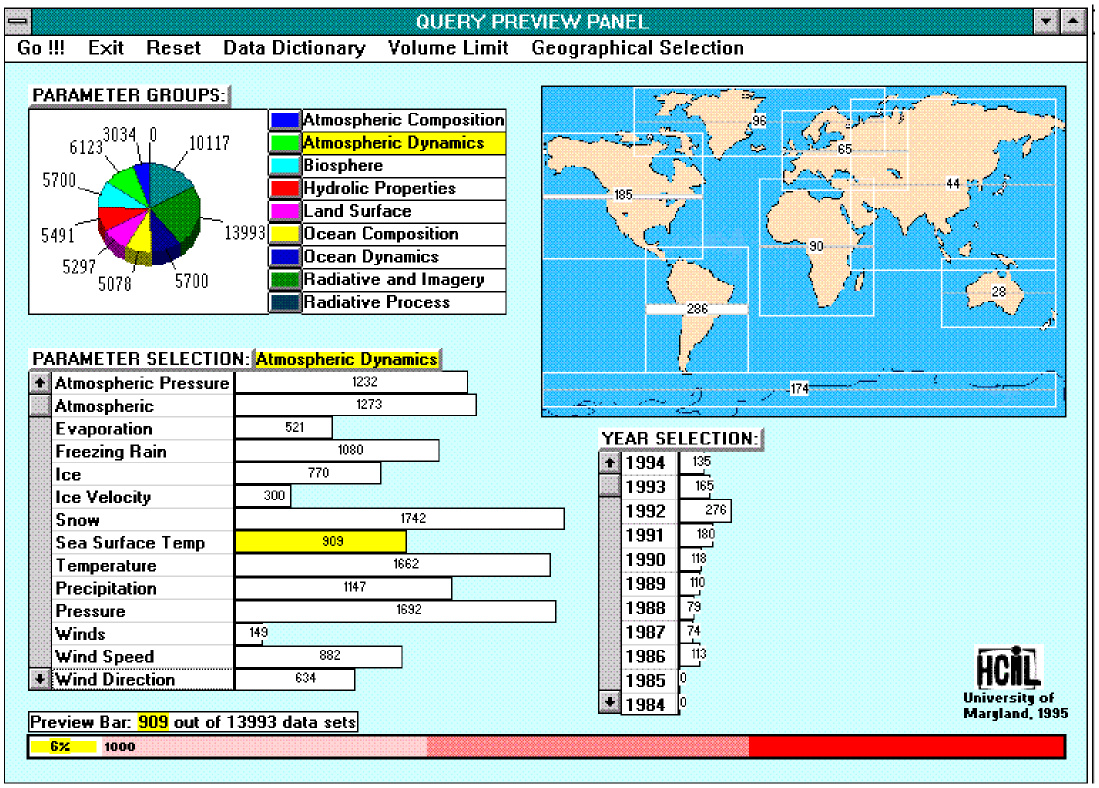


Fig. 2. The query preview panel after selecting Atmospheric Dynamics



QUERY PREVIEW PANEL

Go !!! Exit Reset Data Dictionary Volume Limit Geographical Selection

PARAMETER GROUPS:

🟦	Atmospheric Composition
🟩	Atmospheric Dynamics
🟦	Biosphere
🟥	Hydrolic Properties
🟪	Land Surface
🟨	Ocean Composition
🟦	Ocean Dynamics
🟩	Radiative and Imagery
🟦	Radiative Process

PARAMETER SELECTION: Atmospheric Dynamics

Atmospheric Pressure	1232
Atmospheric	1273
Evaporation	521
Freezing Rain	1080
Ice	770
Ice Velocity	300
Snow	1742
Sea Surface Temp	909
Temperature	1662
Precipitation	1147
Pressure	1692
Winds	149
Wind Speed	882
Wind Direction	634

YEAR SELECTION:

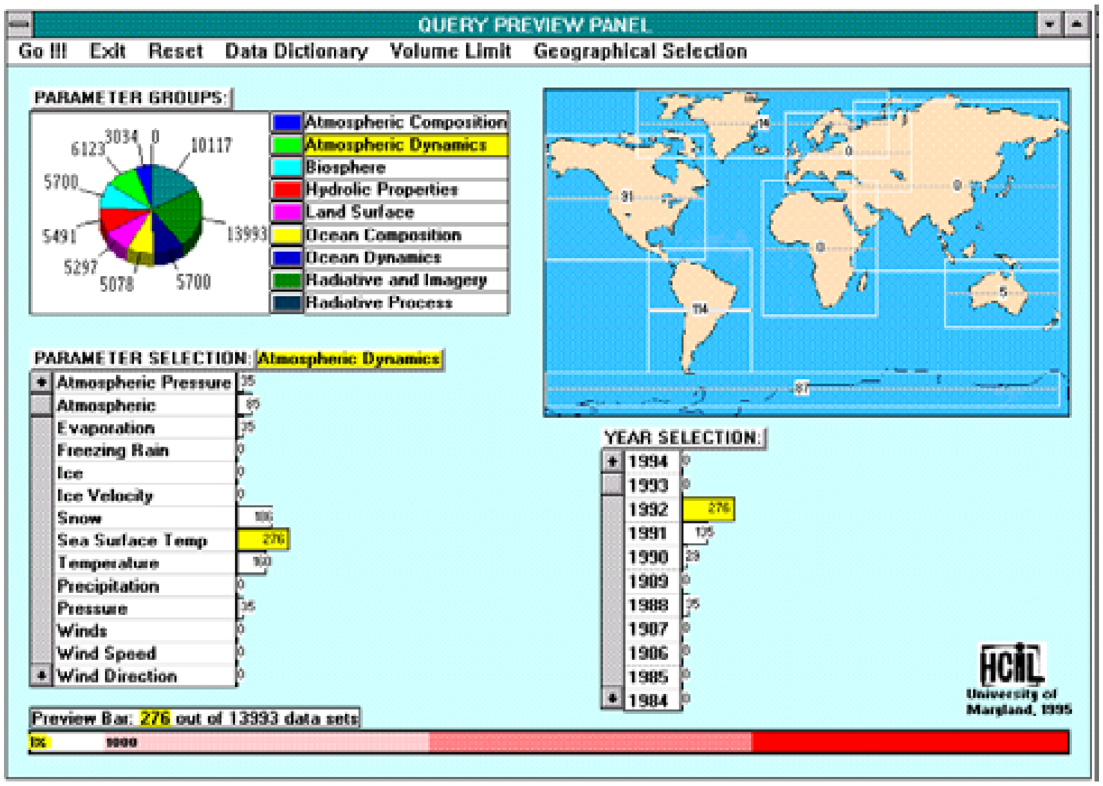
1994	135
1993	165
1992	276
1991	180
1990	118
1989	110
1988	79
1987	74
1986	113
1985	0
1984	0

HCIL University of Maryland, 1995

Preview Bar: 909 out of 13993 data sets

6% 1000

Fig. 3. The query preview panel after selecting Sea Surface Temp



QUERY PREVIEW PANEL

Go !!! Exit Reset Data Dictionary Volume Limit Geographical Selection

PARAMETER GROUPS:

🟦	Atmospheric Composition
🟩	Atmospheric Dynamics
🟦	Biosphere
🟥	Hydrolic Properties
🟪	Land Surface
🟨	Ocean Composition
🟦	Ocean Dynamics
🟩	Radiative and Imagery
🟦	Radiative Process

PARAMETER SELECTION: Atmospheric Dynamics

Atmospheric Pressure	35
Atmospheric	85
Evaporation	35
Freezing Rain	0
Ice	0
Ice Velocity	0
Snow	106
Sea Surface Temp	276
Temperature	100
Precipitation	0
Pressure	35
Winds	0
Wind Speed	0
Wind Direction	0

YEAR SELECTION:

1994	0
1993	0
1992	276
1991	135
1990	29
1989	0
1988	35
1987	0
1986	0
1985	0
1984	0

HCIL University of Maryland, 1995

Preview Bar: 276 out of 13993 data sets

1% 1000

Fig. 4. The query preview panel after selecting 1992

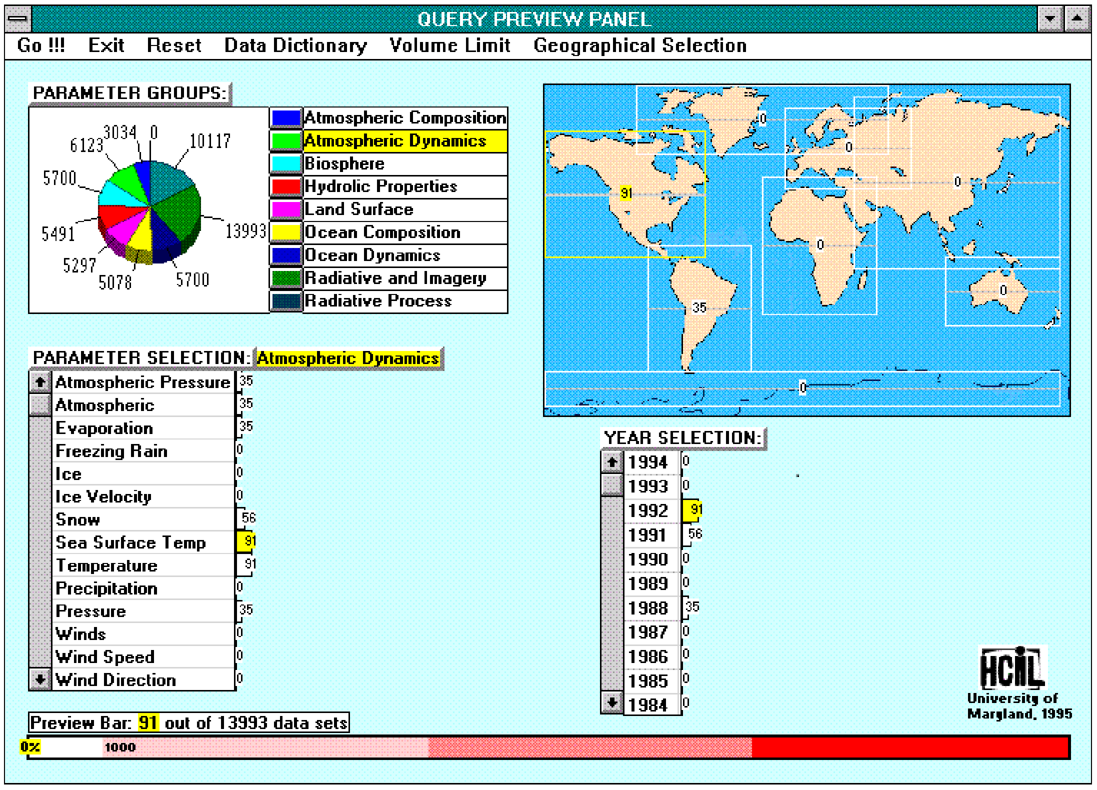


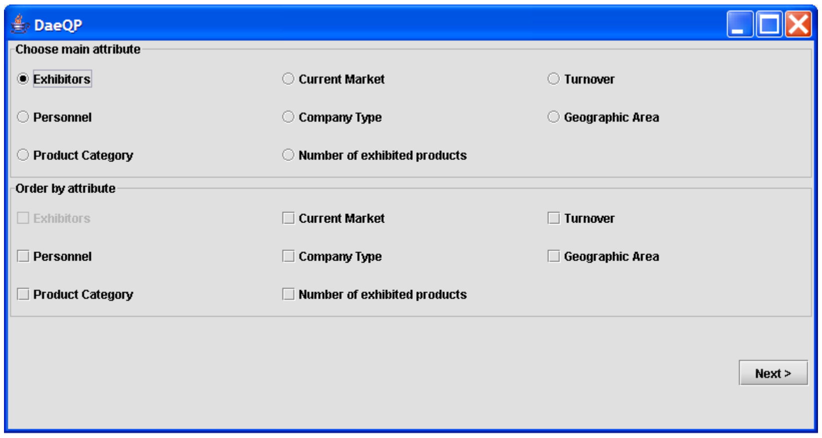
Fig. 5. The query preview panel after selecting North America

III. Interacting with the DaeQP Query Preview System

DaeQP is a another visual query preview tool developed in [1]; it supports the analysis of multi-dimensional data, providing the user with multidimensional overviews and allowing him to perform an appropriate data analysis by direct manipulation of the elements on the screen. Users may filter out uninteresting items and focus on those of interest; once a manageable number of items is obtained, it is easy to browse the details about groups or individual items. The scenario provided is that of an agriculture's trade fair; an organizer wants to retrieve some information about some exhibitors that attended the last edition; as usual, these data are stored in a database. The organizer is looking to some companies with certain characteristics, in order to send customized advertising, together with the invitation to attend the next event. In order to do this, DaeQP starts off from the window in figure 6, asking the user to select an attribute (from the drop-down menu) among the set of the attributes. In the example below, the user chooses "Exhibitors". When the user presses the "Next" button the interface shows the distribution of the Exhibitors along the selected other attributes (see Figure 7).

Note that in the EOSDIS system the main attributes are selected among those that appear in the sets "Geographical selection", "Parameter groups", and "Year selection". From this moment the DaeQP and EOSDIS have exactly the same behaviour as regards the interaction with the user. Each bar, pie chart and geographical grid is labeled with numbers that show how many items with that attribute value are in the selected data set. In DaeQP, starting from the configuration of Figure 7, the user selects companies with specific attribute values. For example, if he is interested in producers with a few employees that operate in north Italy, he clicks on the value or on the bar of each of these attributes. Figure 8 shows the final result after the user has clicked on the value "1-10" of Personnel, on the value "Produttore" of Company Type and on the value "Italia Settentrionale" (Northern Italy) of Current Markets (in analogy with figures 2, 3, 4 and 5).

A click on the Next button will show the list of the exhibitors in this data set; the user can use this list for his purposes, or he can go back to the overview in figure 7 for a different selection of attributes. He can decide to display a different attribute at any moment of the interaction, by clicking on the name of the attribute on top of the window in figure 7 (or figure 8). Similarly, he can remove a displayed attribute by clicking on the related attribute button.


DaeQP

Choose main attribute

(● Exhibitors) ○ Current Market ○ Turnover

○ Personnel ○ Company Type ○ Geographic Area

○ Product Category ○ Number of exhibited products

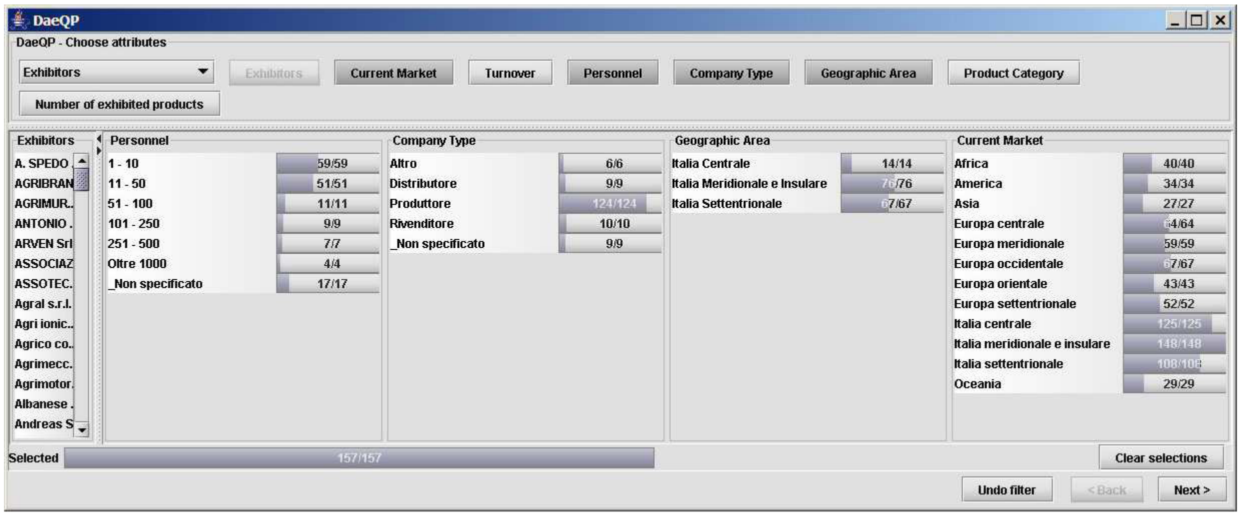
Order by attribute

☐ Exhibitors ☐ Current Market ☐ Turnover

☐ Personnel ☐ Company Type ☐ Geographic Area

☐ Product Category ☐ Number of exhibited products

Fig. 6. Initial selection of attributes


DaeQP

DaeQP - Choose attributes

Exhibitors: A. SPEDO, AGRIBRAN, AGRIMUR.., ANTONIO ., ARVEN Srl, ASSOCIAZ, ASSOTEC., Agral s.r.l., Agri ionic.., Agrico co.., Agrimecc., Agrimotor., Albanese ., Andreas S

Personnel	
1 - 10	59/59
11 - 50	51/51
51 - 100	11/11
101 - 250	9/9
251 - 500	7/7
Oltre 1000	4/4
_Non specificato	17/17

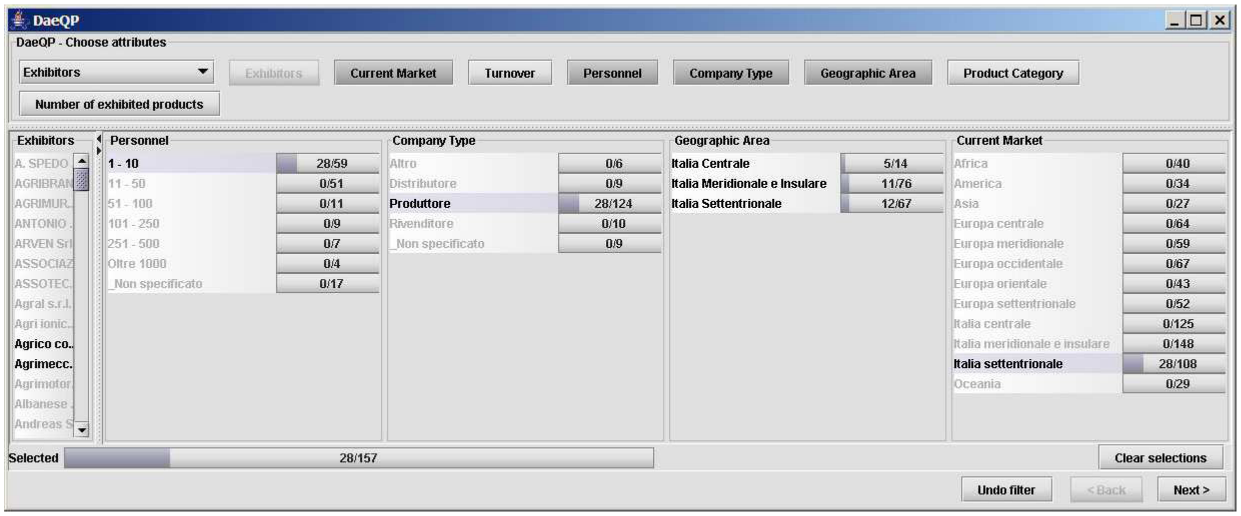
Company Type	
Altro	6/6
Distributore	9/9
Produttore	124/124
Rivenditore	10/10
_Non specificato	9/9

Geographic Area	
Italia Centrale	14/14
Italia Meridionale e Insulare	76/76
Italia Settentrionale	67/67

Current Market	
Africa	40/40
America	34/34
Asia	27/27
Europa centrale	64/64
Europa meridionale	59/59
Europa occidentale	67/67
Europa orientale	43/43
Europa settentrionale	52/52
Italia centrale	125/125
Italia meridionale e insulare	148/148
Italia settentrionale	108/108
Oceania	29/29

Selected 157/157

Fig. 7. Overview of data w.r.t attributes Personnel, Company type, Current Market, and Geographic Area


DaeQP

DaeQP - Choose attributes

Exhibitors: A. SPEDO, AGRIBRAN, AGRIMUR., ANTONIO ., ARVEN Srl, ASSOCIAZ, ASSOTEC., Agral s.r.l., Agri ionic., Agrico co.., Agrimecc., Agrimotor., Albanese ., Andreas S

Personnel	
1 - 10	28/59
11 - 50	0/51
51 - 100	0/11
101 - 250	0/9
251 - 500	0/7
Oltre 1000	0/4
_Non specificato	0/17

Company Type	
Altro	0/6
Distributore	0/9
Produttore	28/124
Rivenditore	0/10
_Non specificato	0/9

Geographic Area	
Italia Centrale	5/14
Italia Meridionale e Insulare	11/76
Italia Settentrionale	12/67

Current Market	
Africa	0/40
America	0/34
Asia	0/27
Europa centrale	0/64
Europa meridionale	0/59
Europa occidentale	0/67
Europa orientale	0/43
Europa settentrionale	0/52
Italia centrale	0/125
Italia meridionale e insulare	0/148
Italia settentrionale	28/108
Oceania	0/29

Selected 28/157

Fig. 8. Query preview after selecting ”1-10”, ”Produttore” and ”Italia settentrionale

IV. A formal model of the user interface

The behavior of interfaces like DaeQP or EOSDIS may be specified by a formal description; as explained in the Introduction, this should help the interface's designer, providing a precise and concise description of the user interface and facilitating a coherent extension of the current system with new functionalities. In this section we introduce a new formalism in order to describe the structure and the behaviour of the previously introduced visual interface. The meaning of the interface (that is, its operational semantics) represents the dialog between the interface and the underlying programs, and will be described by notation-specific semantics, that is pieces of programming languages attached to the interface description, or by formal specification notations. Our formalism is based on a type-inference notation, in which each variable is defined in the domain of the interface's basic objects (in this case, buttons, bars, and pie charts), and each transition rule represents the transition from a given configuration of the interface to the following one. As we will show in the rest of this section, this approach allow us to easily define dynamic interfaces, whose structure can change depending on the user's actions and on the database's values.

A. Types

We recall some definitions on types and type inferences; we will use them as the basis for our interface's specification language. For our purposes, we consider a type as a set of values, in which new values can be built up from the old ones.

If f is a function from A to B we say that f has *type* $A \mapsto B$. In general, we can form type expressions that denotes sets of values, by means of the following rules:

1) The names of sets are type expressions (for example, $\mathcal{N} = \{0, 1, 2, \ldots\}$ is the set of natural numbers, or $\mathcal{ID}$ is the set of identifiers of a programming language);
2) if $A_1, \ldots, A_n$ are types (sets), then $A_1 \times \ldots \times A_n$ is a type expression (the *Cartesian product* of $A_1, \ldots, A_n$);
3) if A and B are types (sets), then $A \mapsto B$ is the set of *functions* from A to B.

We write $x : t$ to indicate that the value of the variable x belongs to type t. An expression is *well-typed* if all operators and functions in the expression are applied to arguments of a suitable type. If e is an expression, $e : t$ denotes that e is well typed and its value has type t (for example, $3 + 5 : N$).

A *type environment* $\tau = [x \mapsto t, ...]$ maps x into its type. It represents the assumptions that are made about x. Given an expression e which contains some occurrences of x, the type of e can be determined only when the type environment τ is defined.

A type inference is based on a set of type inference rules, one for each expression in the language we are describing. For example, consider an expression e_1+e_2, which is the sum of two sub-expressions, each of which is well-typed and has type B in the type environment τ. Then the entire expression is well-typed and has type B, and this is expressed by the following two-premises inference rule:

$$\frac{\tau \vdash e_1 : \mathcal{B} \quad \tau \vdash e_2 : \mathcal{B}}{\tau \vdash e_1 + e_2 : \mathcal{B}}$$

The part above the line is called the *premise*. Now consider an expression (e_1, e_2), which build a pair. If e_1 has type t_1 and e_2 has type t_2, and both are well-typed, then the pair is well-typed and has type $t_1 \times t_2$:

$$\frac{\tau \vdash e_1 : t_1 \quad \tau \vdash e_2 : t_2}{\tau \vdash (e_1, e_2) : t_1 \times t_2}$$

Using type inference rules, the type of a complex expression can be inferred from the assumptions held in the type environment τ. For example, with $\tau = [m \mapsto \mathcal{N}, n \mapsto \mathcal{N}]$ we have

$$\frac{\dfrac{\tau \vdash m : \mathcal{N} \quad \tau \vdash n : \mathcal{N}}{\tau \vdash m + n : \mathcal{N}} \qquad \dfrac{\tau \vdash m : \mathcal{N} \quad \tau \vdash n : \mathcal{N}}{\tau \vdash m - n : \mathcal{N}}}{\tau \vdash (m + n, m - n) : \mathcal{N} \times \mathcal{N}}$$

B. Static description of the interface

In the rest of the paper, we have that

- $P = P_1, P_2, \ldots$ is the set of *principal categories*; $S = S_1, S_2, \ldots$ is the set of *secondary categories*;
- each category (principal or secondary) is a set of *attributes*;
- $States = \{s, d\}$ is the set of *button states* (selected or deselected).
- given a set A, the number of elements of A is denoted with $\#(A)$.

Definition 4.1: A *sequence of principal attributes* is the l-ple

$$ssp := (ssp_1, \ldots, ssp_l)$$

where:

1) $ssp_j := ((p_{j1}, f_{j1}), \ldots, (p_{jn}, f_{jn})) : (P_j \times States)^n$, with $j = 1 \ldots l$ and $n = \#(P_j)$;
2) $p_{ji} \neq p_{jk}$, if $i \neq k$, with $j = 1 \ldots l$ and $i, k = 1 \ldots \#(Pj)$;
3) for each j, there exists at most one i such that $f_{ji} = s$.

Given a sequence of principal attributes $ssp = (ssp_1, \ldots ssp_l)$, with $ssp_j = ((p_{j1}, f_{j1}), \ldots, (p_{jn}, f_{jn}))$, the reader understands from the previous definition that each attribute p_{ji} is selected from the set of principal categories P_j, and that it is associated with a state $f_{ji} : \{s, d\}$; this means that an attribute can be either selected or deselected. Moreover, each attribute appears only once into the list of attributes, and only one attribute into this list can be selected. For instance, figures 1 and 2 show the selection of a sequence of principal attributes.

Definition 4.2: An *initial sequence* is a sequence of principal attributes with $f_{ji} = d$, for each i and j.

Indeed, an initial sequence is a sequence of principal attributes that are all deselected; this corresponds to the initial (empty) view of the panel of a query preview system.

Definition 4.3: Given a sequence of principal attributes $ssp := (ssp_1, \ldots, ssp_l)$, the *semantic* of ssp is the $v+1$-ple

$$[ssp] := (\pi_1, \ldots, \pi_v)$$

where:

1) $\pi_q = \emptyset$; or
2) $\pi_q = ((s_{q1}, n_{q1}, state), \ldots, (s_{qr}, n_{qr}, state)) : (S_q \times N \times States)^r$, with $q = 1 \ldots v$; and
3) the numbers $n_{q1}, \ldots, n_{qr}$ are the result of statistical analysys made on the database, and they depend on the sequence of principal attributes ssp.

The semantic of a sequence of attributes represents what appears in the lower part of the screen of a query preview system's interface, when that sequence has been selected; in figure 2, 7 and 8, the semantic is a set of secondary attributes, together with the related statistical information. Note that some among the sets can be empty and will be specified later, when more principal attributes will be selected (as in figure 1), and that the secondary attributes are themselves selectable in order to filter the information.

Each set π_q represents the semantic associated to the selection of principal attributes in the sequence ssp. Its elements are triples (*secondary attribute*, *counter*, *state*), where all the secondary attribute are initially deselected (in what follows we will introduce a rule to describe what happens when a secondary attribute is selected), and the number *counter* is the result of statistical analysis on the whole database. The set π_q is empty when all the flags of the corresponding principal attribute are deselected, meaning that the user does not want to analyze the section of the database associated to that attribute. For example, in figure 1 the elements of the attribute "Parameter Groups" are all deselected, and the related piece of semantic (the lower left part of the screen) is still empty; when the user selects "Atmospherics Dynamics" among them, the set of secondary attributes appears on the screen, with the related bars and numbers, providing a new view corresponding to that choice. The same happens in figure 7. We will describe how to formalize this behaviour of the interface in the following section.

C. The dynamic behaviour of the interface

In systems similar to DaeQP and EOSDIDS, the transition from a view to another view is steered by the user's activity and by the statistical functions that the system computes. We describe all these events by means of a type-inference system, in which the premises are based on the current configuration of the system (i.e. on the sequence of principal and secondary attributes) and on the action performed (i.e. on the selected attribute). The consequence represents the new configuration as a result of the action. To each sequence of attributes we associate one or more semantics; the way the semantics change is described by means of the following rules. We omit the type environment, when straightforward.

Definition 4.4: Given a sequence of principal attributes $ssp := (ssp_1, \ldots, ssp_l)$, with $ssp_j := ((p_{j1}, f_{j1}), \ldots, (p_{jn}, f_{jn})) : (P_j \times States)^n$, with $j = 1 \ldots l$ and $n = \#(P_j)$, and given the related semantic $[ssp] := (\pi_1, \ldots, \pi_v)$, with $\pi_q = ((s_{q1}, n_{q1}, state), \ldots, (s_{qr}, n_{qr}, state)) : (S_q \times N \times States)^r$, with $q = 1 \ldots v$, the *choice-of-principal-attribute rule* is

$$\frac{(ssp, [ssp]) \qquad (p_{ji}, f_{ji}) : P_j \times States}{(ssp', [ssp'])}$$

where:

1) ssp' is obtained by (a) setting to s the flag f_{ji} in ssp, and (b) setting to d the flag that was previously set to s into ssp, if any; and
2) $[ssp']$ depends on the new sequence of principal attributes ssp' and on the old semantic $[ssp]$.

This rule is introduced in order to describe the step-by-step selection of the sequence of principal attributes: given a category, the user chooses one among the attributes belonging to that category. For example, the user clicks on the "By Continents" button into the "Geographical selection" scroll-down menu in figure 1, obtaining a view (a semantic) corresponding to this choice; then, he clicks for the first time on the "Atmosphere Dynamics" button into the "Parameters groups" menu, obtaining a new view, in which a new set of secondary parameters appears on the left bottom side of the screen. He can now click again on a different attribute of one of the categories, changing the semantic again and again. In this example, the previous rule has been applied twice. In figure 8, the same rule has been applied four times: the principal attributes "Geographic Area", "Company Type", "Personnel" and "Current Market" have been selected, producing the semantic that appears on the lower part of the screen (the four columns with attributes and data), and leaving the other five attributes unselected.

Definition 4.5: Given a sequence of principal attributes $ssp := (ssp_1, \ldots, ssp_l)$, with $ssp_j := ((p_{j1}, f_{j1}), \ldots, (p_{jn}, f_{jn})) : (P_j \times States)^n$, with $j = 1 \ldots l$ and $n = \#(P_j)$, and given the related semantic $[ssp] := (\pi_1, \ldots, \pi_v)$, with $\pi_q = ((s_{q1}, n_{q1}, state), \ldots, (s_{qr}, n_{qr}, state)) : (S_q \times N \times States)^r$, with $q = 1 \ldots v$, the *choice-of-secondary-attribute rule* is

$$\frac{(ssp, [ssp]) \qquad (s_{ji}, f_{ji}) : (S_j \times States)}{(ssp', [ssp]')}$$

where:

1) the flag f_{ji} in $[ssp]$ is set to s and the flag that was previously set to s into $[ssp]$ is set to d, if any, obtaining and intermediate semantic $[ssp]^*$;
2) $[ssp]'$ depends on the new intermediate semantic $[ssp]^*$ and on the old semantic $[ssp]$.

This rule is introduced in order to describe the step-by-step selection of the sequence of secondary attributes: given a selection of principal attributes and the related semantic, the user can choose some among the secondary attributes. For example, the user clicks on the "Sea surface temperature" button into the "Atmospheric Dynamics" menu, as in figure 1; then, he clicks on the "1992" button into the "Year Selection" menu, as in figure 4; then, he clicks on "North America", as in figure 5. Each time the user selects an attribute among the secondary categories, he gets a new semantic, that is a new set of statistical information (see definition 3). In figure 8, this rule has been applied three times, selecting the secondary attributes "1-10", "Produttore", and "Italia settentrionale". Note also that the sequence of principal attributes remains unchanged, while the new semantic depends on a intermediate one which is obtained by changing appropriately the flags into the old semantic.

Definition 4.6: Given a query preview system, a *session* of the system is a type-inference binary tree, where:

1) each leaf is labeled with either $(ssp, [ssp])$, or $(p, f_p) : P_i \times States$, or $(s, f_s) : S_j \times States$; and
2) each node is obtained by applying one between the choice-of-principal-attribute rule or choice-of-secondary-attribute rule.

In summary, the transition from a view to another can be obtained in two different ways: starting from an initial sequence of attributes, the user can repeatedly select or deselect items among the principal attributes, and he can do the same among the secondary attributes, if any; in our notation, this is represented by a type-inference tree. Note that the elements of a generic semantic assume the same role of the elements of a generic sequence of principal attributes; thus, the user can select or deselect them, according to his needs.

V. Conclusions and further work

The possibility of reasoning and proving properties of a session of an interactive system is one of the advantages of using formal notations. For example, we could reason about the interaction between the interface and the human cognitive system, or about the properties of the interface, in order to modify it, or to have a proper feedback. This is easily done by an inspection of the inference tree (a session), seeking for path or states that can be reached. Among the properties that seems easy to verify there are; reachability (a user interaction can generate an effect on a part of the user interface); visibility (each user action is associated with a modification of the user interface); complementarity (the possibility to reach the same configuration of the interface following two or more different modalities); assignment (the possibility to reach a configuration in a unique way); minimal path (if a given configuration can be reached in different ways, we can find the quickest way to do it); absence of deadlock (the same configuration appears more than once in a session). Another advantage of a formal notation is to have a sound set of tools for the evolution of the software. A sensible prosecution of this work could be the further development of the formalism, in order to capture more generic interface's models; a detailed comparison with other well-known formalisms; and some on-field usability tests. Moreover, a clear relation between user-interface properties and tree-inference properties cold be provided.

References

[1] P. Buono, G. Pani,E. Covino and F. Costabile, *A visual tool for multidimensional data analysis*, International workshop of Visual Language and Computing, VLC '05, Banff, 2005.

[2] K. Doan, C. Plaisant and B. Shneiderman, *Query Previews in Networked Information Systems*, Proceedings of the 3rd International Forum on Research and Technology Advances in Digital Libraries, ADL '96, IEEE, 1996.

[3] M. Erwig, *Abstract syntax and semantics of visual languages*, Journal of Visual Languages and Computing, vol. 9, pp. 461-483, 1998.

[4] F. Ferrucci, G. Tortora, M. Tucci and G. Vitiello, *Relation grammars: a formalism for syntactic and semantic analysis of visual languages*, Visual language theory, Springer-Verlag New York, Inc, pp. 219-243, 1998.

[5] E. J. Golin and S. P. Reiss, *The specification of visual language syntax*, Journal of Visual Languages and Computing, vol. 9(2), pp. 141-157, 1990.

[6] J. M. Gooday and A. G. Cohn, *Visual language syntax and semantics: A spatial logic approach*, Proc. AVI '96 Workshop on Theory of Visual Languages, 1996.

[7] R. Helm and K. Marriott, *A declarative specification and semantics for visual languages*, Journal of Visual Languages and Computing, vol. 2, pp. 311-332,1996.

[8] P. Curzon, R. Ruksenas and A. Blandford, *An approach to formal verification of human-computer interaction*, Formal Aspects of Computing (2007) 19: 513-550.

[9] K. Marriott, B. Meyer and K. B. Wittenburg, *A survey of visual language specification and recognition*, Visual language theory, Springer-Verlag New York, pp. 5-85, 1998.

[10] M. Mezzanotte and F. Patern, *Verification of Properties of Human-Computer Dialogues with an Infinite Number of States*, Proc. of the BCS-FACS Workshop on Formal Aspects of the Human Computer Interface, Sheffield Hallam University, 1996.

[11] M. A. Najork and S. M. Kaplan,*Specifying visual languages with conditional set rewrite systems*, Proceedings of the IEEE Workshop on Visual Languages, IEEE, New York, pp. 12-18, 1993.

[12] P. Palanque, R. Bastide, L. Dourte and C. Sibertin-Blanc, *Design of User-Driven Interfaces Using Petri Nets and Objects*, CAiSE '93: Proceedings of Advanced Information Systems Engineering, Springer-Verlag, vol. 17(3), pp. 569-585, 1993.

[13] C. Plaisant, B. Shneiderman, K. Doan and T. Bruns, *Interface and data architecture for query preview in networked information systems*, ACM Trans. Inf. Syst., ACM, vol. 17(3), pp. 320-341, 1999.

[14] C. Plaisant, M. Venkatraman, K. Ngamkajornwiwat, R. Barth, B. Harberts and W. Feng, *Refining Query Previews Techniques for Data with Multivalued Attributes: The Case of NASA EOSDIS*, Advances in Digital Libraries Conference, http://doi.ieeecomputersociety.org/10.1109/ADL.1999.777690, IEEE, pp.320-341, 1999.

[15] H. Thimbleby, *User interface design with matrix algebra*, Journal of ACM Transactions on Computer-Human Interaction, vol. 11.2, June 2004.

[16] D. Varr, G. Varr and A. Pataricza, *Designing the automatic transformation of visual languages*, Science of Computer Programming, vol. 44(2), pp. 205-227, 2002.

[17] D. Wang and J. R. Lee, *Visual Reasoning: its Formal Semantics and Applications*, Journal of Visual Languages and Computing, vol. 4(4), pp. 327-356, 1993.