Software Defined Radio Assisted Localization for Emergency Scenarios

Enrico Del Re[1], Luca Simone Ronga[1], Luca Vettori[1], Simone Morosi[1],
Letizia Lo Presti[2], Emanuela Falletti[3], and Marco Pini[3]

[1] Dipartimento di Elettronica e Telecomunicazioni, Università di Firenze, Italy
[2] Dipartimento di Elettronica, Politecnico di Torino, Italy
[3] NavSAS Group, Istituto Superiore Mario Boella (ISMB) Torino, Italy

Abstract. In the following lines we introduce a Software Defined Radio prototype terminal for Emergency Situations. More specifically this integrated terminal will have NAV/COM functions and so it will be composed by two modules: the Localization and Communication Components, running on the same machine and connected each other via TCP/IP protocol. The Localization component will have to find out the terminal position itself and the Communication component will have to communicate it to other terminals. Both components of this prototype, thus, will be interfaced by their own front-ends, so providing the navigation and communication services.

Keywords: emergency situations, assisted localization, NAV-COM devices, software defined radio terminal.

1 Introduction

The activities of the scientific research communities have been recently motivated towards the topics of the communications and networking technologies for public safety and security. This trend has been enforced by the terrorist menace and the attention which has been given to catastrophic events.

As a result, a big research activity has been accomplished at international level in order to define modern, interoperable communications and networking standards for emergency response and public safety. Communication and navigation services in an emergency scenario are characterized by extremely different requirements from the classical ones. Access to permanent bidirectional links between a control center and teams of interveners (e.g., first aid groups) is crucial in handling emergencies and must be available for the entire duration of the intervention. Besides, knowing on time and accurately the position of the rescue entities (persons and means) is a fundamental enhancement in the coordination and planning of emergency and disaster relief operations, both in terms of efficacy and safety of the interventions. Indeed, the ETSI Recommendation TS 102 181 [1] recognizes the importance of providing “real-time information regarding the position of personnel or vehicles to a command point”. Therefore the integration of communication services and positioning/navigation capabilities in

K. Sithamparanathan et al. (Eds.): PSATS 2010, LNICST 43, pp. 425–435, 2010.

a unique, portable, robust and user-friendly terminal (professional NAV/COM device) results a key feature in this kind of applications.

Navigation capabilities, for example using GNSS (Global Navigation Satellite Systems) services, are necessary to provide the 3D terminal position (with better accuracy possible). Then, localization information should be automatically transmitted to the suitable entities as soon as it is available, without human intervention. Unfortunately, GNSS signal reception is dramatically impaired by Line-Of-Sight (LOS) obstructions and multipath, especially in urban environments, as well as by strong attenuations due to walls and slabs in indoor conditions. Several strategies to improve positioning performance in critical conditions have been studied [3,4,5,6]; among them, the concept of "cooperative localization" seems to be particularly suitable for the scenario addressed in this work, in order to improve overall system reliability by the exploitation of all the available technology at the maximum extent, as expected for a system that can save human lives. The basis for the cooperative localization investigate here is an Assisted GNSS (AGNSS)-like approach, which exploits the COM network architecture with specifically added feature to improve localization performance.

The main objective of this paper is the definition of a prototypical architecture for a reconfigurable and flexible user terminal open to host the requirements of a professional NAV/COM device for emergency interventions. The Software Defined Radio (SDR) technology is the technique chosen to reach this requirements because of its intrinsic flexibility and reconfigurability [2]. As it is well known SDR is a technique for moving digital signal processing as close as possible to the antenna. This kind of solutions also claims for a higher flexibility at the physical layer and a smarter intelligence at the other lower layers: as a result, the enabling SDR technologies are often considered together with a new radio paradigm that implements self-reconfigurability, the so-called Cognitive Radio (CR).

This paper is organized as follows. In Section 2 the Integrated NAV/COM Terminal and considered scenario are presented. In Sections 3 and 4 the Location and Communication components are better described. In Section 5 we present the concluding remarks.

2 Integrated Terminal Architecture

As stated in the introduction, the indoor/outdoor location and tracking capability with better possible accuracy has been recognized as the single most important technological development enhancing emergency response operations. A single GPS receiver has good performance of providing optimal capability outdoor. Nonetheless, a challenging situation is when a terminal attempts the localization both in indoor and in the outdoor with hostile environment (e.g., because of foliage or heavy scattering from dense and metallic structure).

In the considered scenario (depicted in Fig. 1), one or more terminals (associated to a first aid group, for instance) are spread on the territory of the intervention. A certain fraction of the terminals are in LOS visibility with four

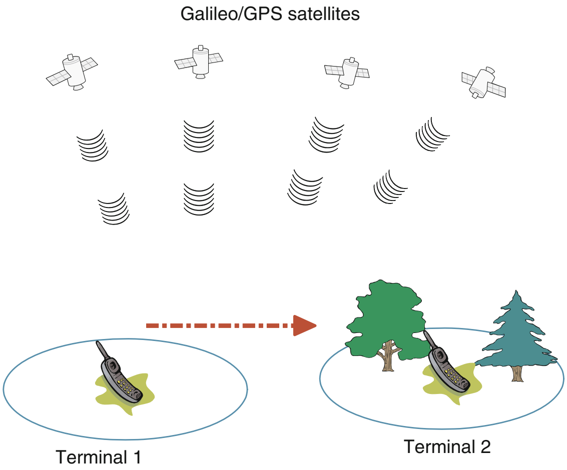


Fig. 1. Considered scenario: the Terminal 1 is in LOS visibility, whereas Terminal 2 is in Non-LOS condition

or more GPS satellites, as Terminal 1 in the figure. On the contrary, the other terminals (Terminal 2 in the figure) are in a Non-LOS condition so that their GPS receiver is severely impaired, but not completely blocked. In this case, an AGNSS-like solution seems profitable, since terminals with reliable GPS position (that could be seen as "anchor nodes") may send their navigation parameters to the impaired ones, such as the ID of satellites with the best C/N$_o$, Doppler frequencies and orbital parameters. Although these local information needs to be refined in the impaired receiver, it could significantly improve performance of the impaired nodes in at least two ways:

1. strongly reducing the acquisition time and false-alarm probability for the medium-impaired terminals, as well as easing the tracking mode;
2. enabling the acquisition and tracking of the necessary satellite signals for strongly-impaired terminals, whose autonomous acquisition process would be too much prone to errors to be reliable.

Notably in such situations the presence of multiple sets of assistance information, received from multiple anchor terminals, should be exploited in the most fruitful way as a form of information diversity, able to potentially increase the reliability and accuracy of the AGNSS procedure.

Consequently, the integration of communication systems to localization/ navigation services in the rescuer's terminal can be extremely effective in a scenario like that has just been envisaged. More specifically a reconfigurable NAV/COM radio terminal may be the key element in the design of an effective and robust emergency network.

With the goal of developing a prototype of such a professional NAV/COM terminal we consider a software-defined radio approach, that is the terminal becomes a software program loaded into a programmable device equipped with proper RF front-ends. Only what is feasible with the present CPU technology will be considered. This reconfigurable SDR terminal will have to take into account heterogeneous communication networks based on terrestrial radio interface and so TETRA, Wi-Fi/WiMAX and UMTS/LTE systems will be considered.

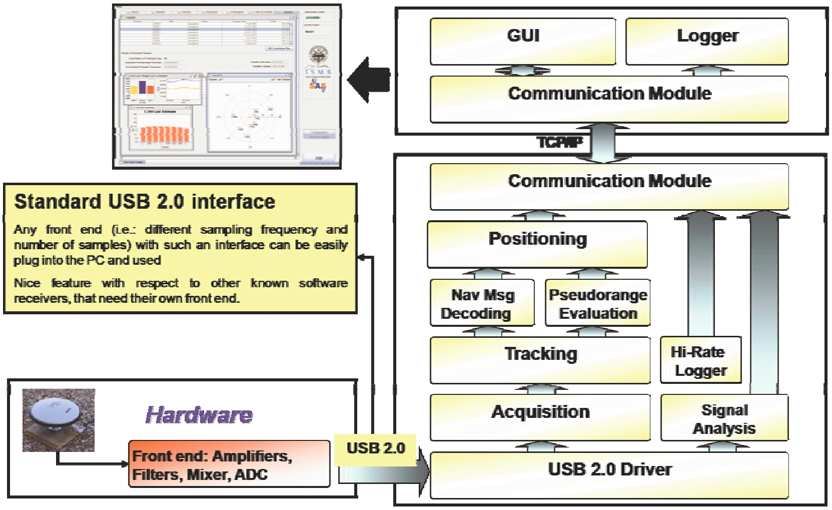
The prototype architecture of our integrated terminal is based on two main modules: the Localization and Communication Component. They are expected to become nothing but two separated software programs running on the same PC, properly interfaced with each other and with their respective navigation and communications radio front-ends, for instance resident on specific USB-connected boards. Thereby, the PC prototype becomes, in fact, a real NAV/COM terminal with cooperative localization capabilities.

3 Localization Component

The SDR localization component is based on the N-Gene platform, an 8-bits fully software GNSS receiver, able to receive the Global Positioning System (GPS) Coarse Acquisition (C/A) code on L1, with the ability to process more than 12 channels in real time, and track live GIOVE-A and GIOVE-B[1] signals transmitted on the E1 band. The receiver has been completely developed at the NavSAS lab in Politecnico di Torino/ISMB during the last two years, with the aim of equipping the lab with a powerful, quickly usable and extremely versatile tool to receive and analyze GNSS signals; the fully-software reprogrammability, together with the real-time processing capability, allow lab researchers to quickly test new receiving algorithms and architectures directly to real or simulated signals, as well as performing completely controlled data collections and signal analysis campaigns. In fact, the receiver can be used with any RF front-end that exports data through an USB interface. The 8-bits quantization allows for implementing all of those signal processing blocks requiring a fine resolution, as for example notch filters for interference mitigation, pulse blanking techniques and multipath mitigation techniques based on the estimation of the parameters of the input signal. The reconfigurability of its software modules makes N-Gene the optimal tool to realize the Integrated Terminal architecture with NAV/COM capabilities introduced in the previous section.

The high level of flexibility is the prominent feature that has made SDR architectures an indispensable tool to evaluate competing designs and has opened the possibility of multi-profile core platforms at commercially affordable costs, which can switch to different application profiles and different performances by means of a few simple software instructions. As a step forward, future GNSS SDR receivers could adaptively reconfigure their own architecture to the most appropriate one for the immediate environment they have to cope with, in order to maximize their efficiency. In a research lab, a real-time fully software architecture represents the ultimate simulation/testing environment, since it provides a

[1] GIOVE-A and GIOVE-B are the two first satellites of the European GNSS system, Galileo).



Any front end (i.e.: different sampling frequency and number of samples) with such an interface can be easily plug into the PC and used

Nice feature with respect to other known software receivers, that need their own front end.

Fig. 2. N-Gene block diagram

single architecture able to process simulated signals generated in controlled environment and live signals. It allows in this way to test and validate novel receiving architectures and signal monitoring procedures without replacing hardware components: all the necessary updates become a software matter. These are the fundamental reasons that has led to the proliferation of SDR solutions in the field of GNSS receivers since the last decade [7].

Figure 2 shows a high level block diagram of the N-Gene software architecture. The left part of the diagram represents the front end, which is the only piece of dedicated hardware required. The USB 2.0 connection makes the receiver particularly versatile to be connected to virtually any RF front-end with USB interface. At the moment the platform has been successfully tested with different commercial and prototype front-ends, featuring 3-to-6 MHz IF bandwidth @L1/L2 and 1-to-8 quantization bits using a sampling frequency in the range 13-to-20 MHz. The right part of Figure 2 represents the N-Gene software routines. The USB driver controls the data flow from the ADC, while the acquisition and tracking blocks are the core of the baseband signal processing. The former seeks for the satellites in view, while the latter synchronizes the incoming codes and carriers with local replicas. As soon as the receiver tracks at least four satellites, it is able to compute the user position and velocity, through the estimation of the distance between the user and the set of satellites (i.e.: pseudorange estimation).

The receiver is controlled by the Graphical User Interface (GUI) through a TCP/IP protocol. This design allows for controlling the receiver from a remote PC, in case this is needed, and in this project represents the enabling feature for the implementation of a cooperative localization architecture among two or more terminals. Indeed, thanks to the software flexibility, the user is allowed to configure the receiver by specifying a wide range of parameters, for example by assigning/excluding specific signals (i.e.: GPS, Galileo or EGNOS) or satellite codes to each channel, configuring thresholds and Doppler search of the signal acquisition, selecting the tracking loop bandwidths, the Delay Lock Loop (DLL) spacing and the integration time. Evidently, similar configuration rights could be assigned to an automatic procedure driven by a remote terminal that provides assistance to the receiver in case of critical LOS conditions, so as to implement a form of cooperative localization. On the other side, the N-Gene-based assistance terminal is able to output its relevant parameters thanks to the same TCP/IP link to an additional "virtual GUI," represented in this setup by the impaired terminal.

The current operative platform of the SW receiver is a Gentoo Linux. Running on a 3 GHz processor (Intel Pentium 4 standard architecture) the maximum CPU load is 50% with 12 satellite channels in navigation and performing PVT (Position, Velocity and Time) computation at 2 kHz rate. These SW characteristics are favorable toward the integration in a single terminal of the Navigation and Communication Components, thanks to the common Operative System (OS) platform and to the fair CPU load featured by the modules.

As far as the N-gene positioning performance are concerned, in open sky the receiver achieves a position accuracy lower than 10 meters rms, using code-based measurements and without applying carrier smoothing techniques. In case of "cold start," the Time-To-First-Fix (TTFF) is lower than 45 s. These performance figures show that the receiver is surely competitive with respect to any other commercial/mass-market solution in terms of positioning performance, while it gains in versatility of use.

For example, it can be used to accurately monitor the signal C/N_o ratio in passing from a LOS (open sky) condition to a partially blocked condition (partial non-LOS), for instance under foliage or in a area populated by high buildings, thanks to its accurate procedure to estimate the C/N_o [8]. As soon as the measure falls below a certain threshold for satellites above the elevation mask, a procedure can be started to decide if assistance is needed and, in the case, to broadcast an assistance request through the Communication Module. A companion receiver, whose Communication Module has sensed the assistance request and whose signal quality control parameters have declared good signal quality, will activate a procedure to communicate back the subset of assistance parameters.

The assistance procedure can be borrowed and re-adapted from the AGNSS standards (N-Gene can already host specific software routines to recovery AGNSS data using an OMA-SUPL protocol [10]). AGNSS is a network-assisted augmentation method that integrates GNSS with information provided by the cellular network (namely, GSM/GPRS and UMTS) to the aim of reducing TTFF and increasing the receiver sensitivity. GNSS receiver sensitivity might be enhanced by reducing the acquisition search space in both Doppler shift and phase domains, thanks to a rough information about the most probable Doppler shift and code phase computed and transmitted by the cellular network. Without resorting to the complete GSM/GPRS/UMTS assistance protocols [9,10], which necessitate in any case of the availability of connection to public services,

Table 1. Possible assistance parameters in AGNSS

Assistance data
Visible satellites
Position of the visible satellites (azimuth, elevation)
Predicted codes phase (and confidence interval)
Code phase confidence intervals ("search window")
Predicted Doppler shifts and Doppler rates
Satellite ephemeris
Satellite almanac
Satellite clock corrections
Navigation time (GNSS time)
Navigation data bits
user terminal location (either coarse or precise estimation)

possibly damaged or overloaded in emergency situations, it is possible to derive ad ad-hoc assistance procedure for the integrated terminal, lighter and possibly more flexible than the standard AGNSS. The list of the common assistance data that may by included in an assistance message is shown in Table 1.

The N-Gene software architecture features a TCP/IP socket and allows for recovering all the parameters involved in the assistance procedure from a remote server. This characteristic is used to connect the receiver to the communication component described below and implement an efficient hybrid NAV/COM system suitable in emergency situations.

4 Communication Component

GNU Radio [11] is an open source software toolkit for building and deploying Software Defined Radios. It is a platform for signal processing on commodity hardware. In fact, when combined with a minimal hardware, GNU Radio allows the construction of radios that the transmitted (and received) waveforms are defined by software.

More in specific GNU Radio is a hybrid system C++/Python that works on several operating systems among which Linux that we will use. The radiocommunication chain can be represented by a graph where the vertexes are the signal processing blocks and the edges represent the data flow between them. The signal processing blocks are implemented in C++ and the graphs are constructed and run in Python. Therefore, Python is used to "glue" the signal processing blocks constituting the communication chain. One interesting feature of GNU Radio is that the parameters, and even the inner structure, of the graph can be changed "on the fly". Thus, our terminal can change its inner structure to adapt to the existing communication systems.

The hardware platform usually combined to GNU Radio is the Ettus Universal Software Radio Peripheral (USRP)[12], a general purpose motherboard that gives access to the radio frequency spectrum and can be connected with any computer (that support GNU Radio) with a USB 2.0 port. The USRP is designed to allow general purpose computers to function as high bandwidth software radios. In essence, it serves as a digital baseband and IF section of a radio communication system.

It has 4 high-speed analog to digital converters (ADCs), each at 12 bits per sample, 64MSamples/sec. There are also 4 high-speed digital to analog converters (DACs), each at 14 bits per sample, 128MSamples/sec. These 4 input and 4 output channels are connected to an Altera Cyclone EP1C12 FPGA. The FPGA, in turn, connects to a USB2 interface chip, the Cypress FX2, and on to the computer. Usually the USRP motherboard supports four daughterboards, two for receiving side and two for transmitting side.

RF front-ends are implemented on these daughterboards. There are some type of them depending by the frequency, ranging from baseband up to 2.9 GHz. We use the daughterboards that operate in the 2.4 GHz band. The basic design philosophy behind the USRP has been to do all of the waveform-specific processing, like modulation and demodulation, on the host CPU. All of the high-speed general purpose operations like digital up and down conversion, decimation and interpolation are done on the FPGA. Finally, to complete the communication chain we use Radio Shack 15-215 Antenna.

For what concerns the physical layer for our terminal, we will consider the OFDM modulation because it is efficiently used in latest broadband wireless communications system, like IEEE 802.11 (Wi-Fi), 802.16 (WiMAX), HYPERLAN and 3GPP Long Term Evolution (LTE). Now we introduce a brief overview of OFDM modulation. The sequential data stream (that could be a PSK or QAM symbol) is converted to a parallel stream and the vectors of which are interpreted as signal in the frequency domain. By applying an inverse fast Fourier transform (IFFT), these data are transformed into the time domain and transmitted on a set of orthogonal carriers with separation Δf[2]. To avoid intersymbol interference (ISI) the last L samples of the body of the OFDM symbol (N samples long) are copied on the head of the symbol, called *cyclic prefix*, to form the complete OFDM symbol. At the receiver we have the inverse blocks of the transmitter and the data are retrieved by means of a fast Fourier transform (FFT).

GNU Radio provides the signal processing blocks implementing the OFDM transmitter/receiver flow graph (depicted in Fig.3). In particular the transmitting chain has a block used to insert "pre-modulated" preamble symbols before each payload. Instead on the receiving side we have a block for the symbol synchronization (realized with Maximum Likelihood method or Pseudo-Noise Sequence correlation) and another block that build an OFDM correlator and equalizer using the known preamble symbols.

More specifically for our simulation we consider a simple OFDM symbol structure for data transmission. We use 200 sub-carriers on 512 elements for the FFT operation with 128 samples for the value of cyclic prefix. Consequently, utilizing the USRPs we obtain a subcarrier spacing equal to 625 Hz (that essentially

[2] To preserve orthogonality between subcarries $\Delta f = \frac{1}{T_u}$ where T_u is the useful OFDM symbol period.

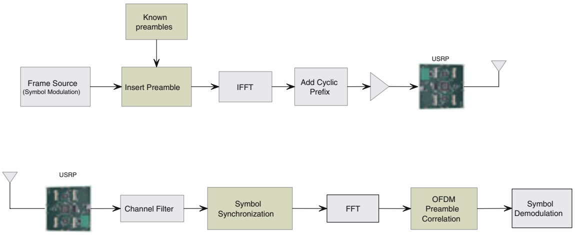
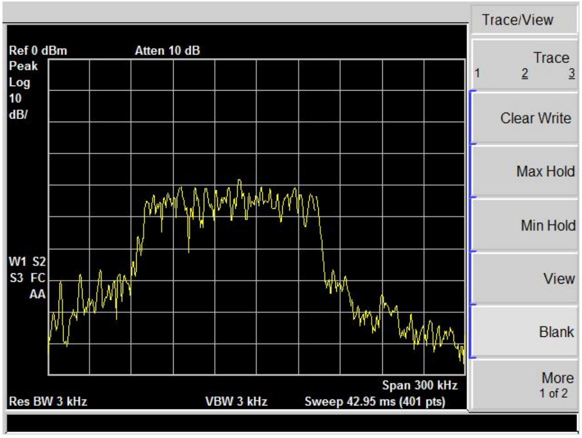


Fig. 3. Transmitting and Receiving chain of OFDM communication system on GNU Radio platform



Ref 0 dBm Atten 10 dB
Peak
Log
10
dB/

W1 S2
S3 FC
AA

Span 300 kHz
Res BW 3 kHz VBW 3 kHz Sweep 42.95 ms (401 pts)

Fig. 4. Bandwidth of ODFM signal created by USRP

depends on the ADC's number of samples and interpolation rate) for a maximum bandwitdh of 125 kHz and a theoretical bitrate of 100 kbps (utilizing a BPSK modulation for every sub-carrier). These values are confirmed by the Fig. 4 that represent the output screen of a spectrum analyzer placed close to the transmitting USRP. Finally, we could consider the implementation of simple coding channel (e.g. block or convolutional coding) in order to increase the performance in term of error rate.

For what concern the interface between NAV and COM modules a reasonable choice seems to be the use of TCP sockets realized by, for instance, Python routines, as we can see from the Fig. 5.

Another interesting feature of GNU Radio is the opportunity to have particular drivers, called universal TUN/TAP drivers [13] for "tunneling" the packets from the USRP to the Linux kernel (via USB 2.0). These TUN/TAP drivers provide the reception and transmission of packet for user space processes. In fact it can be seen as a simple Point-to-Point or Ethernet device, which, instead of receiving packets from physical media, receives the packets from user space

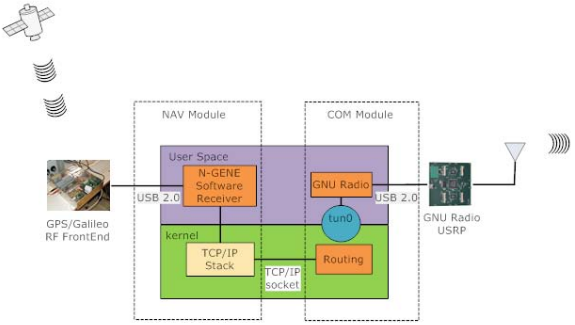


Fig. 5. Integration of NAV/COM Systems in proposed Software Defined Radio Terminal

process and instead of sending packets via physical media writes the packets to the user space process. In order to use the driver, the program has to open /dev/net/tun and issue a corresponding ioctl() to register a network device with the kernel. A network device will appear as tunXX or tapXX, depending on the driver chosen. When the program closes the file descriptor, the network device and all corresponding routes will disappear. Depending on the type of device chosen, the userspace program has to read/write IP packets (with tun) or Ethernet frames (with tap). Another interesting application using TUN/TAP is pipsecd, an userspace IPSec implementation that can use complete kernel routing. GNU Radio using these TUN/TAP drivers can create a program that provides a framework for building our own MAC protocols. The Linux 2.6 kernel (used in Ubuntu 8.10) includes already the tun module. What we have to do is to run two copies of this program on two different machines. So in this situation we can allow two machines to talk each other with the TCP/IP protocols. Thanks to this IP tunnel we can copy with a problem born considering the scenario in Fig.1. In fact, in order to reduce time acquisition by the impaired terminals (like Terminal 2 in figure 1) we have to maintain a time-synchronization between Terminal 1 (in full LOS visibility) and Terminal 2. So, we will try to realize this time-synchronization with the Network Time Protocol (NTP), a client-server TCP/IP protocol used to synchronize the internal clock of computers inside a network. It can be used on Linux platform, by easily configuring the daemon NTPd. Even the Precision Time Protocol, PTP (a protocol similar to NTP but that achieves more accuracy) and the Linux daemon PTPd will be investigate to achieve more accuracy compared to NTP.

5 Concluding Remarks

SDR represents in this work the enabling technology to realize the prototype of an integrated NAV/COM terminal intended to be used by first responders intervening in public emergency situations. The challenging feature is the provision of positioning capabilities in satellite-blocked (or partially blocked) environments, as are likely e.g. in urban or forest interventions. Such critical signal scenarios can be coped with a cooperative localization approach, which involves the exchange of navigation data between two NAV/COM terminals. Two SDR modules are then realized for each user terminal: a Localization SW module connected via TCP/IP to a Communication SW module. Thanks to the fully SW implementation, the presence of hardware components is reduced to the minimum extent, and the whole prototype can be hosted in a general purpose PC, this allowing to easily perform field test campaigns at really affordable costs.

Acknowledgment

This work has been supported by the Italian Ministry of the Education, University and Research (MIUR), under the PRIN 2007 program "SALICE" (`http://lenst.det.unifi.it/salice/`).

References

1. ETSI Recommendation TS 102 181 V.1.1.1 (2005-12), Emergency Communications (EMTEL); Requirements for communication between authorities/organizations during emergencies
2. Del Re, E. (ed.): Software Radio Technologies and Services. Springer, London (2001)
3. van Diggelen, F.: Indoor GPS theory & implementation. In: IEEE/ION Position Location and Navigation Symposium, PLANS 2002, pp. 240–247 (April 2002)
4. Sahmoudi, M., Amin, M.G.: A Maximum-Likelihood synchronization scheme for GPS positioning in multipath, interference, and weak signal environments. In: IEEE Vehicular Tech. Conf., VTC 2006 (September 2006)
5. Razavi, A., Gebre-Egziabher, D., Akos, D.M.: Carrier loop architectures for tracking weak GPS signals. IEEE Trans. on Aerospace and Electronic Systems 44(2), 697–710 (2008)
6. Dovis, F., Lesca, R., Margaria, D., Boiero, G., Ghinamo, G.: An assisted high-sensitivity acquisition technique for GPS indoor positioning. In: IEEE/ION Position, Location and Navigation Symposium, PLANS 2008, pp. 1350–1361 (May 2008)
7. Won, J.H., Pany, T., Hein, G.W.: GNSS Software Defined Radio. Inside GNSS Magazine, 48–56 (July/August 2006)
8. Falletti, E., Pini, M., Lo Presti, L., Margaria, D.: Assessment on low complexity C/N_0 estimators based on a M-PSK signal model for GNSS Receivers. In: IEEE/ION Position Location and Navigation Symposium (PLANS) (May 2008)
9. 3GPP Specications. Requirements for support of assisted global positioning system. 3GPP TS 25.171, v.7.1.0 (June 2006)
10. Open Mobile Alliance (OMA). Secure User Plane for Location (SUPL), OMA-TS-UPL, v.1.0 (June 2007)
11. GNU Radio Website (2009), `http://www.gnuradio.org`
12. Ettus Research LLC Website (2009), `http://www.ettus.com`
13. VTun, `http://vtun.sourceforge.net`