EXTENDABLE ARCHITECTURE FOR PROCESS SIMULATION SYSTEM WITH POSSIBILITY OF WORK WITH LARGE NUMBER OF EXTERNAL LIBRARIES AND SYSTEMS

Pavel Kyurkchiev

Abstract: The main idea of computer science is to develop, automate and facilitate mathematics and particularly mathematical calculations. The engine of this niche are the data simulation and information processing systems. One of the signs for classification of this system is the way they process the data, analytical or simulation. With the development of information technology the system requirements are being increased, which in turn implies the possibility of handling data from a high level of abstraction. Ever – greater challenge is creating architecture permitting the retention of the old functionality and at the same time providing an opportunity for integration with external services. Creating a scalable and versatile system that allows users the ability to work with several methods to describe the simulation is very long and cumbersome process. Simulation and processing system are handle a large number of libraries and external system enabling then to achieve their tasks. With combining the different design patterns, it is possible to achieve such architecture providing flexibility and freedom of expansion.

Keywords: Architecture, Model, Simulation, Systems, Extendable.

1. Introduction

Information technologies are part of modern people everyday life. Great part of these technologies are systems that handle specific mathematical analysis and modelling [1, 6, 7]. Even if people do not notice, technologies are everywhere

around them. Delicate systems operate our cars from the board computer, make our smartphones smart, run our laptops or even the calculus on our desk, every device branded as “smart” is construct around a computer. Each similar system is armed with a great number of external libraries that give her the ability to work with many different types of data.

This article describes an architectural template that provides the possibility for working with a number of external systems and libraries. It also contributes to the unobscured implementation of flexible mathematical models that can be further expanded. The presented system is a combination of selected templates and architectural design patterns which provide foundation for the creation of an expandable framework.

2. Description of analytical and model simulation systems

If we explore a complex group of applications that is responsible for strategical planning, logistics, marketing, managing projects, handling human resources and we analysis the instruments and techniques for modelling used by this group a certain pattern can be observed – the most used software for modelling up to date is MS Excel. Excel has many advantages to the user – ease of access, ability to operate on different platforms and interactive design. The foundation of this software is based on the analysis of different models. Its model of work is very simplified. It includes electronic tables, divided in cells that can be filled with different mathematical formulas and other cells that calculate the results. Although their flexible and simplified method of work, such types of software can't complete all sorts of tasks. To be able to do so, they need to be able to expand their functionality with a variety of add-ons. Great number of problematic classes are handled by dynamic systems like:

- nonlinear behaviour;
- time dependencies;
- memory.

To solve the problems of these classes specific systems are needed [1, 2, 7]. These systems for simulation models are using more complex models. Their final aim is receiving similar data on the basis of a complicated model. The models used in such systems are graphically build from internal libraries that give the chance of working with a variety of models.

Both types of applications have a similar working method. Information (models and mathematical models) is collected from user interface and is loaded in the context of the application. Then there a decision is taken on how to process the

collected data. After that it is send to a specific library. Before being received the data is being processed by the module for presentation of data that has been used which in most cases consists of a group of several methods. After information processing has started, the responsible methods return middle data for specific methods. Processed data is being presented in an easy "view-models" and after that is visualized to the user in an easy to use format.

To create such an architecture as well as its ability to expand and further develop, a specific separation of not only the logical parts but a separation of physical parts in different projects is required. Such separation would improve not only the workings of the systems, but it would give the foundation possibility of expanding with additional modules in the future. It is a good practice that each calculus library and method are separated in different projects. This would ensure the isolation of different logical parts and preserve their state clean and easy for maintain. All these mathematical libraries can be requested later, but to be able to accomplish this they have to inherit the same interface that describes the general range of actions they are created to support.

All the libraries have to situate only one way of access, so that the difficulty of support is reduced. This will be accomplished by realised by implementing a "facade pattern" [3], that works as a foundation for access to different logics. The object that is responsible for that has to be global and must be able to coordinate the actions in the whole system. Because of these characteristics the object would be resource hungry for the system. For this point it has to be initialized only once and if possible in the last stage of life of the system. It also has to be destroyed after finishing its tasks. To respond to all these characteristics, the object has to fill into a "singleton pattern" [4].

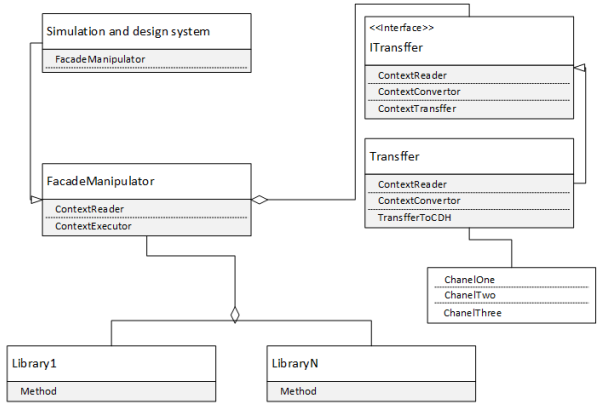

Figure 1. Architecture of extendable simulation system

Figure 1 presents the architecture that contains all components described above with one additional module. This module is called “transfer” and is responsible for the communication with other external systems and libraries. Increase in the number of users directly leads to the increase of load in a system. This is particularly valid for architectures that offer the availability for external communication with other services. One of the major difficulties such architectures meet is the increased number of queries towards them. Like very module for communication, this one would also experience great trouble if a big number of customers use the system at the same time. To achieve minimal loading time when transferring data, the creation of three different channels is required. One, responsible for the input calls, one for output calls and one that is used to store data. The communication channels represent AMQP (Advanced Message Queuing Protocol) queue of messages. The messages are objects encoded under base64. The input channel or “ChannelOne” receives messages under the form of processed data, which is presented in a suitable format before being visualized. The output channel or “ChannelTwo” will send messages as mathematical models for processing. The channel for storing data or “ChannelThree” will keep data or receive finished visualizations of stored objects.

3. Simple Implementation of the architecture

The presented architecture for expandable system for simulation and modelling of processes can be used as a foundation of other applications with great ease and only minor code changes.

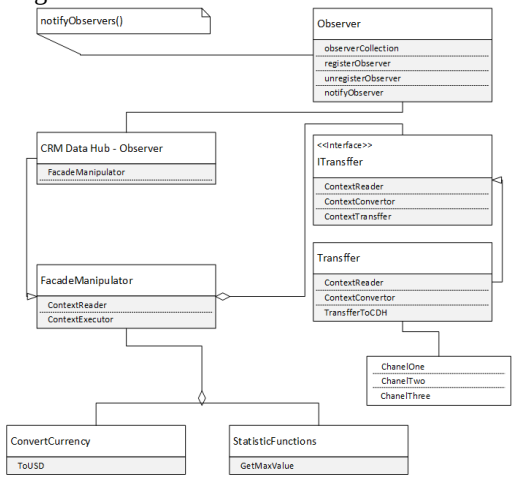

Figure 2. Architecture of CRM Data Hub

We will now explore as an example a system of type CRM Data Hub. It is a system that is responsible for the communication of a great number of subsystems and libraries. Data handlers of such type receive queries for completion. After receiving the queries, information is being processed and a specific external module is being chosen for completion of the described tasks. After that the context of the query is being sent towards the specific subsystem for completion. After completing the query, the data centre returns the data and closes the information channel.

In most cases situating the needs of such types of systems is handled by n-tier architecture that is combined with simplified model for communication with external services. Architecture of such kind gives the opportunity for smooth development and support. However, the possibilities for expansion with external services and libraries without creating disturbance in the structure are limited. When a great number of users are involved the possibility of overload in communication channels is a risk.

With a slight change of the model and the addition of a module for data transformation the architecture would satisfy completely all the needs of such class of applications. Other than the easy adaptation to the requirements of such similar type of systems, the architecture would enhance the performance when experiencing huge load.

The bank industry, unlike other industries, is experiencing constant competition when it comes to offering new types of services. The old systems used to support the banking industry are in a state of complete expansion. This expansion leads to the need of upgrades incorporating external services and libraries. The whole process of continuous development cannot suppress the ever increasing load that is being handled by certain modules, which leads to slowdown in the system. Most systems of this kind use Domain-driven design architectures. DDD is a methodology that provides clean and useful project design. However the possibilities for expansion with external means are very limited. For improving the work of such systems and simultaneously decreasing the load between different modules the system described above can be used. It is capable of enhancing the possibilities of expansion and further lowering and distributing the system load.

4. Conclusion

The creation of such architecture that holds the ability to work with a great variety of libraries like [5] is one of the most common cases of solving the problem of the rising customer needs. In a certain way establishment of such architecture is done with the idea of solving a concrete specific problem. Such architecture, with

only a slight change of logic can be used in any possible direction. To make this happen, the architecture needs to be able to work not only with internal but mostly with external systems. This would allow a huge advance in the future of developing applications.

The distributed architectural system for modelling and simulation provides the developers the ability to easily expand and upgrade. It would allow easy support and integration of outer calculation systems. It could make a project more readable as well as easy to expand from an external team of software developers.

Acknowledgments

The research is partially supported by the Fund NPD, Plovdiv University, No. IT15-FMIIT-004.

References

[1] Kyurkchiev P., Architecture of web based system for symbol programming a real process, *Scientific Conference "Innovative ICT: Research, Development and Application in Business and Education"*, 11–12 November 2015, Hisar, 2015, 55–60.

[2] Stella simulation software, http://www.iseesystems.com

[3] Gamma, E., R. Helm, R. Johnson, J. Vlissides, *Design Patterns: Elements of Reusable Object-Oriented Software*, Addison Wesley, 1994.

[4] Microsoft Corporation, Singleton, https://msdn.microsoft.com/en-us/library/ff650849.aspx

[5] Valchanov, N., T. Terzieva, V. Shkurtov, A. Iliev, Architecture of extensible computation driven systems, Mathematics and mathematical education, *Proc. of 39th spring conference of Union of Bulgarian Mathematicians*, 06–10 April 2010, Albena, Bulgaria, 2010, 207–211.

[6] Valchanov, N., A. Iliev, Implementation of graphical simulation environment for mathematical models, Fundamental and Complementary Science, "Mircea cel Batran" Naval Academy Scientific Bulletin, Constanta, Romania, Volume XIV (2), 2011, 222–228.

[7] Valchanov, N., P. Petkova, A. Iliev, Integration of computational library for simulation mathematical models in web-based system for courses management, *Proceedings of the Jubilee National Conference with international participation "Tradition, directions, challenges"*, Paisii Hilendarski University

of Plovdiv – branch Smolyan, 50 years Scientific and Educational Institution in the Rhodope Mountains, 19–21.10.2012, 2012, 89–94.

[8] Iliev, A., G. Hristozov, T. Terzieva, Software environment for dynamic models presentation with statistics, *National Conference "Education in the Information Society"*, Plovdiv, 2006, 38–43.

Faculty of Mathematics and Informatics
Paisii Hilendarski University of Plovdiv
236 Bulgaria Blvd., 4003 Plovdiv, Bulgaria
e-mail: pavkyu@gmail.com

АРХИТЕКТУРА НА РАЗШИРЯЕМА СИСТЕМА ЗА СИМУЛАЦИЯ НА ПРОЦЕСИ С ВЪЗМОЖНОСТ ЗА РАБОТА С ГОЛЯМ БРОЙ ВЪНШНИ БИБЛИОТЕКИ И ПРИЛОЖЕНИЯ

Павел Кюркчиев

Резюме: Основната област на компютърните науки е да развива, автоматизира и улеснява математиката и в частност математическите изчисления. Двигателят на тази ниша са системите за моделиране и обработка на данни. Един от признаците за класифициране на подобен род системи е начинът на обработка на данните, аналитично или симулационно. С развитието на информационните технологии се увеличават и изискванията към тези системи, което, от своя страна, включва и възможност за боравене с данни от по-високо ниво на абстракция. Все по-голямо предизвикателство представлява създаването на архитектура, допускаща запазването на старата функционалност и предоставяща в същото време възможност за интеграция с външни услуги. Създаването на една разширяема и многофункционална система, предоставяща на потребителите възможност за работа с няколко метода за описване на симулации, е много дълъг и тежък процес. Системите за симулиране и обработка на процеси боравят с голям брой библиотеки и външни системи, даващи им възможност за решаване на техните задачи. Чрез комбиниране на различни шаблони за дизайн е възможно постигането на архитектура, даваща гъвкавост и свобода на разширение.

Ключови думи: Архитектура, Модели, Симулации, Системи, Разширяеми.