FACTA UNIVERSITATIS
Series: **Electronics and Energetics** Vol. 28, No 4, December 2015, pp. 541 - 556
DOI: 10.2298/FUEE1504541P

PRACTICAL ASPECTS OF CELLULAR M2M SYSTEMS DESIGN

Aneta Prijić[1], Ljubomir Vračar[1], Dušan Vučković[2], Danijel Danković[1] and Zoran Prijić[1]

[1]University of Niš, Faculty of Electronic Engineering,
Aleksandra Medvedeva 14, 18000 Niš, Serbia
[2]DELTA, Venlighedsvej 4, 2970 Hørsholm, Denmark

Abstract: This paper highlights some crucial design challenges of Machine–to–Machine (M2M) systems. The focus is on the cellular based, wireless wide area network systems. Design of M2M terminals, used as wireless sensor nodes, is covered in detail, including the criteria for selecting appropriate core and hardware peripherals. Discussion is extended to modeling and design of terminal's embedded software. Communication using framework and backend application software architectures are explored. Practical examples of the described design principles are demonstrated.

Keywords: Machine to machine, cellular communications, wireless sensor networks

1 Introduction

Machine–to–Machine (M2M) is a term widely used to describe communication systems where a group of dedicated devices interacts with some application, in order to perform specific task continuously, without or with minimal human interference. Rapid development of communication technologies has extended potential for application of M2M systems in many sectors such as industry, transportation, energy, retail, healthcare, environmental, buildings, etc [1]. Most of today's M2M systems rely on wireless wide area network (WWAN) by using cellular infrastructure for communication.

Received April 9, 2015
Corresponding author: Aneta Prijić
University of Niš, Faculty of Electronic Engineering, Aleksandra Medvedeva 14, 18000 Niš, Serbia
(e-mail: aneta.prijic@elfak.ni.ac.rs)

A typical wireless M2M system architecture may be segmented to three domains: M2M Device, Network and Application [2]. Simplified view is illustrated in Fig. 1. Devices are typically referred to as M2M terminals. A set

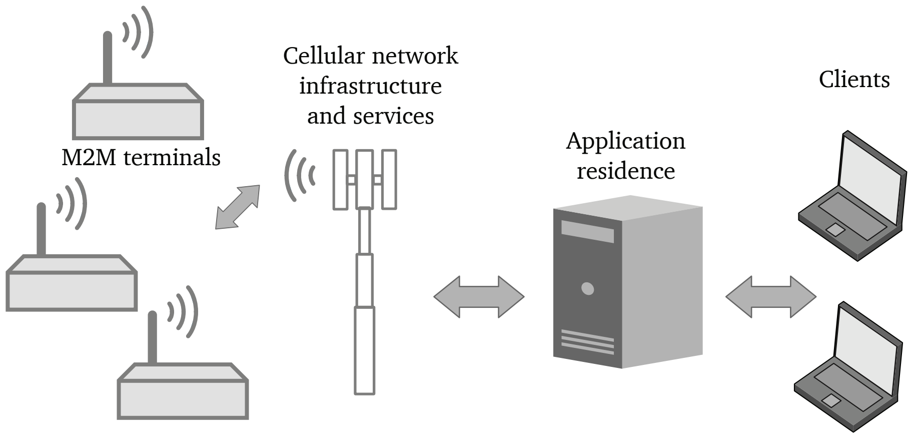


Fig. 1. Illustration of wireless M2M system architecture.

of terminals forms M2M Device domain. Terminals are communicating with the application using WWAN infrastructure and services, including M2M gateway, which is referred to as a Network domain. Communication may be: a) capillary–cellular, where several terminals are wired and connected to the WWAN through common gateway; b) cellular, where terminals are exclusively wireless and connected to the WWAN independently (Fig. 1). It is important to note that terminals are not designed as end–user or consumer devices. Rather, they interact with the environment performing pre–determined tasks and actions. As a result, sets of data are transmitted from the terminals to the application. An application is a point where information are extracted from the data, processed, analyzed, and accessed by clients. Eventually, on the basis of the received data, an application may instructs terminals to perform backward action. Application and clients, along with their interconnection, are belonging to the Application domain. Note that this architecture can be viewed as a form of wireless sensor network, a subset of more general M2M definition [3].

This paper focuses on the practical realization of cellular–based M2M systems. The design of M2M terminal is subject of Section 2. Requirements for main blocks of the terminal are presented and analyzed. Various hardware platforms are compared in terms of flexibility. In Section 3 discussion is extended to the modeling of the terminal's embedded software. Design considerations of a backand application are elaborated in Section 4. Use cases from government and industrial sectors are explored. Potentials and challenges are summarized in Section 5.

2 M2M terminal

Typical M2M terminal in cellular systems may be described using block diagram shown in Fig. 2. Blocks may be identified as power, central, and

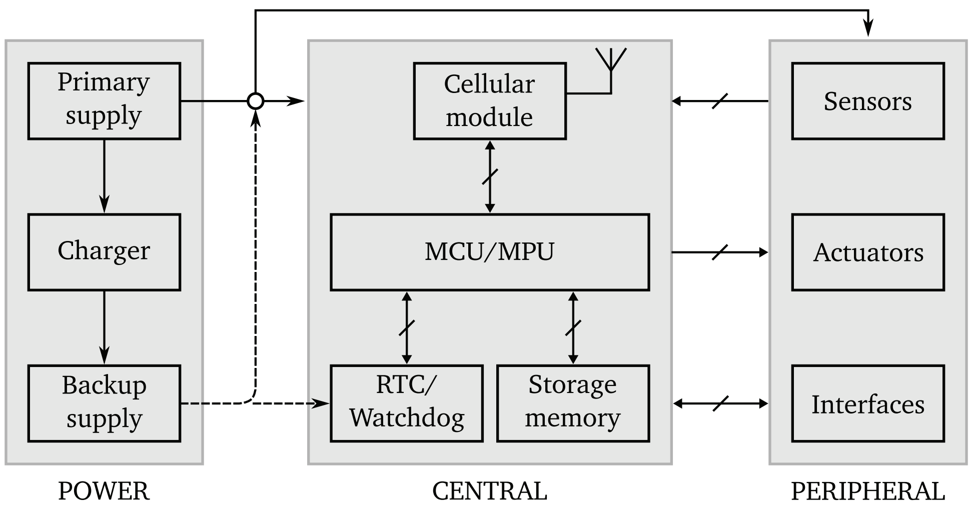


Fig. 2. Block diagram of M2M terminal.

peripheral, respectively. It should be emphasized that the design of each block in reality depends on the targeted application.

Power block consists of the primary and backup power supplies. The presence and implementation of the backup supply is dependent on the desired functionality and required autonomy of the terminal. Terminals whose functionality is related to other machines which are designed to operate exclusively on the line power (POS terminals, ATMs, etc.) normally do not need backup supply. This is also the case with the terminals built in various vehicles, where energy from the vehicle's own battery is used to power the terminal. On the other hand, terminals which act as a standalone devices often require backup supply through rechargeable battery because either their functionality demands uninterrupted operation or they are distributed within the areas where power lines are inaccessible. In the former case, energy harvesting from a renewable source is used to provide primary power. Excess energy harvested from the source is routed through the charger to the backup supply and used to power the terminal once the source is temporarily lost. Photovoltaic panels are typically used to convert solar into electrical energy, and such terminals are widely utilized in environmental monitoring and agriculture.

Terminals are designed using low–power microcontrollers/microprocessors (MCU/CPU). Power consumption of the devices in peripheral block may vary and it mainly depends on the actuators used. However, cellular modules may consume significant power in bursts during network connection and transmission, with the current consumption up to 3 A. Recommended approach is to divide an unregulated DC power input into two branches through DC/DC converters, one to provide regulated power solely to the module, and the other to power rest of the circuitry. Either switching or linear low dropout regulators may be used, with latter being preferred in order to avoid potential EMI issues. Modules' manufacturers usually give their recommendations, while general guidelines are also available [4]. Low ESR capacitors on the regulated output should be used. Modules are not tolerant even to very short dropouts of the supply voltage and in such a condition they are switched off. Tantalum capacitors of decent capacity are choice for the reservoir and they should be located to the module's power pins as close as possible.

Regardless of the necessity for the uninterrupted operation, a backup supply for the Real–Time Clock (RTC) is mandatory. Keeping the time is essential in M2M systems in almost every application and this is particularly true for telemetry and telematic. Coin cell battery is normally used for this purpose, although super capacitor may be also the choice.

Peripheral block is an optional part of M2M terminal. Commercially available terminals normally provide expansion connectors with various interfaces (ADC, GPIO, SPI, etc.) [5, 6, 7, 8]. In many applications sensors and actuators are external devices [9] and the communication with the terminal is established using cables. The terminal may provide power pin at the expansion connector, usually for sensors and additional interfaces, while some actuators may require dedicated power supply.

The central block from Fig. 2 can be implemented by using two system design architectures: i) component–based, where every part of the block is a separate device; ii) integrated, where the complete block appears as a single board/chip device. Key features of each design architecture are summarized in Tab. 1.

In the component–based design architecture, a MCU is connected to cellular module using serial (RS232) communication. Embedded software is executed by the MCU and cellular module is used as a modem only. Variety

Table 1. Key features of the component–based and integrated central block of the M2M terminal.

	Component–based	Integrated
MCU/MPU	Versatile choice	Fixed
Storage memory capacity	Flexible	Fixed
Interfaces	Flexible	Fixed
Backup supply charging	Bulit–in	Built–in
Firmware Over–The–Air	Limited	Full
Multithreading	Limited	Full
Embedded software	Procedural	Object–oriented
Cost	Low to medium	Medium to high

of modules are available, e.g. [10, 11, 12]. Interaction with the sensors and actuators is normally done using MCU's analog to digital converters and general purpose input/output pins, available through external connector. Real time clock is built–in into the module and its backup supply is provided by the battery, so there is no need for the dedicated RTC chip. Common practice is to include notification LE diodes, for startup and diagnostics of the MCU and status of the module. If the extensive data logging is not required, even the external storage memory may be omitted.

The main advantage of this architecture is hardware design flexibility. Depending on the requirements, a broad selection of MCUs is available. Using low–cost MCU, this can be a primary choice for M2M terminals performing relatively simple tasks. The terminals may be realized in a compact form. An example of the terminal using only digital GPIOs is shown in Fig. 3. Such a terminal is suitable for home or industrial automation. On the other hand, high–end MCUs or mixed signal system–on–chip [13] may be used for demanding applications involving sensor signal conditioning and/or more specific interfaces. The disadvantage is in relatively cumbersome procedure required for the embedded software update. MCUs are commonly programmed in–circuit, using dedicated hardware and interfaces, which makes Over–The–Air (OTA) updates virtually impossible.

In the integrated architecture there is only one part of the central block. Microprocessor (normally–ARM based), cellular module (modem), RTC, and storage memory are available in a single package, commonly referred to as "the module". Package can be board–to–board or land grid array and modules are produced in automotive, industrial, and commercial grades. Embedded software is executed by the module only. All standard interfaces

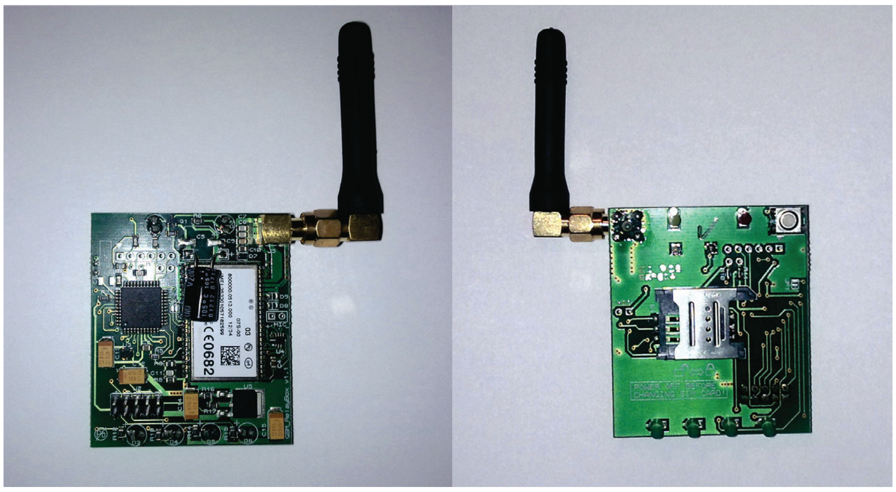


Fig. 3. Photo of the M2M terminal realized using component–based architecture (left – top view, right – bottom view). Board dimensions are 49 mm×57 mm.

(RS232, GPIO, ADC, DAC, SPI) are present on the module, so it can be connected directly to peripherals. Integrated flash memory has usually capacity of a couple of Megabytes or more, which makes the modules suitable for data logging applications. Commercially available standalone terminals are based on this architecture [7, 14]. Custom design is also possible, as shown in Fig. 4. The terminal is a single–board solution and it can be

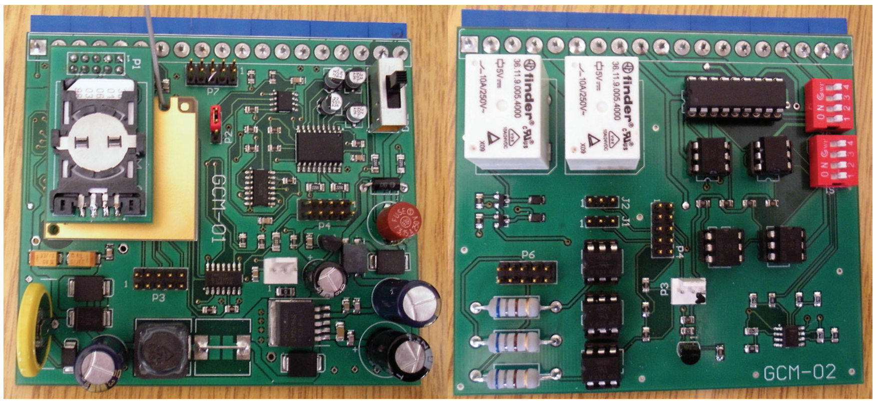


Fig. 4. Photos of the M2M terminal realized using integrated architecture (left), and peripheral board (right). Boards dimensions are 99 mm×88 mm.

used as a standalone device. In addition, it can be connected with a peripheral expansion board providing relays, opto–isolated I/Os, and analog signal conditioning circuitry. Boards can be stacked, so the terminal can be assembled in the enclosure appropriate for industrial environment, as illustrated in Fig. 5. An example of full-featured M2M terminal is shown in Fig. 6. In

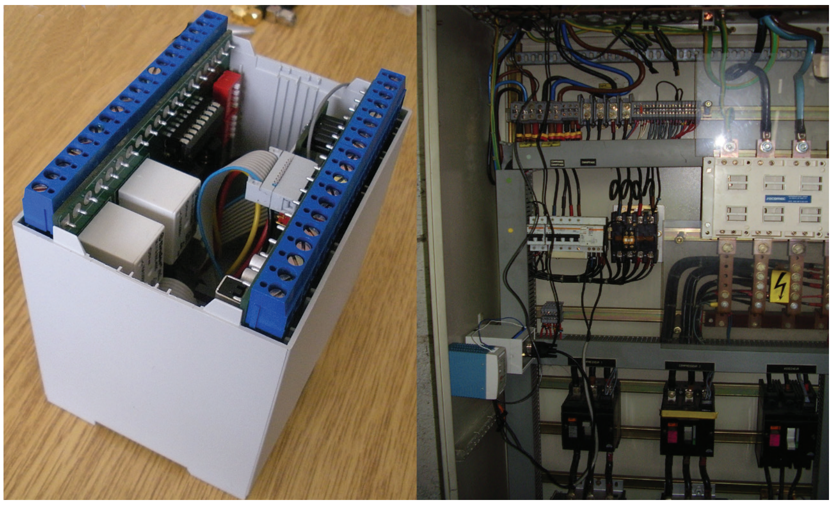


Fig. 5. Assembled M2M terminal (left), used in the industrial environment (right).

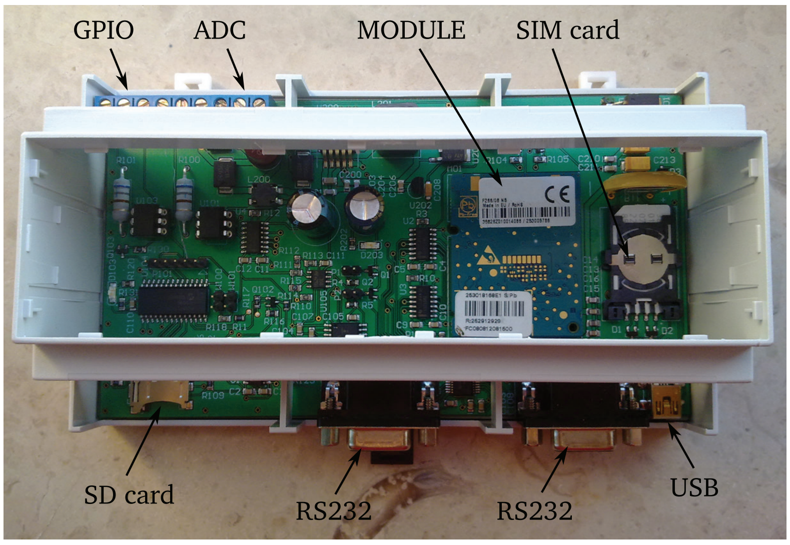


Fig. 6. Assembled M2M terminal based on the module with embedded operating system.

this case, the module is running an embedded operating system, so it can be used as a microcomputer capable to process in–situ relatively large amount of data.

The advantage of this type of M2M terminals is integrated MCU/MPU, which normally allows native multithreading/multitasking and the terminals may be designed to perform complex tasks. In addition, OTA is readily available for the embedded software update. On the other hand, number of the interfaces may be limited for some applications and hardware design can be more challenging due to the fact that modules usually have voltage I/O levels that require translators in order to comply with the standard peripherals. It is to be noted that an entry–level MCU still may be used as an external device in charge for the module's power–up, reset and watchdog functions. In this case, a simple and robust software procedure is executed in MCU, without the need of update. This solution results in a less complex and cheaper design in comparison to the usage of the dedicated circuitry.

3 Embedded software

As indicated in Tab. 1, embedded software development is dependent on the M2M terminal architecture. However, regardless of the architecture type, at early stage of development, a model–based engineering approach may be employed to establish a mapping between the hardware and the embedded software. One way is to use Architecture Analysis & Design Language (AADL) [15], as illustrated in Fig. 7 The model describes M2M terminal suitable

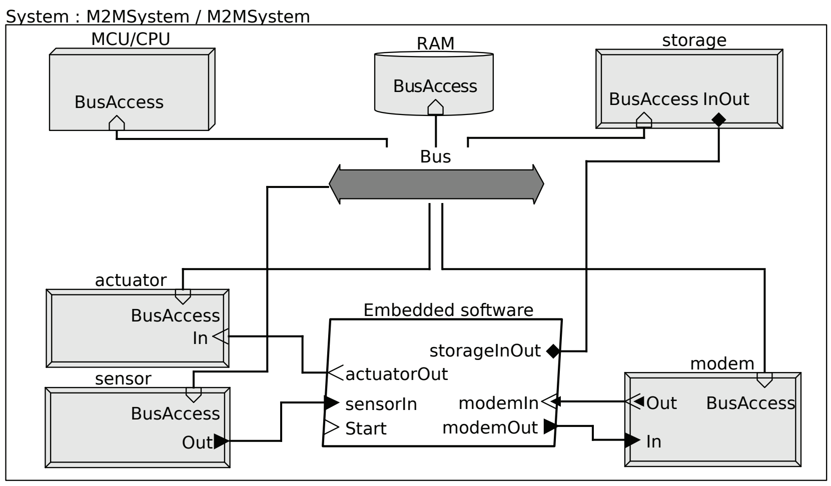


Fig. 7. AADL model of M2M terminal.

for telemetry/telematic applications [16]. Hardware devices communicate over common bus, while embedded software (executed by the MCU/CPU) is represented as a process mapped to the hardware by using virtual ports.

Procedural programming is used for terminals with component–based design architecture. For many MCUs there are well established programming techniques and ready–to–use libraries are available for a broad range of interfaces. Terminals with integrated design architecture are programmed using object–oriented approach. Some modules support high–level programming languages like Java Micro Edition [17, 18] or Python [14]. Other modules are running embedded operating system, typically Linux [19, 20], with language independent application programming interfaces.

A general use–case scenario for the terminal depicted in Fig. 7 may be expressed as:

- ✧ acquire data from the sensor and store them onto the memory,
- ✧ transfer stored data to the application residence periodically and/or on demand,
- ✧ raise alarm towards the application residence in response to the critical data values,
- ✧ control the actuator either using pre–configured setup, usually based on critical data values, or on demand.

The terminal should perform several tasks concurrently. Embedded software is designed using multithreading, usually within a single process [21]. Behavior analysis is used for identifying main threads and their synchronization. The software should listen modem's answers to the issued attention (AT) commands and also for unsolicited result codes (URCs) received from the cellular network. These events occurs asynchronously and are handled by a thread `Listener`, as shown in Fig. 8. Events are processed by using `EventHandler` thread, in the order of their appearance through `EventQueue`. The `Measurement` thread performs data acquisition, storage, issues alarms, and eventually controls the actuator. The `Timer` initiates periodical data transfer, which is performed by a `Transfer` thread.

Requirements for thread synchronization can be identified using the model from Fig. 8, which consists of data components and threads. In a concrete programming implementation, data components are classes having properties and methods. Threads `Measurement` and `Transfer` may need simultaneous access to hardware devices `modem` and `storage` (from Fig. 7). For example, initiation of the data transfer may come either from the `Timer` (periodical) or from the application residence (on demand), via `Listener` and `EventHandler`. Care should be taken to avoid overlapping between periodical and on demand data transfer, and to provide retry option in the

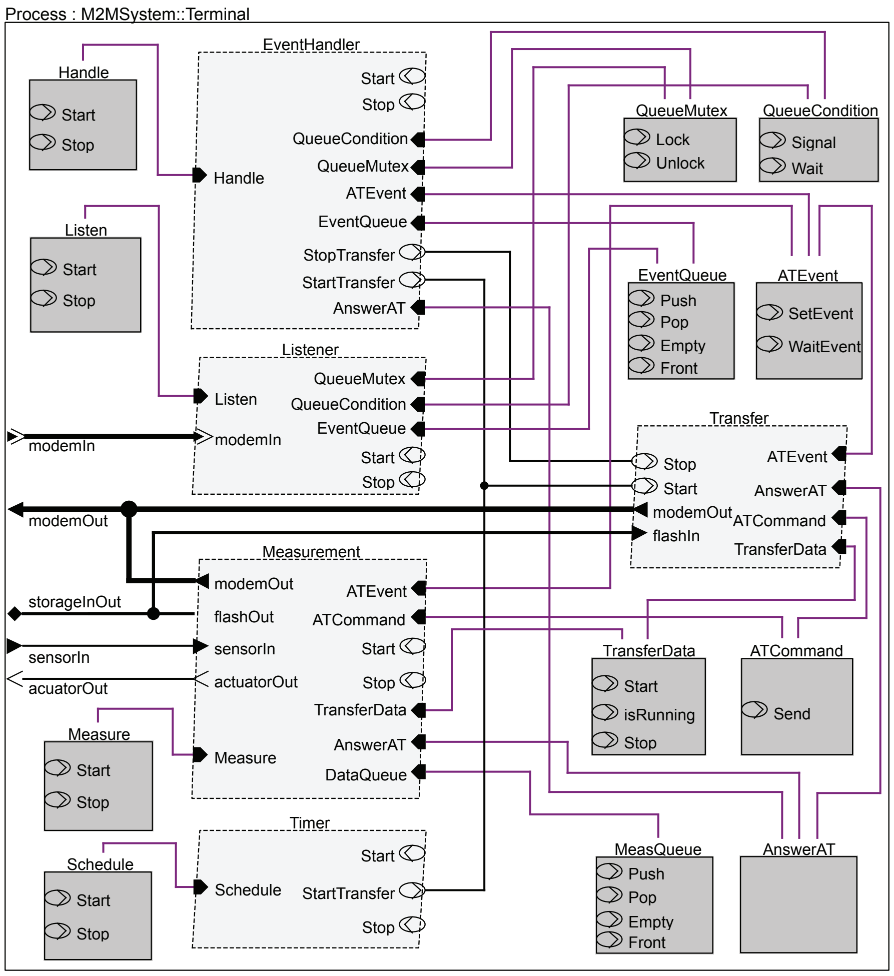


Fig. 8. AADL diagram of the embedded software of M2M terminal with threads and data components [16].

case of interrupted transfer. Also, it is necessary to ensure continual data acquisition or raise of alarm, while the transfer is in progress. Both threads may send AT commands to the modem using `ATCommand` data component and *wait* for the answer described by `ATEvent`. The `EventHandler` process the answer, assigns it to the `AnswerAT`, and *sets* `ATEvent`. The calling thread then can access `AnswerAT` and decide how to proceed.

Based on the representation from Fig. 8, a corresponding AADL model can be developed for each thread [16], and an example is shown in Fig. 9. The main function of the thread is to get data from the storage and transfer

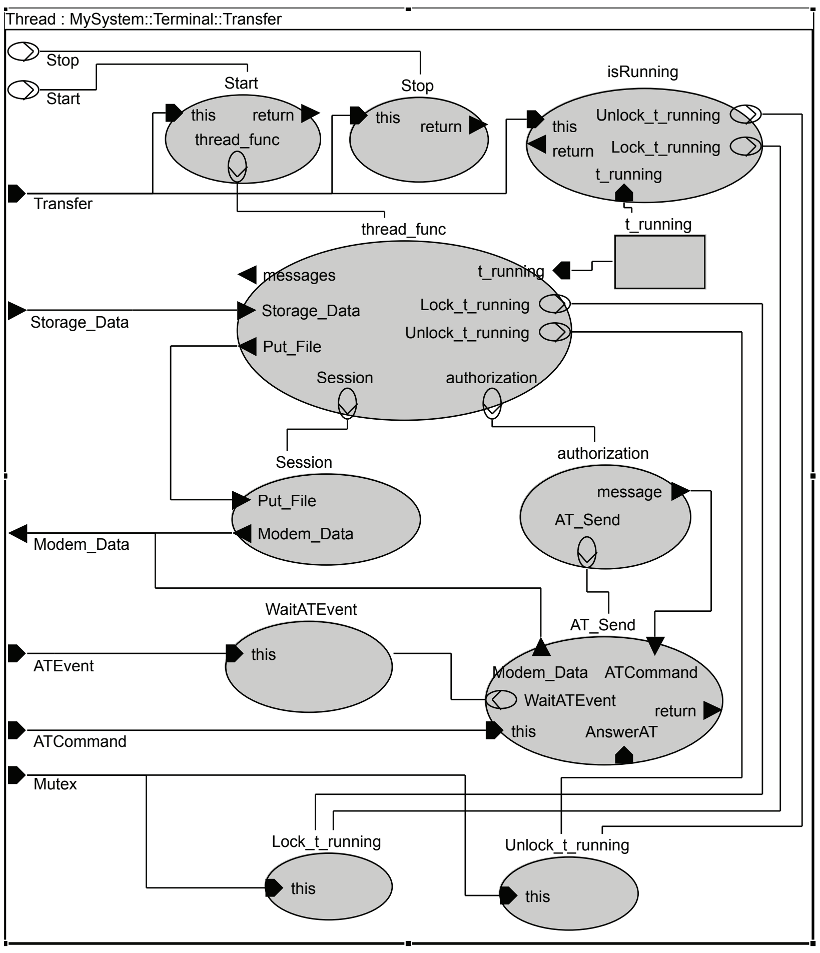


Fig. 9. AADL model of the `Transfer` thread.

them as a file to the application through modem. The thread should ensure that no other instance is running, which is achieved by mutual exclusion (`Mutex`). It also uses AT commands and events to check if data connection is possible on the cellular network before establishing a session, to verify access using authorization, and to report transfer status.

Terminals require configuration parameters describing network, severs, security, critical data values, etc. Parameters are usually kept in the configuration file, which can be updated using OTA procedure. The embedded software should make a backup copy of the configuration file and a feature to issue reconfiguration status notice to the application. Short Message Service (SMS) is frequently employed as a convenient way to change values of particular parameters and to report status. Normal security precautions are authorization of the calls/events coming from the cellular network, usage of passwords, and jamming detection, while encryption may be required in case of sensitive data.

4 Application

In most implementations of M2M systems, an application represents monitoring and command center, residing on a dedicated server. A common software requirement specifications for the application are to:

- ✧ collect data from the terminals,
- ✧ present data to the clients either in a raw or in a processed way,
- ✧ instruct the terminals to perform actions,
- ✧ reconfigure the terminals.

Terminals are not intended to collect and transfer a lot of data. In a telemetric application, measurements are carried out in a reasonably large intervals and results are normally transferred periodically, e.g. once per day. Typical data record contains ⟨date/time⟩⟨measured value⟩ pair, and file size for the transfer is usually few kilobytes. Data transfer requires bearer within the cellular network with services (e.g. GPRS, EDGE, 3G), and the corresponding protocol, normally TCP–based.

Cellular modules have embedded TCP/IP stack, normally supporting protocols like HTTP, FTP and SMTP. File Transfer Protocol (FTP) is convenient solution for data transfer in many M2M applications. In the example illustrated in Fig. 10 FTP server is used to receive data from the terminals. Received data are stored into the database, and are accessible to the Web server through Common Gateway Interface (CGI). Client can view data by sending HTTP request to the Web server. Client can also generate command and configuration files which are forwarded to the terminals. Optionally, a SMS gateway can be used as an alternative way for communication with the terminals. Application may be realized with server side that includes scripting, thus requiring only Web browser on the client's side.

Basic information on each terminal like IMEI, alias, location, etc. are entered into the database and related with the terminal's configuration and data files. Queries are used to generate numerical or graphical representations of the collected data. Datasets from multiple terminals may be viewed and compared, as illustrated in Fig. 11. Data volume may become large over time and archiving procedure for obsolete or redundant records is provided, in order to maintain performance of the application. It is notable

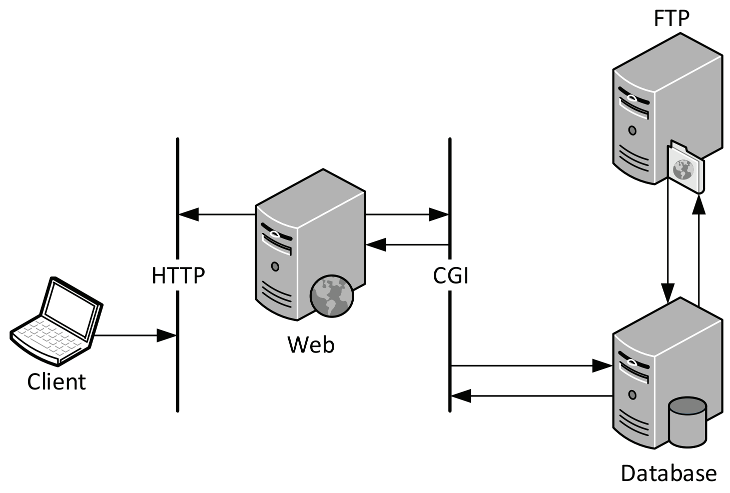
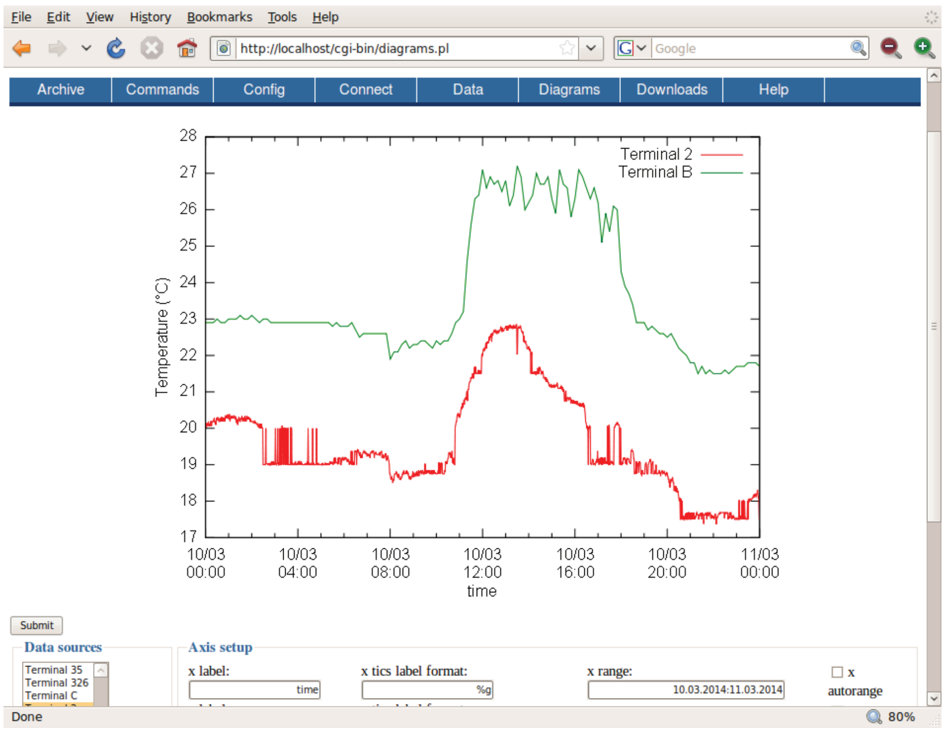


Fig. 10. Implementation example of M2M application architecture.

that powerful and feature–reach applications may be released using open source solutions, which may significantly reduce overall cost of M2M system implementation [22].

A massive M2M system can be implemented in government sectors. As an example, fiscal cash registers in Serbia are equipped with M2M terminals, either external or built–in. Application from tax authorities requires periodical transfer of the turnover, tax and reset data from each cash register's fiscal memory. Terminals are instructed on the scheduled time for transfer, sort of the requested data, and period for which data are requested by reading configuration file from the FTP server at tax authorities residence. Cash register operator is visually notified about the request by LE diode or display. The only human interaction with the terminal is putting the cash register into the transfer mode. Terminals are in virtual private network and GPRS is used as data service. Over 150000 cash registers are integrated into the system.

Designing a proper scheduling for data transmission is a challenge for some M2M systems. Possibility of network congestion always exists, and reliability and security of the transfer is concern [23, 24, 25]. Energy efficiency is also subject of extensive research [3, 26]. Although segmentation may be one of the solutions, in the case of massive M2M systems transmission scheduling schemes are still strongly dependent on both the infrastructure and the concrete application and may require new models [27].



Data sources

Terminal 35
Terminal 326
Terminal C

Axis setup

x label: time

x tics label format: %g

x range: 10.03.2014:11.03.2014

☐ x autorange

Fig. 11. Screenshoot of the M2M application client's interface showing temperature data collected from two terminals located at different sites in 24 hour period.

5 Conclusion

This paper has presented an overview of designing process for cellular M2M systems in the configuration of wireless sensor network. However, M2M systems have potential to expand the communication platforms between wireless devices in a broader scope and with a variety of scenarios [3, 28]. Recent advances in communication technologies have emerged Internet of Things, which may be seen as an extension of M2M systems, although distinction between the two is gradual rather than sharp. Some predictions states that M2M communication in the near future will exceed machine–to–human communication [1]. Nevertheless, it should be emphasized that human control is still of crucial importance in interpreting M2M data and making feedback decisions.

Acknowledgments

This work was supported by the Serbian Ministry of Education, Science, and Technological Development, projects No. TR11018 and No. TR32026, and by Ei–PCB Factory, Niš, Serbia.

References

[1] D. Bosswarthick, O. Elloumi, and O. Hersent, Eds., *M2M Communications: A systems Approach*. New York: Wiley, 2012.

[2] ETSI. (2015) Accessed: February 22, 2015. [Online]. Available: http://www.etsi.org/technologies-clusters/technologies/m2m

[3] S. Ajah, A. Al-Sherbaz, S. Turner, and P. Picton, "Machine-to-machine communications energy efficiencies: the implications of different M2M communications specifications," *Int. J. of Wireless and Mobile Computing*, vol. 8, No.1, no. 1, pp. 15–26, 2015.

[4] Micrel. (2014) ANLPS300: How to Power GSM/GPRS/EDGE/3G/HSPA M2M Modems. Application note. Accessed: February 26, 2015. [Online]. Available: http://www.micrel.com/_PDF/App-Notes/ANLPS300.pdf

[5] Sierra Wireless. (2015) Accessed: March 09, 2015. [Online]. Available: http://www.sierrawireless.com/productsandservices/airlink_gateways_modems_networking_solutions/programmable_modems/fx_series/

[6] HCP. (2015) Accessed: March 09, 2015. [Online]. Available: http://www.hcp.rs/en/products/communications-

[7] Gemalto M2M Terminals. (2015) Accessed: March 09, 2015. [Online]. Available: http://m2m.gemalto.com/products/terminals.html

[8] GT863-3GG. (2015) Telit. Accessed: March 09, 2015. [Online]. Available: http://www.telit.com/products/product-service-selector/product-service-selector/show/product/gt863-3gg/

[9] Z. Prijić and A. Prijić, "Current and Pressure Sensors in M2M System," in *Proc. 52. Conference ETRAN*, Palić, Serbia, June 2008, pp. MO2.3–1–MO2.3–4, (in Serbian).

[10] u–blox. (2015) Cellular modules. Accessed: March 04, 2015. [Online]. Available: http://www.u-blox.com/en/wireless-modules.html

[11] Sierra Wireless. (2015) Accessed: March 06, 2015. [Online]. Available: http://www.sierrawireless.com/productsandservices/airprime_wireless_modules/essential_modules/

[12] Telit. (2015) Accessed: March 06, 2015. [Online]. Available: http://www.telit.com/products/product-service-selector/product-service-selector/show/product/ce910-sl/

[13] Cypress Semiconductor Corporation. (2015) Programmable System–on–Chip. Accessed: March 17, 2015. [Online]. Available: http://www.cypress.com

[14] Telit. (2015) GT864-QUAD/PY. Accessed: March 23, 2015. [Online]. Available: http://www.telit.com/products/product-service-selector/product-service-selector/show/product/gt864-quadpy/

[15] P. Feiler and D. Gluch, *Model-Based Engineering with AADL: An Introduction to the SAE Architecture Analysis & Design Language*, ser. The SEI Series in Software Engineering. Addison-Wesley Professional, 2012.

[16] A. Prijić, Z. Prijić, D. Vučković, and A. Stanimirović, "AADL modeling of M2M terminal," in *27th International Conference on Microelectronics Proceedings (MIEL)*, Niš, Serbia, May 2010, pp. 373 – 376.

[17] "Cinterion TC65i Wireless Module," Datasheet, 2014. [Online]. Available: http://www.gemalto.com/m2m/modules-terminals

[18] "Cinterion EHS6 Wireless Module Global 3G with Java embedded," Datasheet, 2014. [Online]. Available: http://www.gemalto.com/m2m/modules-terminals

[19] Sierra Wireless. (2015) Legato™ Platform Open source embedded platform built on Linux. Accessed: March 24, 2015. [Online]. Available: http://www.sierrawireless.com/productsandservices/airprime_wireless_modules/smart_modules/legato/

[20] "MO300E / XS300E Linux Powered EDGE M2M Modules," Datasheet, Sagem Communications, 2008.

[21] B. Sandén, *Design of Multithreaded Software*, ser. IEEE Computer Society Publications. New Jersey: Wiley, 2011.

[22] Z. Prijić, A. Prijić, D. Vučković, and N. Jordanov, "Practical Implementation of the Industrial M2M System," in *Proc. 54. Conference ETRAN*, Donji Milanovac, Serbia, June 2010, pp. MO2.1–1–MO2.1–4, (in Serbian).

[23] K. Hojgaard-Hansen, T. K. Madsen, and H.-P. Schwefel, "Reducing communication overhead by scheduling tcp transfers on mobile devices using wireless network performance maps," in *European Wireless, 2012. EW. 18th European Wireless Conference*, Poznań, Poland, April 2012, pp. 1–8.

[24] G. Madueño, C. Stefanovic, and P. Popovski, "Reliable Reporting for Massive M2M Communications With Periodic Resource Pooling," *IEEE Wireless Communications Letters*, vol. 3, no. 4, pp. 429–432, Aug 2014.

[25] E. Kartsakli, A. S.Lalos, A. Antonopoulos, StefanoTennina, M. DiRenzo, L. Alonso, and C. Verikoukis, "A Survey on M2M Systems for mHealth: A Wireless Communications Perspective," *Sensors*, vol. 14, pp. 18 009–18 052, 2014.

[26] C. Pereira and A. Aguiar, "Towards Efficient Mobile M2M Communications: Survey and Open Challenges," *Sensors*, vol. 14, pp. 19 582–19 608, 2014.

[27] I. Stojmenović, "Machine-to-machine communications with in-network data aggregation, processing, and actuation for large-scale cyber-physical systems," *IEEE Internet of Things Journal*, vol. 2, no. 2, pp. 122–128, 2014.

[28] J. Kim, J. Lee, J. Kim, and J. Yun, "M2M Service Platforms: Survey, Issues, and Enabling Technologies," *IEEE Communications Surveys Tutorials*, vol. 16, no. 1, pp. 61–76, 2014.