OR.NET RT: How Service-Oriented Medical Device Architecture meets Real-Time Communication

Jonas H. Pfeiffer[1], Martin Kasparick[2], Benjamin Strathen[3], Christian Dietz[1], Max E. Dingler[1], Tim C. Lueth[1], Dirk Timmermann[2], Klaus Radermacher[3], and Frank Golatowski[2]

Abstract— **Today's landscape of medical devices is dominated by stand-alone systems and proprietary interfaces lacking cross-vendor interoperability. This complicates or even impedes the innovation of novel, intelligent assistance systems relying on the collaboration of medical devices. Emerging approaches use the Service-Oriented Architecture (SOA) paradigm based on the internet protocol to enable communication between medical devices. While this works well for scenarios with no or only soft timing constraints, the underlying best-effort communication scheme is insufficient for time critical data. Real-time networks are able to reliably guarantee fixed latency boundaries for example by using time division multiple access communication patterns. However, deterministic real-time networks come with their own limitations such as tedious, inflexible configuration and a more restricted bandwidth allocation. In this contribution we overcome the drawbacks of both approaches by describing and implementing mechanisms that allow the two networks to interact. We introduce the first implementation of a medical device network that offers hard real-time guarantees for control and sensor data and integrates into SOA networks. Based on two application examples we show how the flexibility of SOA networks and the reliability of real-time networks can be combined to achieve an open network infrastructure for medical devices in the operating room.**

I. Introduction

Medical devices capable of exchanging data and understanding each other can lead to various new ideas and innovations that have the potential to assist surgeons and to significantly improve patient outcome [1].

The benefits of collaborating medical devices can have even more impact, when this collaboration takes place under real-time guarantees. These become necessary for closed-loop control scenarios and whenever the patients safety relies on the data exchange.

Throughout this paper we introduce three guiding application examples with real-time requirements (see Section I-A). These examples rely on different types of data exchange whereby only some of the data has to be transferred with hard timing constraints, to allow safe medical systems. A reasonable amount of data does not need these constraints and thus does not have to bear the extra costs and efforts of RT communicaton. A Solution can be a hybrid networking architecture with a coexistence of hard RT communication and more flexible service-oriented data exchange, that we will introduce in this work.

In Section I-B we first classify different types of data with respect to timing requirements and safety considerations. Following the requirements analysis we introduce an architecture for the interconnection of a real-time communication network with a service-oriented medical device network in Section IV. The first successful implementation of this architecture is shown in Section V. It uses Ethernet Powerlink as real-time protocol and a web-service implementation based on the new IEEE 11073 SDC standards for the non-real-time network. Finally, Section VI presents another possible approach using Ethercat as real-time protocol.

A. Guiding Application Examples

[1]Author is with Institute of Micro Technology and Medical Device Technology, Technical University of Munich, 85748 Garching, Germany `firstname.lastname@tum.de`

[2]Author is with the Institute of Applied Microelectronics and Computer Engineering, University of Rostock, 18119 Rostock, Germany `firstname.lastname@uni-rostock.de`

[3]Author is with the Chair of Medical Engineering, RWTH Aachen University, 52074 Aachen, Germany `lastname@hia.rwth-aachen.de`

a) Configurable Footswitch: A well-known issue in everyday life in the operating room (OR) is the amount of cables underneath the operating table. Most of these cables originate from various footswitches used to activate a multitude of devices during interventions. The high number of cables underneath the operating table complicates the device set-up and is a well-known tripping hazard. One universal, networked footswitch capable of coupling with different devices could reduce the complexity and the amount of cabling underneath the table. When devices controlled by that networked foot pedal are able to harm the patient, a temporal and a value correctness of the control data will have to be guaranteed.

b) Navigated Control: The second scenario is a closed-loop control of active surgical instruments. A very common task in most surgical interventions is the removal of tissue. The surgeon is thereby frequently confronted with situations where delicate structures need to be identified and protected from damage. Unfortunately the intraoperative view on the situs in minimally invasive as well as in open procedures is often limited and the identification of preservable structures can be very challenging. A closed-loop control of an instrument's power dissipation could assist the surgeon and help to prevent patient damage. When a surgeon guides an active instrument towards critical and delicate patient structures, such as nerves, the power of the instrument could be reduced gradually and automatically. The method referred to as Navigated Control [2] uses real-time tracking information from a medical navigation system, to calculate the distance between a patient's risk structure and the active instrument.

Different clinical applications of closed-loop Navigated Control have been examined and showed very promising results. Kneissler et al. [3] describe the application and clinical evaluation of navigated milling in spine surgery. Meanwhile [4] shows the results of clinical evaluations for surgeries with Navigated Control at the lateral skull base.

One problem of Navigated Control applications is the costly and time consuming development. For every combination of active instrument and tracking system a new system-of-systems has to be designed and certified. With a vendor-independent and real-time capable networking solution, the integration of the different subsystems to a system-of-systems could be facilitated a lot and more devices could be controlled easily.

c) Surgical Robotics: The third scenario is the integration of robotic systems into networked operating rooms. Surgical robotic systems can particularly benefit from a networked accessibility of real-time information. An exemplary orthopedic intervention is a revision total hip replacement where the precise removal of femoral bone is a challenging task for the surgeon and carries risks for the patient. The use of networked computer assistance with surgical robotics provides support for the surgeon to perform the procedure in a more precise and safer manner, for example by using networked intraoperative multiplanar x-ray images and tracking information for closed-loop control [5][6].

Besides these three guiding applications, there are many other situations and use-cases requiring real-time behavior, when integrated into a medical device network. For example a networked closed-loop control of Patient Controlled Anelgesic (PCA) pumps with physiological sensor information as described in [7]. Or the synchronization of a x-ray machine with a ventilator to prevent the lungs' movements from blurring chest images [8], to name just two of them.

B. Analysis of Real-Time Requirements

To interconnect medical devices in the OR via an open communication infrastructure, different communication requirements have to be taken into account. For safety reasons, the control data for a surgical instrument need to be handled with a different level of precaution than a patient demographics data for example. Data transportation requirements can be defined by the possible consequences in a fault condition. A surgical instrument that does not stop for one second, because its control data arrives too late, might seriously harm the patient. Meanwhile the fact that patient's demographic information arrives one second delayed might not even be noticed by the user.

Generally speaking, one aspect of the real-time requirements for an open, integrated OR are different temporal requirements for varying types of data. Within the OR.NET project we asked the medical device manufacturers for the interface specifications of their products. For every interface they summarized the type, the cyclicality and the maximum tolerable latency of data and the possible consequences in a fault condition. From the different specifications, we distinguished different classes of data:

Class A) Data from clinic servers and patient related data, such as patient demographics or DICOM images, usually have the lowest demands concerning data transmission latency. Delayed data has usually no direct impact on the patient's safety. A traditional network based on best-effort transmission should well suit this type of data.

Class B) Meanwhile device configuration data and device parameters are required to be send fast enough to enable a smooth and comfortable usability of the devices. The end-to-end latency should reach values of approximately $300-400\,\mathrm{ms}$. As a delayed data does not harm the patient, this can be described as soft real-time requirement. Again a best-effort based network could fulfill this requirement.

Class C) Control data that activates devices or triggers the power dissipation of active instruments build another class of data. A loss or delay of such data could have serious consequences for the patient and the transportation thus requires hard real-time constraints. A temporal classification of allowed latencies very much depends on the application. Typical values are in the range of $10-100\,\mathrm{ms}$. This also includes the above mentioned examples of a networked footswitch and the automatic closed-loop control of active instruments. Of course all data needed to close such control loops require the same hard real-time constraint, e.g. surgical navigation systems delivering positioning information.

Class D) A last class of data may not be common in today's devices, but will become increasingly important in the future. Data that requires hard real-time constraints with transmission delays of $< 1\,\mathrm{ms}$. For example to enable a networked real-time control of medical robots with synchronization of multiple axes [9].

The four different classes are not supposed to have fixed latency borders. A classification of the data always depends on the respective application and on the possible consequences in fault conditions. According to the harmonized European standard EN 60601-1 for safety and effectiveness of medical electrical equipment, medical devices have to be designed so that single-fault conditions may not lead to unacceptable risks. False control data that is supposed to change the output power of an active medical instrument has the potential to harm the patient. In order to mitigate such unacceptable risks, applications including the control of active instruments and comparable devices need to be designed with a deterministic behavior. This becomes increasingly important in control scenarios where devices from different manufacturers are supposed to collaborate. Fault conditions might hereby immediately lead to questions of liability. Who is responsible for the failure? The controlled device, the controlling device or the underlying communication infrastructure?

In a deterministic infrastructure a temporal and value correctness are guaranteed. The source and sink of data are predictable at any given point in time, which facilitates the design of fail-safe systems and control scenarios.

II. Situation in the Medical Device Domain

In the strictly regulated field of medical device technology, manufacturers hardly ever enable others to control their devices in form of distributed systems. A cross-manufacturer collaboration of devices in the OR is very rare. Time critical data transmission is commonly achieved by means of simple electrical circuits. Manufacturers tend to market their systems in predefined, small setups (e.g. consisting of a surgical aspirator and its suitable footswitch) [10]. Often, systems of one and the same manufacturer differ substantially regarding their interfaces. Differences occur in the mechanical compatibility (dimensions of the interface, number of pins) as well as the type of signal transmitted (simple circuits, analog signals, digital signals via sophisticated communication protocols).

Following the trend of networked systems, the market leaders in the medical device sector developed proprietary solutions for so called integrated

ORs. Their features reach from efficient and ergonomic access and usability of devices, over a centralized adjustment of parameters, to simple data and image access throughout the OR. The used communication protocols and technologies are kept private and therefore impede a cross-manufacturer interoperability of devices.

III. Related Work

A. The OR.NET Project

The German flagship project OR.NET funded by the German Federal Ministry for Education and Research (BMBF) aimed at developing a comprehensive and vendor-independent interconnection of medical devices in the OR and the whole clinic. The interconnection should be built between medical devices as well as between medical devices and medical information systems. This paper deals with the device-to-device communication. As medical devices and medical applications are very heterogeneous, there is a big variety of communication requirements in terms of reliability, latency, and determinism. Therefore two different types of device-to-device interconnections have been developed:

1) Highly flexible SOMDA-based interconnection
2) Highly reliable real-time capable interconnection

The Service-Oriented Medical Device Architecture (SOMDA) has been developed based on the well-known paradigm of a Service-Oriented Architecture (SOA). The concept of the SOMDA makes the advantages of the SOA like interoperability, plug-and-play, loose coupling, scalability, reusability, maintainability, etc. available for medical devices. A concrete specification for the SOMDA paradigm has been developed and is currently in the process of standardization as a new part of the established IEEE 11073 family of standards. The specification is called IEEE 11073 SDC family or short SDC. Note that the OR.NET project also coined the term Open Surgical Communication Protocol (OSCP) which describes the same as SDC. The IEEE 11073 SDC family consists of three standards: The Medical Communication Profile for Web Services (MDPWS, IEEE 11073-20702) is derived from the OASIS standard DPWS. MDPWS realizes basic aspects like data transmission and dynamic discovery of devices as well as medical specific aspects considering safety requirement like safe data transmission (e.g. single-fault safety), data streaming for waveforms, and a compact data transmission [11]. The Domain Information & Service Model (IEEE 11073-10207) realizes the structural interoperability [12]. Additionally IEEE 11073-20701 defines the allover architecture and the binding between the two former mentioned standards. A detailed description of SOMDA and SDC as its realization is out of scope of this paper. Further information can be found in the given references and in more details in the work from Kasparick et al. [13] as well as Schlichting and Pöhlsen [14]. Software frameworks implementing SDC are available as open source projects: openSDC [15] (Java), OSCLib [16] (C++), and SoftICE [17] (Java, wrapper available for C#).

To realize the second type of device-to-device communication, the authors in [18] conceptualized the Surgical Real-Time Bus (SRTB) and realized a first reference implementation. The SRTB is a parallel communication infrastructure for deterministic and time critical data transfer. It uses a master-slave hierarchy with TDMA communication patterns and can be implemented with real-time Ethernet technology. The scope of this paper is to describe the interconnection between the SRTB and the (SOMDA based) SDC standard.

B. Non-Medical Real-Time SOA

Using SOA in real-time systems is not new. First results of transforming IT technologies have been established especially in industrial automation. Projects like SIRENA, SODA, SOCRADES, IMC-AESOP, and ARROWHEAD contributed in many ways to the embedded web service technologies especially in conjunction with real-time requirements.

Jammes [19] analyses the needs of bringing SOA to embedded devices with real-time capabilities. In his paper Jammes states that a lot of research work has been done to improve the responsiveness of embedded Web Services that are available in IP networking environments. He explains that in order to use a SOA at the lowest levels of process control applications, the targeted performance should be around 1 ms. To achieve this goal a very high performance improvement of the existing solutions is necessary. Five possible solutions to reach that goal were mentioned:

1) SOAP/XML messages with e.g. EXI
2) Hardware-supported XML parsing, by using dedicated processor instructions
3) Simply increase CPU resources and performance.
4) Use of QoS (Quality of Service) priorities to be used both at the device (stack) level and at the network level.
5) Make use of formal approaches to real-time programming.

Cucinotta et al. proposed in [20] a Real-Time SOA to combine DPWS with real-time services. In this architecture real-time communication is executed on a separate channel. Real-time services are directly executed on the lower layer on top of UDP/IP. Web Services instead uses the complete DPWS stack. Parallel to the service systems that are built on top of IP domain specific field busses are supported by the architecture. In their work two separate APIs where used: a common and custom API. The common API supports functions related to Web Services and real-time services in combination with QoS management that ensures real-time capabilities on IP level. The custom API is used for the integration of legacy field bus communication.

Durkop et al. [21] proposes instead of device-level SOA a module-level SOA for the integration of a SOA into real-time systems for industrial automation. The real-time communication is realized on field level and is integrated into the SOA via a module. A Module can be seen as an adapter integrating a proprietary real-time networking device into the IP-network and the external SOA. By this, field devices (e.g. PLCs) are extended with Web Service interfaces and its control logic can be accessed from external SOA.

It can be concluded that the combined usage of a highly flexible interconnection based on a SOA and a highly reliable interconnection based on specialized real-time Ethernet solutions reflects the state of research. To the best of our knowledge, there is no solution available that can hold all requirements of a medical device interconnection in terms of reliability, latency, and determinism using one SOA-based interconnection for all communication. Thus, we follow the approach of dividing the network traffic into the SOMDA interconnection using standard IP networks and the SRTB interconnection using real-time Ethernet.

IV. OR.NET RT: Non-Real-Time SDC Meets SRTB Real-Time Network

Based on the requirements analysis presented in Section I-B and the analysis of available technical solutions (see sections II and III) the network traffic has to be divided into two parts and transferred via different networks: traffic with hard real-time requirements and small amount of data on the one hand and traffic without hard real-time requirements and a huge variety of different traffic patterns (small bandwidth up to very high bandwidth necessary, different periods, events, etc.) on the other hand.

To ensure that the non-real-time network cannot affect the real-time communication, two distinct networks are necessary. At least this requires a logical separation between these networks using technologies like VLAN. Typically this will be done by two physically divided networks.

The SDC-based interconnection implementing the SOMDA is realized using an non-real-time network. It is characterized by high flexibility in terms of plug-and-play functionality, dynamically appearing and disappearing medical devices and functionalities, heterogeneous data rates and traffic pattern, dynamically changing associations between control elements and controlled devices, etc. In contrast to this flexibility the real-time network referred to as SRTB has to be more restrictive to ensure hard real-time capabilities. For example the possible communication participants have to be known at configuration time and plug-and-play capability as well as data rate during the synchronous communication is limited. Thus, there is a tradeoff between flexibility, low configuration and integration effort on the one hand side and the capability to ensure hard real-time on the other hand side.

Due to the requirements and capabilities of the medical device and to the use cases the manufacturers and operators have to decide whether a medical device will be connected only to the SDC network without hard real-time capabilities or to both networks. Implementing both interfaces

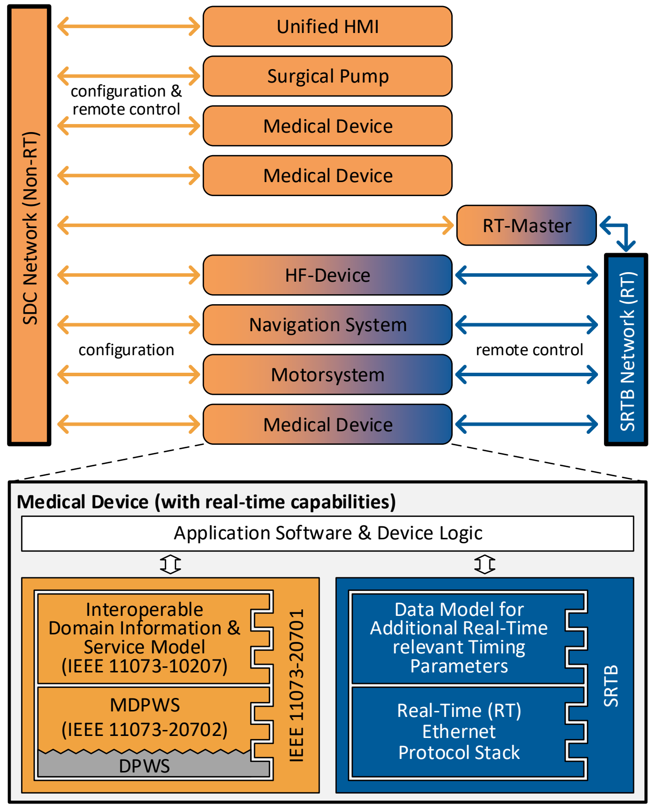

Fig. 1: OR.NET RT architecture: Parallel usage of SDC network (for non-real-time communication) and SRTB network (for real-time communication).

comprises a higher effort for the manufacturer but enables hard real-time communication as well as taking part in the SDC-based network. It is not recommended to connected medical devices only to the SRTB as the flexibility and vertical integration capability gets lost for such a device. Thus there are two types of medical devices:

1) SDC medical devices
2) SDC and SRTB medical devices

Figure 1 illustrates both networks and both kinds of medical devices schematically and show some example devices. The bottom part shows a medical device of the second type containing the SDC stack (left) realizing the interconnection to the SDC network and the SRTB real-time stack (right). The SDC stack contains the implementation of the three standards of the so called IEEE 11073 SDC family: IEEE 11073-10207, -20701, and -20702. The SRTB stack implements the RT communication stack and holds relevant parameters for the real-time communication. These are a minimum and maximum cycle time for each input of a device, the maximum age of data arriving at an input, and the maximum processing time of a device (for details see Dietz et al. [22]).

Note that there is a special network participant connected to the SDC network as well as to the SRTB network: the *real-time master (RT-Master)*. (See middle part of Figure 1.) The real-time network realizations used within OR.NET have a master component that coordinates the network traffic. Thus, connecting the RT-Master to the SDC interconnection enables the configuration of SRTB parameters. It is also possible to realize a configuration of medical devices that cannot be connected to the SDC network (e.g. legacy devices) via the *RT-Master* if necessary. If the used real-time network does not need a master component, it can be removed.

Additionally there is also a specialized pure SDC network participant labeled as *Unified HMI*. (See top middle part of Figure 1.) It represents a Human-Machine-Interface (HMI) that consolidates the information and remote control capabilities of the available medical devices. Such an HMI can be used for the configuration of the SRTB network and its participants. Especially in dynamically reconfigurable systems, like dynamically assignable footswitches, it is essential that the current configuration is displayed and clear to all actors within the OR. Aside from such a *Unified HMI* other components can additionally realize such functionality.

In the next parts of this section we will discuss both networks in detail and explain their interconnection.

A. SRTB Real-Time Network

The *Surgical Real-Time Bus* is designed for highly critical device-to-device communication with low latency requirements, corresponding to the classes C) and D) of Section I-B. Thus, all surgical robot systems and surgical device systems realizing a control loop (see Section I-A for examples) must be connected via the SRTB. Performance of the SRTB depends strongly on the type of implementation: An open-source RT-Stack using standard ethernet components may be limited to transport of class C) data, whereas class D) data typically require dedicated hardware solutions.

Devices connected to the SRTB need a compatible real-time interface implemented on a so called medical device connector. This medical device connector can either be implemented in software only, it can be a piece of hardware integrated into the device or it can be a separate hardware module. Through a separate module wrapping existing interfaces it becomes possible to integrate legacy devices without changing their design. This is an important feature as medical devices can have very long life cycles.

B. SDC Non-Real-Time Network

There are several main areas of application for the SDC network that do not need hard real-time capabilities, like the exchange of vital signs, device parameters, as well as patient demographics and order information. If this information is displayed for humans or is provided for documentary issues hard real-time capability is not necessary. Also the exchange of information from the clinical information systems is not real-time critical. Typically the information systems themselves have high latency and do not react deterministically in terms of response times. The second area is changing device parameters. Typically this does not require hard real-time, especially if it is done by humans. Also the activation of non-critical device functionality or device functionality with a high physical inertia can be done using the SDC network, like activating the flow of a surgical pump.

For the activation of critical device functionalities over a SDC network Kasparick et al. [23] provide safety mechanisms that can be used for medical device having a certain safe activation state. For example, the safe activation state of a surgical motor system is off as the patient cannot be harmed. The described mechanism ensures that network failures like delayed packages or connection-loss are safely recognized and the device passes over to its safe activation state. For medical devices having no certain safe activation state this mechanism is not suitable and a real-time network has to be used. For example the safe activation state for high frequency device cannot be defined clearly. The cutting function shall stop, but the coagulation function shall be available at every time as it is necessary to staunch bleedings.

Summarized the SDC-based interconnection can be used for communication with latency requirements of Class A) and B) (see Section I-B) having low criticality or even high criticality with a certain safe activation state.

C. Interaction between SDC and SRTB: OR.NET RT

Although there is the separation between SDC and SRTB network, there will be interaction between both. We refer to the developed concept of interconnecting SDC and SRTB as OR.NET RT. One important aspect is using the SDC communication to make the SRTB more flexible. Much configuration effort can be done via SDC before the data transfer over SRTB starts. For example parameters of medical devices that are activated via SRTB can be changed via SDC. Also the association between control elements, like a footswitch, and different medical devices or the configuration of a navigated control system as described in the guiding examples (see Section I-A) can be done using the non-real-time communication.

Such influences of SRTB connected devices via SDC have to follow one basic rule: SDC does not disturb SRTB under any circumstances. This means that commands received over SRTB have to be handled with a higher priority by the medical device than commands received from the SDC network. Furthermore, it is not possible to change parameters that affect functionality of the SRTB connected system over the period of time an action is performed. Thus, a blocking period for SDC commands has to be ensured. For example while a navigated control is working or while a dynamic assignable footswitch triggers the activation of a medical device, no configuration changes are allowed via SDC. The length of the blocking period is application specific and depends on the risk management of the device manufacturer.

Technically this will be realized using features of the IEEE 11073-10207 Domain-Information- and Service-Model: During the blocking period the corresponding remote control operations will be deactivated using the *OperatingMode* attribute of the operation state. This informs every interested service consumer that the remote control operation is (temporarily) not available. Additionally the device has to ensure that incoming remote control commands are rejected and the requester will be informed about the rejection. This is especially important for the time when the blocking period starts but this information might not be propagated fast enough in the SDC network.

Note that the reading access from the SDC network should not be affected, under the consideration of processing priority in case of resource constrained systems. This ensures that for example documentary issues can be ensured.

V. Realization: SDC meets SRTB

A SRTB reference implementation at the research institute of Micro Technology and Medical Device Technology (MiMed) of the Technical University of Munich was demonstrated to the public in March 2016. This demonstrator is based on the RT-Ethernet protocol POWERLINK and will be introduced in the following. It was the first implementation offering both hard real-time capabilities for control and sensor data transmission as well as an interface to SDC for configuration purposes.

A. Ethernet Powerlink

Ethernet POWERLINK (EPL) is a real-time protocol targeted at industrial automation. Network participants are called EPL controlled nodes or EPL slaves. There is also a central node called EPL master which orchestrates the real-time communication.

EPL uses a TDMA medium access pattern where time is divided into fixed-size intervals called cycles. Within each cycle each EPL controlled node has a fixed slot in which it sends and receives data. It must not send or receive data at any other point in time which prevents packet collisions that might lead to unbounded delays due to retransmissions. This predefined schedule is repeated cycle after cycle. Schedule execution is orchestrated by the EPL master by polling slave nodes at the respective time slots. The deterministic behavior of EPL makes it easy to guarantee worst case latencies by either manual reasoning or automated static timing analysis that can be conducted before the network is actually operated.

B. MiMed SRTB Master

A concrete implementation of the SRTB master is shown in Figure 2. The SRTB master not only manages the SRTB real-time network but also acts as the bridge between SDC and SRTB. This dual-network-structure is also reflected in the internal architecture of the real-time master as shown in Figure 3. It consists of three separate processing platforms: A real-time unit for managing the SRTB network, a display/control unit to implements direct human machine interaction via a touch display and an SDC unit that allows communication with an SDC network.

Fig. 2: MiMed real-rime Master.

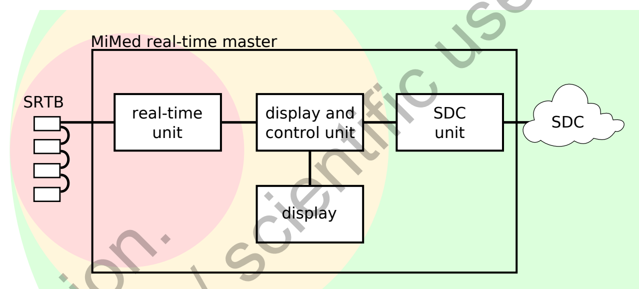

Fig. 3: Internal architecture of the real-time master with different safety levels ranging from high (SRTB, red) to medium (SDC, green).

The real-time unit is the only part of the real-time master that is connected to the SRTB. It acts as EPL master and controls the communication within the real-time network. Besides performing the necessary low-level EPL functionality, such as sending *Start-of-Cycle* packets and polling controlled nodes via *Poll-Request* packets at correct time intervals, the master also implements the overlying data flow by ensuring that data, exchanged between controlled nodes, is correctly forwarded.

The SDC unit connects to the SDC network and allows modification of certain parameters of the SRTB network or devices connected to the SRTB network. Only parameters which do not require

Fig. 4: OR.NET RT demonstrator capable of joint operation between SDC and SRTB at MiMed, Technical University of Munich.

hard real-time requirements are allowed to be modified via this interface. One of such non-critical parameters is the currently controlled medical device (see Section V-C.1) with the restriction that this value cannot be changed while the footswitch is pressed. The SDC unit acts as an SDC device (service provider) that advertises parameters via SDC settable and gettable metrics just as any other SDC service provider. SDC clients (service consumer) can connect, retrieve the set of available metrics, the current value of individual metrics and change the desired settable metrics via the SDC protocol.

Internally, valid set-requests arriving at the SDC unit are gathered, cleared from duplicates and then forwarded to the display/control unit at a fixed time-base to avoid overloading the display/control unit in cases of high SDC traffic load. Between the display/control unit and the real-time unit, a simple cyclic master slave communication protocol is used to change parameters and to report the current network state back to the display/control unit. Throughout the whole communication chain, only full duplex point-to-point connections are used to prevent potential packet loss due to collisions.

C. Application Scenarios

The SRTB demonstrator (see Figure 4) at the MiMed institute of Technical University of Munich implemented several application scenarios, including the guiding examples introduced in Section I-A.

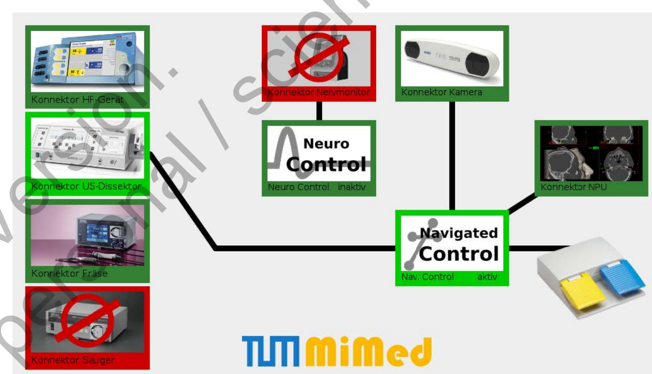

Fig. 5: Master Display that allows the dynamic selection of devices to be controlled via the universal footswitch.

1) Configurable Footswitch: A configurable footswitch, connected via the dynamic real-time network, allows the activation of various devices via one single footswitch. The device to be activated can be selected either via the MiMed-Master's touch screen or remotely via a generic SDC client. Device control is instead realized via the real-time network. The initial implementation comprised an aspirator, an HF-unit, a surgical drill and an ultra-sonic dissector (see Figure 5). It is the first implementation that bridges the gap between hard real-time control requirements and SDC flexibility.

For safety reasons, the initial device to be activated must be selected by the user. Upon selection, a green line connecting the footswitch and the selected device appears on the Master display. While the footswitch is pressed, the line color alternates between grey and green and all other devices are greyed out to avoid change of the network configuration during device operation. Yet, if one tried to select another device during operation, a hint appears on the Master display and change of configuration is not implemented. Even in the event of network failure (including the Master), all medical device connectors have a dedicated fallback-socket that is compatible with the footswitch plug. Thus, all devices can be controlled independently from network failures.

2) Configuration and Control: Aside from their control interfaces, most medical devices with real-time requirements offer the configuration of various parameters via either a user interface or a separate communication interface. Mostly there are no strict timing constraints on the setting of these parameters. Setting these parameters via the real-time network is of little use and consumes precious bandwidth.

As we operate the SDC parallel to the SRTB, we profit from both the flexibility and bandwidth of SDC as well as the hard real-time capabilities of the SRTB. In our implementation, we used the SRTB for control and sensor data transmission of various devices comprising instruments such as drills, HF-units, surgical pumps as well as navigation cameras and neuromonitoring signals. On the other hand, SDC allows us to remotely manipulate the device's configuration parameters such as maximum drill revolution, suction intensity, coagulation mode (monopolar, bipolar) and so forth.

3) Distributed Navigated Control: As explained in Section I-A, Navigated Control describes the method to control the power of active medical instruments. For the implementation of Navigated Control a stereo tracking camera and a navigation computer are coupled to the SRTB. A dedicated function module gathers the tracking data from the stereo camera and constantly monitors the distance between instruments and predefined risk structures. If activated by the user, the module is logically coupled between the networked footswitch and the controlled instrument. In case the distance to the risk structure falls below the threshold the module limits the maximum output power of the instrument. The surgeon always stays in control as the module only limits the footswitch signal and never actively sets the output power.

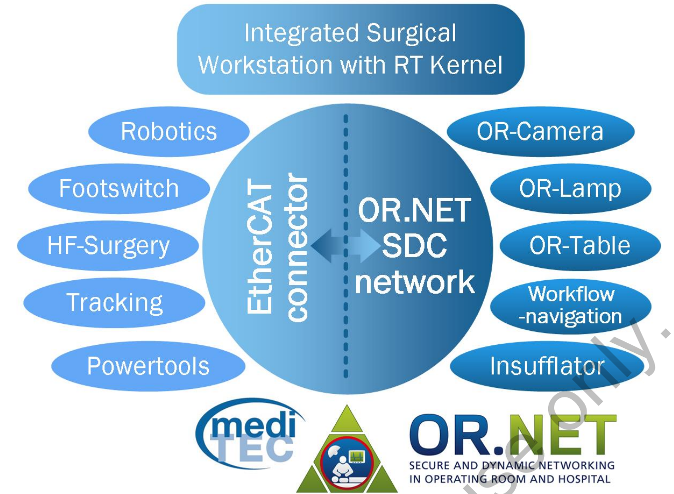

Fig. 6: EtherCAT connector on an integrated surgical workstation to connect real-time medical devices with the OR.NET SDC network.

This module is the first distributed Navigated Control application capable of controlling more than one device simultaneously. It was implemented so that it can be coupled with a surgical drill and an ultra-sonic dissector. Further devices can easily be added in the future.

VI. SDC MEETS SURGICAL ROBOTIC SYSTEMS

For the connection of robotic systems to the SDC network, a connector based on the presented architecture has been developed with EtherCAT at the Chair of Medical Engineering at the RWTH Aachen University. This connector is running on an integrated surgical workstation (Figure 6) to connect real-time with non-real-time SDC networks in operating rooms.

To generate a dynamic trajectory for the robotic system, different sensors, such as, inertial sensors, high-speed optical tracking systems and haptic force feedback devices have been used [24] [25]. High requirements regarding spatial and temporal resolution have to be met to calculate and achieve a precise trajectory for a stable closed-loop control behavior with low network latency. Commercially available motion controllers are usually equipped with CANopen or EtherCAT ports. In this setup we chose EtherCAT.

A. EtherCAT

EtherCAT is standardized according to IEC 61158, IEC 61784, IEC 61800 and ISO 15745. The Ethernet communication operates on ISO layer two with 100Mbit/s in full-duplex mode and is based on a master/slave architecture [26].

B. Application Scenarios

There are different potential application scenarios for semi-active or active robotic systems, among these are (revision) total hip arthroplasty, uni- and total knee arthroplasty, pedicle screw placement, kyphoplasty, vertebroplasty, biopsies or endoscope (tele-)manipulation. In this discussion the focus is on the following two scenarios, including the guiding examples introduced in section I-A:

1) Cement Removal in Revision Total Hip Replacement,
2) Universally-configurable footswitch for the use with different medical device apparatuses from different manufacturers.

Due to the poor visibility of the operating area, the removal of bone cement based on iterative multiplanar x-ray images may carry a high risk of complications, for example, where perforations or fractures require traumatic invasive intervention. Furthermore, the robotic system may need to be reconfigured frequently during the surgical workflow thereby increasing the number of interactions with various footswitches. This could lead to a reduced overview of the available activation options of networked medical devices over various footswitches and increases the risk for errors in the OR.

At the Chair of Medical Engineering at the RWTH Aachen University a modular and versatile mini-robot for minimal invasive orthopedic surgery (MINARO) [25] has been developed (Figure 7) in previous founded research project, called OrthoMIT. The lightweight robot system offers five degrees of freedom for minimal invasive total hip revision surgery, reduces surgical trauma and hospitalization and supports a faster rehabilitation.

The integration of a robotic system into a networked OR over a connector offers the possibility to generate real-time status-reports and enables the medical staff to configure parameters from a central surgical workstation. Furthermore, the milling path

Fig. 7: Modular mini-robot for minimal invasive orthopedic surgery (MINARO).

and the bone cement thickness can be displayed, based on preoperative images and real-time data from a tracking server [5][6].

For the second scenario a universal footswitch has been developed to reduce the number of footswitches in the operating room and has been integrated into the connector via EtherCAT protocol. The GUI panel of the universal footswitch is presented in Figure 8 and shows the working mode with the configured pedals. The previews of available configuration functionalities are shown on the right and on the bottom of the figure.

By pressing the black button for single selection or the black preset pedal, the working mode switches into the configuration mode. In this mode, the available functionalities of the networked medical devices in the SDC environment will be highlighted only for the yellow pedal and the grey rocker switch [27]. For safety reasons during the surgical procedure the blue pedal is always reserved for bipolar coagulation.

A programmable logic controller (PLC) on the real-time connector differentiates between working and configuration mode. The user can only enter the configuration mode when both, the yellow pedal and the grey rocker, are not pressed. During the configuration mode the user can select between different functionalities for the yellow pedal and the grey rocker switch by pressing those pedals. After choosing a different setting the user confirms the configuration by pressing the black pedal again [27]. In case the user aborts the configuration mode by pressing the other black pedal, the previous configuration is restored.

The footswitch information, like ID, battery status, signal strength, configuration process, the selected setup on the pedals and the activation values of the pedals are provided to the surgical

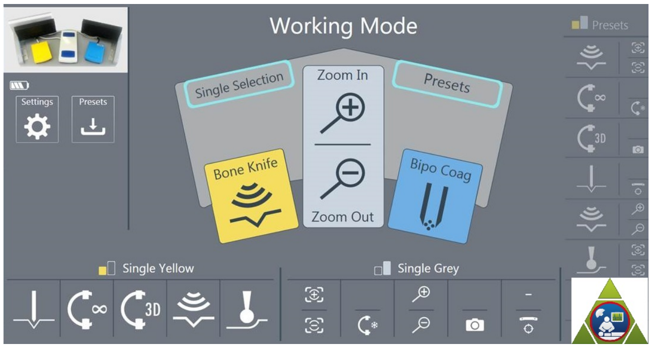

Fig. 8: Graphical panel of the universal footswitch.

workstation through the SDC connector. With this approach, the user can observe the footswitch status on a graphical user interface of the surgical workstation, which is connected to the SDC network.

In general the selected approach can be implemented with other real-time industrial Ethernet based field busses. This shows the flexibility of the real-time architecture presented in the previous sections. The connector provides an interface for sharing data to a central surgical workstation, which has also been developed as part of the OR.NET project. The development of several connectors for further integration of industrial Ethernet protocols and SDC framework increases the opportunities to choose different devices and therefore appropriate treatment procedures.

VII. Conclusion and Future Work

In this article we introduced the OR.NET RT approach to bridge the gap between flexibility requirements that can be achieved by using SDC as well as hard real-time requirements that fall in the domain of SRTB. Our current solution extends medical devices with connectors that allow a simple integration to SRTB. The SRTB Master component offers an interface to SDC that allows for dynamic reconfiguration of the real-time network. All data without hard timing constraints can be transferred via the service-oriented SDC network. The SRTB can be implemented with different industrial Ethernet protocols. This offers a higher flexibility to manufacturers and enables a long term persistence of the proposed architecture. As a next step the SRTB real-time part of the proposed architecture will have to follow the process of standardization that already started rolling for the SDC communication as new part of the IEEE 11073 family.

While there are many research contributions to bring SOA on the device-level with real-time requirements, the available solutions cannot use one comprehensive SOA interconnection. If there are hard real-time requirements a separated real-time network is necessary. The final goal is to use one single IP network with real-time capabilities that can operate within <1 ms. Such a comprehensive real-time SOMDA interconnection would provide capabilities like plug-and-play, loose coupling, scalability, reusability, etc. for all kinds of medical devices and applications. This will improve clinical workflows, lower costs in terms of installation and integration effort as well as in terms of network infrastructure, and reduce the number of cables in the OR. To achieve this goal a couple of problems have to be solved. To give some examples: The communication has to become deterministic and the latency has to be reduced. It has to be guaranteed that non-real-time traffic cannot affect the real-time communication and devices that do not fulfill the specification or being malicious cannot harm the real-time system. New ways of human-machine-interaction will be developed and regulatory issues have to be adapted to enable safety critical dynamic interconnection.

Acknowledgment

This research has been funded within the project "OR.NET - Secure and Dynamic Networking in Operating Room and Hospital" by the German Federal Ministry of Education and Research (BMBF) and in parts within the project "ZiMT - Certifiable Integrated Medical Technology and IT Systems based on Open Standards in the Operating Room and Hospital" by the State of North-Rhine Westphalia and the European Commission - European Regional Development Fund (EFRE, grant no.: EFRE-0800454).

Author Contributions

Sections contributed by the Institute of Micro Technology and Medical Device Technology, Technical University of Munich are I, II, V and VII. Sections contributed by the Institute of Applied Microelectronics and Computer Engineering, University of Rostock are III, IV and VII. Sections contributed by the Chair of Medical Engineering, RWTH Aachen University are VI and VII.

References

[1] H. U. Lemke and M. W. Vannier, "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.

[2] T. C. Lueth, J. Bier, A. Bier, and A. Hein, "Combination of medical devices for removal and manipulation of material, computer assisted and connected to positioning system," *DE10117403, LB Medical GmbH, Berlin*, 2002.

[3] M. Kneissler, A. Hein, M. Matzig, U. Thomale, T. C. Lueth, and C. Woiciechowsky, "Concept and clinical evaluation of navigated control in spine surgery," in *Proceedings of International Conference on Advanced Intelligent Mechatronics*, vol. 2. IEEE, 2003, pp. 1084–1089.

[4] M. Hofer, T. Lueth, A. Dietz, and G. Strauss, "Potential of the navigated controlled surgery at the lateral skull base with the navigated control unit (NCU 2.0)," in *Studies in Health Technology and Informatics*, vol. 173, 2012, pp. 183–185.

[5] S. Heger, M. Niggemeyer, M. de la Fuente, T. Mumme, and K. Radermacher, "Trackerless ultrasound-integrated bone cement detection using a modular minirobot in revision total hip replacement." *Proc Inst Mech Eng H*, vol. 224, no. 5, pp. 681–690, 2010.

[6] J. Benzko, B. Ibach, M. Niggemeyer, and K. Radermacher, "A novel SOA based approach towards the integration of a robotic system into a modular surgical work system and IT network," in *Proceedings CARS (Suppl. 1)*, ser. International Journal of Computer Assisted Radiology and Surgery, vol. 5, CARS. Geneva (Switzerland): Springer, June 2010, pp. 193–194.

[7] A. King, D. Arney, I. Lee, O. Sokolsky, J. Hatcliff, and S. Procter, "Prototyping Closed Loop Physiologic Control with the Medical device Coordination Framework," *Proceedings of the 2010 ICSE Workshop on Software Engineering in Health Care, SEHC '10*, pp. 1–11, 2010.

[8] J. Hatcliff, A. L. King, I. Lee, A. MacDonald, A. Fernando, M. Robkin, E. Vasserman, S. Weininger, and J. M. Goldman, "Rationale and Architecture Principles for Medical Application Platforms," in *Proceedings - 2012 IEEE/ACM 3rd International Conference on Cyber-Physical Systems, ICCPS 2012*. IEEE, apr 2012, pp. 3–12.

[9] G. P. Fettweis, "The Tactile Internet: Applications and Challenges," *IEEE Vehicular Technology Magazine*, vol. 9, no. 1, pp. 64–70, mar 2014. [Online]. Available: http://ieeexplore.ieee.org/document/6755599/

[10] C. Woskowski, "A Pragmatic Approach towards Safe and Secure Medical Device Integration," in *Computer Safety, Reliability, and Security: 33rd International Conference, SAFECOMP 2014, Florence, Italy, September 10-12, 2014. Proceedings*, A. Bondavalli and F. Di Giandomenico, Eds., 2014, pp. 342–353.

[11] M. Kasparick, S. Schlichting, F. Golatowski, and D. Timmermann, "Medical DPWS: New IEEE 11073 Standard for safe and interoperable Medical Device Communication," in *2015 IEEE Conference on Standards for Communications and Networking (CSCN) (CSCN'15)*, Tokyo, Japan, 2015, pp. 212–217.

[12] ——, "New IEEE 11073 Standards for interoperable, networked Point-of-Care Medical Devices," in *Engineering in Medicine and Biology Society (EMBC), 2015 37th Annual International Conference of the IEEE*, Milan, Italy, 2015, pp. 1721–1724.

[13] M. Kasparick, M. Schmitz, B. Andersen, M. Rockstroh, S. Franke, S. Schlichting, F. Golatowski, and D. Timmermann, "OR.NET: A Service-Oriented Architecture for Safe and Dynamic Medical Device Interoperability," *Biomedical Engineering / Biomedizinische Technik*, 2017. [Online]. Available: https://doi.org/10.1515/bmt-2017-0020

[14] S. Schlichting and S. Pöhlsen, "An architecture for distributed systems of medical devices in high acuity environments - A Proposal for Standards Adoption," in *HL7, 2014. 11073/HL7 Standards Week*, San Antonio, Texas, USA, 2014.

[15] "sourceforge: OpenSDC facilitates development of distributed systems of medical devices." [Online]. Available: https://sourceforge.net/projects/opensdc/

[16] SurgiTAIX AG, "Open Surgical Communication Library (OSCLib)." [Online]. Available: https://github.com/surgitaix/osclib

[17] ——, "Software for the Integrated Clinical Environment (SoftICE)." [Online]. Available: https://bitbucket.org/surgitaix/softice

[18] J. H. Pfeiffer, M. E. Dingler, C. Dietz, and T. C. Lueth, "Requirements and Architecture Design for Open Real-Time Communication in the Operating Room," in *IEEE International Conference on Robotics and Biomimetics (ROBIO)*, Zhuhai, China, 2015, pp. 458–463.

[19] F. Jammes, "Real time device level Service-Oriented Architectures," in *2011 IEEE International Symposium on Industrial Electronics*. Gdansk: IEEE, June 2011, pp. 1722–1726.

[20] T. Cucinotta, A. Mancina, G. Anastasi, G. Lipari, L. Mangeruca, R. Checcozzo, and F. Rusina, "A Real-Time Service-Oriented Architecture for Industrial Automation," *IEEE Transactions on Industrial Informatics*, vol. 5, no. 3, pp. 267–277, Aug. 2009.

[21] L. Durkop, H. Trsek, J. Otto, and J. Jasperneite, "A field level architecture for reconfigurable real-time automation systems," in *Factory Communication Systems (WFCS), 2014 10th IEEE Workshop on*. IEEE, May 2014, pp. 1–10.

[22] C. Dietz, T. Lueddemann, M. E. Dingler, and T. C. Lueth, "Automated risk detection for medical device networks with hard real time requirements," in *IEEE/SICE International Symposium on System Integration*, Sapporo, Japan, 2016.

[23] M. Kasparick, M. Rockstroh, S. Schlichting, F. Golatowski, and D. Timmermann, "Mechanism for Safe Remote Activation of Networked Surgical and PoC Devices using Dynamic Assignable Controls," in *Engineering in Medicine and Biology Society (EMBC), 2016 38th Annual International Conference of the IEEE*, 2016, pp. 2390–2394.

[24] V. Cunha-Cruz, A. Follmann, A. Popovic, P. Bast, T. Wu, S. Heger, M. Engelhardt, K. Schmieder, and K. Radermacher, "Robot and computer assisted craniotomy (cranio): From active systems to synergistic man-machine interaction," *Proc. ImechE, Part H: J.Engineering in Medicine*, vol. 224, no. 3, pp. 441–452, 2010.

[25] M. Niggemeyer, M. Müller, A. Niesche, M. de la Fuente, A. Komadinic, and K. Radermacher, "Modular design of a miniaturized surgical robot system," *Biomed Tech (Berl)*, vol. 57, no. 4, pp. 261–268, July 2012.

[26] M. Rostan, J. E. Stubbs, and D. Dzilno, "EtherCAT enabled advanced control architecture," in *2010 IEEE/SEMI Advanced Semiconductor Manufacturing Conference (ASMC)*. Institute of Electrical and Electronics Engineers (IEEE), jul 2010.

[27] J. Dell'Anna, A. Janß, H. Clusmann, and K. Radermacher, "A Configurable Footswitch Unit for the Open Networked

Neurosurgical OR – Development, Evaluation and Future Perspectives," *i-com*, vol. 15, no. 3, pp. 227–247, 2016.