Meta-Model and Platform for quickly build software applications

Yuri Rogozov, Alexey Degtyarev
System analysis and telecommunications.
Taganrog Institute of Technology, Southern Federal University
Taganrog, Russia
rogozov@tsure.ru, alexey.a.degtyarev@gmail.com

***Abstract*— In the development of large and complex applications is difficult to satisfy unique customer requirements without stretching development time. The need to support adaptability of applications is growing because of dynamic business processes. We analyzed of the literature in this domain. The analysis has shown that combination of the existing paradigm of application development is a good approach to support adaptability of business applications. This paper presents a multi-layered Meta-Model for enterprise applications and platforms that we have created on our multi-layered Meta-Model.**

Keywords - metamodel, metamodelling, meta-design, business application development platform, automation.

I. INTRODUCTION

The creation of business applications based on client-server architecture is still an urgent task. In this paper we will pay attention to the corporate business applications. The main features of such applications are heterogeneity (heterogeneity) of business information and variability processes that use this information. This creates a lot of problems in the development of business applications. Many of the studies on software project failures have identified difficulties of accurately capturing user requirements as a major contributor to failure of software development projects [1]. In the domain of software engineering a lot of effort has been spent to improve methods of capturing requirements. However, experience has shown that requirements cannot be initially captured completely and correctly [2]. In our opinion they are not static, but dynamically changing, evolving along with the world around us. To solve this problem software engineering proposed iterative approach to application development. The main idea of the approach is the speedy transfer of user working version [3]. Given the scale and complexity of corporate business applications, we can confidently say that even when using an iterative approach, the average time of development business applications is not less than six months. Approved by the functional and data structure in an early iteration of the project completion can be drastically revised by future system users. It will make some additional risks in the development process. That is why the support of business applications adaptability appears [27]. On the other side the market of the modern information technology, shows that universal automation solutions are ineffective [28], so the business application must be created individually for each customer [29].

To solve the above problems, we need such software tool that would allow: significantly reduce the complexity of each iteration; ensure the creation of individual decisions at different levels of detail (either total subsystem, or individual business functions); modify the functional and data structures at no additional cost, which were approved in early iterations.

Analysis of existing approaches to application development allowed us to determine that each of them focuses on a specific level of granularity, i.e. strictly defined in advance the items that are indivisible or atomic. In some approaches to such elements include operators of classical programming languages, classes and objects [4], in other cases, they include the whole program modules [5]. However, in our opinion, the approach Meta – Design is the most promising. [6]. A comparison of these approaches with its problems allowed us to assume that a multi-level Meta - Model platform was the most promising. To understand what must be the Meta - Model, we have generalized our knowledge in developing enterprise business applications. Then we built meta-level aspects of the model that described the business applications. This Meta-Model should be reflected in the Meta - Model implementation. We relied on our experience and knowledge and tried to make it more concrete. We chose the concept of service-oriented architecture as a starting point [7]. In this paper we propose a multi - level Meta - Model and platform for building client / server enterprise business applications.

Platform with developed AV Script scripting language and the unique structure of an independent data base [8] allow us to bind together the software details of different levels - from the software modules that implement support for business process or some of its functions to a low-level programming constructs perform mathematical operations and access to business information.

II. ANALYSIS OF EXISTING APPROACHES TO SOFTWARE BUILDING

During the development of software engineering a number of approaches to software development have formed. These approaches have both positive sides and restrictions. The most important methods of software development , proven during its existence presented below: a) Classical programming [9]; b) Generative programming [10]; c) Model Driven (Architecture, Design, Engineering) [11]; d) Feature-Driven

Development [12]; e) domain specific (languages ,modeling, design) [13]; f) Metaprogramming [14]; g) Software factory, Frame Technology [15].

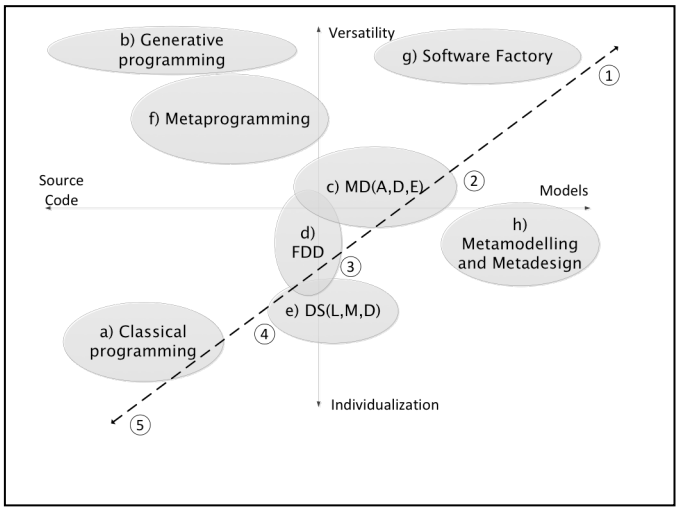


Figure 1. Existing approaches to software building

In our view, the criteria by which we can compare the approaches are following:

- level of personalization of the solutions determines how much depth we take into account the unique requirements of user (if the user wants a button, which shimmers like a rainbow, and we fulfill his requirement, it means that degree of personalization is high, if we make the four colors button only, because we have such templates, it means that degree of personalization is lower);
- level of formalization of the programming process, determines which way displaying the business problem in an existing space of solutions the process is carried out. (For example, if an application requires a creative activity for transformation of business models processes then the degree of formalization is low, and if an application has to perform a set of strictly defined rules for transformation of business models process then the degree of formalization is high) [16]. Then a comparison of approaches could be presented graphically in the following way (see Figure 1). The motion on the dotted line from left to right in Figure 1 shows the increasing abstraction level. So, to create a platform that would allow to reduce the complexity of each iteration, we need to maximize the formalization of the software development process.

Figure 1 shows that requirements to the platform could be met using Meta – design approach as the most attractive in accordance with our requirements [17].

However, the development of software at different levels of detail by constructing a common meta-model is most likely not succeeded. This is primarily because of constructing the complexity single universal Meta – Model [18, 19, 20]. Assumes that the Meta - Model should be multilevel and meet the following requirements: ensure that changes in the level of detail (see Figure 1); ensure the maximum degree of software formalization development processes.

Identify applicable sponsor/s here. If no sponsors, delete this text box. (sponsors)

III. ASPECTS THAT CHARACTERIZE THE CORPORATE BUSINESS APPLICATIONS

We believe that the modern corporate business applications aimed at collecting, storing and processing information in the most general case characterized by the following aspects (see Figure 2): work flow logic; graphic user interface; decision logic; data manipulating function; business objects. Note that some researchers in the software field have a similar view [21],[22],[23],[24], it makes our judgments more reasonable.

Work flow logic - an aspect that reflects the functions sequence performing, business applications and management tasks.

Graphic user interface - an aspect that reflects the interaction of users with information and business application.

Decision logic - an aspect that describes the implementation of logic solutions (calculation formulas, processing information about Business Objects).

Data manipulating function - determines how to work with data that represent real business objects. The standard functions for manipulating data include: the input new data, edit existing data, viewing of existing data, delete existing data.

Business Objects - a simplified model of real business objects, which characterizes them as the way which is important for building enterprise business applications.

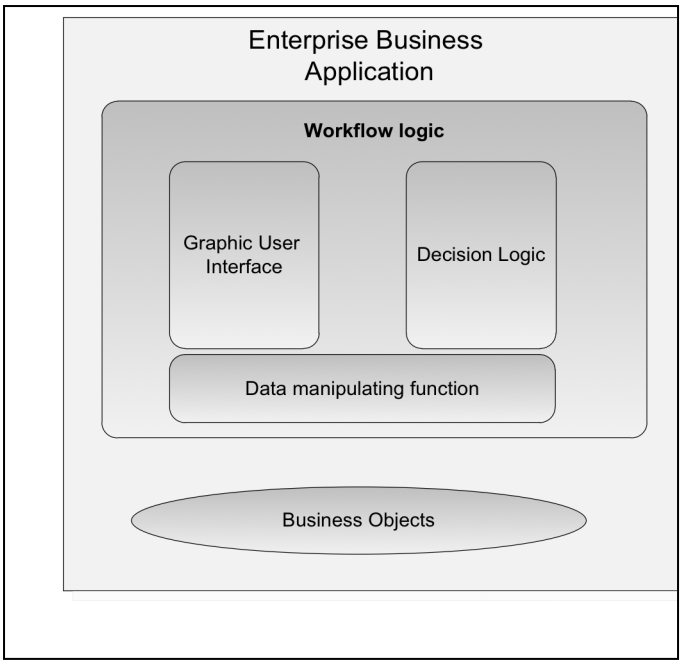


Figure 2. Aspects that characterize the corporate business applications

Figure 2 shows the Meta – Model aspects level that characterize the corporate business applications. This Meta - Model is abstracted from the domain and business objectives, i.e. is interdisciplinary. However, if we attempt to deepen it (refine) to a specific business problem we will be faced with problems. First, these problems are associated with a set of the

domain knowledge, which subject that works out in detail should know. Secondly knowledge of implementation technologies to create the functioning business applications is required. In other words, it is not enough to work out in detail aspects of the Meta - Model to the level of a specific business problem. It is necessary to display the specific business objectives for space implementation. Today it is a serious problem. Assume the following: concretized aspects of the Meta - Model elements, will find their place in any of the approaches to building software that depicted in Figure 1.

Then conceptually multi-layered meta-model space realizations can be represented as a plane that is divided into different levels of abstraction (see Figure 3). In this view, you can find the correspondence between the abstraction level above approaches and input levels of abstraction space realizations (see numerals in Figures 2 and 3). Depth of detail and level of abstraction is limited only by the degree of our knowledge and technology developments i.e. can grow indefinitely.

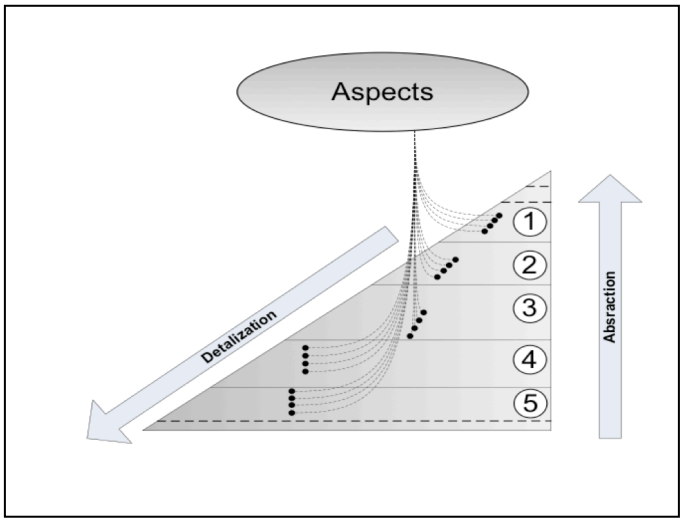


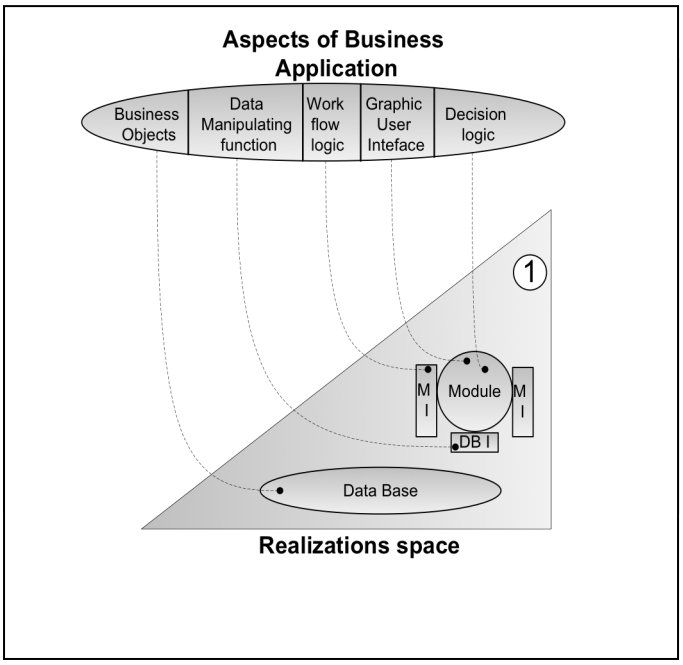
Figure 3. Mapping aspects

IV. MULTI-LEVEL META - MODEL CORPORATE BUSINESS APPLICATIONS

Multi-level Meta- Model was constructing by using the idea behind meta-approach of building prototypes of the object [25]. To determine the highest level of abstraction layered Meta- Model we used the principle of service-oriented architecture [7]. This principle bases that any enterprise business application is represented as a collection of loosely coupled modules. At the level of detail, where the module is an atomic unit, a business application can be represented as follows - Figure 4. By analogy with the terminology of object-oriented approach [26], the module is an object that encapsulates a user interface and business logic. Inter-module interfaces and interfaces to the database are external interfaces of the module. In Figure 4, weak ties are reflected by dotted lines. In accordance with Figure 3, consider how aspects of enterprise business applications are displayed at the proposed area of implementation. Analyzing the figures 2, 3 and 4 set a direct mapping is characterized by the corporate business applications in the proposed space realizations.

Figure 4. Meta-Model on level modules

Business objects are displayed in the database (Data Base), the logic of the process of business applications in the inter-module interfaces (Module interface), user interfaces and



business logic in the module (Module), the standard functions for manipulating data in an interface to the database (data base interface). This map is shown in Figure 5. Displays the next level of detail will be similar.

Figure 5. Mapping aspects of one level

Let's decompose the module into smaller components (see Figure 6) and introduce the corresponding concepts.

Module - is a certain amount of functional units (FU). They are strongly connected (tightly coupled) with each other and have minimal connection with the functional units (FU) of other modules. One of the functional units have to be the main (working with the central or main object in the context of the business objectives of the module) and the remaining subsidiaries.

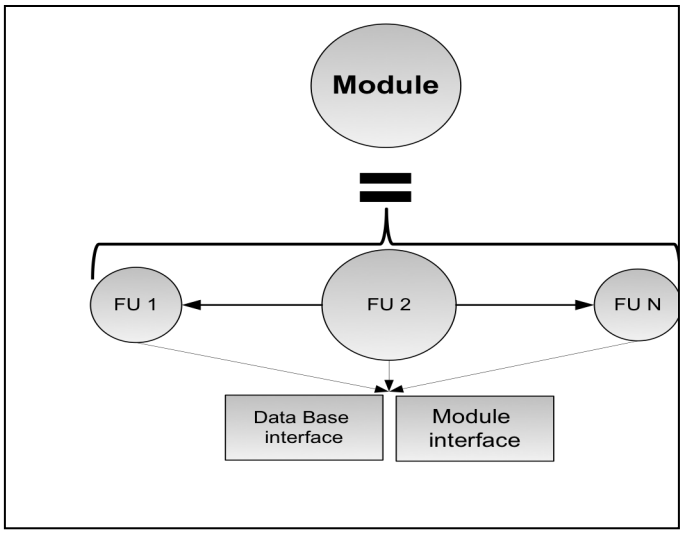


Figure 6. Meta-Model on level functional units

We showed Interface intermodule interactions (Module Interface) interface and work with DB (Data Base interface) on Figure 6 because the functional units themselves may interact with the database and functional units of other modules. In reality, the module interface and data base interface are located in the internal structure of functional units (FU). The solid lines in Figure 6 show that the functional units are combined into modules according to the principle of tightly coupled. We introduce the concept of functional units and continue to decompose.

Functional unit of the module (FE) - a graphical element (shape, window, Web page, etc.) which contains a set of elementary operations (EO). They are intended for display, modify, add or remove a business object's attributes or their values. Elementary operation may cause, another elementary operation or a functional unit of the module and perform other similar operations. In other words, a functional unit allows for a certain set of functions associated business processes.

Main functional unit (MFU) - functional unit of the module with the highest degree of connectivity (informational or functional) with other functional units of modules and comprises a master form. The remaining functional units of the module are subject-forms. We can lead explorer in MS Windows as an illustration. It folders and files are central to the business objects, so a window displaying them with all its functionality, is the primary functional unit. In turn, the window control which users can share any file or folder it can be a child or a subordinate.

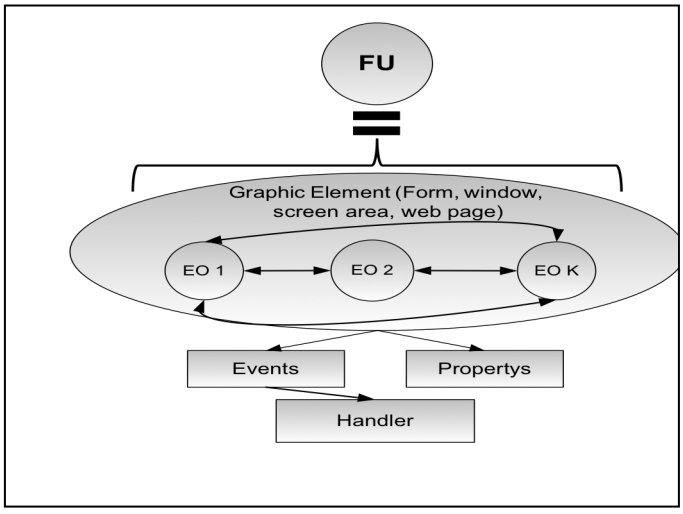


Figure 7. Meta-Model on level Graphic Element

The graphic element of a higher order (see Figure 7) - is a component, which is presented graphically in the window form or the screen. It has events and properties and contain a set of elementary operations. Events and properties of the graphical elements of higher order and a set of elementary operations together provide a complete description of the functional units of the module. Elementary operations can be arbitrarily interact with each other and interact with other functional units. There are two important limitations: an elementary operation cannot directly interact with the unit belonging operation to another graphic element, a functional unit of the module always has one and only one graphic element of a higher order, which represents a box shape of the screen or Web page.

Events - this is actually class methods (in the terminology of object-oriented programming) implements a specific graphic element (such as a mouse click - OnButtonClick (), or loss of window focus - FocusOff (), or closing the window - OnClose ()), each event if so stipulated requirements for business applications, can have its own handler.

Properties - the attributes that characterize a particular instance of a graphical interface. The properties of the graphic element may include the following parameters: name, description, the form position the top left corner along the axis X; the form position of the upper left corner along the axis Y; height form, the breadth of forms; color, font settings (font name, font size, bold, italic, underline), the visibility of the form (visible / invisible).

Elementary operation (EO) - is an elementary graphical elements (buttons, panels, switch, etc.), that connected with a single action function of a business process. The structure of its metadata elementary graphic element is very similar to the graphic element of a higher order. There is one exception, to be exact an elementary graphic element cannot contain elementary operations. Everything else is absolutely equally: events, properties and handlers. Let`s consider the structure of the handler (see Figure 8). The structure is identical for both the graphic element of a higher order (GE) and for an elementary graphic element (EGE).

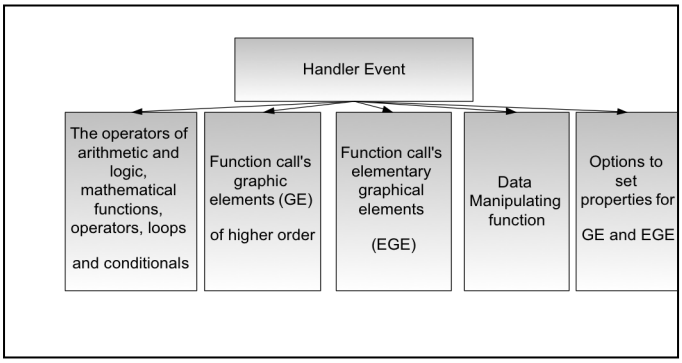


Figure 8. Meta-Model on level handler event

The operators of arithmetic and logic, mathematical functions, operators, loops and conditionals - these operators

are designed to implement the business logic (see Figure 1), their collection may vary.

Function call's graphic elements (GE) of higher order, function call's elementary graphical elements (EGE), options to set properties for GE and EGE - may represent a limited number of universal adaptive functions (performance depends on the method of implementation and using the framework).

Data Manipulating function - designed to implement interfaces with databases. Data Manipulating function can be variable. For stored objects in our database ("Object", "object attributes", "The values of object attributes") we released 12 basic functions for manipulating data. In case of need this set of functions can be expanded. The handler must have a certain syntax. The handler`s syntax can be a modern graphical syntax, built on the cognitive perception of reality as well as the classic text syntax.

A result of realizations stepwise decomposition of elements a Meta - Model was obtained. The Meta - Model has a divergent level of detail. Multi-level meta-atomic elements at the deepest level of detail can be considered: the elements of an event handler, basic graphic elements and graphic elements of the highest order. The last two sections cannot be divided but can be customized. Generalized multi-layered meta-model is shown on Figure 9.

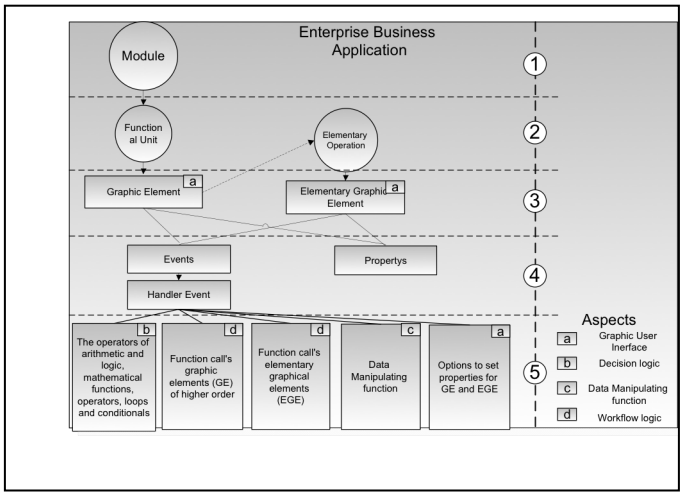


Figure 9. Meta-Model on all levels

We should note that at all levels either explicitly or implicitly event element as part of an integration is used. The syntax of the handler describes the event.

V. A PLATFORM TO QUICKLY BUILD BUSINESS APPLICATIONS

Our platform is based on the multi-level Meta – Model. The Meta - Model presented above has been implemented as bundles of desktop applications and server. Server stores the description of the components, business information and all service information. We tried to make the most simple quickly creation environment of business applications. It contains only the most necessary elements for building business applications. We will see a concise workspace, when we launch Primius platform. Since launching platform Primius, we will see a concise workspace. Main navigation menu consists of 2 major categories: Administration and modules that are available in selected business applications (see Figure 10). Figure 10 shows that the current business application consists of 3 modules - Drugstore, Clinical nutrition, Employee.

The modules are workstations of employees in the enterprise information system. Each application contains several functional units. The work of the existing modules is no differ from other business applications that is why it will not be considered. Figure 11 shows the category of platform administration. This section consists of three sections: Configuration, System Editor, and Business Object Editor. The Configuration partition partially consistent with the hierarchy of levels of multilevel Meta - Model we have developed. The creation and configuration the necessary business applications are made by this section. Section System Editor is designed to create new and modify existing items that are available in the categories section of Configuration. The work of Configuration partition always starts with the choice of Domain. Any further action should be tied to a specific application, so long as the current application is not specified, the other tabs will be unavailable.

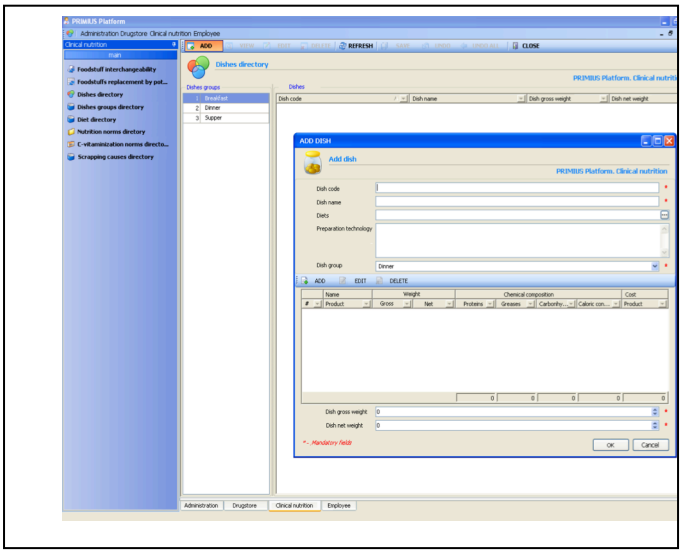


Figure 10. Modules those are available in selected business applications

A configuration on the modules is done by adding them to the list of available in the selected application modules. Every time you add a new module to the application, the metadata of the business data changes. Features of the implementation of this framework were considered in [8]. Configurations of next detail levels of elements realizes similarly. Figure 12 shows the Business Objects Editor. Any business object (real or abstract) is displayed by directory of objects that reference the attributes and the type of relationship between objects and attributes. All business objects and their attributes are linked to a specific system. There are two ways to create or change the structure of business data: automatic and manual.

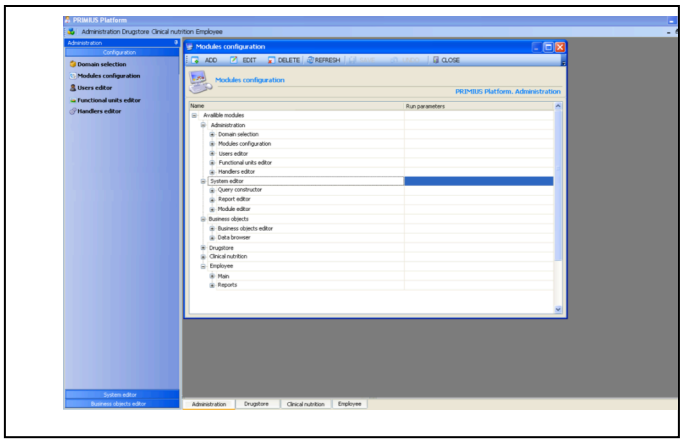


Figure 11. Modules configuration

Automatic mode is available when developer operating at the level of modules. In other cases, the structure of business data should be prepared manually by editing the business objects.

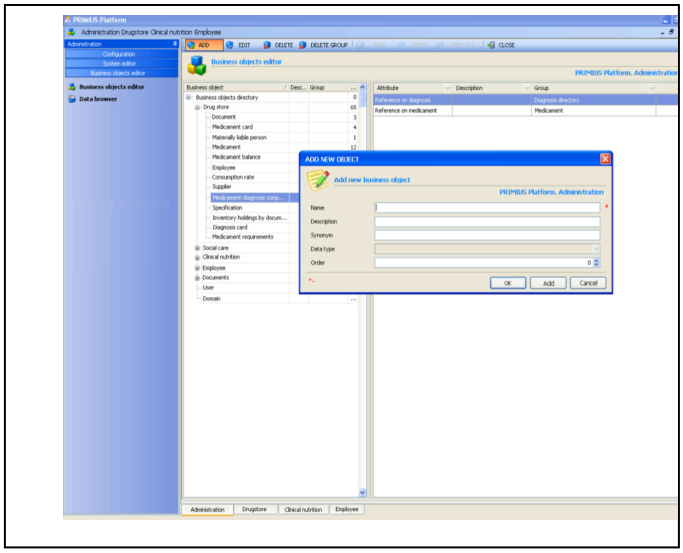


Figure 12. Business object editor

VI. CONCLUSION

In this article we looked at popular approaches to development programs and analyzed them for the important criteria to us. We concluded that to solve the problems of software development which relate to development time, individuality and flexibility of these solutions the platform should use a combination of approaches Meta design and factory applications. We have also assumed that the basis of the platform should be based on a multi-level Meta - Model. Analysis of the aspects characterizing the corporate business applications enabling us to understand what kind of needs multi-level Meta - Model should be. Our platform is based on the Meta - Model that we have developed. We have developed a unique structure of the data [8] to implement the ideas embodied in the Meta - Model. The structure is a common repository that describes elements of Meta, Meta data describing the business objects and data from which these objects work. This platform allows reducing the demands on staff. To explore this indicator we have attracted students to develop real-world business applications. Students were at 2 and 3 year bachelor's degree programs in "Computer Science and Engineering." One of the groups studied the C # and SQL in the IDE Microsoft Visual Studio 2005, another group of students studying specialized algorithmic language AV Scripts, SQL and basic knowledge of operating principles in the platform Primius. Our research group specializes in the development of fairly large business applications for the social protection institutions. We detected the following fact: in both groups to obtain basic knowledge took on average a similar time. However, the results between the groups were very different. Figure 13 reflects the evaluation of professional commitment to the development of applied business applications of groups.

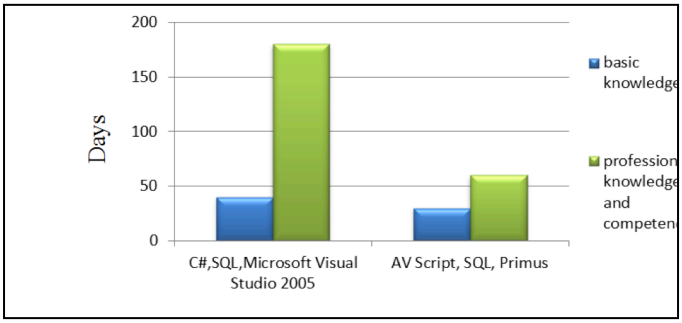


Figure 13. Comparison Chart

When we research and test platform for the control groups we identified areas which require further investigation. One of the identified areas was the task of generation automating the SQL queries. We want to rid the developers using our platform from having knowledge of SQL - they need to concentrate on the business structure of the data but not on relational algebra. This work demonstrates that a natural way of development of software engineering is raising the level of abstraction. Our results show that the combination paradigm of Meta - Design and application factories can produce very good results.

REFERENCES

[1] Standish_Group (1995). Chaos. THE STANDISH GROUP REPORT.

[2] Tao Yue, Lionel C. Briand and Yvan Labiche, "A systematic review of transformation approaches between user requirements and analysis models". //Springerlink.com

[3] Craig Larman, Victor R. Basili (June 2003). "Iterative and Incremental Development: A Brief History"

[4] Gafter, Neal (2006-11-05). "Reified Generkics for Java"

[5] Esteves, J., and Pastor, J., Enterprise Resource Planning Systems Research: An Annotated Bibliography, Communications of AIS, 7(8) pp. 2-54

[6] Cesar Gonzalez-Perez, Brian Henderson-Sellers Metamodelling for Software Engineering, Wiley, 2008, 219 pages

[7] Bieberstein et al., Service-Oriented Architecture (SOA) Compass: Business Value, Planning, and Enterprise Roadmap (The developerWorks Series) (Hardcover), IBM Press books, 2005.

[8] Youri I. Rogozov, Alexander S. Sviridov, Sergey A. Kutcherov, Wladimir Bodrov. Purpose-driven approach for flexible structure-independent database design. Proceedings of the Fifth International Conference on Software and Data Technologies, ICSOFT 2010, Volume 1, p.356-362

[9] Stroustrup, Bjarne (1997). "1". The C++ Programming Language (Third ed.).

[10] Krzysztof Czarnecki and Ulrich W. Eisenecker Generative Programming - Methods, Tools, and Applications Addison-Wesley, 2000, 864 pages

[11] Anneke Kleppe , Jos Warmer , Wim Bast MDA Explained: The Model Driven Architecture(TM): Practice and Promise Addison-Wesley Professional, 2003, 192 pages

[12] Palmer, S.R., & Felsing, J.M. (2002). A Practical Guide to Feature-Driven Development. Prentice Hall. (ISBN 0-13-067615-2)

[13] Steven Kelly, Juha-Pekka Tolvanen Domain-Specific Modeling Wiley-IEEE Computer Society Press,2008, 448 pages

[14] David Abrahams, Aleksey Gurtovoy C++ Template Metaprogramming: Concepts, Tools, and Techniques from Boost and Beyond. Addison-Wesley Professional , 2004, 4000 pages

[15] Jack Greenfield and Keith Short Software Factories: Assembling Applications with Patterns, Frameworks, Models & Tools Wiley, 2004, 500 pages

[16] Рогозов Ю.И., Актуальные проблемы и перспективные направления в области построения информационных систем и процессов: сборник статей международной научно-технической конференции. Таганрог: изд-во ТИ ЮФУ, 2010 г., стр. 9-15

[17] Ye, Y., Fischer, G.: Designing for Participation in Socio-Technical Software Systems. In: Stephanidis, C. (ed.) Proceedings of 4th International Conference on Universal Access in Human-Computer Interaction, Beijing, China, pp. 312–321. Springer, Heidelberg (2007)

[18] Schwabe, D., G. Rossi, et al. (1996). Systematic hypermedia application design with OOHDM. seventh ACM conference on Hypertext, Bethesda, Maryland, United States, ACM Press.

[19] Fratenali, P. and P. Paolini (1998). A conceptual model and a tool environment for developing more scalable and dynamic Web applications. EDBT 98, Valencia, Spain

[20] Schewe, K.-D., B. Thalheim, et al. (2004). Modelling and Stories in Web Information System. Information Systems Technology and its Applications (ISTA), Salt Lake Ciy, Utah, USA.

[21] Visual Rules – www.visual-rules.com/dynamic-applications.html

[22] Athula Ginige. Meta-design paradigm based approach for iterative rapid development of enterprise WEB applications. Proceedings of the Fifth International Conference on Software and Data Technologies, ICSOFT 2010, p.337-343

[23] Webratio – http://www.webratio.com

[24] Mendix – http://www.mendix.com/

[25] Yuri Rogozov, Wladimir Bodrow, META-APPROACH FOR CREATION OF OBJECT PROTOTYPES. Proceedings of ICERI2010 Conference. 15th-17th November 2010, Madrid, Spain.

[26] Grady Booch, Object-Oriented Analysis and Design with Applications, 2007.

[27] L.N. Lyadova et al., Implementation of distant learning portals based on CASE-technology METAS, // International Journal "Information Technologies and Knowledge", 2008. Т. 2. № 5. С. 489—495

[28] L.N. Lyadova, V.Lanin, Documents Management in Dynamically Adaptable Systems Based on Metamodelling // Proceedings of the Congress on Intelligence Systems and Technologies “AIS-IT’10”. Scientific publications in 4 volumes. Moscow: Physmathlit, 2010, Vol. 4. , Moscow, 2010.

[29] Fischer, G. and E. Giaccardi. Meta Design: A framework for the future of end user development. End User Development: Empowering People to flexibly Employ Advanced Information and Communication Technology. H. Lieberman, F. Paterno and V. Wulf, Springer. 9: 427-457