INTERNATIONAL JOURNAL OF MEDICAL INFORMATICS 78 (2009) 83–103

journal homepage: www.intl.elsevierhealth.com/journals/ijmi

A development framework for semantically interoperable health information systems

Diego M. Lopez, Bernd G.M.E. Blobel*

eHealth Competence Center, University of Regensburg Medical Center, Franz-Josef-Strauss-Allee 11, D-93053 Regensburg, Germany

ARTICLE INFO

Article history:
Received 13 February 2007
Received in revised form
20 May 2008
Accepted 30 May 2008

Keywords:
Integrated health care systems
Systems analysis
Software process engineering
Model-Driven Architecture
Semantic interoperability
Unified process
HL7

ABSTRACT

Background: Semantic interoperability is a basic challenge to be met for new generations of distributed, communicating and co-operating health information systems (HIS) enabling shared care and e-Health. Analysis, design, implementation and maintenance of such systems and intrinsic architectures have to follow a unified development methodology.

Methods: The Generic Component Model (GCM) is used as a framework for modeling any system to evaluate and harmonize state of the art architecture development approaches and standards for health information systems as well as to derive a coherent architecture development framework for sustainable, semantically interoperable HIS and their components. The proposed methodology is based on the Rational Unified Process (RUP), taking advantage of its flexibility to be configured for integrating other architectural approaches such as Service-Oriented Architecture (SOA), Model-Driven Architecture (MDA), ISO 10746, and HL7 Development Framework (HDF).

Results: Existing architectural approaches have been analyzed, compared and finally harmonized towards an architecture development framework for advanced health information systems.

Conclusion: Starting with the requirements for semantic interoperability derived from paradigm changes for health information systems, and supported in formal software process engineering methods, an appropriate development framework for semantically interoperable HIS has been provided. The usability of the framework has been exemplified in a public health scenario.

Abbreviations: NET, The Microsoft .NET Framework; ACME ADL, Generic Software Architecture Description Language; ADD, SEI Attribute-Driven Design; ADM, Architecture Development Method; ATAM, Architecture Tradeoff Analysis Method; CDA, HL7 Clinical Document Architecture; CEN, European Committee for Standardization; CMMI, Capability Maturity Model® Integration; COM, Component Object Model; CORBA, Common Object Request Broker Architecture; DODAF, Department of Defense Architecture Framework; FEAF, Federal Enterprise Architecture Framework; HDTF, Health Domain Taskforce; HTML, HyperText Markup Language; IBM, International Business Machines Corporation; ICD-10, International Classification of Diseases; IEEE 1471-2000, IEEE Recommended Practice for Architectural Description; ISO/IEC, International Organization for Standardization/International Electrotechnical Commission; J2EE, Java 2 Enterprise Edition; OMG, Object Management Group; OpenEHR ADL, OpenEHR Archetype Definition Language; RIM, Reference Information Model; RM-ODP, Reference Model for Open Distributed Processing; RUP, Rational Unified Process; SAAM, Architecture Analysis Method; SEI, Software Engineering Institute; SPEM, Software Process Engineering Meta-Model; TOGAF, Open Group's Architecture Framework; UML, Unified Modeling Language; XML, Extensible Markup Language.

* *Corresponding author*. Tel.: +49 9419446767; fax: +49 9419446766.
E-mail addresses: diego.lopez@ehealth-cc.de, dmlopez@unicauca.edu.co (D.M. Lopez).

1386-5056/$ – see front matter © 2008 Elsevier Ireland Ltd. All rights reserved.
doi:10.1016/j.ijmedinf.2008.05.009

1. Introduction

For meeting the challenge for efficient, high quality and sustainable care, increasingly specialized and distributed health systems in developed, and more in developing countries require extended communication and cooperation between all principals involved in patient's care. Principals are all actors within the information management chain such as persons, organizations, systems, devices, applications, components or even single objects [1].

Characterizing a system by its inherent technical, environmental, or policy conditions, communication between different actors requires open, flexible, scalable, service-oriented, intelligent, semantically interoperable, and trustworthy information systems. Requirement analysis, design, implementation, evaluation, use, and maintenance of such complex systems have to follow an agreed process. A formal health information systemS (HIS) development process covering domain perspectives, the views on systems, the composition and decomposition (generalization and specialization) of systems components, comprising not only the architecture development, but also the tooling to be deployed, needs to be provided.

In this paper, a development framework as a set of principles and guidelines as well as methodologies and techniques to be deployed within a unified development process for realizing semantic interoperability in health information systems is proposed, and its usability is exemplified in a public health scenario. For achieving this objective, state of the art approaches for information system's architecture are analyzed and then harmonized towards the framework. As a result, the approach realizes architectural principles by defining an architecture development process in a standard way, documenting, as no other methodology does: tasks, responsible persons, artifacts, products, guidance, phases, and workflows. Particularly important is the use of formal software processes engineering methods and the use of the newest version of the Rational Unified Process (RUP) framework, defined as the facto standard for software systems development. RUP facilitates the flexibility, scalability and reusability of the methodology by providing tooling for delivering the methodology through exportable Web pages and XML Metadata Interchange (XMI) documents. The advantage using the RUP tooling is that the development process's documentation can be viewed (as HTML pages) and shared (as XML files) among software development teams, enforcing them to design consistent models and use of the same guidelines, therefore supporting the design of semantically interoperable models and finally systems.

The paper is organized as follows: In Section 2, business needs and interoperability requirements for health information systems have been defined. Because of its fundamental importance for the approach offered in the paper, systems, systems architecture and systems modeling are systematically and carefully considered in Section 3, thereby introducing the Generic Component Model (GCM). Analyzing and evaluating existing architectural approaches to information systems especially focusing to health-related environments in Section 4, a comprehensive development framework for semantically interoperable health information systems has been derived in Section 5. The Generic Component Model has been used for adapting the Rational Unified Process and harmonizing the outcome with the aforementioned existing approaches. In Section 6, the use of the health information system development framework is exemplified by describing the system architecture of a public health information system, before concluding the paper in Section 7.

In the paper's context, the architecture development framework considers Architecture Vision, Business Architecture, Information System Architecture, and partially Technology Architecture, all expressed by formal models. Other architectural issues such as maintenance, adaptation, governance, etc., are out of scope of the current research paper.

2. Interoperability requirements

The shift in care paradigm towards highly distributed, labor-sharing care settings also concerns the supporting information and communication technology (ICT). Information systems are forced to support communication and co-operation (interoperability) at different levels: technical interoperability at plug, signal and protocol level; structural interoperability realized as simple data exchange; syntactic interoperability as meaningful data exchange with agreed vocabulary; semantic interoperability with common information models and agreed communicating applications' behavior; organization/service interoperability based on common business models and chained services.

Especially in the context of long-term usable e-Health applications such as Electronic Health Record (EHR) systems, several crucial requirements for semantic interoperability reflecting basic business needs have to be met. According to ISO/TR 20514 [2], there are four prerequisites for EHR semantic interoperability: (i) agreement on a standardized reference model, (ii) standardized service interface models to provide interoperability between health services and other services such as demographics, terminology, access control and security (iii) a standardized set of domain-specific concept models, e.g., archetypes and templates for clinical, demographic, and other domain-specific concepts, and (iv) standardized terminologies associated with controlled vocabularies. These requirements only concern informational aspects of HIS, however. From a more comprehensive perspective also including organizational, Technology Architecture as well as other aspects, communication and co-operation between different health information systems and their components in a complex and highly dynamic environment also requires [3]:

- openness, scalability, flexibility, portability;
- distribution at internet level;
- standard conformance;
- business process orientation;
- consideration of timing aspects of data and information exchanged;

- user acceptance;
- lawfulness;
- appropriate security and privacy services.

For achieving the aforementioned characteristics, the HIS development process (requirement analysis, design, implementation, evaluation, use, and maintenance) has to meet the following paradigms:

- architecture focus;
- separation of system architecture and software product development (sustainability, portability);
- distribution, component-orientation (flexibility, scalability, reusability);
- model-driven and service-oriented design (manageability, user acceptance);
- separation of platform-independent and platform-specific modeling, i.e. separation of logical and technological views (portability);
- specification of reference and domain models at meta-level (semantic interoperability);
- interoperability at service level, considering concepts, contexts, knowledge (semantic interoperability, user acceptance, lawfulness);
- common terminology and ontology (semantic interoperability);
- advanced security, safety and privacy services (user acceptance, lawfulness).

Because of complex requirements, semantically interoperable HIS and components can only be developed by meeting the different paradigms within a unified development process.

3. Systems modeling

3.1. *The role of the systems approach*

While information systems in the informatics domain are traditionally and widely just considered from the information and computerization perspective, semantic or even service interoperability requires the integration of many additional perspectives on a real world environment. Therefore, the scope of considerations shall be extended beyond bits and bytes, hard- and software to many other aspects, which are addressed by systems sciences. Thereby, principles and methodologies, but also terminologies and underlying ontologies have been borrowed from system engineering, cybernetics, natural sciences, and philosophy as well. As those principles, methodologies, etc., are frequently ignored, they have to be brought into mind shortly.

The "systems approach" or "systems thinking" proposes the use of the concept of a "system" as the basis for everything in the real world such as entities, phenomena, problems, etc. [4]. Several definitions for a system, starting from basic ones through more complex mathematical and graphical representations of systems, are in use. A basic definition considers a system as "a set of elements standing in inter-relations" [5]. The concept of an "element" refers to physical objects, interrelated activities, energy forms, ideas, concepts, mathematical problems, biological units, etc., but also any systems' components, generally called subsystems. The other key term in system theory is that of "inter-relations". The definition of inter-relations was the distinctive new idea proposed by modern systems theorists: systems analysis is not restricted to the nature of individual component elements and their organization (structure) but also concerns relationships between them, their status and operations (behavior). The inclusion of components' relationships allows for the analysis of a system as a "whole" rather than the study of its parts in isolation. Our uncommon approach coincides with Aristotle's thesis: "The whole is more than the sum of its parts".

The systems approach has been separately used in many different domains such as biology, mathematics, economics, anthropology, psychology, etc. But, according to Bertalanffy [5] and Lewis [6], the fields where system ideas have had more impact have been that of organizational and management sciences as well as engineering. This has been especially driven by the considered entities' growing complexity combined with the increased specialization of their parts.

Focused on the field of information systems engineering, the systems approach has undoubtedly had a strong influence on this discipline. Several approaches are found in the literature for describing the theoretical foundations of information systems' field [7–11]. According to Lewis [6], the idea that the information systems science is a combination of two primary disciplines, i.e. computer and management sciences, with a large number of supporting disciplines such as physiology, sociology, statistics, political sciences, economics, philosophy and mathematics, has been dominating in the last years. Unfortunately, the aforementioned theoretical bases are nowadays widely ignored. In practice, the information systems domain is being pushed separately by computer scientists mainly concerned with the technological aspects of information systems, and by management researchers and domain experts concerned with the organizational and business aspects of information systems.

This problem also occurs in the dedicated field of health informatics [12]. The relationships between technology as well as organizational views and multi-disciplinary aspects are normally devalued when modeling health information systems. Missing deep knowledge, high education, and sufficient experiences for multi-disciplinary work, health information systems development pushed by technologists (IT experts, software engineers) or health professionals leads to extended development cycles as well as to weak solutions from the other domain's perspective frequently bound up with reinventing the wheel in the correspondingly "foreign" domain. This situation results in excellently designed HIS nobody is happy in using it on the one hand and bad as well as unsafe system design desperately defended by the end-user on the other hand. Inherited traditions, different epistemology, ontology, and further aspects provide different environments not allowing for simple connection of the different domains' products. Based on deep involvement in both health and IT domains, the current paper aims to overcome this isolation and to adapt as well as to reuse existing achievements. Following, the ways of thinking and terminologies/ontologies of systems sciences will be adapted and profiled to develop health information systems architectures.

3.2. *The Generic Component Model*

For the sake of analysis of, and interaction with, the reality, any system under consideration must be described by appropriate models. A model is a partial representation of reality. It is restricted to attributes the modeler is interested in. Defining the pragmatic aspect of a model, the interest is depending on the addressed audience, the reason and the purpose of modeling the reality, and using the resulting model for a certain purpose as well as for a certain time instead of the original. Therefore, the model as a result of an interpretation must be interpreted itself. It provides a specific view on the system, created as a tool for communication between parties involved. Considering the common property of any real system of being composed by interacting elements forming a whole, but also recognizing the uniqueness nature of systems, a generic model can be used for the purpose of analyzing systems [13].

Starting in the early nineties, the Generic Component Model was developed at the Magdeburg Medical Informatics Department [14,15]. Based on the systems' ideas, the GCM provides a framework for describing any real system. While objects have been just pushed, the GCM was considering the importance of objects' inter-relations for the resulting aggregations' functionality, therefore proposing the component paradigm instead of simple object orientation. The GCM abstraction model reduces a real system's complexity, following three simplification paths or dimensions, as shown in Fig. 1. The three dimensions can be considered separately, need a combination for real design and implementation at the end, however. The first dimension reduces the complexity of really inter-relating domains by separating them. Domains of interest can be medical, legal, administrative, financial, technical, organizational, social, etc. The second dimension reduces the structural complexity of the system by decomposing them. As a result, the granularity of a system can be increased from business concepts over relations networks and basic services/functions up to basic concepts. For a comprehensive view, the system elements (hereafter referred to as components) must be aggregated (composed) again. The third dimension includes separate views on the system extending the viewpoints (VPs) defined in ISO/IEC 10746 "Information Technology—Open Distributed Processing. Part 2: Reference Model" (RM-ODP) [16]: enterprise VP, information VP, computational VP, engineering VP, and technology VP as phases of the system's development process. The role of a unified system development process is an essential part of the GCM and needs to be further developed. The unified development process covers the whole system lifecycle: planning, analysis, design, implementation, evaluation, use, and maintenance. The process expresses the inter-relationships among domains, the different views on the system, and the composition and decomposition (generalization and specialization) of components.

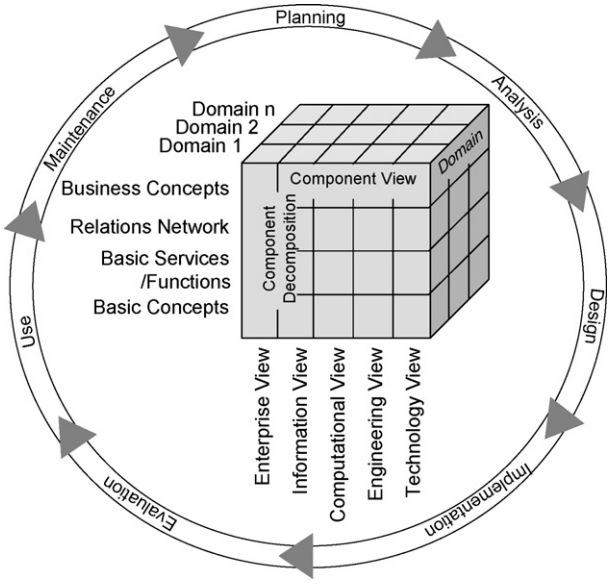

Fig. 1 – The Generic Component Model.

Restricted on information flow and processing, and ignoring other media such as materials or energy, in the next two sections the Generic Component Model principles are used as a framework and reference architecture first for the evaluation of current approaches for developing information systems architectures, and second for the harmonization and proposal of a comprehensive development framework for semantically interoperable health information systems.

4. Evaluation of architectural approaches

For developing a software-intensive health information system, the system's mission, its environmental and contextual conditions, the chosen architectural approach including its rationale as well as the stakeholders addressed by the system and its architectural description have to be clearly defined. ANSI/IEEE 1471-2000 [17] provides the IEEE Recommended Practice for Architectural Description of Software-Intensive Systems, centering on a conceptual framework for architectural description and a statement of information needed for being compliant with the standard. Fig. 2 presents the conceptual model of architectural description as defined in that standard. The concepts in the standard have been appropriately deployed for deriving the architecture development framework presented in the paper.

From a more restrictive ICT perspective than defined in Section 3, a system is a collection of components organized to accomplish functionalities. Correspondently, architecture describes the fundamental organization of a system embodied in its components, their relationships to each other, and to the environment, and the principles guiding its design and evolution [17]. Combining and simplifying those definitions, Information System Architecture can be defined as the description of the system's structure and behavior, including the compounding elements, the relationships among them and their functions. Always, the definition and description of such structure and behavior is a fundamental step in the information system's development process.

A system architecture has its own lifecycle defined as follows [18]: creating the business case; understanding requirements; creating or selecting, then documenting and communicating, thereafter analyzing or evaluating the architecture; implementing the system based on the architecture

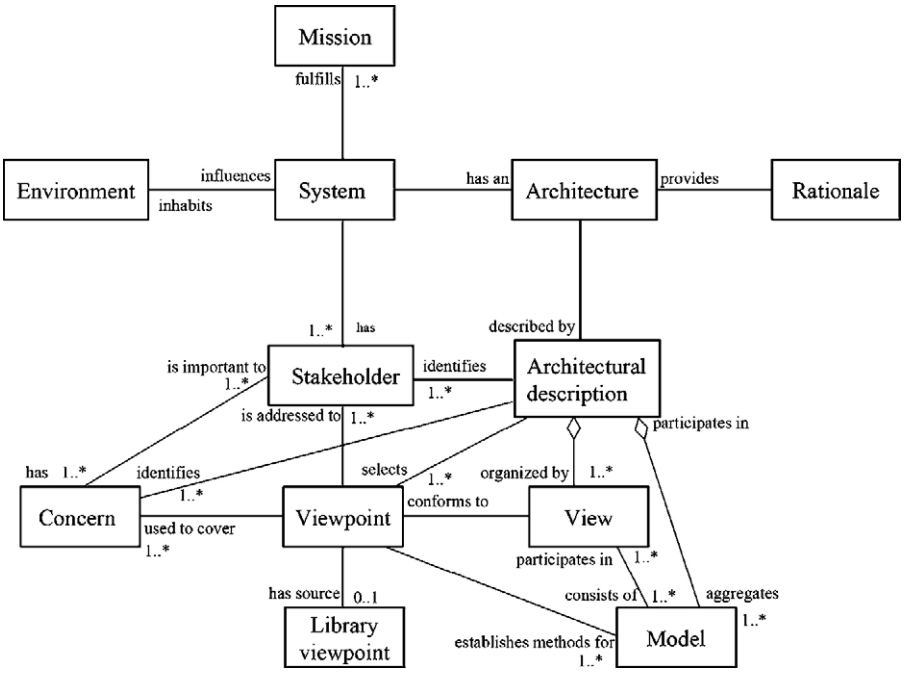

Fig. 2 – Conceptual model of architectural description after IEEE 1471-2000 [17].

and ensuring that the implementation conforms to that architecture. Often, when the objective is to implement a specific software product, the architecture development process is included within the system development methodology. However, when the architecture's objective is reusability of a set of functionalities, components, applications, or families of products; the definition of a specific architecture development methodology is necessary. The former is the problem addressed in this paper.

Architectural approaches are frequently described in many different ways: architectural frameworks, enterprise architectures, architectural models, analysis/design methods, architecture development process, architecture description languages, etc. Table 1 classifies main architectural approaches.

This paper is specifically focused on the identification and evaluation of approaches for modeling system architectures, restricted to the methods for developing such architectures (i.e. architecture development processes). Hence, some of the approaches described in Table 1 can be excluded from the evaluation process. Enterprise architectures are holistic views on architectures, focused on the description of the enterprise process. They normally do not define a specific method for designing the Information System Architecture, but provide a strategy to describe complex information systems. The Zachman Framework is an example of a broadly acknowledged enterprise framework. It proposes a structure to describe different views on the information system: scope, business model, system model, technology model, components; and working system and views on different stakeholders such as Planner, Owner, Designer, Builder and Subcontractor [19]. However, the Zachman Framework does not include a strategic planning methodology for the architecture.

For designing and implementing a solution, the provided enterprise architecture must be combined with architectural models and related methodologies. Architectural models define reference architectures and models to describe the system architecture. An example is the Service-Oriented Architecture (SOA) [20]. SOA is an architectural paradigm that promotes the development of systems as a set of services able to interoperate with other services through well-defined interfaces. SOA puts a layer on top of software components, offering independent software entities directly describing the business processes; however, SOA does not prescribe any specific development methodology in the development course of a distributed system or its architecture.

In the healthcare domain, several architecture standards, reference architectures, architecture description languages are found, most of them focused on Electronic Health Record approaches, e.g., ISO EN 13606-1 [21], OpenEHR [22] and HL7 CDA [23]. However, most of them do not prescribe any architecture development process. A comprehensive analysis of EHR architectural approaches can be found in [24,25]. The CEN EN 12967 standard [26] describes a methodology for HIS development based on RM-ODP. This methodology is only a description of the first three RM-ODP Viewpoints, providing some ideas of UML models to support each view, but not suggesting any development process.

Following, the major architectural approaches in Table 1, which describe in some way architectural development processes that are broadly used in the software engineering and healthcare domains, are analyzed in more detail.

4.1. *The Unified Software Development Process*

The Unified Software Development Process, commonly known as the Unified Process (UP), is a generic software engineer-

Table 1 – Architectural approaches[a]

Architectural approaches	Related approaches	Domain	
		Software engineering	Health informatics
Architectural frameworks	Enterprise architectures	Zachman; FEAF; DODAF; RM-ODP; TOGAF; IEEE 1471-2000	Generic Component Model
Architectural models	Reference architectures; architecture styles	Object-oriented architectures; component-based architectures; MDA; SOA; business process models	ISO EN 13606-1; OpenEHR; HL7 RIM; CDA; GCM specialization
Description languages	Modeling languages	UML; ACME ADL; OCL	OpenEHR ADL
Middleware architectures	Component architectures; service architectures	Web services; CORBA; J2EE; COM; .NET	OMG HDTF; CEN 12967
Architecture development process	Analysis/design methods and models	RUP; SAAM; ATAM; ADD; TOGAF ADM; SEI CMMI; SPEM	HL7 HDF; HL74SOA; CEN EN 12967-1.

[a] The abbreviations introduced in the table are explained in Section 'Abbreviations'.

ing process first proposed by Jacobson et al. [27]. The Rational Unified Process [28], as instance of UP, has progressed as the de-facto standard software engineering process in use today. In order to harmonize the concepts, only the RUP is deployed in the paper. RUP defines a set of phases and workflows necessary to transform user requirements into a software system. It identifies six core workflows (technical workflows): business modeling, requirements, analysis, design, implementation, test, and deployment. To define how the process rolls out over time, the RUP identifies four iterative phases (inception, elaboration, construction, and transition). Each phase is implemented as a set of workflows that can be repeated as many times as necessary (RUP iterations). Correspondently, each workflow is described as set of activities (RUP tasks) to be performed by the different members of the development team (RUP roles) leading to the development of system models and documentation (RUP work products and artifacts) to support the development of software systems. Fig. 3 describes the RUP structure and the inter-relations between main RUP concepts.

The RUP was initially supported and distributed as a commercial product of Rational Software Inc. At that time the process was offered as set of documents and supporting material describing in detail the development process components (task, artifacts, and roles) and the different scenarios (projects) in which it can be used. In 2005 after the

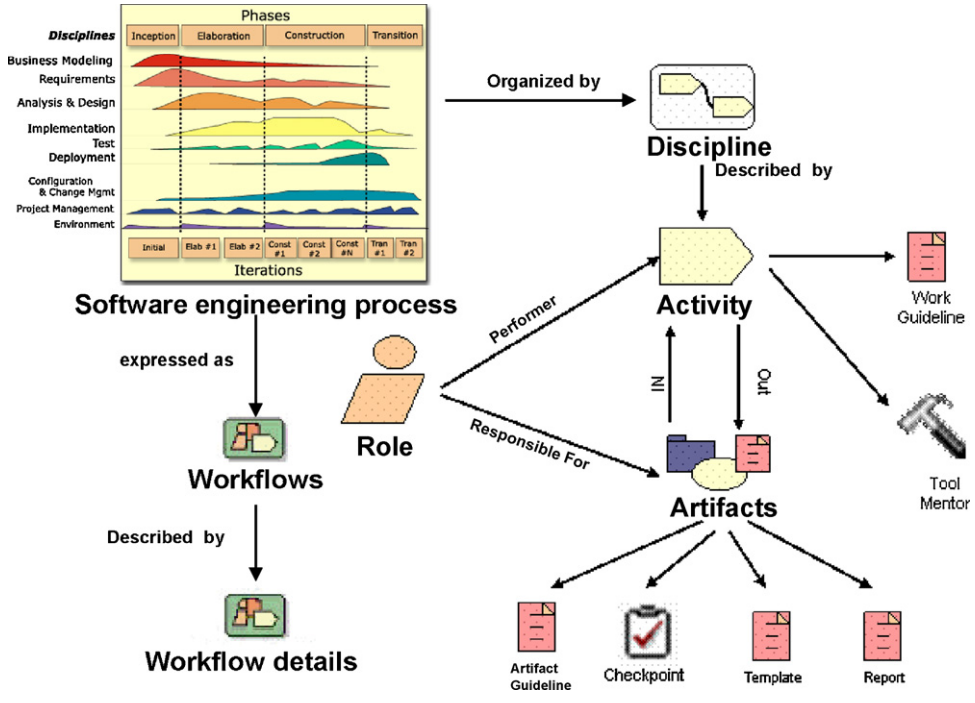

Fig. 3 – RUP structure (adapted from [28]).

product was acquired by IBM, the RUP has been evolved towards a comprehensive software platform called the Rational Method Composer (RMC). The RMC conforms to the OMG Software Process Engineering Meta-Model (SPEM) specification [29]. The RMC has two main functionalities. First it acts as knowledge base of development processes and best practices, where information about the RUP principles, concepts, components, guidance and other supporting material is customized to different application scenarios (e.g., SOA, commercial-off-the-shelf, legacy, maintenance solutions). It also provides the capability to reuse parts of the process configurations using process fragments called capability patterns, and delivery processes. A capability pattern is a special process that describes a reusable cluster of activities in common process areas (requirements, analysis, design, project management, etc.). A delivery process describes a complete and integrated approach for performing a specific type of project (e.g., complete RUP configuration for small projects, medium-sized or large scale projects). The platform allows navigating through the different process descriptions via a Web-based presentation. It is also possible to customize and incorporate new customized processes (often called process best practices) via Method Plug-ins. Method Plug-ins are reusable domain-specific method descriptions (tasks, roles, artifacts, guidance) provided as "process Lego™ pieces" and created to cover that domain-specific knowledge not supported by the RUP.

The second main functionality is the possibility to create new process components (or modify existing ones) such as tasks, roles, artifacts, guidance (also called Method Content) or processes descriptions (phases, workflows, iterations). This functionality is open to create reusable process components that can be used within the organization, shared with other development teams, or even incorporated into the IBM repository of best practices. The new process configuration can be delivered as reusable Method Plug-ins, to be additionally exported as a Web site. The last functionality will be exploited in this paper, tailoring the RUP best practice for large projects in order to provide a process configuration that supports the development of architectures for health information systems.

Finally, RUP defines two primary artifacts related to architecture: (i) the Software Architecture Description (SAD), which describes the architectural views relevant to the project; and (ii) the architectural prototype, which serves to validate the architecture and serves as baseline for further development [27]. The major architectural design activities are described in the analysis and design workflows, and take part in the inception and elaboration phases. Some shortcomings of RUP as architecture development process have been identified in the literature: the limited coverage of the whole architecture lifecycle and the insufficient guidance on quality attribute requirements [30]. To overcome those weaknesses, the incorporation of comprehensive software architecture design methods such as the Software Architecture Analysis Method (SAAM) [31], the Architecture Tradeoff Analysis Method (ATAM) [32], and the SEI Attribute-Driven Design (ADD) method [18] have been proposed. Meanwhile, some of those approaches have been harmonized towards the RUP [32], being feasible to customize it to provide the intended architecture development process.

4.2. *Model-Driven Development and Model-Driven Architecture*

The Model-Driven Development (MDD) approach refers to the intensive use of models in the systems development process. The Object Management Group (OMG) formalized the MDD approach offering the Model-Driven Architecture (MDA) [33] which is a development process characterized by the use of models and mechanisms for automatic transformation between models. It is based on the very well known and extended idea to separate the system's specification and the implementation details which makes it appropriate for the design of systems architectures. The three main goals of MDA are portability, interoperability and reusability by separating the architectural views. An MDA design basically consists of a set of system models reflecting the application's behavior from different conceptual levels together with a set of transformation models that allow for easily transforming from one model to the other. These transformation models are defined based on the platform and a set of transformation rules. MDA distinguishes three model types: (i) computation-independent models (CIM), (ii) platform-independent models (PIM), (iii) platform-specific models (PSM).

MDA broadly defines a system development lifecycle, which does not differ from the traditional software lifecycle. The PIM is a result of system's requirements and analysis stages. The PSM is the main result of the design process, and the PSM to code transformation can be related to the implementation, testing and deployment stages. However, MDA does not prescribe any specific development methodology (defining processes, task, artifacts, etc.) in the development course of a distributed system or its architecture. It specially ignores the enterprise requirements aspects with a resulting enterprise architecture needed. Furthermore, MDA technologies are not explicitly related to identifiable activities within software development processes, since these technologies are being developed to be generally applicable in combination with development processes that may already be anchored in organizations [34].

For practically using the MDA, an MDA-based architecture can easily be developed using the RUP workflow descriptions. The MDA transformations are comparable with the different models described in RUP: transformation of the Business Model and Requirements Model (CIM) into the Analysis Model (PIM); transformation of the Analysis Model (PIM) into the Design Model (PSM); and transformation of the Design Model (PSM) into the Implementation Model (Code).

Another industry standard approach to design MDA-based systems is the Architecture Development Method (ADM) of The Open Group's Architecture Framework (TOGAF) [35], describing how to derive an organization-specific enterprise architecture that addresses business requirements. It is not used in the paper's context, however.

4.3. *HL7 Development Framework*

From the healthcare perspective, the HL7 v3 standard – a widely used interoperability standard – has to be mentioned. HL7 v3 is originally based on a comprehensive development methodology called the HL7 Message Development Frame-

work (MDF), and offers models and artifacts for information modeling in healthcare. However, the MDF approach solely targets the lifecycle of standard messages specifications. Opening the HL7 scope towards semantic interoperability including the appropriate use of exchanged information in the sense of the communicating applications' behavior, the MDF is currently moving towards the HL7 Development Framework (HDF), by that way shifting the HL7 paradigm from message to architecture. Newer HL7 developments such as the EHR-S Functional Model and the SOA Project Group activities have been pushing this move.

The newest HDF specification (HDF v2 [36]) describes a development process integration workflows, artifacts, roles, tools, etc. However, the process has not been formally defined using standard process concepts (e.g., the OMG SPEM standard), and ignoring more mature development processes such us the RUP. HDF's current version includes the following core processes:

- project definition, planning, and approval;
- requirements analysis and documentation;
- specification design, analysis, and harmonization;
- standard profiling: constraints, extensions, and annotations;
- technology specification process (under development);
- change control process.

5. The development framework for semantic interoperability

After evaluating and comparing existing architectural approaches in Section 4, this section presents the outcome of harmonizing the most important ones in an architecture development framework for semantically interoperable health information systems and components.

The development framework aims at providing a comprehensive architecture development process, overcoming the limitation of current architectural approaches, and furthermore supporting the requirements for semantic interoperability when designing healthcare systems. The scope of the architecture development process covers the business analysis, requirements, and design as well as documentation stages in the architecture lifecycle. The herewith described process is restricted to platform-independent aspects of the architecture, covering the information system's enterprise, information, and computational views. The prerequisites for semantic interoperability include distribution, component-orientation, standard conformance, specification of reference and domain models, common terminology and ontology, etc., as described in Section 2.

In the next sections, the development framework for semantically interoperable health information systems and components will be hereafter referenced as HIS-DF.

5.1. *The framework at a glance*

Fig. 4 describes how the GCM core principles are extended in the HIS-DF. In general, the framework is defined as a process of continuously constraining the GCM (steps 1–5). In step 1, because the GCM is open to describe any real system, the system to be analyzed in the context of HIS-DF is first selected. The systems analysis in HIS-DF is restricted to "health information systems". In step 2, the domain dimension of GCM is brought out by separating the domain of interest (health-related domain) from other related domains (financial, administrative, security, etc.). The consideration of the related domains requires an independent analysis as demonstrated, e.g., for modeling security services [37]. In the development process, the separated domains as all other GCM dimensions separately considered below, have to be aggregated again. That is the way how complexity of all domains involved in health systems can be managed.

The third step is to address the structural complexity of HIS. All four granularity levels defined in the GCM are analyzed: basic concepts, basic functions and services, relation networks, and business concepts. This step intends to define the domain knowledge (reference business models, reference information models, domain concepts, and vocabulary) to enable semantic integration. The HL7 Reference Information

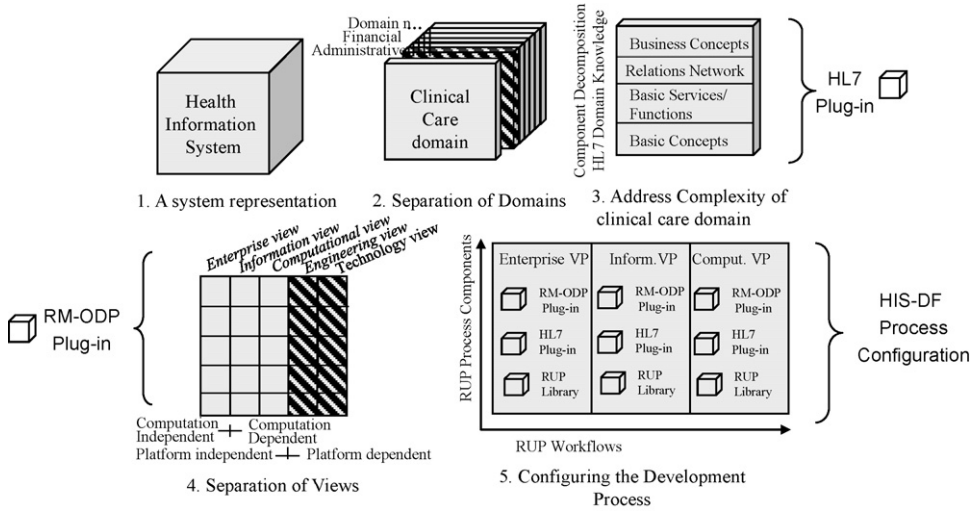

Fig. 4 – HIS-DF core process.

Model (RIM), Domain Information Models, CDA, functional service definitions, OpenEHR Archetypes and health information systems architecture standards (CEN EN 12967, OMG HDTF) are candidate sources of domain knowledge. In the HIS-DF instances presented in the paper, the domain knowledge is restricted to health-related information. In real life systems, especially in the context of distributed, fully integrated and personalized care, all other domains (e.g., security) become relevant, however.

After analyzing the enterprise architecture as well as describing the computation-independent aspects of the business reflected in the Enterprise Viewpoint, in the next phase of modeling (i.e. step 4), the scope of the system architecture is defined by restricting the analysis to platform-independent, or logical, aspects of the system reflected in the ISO/IEC 10746-2 RM-ODP Enterprise VP, Information VP and Computational VP. Platform-specific models are left to the particular system development process that uses the proposed architecture. Nevertheless, the proposed platform-independent models are portable to any environment with specific database models, operating systems' requirements, etc., by using MDA transformations.

RM-ODP provides an abstract model consisting of concepts to describe a system from each viewpoint. However, the standard does not include any specific notation for describing models, neither a particular method nor guidance for building the Enterprise, Informational and Computational Viewpoints. To cope with this problem on the basis of the RUP, HIS-DF defines in step 5 a set of process components (roles, work products, artifacts, guidance, phases, iterations and workflows) aimed at supporting each RM-ODP Viewpoint.

Summarizing, the main contribution of HIS-DF is to offer an extension to the GCM and RUP by using standard healthcare information models and vocabulary to describe the different levels of granularity of system architectures (step 3, realized as reusable method descriptions or RUP Plug-in for HL7) and standard ODP concepts for distributed architecture description (step 4, realized as reusable method descriptions or RUP Plug-in for RM-ODP). Based on that, the HIS-DF methodology is documented by combining existing RUP process components (roles, artifacts, task, guidance, phases, iterations, etc.) with contents defined in previously created Plug-ins completing step 5 in the methodology.

In the following subsections the components of the HIS-DF are developed in detail, especially addressing the steps 3–5 presented in Fig. 4.

5.2. *Tailoring the RUP to develop the HIS-DF components*

Before introducing the HIS-DF components, the process of specialization of the RUP is briefly described. RUP provides a guideline on how the process configuration can be tailored, also offering different example scenarios [38]. This guideline was adapted, and the process of developing HIS-DF was performed as follows:

1. *Select the level of tailoring to be performed*: The RUP offers several mechanisms for tailoring the RUP. The need for reusable and portable processes and the requirements for semantic interoperability enforced the use of Method Plug-ins as tailoring mechanism. Plug-ins are pieces of method descriptions which can be easily used and adapted to other development process implementations using development tools enabled to support XML Metadata Interchange documents.
2. *Tailor the RUP*: The next step is to create the necessary method components not found in the RUP method descriptions. The HIS-DF specification consists of 2 separated Plug-ins: (i) RM-ODP Method Plug-in containing method descriptions for the system architecture analysis based on RM-ODP concepts; (ii) HL7 Method Plug-in containing the method descriptions to support the process of modeling systems based on UML models and HL7 information models.
3. *Method configuration*: HIS-DF is finally developed by combining the existing RUP Method Content and the content developed in the Plug-ins.
4. *Make the method available*: The method configuration is made available for software development teams and stakeholders as a Web site and XMI files, supported on the Rational Method Composer platform. The Eclipse Process Framework (EPF) [39] is an open source alternative to perform the configuration and publication of the HIS-DF. Unfortunately, EPF is not enough for describing the HIS-DF specification because it does not include all the needed process components such as, e.g., the business modeling discipline.

In the following subsections the Plug-ins and the method configurations are detailed.

5.3. *Method Plug-in for RM-ODP*

The Method Content structure follows the Enterprise, Informational and Computational Viewpoints specified in the RM-ODP standard. As already mentioned RM-ODP standards do not prescribe any particular method for building the different viewpoints, but a set of concepts necessary for describing a distributed system. Therefore, an analysis of the standard was performed for selecting and relating the RM-ODP concepts with the RUP definitions. The first step for the development of the Plug-in is to include the RM-ODP concept definitions within the RMC environment and creating an RM-ODP Plug-in with this information. Table 2 provides a short description of each RM-ODP Plug-in element. The full Plug-in content (not included in the table) includes a definition for each concept, the use of the concept referring to the supporting RUP artifacts and task, and its notation using the UML 2.0 Profile for ODP.

The next step is to identify the existing RUP artifacts, task, roles and guidance related to each ODP concept. Table 3 describes the RM-ODP Plug-in elements matching the identified RM-ODP concepts and the corresponding RUP Process Components. For instance the ODP concept of "Community Objective" expresses the objective of an organization. In terms of the RUP, this concept corresponds to a Business Goal artifact.

The full Plug-in elements and their respective content can be consulted in the Plug-in description of the HIS-DF Web site (Left menu, tab "rm-odp_plug-in"). The HIS-DF Web site

Table 2 – RM-ODP Plug-in elements

RM-ODP Plug-in elements	Type	Description
Enterprise Viewpoint		
Community	Concept	A configuration of objects formed to meet an objective.
Community Objective	Concept	Expresses the objective defined for a community in a distributed system of objects. The objective is expressed in a contract which specifies how the objective can be met.
Process	Concept	A graph of steps taking place in a prescribed manner and which contributes to meet an objective.
Enterprise Object	Concept	A model of an entity (abstract or concrete thing of interest) in the Universe of Discourse.
Interaction	Concept	A mechanism to model ODP systems behavior. An interaction describes information interchange between roles in a community.
Policy	Concept	A policy is a set of rules related to a particular purpose.
Information Viewpoint		
Invariant Schemata	Concept	An invariant schema is a set of predicates on one or more information objects which must always be true. The predicates constrain the possible states and state changes of the objects to which they apply.
Static Schemata	Concept	A static schema is a specification of the state of one or more information objects at some point in time; subject to the constraints of any Invariant Schemata.
Dynamic Schemata	Concept	A dynamic schema is a specification of the allowable state changes of one or more information objects; subject to the constraints of any Invariant Schemata.
Computational Viewpoint		
Computational Object	Concept	A computational object models functional decomposition and interacts with other computational objects. Since it is an object, it has state and behavior, and interactions are achieved through interfaces.
Interface	Concept	An interface is an abstraction of the behavior of an object that consists of a subset of the interactions of that object together with a set of constraints on when those interactions can occur.
Object Behavior	Concept	Behavior of an object is a collection of actions with a set of constraints on when those actions may occur.

is available at [40]. For a processable and reusable description of each component, please refer to the XMI Plug-in files in [41]. Instructions on how to install the Plug-in can be found in the RMC documentation.

5.4. *Method Plug-in for semantic interoperability*

The Method Plug-in for semantic interoperability aimed to identify domain knowledge concepts (reference models, reference information models, domain concepts, vocabulary) that could be reused in the different architectural descriptions (from business through requirements and analysis to design models), in that way supporting the design of semantically interoperable components. HL7 artifacts and process components were found the most comprehensive set of medical knowledge available to be reused for the addressed purpose, and also the only ones that in its maturing phases, increasingly cover the three ODP architectural Viewpoints. Alternatively, other sources of domain knowledge for semantic interoperability such as the CEN 13606 EHR Communication standard, OpenEHR archetypes or even the CEN 12967 can be reused/integrated in the process as separated Plug-ins.

Table 4 describes the selected HL7 artifacts, becoming the components of the HL7 Plug-in. The HL7 Plug-in is more complex than the RM-ODP one, consisting of task, artifacts, roles, tools, concepts and guidelines. For example, HL7 Use Case Model – an artifact created for each single HL7 specification in the HDF – is included in the Plug-in as a HIS-DF artifact which can be used as input to the RUP task "Find Actors and Use Cases" to create the Systems Business Use Case Model. Other domain concepts, not identified in Table 4, are also included in the Plug-in as concept definitions, artifacts or tasks. A fundamental artifact in the HL7 Plug-in is the "Harmonize Information Models" task. This artifact is used to harmonize information models and HL7 standards, necessary to support semantic interoperability.

In Table 5, the RUP method components (artifacts, roles, tasks, and guidance) related to each HL7 Plug-in elements have been summarized. The complete Plug-in elements and their respective content can be consulted in the HL7 Plug-in description of HIS-DF Web site (Left menu, tab "HL7 Plug-in"). Correspondently, an XMI processable description the Plug-in is found in [42].

5.5. *Configuring and publishing the HIS-DF method components*

According to the HIS-DF process described in Fig. 4, the practical outcome of the methodology (step 5) is the definition of a set of process components (roles, work products, artifacts, guidance, disciplines, phases, iterations and workflows) to independently addressing the Enterprise, Information and Computational VPs.

Following the RUP structure defined in Fig. 3, any software process is first organized by disciplines, which are described by activities. From the RM-ODP concept's mapping in Table 3, it can be concluded that the HIS-DF Enterprise Viewpoint is completely supported by the RUP Business Modeling discipline and the HIS-DF Information and Computational Viewpoints by

Table 3 – RM-ODP concepts and RUP method elements

ODP concepts	Related RUP concept	Related RUP artifact/guidance	RUP task	Responsible RUP role
Enterprise concepts				
Community	Target Organization	Business Vision	Set and Adjust Objectives	Business-Process Analyst
Community Objective	Business Goal	Artifact: Business Goal	Identify Business Goals	Business-Process Analyst
Process	Business Process	Business Use Case Model	Find Business Actors and Use Cases	Business-Process Analyst—BUCM Contributor
Enterprise Object	Business System	Artifact: Business Systems; Guideline: Business Systems	Business Architectural Analysis	Business Architect
Interaction	Interaction	Artifact: Business Use-Case Realization; Technique: Sequence Diagram in the Business Analysis Model	Business Use-Case Analysis	Business Designer
Policy	Business Rule	Artifact: Business Rule; Guideline: Business Rules	Identify Business Rules/Maintain Business Rules	Business-Process Analyst
Information concepts				
Invariant Schemata	Analysis Class (Analysis Model, Entity Classes only)	Artifact: Analysis Class; Guideline: Analysis Class	Use-Case Analysis; Step: Describe Attributes and Associations	Designer
Static Schemata	Object Diagram (Analysis Model, Object Diagram)	Artifact: Analysis Class; Guideline: Analysis Class	Use-Case Analysis.	Designer
Dynamic Schemata	State Machine Diagram (Design Model, only Statechart Diagram)	Artifact: Design Model; Guideline: Statechart Diagram	Task: Class Design; Step: Define States	Designer
Computational concepts				
Computational Object	Design Component, Design Subsystem	Artifact: Design Subsystem	Task: Identify Design Elements; Step: Identify Classes, Active Classes and Subsystems	Designer
Interface	Interface	Artifact: Interface	Task: Identify Design Elements; Step: Identify Subsystem Interfaces	Designer
Object Behavior	Design Component; Design Subsystem	Artifact: Design Subsystem; Guideline: Design Subsystem	Task: Subsystem Design; Step: Distribute Subsystem Behavior to Subsystem Elements	Designer

Table 4 – HL7 Plug-in method elements

HL7 concepts	Method content type	Description
HL7 Storyboard	Artifact	A Storyboard is a narrative description of a series of steps involving some exchange of information between different participants to achieve the objectives of a healthcare business process.
HL7 Use Case Model	Artifact	A set of Business Use Cases is developed for each single specification developed using the HDF.
HL7 Domain Analysis Model (DAM)	Artifact	Domain Analysis Model is defined for each single specification describing the static syntactic and semantic relationships of importance in the healthcare business process including the responsible parties/entities and the various data elements/structures required by the process.
Design Information Model (DIM)	Artifact	The DIM provides a solution to the information requirements of a particular problem domain.
Dynamic Design Model	Artifact	HL7 Dynamic Design Model describes collaborations and interactions between applications being integrated using HL7 standards. It is described in the HDF by UML State Charts, UML Activity Diagrams, System/Service Interfaces and Sequence Diagrams.
HL7 Domain Glossary	Artifact	A Domain Glossary is defined for each single specification describing the semantics of each item and attribute in the Domain Analysis Model (DAM).
HL7 Business Rule	Artifact	HL7 Business Rule is a declaration of policies or condition that must be satisfied in healthcare business processes. It is based on the RUP Business Rule Definition.
Information Model	Artifact	An information model is a structured specification of the information within a specific domain of interest.
Reference Information Model (RIM)	Artifact	The Reference Information Model is a static model of health and health care information as viewed within the scope of HL7 standards development activities.
Domain Message Information Model (D-MIM)	Artifact	D-MIM is a subset of the RIM that includes a fully expanded set of class clones, attributes and relationships that are used to create messages for any particular domain.
Refined Message Information Model (R-MIM)	Artifact	An information structure that represents the requirements for a set of messages. A constrained subset of RIM which may contain additional classes that are cloned from RIM classes.
Application Role	Artifact	Defined as an abstraction that expresses a portion of the messaging behavior of an information system.
Trigger Event	Artifact	A trigger event is an explicit set of conditions that initiate the transfer of information between system components (application roles).
Common Message Element Types (CMETS)	Artifact	CMET is a message type fragment that is reusable by other message types.
Vocabulary Domain	Artifact	A vocabulary domain is the set of all concepts that can be taken as valid values in an instance of a coded field or attribute of an information model.
Harmonize Design Models	Task	This task describes how to harmonize RUP artifacts with HL7 Reusable Assets. The objective is facilitating semantic interoperability by using standardized reference models.
HL7 Modeling Facilitator	Role	This role supports the development of the system's software architecture, specially concerned with the identification and reuse of HL7 information models in the Harmonization Task.

the RUP requirements, analysis and design disciplines. Hence, the HIS-DF defines also three disciplines (one per viewpoint) specializing the aforementioned RUP disciplines. To facilitate further reusability, for each HIS-DF discipline a capability pattern has to be defined.

The next step in the configuration is to define the task and related method elements for each HIS-DF discipline. This process is based on the relationships described in Tables 3 and 5. The RMC platform supports this procedure by providing HTML editors to establish the relationships between the existing RUP content, and the created HL7 and ODP Method Plug-ins elements. In the process configuration, the roles are directly linked to activities and the artifacts are linked to the activities as input or output results. Correspondently, guidances (concepts, checkpoints, tools) are connected to the input or output artifacts.

The resulting configuration is delivered as a Web site describing their process components and its interrelationships [40]. Fig. 5 offers a snapshot of HIS-DF configuration. The content is organized as three separated HIS-DF disciplines corresponding to each one of the ODP viewpoints (Enterprise, Computational and Informational). The content in each discipline is organized by groups of method elements (reference workflows, task, and guidance). The Web

Table 5 – HL7 concepts and RUP method elements

HL7 concepts	Related RUP concept	Related RUP method component	RUP discipline
HL7 Storyboard	Storyboard	Artifact: Storyboard; Guideline: Storyboard	Business Modeling; Requirements
HL7 Use Case Model	Business Use Case Model	Artifact: Business Use Case Model; Guideline: Business Use-Case Model; Guideline: Use-Case Diagram in the Business Use-Case Model	Business Modeling; Requirements
HL7 Domain Analysis Model (DAM)	Business Analysis Model	Artifact: Business Analysis Model; Guideline: Business Analysis Model	Business Modeling
Design Information Model (DIM)	Analysis Model	Artifact: Analysis Model; Artifact: Analysis Class; Guideline: Analysis Class	Design
Dynamic Design Model	Use-Case Realization	Artifact: Use-Case Realization; Guideline: Use-Case Realization	Analysis and Design
HL7 Domain Glossary	Business Glossary	Artifact: Business Glossary; Checklists: Business Glossary Template: Business Glossary	Business Modeling
HL7 Business Rule	Business Rules	Guideline: Business Rules; Artifact: Business Rule	Business Modeling
Information Model	Analysis Class	Artifact: Analysis Class	Analysis and Design
Reference Information Model (RIM)	Analysis Class	Artifact: Analysis Class	Analysis and Design
Domain Message Information Model (D-MIM)	Analysis Class	Artifact: Analysis Class	Analysis and Design
Refined Message Information Model (R-MIM)	Design Classes	Artifact: Design Class; Guideline: Design Class	Design
Application Role	Business Actors; Business Workers; Actors	Artifact: Business Actors; Artifact: Business Workers; Artifact: Actors	Business Modeling; and Design
Trigger Event	Business Event	Artifact: Business Event	Business Modeling; and Design
Common Message Element Types (CMETS)	Analysis Class	Artifact: Analysis Class	Analysis and Design
Vocabulary Domain	Analysis Class	Artifact: Analysis Class	Analysis and Design
Harmonize Design Models	Asset-Based Development	Artifact: Reusable Asset	Analysis and Design
HL7 Modeling Facilitator	Software Architect	Artifact: Reference Architecture; Artifact: Analysis Model; Artifact: Design Model	Analysis and Design

site (left menu at the bottom) also includes a complete description of each Plug-in's content.

6. Practical example using HIS-DF

After developing the HIS-DF, the demonstration of the practicability of the approach in HIS development project is underway. The National Public Health Surveillance Information System in Colombia (Sistema de Información para Vigilancia en Salud Pública – SIVIGILA [43]) has been selected as scenario for such HIS-DF feasibility study. SIVIGILA is part of the Colombian Health Information System, created with the objective to support the ongoing reporting of notifiable diseases (e.g., malaria, measles, hepatitis, yellow fever and 47 other diseases). For facilitating the notification process, several legacy applications independently developed at local or regional public health offices currently exist. Furthermore in 2006, the National Health Institute (Instituto Nacional de Salud, INS) as governmental institution in charge of SIVIGILA planning, development and implementation, developed a computer-based information system (named SIVIGILA2006) that is being distributed among the public health offices in the country. A new version of the system was released in early 2007.

None of the aforementioned developments considers the requirements for automatic semantic interoperability with other public health information systems (information systems to support vertical programs, e.g., promotion of care and prevention in tuberculosis, child information systems, etc.) or clinical information systems (e.g., electronic patient record systems). The HIS development project introduced in this section illustrates the adaptation and use of HIS-DF for the development of a new architecture for the SIVIGILA Information System.

6.1. *Configuring the development process*

The method configuration described in Section 5 is a generic repository of Method Content able to be adapted to the specific requirements of any software development project. Based on that, a software development project could define its own

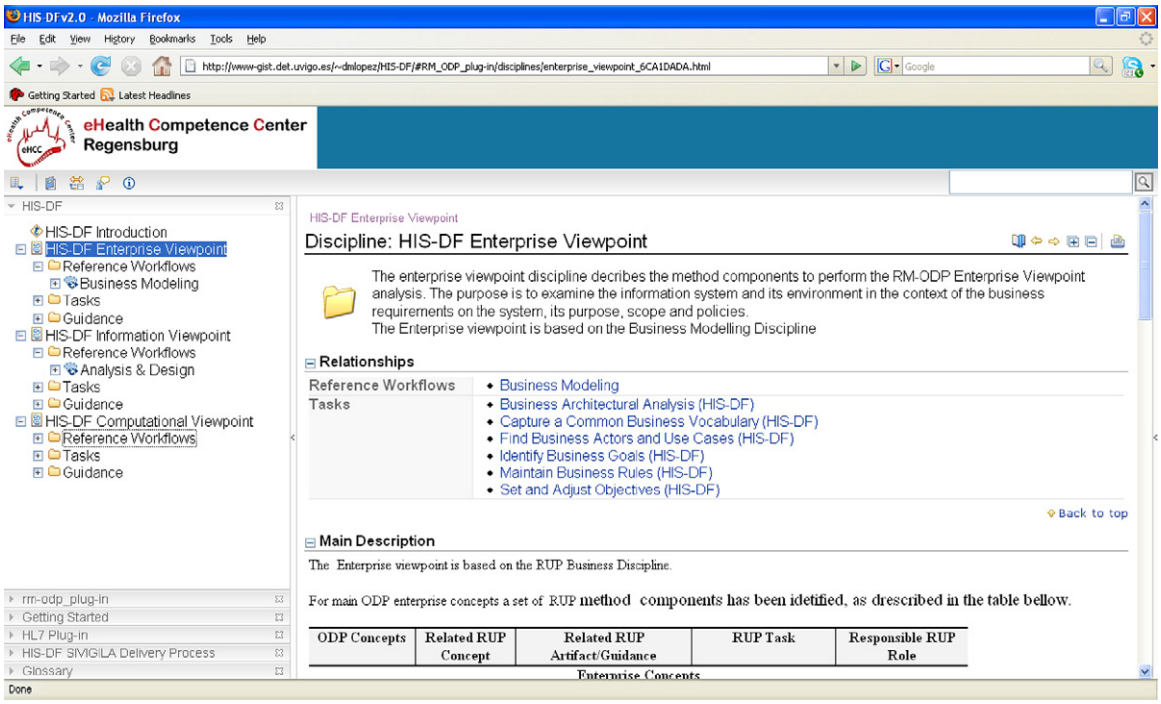

Fig. 5 – The HIS-DF deployed.

development process. For doing that, the HIS-DF Method Content would need to be integrated into phases and workflows which specify how the work shall be organized over the time for meeting the needs of the particular development project.

For the SIVIGILA example, a process configuration (definition of phases and workflows) was performed. In the hypothetic development project for the example, a small-sized development team is defined representing the System Analyst, System Architect and HL7 Modeling Facilitator roles. The RUP phases in the project are inception and elaboration. The Workflows are defined for the Enterprise Viewpoint, Information and Computational disciplines described in the HIS-DF configuration.

The RMC tool also supports that process configuration. It reuses the capability patterns (reusable process configurations) already created for each RM-ODP discipline, and integrates them into a process configuration (called delivery process). The delivery process created for the SIVIGILA example is called "HIS_DF_SIVIGILA" and has been included in the published HIS-DF Web site as reference (left menu under option: HIS_DF SIVIGILA delivery process). Note that the process has been configured for the SIVIGILA scenario. For another HIS development project, a new delivery process must be configured.

The HIS-DF SIVIGILA delivery process as it can be found at the HIS-DF Web site is presented in Fig. 6. The figure shows the hierarchy of RUP phases (inception and elaboration), disciplines (Enterprise, Information and Computational) and workflows sequentially organized in tasks. In the right section of the figure, two additional diagrams are shown. The Activity Diagram in the upper right corner describes the workflow (sequence of tasks) defined for the Enterprise Viewpoint discipline. The Flow Diagram at the bottom represents the work products and roles associated to the Harmonize Information Model Task.

Details on the activities and tasks as well as the associated artifacts, roles, and guidance can be found on the HIS-DF Web site. In the following subsections, some of the tasks and their output artifacts defined in the SIVIGILA process configuration (Fig. 6) will be detailed in the context of the proposed scenario. For a more complete reference, some examples – including those presented here – are added to the process configuration as example guidance. The examples are accessible in the HIS-DF Web site; directly from the related tasks descriptions.

6.2. *Enterprise Viewpoint*

The architectural analysis initiates defining the Enterprise Viewpoint. This view offers a perspective on the SIVIGILA system and its environment, describing the system's purpose, scope and policies. In terms of RUP workflows, it specialize a Business Modeling process where the business workflow at the public health organization in charge of the reporting of communicable diseases, is modeled.

One of the main activities when describing the business process is to clearly define the business objectives. This activity is performed in the task Set and Adjust Objectives (HIS-DF). In this task the artifact Business Vision Artifact is created, describing the project's target organization which is the National Public Health Surveillance System (SIVIGILA). The system consists of healthcare institutions, protocols, norms and resources organized with the objective of supporting the systematic and ongoing collection, analysis, interpretation, delivery and evaluation of health events necessary for health

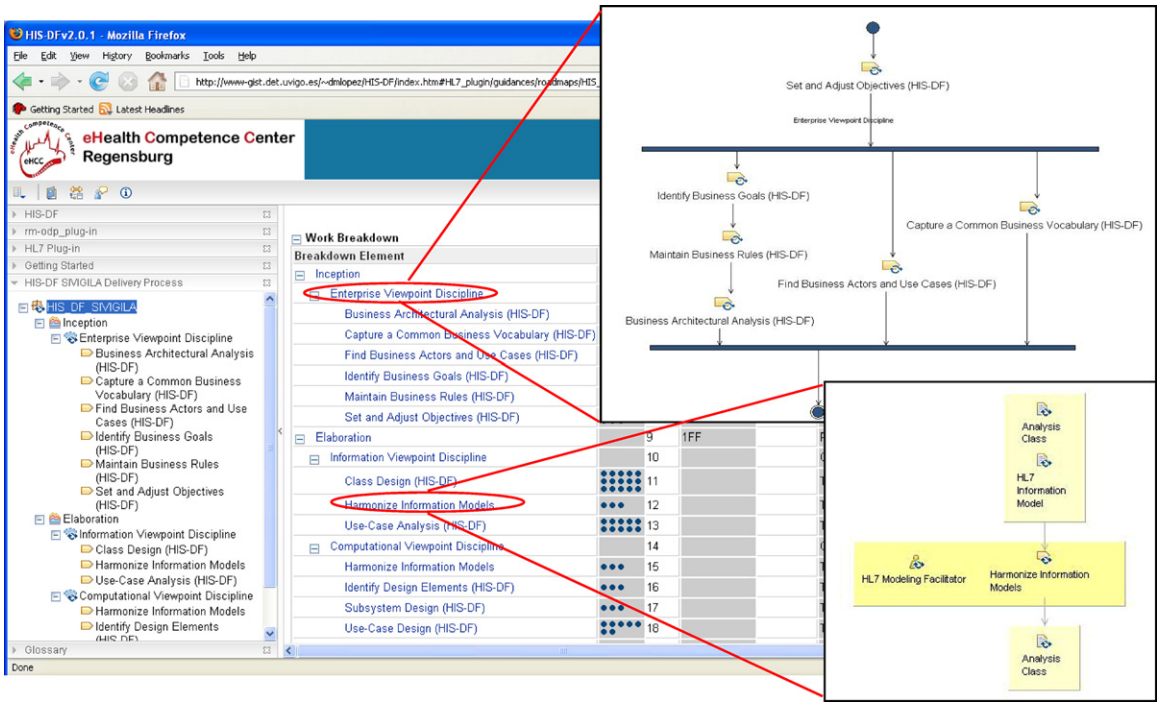

Fig. 6 – HIS_DF SIVIGILA delivery process.

promotion and prevention programs. The system's external actors are: Ministry of Health, Healthcare Service Providers, Public Health Authorities, Health Service Administrators, and any other organization interested in public health information for decision making.

Next, a formal representation of the business process is created using the UML Business Use Case Model developed in the Find Business Actors and Use-Cases task (HIS-DF). Fig. 7 represents the SIVIGILA's Business Use Cases Model. In the diagram, Health Services Providers (public and private institutions offering clinical services, e.g., hospitals, clinics and private doctors' offices) periodically send a report on events of public health interest (diseases, deaths, risk factors, public health interventions, etc.) to the respective Public Health Authority (Public Health Surveillance Offices in its geographical jurisdiction). SIVIGILA internally notifies that information to the respective offices (from local to regional via Public Health Surveillance Offices) and consolidates the information. The information is then analyzed and sent to different decision makers as surveillance protocols for supporting public health promotion, prevention and any other kind of health intervention.

6.3. *Information Viewpoint*

The Information Viewpoint offers a perspective on the information's structure and its semantics. For the sake of brevity, the architectural models presented in this paper

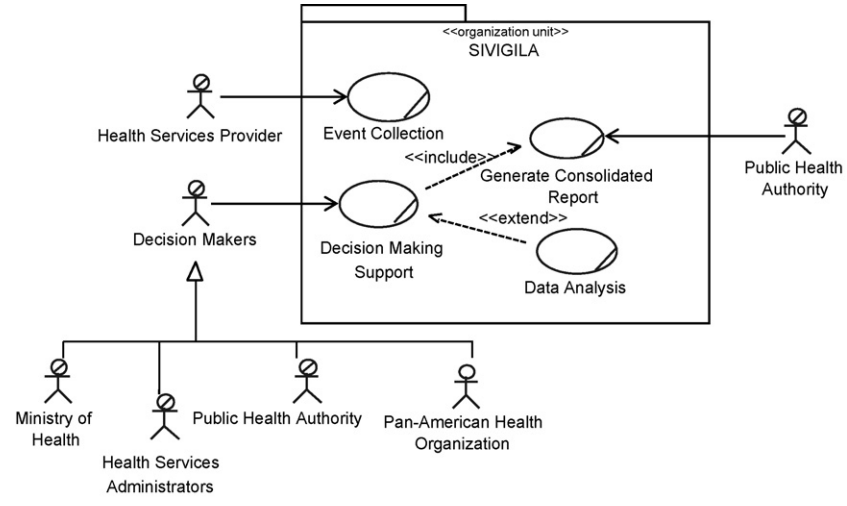

Fig. 7 – SIVIGILA Business Use Case Model.

are restricted to the Analysis Class artifact (Invariant Schemata).

The Analysis Class Model is derived from the aforedescribed Enterprise Viewpoint models and the functionality described in the Requirements discipline, concretely from the Use-Case Model. Due to the complexity of the business process, the architectural analysis of the SIVIGILA Information System is restricted to the "Event Collection" business process. For this process, the requirements for automatic semantic interoperability have been carefully analyzed, considering the need of information sharing with clinical information systems to automatically gather information about notifiable events from Electronic Health Records systems. Fig. 8 describes the high-level Use Case Model for the SIVIGILA Information System. In the system, an Individual Event Register is collected from an Electronic Health Record System (EHR-S) belonging to a "Healthcare Service Provider". Then, the system generates a consolidated report, which is sent to the Territorial Surveillance Assistant at the Territorial Public Health Office in its influencing area, i.e. a local, regional or national public health office. The Territorial Surveillance Assistant can also upload the consolidated report received from a lower-level Healthcare Service Provider in its jurisdiction (e.g., Regional Surveillance Authority receives from the Local Surveillance Authority) or send the report to his upper level territorial authority (e.g., the Regional Surveillance Assistant sends a consolidated report to the National Surveillance Authority).

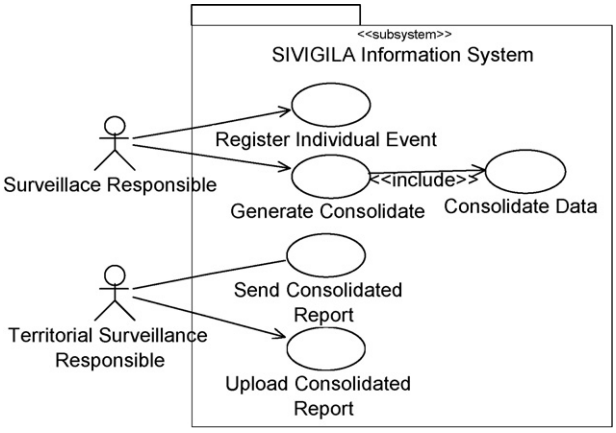

Fig. 8 – SIVIGILA high-level use cases.

Having described the System Use Cases, the Analysis Class Model is derived. This process is performed during the Use-Case Analysis task (HIS-DF). The resulting artifact, the UML Analysis is presented in Fig. 9. The model describes SIVIGILA main informational components. The main entities are "Public Health Event" representing information about the SIVIGILA events, "Person" representing the person associated to the public health case, and "Health Service" related to the attention service where the public health case were diagnosed.

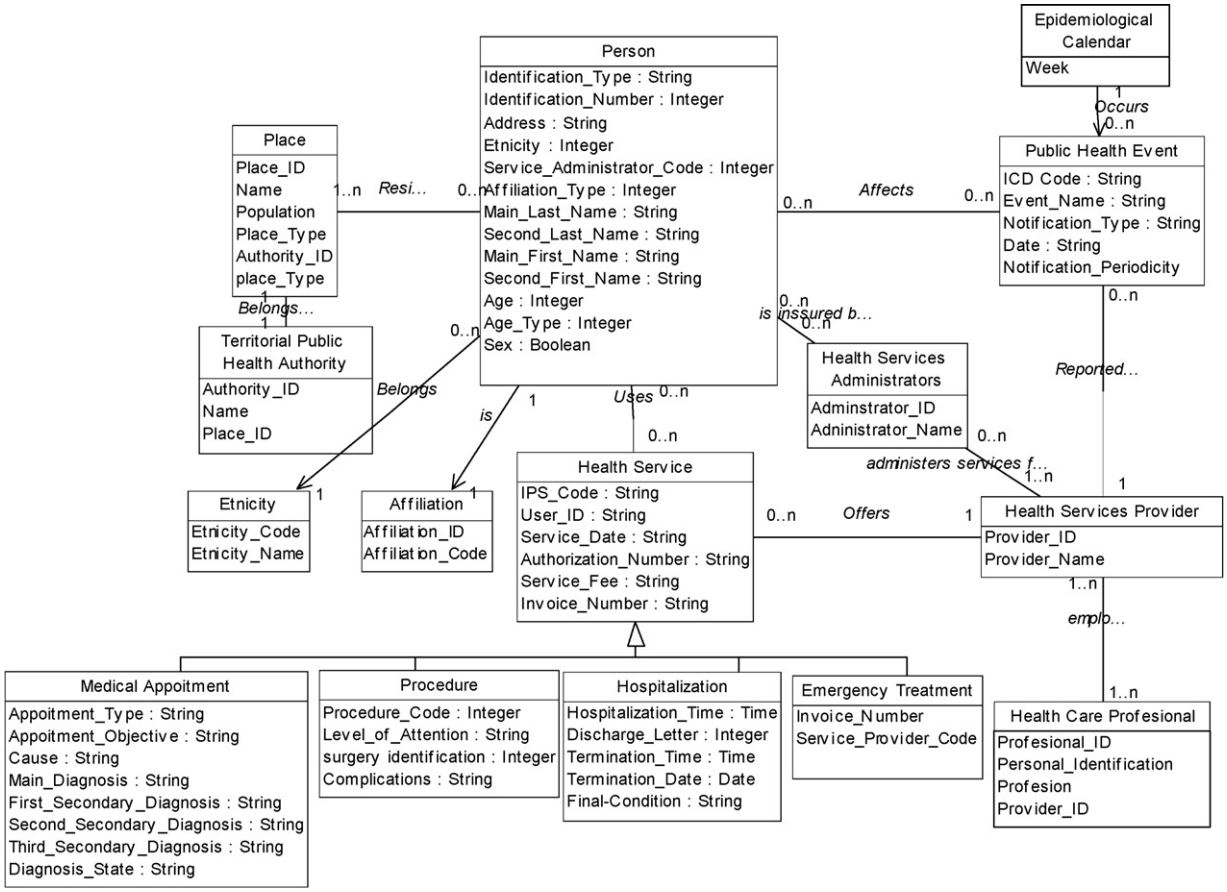

Fig. 9 – SIVIGILA System Analysis Class Model.

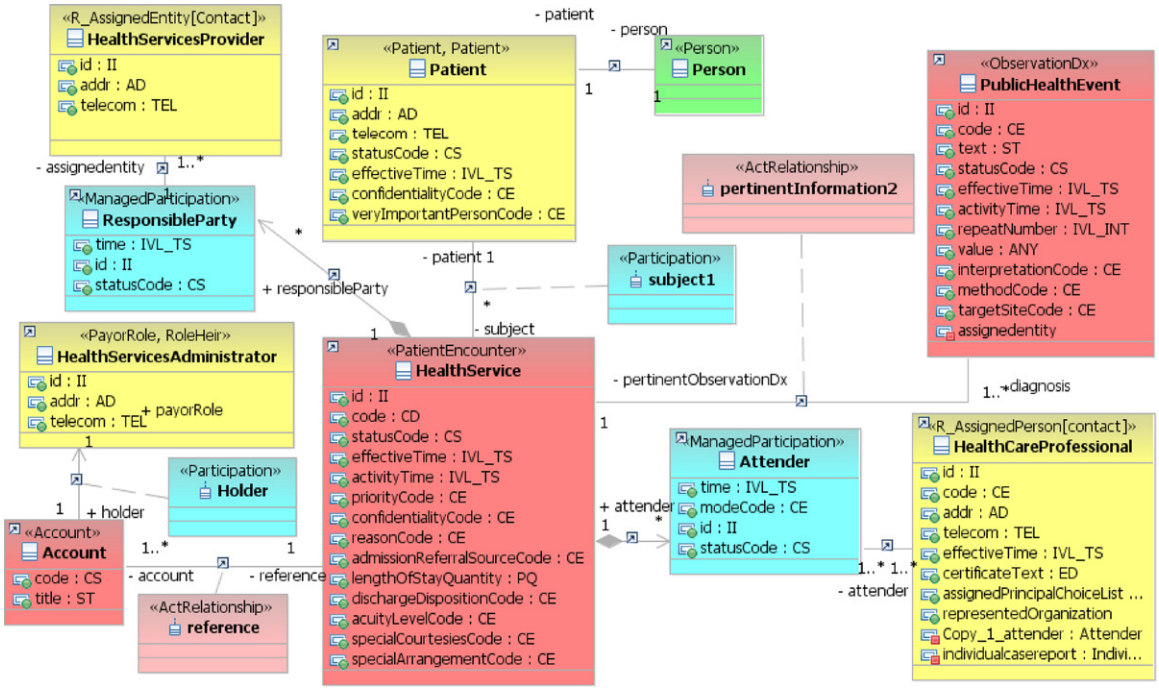

Fig. 10 – The HL7 Harmonized Model.

Finally, in order to demonstrate how the Analysis Models are harmonized with domain reference information models and vocabulary, the Task Harmonize Information Models (HIS-DF) is performed. The Domain Information Models for supporting the informational components are provided in HL7 specifications. Concretely, the Ambulatory Encounter Topic of the Patient Administration Universal Domain [44] has been specialized in the paper's context. Fig. 10 presents the same SIVIGILA System Analysis Class Model (Fig. 9) after being harmonized with the HL7 R-MIM classes. A detailed example of the harmonization process is presented in [45].

6.4. *Computational Viewpoint*

The Computational Viewpoint concerns the functional decomposition of the SIVIGILA Information System by specifying logical components and their interactions through interfaces. The architecture development methodology describes a top-down approach in which the computational components have been derived from the system use case descriptions according to the Enterprise Viewpoint. Also the process of harmonization of design components and interfaces defined in Task: Harmonize Design Models (HIS-DF) is performed.

The Design Component Model, developed during the Task Identify Design Elements (HIS-DF), describes the system design components at a high-level of abstraction. Fig. 11 shows the structural diagram for the SIVIGILA system. The diagram is not exhaustive, but depicts the main components and interfaces. A detailed description of the Design Component Model, including the use of HL7 services specifications to describe components and interfaces is presented in [46].

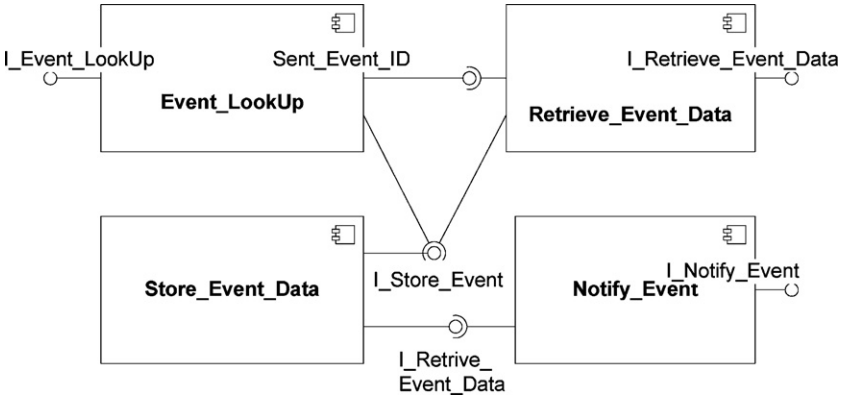

Fig. 11 – SIVIGILA component diagram.

7. Discussion

Starting with the requirements for semantic interoperability derived from paradigm changes for health systems and their supporting health information systems, the need of an architectural approach for analyzing, designing, implementing, and maintaining advanced, sustainable, semantically interoperable HIS has been shown. As the GCM's basic principles have been first published [47,48], it has been received only by small communities of experts due to the radical changes offered to the way of analyzing and designing health information systems at that time [49]. Successfully introducing and deploying those principles and methodologies in European projects (e.g., HARP [50]) or international initiatives such as ISO TC 215, CEN TC 251, HL7, the thoughts are going to find acceptance in those organizations, resulting in substantial improvements of related projects and specifications such as HL7's HDF, the HL7 Architecture Project, or several approved ISO and CEN standards (e.g., EN 12967 [26], EN 13606-4 [51], ISO 22600 [52]).

Combining the GCM with those advancing software process engineering methods, an appropriate architectural development methodology for developing process-oriented and model-driven solutions has been provided.

The architecture development methodology offers a reference knowledge base of tasks, performers (roles), work products (artifacts), guidance, phases, workflows, etc., necessary for the architectural analysis within healthcare software development projects, where semantic interoperability is one of the main requirements. At a higher level of abstraction, the methodology is based on the principles of systems theory borrowed from the Generic Component Model. This approach allows for describing a Health Information System as a set of components reducing the system's complexity but also considering the inter-relation between systems components expressing behavior. This approach is different from many architectural approaches which normally describe only structural aspects of an information system. This formal analysis method also deals with the inherent complexity of HIS by separately considering different domains of interest (not only medical or technical ones) and offering the possibility of abstraction focusing on specific aspects of the system or views. The RM-ODP provides a comprehensive system abstraction by defining viewpoints and a correspondent language for describing components. MDA establishes a process to move from business descriptions over platform-independent aspects of the systems to specific technology details, not addressing the detailed business analysis, however. The IEEE 1471-2000 methodology provides an interesting framework for describing architecture without clear methodologies and migration proposals for reusing existing domain knowledge. All these aspects, found essential in the evaluation of state of the art architectural approaches, are uniquely combined in the HIS-DF.

The weakest aspect in the evaluated architectural approaches for HIS development, as already discussed in Section 4, was the lack of a formal architecture development methodology covering the complete architecture lifecycle, and a clearly defined development process describing tasks, work products, roles, workflows, etc. The RUP was found the most comprehensive source of methods and processes for system development, including architecture development. This development process constitutes the better approach to complete the HIS-DF, providing details on how to deliver a sound architecture description. HIS-DF tasks, responsible persons, products, guidance, phases, and workflows are described specializing (tailoring) the RUP process. RUP facilitates the flexibility, scalability and reusability of the methodology by describing method components and providing guidance and tooling for creating reusable Method Content (RUP Plug-ins) and documenting the methodology through exportable Web pages and XMI files. Note that the Plug-ins it themselves are no more that s series of documents (guidelines providing definitions, reference to artifacts, activities or examples). The advantage using the RUP tooling is that they can be viewed (as HTML pages) and shared (as XML files) among software development teams. It is the responsibility of the development team however, to read and follow those guidelines in the course of its role developing a system. The advantage is that using the proposed reusable and unified development process, the development of consistent models and use of the same guidelines is enforced, therefore supporting the design of semantically interoperable models and systems.

Another important aspect in the methodology is the orientation towards specifying semantically interoperable HIS. The framework is able to integrate existing domain knowledge, e.g., reference models, vocabulary, services specifications and standards into the development process, in order to enhance the expressivity of the resulting architectural models. HL7 specifications have been partially re-used because they are nowadays the richest source of reference for medical and public health knowledge. Furthermore, it is the only approach that progressively supports enterprise and computational specifications additionally to the Information Viewpoint. However, the method is not restricted to HL7 specifications. Other standards such us the CEN 13606 EHR Communication standard, OpenEHR Archetypes, CEN ENV 12967, OMG HDTF, etc., are alternative/complementary candidate sources of domain knowledge. This knowledge can be progressively added to the methodology as new plug-ins or new configurations. In the configuration presented in this paper, HL7 contributes to the specification of informational aspects of a system with standard information models such as RIM, DIMs, R-MIMs and CMETS, and a rich domain vocabulary referencing to HL7-recognized external coding schemes, e.g., Logical Observation Identifiers Names and Codes (LOINC) or the Systematized Nomenclature of Medicine (SNOMED). The enterprise specification is supported by HL7 work products such as Domain Analysis Models (DAMs), Storyboards, Use Case Models, Activity Diagrams, Domain Glossary and Business Rules. Computational specifications are supported by Application Roles, Trigger Events, messages specifications and, as functional requirements, by recent HL7's Service-Oriented Architecture services specifications Entity Identification Service (EIS), Retrieve Locate Update Service (RLUS), Common Terminology Service (CTS) and Decision Support Service (DSS).

While the proposed approach opens new pathways for standardized, comprehensive and customizable development methodologies in the domain of health care, it cannot be

excluded that similar ideas might be piloted in other domains as well. Regarding the generic way of reusing state of the art software engineering methods and tools, the approach of developing systems based on components and services, using domain reference information models, standards and metadata, referring to domain vocabulary, terminology and ontology is domain-independent. Hence, the healthcare context (e.g., [24,53–55]) and other application domains (e.g., e-government [56], e-learning [57], environmental and geospatial sciences [58], civil engineering [59], etc.) are currently struggling with the challenge for appropriately developing semantically interoperable solutions. However, the HIS-DF contributes with new opportunities for analyzing, evaluating, adopting and combining existing models and consistently creating new ones for representing healthcare-specific and healthcare-independent knowledge. At our knowledge, a formal and comprehensive methodology for designing semantically interoperable information systems at the level of details presented in the paper has not been provided so far. Moreover, none of the publicly available RUP Plug-ins [60] (which are RUP specializations for different purposes) are devoted to cover development projects in specific application domains, except perhaps the RUP4STP Plug-in for financial and administrative systems and the secure application development Plug-in. In the healthcare context, traditional RUP has successfully been used as development methodology in several IT healthcare projects, e.g., the HYGEIAnet project [61] and the Electronic Nursing Record System in Korea [62]. Other methodological approaches in healthcare specialize some of the principles of the RUP, e.g., the layered approach for system evolution [63] and the methodology for medical data warehouses [64]. However RUP has not been used or configured before, as a process to describe generic system architectures as detailed in this paper.

Finally, regarding the significance of the described results, the method can be incorporated into any system development project with the objective to develop new information systems and components, family of systems, generic architectures, integration of existing systems, etc. HIS-DF is also useful when developing platform-independent architectures that will further be reused for implementing systems and components in specific implementation technologies. This will be very helpful for national/regional/local information systems. Furthermore, the architectural approach facilitates the adoption of healthcare standards and advanced architectural approaches in health care IT development projects. Such approach is an opportunity especially for small-sized development projects with restricted resources.

The architecture described as example suggested several opportunities to enhance the SIVIGILA system by providing efficient, high quality and sustainable public health services. One opportunity is that the architecture can be completed and used as technical specification (platform-independent specification) to normalize all information systems developments in the country which are often autonomous projects not considering interoperability. Another possibility is to go further and implement the technical specification of SIVIGILA by transferring the architecture to platform-specific models using MDA, thus providing open software components and services that can be reused in local/regional developments. The presented models are currently under evaluation in real systems: Negotiations with the National Health Institute in Colombia (INS) could lead to some pilot applications which improve the current IT technology on the offered basis soon. In the mean time, the research continues towards the Technology Architecture implementing services and covering other architectural issues such as maintenance, adaptation, governance, etc.

Summary points

What is already known on the topic?

- There are diverse approaches in systems and software engineering disciplines for analyzing, designing, implementing and evaluating more or less interoperable information systems.
- An unsolved challenge consists in the consistent integration of different domains' knowledge by harmonizing concept representation, languages and methodologies among domain experts involved in an information system development process.
- Public Health Information Systems in Colombia are mainly legacy applications independently developed at local or regional public health offices, not supporting interoperability at the semantic level.

What has been added by the paper?

- The evaluation of architectural approaches for HIS development demonstrated a lack of a formal and comprehensive architecture development methodology covering the complete architecture lifecycle, and clearly defining activities, artifacts, roles, workflows, etc.
- The paper provides a formal methodological framework which is consistent, comprehensive, customizable, scalable, and reusable enough to support development teams in the design of HIS architectures. The framework is available as HTML and XMI documents.
- HIS-DF facilities the adoption of healthcare standards and advanced architectural approaches in health care IT development projects being an opportunity especially for small-sized development projects with restricted resources.
- The sample architecture suggests several opportunities to enhance the Colombian Public Health Information System by providing efficient, high quality and sustainable semantically interoperable public health services.

8. Conclusion

Semantic interoperability is a basic challenge to be met for new generations of distributed, communicating and co-operating health information systems enabling shared care and e-Health. In the paper, existing architectural

approaches were analyzed, compared and finally harmonized towards a methodology for semantically interoperable health information systems and components development. The methodological framework provides a comprehensive and customizable knowledge base of tasks, performers, artifacts, and guidance; necessary for the architectural analysis within healthcare software development projects. The framework is able to integrate existing domain knowledge, e.g., reference models, vocabulary, services specifications and standards into the development process, in order to enhance the expressivity of the resulting architectural models. The feasibility of the approach was exemplified in the specification, analysis and design of a public health information system.

Acknowledgments

The authors are indebted to thank partners from HL7, OMG and other Standards Development Organizations for collegial collaboration. It is a pleasure also to thank the public health authorities "Instituto Nacional de Salud" and "Dirección Departamental de Salud del Cauca" in Colombia for support and kind co-operation.

The authors would also like to thank Jan Talmon and the anonymous reviewers who contributed a lot clarifying and streamlining the original manuscript.

This work was partially funded by the German Academic Exchange Service DAAD, the Bavarian Research Foundation, the Program Alban (European Union Programme of High Level Scholarships for Latin America), and the University of Cauca.

Contributions: DML wrote the first draft and collected the data. Both authors took part in the conception and design of the study, analysis and interpretation of data, revising it critically for important intellectual content, and revision and approval of the version to be submitted.

REFERENCES

[1] J. Siegel, Quick CORBA 3, John Wiley & Sons, New York, 2001.
[2] ISO DTR 20514, Health informatics—electronic health record: definition, scope and context, 1st ed., International Organization for Standardization, 2005.
[3] B. Blobel, Advanced EHR architectures—promises or reality, Methods Inf. Med. 45 (1) (2006) 95–101.
[4] P. Checkland, Systems Thinking, Systems Practice, John Wiley & Sons, Chichester Sussex, 1981.
[5] L. Bertalanffy, General Systems Theory: Foundations, Development, Applications, Braziller, New York, 1968.
[6] P.J. Lewis, Information-Systems Development. Systems Thinking in the Field of Information-Systems, Pitman, London, 1994.
[7] C. Churchman, R.L. Ackoff, E.L. Arnoff, Introduction to Operations Research, John Wiley & Sons, London, 1957.
[8] M.J. Culnan, B.E. Swanson, Research in management information systems, 1980–1984: points of work and reference, MIS Q. 10 (3) (1986) 289–302.
[9] J. Liebenau, J. Backhouse, Understanding Information. An Introduction, MacMillan, London, 1990.
[10] N. Ahituv, S. Neumamm, Principles of Information Systems for Management, Wm. C. Brown Company Publishers, Dubuque, 1990.
[11] V. Zwass, Foundations of Information Systems, Irwin/McGraw Hill, Boston, 1998.
[12] M. Chiasson, M. Reddy, B. Kaplan, E. Davidson, Expanding multi-disciplinary approaches to healthcare information technologies: what does information systems offer medical informatics? Int. J. Med. Inform. 76 (Suppl. 1) (2007) S89–S97.
[13] B. Blobel, Concept representation in health informatics for enabling intelligent architectures, in: A. Hasman, R. Haux, J. van der Lei, E. De Clercq, F. Roger-France (Eds.), Ubiquity: Technology for Better Health in Aging Societies, vol. 124, IOS Press, Amsterdam, 2006, pp. 285–291.
[14] B. Blobel, Application of the Component Paradigm for Analysis and Design of Advanced Health System Architectures, Int. J. Med. Inform. 60 (3) (2000) 281–301.
[15] B. Blobel, Analysis, Design and Implementation of Secure and Interoperable Distributed Health Information Systems, IOS Press, Amsterdam, 2002.
[16] ITU-T Recommendation X.902 | ISO/IEC IS 10746-2, Information Technology—Open Distributed Processing-Reference Model: Foundations, International Organization for Standardization, 1996.
[17] ANSI/IEEE 1471-2000, Recommended Practice for Architectural Description of Software-Intensive Systems, Institute of Electrical and Electronics Engineers, 2000.
[18] L. Bass, P. Clements, R. Kazman, Software Architecture in Practice, 2nd ed., Addison-Wesley, Boston, 2003.
[19] J. Zachman, A framework for information systems architecture, IBM Syst. J. 38 (2–3) (1999) 454–471.
[20] P. Mikhail, R. Caspar, T. Zahir, The impact of software development strategies on project and structural software attributes in SOA, Lect. Notes Comput. Sci. 3762 (2005) 442–451.
[21] ISO EN 13606-1, Health Informatics—Electronic Health Record Communication—Part 1: Reference Model, International Organization for Standardization, Geneva, 2007.
[22] OpenEHR Foundation (homepage on the Internet), available from: http://www.openehr.org (cited 2008 May 18).
[23] 7HL Inc. (homepage on the Internet), CDA Clinical Document Architecture Release 2.0, available from: http://www.hl7.org/v3ballot/html/infrastructure/cda/cda.htm (updated 2008 March 23; cited 2008 May 18).
[24] B. Blobel, K. Engel, P. Pharow, Semantic interoperability—HL7 Version 3 compared to advanced architecture standards, Methods Inf. Med. 45 (4) (2006) 343–353.
[25] M. Eichelberg, T. Aden, J. Riesmeier, A. Dogac, G. Laleci, A survey and analysis of electronic healthcare record standards, ACM Comput. Surv. 37 (2005) 277–315.
[26] CEN EN 12967, Health informatics—Healthcare Information System Architecture, European Committee for Standardization, 2007.
[27] I. Jacobson, G. Booch, J. Rumbaugh, The Unified Software Development Process, Addison-Wesley, Reading, 1999.
[28] P. Kruchten, The Rational Unified Process. An Introduction, 3rd ed., Addison Wesley, Boston, 2003.
[29] OMG (homepage on the Internet), Software Process Engineering Metamodel Specification, v2.0. Draft adopted specification, available from: http://www.omg.org/docs/ptc/07-02-01.pdf (cited 2008 May 18).
[30] T. Licong, L. Zhang, Z. Bosheng, Q. Guanqun, A gradually proceeded software architecture design process, Lect. Notes Comput. Sci. 3840 (2005) 192–205.
[31] R. Kazman, P. Kruchten, R.L. Nord, J.E. Tomayko, Integrating Software-Architecture-Centric Methods into the Rational Unified Process, Technical Report, Software Engineering Institute, Pittsburg, Carnegie Mellon University, Report No.: CMU/SEI-2004-Tr-011, 2004.

[32] R. Kazman, M. Klein, P. Clements, ATAM: Method for Architecture Evaluation, Technical Report, Software Engineering Institute, Pittsburg, Carnegie Mellon University, Report No.: CMU/SEI-2000-TR-004, ADA382629, 2000.
[33] J. Miller, J. Mukerji (Eds.), MDA Guide Version 1.0.1, Object Management Group, Framingham, 2003.
[34] A. Gavras, M. Belaunde, L. Ferreira, J. Almeida, Towards an MDA-based development methodology, Lect. Notes Comput. Sci. 3047 (2004) 230–240.
[35] T. Blevins, J. Spencer, F. Waskiewicz, TOGAF ADM and MDA® (monograph on the Internet), available from: http://www.opengroup.org/cio/MDA-ADM/MDA-TOGAF-R1-070904.pdf (cited 2008 May 18).
[36] 7HL Inc. (homepage on the Internet), HL7 Development Framework, available from: http://www.hl7.org/v3ballot/html/help/hdf/hdf.htm (cited 2008 May 18).
[37] B. Blobel, R. Nordberg, J.M. Davis, P. Pharow, Modelling privilege management and access control, Int. J. Med. Inform. 75 (8) (2006) 597–623.
[38] IBM Corp. (homepage on the Internet), Rational Unified Process, available from: http://www-306.ibm.com/software/awdtools/rup (cited 2008 May 18).
[39] EPF (homepage on the Internet), Eclipse Process Framework Project (EPF), available from: http://www.eclipse.org/epf (cited 2008 May 18).
[40] HIS-DF (homepage on the Internet), Health Information Systems Development Framework Web site, available from: http://www-gist.det.uvigo.es/~dmlopez/HIS-DF/ (cited 2008 May 18).
[41] ODP_Plug-in (homepage on the Internet), HL7_Plug_in for RUP, available from: http://www-gist.det.uvigo.es/~dmlopez/ODP_Plug_in.zip (cited 2008 May 18).
[42] HL7_Plug-in (homepage on the Internet), HL7_Plug_in for RUP, available from: http://www-gist.det.uvigo.es/~dmlopez/HL7_Plug_in.zip (cited 2008 May 18).
[43] Ministerio de la Protección Social, Decreto 3518 de 2006, Sistema de Vigilancia en Salud Pública, Ministerio de la Protección Social, República de Colombia, 2006.
[44] 7HL Inc. (homepage on the Internet), Domain: Patient Administration, available from: http://www.hl7.org/v3ballot/html/domains/uvpa/uvpa.htm (cited 2008 May 18).
[45] D.M. Lopez, B. Blobel, Formal design of electronic public health records, in: L. Bos, L. Roa, K. Yogesan, B. O'Connell, A. Marsh, B. Blobel (Eds.), Medical and Care Compunetics 3. Series Studies in Health Technology and Informatics, vol. 121, IOS Press, Amsterdam, 2006, pp. 337–348.
[46] D.M. Lopez, B. Blobel, Connecting public health and clinical information systems by using a standardized methodology, in: K.A. Kuhn, J.R. Warren, T. Leong (Eds.), MEDINFO 2007, IOS Press, Amsterdam, 2007, pp. 132–136 (Best Student Paper Award).
[47] B. Blobel, M. Holena, Comparison, evaluation, and possible harmonisation of the HL7, DHE, and CORBA middleware, in: J. Dudeck, B. Blobel, W. Lordieck, T. Bürkle (Eds.), New Technologies in Hospital Information Systems, vol. 45, IOS Press, Amsterdam, 1997, pp. 40–47.
[48] B. Blobel, Assessment of middleware concepts using a generic component model, in: Proceedings of the Conference Toward An Electronic Health Record Europe'97, London, 1997, pp. 221–228.
[49] B. Blobel, F. Roger-France, A systematic approach for analysis and design of secure health information systems, Int. J. Med. Inform. 62 (3) (2001) 51–78.
[50] B. Blobel, G. Stassinopoulos, P. Pharow, Model-based design and implementation of secure, interoperable EHR systems, in: M.A. Musen, C.P. Friedman, J.M. Teich (Eds.), AMIA 2003 Symposium Biomedical and Health Informatics: From Foundations to Applications, American Medical Informatics Association 2003 Proceedings, Bethesda, 2003, pp. 96–100.
[51] CEN EN 13606-4, Health informatics—Electronic Health Record Communication—Part 4: Security, European Committee for Standardization, Brussels, 2007.
[52] ISO TS 22600, Health Informatics: Privilege Management and Access Control, Part 2: Formal Models, International Organization for Standardization, Geneva, 2006.
[53] K. Bernstein, M. Bruun-Rasmussen, S. Vingtoft, S.K. Andersen, C. Nohr, Modelling and implementing electronic health records in Denmark, Int. J. Med. Inform. 74 (2–4) (2005) 213–220.
[54] J. Mykkanen, J. Porrasmaa, J. Rannanheimo, M. Korpela, A process for specifying integration for multi-tier applications in healthcare, Int. J. Med. Inform. 70 (2–3) (2003) 173–182.
[55] J. Phillips, R. Chilukuri, G. Fragoso, D. Warzel, P.A. Covitz, The caCORE Software Development Kit: streamlining construction of interoperable biomedical information services, BMC Med. Inform. Decis. Mak. 6 (2006) 2.
[56] L. Guijarro, Interoperability frameworks and enterprise architectures in e-government initiatives in Europe and the United States, Gov. Inform. Q. 24 (1) (2007) 89–101.
[57] D. Dagger, A. O'Connor, S. Lawless, E. Walsh, V.P. Wade, Service-oriented e-learning platforms: from monolithic systems to flexible services, IEEE Internet Comput. 11 (3) (2007) 28–35.
[58] R. Peachavanish, H.A. Karimi, B. Akinci, F. Boukamp, An ontological engineering approach for integrating CAD and GIS in support of infrastructure management, Adv. Eng. Inform. 20 (1) (2006) 71–88.
[59] Q.Z. Yang, Y. Zhang, Semantic interoperability in building design: methods and tools, Comput. Aided Des. 38 (10) (2006) 1099–1112.
[60] IBM Corp. (homepage on the Internet), Rational Process Library, The industry's most robust collection of best practices guidance, available from: http://www-306.ibm.com/software/awdtools/rmc/library/ (cited 2008 May 18).
[61] M. Tsiknakis, D.G. Katehakis, S.C. Orphanoudakis, An open, component-based information infrastructure for integrated health information networks, Int. J. Med. Inform. 68 (1–3) (2002) 3–26.
[62] H.A. Park, I. Cho, N. Byeun, Modeling a terminology-based electronic nursing record system: an object-oriented Approach, Int. J. Med. Inform. 76 (10) (2007) 735–746.
[63] R. Lenz, K.A. Kuhn, Towards a continuous evolution and adaptation of information systems in healthcare, Int. J. Med. Inform. 73 (1) (2004) 75–89.
[64] N.B. Szirbik, C. Pelletier, T. Chaussalet, Six methodological steps to build medical data warehouses for research, Int. J. Med. Inform. 75 (9) (2006) 683–691.