Intensive Care Window: A multi-modal monitoring tool for Intensive Care Research and Practice

Harald Gjermundrød, Marios Papa, Demetrios Zeinalipour-Yazti, Marios D. Dikaiakos
Department of Computer Science, University of Cyprus, 1678 Nicosia, Cyprus

George Panayi, Theodoros Kyprianou
Intensive Care Unit, Nicosia General Hospital, Nicosia, Cyprus

Abstract

Intensive Care Units are widely considered as the most technologically advanced environments within a hospital. In such environments, physicians are confronted with multiple medical devices that monitor the inpatients. The capability to collect, store, process, and share inpatient monitoring data along with the remarks of the treating physicians can bring tremendous benefits to all aspects of Intensive Care Medicine (practice, research, education). The IC-Window makes it feasible for physicians to extract, view, store, and replay Clinically Interesting Episodes through simple, intuitive user interfaces.

1 Introduction

Each year, more than four million acutely ill patients are admitted to Intensive Care Units (ICUs), only in the U.S.; approximately 500,000 of them do not survive [3]. The physiological condition of ICU patients is marked by rapidly evolving and frequently life-threatening derangements as well as 'silent' yet important alterations in homeostasis. Effective and reliable monitoring using multiple, patient attached 'sensors' is, therefore, of ultimate importance in order to ensure early diagnosis, timely and informed therapeutic decisions, effective institution of treatment and follow up. Furthermore, the capability to collect, store, process, and share inpatient monitoring data along with the remarks of the treating physicians can bring tremendous benefits to all aspects of Intensive Care Medicine (practice, research, education).

A characteristic example is traumatic brain injury (TBI), which is the main cause of death and severe disability among young people in the western world, its treatment being a challenge for the health-care team and medical technology scientists. Mortality and morbidity in TBI are largely attributed to 'secondary' injuries of the brain due to inadequate perfusion and oxygenation after the initial insult (accident). Institution of multi-modality monitoring has been suggested in order to detect and treat these 'secondary' injuries [4]. Multi-modality monitoring data may help physicians identify *bio-patterns* [8] reflecting the prognosis of a patient under a certain treatment, the effects of the treatment, etc.

However, the implementation of multi-modality monitoring in the context of modern ICUs faces a number of important challenges:

- Inpatient physiological parameters are collected and managed by proprietary medical monitoring devices. Typically, these systems have limited support for importing data from other systems or for exporting data into third-party software. Therefore, the ICU physician does not have the tools to integrate and analyse data retrieved from different devices or to deploy and use custom data-analysis algorithms.
- The multi-modal monitoring, required in cases like the traumatic brain injury described above, produces large volumes of data. However, the IT infrastructure found in most hospitals today

does not have the required resources to cope with the storage, curation, and processing of large volumes of data.

- The storage of patient-data off-site and the potential sharing thereof among different sites raise sensitive issues of secure data communication and controlled inter-institutional data sharing.

Grid infrastructures represent a promising framework for ICUs that seek to address the second and the third challenge presented above. Currently, Grid infrastructures assemble an extensive collection of resources and expertise in production-quality operational support: for instance, the *EGEE* Grid (Enabling Grids for E-sciencE) [1], which is the largest Grid infrastructure in operation, assembles over 200 sites around the world with more than 30,000 CPU's and about 5PB of storage. Thus, resources provided by Grids are adequate for storing and managing ICU-related data.

Nevertheless, to exploit the capabilities of Grid infrastructures from inside the ICU, we need to address the first of the challenges described above. Hence, we need to equip ICUs with open and extensible tools that can undertake the retrieval and integration of monitoring data from different medical sensors, and the local buffering and anonymization of integrated data-sets. Furthermore, these tools should enable medical staff to undertake the on-site study, annotation, and filtering of integrated data-sets, and the uploading thereof to the Grid by authorized physicians.

In this paper, we present the design and implementation of the *Intensive Care Window (IC-Window)*, a software tool that enables the retrieval and integration of data from patient-attached medical sensors found in modern Intensive Care Units. IC-Window follows a modular design to retrieve data from different patient monitoring devices. The tool includes a full-fledged interaction protocol and graphical user-interface to interact with the Phillips IntelliVue MP70 medical monitor, currently one of the most technologically advanced and widely used medical monitors. IC-Window is implemented in the context of *ICGrid (Intensive Care Grid)*, a novel data-grid framework that utilizes the EGEE infrastructure to enable the seamless integration, correlation and retrieval of clinically interesting episodes across Intensive Care Units [5].

The remainder of this paper is organized as follows: in Section 2, we present the context of modern Intensive Care Units and the capabilities of the Phillips IntelliVue MP70 medical monitor. Section 3 discusses the design and implementation of the IC-Window tool. We conclude in Section 4 with a summary of our experiences and a discussion of future steps.

2 Context

In this Section we provide an overview of the medical devices found in modern Intensive Care Units and describe the capabilities of the Phillips IntelliVue MP70 monitor.

2.1 Equipment in modern ICUs

Effective and reliable inpatient monitoring is of ultimate importance in Intensive Care Units in order to ensure early diagnosis, timely and informed therapeutic decisions, effective institution of treatment and follow up. In modern ICUs, inpatient monitoring relies upon information extracted primarily from multiple patient-attached medical 'sensors'. Additional information is provided by many sources including clinical examination/observation, laboratory results and medical imaging [2]. Patients' monitoring is available at the bedside and centrally, where main data from all patients are consolidated in a couple of screens. A minimum common setup of monitoring and life supporting medical devices are considered to be the equipment backbone of a modern ICU: the bedside patient monitor, the ventilator and the drug administration pumps.

The *bedside patient monitor* is the most common long term monitoring medical device used in an ICU. It constantly monitors (with variable sampling rate) the physiological parameters of an inpatient through specially designed sensors and invasive or non invasive interfaces (cables, transducers, catheters etc): Cardiac activity (heart rate, ECG and derivatives), circulation (Blood pressure & cardiac output indices, SvO2), respiratory function (respiratory rate, oxygenation, capnography), brain activity (EEG and mathematical derivatives), temperature and metabolic rate etc.

The *ventilator* is one of the most critical, life-supporting medical devices in the ICU. It provides mechanical support ventilation to inpatients when they cannot sustain the work of breathing or their lungs cannot oxygenate the blood under spontaneous breathing conditions.

Drug administration pumps are classified as life supporting devices and they are used for administering drugs to the inpatients in rhythms and quantities selected by the users. This ensures that the correct quantity of the drug is administered over the specified time period.

Departing from the equipment backbone, there are a lot of standalone short-term monitoring devices and life-supporting devices. These devices are not available in all ICUs and their use depends solely on the physicians' expertise as well as the disease type/severity. *Short-term monitoring devices* are used when a specific physiological parameter needs to be acquired for a short period of time and the long term monitor does not have the capability to monitor. *Life-supporting devices* may include devices used for supporting patients with renal failure, cardiac arrest (defibrillators), liver failure etc.

Most of the devices mentioned above are nowadays microprocessor based and have their own screens and data export capabilities.

2.2 The Phillips IntelliVue MP70 monitor

The Philips IntelliVue MP70 [6], which is part of Philips Healthcare Information Technology (HIT), is a bed side patient monitoring device. The device provides a color touch-screen display which depicts the measurements from the attached sensors. A variety of sensors (individual clinical measurement modules) can be attached into the Flexible Module Server in a plug-and-play way. These modules are interchangeable with other monitors provided by Philips (CMS 2002 monitors). One of these modules is the Philips VueLink module which provides an interface to more than 100 third-party specialty measurement devices like mechanical ventilators, gas analyzers, anesthesia machines, etc.

In order to extract the information that is collected (from all the sensors) by the monitor, the IntelliVue provides two interfaces for external device communication. The first interface is an Ethernet port (RJ45 connector) that supports the UDP/IP (User Datagram Protocol/Internet Protocol) protocol, and its network IP address is automatically configured using the BootP protocol. The second interface is a Medical Information Bus (MIB/RS232) port (RJ45 connector), which can use either a fixed or a variable baudrate protocol. On top of these two transport protocols, the IntelliVue provides the *IntelliVue Data Export Interface* communication protocol [7]. This is a connection-oriented, message-based request/respond protocol, based on an object-oriented model concept. All the information are stored as attributes within a set of defined object types. The following objects are defined in the protocol: Medical Device System (MDS), Alert Monitor, Numeric, and Patient Demographics. In order for a client application to access the attributes of instantiated objects, it first has to pull the MDS object. Then, the client gets the information of the instantiated object via queries that return the attribute values of these objects.

3 IntelliVue protocol and IC-Window application

The IC-Window was developed to extract measurements and patient data out of the IntelliVue monitor. A client application was developed to interact with the monitor using the IntelliVue Data Export Interface communication protocol. The extracted measurements are displayed using an intuitive GUI (Graphical User Interface) or saved as tab separated text files for later usage.

3.1 IntelliVue data export interface

As explained above the IntelliVue Data Export Interface is a connection-oriented protocol on top of either MIB/RS232 or UDP/IP, which are connectionless protocols. To achieve communication by the latter, a bootp server is required to give the monitor its IP address. The interaction between the

client and server using this protocol is depicted in Figure 1 and the steps are as follows. The first step of the protocol is to establish the connection, which is done through an *association* (handshake) message exchange. After the connection is established, the client can query the monitor (server) for data with a *Poll Request*. The server replies to the client with the requested data, which is referred to as *Poll Result*. The client can choose to close the connection by sending association release messages to the server. The messages that are sent between the client and server are marshalled/unmarshalled from/to objects from a class type hierarchy.

The different classes in this hierarchy describe the information that is conveyed between the client and server. The classes have a field that is classified as a "header" and an array, which is the body of the object. In the headers there is general information about the body, like the size of the body and what type of information is in it. This class type hierarchy is described in detail in the programming guide [7].

For robustness the protocol incorporates timeouts, i.e. if a certain interval has elapsed from the last message received the server will terminate the connection by sending an association release message to the client. Followed is a more detailed description of the various message types that are part of the protocol.

Connect Indication: The IntelliVue monitor will receive a valid IP address from a Bootp server in the LAN (Local Area Network). When it has received its IP address, it starts periodically to broadcast a Connect Indication message. This message contains device-related information, e.g., serial numbers, network addresses and allows clients on the LAN to locate the IntelliVue monitor.

Association Request: To establish a logical connection, the client sends the Association Request message to the IntelliVue monitor. The Association Request can be used to set optional features of the logical connection between the client and the IntelliVue monitor. For instance the client can request the MTU (Maximum Transport Unit) size, the minimum poll interval, or even request from the server to provide averages of the measurements for a given interval.

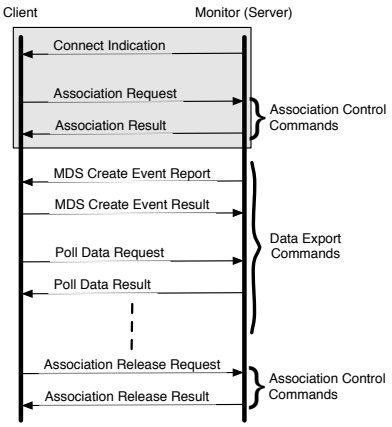

Figure 1. Data export protocol

Association Result: The IntelliVue monitor processes the Association Request and sends an Association Result. The client must parse the Association Result to find out which protocol features the server will provide for the requested connection.

MDS Create Event Report: If the IntelliVue monitor accepts the association, it sends a MDS (Medical Device System) Create Event Report right after the positive Association Result message. The MDS Create Event Report contains information about the monitor, its configuration, and all the devices that are connected to it. The client will use this information to discover the devices connected to the given monitor.

MDS Create Event Result: Here the client confirms the reception of the MDS Create Event Report

Poll Data Request: After establishing an association, the client can send Poll Data Requests to access the data within the IntelliVue monitor.

Poll Result: Depending on the status of the IntelliVue monitor and the options set, a Poll Data Request message can return:

- a single Poll Data Reply
- multiple, linked Poll Data Replies, if the size of the requested data exceeds the MTU
- a continuous number of periodic Poll Data Replies for a time period defined by the client.

Association Release Request: When the client wants to close an association, it sends a Release Request. The monitor will close the connection automatically if the client stays inactive for a specified duration.
Association Release Result: The Intellivue monitor parses the Release Request and responses with an Association Release Result indicating that the Association has been released. The same Association Release Result is also sent in the case of a timeout.

3.2 IC-Window application

The IC-Window application consists of two modules and a GUI that uses the API (Application Programming Interface) of the two modules to present and extract data. The API can be used to create other applications that need to retrieve the measurements from the IntelliVue monitor. The usage of this API involves instantiation of specific objects and a timely correct interaction sequence with the server. The IC-Window GUI presents a predefined set of dialogs that demonstrate the usage of the API to acquire the data, then store the data and present it in graphs.
PhMon Module: This module is the base module for the implementation of the protocol. It processes messages (Byte Arrays) and converts them into data objects (unmarshalling). In addition, it has the functionality of generating messages (Byte Arrays) to be sent to the monitor (marshalling). The message classes are easily created using parameterized constructors. The module has also defined all the constant numeric values that are defined for the protocol and their corresponding human readable meaning (strings). Finally, the module provides the functionality to extract the data out of the instantiated measurement objects to be stored into secondary storage.
PhMonListener Module: This module contains one main class namely the PhMonCommunicator and some helper classes that are used by the main class. The PhMonCommunicator class can send and receive byte arrays from and to the different network interfaces and to and from the PhMon Class. Its function is to listen for Discovery messages from monitors on the network (LAN interface only) using the UDP segmenting classes provided by the .net API. Once a connection is established between the client and the monitor, it is this class that creates and manages the communication timers (e.g. Association Timeout).
IC-Window GUI: It demonstrates a basic usage of the API provided by the PhMon module and the PhMonCommunicator class in order to extract data from the IntelliVue monitor. The GUI is organized as a wizard to help the user interact with various monitors. The first dialog of this wizard listens for incoming Connect Indication broadcasts and as they are received the IP addresses of sending monitors are listed. The user can then select one or more monitors that he/she wants to interact with. The second dialog (see Figure 2) then presents the different sensors that are attached to the selected monitors. The user can then select which of the sensors he/she wants to retrieve measurements from. The user has two choices, a one-time poll of the measurements for the sensors or to continuously poll the measurements for a specified interval. In addition, the user can specify if the measurements should be stored on secondary storage and if so in which directory. When the user presses the *view measurements* the third dialog is displayed (see Figure 3). This dialog presents the measurements from each of the sensors in a time-value series graph.
Use-Case Our driving use-case is as following: An Inpatient is committed to the ICU unit in a clinically interesting condition. The physician would like to save the measurements from the bed-side monitors during the treatment. The reason for this is that the physician would like to replay the incident to future apprentices as part of their training. i.e. what was the treatment undertaken for an inpatient in the specific condition and what was the outcome. Thus, prior to the treatment, the physician starts the ICW application and selects the monitor located by the inpatient's bedside. There are a number of sensors connected to the monitor, but only the ones that will be interesting for the replay scenario are selected (See Figure 2) – there may be cases where all of them are selected. The measurement capturing starts after setting its time duration. As the physician treats the inpatient the ICW application will display the measurements (See Figure 3) at the same time as they are stored to secondary storage. During the training sessions, the physician can replay the incident for the apprentices.

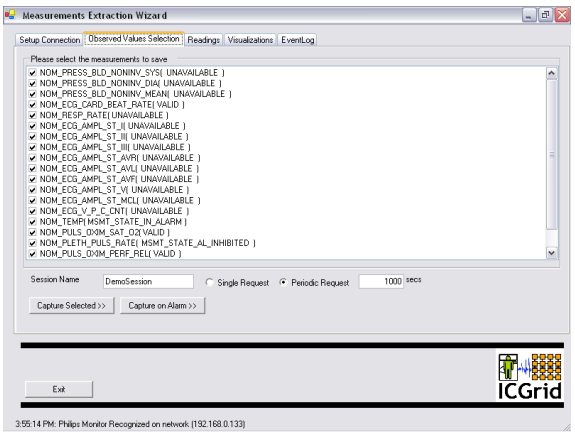

Figure 2. ICW: selecting sensors

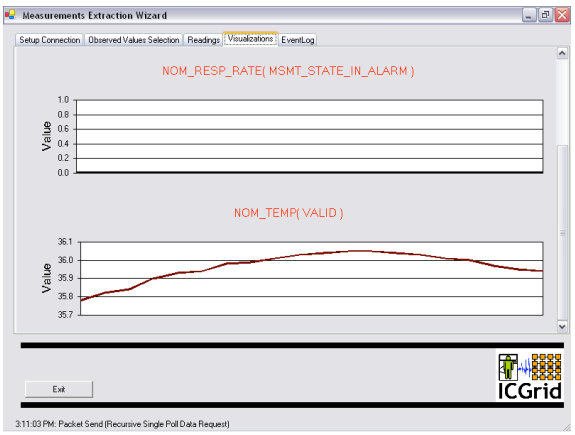

Figure 3. ICW: view measurements

4 Conclusions and future work

This paper presented the IC-Window application, which is an open implementation of the IntelliVue Data Export Interface protocol. The benefits of the application were twofold. The first was to provide an open implementation of the protocol that can be used be the research community. The second is to actually build and deploy an application that uses the protocol to assist the physicians at the ICU to gather and save the measurements for later use, such as training, medical research, post analysis or sharing the information with others.

In the future we plan to extend the IC-Window application to communicate with other medical devices found within the ICU. This will provide an open platform for others to develop applications to interact with these devices, such applications can use the retrieve data for various medical studies etc.

Acknowledgements

This work was supported in part by the European Union under projects EGEE (#IST-2003-508833) and CoreGRID (#IST-2002-004265).

References

[1] Enabling Grids for E-SciencE project. http://www.eu-egee.org/.

[2] J. Carr et al. Intensive and coronary Care Units. In *Introduction to biomedical equipment technology*, chapter 14. Prentice Hall, 4th edition, 2001.

[3] JD Birkmeyer et al. *Leapfrog safety standards: potential benefits of universal adoption.* The Leapfrog Group, 2000.

[4] RJ Multon et al. Head injury and intracranial hypertension. In Hall J et al, editor, *Principles of Critical Care*, pages 1395–1408. McGraw Hill, 3rd edition, 2005.

[5] M. Papa, D. Zeinalipour-Yazti, M.D. Dikaiakos, T. Kyprianou, and G. Panayi. ICGrid: Intensive Care Grid. In *EGEE '06 Conference (poster session)*, September 2006.

[6] Philips. Intellivue mp60 and mp70: Networked patient monitors with portal technology for critical and intermediate care. http://www.medical.philips.com.

[7] Philips. Intellivue patient monitor mp60/70/90: Data export interface programming guide. M8000-9305BM8000-9305B.

[8] L. Sun, P. Hu, C. Goh, B. Hamadicharef, E. Ifeachor, et al. Bioprofiling over Grid for eHealthcare. In V. Hernandez, I. Blaquer, and T. Solomonides, editors, *Challenges And Opportunities of Healthgrids: Proceedings of Healthgrid 2006 (Studies in Health Technology and Informatics)*. IOS Press, 2006.