An integrated OR system based on open standards

Release 1.00

Stefan Bohn, Michael Gessat, Stefan Franke, Arun Voruganti and Oliver Burgert

July 10, 2009

Innovation Center Computer Assisted Surgery (ICCAS), University of Leipzig, Germany

Abstract

An open standards based modular OR integration system was designed which is modelled as component-based service-oriented architecture. Medical devices, IT hardware and software are integrated as independent components and interconnected through a TCP/IP based Ethernet network. The integrated system facilitates service discovery, time synchronization, systems monitoring as well as component control and administration. Each of the components exposes its semantic information such as attributes, functions as well as information about offered services such as streaming with information on the used protocol as service description based on a generic model to peer components which can interpret and make use of these services. An open standards based programming library for the rapid development of server/client components was developed and a prototype implementation within a demonstrator OR lab realized.

Contents

1 Introduction

Today's operating rooms (OR) accommodate a wide variety of medical device hardware and software, modern image acquisition and processing technologies, IT systems and systems for computer assisted surgery (CAS). Unlike in other industry domains, there exists no communication platform within the OR, that enables interoperation among heterogeneous medical devices or IT components. The lack of a common OR infrastructure and the large amount of information sources within the OR leads to unergonomic and uneconomic conditions as well as an impaired surgical workflow. Besides focusing on the surgical intervention, the surgeon is required to mentally integrate all necessary information coming from various spatial distributed IT components. Furthermore, due to the absence of a common OR infrastructure, there is lack of a consistent data flow between preoperative, intraoperative and postoperative phases. Data, which already exist in clinical information systems (e.g. hospital information systems or radiological information systems), need to be entered manually into several OR systems. Portable media are used to transfer data from preoperative planning to intraoperative application as well as interoperatively acquired data to postoperative documentation. These limited state of the art approaches are highly susceptible to type errors or mixing of patient identification information and thus can cause undesirable data inconsistencies in clinical information systems and PACS archives.

Together, all affected participants across the medical device industry, clinical end-users, and researchers recognized the benefits, which would come from plug-and-play device interoperability within the OR as well as ergonomic centralized user access to an integrated system. Several commercial vendors provide such integrated operating rooms (e.g. BrainLab, Storz, Stryker) which mainly integrate their own components using proprietary interfaces. Although a wide variety of communication protocols and standardized physical interfaces for systems intercommunications are available, none of them are established or commonly accepted in the medical device domain until now. There exists no generic communication framework, which provides communication protocols for syntactic and semantic interoperability among medical devices or CAS systems.

The aim of our work is the design and implementation of an integrated OR infrastructure as research platform, which is based on open standards and protocols. This should show the feasibility of vendor-independent system intercommunication within the medical domain and especially in the operating room. Clinical users ask for an efficient and user friendly system that can be applied in different contexts depending on the surgical intervention. Therefore we focus on the generic design of an autonomous and fault-tolerant infrastructure, which can assert safety, reliability, robustness and reconfigurability.

2 Architecture

The development of the OR integration concept started with an analysis of the required system components [1], surgical workflow analysis's [2] and selected clinical use cases [3]. The design of our modular OR integration infrastructure follows the Therapy Imaging and Model Management System (TIMMS) meta-architecture, which was published by Lemke and Vannier in 2006 [4]. It introduces several engines, repositories and communication channels, suitable for all kinds of computer assistance in surgery and therapy.

From this we derived a concrete modular integration architecture, which consists of a system of distributed modules (hardware and software) as well as several system core components. The architecture of the integrated OR system is shown in Figure 1. In order to achieve vendor-independent data exchange, the modules interact using a set of standard protocols for session management, data exchange, remote control, time synchronization as well as systems monitoring. Since the OR is dominated by an ad hoc context, the overall system needs to cope with the absence of any a-priori knowledge about possible peers and particular communication mechanisms.

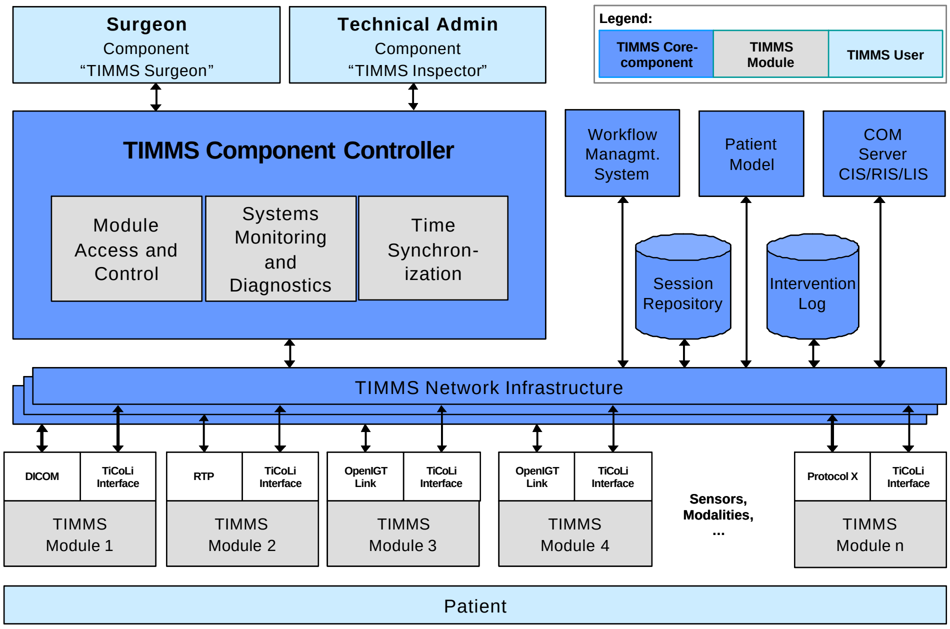

Figure 1 The architecture of the integrated system with TIMMS modules, network infrastructure, core components and end users.

2.1 TIMMS Modules

Modules which act on the patient during the intervention, such as sensors, imaging modalities, or CAS systems consist of hardware and software of various kinds. Each of them strongly differs in the requirements regarding data exchange with peer modules. Online data sources require real-time communication of streaming data, while modelling modules or image acquisition devices produce large data objects at irregular intervals. These need to be transported from one module to another resulting in heavy network load for a short period. At the same time, status messages are exchanged in regular or irregular intervals which also need to be transmitted. These applications bring along different requirements on all levels of the Open Systems Interconnection (OSI) Reference Model. To satisfy all requirements at the same time, the TIMMS backbone network infrastructure needs to be implemented using different technologies on all levels of the OSI Reference Model.

To integrate these various technologies into a homogeneous framework over the complete system operation cycle, each module is abstracted based on a generic device model [5]. The device model describes the functional capabilities of each module, its data, behaviour and the used communication protocols in a generic manner. The device model is exported as service description object to peer modules during the initial negotiation phase of the intercommunication. Thus, using technologies of auto-configuration, TIMMS modules can seamlessly join the network, discover peer modules, connect and retrieve the device description, learn and make use of the offered services to perform their particular tasks within the integrated system.

2.2 TIMMS Core Components

The integrated OR system contains several core components which provide the overall integrated functionality to the end-user such as clinicians as well as the technical administrator. Furthermore the core components perform several managing functions, such as maintaining the patient context, data exchange with clinical information systems, time synchronization and logging.

The TIMMS Component Controller (TCC) is the central managing component which administrates and supervises all interconnected modules within the integration system. The TCC tracks the system health status of the entire networked modules, monitors the data flow on the network and ensures rights and access control for security and safety. Additionally, it provides a time server in order to keep modules performing time critical tasks synchronized. The TCC models and controls the integrated system and allows different user groups to view and access the OR functionality from different perspectives at different workstations. On the one hand, the clinical user at the surgical cockpit gets a condensed view, visualization and smart control of the OR functionality where technical details are abstracted to an appropriate level. The technical or administrator user on the other hand has an in-depth view and access to all parameters of the integrated TIMMS system and is able to adopt specific configuration issues on demand.

The integrated OR system is required to exchange data with different clinical information systems and repositories to enable a consistent data flow between preoperative, intraoperative and postoperative phases. Therefore, a COM-Server maintains access to the IT world outside the OR, e.g. to retrieve patient information from the hospital information system, worklists from PACS archives as well as to store OR reports and acquired data back for surgical documentation and reporting.

The context module and session repository maintain all information about the current intervention, patient information and associated files, e.g. planning data or device profiles. The session repository stores linking information about all data which is generated during the intervention by the TIMMS modules, e.g. screenshots, recordings of biosignals etc. After the intervention the surgeon obtains an overview of all acquired information to choose items which should be used for surgical documentation.

The intervention logger provides a central logging service to record all intervention relevant information from the TIMMS modules and core components, such as the appearance of alarms, user inputs, systems failures and status data for retrospective analysis of the intervention as well as for OR documentation.

Later versions of the integration system will encompass further core components such as a workflow management system for knowledge support and intelligent systems control as well as the patient model component that integrates the diversity of anatomical and functional patient data into one unique representation available for the surgeon.

3 Implementation

The use and application of established communication standards is one of the main requirements for the successful implementation and acceptance of an integrated system. Intercommunications within the proposed integrated OR system are based on a set of standard protocols on top of a TCP/IP based Ethernet network. A C++ library was developed as a general communication interface for TIMMS modules. The TIMMS Communication Library API (TiCoLi API) is a hybrid library, which combines several existing open-source libraries in order to combine the strengths and compensate for the weaknesses each of the used libraries has.

3.1 TiCoLi API Protocol Stack and Services

Figure 2 shows the TiCoLi API protocol stack. TiCoLi API provides a C++ application-level interface to the functionalities of the underlying protocols and specifies several services (e.g. Service Discovery, Message Exchange, Time Synchronization etc.).

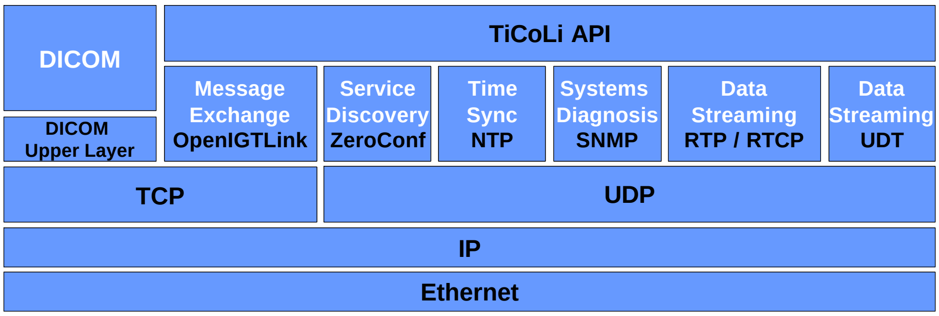

Figure 2 The Protocol Stack of the TIMMS Communication Library API (TiCoLi API) with services and corresponding protocols.

The TiCoLi API uses the OpenIGTLink Library which is provided by NA-MIC [7], the open-source RTP implementation jRTPLIB, POSIX Threads, the ZeroConf implementation BonjourSDK, the TimyXML library and the SNMP implementation of Agent++.

3.2 Service Discovery

Each of the TIMMS modules is required to automatically detect services with certain characteristics within the network. Automatic configuration and plug-and-play service discovery of TIMMS modules are realized using ZeroConf [6]. Within ZeroConf the DNS-Service Discovery (DNS-SD) protocol is applied, which specifies how DNS service messages are used to describe the services a module is offering. Modules which enter or leave the network send short status messages to all components on the network. Thus, modules can be added or removed at runtime without affecting the overall network integrity. The TiCoLi API offers a set application level service primitives to register and discover services within the integrated system.

3.3 Service Description

The service description of a server module is a document which contains information about the services (messaging, streaming, attribute access, method calls) the module currently offers via TiCoLi. The TiCoLi API provides a programming interface where each of the particular variables, methods and streams are registered during server initialization. The service description object is then transparently assembled at runtime. Each server module holds one default message socket open (as registered with ZeroConf) at which peer devices can connect and retrieve the device description.

3.4 Message Exchange, Remote Procedure Calls and Attribute Access

TiCoLi API specifies an acknowledged message exchange service which is based on OpenIGTLink [7]. The message exchange service can be used to exchange the service description objects as well as to exchange application-level messages between client and server. TiCoLi holds references to all registered objects in the service description objects (see 3.3) to transparently facilitate reading or writing of attribute values, execution of remote procedure calls, as well as to initiate and control data streams. The TiCoLi API specifies a range of common data types for attributes (e.g. boolean, numeric, or character strings), status messages as well as service types and stream types.

3.5 Data streaming

The TiCoLi API implements an unicast and multicast streaming service based on the Real-Time Transport Protocol (RTP). RTP is an application-layer protocol which facilitates packet-based data exchange via the User Datagram Protocol (UDP) transport layer. RTP consists of two components, the Data Transfer Protocol and the RTP Control Protocol (RTCP) which is used for exchange of reception quality feedback and for synchronization. TIMMS modules can connect to streams offered by servers by sending a connection request via a previously opened messaging session (see 3.4). Quality of service (QoS) information such as stream and video characteristics are exchanged as a stream-description via messaging session prior to stream connection. The synchronization of the streams is handled by the TiCoLi internally. The streaming service continuously sends the frames to the subscribed client modules according to the frame-rate negotiated during QoS exchange.

3.6 Time Synchronization

Time synchronization within the integrated system is facilitated using the Network Time Protocol (NTP). NTP is designed to compensate for network latencies which are a common problem in packet-switched domains, such as IP based networks. NTP is based on a hierarchical client-server architecture with highly reliable clocks (such as atomic clocks) on the highest level. The implementation in the proposed integrated system follows the IHE Profile “Consistent Time”, where the TCC implements a grouped time server with which each of the modules can be synchronized.

3.7 Systems Monitoring und Diagnosis

The life critical domain within the OR demands safe and reliable operation of the integrated OR components. Therefore, the TiCoLi implements a technical supervision framework that facilitates systems monitoring and diagnosis of the integrated system. Management agents are used to acquire performance and diagnostic information at network backbone level, computer hardware level, and in software applications. The transfer of diagnostic information between management agents and the supervisory module of the TCC is based on the Simple Network Management Protocol (SNMP). Monitoring at code implementation level is facilitated using the Open Group Standard Application Response Measurement (ARM). The combination of ARM and SNMP enables software performance measurements as well as alive-state supervision using software watchdogs [8].

4 Results

As a result, the presented modular OR system integrates medical hardware and software components into a common research platform at different levels (data, functions, application). A prototype setup was established in a demonstrator OR laboratory (Figure 3). The TIMMS modules operate on standard PC hardware and are interconnected over a standard 100/1000Base-T Ethernet network.

Figure 3: Prototype implementation within ICCAS demonstrator OR lab.

Several software applications have been implemented as modules based on the TiCoLi API, such as a tracking streaming server for object tracking, an image-guided surgery application using navigation and patient modelling (Figure 4) [9], streaming of biosignals, a server for display and video routing, as well as PACS access to DICOM objects, e.g. imaging data or reconstructed surfaces from surgical planning software.

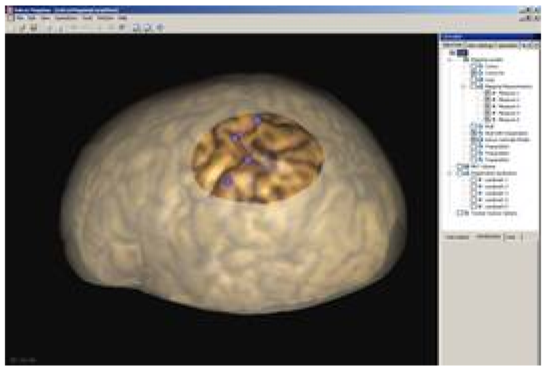

Figure 4: Image-guided-surgery application for brain tumor surgery [9].

The user interface for the clinical user provides centralized access and control using ceiling-mounted booms with touch screen displays (Figure 3). Here software applications and device functions are integrated based on remote display software (using Remote Framebuffer Protocol) as well as video routing technologies.

The user interface for the technical supervisor at the TCC software component provides an overview and access to all connected TIMMS modules (Figure 5). It displays the particular device models as well as enables access to configurable data elements. The supervising and monitoring module provides information to detect system anomalies such as network bottlenecks, cache and hard disc space exceeds or CPU consuming software processes. The visualization comprises simple numerical values of performance measurements as well as graphical trend views for time-dependent values (e.g. network load).

Figure 5: TCC application with detailed view of service description of selected component and user interface for systems monitoring.

The integrated OR system itself is independent of any particular surgical discipline. Due to its modular architecture, the system can be adapted to specific clinical requirements and applications on demand.

5 Conclusion

A generic open standards based infrastructure which supports the integration of components for computer assisted surgery has been developed. The programming interface TiCoLi API enables the rapid development of CAS systems for the proposed integration architecture using methods of auto-configuration, plug-and-play service discovery, message exchange and device control, streaming as well as systems supervision and time synchronization. Since the OR is characterized by rapid changing technologies the proposed systems integration architecture aims for the development of a stable long-term integration solution which can be used in different surgical disciplines. Standard communication protocols as used within the TiCoLi API and technology independent service descriptions offer the potential to solve this issue while providing sufficient flexibility due to the independency of special, programming languages, operating systems or network technologies. Future developments aim on the investigation of semantic interoperability as well as workflow assisted interventions.

Acknowledgments

The Innovation Center Computer Assisted Surgery (ICCAS) at the Faculty of Medicine at the University of Leipzig is funded by the German Federal Ministry for Education and Research (BMBF) and the Saxon Ministry of Science and Fine Arts (SMWK) within the scope of the initiative Unternehmen Region with the grant numbers 03 ZIK 031 and 03 ZIK 032.

References

[1] Korb W, Bohn S, Burgert O et al. *Surgical PACS for the Digital Operating Room - Systems Engineering and Specification of User Requirements*, Stud. Health. Technol. Inform., Vol. 119, pp. 267-272, 2006.

[2] Neumuth T, Jannin P, Strauss G, Meixensberger J, Burgert O. *Validation of Knowledge Acquisition for Surgical Process Models*, J Am Med Inform Assoc, Vol. 16, No. 1, pp. 72-80, 2009.

[3] Burgert O, Korb W, Gessat M et al. *Real Time Issues for Usage of Vision and Image Data in the Future Operating Room*. IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems, Heidelberg, Germany, 2006.

[4] Lemke H, Vannier M. *The operating room and the need for an IT infrastructure and standards.* International Journal of Computer Assisted Radiology and Surgery, Vol. 1, No. 3, pp. 117 - 121, 2006.

[5] Bohn S, Burgert O, Mayoral R. *Towards a Generic Systems Integration and Device Operation Model for the Digital Operating Room*". International Journal of Computer Assisted Radiology and Surgery, Vol. 3, Supplement 1, 2008.

[6] E. Guttman. *Zero Configuration Networking*. Proc. INET 2000, Internet Society, Reston, VA; available at http://www.isoc.org/inet2000/cdproceedings/3c/3c_3.htm, Last visited 03/06/2009.

[7] Tokuda J, Yaniv Z, *OpenIGTLink. In IGSTK Image-Guided Surgery Toolkit: An Open-Source C++ Library.* (Kevin Cleary, Patrick Cheng, Andinet Enquobahrie and Ziv Yaniv eds.), Department of Radiology, Brigham and Women's Hospital, Harvard Medical School, Boston, MA, USA, 2009.

[8] Bohn S, Lessnau M, Burgert O. *Systems Monitoring and Diagnosis of Digital Operating Room (DOR) Equipment using Supervisory Control and Data Acquisition (SCADA) Technology.* International Journal of Computer Assisted Radiology and Surgery, Vol. 4, Suppl. 1, 2009.

[9] Franke S, Bohn S, Strauß G, Burgert O, Gessat M. *Navigated mapping of streamed sensor data.* International Journal of Computer Assisted Radiology and Surgery, Vol. 4, 2009.