

SPINE (Signal Processing in Node Environment): a Framework for healthcare monitoring applications based on Body Sensor Networks

January 2008

Abstract

SPINE (Signal Processing in Node Environment) is a software framework for the design of Body Sensor Network (BSN) applications. It provides developers of signal processing algorithms for the analysis and the classification of sensor data with APIs and libraries of protocols, utilities and data processing functions. This provides application developers with an abstraction that improves interoperability and allows to reduce development time.

This Whitepaper presents the architecture and the main functionalities of the SPINE framework and describes the objectives and the principles of the Open Source project.

Introduction

Body Sensor Networks (BSNs) based on low power wireless sensor nodes placed on the human body and connected remotely to doctors or caregivers promise to enable in the next future a broad variety of health care and assisted living applications. However, there are still several technical challenges to be solved before the potential of BSNs is fully exploited. In particular, designing BSNs applications is still too difficult and time-consuming due to the lack of proper abstractions that support interoperability and hide low-level details to application developers.

The SPINE Open Source project aims to create a community of developers and build a framework useful to decrease development time and improve interoperability among signal processing intensive applications based on BSNs. SPINE provides libraries of protocols, utilities and processing functions, and a lightweight Java API that can be used by local and remote applications to manage the sensor nodes or issue service requests. By providing these abstractions and libraries, that are common to most signal processing algorithms used in BSNs for sensor data analysis and classification, SPINE also provides flexibility in the allocation of tasks among the BSN nodes and allows the exploitation of implementation tradeoffs. For example, SPINE supports distributed implementations of classification systems where signal processing functions are computed on the sensor nodes and the result sent to the BSN coordinator. This allows to reduce the amount of data exchanged between the BSN coordinator and the sensor nodes with respect to implementations where sensor nodes transmit raw sensor data.

The paper is organized as follows. The first section introduces Body Sensor Networks. The “What is SPINE?” section describes the architecture of the framework, its main components and interfaces. “The SPINE Community” section describes how the open source principles will apply to SPINE. Finally, the “Why to use SPINE?” section outlines the main advantages of the framework and describes an application example of a posture recognition prototype based on SPINE.

Body Sensor Networks

Wireless Body Sensor Networks (BSNs) [6] are used in health care monitoring and assisted living applications to connect sensors that are placed on the human body and measure physical parameters of a person. Sensors commonly used in BSNs are pulse monitors for heart rate, accelerometers and gyroscopes for movements, pulse oxymeters for blood oxygen level, spirometers for the amount of air inspired and expired by the lungs.

BSN architectures are usually composed of a coordinator node and sensor nodes connected in a star topology. Sensor nodes transmit raw or interpreted data to the coordinator node, which may further analyze it. In remote monitoring applications the BSN is connected through a gateway and a wide area network to remote locations, from where doctors or caregivers can access the patients' data. In some application scenarios, e.g. localization and safety, a BSN might also interact with sensor nodes placed in its surrounding environment.

BSNs usually must satisfy strict requirements in terms of:

- reliability especially for applications monitoring vital parameters
- privacy and security to ensure that only authorized people, e.g. relatives and caregivers, can access information on personal health or activities
- latency especially in life emergency scenarios
- low power consumption to maximize battery lifetime
- wearability to allow patients carry sensor nodes in their daily life
- extensibility to other sensors and services, e.g. when new health care needs arise
- provisioning of service across locations to support continuous monitoring

These requirements make the design of BSNs challenging especially because BSNs have limited resources in terms of memory, computing power and energy.

Despite their great potential, currently BSNs are still designed in a very inefficient manner due to lack of proper abstractions for interoperability and modularity as well as the lack of methodologies and tools for analysis of implementation tradeoffs. Moreover, there several open technical challenges in security, classification algorithms, multi-sensor data fusion, low-power protocols, quality of service.

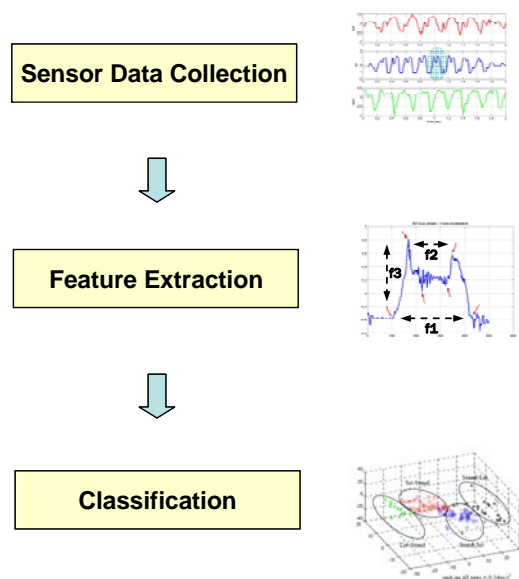


Figure 1: Activity Recognition Systems

A common application of BSNs is the recognition of physical activities or health conditions [6]. Activity recognition systems are usually based on feature extraction and classification algorithms to interpret sensor data and identify patterns of behaviors [7], as shown in Figure 1. Besides the problem of selecting or defining the most suitable classification algorithms for an application, in the design of a BSN application there are also significant implementation challenges, such as the efficient allocation of functional tasks among nodes, due to their limited amount of resources.

What is SPINE?

SPINE is a framework developed in the TinyOS environment (node side) and in Java language (server side) to support developers of assisted living applications through libraries of functions and protocols that can be used to shorten development time.

SPINE is developed as an Open Source project to establish a broad community of users and developers that will contribute to extend the framework with new capabilities and applications.

The SPINE Community – the Open Source Project

SPINE is distributed under LGPL (Lesser GNU Public License) [5]. LGPL enables full exploitation of SPINE, even in a business environment, while enforcing the constraint that any modification of SPINE source code and any derivative work be returned to the community under the LGPL license itself. No restrictions, instead, are put on applications and other categories of software that uses SPINE. In particular LGPL assures right to:

- make and distribute copies of SPINE
- have access to the software source code
- make improvements to the program
- incorporate SPINE into a proprietary program

The LGPL mandates also some duties such as:

- not to keep modifications private
- not to change the license of SPINE and its modifications

The SPINE project is supported by a website [1] where users can find release code and documentation, report possible bugs and browse a collection of useful links maintained by the SPINE team. Moreover, two mailing lists are available to developers for discussing technical issues or just for staying tuned about the project, e.g. to be informed about new releases. SPINE welcomes contributions of the Open Source community that can be given under different forms: simply making publicly known the use of SPINE, reporting and fixing bugs and documentation, replying and giving support to less-experienced users on the mailing list, contributing with new add-ons and software modules.

The Architectural model

SPINE relies on a BSN architecture with star topology including one or more sensor nodes and a BSN coordinator node. The coordinator typically manages the BSN, collects and possibly analyzes the data received from the sensor nodes, and acts as a gateway to connect the BSN with wide area networks for remote data access. Figure 2 visualizes a typical architecture with star topology. The first release of SPINE (SPINE 1.0) supports only BSNs with star topologies. However, it is envisioned that future releases will be extended to other topologies and architectures including sensor nodes placed in the surrounding environments and interacting with the BSN.

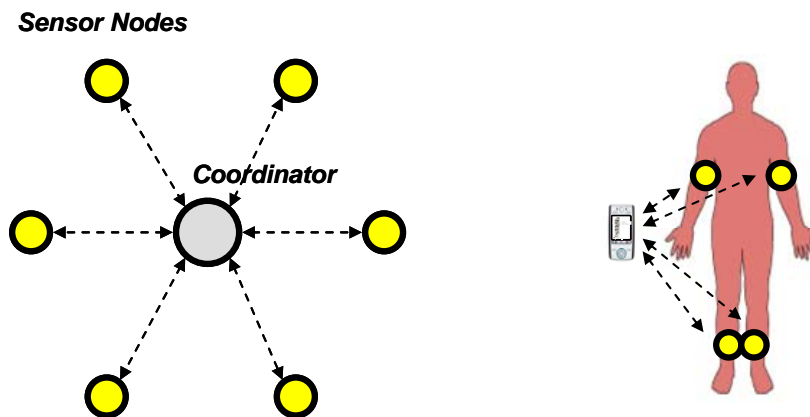


Figure 2: SPINE 1.0 Network Architecture

SPINE has two main SW components: one on the sensor nodes and the other on the BSN coordinator.

The sensor node component, designed in TinyOS environment and written in nesC [2], includes several utilities for signal processing such as data storage buffers, mathematical function libraries and common feature extractors used in signal processing. Furthermore, it includes also an over-the-air communication protocol to transfer data with the coordinator.

The coordinator component consists of a Java-based interface that an application running on the gateway itself or on a remote server can use to manage the sensor nodes or make service requests. SPINE provides a lightweight Java API that is easily portable to devices of various capabilities that can be used as gateway, such as a PC or a mobile phone.

The functional components of the SPINE architecture are visualized in Figure 3.

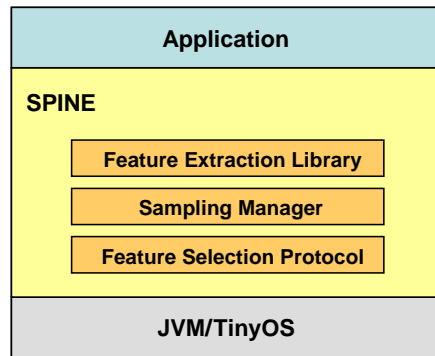


Fig. 3: SPINE 1.0 Functional Architecture

SPINE provides a flexible framework to support developers in designing distributed signal processing algorithms and exploiting implementation tradeoffs. In particular, it supports both centralized and distributed implementations and therefore offers developers the flexibility to select the implementation architecture that is most suitable for each application.

Centralized architectures maximize the amount of functions executed on the coordinator (e.g. an access gateway or a mobile phone). In a fully centralized approach, sensor nodes typically transmit the raw sensor data to the coordinator that extracts features and executes classification algorithms.

In distributed architectures also sensor nodes perform some data processing. For example, sensor nodes may compute features locally and transmit them to the coordinator. Computing features locally and sending the results instead of transmitting the raw sensor data often allows more efficient utilization of the wireless bandwidth and relevant saving of the energy of the nodes. SPINE supports distributed architectures also through a Feature Selection Protocol (FSP) for the communication between the coordinator and the sensor nodes. Using FSP the coordinator can select the features that each sensor node should compute and the sensor nodes can send these features back to the coordinator. In addition to requesting a node to compute specific features locally, FSP also allows the coordinator to specify:

- the interval over which a feature is to be calculated
- the frequency with which the nodes should send the data to the gateway, e.g. at regular intervals, upon request, or when the values reach a specified threshold.

The first release of SPINE has been used for the design of a posture recognition prototype, as described below. Hence, the libraries in SPINE 1.0 include only features relevant to the analysis of data from accelerometers and gyroscopes, as well as on-mote implementation of code that interfaces accelerometer and gyroscope sensors with internal

buffers. However, the framework is not intended to be limited to applications using accelerometer and gyroscope sensors. Instead, the architecture of the framework is general enough to allow future extensions with new libraries of features and interfaces for other types of sensors.

Why to use SPINE?

The SPINE Framework simplifies the design of assisted living applications based on Body Sensor Networks through APIs and libraries of functions and protocols.

Open Source. SPINE is an open-source project that involves the contributions and collaborations of the user community. This user-driven approach allows both users and developers to contribute with suggestions and new code.

Interoperability through APIs. SPINE provides local and remote applications with lightweight Java APIs that are easily portable to devices of various capabilities, such as PCs or mobile phones, used as BSN coordinator.

Higher abstractions. SPINE simplifies the task of application developers by raising the level of abstraction. The layer defined by the SPINE libraries allows designers to focus on application-specific issues and program at a higher level of abstraction than TinyOS.

Distributed implementations of classification algorithms. SPINE simplifies the development of assisted living applications that require complex signal processing algorithms and classifiers. SPINE supports distributed implementation of classification algorithms to reduce the amount of data to be transmitted and save energy.

Reusability. The service discovery function allows the coordinator to recognize the functions of sensor nodes that have already been configured. This flexibility allows the same node to be reused in many different application scenarios without reconfiguration of the embedded code.

Example Application: Posture Recognition

Telecom Italia and WSN Lab have developed a prototype of a posture recognition system based on SPINE [4]. The demonstration prototype, whose components are visualized in Figure 3, implements an application that recognizes the posture (e.g. lying prone or sitting) or the movement (e.g. walking or running) of a person. In addition to recognizing selected postures and movements at any given time, the application sends an alarm when a critical situation is detected (e.g. if the system detects that the monitored person has fallen).

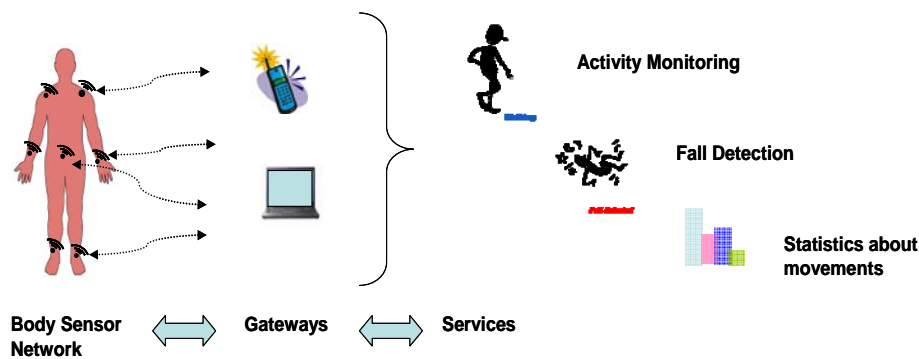


Fig. 3: Posture Recognition system

The hardware platform is based on Tmote Sky [3] motes and a board including a 3-axis accelerometers and two 2-axis gyroscopes [4]. The experimental setup includes several wireless sensor nodes placed at various places of the human body and a PC that collects the data from the motes and executes the application.

Future directions

The SPINE team is planning several extensions of the framework and expects lots of contributions from the research community. Possible extensions include:

- support for new sensors in addition to accelerometers and gyroscopes. Due to the modularity of the SPINE framework, new sensors may be added and supported by the framework itself.
- support for implementations with classification functions executed on the sensor nodes
- definition and implementation of a platform independent abstraction layer that facilitates porting SPINE from TinyOS to other design environments
- support for multi-sensor data fusion
- development of new applications based on the data received by the sensors

The SPINE team welcomes any contribution in those directions as well as any innovative idea about using and enhancing SPINE.

References

- [1] SPINE website, <http://spine.tilab.com>
- [2] TinyOS website, <http://www.tinyos.net>
- [3] MoteIV website: <http://www.moteiv.com>
- [4] R. Gravina, A. Guerrieri, S. Iyengar, F. Tempia Bonda, R. Giannantonio, F.L. Bellifemine, T. Pering, M. Sgroi, G. Fortino and A. Sangiovanni-Vincentelli, "Demo: SPINE (Signal Processing in Node Environment) framework for healthcare monitoring applications in Body Sensor Networks", Proc. of the 5th European conference on Wireless Sensor Networks 2008 (EWSN'08), Bologna, Italy, Jan 30 – Feb 1, 2008
- [5] LGPL license, <http://www.opensource.org/licenses/lgpl-license.php>
- [6] Proceedings of the International Workshop on Body Sensor networks, BSN 2004-2007.
- [7] R. O. Duda, P. E. Hart, D. G. Stork, "Pattern Classification", 2nd ed., Wiley Interscience.

Contacts

- Roberta Giannantonio, roberta.giannantonio@telecomitalia.it
- Fabio Bellifemine, fabioluigi.bellifemine@telecomitalia.it
- Marco Sgroi, marco.sgroi@wsnlabberkeley.com