JOURNAL OF INFORMATION SCIENCE AND ENGINEERING 22, 625-640 (2006)

TeMeFr: Towards a Reuse-Based Development for Conference-Oriented Telemedicine Systems

HEWIJIN CHRISTINE JIAU, JINGHONG COX CHEN, KUO-FENG SSU AND JIM-MIN LIN*
Department of Electrical Engineering
National Cheng Kung University
Tainan, 701 Taiwan
**Department of Information Engineering*
Feng Chia University
Taichung, 407 Taiwan

Conventional development of telemedicine systems usually leads to a medical information driven paradigm that tends to fail in capturing suitable abstraction when developing conference-oriented telemedicine systems for assisting the proceeding of telemedicine practices. By reviewing publicly available telemedicine systems through commonality and variability analysis, we have identified common design concerns and tried to provide a framework for supporting reuse-based development. Based on identified common design concerns, a generic model of conference-oriented telemedicine systems and layered architecture are illustrated as a basis for analyzing typical design problems encountered when developing similar systems. As the main part of our work, we have proposed seven commonly occurring design problems and realized a framework implementation to provide a general resolution to these design problems based on the generic model and layered architecture of conference-oriented telemedicine systems. Finally, the details of the proposed framework are explained and a comparison with related telemedicine systems is given.

Keywords: frameworks, software architecture, object-orientation, telemedicine

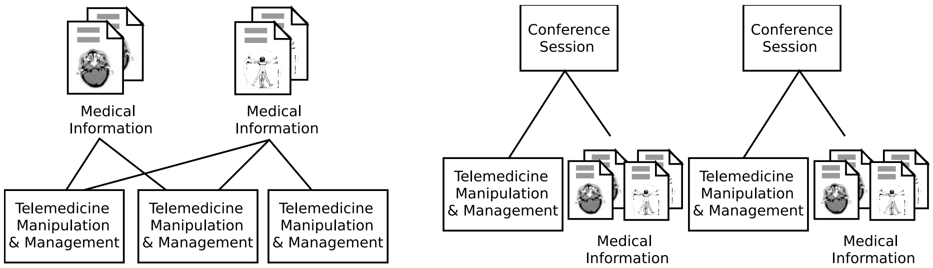
1. INTRODUCTION

Currently, there are many publicly available telemedicine systems. They are developed for diverse needs and are equipped with specialized functional components that aid telemedicine practices. Some of them focus on the construction and management of Electronic Medical Records (EMR), such as *OpenClinic* [1], a medical records system developed for private clinics, *OpenEMR* [2], an open source medical clinic management and electronic record application, and *FreeMED* [3], a web-based medical records and patient management system with many other features. Others [4-6] act as intermediate components and fulfill partial functionality of a telemedicine system.

Most of the above telemedicine systems focus on the manipulation and management of medical information or other medical information related concerns [7-10]. Such design focus helps to abstract the relationship between medical information, manipulation and management, which we call the *Medical Information Driven Paradigm* of telemedi-

Received January 15, 2004; revised April 23 & August 17, 2004; accepted April 18, 2005.
Communicated by Pau-Choo Chung.

cine systems. Unfortunately, when developing a software system for assisting telemedicine practices, the medical information driven paradigm fails to capture suitable granularity of abstractions in the problem domain. Because, in the problem domain of software systems for assisting telemedicine practices, requirements are described around a central perspective of how a specific telemedicine practice is organized in what level of granularity, which reflects the concept of a *Conference Session*. A conference session is a conceptual, organizational unit of telemedicine practice. In the abstraction of a conference session, participants of telemedicine practices (*telemedicine clients*), functionalities assisting telemedicine practices (*telemedicine manipulation and management*), and resources required for telemedicine practices (*medical information*) are encapsulated as a cohesive unit. Thus, given a cohesive organization of telemedicine clients, telemedicine manipulation and management, and medical information, these elements comprising a conference session will collaborate for assisting telemedicine practices and are bundled with contextual meaning of system behavior that reflects specific requirements of a conference session. In contrast to the medical information driven paradigm, the concept of conference session introduces to the *Conference-Oriented Paradigm* for the development of telemedicine systems. Fig. 1 shows the conceptual differences between medical information driven paradigm and a conference-oriented paradigm.



(a) Medical information driven paradigm. (b) Conference-oriented paradigm.

Fig. 1. Conceptual differences between (a) medical information driven paradigm and (b) conference-oriented paradigm.

Consider the commonality of telemedicine systems through a conference-oriented paradigm, as explained above, such as a conference session and the required behaviors for assisting telemedicine practices. We find that developers of different telemedicine systems had to make a series of decisions that reflects the common design concerns among these systems especially in the software architecture aspect. Common design problems in software architecture can be identified. If we try to provide general solutions for them, we will be rewarded with most of the benefits from reusing these solutions. By reusing solutions, developers will have a robust software architecture along with an extensible infrastructure. This allows them to easily integrate desired functional components for specified telemedicine needs, and attain shortened development life cycle and reduced development cost.

In our approach, we consider each telemedicine system as a conference-oriented information system that is specialized for telemedicine practices. This specialization is done by tweaking the front end for the presentation and manipulation of medical information for different telemedicine practices and the back end for the acquisition of diverse medical information sources. Beyond the specialization parts, the architecture of a telemedicine system is usually composed of three major parts: 1) data abstraction that encapsulates specified medical information according to different telemedicine needs, 2) facilities that allow users to manipulate and manage encapsulated medical information, and 3) intermediate structure for communication and coordination of the above two architectural parts. Among these three major parts, we have proposed seven common design problems to be considered when realizing such an architecture. Furthermore, in order to achieve a reusable architecture, this paper illustrates a general resolution to the seven problems as a framework of a telemedicine system which was embedded with our observations of publicly available telemedicine systems and experiences in developing conference-oriented telemedicine systems.

Based on the above mentioned concerns, a framework named as *TeMeFr* is proposed to allow for reuse. The software architecture, components, and design considerations of the framework are explained with regards to the proposed seven common deign problems. This paper is organized as follows. Before giving the details of our framework, we give a general analysis for conference-oriented telemedicine systems in section 2. Section 3 describes the software architecture and components of *TeMeFr*. The seven common design problems of telemedicine systems are described. Our solutions to these design problems are then explained in the remainder of the section by means of software architecture, components, and their collaborations. A concise feature comparison of *TeMeFr* with other similar telemedicine systems is given in section 4 with different categories of feature sets that reflect subdivision of the telemedicine domain. Finally, we summarize *TeMeFr* and describe directions for future work in section 5.

2. A GENERAL SYSTEM ANALYSIS FOR CONFERENCE-ORIENTED TELEMEDICINE SYSTEMS

A common solution for any modern conference-oriented telemedicine systems is a layered architecture that facilitates separation of considerations. In a layered architecture, a data abstraction layer may be introduced for separation of data abstraction for medical information. With the help of the separated data abstraction layer from the concept of conference session, the development of conference-oriented telemedicine systems, especially the design of the conference session, will be independent from out-sourcing medical information sources. Also, an abstracted view of medical information is thereby provided, which makes adapting to changes of medical information sources easier.

It would be even better for developers to introduce existing solutions from other domains of computing for design concerns of those separated layers. Fig. 2 illustrates a general layered architecture for conference-oriented telemedicine systems that emphasizes the separation of concerns to have a clear-cut design goal for the concept of conference session.

The major responsibility of a conference session to assist telemedicine practices is the realization of required telemedicine manipulation. For example, in a teleradiology system, the specific manipulation may include zooming, rotation, edge finding, etc.

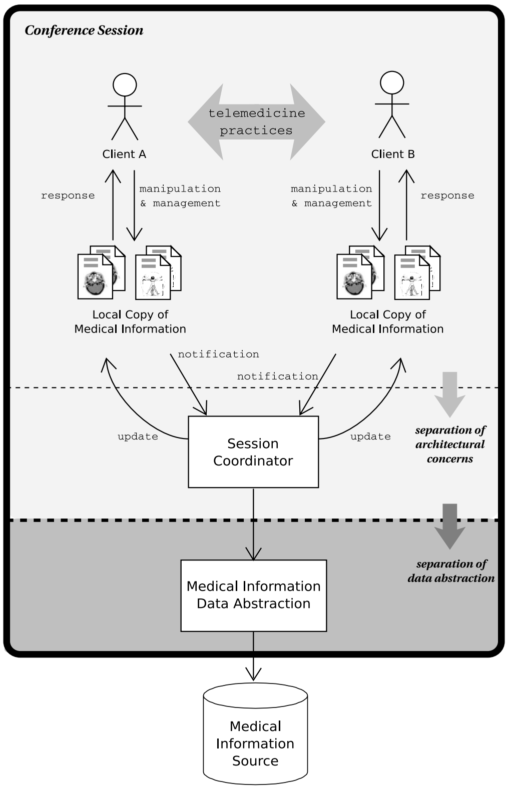


Fig. 2. A general layered architecture for conference-oriented telemedicine systems.

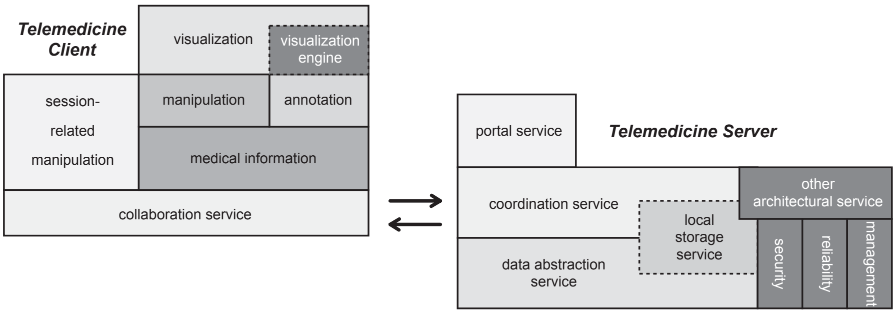


Fig. 3. A block architecture for conference-oriented telemedicine systems.

For the realization of a telemedicine manipulation mechanism, there are two major concerns: 1) how to separate the abstraction of telemedicine manipulation from possible implementations so that changes in implementation can be isolated without affecting the manipulation mechanism; 2) how to make it easy and effective to perform the coordination of telemedicine manipulation triggered by different clients. In the layered architecture shown in Fig. 2, the design concerns of the realization of the telemedicine manipulation mechanism are tangled in the layer of conference session, which needs further refinement to reflect these detailed concerns. We use a block architecture for conference-oriented telemedicine systems given in Fig. 3 to represent a elaboration of the layered architecture in Fig. 2.

The architecture shown in Fig. 3 distinguishes two different roles played by the client and server of a conference-oriented telemedicine system. The client architecture realizes the concept of the conference session layer, especially the telemedicine manipulation mechanism; the server architecture supports the implementations of separated concerns including data abstraction and architectural concerns. Looking deeper at the client architecture, we find that the telemedicine manipulation mechanism is realized with the participation of five inner blocks: *medical information* block, *manipulation* block, *annotation* block, *visualization engine* block, and *visualization* block. These blocks are designed for the realization of the telemedicine manipulation mechanism and for the fulfillment of the design considerations mentioned above. Here is an example scenario of telemedicine manipulation as illustrated in Fig. 4:

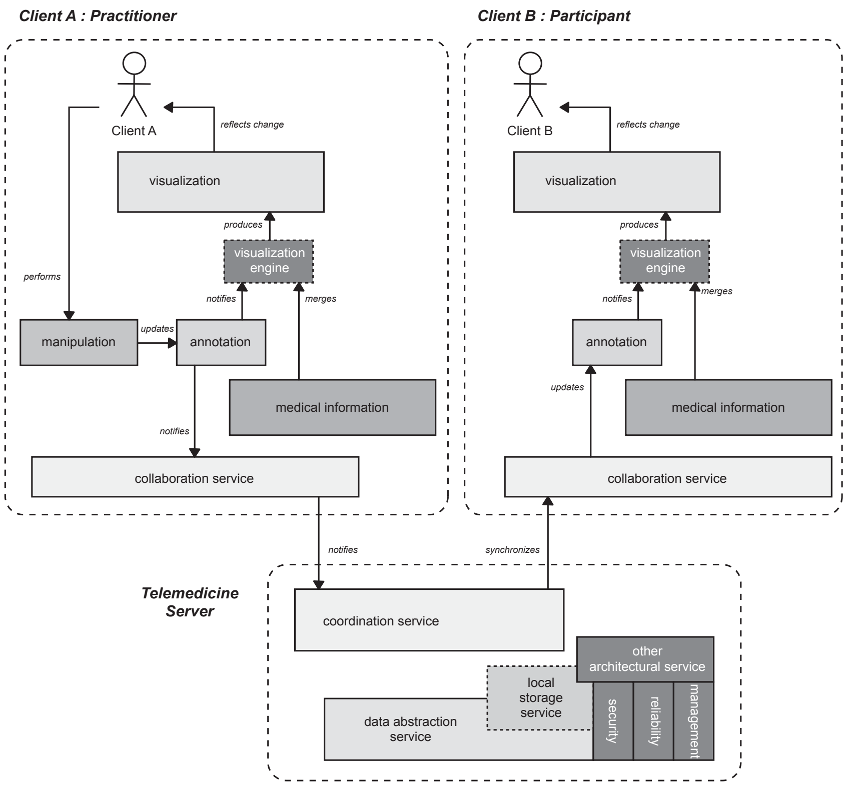


Fig. 4. An example scenario of telemedicine manipulation.

1. *Client A*, as a practitioner of telemedicine manipulation, performs specific telemedicine manipulation.
2. The *manipulation* updates corresponding *annotation* of the target of manipulation.

3. The change of *annotation* notifies both the *visualization engine* for local visualization of manipulation effect and the *collaboration service* for extending the manipulation effect in the global context of the conference session.
4. Locally in the conference session of *Client A*, *medical information* (target of manipulation) is merged with *annotation* by the *visualization engine* to produce a concrete *visualization* of *medical information* that reflects changes triggered by specific telemedicine manipulation.
5. On the other hand, *collaboration service* will notify the *coordination service* in the telemedicine server about the change of *annotation*.
6. The *coordination service* then synchronizes the participants in the global context of the conference session.
7. The synchronization message is sent to the *collaboration service* of a participant like *Client B*.
8. The *collaboration service* updates the corresponding *annotation* of the manipulation target.
9. The local visualization of manipulation effect proceeds as in *Client A* to produce a concrete *visualization* of *medical information* that reflects changes triggered by specific telemedicine manipulation.

So far, in this section, we have provided a general system analysis for conference-oriented telemedicine systems, the layered architecture in Fig. 2 for separation of design considerations, a block architecture in Fig. 2 for the refinement of layered architecture, and, finally, a scenario in Fig. 4 for the explanation of the telemedicine manipulation mechanism. We will further introduce a framework called *TeMeFr* in the following section, which is a concrete implementation according to the analysis and generic models described in this section.

3. ***TeMeFr*: AN ARCHITECTURAL FRAMEWORK FOR TELEMEDICINE SYSTEMS**

TeMeFr serves as a purely telemedicine client for distributed medical data sources. Externally, it communicates with DICOM image servers, RIMS, or other medical data sources using standard DICOM protocol for the acquisition of requested medical records. Internally, XML is used as the major data format for information exchange among components that constitute the entire *TeMeFr*. From a functional perspective, *TeMeFr* facilitates telemedicine services by providing functionalities of teleradiology, teleconsultation [11-13], diagnostic reporting [14], etc. The kernel part of *TeMeFr* is a robust implementation of Conference-Oriented Management of Image Studies (COMIS), which manages medical records as a conference session for teleconsultation to provide more context-aware information and meaning of medical records for clinical works. We give a more comprehensive discussion of features in section 4.

3.1 ***TeMeFr*: The Software Architecture Overview**

The kernel part of the *TeMeFr* evolved using design solutions for repeatedly occurring problems of telemedicine systems. We found these design problems common to most telemedicine systems, some relate to architectural aspect of a telemedicine system, while others are issues to be considered from the data abstraction view. The following is a summary of these commonly occurring design problems in telemedicine:

1. To provide an abstract view of diverse medical data sources, this data abstraction view must be adjustable for different purposes of telemedicine and flexible for configuration of application.
2. Internal policies for medical data acquisition must be adjustable for different workloads of various telemedicine applications.
3. The preservation of data privacy must be a fundamental requirement of component's operation.
4. Architectural robustness has to be well planned in order to provide a reliable platform for possible telemedicine practices.
5. Coordination and management of various telemedicine clients must be configurable for different types of telemedicine activities.
6. Built-in functional components should also be replaceable by other customized components which possess equivalent capabilities and reflect interface conformity.
7. Components that facilitate specific telemedicine needs should be flexibly integratable atop the infrastructure.

The software architecture of *TeMeFr*, as shown in Fig. 5, is a general resolution of these common design problems in telemedicine domain. These design problems can be further categorized into three levels of concerns: the data-abstraction level, the control logic level, and the application level. In order to separate diverse levels of design problems, we utilize the typical *three-tier architecture* as the skeleton of *TeMeFr*. The primary advantage of using a three-tier architecture is that designers can easily switch policies or strategies of different tiers and localize unnecessary change propagation.

The presence of *COMIS Persistence Layer*, the bottom layer of *TeMeFr* software architecture, provides an abstract view of medical data sources and handles issues related to the data-abstraction level. The middle layer of *TeMeFr*, *Session Management Layer*, is a realization for control logic level design concerns, which acts as a configurable channel between the top and bottom layers of *TeMeFr*, and provides a flexible mechanism with supportability for managing telemedicine practices. The top layer is the *TeMeFr Application Layer*, which aims at the deployment of functional components and their interoperability with other layers.

3.2 *TeMeFr*: As a General Resolution for Common Design Problems in Telemedicine Domain

3.2.1 Providing an abstract view of diverse medical data sources

The *COMIS component* in *COMIS Persistence Layer* of the software architecture is responsible for the provision of an abstract view of diverse medical data sources: external and internal. Externally, the *COMIS component* handles all connectivity problems with various medical data sources that serve as the data backend for telemedicine practices based on *TeMeFr*. Thus, the *COMIS component* complies with medical standards.

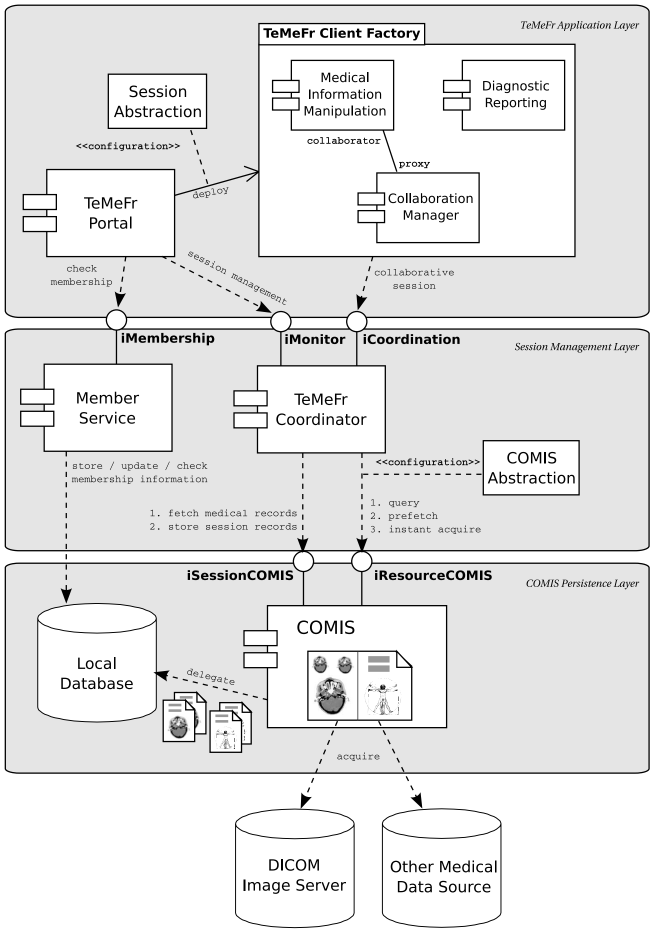


Fig. 5. Software architecture of *TeMeFr*.

Internally, the *COMIS component* organizes medical data in the unit of telemedicine practice, i.e., a conference session. Every telemedicine practice based on *TeMeFr* is treated as a conference session. While organizing a conference session, we have to configure the *COMIS Abstraction* to organize necessary medical data, which may vary in respect to different types of telemedicine practices, for holding a conference session.

3.2.2 Setting up tunable policies for medical data acquisition

The policies for medical data acquisition should be practical to suit various contexts of telemedicine activities. For example, medical data used in a distributed telemedicine system are usually stored in diverse medical data servers, some of them located outside hospitals and even be sheltered by firewalls. In order to start a conference session on schedule, the medical data acquisition is usually performed before the scheduled date, and pre-fetched medical data would be moved to local database for temporary storage. When the conference session is activated, the access to medical data would be a local process within the infrastructure, avoiding the latency of traveling through outside networks. The pre-fetching processes scheduled for different conference sessions will be managed by the *TeMeFr Coordinator component* to avoid possible resource race conditions.

When the context of telemedicine practices is switched to applications with centralized fashion, in which medical data sources may be located in the same intranet as the telemedicine system serving the conference session, the pre-fetching policy would not be necessary and could be turned off. In such systems, the *TeMeFr Coordinator component* has to play a heavy role in the coordination of on-demand acquisition policy. Both the pre-fetching policy, on-demand acquisition policy and even other customized policies are part of the configuration presented in the *COMIS Abstraction*.

3.2.3 Preserving medical data privacy

Data privacy is always a first-priority issue for informatics, especially in the telemedicine domain. When a conference session is in progress, acquired medical data may contain information related to patient's privacy.

There are two key designs for the preservation of medical data privacy. First, when the *COMIS component* acquires medical data from medical data sources, headers of medical data, which are irrelevant to the data itself, are suggested to be filtered out before entering local storage or before further processing by other components. Second, for consultation purposes, medical data would be transferred to various clients in a conference session, and these data will be erased from local storage after the termination of conference session, which is controlled by the *Collaboration Manager component* deployed to every destined clients along with other functional components in the *TeMeFr Client Factory package*. The filtering strategy of medical data configuration is also part of the *COMIS Abstraction*, while the erasure of transferred medical data is a configurable option in the *Session Abstraction*.

3.2.4 Planning for architectural robustness

When utilizing telecommunication technologies in telemedicine practices, the architectural robustness of a telemedicine system is challenged by unexpected obstacles for the proceeding of a conference session, for example, the network failure among connections within a conference session or operational exceptions occurring in clients of telemedicine practices. In the design of *TeMeFr*, we focus on architectural robustness.

We have implemented several *Fault Tolerance* mechanisms for reliability. For example, the conference session will continue disregarding some network problems caused by arbitrary disconnection of clients or exceptions caused by re-entrance of disconnected clients.

3.2.5 Supporting configurability for coordination and management of telemedicine activities

The design problems related to coordination and management of telemedicine activities are mainly realized in the *TeMeFr Coordinator component*, the *Collaboration*

Manager component, and their interactions during a conference session. The policies of coordination and management of telemedicine activities are configurable through specification options of the *Session Abstraction* according to the characteristics of different telemedicine activities. In addition, there are options for role-based access-right control, which is achieved with the help of the *Member Service component*. The *TeMeFr Coordinator component* and the *Member Service component* are located in the *Session Management Layer* of *TeMeFr*, so that mechanisms implemented for reliability or other architectural robustness concerns can be easily replaced or integrated into the current design of *TeMeFr* with change propagation minimized inside this layer.

3.2.6 Allowing software componentry based on interface conformity

Software componentry is a realization of component-based development, where, like the idea of hardware components, components are made replaceable, interchangeable, and reliable. This goal is reached by clear declaration of component interfaces. In *TeMeFr*, components are encapsulated by their interfaces that define unique responsibilities and capabilities and are well-organized in different layers of the software architecture. As long as the interface conformity is complied with, components will be interchangeable, replaceable, and customizable.

3.2.7 Providing extensibility for various telemedicine needs in application level

This design problem is an extension of a previous one, *Allowing Software Componentry Based on Interface Conformity*, and could be solved based on interface conformity with an extended design. This happens primarily in the *TeMeFr Application Layer* of the *TeMeFr* architecture, and we have packaged components that satisfy different telemedicine needs into a single software package, the *TeMeFr Client Factory package*. The extensibility is achieved by using an *Abstract Factory Pattern*-like design for the creation of deployable packages. According to different needs of telemedicine systems, domain specific components are dynamically selected and integrated on-the-fly as a deployable package to telemedicine clients, where the *Abstract Factory Pattern*-like design serves such creation process.

4. COMPARISON OF SYSTEMS

We have described the overview of *TeMeFr*, its software architecture, and its common design issues. In this section, *TeMeFr* is compared with other related telemedicine systems.

4.1 Related Works

Table 1 presents a representative summary of features in each of the related works and *TeMeFr*. A √ means the feature is generally implemented by that system, and × means the feature is not implemented. Symbol • indicates the implementation of a particular feature is not for general purposes but only for specific context or certain consideration.

Table 1. Feature Comparison.

Feature	TeMeFr	TeleMed	JReads
General image viewer	√	√	√
Image manipulation	√	√	√
Runs in a browser	√	√	√
Platform independence	√	√	√
Complete access to patient records	•	√	×
User authentication	√	√	√
Realtime teleconsulation	√	√	×
Session management	√	×	×
VPR concept	•	√	×
COMIS view	√	×	×
Potential for data mining	√	√	×
Fault tolerance	•	×	×
Encryption of patient data	√	√	•
Patient record privacy	√	×	×

It lists features that are viewed as significant through the development and evolution of *TeMeFr* and is definitely not an exhaustive list of functionalities of any of these systems. Each system indeed has its own purpose, context, and focus that differs from others.

TeleMed [15] reaches the goal of assembling virtual patient records [16] (VPR) from scattered data sources. TeleMed brings together all relevant medical records of a specific patient and presents them to the user in a unified visualization, which is based upon the VPR concept. VPR is provided to a physician as a complete graphical patient record that includes pharmacokinetic treatments, bacteriology reports, and clinical reports, as well as a complete set of CT sequences over the treatment history of the patient.

JReads [17] is a medical image viewer which is capable of functioning in a teleradiology environment, with necessary image manipulation capabilities and additional image analysis tools. Rather than provide a higher level view of patient records as does TeleMed, JReads focuses on just teleradiology which is known as a very important aspect of telemedicine.

The *TeMeFr* proposed in this paper encapsulates teleconsultation and teleradiology facilities with reliability, security, and privacy needed for critical requirements of health care in the telemedicine domain, which is achieved via fault tolerance, access control, and the implementation of COMIS view. Features listed in Table 1 could be further divided into three categories which will be discussed in detail in the following subsections.

4.2 Image Viewer for Teleradiology

The first category of Table 1 is the feature set that is generally possessed by image viewers for teleradiology. In spite of the feature *Complete access to patient records*, the *TeMeFr* and TeleMed are functionally equivalent with JReads as a general medical image viewer. *TeMeFr* gets a bullet • on *Complete access to patient records* because of *Patient* *record privacy* and *User authentication* based on the mechanism of access control. Since there is no particularly individual user of a telemedicine system that has full access to all possible patient records, we implemented a conditional access control mechanism, incorporating the feature *User authentication* for access to patient records, which is similar to the mechanism of *Role/Access Rules* in a content management system.

4.3 Facilities for Teleconsultation

Not surprisingly, JReads gets all × in the second category of Table 1, which stands for features of teleconsultation, since JReads is considered to be a general medical image viewing system. The differences between *TeMeFr* and TeleMed originate from the handling of patient records introduced in these two systems: While we create a COMIS view in the *TeMeFr* implementation, TeleMed proposes the concept of Virtual Patient Records (VPR). COMIS, which manages patient records from the perspective of a specific conference session for teleconsultation, can be considered as a specialization of VPR that organizes patient records with a granularity of the individual patient. Though VPR presents a more complete view of patient history, the primary advantage of COMIS is that it provides more context- aware information and meaning for clinical works based on telemedicine technology.

The *Session management* feature of *TeMeFr* provides a workflow-like management of the conference session for teleconsultation, which includes preserving the conference session, notification and establishment of a session, coordination or concurrency control during the session (based on a simple token mechanism), voting after consultation for result storage and conclusion, and finally reporting the diagnosis based on medical records acquired during the session. In order to provide a reliable environment, we also implement a fault tolerance mechanism to deal with network failures or possible data or message loss in *TeMeFr*.

4.4 Non-functional Requirements on Reliability, Security, and Privacy

Medicine is more conservative than other fields when it comes to adopting information technology. Thus, design and implementation of telemedicine systems should share common characteristics with mission-critical systems or safety-critical systems, which demand usability, reliability, security, etc., not only supporting the high quality of care but also preserving as much traditional culture of medicine as possible.

In *TeMeFr*, we implement the necessary *Fault tolerance* mechanism specifically for the reliability of conference sessions for teleconsultation, which helps prevent problems due to inadequate coordination, message dispatching, and concurrency control. Currently, *Fault tolerance* is supported by the conference session for teleconsultation (that's the reason why we put a • here), and we have planned to extend the scope of current support for *Fault tolerance* in order to evolve the *TeMeFr* to be more reliable and usable. While both TeleMed and JReads are designed for general purpose, this may be the reason that neither of them implements a fault tolerance mechanism. *Encryption of patient data* is a fundamental requirement for any telemedicine system that may break the border of hospitals, regions, or nations through the Internet. As shown in the bottom category of Table 1, all three systems share this basic feature.

Though VPR may be a trivial, natural view of managing patient records stored in a distributed medical repository, it indirectly causes the problem of *Patient record privacy*, and this would encounter some legal problems over protection of medical data. Medical data that are related to patient privacy, such as information in the header of a DICOM image, would be filtered out before being sent from the conference session for teleconsultation. Moreover, medical data will not be stored locally to help ensure *Patient record privacy*. Medical data retrieved or dispatched through COMIS would only be stored in the main memory of each client and will disappear with the termination of the conference session.

4.5 Implementation Details

Accidentally, all three telemedicine systems discussed above, namely, TeleMed, JReads, and *TeMeFr*, chose the Java programming language for the implementation. JReads and TeleMed adopted CORBA in their architectural designs, and TeleMed utilizes a specific CORBA implementation, CORBAmed, which is specialized for medical needs. While we did not apply specific Java technology in the implementation of *TeMeFr*, we embeded COMIS in its software architecture and chose XML as the major data format for internal data exchange. Such decisions help *TeMeFr* eliminate the burden of being a vertical system as JReads or TeleMed, and preserves most portability, flexibility, and extensibility.

5. SUMMARY AND CONCLUSIONS

A conference-oriented paradigm helps to capture suitable abstraction of user requirements for systems that support telemedicine practices. In this paper, we conducted a general analysis based on a generic model for conference-oriented telemedicine systems. By doing so, common design concerns were identified and a layered architecture was proposed for separation of concerns. Along with observations of publicly available telemedicine systems, the generic model for conference-oriented telemedicine systems, the layered architecture, and common design concerns serve as the foundation towards reuse-based development for the telemedicine domain. As the main part of this paper, we proposed an architectural framework, *TeMeFr*, for the purpose of supporting reuse-based development for the telemedicine domain. This framework is designed and realized to provide general resolution to seven common design problems derived from the generic model for conference-oriented telemedicine systems, the layered architecture, and common design concerns.

In addition to the general resolution of design problems, *TeMeFr* provides an extensible platform for developing functional components for specialized telemedicine practices. We believe by reusing *TeMeFr* as a software architecture for developing telemedicine systems developers are supported with a robust basis, can pay more attention to specialized functional components, and thus will be rewarded with most of the benefits of architectural reuse, such as increased software quality, shortened development life cycle and reduced development cost. Also, the generic model for conference-oriented telemedicine systems and the layered architecture will help in reasoning or studying currently available telemedicine systems through the conference-oriented paradigm, which may lead to possible reuse of telemedicine components for those systems.

REFERENCES

1. J. A. Chavarría, "OpenClinic: an open source clinic system," http://openclinic.sourceforge.net/.
2. Pennington Firm, "OpenEMR: open source EMR and practice management," http://www.openemr.net/.
3. J. Buchbinder, "FreeMED: free and open source software in medicine," http://www.freemed.org/.
4. A. P. Ho, "OIO: open infrastructure for outcomes," http://www.txoutcome.org/.
5. UCLA Medical Imaging Informatics, "DataServer: an open source XML gateway," http://www.mii.ucla.edu/dataserver/.
6. J. Buchbinder, "FreeB: a medical billing system," http://www.freeb.org/.
7. M. Proctor and G. Creech, "Object-oriented modeling of patients in a medical federation," *IEEE Transactions on Information Technology in BioMedicine*, Vol. 5, 2001, pp. 244-247.
8. E. Karavatselou, G. Economou, C. Chassomeris, V. Mylonas, and D. Lymberopoulos, "OTE-TS – a new value-added telematics service for telemedicine applications," *IEEE Transactions on Information Technology in BioMedicine*, Vol. 5, 2001, pp. 210-224.
9. E. Carson, D. Cramp, A. Morgan, and A. Roudsari, "Clinical decision support, systems methodology, and telemedicine: their role in the management of chronic disease," *IEEE Transactions on Information Technology in BioMedicine*, Vol. 2, 1998, pp. 80-88.
10. C. Combi, G. Cucchi, and F. Pinciroli, "Applying object-oriented technologies in modeling and querying temporally oriented clinical databases dealing with temporal granularity and indeterminacy," *IEEE Transactions on Information Technology in BioMedicine*, Vol. 1, 1997, pp. 100-127.
11. P. Shah, R. Martinez, and B. Zeigler, "Design, analysis, and implementation of a telemedicine remote consultation and diagnosis session playback using discrete event system specification," *IEEE Transactions on Information Technology in BioMedicine*, Vol. 1, 1997, pp. 179-188.
12. J. Bai, Y. Zhang, and B. Dai, "Design and development of an interactive medical teleconsultation system over the world wide web," *IEEE Transactions on Information Technology in BioMedicine*, Vol. 2, 1998, pp. 74-79.
13. J. Zhang, J. Stahl, H. Huang, X. Zhou, S. Lou, and K. Song, "Real-time teleconsultation with high-resolution and large-volume medical images for collaborative healthcare," *IEEE Transactions on Information Technology in BioMedicine*, Vol. 4, 2000, pp. 178-185.
14. E. Neri, J. Thiran, D. Caramella, C. Petri, C. Bartolozzi, B. Piscaglia, T. D. B. Macq, G. Cosnard, B. Maldague, and J. Pauw, "Interactive DICOM image transmission and telediagnosis over the European ATM network," *IEEE Transactions on Information Technology in BioMedicine*, Vol. 2, 1998, pp. 35-38.

15. D. W. Forslund, J. E. George, E. M. Gavrilov, T. A. Staab, T. E. Weymouth, and S. Kotha, "TeleMed: development of a Java/CORBA-based virtual electronic medical record," in *Proceedings of the Symposium on Pacific Medical Technology*, 1998, pp. 16.
16. D. G. Kilman and D. W. Forslund, "An international collaboratory based on virtual patient records," *Communications of the ACM*, Vol. 40, 1997, pp. 111-117.
17. S. P. Laird, J. S. Wong, S. William J, B. J. Erickson, and P. C. de Groen, "Design and implementation of an internet-based medical image viewing system," *Journal of Systems and Software*, Vol. 66, 2003, pp. 167-181.

Hewijin Christine Jiau (焦惠津) received the B.E. in Electrical Engineering from National Cheng Kung University, M.S. in Electrical Engineering and Computer Science from Northwestern University, and Ph.D. in Computer Science from the University of Illinois at Urbana-Champaign. She is an Associate Professor in the Department of Electrical Engineering, National Cheng Kung University (NCKU), Taiwan. Before serving at NCKU, she was an Assistant Professor in the Department of Information Engineering and Computer Science, Feng Chia University, Taiwan. Her research interests include software reuse, object technologies, information integration, data mining, and database applications on the Internet. She is a member of the IEEE and the ACM.

Jinghong Cox Chen (陳敬泓) received the B.E. degree in Electrical Engineering from National Cheng Kung University, Taiwan. He is currently pursuing the Ph.D. degree in the Department of Electrical Engineering, National Cheng Kung University, Taiwan. His research interests include software reuse, object technologies, design patterns, multi-paradigm design, information architecture, and software componentry.

Kuo-Feng Ssu (斯國峰) received the B.S. in Computer Science and Information Engineering from National Chiao Tung University and Ph.D. in Computer Science from the University of Illinois at Urbana-Champaign. He is an Associate Professor in the Department of Electrical Engineering, National Cheng Kung University, Taiwan. His research interests include dependable systems, mobile computing, and distributed systems. He is a member of the IEEE, the ACM, and the Phi Tau Phi honor scholastic society.

Jim-Min Lin (林志敏) received the B.S. degree in Engineering Science and the M.S. and the Ph.D. degrees in Electrical Engineering, all from National Cheng Kung University, Tainan, Taiwan, R.O.C., in 1985, 1987, and 1992, respectively. Since 1993 he has been with Feng Chia University, where he is currently a professor of Department of Information Engineering and Computer Science. During August 2001 and July 2003, he was on-leave and served as the Chairman of the Department of Information Management, Asia University, Taichung, Taiwan. He also serves as the Associate Editor-in-Chief of the Journal of Computer, Computer Society of the R.O.C., since January 2006. His research interests include operating system, software integration/reuse, embedded systems, software agent technology, and computer arithmetic.