

Micro system data format MSD for interoperability between wearable devices

¹B Jettkant, ²P Dültgen

¹BG Kliniken Bergmannsheil, Ruhr Uni-Bochum

²Lehrstuhl für Produktionssysteme, Ruhr-Universität Bochum
Bochum, Germany

Keywords: Microsystem Data Format, MSD, Section, Container, Message, UTC, CRC, remote

1. Introduction

In 2001 the initiative “Implantable and Extracorporal Modular Microsystem Platform (IMEX) inside VDE was started to analyse widely used interfaces between components of modular constructed medical microsystems. As a result a unified medical format is recommended: scalability from 8 bit to 64 bit processors, amount of data from some characters up to several 100 MB data, embedding of existing data formats (like JPG, DCM...), identification of communication partners, timestamp and date in UTC, chronological order of messages, prioritization of messages, integrity of messages, differentiation between data, commands, execution and remote access, encryption and dig. Signature, path of transmission, sender name, address, router, priority, timeout values for transmission and execution (real time), event markers, handshake support for transmission, support for actuators, sensors and data transmission.

The defined MSD-format relates to data transmission between microsystems and their communication partners. These partners can be other microsystems, communication gateways or computers. The communication hardware is thereby not taken into consideration but the transmission format. Its flexibility allows the usage of established communication hardware and protocols. The format is scalable thus being able to contain a single measurement value as well as large aggregates of data e.g. tele-medicine [1]. It can include additional information indicating the origin of data, compression, encryption, transformation methods and other information.

2. The Data Format

There exists only an inhomogeneous and vendor-specific collection of several protocols which are incompatible with each other. For this reason, the original sensor data (or the original data object) are expanded by specific additional information, which are described below. The basic construction can be organized in three parts: Message, Container and Section.

The topmost structure, which contains unique information about the sender, recipient etc. is the Message. The Containers are imbedded in the Message in a specific sequence, as a sorted collection of datagrams. Containers become expedient and necessary, when the medical device acquires several vital parameters. If only one value is observed, it may be omitted. The actual user data are embedded within the Container and in a formal frame called Section which ensures the safe transport using information about the data structure, sequence processing priority and life cycle. All three parts can be organized, once again into similar modules. Message, Container and Section follow the basic construction: Header block (static and dynamic), Data object and Data integrity

The Section comprises the raw data, a supplementary description (header block) of the raw data as well as the data integrity. It can also contain expanded information about the encryption method. The raw data are covered with information about the coding method and details about the end of the data. All information are implemented using a binary TAG notation. This way, the requirement for XML conformity is met.

The Header of the Section begins with 3 information bytes. The first one contains information, as to whether it is a Section, Container or Message (Data Structure)., byte order, indication of handshaking etc. Every byte ends with one extension bit. The second byte contains among other things the priority. This is important, when several different data streams are sent simultaneously. A set LifeTime ensures that the receiving end confirms the receipt within the LifeTime, otherwise the packet can not be considered as having being received. A set TimeDivisor defines the underlying time unit as part of a second. Finally sender's name can be set. Clarity can be guaranteed using serial numbers or MAC addresses [2]. Actual details such as date stamps are attached to the dynamic part of the header protocol. The third byte contains further information whether specific details are available in the dynamic part, for instance whether data was compressed or a checksum upon the header had been attached. The dynamic header varies in the length, depending on the status bits set in the static header. It always begins with the version number and the type of encoding.

A continuous numerical 32-bit signed integer SectionCounterNumber identifies the unique sequence of the Sections. Since synchronisation of all communications partners is not feasible, the sequence of individual Sections cannot be determined, using UTC time stamp [3]. The Section Type indicates the type of the data object

using four consecutive letters. A common file extension, generally identical with Windows/ DOS file extensions permits automated processing. The next bytes are about priority and LifeTime, access right administration, all based on UNIX. Almost important is the length of the data object for validation of the complete transmission. The value refers exclusively to the decoded user data in the data section (SectionDataObject).

Subsequently, the actual data object follows. This is later stored in the database as referenced file. The checksum upon the data object is calculated and stored as an array of char. The width of the array is determined by the applied CRC method (CRC8-CRC128) [4,5].

It is possible to store a crypto-signature in the Header of Containers and Messages. The encryption can be added by means of the Cipher-Status Information. Only the file contents are encoded. Symmetric, public key and hybrid algorithms are supported.

An important supplement to the messages is the specification of the address [6]. The address of a sender, receiver, router and so forth is defined as a character strings without \0-termination. Beside all alphanumeric 7 Bit ASCII characters all 32-bit Unicode characters are allowed, however exclusively in hexadecimal representation. Other special characters are used for operators. Lists of comma-separated addresses in brackets () can also be created. It is possible to give the address in different formats, i.e. in decimal, binary or hexadecimal form, as IP format or as a substring. In doing so, the respective format must be clearly identified using a prefix. As to be shown, operators are used for routing [`<>/\`], for delimiting of addresses or lists [, | ()] or for priority definition [:]. In all these cases, the operator associativity is from right to left. The path is the description of the route. The layout of this "way description" is as follows: Adresse(s) Operator(to from) Address : Priority.

The format is not only meant for data transfer from medical equipment to the database. Depending upon the status set, several tasks can be executed. Criterion for this is how the data object was defined. As Section Data, it contains the vital parameters acquired by the medical transmitting device. Section Command is needed for confirmation of receipt of acknowledgement (LifeTime). The data object is a simple Acknowledge-flag.

Since the deployed medical devices can also act as actuator applications, Section Execute defines the executable commands. The encapsulated object then contains for example vendor-specific control data or instruction tables [7].

Remote maintenance-related problem definitions can be declared using the Remote Section. The data object in this case is for example a BIOS update [8]. By means of the Section Remote Information-Header, the protocol detects that the transmitted data is a command for remote maintenance.

The most important advantages of the protocol can be summarized as follows. Arbitrary data contents can be sent. The actual data object is enclosed and expanded by additional information. Missing information are skipped. Future manufacturer-specific protocols can be simply integrated. The MSD-format does not know anything about the content of the data object. Neither does the accepting database: it stores the object as a file. The XML-consistent tagging allows a simple and structured reading into the source code. Furthermore the contents can be better imported into the database. This way, future expansions, and in relational databases as well, are maintainable. The.msd data format, which is currently going through international standardization committees, is characterized by extreme flexibility and expansion capability. In the future, small and medium-sized enterprises (SME's) will thus gain the confidence to deploy common and established standard using the data protocol. The authors hope for a wide acceptance of the standard and its successful application not only in medicine

References

- [1] Biskup, K., Clasbrummel, B., Gerboth, A. Erste Erfahrungen mit der poststationären Betreuung traumatologischer Patienten mittels Televisite. Biomed Tech Vol. 47-1, 2002
- [2] Linux-Wegweiser zur Installation & Konfiguration, Online-Version by O'Reilly GmbH & Co.KG, 2000
- [3] David L. Mills Network Working Group, Network Time Protocol, RFC-1119, RFC-1059
- [4] Williams, R. N.: A Painless Guide to Error Detection Algorithms [online], Rocksoft (1993). Available from: ftp://ftp.rocksoft.com/papers/crc_v3.txt
- [5] Press, W. H., Flannery, B. P., Teukolsky, S. A., Vetterling, W. T. Cyclic Redundancy and Other Checksums. Ch. 20.3 in Numerical Recipes in C: The Art of Scientific Computing, 2nd ed. Cambridge, England: Cambridge University Press, pp. 896-903, 1998.
- [6] Das AX-25 Protokoll (IARU), Packet Radio Book, DARC 93, Available from: www.loisl.de,
- [7] Molitor, E. et al. Empfehlung zur Fernwartung medizinischer EDV-Systeme, Friedrich-Wilhelms-Universität Bonn, 2002
- [8] Gubbels, H. Basisdienste für Anwendungen in ubiquitären Rechnersystemen, Universität Stuttgart, Fakultät Informatik, 2002

Birger Jettkant
BG Kliniken Bergmannsheil Bochum
Bürkle-de-la Camp Platz 1
D-44789 Bochum
email : Birger.Jettkant@ruhr-uni-bochum.de