Transforming an Enterprise E-Health System from Process Oriented to Model Driven Architecture

Blagoj Atanasovski[*], Miloš Bogdanović[**], Goran Velinov[&], Leonid Stoimenov[**], Dragan Sahpaski[*], Irena Skrceska[!], Margita Kon-Popovska[&], Dragan Janković[**], Boro Jakimovski[&]

[*] Sorsix International, Skopje, Macedonia
[**] University of Nis, Faculty of Electronic Engineering, Niš, Serbia
[&] Faculty of Computer Science and Eng., Ss. Cyril and Methodius University in Skopje
[!] Faculty of Informatics, European University, Skopje
blagoj.atanasovski@sorsix.com milos.bogdanovic@elfak.ni.ac.rs goran.velinov@finki.ukim.mk
leonid.stoimenov@elfak.ni.ac.rs dragan.sahpaski@sorsix.com irena@eurm.edu.mk
margita.kon-popovska@finki.ukim.mk dragan.jankovic@elfak.ni.ac.rs boro.jakimovski@finki.ukim.mk

***Abstract*— The e-health platform Pinga is an integrated health information system represents a central electronic system, in which all medical and health related information about patients, health workers, facilities, documents, and procedures is stored and processed. The platform is implemented in Serbia as MojDoktor (My Doctor), and Macedonia as MojTermin (My Appointment). The architecture of the system was designed to allow for process oriented development with agile methodologies. This methodology allowed for fast deployment and adoption but a change in the architecture to a more formal approach is required to assure its extensibility, soundness, interoperability and standardization. In this paper we propose a transformation of the design framework from Process Oriented to a Model-Driven Architecture.**

I. INTRODUCTION

The health platform Pinga represents a central electronic system, in which all medical and health related information is stored and processed on-line. The system allows integration with existing and future systems used by hospitals, clinics, the Ministry of Health, and other public health institutions [1]. Pinga can be classified as a system that integrates several components: an EHR (Electronic Health Record), Electronic Prescriptions, Electronic Referrals, Hospital Stay and Surgeries Information System, Laboratory Information System and a Radiology Information System.

Pinga keeps the data in a centralized system. The various health organizations integrated with the system have autonomous heterogeneous systems who keep data locally, but a part of that data is written in the central database in the process of integration with Pinga. The platform was implemented in Macedonia as MojTermin (MyAppointment). Pinga integrates all existing electronic information systems used in public and private health organizations that have an agreement with the Public Health Fund. Integration is done through a precisely defined API accessible through web services with a HTTPS connection. The access to the web services can be public or protected. Public services provide access to public data about the health organizations and specialties, doctors, resources, available timeslots etc. Protected services provide access to patient data, referral, exam and prescription details etc. Protected services require authentication. For the functionalities not covered by any existing system a web interface provides authorized access. The organizations, that did not use any kind of information system before, use the web interface. A complete high-level architecture of the system is given in Figure 1.

From a data collection point of view the system continuously creates new and updates existing records. In Serbia, the system creates records for over 250000 prescriptions, 80000 referrals and 190000 exams, each working day. The system handles over 4800000 requests through the APIs.

The authors in [8] recognize that long term strategy targets for a national e-health system should be standardization on a national level, progressing to full interoperability on a European level. The current architecture design enabled an implementation of the processes and to create a functional system. This allowed MojDoktor to be built and functional in 9 months.

A change to a more formal approach is required in order to transform the platform to be interoperable and standardized, and do so in a secure and sound way. The current implementation uses standardized international codes for doctor specialties, diagnosis (ICD10), drug generics (ATC). And the next step is to make the system compliant with EHR standards like HL7 or OpenEHR.

II. RELATED WORK

When building an integrated health information system, the potential to incorrectly implement the requirements is very high, due to the complexity of the processes in health care. A change in any of the requirements introduces a need for a correction of the implemented system, and can lead to regression.

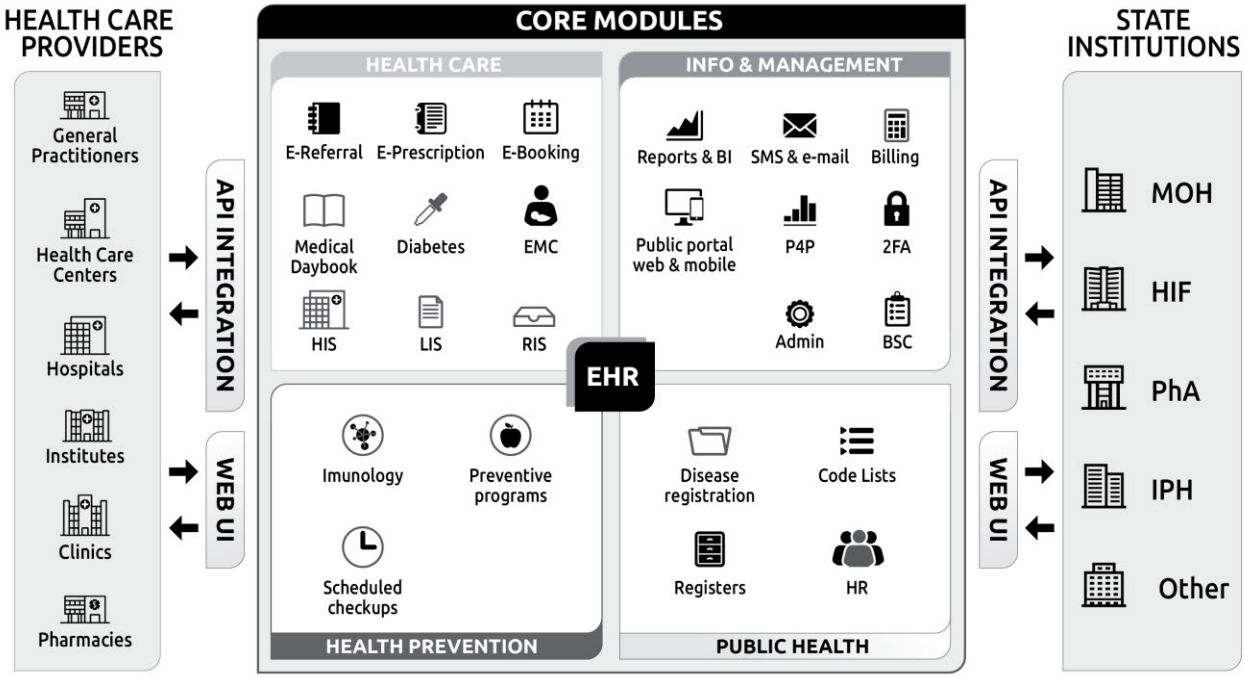

Figure 1. High Level Architecture of the Pinga platform

This can be avoided if the system is established using a well-defined framework/architecture. Model Driven Architecture (MDA) has proven to be one of the best choices when requirements changes are expected [11]. MDA focuses on the consequent utilization of diagrammatic models, to describe the contextual situation, which is successively transferred through different model layers into a technical model [2]. MDA specifies three layers:

1. Computation Independent Model (CIM), the business model;
2. Platform Independent Model (PIM), software engineering model without technology aspects;
3. Platform Specific Model (PSM), technology-related aspects of the target platform.

Business models extracted from domain experts are transformed through the layers, to derive the software. Rules defined for transformation between the layers allow changes in the business process to be easily transformed into correct code changes [3, 4].

Several attempts in using MDA for e-health systems have been published. The authors in [5] attempt to create a specific MDA for the healthcare domain. They needed a mechanism that fosters a sustainable tele-health platform in terms of a methodology that provides an efficient way to deploy new artifacts on the platform and ensure these artifacts are compliant to the platform properties. They show how and MDA approach can be used in order to build a methodological fundament for a systematic creation of an application system. Their use-case is an application for IT based workflow support for an interdisciplinary stroke care project.

In [6], with MDA, tools are developed for data collection to allow non-informatics experts to model requirements in their local domain. They recognized a problem in the process of implementing local improvement initiatives in healthcare systems, so they developed a Model - Driven framework and implementation that allows local teams in medical organizations to specify the metrics to track their performance during an intervention, together with data points to calculate these metrics. Based on the metric specification the software generates appropriate data collection pages.

In [7] MDA augmented with formal validation was used to develop m-health components promising portability, interoperability, platform independence and domain specificity. The authors are developing systems based on inter-communicating devices worn on the body (Body Area Networks) that provide an integrated set of personalized health-related services to the user. This mobile healthcare application feeds captured data into a healthcare provider's enterprise information system.

In their previous work [13] the same authors propose an extension of the model-driven approach where formal methods are used to support the process of modelling and model transformation. The semi-formal modeling with UML is verified by using tools such as SPIN [17] for model checking and TORX [18] model testing. They present [14], [15] and [16] as possible formal approaches to the transformation.

In this paper we present a transformation from Process Oriented design to a Model-Driven architecture for the Pinga platform.

III. METHODOLOGY

By analysis of the requirements of health institutions, several functional modules have been discovered and implemented in Pinga (Public Portal [9] [10], Electronic Health Record, Reporting and BI, Integration etc.). Different modules have disjoint sets of processes; each can depend on one or more processes from other modules.

The processes have already been defined from a business perspective, out of the functional requirements of our client. We documented them using BPMN, and those definitions represent the CIM layer of our architecture.

For each module we define a process-domain-library (process-meta-model). This domain library

enforces semantic restrictions on the processes and allows us to develop tools that guarantee the soundness of the model (safe invocation, safe termination). This domain-library is part of the PIM of our architecture. Using class diagrams, models for specific processes can be created that conform to the domain-library.

A change in the definition of a process, or a new process, will update the PIM and run verification. Problems can be discovered before any code is written. When a new standard is implemented a PIM→PIM transformation can be applied to obtain a design that will generate PSM models compliant with the standard.

IV. Solution

A state's healthcare is an enormous collection of processes, rules, people and resources backed by legislative. To create a holistic model is impossible in one iteration. In each iteration focus is given to certain processes or improvements. We present an example of how we model the Electronic Health Record (EHR) module. By using workshops with domain-experts, models using BPMN or UML Activity Diagrams of the processes are created.

The domain-library for the EHR defines Resources, Documents and Processes. Resources and Documents have an identifying type. A resource can consume a set of documents of a certain type, and produce a set of documents. A process can be represented as a directed graph with a starting vertex and an ending vertex. Its execution is a graph traversal. Each vertex can be a sub process (subgraph) or a resource.

A process is invoked by a set of documents and a set of available resources. The results from one vertex are given to the adjacent vertexes as input. The ending vertex produces the output of the process. Only vertexes with satisfied input requirements can be executed. Each process can be validated with the following conditions being fulfilled: process can start and end, no loops exist or no unwanted scenarios can happen.

Figure 2 presents a part of the domain library using a UML class diagram.

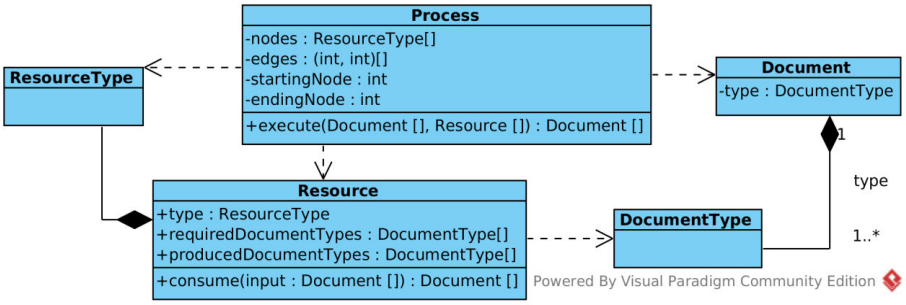

Figure 2. High Level Architecture of the Pinga platform

V. Conclusion

The Pinga platform is a working system, was implemented in reasonable times, and is constantly upgraded with new features. The current implementation practice does not hinder the usage or improvement of the system. However, problems can arise in the future if the current implementation technology becomes outdated or obsolete.

The lack of a formally defined Platform Independent Model increases the risk the platform becoming stuck in a state of inertia that will require a complete overwrite. Additionally our goals for creating an architecture that will support standardization and interoperability are complemented by the nature of MDA itself.

By having an MDA design, new implementation infrastructure can be integrated and supported, existing functionality can be more rapidly migrated to new platforms and environments, and quality can be gained by the formal separation of concerns, plus the consistency and reliability of artifacts produced contribute to the quality of the overall system. And most importantly, integration with external systems is significantly facilitated [12].

References

[1] MojDoktor technical documentation, internal document property of Sorsix International

[2] Object Management Group: OMG Unified Modeling LanguageTM (OMG UML), Super-structure, Ver. 2.4.1, http://www.omg.org/spec/UML/2.4.1/Superstructure/PDF/ (2011).

[3] Soley, R., Group, O.S.S.: Model driven architecture (2000). OMG white paper. 308.

[4] Mellor, S.J.: MDA distilled: principles of model-driven architecture. Addison-Wesley Professional (2004)

[5] Schlieter, Hannes, et al. "Towards Model Driven Architecture in Health Care Information System Development." Wirtschaftsinformatik. 2015.

[6] Curcin, Vasa, et al. "Model-driven approach to data collection and reporting for quality improvement." Journal of biomedical informatics 52 (2014): 151-162.

[7] Jones, Val, Arend Rensink, and Ed Brinksma. "Modelling mobile health systems: an application ofaugmented MDA for the extended healthcare enterprise." EDOC Enterprise Computing Conference, 2005 Ninth IEEE International. IEEE, 2005.

[8] Velinov, Goran, et al. "EHR System MojTermin: Implementation and Initial Data Analysis." MIE.2015.

[9] MojTermin Public Portal, http://mojtermin.mk. Accessed 30 January 2017.

[10] MojDoktor Public Portal, https://www.mojdoktor.gov.rs. Accessed 30 January 2017.

[11] Loniewski, Grzegorz, Emilio Insfran, and Silvia Abrahão. "A systematic review of the use of requirements engineering techniques in model-driven development." International Conference on Model Driven Engineering Languages and Systems. Springer Berlin Heidelberg, 2010.

[12] Truyen, Frank. "The fast guide to model driven architecture the basics of model driven architecture." Cephas Consulting Corp (2006).

[13] Jones, V. M., et al. "A formal MDA approach for mobile health systems." (2004).

[14] Correctness Preserving Transformations for the Early Phases of Software Development; T.Bolognesi, D. De Frutos, R. Langerak, D. Latella.I,IN Bolognesi T, van de Lagemaat J and Vissers C.A. (ed), LOTOSphere: Software Development with LOTOS, pp. 348-368, Kluwer Academic Publishers, 1995.

[15] Jones V (1995). Realization of CCR in C, In Bolognesi T, van de Lagemaat J and Vissers C.A. (ed), LOTOSphere: Software Development with LOTOS, pp. 348-368, Kluwer Academic Publishers, 1995.

[16] Jones VM (1997) Engineering an implementation of the OSI CCR Protocol using the information systems engineering techniques of formal specification and program transformation. University of Twente, Centre for Telematics and Information Technology Technical Report series no. 97-19. ISSN 1381-3625.

[17] G.J. Holzmann, (2003) The Spin Model Checker: Primer and Reference Manual, AddisonWesley, ISBN 0-321-22862-6

[18] J. Tretmans and A. Belinfante. Automatic testing with formal methods. In EuroSTAR'99: 7th European Int. Conference on Software Testing, Analysis & Review, Barcelona, Spain, November 8-12, 1999. EuroStar Conferences, Galway, Ireland.