Proactive Agriculture: An Integrated Framework for Developing Distributed Hybrid Systems

Christos Goumopoulos[1], Achilles Kameas[1,2], and Brendan O'Flynn[3]

[1] Research Academic Computer Technology Institute, Distributed Ambient Information Systems Group, N. Kazantzaki, 26500 Rio Patras, Hellas
{goumop, kameas}@cti.gr
[2] Hellenic Open University, 23, Sahtouri Str, Patras, Hellas
kameas@eap.gr
[3] Tyndall National Institute, Lee Maltings, Prospect Row, Cork, Ireland
boflynn@tyndall.ie

Abstract. In this paper we discuss research work that enables the development of hybrid systems consisting of communicating plants and artefacts and we investigate methods of creating "interfaces" between artefacts and plants in order to enable people to form mixed, interacting communities. Our research objective is to develop hardware and software components that enable a seamless interaction between plants and artefacts in scenarios ranging from domestic plant care to precision agriculture. This paper deals with the approach that we follow for the development of such hybrid systems and discusses both hardware and software architectural aspects, with a special focus on describing the modular platform for wireless sensor network implemented and the distributed context management process followed. The latter imposes a proactive computing model by looping sensor data with actuators through a decision-making layer. The deployment of the system in a precision agriculture application is also presented.

1 Introduction

Currently, there are few discussions on the integration of biological elements of the real (natural) environment into pervasive computing applications [1, 2, 3]. In this paper, we present our research efforts to create digital interfaces to nature, in particular to selected species of plants. Our approach goes beyond the use of sensor networks for environmental monitoring [4] by emphasizing the development of a system architecture that incorporates the plants and associated computation as an integral part of the system, and allows the interaction of plants and artefacts in the form of synergistic and scaleable mixed societies. The ambient intelligence technology is used to encompass plant requirements, by establishing a three-way interaction between plants, people and objects. This approach enables the development of hybrid systems consisting of communicating plants and artefacts in scenarios ranging from domestic plant care to precision agriculture.

Precision agriculture is an agricultural concept relying on the existence of in-field variability across an array of cropping systems [5]. Thanks to developments in the field of wireless sensor networks as well as miniaturization of sensor systems, new trends have emerged in the area of precision agriculture. Wireless networks allow the deployment of sensing systems and actuation mechanisms at a much finer level of granularity, and in a more automated implementation than has been possible before.

At present the information gathered by sensor networks deployed in a field are mainly used for monitoring and reporting on the status of the crops [1, 6, 7]. However, agricultural environments make a good candidate for using proactive-computing approaches for applications which require a faster than human response time or which requires precise, time-consuming optimization.

In this paper we describe an integrated framework for developing hybrid systems. A hybrid system consists of various entities including software components, hardware components (sensors, actuators and controllers), datastores (knowledge base, raw data), biological elements (plants) and environmental context. By positioning sensors around particular plants (proximal remote sensing) the delivered technology is capable of reacting (via actuators) to stimuli (perceived via sensor networks), aiming to maintain a coherent plant state and support efficient plant growth. Our research has focused on the provision for proactive applications by deploying sensor networks and connecting sensor data with actuators through a decision-making layer which attempts also to resolve aspects of data uncertainty.

The remainder of the paper is organized as follows. Section 2 presents the rationale behind mixed societies of communicating plants and artefacts and describes its basic elements. The pillars of the integrated framework defined for developing hybrid systems are discussed in the next section. A layered modular architecture of the system is proposed to enable system flexibility and extensibility, while the need to cope with uncertainty of the data is also explored. A detailed example application from the precision agriculture domain is given in section 4. Related work is discussed in section 5 and finally the conclusions of our work are presented.

2 Mixed Societies of Communicating Plants and Artefacts

The communicating plant concept [8] fits well within the vision of Ambient Intelligence where the virtual (computing) space will be seamlessly integrated with our physical environment. By regarding plants as virtual “components”, which can communicate with other artefacts in the digital space we can shape mixed societies of them. From an engineering perspective, a mixed society of communicating plants and artefacts can be regarded as a multi-layered, hierarchical distributed system, which will globally manage the resources of the society, its function(s) and its interaction with the environment.

An ePlant component, in particular, may represent the digital self either of a specific plant or a group of plants (a group may be defined in terms of a specific plant species or in terms of plant vicinity, a number of plants in a geographical region) and is responsible for the back-end computation with respect to the sensor network computing. Through a software layer (middleware), ePlant communicates with the sensor network, implements a decision-making scheme for assessing plant states and alarms and handles the interaction with other eEntities.

eEntities that represent domain-specific objects with the capabilities of information processing and exchange are also called artefacts. These artefacts have the capability of communicating with other artefacts based on local networks, as well as accessing or exchanging information at a distance via global networks. In our case artefacts may represent expressive devices (speakers, displays, etc.), resource-providing devices (e.g. lamps, irrigation/fertilization/shading system) or any other everyday object (e.g., cell phone, camera).

Sensor systems range from standalone sensor devices to wireless sensor networks monitoring micro-climates in a crop field. Standalone sensor devices may be shared among a number of ePlants so that the context needs to be determined. On the other hand, the actuator systems will allow the plant to influence the environment that it resides in.

3 Integrated Framework for Developing Hybrid Systems

3.1 Wireless Sensor/Actuator Network Hardware Platform

The hardware platform used is the 25mm mote developed at Tyndall [9, 10]. The Tyndall25 mote is a miniaturised wireless sensor platform that addresses the issues of reconfigurability, power-efficiency and size which are desirable and necessary characteristics for a wireless sensor network platform (Figure 1).

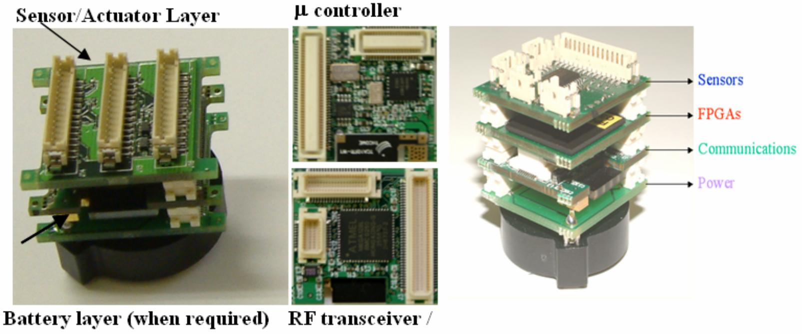


Fig. 1. Tyndall25 modular platform

The hardware platform is analogous to a Lego™-like 25mm x 25mm stackable system. Layers can be combined in an innovative plug and play fashion and include communication, processing, sensing and power supply layers. The communication layer is comprised of a microcontroller, RF transceiver and integrated antenna [11]. The module contains an Atmel ATMega128L microcontroller and a Nordic VLSI 2401 RF transceiver both of which are combined on a single layer. The microcontroller is equipped with 128KB in-system flash memory and can be programmed to handle analogue to digital conversion of sensor data and the communication networking protocols necessary for interfacing with the RF transceiver to achieve communication with other motes. Stacked upon this RF microcontroller layer is the custom sensor/actuator interface layer. On the software side, the microcontroller runs a tailored version of TinyOS [12], an optimised operating system that allows fast configuration of the sensor nodes implementing active message protocol (AMP) [13]. The power layer may include batteries or other energy supply or power harvesting mechanisms. Finally, an optional FPGA layer can be integrated into the system whenever high-speed Digital Signal Processing is required.

The sensor/actuator interface layer allows any combination of eight different sensors or actuators to be connected to the 25mm module. In the agricultural application domain, the sensor interface portion allows soil moisture probes, thermistors and ethylene sensors to be interrogated by the controlling software running on the mote.

3.2 Software System Architecture

Figure 2 illustrates a general overview of the system architecture. In the lower layer various sensors/actuators that can form collectively sensor networks, provide the raw data. In the drivers' layer, a specific driver is designed and implemented for each sensor device/network implementing the communication protocol with the hardware.

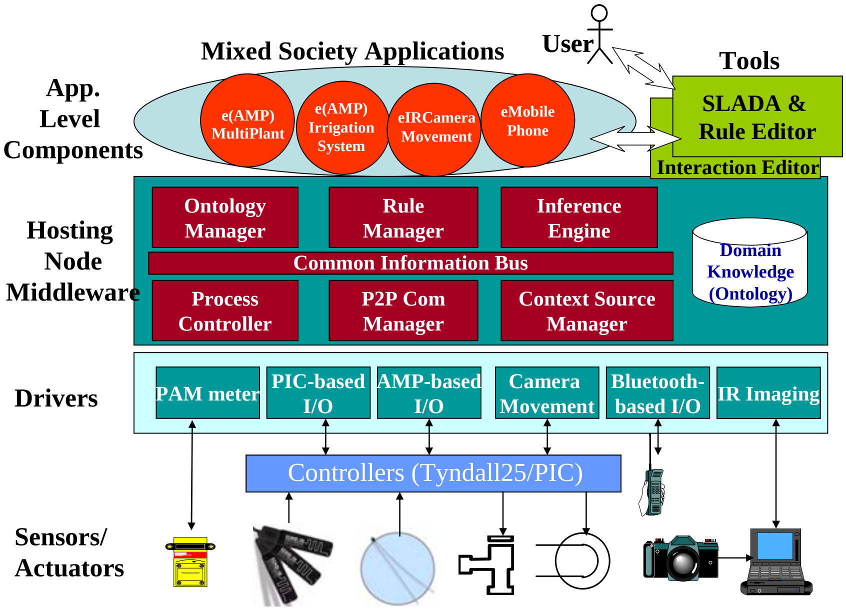


Fig. 2. Integrated System Architecture

The role of a hosting node in the distributed management system reflected by mixed societies is mainly to act as a gateway between the wireless sensor network and domain-specific applications. In that sense the hosting node software represents a middleware layer that supports interaction with other nodes, back-end monitoring and performing of control/management services. In particular, the *P2P Communication*

Module is responsible for application-level communication and interaction between the various hosting nodes. The *Process Controller* is the coordinator module and the main function of this module is to monitor and execute the reaction rules defined by the supported applications. These rules define how and when the infrastructure should react to changes in the environment. The *Context Source Manager* is responsible for dealing with contextual information and in particular with context gathering, inference, aggregation, history and monitoring. It handles the runtime storage of node's context values, reflecting both the hardware environment (sensors/actuators) at each particular moment (primitive context), and properties that are evaluated based on sensory data and P2P communicated data (composite context). The *Inference Engine* is responsible for the evaluation of composite properties (e.g., state assessment) according to a set of rules, which are obtained from plant science research; its implementation is currently based on the Jess (Java Expert System Shell) environment [14]. The rule management can be separated from the evaluation logic by using a high-level rule language and a translator that translates high-level rule specifications to XML that can be exploited then by the evaluation logic. This management is the responsibility of the *Rule Manager* module. To support the development of applications we have defined an *Ontology* that encodes domain knowledge (description of eEntities, sensors, actuators, parameters, states; and application logic as aggregation, inferring and action rules). Thus an *Ontology Manager* module has been defined for the manipulation of the knowledge represented into the ontology and to provide the other modules of the system with parts of this knowledge with a level of abstraction. Details on the organisation of the ontology can be found in [15, 16].

The application level components hold the logic that specifies the conditions under which actions are to be triggered. The conditions are specified in terms of correlation of events. Events are specified up front and types of events are defined in the ontology. The Inference Engine subscribes to events (specified in applications logic) and the Context Manager generates events and notifies the Inference Engine when the subscribed events occur. When the conditions hold, the Process Controller performs the specified actions, which could consist of, e.g., sending messages through the P2P Communication Manager and/or request an external service (e.g., toggling irrigation).

When building context-aware applications in pervasive computing environments one faces the difficult problem of dealing with uncertain context information. Quality indicators can be specified so that the end-user can make judgements on the confidence level that the information entails. We model uncertainty in our environment by enhancing the rules with *certainty/confidence factors* (*CF*) about how certain the conclusions drawn from the rules may be. Certainty factors are guesses by an expert about the relevance of evidence. We are using the scale -1 to 1 and we assume the following interpretation: as the CF approaches 1 the evidence is stronger for a hypothesis; as the CF approaches -1 the confidence against the hypothesis gets stronger; a CF around 0 indicates that there is little evidence either for or against the hypothesis.

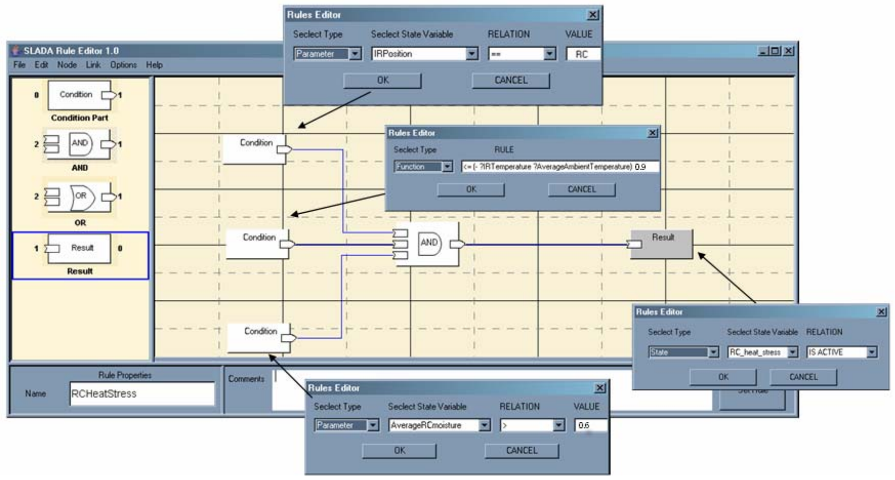
Certainty factors may apply both to facts and to rules, or rather to the conclusion(s) of rules. Conditions for rules are formed by the logical "*and*" and "*or*" of a number of facts. The certainty factors associated with each condition are combined to produce a certainty factor for the whole condition. For two conditions *P1* and *P2* it holds that:

CF(P1 and P2) = min(CF(P1), CF(P2)) and *CF(P1 or P2) = max(CF(P1), CF(P2))*. The combined CF of the condition is then multiplied by the CF of the rule to get the CF of the conclusion. The CF scheme has been implemented through FuzzyJ ToolKit [17], a library that can be integrated with JESS for the provision of fuzzy reasoning.

3.3 Rule Editor

The *Rule Editor* works in cooperation with the *Supervisor Logic and Data Acquisition Tool* (*SLADA*); the former manages dynamically the rules taking part in the decision-making process while the later can be used to view knowledge represented into the Ontology and monitor plant/environmental parameters. The Editor provides a Graphical Design Interface for managing rules, based on a user friendly node connection model. The advantage of this approach is that rules will be changed dynamically in a high-level manner without disturbing the operation of the rest of the system.

Figure 3 shows the design of the Heat stress calculation rule for the ePlant. The rule consists of three conditions combined with a logical AND gate. The first condition checks the applicability of a specific area (Right Center) of the field layout for which we need to evaluate the heat stress state. The second condition checks whether the absolute difference between environmental and average temperature, in that area, is below 0.9° C. The third condition checks whether the average moisture in the specific area is over 60%. The rule, as designed, states that when all three conditions are met then the heat stress state of the RC area must be set to active.



Seclect Type: Parameter | Seclect State Variable: IRPosition | RELATION: == | VALUE: RC

Seclect Type: Function | RULE: (<= (- ?IRTemperature ?AverageAmbientTemperature) 0.9

Seclect Type: State | Seclect State Variable: RC_heat_stress | RELATION: IS ACTIVE

Seclect Type: Parameter | Seclect State Variable: AverageRCmoisture | RELATION: > | VALUE: 0.6

Rule Properties
Name: RCHeatStress
Comments

Fig. 3. Editing the Heat Stress rule of the ePlant

Using a rule editor for defining application business rules emphasizes system flexibility and run-time adaptability. In that sense, our system architecture can be regarded as a reflective architecture that can be adapted dynamically to new requirements. The decision-making rules can be configured by domain experts external to the execution of the system. End-users may change the rules without writing new code. This can reduce time-to-production of new ideas and domain-specific research results to a few minutes. Therefore, the power to customize the system is placed in the hands of those who have the knowledge to do it effectively.

4 Precision Agriculture Example Application

The example application described in this section is composed of a strawberry plant where the plant is controlling irrigation and supplementary light. Irrigation is applied according to the specific requirements of the plants in different parts of the crop array, thus illustrating the precision delivery of agricultural inputs.

4.1 Plant/Environmental Signals

The plant/environmental signals explored for the application development are: Plants' leaf Temperature (PT), Chlorophyll Fluorescence (CF), Ambient Temperature (AT), Ambient Light (AL) and Soil Moisture (SM). For each signal a different type of sensor is required. Table 1 summarizes the signals and the corresponding sensors used as well as the associated knowledge that will be stored in the ontology for supporting the monitoring and decision-making process.

Table 1. Plant/Environmental Signals and Sensors

Signal	**Measuring Sensor**	**State Assessment**	**Possible Actions**
CF	PAM meter[1]	photo-stress; photosynthetic efficiency	light control; estimate/adapt threshold values for providing input resources
PT	thermistor	drought stress; heat stress	irrigation/misting
SM	Probe EC-10[2]	drought stress	irrigation
AT	thermistor		
AL	PAR meter[3]	photo-stress	light control

Heat stress can occur independently of water stress when the ambient environmental temperature gets very high and plant transpiration cannot maintain leaf cooling. Therefore, if the plant has adequate water (determined by the SM probe) but the plant temperature is high this means that it is heat stressed and requires misting to cool it. However, if the temperature is high and the moisture content low, then pot irrigation is required. The CF and AL parameters are used to determine photo-oxidative stress and adjust supplementary light.

---

[1] Junior PAM, Gademann Instruments: http://www.gademann.com/

[2] ECHO probe model EC-10: http://www.ech2o.com/specs.html

[3] Skye SKP215 Quantum Sensor: http://www.alliance-technologies.net

4.2 Prototype Setup and Evaluation

The prototype setup consists of an array of 96 plants placed in a glasshouse, arranged in an array of 12 by 8. The setup consists of 4 different zones: Left-Edge (LE), Right-Edge (RE), Left-Center (LC), Right-Center (RC) and also one zone specified for misting which coincides with the RC zone. The setup integrates the thermistors and soil moisture probes into one system that can irrigate when required and also determine when to stop the irrigation. This deployment takes into account differences in the location of the plants in the overall area and will allow for independent irrigation of edge or centre zone plants as required. Each zone can be controlled using individual solenoids. Misting can be applied only to the RC due to infrastructure limitations.

A total of 10 Tyndall25 motes are required to implement the above prototype: 8 modules are used for connecting the various sensors, each one ‘supervising’ the sensors in the neighbourhood of an array of 3 by 4 plants; 1 module is sensorless and is used as a communication relay with the hosting node; and 1 module is used for controlling the irrigation system. The nodes are housed tightly in IP-67 rated water-proof packaging to withstand the harsh conditions of the field. The sensor nodes are manually placed however the mapping to the zones is administered at a higher level in the hosting node (ePlant), as part of its description. For energy-efficiency and power consumption considerations, the sensor nodes are reporting data once per five minutes. The data collected by the sensor nodes is gathered by the hosting node, for local processing and logging. Interaction then is possible between the hosting node and other devices for managing the delivery of agricultural input according to an adaptable decision-making scheme.

The application business logic is expressed upon a set of plant parameters, plant states and actions to be performed. Table 2 illustrates such variables defined in the ontology of the application.

Table 2. Application business logic variables

Parameters	States	Action Requests
`AmbientAvgTemp`	`“Z”DroughtStress`	`“Z”NeedIrrigation`
`“Z”AvgTemp`	`“Z”HeatStress`	`“Z”NeedMisting`
`“Z”AvgMoisture`		

The “Z” prefix in the name of a variable is substituted by one of the possible zone names of the crop array (LE, RE, LC, RC). For the NeedMisting variable the prefix can be omitted since there is only one zone specified for misting. Two additional parameters must be defined for the prototype to be properly working; the duration of irrigation/misting and an idle time which specifies the amount of time the rules should be disabled, after the action is performed. This is to allow the ecosystem to absorb the changes. The values used for the application were 1 min and 4 hours respectively.

The actual logic of the prototype is captured in a set of rules. Table 3 contains the applicable rules for the RC zone. Rules for evaluating the plant states and actions to be performed are shown. Confidence Factor values are also included. CF values in square brackets are defined by the domain-expert, while in curly brackets by the system Inference Engine. The user for example can specify a policy where actions with confidence below 50% shouldn't be triggered but the user should be notified.

Table 3. Application rules with Confidence Factors shown

Rule	**Body**
RCDrought Stress *[CF=0.8]*	IF RCAvgTemp–AmbientAvgTemp>0.75°C *[CF=0.9]* THEN RCDroughtStress ← TRUE ELSE RCDroughtStress ← FALSE *{CF=0.72}*
RCHeat Stress *[CF=0.9]*	IF RCDroughtStress *{CF=0.72}* AND RCAvgMoisture>60% *[CF=0.9]* *{CF=min(0.72, 0.9)=0.72}* THEN RCHeatStress ← TRUE ELSE RCHeatStress ← FALSE *{CF=0.65}*
RCNeed Irrigation *[CF=1]*	IF RCDroughtStress *{CF=0.72 }* AND NOT RCHeatStress *{CF=0.65 }* *{CF=min(0.72, 0.65)=0.65}* THEN RCNeedIrrigation ← TRUE ELSE RCNeedIrrigation ← FALSE *{CF=0.65}*
Need Misting *[CF=1]*	IF RCDroughtStress *{CF=0.72 }* AND RCHeatStress *{CF=0.65}* *{CF=min(0.72, 0.65)=0.65}* THEN NeedMisting ← TRUE ELSE NeedMisting ← FALSE *{CF=0.65}*

The reliability of the wireless sensor network is of great importance as lost of data may hinder the decision support layer of the system and thus the correct delivery of inputs. There are several measures that have been taken to alleviate this risk. First each sensor node will store each measurement in its local memory and will overwrite it when an acknowledgement is received. In addition the use of sequence numbers in the packets allows the hosting node to detect easily lost packets, if the MAC-layer fails to deliver them after attempting a number of retransmissions.

On the agronomic part of the experiment the instrumentation of the strawberry field with the wireless sensor network and the plant-driven irrigation leads to a notable reduction in water consumption (15-20%) with respect to traditional agricultural practices involving user defined timed irrigation based on rules of thumb. The later was applied in a parallel setup for the same growing period (early development stage) of the crop. The deployment of smart water management on a large farming scale is extremely important given the irrigation needs of the agricultural sector (irrigation uses up to 80% of total water in some regions) and the decreasing availability of water for irrigation.

5 Related Work

Attempts to use environmental sensor networks in order to improve crop cultivation by monitoring and reporting on the status of the field are reported in [1, 7]. These approaches provide decision-support to the user who responds by providing the required treatment. In the same way the approach discussed in [6] uses a centralized architecture to gather data followed by an analysis phase so that a grower becomes able to examine crop conditions in trial-and-error scheme. This is in contrast to our plant-driven distributed management system that imposes a proactive computing model for the crop treatment.

A UbiComp application called PlantCare that takes care of houseplants using a sensor network and a mobile robot are investigated in [18]. The proposed framework in this paper supports both ambient and agricultural applications where in the later case the integration of a large number of sensors and the complexity of the communication and the decision-making processes are the focal points.

MoteWorks^{TM} is a general-purpose software platform for the development of wireless sensor network systems [19]. MoteWorks, utilizes comparable solutions to our framework at the mote network tier level. However, at the middleware tier, the objectives differ and are not comparable, as our approach views applications in the form of cooperating objects in our natural environment with context management and adaptive decision-making requirements. Instead, MoteWorks in order to be generic provides a simple API from the Intra/Internet to the wireless sensor network.

On the hardware platform side comparisons with other similar sensing nodes in this class, namely the Mica2, Mica2Dot and Intel motes, revealed advantages and disadvantages of each [10]. The modular nature and robust connectivity mechanism of the Tyndall mote made it ideal for use in the application domain explored.

6 Conclusions and Future Directions

We have been involved with a facet of precision agriculture that concentrates on plant-driven crop management. By monitoring soil, crop and climate in a field and providing a decision support system, it is possible to deliver treatments, such as irrigation, fertilizer and pesticide application, for specific parts of a field in real time and proactively. We have presented in this paper an integrated framework consisting of hardware, software components and a rule editor that support efficiently the development of distributed hybrid systems.

Moving our research towards to a more autonomous system with self-adaptation and self-learning characteristics, we have been exploring ways of incorporating learning capabilities in the system. Machine-learning algorithms can be used for inducing new rules by analysing logged datasets to determine accurately significant thresholds of plant-based parameters. Finally, regarding the sensor network platform work is underway to implement a version of the current 25mm square form factor transceiver node in a 10mm and 5mm cube form factor.

Acknowledgement

This paper describes research carried in the PLANTS (IST-38900) and ASTRA (IST-29266) projects; the authors wish to thank their fellow researchers in the consortiums.

References

1. Burrell, J., Brooke, T., Beckwith, R.: Vineyard Computing: Sensor Networks in Agricultural Production. IEEE Pervasive Computing **3(1)** (2004) 38–45
2. Bohlen, M., Tan, N.: Garden Variety Pervasive Computing. IEEE Pervasive Computing **3(1)** (2004) 29-34
3. Mainwaring, A., Polastre, J., Szewczyk, R., Culler, D.: Wireless Sensor Networks for Habitat Monitoring. In: Proceedings of the 1st ACM International Workshop on Wireless Sensor Networks and Applications, ACM Press (2002) 88–97
4. Martinez, K., Hart, J., Ong, R.: Environmental Sensor Networks. IEEE Computer **37(8)** (2004) 50-56
5. Koch, B., Khosla, R.: The Role of Precision Agriculture in Cropping Systems. Crop Production **8** (2003) 361-381
6. PhyTech, Phytomonitoring™. http://www.phytech.co.il/introduction.html
7. Zhang, W., Kantor, G., Singh, S.: Integrated Wireless Sensor/Actuator Networks in an Agricultural Application. In: Proceedings of the 2nd ACM International Conference on Embedded Networked Sensor Systems (SenSys), ACM Press (2004) 317
8. Calemis, I., Goumopoulos, C., Kameas, A.: Talking Plant: Integrating Plants Behavior with Ambient Intelligence. In: Proceedings of the 2nd IET International Conference on Intelligent Environments, IET Press (2006) 335-343
9. O'Flynn, B., et al.: The Development of a Novel Miniaturized Modular Platform for Wireless Sensor Networks. In: Proceedings of the 4th International Conference on Information Processing in Sensor Networks (IPSN'05), IEEE Computer Society Press (2005) Article 49
10. Bellis, S.J., et al.: Development of Field Programmable Modular Wireless Sensor Network Nodes for Ambient Systems. Computer Communications J. **28(13)** (2005) 1531-1544
11. O'Flynn, B., et al.: A 3-D Miniaturised Programmable Transceiver. Microelectronics international J. **22(2)** (2005) 8-12
12. TinyOS Community Forum. http://www.tinyos.net.
13. Buonadonna, P., Hill, J., Culler, D.: Active Message Communication for Tiny Network Sensors. UC Berkeley Technical Report (2001). http://www.tinyos.net/papers/ammote.pdf
14. Jess - the Rule Engine for the Java Platform. http://herzberg.ca.sandia.gov/jess/
15. Goumopoulos, C., et al.: The PLANTS System: Enabling Mixed Societies of Communicating Plants and Artefacts. In: Proceedings of the 2nd European Symposium on Ambient Intelligence (EUSAI 2004), LNCS Vol. 3295, Springer-Verlag (2004) 184-195
16. Christopoulou, E., Goumopoulos, C., Kameas, A.: An ontology-based context management and reasoning process for UbiComp applications. In: Proceedings of Joint sOc-EUSAI'2005 conference, ACM Press (2005) 265-270
17. FuzzyJ ToolKit. http://www.iit.nrc.ca/IR_public/fuzzy/fuzzyJToolkit2.html
18. LaMarca A., et al.: PlantCare: An Investigation in Practical Ubiquitous Systems. In: Proceedings of the 4th International Conference on Ubiquitous Computing, LNCS Vol. 2498, Springer-Verlag (2002) 316-332
19. Crossbow Technology Inc.: MoteWorks. http://www.xbow.com/Products/Product_pdf_files/Wireless_pdf/MoteWorks_OEM_Edition.pdf