UDC 004.7

Zykov S.
Discrete Mathematics and Cryptology Department,
Moscow Engineering Physics Institute (State University),
Kashirskoye Av., 31, Moscow, 115409, Russia,
E-mail: szykov@itera.ru

SUPPORTING THE ENTIRE LIFECYCLE: ENTERPRISE CONTENT MANAGEMENT METHODOLOGY

The paper considers content management in web portals, which are built on heterogeneous enterprise information systems (IS). A huge data bulk is accumulated by the enterprises, and it tends to increase further. Global distribution of the heterogeneous data and its weak-structured character complicate its uniform management. A problem-oriented approach for enterprise content management is presented, which includes a formal model and a software development toolkit. Implementation results demonstrate significant terms and cost reduction.

Keywords – enterprise information system, enterprise portal, data model, abstract machine, content management.

1. Introduction

State-of-the-art enterprise IS currently handle terabytes data (and metadata) of heterogeneous sources, which tend to change to petabyte-sized ones shortly. Such huge data volumes demand new models, methods and tools to manage them uniformly. A positive approach to solve the problem is portal-based enterprise content management (ECM). However, major software producers (Microsoft, Oracle, BEA etc.) lack adequate formal models in their ECM schemes, which makes the process vendor-dependent and non-uniform. Thus, the approaches known as yet either have model-level methodological "gaps" or do not result in enterprise-level solutions with practically applicable implementation features (scalability, expandability, availability, fault tolerance etc.).

The proposed ECM model, methodology and toolkit are parts of the integrated IS lifecycle support for heterogeneous portal-based environment [7].

ECM modelling methods suggested are based on a creative synthesis of finite sequences [1], categories [2,3], computations [5] and semantic networks [4].

2. The ECM Data Model

An object-based data model is suggested with data the following data representation:

$$Data = <class, object, value> \quad (1)$$

In Eq.1, under a *class* a collection of objects of the integrated enterprise database is implied. An *object* means an element of ECM IS portal page template. A *value* means a template data object instantiation and represents the problem domain dynamics.

Compared to results known as yet, benefits of the model suggested are more adequate mapping of heterogeneous weak-structured dynamic and static problem domains as well as event-driven (meta)data control in the global computational environment.

The data model is based on two-level *conceptualization* [6], i.e., the process of establishing relations between problem domain concepts. Objects, according to assigned types, are assembled into assignment-dependent collections, thus forming variable domains, which model dynamic and static problem domains.

The data model compression principle allows model application to classes, objects and values separately and to the data elements on the whole.

The integrated object model for data, metadata and states is characterized by structural hierarchical organization, scalability, metadata encapsulation and readability. Extendibility, adequacy, neutrality and semantic soundness of the formalism provide problem-oriented IS development with (meta)data object adequacy maintenance throughout the entire lifecycle.

The multi-parameter functional is introduced Eq.2:

$$F = F((v), (e), \ldots)(s)(p) \quad (2)$$

On the basis of Eq.2, where respective values in brackets represent client interface parameters, data access device parameters, user personal preferences and user registration status, a problem-oriented object portal personalization model has been built, based on functional $||F||$ evaluation.

Abstract machine for content management (AMCM) [7] is suggested as an ECM IS model, improving categorical abstract machine [2]. At any given moment AMCM is determined by its state; dynamics of its work cycle can be formalized by explicit enumeration of possible state changes.

From the formal model viewpoint, when portal page templates are mapped into the pages, variable *binding* evaluates the variables that represent template elements and their values, i.e. portal page elements.

AMCM semantics can be described in terms of semantic domain theory [5]:

1. Standard (most commonly used within the model framework) domains are defined;
2. Finite (containing explicitly enumerable elements) domains are defined;
3. Domain constructors (operations building new domains out of the existing ones) are defined;
4. Complex domains are built out of the standard ones using constructors, which include functional space [D1→D2], Cartesian product [D1×D2×…×Dn], disjunctive sum [D1+ D2+… +Dn] and sequence D*.

Let us collect all possible language identifiers into Ide domain, commands – into Com domain, and expressions – into Exp domain. AMCM language contain environment model is constructed as follows:

$$St = Mem \times In \times Out; \quad (3)$$

$$Mem = Ide \rightarrow [Val + \{unbound\}]; \quad (4)$$

$$In = Val^*; \quad (5)$$

$$Out = Val^*; \quad (6)$$

$$Value = T_1 + T_2 + \ldots. \quad (7)$$

AMCM state, Eq. 3, is defined by "memory" state, Eq. 4, including input values (i.e. content, Eq. 5) and output values (i.e. web pages, Eq. 6) of the abstract machine. Therewith, under memory a mapping from identifier domain into value domain is implied, which is similar to lambda calculus variable binding. For correct exception handling, unbound element should be added to the domains. Value domain, Eq. 7, is formed by disjunctive sum of domains, which contain content types of AMCM language.

Semantic statements describe denotates (i.e. correct constructions) of AMCM (meta)data object manipulation language.

Semantic statements for basic AMCM language commands and expressions are presented in [7].

Constant denotates are their respective values in a form of ordered pair of <variable, value>, while program state remains unchanged.

Identifier denotates are identifiers bound with their values (if binding is possible) in a form of ordered tuples of <variable_in_memory, identifier, state>, while the state remains unchanged (an error message is generated if the binding is impossible).

Thus, ECM IS template binding with the content may result in AMCM state change and in a number of limited, predefined cases (particularly, under template and content type incompatibility) – in error generation.

Semantic statement for an AMCM command, which assigns content to template element, results in state change with substitution of content value by the identifier in memory.

3. ECM Customization

Let us apply the computational models introduced to the target ECM IS and the portal.

The problem domain model is based on two-level compression [6], which is interpreted here as establishing relationships between data object classes C of the integrated problem domain D, Eq.8:

$$C = Iw:[D] \ \forall v:D\ (w(v) \leftrightarrow \Delta) = \{v:D \mid \Delta\}, \quad (8)$$

In Eq. 8, C and D are in a relation of partial order with each other (C ISA D), and Δ is a condition of data object w belonging to class C (according to a problem domain expert).

Complex, "multi-dimensional" data objects are modeled as n-arity data object relationship, Eq. 9:

$$R^n = Iw: [V_1,..,V_n] \ \forall v_1:V_1 \ldots \forall v_n:V_n\ (w\ [v_1,\ldots,v_n] \leftrightarrow \Gamma) = \{[\ v_1:V_1,\ldots,v_n:V_n] \mid \Gamma\}. \quad (9)$$

Frame representation of Eq. 9 is given in [7].

Thus, any class of data objects is a collection of ordered pairs (vi:Vi), where vi is i-th attribute of the class, and Vi is type of the class.

However, class attributes contain not only data, but also metadata (e.g., object dimensions, integrity constraints, and a bit mask), defining class object access rights in ECM IS templates.

Under assignment a1, class C is instantiated with Δk template of ECM IS web page, Eq. 11. Therewith, evaluation of template collection M assigns the value "true" to its only element mi, the index of which equals to the template number (k), Eq. 10:

$$M= (m_1,\ldots, m_k,\ldots, m_N), \ \forall i=1,\ldots,N\ m_i \in \{0,1\}; \quad (10)$$

$$[M|\Delta_k] = (m_1^*,\ldots, m_k^*,\ldots, m_N^*), \text{ where } m_i^* = 1,\ i = k \text{ and } \ m_i^* = 0,\ i \neq k. \quad (11)$$

Besides, the metadata attributes v1,…,vn are instantiated with metadata objects, according to constraint conditions ti of template Γ- see Eq. 12:

$[(v_1{:}V_1,\ldots,v_n{:}V_n)]t_i = ([v_1]|\Gamma(t_1),\ldots, [v_n]|\Gamma(t_n)) = = (v_1{'}{:}V_1{'},\ldots,v_n{'}{:}V_n{'})$, where $V_i{'}ISA\ V_i,\ \forall i=1,\ldots,n$. (12)

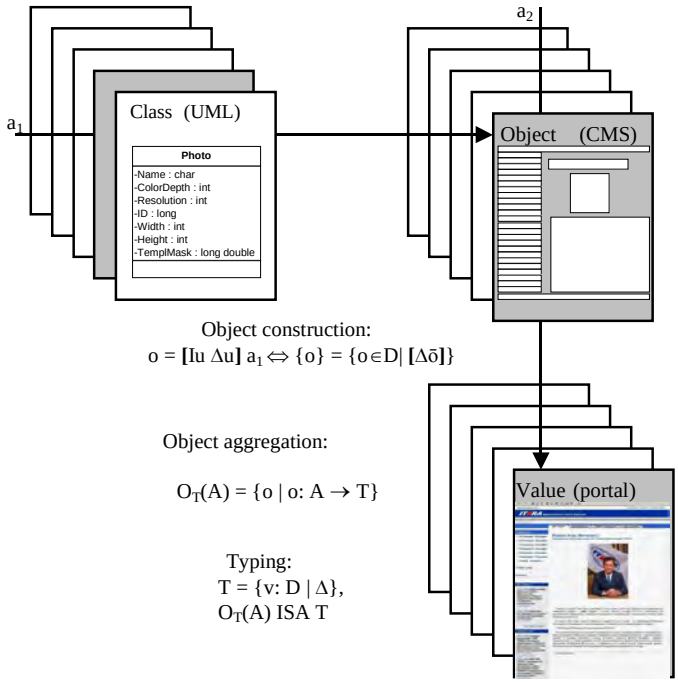

Fig.1 The ECM model and portal page generation.

The second assignment a2 results in (v1',…,vn') instantiation of the ECM IS web page template with content values of (c1,…, cn), Eq. 13:

$$[(v_1{'}{:}V_1{'},\ldots,v_n{'}{:}V_n{'})]c = (v_1{'}/c_1,\ldots, v_n{'}/c_n), \text{ where } c_1{:}C_1,\ldots, c_n{:}C_n;\ C_1\ ISA\ V_1{'},\ldots, C_n\ ISA\ V_n{'}. \quad (13)$$

Abstraction level of the model elements decreases from classes to objects, and to their values. Data object expandability provides extendable software development.

Object classes u are defined by means of description Iu Δ(u) with the values of [Iu Δ(u)], where Δ is a selection criterion. Assignments a1∈A and a2∈A transform the classes first into objects o = [Iu Δ(u)] a1 and then to their values: c = o a2.

Two-way assignment character (from classes to values and backwards) provides an adequate model of reengineering; description mechanism simplifies bi-directional modeling.

Variable domains represented by Eq. 14:

$$O_T(A) = \{o \mid o : A \rightarrow T\} \qquad (14)$$

are constructed as collections of objects o with types T, selected from problem domain D by selection criteria predicates Δ, while the collection of possible data objects o is contained in D, and the collection of actual ones, OT (A) is contained in T.

Thus, the basic modeling principle can be summarized as shown in Eq.15:

$$[\ class\ of\ objects\ ]:\ \ assignment \rightarrow object, \qquad (15)$$

where the bracketed part refers to the language level of class description, and the rest refers to the problem domain level. The principle means that to declare a class we use selection criteria, which identify functions from assignments into objects, i.e. class is treated like a process.

Data objects are identified as shown in Eq.16:

$$[\ object\ class\ ]:\ \ assignment \rightarrow object \blacktriangleright object \blacktriangleright assignment \rightarrow value \blacktriangleright value. \quad (16)$$

In Eq. 16, the symbol "➤" implies abstraction level decrease.

Thus, the diagram shown in Fig.1 illustrates the compression principle Eq. 17:

$$o = [Iu\ \Delta u]a_1 \Leftrightarrow \{o\} = \{o \in D \mid [\Delta(\bar{o})]\} \qquad (17)$$

and the essence of problem domain modeling process, which transfers language-level classes to problem-domain ones by the evaluation function [•]. An object class is modeled by its selection criteria Δ and its description I. Evaluation function maps problem domain objects to data definition language ones.

Fig.1 also gives an example of ECM modeling.

The example starts from a UML representation of a data class (digital photo image) with the attributes:

ID: char;

Name: int;
ColorDepth: int;
Resolution: long;
Width: int;
Heght: int;
TemplMask: long double;
The latter attribute is ECM template bit mask.

The first assignment a_1 fixes certain template attributes in the resulting enterprise portal web page. The second assignment a_2 instantiates the ECM template web page by the content values.

IS development methodology has been customized for enterprise portals, including information and interface (i.e. data and metadata) of Internet and Intranet sites [7].

A formal procedure for heterogeneous warehousing is suggested that allows user interaction with the integrated distributed (meta)database in a certain state, depending on dynamical script-activated assignments. Scripts depend on user-triggered events and provide transparent intellectual front-end interaction. Dynamic (meta)database access profiles provide reliable and flexible personalization, high fault tolerance and data security.

4. Conclusion

In this paper the novel ECM development approach description is given. The approach has been practically approved by constructing Internet and Intranet portals for ITERA International Group of Companies (http://www.iteragroup.com).

In terms of system architecture, the ECM IS provides assignments (depending on front-end position in data access hierarchy) with effective rights for (meta)data entry, modification, analysis and report generation (from administrator level down to reader one). Problem-oriented form designer, report writer, online documentation and administration tools make an interactive interface construction toolkit. (Meta)database supports integrated storage and online access to data and metadata (e.g., object dimensions, integrity constraints, representation formats, etc.).

Implementation process included fast prototyping (with mySQL DBMS and Perl scripting) and full-scale integrated Oracle-based implementation. The fast prototype proved adequacy of the (meta)data model and of the approach on the whole.

Upon prototype testing, the ECM IS has been developed, which manages Intranet and Internet portals (http://www.iteragroup.com).

Implementation proved 40% terms and costs reduction (on the average) as compared to commercial software available. Functional features have been essentially improved and include better ERP and legacy IS integration, easier handling of complicated objects and smarter report generation. Advanced personalization and access control have substantially reduced risks of (meta)data damage or loss [7].

An ECM construction methodology with lifecycle support introduced is a part of the integrated approach to enterprise IS development, which provides adequate, consistent and integrate (meta)data manipulation during its entire lifecycle.

A set of (meta)data object models have been constructed. It includes state-based dynamic models for problem domain and development tools. The models provide integrated (meta)data object manipulation in weak-structured heterogeneous problem domains.

A fast event-driven prototype and the full-scale ECM IS have been implemented. The IS manages Internet and Intranet portals for a corporation with around 10,000 employees of over 150 companies, located in 29 countries [8].

References

[1] Barendregt H.P. *The lambda calculus (rev. ed.)*, Studies in Logic, 103, North Holland, Amsterdam, 1984.

[2] Cousineau G., Curien P.-L., Mauny M. The categorical abstract machine. In: *Science of Computer Programming* 8(2): 173-202, 1987.

[3] Curry H.B., Feys R. *Combinatory logic*, Vol.I, North Holland, Amsterdam, 1958.

[4] Roussopulos N.D. *A semantic network model of data bases*, Toronto Univ., 1976.

[5] Scott D.S. Domains for denotational semantics. In: *ICALP* 1982, 577-613.

[6] Wolfengagen V.E. Event-driven objects. In: *CSIT'1999*. MEHhI, Moscow, Russia, 1999, Vol.I. p.p. 88-96, Sept. 1999.

[7] Zykov S.V. Integrated Methodology for Internet-based Enterprise Information Systems Development. In: *WEBIST 2005*, Portugal, Setubal, INSTICC, 2005, pp.168-175, Apr. 2005.

[8] Zykov S.V. Large-Scale Internet Systems Building Technology. In: *TECHNOMAT & INFOTEL 2006*, Bulgaria, Bourgas. Science Invest LTD – branch Bourgas, 2006, pp.18-29, May 2006.