Dynamic Remote Control through Service Orchestration of Point-of-Care and Surgical Devices based on IEEE 11073 SDC*

Martin Kasparick[1], Malte Schmitz[2], Frank Golatowski[1], and Dirk Timmermann[1]

***Abstract*— Nowadays, the staff of modern operation rooms (ORs) and intensive care units (ICUs) has to handle increasingly complex medical devices and their user interfaces. Inconsistent and often non-sterile user interfaces lead to error-prone and slow reconfiguring actions which in the end may even harm the patient. To overcome these issues interconnected medical devices are necessary. We introduce a new concept for flexible and easy-to-use remote controls which allow to control a range of different devices from different manufacturers. Current solutions are vendor-, and mostly even device-specific and tightly coupled. The effort for manufacturers is high and the maintainability is bad. Thus, controls that can be assigned dynamically to different medical devices are rare or mostly not available. Yet such dynamic controls are badly needed to improve clinical workflows especially in ORs and ICUs. We establish such a remote control setup using the service-oriented architecture defined in the IEEE 11073 SDC standards family. The presented concept is based on dynamic service orchestration to overcome existing problems: The control device and the controlled medical device are published as independent services in the network and an additional composed service interconnects them. We successfully implemented this concept for dynamically assignable controls in a real-world demonstrator with several medical devices from more than five different manufacturers. Performance evaluations show its practicability.**

I. INTRODUCTION

Point of care (PoC) medical device systems in the hospital become more and more complex, especially in OR and the ICU. Most devices provide their own user interface with a variety from simple switches to complex touch-based graphical user interfaces (GUI). Typically, the PoC device can only be configured from its own user interface. This leads to many problems, for example when the medical device is not reachable for the physician or the nursing staff. In this case the actor has to move to the device or ask somebody to do the configuration, which is time-wasting and fault-prone. Avoidable moving through ICU or OR should be minimized due to many tripping hazards. Especially in the OR, it is often impossible for the surgeons to change settings of the used surgical devices as it is not allowed to touch the base station of the devices outside the highly sterile area of the OR. For example, the surgeon has to ask somebody for changes of the device settings of the high frequency (HF) device, endoscopic camera, etc., as only the hand sample is sterile but the device itself is not [1]. Thus, it is necessary to overcome the device and vendor borders and enable a remote control of PoC medical devices. The aim is to give the physicians and nursing staff the possibility to interact with all medical devices using the user interface that is available and usable at any point of time and any place. Thus, it should be possible to use existing switches that are located at one device to control another device potentially from another vendor.

The described problems can be encountered with integrated systems of medical devices where the devices are interconnected to each other. But currently only proprietary monolithic solutions by some major vendors are available whereas a vendor-independent interconnection is yet missing. Thus, a comprehensive interconnection is not possible or very expensive as clinic operators are limited to products of some specific vendors. This problem is addressed by the new proposed IEEE 11073 SDC standards for networked PoC medical devices: IEEE 11073-10207, -20701, and -20702, which establish a *Service-Oriented Medical Device Architecture (SOMDA)*. These open standards allow a vendor-independent interconnection and will overcome isolated solutions.

The interconnection of controls and the medical devices to be controlled can be realized using these standards in many different ways. Therefore, concepts have to be developed how this interconnection will take place. This paper addresses two major aspects of this field:

- Modelling the association of a control to a medical device.
- Realization and configuration of the interconnection between control and controlled medical device.

In a complex system of interconnected control elements and medical devices it is very important to know which switch is connected to which operation of which device. On the one hand this is important because this information has to be available for the physician to avoid faulty operation. On the other hand it is reasonable to allow a remote configuration of the association. To realize this we present a minimum modeling principle in Section III.

Furthermore we propose a new method how to realize the dynamic interconnection between controls and medical devices (see Section IV). The basic idea is to separate the concerns of providing a control device and controlling a medical device. This means that the control device will not have any client functionality. It only provides the information whether it is active or not as a basic service. The interconnection to the controlled medical device is done by a separate composed service. This idea follows one of the basic principles of SOAs which is stated in the SOA Manifesto [2]

*This work has been partially funded by the German Federal Ministry of Education and Research (BMBF) as part of the OR.NET project.

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

[2]Author is with the Institute for Software Engineering and Programming Languages, University of Lübeck, 23562 Lübeck, Germany firstname.lastname@isp.uni-luebeck.de

as follows: “Separate the different aspects of a system that change at different rates.” Compared with the traditional realization where the control device invokes the services of the medical device directly there are several advantages: 1) The control device can be manufactured independently of the controlled medical devices. Due to the simplicity of such control devices and their interfaces the device can be reused for many different purposes. 2) Maintenance and debugging of the control devices is simplified by their simplicity as well. 3) IEEE 11073 SDC aims at keeping the device implementation as simple as possible. As the control devices do not need to implement any client functionality their implementation is suitable for resource-constrained embedded systems. Altogether, control devices are cheap to produce and new instances can be added easily to existing systems.

The use cases of the OR.NET project [3] are related to the OR, thus the examples given in this paper focus on surgical devices. As the proposed IEEE 11073 SDC standards include all PoC devices and the described mechanisms are generic, the concept is fully applicable for PoC devices in general.

In the paper we will use the following wording convention: A *control* describes any imaginable unit that allows the user to make an input in terms of pressing, clicking, sliding, etc. To give some examples: a switch that can be on and off, a foot pedal that can provide a (possibly) continuous press-depth, a button or a slider on a touch screen, etc. A *control device* is one piece of hardware that can provide one or more *controls*, for example foot switches, touch screens, control panels, etc. In terms of the SOA a *client* is a component that uses services of a *device*, like a *get* or a *set* service. The *client* does not provide any services to the other participants.

II. STATE OF THE ART

A. IEEE 11073 SDC Interoperability Standards

The focus of the three proposed IEEE 11073 SDC (System and Device Connectivity) standards is to enable a dynamic, vendor-independent, and interoperable interconnection between PoC medical devices. Currently, they are in the process of standardization. The IEEE 11073-20702 specifies the data transmission technology. It is derived from the OASIS standard Devices Profile for Web Services (DPWS). The Medical DPWS (MDPWS) defines extensions and restrictions to ensure medical safety [4].

The structural interoperability is addressed by the IEEE 11073-10207 that defines the Domain Information & Service Model. It allows the modelling of medical devices as a tree hierarchy with a depth of four. On the lowest level, *Metrics* represent the measurements, status, settings, etc. of the device. Additionally, remote control capabilities can be provided. E.g., set operations can be defined to change the value of a metric by setting a new (absolute) value. An *Activate Operation* allows a remote triggering of functionalities of arbitrary complexity, like increase or decrease of a value or turning on and off medical functionalities. If necessary, activate operations can be used with an argument. Every element of the device description tree including the remote control operations is semantically annotated by a code belonging to a coding system. Furthermore, additional information like ranges and step widths of metrics and operations can be declared. This ensures vendor-independent interoperability [5].

IEEE 11073-20701 describes the allover SOMDA architecture based on the paradigm of a service-oriented architecture (SOA) and defines the binding between the Domain Information & Service Model and Medical DPWS.

B. Connection between Controls and PoC Medical Devices

Currently, interconnection between controls and medical devices is realized in the way that the control implements client functionality. Thus, it is able to invoke the remote control operations of the medical device. This concept contradicts the paradigm of a SOA to separate device and client functionality. The need to implement the client functionality at the control devices leads to a high effort for the manufacturers and bad maintainability. Thus, currently available controls do not support a dynamic interconnection to other devices, especially to other vendors, and the barrier to entry is high for new products. Furthermore the additional client implementation needs more hardware resources. To overcome these problems we present a new concept in this paper.

III. MODELING CONTROL ASSOCIATIONS

For dynamically interconnected systems of networked medical devices it is necessary to be able to monitor and configure associations between dynamically assignable controls (e.g., foot switches, handhold buttons, etc.) and the remote controlled devices (e.g., surgical drill, pump, OR-table, etc.). Especially, this is important for safety-critical remote activation of device functionalities like the activation of a surgical drill. Thus, the current association of a dynamically assignable control has to be clear to the physician and all participants of the surgery. For example, the association can be displayed on a dedicated monitor, the information can be embedded into the surgical workflow like an overlay on the endoscope or microscope image, or the configured effect of a switch can be displayed directly at the switch device if it provides an appropriate display. To realize this, we propose to describe the association by two dedicated string metrics that will be available for every single control of the control device: 1) the *UDI* of the device the control is associated with and 2) the handle of the associated activate operation.

This is the minimum set of information that is needed to model the association unambiguously, as the *UDI (Unique Device Identification)* is unique by definition and the handle has to be unique within the scope of the medical device. Thus, for every medical device ensemble this information allows every participant of the device ensemble to be informed about the associations between controls and controlled devices. This is one key enabler to display this information where it is needed by the actors within the OR. In order to allow a remote configuration of the assignment the control can define set operations to modify the two metrics.

IV. DYNAMIC SERVICE ORCHESTRATION

Currently, control devices implement the client functionality of triggering some action at the device to be controlled. As

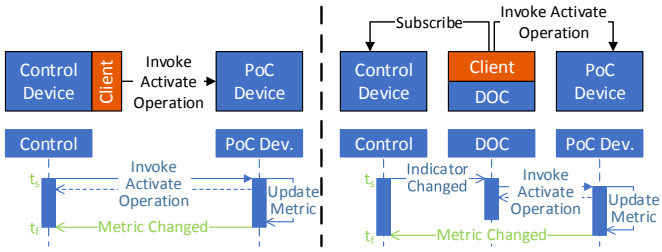

Fig. 1. Comparison between traditional tight coupling (left) and proposed loose coupling using dynamic service orchestration (right) between controls and PoC devices and (simplified) sequence diagram including an additional event for the technical evaluation of the RTT (green)

described above, this realization has strong disadvantages. Bringing the concept of service orchestration to the given scenario of the dynamic interconnection between controls and medical devices means that the control will only provide its origin functionality: Indicate whether it is pressed or not. The client functionality should be done by another component that provides the interconnection realized as a composed service. It is called *Dynamic Orchestration Component (DOC)*. The *DOC* orchestrates the services of the control devices indicating whether a control is utilized or not on the one hand and the services of the medical devices that will be controlled by control on the other hand. Fig. 1 illustrates the dynamic orchestration compared with the traditional realization.

A control device provides one *indicator metric* for every control that indicates whether the control is pressed or not. For example a foot switch with three pedals provides three *indicator metrics*. There are different types of the controls like binary, discrete, or continuous. They are modelled by a corresponding metric type (enumeration or numeric) with additional descriptive information defined in IEEE 11073-10207, e.g. the precise semantic defined by a term code and information about limits and step widths. The control device provides the ability for clients to retrieve the metric state by a get service (polling) or to subscribe to the metrics where the metric state is retrieved by events that fire by a metric change or periodically.

The medical device to be controlled offers activate operations according to its capabilities. The specific functionality of the activate operation and its semantics is described by the operation type as a coded value.

A. Dynamic Orchestration Component (DOC)

The interconnection between control and medical devices that should be controlled is realized by the *DOC*. Thus, it has to discover all controls and all medical devices that offer activate operations within the medical device ensemble. The *DOC* subscribes to the *indicator metrics* of the control devices and invokes the associated activate operation of the medical devices if an *indicator metric* change has been received. The decision whether controls or activate operations fit into the concept and whether a concrete control can be associated with a concrete activate operation will be made according to their descriptions. So, the indicator metric type of the control and the type of the activate operation have to be compatible. The IEEE 11073 SDC device description contains semantic annotations for the indicator metric and the activate operation which allow for automatic unit translations in some cases. For example, the range of an indicator metric between 0 and 255 (8 bit value) could be automatically mapped to a percentage argument of an activate operation. Whether such a mapping can be done or not depends on the concrete functionality and semantic that is defined in the metric type and unit and the activate operation type.

The configuration of the association between the controls and the medical devices will be modeled by two metrics using the mechanisms described in Section III. Thus, the *DOC* has to subscribe to changes of these metrics.

We recommend that the *DOC* provides a human-machine-interface (HMI) like a touch-based graphical user interface (GUI). This would be used by the actors within the OR to configure the associations. Nevertheless, as the controls provide the metrics that model the association the configuration is not restricted to the HMI provided by the *DOC*. E.g., the control device could provide its own HMI to configure the association. If suitable activate operations are implemented, it would be even possible to switch between association configurations by using controls that are part of the system.

B. Advantages and Challenges

As initially noted, one of the principles of IEEE 11073 SDC is to keep the device implementation as simple as possible and transfer the main workload to the clients. The introduced dynamic service orchestration enables very simple control devices without the need to implement client functionality. Such control devices can be easily implemented on resource constrained systems. There are already many control elements available at several devices, like switches at the handholds of the microscope or endoscope that are used inside the highly sterile area of the OR. The simple implementation will allow the vendors to make these existing controls available for a dynamic interconnection to other devices within the OR and will reduce the efford for new ones.

Currently, the conformance declaration for devices controlling other medical devices must be very specific about the controlled device. However, we state that dynamic orchestration may lead to a new certification process: Using this approach, interconnected controls can be treated as supplemental material like for example a classical wired foot switch is treated today. The manufacturer of the control will need to make certain assurances which may be checked by the *DOC* for compatibility with the controlled medical device and especially the criticality of the controlled action.

V. Demonstrator

The feasibility of the mechanisms described in this paper has been evaluated with real medical devices in real-world demonstrators within the OR.NET project (see left part of Fig. 2). Therefore we implemented the *DOC* that provides the association between controls and medical devices. As control devices the demonstrator includes two different foot switches from different vendors, each with three separate pedals, as well as a camera head of an endoscope with two buttons.

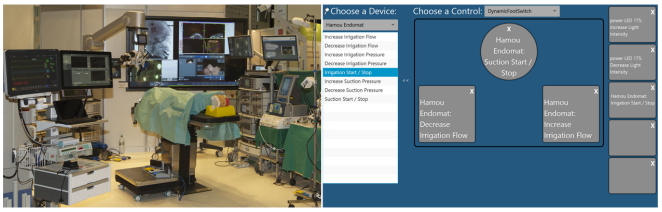

Fig. 2. Left: OR.NET demonstrator including the concept of dynamic orchestration at conhIT exhibition 2016; Right: HMI prototype for *DOC*

This demonstration includes more than ten different devices from more than five manufacturers with over 50 activate operations that can be associated to the controls. To give some examples: increase and decrease parameters like light intensity of an endoscopic light source, rotation limit of a surgical shaver, flow of surgical pumps; change parameters of a HF-device; switch through the images of a DICOM viewer. The implementation of the medical devices and the control devices was done by us and other partners within the OR.NET project using two different reference implementations of the proposed IEEE 11073 SDC standards: openSDC [6] (Java) and OSCLib [7] (C++). The *DOC* was implemented using the openSDC library. Beside the core functionality of interconnecting the controls and PoC devices dynamically, the *DOC* provides a HMI to configure the associations (see right part of Fig. 2). This HMI is intended to be used on a touchscreen. The configuration is done via drag-and-drop by pulling the activate operation of a PoC device from the left part of the HMI to the intended control that is displayed in the middle part. Currently, the visualization of the control devices is very rudimentary and reasonable concepts have to be developed in the future.

In the demonstrator the correctness of the dynamic device orchestration was monitored with the *OSCP Swiss Army Knife* [8]. This generic client discovers all devices available and shows their metrics. For the purpose of verification a modified version of the control associations described in Section III is used: We were able to describe the desired temporal behavior of a control device controlling a medical device in terms of UDIs and metric handles in a temporal logic expression. With the help of generated monitors the fulfillment of this behavior could be checked at runtime.

VI. Technical Evaluation

In addition to the real-world demonstrator we implemented a test setup to evaluate the delay of the activate operation triggering process. The described dynamic service orchestration forces a two-hop communication with an additional processing step in the *DOC*. In contrast, the traditional connection that forces the control device to implement client functionality uses a one-hop communication. Therefore, we compare the delays to evaluate the proposed mechanism.

A. Test Setup

To realize the evaluation we use a physical network of interconnected, physically separated devices. In order to automate the trigger of the controls we used test applications on the controls and devices. The test setup for the proposed mechanism using the *DOC* includes three components: The control provides the change of the indicator metric. The medical device provides an activate operation. For the *DOC* the same software implementation is used as shown in the real-world demonstrator. The test setup for the traditional mechanism has two components: The control implements the client functionality that is able to invoke the activate operation of the medical device. We use the same medical device component for both evaluations. All involved software components are implemented by using the openSDC library in Java. As switch we use the D-Link DES-1008D 100 Mbit/s Fast Ethernet Switch. We evaluate our approach on three different hardware platforms for control device and medical device. We measured the RTT with different JVMs and present the best setup for every platform. The *DOC* is hosted on the PC in every case as it can be assumed that such a component will not be implemented on resource constrained system in real-world scenarios:

- PC (Intel i7-4770; 3.4GHz; 32GB RAM; Oracle JVM 8)
- Raspberry Pi Model B (ARMv6; 700 MHz; 512 MB RAM; Oracle JVM 7)
- Intel Galileo (Intel Quark SoC X1000; x86 architecture; 400 MHz; 256 MB RAM; openJDK 8)

B. Evaluation Measurement Process

To evaluate the system delay we measure the round trip time (RTT). The sequence diagrams shown in the lower part of Fig. 1 illustrate the measurement process: The measurement starts at t_s directly before the trigger of the control is simulated. On the right the *DOC* listens to state change events of the control and invokes an activate operation on the controlled device. In the classic connection on the left the control directly invokes the activate operation of the controlled device. For the measurement in both cases the medical device then raises a state change event which is notified. The measurement stops at t_f directly after the control has received and processed this metric change. Note that this implemented client functionality in the control device is just for the purpose of measuring the RTT and not needed in real applications. Additionally, this way we measure more than the pure invocation of the activate operation, but all recordings take place on the same device. Thus, we do not have to care about high precision time synchronization.

C. Results and Discussion

Table I compares the median and the standard deviation of the measured RTTs of the two approaches for the different hardware platforms. The measured RTTs are also shown in Fig. 3 as boxplots. The shown results are based on 2.500 iterations of the measurement. Due to Java-specific concepts like adaptive optimization at runtime, just-in-time compilation, and dynamic memory allocation, we omitted the first measurements ("warm-up phase") to ensure comparability.

For the powerful PC hardware the RTT is the lowest by far with 10.77 ms. Using Raspberry Pi and Intel Galileo the system is much slower with 123.69 ms and 140.85 ms, respectively. This is with ease within the 500 ms to 1 s reaction time of humans when muscular activity is included [9].

TABLE I

MEASURED RTTS FOR REMOTE INVOCATION OF ACTIVATE OPERATION

Hardware Platform	Traditional Connection Median [ms]	Traditional Connection Std. Dev. [ms]	Dynamic Orchestration Median [ms]	Dynamic Orchestration Std. Dev. [ms]	Overhead [%]
PC	8.37	1.75	10.77	2.19	28.72
Rasp. Pi	112.16	13.27	123.69	16.96	10.28
Galileo	126.01	26.65	140.85	29.50	11.78

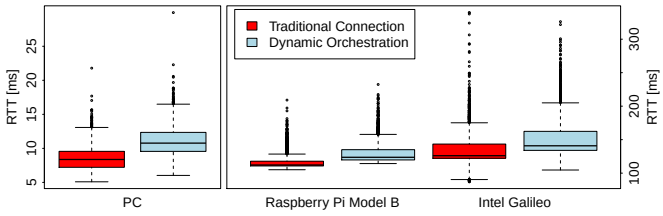

Fig. 3. Distribution of measured RTTs (note different scales)

Comparing the RTT values of the traditional connection with the dynamic orchestration, the overhead can be recognized. Table I shows the delay overhead based on the medians. For the embedded hardware platforms there is an overhead of about 10.28 % and 11.78 %, respectively. Compared with the overhead using PC hardware for the complete system, of about 28.72 %, these values are lower. As the *DOC* is running on PC hardware in all three cases, the additional processing time on the *DOC* is always the same, which leads to an increased relative overhead in the faster case of the PC hardware. As already explained it can be assumed the *DOC* will not be implemented on resource constrained systems as it is the case for the devices. Due to the very low reaction time using PCs the overhead of about 28.72 % is acceptable.

In Fig. 3 the scatter of the measured RTT values can be seen, including outliers. These values are also within the mentioned threshold of [9]. As the components are implemented in Java using the openSDC library running on non-real-time systems, the scatter can be minimized by the usage of a real-time Java Virtual Machine and a real-time operating system [10].

It can be concluded that performance of the proposed new concept is acceptable for medical real-world scenarios. The overhead compared to the traditional concept is reasonable. As latency optimization was not in the scope of this work there is much potential for improvements in future work.

VII. CONCLUSION AND OUTLOOK

In this paper we have shown how dynamic service orchestration can be used to interconnect control devices with medical devices in a flexible and loosely coupled way. Following SOA principles, particularly separation of concerns and reduction of implicit dependencies, we overcome many issues of existing solutions: Control devices no longer need to know anything about the controlled medical device. Neither needs the manufacturer to implement specific interfaces for every targeted medical device nor need the hardware be capable of connecting to another medical device at all. A network-capable control device simply needs to publish its current state into the network. Although this paper focuses on the application area of ORs the presented solution is not limited to ORs. In its generality it can be applied to any application area of PoC medical devices. The proposed concept is also fully compatible with mechanisms to increase the safety of continuous remote activation in potentially unsafe networks with problems like connection loss and jitter [11].

Flexible and dynamically assignable controls will ease the work for physicians and nursing staff as PoC medical devices can be configured at the place and at the time it is necessary. On the long run this will increase the patients' safety.

As next step we want to address the semantic description of controls: We need to establish coded values for different kinds and layouts of switch controls. These can be used to improve the sanity checks of configured mappings in the *DOC*. For example, minimum button shapes and colors for hazardous actions could be ensured.

Our current implementation uses one *DOC*. Of course, a backup system can be implemented which can take over immediately, because the configured mapping can be read back from the control devices. Nevertheless, a single *DOC* may become a bottleneck. To fix this limitation we want to use a distributed system of *DOCs*, sharing the workload.

With the real-world demonstration and the technical evaluation in this paper we have already proven the technology being ready to use today. Nevertheless, further steps towards a renewed certification process are needed before we will see such flexible and universal remote control of PoC medical devices, e.g., in ORs and ICUs. We state that on the long run our concept paves the way for an easier and more flexible certification process, as control devices and medical devices can be manufactured and certified independently.

REFERENCES

[1] J. Dell'Anna, A. Janss, M. Blaar, A. Höllig, H. Clusmann, and K. Radermacher, "Analysis of User-Induced Risks in the Neurosurgical OR," in *5th International Conference on Applied Human Factors and Ergonomics AHFE*, Kraków, Poland, July 2014, pp. 352–358.

[2] A. Arsanjani, G. Booch, T. Boubez, P. Brown, D. Chappell, J. deVadoss, T. Erl, N. Josuttis, D. Krafzig, M. Little, *et al.*, "SOA Manifesto," October 2009. [Online]. Available: http://www.soa-manifesto.org/

[3] "OR.NET - Sichere dynamische Vernetzung in Operationssaal und Klinik," 2015-03-23. [Online]. Available: www.ornet.org

[4] M. Kasparick, S. Schlichting, F. Golatowski, and D. Timmermann, "Medical DPWS: New IEEE 11073 standard for safe and interoperable medical device communication," in *IEEE CSCN*, 2015, pp. 212–217.

[5] ——, "New IEEE 11073 standards for interoperable, networked point-of-care Medical Devices," in *IEEE EMBC*, Aug 2015, pp. 1721–1724.

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

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

[8] M. Leucker, M. Schmitz, and D. à Tellinghusen, "Runtime verification for interconnected medical devices," in *ISoLA*, ser. LNCS. Springer, 2016, In press.

[9] G. P. Fettweis, "A 5G wireless communications vision," *Microwave Journal*, vol. 55, no. 12, pp. 24–36, 2012.

[10] B. Konieczek, M. Rethfeldt, F. Golatowski, and D. Timmermann, "Real-Time Communication for the Internet of Things using jCoAP," in *IEEE ISORC*, April 2015.

[11] 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 *IEEE EMBC*, Aug 2016, In press.