A Software Architecture Style for Medical Process Re-engineering

Umesh Banodha & Kanak Saxena

Abstract - Software Architecture has been recognized both by Academia and software industries as the most promising approach to tackle the problems in the era of Finance, Communications, Medical, Run-time applications etc. Software architecture style is a classification of SA. Software architecture style made the communication more precise and convenient at the level of SA. Different style has different system characteristics. In this paper we propose to model the architecture of medical process Re-engineering represented as instance of UML class diagram based on Service-Oriented Architectural style which is still lacking in the field of medical Domain. We describe architecture of medical process re-engineering model which provides clinical staff, patients and other individuals with knowledge and person – specific information, intelligently filtered and presented at appropriate times to enhance health and health care with the concept of using software re-engineering. The Re-engineering process helped to improve understanding about the processes and led to the conclusions with deletion of subsequent improvements to those processes. We discuss the properties that are interesting to be analyzed with the process of analysis and reengineering.

Index Terms - Architecture style, MPR model, Reengineering, Reusability, Software architecture.

I. INTRODUCTION

Architecture Styles

An architectural style is a set of principles. An architectural style improves partitioning and promotes design reuse by providing solutions to frequently recurring problems [3, 4]. "…a family of systems in terms of a pattern of structural organization. More specifically, an architectural style determines the vocabulary of components and connectors that can be used in instances of that style, together with a set of constraints on how they can be combined. These can include topological constraints on architectural descriptions (e.g., no cycles). Other constraints are having to do with execution semantics, might also be part of the style definition." Architectural styles provide a few benefits. The major benefit is that they provide a way to have a conversation that is technology agnostic with common Language.
This allows us to facilitate a higher level of conversation that is inclusive of patterns and principles, without getting into the specifics.

Manuscript received july 09, 2011: revised july 30, 2011.
U.banodha is with the Samrat Ashok Technological Institute, Vidisha, INDIA (phone : +91-9425640876 :Fax : +91-7592-251082: email : : banodha@gmail.com)
K.saxena is with the Samrat Ashok Technological Institute, Vidisha, INDIA (email : ksv1909@yahoo.com)

Process modeling approach

Process modeling is a method that helps to understand the actions, work flow, and tasks of an organization, and how the tasks are executed. Process modeling captures the process flow and the actors in the process, and the tasks performed by these actors. The focus in process modeling is on the functional processes which are entities that start with a certain event and end with a certain result. A process has always an input and an output, input triggers the process and process results in an output [5, 6].

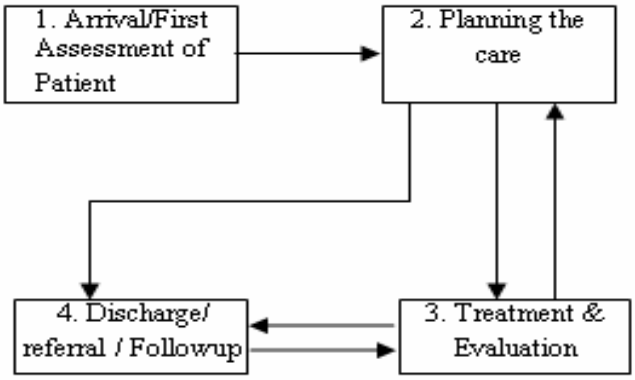

Fig. 1. Process Modeling Approach

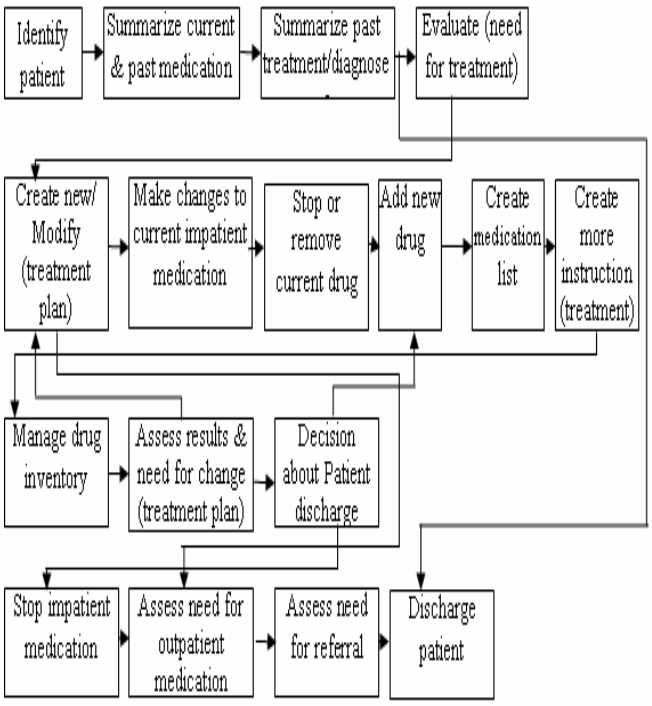

Fig. 2. Detailed steps of modeling approach

ISBN: 978-988-18210-9-6
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

The process consists of four steps (Fig.1) in the highest abstraction level. Each of the steps and the relations between them are depicted in more detail in Fig. 2. Process begins when the patient arrives to the reception of a doctor or a nurse. It ends when the patient is discharged. The actor of the processes is a doctor unless mentioned otherwise [1, 7].

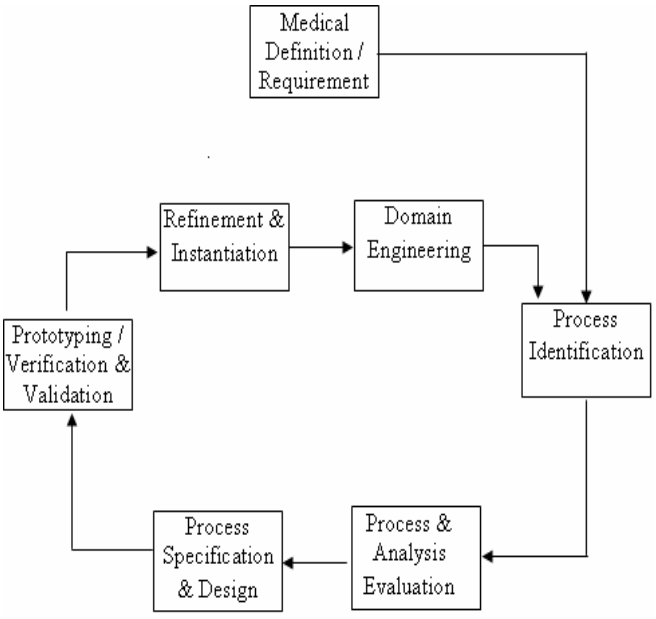

Fig. 3. A MPR Model

Medical process re-engineering (MPR) model

A medical process is "a set of logically related tasks performed to achieve a defined outcome", as shown in Fig. 3. The model defines seven activities: [2, 17]

Medical Definition: Medical Goals are identified within the context of the promotion and maintenance of health, saving and extending of life. This can be done in the context of cost reduction, time reduction, quality improvement, personnel development, and empowerment. Goals may be defined at the initial level or for a specific stage of the medical process.

Process Identification: Processes that are critical to achieve the goals defined in medical definition are identified. They may then be prioritized by importance, need for change, or in any other way that is more appropriate for the re-engineering activity in view of MPR.

Process Evaluation: The existing process is thoroughly analyzed and measured. Process tasks that are identified are evaluated in terms of the costs, times consumed by process tasks and quality/performance problems are isolated.

Process specification and design: Based on information obtained during the first three MPR activities, use cases are prepared for each process which contains the specification of the process, a new set of tasks, which are obtained from medical design control [8]. The Medical design control includes project verification & validation plans, creation of input &design output documents, requirements specification with descriptions and many more.

Prototyping: A redesigned medical process must be prototyped before it is fully integrated into the medical domain. This activity "tests" the processes so that refinements can be made with precision and time. The tests covers validation and verification at the following levels – planning, system level, Mechanical, electrical process validation and field or/and clinical level.

Refinement and instantiation: Based on feedback from the prototype, the medical process is refined and then instantiated within a medical system.

Domain Engineering: The outcomes possible in form of specific development, embedded system, electro-mechanical systems, critical system design technique for safety, reliability &compliance, fault free analysis, failure mode effects analysis, risk analysis of all considerable factors, reliability analysis, decision making/design, trade-off analysis, engineering tool & product selection, design for standards & regulatory compliance.[15]

II. MODELING

Modeling of architectural style is presented with the UML & transformation rules; model includes a static and dynamic model. The static part defines the set of possible medical process components, connectors and constrains the way in which these elements can be combined together. The dynamic part specifies how a given medical process re-engineering architecture can evolve in reaction to planned reconfigurations or unanticipated changes of the medical process environment. We study service-oriented architecture style as a case study for medical process re-engineering [9].

Static model

With reference [11] the architectural style of static model of medical process re-engineering contains, three types of elements: Medical Definition, communication, and specification documents (Reports). In (Fig. 4) the Medical Definition and Medical Services (Tests/Medication/Future check-up) are mandatory required for the static model. In this case, a Medical Definition can play different Roles at the each iteration of Medical process re-engineering, i.e., a Service Provider(Medical Labs/Hospitals) can require the communication with Patients and vice versa. The Diagnosis Agency is considered as subclass of Medical Labs/Hospitals because it provides services dedicated especially to publishing (subscribes) and querying the service specifications. A Service Requestor (Patients: require for recovery) interacts with Medical Services via a particular Session instance, contains the information about the present state of the interaction for each patient. [10]

A medical process re-engineering needs to initiate a reconfiguration usually has to communicate this to other affected medical components, we provide the necessary types of messages: The Query message is used when searching the diagnosis agency for Doctors/ Expert Knowledge Banks, and the Documents to specialist doctor

ISBN: 978-988-18210-9-6
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

and Discharge the patient messages are used for the creation and cancellation of a session [10].

We also include representations of Reports in the model. There are two types of Reports: Requirements (Future medication) and serviceSpecifications (Doctors / Expert Knowledge Banks), which contains a set of Properties. In the case of a Future medication, these properties are required by a patient for the tests/medication/future check-up. Doctors / Expert Knowledge Banks describing a particular tests/medication, these properties are guaranteed by the Medical Lab / Hospitals with certain assumptions.

The associations between the classes define how the above mentioned medical elements can be linked in a concrete architecture, constrained by the given cardinalities. Other constrains and well-formed rules can be added as OCL expressions [11]. For instance, the following expression restricts the allowed link:implies links between confirm Diagnosis link:toconfirm to other component which satisfy a logical implication:

Context confirm Diagnosis inv: self.implies →
for all (p | self: expression implies patients expression)

For instance, the following expression restricts the allowed implies links between Consult to Doctor to other component which satisfy a logical implication:

Context consult to doctor inv: self.implies →
for all (p | self: expression implies patients expression)

Architecture compliant with the style can be regarded as an instantiation of the class model like in Fig. 5: Medical component comp2, which provides service s1 (Medical Lab / Hospitals) to the patient sr1, also plays the patient (other) role sr2 and uses the Tests / Medication s2. This is necessary to guarantee diagnosis p4 of the Doctor/Expert knowledge bank whose assumptions are satisfied by s2. In this situation the session se2 (consulting to doctors) is required to serve session se1 (consulting to doctors).

Dynamic model

We use transformation rules to capture the dynamic aspects of the architectural style. There are two different ways of visualizing a transformation rule. One is to present a rule as a pair of two instance of the architectural style. The Fig.6 defines the pre-conditions for the rule application on patients' requirements for known medication and the Fig.7 defines the post-conditions. In order to apply the rule, a matching on Fig.6 with the actual architecture has to be found [10]. Fig. 7 shows an example of such a rule in which a patient sends a request to the tests/Medication it would like to connect to with precondition that the patient has to know a Doctor/expert knowledge bank which satisfies patient (couldSatisfy) for its Future Enhancement and as post condition the documents to specialist Doctor/expert is created and linked to all confirm diagnosis of the Requirements. This is done because the Medical Labs/Hospital that receives the request has to confirm all the required properties before a successful connection to the Tests/Medication can be established [10].

This constraints shows the post conditions when the patients aware of pre-requisites:

Context medication :: Future Enhancement( )
Post : Future Enhancement / Checkup
If Doctor/Expert bank = satisfy
Then Toconfirm Diagnosis or Specialist Doctor
Else Tests or Future checkup
EndIf

We are illustrating the process by taking an instance. Therefore, we omit the rules which deal with the just created Request and try to confirm all required properties on the Medical Labs/Hospital side. When the Medical Labs/Hospital can actually guarantee or confirm the Diagnosis of diseases, the link toConfirm between that diseases and the Request of patients is deleted as it is already confirm. Thus, if all properties have successfully been confirmed to the Medical Labs/Hospital, a new session for the Tests/Medication is established as shown in Fig.7.

This rule contains a negative application condition which prevents its application if there are no properties in the Future medication /enhancement which still have to be confirmed by the Medical Labs/Hospital. It creates a new session instance which realizes the connection between the patients and the Tests/Medication. Since the request of patient has been fulfilled, the corresponding message can be deleted. After that, the binding of the patients to the new Tests/Medication has been completed.

III. ANALYSIS

We identify automated means to formally reason about the correctness and consistency of architectural styles and concrete architectures captured by high level specifications in the form of structural UML diagrams. The analysis tasks is to show that the model of an architectural style fulfills the informal requirements with a concrete implementation of it i.e. show that the style and the application is consistent from both a static and dynamic point of view. In the presence of faults, a carefully constructed fault model, based on re-engineering concepts with aims at formalizing what changes in the context are encountered. Afterwards, we can first assess the fault-tolerant capabilities of the style itself by proving consistency when certain well-formed ness constraints are not satisfied by the application or the medical context. After identifying the dependability bottlenecks where certain repair actions are indispensable, new rules can be introduced to the operational description of the style to provide such repair mechanisms.

Thus, the model checking process may continue with an extended rule set The structural description of the architectural style, the transformation system, and an arbitrary (bounded) model instance of a given application, we can automatically generate a state transition system [12,13] and verify properties by model checking. Previous

ISBN: 978-988-18210-9-6
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

techniques for validation preserve all information of the modeled system, in the model checking case only dynamic parts of the application (i.e., those that can be altered by a rule) are projected into the target transition system while static parts are simplified by a compile time preprocessing in order to obtain a manageable state space. Properties to be verified are captured in the specification language of the model checker tool, which typically take the form of temporal logic formulae [14, 15], or simple transitions that are not allowed to fire during model evolution [16].

IV. CONCLUSIONS

The paper presents the impact of architectural styles on the medical domain with the re-engineering concepts. The current work is on experimenting different solutions to express rules, constraints, and control mechanisms based on the knowledge base to find the right balance between expressiveness and analyzability with the applicability of re-engineering on various MPR components. Rules can be extended to address adaptability and the capability of full or partial automatic recovery in the complete process. If we consider modern scenarios where applications are ubiquitous and they must adapt their behavior to the context in which they are executed, a re-engineering approach to modeling these aspects is essential. Rules offer a clean and neat way to specify how the architecture should react to the different scenario with the analysis capabilities as model-checking and simulation, complement the design with the capability of automatic reasoning and predicting the behavior of specified architectures. In a similar way, rules can specify the self-healing capabilities of MPR components which are associated with the way in a specific style or family of architectures styles.

REFERENCES

[1] P. Nykanen and J. Makinem, “Integration of medication information in electronics patient record systems”, The dementia patient case. Turku School of economics, Research report LTH-1:2007, Turku, 2007.

[2] ”An inventory of the certification criteria for electronic patient records”, Eurorec organization, Brussels, 2006, www.eurorec.org.

[3] Binder, R., “Design for reuse is for real, American Programmer”, vol.6, no.8, August 1993, 30-37.

[4] Lim, W. C., “Effects of Reuse on Quality, Productivity and Economics”, IEEE Software, Sept.1994, 23-30.

[5] Wang, ”Modelling information architecture for the organization. Information and Management” 32, 6, 1997, pp. 303-315.

[6] R. Weber, “Conceptual modeling and ontology: Possibilities and pitfalls”, Journal of Database Management,14, 2, 2003, pp. 1-20.

[7] Makinen,J. Et.Al., ”Process Models of medication Information”, Procd. Of the 42nd Hawaii International Conf. on System sciences, 2009, 1-7.

[8] Clark,L.A., et. Al., “Using software Engineering Technology to improve the quality of Medical Processes”, ICSE’08, May, 10-18, Germany.

[9] M. Champion, C. Ferris, E. Newcomer, and D. Orchard. “Web Service Architecture”, W3C Working Draft, 2002. http://www.w3.org/TR/2002/WD-ws-arch-20021114.

[10] Luciano Baresi, et.al.,”Modeling and Analysis of Architectural Styles Based on Graph Transformation”, 2003, 6th ICSE workshop on component base software engineering automated reasoning and prediction, Portland, 67-72.

[11] Object Management Group. “UML specification version”,1.4, 2001. http://www.omg.org/uml/.

[12 P. Bottoni, A. Sch¨urr, and G. Taentzer. “Efficient Parsing of Visual Languages based on Critical Pair Analysis and Contextual Layered Graph Transformation”, In Proc. IEEE Symposium on Visual Languages, September 2000. Long version available as technical report SI-2000-06, University of Rom.

[13] D. Varr´o.”Towards symbolic analysis of visual modeling languages”,In Paolo Bottoni and Mark Minas, editors, Proc. GT-VMT 2002: International Workshop on Graph Transformation and Visual Modelling Techniques, volume 72 of ENTCS, .Elsevier , Barcelona, Spain, October 11-12 2002,. Pages 57–70.

[14] G. Holzmann,” The model checker SPIN”,IEEE Transactions on Software Engineering, 23(5):279–295, 1997.

[15] S. Bensalem, V. Ganesh, Y. Lakhnech, C. Munoz, S. Owre, H. Rueß, J. Rushby, V. Rusu, H. Sa¨ıdi, N. Shankar, E. Singerman, and A. Tiwari. An overview of SAL. In C. Michael Holloway, editor, LFM 2000: Fifth NASA Langley Formal Methods Workshop, pages 187–196, 2000.

[16] The Murphi Model Checker. http://verify.stanford.edu/dill/murphi.html.

[17] Roger s. pressman, “software engineering: a practitioner’s approach”, mcgraw-hill international edition, VI edition.

ISBN: 978-988-18210-9-6
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

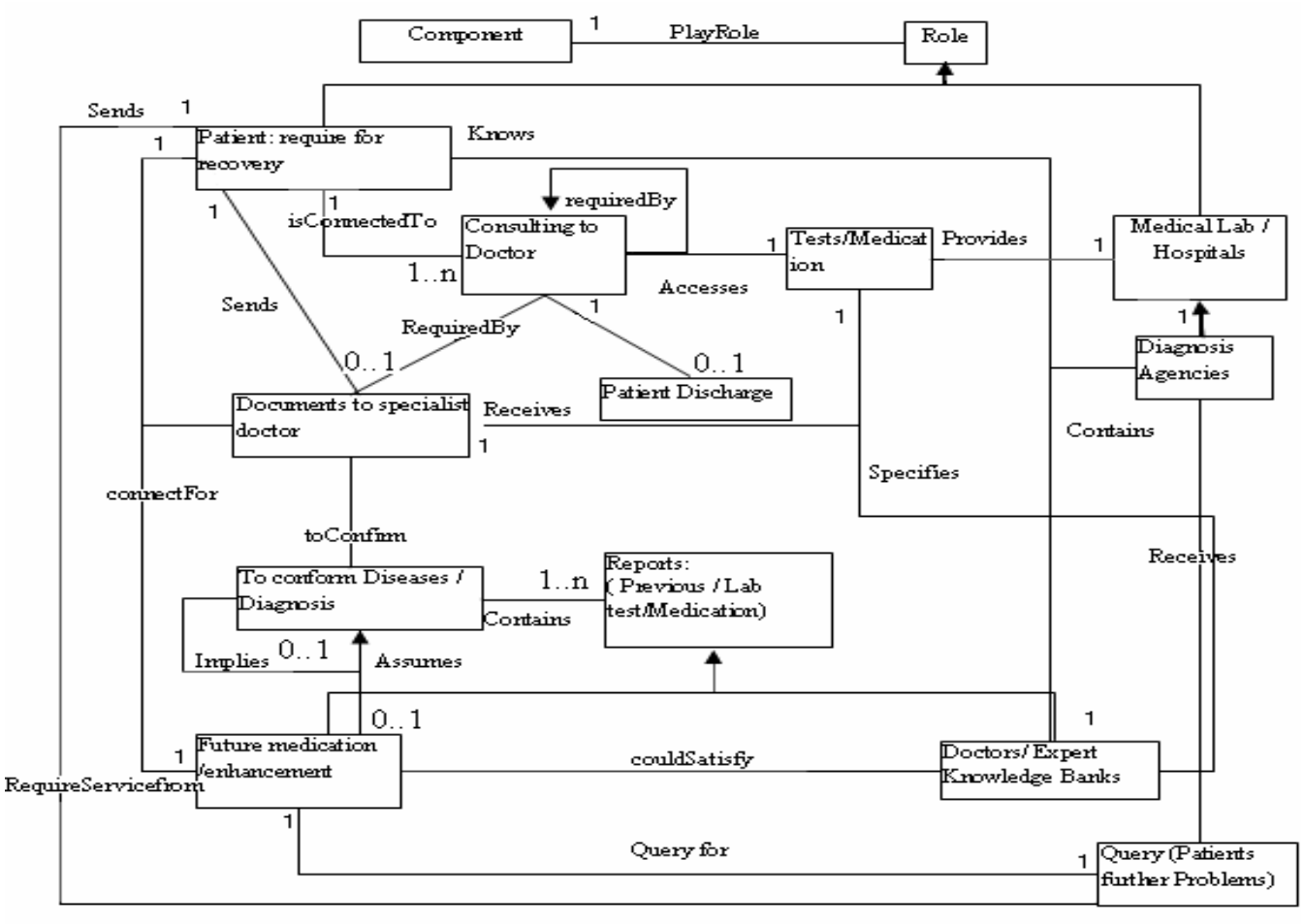

Fig. 4. Service-Oriented architectural style for MPR

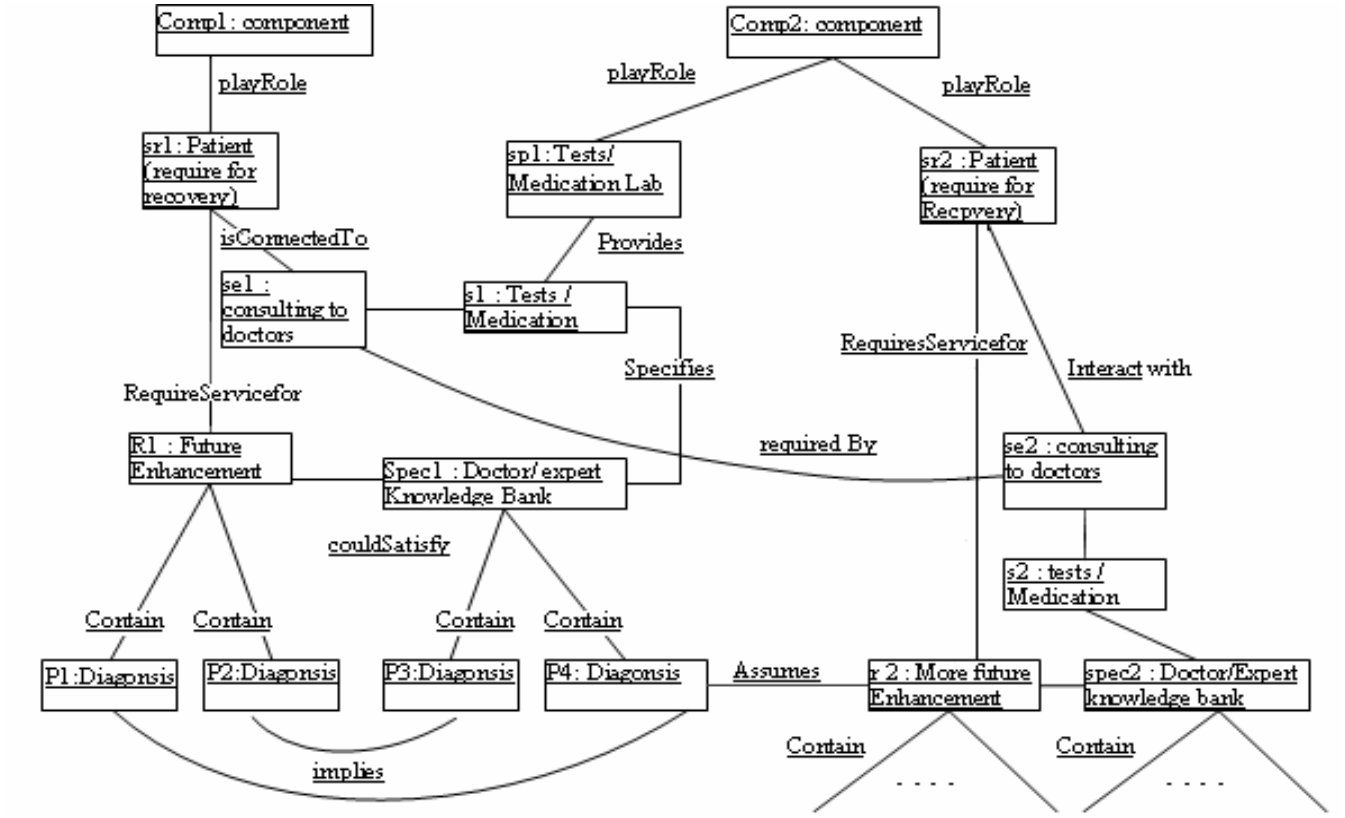

Fig. 5. Instance of MPR (service-Oriented architectural style)

ISBN: 978-988-18210-9-6
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

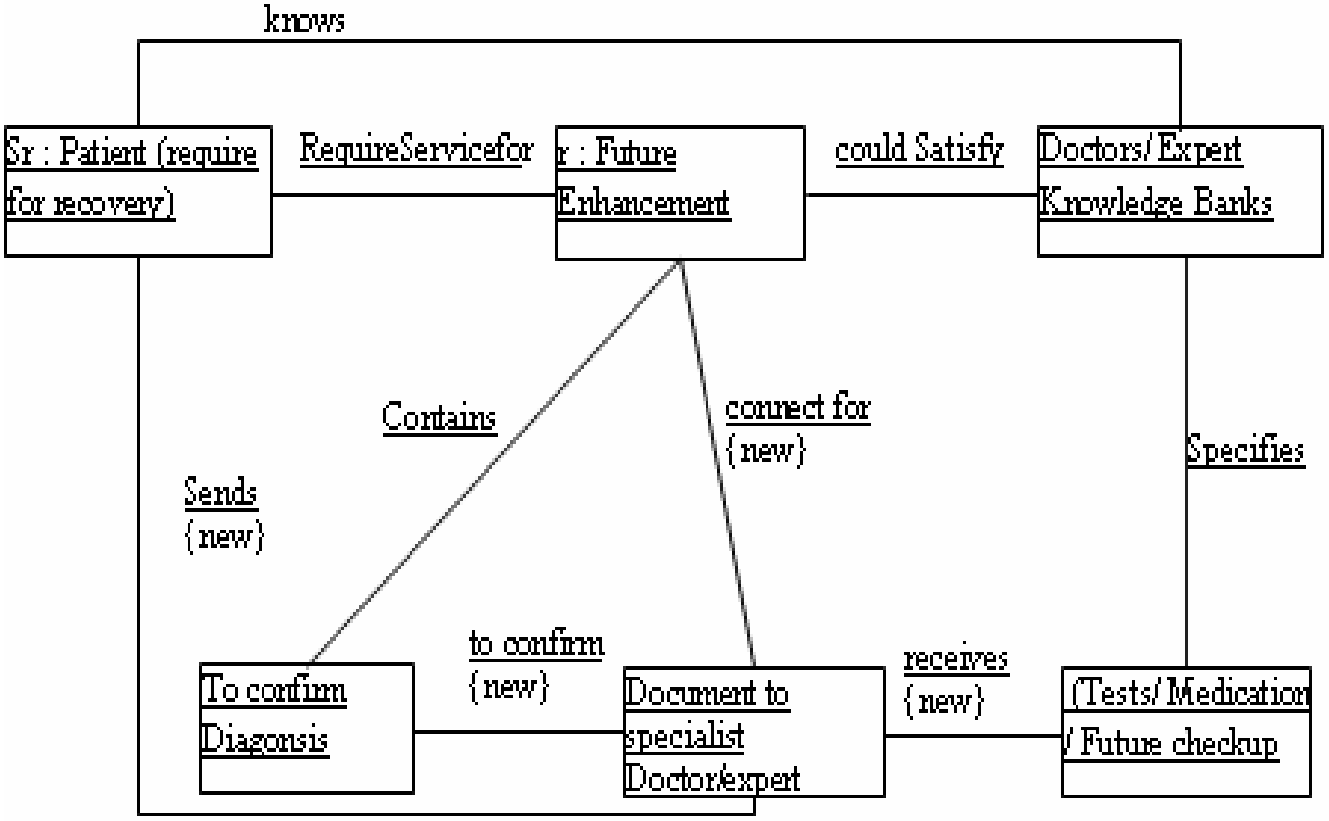

Fig. 6 Creating a patient requirement for a known medication

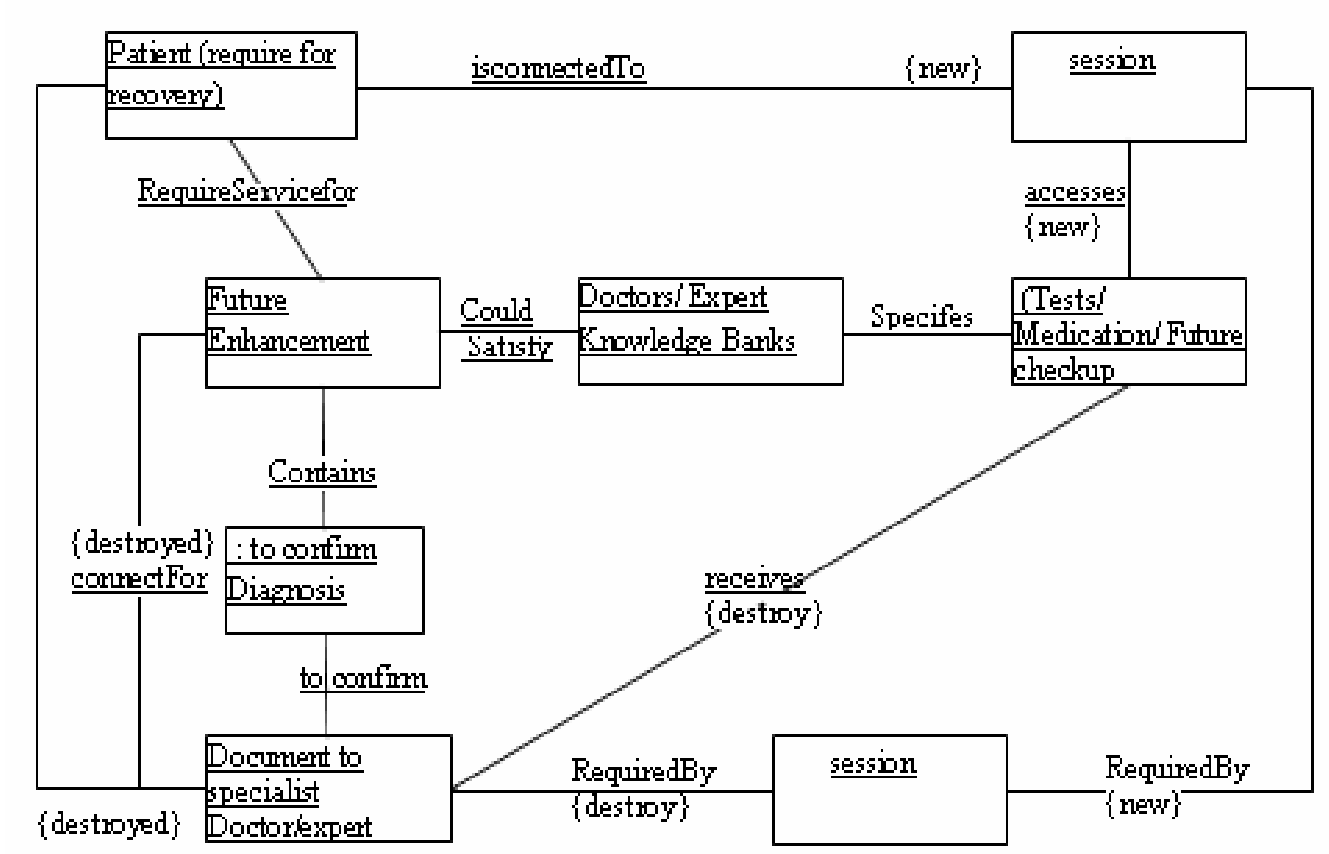

Fig. 7. Connection between Patient and Medication/ Tests and Message deletion

ISBN: 978-988-18210-9-6
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)