Robust design of web services supporting the home administration of drug infusion in pediatric oncology

Chiara Parretti*[1], *Elaheh* Pourabbas[2], *Fernando* Rolli[1], *Fabrizio* Pecoraro[3], *Paolo* Citti[1], *Alessandro* Giorgetti[1]

[1]Department of Innovation and Information Engineering, Guglielmo Marconi University, Via Plinio 44 - 00193 Rome, Italy
[2]Institute for System Analysis and Computer Science "A. Ruberti", National Research Council, Via dei Taurini, 19 - 00185 Rome, Italy
[3]Institute for Research on Population and Social Policies, National Research Council, Via Palestro, 32 - 00185 Rome, Italy

Abstract. Cancer home care is a sector of particular relevance for the Italian health service. The budget ceilings imposed by public finance pose the need to reduce hospitalisation costs, moving patients from treatment to home care as much as possible. This is especially true in the field of pediatric oncology, where a protected family environment offers a variety of benefits to young patients and their families. However, in order to guarantee adequate assistance services, an integrated information system for management must be devised as the center of gravity of a coordinated network of, even heterogeneous, actors. This paper focuses on the identification and evolving development of web services for regulation and control of home administration of drug infusion in pediatric oncology. The Service Oriented Architecture (SOA) remains the software architecture of reference, whereas the methodological approach is open to a reconsideration of the continuous improvement of the whole process in light of the ceaseless progress of medical science and information technology. In this paper, his goal is achieved by using a methodological design approach that combines the UML modeling based on Case Stories and the process optimization of Axiomatic Design. Case Stories allow the formalization of end users' requirements in UML language, easily interpreted by software developers. On the other hand, Axiomatic Design allows optimizing the design process by identifying the most robust solutions in terms of greater logical coherence and lesser systemic complexity. We analyze a case study focused on the design of web services supporting the home administration of drug infusion, for young cancer patients in home care. Then, we show how the UML-modeling methodology can be used to design new services on the basis of specific Use Cases. Finally, through the Axiomatic design approach we verify the proposed solutions, by identifying the robust set of web services to be implemented.

1 Introduction

1.1 Premise

Cancer home care is an important field for the national health service. The spending limits imposed by public finance pose the need to reduce in-patient costs in hospitals, relocating home care as much as possible [1]. This is even more true in the field of pediatric oncology, where a protected family setting offers several benefits to young patients and their families. In order to guarantee suitable care services, however, it is necessary to devise an Integrated Information Management System [1-3]. This can be achieved by developing an open platform of services, constituting the center of gravity of a well coordinated network of heterogeneous actors [4-6]. The primary purpose of such a network is the provision of medical, social and psychological interventions with a service level adequate to the gravity of clinical cases, which unfortunately can also be, extreme. In this study, we tackle the application of a methodology of design and implementation of information systems as a valid communication tool among these actors. In this context, the Service oriented Architecture (SOA) remains the software architecture of reference [7, 8], whereas the methodological approach is open to a reconsideration of the continuous improvement of the whole process in light of the ceaseless progress of medical science and information technology.

1.1 Scope

1.1.1 Context

The analyzed case study focuses on the design of web services supporting the home administration of drug infusion, in home care, for young cancer patients being treated at the Gianna Gaslini Institute[a] in Genoa. This study is part of the Health@Home (H@H) Project promoted by the National

[a] https://www.gaslini.org/

Research Council (CNR) of Italy and financed by the Ministry of Education, Universities and Research [7]. The general objective of H@H is creating a network of integrated health and social services and interoperable devices/systems to better improve the *quality of life* of people/families both with difficulties and with any particular problems, through solutions, which guarantee the greatest possible autonomy within their own environment. The experience of the Gaslini Institute is an authentic best practice considering the similar centers operating in health systems with universal coverage welfare. Thus, such experience has been posed as object of our analysis.

1.1.2 Project phases

The H@H Project has been launched by proposing a methodology, based on Unified Modelling Language (UML), to identify health and social care web services on the basis of Case Stories. [7, 8]. In this paper, this methodology is also used to study the current process of home administration of drug infusion at the Gaslini Institute. Thus, it is required the to identify the web services necessary to implement the overall mechanism of data exchange in the H@H platform. At this point, it seems appropriate to use the Axiomatic Design optimization procedure, so that starting from system use cases a robust set of web services to be implemented could be defined. Both above-mentioned methodologies are combined in order to be applied in sequence. The first design phase, called "As is", represents the picture of the non-optimized process. Then, the application of the Axiomatic Design follows, which leads to the identification of an optimized set of web services (To be phase).

Table 1. Glossary.

Abbreviation	Definition
AD	Axiomatic Design
CA	Customer Attribute
DP	Design Parameter
FR	Functional Requirement
H@H	Health@Home Project
PV	Process Variable
R	Reangularity
SOA	Service-Oriented Architecture
UML	Unified Modeling Language
WS	Web Service

1.1.3 Process optimization

The proposed method led, also, the rethinking of the entire administration of the in-home medication regimen process; in this case, the definition of the H @ H platform constitutes an essential technological lever to allow the use of different services for patients [7, 8]. This platform cannot only transmit web services of direct assistance to cancer patients, but can also support access to various socio-educational services. SOA is an open type architectural protocol, which makes the system integrable and interoperable with a myriad of Web services already available or easily implemented. In this context, Axiomatic Design can play a dual roleInnanzitutto, può ridefinire le attività del processo di somministrazione di farmaci anche in termini di ottimizzazione snella [9-13]. Secondly, this operational definition is the premise for identifying the set of robust web services to be implemented. The following paragraphs will emphasize the process of breaking down user requirements. Each level of functional decomposition allows, through a top-down logic, to first identify the web services to be offered to young patients, then the elementary actions of which they are composed. Finally, subsequent breakdowns may make it possible to construct the conceptual design of the system to be entrusted to the work of the developers.

2 Methodology

2.1 Definitions

In this paper, we will refer to terms such as case study, case story and use case. A case study is a factual representation of what happened along with some analysis that provides insights and learning for the future [13]. In the social and life sciences, a case study is a research method involving in-depth and detailed examination of a subject of study (the case), as well as its related context conditions [13, 14]. Whereas, a case story depicts what happened through people, place, and plot and brings emotional context into the portrayal of what happened. Finally, a use case is a list of actions or event steps typically defining the interactions between a role (known in - UML as an actor) and a system to achieve a goal [15]

2.2 Methodological approach

The proposed methodology is based on using the UML modeling techniques on Case Stories and on a process of axiomatic decomposition of functional requirements of the system [16]. The main purpose of the system analysis methodologies referring to the Use Cases aim to capture the system requirements to be implemented from the real world.

Therefore, as a first step, Case Stories are built. They describe the system functioning from the involved user-actor's point of view [8, 17]. They are then combined, analyzed and translated into UML language-type Use Cases [18]. This level of formalization allows us to identify the interactions among the parties and to define an accurate sequence diagram. At this stage, it is already possible to identify web services for applying to the implementation or re-design. However, without a proper optimization process, the result would be highly related to the software designers' skill and experience [19, 20]. For this reason, it is proposed to improve the UML modeling techniques, turning to a selection of web services to be implemented or re-designed on a quantitative basis [20, 21]. In this sense, axiomatic design provides us with a powerful assessment tool, based on

a solid theoretical basis, consisting of axioms and corollaries, easily applicable[22-24]. In fact, the relations among the different components of the system are reported in terms of design matrices. This allows us to keep trace of the decomposition operations and to attribute a measurement to different decompositions, but referable to the same level of detail.

2.3 Method of application of axiomatic selection

The application of axiomatic decomposition foresees a mapping process among four conceptual domains, i.e., customer functional, physical and process domains (Figure 1). In this context, Case Stories represent the description in semi-formal language of the system behavior with respect to the requests of the various actors [15-18]. Therefore, in order to activate the axiomatic selection, we can consider Case Stories as customer attributes (CA) of the process. Hereinafter, the Use Cases are identified as high-level functional requirements (FR). Design parameters (DP) are created by designers. They correspond to the collaborations (interactions) among various Use Cases. In this way, first-level decomposition describes the network of interactions among Use Cases [16, 20]. Finally, we can consider the process variables (PV) coinciding with the applications, which implement services to be selected (WS) [16]. They represent the implementation tools of the data communication system, described in a formal way through UML diagrams [24]. Axiomatic Design continues with successive decompositions of the selected web services up to a certain level of detail that allows their implementation by developpers. In the last stage of decomposition, the decomposed functional requirements are elementary functions, whereas the design variables (DP) are a data sets corresponding to the objects of upper level classes [24-26]. In the SOA architecture we can identify the Web services (WS) to be implemented based on the project matrix between functional requirements (FR) and design parameters (DP). In the case that we will present later, we will neglect the design of the implementation phase.

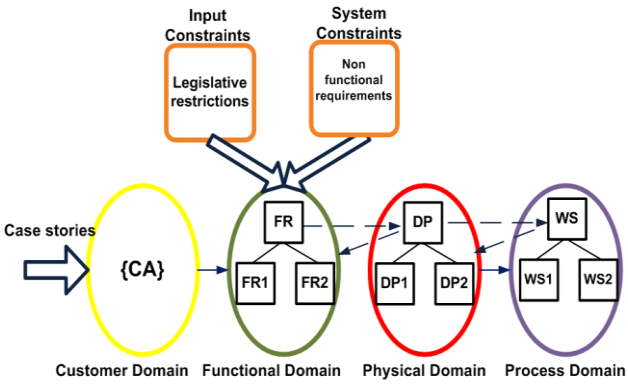


Fig. 1 Axiomatic process of functional decomposition

2.4 Project constraints

During the axiomatic decomposition phase of the FR it is necessary to consider the constraints of the project. We can distinguish two classes of limitations in the implementation of web services in a health care setting (Figure 1). The first limitation concerns the regulatory restrictions related to the processing of health data. In fact, these data are classified as highly sensitive. Therefore, the legislature has established a set of strict rules to protect patient privacy, where non-compliance is a criminal offense. The second limitation concerns non-functional constraints. For example, medical equipment must be as compatible as possible with the SOA, as to limit the manual intervention of communication of health parameters to the management platform. Figure 2 schematically illustrates the modalities of application of AD.

3 Case study

3.1 Home administration of drug infusion in pediatric oncology

The case study under review is about the optimization process of drug infusion dispensing. To this end, a number of interviews have been carried out with the healthcare workers at Gaslini Hospital in Genoa, parents, patients and other actors. The actors involved are stylized in the general scenario of the process of Figure 2 with the little man icon.

This interview allowed us to identify the primary and recurrent activities for this process. These activities can be summarized in the following synthetic semiformal description (High-level Case Story)

- Checking for drug availability (locker state, non-automated);
- Change of therapy;
- Change of modality;
- Change of drug;
- Prescription of drug by the hospital doctor;
- Withdrawal of drug at the hospital pharmacy;
- Administration of drug by parents or healthcare professional (nurse, doctor);

Practically, administration of drug is activated by the management device. At this point, the system detects that the drug is about to end and sends the alert to the control center.

The control center contacts the hospital doctor who prescribes the medicine administered by the hospital pharmacy where the patient's parent goes to collect it. At the end of the process, the availability of the drug is updated.

3.2 Process re-design

Process re-design of drug infusion administration foresees the implementation of an IT centralized Platform (H@H Platform). The exchange of information among various

involeved actors is carried out through web services in SOA architecture. Identifying the web services to be implemented consists in re-designing the whole administration process, considering the constraints and intrinsic features of the current process [9, 10]. As a result, the first phase of the selection process has been characterized by a series of in-depth interviews with the healthcare workers at Gaslini Hospital in Genoa, parents, patients and other actors. This activity allowed preparing a series of Case Stories, which describe in a semi-structured way the state of the art of the completely pediatric home care process. Each Case Story represents a single portion of the entire process [17, 8]. In this work, we tackle a twofold challenge. On the one hand, it is necessary to formalize the proposal for a non-optimized solution in terms of UML [18]. The second challenge regards the attempt to optimize the entire process based on an axiomatic approach [16]. This translates into the re-engineering of the entire process [11, 12]. In this context, the H@H platform becomes the focus of change. Axiomatic Design will allow to identify a robust set of web services to be conveyed through the platform.

3.3 State of the art (As is)

3.3.1 General scenario of the process of prescribing and dispensing drug infusion

From the above-mentioned Case Story, it is possible to outline the general scenario of the process of home administration of drug infusion. This general scenario already represents a first solution for the re-engineering of the process. It foresees the introduction of the H@H Platform and the use of web services as a tool for information exchange. In terms of UML, Figure 2 illustrates the high-level design of the process. The represented actors and Use Cases are described in detail in section 3.3.2 hereafter.

3.3.2 Process Use Cases

In the diagram shown in Figure 2, the main Use Cases of the Case Story relating to the prescription and dispensing of drug infusion are highlighted. In addition, Use Cases describing the management of the device that dispensing the drug are reported. In particular, the Use Cases are the following:

1) Prescribe therapy or prescription of the drug therapy (dose and drug) made by the hospital doctor;

2) Dispense drug or the issue of the drug on the basis of the doctor's prescription. This activity is carried out by a pharmacy in a hospital or a territorial pharmacy depending on the type of drug;

3) Administer drug or administration of the drug by a medical device (for example, for drugs injected by continuous infusion) or by the patient's parents.

These three activities can include (extend) the update of the medical record managed by the H@H Platform. The specific activities on the device, on the other hand, are the following:

1) Check device functionality or the control of variables indicating the functioning of the medical device. A specific case of this Case is given by the analysis of malfunctions (Check device malfunction);

2) Check drug availability, describes a functionality of the Platform that verifies when the drug is about to end on the basis of consumption, and purchase of the drug itself;

3) Manage alert or management of the alerts on the basis of the functions mentioned at point 1 (malfunctioning, management of the drug, etc.). These alerts are primarily

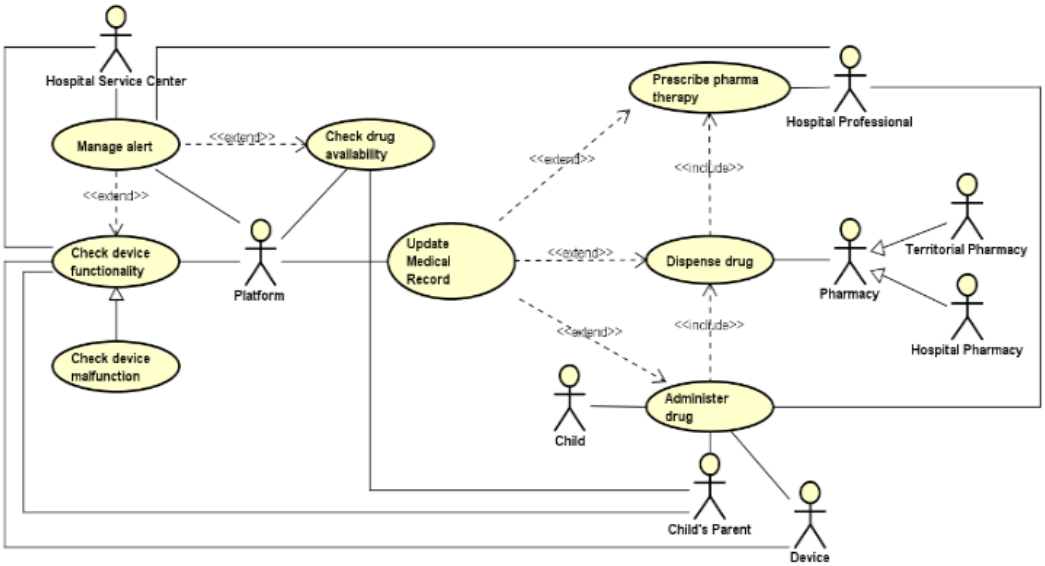


Fig. 2. General process scenario

managed by the hospital service center, which sends an alert to the hospital doctor.

3.3.3 Representation of the general scenario in terms of interactions among Use Cases

The same Case Story allows us to identify the interactions (collaborations) that can be defined among various Use Cases [16, 20]. At the level of general scenario such relations can be represented by the collaboration diagram reported in Figure 3.

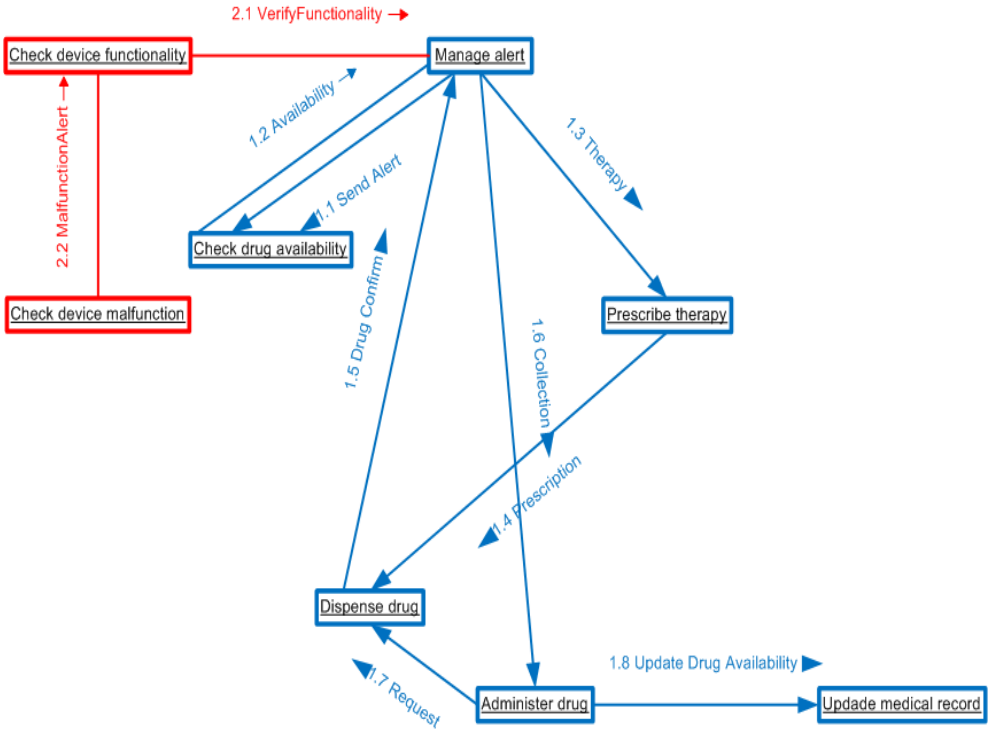


Fig. 3 Collaboration diagram

In the following, we limit the analysis only to the simple process of prescribing and dispensing drug infusion. Alert Use Cases that are not dependent on the availability of the drug will not be treated. In this case, the boundary of the application is the set of Use Cases and relations indicated in blue color. The components shown in red color represent, instead, the management of other types of alert reporting.

They are mostly reports of technical malfunction of the administration devise of the drug infusion.

3.3.4 Matrix representation equivalent to the collaboration diagram

Based on the general scenario and the relevant collaboration diagram we can build the first-level design matrix. This matrix represents the series of actions that allow the execution of the whole process. This representation is obtained by considering Use Cases as fuctional requirements (FRs) [16, 20]. As a consequence, they are inserted in order of activation according to the standard execution of the process on the matrix rows. In the columns of the design matrix, we consider as design parameters (DPs) the possible interactions (collaborations), which may exist among different Use Cases during the execution process. In particular, a collaboration between Use Case X and Use Case Y can be defined as the Use Case X method that enables any kind of processing in the Use Case Y. In other words, it can be said that the Use Case X interacts or cooperates on the Use Case Y only when it activates a method that causes Use Case Y to change its state. Therefore, Get () methods do not produce collaborations. In relation to the previous collaboration diagram there are the Following collaborations among first-level Use Cases (Figure 4). As you can see, the design matrix associated with the current representation of the process state of the art does not comply with the axiom of independence [23]. The matrix is neither diagonal nor triangular. This means that the Use Cases are not independent from one another.

	DP1 Collaboration Manage alert	DP2 Collaboration Check drug availability	DP3 Collaboration Prescribe therapy	DP4 Collaboration Dispense drug	DP5 Collaboration Administer drug	DP6 Collaboration Updade medical record
FR1 Manage alert	x	x		x		
FR2 Check drug availability	x	x				
FR3 Prescribe therapy	x		x			
FR4 Dispense drug			x	x	x	
FR5 Administer drug				x	x	
FR6 Updade medical record					x	x

Fig.4. First-level design matrix

3.3.5 Considerations on functional dependencies of first-level matrix

The first-level design matrix does not respect the axiom of independence because there are functional dependencies among Use Cases. In particolar, we can identify the following critical situations:

- FR1: The Use Case Manage alert is activated by the Availability function of the Use Case Check drug availability. This activation makes the Use Case Manage alert dependent on the Use Case Check drug availability;
- FR1: The Use Case Manage alert is activated by the Drug Confirm feature of the Use Case Dispense drug. This activation makes the Use Case Manage alert dependent also on the Use Case Dispense drug;
- FR4: The Use Case Dispense drug is activated by the Request feature of the Use Case Administer drug. This activation makes the Use Case Dispense drug dependent on the Use Case Administer drug.

Observing the previous collaboration diagram, we notice that the three identified critical situations correspond to the three cycles in Figure 3. This means that the axiom of independence ensures that there are no loops/cycles in the process, in which the information is likely to remain in a state of indeterminacy.

3.4 First-level optimization

3.4.1 Use Cases re-definition

This situation leads us to the position to intervene to optimize the process. From the point of view of axiomatic design we can introduce three new Use Cases. In this way we would have a functional decoupling, which would lead to a lower triangular type of the design matrix, in order to determine a well-defined sequential path of actions. This intervention, in general, corresponds to transform a directed cyclic schema into a directed acyclic one, As is well known [27], a network of nodes and arcs is acyclic if an only if it possesses a topological ordering of its nodes. It means, whenever we assign a label to a node at an iteration, the node has only outgoing arcs and they all must necessarily point to nodes that will be assigned higher labels in subsequent iterations.

Consequently, this labeling gives a topological ordering of nodes. Provided that the nodes are labelled in ancestral order (parents come before children) a directed acyclic graph can be represented as a trinagular adjacency matrix. (traniangular superior). Note that, by labelling nodes in an opposite way, we obtain a trangular inferior matrix. A network that contains a directed cycle has no topological ordering, and conversely. For this reason, we introduce the following Use Cases:

- FR2.1 Manage drug availability. This Use Case concerns the management of the availability of drug infusion. It is linked to the Use Case Check drug availability by an inclusion relation (<include>);
- FR4.1 Manage dispense drug. This Use Case provides confirmation of drug availability at the pharmacy. It is linked to the Use Case of Dispense drug by an inclusion relation (<include>);
- FR5.1 Withdrawal drug. This Use Case concerns the drug withdrawal by the child's parents at the pharmacy that provide the availability. It is linked to the Use Case of Administer drug by an inclusion relation (<include>).

Figure 5 represents the collaboration diagram of the process of prescribing and dispensing drug infusion. As we observe in this figure, there are no more cycles, and consequently, the process follows a well defined sequential path.

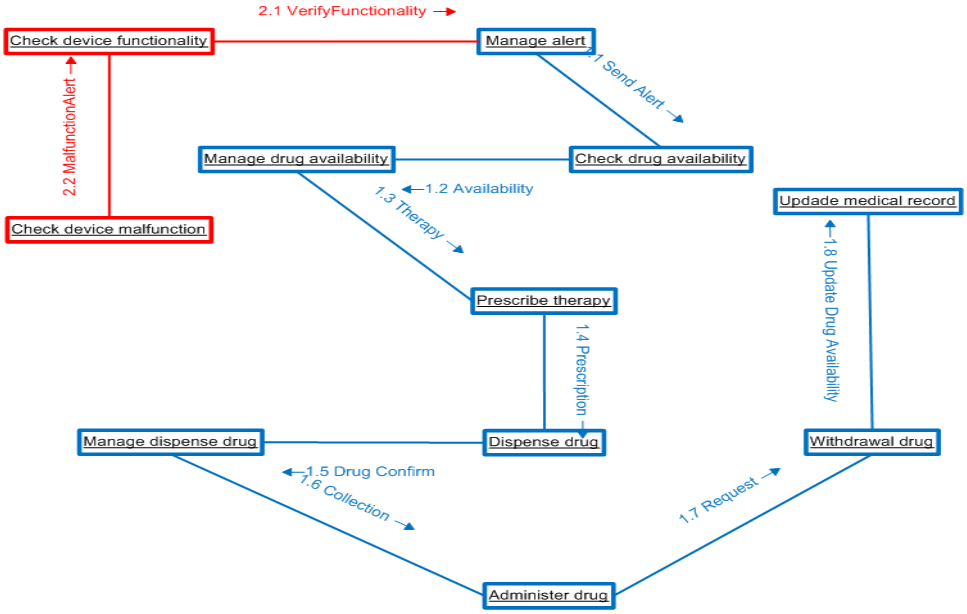


Fig.5 Collaboration diagram of the optimized process

3.4.2 Applying the axiom of independence

The introduction of these three use cases allows us to re-define the first-level design matrix, as shown in Figure 6.

	DP1 Collaboration Manage alert	DP2 Collaboration Check drug availability	DP2.1 Collaboration Manage drug availability	DP3 Collaboration Prescribe therapy	DP4 Collaboration Dispense drug	DP4.1 Collaboration Manage dispense drug	DP5 Collaboration Administer drug	DP5.1 Collaboration Withdrawal drug	DP6 Collaboration Updade medical record
FR1 Manage alert	X								
FR2 Check drug availability	X	X							
FR2.1 Manage drug availability		X	X						
FR3 Prescribe therapy			X	X					
FR4 Dispense drug				X	X				
FR4.1 Manage dispense drug					X	X			
FR5 Administer drug						X	X		
FR5.1 Withdrawal drug							X	X	
FR6 Updade medical record								X	X

Fig.6 Optimized process design matrix

The new design matrix is of a lower triangular type. Therefore, at this level the process of prescribing and dispensing the drug infusion meets the indications of the axiom of independence. In fact, the process described in the Use Story represents a coherent sequence of activities.

3.4.3 Level assessment of the functional independence

Axiomatic Design also allows improving the degree of functional independence of the various Use Cases of the scenario. In this case, we will have to resort to the evaluation of the Reangularity of the matrix (R) [20, 21]. For this purpose, we should first assume that the elements of the design matrix (A_{ij}), which represent interactions, have a value of 1. Vice versa we can pose the same elements (A_{ij}) equal to 0. This measure indicates how close a square matrix is to the corresponding unit matrix. R is a value between 0 and 1. Reangularity relates the angles between the axes of the design parameters, while semiangularity measures the magnitude of the diagonal elements [27- 29] as follows:

$$R = \prod_{\substack{i=1,n-1 \\ j=1+i,n}} \sqrt{1 - \frac{(\sum_{k=1}^{n} A_{ki}A_{kj})^2}{(\sum_{k=1}^{n} A_{ki})^2(\sum_{k=1}^{n} A_{kj})^2}} \tag{1}$$

$$S = \prod_{j=1}^{n} \frac{|Ajj|}{\sqrt{\sum_{k=1}^{n} A_{kj}^2}} \tag{2}$$

The Reangularity index can be expressed as follows:

$$R = sin\theta = \sqrt{(1 - cos^2\theta)} \tag{3}$$

where θ represents the angle between columns vectors. The ideal uncoupled design ($R = S = 1$) is obtained by $\theta = 90^o$ that is when axis are orthogonal. Otherwise, the design can be decoupled ($R = S < 1$) or coupled ($R \neq S < 1$).

Accordingly, in our case for R=1, the matrix is unitary, so all the Use Cases would be independent. This would be the ideal solution. In this case, the application of Eq. (1) gives us a

reangularity value R=0,258345. This means that any other reconfiguration of the same design matrix, while respecting the independence axiom, must not have a value of R below this threshold.

3.4.4 Complexity vs. functional decoupling

The degree of reangularity could be improved by introducing a new Use Case, but in this case, we would have to deal with the axiom of information. In fact, the information content of the new representation of the design matrix should be at most equal to the one of the original design matrix. The axiom of information thus limits the degree of functional decoupling that may be introduced at the design phase. This axiom allows us to avoid that the new design configuration does not enhance the level of complexity of the system [23]. As regards information systems, it is convenient to estimate their size in terms of Function Points [30-32]. Function Points represent the functional point of view of the end-user. In addition, they also have the advantage of being a universally recognized measurement standard [32]. In this way, we can put the Function Point estimation deduced from the process optimized in the previous paragraphs, as an upper limit, with respect to any improvements in the degree of functional decoupling, which would result in an increase in the level of complexity of the system.

3.5 Second-level optimization

3.5.1 Decomposition Use Cases by zigzagging process

The detailed information, taken from the Case Story of the drug infusion administration process, allows us to identify the sequence of actions that trigger the execution of the process. These actions constitute the second-level functional requirements, or the decomposition of the corresponding Use Cases. At this point, the zigzagging process between the Functional Domain and the Physical Domain allows to associate, with each action of the sequence of execution of the process (FRij), a specific second-level interaction (DPij).

3.5.2 Second-level matrix representation FR-DP

The rules for generating the second-level FR-DP design matrix are the same as those followed in paragraph 3.3.4. In particular, as in the previous case, we can state that there is a collaboration from the Use Case X towards the Use Case Y, if there is a method related to the Use Case X that changes the status of the Use Case Y. In the same way, we can define the interactions among second-level activities, such as the behaviour of a given action that induces a processing towards another action. The greater detail of the second-level of decomposition allows us to identify the actions of the drug administration process. They correspond to the methods of the same Use Cases constituting the general scenario of the process. In a preliminary way, we identify the actions that generate first-level collaborations. This means selecting communication actions among Use Cases. They are shown in Figure 7.

From	To	Metodo	Collaboration
FR1 Manage Alert	FR2 Check Drug Availability	FR11: SendDeviceAlert()	Collaboration 1
FR2 Check Drug Availability	FR2.1 Manage Drug Availability	FR24: SendAlertAV()	Collaboration 2
FR2.1 Manage Drug Availability	FR3 Prescribe therapy	FR12: SendInformAvailability()	Collaboration 3
FR3 Prescribe therapy	FR4 Dispense drug	FR14: SendPrescription()	Collaboration 4
FR4 Dispense drug	FR4.1 Manage Dispense drug	FR42: ConfirmDrugAvailability()	Collaboration 5
FR4.1 Manage Dispense drug	FR5: Administer Drug	FR15: SendDrugAvailability()	Collaboration 6
FR5: Administer Drug	FR5.1 Withdrawal drug	FR52: DrugCollection()	Collaboration 7
FR5.1 Withdrawal drug	FR6 Update Medical Record	FR52: DrugCollected()	Collaboration 9

Fig. 7. List of interactions activating Use Cases

3.5.3 Second level design matrix considerations

At this point, starting from Figure 6 and considering the collaborations among Use Cases shown in Figure 7, it is possible to build the second level FR-DP design matrix (Figure 9 attached). The Xs in Figure 9 can be interpreted in this way. The black Xs are internal elaborations to the specific Use Case. For this reason, they are graphically enclosed in a square. The red Xs are, instead, external to the squares that delimit the actions inside the individual Use Cases. They represent the methods that, starting from a specific Use Case, activate a process in another Use Case. These actions are performed using the methods shown in Figure 7. The description of the elements of the Figure 9 are attached in the summary matrix of Table 2.

3.6 Considerations around the axiomatic decomposition levels of the system

3.6.1 Stratification of functional decomposition levels

The axiomatic decomposition of functional requirements can be pushed up to a level of stratification such that the produced UML diagrams can be used to automatically produce the software. In fact, there are commercially suitable cases, which automatically generate software on the basis of particularly detailed UML diagrams [33]. Figure 8 illustrates the stratification of analysis levels. Each decomposition level reports the corresponding UML diagrams. However, the following question arises: is it advisable to proceed to a decomposition of elementary detail? The following paragraph presents reflections on this question.

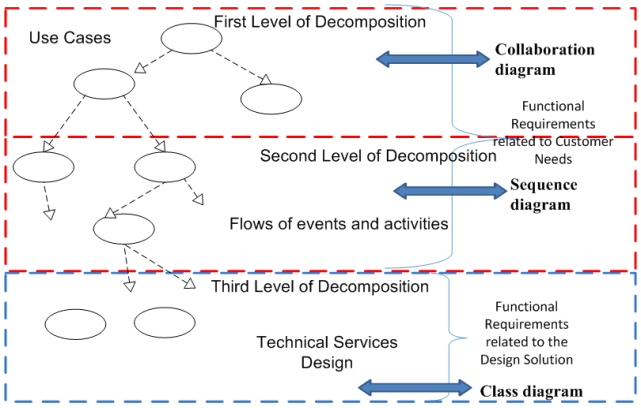


Fig. 8. Stratification of functional decomposition levels

3.6.2 The limits of an exaggerate stratification

Excessive stratification of axiomatic decomposition levels can lead to a paralysis of software deployment and upgrade activities [33]. Axiomatic Design methodology and UML modeling are designed for software development processes that follow either the cascading or the V model. These models are suitable for large, centrally driven software implementations, but are quite rigid for developing procedures with not particularly complex functionality in a web environment [34]. The most real risk is to allocate too many resources to the production of project documentation, which then the continuous system updates risk to make quickly obsolete. To avoid this, in recent years many software houses have preferred to use more efficient development techniques based on agile software development [33, 34].

3.6.3 Agile software development

Agile software development is based on the close connection between developer and end-user with the prototype production of different versions of the software product. In this way, programmers focus on implementing a limited number of features, which they periodically show to the end user. If the prototype passes the customer's tests, further functionalities are implemented. Otherwise, the end user's comments are noted down and the software is updated. This methodology allows for a participatory implementation between the service provider and the customer. In addition, the basic versions of the procedure can be brought into production immediately, and the missing functionalities can be put into production for subsequent releases [33]. As you can see, this methodology is particularly efficient in emergency situations, but the lack of a complete trace between the user specification and the technical solution adopted can lead to situations of annoying misunderstanding between programmer and end user [33]. In this case, there may be cases of publication of procedures that do not completely meet the user specifications, for which the client and supplier are blaming themselves for the causes of failure [34].

3.6.4 How to combine AD and UML design with agile software development techniques?

In order to avoid the situations described in paragraph 3.6.2 some authors [33, 34] have proposed the use of UML modeling or Axiomatic Design, also, in the case of agile programming and web environment. In this case, the stratification of the axiomatic decomposition is limited to high levels of detail intended for the understanding of the final client. The diagrams produced will be simple sketches, shared by the end-user, with which the programmers will proceed in their work. Instead, the technical solutions to be adopted will remain the exclusive competence of the developers. In our case, this solution corresponds to using only the UML diagrams of the first two layers of Figure 9.

The adoption of this solution corresponds to limiting the application of Axiomatic Design to the levels of greater abstraction of the process of administration of infusion drug.

This allows us to keep the general requirements of the system under control, delegating to each actor the technological adaptation best suited to their own infrastructure. The H@H Platform continues to be the coordination center for all activities, updating and making available UML diagrams and XSDL diagrams on the information to be exchanged.

3.7 Web Service Identification Process

3.7.1 Selection approach

Mapping between the Physical Domain and the Process Domain leads us to the definition of the so-called web services matrix (WS). This matrix is mirroring the optimized design matrix, due to the interdomain zigzagging process [16]. It schematized the dependencies between the interactions with the previously defined Use Cases (Collaborations) and the web services to be implemented.

Interactions are the rows of the matrix, while WS are the columns. The Process Domain allows the designer to assign a specific mode of implementation to the functional requirements expressed as Use Cases. The zigzagging process is iterative. Therefore, the selection of an implementation intervention may require the reformulation of some functional requirements. This means that functional decomposition takes place simultaneously on two domains.

This separation between the functional requirements side represented by the mapping between the Functional domain and the Physical Domain and the implementation side represented by the relations between the Physical Domain and the Process Domain is particularly significant. In this way, functional requirements are decoupled from

implementation solutions. This allows a dynamic approach to the management of the prescription and dispensing process of drug infusion to be pursued. In fact, the introduction of new therapies or new technological tools may also require a reconsideration of the functional requirements of the system.

This mechanism allows to verify the existence of the same functional requirements in a constantly evolving context such as the health sector. Figure 10 summarizes this process of parallel double mapping and the contextual application of the axioms of independence and information.

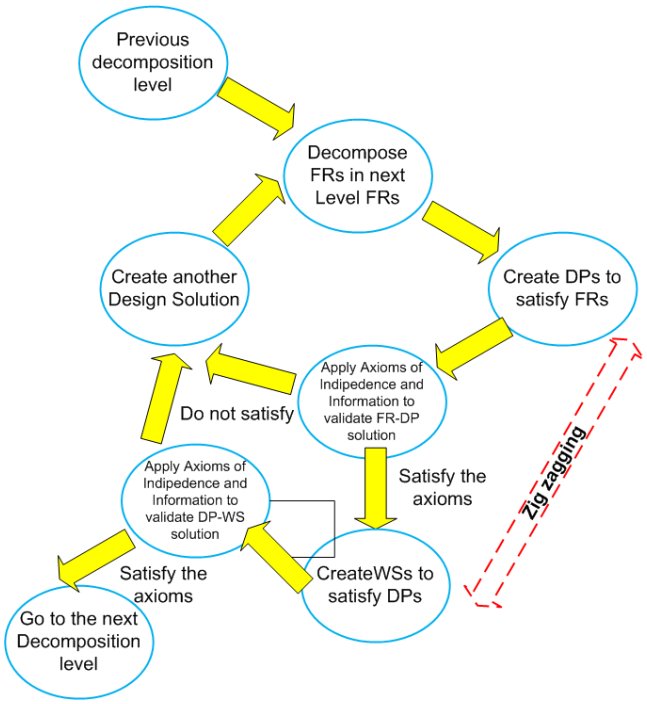


Fig. 9. Axiomatic process of web services selection

3.7.2 Web services Identification

In our case the approach proposed in paragraph 3.7.1 can be simplified. The H @ H project was developed based on the SOA architecture. This means that Web services can be identified based on the project matrix of Figure 9 [16].This is equal saying that each web service is logically consistent, as the correlation matrix of Figure 9 respects the independence axiom. Moreover, respecting the information axiom means minimise the system complexity. The axiomatic decomposition has involved two levels only. These two levels have allowed us to identify the web services and the relevant methods (actions) to implement. They represent the conceptual scheme of the system, as illustrated by Table 2. Further levels of decomposition allow us, instead, to define the logical design of the system. In this case, every interaction becomes data file, while methods become elementary processing.

4 Conclusions

Home administration process of drug infusion in pediatric oncology has several issues of concern. First of all, the continuity in drug infusion administration must be ensured.

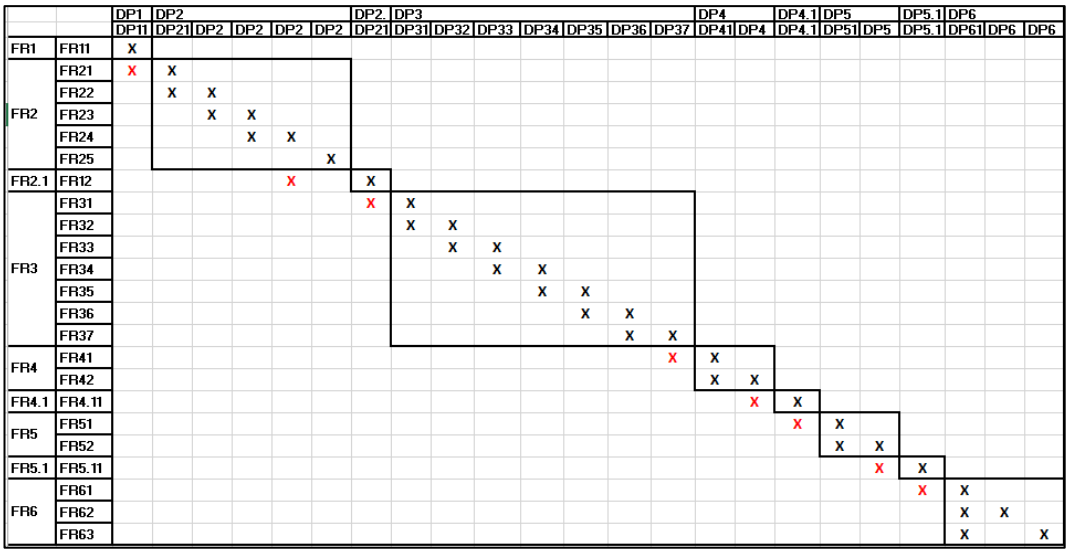
This means not only tracking medical devices, so to settle any cause malfunctions within a reasonable time, preventing weak patients, such as those having cancer from being damaged. It should also be considered that the process itself of drug dispense implicates various actors involved, also outside the treatment center of reference for the patient.

Therefore, web services to be implemented must allow the identification of the pharmacies nearest to the patient's home, verify and update the availability of the drug bags and manage the yields. The H@H Platform is an open-type. This type guarantees the interoperability with heterogenous systems, but also the possibility to extend services to provide the young patients with, as a psychological support or to integrate them with the social and school service of the area. In this sense, on the basis of Case Stories, we have shown that UML-modeling methodology allows to design new services also on the basis of specific Use Cases. Then, through the Axiomatic design approach we verified the proposed projectual solutions, by identifying the robust set of web services to be implemented.

References

1. F. O'Hagan (2017). *Work, organizational practices, and margin of manoeuver during work reintegration*. Disability and rehabilitation, pp. 1-10.
2. S. Landers, et al., (2016) *The Future of Home Health Care-A Strategic Framework for Optimizing Value*, Home Health Care Management & Practice, 28(4) 262 – 278.
3. F. Wang, (2012) *Measurement, Optimization, and Impact of Health Care Accessibility: A Methodological Review*, Annals of the Association of American Geographers, 102(5): 1104–1112.
4. G. Du, X. Liang, C. Sun, (2017) *Scheduling Optimization of Home Health Care Service Considering Patients' Priorities and Time Windows*, Sustainability 9, 253.
5. I. Giorgi, G. Calsamiglia, M. Negri, F. Scafa, R. Colombi, S. M. Candura, & O. Bettinardi. (2007). *Management of occupational stress among patients with cardiac diseases.* Giornale italiano di medicina del lavoro ed ergonomia, 29(3 Suppl), pp. 695-696.
6. K. Avila, P. Sanmartin, D. Jabba, M. Jimeno, (2017) *Applications Based on Service-Oriented Architecture (SOA) in the Field of Home Healthcare*, Sensors, 17(8): 1703.
7. F. Pecoraro, D. Luzi, E. Pourabbas, F. Ricci (2016) : *A Conceptual Model for Integrating Social and Health Care Services At Home: The H@H Project*, IEEE 18th International Conference on E-health Networking, Applications & Services (Healthcom'16), p. 375-380, 14-17 September 2016, Munich-Germany.

8. F. Pecoraro, D. Luzi, E. Pourabbas, F.L. Ricci, (2017) *A methodology to identify health and social care web services on the basis of Case Stories*. Proceedings of E-Health and Bioengineering Conference (EHB).
9. JS. Peck, SG. Kim (2010*). Improving patient flow through axiomatic design of hospital emergency departments*, CIRP Journal of Manufacturing Science and Technology, Vol. 2, Issue 4, 255-260.
10. JS. Peck, SG. Kim (2010). *Axiomatic approach for efficient healthcare system design and optimization*, CIRP Annals, Vol. 59, Issue 1, 469-472.
11. DT. Matt, E. Rauch, VM. Franzellin (2015). *An axiomatic design-based approach for the patient-value-oriented design of a sustainable Lean healthcare system*. International Journal of Procurement Management, Vol. 8, Issue 1-2, 66-81.
12. G. Arcidiacono, DT. Matt, E. Rauch, (2017). *Axiomatic Design of a Framework for the Comprehensive Optimization of Patient Flows in Hospitals*, Journal of Healthcare Engineering, vol. 2017, Article ID 2309265, 9 pages.
13. DT. Matt, G. Arcidiacono, E. Rauch (2018). *Applying Lean to Healthcare Delivery Processes-a Case-based Research*. International Journal on Advanced Science, Engineering and Information Technology, 8(1), 123-133.
14. Z. Zainal, (2007) *Case study as a research method*, Jurnal Kemanusiaan bil.9.
15. B. Flyvbjerg, Case Study (2011). Norman K. Denzin and Yvonna S. Lincoln, eds., *The Sage Handbook of Qualitative Research,* 4th edition, (Thousand Oaks, CA: Sage, 2011), chapter 17, pp. 301-316.
16. I. Jacobson, M. Christerson, JP. Onsson, G. Övergaard. (1992), *Object-Oriented Software Engineering - A Use Case Driven Approach*, Addison-Wesley.
17. C. Parretti, F. Rolli, E. Pourabbas, P. Citti, (2018) *Axiomatic Selection of Health and Social Care Web Services on the Basis of Use Cases*, the 12th International Conference on Axiomatic Design (ICAD 2018).
18. S. Crowe, K. Cresswell, A. Robertson, G. Huby, A. Avery, A. Shikh, (2011) *The case study approach*, Medical Research Methodology, 1.
19. D.Tarenskeen, R.Bakker, S.Joosten, (2015) *Applying the V Model and Axiomatic Design in the Domain of IT Architecture Practice*; Procedia CIRP 34, 263-268.
20. P. Pimentel, C. Stadzisz, (2006) *A Use Case based Object-Oriented Software Design Approach using The Axiomatic Design Theory*. Proceedings of ICAD2006 Fourth International Conference on Axiomatic Design, 4:1-8.
21. N.P. Suh, (1990) *The Principles of Design*, Oxford Series on Advanced Manufacturing.
22. A.M. Farid, (2016) *An engineering systems introduction to axiomatic design. In Farid AM, Suh NP (eds) Axiomatic design in large systems: complex products, buildings and manufacturing systems*, Chap. 1, Springer, Berlin, Heidelberg, 1–47.
23. N.P. Suh, Axiomatic Design - *Advances and Applications*, (Oxford University Press, 2001).
24. S.H. Do, N.P. Suh, (2000) *Object-oriented software design with axiomatic design*, Proc. CIRP 49, 278-84.
25. F. Rolli, A. Giorgetti, P. Citti, M. Rinaldi, (2016) *Information content evaluation to obtain robustness of the management in Italian fiscal process (Part 2): optimization of Italian income certification process of the year 2016*, Proc. CIRP 53, 63-69.
26. F. Rolli, A. Giorgetti, P. Citti, M. Rinaldi, (2016) *Improvement of the compilation process of the Italian income certifications: a methodology based on the evaluation of the information content (Part 1)*, Proc. CIRP 53, 56-62.
27. R. K. Ahuja, T. L. Magnanti and J. B. Orlin, (1993) *Network Flows: Theory, Algorithms, and Applications*, Prentice-Hall, Inc., Upper Saddle River, New Iersey, USA.
28. D. Silverstein, P. Samuel, and N. Decarlo, *Axiomatic Design*, vol. 1. 2011.
29. J. Delaš, S. Škec, and M. Štorga, (2018) *Application of Axiomatic Design principles in conceptual design*, the 12th International Conference on Axiomatic Design (ICAD 2018).
30. A. J. Albrecht, (1979) *Measuring Application Development Productivity*, Proceedings SHARE/GUIDE IBM Applications Development Symposium, October 1979.
31. IFPUG - *Function Point Counting Practices Manual*, Release 4.1 - Westerville - Ohio, 1999.
32. R. Meli, L. Santillo, (1997) *Function point estimation methods: a comparative overview*, Data Processing Organization srl, IFPUG - Fall Conference - September 15-19, Scottsdale, Arizona USA.
33. M. Fowler, (2010) UML Distilled, Addison-Wesley.E. Puik, D. Ceglarek, (2018), *Application of Axiomatic Design for Agile Product,* the 12th International Conference on Axiomatic Design (ICAD 2018).
34. E. Puik, D. Ceglarek, (2018), *Application of Axiomatic Design for Agile Product*, the 12th International Conference on Axiomatic Design (ICAD 2018).



		DP1	DP2					DP2.	DP3							DP4		DP4.1	DP5		DP5.1	DP6		
		DP11	DP21	DP2	DP2	DP2	DP2	DP21	DP31	DP32	DP33	DP34	DP35	DP36	DP37	DP41	DP4	DP4.1	DP51	DP5	DP5.1	DP61	DP6	DP6
FR1	FR11	X																						
FR2	FR21	X	X																					
	FR22		X	X																				
	FR23			X	X																			
	FR24				X	X																		
	FR25						X																	
FR2.1	FR12					X		X																
FR3	FR31							X	X															
	FR32								X	X														
	FR33									X	X													
	FR34										X	X												
	FR35											X	X											
	FR36												X	X										
	FR37													X	X									
FR4	FR41														X	X								
	FR42															X	X							
FR4.1	FR4.11																X	X						
FR5	FR51																	X	X					
	FR52																		X	X				
FR5.1	FR5.11																			X	X			
FR6	FR61																				X	X		
	FR62																					X	X	
	FR63																					X		X

Fig.9. Second level design matrix

Table 2. Second level web service matrix

Use case	Action	First-level interaction	Second-level interaction	Method	Web Service
FR1: Manage alert	FR11: SendDeviceAlert	DP11: Collaboration SendDeviceAlert	DP1: Collaboration Manage alert	WS11: SendDeviceAlert	WS1: Manage alert
FR2: Check drug availability	FR21: ElaborateDevice Alert	DP21: Collaboration ElaborateDevice Alert	DP2: Collaboration Check drug availability	WS21: ElaborateDevice Alert	WS2: Check drug availability
	FR22: GetInformationDevice	DP22: Collaboration GetInformationDevice		WS22: GetInformationDevice	
	FR23: ElaborateInformDevice	DP23: Collaboration ElaborateInformDevice		WS23: ElaborateInformDevice	
	FR24: SendAlertAV	DP24: Collaboration SendAlertAV		WS24: SendAlertAV	
	FR25 AlertReceived	DP25: Collaboration AlertReceived		WS25: AlertReceived	
FR2.1: Manage drug availability	FR21: SendInformAvailability	DP21: Collaboration SendInformAvailability	DP2.1: Collaboration Manage drug availability	WS21: SendInformAvailability	WS2.1.: Manage drug availability
FR3: Prescribe therapy	FR31: ElaborateAlertAV	DP31: Collaboration ElaborateAlertAV	FR3: Prescribe therapy	WS31: ElaborateAlertAV	WS3: Prescribe therapy
	FR32: GetInformAlertAV	DP32: Collaboration GetInformAlertAV		FR32: GetInformAlertAV	
	FR33: ElaborateInformAlertAV	DP33: Collaboration ElaborateInformAlertAV		WS33: ElaborateInformAlertAV	
	FR34: GetPatientInform	DP34: Collaboration GetPatientInform		WS34: GetPatientInform	
	FR35: ElaboratePatientInformation	DP35: Collaboration ElaboratePatientInformation		WS35 ElaboratePatientInformation	
	FR36: InsertPrescription	DP36: Collaboration InsertPrescription		WS36: InsertPrescription	
	FR37: SendPrescription	DP37: Collaboration SendPrescription		WS37: SendPrescription	
FR4: Dispense drug	FR41: ElaboratePrescription	DP41: Collaboration ElaboratePrescription	DP4: Collaboration Dispense drug	WS41: ElaboratePrescription	WS4: Dispense drug
	FR42: ConfirmDrugAvailability	DP42: Collaboration ConfirmDrugAvailability		WS42: ConfirmDrugAvailability	
FR4.1: Manage dispense drug	FR4.11: SendDrugAvailability	DP4.11: Collaboration SendDrugAvailability	DP4.1: Collaboration Manage dispense drug	WS4.11: SendDrugAvailability	WS4.1: Manage dispense drug
FR5: Administer drug	FR51: ElaborateDrugAvailability	DP51: Collaboration ElaborateDrugAvailability	DP5: Collaboration Administer drug	WS51: ElaborateDrugAvailability	WS5: Administer drug
	FR52: DrugCollection	DP52: Collaboration DrugCollection		WS52 DrugCollection	
FR5.1: Withdrawal drug	FR5.11: DrugCollected	DP5.11: Collaboration DrugCollected	DP5.1 Collaboration Withdrawal drug	WS5.11 DrugCollected	WS5.1: Withdrawal drug
FR6: Updade medical record	FR61: UpdateDrugAV	DP61: Collaboration UpdateDrugAV	DP6: Collaboration Update Medical Record	WS61: UpdateDrugAV	WS6: Updade medical record
	FR62: DrugCollectedPre	DP62: Collaboration DrugCollectedPre		WS62: DrugCollectedPre	
	FR63: DrugCollectedAV	DP63: Collaboration DrugCollectedAV		WS63: DrugCollectedAV	