Paper

# Monitoring of a Cloud-Based Environment for Resilient Telecommunication Services

Grzegorz Wilczewski

*Faculty of Electronics and Information Technology, Warsaw University of Technology, Warsaw, Poland*

**Abstract—This article depicts insights and in-depth presentation of a new tool, specifically designed for Data Center resources monitoring purpose. It enables physical and virtual resources monitoring and is capable of performing advanced analysis on the resulting, measured data. Here in presented are exemplary scenarios conducted over the proprietary Data Center unit, delivering specific information on the behavior of the analyzed environment. Presented results create a base layer for a high level resiliency analysis of telecommunication services.**

***Keywords—cloud computing, Data Center, resiliency, resources monitoring.***

## 1. Introduction

Nowadays, most telecommunication services are stated upon a concept of a virtualized, cloud-based functionalities serving contemporary telco products, for instance video streaming capabilities, storage services or big data processing functionalities [1]. Majority of workload volume is digested within Data Center units (DC), being it a vast computational power and massive storage spaces. Services being deployed utilizing such solutions require emerging but reliable systems overcoming possible flaws in the DCs design [2], [3]. Reliability of a cloud environment is the key feature that determines the success of a service being created within such environment [4]. Popular cloud-based services contribute to the XaaS model – *X as a Service*, where *X* defines what users can choose from the Cloud Service Provider (CSP) offer. Is it infrastructure, platform or application or many others, the common point is to deliver stable fundament for the overlaid services.

Creating reliable Cloud or DC environment requires multiplicity of features being supported by the discussed unit management system. One of the upmost importance characteristic of that system is to deliver a crude monitoring functionality. However the concept is not only limited to a basic parameters or resources observation but also requires appropriate examination of those gathered results. Active monitoring function has to cope with both hardware and software domains of the Data Center environment, moreover in both classification of the resources, namely, virtual and physical [5], [6]. The analysis of the current state of the Cloud unit delivers essential data for the CSP to act with. Whenever troubleshooting of a current problem is required or appropriate adjustment of a Service Level Agreement (SLA) is necessary, the monitoring application clarifies what element or node of the environment is responsible for such a state of a matter [7], [8]. Moreover, monitoring variety of parameters can be a successful approach towards isolation of specific virtual units whenever the assumption is that user's activities led to unstable or overloaded working conditions.

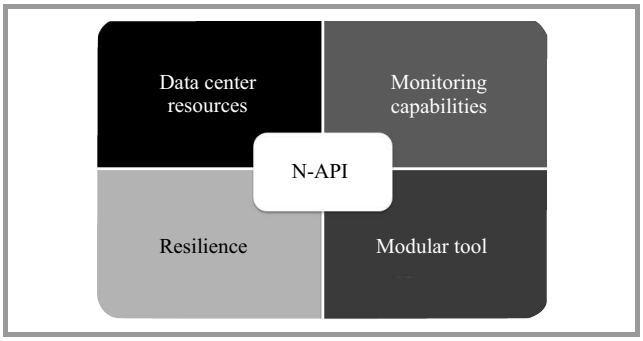


***Fig. 1.*** Functional stages of N-API project.

To overcome the aforementioned scenarios and to fulfill previously assessed functionalities the concept of an N-API project was defined. The following paper covers the insights towards design, creation and initial test-bed investigation of a tool in its native, cloud environment. The common goals of the project, presented within Fig. 1, define the following features of the N-API tool:

- an unified structure of N-API supporting networking Application Programming Interface (API) for the required functionalities,
- a definition and description of a set of resources and parameters of a DC (concerning processing capabilities, storage information, networking means and topology),
- a monitoring capability in all of the DCs structural layers (both virtual and physical),
- a modular structure of itself, supporting upcoming analysis expansion features and additional vendor-specific development,
- a resiliency evaluation module.

Table 1
N-API S2 REST functionalities – available methods

| Method | Parameters | Exemplary call | Returned | Comments |
|---|---|---|---|---|
| getVMS | None | http://localhost:8084/napi/analysis/getVMS | {"vmIds":[id1, id2, ..., idn]} | • Check the list of all available VMs of a DC |
| getCPU Analysis | vmId timePeriod | http://localhost:8084/napi/analysis/getCPUAnalysis?vmId=488&timePeriod=hourly | AnalysisResult Combined | • vmId has to be one of the vmIds elements set<br>• timePeriod has to be one of the following set: hourly, daily, weekly, monthly |
| getMemory Analysis | vmId timePeriod | http://localhost:8084/napi/analysis/getMemory Analysis?vmId=488&timePeriod=hourly | AnalysisResult Combined | • vmId has to be one of the vmIds elements set<br>• timePeriod has to be one of the following set: hourly, daily, weekly, monthly |
| getDisk Analysis | vmId TimePeriod | http://localhost:8084/napi/analysis/getDiskAnalysis ?vmId=488 timePeriod =hourly | AnalysisResult Combined | • vmId has to be one of the vmIds elements set<br>• timePeriod has to be one of the following set: hourly, daily, weekly, monthly |

Considering listed attributes of a tool being designed one has to divide the overall project into a three, complementary steps being realized:

- design and deployment of a functional API for DC resource monitoring,
- positioning of parameters being taken into consideration while evaluating measured unit characteristic (i.e., storage utilization, computational resources charge, hierarchy and topology of a Data Center),
- design and development of an application utilizing aforementioned modules.

## 2. Application Programming Interface

Approaching realization of the predefined project goals required access to the deployed Data Center environment. As a model one, the unit serving telecommunication services, manufactured by Cisco was selected. The management system being utilized within this unit was of the UCS Director software package. Thus, the designed N-API API is compatible with the mechanisms, functions and data model of the mentioned Cisco product line. Architecture of the designed API (presented on the Fig. 2) delivers essential information of how the tool is constructed and what are the possible utilization schemes. The programming interface of the N-API enables to retrieve essential information about the monitored DC unit, while API functions are used directly by the Application (APP) module to deliver raw and processed information about the environment (from the Analysis Extension section).

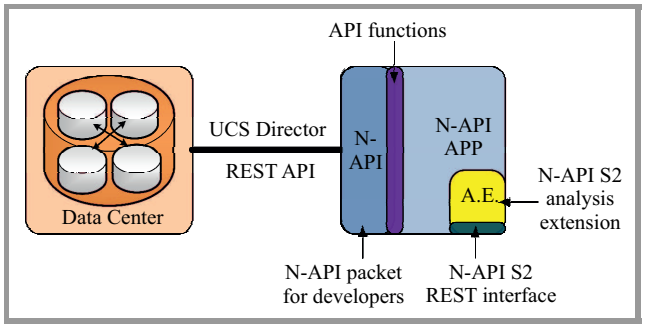


*Fig. 2.* Insights of the N-API architecture.

The N-API tool is purely designed as a networking, mobile software package, thus it is delivered in Java environment supporting data representation and manipulation by JSON (JavaScript Object Notation) and REST (Representational State Transfer) functionalities. In case of available connectivity schemes (depicted on the graph in Fig. 3) the interaction may be led in two approaches: either local regime or remote access. To support efficient security while accessing DC unit, appropriate secure API key is being utilized. Latter, a user can define specific connectivity socket (i.e., protocol, port, address) as well as establish a Joint Singleton interaction, what improves performance of an access towards monitored Data Center unit, deployed with use of the Cisco UCS package.

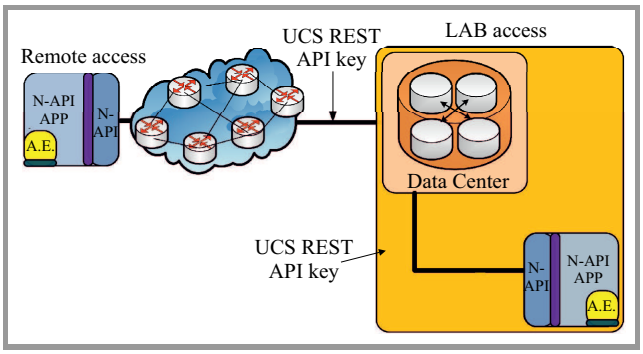


*Fig. 3.* Available connectivity schemes.

Monitored, raw data of the specified parameters is being delivered by a detailed description report files. Eligible entities possessing such a functionality are contained is the following context units (i.e., reflecting the hierarchy of the DC):

- *GlobalClient* – reveals the overall DC's structural outline, especially topology; is the highest order element in the hierarchical layout;

- *Cloud* – presents information concerning all of the Cloud units being realized in the Data Center; supports user with the intrinsic data about addressing, structure, contents and topology of the Cloud unit;
- *VirtualDataCenter* (vDC) – intermediate partition of a Cloud unit; enables layer isolation on the level required for IaaS and PaaS functionality support;
- *VirtualMachine* (VM) – is the lowest order entity in the hierarchy of the monitored Data Center unit; contributes to the basic SLA and billing handling; guarantees the lowest level of separation and isolation within the DC resources; initiates foundation for *SaaS* services;
- *HostNode* and *DataStore units* – deliver essential information about physical assets of the Data Center; presents configuration of selected peripherals and delivers network topology of the analyzed environment.

Table 2
N-API analysis results – returned response structures

| | |
|---|---|
| Analysis Result Combined | {“xyResults”:[ AnalysisResultXYChart ], “barResults”:[ AnalysisResultBarChart ], “pieResults”:[ AnalysisResultPieChart ], “singleResults”:[ AnalysisResultSingle ], “comment”:“comment”} |
| Analysis Result XYChart | {“series”:[{ “xValues”:[1.1,2.2,3.3 ...], “yValues”:[1.1,2.2,3.3 ...], “comment”:“comment”},...], “chartTitle”:“Title”,“units”:“range label”, “domainLabel”:“label”, “comment”:“comment”} |
| Analysis Result BarChart | {“series”:[{“values”:[{“xValue”: 1.1, “comment”:“comment”},...], “comment”:“seriesName”},... ], “chartTitle”:“Title”,“units”:“range label”, “domainAxisLabel”:“label”, “comment”:“comment”} |
| Analysis Result PieChart | {“values”:[{“xValue”:1.1,“comment”: “bar label”}, ...], “chartTitle”:“Title”,“units”: “range label”,“comment”:“comment”} |
| Analysis Result Single | {“xValue”:1.1,“comment”:“comment”} |

Nonetheless, architecture depicted on the Fig. 2 requires clarification on the REST functionalities presented in the Analysis Extension part of the N-API S2 module. Supported monitoring activities are expanded by the analysis module that supplies user or third party Cloud Manager with essential data elements/structures. The designed interface utilizes simple GET functionality and enables methods for specific inquiry. Parameters of those calls (i.e. VM identifiers) as far as an exact actions are presented in the positioning across Table 1. Resulting responses, given by means of JSON data modules are presented in the Table 2, accordingly. Whenever called action cannot be executed, i.e. due to user’s inappropriate parameter input, the server returns http 500 status.

## 3. Monitoring Application and Results Analysis

Previous sections presented the insights towards N-API’s utilization and its capabilities, thus herein are discussed functionalities of the application, being a complementary part of the N-API package (denoted across Figs. 2 and 3 by N-API APP phrase). To start with, it is a standalone ap-

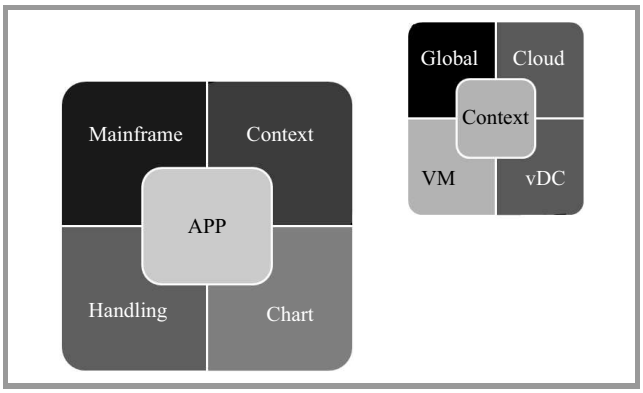


***Fig. 4.*** Functional modules of N-API application.

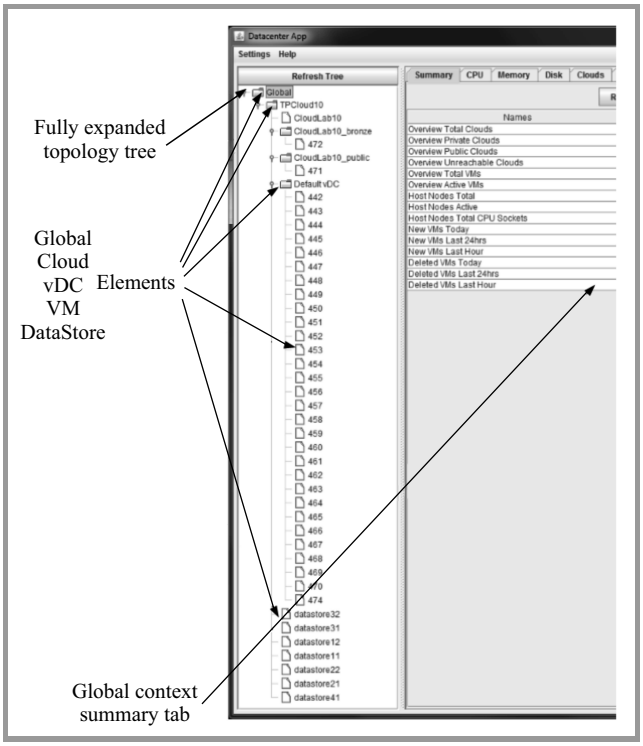


***Fig. 5.*** Topology tree – discovered hierarchy.

Table 3
Functionalities of the N-API Analysis Extension toolkit

| Tab | Tool/plot | Comment |
|---|---|---|
| Interpolation | Input values | Plot of the original values of the performance indicator. |
| | Input values;<br>Chunked grouped by counting;<br>Operation on chunk: median | Plot of the median value of the chunked samples from the original set of values. |
| | Input values;<br>loess interpolation;<br>Cut to max 100.0, min 0.0. | Plot of the interpolated values of the performance indicator. Interpolation method: LOcal regrESSion algorithm with the default configuration (please refer to the Apache Commons Math 3 v3.3 toolbox). Whenever interpolated values overshoot the boundary value, the cutoff below 0 and above 100 is applied. |
| Pie chart | Percentile containers | Pie chart represents the total time the VM's performance indicator spent in a specific percentile range. Containers represent intervals: 0–50%; 50–75%; 75–90%; 90–100%. |
| Resiliency monitoring | Color indicators | Visualization of the resiliency monitoring by means of distinctive color markers, representing total time spent in the appropriate percentile container. In case of a RED marker there is a Note informing about suggested activation of a chosen resiliency mechanism. |
| Bar chart | Minimum | Displays the minimum value from the analyzed set of values. |
| | Maximum | Presents the maximum value out of the analyzed sequence. |
| | Arithmetic mean | Calculates an arithmetic mean from the analyzed sequence |
| | Median | Calculates a median value from the analyzed set of values. |
| | Standard deviation | Calculates standard deviation value out of the considered set. |

plication what means it only requires a compatible runtime environment to be able to function properly and deliver support package for developing purposes. It has specific modules and plug-ins integrated within programming environment, alongside with the JFreeChart and Apache Commons Math 3 v3.3 toolboxes. The general structure of the N-API APP is presented on the Fig. 4. What is noticeable is the modular built that supports basic activities delivered by means of the API (Context Block) and enables handling

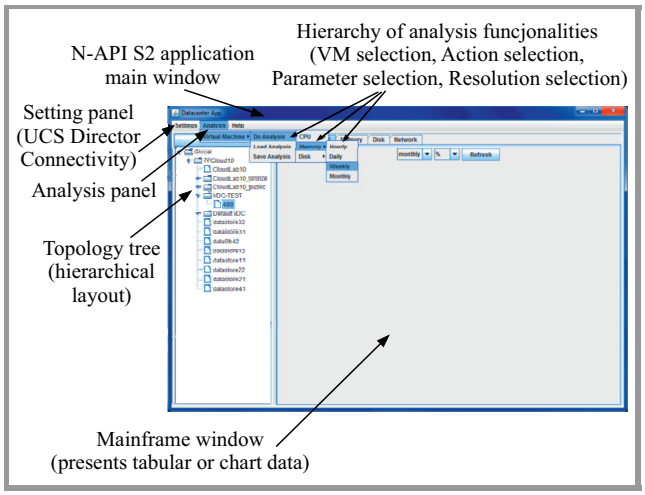


*Fig. 6.* Analysis Extension GUI.

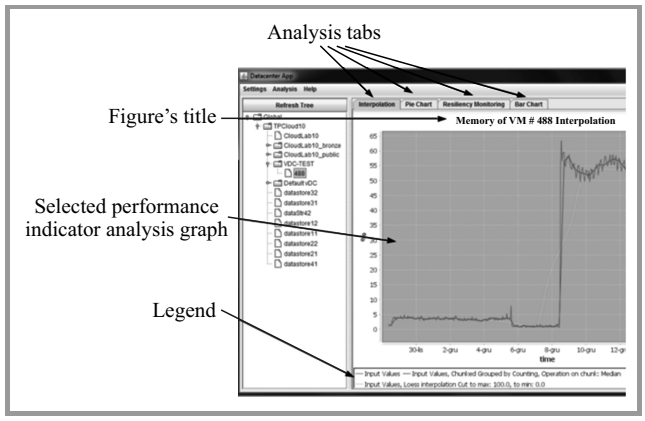


*Fig. 7.* Analysis of the selected VM's resources.

of the monitoring data (both raw and analyzed) in a form of an intuitive Graphic User Interface (GUI). Designed N-API application enables user to discover following information concerning monitored DC unit:

- topology and structure in a form of an expandable, active tree,
- status of the peripherals and inventory of the structural unit, i.e. configuration of the VM entity,
- monitoring data of the essential resources: CPU, RAM memory, storage space.

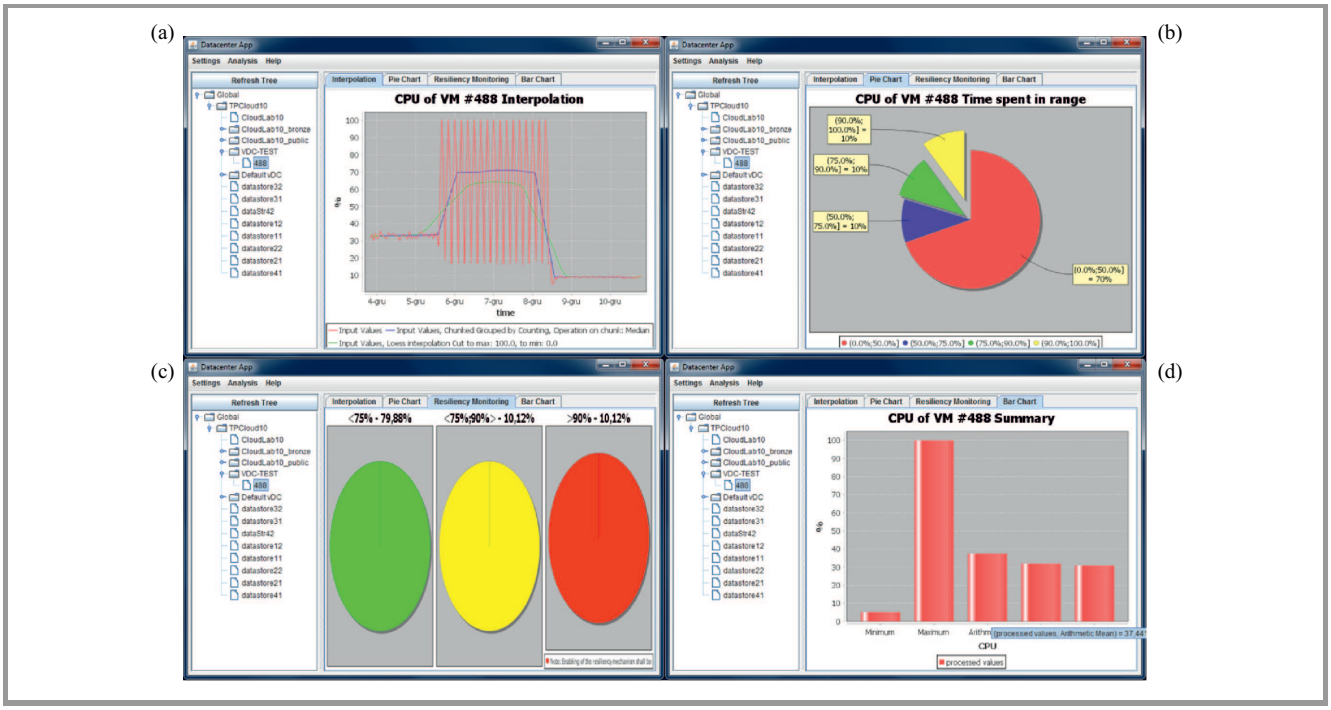


***Fig. 8.*** Resiliency evaluation tools of the N-API: Interpolation method (a), Pie chart analysis (b), Resiliency monitoring (c), Bar chart samples analysis (d).

Functionality-wise, the application delivers reporting capabilities for the export/process data in a range of selectable units (relative to the actual resource metric): CPU (MHz, in percent), RAM and Storage (GB, in percent). In case of defining temporal resolution of the report, following options are exposed: instant report (average value of parameter over the period of 5 min.), historical data (resolution: hourly, daily, weekly, monthly).

Concerning working with the executed application, the graphical representations of the selected states are delivered over the sequence from Fig. 5 to Fig. 7. Describing operation scheme of the application, one has to start with the activation of the connectivity module. After selecting appropriate socket and its configuration a secure link with the DC is being established. Whenever a successful connection is present the Context section is being activated and specific data obtained. Scrolling through the topology tree (Fig. 5) one can select a VM of choice and afterwards perform a set of analysis methods. In order to achieve that goal selection of the Analysis List from the top bar of the window is necessary. Mechanisms incorporated within Analysis Tab enable users to perform specific actions over the Virtual Machine unit, and those are: Do Analysis, Load Analysis, Save Analysis. The intermediate storage format of the data is coherent across the whole project and contributes to the .json format files. Furthermore, the application is capable of presenting the analysis of performance data for the following group of resources: CPU, RAM and Disk (Storage). Time resolutions are in coherence with instant and historical reports. Worth noticing is the fact of an abundant data being transferred whenever high resolution of samples across longest period is required. Positioning in the Table 3 covers the information about Analysis Extension module, while composed graphics in the Fig. 8 depict outcomes of all available and applied analysis tools.

## 4. Conclusions

Testing the designed API and application within its native, Data Center environment delivered a vast range of results. Selected scenarios covered various states of workload being applied to the controlled unit. Performed analyses of the overloaded resources delivered unique characteristics of monitored Cloud structure (depicted by means of Fig. 8). Deployed N-API tool enables CSPs to call the appropriate actions whenever troubleshooting or SLAs improvement are to be done. Advanced statistical analysis might improve resource allocation or introduce dynamically adjustable pools of resources for VMs. Finally, one ought to perceive the upmost importance of the resiliency analysis within a DC unit, in order to create a successful telecommunication service.

## References

[1] G. Wilczewski, "Analysis of content quality evaluation within 3DTV service distribution systems", *J. Telecommun. Inform. Technol. (JTIT)*, no. 1, pp. 16–20, 2014.

[2] G. Wilczewski, "Utilization of the software-defined networking approach in a model of a 3DTV service", *J. Telecommun. Inform. Technol. (JTIT)*, no. 1, pp. 32–36, 2015.

[3] S. Musman and S. Agbolosu-Amison, "A measurable definition of resiliency using mission risk as a metric", Mitre Tech. Rep., McLean, VA, 2014.

[4] Y. H. Khalil, A. Elmaghraby, and A. Kumar, "Evaluation of resilience for data center systems", in *Proc. IEEE Symp. Comp. Commun. ISCC'08*, Marrakech, Morocco, 2008, pp. 340–345.

[5] Riverbed Migration Mitigation, The Complete Data Center Consolidation Guide, Corporate Whitepaper, 2012.

[6] Cisco Data Center Solution Brief, Mitigate Risk for Data Center Network Migration, Corporate Whitepaper, 2014.

[7] R. Nasiri and S. Hosseini, "A case study for a novel framework for cloud testing", in *Proc. 11th Int. Conf. Electron., Comp. Comput. ICECCO 2014*, Abuja, Nigeria, 2014, pp. 52–56 (doi: 10.1109/ICECCO.2014.6997566)

[8] D. Agarwal and S. K. Prasad, "AzureBench: Benchmarking the storage services of the azure cloud platform", in *Proc. IEEE 26th Int. Parallel and Distrib. Process. Symp. Worksh. & PhD Forum IPDPSW 2012*, Shanghai, China, 2012, pp. 1048–1057.

**Grzegorz Wilczewski** received his B.Sc. and M.Sc. degrees in Electrical and Computer Engineering from Warsaw University of Technology, Poland, in 2009 and 2011, respectively. He is currently a Ph.D. candidate at Warsaw University of Technology. His research interests include 3DTV service quality monitoring, 3D imagery and digital signal processing.
E-mail: g.wilczewski@tele.pw.edu.pl
Faculty of Electronics and Information Technology
Warsaw University of Technology
Nowowiejska st 15/19
00-665 Warsaw