COMPUTER METHODS AND PROGRAMS IN BIOMEDICINE 117 (2014) 168–178

journal homepage: www.intl.elsevierhealth.com/journals/cmpb

Event-driven, pattern-based methodology for cost-effective development of standardized personal health devices

Miguel Martínez-Espronceda[a,*], Jesús D. Trigo[a], Santiago Led[a], H. Gilberto Barrón-González[b], Javier Redondo[a], Alfonso Baquero[a], Luis Serrano[a]

[a] Electrical and Electronics Engineering Department, Public University of Navarre (UPNA), Pamplona, Spain
[b] The Coordination of Electronic and Communication Engineering, Autonomy University of Nuevo León, University Avenue s/n, University City, San Nicolás de los Garza, NL 66450, Mexico

ARTICLE INFO

Article history:
Received 13 March 2014
Received in revised form
21 July 2014
Accepted 22 July 2014

Keywords:
Cost-effective
Event-driven
Interoperability
ISO/IEEE11073
Pattern-based
Personal health device (PHD)

ABSTRACT

Experiences applying standards in personal health devices (PHDs) show an inherent trade-off between interoperability and costs (in terms of processing load and development time). Therefore, reducing hardware and software costs as well as time-to-market is crucial for standards adoption. The ISO/IEEE11073 PHD family of standards (also referred to as X73PHD) provides interoperable communication between PHDs and aggregators. Nevertheless, the responsibility of achieving inexpensive implementations of X73PHD in limited resource microcontrollers falls directly on the developer. Hence, the authors previously presented a methodology based on patterns to implement X73-compliant PHDs into devices with low-voltage low-power constraints. That version was based on multitasking, which required additional features and resources. This paper therefore presents an event-driven evolution of the patterns-based methodology for cost-effective development of standardized PHDs. The results of comparing between the two versions showed that the mean values of decrease in memory consumption and cycles of latency are 11.59% and 45.95%, respectively. In addition, several enhancements in terms of cost-effectiveness and development time can be derived from the new version of the methodology. Therefore, the new approach could help in producing cost-effective X73-compliant PHDs, which in turn could foster the adoption of standards.

Abbreviations: APDU, application protocol data unit; ASN.1, abstract syntax notation one; DIM, domain information model; ECG, electrocardiogram; FSM, finite state machine; IPC, inter-process communications; LV–LP, low-voltage low-power; MDER, medical device encoding rules; MDS, medical device system; PHD, personal health device; RAM, random access memory; RO, read-only; RW, read-write; X73, ISO/IEEE11073; X73PHD, ISO/IEEE11073 for personal health devices.

* *Corresponding author*. Tel.: +34 948 16 9671.
E-mail address: miguel.martinezdeespronceda@unavarra.es (M. Martínez-Espronceda).

http://dx.doi.org/10.1016/j.cmpb.2014.07.008

1. Introduction

Nowadays, developed and emerging economies are fostering solutions that enhance the healthcare quality, for example, by offering new services based on technological systems [1]. Population demands, however, may challenge the budgetary balance, since such services are offered depending – among other factors – on the associated cost [2]. Affordable personal health devices (PHDs) will therefore play a crucial role in new telemonitoring scenarios. At the same time, standardization of PHDs has been found as a key enabler for the provision of high quality services [3]. However, the application of standards in PHDs presents an inherent trade-off between interoperability and processing load. Therefore, developing implementation methodologies able to limit the processing load to a minimum is of utmost importance.

The ISO/IEEE11073 (X73) family of standards for PHDs (X73PHD) is aimed at providing seamless integration of PHDs into personal health solutions [4]. However, while it facilitates the exchange of medical measurements in a transparent way, the complexity of the protocol requires higher performance hardware in comparison to non-standardized ad-hoc protocols [5].

All these previous considerations led the authors to propose a patterns-based methodology as a solution to optimize the implementation of PHDs with low-voltage low-power (LV–LP) constrains [6] (hereinafter referred to as version 1.0). The patterns-based approach relied on the low variability between the fields of the X73PHD-compliant frames. This method has been applied during 3 years in the NAsisTIC project [7], as well as in other related experiences [8–10]. Version 1.0 was based on the multitasking paradigm. This implied the need of a module that provides some operating system features – threading and inter-process communications (IPC). Such features required additional resources, especially in terms of processor cycles, as well as a more complex programing and debugging process. Hence there was a need to thoroughly revise and refine the principles defining version 1.0 of the methodology. Thus, the main objective of this work is to define an improved version, hereinafter referred to as version 2.0. Particularly, the specific objectives include testing version 2.0 and comparing the results to version 1.0.

This paper therefore presents the new event-driven, patterns-based methodology and discusses the lessons learned from a 3-year experience using the previously proposed methodology. The new software architecture, along with the enhanced methodology, is presented in Section 2. Results are provided in Section 3. A comprehensive discussion of the results is conducted in Section 4. Finally, conclusions are drawn in Section 5.

2. Materials and methods

The X73PHD standard is aimed at providing an interoperable interface between PHDs and gateways (also referred to as “agents” and “managers” in X73PHD, respectively). To do that, X73PHD specifies an application layer for this agent-to-manager point-to-point communication based on the concepts of the Domain Information Model (DIM), the service model, and the Finite State Machine (FSM). On one hand, the DIM is an object oriented static model that represents data managed by agents. On the other hand, the service model and FSM define the dynamic behavior that allows managers to interoperate with agents and access their DIMs. The X73PHD family is mainly specified in the optimized exchange protocol (X73-20601) and PHD specializations (X73-104yy). The former defines a general framework and a general PHD model to virtually define any possible PHD configuration, whereas the latter detail particularizations of the general model to a specific class or type of PHD (such as blood pressures, thermometers, medication monitors, etc.).

The interactions between the agent and the manager can be observed in the transport layer as application protocol data units (APDUs). The APDUs are usually generated for each transaction of DIM-related objects and attributes. Nevertheless, considering a specific agent configuration and a specific type of APDU, many parts of the APDU are fixed in position (offset from APDU’s beginning) and content. Most of these fields are a consequence of using medical device encoding rules (MDER) encoding and meta-information included in Abstract Syntax Notation One (ASN.1) structures and they can be computed very efficiently without needing to encode/decode the whole APDU. For example, length fields can be obtained as the remaining length of the APDU. Others would be directly generated from the measurements (value and time stamp). In this context, the authors defined the APDU-pattern as a set of rules that can be used to analyze and synthesize a specific type of APDU in a specific agent configuration. The patterns-based methodology is based on this concept to limit the overhead due to APDU management.

Version 1.0 of the patterns-based methodology and the high level software architecture is shown in Fig. 1(a), lying between the application and transport layers. In this figure, the influence of the IPC and threading module is shown, which adds complexity and need for resources. In this paper, an evolution of this concept is presented. The new software architecture (version 2.0) is shown in Fig. 1(b), also lying between the application and transport layers. In this new version, the dependency with the IPC and threading module has been eliminated. The application layer adapts agent inputs to the X73PHD Kernel. The Pattern Library logically represents the collection of APDU patterns for an agent configuration as mentioned above. The X73PHD Kernel manages the X73PHD protocol state, analyzing and synthesizing APDUs in an X73PHD communication, as defined by the Pattern Library. This program is driven by the application layer and manages the transport layer. The transport layer interface provides methods to the X73PHD Kernel to initiate a connection procedure, to stop a connection, to abort it, to send and APDU and to receive APDUs. There can be several reliable or best effort channels, which are used according to the X73PHD core document (ISO/IEEE11073-20601).

The idea of APDU patterns is useful to explain the fundamentals of the patterns-based methodology. Nevertheless, in terms of software, the Patterns Library is usually embedded into the analysis and synthesis source code, making them difficult to be differentiated. There are three main algorithms, which manage the X73PHD. These correspond to the APDU