11th International Conference on Hydroinformatics
HIC 2014, New York City, USA

# A NEW ALARM GENERATION CONCEPT FOR WATER DISTRIBUTION NETWORKS BASED ON MACHINE LEARNING ALGORITHMS

CHRISTIAN KÜHNERT (1), THOMAS BERNARD (1), IDEL MONTALVO (2), REIK NITSCHE (3)
*(1): Fraunhofer Institute of Optronics, System Technologies and Image Exploitation (IOSB), Fraunhoferstraße 1, 76131 Karlsruhe, Germany*
*(2): 3Sconsult,76137 Karlsruhe,Germany*
*(3): TZW: DVGW-Technologiezentrum Wasser, 01326 Dresden, Germany*

Water Distribution Networks are complex systems containing a large amount of sensors placed in the network. An important task of water quality sensors is to give information to the operators if a contamination has occurred in the network. Generally, the overall analysis of the sensor data is performed manually by the operators since data-driven alarm generation systems for protecting a network in real time are not established at water utilities. This has several reasons: At first, the parameterization of these modules is often very complex and time consuming. Secondly, in most cases these systems generate too many false positive alarms due to special operational actions like sensor calibration or flushing of pipes.
In this paper an approach is presented which addresses both problems. First, a self-configuring alarm generation module is proposed which only needs a few parameters to be set. Next, using the module, it is shown that the amount of false positive alarms can be reduced, if known events are used for training the module. This approach is proved in experiments performed on a laboratory plant.

## INTRODUCTION

Water Distribution Networks (WDNs) are critical infrastructures that are exposed to deliberate or accidental chemical, biological or radioactive contamination. During the last years powerful multi-parameter sensors for water quality monitoring in WDNs have been developed in order to monitor the water quality. These sensors measure several physical and chemical water parameters like conductivity, pH, free chlorine, redox potential, diffusion, and turbidity.

### Aim of the alarm generation module

The aim of the alarm generation module is the online monitoring of the water distribution network. On the basis of measured historical sensor data, machine learning approaches can be applied to automatically generate a model for monitoring the water quality and quantity. The advantage of this approach is that no analytically-formulated expertise is needed a priori, and thus the user is not hampered by "unsafe" assumptions about the physical / biochemical behavior of the drinking water.
An alarm is generated, if a *novelty* occurs in the acquired measurements. In that case measurements can describe one station in the WDN which covers several sensors or sensors

placed in the network itself. In both contexts, a novelty is defined as a detected unknown state of the WDN. Unknown states can be detected using data from the water quantity as well as water quality. The output of the module is a continuous alarm index value. This value quantifies the difference from the trained model to the acquired measurement data. If the alarm index passes a certain threshold, an alarm is detected.
In the past, several approaches have been investigated and implemented. In [7] an event detection software called CANARY is proposed which contains several statistically based algorithms. The company whitewater has developed a software tool called BlueBox [10], which can be used to perform an enhanced data analysis on water quality and quantity data. Other approaches are presented in [4] by using Support Vector Machines and in [1] by using genetic algorithms. Still, all proposed approaches have not be widely applied in WDNs. Two possible reasons for that are:

- The parameterization of an alarm generation software is complex and time consuming. Appropriate parameters for the machine learning algorithm need to be selected. Additionally, the selection of the alarm threshold is not an easy task.
- A lot of abnormalities detected by the software in the data are due to special operational actions. These leads to false alarms, reducing the credibility of the module. Examples of operational actions which lead to false alarms are sensor calibrations, flushing of pipes or rapid changes of water quality due to mixing of different water resources.

In this paper a new approach is presented which reduces the impact of both problems. The development is part of the project SMaRT-OnlineWDN [8]. The main objective of this project is the development of an online security management toolkit for WDNs that is based on sensor measurements of water quality as well as water quantity.
The paper is structured in two parts:
In the first part, the methodology of the alarm generation module is explained. This covers the preprocessing step, the calculation of the alarm index and the calculation of the alarm threshold. In the second part, results of experiments performed on a laboratory plant are presented. The paper closes with a short summary and proposals for future research.

## METHODOLOGY OF THE ALARM GENERATION MODULE

The proposed alarm generation module covers several steps. Initially, a normalization of the measurements is performed. As multivariate statistical method the principal component analysis (PCA) is used. The PCA is applied to the normalized measurements. In the following sections, it is explained in detail how the alarm index and finally the generation of the alarm threshold are obtained.

### Normalization

Since the event detection module works with different types of quality and quantity parameters (e.g. pressure, conductivity) the measurements need to be normalized. For normalization of the data, the z-score normalization [5] is used, which sets mean value of the measurements to zero and the standard deviation to one. The z-score normalization is defined as

$$z[k] = \frac{x[k] - \mu}{\sigma} \tag{1}$$

while $x[k]$ with $k = 1 \dots n$ is the original measurement data of one process variable. The variable $\mu$ describes its mean value and $\sigma$ the standard deviation. The original set of process variables is defined as $X = \{x_1[k], x_2[k], \dots, x_m[k]\}$, the normalized one as $Z = \{z_1[k], z_2[k], \dots, z_m[k]\}$,. The normalization is performed for each process variable in $X$.

**Principal Component Analysis**

The principal component analysis (PCA) is a procedure of multivariate statistics to structure large data sets. It is usually used as a data mining method [5] or for model reduction [9]. Applications of the PCA in terms of process supervision can be found for example in [3] and [6].

The main concept of a PCA means to perform an orthogonal transformation to map the set of correlated variables into a set of linear, uncorrelated ones. The PCA assumes that the acquired measurements are stationary Gaussian distributed random variables. The resulting uncorrelated variables are then called the *principal components* of the variable set. Mathematically, the principal components then cover the variance accounted for in the data set. This means that only a small amount of principal components needs to be used for monitoring. Usually, the first two principal components are used to describe the state trajectory of the system. Further information on how the PCA can be used for process supervision can be found e.g. in [3].

The calculation of the principal components is carried out by computing the eigenvectors of the covariance matrix. The covariance matrix $\Sigma \epsilon\ R^{m \times m}$ is defined as

$$\Sigma = \begin{bmatrix} \sigma_{11}^2 & \sigma_{12}^2 & \cdots & \sigma_{1m}^2 \\ \sigma_{12}^2 & \sigma_{22}^2 & \dots & \sigma_{2m}^2 \\ \vdots & \vdots & \ddots & \vdots \\ \sigma_{1m}^2 & \sigma_{2m}^2 & \dots & \sigma_{mm}^2 \end{bmatrix} \tag{2}$$

with $\sigma_{ij}^2$ being the covariance of the two standardized variables $x_i[k]$ and $x_j[k]$ in the variable set. Next, the eigenvalues $\lambda$ of the covariance matrix are calculated and sorted in ascending order. This results in the final diagonal matrix $\Lambda \epsilon\ R^{m \times m}$:

$$\Lambda = \begin{bmatrix} \lambda_1 & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & \lambda_m \end{bmatrix} \text{ with } \lambda_1 \geq \cdots \geq \lambda_m. \tag{3}$$

In a next step, the corresponding eigenvectors of the eigenvalue matrix $\Lambda$ are calculated and summarized in columns. This results in the matrix $\Gamma \epsilon\ R^{m \times m}$

$$\Gamma = \begin{bmatrix} \gamma_{11} & \gamma_{12} & \cdots & \gamma_{1m} \\ \gamma_{21} & \gamma_{22} & \cdots & \gamma_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ \gamma_{m1} & \gamma_{m2} & \cdots & \lambda_{mm} \end{bmatrix} \tag{4}$$

Finally, the matrix $\Gamma$ is used to perform the linear transformation

$$Z \rightarrow Y = \Gamma^T Z \tag{5}$$

while $Y$ contains the principal components. For example, $y_1[k] = \gamma_{11}\ x_1[k] + \cdots + \gamma_{m1} x_m[k]$ is the time series of the first principal component when having new acquired measurements. For the calculation of the alarm index, which is described more detailed in the next subsection. The first two principal components are used since this usually covers most of the variance accounted for in the data set.

**Stages of the Event detection**

To check if an event has occurred, in the measurements, three steps need to be performed. First the model has to be trained using offline data from the WDN. Next, the alarm index is calculated using online data. Finally, it has to be checked if the index passes a certain threshold.

*Training the model*

Initially, training data is selected, which defines the “normal state” of the WDN. For WDNs, in general, this selection is performed by experienced personnel of the water utilities. After

performing the normalization (1), this data is used to calculate the principal components given in equation (5). In a second step, new measurement data coming from the process devices and the calculated principal components, are used to generate the alarm index.

*Calculating alarm index*

The alarm index at sample $k$ is calculated with the measurement data $x^{new}[k] \epsilon R^m$. $x^{new}[k]$ is normalized by using the z-score $z^{new}[k] = \frac{x^{new}[k]-\mu}{\sigma}$ for each value in $x^{new}[k]$ with mean and standard deviation from the training data.

The alarm index is then calculated by taking the dot product between the selected number of principal components $\Gamma \epsilon R^{m \times l}$ with $1 \leq l \leq m$ and the data $Z_{new}[k] = \{z_1^{new}[k], \dots, z_m^{new}[k]\}$ given as:

$$\boldsymbol{Y_{new}}[k] = \Gamma^T Z_{new}[k] \tag{6}$$

$\boldsymbol{Y_{new}}$ is a vector with length *l*. Finally, the alarm index $Y_{new}^{Alarm} \epsilon R$ results from summing up the values in $Y_{new}$

$$Y_{new}^{Alarm}[k] = \sqrt{\boldsymbol{Y_{new}}[k] \cdot \boldsymbol{Y_{new}^T}[k]} \tag{7}$$

Since the alarm value $Y_{new}^{Alarm}$ is defined between $0 \leq Y_{new}^{Alarm} \leq \infty$ a dynamic threshold needs to be calculated that defines if an event has occurred or not.

*Calculation of the threshold*

The threshold is generated by calculating the alarm index using the training data, the selected number *l* of principal components and by calculating the threshold of the alarm index in terms of the variance from the training data. Therefore, the alarm index is calculated using the training data to generate $Y_{train}^{Alarm}$. This is performed by calculating $Y_{train}[k] = \Gamma^T Z[k]$ for each sample $k = [1, \dots n]$ in the training set which finally leads to the alarm index

$$Y_{train}^{Alarm}[k] = \sqrt{\boldsymbol{Y_{train}}[k] \cdot \boldsymbol{Y_{train}^T}[k]} \tag{8}$$

In a next step the threshold $Q_{threshold} \epsilon R$ for the alarm index is calculated as:

$$Q_{threshold} = K \cdot VAR(Y_{train}^{Alarm}) \tag{9}$$

In the following, the parameterization is set to $K = 6$. If $Y_{new}^{Alarm} > Q_{threshold}$ counts, an alarm is generated.

It is important to mention that this algorithm only needs two parameters to be set, namely the number of principal components *l* and the parameter *K* for calculating the threshold. Both parameters are statistically interpretable values.

## EXPERIMENTAL RESULTS

### Laboratory plant

The developed methodology has been tested on a laboratory water distribution network. Two configurations with different outflow conditions have been investigated. The two configurations are sketched in Figure 1: Configuration 1 and 2 of the plant to perform experiments. Once the outlet Out8 is used as outflow, once the outflow Out6.. In this installation of the laboratory network five conductivity sensors, two flow meters and one pressure sensor are installed. To simulate a contamination, an injection including a dosing pump is installed at the inflow. The pump injects NaCl solution into the network, leading to an increase of conductivity, being measured by the conductivity sensors. The two configurations of the laboratory network can be characterized as follows:

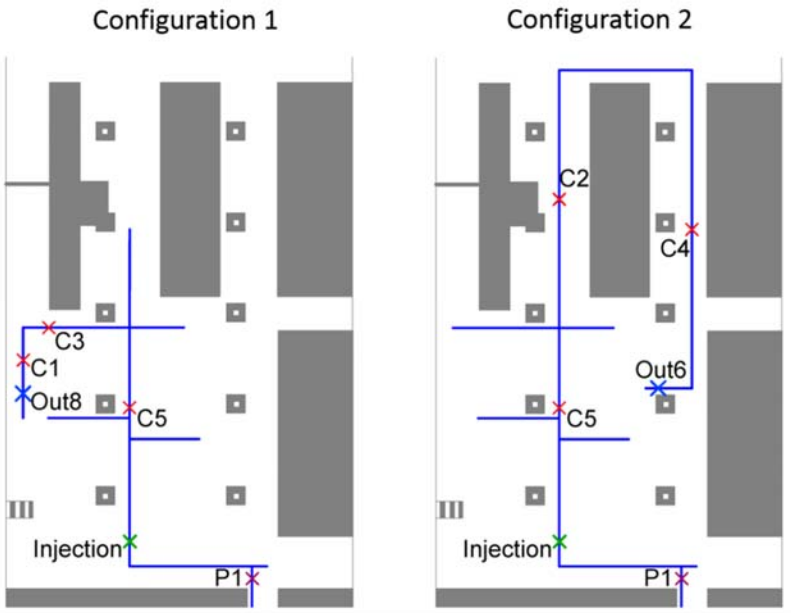


Figure 1: Configuration 1 and 2 of the plant to perform experiments. Once the outlet Out8 is used as outflow, once the outflow Out6.

*Configuration 1*
In configuration 1 the stop cock at the outflow Out6 is closed and at outflow Out8 is open while passing the conductivity sensors C1, C3 and C5. The conductivity sensors C2 and C4 do not measure an increase of conductivity if NaCL is injected since the flow rate in that pipe section is zero. The acquired measurements with and without injected NaCl are shown in Figure 2.

*Configuration 2*
In configuration 2 the stop cock at the outflow Out8 is closed and open at outflow Out6. In that case, the water flow passes the conductivity sensors C2, C4 and C5. The sensors C1 and C3 do not measure an increase of conductivity after NaCL injection. The measurements with and without injected NaCl are given in Figure 3.

**Tested scenarios**
The acquired measurements of the two network configurations have been used to test the alarm generation module for different scenarios. At all, five different scenarios are investigated. The resulting alarm indices, including their calculated thresholds, are given in Figure 4. The scenarios are explained as follows:
***Injection Scenario 1*:**
Training data*: Clean water config. 1, clean water config. 2*
Test data*: Contaminated water config. 1*
This scenario covers the basic functioning of the module. Clean water is used for training, contaminated data for testing. For clean water the alarm index stays below the threshold, for contaminated water, the threshold is passed by large meaning that an event has be detected (see alarm index in figure 4, subplot 1).
***Injection Scenario 2*:**
Training data*: Clean water config. 1, clean water config. 2*
Test data*: Contaminated water config. 2*

This scenario is similar to the first one. Only, that in this case contaminated water from configuration 2 is used. Like in the first scenario, the contamination is detected by the module (see alarm index in figure 4, subplot 2).

***Injection Scenario 3*:**

Training data*: Clean water config. 1, clean water config. 2, contaminated water config. 1*

Test data*: Contaminated water config. 1*

In this scenario the contaminated water from configuration 1 is used for training and testing the module. The thought of this scenario is to test if it is possible to train specific states of a system and store it in the module as being normal. The resulting alarm index (figure 4, subplot 3) shows, that the contaminated data leads to a small increase of the alarm index, but stays below the threshold. Hence, no event is raised by the module.

***Injection Scenario 4*:**

Training data*: Clean water config. 1, clean water config. 2, contaminated water config. 1*

Test data*: Contaminated water config. 2*

Scenario 3 showed that specific states of a system can be stored in the module. This scenario checks if it is still possible to detect other events coming from other system configurations. The resulting alarm index in figure 4, subplot 4, indicates that this is still possible. Since the contaminated water from configuration 2 was not used for training, the module still detects the event. The alarm index passes the calculated threshold by large.

***Injection Scenario 5*:**

Training data*: Clean water config. 1, clean water config. 2, contaminated water config. 1, contaminated water config. 2*

Test data*: Contaminated water config. 1, contaminated water config. 2*

In this scenario both the training data and test data contain information from contaminations in configuration 1 and configuration 2. Due to that, no events are detected in the test data (see alarm index in figure 4, subplot 5). This indicates that it is possible to store several different patterns of a system in the module that are then declared as a normal state of the WDN.

## SUMMARY AND FUTURE WORK

In this paper, an easy-to-parameterize event detection algorithm based on principal component analysis has been presented. This algorithm only needs two parameters to be set, namely the number of principal components and the sigma-environment for calculating the threshold. Both parameters are statistically interpretable values.

Next, several experiments at a laboratory network were performed to check the performance of the developed module. It has been tested if different process patterns can be stored in the module as being normal process behavior. Therefore, data containing clean and contaminated water from the laboratory network for two configurations were used. The results showed that for these two configurations of the plant, it is possible to learn patterns as normal states of the WDN.

Future work will focus on the construction of a complete diagnosis chain. This covers the step to localize the source of the contamination in the WDN after an event has occurred and a classification of the event. Furthermore, the another focus will be on the optimization of the selected threshold. Additionally, the extension of the PCA using different kernel methods needs to be checked. Possibly, kernel methods lead to better results, since they are a nonlinear methods.

## ACKNOWLEDGEMENTS

The project is supported by the German Federal Ministry of Education and Research (BMBF; reference project: 13N12180) and by the French Agence Nationale de la Recherche (ANR; reference project: ANR-11-SECU-006).

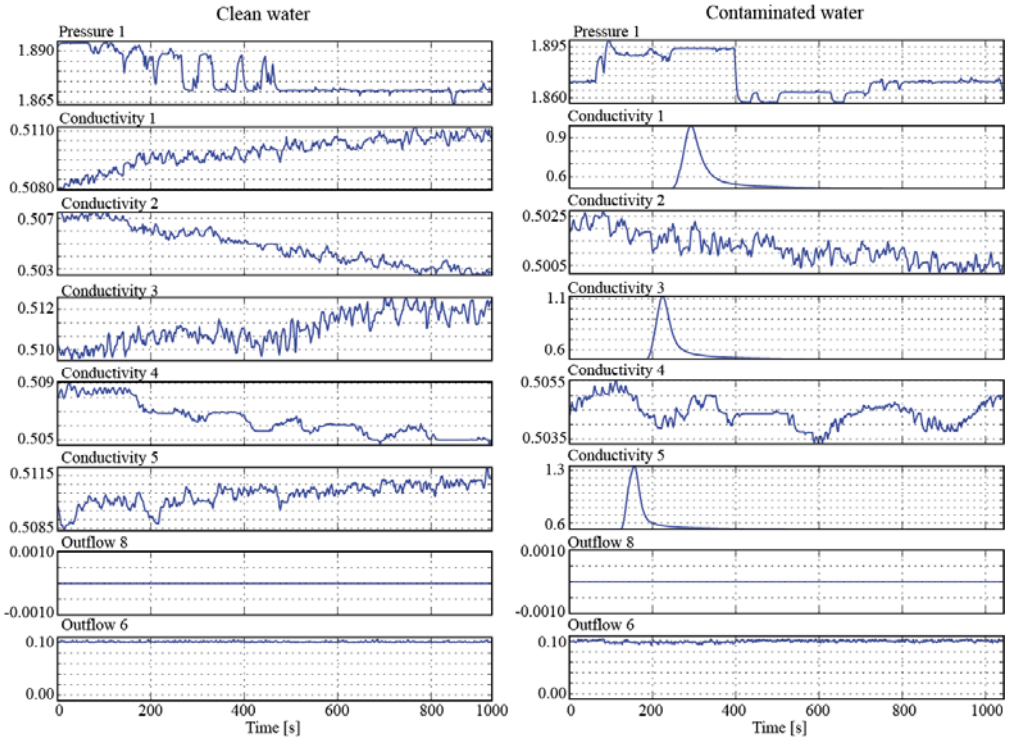


Figure 2: Measurements from configuration 1 of the laboratory plant.

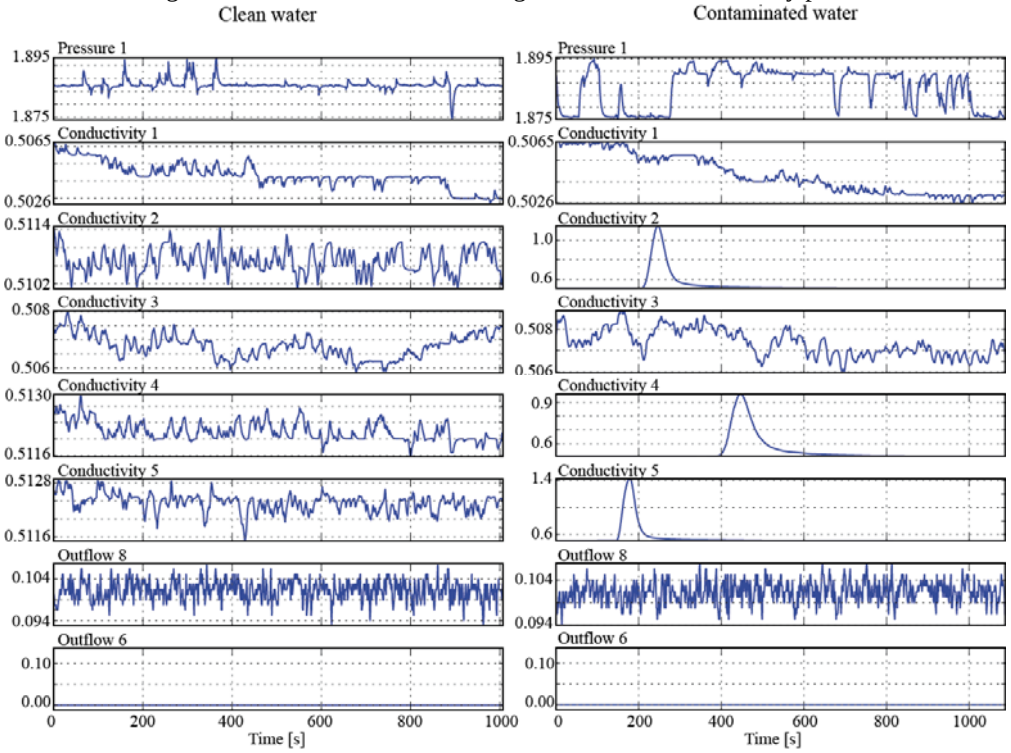


Figure 3: Measurements from configuration 2 of the laboratory plant.

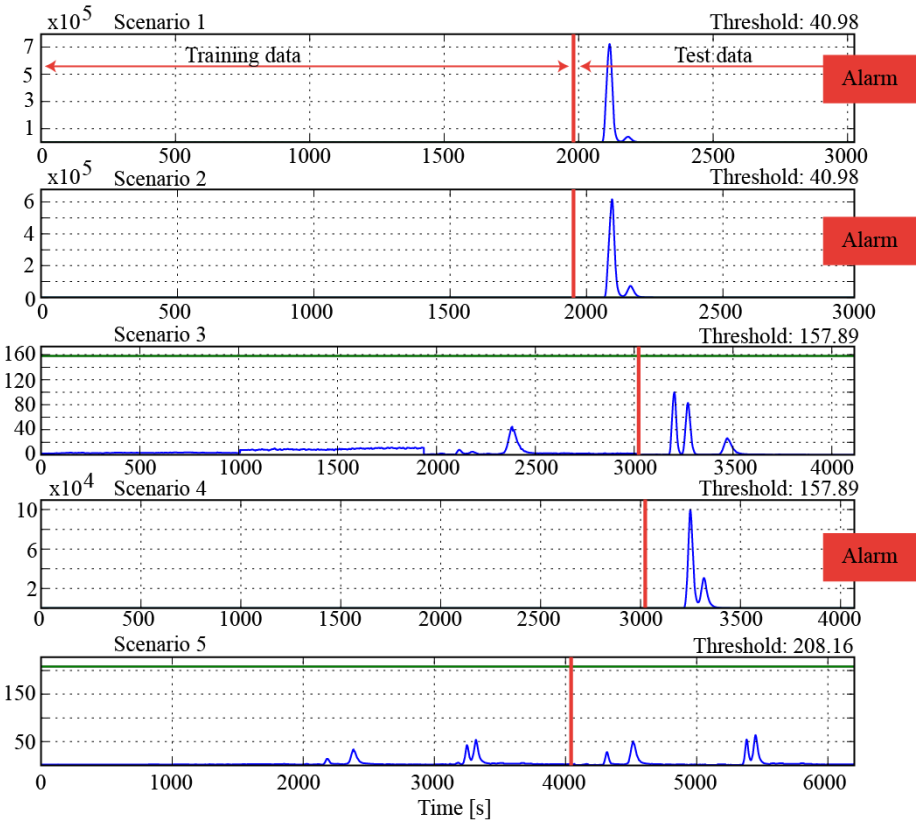


Figure 4: Alarm indices for the investigated scenarios.

**REFERENCES**


[1] Arad. J., Housh M., Perelman, L., Ostfeld A.: "Contamination event detection utilizing genetic algorithm", WDSA, Adelaid (2012)

[2] Grey R.M.: "An introduction to statistical signal processing". Stanford University, 2010

[3] Iserman, R.: "Fault-Diagnosis Systems", Springer (2006)

[4] Nurik, O., Ostfeld A.: "A weighted support vector machine classifier for contamination event detection in water distribution systems", WDSA, Adelaide (2012)

[5] Larose, D.: *"Data mining methods and models"* Wiley (2006)

[6] Mazur K., Borowa A., Brdys M.A.: "Condition monitoring using PCA based method and application to wastewater treatment plant operation", 1st IFAC Workshop on Applications of Large Scale Industrial Systems, Helsinki (2006)

[7] Murray R., et. al. : Contamination Warning Systems: Development, Testing, and Application of CANARY. EPA-600-R-10-036. U.S.

[8] Piller O., Gilbert D., Sedehizade, F., Lemoine C., Sandraz A., Werey C., Weber J., Deuerlein J., Korth A., Bernard T.: "SMaRT-Online$^{WDN}$: Online Security Management and Reliability Toolkit for Water Distribution Networks". WISG2013 Workshop Interdisciplinaire sur la Sécurité Globale (2013)

[9] van der Maaten L.J.P., Postma H.J., van den Herik H.J.: "Dimensionality Reduction: A Comparative Review". Technical Report TiCC (2009)

[10] Whitewater Bluebox ™; www.w-water.com (access on March 28, 2014)