Resource-Efficient Wireless Monitoring based on Mobile Agent Migration

Kay Smarsly*[a], Kincho H. Law[a], Markus König[b]
[a]Dept. of Civil and Environmental Engineering, Stanford University, Stanford, CA 94305, USA
[b]Dept. of Civil and Environmental Engineering, Ruhr-Univ. Bochum, 44780 Bochum, GERMANY

ABSTRACT

Wireless sensor networks are increasingly adopted in many engineering applications such as environmental and structural monitoring. Having proven to be low-cost, easy to install and accurate, wireless sensor networks serve as a powerful alternative to traditional tethered monitoring systems. However, due to the limited resources of a wireless sensor node, critical problems are the power-consuming transmission of the collected sensor data and the usage of on-board memory of the sensor nodes. This paper presents a new approach towards resource-efficient wireless sensor networks based on a multi-agent paradigm. In order to efficiently use the restricted computing resources, software agents are embedded in the wireless sensor nodes. On-board agents are designed to autonomously collect, analyze and condense the data sets using relatively simple yet resource-efficient algorithms. If having detected (potential) anomalies in the observed structural system, the on-board agents explicitly request specialized software agents. These specialized agents physically migrate from connected computer systems, or adjacent nodes, to the respective sensor node in order to perform more complex damage detection analyses based on their inherent expert knowledge. A prototype system is designed and implemented, deploying multi-agent technology and dynamic code migration, in a wireless sensor network for structural health monitoring. Laboratory tests are conducted to validate the performance of the agent-based wireless structural health monitoring system and to verify its autonomous damage detection capabilities.

Keywords: Wireless Sensor Networks, Agent-Based Monitoring, Mobile Agent Migration, Collective Intelligence, Distributed Intelligent Systems, Autonomous Damage Detection

1. INTRODUCTION

In the United States, more than 150,000 bridges – about 25% of the U.S. bridges – are considered structurally deficient [1]. In other countries, the situations are similar. In Germany, for example, more than 80% of the Federal highway bridges show signs of deteriorations that affect durability and longevity of the structures. The required repair and maintenance costs are estimated at more than €6.8 billion [2, 3]. Similar problems apply to other engineering structures, such as dams, buildings or wind turbines, which are subjected to ageing and environmental impacts. Thus, innovative structural health monitoring (SHM) systems, that are capable of continuously assessing the actual conditions of engineering structures by automatically sensing and analyzing relevant structural data, are needed. SHM systems built upon wireless sensor technology, having proven to be both accurate and inexpensive, are increasingly popular [4].

Composed of several wirelessly connected sensor nodes, wireless sensor networks are capable of self-interrogating collected monitoring data for signs of structural damage using sensor-based embedded engineering algorithms [5-7]. Referred to as "smart structures" or "intelligent infrastructure", today's state-of-the-art wireless sensor networks are embedded in the structure automatically collecting, analyzing, condensing and communicating vast amounts of data obtained from the structure and from its environment. Valuable information about the structure is collected in real-time to be used for local damage detection and, furthermore, for detecting global emergent structural patterns, in particular structural anomalies. These patterns could serve as a basis for gaining holistic knowledge about the structural system.

However, a collaborative self-interrogation of large amounts of measured data executed by inherently resource-poor sensor nodes entails high energy consumption. In addition, the utilization of sophisticated embedded engineering algorithms, needed for an accurate safety assessment, requires comprehensive computational power of the sensor nodes. Thus, the objective of the research presented in this paper is twofold: First, an efficient as well as accurate self-

---

* smarsly@stanford.edu; phone +1 (650) 283-5586; fax +1 (650) 723-7514; http://www.structural-health-monitoring.com

assessment of the structural condition is envisaged, to be collaboratively performed by the sensor nodes. Second, the resource consumption of the sensor nodes is to be reduced with respect to memory utilization (due to embedded algorithms used) and power consumption (due to communicated data). Both self-assessment and resource efficiency is realized through the concept of *mobile agent migration*, which is implemented into a wireless sensor network for the purpose of decentralized structural health monitoring.

Representing a topic of increasing importance in science and in engineering practice, multi-agent technology provides means and tools to create decentralized software systems that are composed of collaborating software entities ("software agents"). Software agents are self-contained, fine-grained computational entities working towards their goals on behalf of another artificial entity or a human individual. Agent properties commonly include concepts of pro-activity, reactivity, autonomy and communication [8-11]. A software agent is capable of acting with a certain degree of flexibility and autonomy, deciding on its own which actions are appropriate to achieve its goal and which other cooperative agents might be requested for assistance. The linkages of such agents form a distributed multi-agent system being scalable and easily to be extended or modified by adding further specialized software agents.

In the last decade, considerable success has been encountered in porting multi-agent technology on mobile devices such as cell phones, smart phones or wireless sensor nodes [12-16]. Thereby, the distinctive strengths of multi-agent systems, such as modularity, flexibility and extensibility, are utilized on mobile applications facilitating dynamic distributed computing. The majority of the main manufacturers of mobile devices supports some form of the Java programming language [17]. Consequently, most approaches towards mobile multi-agent systems, i.e. multi-agent systems distributively running on mobile devices, are based on Java, typically on the Connected Limited Device Configuration (CLDC) subset of the "Java Platform, Micro Edition" (Java ME). Recent developments of mobile multi-agent systems are, e. g., DARPA CougaarME [14], AFME [18], MicroFIPA-OS [19], 3APL-M [20, 21] and JADE-LEAP [22-26]. These agent platforms are designed to face a common but unique set of technical challenges: Agent infrastructure, agent communication and agent interaction protocols must be provided on computationally constrained devices in open, dynamic and decentralized environments. Owing to the limitations of the mobile devices, mobile agent platforms usually use a lightweight Java virtual machine and significantly smaller APIs than those designed for desktop environments [27, 17].

In recent years, it has been recognized that the performance and the dynamic behavior of mobile multi-agent systems can further be enhanced by dynamic code migration. Having already demonstrated high effectiveness in conventional wired decentralized systems, dynamic code migration represents an emerging and powerful paradigm that is already supported by some mobile multi-agent systems of the latest generation [28, 29]. Code migration, i.e. agents physically migrating from one mobile device to another including dynamic behavior, actual state and specific knowledge, enables mobile multi-agent systems to dynamically adapt to certain changes and altered conditions of their environment, and to reduce network load and latency. A few approaches towards mobile multi-agent systems have already been implemented, mostly being applied to mobile commerce, medical applications and distributed traffic detection (e.g. [16, 30-32]). Also, code migration has already been used in distributed engineering applications [33, 34]. However, the potential of agent migration in wireless structural health monitoring systems has not yet been investigated, nor have migration-based wireless sensor networks been implemented to solve current SHM problems.

In response to the abovementioned limitations of current wireless sensor networks, this research proposes an innovative approach that incorporates the integration of multi-agent technology and dynamic code migration into a wireless SHM system

1. to allow the sensor nodes collaboratively self-assessing the condition of the observed structure and
2. to enhance the resource efficiency of the sensor nodes with respect to data communication and on-board memory usage.

This paper is organized as follows: First, an overview of the proposed agent-based wireless SHM system is given. Focusing on the embedding of multi-agent technology into wireless sensor nodes, the development of the prototype system is shown. Specifically, the design and implementation of mobile agents is described. To validate feasibility and performance of the prototype system and to verify its autonomous damage detection capabilities, validation tests are conducted in the laboratory. The results are discussed and the performance is compared to traditional approaches commonly used in current wireless sensor networks. The paper concludes with an outlook on future research that is envisaged to further improve the proposed concept.

2. DESIGN AND IMPLEMENTATION OF AN AGENT-BASED WIRELESS SENSOR NETWORK

To reduce the quantities of communicated measured data and to economically utilize the restricted computing resources, mobile agents – representing "on-board agents" – are embedded in the wireless sensor nodes. The total set of agent-based sensor nodes forms a mobile multi-agent system. The on-board agents are designed to autonomously collect, analyze, condense and communicate the measured data of a monitored structure; they are continuously executing relatively simple yet resource-efficient algorithms in real-time at relatively low sampling rates. If having detected (potential) anomalies, specific algorithms and further knowledge is required for a more comprehensive interrogation of the data. Thus, specialized software agents ("migrating agents") are requested by the on-board agents on demand to physically migrate from an on-site local computer system or from adjacent sensor nodes to the respective sensor node. Possessing the required expert knowledge and specific algorithms, a migrating agent is capable of performing appropriate decision-making directly on a sensor node.

2.1 Architecture of the agent-based wireless SHM system

The prototype SHM system is composed of three basic components as graphically depicted in Fig. 1: (*i*) wireless sensor nodes, (*ii*) a base station and (*iii*) a local computer.

i. Each wireless sensor node hosts a set of on-board agents. In case of detected anomalies, the on-board agents proactively adapt their behavior to the new situation, e.g. by modifying the data sampling rate, and request expert knowledge from other sources within the SHM system. A collection of wireless sensor nodes forms a cluster which is managed by a head node that performs administrative tasks and hosts migrating agents (but does not collect or analyze sensor data).

ii. The base station serves as an interface between the wireless sensor nodes and the local computer installed on-site. It forwards sensor data and information, assembled by the agents, from the wireless sensor nodes to the local computer for persistent storage and further processing. Vice versa, commands sent from the local computer are communicated via the base station to the wireless sensor nodes.

iii. The local computer receives and processes the information from the sensor nodes. It also allows users to interact with the wireless sensor network and connects external resources to the wireless sensor network.

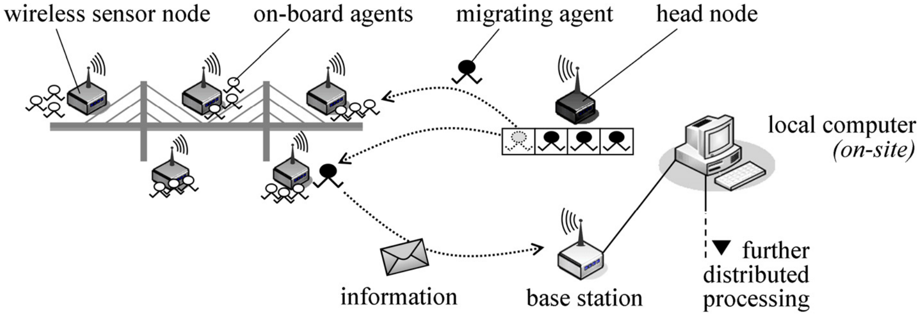


Figure 1. Architecture of the agent-based wireless SHM system.

2.2 Overview of the wireless sensing unit hardware

To demonstrate the concept of an agent-based wireless SHM system, Java-based wireless sensing units, named SunSPOTs, are employed for the prototype implementation. SunSPOTs, manufactured by Sun Microsystems, have already proven their practicability and performance in a multitude of scientific projects [35-37]. As a distinct advantage, unlike common embedded applications for wireless sensor networks which are usually written in low-level native languages such as C/C++ and assembly language, the sensing units comprise a fully capable Java ME, that is widely used, for example, on advanced mobile phones. The computational core of a sensing unit is an Atmel AT91RM9200 system on a chip (SoC) incorporating a 32-bit ARM920T ARM processor with 16 kB instruction and 16 kB data cache memories executing at 180 MHz maximum internal clock speed. The SoC includes several peripheral interface units such as USB host port, USB device port, Ethernet MAC, programmable I/O controller, serial peripheral interface controller, two-wire (I^2C) interface, etc. Memory of the sensing unit is a Spansion S71PL032J40 consisting of 4 MB flash memory and 512 kB RAM.

For wireless communication, the IEEE 802.15.4-compliant Texas Instruments (Chipcon) CC2420 single-chip transceiver is deployed operating on the 2.4 GHz unregulated FCC industrial, scientific and medical (ISM) band. For acceleration measurements, the low-power 3-axis linear accelerometer LIS3L02AQ, manufactured by STMicroelectronics, is used. Consisting of a Micro-Electro-Mechanical System (MEMS) sensor element, the LIS3L02AQ measures a bandwidth of 4.0 kHz in x- and y-axis and 2.5 kHz in z-axis over a scale of ± 6 g. In addition, the wireless sensing unit provides an integrated temperature sensor, an ambient light sensor, 2 momentary switches, facilitating the user interaction with the unit, 6 analog inputs as well as 5 general purpose I/O pins and 4 high current output pins.

On the software side, the core of the unit is the Squawk virtual machine that is compliant with the Connected Limited Device Configuration (CLDC) 1.1 Java ME configuration. The Squawk virtual machine runs on the wireless sensing unit without an underlying operating system. Instead, compact operating system functionalities are included in the Squawk virtual machine. As a result, memory is saved that would otherwise be consumed by the operating system. In addition, the Squawk virtual machine executes directly out of the flash memory. As Squawk is mostly written in Java, further memory savings arise because Java byte code is a more efficient representation than its equivalent in native code. Whereas most Java virtual machines run a single application, Squawk can run multiple applications, each being represented as an object and completely isolated from all other applications [38-40]. In total, a high degree of portability, flexibility, extendibility and maintainability as well as an ease of debugging is achieved which makes Squawk a powerful virtual machine well-suited as a foundation for prototyping Java-based multi-agent systems for wireless structural health monitoring.

2.3 Embedded software design and implementation

For the prototype implementation of the agent-based wireless SHM system, on-board agents as well as migrating agents are embedded into the sensor nodes forming a mobile multi-agent system. The mobile multi-agent system is designed based on the "MAPS" agent architecture as proposed by Aiello *et al.* [30, 41, 42]. The MAPS architecture is characterized by components offering a set of services to the agents such as message transmission, agent creation, agent cloning, agent migration, timer handling and, also, access to sensor node resources including, e.g., sensors, actuators, flash memory, battery or radio.

Technically, both on-board agents and migrating agents are implemented as components interacting through events. The component- and event-based approach allows modeling the dynamic behavior of the mobile multi-agent system through multi-plane state machines [43, 44]. A multi-plane state machine consists of several functions, variables and planes. One plane represents one behavior of an agent corresponding to the agent's role within the mobile multi-agent system. Accordingly, an agent that assumes several roles is represented through a composite behavior integrating several planes. A fundamental part of a plane is an automaton that controls the dynamic behavior of a plane, and thus of the agent, using Event-Condition-Action (ECA) rules. ECA rules within the mobile multi-agent system are represented by the triplet

$$r_{MMAS} = \langle E, C, A \rangle, \tag{1}$$

where E is the event set, C is the condition set and A are the atomic actions to be taken. An action of an ECA rule, transferring the automaton in the next state, is triggered when the event is detected and the condition is satisfied. The events of an agent, triggering actions of other agents in the mobile multi-agent system, are communicated asynchronously between the agents using unicast, multicast or broadcast inter-agent communication.

Fig. 2 shows an abridged UML class diagram illustrating the main classes of the implemented mobile multi-agent system. In particular, the classes related to the on-board and migrating agents are illustrated. The implementation of these agents and the corresponding agent behaviors is described in the following subsections.

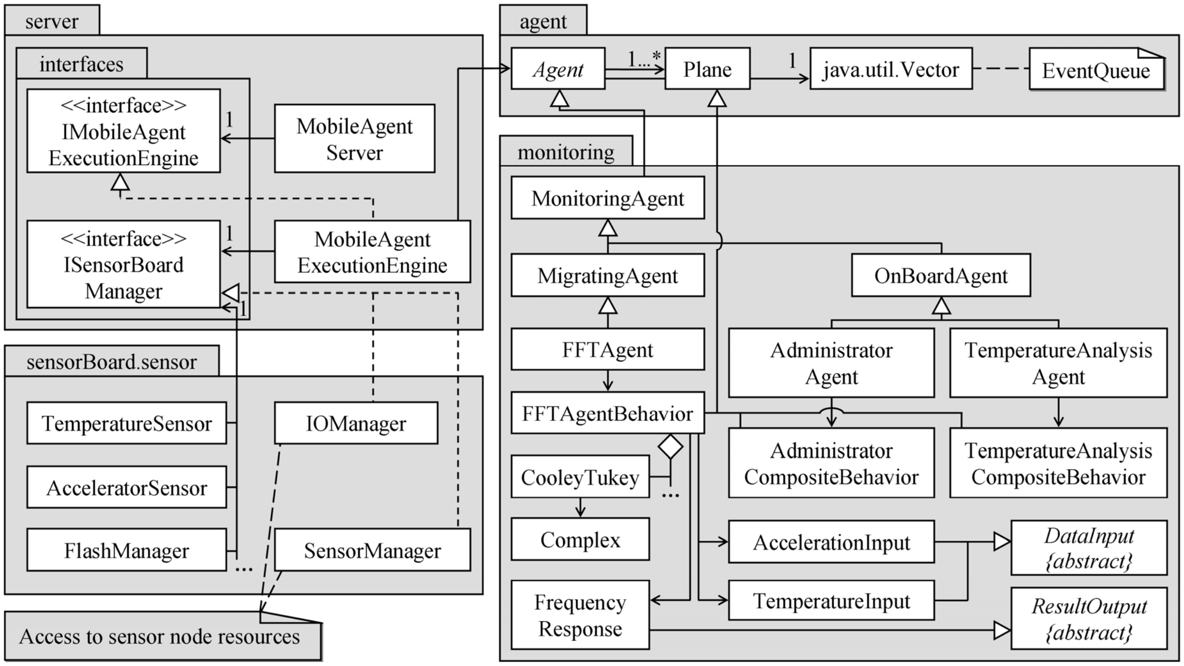


Figure 2. Abridged UML class diagram of the mobile multi-agent system extending the MAPS architecture.

On-board agents

Two on-board agents, the TemperatureAnalysisAgent and the AdministratorAgent, are prototypically implemented to be situated on each sensor node. The AdministratorAgent is responsible for the administration of a sensor node; it manages, for example, hardware and network features and provides information about memory usage, battery status and radio configurations. The TemperatureAnalysisAgent is designed to continuously collect and analyze temperature data from the observed structure. Its purpose is to detect anomalies, i.e. abnormal temperature changes, based on resource-efficient embedded algorithms. As illustrated in Fig. 3, for continuous temperature interrogations the TemperatureAnalysisAgent senses periodically temperature data via the on-board ADT7411 temperature sensor and compares the measurements with threshold values. Threshold values as well as sampling rates can be modified by the agent or, through the local computer, by human individuals. In case of detected anomalies, the TemperatureAnalysisAgent communicates the observed symptoms to the head node (s. Fig. 1) and requests specialized migrating agents capable of investigating the observed anomaly in detail. Simultaneously, the TemperatureAnalysisAgent increases the temperature sampling rate. The dynamic agent behavior described is implemented in the TemperatureAnalysisCompositeBehavior class in terms of a state machine illustrated in Fig. 3.

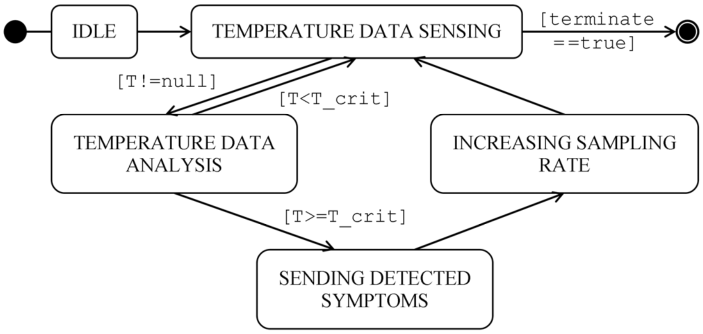


Figure 3. Dynamic agent behavior implemented as ECA automaton.

Migrating agents

The migrating agents are capable of physically migrating from one node to another including their dynamic behavior, actual state and specific knowledge. Hosted on a head node, migrating agents are sent to a sensor node if potential anomalies are observed by the on-board agents situated on the respective sensor node. Arrived on a sensor node, the migrating agents apply their inherent analysis capabilities to achieve new information about the structural condition and send the analysis results to the connected local computer. On the local computer, the information can be assembled providing, together with the information received from other sensor nodes, a holistic picture about the current structural condition.

The agent migration is implemented utilizing methods for hibernation/dehibernation and serialization/deserialization of objects provided by the Squawk Java virtual machine. Assuming agent migration from a head node H (source node) to a regular sensor node S (destination node), the destination node is contacted by the source node through a message. Next, a socket is opened based on the radiostream protocol. The radiostream protocol – a peer-to-peer protocol implemented on top of the MAC layer of the standard IEEE 802.15.4 – provides a reliable, buffered and stream-based communication between two sensor nodes. After having received the message from source node H, the destination node S sends an acknowledgement back to the node H, whereupon node H establishes a radiostream connection with node S. The migrating agent on H is paused, hibernated, serialized into a byte array and sent in a message to the destination node S – including the code, all relevant data and execution state. After having received the message, the destination node S deserializes, dehybernates and activates the migrating agent.

For the prototype system, the FFTAgent, a migrating agent capable of analyzing modal properties, is implemented. The FFTAgent, when migrated to a sensor node S, accesses the sensing hardware of the node and collect acceleration time history records. The acceleration data is used for on-board damage detection as follows: To accurately identify the primary modal frequencies of an observed structure at the location of sensor node S, the FFTAgent analyzes the collected time history data based on embedded fast Fourier transforms (FFT). The agent compares the computed primary modal frequencies to the frequencies of the healthy state being part of its internal knowledge. For calculating primary modal frequencies and frequency response functions from time history data, the FFTAgent uses the computationally efficient Cooley-Tukey algorithm [45]. As shown in Fig. 2, the agent behavior is encoded in a modular fashion in the class FFTAgentBehavior, which aggregates the CooleyTukey class and is associated with the FrequencyResponse class handling the calculated frequency response function. The diagnostic results obtained by the FFTAgent are sent to the local computer for further processing.

3. VALIDATION TESTS

To validate the migration-based concept, laboratory tests have been conducted. The tests serve as a proof of concept of the implemented agent-based wireless SHM system. According to the two main objectives of this research, two goals are pursued by conducting the laboratory tests: First, system performance data is collected to determine the resource efficiency of the prototype system. Second, the capabilities of the system are examined with respect to autonomously detecting structural changes in a decentralized-cooperative fashion. To this end, two test series are conducted using an aluminum plate as well as an aluminum beam (Fig. 4). The aluminum plate experiment is primarily intended for collecting the performance data. The aluminum beam experiment, which is considered in the following subsection in more detail, is used both for collecting the performance data and for validating the autonomous health monitoring capabilities. For that purpose, the aluminum beam is exposed to heat. The thermally induced damage is to be detected by the agent-based wireless SHM system in real-time. Furthermore, the structural condition of the aluminum beam is to be assessed autonomously by the agents involved. The results of the condition assessment are then sent to the connected local computer in the form of an automatically generated safety report.

3.1 Validation test setup

An aluminum beam is mounted on a laboratory bench as illustrated in Fig. 4. The cantilever beam is $L = 810.0$ mm long, $w = 25.4$ mm wide and $t = 3.2$ mm thick. The sensor nodes S_1, S_2 and S_3 are installed on the fixed end (S_1), in the middle (S_2) and on the free end (S_3) of the cantilever beam. Every sensor node hosts the previously introduced (and relatively simple) on-board agents TemperatureAnalysisAgent and AdministratorAgent. A separate node, the head node H, hosts the prototypically implemented (and relatively complex) migrating agent FFTAgent. The mobile multi-agent system is thus composed of the agents that are situated on the sensor nodes S_i and on the head node H. The base station B connects the mobile multi-agent system to the local computer.

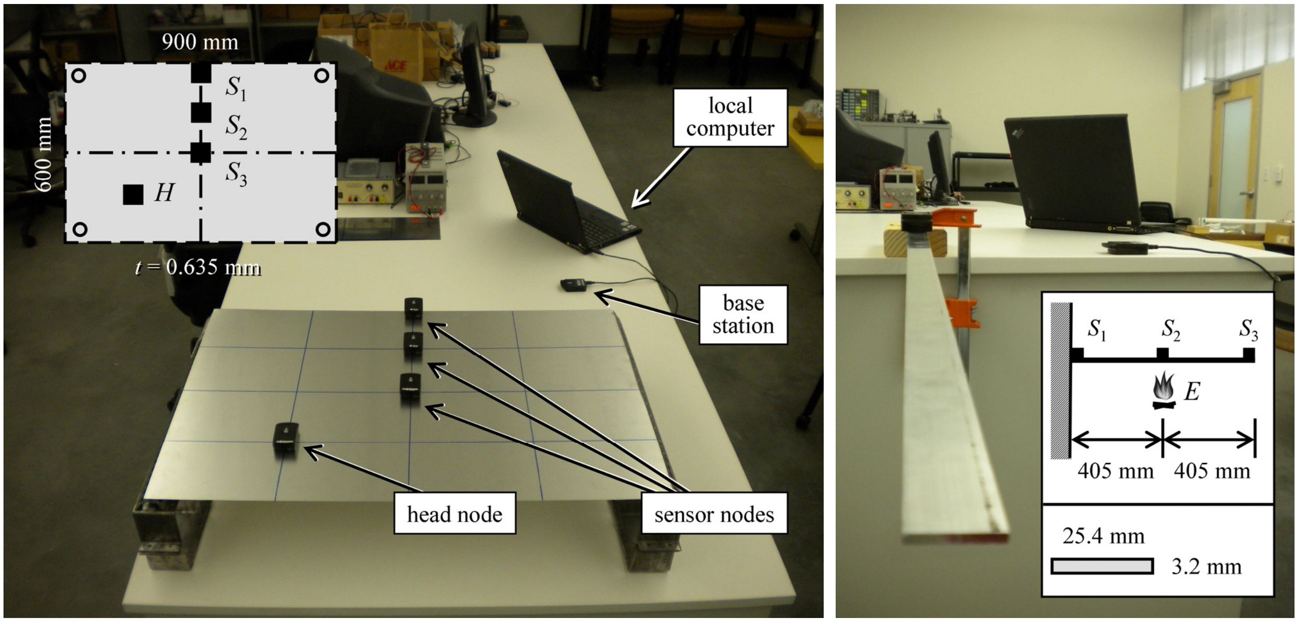


Figure 4. Aluminum plate assembled with agent-based wireless SHM system (left) and aluminum beam during instrumentation (right).

3.2 Autonomous health monitoring based on agent migration and agent cooperation

To determine the initial (undamaged) state of the cantilever beam, the free end (location of sensor node S_3) is excited by a vertical deflection forcing the cantilever to vibrate at its characteristic frequency. Fig. 5 illustrates the vertical acceleration response measured at the location of sensor node S_3. Fig. 6 shows the frequency response function. Representing the initial state of the structure, 2.42 Hz is identified from the frequency response function as the first modal frequency.

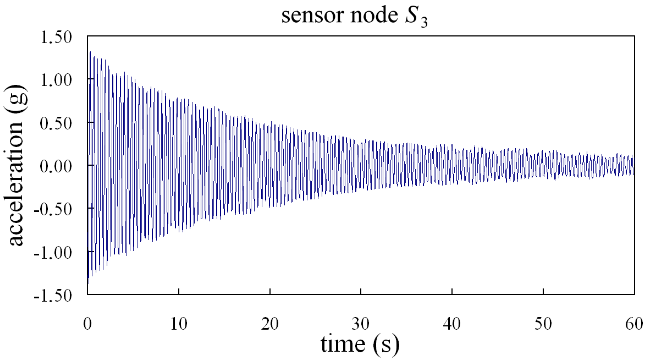


Figure 5. Vertical acceleration time history records collected at location of S_3.

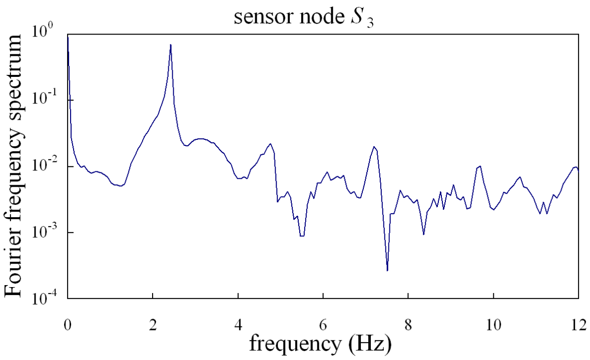


Figure 6. Frequency response function of the cantilever beam.

To induce thermal damage to the cantilever beam, an electric coil heating element is installed below the cantilever as a heating source. The on-board TemperatureAnalysisAgents operating on each sensor node are continuously sensing and interrogating temperature measurements at sampling rates of 0.1 Hz. For the experiment, the temperature threshold value T_{crit} = 50 °C is given to the agents. The threshold value indicates an anomaly, at which the mobile multi-agent system is intended to take further actions on behalf of the temperature-sensing on-board agents. Fig. 7 shows the heating process of the cantilever beam recorded at the locations of the nodes S_1, S_2 and S_3. The critical temperature is reached at location S_2 at about t = 320 s after starting the heating process.

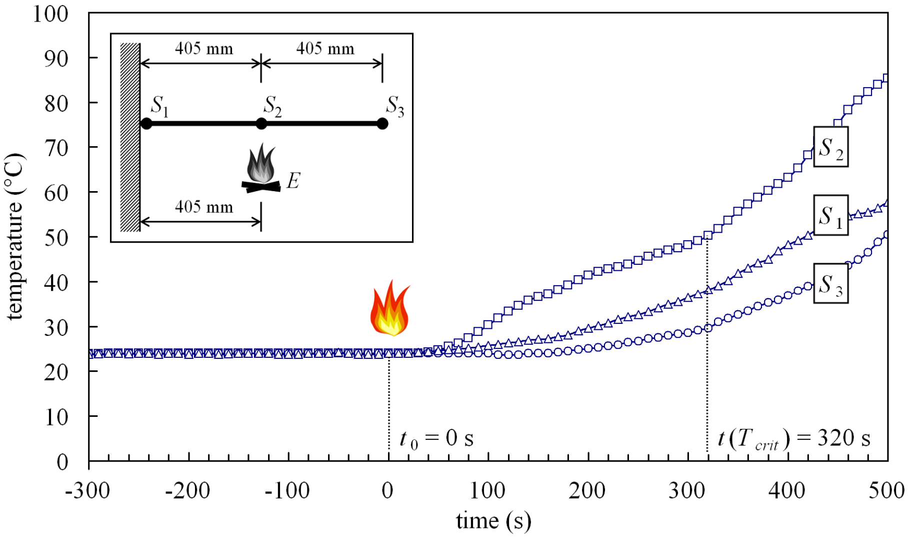


Figure 7. Heating process inducing thermal damage recorded at locations of S_1, S_2 and S_3.

As soon as having detected an anomaly, the TemperatureAnalysisAgent situated on sensor node S_2 increases the temperature sampling rate and – because of its limited knowledge not being able to analyze the current situation in more detail – communicates the detected symptoms to the head node H. Based on the received symptoms, the head node initializes a FFTAgent. The agent, assembled with all required knowledge such as modal properties of the undamaged structure, is sent to sensor node S_2 in order to analyze the current structural condition.

After arrival at sensor node S_2, the migrating FFTAgent accesses the sensor node hardware and starts collecting acceleration data using a sampling rate of 40 Hz. The FFTAgent executes fast Fourier transforms and derives the frequency response function from the acceleration time history data. Using the calculated frequency response function as a basis, the agent identifies the current first modal frequency of the cantilever beam as 2.27 Hz, and compares it to the first modal frequency of the initial condition to identify structural changes (Fig. 8).

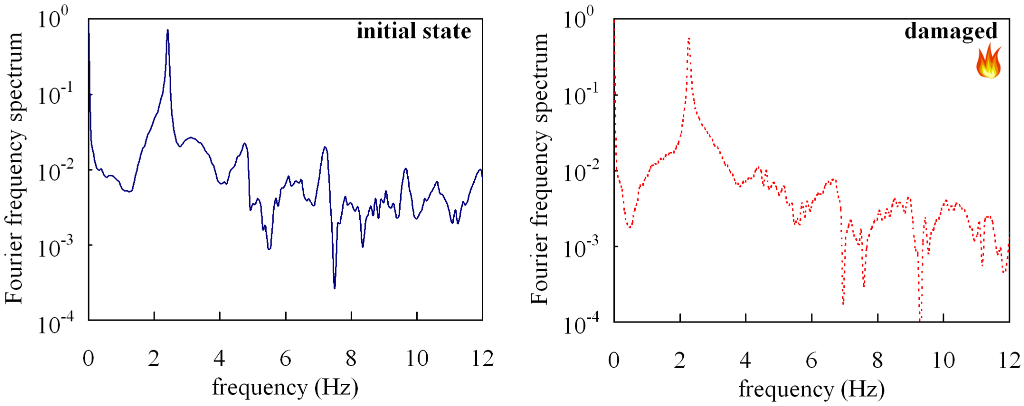
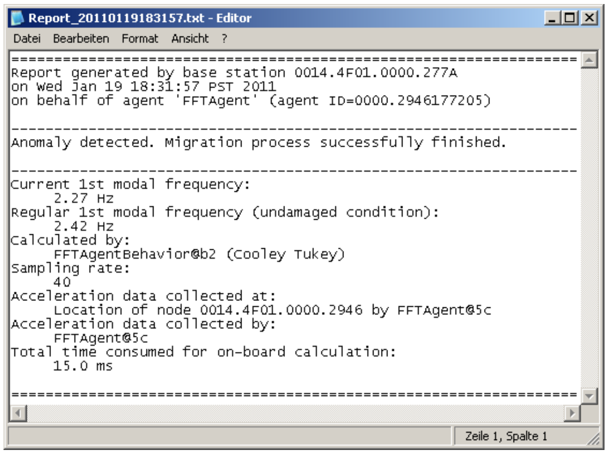


Fig. 8: Frequency response function of the cantilever beam undamaged (left) and damaged (right) as calculated by the migrating FFTAgent.

The newly acquired information on the structural condition is transmitted by the FFTAgent from sensor node S_2 to the base station. The base station assembles the achieved information and creates a safety report. The report is automatically stored by the base station on the local computer accessible by human individuals (Fig. 9).



```
==================================================================
Report generated by base station 0014.4F01.0000.277A
on Wed Jan 19 18:31:57 PST 2011
on behalf of agent 'FFTAgent' (agent ID=0000.2946177205)

------------------------------------------------------------------
Anomaly detected. Migration process successfully finished.

------------------------------------------------------------------
Current 1st modal frequency:
     2.27 Hz
Regular 1st modal frequency (undamaged condition):
     2.42 Hz
Calculated by:
     FFTAgentBehavior@b2 (Cooley Tukey)
Sampling rate:
     40
Acceleration data collected at:
     Location of node 0014.4F01.0000.2946 by FFTAgent@5c
Acceleration data collected by:
     FFTAgent@5c
Total time consumed for on-board calculation:
     15.0 ms

==================================================================
```

Fig. 9: Example of a safety report generated by the base station on behalf of the migrating agent.

3.3 Experimental results

During the validation tests, performance data is collected from the agent-based wireless SHM system and compared to current approaches commonly implemented in wireless SHM systems. The size of transmitted data as well as the utilization of internal node memory is recorded. Due to the deployment of agents migrating on demand in order to execute algorithms directly on a sensor node, a 96.4% reduction of wirelessly transferred data has been achieved compared to transferring the collected raw sensor data to a central server for analyses.

As comparing to certain traditional approaches that do perform embedded algorithms directly on a sensor node, significant enhancements have been made by the implemented migration-based approach in terms of reduced memory consumption. Reasons for the resource efficiency achieved are: (*i*) sensing and on-board storage of unnecessary measurements is largely avoided and (*ii*) complex on-board calculations are only carried out by the specialized migrating agents in case that potential (or suspected) anomalies are identified. Behind the scenes, both (collected measurements and migrating agents) are realized as Java objects, which are not a priori implemented on a sensor node; they are created and migrated during runtime only if necessary. Furthermore, if they are no longer needed, the objects are marked and swept by means of "garbage collection" by the Squawk Java virtual machine. According to the present study, in one single monitoring sequence, as described above, the memory consumption of a sensor node has been reduced by 71.0 kB compared to the conventional execution of embedded algorithms. Therein, the FFTAgent and all associated objects, such as agent behavior, agent state and algorithms used, consume only 3.2 kB. The objects representing the collected measurements consume 67.8 kB[1].

4. CONCLUSIONS

The design and implementation of an agent-based wireless SHM system, able to autonomously detect structural changes, has been presented. With the SHM system, the concept of agent migration has, for the first time, successfully been applied for wireless structural health monitoring. Validation tests, serving as a proof of concept of the proposed approach, have been conducted to test the decentralized information processing and the autonomous monitoring capabilities of the agent-based wireless SHM system. In the two tests, the sensor nodes have been mounted on an aluminum plate and an aluminum beam. The aluminum beam has been exposed to heat causing thermally induced damage. As a result, the damage has autonomously been detected by the agent-based wireless SHM system employing agent migration, distributed information processing and agent cooperation. Furthermore, the wireless transmission of collected sensor data within the SHM system as well as the memory consumption of the sensor nodes has been reduced compared to traditional approaches commonly implemented in wireless SHM systems.

Future improvements and modifications can be made to the current implementation of the agent-based wireless SHM system. Research efforts are already underway exploring the applicability of the agent-based concept in other

---

[1] Performing a 4,096-point on-board Cooley-Tukey FFT analysis.

engineering fields, which seems to be promising and reasonably practicable because of the portability, modularity and flexibility of the agent-based system. Specifically, ground water monitoring and ground motion detection are emerging disciplines in which the proposed migration-based monitoring concept could advantageously be deployed.

Beyond that, future work may include further developments that can facilitate the collaborative behavior of the agents entailing a more accurate and efficient decentralized safety assessment. To this end, further agents with a higher degree of autonomy can be designed and embedded in the sensor nodes. In the same context, the decentralized information collected from the mobile agents can be stored in a central information pool, that comprises artificial models of the observed structure and is continuously updated by the agents. For that purpose, well-established and widely used engineering applications such as databases and software tools like MATLAB or Octave, both providing Java-based interfaces, can be integrated into the agent-based wireless SHM system.

5. ACKNOWLEDGMENTS

This research is partially funded by the German Research Foundation (DFG) under grant SM 281/1-1 awarded to Dr. Kay Smarsly. The research is also partially supported by the US National Science Foundation (Grant No. CMMI-0824977). Any opinions expressed in this paper are those of the authors and do not necessarily reflect the opinions of the German Research Foundation and the National Science Foundation.

REFERENCES

[1] HSNW, "150,000 U.S. bridges are rated 'deficient'," Online: http://homelandsecuritynewswire.com/150000-us-bridges-are-rated-deficient, Homeland Security News Wire, Locust Valley, NY, USA, July 26, 2010.

[2] Sedlacek, G., "Sicherheit von Brücken: Bauwerksprüfer Gerhard Sedlacek registriert wachsende Schäden durch steigende Verkehrslasten, für die ältere Brücken nicht berechnet wurden," VDI Nachrichten, VDI Verlag GmbH, Düsseldorf, Germany, Aug. 10, 2007.

[3] Gerwens, S., "Verkehrsetat 2011 - Informationen zu den Haushaltsberatungen von Bundestag und Bundesrat," Report, Pro Mobilität - Initiative für Verkehrsinfrastruktur e.V., Berlin, 2010.

[4] Lynch, J. P. and Loh, K., "A Summary Review of Wireless Sensors and Sensor Networks for Structural Health Monitoring," Shock and Vibration Digest, 38(2), 91-128, 2006.

[5] Lynch, J. P., "Overview of Wireless Sensors for Real-Time Health Monitoring of Civil Structures," Proc. of the 4th International Workshop on Structural Control and Monitoring, New York City, NY, USA, 2004.

[6] Wang, Y., Swartz, R. A., Lynch, J. P., Law, K. H., Lu, K.-C. and Loh, C.-H., "Wireless feedback structural control with embedded computing," Proc. of the SPIE 11th International Symposium on Nondestructive Evaluation for Health Monitoring and Diagnostics, San Diego, CA, USA, 2006.

[7] Wang, Y., Swartz, R. A., Lynch, J. P., Law, K. H., Lu, K.-C. and Loh, C.-H., "Decentralized civil structural control using real-time wireless sensing and embedded computing," Smart Structures and Systems 3(3), 321-340, 2007.

[8] Wooldridge, M. J. and Jennings, N. R., "Intelligent Agents: Theory and Practice," Knowledge Engineering Review 10(2), 115-152, 1995.

[9] M. Wooldridge and N. R. Jennings, "Software Engineering with Agents: Pitfalls and Pratfalls," IEEE Internet Computing 3(3), 20-27, 1999.

[10] Weiss, G. [Multiagent systems: a modern approach to distributed artificial intelligence], MIT Press, Cambridge, MA, USA, 1999.

[11] Jennings, N. R. and Wooldridge, M. J. [Agent Technology - Foundations, Applications, and Markets], Springer, New York, NY, USA, 1998.

[12] Muldoon, C., Tynan, R., O'Hare, G. M. P. and O'Grady, M. J., "Agent-based coordination for the sensor web," Proc. of the 2010 ACM Symposium on Applied Computing, Sierre, Switzerland, 2010.

[13] Fok, C.-L., Gruia-Catalin, R. and Lu, C., "Mobile Agent Middleware for Sensor Networks: An Application Case Study," Proc. of the 4th International Conference on Information Processing in Sensor Networks (IPSN'05), Los Angeles, CA, USA, 2005.

[14] Wright W. and Moore D., "Design considerations for multiagent systems on very small platforms," Proc. of the Second International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS-03), Melbourne, Australia, 2003.

[15] Berger, M., Rusitschka, S., Toropov, D., Watzke, M. and Schichte, M., "Porting Distributed Agent-Middleware to Small Mobile Devices," Proc. of the Workshop on Ubiquitous Agents on Embedded, Wearable, and Mobile Devices, Bologna, Italy, 2002.

[16] Mihailescu, P., Binder, W. and Kendall, E., "MAE: A Mobile Agent Platform for Building Wireless M-Commerce Applications," Proc. of the 8th ECOOP Workshop On Mobile Object Systems: Agent Applications and New Frontiers. Malaga, Spain, 2002.

[17] O'Hare, G. M. P., O'Grady, M. J., Muldoon, C. and Bradley, J. F., "Embedded Agents: A Paradigm for Mobile Services," International Journal of Web and Grid Services 2(4), 379-405, 2006.

[18] Muldoon, C., O'Hare, G. M. P., Collier, R. W. and O'Grady, M. J., "Agent Factory Micro Edition: A Framework for Ambient Applications," Proc. of Intelligent Agents in Computing Systems, Reading, UK, 2006.

[19] Tarkoma, S. and Laukkanen, M., "Supporting Software Agents on Small Devices," Proc. of the First International Joint Conference on Autonomous Agents and Multi-Agent Systems, Bologna, Italy, 2002.

[20] Koch, F., "3APLM Platform for Deliberative Agents in Mobile Devices," Proc. of The International Joint Conference on Autonomous Agents and Multiagent Systems, Utrecht, The Netherlands, 2005.

[21] Koch, F., "3APL-M Platform for Lightweight Deliberative Agents. Feb-2005. Version: 1.4," Intelligent Systems Group, Institute of Information and Computing Sciences, Utrecht University, The Netherlands, 2005.

[22] Caire, G. and Pieri, F., "LEAP User Guide. 14-April-2010. LEAP 4.0," Online: http://jade.tilab.com/doc/tutorials/LEAPUserGuide.pdf, Telecom Italia S.p.A., Torino, Italy, 2010.

[23] Grimshaw, D., "JADE Administration Tutorial. 26-March-2010. JADE 4.0," Online: http://jade.tilab.com/doc/tutorials/JADEAdmin/index.html, Telecom Italia S.p.A., Torino, Italy, 2010.

[24] Bellifemine, F., Caire, G., Poggi, A. and Rimassa, G., "JADE - A White Paper," EXP in Search of Innovation 3(3), 2003.

[25] Bellifemine, F., Caire, G., Trucco, T. and Rimassa, G., "JADE Programmer's Guide. 08-April-2010. JADE 4.0," Online: http://jade.tilab.com/doc/programmersguide.pdf, Telecom Italia S.p.A., Torino, Italy, 2010.

[26] Moreno, A., Valls, A. and Viejo, A., "Using JADE-LEAP to implement agents in mobile devices," EXP in Search of Innovation 3(3), 2003.

[27] Sanneck, H., Berger, M. and Bauer, B., "Application of Agent Technology to Next Generation Wireless / Mobile Networks," Proc. of the Second World Wireless Research Forum, Helsinki, Finland, 2001.

[28] Warnier, M., Oey, M. A., Timmer, R. J., Overeinder, B. J. and Brazier, F. M. T., "Enforcing Integrity of Agent Migration Paths by Distribution of Trust," International Journal of Intelligent Information and Database Systems 3(4), 2009.

[29] Cucurull, J., Overeinder, B. J., Oey, M. A., Borrell, J. and Brazier, F. M. T., "Abstract Software Migration Architecture Towards Agent Middleware Interoperability," Proc. of the Second International Multiconference on Computer Science and Information Technology, Wisla, Poland, 2007.

[30] Aiello, F., Gravina, R., Guerrieri, A. and Fortino, G., "MAPS: A Mobile Agent Platform for WSNs based on Java Sun Spots," Third International Workshop on Agent Technology for Sensor Networks, Budapest, Hungary, 2009.

[31] Chen, B., Cheng, H. H. and Palen, J., "Integrating mobile agent technology with multi-agent systems for distributed traffic detection and management systems," Transportation Research Part C: Emerging Technologies 17(1), 1-10, 2009.

[32] Herbert, J., O'Donoghue, J., Ling,G., Fei, K. and Fok, C.-L., "Mobile Agent Architecture Integration for a Wireless Sensor Medical Application," Proc. of the 2006 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology, Hong Kong, China, 2006.

[33] Peng, J. and Law, K. H., "Building Finite Element Analysis Programs in Distributed Services Environment," Computers & Structures, 82(22), 1813-1833, 2004.

[34] Liu, D., Peng, J. and Law, K. H., "Integrating Engineering Web Services with Distributed Data Flows and Mobile Classes," Proc. of the ASCE International Conference on Computing in Civil Engineering, Cancun, Mexico, 2005.

[35] Sun Microsystems, Inc., "Sun Small Programmable Object Technology (Sun SPOT) Theory of Operation," Santa Clara, CA, USA: Sun Microsystems, Inc., 2008.

[36] Cifuentes, C., Arseneau, E., White, D. and Simmons, D., "Simplified Development of Wireless Sensor and Actuator Applications Using Java Technology," Sun Microsystems, Inc., JavaOne Conference, San Francisco, CA, USA, 2006.

[37] Cifuentes, C., White, D. and Arseneau, E., "Squawk: A Java VM for Wireless Sensor and Actuator Devices," Sun Microsystems, Inc., JavaOne 2006 Conference, San Francisco, CA, USA, 2006.

[38] Arseneau E., White, D. and Chang Y., "The Oracle Squawk Project," Online: http://labs.oracle.com/projects/squawk, Oracle Corporation, Redwood Shores, CA, USA, 2007.
[39] Simon, D., Cifuentes, C., Cleal, D., Daniels, J. and White, D., "Java on the Bare Metal of Wireless Sensor Devices: The Squawk Java Virtual Machine," Association for Computing Machinery, The Second International Conference on Virtual Execution Environments, Ottawa, Canada, 2006.
[40] Smith, R. B., Cifuentes, C. and Simon, D., "Enabling Java TM for small wireless devices with Squawk and SpotWorld," Association for Computing Machinery, Second Workshop on Building Software for Pervasive Computing, San Diego, CA, USA, 2005.
[41] Aiello, F., Fortino, G., Gravina, R. and Guerrieri, A., "A Java-Based Agent Platform for Programming Wireless Sensor Networks," Advanced online access, The Computer Journal, 2010.
[42] Aiello, F., Carbone, A., Fortino, G. and Galzarano, S., "Java-based Mobile Agent Platforms for Wireless Sensor Networks," Proc. of the International Multiconference on Computer Science and Information Technology, Wisla, Poland, 2010.
[43] Bölöni, L. and Marinescu, D. C., "A Multi-Plane State Machine Agent Model", Proc. of the 4th International Conference on Autonomous Agents, Barcelona, Spain, 2000.
[44] Dobrowolski, G., "Programming an Agent as Abstract State Machine", Proc. of the International Workshop of Central and Eastern Europe on Multi-Agent Systems, Budapest, Hungary, 2005.
[45] Press, W. H., S. A. Teukolsky, W. T. Vetterling and Flannery, B. P. [Numerical Recipes in C: The Art of Scientific Computing], Cambridge University Press, New York, NY, USA, 1992.