Intelligent Design of Class Structure Model based on Ontological Data Analysis

[1]A.N. Kovartsev <kovr_ssau@mail.ru>,
[1]V.S. Smirnov <victorsmirnov92@gmail.com>
[2]S.V. Smirnov <smirnov@iccs.ru>
[1]Samara State Aerospace University,
443086, 34 Moskovskoye shosse, Samara, Russia
[2]Institute for the Control of Complex Systems of RAS,
443020, 61 Sadovaya st., Samara, Russia

Abstract. This paper investigates a formal approach which supports a critically significant step in object oriented analysis and software engineering. It is proposed to create an object class structure model based on an Ontological Data Analysis of a targeted domain empirical data. This technology is a development of the well-known method of Formal Concept Analysis and is able to work with incomplete (contradictory, inaccurate, vague, etc.) empirical information on domain, naturally supports the construction of arbitrary binary relationships between classes of objects and takes into account available to researcher information about the interconnection between actual for the designer domain objects properties. Multi-valued vector logic models and means are usedin order to factor in the realities of the empirical data accumulation.In concurrence with this a nonstrict formal context is being formed to display the conceptual domain structure. In this context truth values of basic semantic proposition of the form “x object has y property” are presented in a vector form. Its transformation into a binary formal context, for which formal concepts output effective algorithms are known, is done using intellectual alpha approximation algorithm which takes into account typical relationships between the objects properties and, above all, a conceptual conjugation of object properties arising from the fundamental cognitive designer’s procedures – conceptual scaling of the objects properties detected. A properties inclusion partial order between derived from the context formal concepts appears which is known as inheritance of properties in object-oriented analysis. Determined by this ratio a formal conceptclosed lattice is transformed into a model that describes an objects class structure, according to a number of pragmatic design principles of this key software component.

Keywords: Object-Oriented Analysis and Design; Class Structure Model; Formal Methods; Ontological Data Analysis.

DOI: 10.15514/ISPRAS-2015-27(3)-5

For citation: Kovartsev A.N., Smirnov V.S., Smirnov S.V. Intelligent Design of Class Structure Model based on Ontological Data Analysis. *Trudy ISP RAN/Proc. ISP RAS*, vol. 27, issue 3, 2015, pp. 73-86. DOI: 10.15514/ISPRAS-2015-27(3)-5.

1. Introduction

Creating a *Class Structure Model* in object-oriented (OO) analysis and software engineering still remains an expert's experience realization subject [1-7]. Object and classes are the basis for the all next steps of analysis, however they "are there just for picking" (i.e. naturally appear in a statement of a problem) or are borrowed from colleagues (with or without any modification) [5]. In other words in practice there is no any systematic procedure or formalism supporting the critical for the further software engineering step.

At the same time the majority of coryphaeus in OO methodologies pointed out the necessity of a certain conceptual analysis of domain for "concepts" description. That is why a strict mathematical theory *Formal Concept Analysis* (FCA) [8] enthused object-oriented analysis and software engineering experts. Numerous researches and developments using FCA for creating Class Structure Model were accomplished. For example [9-11].

FCA is a theoretically well-founded and actively developing method of data analysis, which reflects the classical approach to a concept as a fundamental epistemological element defined by *extent* and *intent*.

Let's illustrate FCA's potential by an example of well-known OO methodologies taxonomy generating in terms of their diagram techniques [10]. Table 1 describes the match *I* between two sets: set of "*objects*" *G* – methodologies and set of "*attributes*" *M* – techniques.

Table 1. OO methodologies and their techniques.

Diagram types *OO methodologies*	Use case diagram	Class diagram	Sequence diagram	Collaboration diagram	Statechart diagram	Activity diagram	Component diagram	Deployment diagram	Timing diagram	Data flow diagram	Object diagram	State transition graph	Fence diagram	Domain chart
UML	×	×	×	×	×	×	×	×						
Booch		×		×	×		×	×	×					
Coad&Yourdon		×			×					×				
Jacobson	×	×	×		×						×	×		
Martin&Odell		×		×	×	×							×	
Rumbaugh		×	×	×	×					×				
Shlaer&Mellor		×		×	×		×			×				×

The tuple $\boldsymbol{K} = (G, M, I)$ – *Formal Context* (FC) – puts together the basic data for FCA. Particularly, using FCA methods we can establish from $\boldsymbol{K}$:

- set of *formal concepts* $\mathbf{B}(\boldsymbol{K}) = \{(X, Y) \mid X \subseteq G, Y \subseteq M, X = Y', Y = X'\}$, wherein: (X, Y) – Formal Concept; X - extent and Y - intent of a concept;

«'» - Galois statements; $X' = \{m \mid m \in M, \forall g \in X\text{: } gIm\}$ the set of attributes common to all objects in X; $Y' = \{g \mid g \in G, \forall m \in Y\text{: } gIm\}$ the set of objects that have all attributes from Y;

- complete *concept lattice* ($\boldsymbol{B}(\boldsymbol{K})$; $\leq$), in which sub-/super concept relation $(X_1, Y_1) \leq (X_2, Y_2)$, iff $X_1 \subseteq X_2$, ($\Leftrightarrow Y_1 \supseteq Y_2$).

From context in Table 1 we can extract a set of formal concepts, shown in Table 2. Meaningfully these are all generalizations of OO methodologies in the aspect of diagram techniques. *Partial order* (“inheritance”) between extracted concepts is shown in the grid in Fig. 1.

Therefore FCA delivers *domain’s conceptual structure* from available data in the form “objects-attributes”. This structure was proposed by FCA’s protagonists as a basis for creating a model describing the designed software class structure.

However, it emerged that FCA usability is limited.

- Construction of *arbitrary relationships* between object classes is not supported, except for the generalization relationship “is-a”.
- Contradictions in the original data – a set of *Basic Semantic Propositions* of the form “*x* object has *y* attribute” are prohibited. Especially the possibility of taking into account the evidence “for” and “against” the truth of such judgments.

Table 2. Formal concepts in domain “OO methodologies and their techniques”.

Name	*Extent*	*Intent*
1		Use case diagram, Class diagram, Sequence diagram, Collaboration diagram, Statechart diagram, Activity diagram, Component diagram, Deployment diagram, Timing diagram, Data flow diagram, Object diagram, State transition graph, Fence diagram, Domain chart
2	UML	Use case diagram, Class diagram, Sequence diagram, Collaboration diagram, Statechart diagram, Activity diagram, Component diagram, Deployment diagram
3	Booch	Class diagram, Collaboration diagram, Statechart diagram, Component diagram, Deployment diagram, Timing diagram
4	UML, Booch	Class diagram, Collaboration diagram, Statechart diagram, Component diagram, Deployment diagram
5	Coad&Yourdon, Rumbaugh, Shlaer&Mellor	Class diagram, Statechart diagram, Data flow diagram
6	UML, Booch, Coad&Yourdon, Jacobson, Martin&Odell, Rumbaugh, Shlaer&Mellor	Class diagram, Statechart diagram
7	Jacobson	Use case diagram, Class diagram, Sequence diagram, Statechart diagram, Object diagram, State transition graph
8	UML, Jacobson	Use case diagram, Class diagram, Sequence diagram, Statechart diagram
9	Martin&Odell	Class diagram, Collaboration diagram, Statechart diagram, Activity diagram, Fence diagram

10	UML, Martin&Odell	Class diagram, Collaboration diagram, Statechart diagram, Activity diagram
11	UML, Booch, Martin&Odell, Rumbaugh, Shlaer&Mellor	Class diagram, Collaboration diagram, Statechart diagram
12	Rumbaugh	Class diagram, Sequence diagram, Collaboration diagram, Statechart diagram, Data flow diagram
13	UML, Rumbaugh	Class diagram, Sequence diagram, Collaboration diagram, Statechart diagram
14	UML, Jacobson, Rumbaugh	Class diagram, Sequence diagram, Statechart diagram
15	Shlaer&Mellor	Class diagram, Collaboration diagram, Statechart diagram, Component diagram, Data flow diagram, Domain chart
16	UML, Booch, Shlaer&Mellor	Class diagram, Collaboration diagram, Statechart diagram, Component diagram
17	Rumbaugh, Shlaer&Mellor	Class diagram, Collaboration diagram, Statechart diagram, Data flow diagram

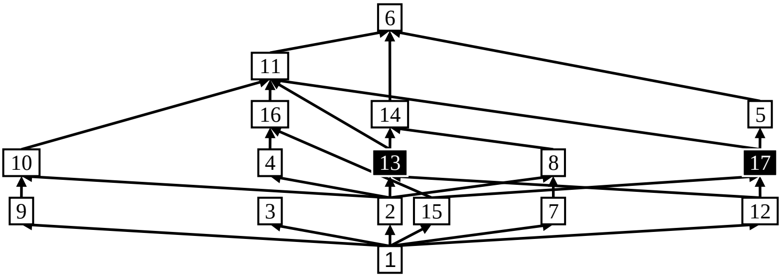

Fig. 1. Line diagram of concept lattice "OO methodologies and their techniques".

- Available to the designer information about the *relationship between attributes* of objects is ignored – the so-called attributes' "*constraints of existence*".

Although it is somewhat dampened the interest in FCA in software engineering, the method continued to develop, especially in the field of ontological modeling, for example [12, 13].

The main point of this paper is to draw developers' (especially, class structure model designers) attention to *Ontological Data Analysis* (ODA), the FCA evolution which can process vague and controversial data of modeled reality, discover arbitrary relationships between object classes and consider properties' limits of existence [14-16].

The topic of the article comes out in Fig. 2 diagram of ODA realization for class structure model design.

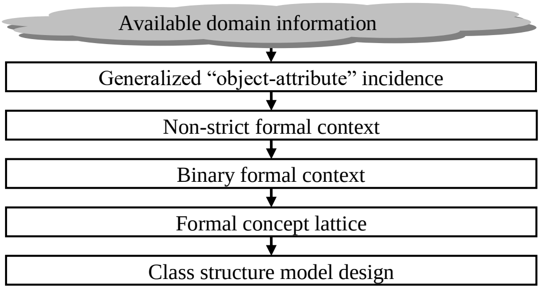
Available domain information
Generalized “object-attribute” incidence
Non-strict formal context
Binary formal context
Formal concept lattice
Class structure model design

Fig. 2. Ontological Data Analysis diagram for domain class structure model design

2. Ontological Data Analysis and Formal Concept Analysis

ODA is a customization and a pragmatic readjustment of FCA.

For FCA primary source of initial data is a multi-valued context – “objects-attributes” incidence (OAI) where observed domain objects’ attributes of researcher’s interest are noted.

In ODA the format of OAI is getting more complicated in order to represent domain empiric information, such as multiple independent object’s attribute records, discovering the same attribute with procedure sharing, confidence differentiation for different sources of information etc.

Besides that, as long as relations presence in ODA is treated as objects’ inner attributes demonstration, in OAI special associated attributes-valences pairs are used to represent arbitrary binary relations. This approach allows us to naturally “insert” a modeling of arbitrary relations between objects to FCA [15].

Only “weak” Basic Semantic Propositions' estimations for domain could be extracted from such generalized OAI. These estimations form in ODA a non-strict FC for conceptual framework extraction. Whereas for FCA usage a *binary* FC is necessary. Therefore ODA offers an approach for generating such FC from initial non-strict FC.

3. Non-strict Formal Context generation

In OAI (general scientific form for logging empirical information) rows correspond to domain objects, columns correspond to set of objects’ attributes that are recorded by measurement procedures available to the analyst. Table cells (matrix *V*) store the measurement results:

- set of objects $G^* = \{g_i\}_{i\,=\,1,\ldots,\,r}$, $r = |G^*| \geq 1$,
- set of attributes $M = \{m_j\}_{j\,=\,1,\ldots,\,s}$, $s = |M| \geq 1$,
- attributes measurement results $V = (v_j)_{i=1,\ldots,\,r;\,j=1,\ldots,\,s}$.

Generalized OAI is represented by tuple (G^*, M, Se, Pr, A), where:

- $Se = \bigcup_{i=1}^{r} Se_{(i)}$ - the set of all series of measurements, $|Se| = \sum_{i=1}^{r} |Se_{(i)}| = m$; $Se_{(i)} = \{se_{(i)k}\}_{k=1,...q_{(i)}}$, $q_{(i)} \geq 1$, $i = 1,\ldots, r$ – series of measurements, applied to object $g_i \in G^*$.
- $Pr = \bigcup_{j=1}^{s} Pr_{(i)}$ - arsenal of measurement procedures, $|Pr| = \sum_{j=1}^{s} |Pr_{(j)}| = n$; $Pr_{(j)} = \{pr_{(j)k}\}_{k=1,...p_{(j)}}$, $p_{(j)} \geq 1$, $j = 1,\ldots, s$, - set of measurement procedures used to estimate the value of the attribute $m_j \in M$, where any procedure $pr_{(j)k}$ has a degree of confidence in its results $t_{(j)k}$.
- $A = (a_{ij})_{i=1,\ldots, m;\, j=1,\ldots, n}$ – matrix of measurements series results *Se* of attributes *M* of objects from sample G^*, made using measurement procedures *Pr*. This matrix elements can be linguistic constants **NM**, **None**, **Failure** and **X**:
  None – a result that demonstrates a finding of a measured attribute value outside of sensitivity threshold and the dynamic range of a measuring instrument; it shows a “semantic mismatch” of the object and the measuring procedure etc;
  Failure – a result that records measurement failure (denial, measurement means malfunction, abstention, etc);
  NM (*not measured*) – a result indicating that as a matter of fact in this series of measurements corresponding property was not measured;
  X replaces any symbol of scales of dynamic ranges of measurement procedures *Pr*.

Non-strict FC is a tuple (G^*, *M*, *I*), where G^* - empirical training set of missile defense, *M* - number of attributes of objects recorded by measuring procedures available to the researcher, *I* - matrix estimates all the Basic Semantic Propositions, each element b_{ij} determined in accordance with the multi-valued logic V^{TF} vector $\langle True, False \rangle$ [17]:

$$\|b_{ij}\| = \langle b^{+}_{ij}, b^{-}_{ij} \rangle;\ b^{+}_{ij}, b^{-}_{ij} \in [0, 1],$$

wherein component *True* b^{+}_{ij} formed certificate confirming the Basic Semantic Proposition and the component *False* b^{-}_{ij} - denying it.

Building a non-strict incidence “objects-attributes” *I* begins with the transition from the primary data, structured in the form of a matrix *A*, to their semantic interpretation in the form of non-strict incidence “series-procedures” *I'*:

$$\left\|b_{ij}\right\|_{i=1,\ldots,m;\, j=1,\ldots,n} = \left\{ \begin{array}{ll} \mathbf{T} = \langle 1,\ 0 \rangle, & \text{if } a_{ij} = \mathbf{X}; \\ \mathbf{F} = \langle 0,\ 1 \rangle, & \text{if } a_{ij} = \mathbf{None}; \\ \mathbf{N} = \langle 0.5,\ 0.5 \rangle, & \text{if } a_{ij} \in \{\mathbf{Failure}, \mathbf{NM}\}. \end{array} \right\},$$

where **T**, **F** and **N** - truth constants V^{TF} logic of “True”, “False” and “Neutral” respectively.

Then line I' is transformed into a non-strict incidence “objects-attributes” I by combining the truth values of basic semantic judgments obtained for the object g_i in all series, and property m_j - all procedures (taking into account confidence in each procedure). Alignment is performed on various compositional rules V^{TF} logic [17].

4. Creating a binary Formal Contexts

Incidence “objects-attributes” I of non-strict FC can be expanded in his binary alpha-section, for example,

$$I = \bigcup\nolimits_{\alpha^+,\, \alpha^- \in [0,\,1]} \left\langle \alpha^+,\ \alpha^- \right\rangle \cdot I^{(\alpha)}\ ,$$

$$I^{(\alpha)} = (b^{(\alpha)}{}_{ij})_{i=1,\dots,\,r;j=1,\dots,\,s}\,,\quad b^{(\alpha)}{}_{ij} = \begin{cases} \textbf{True}, & \text{if } b_{ij}^{+} \geq \alpha^{+} \wedge b_{ij}^{-} \leq \alpha^{-}; \\ \textbf{False} & \text{in the opposite case} \end{cases},$$

wherein the alpha-section $I^{(\alpha)}$ - normal (binary) level corresponding vector $\alpha = \langle \alpha^+,\ \alpha^- \rangle$.

In practice, alpha-section $I^{(\alpha)}$ usually used as an approximation of so called «α-approximation» the original was not-strict incidence I. However, this method in the problem of forming a binary FC on its lack of rigor prototype is generally incorrect because the set of measured properties of M may exist a priori relationship “constraints of existence”.

Characteristic types of this kind of binary relations are considered in [18]. So a couple of properties m_j, $m_k \in M$, $j \neq k$ for each object data domain (and hence, for $\forall g_i \in G^*$) can be:

- *inconsistent* if, possessing property m_j, object g_i obviously does not have property m_k, and vice versa;
- *caused* if, possessing property m_j, object g_i indisputably has the property m_k, although the reverse may be wrong;
- *interdependent* if possessing property m_j, object g_i definitely has the property m_k, and vice versa.

The usual method of alpha-section is insensitive to such relations. Therefore, its application to the formation of a binary FC original non-strict context may lead to a violation of “constraints of existence.”

The idea of intelligent alpha-sectional non-strict FC is available for the formalization of context “constraints of existence” as a single predicate “α-section correctly” with argument “Threshold α of confidence in the source data” followed by the identification of the tolerance range α, delivering such a predicate **True**.

In general, set the specified area for non-strict FC is very difficult; it is possible and that it is empty. Therefore, to solve the problem correctly binary approximation non-strict FC in the ODA path is a reasonable compromise. Work with a common threshold of confidence α proposed to replace the manipulation of a set of thresholds of confidence in the data fragments that describe each object $g_i \in G^*$ at the level of each separately taken “constraints of existence”.

A very important case is when the inconsistently of attributes is the result of a fundamental cognitive procedure, known in FCA as a *conceptual scaling* [8]. This case is considered in [16], where proposed the method of rational alpha-section non-strict FC.

5. Formation of Class Structure Model

Analysis of binary FC allows deduce all the formal domain concepts. Formal concepts are partially ordered by inclusion of extensions (the extension of the concept - a set of objects, which are described by means of this concept) and form a complete lattice [8]. To use this result in the design of the software necessary to transform formal concept lattice in Class Structure Model.

Formal concepts according to the formation of their extensions are divided into three types:

- The concepts of the first type describe objects really exist in the analyzed domain. These concepts define a class of objects that deserve the naming of “*fundamental*”.
- The concepts of the second kind - only generalize other notions. In software design these classes are known as “*virtual*”.
- The third type of concepts is characterized by combining these features concepts first and second kinds.

When designing the Class Structure Model pragmatic considerations require confine fundamental and virtual classes of objects. In general, you can specify the following principles of formal concept lattice transformations in Class Structure Model:

- all the concepts of the lattice are candidates for fundamental classes of the model;
- the fundamental class becomes the minimum (in the terminology of lattices) concept containing the object in its extension;
- attribute is preserved to the maximum of the concepts contained this attribute in its intension;
- the highest concept lattice (his sign - power extension equal to the of objects) is certainly excluded from the model, if its intention is empty;
- the smallest concept lattice (his sign - the power intention equal to the power set of attributes) are known to be excluded from the model if its extension is empty;
- analysis of candidates in the fundamental classes begins with the smallest concept, and conducted by levels nearest super-concepts.

Algorithm which follows those principles is shown in table 3.

Table 3 – Concept lattice conversion into a class structure model algorithm

Step	Operation
1	The original version of the model is formed as *a copy* of the formal concept lattice.
2	In the model is searched *the greatest concept*. If *the intension of this concept is empty*, it is excluded from the model with break his ties with sub-concepts.
3	In the model is searched *the smallest concept*. If *extension* of the smallest concept is *empty*: • this concept is excluded from the model with the breaking its ties with super-concept; • a set of candidates in fundamental classes is formed of his closest super-concepts. If extension of the smallest concept is not empty, then a set of candidates in fundamental classes is formed of *one* smallest concept.
4	Loop through a set of candidates.
4.1	For each super-concept of the candidate under consideration excludes objects from extension that are within the extension of this candidate (the extension super-concept is always not less than the extension sub-concept).
4.2	In consideration of the candidate from the intension excludes any attribute that is part of the intension of at least one super-concept (a combination of all super-concept's intension is always not more than concept intent, which they are).
4.3	If the candidate has no sub-concepts, it is recorded as the fundamental class. In such case one of two alternatives is implemented: • if the candidate has no sub-concepts, it is recorded straight as a fundamental; • otherwise for this candidate creates a new sub-concept, in which the extension is transferred (and only extension) of the candidate. This new sub-concept is fixed as the fundamental class of objects. The intension of such fundamental class is empty. The candidate is retained in the model as a virtual class with an empty extension.
4.4	Promising set of concepts-candidates is unalterably filling with super-concepts of a current candidate.
5	Promising set of candidates is being reduced: remains only root concepts of generalization relationship, which is determined in a promising set of concepts-candidates.
6	If a set of promising candidates is not empty, then algorithm repeats from Step 4.
7	Classes with an empty extent and intent are excluded from a formed set. These could be only intermediate (i.e. not root or node class) classes of developed taxonomy.

In Fig. 3 class taxonomy after converting the formal concept lattice, shown in Fig. 2. Conspicuous is the fact that concepts 13 and 17 (highlighted in Fig.2) are absent in this taxonomy. Both of these concepts are losing their extent and intent after the conversion. Besides that in Fig. 3 the concept number 5 determines a fundamental class (all similar classes are highlighted) in order to describe the intent (and only intent) for which a special virtual class 05 implemented into the model.

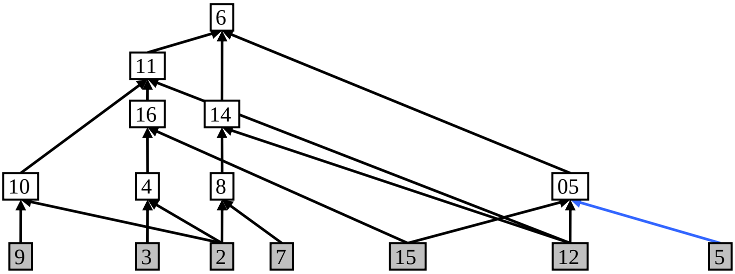

Fig. 3. Taxonomy of classes in domain “OO methodologies and their techniques”.

5. Conclusion

Formal Concept Analysis (FCA) has shown its benefits in many application areas – including the field of Software Engineering. Its use is especially valuable in the early stages of software development associated with the identification of a domain object types (classes) and relationships between these types.

Methodical equipment of the Ontological Data Analysis significantly expands and strengthens these advantages:

- can deal with incomplete and contradictory information about the data domain, namely a situation is typical for the beginning of the software life cycle;
- organically describes and analyzes arbitrary relations between classes of domain;
- take into account numerous priori known analyst relationship between the properties of domain (actually an additional cognitive resource that did not use the classic FCA).

Finally, the arsenal includes ODA pragmatically oriented algorithm for transforming formal concept lattice model in describing the structure of the classes. Formed model differs in that only describes two kinds of classes with a fundamentally different technical realization.

Acknowledgment

This work was conducting research on the topic “Development of the basic theory of intersubjective manage using ontological model of the situation” within the state task Institute for the Control of Complex Systems of Russian Academy of Sciences for 2013-2015, as well as public support of the Ministry of Education and Science of the Russian Federation in the framework of implementation of the Program of improving the competitiveness of Samara State Aerospace University among the world’s leading research and education centers for 2013-2020.

References

[1]. G. Booch. Object-Oriented Analysis and Design with Applications (2 ed.). Benjamin-Cummings Publishing, 1994. 608 p.

[2]. P. Coad and E. Yourdon. Object-Oriented Analysis. Prentice Hall, 1990. 223 p.

[3]. J. Martin and J. Odell. Object-Oriented Analysis and Design. Prentice Hall, 1992. 515 p.

[4]. S. Shlaer and S.J. Mellor. Object Lifecycles, Modeling the World in States. Yourdon Press, 1991. 268 p.

[5]. B. Meyer. Object oriented software construction (2 ed.). Prentice Hall, 1997. 1296 p.

[6]. G.N. Kalyanov. CASE strukturnyj sistemnyj analiz (avtomatizacija i primenenie) [CASE structural systems analysis (automation and application)]. Lori, Moscow, 1996. 242 p. (in Russian).

[7]. A.M. Vendrov. CASE-tekhnologii: sovremennye metody i sredstva proektirovanija informacionnykh sistem [CASE-technology: modern methods and tools for the design of information systems]. Finansy i statistika [Finance and Statistics], Moscow, 1998. 176 p. (in Russian).

[8]. B. Ganter and R. Wille. Formal Concept Analysis. Mathematical foundations. Springer-Verlag, Berlin-Heidelberg, 1999. 290 p.

[9]. R. Godin, H. Mili, G.W. Mineau, R. Missaoui, A. Arfi and T.-T. Chau. Design of Class Hierarchies based on Concept (Galois) Lattices. Theory and Application of Object Systems (TAPOS), 1998, 4(2), pp. 117-134.

[10]. S. Düwel, W. Hesse. Bridging the gap between Use Case Analysis and Class Structure Design by Formal Concept Analysis. In: J. Ebert, U. Frank (Hrsg.): Modelle und Modellierungssprachen in Informatik und Wirtschaftsinformatik. Proc. “Modellierung 2000”. Fölbach-Verlag, Koblenz, 2000, pp. 27-40.

[11]. W. Hesse and T. Tilley. Formal Concept Analysis Used for Software Analysis and Modelling. Formal Concept Analysis (Foundations and Applications) / LNAI 3626. Eds.: B. Ganter, G. Stumme and R. Wille. Berlin-Heidelberg, Springer-Verlag, 2005, pp. 288-303.

[12]. H.-M. Haav. A Semi-automatic Method to Ontology Design by Using FCA. Proc. of the CLA 2004 International Workshop on Concept Lattices and their Applications (Ostrava, Czech Republic, 2004, September 23-24). Eds.: V. Snasel, R. Belohlavek. TU of Ostrava, Dept. of Computer Science, 2004, pp. 13-24.

[13]. C. De Maio, L.V. Fenza and S. Senatore. Towards Automatic Fuzzy Ontology Generation. In: Proc. of the 2009 IEEE Int. Conf. on Fuzzy Systems (Jeju Island, Korea, 2009, August 20-24), pp. 1044–1049.

[14]. S.V. Smirnov. Ontologicheskij analiz predmetnykh oblastej modelirovanija [Ontological analysis of modeled domains]. Izvestija Samarskogo nauchnogo centra RAN [Proc. of the Samara Scientific Center of RAS], 2001, vol. 3(1), pp. 62-70 (in Russian).

[15]. S.V. Smirnov. Postroenie ontologij predmetnykh oblastej so strukturnymi otnoshenijami na osnove analiza formal'nykh ponjatij [Designing of ontologies by using Formal Concept Analysis in domains with arbitrary relationships]. Znanija – Ontologii -Teorii: Materialy Vserossijskoj konferencii s mezhdunarodnym uchastiem [Knowledge – Ontology – Theory: Proc. of the National Conf. with international participation] (Novosibirsk, Russia, 2011, October 3-5), vol. 2. Institut matematiki SO RAN [Sobolev Institute of Mathematics], Novosibirsk, 2011, pp. 103-112 (in Russian).

[16]. V.P. Ofitserov, V.S. Smirnov and S.V. Smirnov. Metod al'fa-sechenija nestrogikh formal'nykh kontekstov v analize formal'nykh ponjatij [Method alpha-section non-strict formal contexts in Formal Concept Analysis]. Problemy upravlenija i modelirovanija v

slozhnykh sistemakh: Trudy XVI mezhdunarodnoj konferencii [Complex System: Control and Modeling Problem. Proc. the XVI Int. Conf.] (Samara, Russia, 2014, June 30 – July 03). Samarskij nauchnyj centr RAN [Samara Scientific Center of RAS], Samara, 2014, pp. 228-244 (in Russian).

[17]. L.V. Archinski. Vektornye logiki: osnovanija, koncepcii, modeli [Vector logic: foundation, concepts and models]. Irkutsk: Irkutskij gosudarstvennyj universitet [Irkutsk State University], 2007. 228 p. (in Russian).

[18]. N. Lammari and E. Metais. Building and maintaining ontologies: a set of algorithms. Data & Knowledge Engineering, 2004, vol. 48(2), pp. 155-176.

Рациональное проектирование модели, описывающей структуру классов объектов, на основе онтологического анализа данных

[1]А.Н. Коварцев <kovr_ssau@mail.ru>,
[1]В.С. Смирнов <victorsmirnov92@gmail.com>
[2]С.В. Смирнов <smirnov@iccs.ru>
[1]СГАУ (НИУ), 443086, Россия, г. Самара, ул. Московское шоссе, дом 34
[2]ИПУСС РАН, 443020, Россия, г. Самара, ул. Садовая 61

Аннотация. Рассматривается формальный метод, обеспечивающий поддержку критически важного шага в объектно-ориентированном анализе и проектировании программного обеспечения. Предложено формировать модель, описывающую структуру классов объектов, на основе онтологического анализа эмпирических данных о целевой предметной области проектирования. Эта технология является развитием известного метода анализа формальных понятий и способна работать с неполной (противоречивой, неточной, нечеткой и т.п.) эмпирической информацией о предметной области, органично поддерживает построение произвольных отношений между классами объектов и принимает во внимание имеющиеся у исследователя сведения о взаимосвязи актуальных для проектировщика свойств объектов предметной области. Для учета реалий накопления эмпирических данных используются модели и аппарат многозначной векторной логики. При этом для задачи вывода понятийной структуры предметной области формируется нестрогий формальный контекст. Его преобразование в бинарный формальный контекст, для которого известны эффективные алгоритмы вывода формальных понятий, производится с использованием интеллектуального алгоритма альфа-аппроксимации, учитывающего типичные зависимости между свойствами объектов и, прежде всего, концептуальную сопряженность свойств объектов, возникающую в результате фундаментальной познавательной процедуры проектировщика – концептуального шкалирования регистрируемых свойств объектов. Между выведенными из формального контекста понятиями фиксируется частичный порядок по вложению свойств, известный в объектно-ориентированном анализе как наследование свойств. Определяемая этим отношением замкнутая решетка формальных понятий трансформируется в модель, описывающую структуру классов объектов, в соответствии с рядом прагматических принципов проектирования этого ключевого компонента программного обеспечения.

Ключевые слова: объектно-ориентированный анализ и проектирование; модель, описывающая структуру классов; формальные методы; онтологический анализ данных.

DOI: 10.15514/ISPRAS-2015-27(3)-5

Для цитирования: Коварцев А.Н., Смирнов В.С., Смирнов С.В. Рациональное проектирование модели, описывающей структуру классов объектов, на основе онтологического анализа данных. Труды ИСП РАН, том 27, вып. 3, 2015 г., стр. 73-86 (на английском языке). DOI: 10.15514/ISPRAS-2015-27(3)-5.

Список литературы

[1]. G. Booch. Object-Oriented Analysis and Design with Applications (2 ed.). Benjamin-Cummings Publishing, 1994. 608 p.

[2]. P. Coad and E. Yourdon. Object-Oriented Analysis. Prentice Hall, 1990. 223 p.

[3]. J. Martin and J. Odell. Object-Oriented Analysis and Design. Prentice Hall, 1992. 515 p.

[4]. S. Shlaer and S.J. Mellor. Object Lifecycles, Modeling the World in States. Yourdon Press, 1991. 268 p.

[5]. B. Meyer. Object oriented software construction (2 ed.). Prentice Hall, 1997. 1296 p.

[6]. Г.Н. Калянов. CASE структурный системный анализ (автоматизация и применение). – М.: Лори, 1996. 242 с.

[7]. А.М. Вендров. CASE-технологии: современные методы и средства проектирования информационных систем. – М.: Финансы и статистика, 1998. 176 с.

[8]. B. Ganter and R. Wille. Formal Concept Analysis. Mathematical foundations. Springer-Verlag, Berlin-Heidelberg, 1999. 290 p.

[9]. R. Godin, H. Mili, G.W. Mineau, R. Missaoui, A. Arfi and T.-T. Chau. Design of Class Hierarchies based on Concept (Galois) Lattices. Theory and Application of Object Systems (TAPOS), 1998, 4(2), pp. 117-134.

[10]. S. Düwel, W. Hesse. Bridging the gap between Use Case Analysis and Class Structure Design by Formal Concept Analysis. In: J. Ebert, U. Frank (Hrsg.): Modelle und Modellierungssprachen in Informatik und Wirtschaftsinformatik. Proc. “Modellierung 2000”. Fölbach-Verlag, Koblenz, 2000, pp. 27-40.

[11]. W. Hesse and T. Tilley. Formal Concept Analysis Used for Software Analysis and Modelling. Formal Concept Analysis (Foundations and Applications) / LNAI 3626. Eds.: B. Ganter, G. Stumme and R. Wille. Springer-Verlag, Berlin-Heidelberg, 2005, pp. 288-303.

[12]. H.-M. Haav. A Semi-automatic Method to Ontology Design by Using FCA. Proc. of the CLA 2004 International Workshop on Concept Lattices and their Applications (Ostrava, Czech Republic, 2004, September 23-24). Eds.: V. Snasel, R. Belohlavek. TU of Ostrava, Dept. of Computer Science, 2004, pp. 13-24.

[13]. C. De Maio, L.V. Fenza and S. Senatore. Towards Automatic Fuzzy Ontology Generation. In: Proc. of the 2009 IEEE Int. Conf. on Fuzzy Systems (Jeju Island, Korea, 2009, August 20-24), pp. 1044–1049.

[14]. С.В. Смирнов. Онтологический анализ предметных областей моделирования // Известия Самарского научного центра РАН, 2001, Т. 3, №1, с. 62-70.

[15]. С.В. Смирнов. Построение онтологий предметных областей со структурными отношениями на основе анализа формальных понятий // Знания – Онтологии – Теории: Материалы Всероссийской конф. с международным участием (3-5 октября 2011 г., Новосибирск, Россия). Т. 2. Новосибирск: Институт математики СО РАН, 2011, с. 103-112.

[16]. В.П. Офицеров, В.С. Смирнов, С.В. Смирнов Метод альфа-сечения нестрогих формальных контекстов в анализе формальных понятий // Проблемы управления и моделирования в сложных системах: Труды XVI международной конф. (30 июня - 03 июля 2014 г., Самара, Россия). – Самара: Самарский научный центр РАН, 2014, с. 228-244.

[17]. Л.В. Аршинский. Векторные логики: основания, концепции, модели. - Иркутск: Иркутский государственный ун-т, 2007. 228 с.

[18]. N. Lammari and E. Metais. Building and maintaining ontologies: a set of algorithms. Data & Knowledge Engineering, 2004, vol. 48(2), pp. 155-176.