Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 123 (2018) 386–392

Procedia Computer Science

www.elsevier.com/locate/procedia

8th Annual International Conference on Biologically Inspired Cognitive Architectures, BICA 2017

Architecture for Modular Type System for Information Systems Based on Relational-Applicative Technologies*

Vladimir Roslovtsev[1]

[1]*NRNU MEPhI, Moscow, Russia*

vladiros@gmail.com

Abstract

In the recent years a tendency in information systems development has manifested itself to use a (meta)data framework. While using software frameworks (user interface elements and engines, whole functional subsystems) has been common practice for decades by now, it is only relatively recently that developers began more and more often to shift away from relying on static subject domain conceptual schemes. This is in response partly to the tendency of growth of subject domain model in conceptual diversity, and partly in attempt to capture the shifting nature of subject domains. In this paper we describe an architecture and supporting data structures for a reusable engine that allows describing concepts, storing data objects with respect to the conceptual structure, and controllable modification of data and concepts.

Peer-review under responsibility of the scientific committee of the 8th Annual International Conference on Biologically Inspired Cognitive Architectures

Keywords: Modular Type System, Information Systems, Relational-Applicative Objects.

1 Introduction

Information systems automate, support or directly perform information processes in subject domains. And information systems, generally speaking, *contain* models of the (information) processes that in reality run outside these systems. The communication with the outside world is implemented either automatically using software and hardware intermediates, or via a human operator (whose responsibility is to feed the necessary data into the information system). Besides that, programs often *are* the environment in which processes directly exist and execute. Developing an information system generally consists of solving two major tasks:

- modeling objects in the domain, their structure, characteristics, relationships with other objects (where relationships sometimes are considered objects themselves);
- modeling processes, running in the domain, that involve objects, create, alter, transform them into objects, even destroy them, and processes themselves sometimes are considered objects.

The desired properties for both kinds of models are:

- their minimalism in terms of involved initial entities,
- completeness in the sense of ability to represent *all pertinent* features of objects, and represent them *adequately*.

Peer-review under responsibility of the scientific committee of the 8th Annual International Conference on Biologically Inspired Cognitive Architectures

10.1016/j.procs.2018.01.060

Achieving that goal helps, in turn, in obtaining certain useful results: conceptual integrity of the resulting program systems; compactness (and thus clarity and ease of understanding and use) of the conceptual framework of the domain; a high potential for analysis of the model; a relatively higher degree of reusability of the program code; better options for extensibility and customization of the program system.

One way of implementing such an approach in practice is to create a reusable core of an information system that can be then extended as needed for any specific application. In our case, the main part of the core would be facilities for describing elements of a conceptual model of a domain, including a database fragment. I this paper we describe our preferred approach to conceptual modeling and an architecture for the 'core' information system based on combined use of principles of the relational model and applicative computational systems (Barendregt, 2012).

The remainder of the paper is organized as following. Section 2 is a short overview of semantic models and our approach to conceptual modeling. Section 3 briefly presents the most important basics of applicative computational system relevant to the current discussion. And section 4 and 5 outline overall organization and certain implementation issues of the type system. Section 6 is a short conclusion.

2 Structure of Semantic Description of Subject Domains

The work of a system analyst performing conceptual modeling of a subject domain consists of several stages (Wolfengagen V. E., 1990):

1. selection and naming of individuals, as well as relationships between them, pertinent in the context of a specific problem;
2. classification of the selected objects and relationships, creating a system of partially-ordered classes (of objects and relationships both), with respect to the inclusion relation $\subseteq$);
3. selection (or synthesis) of a suitable formal language with necessary means to express the selected objects, relationships and the classification scheme;
4. injecting the discovered objects, relationships and classes into the language framework as constants of some kind or other.

Several such languages are well known and used, including: the entity-relationship model (ER-model), a number of variations of description logic (DL), the language (algebra) of frames.

Concepts. The central idea around which the modeling process revolves is to set up what is called 'conceptual structure' of the subject domain. By observing individual objects (often called individuals) we can formulate certain criteria according to which a number of individuals can be grouped to form a concept: the criteria are the intension of the concept, and the set of individuals is its extension (at a particular possible world, or state). Intensions of two concepts can be conveniently expressed as predicates, and usually in such a way that it is possible to infer whether one of the predicates 'includes' (implies) the other. It allows to establish the $\subseteq$ relation on the set of concepts. Partly as a defining factor for the $\subseteq$ relation, and partly as being of interest on its own, definition of each concepts is captured: is it initial, a restriction of an existing concept, a union (intersection) of two existing concepts, etc. In the ER-model concepts are represented through the entities, in DL and frames languages they are derived from initial domains and/or concepts.

Elements of a concept are usually described through a number attribute values; individuals of the same concept usually possess the same set of attributes, and differences in attributes values is one way to distinguish between individuals. The other way is individuals' structure: whether the object is believed atomic or complex, and, in the latter case, what are its constituents. Structure can be captured