Human Aware Superorganisms

Nicola Bicocchi
Dipartimento di Ingegneria
Enzo Ferrari
Università di Modena e Reggio Emilia
nicola.bicocchi@unimore.it

Damiano Fontana
Dipartimento di Scienze e Metodi dell’Ingegneria
Università di Modena e Reggio Emilia
damiano.fontana@unimore.it

Franco Zambonelli
Dipartimento di Scienze e Metodi dell’Ingegneria
Università di Modena e Reggio Emilia
franco.zambonelli@unimore.it

Abstract
Massive networks of wearable devices have recently become a key scenario for pattern recognition technologies. Applications range from implicit human-machine interactions, to autonomous monitoring of user habits and activities. This paper presents a framework providing developers with tools to orchestrate the continuous process of collecting and classifying data streams in aware-systems. It supports service oriented, reconfigurable components and provides a solid background to put at joint work specification- and data-driven approaches. It also provides an innovative meta-classification scheme allowing to implement applications by editing a simple state automata. Experimental results suggest that the approach could be integrated in a number of applications for: (i) improving energy efficiency, (ii) improving classification accuracy and (iii) improving software engineering of aware systems.

ACM Classification Keywords
D.2.8 [Software Engineering]: Metrics—complexity measures, performance measures; D.2.2 [Software Engineering]: Design Tools and Techniques; F.1.1 [Computation by Abstract Devices]: Models of Computation
Introduction
The widespread adoption of sensor networks, actuators and computational resources capable of interacting with people is transforming urban environments as well as domestic spaces. People are increasingly exposed to machines capable of understanding their health parameters, behaviours, activities, daily routines and acting accordingly. This trend could be considered as the actual implementation of what Mark Weiser envisioned three decades ago.

However, the design and deployment of such services in future urban scenarios is still not trivial. In fact, it shakes current approaches until their foundations rooted in top-down design. Designing with a top-down approach means that all the requirements of a software architecture have to be taken in account a priori; systems engineered in this way have a predictable and measurable behaviour but are not well suited to cope with dynamic execution contexts. Contrarily, bottom-up design delivers robust systems and can be fruitfully used in pervasive environments. However, predicting and controlling their behaviour by design is not an easy task. Short coming urban scenarios call for a balanced trade off between the two approaches.

Awareness appears to be one of the key driver to guide this tradeoff. In fact, while top-down design is still required given our current technological level, the awareness of operating conditions could be used to provide systems with bottom-up features proved to be essential in dynamic, interconnected, heterogeneous environments. Awareness modules are called for knowledge collection, organisation, and reasoning. Given their conception, and their task of providing adaptation capabilities to top-down artefacts called to operate open-ended environments, awareness modules themselves are needed to be rooted around adaptation concepts. In this context, this work describes an innovative awareness framework - rooted around the concept of dynamic service reconfiguration - able to gather data from a number of different sources and classify them using general-purpose algorithms.

The rest of the paper is organised as follows. Section Reconfigurable components for knowledge collection and understanding describes motivations and challenges behind this work. Section Architecture presents the global architecture of the awareness framework we are developing. Finally, Section Related Work discusses related work and Section Conclusion concludes the paper.

Reconfigurable components for knowledge collection and understanding
Recent works show that it is possible to recognise personal and social behaviours, urban and natural events by analysing users activities both in physical and digital domains. However, they stress the relevance of mapping the intrinsic dynamism of real-world with engineering mechanisms pointing out three main challenges: (i) different approaches and algorithms are needed to effectively deal with the many facets of the real-world situations; (ii) for the same classification problem, it is still not feasible to deal with every possible operative context with a single technique. For example, in continuous sensing it has been shown the relevance of modifying classification parameters at runtime; (iii) classification accuracy is inversely proportional to the number of treated classes [11] [5].

Thus, modern architectures stemming from these requirements should exhibit a certain degree dynamism and flexibility and make applications able to autonomously
interoperate and reconfigure. Service oriented and dynamically reconfigurable components have been proposed [8].

Despite reconfigurable components could provide a higher degree of flexibility to applications, the problem of driving reconfiguration processes is still open. In fact, every system capable of changing its internal structure or parameters must take decisions according to its environment. These decisions could be taken on different basis and could be the output of different reasoning processes. We decided to drive internal reconfigurations with state-based automata because of their simplicity and generality. Each status is associated with a set of sensors specifically configured, while transitions and consequent reconfigurations are triggered by specific conditions. Moreover, such models can be monitored, enriched, updated at run time, and even shared among different applications and enriched over time by a community of developers. Engineers, in this way, could put at work a number of classifiers agnostic about the domain of the problem and dynamically reconfigure them using predefined set of rules.

Summarising, the key idea behind this work consists in using service oriented, reconfigurable components to fill the gap between specification- and data-driven models. In fact, while data-driven models can be used to classify data streams, specification-driven models, such as automata, can drive the reconfiguration of the architecture. This meta-classification scheme leads to three main advantages:

• Improve energy efficiency. In particular, for each specific situation the less energy demanding sensors and classifiers could be used. For example, it is possible to roughly recognise the vehicle used by a user with either GPS or accelerometer or microphone. An energy constrained system could constantly monitor its energy consumption and select the most appropriate trade-off. Furthermore, classification algorithms could be parameterized to be less precise in favour of a minor computational complexity. Finally, in the near future, the number of sensors available will rapidly increase and it is up to the awareness framework to select the most suitable ones to recognize the current context, by improving accuracy.

• Improve classification accuracy. Instead of having a single classifier looking for a wide spectrum of classes, several classifiers could be used to recognise only specific situations. These could be activated on-demand by taking in consideration the current context awareness of the framework. Specific classifiers are less prone to over fitting, are thus more general and could be used without expensive re trainings on other problems. Furthermore, as the number of situations (i.e., classes) increases combining simpler classifiers often gives better accuracy.

• Improve software engineering. Organising the framework around the idea of reconfigurable components (i.e., sensors, classifiers, meta classification schemes) leads to modularity and composability of an innovative software ecosystem. On one side we will provide the community with a set of frequently used sensors and classifiers. On the other, users will be able to deploy their own both by: (i) using libraries already included in the framework or (ii) developing their own algorithms.
Figure 1: The framework architecture is structured around three layers, namely sensor, classifier and awareness layer.

Architecture

This section details an awareness architecture supporting both individual and collective awareness incorporating concepts described above. Specifically, its goal is to provide general-purpose awareness that could be used as a starting point for many diverse applications. Developers are only required to select the needed modules, define the topology of data flows and specify their reconfiguration strategies as depicted in Figure 1.

The architecture is structured around three layers, namely sensor, classifier and awareness layer. Each layer can host multiple modules connected each other via application-definable topologies. The data flow from sensors through the whole architecture by means of in-memory queues enabling modules decoupling and many-to-many asynchronous communications. Each layer can host multiple modules.

The sensor layer hosts modules that are in charge of retrieving raw data from physical sensors and preprocess them. An example could be a module acquiring images from a camera and cropping and resizing them. Other examples could be modules acquiring facts from social networks, such as Twitter, Facebook or Foursquare. At the time of writing, we have already implemented modules for reading data from Android devices.

The classification layer hosts modules that consume data coming from the sensor layer and classify (i.e., generate semantically richer information) them. An example could be a module able to classify the activity performed by a user by processing accelerometer data. At the time of writing, we have implemented modules for classifying user activity, location, speed, vehicle used on the basis on common smartphone sensors. It is worth noting that our goal is to build a general-purpose awareness framework that could be used as common basis for both research and application development, not to solve every possible classification problem. Specific applications will need their own modules to be developed.

The awareness layer hosts modules consuming labels produced in the classification layer and feeding external applications with situational information. These modules might have different goals depending on the application. However, they could be divided into two main classes.

The former comprises modules delegated to sensor fusion processes. These modules receive labels, eventually conflicting, coming from multiple classification modules and apply algorithms to achieve higher semantical levels. For example, commonsense knowledge has been recently proposed [2] and could be integrated at this level.

The latter, instead, is related with the capability of the
framework of monitoring and controlling itself. In a sense, the awareness layer could be the key of building a self-aware awareness module. For example, it would be possible to integrate within this level modules observing the internal status of the framework and activating different classifiers and sensors depending on operating conditions. This capability could be used to achieve both improved classification accuracies and reduced power consumption levels by continuously selecting to most suitable classifiers and sensors.

From an engineering viewpoint, the architecture is implemented on the top of industrial-level Java technologies. Each module is actually an OSGi component able to meet the requirements mentioned in Section [1].

On top of OSGi, we have an iPOJO layer. iPOJO is a container-based framework handling the lifecycle of Plain Old Java Objects (POJOs) and supporting management facilities like dynamic dependency handling, component reconfiguration, component factory, and introspection. Moreover, the iPOJO container is easily extensible and allows pluggable handlers, typically for the management of non-functional aspects.

On top of the iPOJO framework we build the support for the staged and layered architecture by making use of Apache Camel. This framework provides components with the capability of asynchronously processing data streams and communicate through in-memory queues. These queues allow modules belonging to different layers to continuously communicate each other with minimum hardware requirements. Considering that pattern classification and analysis has a central role in situation awareness, we wrapped well-know data manipulation libraries within the framework such as Weka and jMIR.

Related Work
In the last years, researchers prototyped sensing systems able to acquire detailed situational information from data streams [7, 6, 3] using both specification-driven (e.g., logic ontologies, logic programming, fuzzy logic) or data-driven approaches (e.g., support vector machine, decision trees, neural networks). The most prominent works have been surveyed in [11] and [5].

However, the majority of these systems lacks in generality and addresses specific recognition problems, by making use of a pre-defined set of sensors. Few of them tried to make use of both the approaches designing frameworks that are resource efficient and robust in a large plethora of situations. For example, in [8] authors make use of processing pipelines on various sensors to show how processing pipelines and dynamic reconfiguration could be used in continuous sensing. The framework doesn’t focus on a generic approach enabling re-configuration and runtime adaptation. In [9], [4] and [10] authors propose to optimise the sensing process in terms of power saving. In particular, [9] exploits a specification-driven reasoning technique to learn relationship among context attributes to optimise the internal logic of an awareness framework. However, like previous works, these frameworks focus only the optimisation of energy consumption in continuous sensing.

To the best of our knowledge, our framework represents a first attempt to integrate the two approaches in a general way. Thanks to its general and self-aware architecture, it is able to implement and make full use of all the strategies and optimisations proposed in the previous works and it is able to deal with more complex scenarios that require flexibility and adaptability as foundational basis.
Conclusion
In this paper we proposed an innovative awareness framework suitable for pervasive scenarios. It has been conceived around the concept on reconfiguration and built using industrial-level tools. Its modular and portable architecture makes it suitable for a number of different applications in which awareness either individual or collective could be used to trigger adaptation processes.

Acknowledgments
Work supported by the ASCENS project (EU FP7-FET, Contract No. 257414).

References