A Multi-Agent-Based Collaborative Framework for a Self-Managing Structural Health Monitoring System

Kay Smarsly[1], Kincho H. Law[1], Dietrich Hartmann[2]

[1]*Stanford University, Department of Civil and Environmental Engineering, Y2E2 Building, 473 Via Ortega, Stanford, CA 94305-4020, USA*

[2]*Ruhr-University Bochum, Department of Civil and Environmental Engineering, Universitaetsstr. 150, D-44801 Bochum, GERMANY*

Abstract. The deterioration of civil infrastructure due to ageing, altered requirements, excessive loading or inadequate maintenance underpins the urgent need for reliable and cost-effective monitoring systems. This paper presents a framework for monitoring the condition of civil infrastructure. A self-managing software framework based on multi-agent technology is designed to remotely access and autonomously process collected information about the monitored structure. The distributed software framework ensures automated anomaly detection, supports collaborative diagnostic tools and enhances communications among distributively located users participating in the monitoring activities. The multi-agent framework has been implemented and validated for the monitoring of a 500 kW wind turbine in Germany. The long-term field instrumentation shows the practicability, efficiency, fault-tolerance and robustness of the system for structural health monitoring applications. This research has demonstrated a practical adoption of a multi-agent-based structural health monitoring system for the long-term deployment in the field.

Keywords: Multi-Agent Technology, Structural Health Monitoring, Distributed Computing, Autonomous Computing, Wind Turbine Monitoring

Introduction

Many civil engineering structures, such as wind turbines considered here, are approaching or have already exceeded their designed service life. Due to ageing, environmental factors and, often, inadequate maintenance, the conditions of these structures are progressively deteriorating, urgently needing inspection and repair works. For example, in Germany more than 20,000 bridges require immediate rehabilitation (Smarsly and Hartmann 2009a). Also in the U.S., more than 26 % of the bridges are rated either structurally deficient or dilapidated, requiring an estimated annual investment of about $17 billion for rehabilitation (ASCE 2009). Similar situations apply to other types of civil engineering structures. For wind turbines, which are usually designed to work for 120,000 hours of operation throughout their design lifetime of about 20 years, at least 25 % of the total expenses during their lifetime are operation and maintenance costs (EWEA 2010, DWIA 2010).

New concepts and technologies are needed to develop reliable and cost-effective strategies for monitoring the integrity and health of engineering structures. Although safety and structural performance are crucial to the economic and societal welfare, current monitoring practice, mostly carried out by visual inspections, is too costly, labor intensive and ineffective due to the lag time between inspection intervals. Recent developments in the field of structural health monitoring (SHM) have demonstrated that new methodologies are available to supplement visual inspections and enhance safety. Research in SHM represents an active field encompassing numerous research directions where various state-of-the-art reviews have been reported (Sohn *et al.* 2001, Chang *et al.* 2003, Elgamal *et al.* 2003a, b, Glaser *et al.* 2007, Spencer *et al.* 2007, Liu *et al.* 2006). Mathematical and numerical approaches have been developed for system identification, damage detection and safety assessment (see, for examples, Farrar *et al*. 2005, Koh *et al*. 2006, Koh and Perry 2007, Zhou and Yan 2006 and Sohn *et al*. 2004). In addition, the advent of wireless technologies allows for robust monitoring systems, including considerable benefits with respect to decentralized data processing and installation costs (Straser *et al.* 1998, Lynch *et al.* 2002,

Spencer *et al.* 2004). Furthermore, with the continuing enhancements of computational power, energy efficiency and data communications, new SHM concepts have been validated by experimental studies in the laboratory and also successfully been applied in the field (Lynch *et al.* 2005, Wang *et al.* 2005a, b, Law *et al.* 2008, 2009, Lynch 2005). Automated SHM systems offer the opportunity to accurately observe the current conditions of engineering structures and to support the detection of structural damage as early as possible before any catastrophe and potential loss of life. With a robust SHM system, the assessment of limit states as well as serviceability of structures can be greatly improved and the expenses for quality control, repair and maintenance work can significantly be reduced.

While most current research in SHM focuses on the implementation and validation of damage detection algorithms, sensors and sensor networks, robust software platforms designed to facilitate collaborative efforts that support data usages, data archiving and automated processing of monitoring tasks have received little attention. In this research, an integrated monitoring concept is proposed coupling artificial intelligence (AI) techniques, well-established advanced engineering methods and decentralized Internet-enabled approaches. Specifically, multi-agent technology, evolved from distributed AI research, is introduced as a conceptual foundation to create an autonomous SHM system for continuously assessing structural performance and safety. As demonstrated by Smarsly and Hartmann (2007, 2009a), AI has reached a level of maturity that allows creating highly sophisticated AI-based SHM systems capable of performing the monitoring tasks, such as data acquisition or data interrogation, in a fully autonomous and distributive fashion. Furthermore, users' interactions as well as autonomous applications are facilitated and, as a result, a proactive support of human experts in charge of monitoring can be achieved (Smarsly and Hartmann 2010, Smarsly *et al*. 2007, Smarsly 2008, 2010). The agent-based approach leads to a high degree of flexibility, modularity and extendibility of the SHM system. By exchanging and interpreting messages, the collaborating agents, distributed across many computer hosts, decide collectively how to assign and handle the monitoring tasks.

Based on prior research experiences with wind turbines (Lachmann *et al.* 2009, Smarsly and Hartmann 2009b, c), the proposed monitoring concept is designed, implemented and validated for the monitoring of a 500 kW wind turbine in Germany. The wind turbine serves as a reference structure for comprehensive system tests and continuous long-term field validation studies. This research has demonstrated, for the first time, a practical adoption of a multi-agent system for structural health monitoring applications in the field.

This paper is organized as follows: First, an overview of a SHM system for wind turbines is given. Second, the hardware sensor technologies deployed on the wind turbine are briefly discussed. The paper focuses on a decentralized software platform, which is designed to provide reliable data acquisitions and autonomous data interrogations (for real-time monitoring) and to support collaborations among users through agent-based monitoring services (for on-demand monitoring). Long-term monitoring results on the wind turbine are presented followed by a brief discussion on future research directions.

Overview of the SHM System

As illustrated in Fig. 1, the SHM system for the wind turbine structure comprises of three monitoring subsystems, an on-site hardware subsystem M_1, a real-time monitoring subsystem M_2 and an on-demand monitoring subsystem M_3. Installed in the observed engineering structure, the hardware subsystem M_1 includes a dense array of sensors connected to data acquisition units, which are controlled by a computer located on-site in the structure. As depicted in Fig. 1, the hardware subsystem M_1 continuously senses and temporarily stores data sets relevant to the monitoring of the structure. Simultaneously, the stored data, termed "*primary monitoring data*", is analyzed with respect to anomalies and streamed to the real-time monitoring subsystem M_2 which is installed off-site in a laboratory. The subsystem M_2 is designed to autonomously execute crucial monitoring tasks, such as converting the collected data sets from primary

into "*secondary monitoring data*" (easily interpretable by human users), interrogating the data, storing the data persistently and providing system access to the human users (Smarsly and Hartmann 2009). The secondary monitoring data can be retrieved by authorized users ("human users") and by intelligent software agents ("artificial users") that are characterized being social and autonomous computational entities (Ferber 1999, Bellifemine *et al.* 2003). The third subsystem, the on-demand monitoring subsystem M_3, allows human users and software agents remotely logging into the real-time monitoring subsystem M_2.

The interfaces connecting M_2 and M_3 are designed to provide remote access to the SHM system. A *web interface connection* and a direct *database connection* as shown in Fig. 1 support collaborations among the users and provide remote access to the collected monitoring data. The web interface connection is designed to facilitate interactions by the human users through web browsers, e.g. to conduct monitoring tasks with respect to safety and performance assessment of the observed structure. The database connection, primarily intended for supporting data analyses, is utilized by both the human users and the software agents for remotely accessing the SHM system and for downloading and analyzing the monitoring data.

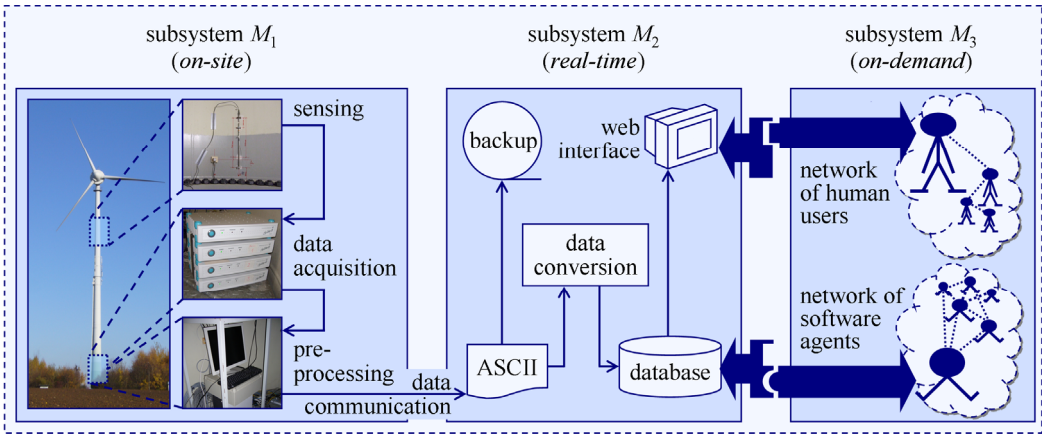


Fig. 1 Architecture of the SHM system

On-Site Hardware Subsystem

The reference structure, a 500 kW wind turbine, is fully instrumented with sensors, data acquisition units and a computer, representing the on-site hardware subsystem M_1 as illustrated in Fig. 1. The sensors are installed inside the steel shaft of the wind turbine, outside the structure and on the foundation. As shown in Fig. 2, six three-dimensional PCB-3713D1FD3G piezoelectric accelerometers, manufactured by PCB Piezotronics (PCB 2010), are placed at five levels in the wind turbine shaft. In addition, at the foundation of the wind turbine three single-axis PCB-393B12 piezoelectric seismic ICP accelerometers are installed. Since small accelerations are expected at the foundation, the sensitivity of the sensors is 10,000 mV/g with a measurement range of ± 0.5 g and a frequency range from 0.6 to 450 Hz. The acceleration data are acquired at a sampling rate of 100 Hz.

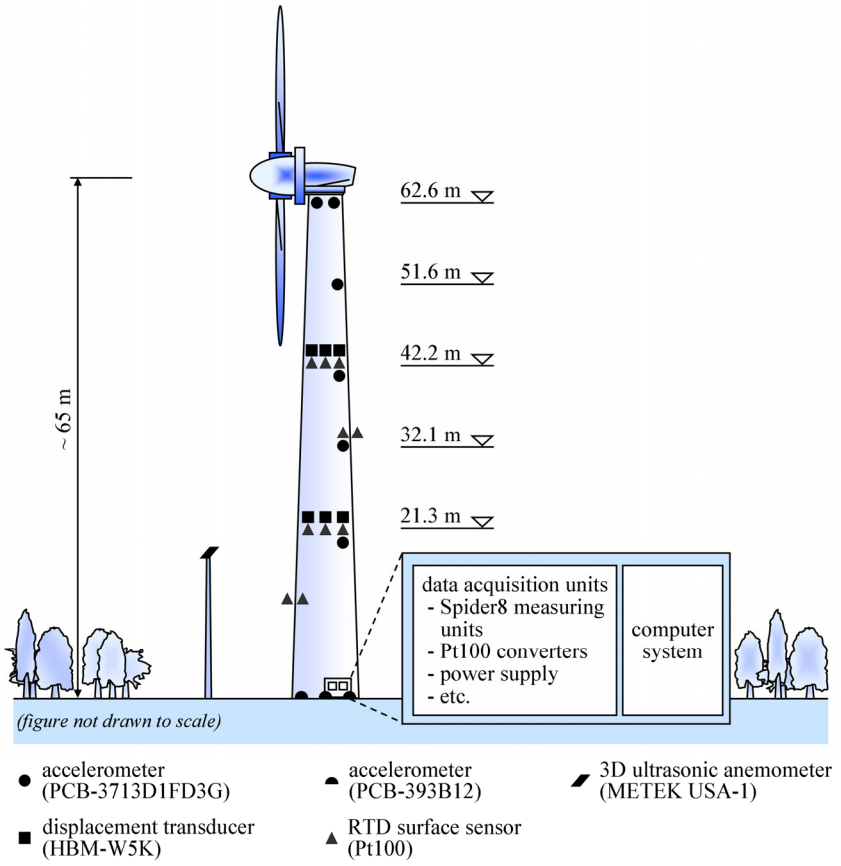


Fig. 2 On-site hardware subsystem

In addition to measuring accelerations along the height of the structure, six inductive displacement transducers, type HBM-W5K, are installed at two different levels inside the shaft. The transducers are complemented by Pt100 resistance temperature detectors to adequately consider temperature influences on the displacement measurements. At two further levels both inside and outside the wind turbine, additional temperature sensors are placed, which are primarily proposed for determining the temperature gradient of the wind turbine shaft. For the temperature data acquisition, three 4-channel Picotech RTD input modules PT-104, each acquires temperature data at 1 Hz, are installed and, through RS232 to USB interface converters, connected to the computer located in the wind turbine. For evaluating complete three-dimensional wind information, an ultrasonic anemometer is mounted on a telescopic mast at a height of 15 m near the wind turbine. The anemometer, a USA-1 manufactured by Metek (Metek 2010), continuously monitors the horizontal wind speeds (0…60 m/s) and directions (0…359°), vertical wind speeds (0…60 m/s) and temperatures (–40…+60°C). Fig. 3 illustrates the instrumentation of the sensors inside the wind turbine shaft and the anemometer in a nearby location.

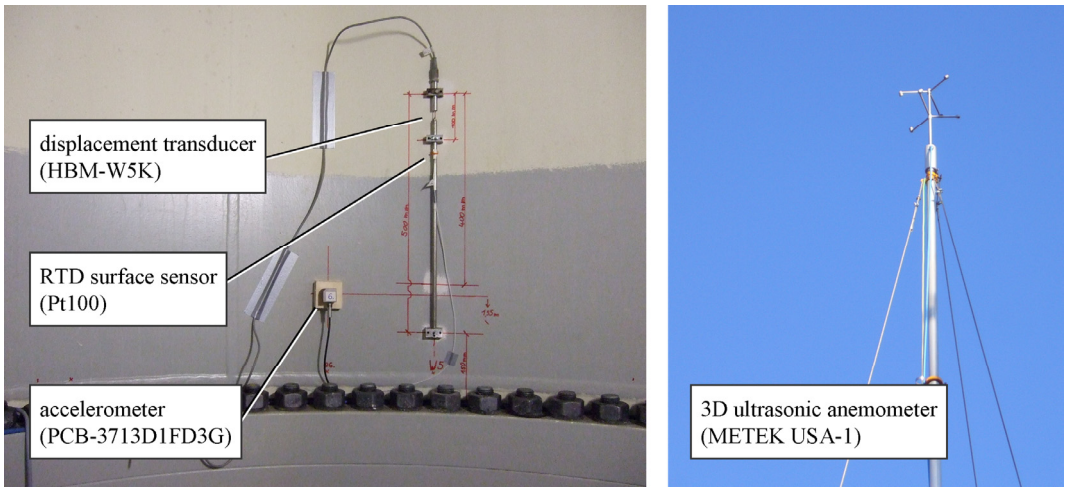


Fig. 3 Displacement transducer, temperature sensor and accelerometer inside the wind turbine shaft (left) and three-dimensional anemometer (right) being part of the on-site hardware subsystem

Representing the state-of-the-art hardware recommended for wind turbine monitoring (IEC 2006, Quecedo *et al.* 2007, Johansen 2008, Johnsson and Højholt 2007), four coupled Spider8 measuring units are placed inside the wind turbine for collecting and processing the sensor data. Each measuring unit includes 8 digital amplifiers, using 4.8 kHz carrier-frequency technology. All channels are operating with individually synchronized A/D converters ensuring simultaneous measurements on all channels at sampling rates between 1 Hz and 9,600 Hz. The sensor data, being sampled and digitized, is forwarded to the local computer placed inside the wind turbine for temporary storage and further processing of the data into a readable (ASCII) format facilitating on-site testing and debugging. To illustrate the primary monitoring data format, Fig. 4 shows a representative part of an ASCII file generated by the installed temperature data acquisition unit. Over a telecommunication line, the collected primary monitoring data is transferred from the wind turbine (subsystem M_1) to the real-time monitoring subsystem M_2.

Datum→	Zeit	Channel·1	Channel·2	Channel·3	Channel·4	Channel·1	Channel·2	Channe
→	→	°C	°C	°C	°C	°C	°C	°C
01.06.2009	07:59:28	22,378	******	******	******	******	22,692	22,726
01.06.2009	07:59:29	22,378	******	******	******	******	22,692	22,726
01.06.2009	07:59:30	22,377	******	******	******	******	22,692	22,725
01.06.2009	07:59:31	22,377	******	******	******	******	22,692	22,725
01.06.2009	07:59:32	22,377	******	******	******	******	22,692	22,725
01.06.2009	07:59:33	22,380	******	******	******	******	22,687	22,727
01.06.2009	07:59:34	22,380	******	******	******	******	22,687	22,727
01.06.2009	07:59:35	22,380	******	******	******	******	22,687	22,727
01.06.2009	07:59:36	22,378	******	******	******	******	22,688	22,729
01.06.2009	07:59:37	22,378	******	******	******	******	22,688	22,729

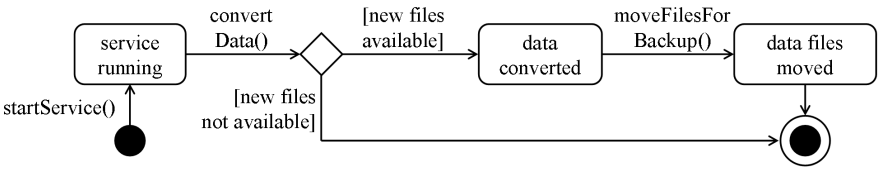
Fig. 4 Primary monitoring data generated by a 3×4-channel temperature data acquisition unit

Real-Time Monitoring Subsystem

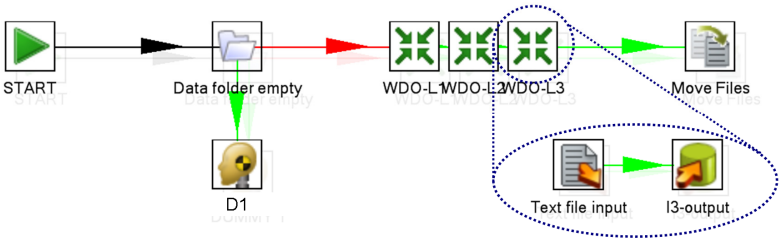
To extract useful knowledge with respect to the structural behavior of the wind turbine, the primary monitoring data collected is automatically backed-up and condensed. After being converted into secondary monitoring data (new structured with proper labels and definitions), the data is stored into a central database system providing access to every participating (human or artificial) actor. The purpose of the real-time monitoring subsystem M_2 is thus to perform in real-time the data archival and data

processing tasks such as condensing, converting and storing of the acquired data sets. For that purpose, the subsystem M_2 comprises (a) server systems for on-line data synchronization, data conversion and for service-oriented data transmission, (b) RAID-based storage systems for periodic backups, (c) database systems for persistent data storage and (d) web interfaces for remotely accessing and visualizing the monitoring data.

In order to automate the conversion of the primary data into secondary data, metadata is added to provide definitions of installed sensors, logger IDs, output specification details, date and time formats, etc. For automated data conversion, the open-source tool "PDI" (Pentaho Data Integration) is integrated into the monitoring subsystem M_2. As a commercially available software, the PDI tool offers metadata-driven conversion and data extraction capabilities (Roldan 2009, Castors 2008). Fig. 5a shows a series of tasks, expressed in terms of state machine protocols, that are explicitly defined for executing the data conversion process. In essence, the data conversion process includes (i) starting the conversion service, (ii) data processing and (iii) moving the input data files for backup. Fig. 5b depicts the corresponding implementation of the conversion service with the PDI tool.



(a) Conversion service expressed in terms of a state machine protocol



(b) PDI representation of the conversion service

Fig. 5 Abridged illustration of the automated data conversion

The converted data, i.e. the secondary monitoring data, is persistently stored in a MySQL database system to be accessed by other middleware, by application tools and – in particular – by both the human users and the software agents for further analyses. Fig. 6 displays the data obtained from the six temperature sensors s_1, ..., s_6 after the conversion of the primary monitoring data (shown in Fig. 4) into secondary monitoring data.

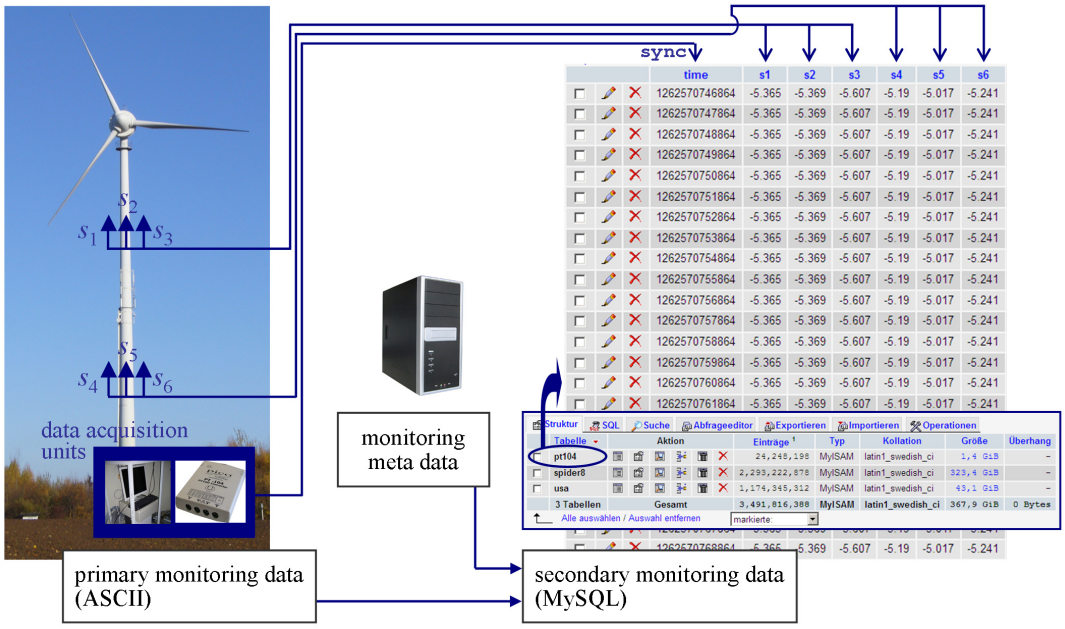


Fig. 6 Example data structure (Smarsly and Hartmann 2009b)

A Multi-Agent Software Platform for On-Demand Monitoring

The on-demand monitoring subsystem M_3 is realized as a decentralized multi-agent system. As the term implies, the multi-agent system is composed of several interconnected agent systems. Each agent system hosts specialized software agents that are introduced to accomplish specific monitoring tasks, such as interrogating the data sets, in cooperation with each other. The software agents are designed as *autonomous*, *reactive* as well as *social* computational entities:

- **Autonomy**: The software agents are capable of controlling their own actions and taking their own decisions. Unlike communication in object-oriented programming, where objects invoke methods of other objects, the agents perform their actions according to their self-interests in a self-contained fashion. Upon receiving a request (either from a human or from another agent) for performing a certain action, an agent can select to answer or to discard the request.
- **Reactivity**: Reactive agent behavior allows triggering appropriate monitoring actions in response to external events in a timely manner (e.g. in case of an anomaly detected in the observed data sets).
- **Social ability**: The agents interact with other agents and with human users to achieve their goals and, thus, to handle the overall monitoring task cooperatively.

The multi-agent software design facilitates modular development of monitoring tools, task execution, and integration of monitoring efforts that are not necessarily co-located at a specific site. The software agents cooperatively solve the monitoring tasks communicating through agent messages. Thus, a high degree of flexibility, modularity and extendibility is achieved.

Architecture of the Multi-Agent System

In contrast to the monitoring subsystems M_1 and M_2, which continuously monitor the structure and collect, process and archive the sensor data, the subsystem M_3 is operating according to an on-demand basis by a user, a software agent or an application tool. The on-demand monitoring subsystem M_3 is designed and implemented as a multi-agent system *MAS*. The purpose of *MAS* is to provide a distributed network of cooperating software agents, each performing a different monitoring task. As shown in Fig. 7, the multi-agent system *MAS* is composed of interconnected and spatially distributed agent systems AS_j in which the software agents $a_{i,j}$ are situated.

A main agent system AS_{main} provides agent registering and addressing information, remote monitoring mechanisms, security functionalities as well as infrastructure and management services. The main agent system hosts two special agents, namely the "Agent Management System (AMS) agent" and the "Directory Facilitator (DF) agent". The AMS agent is responsible for the overall management actions, including creating or terminating other agents. The DF agent implements a yellow page service, allowing the agents to advertise their services as well as to look up other agents that offer the services needed to accomplish a specific monitoring task cooperatively. The AMS and DF agents together provide the overall configuration description of the *MAS*. Through the management information and functions residing in the AS_{main}, distributed software agents can locate, interact and collaboratively work with other autonomous software agents. In addition, AS_{main} includes a "supervisor agent" $a_{s,main}$ for managing the overall monitoring process and a "database agent" $a_{d,main}$ for providing remote access to the central MySQL database residing in the monitoring subsystem M_2.

Additional agent systems AS_j can be launched remotely to perform a variety of structural monitoring functions as long as they are registered to the main agent system AS_{main}. As a result, the capabilities of the multi-agent system *MAS* are extended by those functions offered by the software agents of the remote agent systems AS_j. As opposed to the main agent system AS_{main} that hosts the software agents to perform the *general* system functions, each remote agent system AS_j consists of *specialized* agents responsible for solving specific sets of monitoring tasks.

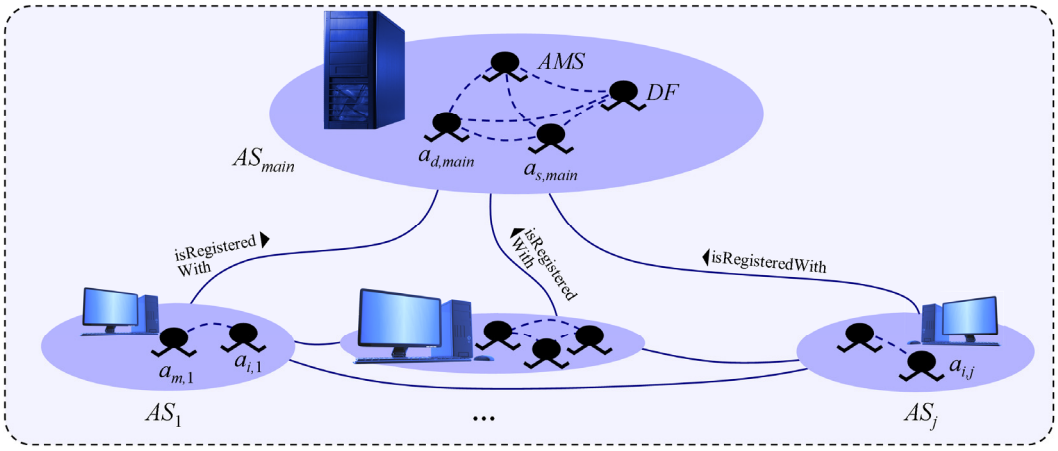


Fig. 7 Distributed architecture of the multi-agent system

The multi-agent system *MAS* is implemented using JADE (Java Agent DEvelopment Framework). JADE is a widely used Java-based software framework supporting the implementation of multi-agent systems through a middleware that complies with current agent standardizations (Bellifemine *et al.* 2004, 2007). As a distinct advantage supporting a decentralized monitoring, all agent systems AS_i constituting *MAS* can easily be distributed over several computer systems. Functions are provided by the JADE technology to allow agents, regardless of the operating system, to clone themselves or to migrate from one computer to another. Beyond that, the *MAS* is protected against security threats by a number of mechanisms that are provided by JADE. As to that, a high level of security at both the infrastructure and the application layer is ensured through the utilization of authenticators and a communication based on signed and encrypted messages. Specifically, software agents trying to accomplish monitoring tasks are allowed to perform only certain privileged actions.

Implementation of Software Agents

The prototype implementation of the multi-agent system *MAS* includes the main agent system AS_{main} and a remote agent system. An agent system AS_j can be installed on any computer that is connected to AS_{main}. To illustrate the design of the multi-agent system, the implementation of a remote agent system AS_1, which

comprises an "interrogator agent" $a_{i,1}$ implemented for analyzing the monitoring data and a "mail agent" $a_{m,1}$ for sending notifications to the responsible individuals in case of detected anomalies, is described in this section.

An Interrogator Agent

The interrogator agent $a_{i,1}$ of AS_1 analyzes the sensor data collected from the structure to detect potential anomalies. Many well-established descriptive statistics-based algorithms for detecting anomalies have been proposed in the literature, e.g. using the mean value along with a multiple of standard deviation, utilizing the median and the interquartile range or combining diverse test procedures, such as Grubb's test, Hampel's test, Walsh's test, Dean-Dixon-Test, Chauvenet's criterion, Nalimov-Test, Peirce's criterion, etc. (Sachs and Hedderich 2009). Implementing a simple outlier test procedure, the interrogator agent $a_{i,1}$ uses the mean value and a multiple of standard deviation on a collected data set to detect anomalies. For a data set with n measurements, a measurement x_k ($k = 1...n$) is considered an outlier if it exceeds the range

$$[\bar{x} - \lambda\sigma,\ \bar{x} + \lambda\sigma] \tag{1}$$

where

$$\sigma = \sqrt{\frac{1}{n-1}\sum_{k=1}^{n}(x_k - \bar{x})^2}, \quad \bar{x} = \frac{1}{n}\sum_{k=1}^{n}x_k \tag{2}$$

In Eq. 1, λ (say 3, a commonly used value for outlier detection) is the detection sensitivity to be defined by $a_{i,1}$ or by a user.

With the prototype multi-agent system, the outlier test procedure can be carried out in two ways. The interrogator agent can first acquire the sensor data from the real-time monitoring database via the database agent $a_{d,main}$ of AS_{main} and then perform data analyses, for example, using a procedure implementing Eq. 1

and Eq. 2. Alternatively, with the MySQL database connected to the multi-agent system, the data analyses can be executed by directly querying the database using a Structured Query Language (SQL) command:

```
query =   "SELECT @mean := AVG("+s+"), @std:= STDDEV_POP("+s+") FROM "+t+";"+
          "SELECT "+s+" FROM "+t+" WHERE ABS("+s+"-@mean) > @std*3;";
```

In the query statement above, *s* defines the sensor to be observed and *t* is the database table corresponding to the data set stored according to a particular sensor type such as accelerometers, displacement transducers or temperature sensors.

For both approaches, the interrogator agent interacts with the database via the database agent $a_{d,main}$. The connection to the database is realized through a programming interface based on the Java Database Connectivity (JDBC). JDBC is an industry standard for database-independent connectivity between Java applications, such as software agents considered here, and various databases; JDBC provides a call-level API for SQL-based database access. The security of database requests and data transmission are provided by the database system, i.e. the MySQL database, which requires password and username as well as secure drivers to be specified by an agent trying to access the database system. Fig. 8 shows an abridged example of how a database connection is established using the required specifications of URL, database driver, username and password (`connect` method). As shown in the figure, an SQL command query, which is issued to request for the sensor data set or to request for performing an outlier test, can be dynamically executed. As can be seen, beyond the database query described above, other complex queries can be executed by the agent depending on its goals (`interrogate` method). Furthermore, the interrogator agent can be implemented with complex damage detection procedures interacting with the real-time monitoring database.

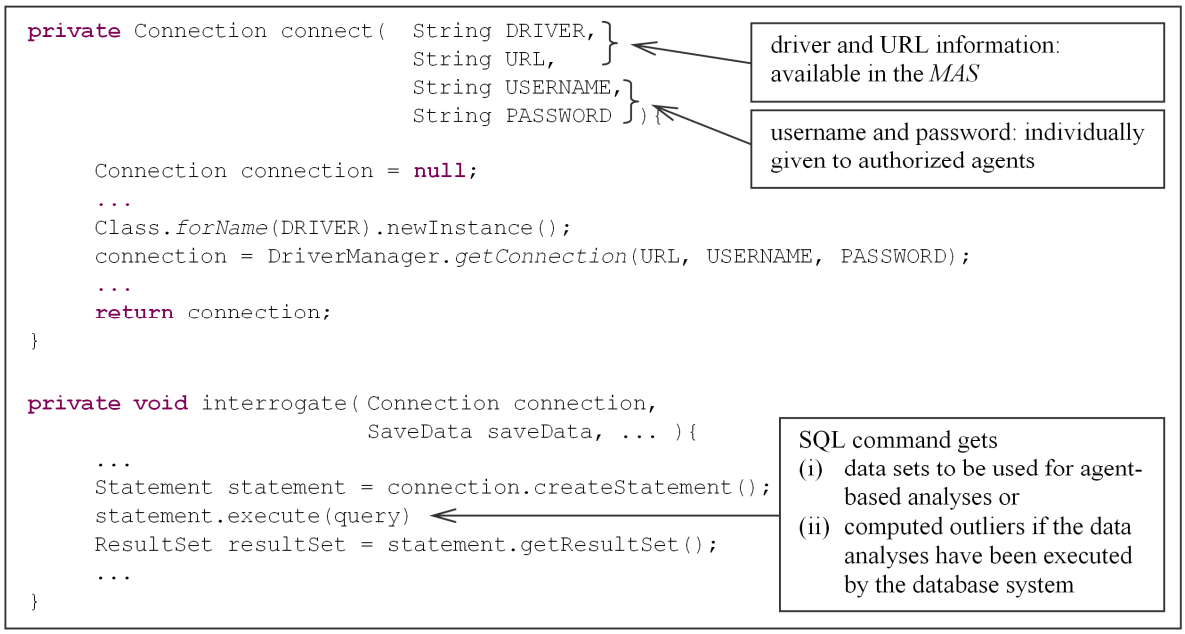


Fig. 8 Abridged example of establishing and utilizing a database connection

A Mail (Notification) Agent

Upon the detection of an anomaly, responsible individuals are immediately notified. On behalf of the interrogator agent $a_{i,1}$ the notification is issued by the mail agent $a_{m,1}$ via an agent-based email messaging service in a two-step process. In the first step, the mail agent $a_{m,1}$ collects all relevant data, which is necessary for composing the emails. The mail agent $a_{m,1}$ creates the emails using the information about the observed anomaly that are provided by $a_{i,1}$. Furthermore, metadata, such as addresses of the recipients or the email server to be used, are acquired for the automated emailing; these metadata are stored in the configuration files in AS_{main} as part of the multi-agent system. In other words, the email content is created based on the anomaly information received from $a_{i,1}$, and the email header is created based on the metadata obtained by $a_{m,1}$ from the configuration files resided in AS_{main}.

Once the emails are composed, they are sent by $a_{m,1}$ to the email clients (the human experts) as recommended by AS_{main} using the Simple Mail Transfer Protocol (SMTP) – an Internet standard for email

transmission – for delivery of the emails. For that purpose, $a_{m,1}$ communicates with a SMTP server that is specified (and can be changed any time by the users, as approved by and registered with AS_{main}) in the configuration files. The SMTP server relays the emails on to the SMTP servers of the email recipients in order to reach the responsible users through an email client. Secure email messages are ensured by username- and password-based authentications that the mail agent $a_{m,1}$, like a human user, needs to specify when trying to access the SMTP server.

Agent Interaction

As discussed earlier, the monitoring tasks executed within the multi-agent system are conducted cooperatively by a collection of autonomous software agents involved. At the global level, an agent system, such as AS_1 connects and interacts with other autonomous agent systems AS_j using the information provided by the main agent system AS_{main}, as illustrated by the mail agent service. The actions undertaken by an agent are based on the agent interactions with other agents (in the multi-agent system) and on the local agent behaviors. An agent behavior defines how the agent perceives, analyzes and, in particular, reacts to various external events (e.g. sensor malfunctions, detected anomalies or unavailable monitoring data). Agent behaviors can be executed either sequentially or concurrently. Each individual behavior has the relevant information about interaction protocols and communication details as well as specifications of languages required to carry out the monitoring tasks.

To illustrate the interactions between the agents, Fig. 9 shows three agent behaviors, `CyclicInterrogation`, `MailInteractionInitiator` and `MailInteractionResponder`, that are needed for executing the "anomaly detection" described within the agent system AS_1. The behaviors `CyclicInterrogation` and `MailInteractionInitiator` belong to the interrogator agent $a_{i,1}$. The third agent behavior `MailInteractionResponder` is an integral part of the mail agent $a_{m,1}$. As shown in Fig. 9, the anomaly detection executed by $a_{i,1}$ is initiated by its `CyclicInterrogation` behavior. The

`CyclicInterrogation` behavior periodically requests and analyzes the monitoring data with respect to anomalies and inconsistencies. Upon detecting an anomaly, $a_{i,1}$ requests the email notification service to notify the responsible human experts. To this end, $a_{i,1}$ uses its `MailInteractionInitiator` behavior for contacting mail agent $a_{m,1}$. The notification of the human users through email is accomplished by the mail agent's `MailInteractionResponder` behavior. Upon receiving and accepting a notification request, $a_{m,1}$ informs $a_{i,1}$ about its decision on agreeing or refusing to handle the request. Once agreed, $a_{m,1}$ processes the request by preparing and sending the email notifications to the users responsible for handling the detected anomaly, as described earlier. If the emails are sent successfully, $a_{m,1}$ informs $a_{i,1}$ and the monitoring procedure is finalized; otherwise, a failure message is sent from $a_{m,1}$ to $a_{i,1}$, which may need to take other actions, if necessary.

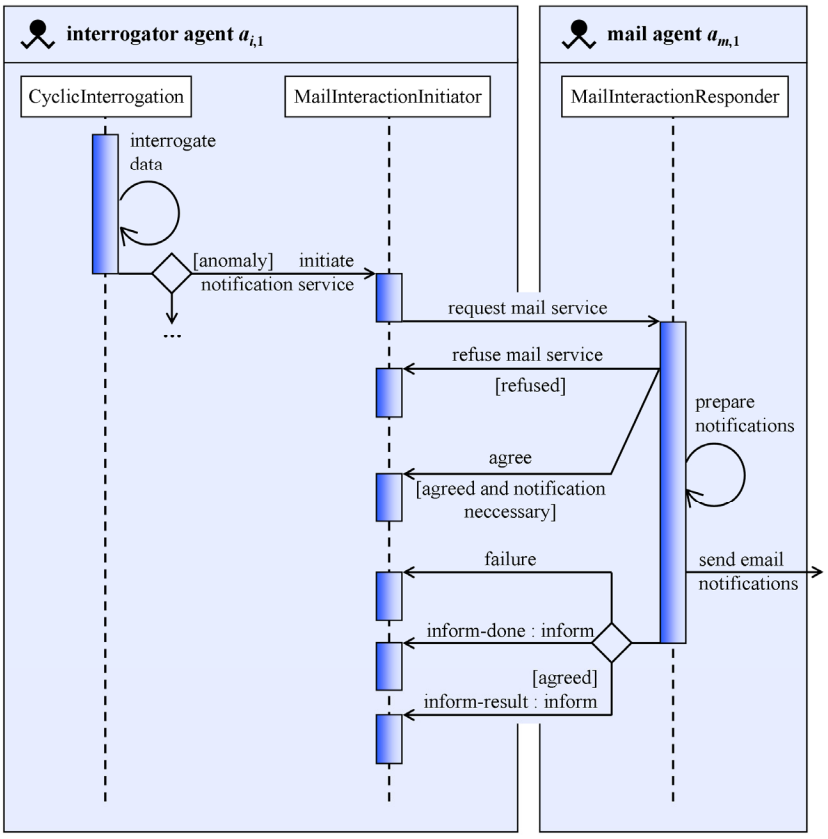


Fig. 9 Monitoring procedure executed in case of a detected anomaly

To ensure extensibility and interoperability, the multi-agent system *MAS* is implemented in compliance with the specifications issued by the Foundation for Intelligent Physical Agents (FIPA). FIPA, the IEEE Computer Society standards organization for agents and multi-agent systems, promotes agent-based technology, interoperability of agent applications and the interoperability with other technologies (FIPA 2004, 2002a, b, c). As shown in Fig. 10, the FIPA specifications can be viewed in terms of different categories: Applications, abstract architecture, agent communication, agent management and agent message transport. These specifications are mapped to the monitoring agents, such as the interrogator agent for anomaly detection.

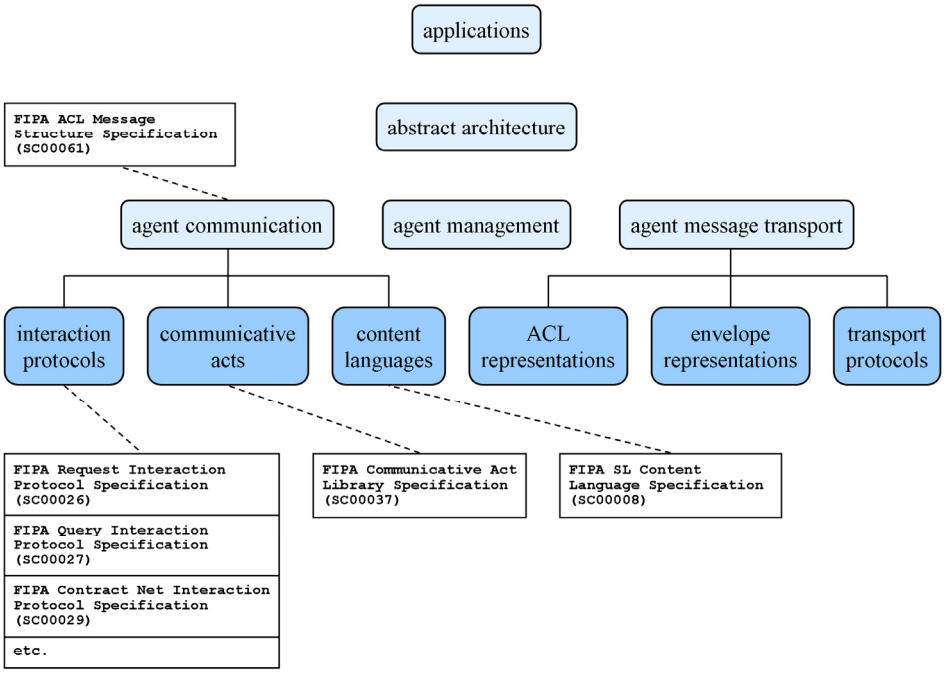
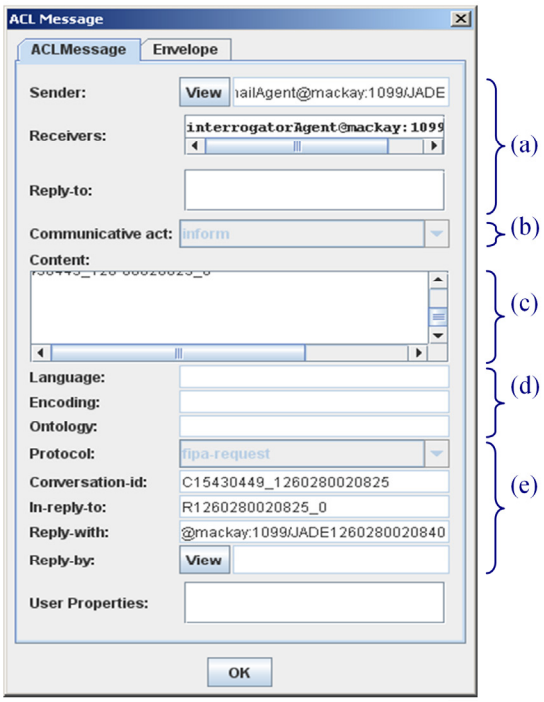


Fig. 10 FIPA specifications grouped by category (extract)

Besides ensuring extensibility and interoperability, adhering to the FIPA specifications provides considerable advantages with respect to performance and robustness of the implemented multi-agent system. The FIPA agent communication specifications, for example, ensure a common understanding of two or more communicating agents based on the FIPA Agent Communication Language, ACL (FIPA 2002d). FIPA ACL, used for the communication within the *MAS*, is based on speech act theory. In

essence, speech act theory treats communication as action. Thus, every software agent performs communicative actions in the same way like other actions. In order to achieve its goals and intentions, a monitoring agent uses performative verbs, corresponding to different types of communicative acts, to specify its communicative intentions and to describe the pragmatics of an ACL message that is sent to another agent. FIPA ACL provides an extensible library of communicative acts, also referred to as "performatives", such as "inform", "request", "refuse", "propose", etc. (FIPA 2002c).

Fig. 11 depicts an ACL message sent within the previously introduced agent interaction executed in case of a detected anomaly. The ACL message shown in Fig. 11 is composed and sent by the mail agent $a_{m,1}$ to inform $a_{i,1}$ that the human users have been notified about a detected anomaly. As defined in the FIPA ACL Message Structure Specification (s. FIPA 2002d), the ACL message includes detailed information on the participants in the communication (such as sender and receiver) as well as type of communicative act, message content and conversation controls (e.g. protocol, conversation id, etc.).



FIPA parameter category
a) participants in communication
b) type of communicative act
c) message content
d) content description
e) conversation controls

Fig. 11 ACL message sent within the agent interaction

Validation of the SHM System Prototype

To assess the reliability and the practicability of the SHM system prototype, long-term field validation tests are conducted using the wind turbine (Fig. 12). One objective of this long-term instrumentation is to investigate the robustness, fault-tolerance and performance of the self-managing SHM system implemented using a multi-agent-based approach. The hardware and (agent-based) software subsystems of the SHM system prototype are implemented and installed at spatially distributed locations. Different system setups have been tested with respect to the locations of the software subsystems. In the illustrated system setup, the hardware and software subsystems are installed at the following locations:

- The on-site hardware subsystem (including sensors, data acquisition units and computer) is installed with the wind turbine located in Dortmund (Germany).

- The real-time monitoring subsystem and the centralized MySQL database system reside at the Institute for Computational Engineering at the Ruhr-University Bochum (Germany).

- The on-demand multi-agent system *MAS* comprising the agent systems AS_{main} and AS_1 is distributedly located at different institutes:

    - The interrogator agent $a_{i,1}$ and the mail agent $a_{m,1}$ ($a_{i,1}$, $a_{m,1} \in AS_1$) for detecting malfunctions and generating alerts are located at Stanford University (USA).

    - The email server (i.e. the SMTP host), used by the agent $a_{m,1}$ at Stanford University, as well as the main agent system (AS_{main}) are situated at the Ruhr-University Bochum.

The following discussion focuses on the self-detection of system malfunctions for temperature monitoring and data acquisition.

Fig. 12 reference structure

Remote Monitoring of Temperature Data Acquisition

System and component malfunctions, such as breakdowns of sensors, temporary unavailability of a data acquisition unit (DAU) or a disconnected server system, occur often in a real-time SHM system and could cause the loss of large amounts of valuable monitoring data. For example, a malfunction on the temperature DAUs (which collect data at a sampling rate of 1 Hz from 10 temperature sensors) undetected for a week would cause the loss of more than 6,000,000 temperature readings. The value and reliability of a monitoring system can be substantially hampered if such malfunctions are not detected early.

For the temperature DAUs, malfunctions, such as abnormal measurements outside certain temperature ranges, are implicitly indicated by anomalies within the data sets collected by the units themselves. One potential malfunction, or cause for alert, for a DAU is characterized by a large number of identical

measurements: The last normally collected value (sensed before the occurrence of the malfunction) is stored repeatedly by the DAU substituting the subsequent temperature values. To detect such an anomaly, the interrogator agent $a_{i,1}$ at certain time intervals extracts and analyzes the temperature data sets stored in the centralized MySQL database (Smarsly *et al.* 2004). For this purpose, a set of implemented configuration files is available in the multi-agent system as shown in Fig. 13. The configuration file `analysis.shm`, used by $a_{i,1}$, specifies interrogation parameters, database URL, database driver, sensor specification, analysis details, etc. Furthermore, the configuration file `config.shm` defines general system information such as scheduled interrogation intervals. Here, the data interrogation with respect to DAU malfunctions is executed twice a day. The configuration file `mail.shm` specifies the parameters necessary to launch the email service when a malfunction is detected; these parameters include, for example, the email addresses of the authorized recipients and the address of the SMTP host utilized by the mail agent $a_{m,1}$.

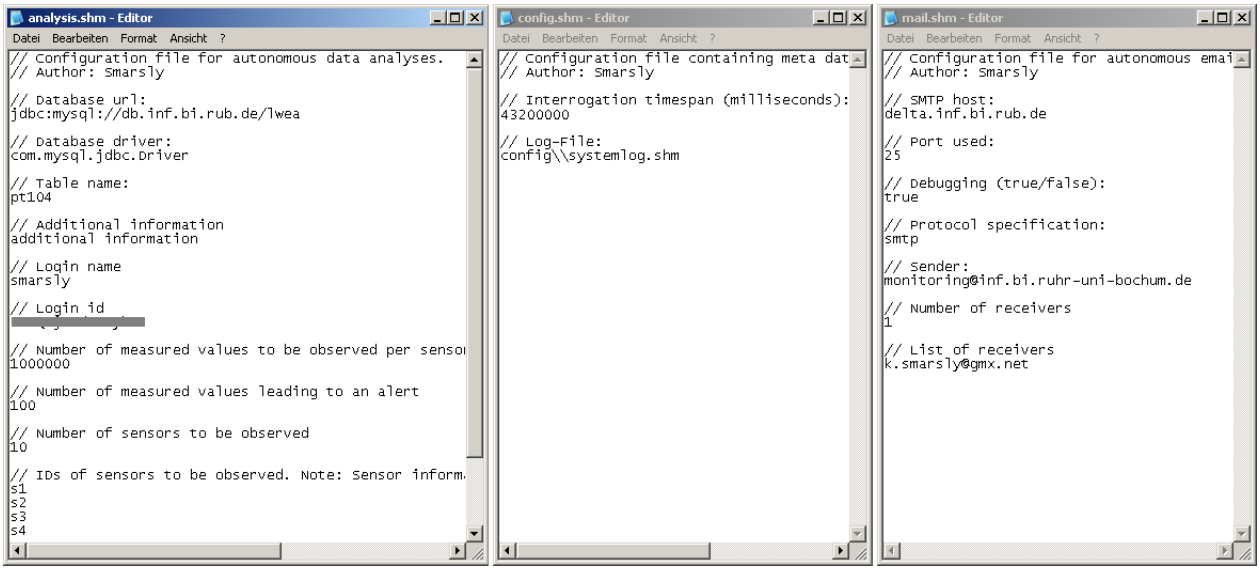


Fig. 13 Configuration files applied for data interrogation, system configuration and email notification

The monitoring sequence proceeds according to the implemented agent interactions as defined in the interaction protocol shown in Fig. 9. The interrogator agent $a_{i,1}$ scans the collected data sets, based on

experience, for 100 identical values as specified in the configuration file `analysis.shm`. Fig. 14 shows a detected anomaly from the data sets collected from the temperature DAUs. The anomaly is characterized by a large number of identical measurements, which, as a result of a DAU-internal system crash, are repeatedly stored by the DAU instead of regular measurements. The anomaly was observed by the interrogator agent in January, 2010, and has led to an undesirable interruption of the automated data acquisition process. As shown in Fig. 14, the anomaly can be visualized through a dynamically generated web interface connection and the corresponding numerical values are retrieved using the database connection that are both provided by the real-time monitoring subsystem.

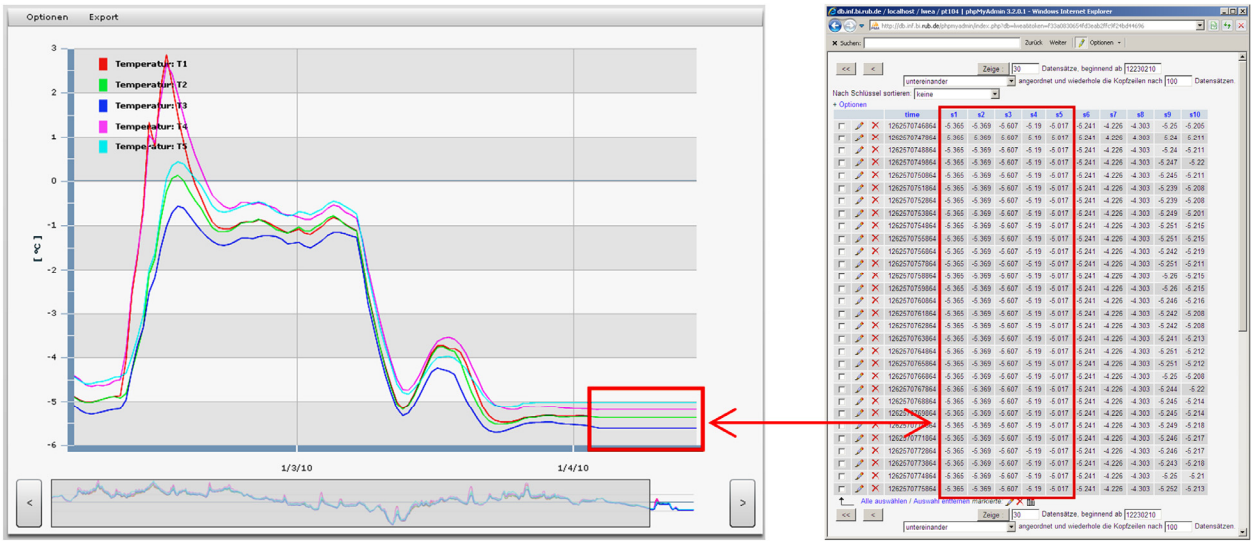


Fig. 14 Graphical and numerical representation of a detected anomaly using the web interface (left, source: Hartmann and Höffer 2010, modified) and the database connection (right)

Once a malfunction is detected, the SHM system prototype generates an alert to inform the authorized users immediately about the anomaly so that appropriate measures can be taken in order to avoid the loss of valuable monitoring data. Consequently, the interrogator agent $a_{i,1}$ requests the mail agent $a_{m,1}$ to inform the human users via email about the malfunction discovered in the temperature acquisition process. Fig. 15 documents the interaction between the participating agents, which is conducted to accomplish the agent-based email notification cooperatively. It should be noted that, although the mail agent $a_{m,1}$ and the

interrogator agent $a_{i,1}$ are the main actors, additional information (such as agent identifiers, agent services provided, etc.) are supplied by the Directory Facilitator agent in the main agent system which possesses the management information. As a result, $a_{m,1}$ generates an email to submit it to the recipients specified in the `mail.shm` configuration file. The email content includes detailed information on the detected malfunction and is generated using the analysis results provided by $a_{i,1}$. The email header is assembled based on the information taken from the configuration file as shown in the excerpt of the email displayed in Fig. 16.

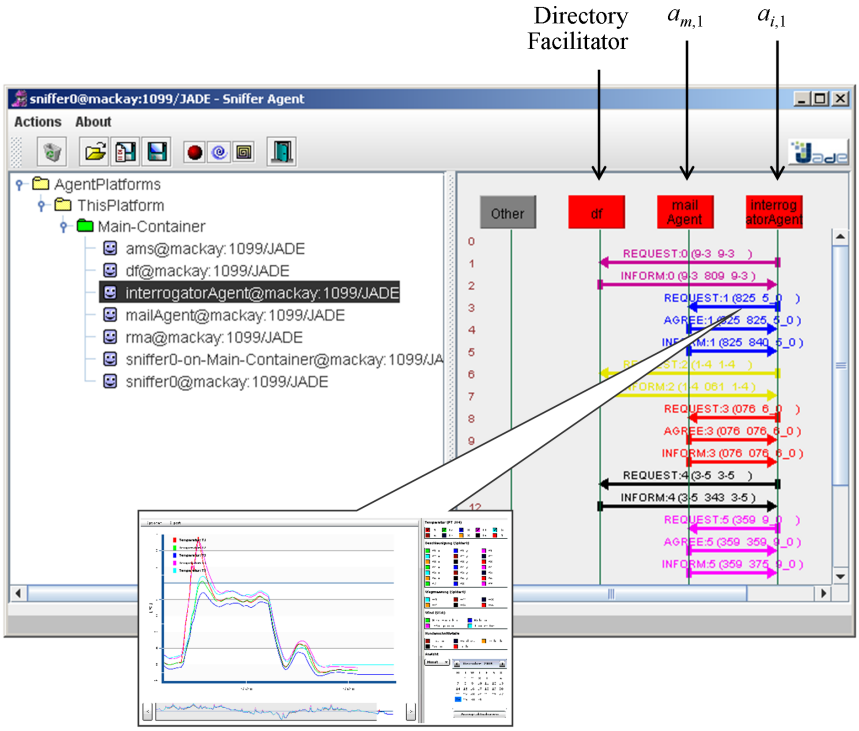


Fig. 15 Agent interactions within the email notification procedure

Von: monitoring@inf.bi.ruhr-uni-bochum.de
An: k.smarsly@gmx.net, stefan.lachmann@rub.de, f.hegemann@web.de
Betreff: [Generated Email] Wind Turbine Safety Report
Datum: 04.01.2010 12:46:57

This email has autonomously been generated and was sent by a software agent. Do not reply.

This type of message is sent by the 'InterrogatorAgent'
i.) if anomalies within the temperature acquisition process have been detected or
ii.) if the multi-agent system has succesfully been launched.
The following agent message has been received. Content:

Analysis properties:
Database driver: com.mysql.jdbc.Driver
Database table: pt104
Additional specifications: -
Login name: smarsly
Login ID: [redacted]
Number of values observed per sensor: 1000000
Number of significant values: 100
Sensors analyzed: s1;s2;s3;s4;s5;s6;s7;s8;s9;s10

Anomaly detected:
Database: lwea
Table: pt104
Date: Mon Jan 04 02:45:19 CET 2010
Sensor: s1
Last value measured: -5.365

…

Properties:
IP address: 134.147.216.69
Internet host: tunsql
Hap: tunsql:1099/JADE
Responsible agent: mailAgent@tunsql:1099/JADE
Agent container: tunsql:1099/JADE
Container state: Ready(4)
Agent platform: tunsql:1099/JADE
Mail host: delta.inf.bi.rub.de
Port: 25
Debugging: true
Protocol: smtp
Sender: monitoring@inf.bi.ruhr-uni-bochum.de
Number of receivers: 3
Local mail configuration file: config\mail.shm

Fig. 16 Autonomously generated email for agent-based email notification

Discussion on the Validation Test Results

The SHM system prototype and the multi-agent monitoring system have been installed and are in service since December, 2009, continuously monitoring the operation of the wind turbine. Since then, the prototype has been able to detect all the malfunctions that occurred. In particular the multi-agent approach

has proven to be imperative, because not only the exceptional events (such as power blackouts) can be detected, but also regular events are taken into account (such as software updates of the operating system that requires reboot of the computer and, hence, temporary termination of all software applications). For all these events, the SHM system prototype, incorporated with self-management functions, is able to reconfigure and restart itself.

The malfunctioning of the commercial temperature DAUs described, which was mainly caused by the cold winter 2009-2010 in Europe (–20°C and less), was one of the many examples. For each self-detected malfunction, the software agents were able to autonomously notify the users via email. As for the example illustrated, the affected DAUs were remotely restarted by the responsible technicians early to prevent the loss of a large amount of valuable measured data. All monitoring tasks performed by the SHM system prototype are permanently logged by the software agents involved. Fig. 17 shows an example of a log file that has dynamically been generated by the supervisor agent $a_{s,main}$. These information could be useful for future optimizations of the system functionalities and for enhancing the total system performance.

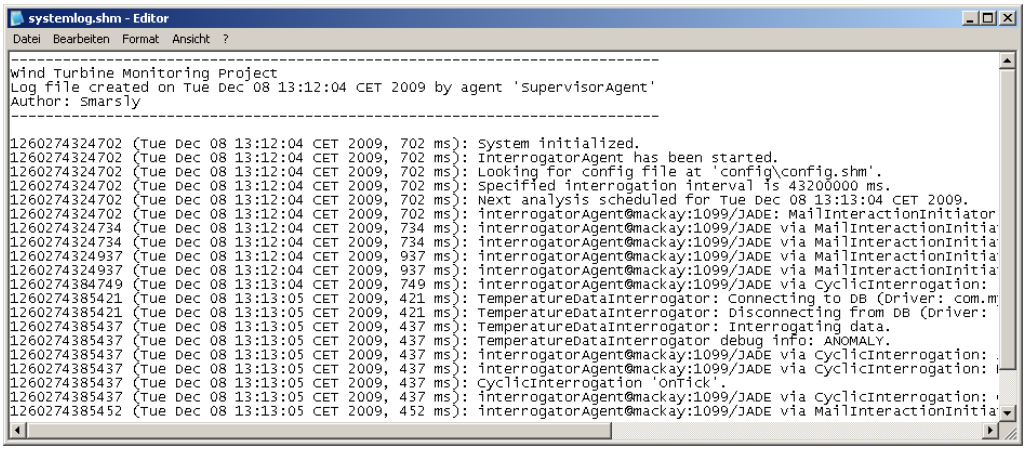


Fig. 17 Log file generated by the supervisor agent (illustrating the starting sequence of the system)

In addition to validating the system functionalities, laboratory tests have been conducted to evaluate the system performance. One critical aspect of a SHM system is its ability to handle and to process a large amount of monitoring data. Following the example scenario on the temperature data collected from 10 different temperature sensors installed in the wind turbine, the time consumed for interrogating the data can be taken as an indicator for the system performance. Different quantities of secondary monitoring data, taken from the database system, have been analyzed. Various sizes of data packets, ranging from 1,500 and 1,250,000 temperature measurements (corresponding to 150 to 125,000 analyzed measured values per sensor), have been analyzed by the interrogator agent $a_{i,1}$. For a sampling rate of 1 Hz, the packets represent the data sets collected during the timeframes from about 150 seconds (smallest packet) to about 35 hours (largest packet).

Fig. 18 shows the amount of time for the agent $a_{i,1}$ to interrogate the temperature data sets. An important aspect for the interpretation of the results plotted in the figure is that the times needed for autonomous execution of all required agent-oriented tasks, such as automated preparation of ACL messages, addressing, remote connecting, etc., are included. Analyzing a medium-sized packet (say 900,000 values representing measurement data collected from the structure during 25 hours) takes about 1.7 seconds, which is equivalent to approximately 525,000 measurements analyzed per each second. As shown in Fig. 18, the number of analyzed monitoring data scales almost linearly with respect to time, indicating that the quantity of processed data does not affect the performance of the agent-based SHM system prototype significantly. However, for extremely large data packets (more than 1,100,000 values analyzed at once) and extremely small data packets (less than 5,000 values), the performance of the interrogator agent scales nonlinearly. For small size data packets, the time consumed for preparing and executing the agent-oriented tasks strongly affects the performance. With respect to large data packets, the performance deterioration can possibly attribute to the underlying JADE infrastructure and the Java language dealing with resource management and consumption.

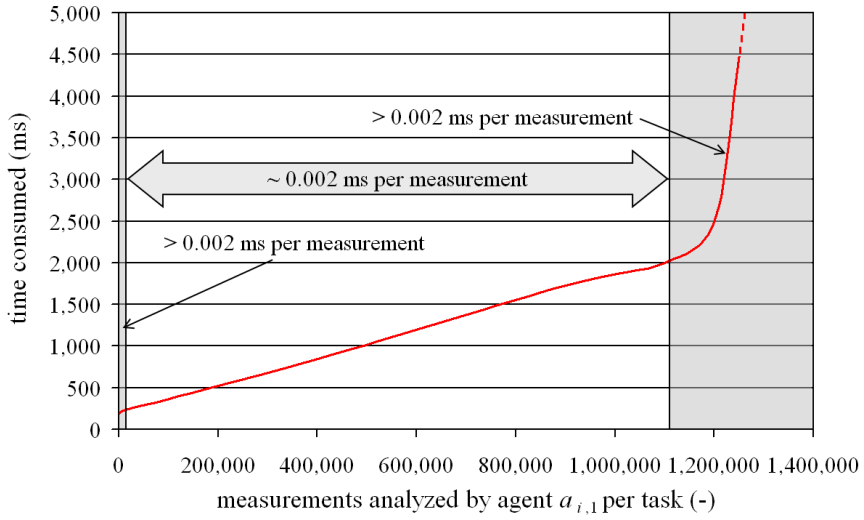


Fig. 18 Performance tests of the SHM system prototype

Summary and Conclusions

The emerging engineering problems resulting from ageing, altered requirements, excessive loading or inadequate maintenance of civil infrastructures underpin the urgent need for reliable and cost-effective monitoring systems. In this paper, a new monitoring concept has been introduced that synergistically couples multi-agent technology and advanced computing methods for structural health monitoring. Design and implementation of the proposed concept have been illustrated by means of a system prototype for the decentralized real-time monitoring of a wind turbine. In order to assess the reliability and the practicability of the prototype, performance tests in the laboratory as well as long-term validation on the wind turbine have been conducted. In particular, the robustness, fault-tolerance and performance of a multi-agent system, as an integral part of the prototype, have been examined focusing on the self-detection of anomalies and system malfunctions in detail.

The validation tests have demonstrated that the multi-agent system is capable of reliably detecting anomalies and system malfunctions such as sensor breakdowns or temporarily unavailable resources.

When having detected a system malfunction, the multi-agent system autonomously informs the responsible individuals participating in the monitoring activities via agent-based email notifications. Without such self-diagnostic and self-management capabilities as in current conventional monitoring systems, large amounts of valuable monitoring data can be lost. The multi-agent system is kept in operation since December, 2009, continuously monitoring the wind turbine. In total, the results from the validation tests conducted have corroborated the capability and reliability of the multi-agent system for autonomous structural health monitoring.

In summary, this research has demonstrated that multi-agent technology can successfully be transferred from the laboratory into monitoring practice for long-term deployment under real-world conditions. Because of its modularity and extendibility, multi-agent technology is a promising paradigm for the implementation of structural health monitoring systems, allowing continuing upgrades and new advancements to be incorporated in an autonomous distributed manner. Although great success has been encountered in the present study, the SHM system prototype can further be improved in a number of areas. For example, additional field tests can help to improve the functionalities of the prototype with respect to the algorithms embedded into the software agents for anomaly detection. Intelligent wireless sensor nodes and mobile agent technologies can be introduced to advance the performance of the prototype. As a result, a more comprehensive decentralization and a collaboration of the sensor nodes can be achieved, leading to the emergence of a higher level of collective group behavior of the sensor nodes. Accordingly, research efforts are already underway exploring the feasibility of evoking collective group behavior, also referred to as “collective intelligence”, for achieving more flexible, holistic and fault-tolerant monitoring systems for civil infrastructure.

Acknowledgments

The authors gratefully acknowledge the financial support of the German Research Foundation (DFG) through the grants SM 281/1-1 and HA 1463/20-1. This research is also partially supported by the National Science Foundation (grant CMMI-0824977).

References

ASCE – American Society of Civil Engineers (2009), "2009 Report Card for America's Infrastructure", [online]. Available at: http://www.infrastructurereportcard.org [Accessed March 25, 2010].

Bellifemine, F., Caire, G., Pogg, A. and Rimassa, G. (2003), "JADE - A White Paper", *EXP Online*, 3(3), 6-19.

Bellifemine, F., Caire, G. and Greenwood, D. (2004), *Developing Multi-Agent Systems with JADE*, John Wiley & Sons, Ltd, West Sussex, UK.

Bellifemine, F., Caire, G., Trucco T. and Rimassa, G. (2007), *Jade Programmer's Guide*, [online]. Available at: http://jade.tilab.com/doc/programmersguide.pdf [Accessed March 31, 2010].

Castors, M. (2008), "PDI Performance tuning check-list", *Technical Report*, Pentaho Corporation, Orlando, FL, USA.

Chang, P.C., Flatau, A. and Liu, S.C. (2003), "Review Paper: Health Monitoring of Civil Infrastructure", *Structural Health Monitoring*, **2**(3), 257-267.

DWIA – Danish Wind Industry Association (2010), "Wind Energy Economics", [online]. Available at: http://www.talentfactory.dk/en/tour/econ/economic.htm [Accessed May 3, 2010].

Elgamal, A., Conte, J.P., Masri, S., Fraser, M., Fountain, T., Gupta, A., Trivedi, M. and El Zarki, M. (2003a), "Health monitoring framework for bridges and civil infrastructure", *Proceedings of The 4th International Workshop on Structural Health Monitoring*, Stanford, CA, USA, September.

Elgamal, A., Conte, J.P., Fraser, M., Masri, S., Fountain, T., Gupta, A., Trivedi, M. and El Zarki, M. (2003b), "Health Monitoring for Civil Infrastructure", *Proceedings of The 9th Arab Structural Engineering Conference (9ASEC)*, Abu Dhabi, United Arab Emirates, November.

EWEA – The European Wind Energy Association (2010), "Operation and Maintenance Costs of Wind Generated Power", [online]. Available at: http://www.wind-energy-the-facts.org [Accessed May 3, 2010].

Farrar, C.R., Allen, D.W., Ball, S., Masquelier, M.P. and Park, G. (2005), "Coupling sensing hardware with data interrogation software for structural health monitoring", *Proceedings of the 6th International Symposium of Dynamic Problems of Mechanics*, Ouro Preto, Brazil, February.

Ferber, J. (1999), *Multi-Agent Systems: An Introduction to Distributed Artificial Intelligence*, Addison-Wesley, London, UK.

Foundation for Intelligent Physical Agents – FIPA (2002a), SC00001:2002 *FIPA Abstract Architecture Specification*, Alameda, CA, USA.

Foundation for Intelligent Physical Agents – FIPA (2002b), SC00026:2002 *FIPA Request Interaction Protocol Specification*, Alameda, CA, USA.

Foundation for Intelligent Physical Agents – FIPA (2002c), SC00037:2002 *FIPA Communicative Act Library Specification*, Alameda, CA, USA.

Foundation for Intelligent Physical Agents – FIPA (2002d), SC00061:2002 *FIPA ACL Message Structure Specification*, Alameda, CA, USA.

Foundation for Intelligent Physical Agents – FIPA (2004), SC00023K:2004 *FIPA Agent Management Specification*, Alameda, CA, USA.

Glaser, S., Li, H., Wang, M., Ou, J. and Lynch, J.P. (2007), "Sensor Technology Innovation for Advancement of Structural Health Monitoring: A Strategic Program Plan of US-China Research for the Next Decade", *Smart Structures & Systems*, **3**(2), 221-244.

Hartmann, D. and Höffer, R. (2010), "*Lifespan assessment of wind energy converters through system identification (Lebensdauerabschätzung von Windenergieanlagen mit fortlaufend durch Systemidentifikation aktualisierten numerischen Modellen)*", research project funded by the German Research Foundation (DFG) through the research grant HA 1463/20-1, Ruhr-University Bochum, Bochum, Germany.

IEC - International Electrotechnical Commission (2006), *IEC standard series 61400-25 (wind power communication standard),* Geneva 20, Switzerland.

Johansen, K. (2008), "Standardising plant communication", *Wind Power Technology, EIA Bilingual Magazine*, 2008-2009, 20-22.

Johnsson, A. and Højholt, L.E. (2007), "Use of IEC 61400-25 to secure access to key O&M data", *Proceedings of the European Offshore Wind conference 2007*, Berlin, Germany, December.

Koh, C.G., Tee, K.F. and Quek, S.T. (2006), "Condensed model identification and recovery for structural damage assessment", *Journal of Structural Engineering*, **132**(12), 2018-2026.

Koh, C.G. and Perry, M. (2007), "Structural damage quantification by system identification", *Journal of Earthquake and Tsunami*, **1**(3), 211-231.

Lachmann, S., Baitsch, M., Hartmann, D. and Höffer, R. (2009), "Structural lifetime prediction for wind energy converters based on health monitoring and system identification", *Proceedings of The 5th European and African Conference on Wind Engineering*, Florence, Italy, July.

Law, K.H., Swartz, A.R., Lynch, J.P. and Wang, Y. (2008), "Wireless Sensing and Structural Control Strategies", *Proceedings of The Fourth International Workshop on Advanced Smart Materials and Smart Structures Technologies*, Tokyo, Japan, June.

Law, K.H., Swartz, A., Lynch, J.P. and Wang, Y. (2009), "Decentralized Control Strategies with Wireless Sensing and Actuation", *Proceedings of the 2009 NSF CMMI Research and Innovation Conference*, Honolulu, HI, USA, June.

Liu, S.-C., Tomizuka, M. and Ulsoy, G. (2006), "Strategic issues in sensors and smart structures", *Structural Control and Health Monitoring*, **13**(6), 946 – 957.

Lynch, J.P., Law, K.H., Kiremidjian, A.S., Kenny, T.W. and Carryer, E. (2002), "A wireless modular monitoring system for civil structures", *Proceedings of The 20th International Modal Analysis Conference – IMAC*, Los Angeles, CA, USA, February.

Lynch, J.P. (2005), "Design of a wireless active sensing unit for localized structural health monitoring", *Structural Control and Health Monitoring*, **12**, 405–423.

Lynch, J.P., Wang, Y., Law, K.H., Yi, J.H., Lee, C.G. and Yun, C.B. (2005), "Validation of large-scale wireless structural monitoring system on the Geumdang Bridge", *Proceedings of The 9th International Conference on Structural Safety and Reliability*, Rome, Italy, June.

Metek GmbH (2010), Fritz-Strassmann-Str. 4, 25337 Elmshorn, Germany.

PCB Piezotronics, Inc. (2010), 3425 Walden Avenue, Depew, NY, USA.

Quecedo, E.S.T., Canales, I., Villate, J.L., Robles, E. and Apnanz, S. (2007), "The use of IEC 61400-25 standard to integrate wind power plants into the control of power system stability", *Proceedings of the European Wind Energy Conference and Exhibition*, Milan, Italy, May.

Roldan, M.C. (2009), "Pentaho Data Integration (Kettle) Tutorial", *Technical Report*, Pentaho Corporation, Orlando, FL, USA.

Sachs, L. and Hedderich, J. (2009), *Angewandte Statistik*, 13th ed., Springer-Verlag GmbH, Heidelberg, Germany.

Smarsly, K., Mittrup, I., Bilek, J. and Bloch, M. (2004), "Implementation of a Software Agent for the Administration of Complex Data in Civil Engineering" (in German), *Proceedings of the Forum Bauinformatik 2004*, Braunschweig, Germany, September.

Smarsly, K. and Hartmann, D. (2007), "Artificial Intelligence in Structural Health Monitoring", *Proceedings of the SEMC 2007 – Third International Conference on Structural Engineering, Mechanics and Computation*, Cape Town, South Africa, September.

Smarsly, K., Lehner, K. and Hartmann, D. (2007), "Structural Health Monitoring based on Artificial Intelligence Techniques", *Proceedings of the ASCE Workshop on Computing in Civil Engineering 2007*, Pittsburgh, PA, USA, July.

Smarsly, K. (2008), "Autonomous Monitoring of Safety-Relevant Civil Engineering Structures", *Doctoral Thesis (in German)*, Ruhr-University Bochum, Bochum, Germany.

Smarsly, K. and Hartmann, D. (2009a), "AMBOS - A self-managing system for monitoring civil engineering structures", *Proceedings of the 16th EG-ICE Conference of the European Group for Intelligent Computing in Engineering*, Berlin, Germany, July.

Smarsly, K. and Hartmann, D. (2009b), "Real-time monitoring of wind energy converters based on software agents", *Proceedings of The 18th International Conference on the Applications of Computer Science and Mathematics in Architecture and Civil Engineering*, Weimar, Germany, July.

Smarsly, K. and Hartmann, D. (2009c), "Multi-scale monitoring of wind energy plants based on agent technology", *Proceedings of The 7th International Workshop on Structural Health Monitoring*, Stanford, CA, USA, September.

Smarsly, K. (2010), "An Autonomous Computing Approach towards Monitoring of Civil Engineering Structures", *Asian Journal of Civil Engineering (Building and Housing)*, **11**(2), 149-163.

Smarsly, K. and Hartmann, D. (2010), "Agent-Oriented Development of Hybrid Wind Turbine Monitoring Systems", *Proceedings of the XVII. Workshop on Intelligent Computing in Engineering*, Nottingham, UK, June.

Sohn, H., Farrar, C.R., Hunter, N.F. and Worden, K. (2001), "Structural health monitoring using statistical pattern recognition techniques", *Journal of Dynamic Systems, Measurement, and Control*, **123**, 706-711.

Sohn, H., Park, H.W., Law, K.H. and Farrar, C.R. (2004), "Minimizing Misclassification of Damage using Extreme Value Statistics", *US-Korea Joint Seminar/Workshop on Smart Structures Technologies*, Seoul, Korea, September.

Spencer, B.F., Ruiz-Sandoval, M.E. and Kurata, N. (2004), "Smart sensing technology: opportunities and challenges", *Structural Control and Health Monitoring* **11**(4), 349-368.

Spencer, B.F., Nagayama, T., Sim, S.H. and Miyamori Y. (2007), "Issues in SHM employing smart sensors", *Journal of Smart Structures and Systems*, **3**(3), 299-320.

Straser, E.G., Kiremidjian, A.S., Meng, T.H. and Redlefsen, L. (1998), "Modular, wireless network platform for monitoring structures", *Proceedings of The 16th International Modal Analysis Conference – IMAC*, Santa Barbara, CA, USA, February.

Wang, Y., Lynch, J.P. and Law, K.H. (2005a), "Wireless structural sensors using reliable communication protocols for data acquisition and interrogation", *Proceedings of The 23rd International Modal Analysis Conference – IMAC*, Orlando, FL, USA, January.

Wang, Y., Lynch, J.P. and Law, K.H., (2005b), "Design of a low-power wireless structural monitoring system for collaborative computational algorithms", *Proceedings of The SPIE 10th Annual International Symposium on Nondestructive Evaluation for Health Monitoring and Diagnostics*, San Diego, CA, USA, March.

Zhou, L.L. and Yan, G. (2006), "HHT method for system identification and damage detection: an experimental study", *Smart Structures and Systems*, **2**(2), 141–154.