Control and Cybernetics

vol. **32** (2003) No. 1

Yet another object-oriented data model and its application[1]

by

Beata Jankowska

Department of Control, Robotics, and Computer Science
Poznań University of Technology

Abstract: In this paper we propose a certain object-oriented data model. It originates from the calculus for complex objects by Bancilhon and Khoshafian (1989). These two models differ mainly in terms of set objects interpretation. In our model a set object stands for a set of items denoting different forms of the same object. As a consequence, a new *sub object* relation must be defined. Then, new functions of *union, intersection* and *complement* are defined. It is proved that the new data model is not only a lattice, but also a Boolean algebra. Finally, the model is expanded into a new calculus for objects. It is shown that the calculus provides a firm background for some interesting query language.

Keywords: data model, calculus for objects, query language.

1. Introduction

Among all of the professional database management systems, the systems with object-oriented databases are now the main object of interest for researchers, ODMG (1997). The basic notions of an object-oriented data model are these of an "object" and a "partial order relation" defined on a set of all objects. By means of these notions one can easily represent a hierarchical world structure.

There were a number of different attempts to build formal foundations for object-oriented systems. The most known of them originate from AI theories. They consider representing the common-sense knowledge in structures called "frames" or "feature structures". Frames, Minsky (1974), are relatively large structures, which exemplify typical instances or cases. They inherit default assumptions that can be displaced when more specific information is available. Feature structures, Carpenter (1992), correspond to "tuples" in other models. Both kinds of structures seem to be good tools for creating systems with partial knowledge representation.

Other attempts to build such foundations are founded on logic. A leading example of this trend is LIFE, Ait-Kaci (1993). It reconciles styles from functional programming, logic programming, and object-oriented programming. From the theoretical point of view, LIFE implements a constraint logic programming language with equality (unification) and entailment (matching) constraints over order-sorted feature terms.

The algebraic approach to a subject of defining object-oriented systems is quite rare. We can mention here the proposals of CA algebra, Nilsson (1993), or the AQUA data model and algebra, Leung et al. (1993).

The data model presented in this paper belongs to the algebraic trend. It comes down from the "calculus for complex objects" by Bancilhon and Khoshafian (1989). This calculus views objects in a broad manner, allowing the three different forms of them: elementary (atom or special), tuple or set. On the set **O** of all these objects the partial order *sub_object* relation ($\leq$), the *union* function ($\cup$) and the *intersection* function ($\cap$) are defined. It is proved that the algebra $A = (\mathbf{O}, \{\cup, \cap\})$ is a lattice. The notion of object is expanded to the notions of *object_formula* and *object_rule*, being an ordered pair of object formulae. Then, the functions of object formula *interpretation*, object rule *application* and object *closure* are defined. By their means the semantics of object formulae and the fixpoint semantics of a set of rules are specified.

2. Yet another calculus for objects

Let us change the presented calculus in some respects now. First of all, let us give the notion of a set object a new, intuitively clear meaning. From now on, a set object will stand for a set of items denoting different forms of the same object. At a given moment the object can assume only one of these forms.

In order to simplement this change let us remind the definition of an object by Bancilhon and Khoshafian. *Objects* are defined recursively there, as:

- integers, floats, strings, and booleans (we call them atomic objects),
- two special objects TOP ($\top$ – the inconsistent object) and BOTTOM ($\bot$ – the undefined object),
- tuple objects of the form: $[a_1{:}\, o_1;\ a_2{:}\, o_2;\ \ldots;\ a_n;\ o_n]$, where $a_1, a_2, \ldots, a_n$ are distinct attribute names and $o_1, o_2, \ldots, o_n$ are some objects,
- set objects of the form $\{o_1, o_2, \ldots, o_n\}$, where $o_1, o_2, \ldots, o_n$ are pair-wise different (not the same) objects[2].

Let **O** stand for the set of all such objects. Let A stand for the set $\{a_1, a_2, \ldots, a_z\}$ of the names of all the attributes used in tuple objects.

Now let us introduce the auxiliary *identity* relation $=_{id}$ specified on the set **O**.

[2]It is not obvious if Bancilhon and Khoshafian consider set or multiset objects; we assume

DEFINITION 2.1 Any two objects $o_1, o_2 \in \mathbf{O}$ are in the relation $=_{id}$, i.e. $o_1 =_{id} o_2$, if and only if:

- o_1 and o_2 are the same elementary (atomic or special) objects,
- o_1 and o_2 are tuple objects of the forms $[a_{11} : o_{11}; a_{12} : o_{12}; \ldots; a_{1z} : o_{1z}]$ and $[a_{21} : o_{21}; a_{22} : o_{22}; \ldots; a_{2z} : o_{2z}]$, such that:
  - for each pair a_{1i}, a_{2j} of the same attributes from the first and the second tuples respectively the relation $o_{1i} =_{id} o_{2j}$ is true,
- o_1 and o_2 are set objects of the forms $\{o_{11}, o_{12}, \ldots, o_{1m}\}$ and $\{o_{21}, o_{22}, \ldots, o_{2m}\}$, such that element objects from these sets are pair-wise in the relation $=_{id}$.

Let us observe, that the above *identity* relation resembles very much the *equality* relation of Bancilhon and Khoshafian.

DEFINITION 2.2 An *object* in the new calculus for objects is:

- a special object BOTTOM ($\top$-the inconsistent object) or TOP ($\bot$-the undefined object),
- a boolean or else an integer, float or string from a finite subset of integer, float or string values (we call it an atomic object),
- a tuple object of the form: $[a_{i1}{:}o_{i1}; a_{i2}{:}o_{i2}; \ldots; a_{iz}{:}o_{iz}]$, where $a_{i1}, a_{i2}, \ldots, a_{iz}$ are all the elements of the set $\mathbf{A}$ (specified in any order) and $o_{i1}, o_{i2}, \ldots, o_{iz}$ are any objects,
- a set object $\{o_1, o_2, \ldots, o_n\}$, satisfying the following:
  - none element object o_i $(1 \leq i \leq n)$ is of a set object form,
  - none element objects o_i, o_j $(1 \leq i \neq j \leq n)$ are in the identity relation $=_{id}$.

Let $\underline{\mathbf{O}}$ stand for the set of all such objects.

To simplify notation, let us assume that attributes of TOP ($\bot$) value may be deleted from specifications of tuple objects from the set $\underline{\mathbf{O}}$. Under this assumption the following exemplary relations hold: $[a_1 : o_1; a_3 : \bot; a_9 : o_9] =_{id} [a_1 : o_1; a_9 : o_9; a_5 : \bot] =_{id} [a_9 : o_9; a_1 : o_1]$, where $o_1, o_9 \in \underline{\mathbf{O}}$.

EXAMPLE 2.1 Here are some examples of objects in the sense of Definition 2.2:

$\bot$, $\top$, `true`, `false`,
`1, 9, -25,`
`3.5, -81.17, 54.7E-2,`
`'Alice and George', 'oto lancuch znakow',`
$[a_1 : -15;\ a_2 : [a_1 : +9;\ a_3 : \{$`'Cracow','Warsaw'`$\}];\ a6 : 2.5]$,
$\{13, 19, -94, 2\}$.

Considering Definition 2.2, let us pay attention to the restrictions put on the

a finite set of values. This restriction is necessary on account of new definitions of the *equality* and *sub_object* relations. On the other hand, note that any computer architecture makes it possible to implement just finite sets of values. The second restriction concerns the set object form. It obviously results from the semantics imposed on the set objects in the new calculus for objects. In this calculus the former object $\{3, 2.1, \{'\mathtt{ABC}', '\mathtt{DEF}'\}\}$ will be replaced by the object $\{3, 2.1, '\mathtt{ABC}', '\mathtt{DEF}'\}$.

The second difference between the two calculi for objects consists in the way of interpreting the *sub_object* relation $\leq$. The former *sub_object* relation $\leq$ will be replaced by the dual *sub_object* relation $\leqq$ of the following interpretation: $o_1 \leqq o_2$ means that the set of (real) entities represented by o_1 is smaller than the set of (real) entities represented by o_2. Let us observe that this new relation shows correctly the hierarchical nature of the world of real entities.

The two considered changes have subsequent consequences, resulting in completely new definitions of the *equality* and *sub_object* relations, as well as the *union* and *intersection* functions in the new calculus for objects.

In the beginning we define the *equality* relation on the set $\underline{\mathbf{O}}$ of all objects in the sense of Definition 2.2 and the *first_normal* form for objects from this set. Let $\underline{\mathbf{AO}}$ stand for the set $\{ao_1, ao_2, \ldots, ao_t\}$ of all the atom objects. Let $\underline{\mathbf{TO}}$ and $\underline{\mathbf{SO}}$ stand for the sets of all objects of a tuple form and a (new) set form, respectively.

DEFINITION 2.3 The *equality* relation ($\equiv$) specified on the set $\underline{\mathbf{O}}$ of all objects is the least equivalence relation, complying with the following requirements:

- $\forall(o_1, o_2 \in \underline{\mathbf{O}})((o_1 =_{id} o_2) \rightarrow (o_1 \equiv o_2))$,
- $\forall(o \in (\underline{\mathbf{O}} - \underline{\mathbf{SO}}))(o \equiv \{o\})$,
- $\forall(o \in \underline{\mathbf{O}})(((o =_{id} [a_1 : o_1; a_2 : o_2; \ldots; a_z : o_z]) \wedge (\exists(1 \leq i \leq z)(o_i \equiv \top))) \rightarrow (o \equiv \top))$,
- $\{\,\} \equiv \top$,
- $\{ao_1, ao_2, \ldots, ao_t, [\,]\} \equiv \bot$,
- $\forall(1 \leq i \leq z)\forall(n \in N)\forall(o_1, o_2, \ldots, o_{i-1}, o_{i+1}, \ldots, o_z, o_{i1}, o_{i2}, \ldots, o_{in} \in \underline{\mathbf{O}})$
  $([a_1 : o_1; a_2 : o_2; \ldots; a_i : \{o_{i1}, o_{i2}, \ldots, o_{in}\}; \ldots; a_z : o_z] \equiv$
  $\{[a_1 : o_1; a_2 : o_2; \ldots; a_i : o_{ij}; \ldots; a_z : o_z] | 1 \leq j \leq n\})$,
- $\forall(1 \leq i \leq z)\forall(o_1, o_2, \ldots, o_{i-1}, o_i, o_i', o_{i+1}, \ldots, o_z \in \underline{\mathbf{O}})$

  $(((o =_{id} [a_1 : o_1; a_2 : o_2; \ldots; a_i : o_i; \ldots; a_z : o_z])$
  $\wedge(o' =_{id} [a_1 : o_1; a_2 : o_2; \ldots; a_i : o_i'; \ldots; a_z : o_z]) \wedge (o_i \equiv o_i')) \rightarrow (o \equiv o'))$,

- $\forall(o \in \underline{\mathbf{SO}})((o =_{id} \{o_1, o_2, \ldots, o_n\}) \rightarrow$

  $\forall(1 \leq i \leq n)(((o_i \equiv \{o_{i(1)}, o_{i(2)}, \ldots, o_{i(k)}\})$
  $\wedge(\{1 \leq j \leq k | \neg\Sigma(1 \leq m \neq i \leq n)(o_m =_{id} o_{i(j)})\} = \{p1, p2, \ldots, pr\}))$

EXAMPLE 2.2 The following pairs of objects are in the *equality* relation $\equiv$:

$'ADAM' \equiv 'ADAM'$
$5.31 \equiv 5.31$
$-7 \equiv \{-7\}$
$[a_3 : 5; a_1 : \bot] \equiv [a_3 : 5]$
$[a_1 : true; a_3 :' rain'; a6 : \top] \equiv \top$
$[a_7 : \{3, 19, 76\}; a_2 : false] \equiv$
$\{[a_7 : 3; a_2 : false],\ [a_7 : 19; a_2 : false],\ [a_7 : 76; a_2 : false]\}$
$[a_2 : \{'rain'\}; a_3 : true;] \equiv [a_2 :' rain'; a_3 : true;]$
$\{[a_1 : \{1, 2\}; a_2 : true],\ [a_1 : 1; a_2 : true]\} \equiv$
$\{[a_1 : 1; a_2 : true], [a_1 : 2; a_2 : true]\}$

Let $\underline{O1}$ stand for the set of objects being in the so-called *first_normal* form.

DEFINITION 2.4 We say that an object $o \in \underline{O}$ is in the *first_normal* form ($o \in \underline{O1}$) if and only if it is of a set form $\{o_1, o_2, \ldots, o_n\}$ satisfying the following:

- $\neg\exists(1 \leq i \neq j \leq n)(o_i \equiv o_j)$,
- $\neg\exists(1 \leq i \leq n)(o_i \equiv \top)$,
- $\forall(1 \leq i \leq n)((o_i =_{id} [a_1 : o_{i1}; a_2 : o_{i2}; \ldots; a_z : o_{iz}]) \rightarrow (\forall(1 \leq j \leq z)(o_{ij} \in \underline{O1})))$.

Obviously, $\underline{O1} \subset \underline{O}$. To simplify notation of the objects from $\underline{O1}$, assume that the attributes of the $\{\bot\}$ value may be deleted from specifications of their tuple sub-objects. Under this assumption the following exemplary relations hold: $\{[a_1 : o_1; a_3 : \{\bot\}; a_9 : o_9]\} =_{id} \{[a_1 : o_1; a_9 : o_9; a_5 : \{\bot\}]\} =_{id} \{[a_9 : o_9; a_1 : o_1]\}$, where $o_1, o_9 \in \underline{O1}$.

EXAMPLE 2.3 Here are some examples of objects in the *first_normal* form:

$\{\}$
$\{5.31, true,' rain'\}$
$\{[a_7 : \{3, 19, 76\}; a_2 : \{false\}]\}$
$\{5.31, [a_2 : \{'rain'\}; a_3 : \{true\}]\}$

and objects not in the *first normal* form:

$\{\top\}$
$[a_7 : \{3, 19, 76\}; a_2 : \{false\}]$
$\{5.31, [a_2 :' rain'; a_3 : \{true\}]\}$
$\{[a_7 : \{[a_2 : \{3, 19\}; a_5 : \{false\}]\}]\},$
$[a_7 : \{[a_2 : \{3\}; a_5 : \{false\}], [a_2 : \{19\}; a_5 : \{false\}]\}]\}$

Let us now introduce the auxiliary notion of the so-called *object_depth*. We will use it while proving lemmas and theorems formulated later on.

- *object depth* $(\top)$ = *object depth* $(\bot)$ = *object depth* $(\{\})$ = 1,
- if $o \in \underline{\mathbf{AO}}$, then *object depth* $(o) = 1$,
- if $o \in \underline{\mathbf{TO}}$ and $o =_{id} [a_1 : o_1; a_2 : o_2; \ldots; a_z : o_z]$, then
  object depth $(o) = \max\{$*object depth* $(oi) | 1 \leq i \leq z\} + 1$,
- if $o \in \underline{\mathbf{SO}}$ and $o =_{id} \{o_1, o_2, \ldots, o_n\}$, then
  object depth $(o) = \max\{$*object depth* $(o_i) | 1 \leq i \leq n\}$.

EXAMPLE 2.4 These are examples of some objects and their *object depth*:

object depth$(\top) = 1$
object depth$('\texttt{ADAM}') = 1$
object depth$([\mathtt{a}_1 : \mathtt{true}; \mathtt{a}_3 :' \mathtt{rain}'; \mathtt{a}_6 : \top]) = 2$
object depth$([\mathtt{a}_1 : \top; \mathtt{a}_2 :' \mathtt{rain}'; \mathtt{a}_3 : [\mathtt{a}_1 : \mathtt{true}; \mathtt{a}_2 : [\mathtt{a}_5 :' \mathtt{snow}']; \mathtt{a}_3 : \bot]]) = 4$
object depth$(\{'\mathtt{ADAM}', '\mathtt{MACIEJ}'\}) = 1$
object depth$(\{[\mathtt{a}_1 : \mathtt{true}; \mathtt{a}_2 :' \mathtt{snow}'; \mathtt{a}_3 : \bot],$
$[\mathtt{a}_1 : \bot; \mathtt{a}_3 : [\mathtt{a}_1 : \mathtt{true}; \mathtt{a}_2 : \{'\mathtt{rain}'\}]]\}) = 3$
object depth$(\{[\mathtt{a}_7 : \{[\mathtt{a}_2 : \{3, 19\}; \mathtt{a}_5 : \{\mathtt{false}\}]\}]\}) =$
object depth$(\{[\mathtt{a}_7 : \{[\mathtt{a}_2 : \{3\}; \mathtt{a}_5 : \{\mathtt{false}\}], [\mathtt{a}_2 : \{19\}; \mathtt{a}_5 : \{\mathtt{false}\}]\}]\}) = 3$

Let us also observe that there exist such pairs of objects $o_1, o_2 \in \underline{\mathbf{O}}$ for which the formula $(o_1 \equiv o_2) \wedge ($*object depth*$(o_1) \neq$ *object depth*$(o_2))$ is true (the first one and the third object from Example 2.4). However, for any objects $o_1, o_2 \in \underline{\mathbf{O1}}$ we have: $(o_1 \equiv o_2) \rightarrow ($*object depth*$(o_1) =$ *object depth*$(o_2))$ (the last pair of objects from Example 2.4).

LEMMA 2.1 *For each object* $o \in \underline{\mathbf{O}}$ *there exists such an object* $o' \in \underline{\mathbf{O1}}$ *for which the relation* $o \equiv o'$ *is satisfied*[3].

Now, we can define the *sub_object* relation and the *second_normal* form for the objects from the set $\underline{\mathbf{O1}}$.

DEFINITION 2.6 The *sub_object* relation $(\leqq)$ specified on the set $\underline{\mathbf{O1}}$ of all objects being in the *first_normal* form may be recursively defined as follows:

- $\forall(o_1, o_2 \in \underline{\mathbf{O1}})((o_1 \equiv o_2) \rightarrow (o_1 \leqq o_2))$,
- $\forall(o \in \underline{\mathbf{O1}})(\{\} \leqq o \leqq \{\bot\})$,
- $\forall(o_1, o_2 \in \underline{\mathbf{O1}})((o_1 =_{id} \{[a_1 : o_{11}; a_2 : o_{12}; \ldots; a_z : o_1 z]\}$
  $\wedge o_2 =_{id} \{[a_1 : o_{21}; a_2 : o_{22}; \ldots; a_z : o_2 z]\}$
  $\wedge \forall(1 \leq i \leq z)(o_{1i} \leqq o_2 i)) \rightarrow (o_1 \leqq o_2))$,
- $\forall(o_1, o_2 \in \underline{\mathbf{O1}})(((o_1 =_{id} \{o_{11}, o_{12}, \ldots, o_{1m}\}) \wedge (o_2 =_{id} \{o_{21}, o_{22}, \ldots, o_{2n}\})$
  $\wedge(\forall(1 \leq i \leq m)\exists(\{o'_{1i}, o'_{21}, o'_{22}, \ldots, o'_{2q}\} \in \underline{\mathbf{O1}})$
  $((o_2 \equiv \{o'_{1i}, o'_{21}, o'_{22}, \ldots, o'_{2q}\}) \wedge (\{o_{1i}\} \equiv \{o'_{1i}\}))))) \rightarrow (o_1 \leqq o_2))$,
- no other pair of objects $o_1, o_2 \in \underline{\mathbf{O1}}$ can be in the relation $\leqq$.

EXAMPLE 2.5 The following pairs of objects are in the *sub_object* relation ($\leqq$):

$\{3.5\} \leqq \{\perp\}$
$\{\,\} \leqq \{3.5\}$
$\{[a_1 : \{2.5, -5.\}; a_2 : \{\texttt{true}\}]\} \leqq$
$\{[a_1 : \{2.5\}; a_2 : \{\texttt{true}\}], [a_1 : \{-5.\}; a_2 : \{\texttt{true}\}]\}$
$\{[a_1 : \{-5.\}; a_2 : \{\texttt{true}\}; a_7 : [a_1 : \{-2.3, 0.5\}; a_3 : \{'\texttt{BIG}'\}]]\} \leqq$
$\{[a_1 : \{-5., 7.0\}; a_7 : [a_1 : \{0.5, -2.3\}]]\}$
$\{[a_1 : \{3.5, 18.0\}; a_3 : \{'\texttt{mouse}', '\texttt{cat}'\}]\} \leqq$
$\{[a_1 : \{3.5, 7.2, 18.0\}; a_3 : \{'\texttt{mouse}', '\texttt{dog}'\}],$
$[a_1 : \{3.5, 18.0\}; a_3 : \{'\texttt{cat}', '\texttt{dog}'\}]\}$

For the last, most complicated case of the Example 2.5 we have:

$\{[a_1 : \{3.5, 7.2, 18.0\}; a_3 : \{'\texttt{mouse}', '\texttt{dog}'\}],$
$[a_1 : \{3.5, 18.0\}; a_3 : \{'\texttt{cat}', '\texttt{dog}'\}]\} \equiv$
$\{[a_1 : \{3.5, 18.0\}; a_3 : \{'\texttt{mouse}', '\texttt{dog}'\}],$
$[a_1 : \{7.2\}; a_3 : \{'\texttt{mouse}', '\texttt{dog}'\}],$
$[a_1 : \{3.5, 18.0\}; a_3 : \{'\texttt{cat}', '\texttt{dog}'\}]\} \equiv$
$\{[a_1 : \{3.5, 18.0\}; a_3 : \{'\texttt{mouse}', '\texttt{dog}', '\texttt{cat}'\}],$
$[a_1 : \{7.2\}; a_3 : \{'\texttt{mouse}', '\texttt{dog}'\}]\} \equiv$
$\{[a_1 : \{3.5, 18.0\}; a_3 : \{'\texttt{mouse}', '\texttt{cat}'\}],$
$[a_1 : \{3.5, 18.0\}; a_3 : \{'\texttt{dog}'\}],$
$[a_1 : \{7.2\}; a_3 : \{'\texttt{mouse}', '\texttt{dog}'\}]\}.$

The above equalities follow from the sixth specific point of Definition 2.3. Finally, from the fourth specific point of Definition 2.6 we conclude the correctness of the last *sub_object* relation $\leqq$.

Let $\underline{\mathbf{O2}}$ stand for the set of objects being in so-called *second_normal* form.

DEFINITION 2.7 We say that an object $o \in \underline{\mathbf{O1}}$ is in the *second normal* form ($o \in \underline{\mathbf{O2}}$) if and only if it is of a set form $\{o_1, o_2, \ldots, o_n\}$ satisfying the following:

- $\neg\exists(1 \leq i \neq j \leq n)(oi \leqq oj)$,
- $\forall(1 \leq i \leq n)((oi =_{id} [a_1 : o_{i1}; a_2 : o_{i2}; \ldots; a_z : o_{iz}])$
  $\rightarrow \forall(1 \leq j \leq z)(o_{ij} \in \underline{\mathbf{O2}}))$.

Obviously, $\underline{\mathbf{O2}} \subset \underline{\mathbf{O1}}$.

LEMMA 2.2 *For each object $o \in \underline{\mathbf{O1}}$ there exists such an object $o' \in \underline{\mathbf{O2}}$, for which the relation $o \equiv o'$ holds.*

Let us denote the two deterministic algorithms from Lemmas 2.1 and 2.2:

- by *fnf* – the algorithm of obtaining for an object $o \in \underline{\mathbf{O}}$ – an object $o' \in \underline{\mathbf{O1}}$ such that $o \equiv o'$,
- by *snf* – the algorithm of obtaining for an object $o' \in \underline{\mathbf{O1}}$ – an object

Consequently, we will write: $fnf(o) = o'$ and $snf(o') = o''$.

LEMMA 2.3 *For each object* $o \in \underline{\mathbf{O2}}$ *there exists an object* $o' \in \underline{\mathbf{O2}}$ *such that* $o \equiv o'$ *and each set object* o'' *being a sub-component of* o' *consists of one element only, i.e.* $o'' =_{id} \{o_1\}$, *where* o_1 *is a special, atom or tuple object. There exists exactly one such object* o': *we call it an object in the elementary form.*

EXAMPLE 2.6 Let us assume, that:

$\mathtt{o} =_{\mathtt{id}} \{5, 3.5, [\mathtt{a}_1 : \{'\mathtt{or}','\mathtt{and}'\}; \mathtt{a}_3 : \{3.5, 8.0\mathtt{E}-1\}],$
$[\mathtt{a}_1 : \{'\mathtt{or}','\mathtt{and}','\mathtt{xor}'\}; \mathtt{a}_3 : \{3.5\}]\}$

Then, the only object $o' \in \underline{\mathbf{O2}}$ satisfying the constraints of Lemma 2.3 is of the form:

$\mathtt{o}' =_{\mathtt{id}} \{5, 3.5, [\mathtt{a}_1 : \{'\mathtt{or}'\}; \mathtt{a}_3 : \{3.5\}], [\mathtt{a}_1 : \{'\mathtt{and}'\}; \mathtt{a}_3 : \{3.5\}],$
$[\mathtt{a}_1 : \{'\mathtt{or}'\}; \mathtt{a}_3 : \{8.0\mathtt{E}-1\}], [\mathtt{a}_1 : \{'\mathtt{and}'\}; \mathtt{a}_3 : \{8.0\mathtt{E}-1\}],$
$[\mathtt{a}_1 : \{'\mathtt{xor}'\}; \mathtt{a}_3 : \{3.5\}]\}$

LEMMA 2.4 *The sub_object relation* $\leqq$ *defined on the set* $\underline{\mathbf{O2}}$ *of objects being in the second_normal form is a partial order relation.*

From now on we will consider objects in the *second_normal* form only: thus, "object o" will mean an object $o \in \underline{\mathbf{O2}}$.

Successively, in the new calculus for objects the *union* and the *intersection* functions are defined. Although both these functions have their prototypes in the calculus for complex objects, let us observe a quite new, compact definition of the *union* function. It is a consequence of the new object interpretation: each object is now considered as a set object. In particular, an atom, special or tuple object is considered an empty or single-element set object.

DEFINITION 2.8 For any two objects $o_1, o_2 \in \underline{\mathbf{O2}}$, the *union* function $\cup$ is defined in the following way:

$- \forall(o_1, o_2 \in \underline{\mathbf{O2}})((o_1 =_{id} \{o_{11}, o_{12}, \ldots, o_{1m}\} \wedge o_2 =_{id} \{o_{21}, o_{22}, \ldots, o_{2n}\}) \rightarrow$
$(o_1 \cup o_2 = snf(fnf(\{o_{11}, o_{12}, \ldots, o_{1m}, o_{21}, o_{22}, \ldots, o_{2n}\}))))$.

EXAMPLE 2.7 From Definition 2.8 we obtain:

$\{3.5, 18, \mathtt{true}, [\mathtt{a}_1 : \{'\mathtt{big}','\mathtt{small}'\}; \mathtt{a}_3 : \{\mathtt{false}\}],$
$[\mathtt{a}_5 : \{'\mathtt{rain}','\mathtt{snow}'\}],'\mathtt{ADAM}','\mathtt{Maciej}',$
$[\mathtt{a}_2 : \{[\mathtt{a}_1 : \{'\mathtt{A}','\mathtt{B}'\}]\}]\} \cup$
$\{-26, 18, \mathtt{false}, [\mathtt{a}_5 : \{'\mathtt{rain}','\mathtt{snow}'\}],'\mathtt{Adam}','\mathtt{Maciej}',$
$[\mathtt{a}_1 : \{'\mathtt{big}','\mathtt{small}','\mathtt{other}'\}], 3.5,$
$[\mathtt{a}_2 : \{[\mathtt{a}_1 : \{'\mathtt{A}'\}], [\mathtt{a}_1 : \{'\mathtt{B}'\}]\}]\} =$
$snf(fnf(\{3.5, 18, \mathtt{true}, [\mathtt{a}_1 : \{'\mathtt{big}','\mathtt{small}'\}; \mathtt{a}_3 : \{\mathtt{false}\}],$
$[\mathtt{a}_5 : \{'\mathtt{rain}','\mathtt{snow}'\}],'\mathtt{ADAM}','\mathtt{Maciej}',$
$[\mathtt{a}_2 : \{[\mathtt{a}_1 : \{'\mathtt{A}','\mathtt{B}'\}]\}], -26, \mathtt{false},$
$'\mathtt{Adam}', [\mathtt{a}_1 : \{'\mathtt{big}','\mathtt{small}','\mathtt{other}'\}],$

$snf(\{3.5, 18, \mathtt{true}, [\mathtt{a}_1 : \{\mathtt{'big'}, \mathtt{'small'}\}; \mathtt{a}_3 : \{\mathtt{false}\}],$
$[\mathtt{a}_5 : \{\mathtt{'rain'}, \mathtt{'snow'}\}], \mathtt{'ADAM'}, \mathtt{'Maciej'},$
$[\mathtt{a}_2 : \{[\mathtt{a}_1 : \{\mathtt{'A'}, \mathtt{'B'}\}]\}], -26, \mathtt{false},$
$\mathtt{'Adam'}, [\mathtt{a}_1 : \{\mathtt{'big'}, \mathtt{'small'}, \mathtt{'other'}\}]\}) =$

$\{3.5, 18, \mathtt{true}, [\mathtt{a}_5 : \{\mathtt{'rain'}, \mathtt{'snow'}\}], \mathtt{'ADAM'}, \mathtt{'Maciej'},$
$[\mathtt{a}_2 : \{[\mathtt{a}_1 : \{\mathtt{'A'}, \mathtt{'B'}\}]\}], -26, \mathtt{false}, \mathtt{'Adam'},$
$[\mathtt{a}_1 : \{\mathtt{'big'}, \mathtt{'small'}, \mathtt{'other'}\}]\}$

DEFINITION 2.9 For any two objects from the set $\underline{\mathbf{O2}}$, the intersection function $\sqcap$ is defined recursively as follows:
- $\forall(o \in \underline{\mathbf{O2}})(\{\,\} \sqcap o = o \sqcap \{\,\} = \{\,\})$,
- $\forall(o \in \underline{\mathbf{O2}})(\{\bot\} \sqcap o = o \sqcap \{\bot\} = o)$,
- $\forall(ao_i, ao_j \in \underline{\mathbf{AO}})((ao_i =_{id} ao_j) \rightarrow (\{ao_i\} \sqcap \{ao_j\} = \{ao_i\} = \{ao_j\}))$,
- $\forall(o_1, o_2 \in \underline{\mathbf{O2}})((o_1 =_{id} \{[a_1 : o_{11}; a_2 : o_{12}; \ldots; a_z : o_{1z}]\}$
$\wedge\, o_2 =_{id} \{[a_1 : o_{21}; a_2 : o_{22}; \ldots; a_z : o_{2z}]\}) \rightarrow$
$(o_1 \sqcap o_2 = \{[a_1 : o_{11} \sqcap o_{21}; a_2 : o_{12} \sqcap o_{22}; \ldots; a_z : o_{1z} \sqcap o_{2z}]\}))$,
- $\forall(o_1, o_2 \in \underline{\mathbf{O2}})((o_1 =_{id} \{o_{11}, o_{12}, \ldots, o_{1m}\} \wedge o_2 =_{id} \{o_{21}, o_{22}, \ldots, o_{2n}\}) \rightarrow$
$(o_1 \sqcap o_2 = (\{o_{11}\} \sqcap \{o_{21}\}) \sqcup (\{o_{11}\} \sqcap \{o_{22}\}) \sqcup \ldots \sqcup (\{o_{11}\} \sqcap \{o_{2n}\})$
$\sqcup (\{o_{12}\} \sqcap \{o_{21}\}) \sqcup (\{o_{12}\} \sqcap \{o_{22}\}) \sqcup \ldots \sqcup (\{o_{12}\} \sqcap \{o_{2n}\})$
$\sqcup \ldots$
$\sqcup (\{o_{1m}\} \sqcap \{o_{21}\}) \sqcup (\{o_{1m}\} \sqcap \{o_{22}\}) \sqcup \ldots \sqcup (\{o_{1m}\} \sqcap \{o_{2n}\}))))$,
- for any other pair of objects $o_1, o_2 \in \underline{\mathbf{O2}}$ we have: $o_1 \sqcap o_2 = \{\,\}$.

EXAMPLE 2.8 From Definition 2.9 we obtain:
$\{\,\} \sqcap \{3.5, 18\} = \{3.5, 18\} \sqcap \{\,\} = \{\,\}$
$\{\bot\} \sqcap \{3.5, 18\} = \{3.5, 18\} \sqcap \{\bot\} = \{3.5, 18\}$
$\{\mathtt{'ADAM'}\} \sqcap \{\mathtt{'ADAM'}\} = \{\mathtt{'ADAM'}\}$

$\{[\mathtt{a}_1 : \{\mathtt{'big'}, \mathtt{'small'}\}; \mathtt{a}_3 : \{\mathtt{true}, \mathtt{false}\}; \mathtt{a}_5 : \{3.5, 18, -45\}]\} \sqcap$
$\{[\mathtt{a}_1 : \{\mathtt{'big'}, \mathtt{'small'}, \mathtt{'other'}\}; \mathtt{a}_3 : \{\mathtt{true}\}]\} =$
$\{[\mathtt{a}_1 : \{\mathtt{'big'}, \mathtt{'small'}\}; \mathtt{a}_3 : \{\mathtt{true}\}; \mathtt{a}_5 : \{3.5, 18, -45\}]\}$

$\{3.5, 18, \mathtt{true}, [\mathtt{a}_1 : \{\mathtt{'big'}, \mathtt{'small'}\}; \mathtt{a}_3 : \{\mathtt{false}\}],$
$[\mathtt{a}_5 : \{\mathtt{'rain'}, \mathtt{'snow'}\}], \mathtt{'ADAM'}, \mathtt{'Maciej'},$
$[\mathtt{a}_2 : \{[\mathtt{a}_1 : \{\mathtt{'A'}, \mathtt{'B'}\}]\}]\} \sqcap$
$\{-26, 18, \mathtt{false}, [\mathtt{a}_5 : \{\mathtt{'rain'}, \mathtt{'snow'}\}], \mathtt{'Adam'}, \mathtt{'Maciej'},$
$[\mathtt{a}_1 : \{\mathtt{'big'}, \mathtt{'small'}, \mathtt{'other'}\}], 3.5,$
$[\mathtt{a}_2 : \{[\mathtt{a}_1 : \{\mathtt{'A'}\}], [\mathtt{a}_1 : \{\mathtt{'B'}\}]\}]\} = \ldots =$
$\{3.5\} \sqcup \{18\} \sqcup$
$\{[\mathtt{a}_1 : \{\mathtt{'big'}, \mathtt{'small'}\}; \mathtt{a}_3 : \{\mathtt{false}\}, \mathtt{a}_5 : \{\mathtt{'rain'}, \mathtt{'snow'}\}]\} \sqcup$
$\{[\mathtt{a}_1 : \{\mathtt{'big'}, \mathtt{'small'}\}; \mathtt{a}_3 : \{\mathtt{false}\}]\} \sqcup$
$\{[\mathtt{a}_1 : \{\mathtt{'big'}, \mathtt{'small'}\}; \mathtt{a}_2 : \{[\mathtt{a}_1 : \{\mathtt{'A'}\}], [\mathtt{a}_1 : \{\mathtt{'B'}\}]\}; \mathtt{a}_3 : \{\mathtt{false}\}]\} \sqcup$
$\{[\mathtt{a}_5 : \{\mathtt{'rain'}, \mathtt{'snow'}\}]\} \sqcup$
$\{[\mathtt{a}_1 : \{\mathtt{'big'}, \mathtt{'small'}, \mathtt{'other'}\}; \mathtt{a}_5 : \{\mathtt{'rain'}, \mathtt{'snow'}\}]\} \sqcup$

$\{'\texttt{Maciej}'\} \underline{\cup} \{[a_2 : \{[a_1 : \{'\texttt{A}','\texttt{B}'\}]\}; a_5 : \{'\texttt{rain}','\texttt{snow}'\}]\} \underline{\cup}$
$\{[a_1 : \{'\texttt{big}','\texttt{small}','\texttt{other}'\}; a_2 : \{[a_1 : \{'\texttt{A}','\texttt{B}'\}]\}]\} \underline{\cup}$
$\{[a_2 : \{[a_1 : \{'\texttt{A}'\}], [a_1 : \{'\texttt{B}'\}]\}]\} =$
$\{3.5\} \underline{\cup} \{18\} \underline{\cup} \{[a_1 : \{'\texttt{big}','\texttt{small}'\}; a_3 : \{\texttt{false}\}]\} \underline{\cup}$
$\{[a_5 : \{'\texttt{rain}','\texttt{snow}'\}]\} \underline{\cup} \{'\texttt{Maciej}'\} \underline{\cup}$
$\{[a_2 : \{[a_1 : \{'\texttt{A}'\}], [a_1 : \{'\texttt{B}'\}]\}]\}$

The last specific point of the Example is a comprehensive one. It is worth analysing.

LEMMA 2.5 *Under the sub object relation* $\leqq$ *it can be proved for any two objects* $o_1, o_2 \in \underline{\mathbf{O2}}$ *that:*
- $o_1 \underline{\cup} o_2 \equiv \sup\{o_1, o_2\}$,
- $o_1 \underline{\cap} o_2 \equiv \inf\{o_1, o_2\}$.

The following theorem is an immediate corollary of Lemma 2.5.

THEOREM 2.1 *The algebra* $A = (\underline{\mathbf{O2}}, \underline{\cup}, \underline{\cap})$ *is a lattice.*

Now, in the new calculus for objects the *complement* function is being defined. This function has no equivalent in the calculus by Bancilhon and Khoshafian.

DEFINITION 2.10 For any object from the set $\underline{\mathbf{O2}}$, the *complement* function $(\underline{\neg})$ is defined recursively as follows:
- $\underline{\neg}(\{\}) = \{\bot\}$,
- $\underline{\neg}(\{\bot\}) = \{\}$,
- $\forall(ao_i \in \underline{\mathbf{AO}})(\underline{\neg}\{ao_i\} = \{ao_1, ao_2, \ldots, ao_{i-1}, ao_{i+1}, \ldots, ao_t, []\})$,
- $\forall(o \in \underline{\mathbf{O2}})((o =_{id} \{[a_1 : o_1; a_2 : o_2; \ldots; a_z : o_z]\}) \rightarrow$

$$(\underline{\neg}(o) = \underline{\cup} (\{[ai : (\underline{\neg} o_i)]\} | 1 \leq i \leq z) \underline{\cup} \underline{\mathbf{AO}}),$$

- $\forall(o \in \underline{\mathbf{O2}})((o =_{id} \{o_1, o_2, \ldots, o_n\}) \rightarrow$

$$(\underline{\neg}(o) = ((\underline{\neg}\{o_1\}) \underline{\cap} (\underline{\neg}\{o_2\}) \underline{\cap} \ldots \underline{\cap} (\underline{\neg}\{o_n\}))))).$$

EXAMPLE 2.9 For the purpose of explaining the semantics of the *complement* function $\underline{\neg}$ let us assume that:

$$\underline{\mathbf{AO}} = \{1, 2, 3, 1.0, '\texttt{ADAM}', '\texttt{Maciej}', '\texttt{snow}', '\texttt{rain}', \texttt{true}, \texttt{false}\}$$

In such case we obtain:
$\neg(\{'\texttt{ADAM}'\}) =$

$$\begin{aligned}
&\sqsupset(\{[a_1 : \{'ADAM','Maciej'\}; a_2 : \{[a_1 : \{true\}; a_3 : \{1, 1.0\}]\}]\}) = \\
&\{[a_1 : \{1, 2, 3, 1.0,' snow',' rain', true, false, []\}]\} \sqcup \\
&\{[a_2 : \sqsupset\{[a_1 : \{true\}; a_3 : \{1, 1.0\}]\}]\} \sqcup \\
&\{1, 2, 3, 1.0,' ADAM',' Maciej',' snow',' rain', true, false\} = \\
&\{[a_1 : \{1, 2, 3, 1.0,' snow',' rain', true, false, []\}]\} \sqcup \\
&\{[a_2 : (\{[a_1 : \{1, 2, 3, 1.0,' ADAM',' Maciej',' snow',' rain', false, []\}]\} \sqcup \\
&\qquad \{[a_3 : \{2, 3,' ADAM',' Maciej',' snow',' rain', true, false, []\}]\})]\} \sqcup \\
&\{1, 2, 3, 1.0,' ADAM',' Maciej',' snow',' rain', true, false\} = \\
&\{[a_1 : \{1, 2, 3, 1.0,' snow',' rain', true, false, []\}]\} \sqcup \\
&\{[a_2 : \{[a_1 : \{1, 2, 3, 1.0,' ADAM',' Maciej',' snow',' rain', false, []\}], \\
&\qquad [a_3 : \{2, 3,' ADAM',' Maciej',' snow',' rain', true, false, []\}]\}]\} \sqcup \\
&\{1, 2, 3, 1.0,' ADAM',' Maciej',' snow',' rain', true, false\} = \\
&\{[a_1 : \{1, 2, 3, 1.0,' snow',' rain', true, false, []\}], \\
&\ [a_2 : \{[a_1 : \{1, 2, 3, 1.0,' ADAM',' Maciej',' snow',' rain', false, []\}], \\
&\qquad [a_3 : \{2, 3,' ADAM',' Maciej',' snow',' rain', true, false, []\}]\}], \\
&\ 1, 2, 3, 1.0,' ADAM',' Maciej',' snow',' rain', true, false\}
\end{aligned}$$

LEMMA 2.6 *For any three objects* $o_1, o_2, o_3 \in \underline{\mathbf{O2}}$, *the following equalities are satisfied:*

- $o_1 \sqcap (o_2 \sqcup o_3) \equiv (o_1 \sqcap o_2) \sqcup (o_1 \sqcap o_3)$,
- $o_1 \sqcup (o_2 \sqcap o_3) \equiv (o_1 \sqcup o_2) \sqcap (o_1 \sqcup o_3)$,
- $(o_1 \sqcap (\sqsupset o_1)) \sqcup o_2 \equiv o_2$
- $(o_1 \sqcup (\sqsupset o_1)) \sqcap o_2 \equiv o_2$.

As a direct consequence of Theorem 2.1 and Lemma 2.6 we obtain Theorem 2.2.

THEOREM 2.2 *The algebra* $B = (\underline{\mathbf{O}}, \sqcup, \sqcap, \sqsupset)$ *is a Boolean algebra.*

Now let us redefine the syntax of *object_formula* and *object_rule* notions from the calculus for complex objects. We must also redefine their semantics expressed by means of the *interpretation, application* and *closure* functions.

DEFINITION 2.11 An *object_formula* is defined as an object in the *second_normal* form, with a number (perhaps zero) of individual variables $(A_1, A_2, \ldots)$ being used instead of the component or sub-component objects in it.

DEFINITION 2.12 An *extended_object_formula* is defined as an *object_formula* optionally enriched by a formula of FOL. The *extended_object_formula* has a form: $f/p/$ or $f//$, where f is an object formula and p is a formula of FOL. These and only these individual variables may be used in the formula p which occur in the object formula f.

The most important assumption is that all individual variables occurring in the formula of FOL represent atom objects. These individual variables must be valuated in the set $\underline{\mathbf{AO}}$ of all atom objects. The formula of FOL constrains the set of their possible valuations.

The set of all *extended_object_formulas* will be denoted as F

EXAMPLE 2.10 Here are some examples of *extended object formulae*:
$\mathtt{A_1}//$,
$\{\mathtt{A_1}\}//$,
$\{35.6\}//$,
$\{[\mathtt{a_1} : \{'\mathtt{string}'\}; \mathtt{a_3} : \mathtt{A_1}]\}//$,
$\{[\mathtt{a_1} : \{'\mathtt{string}'\}; \mathtt{a_3} : \{\mathtt{A_1}\}; \mathtt{a_6} : \{\mathtt{A_2}\}]\}/ \leq (\mathtt{A_1}, \mathtt{A_2})/$,
$\{35.6, [\mathtt{a_2} : \{\mathtt{A_1}\}; \mathtt{a_3} : \{'\mathtt{string}'\}; \mathtt{a_5} : \{2, 6, \mathtt{A_1}, \mathtt{A_2}\}]\}/ < (\mathtt{A_1}, \mathtt{A_2}) \wedge > (\mathtt{A_1}, 6)/$.

DEFINITION 2.13 An *extended_object_rule* is defined as an ordered pair $\langle f_1/p_1/, f_2/p_2/\rangle$ of extended_object_formulae $f_1/p_1/$ and $f_2/p_2/$ such that:

- the set $V1$ of variables from f_1 is a subset of the set $V2$ of variables from f_2,
- if $V2 = \{A_1, A_2, \ldots, A_n\}$, then the sentence: $\forall(A_1, A_2, \ldots, A_n)(p_2 \rightarrow p_1)$ must be a tautology.

The set of all *extended_object_rules* will be denoted $\underline{\mathrm{RU}}$.

DEFINITION 2.14 Let $f/p/$ be an *extended_object_formula* with variables $\{A_1, A_2, \ldots, A_n\}$. Let I be an interpretation of predicate constants from the FOL formula p. A well-formed substitution for $f/p/$ under I is defined as $\sigma =_{id} \{o_1/A_1, o_2/A_2, \ldots, o_n/A_n\}$, with $o_i \in \underline{\mathbf{O}} (1 \leq i \leq n)$ fulfilling the conditions:

- for any A_i $(1 \leq i \leq n)$ from the FOL formula p there must be: $o_i \in \underline{\mathbf{AO}}$,
- the FOL formula p is satisfied for the substitution σ under the interpretation I,
- the result $o =_{id} \sigma(f)$ of substitution σ on f is an object such that $o \in \underline{\mathbf{O}}$.

We call the object o an instantiation of the *extended_object_formula* $f/p/$.

Finally, in the new calculus for objects the *interpretation*, *application*, and *closure* functions are defined. They have their prototypes in Bancilhon, Khoshafian (1989).

DEFINITION 2.15 Let $f/p/$ be an *extended_object_formula*. Let I be an interpretation of predicate constants from p. Let o be an object in the *second_normal* form. The *interpretation* of $f/p/$ with respect to o under I is a class $C1 = f/p/(o, I)$ of all objects o' such that:

$o' =\underline{\cup} \{\sigma(f) | \sigma$ is a well-formed substitution for $f/p/$ under I such that $snf(fnf(\sigma(f))) \leqq o\}$.

From commutability of the *union* function $\underline{\cup}$ we conclude that all objects

DEFINITION 2.16 Let $r =_{id} \langle f_1/p_1/, f_2/p_2/\rangle$ be an *extended_object_rule*. Let I be an interpretation of predicate constants from p_1 and p_2. Let o be an object in the *second_normal* form. The *application* of r on o under I is a class $C2 = r(o, I)$ of all objects o' such that:

$$o' = \underline{\cup} \{\sigma(f_1) | \sigma \text{ is a well-formed substitution for } f_1/p_1/ \text{ and } f_2/p_2/ \text{ under } I \text{ such that } snf(fnf(\sigma(f_2))) \leqq o\}.$$

DEFINITION 2.17 Let sr be a set of *extended_object_rules*. Let I be an interpretation of predicate constants from the FOL formulas occurring in the rules of sr. Let o be an object in the *second_normal* form. The *application* of sr on o under I is a class $C3 = sr(o, I)$ of all objects o' such that:

$$o' = \underline{\cup} \{o'' \in r(o, I) | r \text{ in } sr\}.$$

DEFINITION 2.18 Let o be an object in the *second_normal* form. Let $r =_{id} \langle f_1/p_1/, f_2/p_2/\rangle$ be an *extended_object_rule* and sr – a set of *extended object rules*. Let I be an interpretation of predicate constants from p_1 and p_2. Object o is closed with respect to r under I if the relation $r(o, I) \leqq o$ is true. Object o is closed with respect to the set of rules sr under I if it is closed with respect to every rule in sr under I.

DEFINITION 2.19 Let o be an object in the *second_normal* form. Let sr be a set of *extended_object_rules*. The *closure* c (o, sr, I) of o with respect to sr under I is the unique minimal object closed with respect to sr under I, if it exists.

The following theorem is a consequence of Theorem 2.1, monotonicity of the *application* (it can be proved directly from Definitions 2.16 and 2.17) and result of Tarski.

THEOREM 2.3 *For any set of extended object rules* $sr \subseteq \underline{\mathbf{RU}}$, *any interpretation* I *of predicate constants from the FOL formulas occurring in the rules of* sr *and any object* $o \in \underline{\mathbf{O2}}$, *if the closure* c (o, sr, I) *exists, it is the limit of the following sequence:*

$$o_1 = o,$$
$$o_2 = sr(o_1, I),$$
$$o_3 = sr(o_2, I),$$
$$\ldots,$$
$$o_n = sr(o_{n-1}, I).$$

Proof. Can be formulated in exactly the same way, as the proof of Theorem 4.1

It was already proved (Theorem 2.2) that our object-oriented data model is not only a lattice (as it prototype was) but also a Boolean algebra. We will show that the proposed extended form of object formulae additionally increases its expressive power. An adequate example will be presented later.

3. A query language to communicate with a database

On the basis of the new calculus for objects an interesting query language can be created. This language is object-oriented and possesses a clear, hierarchical structure. It may be used to communicate with some databases, for instance databases with complex nested values or NF^2 (non-first-normal-form) databases, Abiteboul, Hull, Vianu (1995).

The kernel layer (0) of language operations consists of the following elementary operations: *union, intersection, complement, application* and *closure.* In the first layer (1) of language operations there are the simplest compound operations – namely those which can be defined by means of operations from the layer (0). In turn, the second layer (2) contains such compound operations, which can be defined by means of operations coming from the layers (0) and (1), and so on.

Here are the examples of a few compound operations.

If we introduce the *difference* function $(\underline{-})$ of the expected semantics: $\forall(o_1, o_2 \in \underline{\mathbf{O2}})\ (o_1 \underline{-} o_2 = o_1 \sqcap (\neg o_2))$, we will put it into layer (1). In turn, the *symmetric_difference* function $(\underline{\div})$ of the semantics: $\forall(o_1, o_2 \in \underline{\mathbf{O2}})(o_1 \underline{\div} o_2 = (o_1 \underline{-} o_2) \sqcup (o_2 \underline{-} o_1))$ will be put into the layer (2).

By means of the new language all but some special SQL operations (Date, Darwen, 1994) can be defined. By the special operations we mean aggregate functions, grouping and ordering, which do not have their equivalents in the relational calculus.

EXAMPLE 3.1 Here are the examples of some SQL queries and their translations into the new language:

SELECT $a_{(i1)}, a_{(i2)}, \ldots, a_{(ir)}$ FROM R_i,

where

$\{a_{(i1)}, a_{(i2)}, \ldots, a_{(ir)}\}$ is a subset of A,

R_i is a relation stored in some SQL database,

can be expressed by means of the following function call:

$project1(\{a_{(i1)}, a_{(i2)}, \ldots, a_{(ir)}\}, o_i)$ of the semantics:

$\{\{[a_{(i1)} : A_{(i1)}; \ldots; a_{(ir)} : A_{(ir)}; \#a_{(k1)} : A_{(k1)}; \ldots; \#a_{(ks)} : A_{(ks)}]\}//,$

$\{[a_{(i1)} : A_{(i1)}; \ldots; a_{(ir)} : A_{(ir)}; a_{(k1)} : A_{(k1)}; \ldots; a_{(ks)} : A_{(ks)};$

$a_{(m1)} : A_{(m1)}; \ldots; a_{(mt)} : A_{(mt)}]\}//\rangle(o_i, I),$

where

the *project1* comes from the layer (1) of language operations,

$o_i \in \underline{\mathbf{O2}}$ is the set of tuple objects forming the relation R_i,

at – a function assigning for an SQL relation the set of all its attributes,
$\{a_{(k1)}, a_{(k2)}, \ldots, a_{(ks)}\} = pk(R_i) - \{a_{(i1)}, a_{(i2)}, \ldots, a_{(ir)}\}$,
pk – a function assigning for an SQL relation the set of all attributes from its primary key,
$\#a_i$ is a hidden attribute, i.e. an attribute not shown in a tuple object representation,
I is a default (empty) interpretation;

SELECT $a_{(i1)}, a_{(i2)}, \ldots, a_{(ir)}, a_{(j1)}, a_{(j2)}, \ldots, a_{(js)}$ FROM R_i, R_j,
where
$\{a_{(i1)}, a_{(i2)}, \ldots, a_{(ir)}, a_{(j1)}, a_{(j2)}, \ldots, a_{(js)}\}$ is a subset of A,
R_i, R_j are such relations from SQL database that:

$$((at(R_i) \cap at(Rj) = \emptyset) \wedge (\{a_{(i1)}, a_{(i2)}, \ldots, a_{(ir)}\} \subseteq at(R_i))$$
$$\wedge(\{a_{(j1)}, a_{(j2)}, \ldots, a_{(js)}\} \subseteq at(Rj))),$$

can be expressed by means of the following function call:
$project2(\{a_{(i1)}, a_{(i2)}, \ldots, a_{(ir)}\}, \{a_{(j1)}, a_{(j2)}, \ldots, a_{(js)}\}, o_i, o_j)$, of the semantics:

$$project1(\{a_{(i1)}, a_{(i2)}, \ldots, a_{(ir)}\}, o_i) \sqcap project1(\{a_{(j1)}, a_{(j2)}, \ldots, a_{(js)}\}, o_j),$$

where
the *project2* comes from the layer (2) of language operations,
$o_i, o_j \in \underline{\mathbf{O2}}$ are the sets of all tuple objects forming the relations R_i and R_j respectively;

SELECT $*$ FROM R_i WHERE f,
where
$*$ stands for the list of names of all attributes from the set A,
R_i is a relation stored in some SQL database,
f is a classical formula, in which attributes (from the set A) act as individual variables and atom objects (from the set $\underline{\mathbf{AO}}$) act as individual constants,
I is presumed to be an interpretation of predicate constants from f,
can be expressed by means of the following function call:

$selection(\{a_{(i1)}, a_{(i2)}, \ldots, a_{(iw)}\}, f, o_i)$, of the semantics:

$$\langle\{[a_{(i1)} : A_{(i1)}; \ldots; a_{(iw)} : A_{(iw)}]\}//,$$
$$\{[a_{(i1)} : A_{(i1)}; \ldots; a_{(iw)} : A_{(iw)}]\}/f/\rangle(o_i, I),$$

where
the *selection* comes from the layer (1) of language operations,
$o_i \in \underline{\mathbf{O2}}$ is the set of tuple objects forming the relation R_i,
$at(R_i) = \{a_{(i1)}, \ldots, a_{(iw)}\}$.

Finally, one more example illustrating expressive power of the *closure* function from the kernel layer of operations. The problem and the scheme of its solution

EXAMPLE 3.2 Suppose that F is a "family" relation stored in some SQL database. Let us assume that $pk(F) = \{\text{name}, \text{y\_birth}, \text{children}\}$. Let us find the set of all descendants of Abraham, who were born before 1750. Such set can be obtained from the values of "descoA" attributes of tuples belonging to the following object o':

$$
\begin{aligned}
o' =_{id}\ & c(o, \{\langle\{[\texttt{descoA} : \{\texttt{Abraham}\}]\}//, \{\bot\}//\rangle, \\
& \langle\{[\texttt{descoA} : \{A_1\}]\}//, \\
& \{[\texttt{family}:\{[\texttt{name}:\{A_2\};\texttt{children}:\{[\texttt{name}:\{A_1\};\texttt{y\_birth}:\{A3\}]\}]\}, \\
& \texttt{descoA} : \{A_2\}]\}/\langle(A3, 1750)/\rangle\}, I),
\end{aligned}
$$

where

$o \in \underline{O2}$ is the set of all tuple objects forming the relation F,
I is presumed to be an usual arithmetic interpretation of the *less_than* predicate $<$.

Let us observe that the problem has no solution in the calculus for complex objects. It is due to the presence of the constraint "who were born before 1750". The *application* and *closure* functions have more expressive power than their prototypes *application* and *closure* in Bancilhon, Khoshafian (1989).

4. Concluding remarks

We presented a new object-oriented data model and a query language to communicate with databases of a certain kind. The model originates from Bancilhon, Khoshafian (1989). It differs from the prototype mainly in the interpretation of set objects. It was proved (Theorem 2.2), that this model is not only a lattice, but also a Boolean algebra.

On the basis of this new object-oriented data model we defined a query language. Its most important feature is a hierarchical structure: it is built of separate layers of increasingly complex operations. The kernel layer (0) consists of elementary operations, taken directly from the calculus proposed. A layer (n) (for each $n \geq 1$) consists of such operations which can be defined by means of operations from the layers (0), (1), ..., (n − 1).

At present the language is being implemented. The kernel layer of operations was implemented in Prolog. The other layers can be implemented by means of DCG (extending Prolog syntax) or by means of YACC generator.

From the issues that remain open the following are the most important:

- how to extend the new query language to a comprehensive database language, having statements for data definition, query and update?
- is it possible to extend the new calculus for objects in such a way that the result could become a background for a query language which would serve to communicate with an object-oriented database? especially – an active

References

ABITEBOUL S., HULL R. and VIANU V. (1995) *Foundations of Databases.* Addison-Wesley, Reading, Mass.

AIT-KACI, H. (1993) An Introduction to LIFE - Programming with Logic, Inheritance, Functions, and Equations. *Proceedings of the 10th International Logic Programming Symposium*, 1–17, Vancouver, BC, Canada, October.

BANCILHON F. and KHOSHAFIAN S. (1989) A Calculus for Complex Objects. *Journal of Computer and System Sciences* **38**, 326–340.

CARPENTER B. (1992) The Logic of Typed Feature Structures. *Cambridge Tracts in Theoretical Computer Science* **32**, Cambridge University Press.

DATE C.J. and DARWEN H. (1994) *A Guide to the SQL Standard.* Third Edition, Addison-Wesley, Reading, Mass.

LEUNG T., MITCHELL G., SUBRAMANIAN B., VANCE B., VANDENBERG S. and ZDONIK S. (1993) The AQUA Data Model and Algebra. *Proc. 4th Intl. Workshop on Database Programming Languages*, NYC, August-September.

MINSKY M. (1974) A Framework for Representing Knowledge. MIT-AI Laboratory Memo 306, June.

NILSSON J.F. (1993) A Concept Object Algebra $CA^{+6[:]}$, in: Kangassalo H., Jaakkola H., Hori K., Kitahashi T., (eds.) *Information Modelling and Knowledge Bases IV: Concepts, Methods and Systems*, 42–55, IOS, Amsterdam.

ODMG Team (1997) *The Object Database Standard ODMG, Release 2.0.* R.G.G. Cattel (ed.), Morgan Kaufmann.

Appendix

Proof of Lemma 2.1. Let us prove the lemma by induction on *object depth* of o. In the case of *object depth*$(o) = 1$, o must be:

- the special object $\top$, or
- the special object $\bot$, or
- an atom object ao_i, or
- a set object $\{ao_{i1}, ao_{i2}, \ldots, ao_{in}\}$ or $\{ao_{i1}, ao_{i2}, \ldots, ao_{in}, \bot\}$, consisting of any number (perhaps zero) of atom objects and, optionally, the special object $\bot$, or
- a set object $\{\top, ao_{i1}, ao_{i2}, \ldots, ao_{in}\}$ or $\{\top, ao_{i1}, ao_{i2}, \ldots, ao_{in}, \bot\}$, consisting of the special object $\top$ and any number (perhaps zero) of atom objects and, optionally, the special object $\bot$.

Then the required object o' must have the form, respectively:

- $\{\}$,
- $\{\bot\}$,
- $\{ao_i\}$,
- [illegible]

– $\{ao_{i1}, ao_{i2}, \ldots, ao_{in}\}$ or $\{ao_{i1}, ao_{i2}, \ldots, ao_{in}, \perp\}$.

It follows directly from Definition 2.3, that in all the above cases the relation $o \equiv o'$ is true.

Let us assume that the lemma holds for each object $o_1 \in \underline{\mathbf{O}}$ such that *object_depth*$(o_1) \leq n-1$ (with $n > 1$). We will prove, that it also holds for each object $o_2 \in \underline{\mathbf{O}}$ fulfilling the condition: *object_depth* $(o_2) = n$. Let us observe that only o_2 of a tuple form $[a_1 : o_{21};\ a_2 : o_{22};\ \ldots;\ a_z : o_{2z}]$ or of a set form $\{o_{21}, o_{22}, \ldots, o_{2n}\}$ may fulfil this condition. Let us consider the first of the two cases. In this case the required object o'_2 may take the following form: $\{[a_1 : o'_{21}; a_2 : o'_{22}; \ldots; a_z : o'_{2z}]\}$, where o'_{2i} $(1 \leq i \leq z)$ is such an object in the *first_normal* form, that the equality $o_{2i} \equiv o'_{2i}$ holds. The existence of objects o'_{2i} results from Definition 2.5 (*object_depth*$(o_{2i}) \leq n-1$) and from the induction hypothesis. Next, the equality $o_2 \equiv o'_2$ results from Definition 2.3.

If the object o_2 is of a set form $\{o_{21}, o_{22}, \ldots, o_{2n}\}$, then o'_2 will be an object of a set form $\{o'_{2(j1)}, o'_{2(j2)}, \ldots, o'_{2(jm)}\}$, where $jm \leq n$, obtained in the following way:

- for each atom object o_{2i} from the set o_2 one must put into the set o'_2 the atom object $o'_{2(jk)} =_{id} o_{2i}$,
- for each such tuple object o_{2i} from the set o_2, that: $\neg(o_{2i} \equiv \top) \wedge \neg\exists(1 \leq m \leq i-1)(o_{2i} \equiv o_{2m})$ one must put into the set o'_2 such a tuple object $o'_{2(jk)}$, that $\{o'_{2(jk)}\}$ is in the *first_normal* form and the relation $o_{2i} \equiv \{o'_{2(jk)}\}$ holds – see the previous case,
- no more object can be added to the set o'_2.

The algorithm of obtaining the set o'_2 and Definition 2.3 guarantee that o'_2 is in the *first_normal* form and the equality $o_2 \equiv o'_2$ holds. ■

Proof of Lemma 2.2. A proof can be obtained in a similar way as that of Lemma 2.1: by induction on *object_depth* of o. ■

Proof of Lemma 2.3. The proof is again by induction on *object_depth* of o. In the case of *object_depth*$(o) = 1$, o must be a set object consisting of any number (including zero) of atom objects or the special object $\perp$. Then the required object o' will take a form identical to that of the object $o : o' =_{id} o$. Obviously, no other object $o'' \in \underline{\mathbf{O2}}$ can fulfil the condition $o \equiv o''$.

Let us now assume, that the lemma holds for each object $o_1 \in \underline{\mathbf{O2}}$ such that *object_depth* $(o_1) \leq n-1$ (with $n > 1$). We will prove that it also holds for each object $o_2 \in \underline{\mathbf{O2}}$ fulfilling the condition: *object depth* $(o_2) = n$. Let us observe that only o_2 of a set form $\{o_{21}, o_{22}, \ldots, o_{2n}\}$ with at least one o_{2i} $(1 \leq i \leq n\}$ of a tuple form $[a_1 : o'_{i1};\ a_2 : o'_{i2};\ \ldots;\ a_z : o'_{iz}]$ may fulfil this condition. Let us remark that for each object o'_{ij} $(1 \leq j \leq z)$: *object_depth* $(o'_{ij}) \leq n-1$. Then, from the induction hypothesis, it is immediate that for each object o'_{ij}

$\{o''_{ij1}, o''_{ij2}, \ldots, o''_{ij(mj)}\}$ and the relation $o'_{ij} \equiv o''_{ij}$ is true. As the result, from Definition 2.3 we obtain: $o_2 =_{id} \{o_{21}, o_{22}, \ldots, o_{2(i-1)}, [a_1: o'_{i1};\ a_2: o'_{i2};\ \ldots;\ a_z: o'_{iz}], o_{2(i+1)}, \ldots, o_{2n}\} \equiv \{o_{21}, o_{22}, \ldots, o_{2(i-1)}, [a_1: o''_{i1};\ a_2: o''_{i2};\ \ldots;\ a_z: o''_{iz}], o_{2(i+1)}, \ldots, o_{2n}\} =_{id} o_3$. Let us assume, that $\{k1, k2, \ldots, ks\} = \{1 \leq p \leq m1 | (\neg\exists(1 \leq r \neq i \leq n)(\{[a_1: o''_{i1p}; a_2: o''_{i2}; \ldots; a_z: o''_{iz}]\} \leqq \{o_{2r}\}))\}$. Then, again from Definition 2.3, we can deduce: $o_3 \equiv \{o_{21}, o_{22}, \ldots, o_{2(i-1)}, [a_1: \{o''_{i1(k1)}\}; a_2: o''_{i2}; \ldots; a_z: o''_{iz}], [a_1: \{o''_{i1(k2)}\}; a_2: o''_{i2}; \ldots; a_z: o''_{iz}], \ldots, [a_1: \{o''_{i1(ks)}\}; a_2: o''_{i2}; \ldots; a_z: o''_{iz}], o_{2(i+1)}, \ldots, o_{2n}\}$. We can apply the same procedure to the objects $o''_{i2}, o''_{i3}, \ldots, o''_{iz}$ and next to all the remaining objects o_{2j} $(1 \leq j \neq i \leq n)$ of *object_depth* ≥ 2. The final result of such an application will be an object o_f in the *elementary* form such that $o_2 \equiv o_f$. Let us complete these considerations by stating that o_f is the only object from the set **O2** fulfilling both of the lemma constraints. The last conclusion can be deduced from the fact that for any two objects o_1, o_2 in the *elementary* form $o_1 \equiv o_2$ if and only if $o_1 =_{id} o_2$. ■

Proof of Lemma 2.4. While reflexivity is obvious (it follows directly from Definition 2.6), let us fix our attention on the proof of transitivity and anti-symmetry properties.

Let us assume that $o_1, o_2 \in$ **O2** are objects such that $o_1 \leqq o_2$ and $o_1 =_{id} \{o_{11}, o_{12}, \ldots, o_{1m}\}$ and $o_2 =_{id} \{o_{21}, o_{22}, \ldots, o_{2n}\}$. From Definition 2.6 we deduce that for each object $\{o_{1i}\}$ $(1 \leq i \leq m)$ there exist object $\{o'_{1i}, o'_{21}, \ldots, o'_{2k}\} \in$ **O2** such that $o_2 \equiv \{o'_{1i}, o'_{21}, \ldots, o'_{2k}\}$ and $\{o_{1i}\} = \{o'_{1i}\}$. On the other hand, from Lemma 2.3 it follows that for $\{o_{1i}\}$ as well as for $\{o'_{1i}\}$ there exists exactly one object o''_{1i} (respectively o'''_{1i}) in the *elementary* form, for which the relation $\{o_{1i}\} \equiv o''_{1i}$ (respectively $\{o'_{1i}\} \equiv o'''_{1i}$) holds. From Definition 2.3 and Lemma 2.3 it is immediate that the objects o''_{1i} and o'''_{1i} must be identical. Let o'_1, o'_2 be objects in the *elementary* form such that $o_1 \equiv o'_1$ and $o_2 \equiv o'_2$. As a consequence of the former deduction we conclude that the relation $o_1 \leqq o_2$ is true if and only if each element belonging to the set object o'_1 belongs also to the set object o'_2. Finally, for any objects $o_1, o_2, o_3 \in$ **O2** we obtain: if $o_1 \leqq o_2$ and $o_2 \leqq o_3$, then $o_1 \leqq o_3$.

The proof of the property of anti-symmetry can be achieved according to the same scheme. ■

Proof of Lemma 2.5. The proof of the lemma is decomposed into a sequence of intermediary steps. Let us first assume that:

- o_1 and o_2 are any objects belonging to **O2**,
- $o_3 = o_1 \cup o_2$,
- $o_4 = o_1 \cap o_2$,
- o'_1, o'_2, o'_3 and o'_4 are objects in the *elementary* form and such that, respectively:

  $o_1 \equiv o'_1$, $o_2 \equiv o'_2$, $o_3 \equiv o'_3$ and $o_4 \equiv o'_4$.

(0) It is easy to conclude that:

- any element belongs to the set o'_3 if and only if it belongs to the set o'_1 or to the set o'_2,
- any element belongs to the set o'_4 if and only if it belongs to the set o'_1 as well as to the set o'_2.

The first part of the conclusion can be deduced directly from Lemma 2.3 and Definition 2.8. The second part will be proved by contradiction. Let us assume that in the set o'_4 there exists an element o'_{4i} which belongs neither to the set o'_1 nor to the set o'_2. The element o'_{4i} must be obviously an atom or a tuple in the *elementary* form. In the first case the fact of existence of o'_{4i} in the set o'_4 directly implies (Lemma 2.3, Definition 2.9) the existence of o'_{4i} in the set o'_1 as well as in the set o'_2. In this way we come to a contradiction. Next, a tuple form of o'_{4i} obligatorily implies that in the set objects o_1 and o_2 there exist tuples, respectively o_{1j} and o_{2k}, fulfilling the condition: $\{o_{1j}\} \sqcap \{o_{2k}\} = \{o_{4l}\}$ such that $\{o'_{4i}\} \leqq \{o_{4l}\}$. But in such case the relations $\{o'_{4i}\} \leqq \{o_{1j}\}$ and $\{o'_{4i}\} \leqq \{o_{2k}\}$ must be true (Definition 2.9, Lemma 2.3, Definition 2.6). As a simple consequence o'_{4i} must belong to the set o'_1 as well as to the set o'_2. So, again we come to a contradiction. The same reasoning will lead us to a contradiction if we assume that in both of the sets o'_1 and o'_2 there exists an element $o'_{(1-2)i}$, which does not belong to the set o'_4.

(1) Let us now prove the first of the facts being the contents of this lemma. Considering the equality $o_1 \sqcup o_2 = \sup\{o_1, o_2\}$, the truth of the thesis: $(o_1 \leqq (o_1 \sqcup o_2)) \wedge (o_2 \leqq (o_1 \sqcup o_2))$ can be achieved directly from the intermediate conclusion derived in the proof of Lemma 2.4 and the first conclusion proved in step (0) of this lemma. On the other hand, the truth of the thesis: $\neg\exists(o' \in \underline{\mathbf{O2}})((o_1 \leqq o') \wedge (o_2 \leqq o') \wedge (o' \leqq (o_1 \sqcup o_2)) \wedge \neg(o' \equiv (o_1 \sqcup o_2)))$ can be proved indirectly, by contradiction. Let us negate the last thesis. If o' exists, then, on the strength of the intermediate conclusion derived in the proof of Lemma 2.4, each element from the set object o'_1 and each element from the set object o'_2 must also exist in the set object o'' in the *elementary* form such that $o' \equiv o''$. Continuing, each element from the set object o'' must also belong to the set o'_3. But from the first conclusion proved in step (0) of this lemma it follows that in o'_3 there are all such and only such elements, which exist in o'_1 or o'_2. As a consequence, the relation $o' \equiv (o_1 \sqcup o_2)$ must be true. In this way we come to a contradiction.

(2) The proof of the equality $o_1 \sqcap o_2 \equiv \inf\{o_1, o_2\}$ can be achieved according to the same scheme. ■

Proof of Lemma 2.6. (1,2) The first and second equalities can be easily derived from Definition 2.3, Lemma 2.3, the intermediate conclusions proved in step (0) of Lemma 2.5, and the known De Morgan laws.

(3) To derive the third of the equalities it is enough to prove that: $o_1 \sqcap (\sqsupset o_1)) \equiv \{\,\}$. It will be proved by induction on *object_depth* of o_1. Observe that for *object_depth*$(o_1) = 1$ o_1 must be a set object $\{ao_{(k1)}, ao_{(k2)}, \ldots, ao_{(ks)}\}$

$\{\perp\}$. In the former case, directly from Definition 2.10 we obtain: $(\neg o_1) = ((\neg\{ao_{(k1)}\}) \sqcap (\neg\{ao_{(k2)}\}) \sqcap \ldots \sqcap (\neg\{ao_{(ks)}\}))$. Then, from Definitions 2.10 and 2.9 it is easy to see that $(\neg o_1) = \{ao_{(l1)}, ao_{(l2)}, \ldots, ao_{(lt)}, []\}$, where $\{ao_{(l1)}, ao_{(l2)}, \ldots, ao_{(lt)}\} = \underline{\mathbf{AO}} - \{ao_{(k1)}, ao_{(k2)}, \ldots, ao_{(ks)}\}$. Finally, again from Definition 2.9, we have: $(o_1 \sqcap (\neg o_1)) = \{\}$. Similarly, in the latter case, directly from Definition 2.9 we obtain: $\{\perp\} \sqcap \{\} = \{\}$.

Let us now assume, that the equality $o_1 \sqcap (\neg o_1)) \equiv \{\}$ holds for any object $o_1 \in \underline{\mathbf{O2}}$ such that *object_depth*$(o_1) \leq n-1$ (with $n > 1$). We will prove that it also holds for any object $o_3 \in \underline{\mathbf{O2}}$, fulfilling the condition: *object depth* $(o_3) = n$. Let us first assume that $o_3 =_{id} \{[a_1 : o_{31}; a_2 : o_{32}; \ldots; a_z : o_{3z}]\}$. From the first equality of this lemma, Definitions 2.10, 2.9 and 2.8 and the induction hypothesis one can derive: $(o_3 \sqcap (\neg o_3)) = \{[a_1 : o_{31}; a_2 : o_{32}; \ldots; a_z : o_{3z}]\} \sqcap (\neg\{[a_1 : o_{31}; a_2 : o_{32}; \ldots; a_z : o_{3z}]\}) = \{[a_1 : o_{31}; a_2 : o_{32}; \ldots; a_z : o_{3z}]\} \sqcap (\{ao_1, ao_2, \ldots, ao_t\} \sqcup \{[a_1 : (\neg o_{31})]\} \sqcup \{[a_2 : (\neg o_{32})]\} \sqcup \ldots \sqcup \{[a_z : (\neg o_{3z})]\}) \equiv (\{[a_1 : o_{31}; a_2 : o_{32}; \ldots; a_z : o_{3z}]\} \sqcap \{ao_1, ao_2, \ldots, ao_t\}) \sqcup (\{[a_1 : o_{31}; a_2 : o_{32}; \ldots; a_z : o_{3z}]\} \sqcap \{[a_1 : (\neg o_{31})]\}) \sqcup (\{[a_1 : o_{31}; a_2 : o_{32}; \ldots; a_z : o_{3z}]\} \sqcap \{[a_2 : (\neg o_{32})]\}) \sqcup \ldots \sqcup (\{[a_1 : o_{31}; a_2 : o_{32}; \ldots; a_z : o_{3z}]\} \sqcap \{[a_z : (\neg o_{3z})]\}) = \{\} \sqcup \{[a_1 : (o_{31} \sqcap (\neg o_{31})); a_2 : o_{32}; \ldots; a_z : o_{3z}]\} \sqcup \{[a_1 : o_{31}; a_2 : (o_{32} \sqcap (\neg o_{32})); \ldots; a_z : o_{3z}]\} \sqcup \ldots \sqcup \{[a_1 : o_{31}; a_2 : o_{32}; \ldots; a_z : (o_{3z} \sqcap (\neg o_{3z}))]\} = \{\}$. Next, let us assume, that $o_3 =_{id} \{o_{31}, o_{32}, \ldots, o_{3m}\}$, where $m \geq 2$. In such a case, from Definitions 2.10 and 2.8 and the first equality of this lemma, we have: $(o_3 \sqcap (\neg o_3)) = \{o_{31}, o_{32}, \ldots, o_{3m}\} \sqcap (\neg\{o_{31}, o_{32}, \ldots, o_{3m}\}) = \{o_{31}, o_{32}, \ldots, o_{3m}\} \sqcap ((\neg\{o_{31}\}) \sqcap (\neg\{o_{32}\}) \sqcap \ldots \sqcap (\neg\{o_{3m}\})) = (\{o_{31}\} \sqcup \{o_{32}\} \sqcup \ldots \sqcup \{o_{3m}\}) \sqcap ((\neg\{o_{31}\}) \sqcap (\neg\{o_{32}\}) \sqcap \ldots \sqcap (\neg\{o_{3m}\})) \equiv (\{o_{31}\} \sqcap ((\neg\{o_{31}\}) \sqcap (\neg\{o_{32}\}) \sqcap \ldots \sqcap (\neg\{o_{3m}\}))) \sqcup (\{o_{32}\} \sqcap ((\neg\{o_{31}\}) \sqcap (\neg\{o_{32}\}) \sqcap \ldots \sqcap (\neg\{o_{3m}\}))) \sqcup \ldots \sqcup (\{o_{3m}\} \sqcap ((\neg\{o_{31}\}) \sqcap (\neg\{o_{32}\}) \sqcap \ldots \sqcap (\neg\{o_{3m}\})))$. Finally, as a consequence of the former case concerning the o_3 form, the induction hypothesis, Lemma 2.5 and Definition 2.8 we deduce: $(o_3 \sqcap (\neg o_3)) \equiv \{\}$. This ends the proof of the third equality of the lemma.

(4) The proof of the equality $(o_1 \sqcup (\neg o_1)) \sqcap o_2 \equiv o_2$ can be achieved according to the same scheme as the former one. ■