INFORMATICA, 1995, Vol. 6, No. 4, 445–456

BUILDING STANDARD USER INTERFACE: DBMS "MICROPOISK" APPROACH

Valery O. GRECHKO and Vadim G. TULCHINSKY

Glushkov Institute of Cybernetics
Glushkov avenue 40, 252207 Kiev, Ukraine

Abstract. The algebra–grammar model of user interface (UI) and the method of UI creation and updating by combining the visual formalisms are proposed. The UI building tools for applications of "MicroPoisk" DBMS are described. Special attention is paid to the requirements of the UI standrads.

Key words: user interface, model, database, object, prototype, visual formalism.

1. Introduction. At present, commercial DBMSs propose users two classes of tools to support the creation of user interface (UI): firstly, toolkits and, secondly, UI editors and manager systems (UIMS). UI editors and UIMS are based on particular sets of graphic components. They permit description of the UI graphically. Thus the development of interface is considerably simplified and accelerated.

Unfortunately, there are many tasks which cannot be described by a set of standard objects, or such description will not be evident or useful. The only way to create UI for such tasks is programming with the help of toolkits.

The new "post-UIMS" technology of UI development appeared recently. It is presented by such systems as Nephew (Szekley, 1989), UI Development Environment (Foley *et al.*, 1988), Unidraw toolkit (Vissides and Linton, 1990), Workspaces UIMS (Olsen *et al.*, 1992) and Application Creation Environment (Jonson *et al.*, 1993). The widgets manipulation has been replaced in this technology by the logic data structures manipulation. The data structures are displayed by adapting visual applied modules, called *visual formalisms* (VF). The term "visual formalism" was offered by David Harel (1988). VF is "visual" because it should express and interpret semantics for a user through a set of visual forms. And VF is a "formalism" because it is manipulated and analyzed

by a computer and hence VF must satisfy formal requirements.

The significant part of "post-UIMS" tools is based on databases. Most of Object–Oriented DBMS (ONTOS, Object Store (Wilson, 1992), O2 (Borras *et al.*, 1992), Face Kit (King and Novak, 1992)) and Rapid Application Developers for Client–Server architecture (Michailov, 1995) use "post-UIMS" ideas.

The offered paper is devoted to the methods of the UI creation and updating for applications of "MicroPoisk" (Grechko *et al.*, 1993) entity–relationship DBMS. Our research is concentrated on combining the VF to develop the UIs for complex applications. Special attention is paid to the requirements of UI Standards.

2. The algebra–grammar UI model. Windowed UI is formed of windows with three types of dialogue. According to the Common User Access standard of IBM, they are *primary*, *secondary* and *pop-up*. A dialogue can be opened by user or by program. Then some action can be performed. In particular, child window (subdialogue) can be opened. Then it can be closed (finished). VF corresponds to a field of a window. VF can be considered as a secondary dialogue, but VF is an indecomposable element of UI. A dialogue (compound dialogue) manipulates data by the VFs corresponded to the appropriate data types.

Most primary and some secondary dialogues may have unlimited number of examples simultaneously. Without loss of generality we'll consider configuration of UI with fixed (sufficient) maximal example numbers for each primary dialogue and secondary subdialogue of any dialogue.

A *state of UI* may be described as a bounded set of opened dialogues. If a pop-up dialogue is opened then its parent dialogue and all its brotherly dialogues with all their posterity are inaccessible for user in the appropriate state of UI. Any state of UI may be derived by the sequence of the "open dialogue" operations from the initial state. There is the only pseudo-dialogue of the program start accessible in the initial state of UI.

Let two dialogues are *equivalent* if their sets of VFs are equal. Let two states of UI are *equivalent* if their sets of accessible dialogues are equal. Let two UIs are *equivalent* if they have the same set of dialogues and the same set of primary dialogues and there are one-to-one correspondings of the "dialogue $\rightarrow$ pop-up subdialogue" and the "dialogue $\rightarrow$ secondary subdialogue" relationships in both ones.

The enterprise model is considered as the hierarchy of data types $(T, \leqslant_t)$ and the set of the necessary tasks D. Each task from D is described by a set of data types from T. Simultaneous access (for viewing, updating, choosing, etc.) to data of these types is necessary for solving of the task. This set is named the *information set of the task.* The set of types of data corresponded to the VFs of all accessible dialogues is named *information set of the state of UI.*

The *Grammar of object representation of interface* (GORI) G is described by (T, N, A, P', P'') (Grechko *et al.*, 1994). Here the set of terminal symbols T is the set of data types from the enterprise model; axiom A corresponds to the initial pseudo-dialogue (dialogue of the program start); the set of nonterminal symbols $N = \{A\} \cup N^0$, where N^0 is the set of dialogues of the UI windows; the set P' includes the production p_0^A: $A \rightarrow W_1 W_2 \cdots W_n$ from axiom and the only production of kind p_0^B: $B \rightarrow C_1 C_2 \cdots C_m$ from any other nonterminal; the set P'' includes any productions of the type p_i^B: $B \rightarrow O_i$. Here W_i is the i-th primary dialogue; C_i is a secondary subdialogue of the dialogue B if nonterminal or a VF corresponded to appropriate data if terminal; O_i is i-th pop-up subdialogue of the dialogue B.

The sentential forms of the GORI G are considered as elements of the algebra system $C(N \cup T)$ of strings of terminal and nonterminal symbols. Its signature includes the operation of free multiplication $\cdot$ (string concatenation) and following 3 relations:

1. $=_c$: $C(N \cup T) \times C(N \cup T) \longrightarrow E_2 = \{\text{FALSE, TRUE}\}$.
   For arbitrary strings β and γ from $C(N \cup T)\beta =_c \gamma$ if and only if the sets of symbols of strings β and γ are equal. Therefore every set of terminal and nonterminal symbols may be looked on as a string of such symbols in the free order.
2. $\leqslant_c$ $C(N \cup T) \times C(N \cup T) \longrightarrow E_2$
   $\beta \leqslant_c \gamma$ if and only if the set of symbols of string β is present the subset of set of symbols of string γ.
3. $\leqslant_t$: $C(N \cup T) \times C(N \cup T) \longrightarrow E_2$
   $\beta \leqslant_t \gamma$ if and only if to each symbol x of string β there is the same symbol of string γ, or (for terminal symbols) such terminal y of string γ, that $x \leqslant_t y$ in the hierarchy of enterprise model data types.

Let GORI $G_1 = (T, N_1, A_1, P_1', P_1'')$ is equivalent GORI $G_2 = (T, N_2, A_2, P_2', P_2'')$ if such identity mapping f: $N_1 \longrightarrow N_2$ exists that

1. $f(A_1) = A_2$;
2. $\forall p_1 \in P_1'$: $B_1 \to C_1^1 C_1^2 \cdots C_1^n \exists p_2 \in P_2'$: $sym(p_2) = f(B_1) \wedge res(p_2) =_c f(C_1^1) f(C_1^2) \cdots f(C_1^n)$ and $\forall p_2 \in P_2'$: $B_2 \to C_2^1 C_2^2 \cdots C_2^m \exists p_1 \in P_1'$: $sym(p_1) = f^{-1}(B_2) \wedge res(p_1) =_c f^{-1}(C_2^1) f^{-1}(C_2^2) \cdots f^{-1}(C_2^m)$;
3. $\forall p_1 \in P_1''$: $B_1 \to C_1 \exists p_2 \in P_2''$: $sym(p_2) = f(B_1) \wedge res(p_2) = f(C_1)$ and $\forall p_2 \in P_2''$: $B_2 \to C_2 \exists p_1 \in P_1''$: $sym(p_1) = f^{-1}(B_2) \wedge res(p_1) = f^{-1}(C_2)$

Here and later $sym(p)$ is a left part of a production p and $res(p)$ is right part of a production p.

The introduced relation of GORI is equivalention by the definition of $\leqslant_c$.

Properties of GORI are described by the next simple theorems.

Theorem 1. *There is one-to-one correspondence between the set of UIs and the set of GORIs in given enterprise model.*

Proof. It is enough to describe UI corresponding to the given GORI $G = (T, N, A, P', P'')$. Dialogues of resulting UI correspond to nonterminals of the GORI without an axiom. Each dialogue includes VFs corresponding to data types of the terminals from the right part of the appropriate production belonging to P'. Primary dialogues correspond to the nonterminals enumerated in the right part of the production from A. Any other production belonging to P' gives rise to the set of "dialogue $\to$ secondary subdialogue" relationships between the nonterminals from its left part and all nonterminals from its right part. The "dialogue $\to$ pop-up subdialogue" relationships should be set according to productions of P''. According to definition resulting UI is equivalent to the initial one. The theorem has been proved.

Theorem 2. *If $\beta \in C(T)$ is the information set of UI then there is a sentential form $\gamma \in C(N \cup T)$ of its GORI $G = (N, T, A, P', P'')$ such that $\beta \leqslant_c \gamma$.*

Proof. Let X_0 is a state of the UI with the information set β. Then there exists a sequence of the opened dialogues which derives X_0. Let us exclude all inaccessible dialogues from this sequence and group brotherly dialogues in the sequence after the first one. Resulting X_1 set of the UI is equivalent to X_0. By definition its information set $\beta_1 =_c \beta$. Let us complete each group of brotherly dialogues by all other possible subdialogues of their parent. Resulting

X_2 set of the UI has information set β_2. All dialogues of X_2 are accessible and therefore $\beta \leqslant_c \beta_2$. Opening of each group of brotherly dialogues of last sequence corresponds to the production of G from the nonterminal of their parent dialogue. If the group consists of a pop-up dialogue then the production belongs to P'' otherwise it belongs to P'. Thus a corresponding sequence of the GORI productions is obtained. The application of each production adds the VFs data types of the dialogue of its left part to the set of terminals of the resulting string. Let the sequence of productions derives the string γ_0. If the belonging to P' productions from all nonterminals are applied to the γ_0 string then the searching γ string is derived. $\beta \leqslant \gamma$ because $\beta_2 \leqslant_c \gamma$ by definition. The theorem has been proved.

Lemma 1. *The dialogue can be accessible in some state of the UI if and only if its nonterminal is accessible in the appropriate GORI.*

Proof. First conclusion of the lemma can be proved similarly to Theorem 2. If B is an accessible nonterminal then there is a sequence of productions that derives a string β: $B \leqslant_c \beta$. Appropriate sequence of the "open dialogue" operations derives the state of UI with an accessible dialogue B. Lemma has been proved.

Let UI is *nonredundant* if it has neither useless windows nor recursive subordination of windows.

Lemma 2. *GORI of the nonredundant UI is finite and UI of the finite GORI without inaccessible nonterminals is nonredundant.*

Proof follows from the definition and the similar properties of the appropriate context–free grammar.

Theorem 3. *For any non-axiom nonterminal $B \in N \setminus \{A\}$ of nonredundant GORI $G = (T, N, A, P', P'')$ there exists context–free grammar $G_B = (T, N, B, P')$ which language consists of the only string.*

Proof. The language of the G_B is not empty because G has no recursive nonterminals and P' has a production from each nonterminal. The language has no more than one element because such production is unique. The theorem has been proved.

Therefore the only terminal string is derived from every non-axiom nonterminal B by productions of set P'. This string is called the *information set of*

dialogue B.

Thus the analysis of GORI permits the supervision of the properties of appropriate UI, in particular:

- applicability of structure of the given UI for solving of some set of tasks (completeness). The task with the information set β may be solved only if the GORI has a sentential form γ: $\beta \leqslant_c \gamma$;
- absence of useless windows, recursive subordination of windows and dialogue type mismatch (consistency). The GORI of a consistent UI has neither inaccessible nor recursive nonterminal. Absence of dialogue type mismatch is described by a logic expression

$$res(p^A) \bigcap \prod_{p \in P' \setminus \{p^A\}} res(p) = \varnothing$$

and

$$\prod_{p \in P'} res(p) \bigcap \prod_{p \in P''} res(p) = \varnothing;$$

- conformity to the requirements of UI Standards to semantics of interaction of dialogues: presentativity, compatibility and context. According to presentativity, for every information set β of the task from D there should be a nonterminal B with the equal information set γ: $\beta \leqslant_c \gamma$. According to compatibility, the GORI should not include two different nonterminals with the equal information sets. According to context, a nonterminal with the information set β may be derived from a nonterminal with the information set γ only if $\beta \leqslant_t \gamma$.

Let GORI is *correct* if its UI is consistent, complete in given enterprise model and satisfy to the UI Standards requirements.

Let us introduce some operations of the correct GORI transformation.

1. $+$: $\Gamma(T, \leqslant_t, D) \times (N, N) \longrightarrow \Gamma(T, \leqslant_t, D)$
   The $G + (B, C)$ expression means the setting of the child–parent relationship between the B and C dialogues. The dialogue B becomes the secondary subdialogue of C if result GORI is correct. Otherwise B becomes a pop-up subdialogue of C. This operation doesn't change GORI if a result of such transformation is not correct.
2. $-$: $\Gamma(T, \leqslant_t, D) \times (N, N) \longrightarrow \Gamma(T, \leqslant_t, D)$
   The $G+(B, C)$ expression means the break of the child–parent relationship between B and C dialogues. If there isn't such relationship then

transformation isn't performed. If C was the only parent of B in GORI G then B becomes a primary dialogue.

3. $\oplus$: $\Gamma(T, \leqslant_t, D) \times N \setminus \{A\} \longrightarrow \Gamma(T, \leqslant_t, D)$
   The $G \oplus B$ expression means the transformation of a secondary dialogue B to the pop-up. If B is not a secondary dialogue of the GORI G then G is not changed.
4. $\otimes$: $\Gamma(T, \leqslant_t, D) \times C(T) \longrightarrow \Gamma(T, \leqslant_t, D)$
   The $\times(G, \beta, B)$ or simple $G \times \beta$ expression means the addition of the new primary dialogue (B) with the information set β to the GORI G. If G has dialogue B or other dialogue with the information set β then GORI is not changed.

Let us define the signature $S = (+, -, \oplus, \otimes)$ and all the operations of the signature have the equal preferences and execute from left to right.

Lemma 3. *Let $G = (T, N, A, P', P'') \in \Gamma(T, \leqslant_t, D)$ and both B and C belong to $N \setminus \{A\}$ and $p \in P' \cup P''$: $C =_c sym(p)$ and $B \leqslant_c res(p)$. If $p \in P'$ then $G - (B, C) + (B, C) = G$ else $G - (B, C) + (B, C) \oplus B = G$.*

Proof. If C is not the only parent of B then $G-(B,C)+(B,C) = G$ and if B is a pop-up dialogue then $G \oplus B = G$. Else if B is a secondary subdialogue of C then according to presentativity its information set is the subset of the information set of C. Therefore after the evaluation of $G - (B, C) + (B, C)$ the dialogue B again becomes a secondary subdialogue of C and G is restored. According to context $G-(B,C)+(B,C) \oplus B$ is similarly equal G for pop-up B. The lemma has been proved.

Theorem 4. *The set $\Gamma(T, \leqslant_t, D)$ of the correct in given enterprise model GORIs and the set $N \setminus \{A\}$ of all nonterminals without the common axiom and the set $C(T)$ of strings of terminal symbols with signature $S = (+, -, \oplus, \otimes)$ form the 4-based algebra $A(N, \leqslant_t, D)$ and the algebra basis includes the only GORI.*

Proof. The set $\Gamma(T, \leqslant_t, D)$ is closed under operations of the signature according to the definition.

Let $G_0 = (T, N_0, A, P_0', P_0'') \in \Gamma(T, \leqslant_t, D)$ and the set of the information sets of its dialogues is equal to D. Let $G = (T, N, A, P', P'') \in \Gamma(T, \leqslant_t, D)$. Let us prove that G_0 can be transformed to G by the operations of algebra signature. Let us execute the $G_0-(B,C)$ operation for the each $(B, C) \in N_0 \times$

N_0 pair of nonterminals of G_0 GORI and then operation $\otimes(G_0, \inf(M), M)$ for any $M \in N \setminus N_0$. Resulting G_1 GORI has the primary dialogues only. Let Q_1 is a sequence of the $G-(B,C)$ operations for each $(B,C) \in N \times N$ pair of the nonterminals of G GORI. Execution of Q_1 transforms G to the G_1 GORI too. Let Q_2 is a subsequence of Q_1 including the all operations that are not identity mapping of GORI. The opposite transformation exists for each operation of Q_2 according to Lemma 3. Let Q is the inverse to Q_2 sequence of opposite transformations. Its execution changes GORI G_1 to G. The theorem has been proved.

Trivial basic GORI of the algebra consists of the primary dialogues only. More interesting basic GORI is built from trivial by the next algorithm.

$G := G_0$;
$S := N \setminus \{A\}$;
if $\exists B, C \in S$: $B \neq C$ then
  $M := \min(S); S := S \setminus \{M\}$;
  if $\exists X \in S$: $\inf(M) \leqslant_c \inf(X)$ then
    $W := \bigcup_{Y \in S:\ \inf(M) \leqslant_c \inf(Y)} Y$;
    while $\exists Y, Z \in W$: $\inf(Y) \leqslant_c \inf(Z) \wedge$
        $\neg \inf(Z) =_c \inf(Y)$ do
          $W := W \setminus \{Z\}$;
    end-while;
  else if $\exists X \in S$: $\inf(\min(S)) \leqslant_t \inf(X)$ then
    $W := \bigcup_{Y \in S:\ \inf(M) \leqslant_t \inf(Y)} Y$;
    while $\exists Y, Z \in W$: $\inf(Y) \leqslant_t \inf(Z) \wedge$
        $\neg \inf(Z) \leqslant_t \inf(Y)$ do
          $W := W \setminus \{Z\}$;
    end-while;
  else
    $W := \varnothing$;
  end-if;
  while $\exists X \in W$ do
    $G := G + (M, X)$;
    $W := W \setminus \{X\}$;
  end-while;
end-if.

Here $S, W \subseteq N$; $\inf(B)$ is an information set of the dialogue B; $\min(S)$ is an element of a set S that for each $B \in S$ if $B \leqslant_t \min(S)$ then $\min(S) \leqslant_t B$.

$\min(S)$ exists in any finite S by transitivity of $\leqslant_t$.

3. "MicroPoisk" applications UI support. The "post-UIMS" tools are based on different protocols. Protocols are used to get corresponding data structures, the names and expected arguments for VFs of the lower level, necessary for task and supported by VF editing operations, etc. The replacement of complicated protocols by the search for information in special database is presented as an expedient step for the VF on databases. This special database is called *metabase* or repository.

The "MicroPoisk" metabase consists of the special graph descriptions of subschemes. This graph description includes information for VF coordination and is based on a *metagraph*. "MicroPoisk" tools include some VFs for manipulation of such graph and its bypassing trees.

Metagraph is a connected oriented graph. Every node of the metagraph corresponds to the class of entities. Every arc corresponds to the existing class of relationships between classes of entities corresponding to the nodes connected by this arc. The only node is selected and called the *metagraph root*. Different nodes of the metagraph may correspond to the same class of entities. In this case different sets of entities may be represented by them.

The virtual arcs are determined by queries. Thus any node of the metagraph represents whether a whole set of entities of corresponding class or its subset.

We call *metatree* the tree of the paths from a root node in a given metagraph. Consider metatree as the structural diagram of a hierarchical model of data. It is possible to construct the extension of a virtual database whose scheme is set by this structural diagram. The extension is a wood of trees. The nodes correspond to entities and the arcs correspond to relationships between appropriate entities. The root of each tree is some entity from a set represented by the metagraph root. Each tree is refered to as a *tree of the virtual database*. We call such maximal subtree of the tree of the virtual database that don't have an entity which is fixed to a node and its own descendant a *covering*. The wood of covering trees of virtual database is called an *information graph*.

An information graph provides the user's "view" on a database for special tasks and extremely simplified visual forms of queries by direct navigation. This VF is developed to grant users of databases with complicated logic data structure the evident and convenient interactive tools.

Metagraph is not only a means for subscheme description. It is a basic data

structure for the maintenance of VF coordination. "MicroPoisk" metabase includes relationships connecting the metagraph nodes with corresponding entities of VFs and computation programs. This permits:

- the simple registration and specification of a new VF by the creation of the corresponding entity of the metabase;
- the semantic specialization of VFs for different sets of entities;
- the simultaneous transforming of UIs by updating entity of a VF in the metabase;
- the VF hierarchy and coordination maintenance for the UI development on the base of GORI.

The enterprise model is evaluated directly from the database intentional if UI develops on the base of GORI. In particular, for entity–relationship models the hierarchy of data types may be developed as a combination of three relations:

1. the data type of an entity or a group attribute dominates on the data types of its own attributes;
2. an order of entity classes derives from relationships of the "one to many" type;
3. some additional problem–oriented order of sets of entities is determined by the arcs of the metagraph.

The data types hierarchy should be received by exception of strongly connected subgraphs (cycles) from the orgraphs of the relations and their combination.

4. Conclusion. Thus the algebra–grammar UI model is proposed. The model provides the UI architecture analysis and design. It is a tool for developing a correct UI that is complete in the enterprise model and satisfies the requirements of UI Standards. The graph subscheme and the VFs for database fragment visualization for entity–relationship databases are proposed. "MicroPoisk" DBMS is used as the example of evaluation of the structural part of enterprise model and the VF coordination support for the model is described.

REFERENCES

Borras, P., J.C.Mamou, D.Plateau, B.Poyet and D.Tallot (1992). Building user interfaces for database applications: the O2 experience. *Sigmod Record,* **21**(1), 32–38.

Foley, J., C.Gibbs, W.C.Kim and S.Kovacevic (1988). Knowledge–based user interface management system. *Proc. of the ACM Conference on Computer–Human Interaction,* NY: ACM Press. 67–72.

Grechko, V.O., Yu.V.Kapitonova and C.B.Pogrebinsky (1993). Instrumentalnie sredstva intelektualizacii baz dannych. *Kibernetika i sistemnyi analiz,* **2**, 120–129 (in Russian).

Grechko, V.O., E.N.Kulish and V.G.Tulchinsky (1994). Postroenie prilozenij dlya relyacionno–setevych baz dannych na osnove grafovych prototipov. *Upravlyayuchie sistemy i mashiny,* **6**, 83–88 (in Russian).

Harel, D. (1988). On visual formalism. *Comm. of ACM,* **31**, 514–520.

Jonson, J.A., B.A.Nardi, C.L.Zarner and J.R.Miller (1993). ACE: building interactive graphical applications. *Comm. of ACM,* **36**(4), 92–101.

King, R., and M.Novak (1992). Building reusable data representaions with FaceKit. *Sigmod Record,* **21**(1), 11–17.

Michailov, D. (1995). Obzor sredstv razrabotki prilozenyi klient/server. *Computer World,* **6**(29), 8–10 (in Russian).

Olsen, D.J., T.McNeill and D.Michell (1992). Workspaces: an architecture for editing collections of object. *Proc. of the ACM Conference on Computer–Human Interaction,* NY: ACM Press. 262–272.

Szekley, P. (1989). Standardizing the interface between application and user interface management systems. *Proc. of the ACM SIGGRAPH Symposium on User Interface Software and Technology,* 34–42.

Vissides, J.M., and M.A.Linton (1990). Unidraw: a framework for building domain–specific graphical editors. *ACM Trans. Inf. Syst.,* **8**(3), 237–268.

Wilson, P.R. (1992). *An overview of three commercial object–oriented DBMSs: ONTOS, ObjectStore, and O2,* **21**(1), 93–104.

Received November 1994

V.O. Grechko received the M.S. degree in automatic control systems from Moscow Physical Technical Institute in 1975, and PhD degree in computer science from the Glushkov Institute of Cybernetics in 1982. Since 1990 to present he has been a Senior Scientist at the Glushkov Institute of Cybernetics, Kiev. His research interests include software tools, databases, knowledge bases, user interface.

V.G. Tulchinsky received the M.S. degree in software from Moscow Institute of Electronic Engineering in 1991. He joined the Glushkov Institute of Cybernetics in 1993 and is currently a Research Assistant at Department of Programming Automatisation. His current research interests include software tools, enterprise models, databases and user interface.

APIE STANDARTIZUOTŲ NAUDOTOJO INTERFEISŲ KONSTRAVIMĄ, NAUDOJANT DUOMENŲ BAZIŲ VALDYMO SISTEMĄ „MIKROPOISK"

Valerij O. GREČKO, Vadim G. TULČINSKIJ

Darbe apžvelgiami metodai, skirti naudotojo interfeisui korporacijų informacinėse sistemose konstruoti. Nagrinėjami duomenų vizualizavimo moduliai, vadinami vizualiniais formalizmais. Ypatingas dėmesys skirtas interfeisų standartizavimui. Siūlomas algebrinis–gramatinis informacinės sistemos naudotojo interfeiso modelis, įgalinantis projektuoti išsamius, darnius ir tenkinančius nustatytus standartus interfeisus.