

A VISUAL MODELLING LANGUAGE TO BUILD HEALTHCARE APPLICATIONS BASED ON A REUSABLE SOFTWARE ARCHITECTURE

UMA LINGUAGEM DE MODELAGEM VISUAL PARA CRIAÇÃO DE APLICAÇÕES DE SAÚDE COM BASE EM UMA ARQUITETURA DE SOFTWARE REUTILIZÁVEL

Marcio Alexandre Pereira da SILVA¹, Valeria Cesário TIMES²,
André Magno Costa de ARAÚJO³, Paulo Caetano da SILVA⁴

1 Federal University of Pernambuco, Brazil. Is a Ph.D. student in computer science in the Informatics Center, Federal University of Pernambuco, Brazil. His research interests include health information systems, adaptive software, and distributed, decoupled and reusable software architectures. He is a professor in a private university in Sergipe, and has professionally worked on software development as software engineer. Contact him at maps3@cin.ufpe.br.

2 Federal University of Pernambuco, Brazil. Is an associate professor in the Informatics Center, Federal University of Pernambuco, Brazil. Her research interests include data warehouses, geographic information systems, health information systems, financial information systems, autonomous databases, data privacy, trajectory analysis, cloud databases, and advanced database applications. Times received a Ph.D. in computer science from the University of Leeds, United Kingdom, and participated in a postdoctoral program at the University of Ottawa, Ontario, Canada. Contact her at vct@cin.ufpe.br.

3 Federal University of Alagoas, Brazil. Is an assistant professor at Federal University of Alagoas, Brazil. He has taught software engineering and database design in many private institutions in Brazil and worked on software development projects as software engineer. His research interests include software architecture, data modeling, big data and software development. He is author and co-author of many articles published in IEEE and Springer which were presented as lectures in notable American and European conferences. Araújo received a Ph.D. in computer science from the Federal University of Pernambuco in 2018. Contact him at amcaraujo@gmail.com.

4 Salvador University, Brazil. Is a professor in the master program in systems and computing at He is also an analyst at the Central Bank of Brazil. His research interests include financial and auditing information system, health information system, blockchain, XBRL, SOA, advanced database applications and XML. Silva received a Ph.D. in computer science from the Federal University of Pernambuco, and participated in a postdoctoral program at the Rutgers University, New Jersey, USA. He is author and co-author of many books and chapters, and articles published in IEEE, IARIA, and ACM. Contact him at paulo.caetano@unifacs.br.

ABSTRACT. Reusability is an essential attribute in a software lifecycle, as it improves the usefulness of applications and reduces maintenance and development costs. However, legacy health applications use software architecture models that hinder the reusability of its components. This paper proposes the Health Software Modelling Language (HSML), which enables the Model-Driven Development (MDD) of healthcare software using acknowledged health standards and a reusable software architecture. Five metamodels and constructors are specified to build the HSML. A modelling tool is built, which allows software developers to model Health Information

Systems. After that, the model is transformed into computer codes. To validate the solution presented herein, a legacy health application was redesigned using the HSML tool in the real-life scenario of a Brazilian healthcare institution. As a result, HSML increases the reusability of the healthcare software and allows the migration of legacy healthcare software to a new software whose environment offers more reusability.

Keywords: Reusability. Health Information System. Microservices. OpenEHR Archetype. HL7 Fast Healthcare Interoperability Resources (FHIR).

RESUMO: A reutilização é um atributo essencial no ciclo de vida de um software, pois melhora a utilidade dos aplicativos e reduz os custos de manutenção e desenvolvimento. No entanto, os aplicativos legados de saúde usam modelos de arquitetura de software que impedem a reutilização de seus componentes. Este artigo propõe a HSML (Health Software Modeling Language), a qual permite o desenvolvimento de software orientado a modelos (Model-Driven Development) e de padrões de saúde reconhecidos. Cinco metamodelos e construtores foram especificados para construir a HSML. Uma ferramenta de modelagem foi desenvolvida, a qual permite que os desenvolvedores de software modelem os Sistemas de Informações de Saúde (SIS) com base em HSML. Depois disso, o modelo criado pela ferramenta é transformado em códigos de computador. Para validar a solução aqui apresentada, um aplicativo de saúde legado de uma instituição de saúde brasileira foi reprojeto usando a HSML. Como resultado, a HSML aumentou a capacidade de reutilização de software de saúde, e permitiu a migração de um software legado de saúde para um novo software cujo ambiente oferece mais capacidade de reutilização.

Palavras-chave: Reutilização. Sistema de Informação em Saúde. Microserviços. Arquetipo OpenEHR. Recursos Rápidos de Interoperabilidade em Saúde HL7 (FHIR).

1. INTRODUCTION

Software Engineering uses systematic methodologies to develop any software (PRESSMAN & MAXIM, 2019). An important phase in software development is the construction of software artifacts, which is performed by software developers through coding using programming languages or modeling software components

through visual modelling languages. The latter is called Model-Driven Development (MDD) (BRAMBILLA et al. 2017).

One of the tasks in building software artifacts is to specify how the components from a software architecture interact with each other. A software architecture design consists in defining its components, their external properties and their relationships with other components (BASS

et al. 2012). There is a consensus that software architecture plays a central role in software development and its lifecycle phases, after delivering the software (CHA et al. 2016).

The component definition of a software architecture using MDD is an approach widely found in Industry and Academia, which facilitates software development (and the definition of its software architecture), because it encapsulates the complex concepts into a graphical representation. Thus, software developers do not deal with the technical difficulties of the standards which are inherent to information systems (RADEMACHER et al. 2019; ISO, 2020). In the healthcare software domain, for instance, many studies encourage the use of MDD, since it can facilitate the application of Health standards in the development of Health Information System (HIS) (CHRISTENSEN & ELLINGSEN, 2016; ELLOUZZEA et al. 2017).

Concomitantly, there is a challenge in the component definition of the software architecture: How to specify its components in a way that its execution can be shared as a resource for more than one software, increasing the reusability of these components? One solution is to create software architectures based on “reusable components”, which means that all these components are not interconnected and thus, independent, and can be used by different sets of software (SILVA et al. 2019).

Due to the need for improvement and evolution in the way software architectures are designed, a new approach has emerged in software engineering. Known as microservices, this approach decomposes applications in small components whose tasks are basic functions. Each component becomes decoupled and operates as a service (a small service or microservice), which

can be implemented and deployed independently (KAKIVAYA et al. 2018). Because of its decoupled nature, this microservice-based software architecture holds components that can be shared by more than one software. Some studies suggest the use of microservices in HIS to increase the reusability in the software architecture (SILVA et al. 2019).

In the Healthcare domain, reusability is widely debated and encouraged to create more standardized HIS worldwide. For instance, there is efforts in the healthcare community to use standards to format healthcare data, e.g., using Archetype (OPENEHR FOUNDATION, 2020), and to exchange such data between HIS, e.g., using HL7 Fast Health Interoperability Resource (FHIR) (HL7 INTERNATIONAL, 2020). An archetype models the healthcare data attributes and provides resources that allow its reusability among HIS (OpenEHR FOUNDATION, 2020), and FHIR is a reusable standard for healthcare data exchange in any HIS (HL7 INTERNATIONAL, 2020).

Some state-of-the-art studies propose the use of MDD to build HIS, including archetypes solutions (CHRISTENSEN & ELLINGSEN, 2016; ELLOUZZEA et al. 2017; OMG, 2018). Other studies propose the dynamic generation of software architecture components (Think!EHR Platform, 2017; GOMES et al. 2018; SILVA et al. 2019; ARAÚJO et al. 2020). These studies bring important contributions to state-of-art and state-of-practice in the healthcare domain. However, there are some limitations in the use of MDD for building reusable software architecture when the HIS uses archetypes for its health data and HL7 FHIR for the interaction between its components. For example, when components of the generated software architecture cannot be used by different

HIS (i.e., there is no reusability in the software architecture). Considering the studies that propose the dynamic component generation, developers that use this solution can use the MDD approach for building the HIS.

This paper proposes the Health Software Modelling Language (HSML) which provides resources for the MDD of HIS with a reusable software architecture using widely adopted health standards such as archetype and HL7 FHIR. To build the HSML, five metamodels and constructors are specified, along with their relationship rules. A modelling environment - with its transformation in computer codes - is built based on the Sirius platform and MOF M2T language (ECLIPSE FOUNDATION, 2020), Template4EHR tool (ARAÚJO et al. 2020), and Microservice4EHR tool (SILVA et al. 2019).

To validate the HSML, two real-world institutions were used to update their HIS. One of them executes the “risk-rated patient care” protocol (SERVIN et al. 2020) in Brazilian public hospitals. The other one executes the “blood donation” protocol (Health Brazil Ministry, 2002) in Brazilian blood donation centers. In this assessment, two legacy software are presented, subsequently redesigned and updated using HSML. To measure the reusability of the new HIS built from HSML, this paper extends a formal modeling based on Graph theory (MOHR, 2014). The main results discussed suggest that health software development with HSML increases reusability at the software architecture level.

This paper is organized as follows; Section II presents the basic concepts and the related works. A detailed description of HSML is given in Section III. Assessment and results are shown in Section IV and V respectively. Conclusion and

future work are presented in Section VI.

2. BACKGROUND AND RELATED WORKS

This section contextualizes both healthcare specifications: OpenEHR archetypes (Section 2.1) and HL7 FHIR (Section 2.2), conceptualizes reusable and non-reusable architectures (Section 2.2), and provides an analysis of related works identified both in Academia and Industry (Section 2.3).

2.1. OpenEHR Archetype

The OpenEHR Foundation – an international organization composed of several Health domain specialists - has specified archetype as a standard for the computational modeling of the Electronic Health Record (EHR) (OpenEHR FOUNDATION, 2020).

An archetype is based on a dual model to create a unique EHR, preserving the history and evolution of patient clinical data, whose exchange and reusability can be applied in other healthcare domains (ARAÚJO et al. 2020). The dual model architecture allows the separation of (i) clinical and demographic properties, and (ii) standards and terminologies that give a semantic meaning to healthcare data. The first level of the Dual model regards the components of programming language, the information exchange language, and all other components related to software development. The second level is represented by archetypes and templates (MONER et al. 2018).

In an archetype, attributes specification is performed through constructors whose data input are called generic data structures, which allow the modeling of EHR heterogeneity through

types such as: ITEM_SINGLE, ITEM_LIST, ITEM_TREE and ITEM_TABLE (JOFREY et al. 2012).

ITEM_SINGLE models a single data attribute (e.g., patient gender), while ITEM_LIST gathers a set of attributes into a list (e.g., patient address composed by street name, number and zip code). ITEM_TREE specifies a data hierarchical structure which is logically represented as a tree (e.g., physical and neurological evaluation of a patient). Finally, ITEM_TABLE models the data elements through lines for element definitions and columns for information values (e.g., a clinical report in a table format, the clinical exams are shown in lines, and their respective values are presented in columns).

Each attribute from any data structure is characterized by a data type and may also have a set of domain constraints and associated terminologies. Terminologies give a semantic meaning to clinical data and can be represented by Health technical terms or textual information defined by a domain specialist.

Currently, the archetype standard has been adopted worldwide in both Academia and Industry (OpenEHR FOUNDATION, 2020). Some state-of-art and state-of-practice tools propose the use of a set of archetypes in order to dynamically build software artefacts, e.g., Graphical User Interface (GUI), data schema, and Application Programming Interfaces (APIs) (Think!EHR PLATFORM, 2017; GOMES et al. 2018; ARAÚJO et al. 2020). State-of-art studies suggest that building software from a set of archetypes is a good practice for maintaining healthcare software in compliance with international health standards. Following this perspective, HSML is specified to generate a reusable software architecture from a set of archetypes.

2.2. HL7 FHIR

The Health Level Seven (HL7) maintains the FHIR standard. FHIR is a platform specification that defines a set of capabilities that use healthcare processing in many different contexts. As a primary component, FHIR features the Application Programming Interfaces (APIs), which is a collection of well-defined interfaces for interoperating between two applications through the Internet.

Although not required, the FHIR specification suggests the use of Representational State Transfer (REST) interfaces for API implementation. REST is an interaction protocol to establish a fast communication between information systems. With this suggestion, FHIR standardizes the communication among API-based HIS.

FHIR has been widely adopted in the HIS of several countries (SEMENOV et al. 2018). By leveraging this standard, HSML tool has also been specified to dynamically generate a reusable software architecture whose interaction between its components is based on FHIR API standards.

2.3. Reusable and Non-Reusable Architectures

For decades, the use of software architectures whose components are interconnected and interdependent (i.e., non-reusable) is referred to as Monolithic Architecture. In the Monolithic Architecture shown in Figure 1, components in the Business Logic layer do not perform tasks independently and cannot be shared by other software. In fact, the monolithic

architecture components share the resources of the same machine (computer) and mandatorily executes their tasks for only one software. For instance, in monolithic applications (i.e., applications built on Monolithic Architecture), it is not possible to operate the Business Logic layer components of a software with those of another software (DRAGONI et al. 2017). To mitigate this lack of reusability, studies on the use of small services (or microservice) built into Business Logic layers have been carried out. This means that the Business Logic layer is made of a set of microservices which perform tasks independently and can be shared by other sets of software (KAKIVAYA et al. 2018; SILVA et al. 2019).

The Microservice architecture depicted in Figure 1 is an approach to develop a single software composed of a set of small services with very specific and bounded tasks (i.e., microservices). Each microservice models either components from the Business Logic or Data Access layers, and does not share the resources of the same machine (computer). Each microservice has its own database. The Microservice is also able to interoperate with different sets of microservices in order to reach the business targets of any software. The access to each microservice is performed through the Internet or a local area network (LARRUCEA et al. 2018).

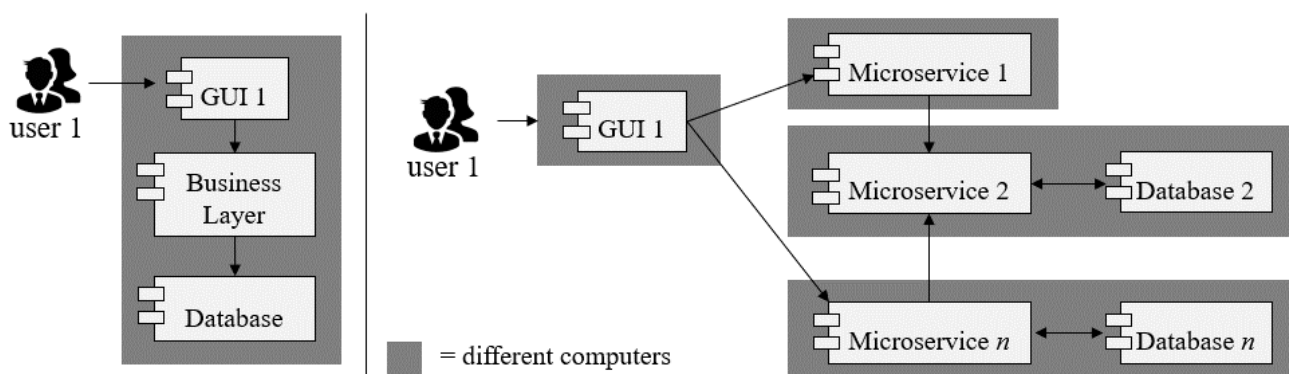


Figure 1. Example of Monolithic and Microservice architectures.

The adoption of microservice architecture promotes software adaptation to new technological demands (e.g., cloud, big data) (KAKIVAYA et al. 2018), as well as non-functional requirements (e.g., maintainability, scalability, decouplability) (DRAGONI et al. 2017, LARRUCEA et al. 2018). Because of these features, in this study we have specified the use of microservices in the DHS. Some works propose tools that build

software architectures from a set of archetypes, which are shown below.

2.4. Related Works

Works that provide healthcare solutions through model-driven methodology have been found from 2002 to the current days.

In the last decade, many studies have

shown the use of the Unified Modelling Language (UML) as a tool for modelling Health software (AGGARWAL, 2002; RAISTRICK, 2004). Other studies also show UML as a limited tool, unable to model all HIS requirements (VASILAKIS et al. 2008). Conversely, in the face of UML limitations, there was a rise of research and solutions for modelling Health software requirements with other resources, e.g., domain-specific modelling language (DSML), as shown below.

Khambati et al. (2008) have developed two Domain-Specific Modelling Language (DSML), one of them models health data, the other one is used for mapping such data at the application level. Rad et al. (2009) model the clinical process and web services with both Business Process Execution Language (BPEL) and Web Services Choreography Description Language (WS-CDL). Martínez-Costa et al. (2009) built a MDSE tool that models archetypes and converts them into Ontology Web Language (OWL) format. Chang (2011) extends the UML for health data modeling and management into relational databases. Gomes et al. (2012) propose a DSML to model archetypes and to interact with Acme language for describing health software architecture.

Braun et al. (2015) created the Business Process Model and Notation for Clinical Process (BPMN4CP), a BPMN extension that allows the modelling of the clinical process. Reichherzer et al. (2015) have developed a DSML in order to model Health data from the Military Data Repository and to interact with UML Use Case diagrams. Telang et al. (2015) show the advantage of using Comma notation, in detriment of the HL7 Messaging, for clinical process modeling. Kannan et al. (2016) propose an agile methodology based on UML and a DSML, for building models for health data and

the clinical process. Demski et al. (2016) propose a DSML for modelling healthcare data based on OpenEHR archetypes. The study of Christensen & Ellingsen (2016) corroborates the notion that using OpenEHR standards helps the MDD of healthcare data and software, being necessary a set of professionals with well defined roles, responsibilities and contributions.

Sun et al. (2017) create a simulator, through which a mathematical modelling evaluates the use of health care units (e.g., hospitals). Ellouze et al. (2017) present a MDSE tool for generating context-sensitive graphical interfaces, according to OpenEHR standards. Gannud et al. (2017) describe a model to model the historical relation between patients and the hospital departments where they are admitted. OMG (2018) proposes a standard for the MDD of archetype-based health data. Mallya & Kothari (2018) suggest a tool that synchronizes the model and the software requirements in real time. Kotronis et al. (2018) show an application and extension of the SysML (System Modelling Language) for modelling HIS using the Internet of Things.

The works mentioned above represent an important advance in the state of the art. Through their results, they demonstrate that MDD has significantly contributed to the healthcare sector. However, there is still a lack of studies addressing a visual modelling language which enables the Model-Driven Development (MDD) for healthcare software using both Health standards already established in the Industry and a reusable software architecture. The Healthcare Software Modelling Language presented in Section 3 addresses the open questions identified in state of the art and offers an alternative for the healthcare sector to build software systems based on a reusable

software architecture.

3. THE HEALTHCARE SOFTWARE MODELLING LANGUAGE (HSML)

HSML describes healthcare software in a high-level representation, as a visual diagram.

As shown in Figure 2, three standards (HL7 FHIR, OpenEHR Archetype and Reusable Software Architecture) have been used to specify the constructors that compose the HSML tool, which is made of three parts: HSML metamodel, model and transformation, to be explained below.

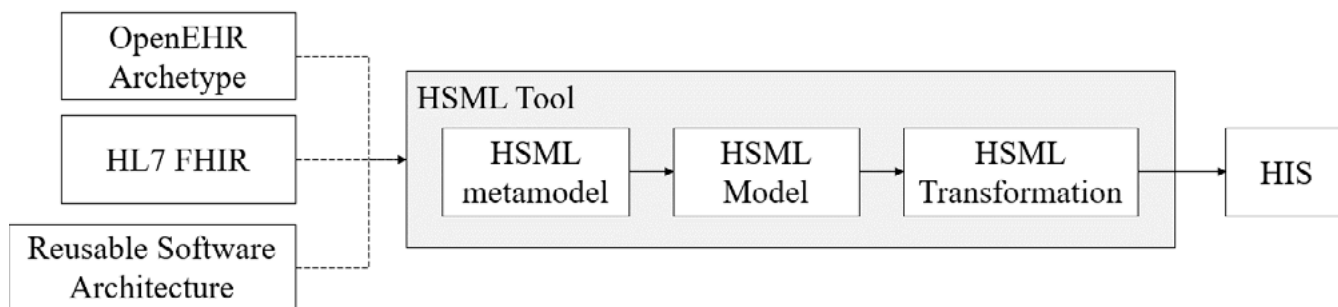



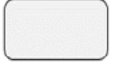



Figure 2. Overview of the HSML.

Table 1. shows the visual notations (constructors) of the HSML, which represent

concepts from OpenEHR archetype, HL7 FHIR, and reusable software architecture.

Table 1. HSML metamodels, concepts and its visual notation.

Metamodel	Concept	Visual Notation
Healthcare Software	Any healthcare software that uses archetypes to format data.	
GUI	Part of the software through which the user interacts (e.g., a web form or a mobile form).	
Relation	A binary relation between two metamodels.	
Reusable Component	Any reusable software component that performs the CRUD operations of a specific GUI.	
Database	A tool that stores data.	

Each HSML metamodel has been specified with a set of properties. Healthcare Software contains Name (the software name) and URL (the address to access this software). GUI has the

following properties: Business Process (the name of the process that this metamodel represents), Code (the code that generates the respective GUIs) and Archetype (the set of archetypes

that make the GUI). Relation contains the Label property (for describing any relation, if necessary). In Microservice, the properties are: Name (name of the microservice or its task), URL (address to access this microservice), Implementation Language (programming language that implements this software component, e.g., Java), and Exist (a Boolean data that defines whether this microservice exists or not on the Internet).

The relationship rules between HSML metamodels follows that, as depicted in Figure 3, the HSML binary relations is triple (i.e., a tuple made by three HSML constructors). To

mathematically represent the relationship rules, let us assume that the constructors Health Software, Relation, GUI, Reusable Component and Database are respectively A, R, G, C and D. Thus, to model the constructors $\{A \wedge R \wedge G \wedge C \wedge D \in \text{HSML}\}$ and to model all relationship, $\{1A n R n G \wedge 1G 1R 1C \wedge 1C 1R 1D \mid n = 1 \text{ or } n < 1\}$. This mathematical representation means that a Healthcare Software constructor can relate to one or more GUI constructors. Each GUI constructor relates to a Reusable Component constructor, each of which relates to one Database constructor.

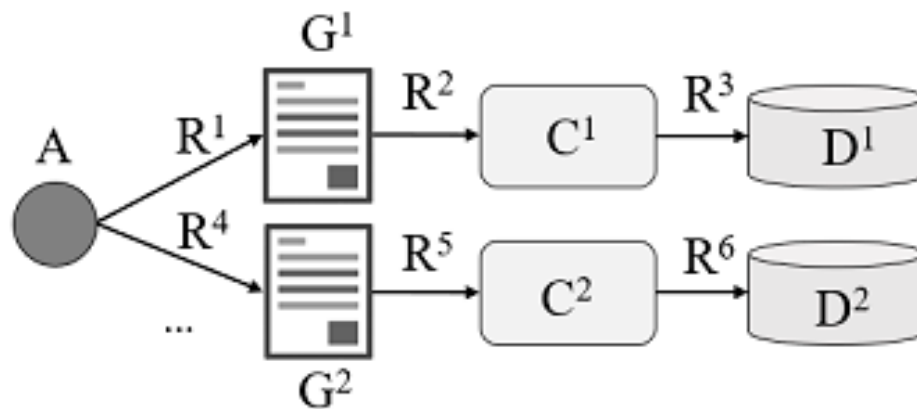


Figure 3. Relationship rules of the HSML constructors.

With the constructors and their relationship rules, software developers create HSML models, which are instances of healthcare software that are represented via a diagram. In this study, a HSML tool was built using the Sirius Platform, a free and open source solution available on the Internet. Three steps were needed to create the HSML tool using the Sirius platform. Firstly, all metamodels are defined into Sirius. Secondly, the modelling environment is dynamically generated from the set of metamodels. After that, to execute the HSML transformation, a MOF M2T language

script was used to verify the constructors (and their relations) used in the modeling environment, finally, all HSML models are transformed to a JSON document. To dynamically generate the reusable software architecture from the HSML model, the JSON document is used as input in the Microservice4EHR tool.

Even though OpenEHR archetype and the FHIR standard are not explicit in the HSML constructors, the archetypes are used in the GUI constructor, and the HL7 FHIR standard is applied in the computer code generation.

4. MIGRATING A LEGACY HIS TO THE REUSABLE SOFTWARE ARCHITECTURE

In this section, two real-world scenarios are adopted to assess the HSML tool. The Health domain is the Brazilian National Health System, which is governed by the Health Ministry. Two legacy HIS - that execute healthcare protocols called “risk-rated patient care” and “blood donation” - are presented. These HIS are redesigned using the HSML tool and then, the computer codes are dynamically generated. After this code generation, this study evaluates the reusability of both legacy HIS and the newly generated HIS.

4.1. “Risk-rated Patient Care” HIS

The Brazilian Health Ministry has specified a risk-rated care protocol to be applied in all public hospitals. These protocol are made of three clinical processes: (i) a patient looks for a Health center (e.g., hospital), (ii) the patient is welcomed by employees or interns and sent to fill in the reception form, (iii) the patient is sent to a sector that evaluates his/her risks. This last protocol is done by a nurse.

In São Luiz, Maranhão (northeastern Brazil), there are legacy software made of three Graphical User Interface (GUI) (e.g., a webform) to perform the last two clinical processes normalized by the risk-care protocol mentioned above. The first one is a webform for patient admission, which includes personal data (e.g., name, identification number, date, time, gender, age, address, phone). The second one is a webform for initial healthcare data (e.g., main complaint, brief history, direct observation, blood pressure, temperature, pulse, pain intensity scale). The last one is a webform

in which a nurse rates the patient’s risk on a scale of four levels (red, yellow, green, blue), data about drugs and addictions, allergies, pre-existing conditions, local time and the name of the professional who performed these observations (i.e., a nurse).

The software architecture of these legacy software are mostly monolithic, where data is not formatted using OpenEHR archetypes and whose database is hosted in a local server. The HSML provides an updated version, whose data are formatted by archetypes and software architecture components are microservices. All steps are detailed below.

Firstly, a set of archetypes is identified in OpenEHR CKM (Clinical Knowledge Manager) to support all clinical processes, e.g., admission. `v0.adl`, `individual_personal.v1.adl`, `pulse.v1.adl`, `body_temperature.v2.adl`, `blood_pressure.v2.adl`, `critical_pain_observation_tool.v0.adl`, `problem_list.v1.adl`, `medication_substance.v0.adl`, `clinical_synopsis.v1.adl`, `health_risk.v1.adl`.

Secondly, a HSML model is created. Figure 4 illustrates the Risk-rated Protocol Care HIS, which contains three webforms that models three clinical processes: Patient Admission, Previous Screening, and Patient’s Risk. Each GUI constructor interacts with a respective Reusable Component constructor, which communicates with a Database. Properties are shown in grey colour. All URLs, ports and names to access reusable components (e.g., microservices) and databases are shown in the model. Some of them are only partly shown for security reasons.

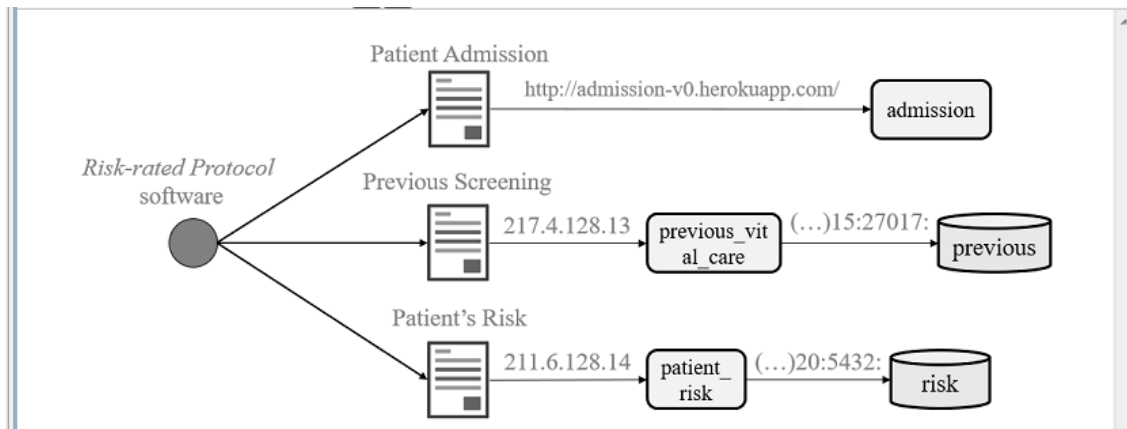


Figure 4. The HSML-based “Risk-rated Patient Care” HIS.

In this study, the property code of each GUI constructor has been filled out with the HTML file created by Template4EHR. This tool has as input a set of OpenEHR archetypes and as output a HTML file made from these archetypes. Some properties of metamodels do not appear in the model, to offer a better visualization of the model between the development teams. However, whether a software engineer clicks on the metamodels, all properties appear in the HSML tool properties bar.

M2T transformation generates a new version of the legacy software called Risk-rated

Protocol, whose EHR uses OpenEHR archetypes and software architecture uses microservices. Figure 5 shows all components generated to make the architecture of the new software. In User Layer, each webform communicates with its respective Connector, which in turn interacts with a Microservice. Connector 1 processes all data from the GUI and sends it to Microservice A, which performs the registration of personal data. Connector 2 and Microservice B process the registration of initial vital data. Connector 3 and Microservice C run data about risk-rated records.

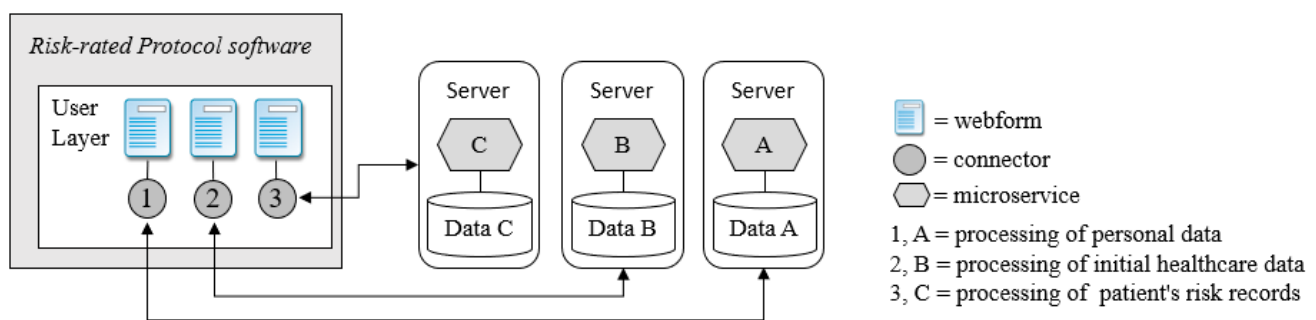


Figure 5. New software architecture of Brazilian public hospital built by HSML.

Figure 6 shows an example of a HTML-based graphical interface related to the first functional requirement from the Risk-rated Patient Care HIS, which is the admission data. This webform is

based on archetype individual_persoanl.v0.adl. In this webform, a professional from a public hospital types data related to the patient.

The screenshot shows a webform for patient registration. It is organized into a tree structure. The root is 'Patient', which contains 'Personal name'. 'Personal name' has a cardinality of [0..*] and contains a 'Structured name' section. 'Structured name' has three fields: 'Given name [1..1]' (text input), 'Middle name [0..*]' (dropdown), and 'Family name [1..1]' (text input). To the right of this is the 'Identifier' section with an empty text input. Below that is 'Date of Birth' with a date input containing '18/04/2020'. Next is 'Sex' with a dropdown menu showing 'Male'. Below that is 'Address [0..*]' with an 'Address Type [1..1]' dropdown menu showing 'Residential'. Underneath 'Address Type' is a 'Structured address' section containing a 'Property number' text input field.

Figure 6. Webform of the patient registration.

4.2. “Blood Donation” HIS

In conformity with norms of the Brazilian Health Ministry, a blood donation center in Aracaju city presents as functional requirements the registration of (i) the donor’s personal data, (ii) prior screening, (iii) haematological testing, and (iv) interview with the donor. In the investigated scenario, the blood donation center used a web application whose architecture is monolithic. User and Business Layer components are hosted on a server located in the blood donation center itself, which is accessed through a local network. In the User Layer are the graphical user interface webforms developed in Java. In the Business Layer, there is a set of Module components which models four functional requirements based on norms of the Brazilian Health Ministry. These Module components are interconnected, interdependent, and share resources from the same computer. All Module components are developed in Java. In the data layer, a proprietary database is used, which is accessed through the Internet. There is no standard in the interoperability between software components and the formatting of health data.

Starting the update of the legacy software, the functional requirements already in use were initially observed. Afterwards, archetypes were identified to support these functional requirements, which are: individual_personal.v1.adl, height.v1.adl, body_weight.v1.adl, blood_pressure.v2.adl, lab_testblood_glucose.v1.adl, pathology_test.v1.adl, pulse.v1.adl, tobacco_use_summary.v1.adl, alcohol_use_summary.v1.adl, problem_list.v1.adl, medication_substance.v0.adl, clinical_synopsis.v1.adl.

Figure 7 illustrates a model of a Blood Donation HIS that contains four webforms that model the registration of donor personal data, prior screening, haematological screening, and interview with the donor. The first GUI constructor (“Donor Personal Data”) reuses an existing component available on the Internet (which is also used in the previous migration shown in Section 4.1). The other GUI constructors interact with their respective Reusable Component constructors, each of which communicates with its Database constructor. Properties are shown in grey color. All URLs, ports and names to access microservices and databases are shown in the model.

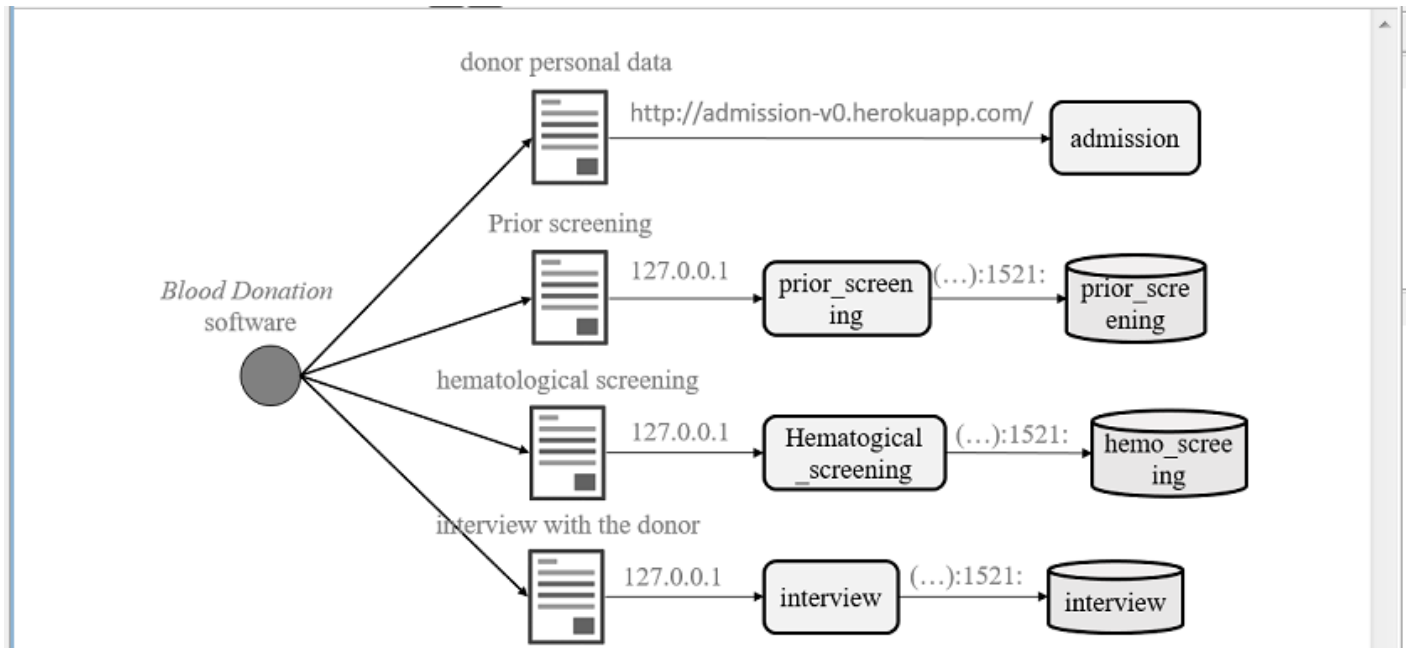


Figure 7. A HSML-based “Blood Donation” HIS.

After the HSML transformation, a new software architecture can be used to replace the legacy software architecture previously used in the blood donation center. In this assessment, four Connectors have been built and deployed at the user layer. Each Connector is related to a respective HTML GUI and requests the execution of tasks from a set of microservices. Four Microservice have also been generated by the proposed tool and deployed to servers on the Internet. Each Microservice performs the task of a single functional requirement specified by the Brazilian Health Ministry. Each microservice is autonomous and independent in the execution of its tasks. It can be hosted on a specific server and made available on the Internet (i.e., any software or other microservice can interact with it) and it does not share resources. Each microservice also has its own database. Health data transmitted and processed in this software architecture is based on archetypes (specified by the OpenEHR Foundation) and the interoperability between

software architecture components is based on FHIR’s REST APIs (specified by HL7 International).

Algorithm samples of the generated codes by HSML tool are available at <https://micro4ehr.herokuapp.com/files/02.txt>. These codes are built in HTML, Javascript, XML and Java.

4.3. Reusability Evaluation

This section shows results of the reusability of legacy software architectures and that one generated when the HSML tool is used for migrating between HIS versions.

To demonstrate such reusability, a reusability modeling service based on Graph theory is extended in this paper, as mentioned in the introduction. In the $G=(V,E)$ Graph modeling, vertices (V) are services and edges (E) are the interactions between two services. The reusability is measured by observing how many edges interact with a vertex (service).

Extending this Graph-based modeling,

we have specified two kinds of vertices: a set of microservice (M) and a set of software (S). An edge between them means a specific software interacts with a specific microservice, i.e., the software can use the interface of the microservice to send and receive data.

To show all probabilities of reuse, we have also specified two kinds of vertices and edges: real, for real-world environment cases, and hypothetical, for cases that may happen in design. To consider a hypothetical scenario is an important feature in microservice-based software architecture, because the infrastructure that processes the microservices has to be prepared for the growing reusability.

Thus, vertices are made of real and hypothetical microservices (Mr, Mh) and real and hypothetical software (Sr, Sh), while edges are made of real and hypothetical edges (Er, Eh). Visually, a real vertex is grey with a solid border line, real edge has a solid border line, hypothetical vertex is white with a dotted border line, and hypothetical edge has dotted lines. The

mathematical formula is $G = \{(Mr,h, Sr,h), (Er, Eh) \mid (Mr,h, Sr,h) \in V, (Er,h) \in E\}$.

Figure 8 shows the Risk-rated Patient Care (s1) and Blood Donation (s2) softwares, which interact with their own microservices. The microservice mA is real and performs the first functional requirement (i.e., record of personal data). The microservices mB and mC are also real and process the registration of initial vital data and risk-rated care data, respectively. Since all Brazilian public hospitals follow the same norms from the Health Ministry, let us also consider three hypothetical public hospitals (from different locations such as example Pernambuco, Alagoas and Sergipe). Their softwares (s3, s4, s5) would perform the same functional requirements, each of which can reuse existing microservices (mA, mB, mC). The real-world blood donation center software (s2) reuses microservice mA and interacts with three other microservices (mD, mE, mF), which respectively executes the registration of the previous screening, haematological screening, and interview.

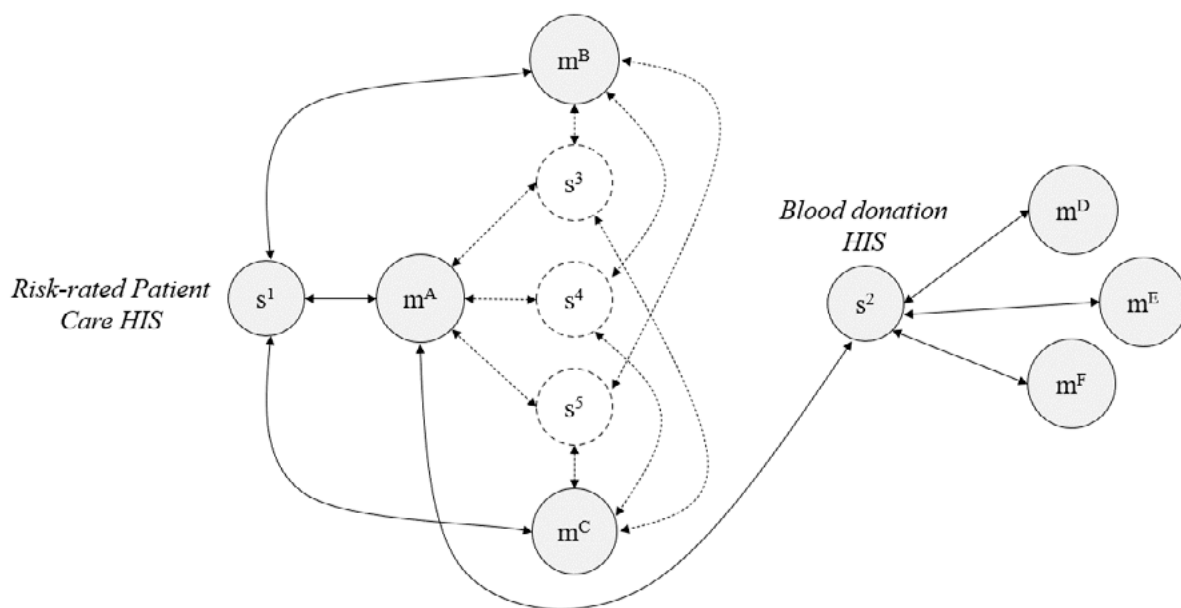


Figure 8. A graph-based reusability representation.

This graph representation (Figure 8) demonstrates the reusability of microservices by a variety of software (s1 and s2). Also, the modeling of hypothetical scenarios (s3, s4 and s5) is possible. This hypothetical representation brings important needs to the design stakeholders: (i) higher processing power by mA microservice infrastructure, and (ii) how the security configuration of the mA microservice must be implemented to be accessible by other varieties of software.

5. CONCLUSION AND FUTURE WORKS

This paper introduces HSML, a tool that allows software developers to build Health Information System (HIS) using OpenEHR archetypes to store/model healthcare data through the Model-Driven Development (MDD) approach. A set of metamodels are specified in this paper, through which software developers can model any archetype-based HIS and dynamically generate the software artifacts that make the HIS. HSML is specified to support international health standards such as archetype and HL7 FHIR. Following the norms from the Brazilian Health Ministry, the assessment of

HSML migrates two legacy healthcare software belonging to the National Health System, whose software architecture reusability is inexistent. In the migration, a new version of the HIS is built. In the code transformation, HSML generates a software architecture whose components are microservices, which can be shared (or reused) by different sets of software.

This study also extends the modeling based on Graph theory, which demonstrates the reusability of the software architecture components in a real or hypothetical scenario. As a result, two HIS that execute the “Risk-rated Patient Care” and “Blood Donation” protocols could use the same component hosted on the Internet. Such reusability is demonstrated by a graph representation specified in this study, in which hypothetical reusability scenarios are also presented using other healthcare software used in Brazilian National Health System.

As future works, we consider (i) a HSML modeling for more complex scenarios, and (ii) an assessment on the adaptability, interoperability and scalability of the components generated by HSML.

REFERENCES

AGGARWAL, V. (2002) “The application of the unified modeling language in object-oriented analysis of healthcare information systems,” **J Med Syst.** 2002 Oct; 26(5):383-97. DOI: 10.1023/a:1016449031753.

ARAUJO, A., TIMES, V. and SILVA, M. (2020) “A Tool for Generating Health Applications

Using Archetypes,” in **IEEE Software**, vol. 37, no. 1, pp. 60-67, Jan.-Feb. 2020. Doi: 10.1109/MS.2018.110162508.

BASS, L., CLEMENTS, P. AND KAZMAN, R. (2012). **Software Architecture in Practice** (3rd ed.). Addison-Wesley Professional.

BRAMBILLA, M., CABOT, J., WIMMER, M., BARESI, L. (2017) “**Model-Driven Software**

Engineering in Practice: 2nd Edition,” 2nd Edition, Morgan & Claypool.

BRAUN, R., SCHLIETER, H., BURWITZ, M., ESSWEIN, W. (2015) “Extending a Business Process Modeling language for Domain-Specific Adaption in Healthcare”, in: Thomas. O.; TEUTEBERG, F. (Hrsg.): **Proceedings der 12. Internationalen Tagung Wirtschaftsinformatik (WI 2015)**, Osnabruck, S. 468-481.

CHA, J., KIM, J., AND JEONG, Y. (2016). “Architecture Based Approaches Supporting Flexible Design of Self-Adaptive Software,” 2016 Int. Conf. on **Computational Science and Computational Intelligence (CSCI)**, Las Vegas, NV, pp. 1424-1425. doi: 10.1109/CSCI.2016.0280.

CHANG, P.H. (2011) “Modeling the Management of Electronic Health Records in Healthcare Information Systems,” Int. Conf. **on Cyber-Enabled Distributed Computing and Knowledge Discovery**, Beijing, pp. 580-584. doi: 10.1109/CyberC.2011.98.

CHRISTENSEN, B. AND ELLINGSEN, G. (2016) “Evaluating Model-Driven Development for large-scale EHRs through the openEHR approach,” **Int. J. of Medical Informatics**, Elsevier. <http://dx.doi.org/10.1016/j.ijmedinf.2016.02.004>.

DEMSKI, H., GARDE, S. AND HILDEBRAND, C. (2016) “Open data models for smart health interconnected applications: the example of openEHR,” **BMC Medical Informatics and Decision Making**, 16:137. DOI 10.1186/s12911-016-0376-2.

ECLIPSE FOUNDATION (2019), **Sirius** [Online]. Available: <https://www.eclipse.org/sirius/>. [Accessed 15 April 2020].

ELLOUZE, A.S., TLILIA, S.H. AND BOUAZIZ, R. (2017) “A Model-Driven Based Methodology for the Generation of Context-

Aware Medical Interfaces from OpenEHR Archetypes,” **J Health Med Informat** 8: 279. doi: 10.4172/2157-7420.1000279.

GOMES, A.T.A., ZIVIANI, A., CORREA, B.S., TEIXEIRA, I.M., MOREIRA, V.M. (2012) “SPLiCE: a software product line for healthcare,” In Proceedings of the 2nd ACM SIGHIT Int. **Health Informatics Symposium (IHI '12)**. ACM, New York, NY, USA, 721-726. DOI: <http://dx.doi.org/10.1145/2110363.2110447>.

HEALTH BRAZIL MINISTRY (2002) **Ordinance No. 2048** [Online]. Available: http://bvsm.saude.gov.br/bvs/saudelegis/gm/2002/prt2048_05_11_2002.html (Portuguese). [Accessed 16 April 2020].

HL7 INTERNATIONAL (2020). **HL7 FHIR R4** [Online]. Available: <https://www.hl7.org/fhir/>. [Accessed on 13 April 2020].

ISO (2020) [Online]. Available: <https://www.iso.org/>. [Accessed 16 April 2020].

KAKIVAYA, G. ET AL. (2018) “Service fabric: a distributed platform for building microservices in the cloud,” in **Proceedings of the 13 EuroSys Conf.** (EuroSys '18). ACM, New York, NY, USA, 2018, Article 33, 15 pages. DOI: 10.1145/3190508.3190546.

KANNAN, V., FISH, J.C., AND WILLET, D.L. (2016) “Agile model driven development of electronic health record-based specialty population registries,” **IEEE-EMBS Int. Conf. on Biomedical and Health Informatics (BHI), Las Vegas, NV, 2016**, pp. 465-468. doi: 10.1109/BHI.2016.7455935.

KHAMBATI, A., GRUNDY, J., WARREN, J. AND HOSKING, J. (2008) “Model-Driven Development of Mobile Personal Health Care Applications,” 23rd IEEE/ACM Int. **Conf. on Automated Software Engineering, L'Aquila, 2008**, pp. 467-470. doi: 10.1109/ASE.2008.75.

- KOTRONIS, C., NIKOLAIDOU, M., DIMITRAKOPOULOS, G., ANAGNOSTOPOULOS, D., AMIRA, A. and Bensaali, F. (2018) "A Model-based Approach for Managing Criticality Requirements in e-Health IoT Systems," **13th Annual Conf. on System of Systems Engineering (SoSE), Paris**, pp. 60-67. doi: 10.1109/SYSOSE.2018.8428764.
- LARRUCEA, X., SANTAMARIA, I., COLOMOPALACIOS, R. AND EBERT, C. (2018) "Microservices," in **IEEE Software**, vol. 35, no. 3, pp. 96-100, May/June 2018. Doi: 10.1109/MS.2018.2141030.
- MALLYA, R. AND KOTHARI, S. (2018) "Self-Adaptive Woman Health Monitoring System Using MAPE Components," **3rd Int. Conf. for Convergence in Technology (I2CT), Pune**, pp. 1-7. doi: 10.1109/I2CT.2018.8529760.
- MARTÍNEZ-COSTA, C., MENÁRGUEZ-TORTOSA, M., FERNÁNDEZ-BREIS, J.T., MALDONADO, J.A. (2009) "A model-driven approach for representing clinical archetypes for Semantic Web environments," **J. of Biomedical Informatics**, Volume 42, Pages 150-164, <https://doi.org/10.1016/j.jbi.2008.05.005>.
- MOHR, F. (2014). A Metric for Functional Reusability of Services. In: Schaefer I., Stamelos I. (eds) **Software Reuse for Dynamic Systems in the Cloud and Beyond**. ICSR 2015. **Lecture Notes in Computer Science**, vol 8919. Springer, Cham.
- OPENEHR FOUNDATION (2020). **OpenEHR**. Accessed on: Apr.15, 2020. [Online]. Available: <https://www.openehr.org>.
- PRESSMAN, R., MAXIM, B. (2019) **Software Engineering: A Practitioner's Approach**. McGraw-Hill Education, ISBN: 1259872971.
- RAD, A.A., BENYOUCEF, M. AND KUZIEMSKY, C.E. (2009) "An Evaluation Framework for Business Process Modeling Languages in Healthcare," **Journal of Theoretical and Applied Electronic Commerce Research**, VOL 4, ISSUE 2, p. 1-19; DOI: 10.4067/S0718-18762009000200002.
- RADEMACHER, F., SORGALLA, J., SACHWEH, S. AND ZÜNDORF, A. (2019) "Viewpoint-Specific Model-Driven Microservice Development with Interlinked Modeling Languages," **IEEE International Conference on Service-Oriented System Engineering (SOSE)**, San Francisco East Bay, CA, USA, 2019, pp. 57-5709.
- RAHMAN, M., GAO, J. (2015) "A Reusable Automated Acceptance Testing Architecture for Microservices in Behavior-Driven Development," **IEEE Symposium on Service-Oriented System Engineering**, pp. 321-325.
- RAISTRICK, C. (2004) "Applying MDA and UML in the development of a healthcare system". In **UML Modeling Languages and Applications**, Nuno Jardim Nunes, Bran Selic, Alberto Rodrigues da Silva, and Ambrosio Toval Alvarez (Eds.). Springer-Verlag, Berlin, Heidelberg 203-218.
- REICHERZER, T., COFFEY, J., GONEN, B. AND GILLET, I. (2015) "Knowledge modeling in the health care domain to support software development & maintenance," **3rd Int. Conf. on Model-Driven Engineering and Software Development (MODELSWARD)**, Angers, pp. 470-476.
- SEMENOV, I. KOPANITSA, G. DENISOV, D. ET AL. (2018) "Patients Decision Aid System Based on FHIR Profiles," **J Med Syst**, 42: 166, Springer US. <https://doi.org/10.1007/s10916-018-1016-4>.
- SILVA, M.A.P., TIMES, V.C., ARAÚJO, A.M.C. (2019) "A Microservice-Based Approach for Increasing Software Reusability in Health Applications," **2019 IEEE/ACS 16th**

International Conference on Computer Systems and Applications (AICCSA), Abu Dhabi, United Arab Emirates, 2019, pp. 1-8.

SERVIN, S.C.N., PINHEIRO, E., MACIEL D.O. ET AL. (2020) “**Risk-rated Healthcare Protocol**,” [Online]. Available: http://bvsmis.saude.gov.br/protocolo_acolhimento_classificacao_risco.pdf (Portuguese). [Accessed 15 April 2020].

SUN, X., LI, M., MENG, C., KONG, N., MENG, H. AND HYER, K. (2017) “Data-driven simulation for healthcare facility utilization modeling and evaluation,” In **Proceedings of the Winter Simulation Conf. (WSC)**. IEEE Press, Piscataway, NJ, USA, Article 233, 12 pages.

TELANG, P.R., KALIA, A.K., SINGH, M.P. (2015) “Modeling Healthcare Processes Using Commitments: **An Empirical Evaluation**,” **PLoS ONE** 10(11): e0141202. doi: <https://doi.org/10.1371/journal.pone.0141202>.

THINK!EHR PLATFORM (2020). **EHR Score** [Online]. Available: <https://www.ehrscape.com>. [Accessed on 18 April 2020].

VASILAKIS, C., LECZNAROWICZ, D., LEE, D.C. (2008) “Application of Unified Modelling Language (UML) to the modelling of health care systems: An introduction and literature survey,” **Int. J. of Healthcare Information Systems and Informatics**, 3(4), 39-52. <https://doi.org/10.4018/jhisi.2008100103>.