



A Serverless Real-Time Data Analytics Platform for Edge Computing

Stefan Nastic, Thomas Rausch, Ognjen Scekic, and Shahram Dustdar • *TU Wien*

Marjan Gusev, Bojana Koteska, Magdalena Kostoska, and Boro Jakimovski • *Ss. Cyril and Methodius University*

Sasko Ristov and Radu Prodan • *University of Innsbruck*

A novel approach implements cloud-supported, real-time data analytics in edge computing applications. The authors introduce their serverless edge-data analytics platform and application model and discuss their main design requirements and challenges, based on real-life healthcare use case scenarios.

With the increasing growth of the Internet of Things (IoT) and edge computing,¹⁻³ an abundance of geographically dispersed computing infrastructure and edge resources remain largely underused for data analytics applications. At the same time, the value of data becomes effectively lost at the edge by remaining inaccessible to the more powerful data analytics in the cloud due to networking costs, latency issues, and limited interoperability between edge devices. The reason for both of these shortcomings is that today's cloud models do not optimally support data analytics at the volume and variety of data originating from sensors and edge devices, typically characterized by high latencies and response times. There is a hard line between the edge and the cloud parts of analytics applications in terms of responsibilities, design, and runtime considerations. While contemporary solutions for cloud-supported, real-time data analytics mostly apply analytics techniques in a rigid bottom-up approach regardless of the data's origin,^{4,5} doing data analytics on the edge forces developers to resort to ad hoc solutions specifically tailored to the available infrastructure. The process is

largely manual, task-specific, and error-prone and usually requires good knowledge of the underlying infrastructure. Consequently, when faced with large-scale, heterogeneous resource pools, performing effective data analytics is difficult, if not impossible.

A promising approach to address these issues is the serverless computing paradigm. Serverless computing is an emerging cloud-based execution model in which user-defined functions are seamlessly and transparently hosted and managed by a distributed platform.⁶ There are multiple commercial and open source implementations of serverless platforms, such as Amazon Web Services Lambda (see <http://aws.amazon.com/lambda>), Apache OpenWhisk (<http://openwhisk.org>), or OpenLambda.⁶ The benefits of the serverless model become especially evident in the context of the described cloud and edge model, as both models seek to mitigate inefficient, error-prone, and costly infrastructure and application management.

In this article, we propose a unified cloud and edge data analytics platform, which extends the notion of serverless computing to the edge and facilitates joint programmatic resource and

analytics management. We introduce a reference architecture for our platform and a serverless application execution model, which enable uniform development and operation of analytics functions, thereby freeing users from worrying about the complexity of the underlying edge infrastructure. Finally, we outline a set of research challenges and design principles to serve as a road map toward a fully-fledged platform for cloud and edge real-time data analytics.

Use Case Scenarios

To better understand the need for distributed cloud and edge processing, we present two scenarios from IoT mobile healthcare (mHealth) and discuss holistic data analytics.

Use Case 1: Measuring Human Vital Signs in Disasters

In case of a major disaster, prompt paramedic attention is crucial to save people's lives. An extension of major disaster protocol includes wearable biosensors that can be attached to injured people by on-site medics, providing critical information about the patient's medical condition and helping determine a priority queue for further patient processing.

As soon as the sensors are attached, they start emitting data to nearby portable edge devices, such as a smartphone. Such devices perform only basic, lightweight analytics, such as triage decision tree, to determine the overall stability of the injured person and inform onsite medics. Once a network connection becomes available, selected data can be transferred to the cloud for more complex analysis that will predict and prioritize more precisely the patients' treatments, helping improve coordination between hospitals, optimize the time needed to reach a hospital, and provide timely information about patients' conditions.

Use Case 2: Measuring Human Vital Signs in Everyday Life

In this use case, a wearable biosensor measures a patient's vital signs during everyday life activities. Sensors continuously stream data of electrocardiogram readings to a nearby edge device that performs simple data analytics to monitor the patient's health condition. If an anomaly such as a heart failure is detected, the system immediately notifies medical emergency services. Preprocessed and filtered data are subsequently sent to the cloud, where comprehensive data analytics can be performed to gain better insight into the patient's overall condition, and help with diagnostics.

Holistic Data Analytics in mHealth

The presented use cases demonstrate the need for consolidating cloud- and edge-based data analytics techniques. To enable prompt reactions to a patient's changing health condition, low-latency algorithms should process the data at the edge in real-time. Conversely, detecting patterns in large amounts of historic patient data requires analytics techniques that depend on cloud storage and processing capabilities. In an architecture that combines cloud and edge data analytics, edge devices should act as a data gateway. Preprocessed and filtered data can be sent to the cloud, where they become persistent and highly available for compute-intensive analytics. To consolidate different techniques, and transparently handle data management, a holistic data analytics platform that unifies cloud and edge resources is needed.

Serverless Platform

Our main objective is to provide a full-stack platform for supporting real-time data analytics across cloud and edge in a uniform manner. The key role of the distributed cloud and

edge platform is to facilitate automated management of the underlying resource pool and optimal placement of analytics functions to support the envisioned serverless execution model. This approach enables combining the benefits of the edge (lower response time and heterogeneous data management) with the computational and storage capabilities of the cloud. For example, time-sensitive data, such as life-critical vital signs, can be analyzed at the edge, close to where data are generated instead of being transported to the cloud for processing. Alternatively, selected data can be forwarded to the cloud for further, more powerful analysis and long-term storage.

Platform Use and Architecture Overview

Figure 1a shows a high-level view of the platform and the main top-down control process (left) and bottom-up data management and delivery process (right). The proposed serverless data analytics paradigm is particularly suitable for managing different granularities of data analytics approaches bottom-up. This means that the edge focuses on local views (for example, per edge gateway), while the cloud supports global views, that is, combining and analyzing data from different edge devices, regions, or even domains. Data are collected from the underlying devices and delivered to the applications via consumption APIs. More importantly, the data analytics can be performed on edge nodes, cloud nodes, or both, and delivered from any of the nodes directly to the application, based on the desired view. Moreover, the top-down control process allows decoupling of application requirements (the what) from concrete realization of those requirements (the how). This allows developers to simply define the analytics function behavior and data-processing business logic and application goals (for example,

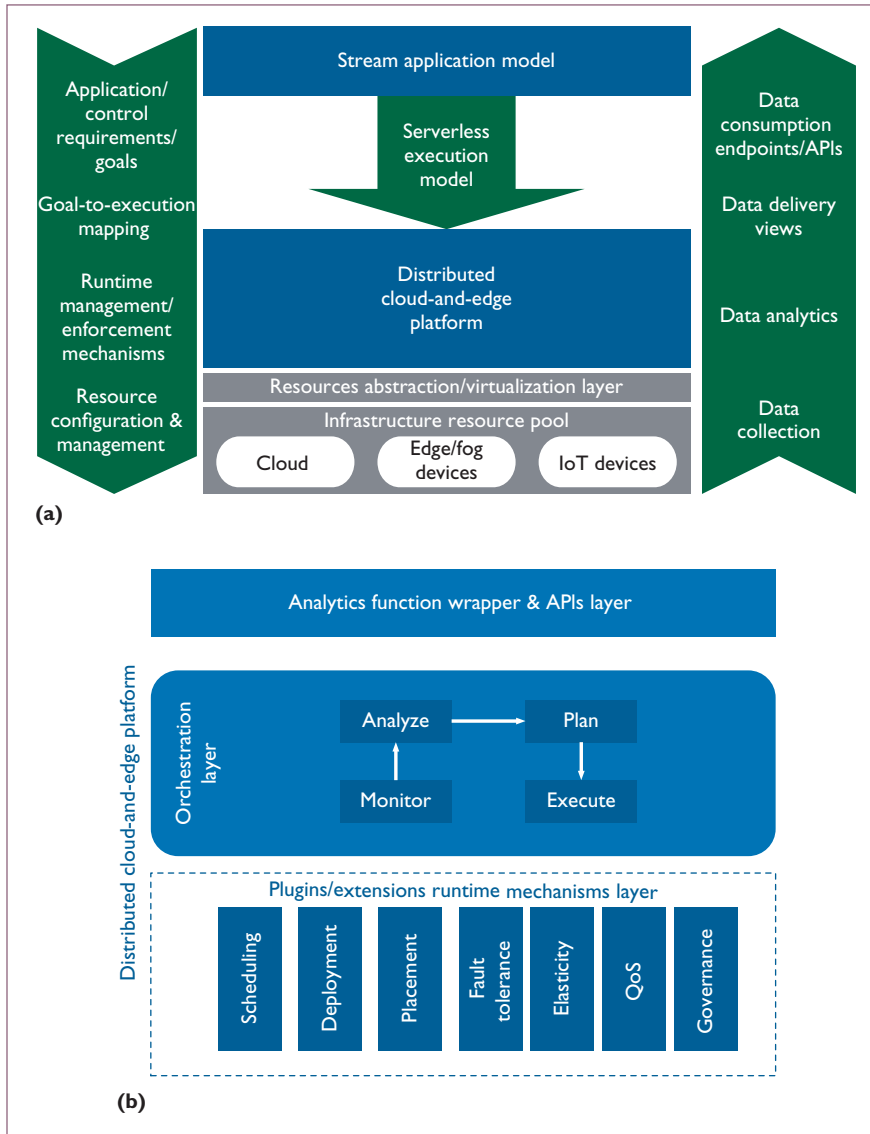


Figure 1. Cloud and edge real-time data analytics platform. (a) High-level usage context. (b) Internal software architecture.

regarding provisioning) instead of dealing with the complexity of different management, orchestration, and optimization processes.

Figure 1b shows the platform’s core architecture. We detail the layers of the architecture in the following.

The analytics function wrapper and APIs layer. This layer focuses on executing and managing user-provided data analytics functions – for example, delivering required data to the function and creating the

resulting endpoints. To this end, this layer wraps the analytics functions in executable artifacts such as Linux containers and relies on the underlying layers to perform concrete runtime actions and execution steps.

The orchestration layer. This layer interprets and executes user-defined real-time analytics functions, requirements, and configuration models. It acts as a gluing component, bringing together an application’s configuration model, user-defined

analytics functions, and the platform’s runtime mechanisms. Therefore, the orchestration layer receives the application configuration directives, in terms of high-level objectives such as optimizing network latency. It interprets and analyzes these goals and decides how to orchestrate the underlying resources, as well as the user-defined functions, by invoking the underlying runtime mechanisms. To this end, this layer contains micro (edge-based) and macro (cloud-based) orchestration and control loops. For example, it can use the scheduling and placement mechanisms to determine the most suitable node (cloud or edge) for an analytics function to reduce the network latency.

The runtime mechanisms layer.

This is an extensible plug-in layer, providing mechanisms to support executing the actions initiated by the orchestration layer. The deployment, scheduling, elasticity, and basic reasonable defaults for the quality of service (QoS) are core runtime mechanisms. More precisely, the platform determines the minimally required elastic resources, provisions them, deploys, and then schedules and executes analytics functions, which will satisfy QoS requirements. On the other hand, the governance, placement, fault tolerance, and extended QoS mechanisms are optional. For example, in some cases, the data could be confidential and some geographical regions should be excluded. Placing the functions closer to the data and deciding whether to use cloud or edge resources could improve the QoS. Additionally, having a *k*-fault-tolerant platform that will mitigate failure risks to acceptable levels could further improve the QoS.

Serverless Stream Model

To facilitate the serverless execution of edge real-time data analytics applications, we propose an

extension of the traditional stream processing model. In our serverless stream model (see Figure 2), the transformation function is the core concept and encapsulates user-defined data analytics logic to process data along the stream. These functions are then composed into topologies that enable complex data processing applications. In our model, we consider streams as first-class citizens – that is, streams are defined through all of the presented concepts, as well as function wrappers and stream contracts.

The wrapper is responsible for encapsulating the transformation functions and exposing a thin API layer, enabling the analytics function layer to treat functions as microservices. This lets our system transparently schedule and deploy functions using container-based deployment strategies, and compose functions to complex topologies. For stateful functions, these wrappers also provide implicit state management. The wrapper transparently handles state replication and migration, and access to a function's state is controlled via the exposed API.

The contract is a high-level description of how the platform manages deployment and execution of streams and their functions. Specifically, a contract gives a user fine-grained control over the platform's runtime mechanisms – that is, placement and scaling policies, governance rules, and QoS requirements. A contract is divided into sections, where each section specifies a respective runtime mechanism. The placement section allows control over how analytics functions are deployed across the infrastructure, such as which cloud or edge resources to use. The scaling section allows control over elasticity strategies – that is, how the platform should adapt to varying workloads. Governance rules allow additional definition of restrictions

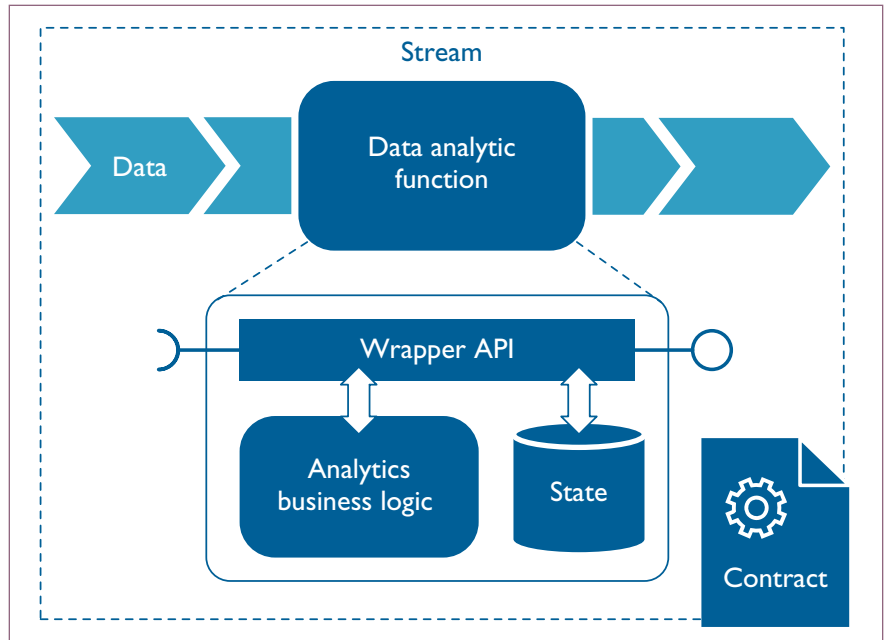


Figure 2. Serverless stream application model. The transformation function is the core concept and encapsulates user-defined data analytics logic to process the stream data.

regarding security or privacy, such as the exclusion of certain geographical regions in the distributed infrastructure. The QoS section allows control over QoS requirements the platform should respect, for example, maximum stream latency or minimum stream throughput. Unless explicitly defined in respective sections, the platform will enact sane defaults for each runtime mechanism.

Design Requirements and Challenges

This section outlines the design requirements and challenges to realize our real-time data analytics platform and its main runtime mechanisms in edge.

Provisioning Data Analytics Functions and Edge Resources

The serverless paradigm and application execution model undoubtedly has the potential to offer a wide range of benefits for provisioning and managing real-time edge data

analytics functions. Unfortunately, due to the inherently different nature of edge infrastructure, for example, in terms of available resources, networks, and so on, provisioning solutions designed for cloud-based serverless landscapes are hardly applicable out of the box in this new computing environment. Fundamental architecture and design assumptions behind such approaches need to be reexamined and specifically tailored for the edge infrastructure to support seamless provisioning of both infrastructure resources and application components.

In our previous work, we developed models⁷ and middleware⁸ that enable provisioning edge/IoT resources and applications across large-scale IoT and edge deployments. Although, these solutions offer numerous advantages to the application developers and operations managers – such as a logically centralized point of operation in a geographically dispersed infrastructure, uniform interaction patterns with both cloud and edge resources,

Related Work in Scalable Data Analytics Applications

Traditionally, scalable data analytics applications have been realized with cloud-supported, distributed, data-stream processing systems. Maintaining low end-to-end latencies under high data velocity is a major challenge for such systems, particularly in large-scale Internet of Things scenarios. Systems like StreamCloud¹ and Twitter Heron² were developed to handle massive amounts of data, using concepts such as auto-parallelization of stream operators, clustering, and elastic scaling. While these approaches address scalability issues, they don't consider edge-specific features such as locality awareness, which are crucial for achieving low-latency, real-time analytics.

Different approaches extended traditional stream processing with novel algorithms for deploying and scheduling operators at the edge. For example, Apostolos Papageorgiou and colleagues extended the stream topology deployment

algorithm of Apache Storm.³ In particular, their deployment optimization approach incorporates quality of service (QoS) metrics of topology-external interactions to reduce communication latencies within stream-processing topologies. Valeria Cardellini and colleagues propose a similar approach, where the scheduling algorithm takes into account network QoS metrics, such as latency, between stream operators.⁴

Only a few efforts have been made to develop novel architectures for data analytics platforms in the edge. Mahadev Satyanarayanan and colleagues propose GigaSight, a hybrid architecture for computer-vision analytics based on cloudlets.⁵ In GigaSight, cloudlets filter and process mobile video streams in near real time. Only processing results, such as recognized objects, and corresponding metadata are sent to the cloud, thereby reducing end-to-end latencies as well as bandwidth usage.

References

1. P. Valduriez et al., "StreamCloud: An Elastic and Scalable Data Streaming System," *IEEE Trans. Parallel and Distributed Systems*, vol. 23, no. 12, 2012, pp. 2351–2365.
2. S. Kulkarni et al., "Twitter Heron: Stream Processing at Scale," *Proc. ACM Sigmod Int'l Conf. Management of Data*, 2015, pp. 239–250.
3. A. Papageorgiou, E. Poormohammady, and B. Cheng, "Edge-Computing-Aware Deployment of Stream Processing Tasks Based on Topology-External Information: Model, Algorithms, and A Storm-Based Prototype," *Proc. IEEE Int'l Conf. Big Data*, 2016, pp. 259–266.
4. V. Cardellini et al., "On QoS-Aware Scheduling of Data Stream Applications Over Fog Computing Infrastructures," *Proc. IEEE Symp. Computers and Comm.*, 2015, pp. 271–276.
5. M. Satyanarayanan et al., "Edge Analytics in the Internet of Things," *IEEE Pervasive Computing*, vol. 14, no. 2, 2015, pp. 24–31.

and utility-based resource consumption – there's still a number of challenges to address to enable our vision of a serverless real-time data analytics platform for the edge. These include fully automated provisioning solutions, where user interaction is limited to providing high-level policies and goals that need to be fulfilled by the platform; design and implementation of provisioning and orchestration mechanisms for the edge network (for example, based on network function virtualization slicing or software-defined networks); and models and techniques for secure edge-resource negotiation based on smart contracts and Blockchain technologies.

Availability and Scheduling of Loosely Coupled Edge Resources

The analytics wrapper and APIs layer (see Figure 1b) is an abstraction level that addresses management of data

analytics functions. These abstraction concepts that arrive at different speeds stochastically must be mapped to real runtime mechanisms. This orchestration and configuration cannot occur in an easy and predictable manner, because of the limited resources in the runtime mechanisms layer. If there exist unlimited, tightly coupled resources as in the cloud, then mapping is ideal because each new function will be mapped to a new resource runtime mechanism in the described monitor, analyze, plan, and execute loop.

Resource availability in edge computing complicates the execution of the planned activities by the orchestration layer. Multiple methods can solve the problem of limited available resources and required processing, communication, or storage demands, such as inserting queues in front of each resource in the pool, or a common (grouped) queue.

Scheduling tasks over provisioned resources is by itself a challenge, as it is an NP-hard, real-time problem. Scheduling in the edge's loosely coupled infrastructure is even more challenging than for the cloud's tightly coupled one. Even if the edge provisions elastic resources, the edge's distribution will generate huge network latency because of its slower wide area network (WAN) compared to the cloud's high-speed LAN. Therefore, the platform should provide lightweight algorithms for optimizing the real-time decision making for scheduling, considering multiple conflicting criteria, such as energy efficiency, dependability, real-time response, resiliency, reliability, time-predictability, fault tolerance, and system cost. To reduce the tradeoff between the acceptable schedule that can be calculated fastest (in real time), a footprint analyzer can be implemented, whose historical outputs can be used

as heuristics to predict edge resource behavior without interrupting real-time data analytics.

Resources Virtualization and Heterogeneous Infrastructure Mapping

Most of the data collection devices (sensors) in the infrastructure resource pool are abstracted by their virtualized model in the resources abstraction layer. The main idea of the model is to transfer data and re-allocate the data analytics functions via cloud/fog computing servers.

We address the autonomous function of IoT devices as a challenge that our model will face. Autonomous processing means that certain data analytics algorithms are also performed on a lower level – that is, the processing is located closer to the source (in the edge). This leads to a *dew computing* design, including a dew server (edge device) on the path between the IoT devices/sensors and the cloud. Our model implements the autonomous function by mapping the main data analytics functions directly to the IoT devices.

In this context, an open challenge is to solve the interoperability and integration between different device implementations on the infrastructure level. Our resource virtualization layer directly addresses these issues, because each device is presented by its functions.

Finally, portability on the lower layer means to transfer defined functions between various devices. For example, this is addressed by the fault tolerance and elasticity runtime mechanisms to replicate or enable a spare device that will continue performing the defined functions if one resource fails or share the load if the load increases.

Rapid Elasticity at the Edge

Cloud-based serverless platforms use commodity infrastructure and have small footprint and short execution

duration, combined with statistical multiplexing of a large number of heterogeneous workloads over time.⁹ Edge, on the other hand, has different characteristics because of its different infrastructure – that is, infrastructure deployment and its geographic dispersion, the topology of network connectivity, and locality-awareness. One challenge is the discoverability of resources. Given the number of heterogeneous devices, standard discovery mechanisms could become burdened with massive workloads; as a solution, and to ensure elasticity, dynamic strategies should be in place for sharing resources. Another perspective is the geographic dispersion, where edge devices are scattered around the network and have limited capacity. A framework should be established, which can dynamically select appropriate devices in regard to proximity to data sources, while considering latency and mobility. The framework should differentiate critical and non-critical functions by setting priorities to the services in the case of a heavy workload in a single location. The topology of the network connectivity represents another issue in rapid provisioning of processing capabilities. It should allow imperceptible handling of any increased workload without increasing latencies and cope with heterogeneous devices, while maintaining a secure environment. As a result, the communication protocols among the nodes should be carefully selected and established.

Data Management

Apart from the five Vs (volume, velocity, variety, veracity, and value) for big data analytics, some real-time analytics applications require additional data management, which should be performed in edge, as well. The data must be preprocessed before going through the transformation functions. Further, data ingestion,


transcription services, deduplication, and natural language-processing algorithms are required.

Many real-time analytics, such as our first use case, require updating the model in real time according to the shorter data horizon. This requirement refreshes the incremental learning algorithms, which will compensate the lack of edge resources compared to the cloud. On the other hand, the lack of storage capacity is compensated with the smaller data obsolescence – that is, prediction upon smaller data series. The second use case is a good representative of this. In both use cases, the data can be forgotten or destroyed after being processed, which will mitigate the risk of data leakage and bring the data privacy and security to an acceptable level. For other real-time analytics, such as longer data horizons or data obsolescence, the same challenges remain, as is the case in the cloud.

To tackle these requirements, one challenge is to design a whole workflow of dependent processes in the orchestration layer. Placement, QoS, and fault tolerance should be added to the core runtime mechanisms for these types of real-time analytics, as struggling or failure of some execution nodes could lead to service-level agreement violations.

Despite the elastic computing power of cloud, along with high-speed networks, real-time analytics in the novel edge computing landscape are becoming ever-more challenging. Our platform facilitates real-time data analytics over cloud and edge computing in a uniform manner. The proposed serverless stream model makes transparent the underused underlying heterogeneous edge infrastructure and enables easier and more intuitive development of various real-time analytics functions,

without worrying about nonfunctional requirements.

Our model will switch the current view of centralized premise and cloud real-time analytics into more distributed, edge, ubiquitous, real-time analytics, in which the data's value won't be lost at the edge and all computing layers will be used evenly. Our vision is that all computing layers work together like a team, without making the cloud the team's most important player. Our platform will act as a team manager that will follow the road map toward a fully-fledged platform for cloud and edge real-time data analytics. It will overcome the challenges of provisioning data analytics functions at edge resources, making them highly available and rapidly elastic, and processing and managing the data in real time without (or before) passing it to the cloud. 

Acknowledgments

This work is partially supported by JPI Urban Europe, ERA-NET, under project 5631209 and by bilateral MKD-AUT project 18779 (Scalability and Elasticity Performance of Cloud Services).

References

1. M. Satyanarayanan, "The Emergence of Edge Computing," *Computer*, vol. 50, no. 1, 2017, pp. 30–39.
 2. V. Bahl, "Cloud 2020: Emergence of Micro Data Centers (Cloudlets) For Latency Sensitive Computing," keynote address, Devices and Networking Summit, 21 Apr. 2015; <https://channel9.msdn.com/Events/Microsoft-Research/Devices-and-Networking-Summit-2015/Closing-Keynote>.
 3. F. Bonomi et al., "Fog Computing and Its Role in the Internet of Things," *Proc. MCC Workshop Mobile Cloud Computing*, 2012, pp. 13–16.
 4. P. Valduriez et al., "StreamCloud: An Elastic and Scalable Data Streaming System," *IEEE Trans. Parallel and Distributed Systems*, vol. 23, no. 12, 2012, pp. 2351–2365.
 5. S. Kulkarni et al., "Twitter Heron: Stream Processing at Scale," *Proc. ACM Sigmod Int'l Conf. Management of Data*, 2015, pp. 239–250.
 6. S. Hendrickson et al., "Serverless Computation with OpenLambda," *Proc. Usenix Conf. Hot Topics in Cloud Computing*, 2016, pp. 33–39.
 7. S. Nastic et al., "Provisioning Software-Defined IoT Cloud Systems," *Proc. Int'l Conf. Future Internet of Things and Cloud*, 2014; doi:10.1109/FiCloud.2014.52.
 8. S. Nastic, H.-L. Truong, and S. Dustdar, "A Middleware Infrastructure for Utility-Based Provisioning of IoT Cloud Systems," *Proc. 1st IEEE/ACM Symp. Edge Computing*, 2016; doi:10.1109/SEC.2016.35.
 9. D. Breitgand et al., "SLA-Aware Resource Over-Commit in an IaaS Cloud," *Proc. 8th Int'l Conf. Network and Service Management*, 2012, pp. 73–81.
- Stefan Nastic** is a postdoctoral research assistant at the Distributed Systems Group (DSG), TU Wien, Austria. His research interests include Internet of Things (IoT) and edge computing, cloud computing, big data analytics, and smart cities. Nastic has a DrTech in programming, provisioning, and governing IoT cloud systems from TU Wien. Contact him at snastic@infosys.tuwien.ac.at.
- Thomas Rausch** is a PhD student at the DSG, TU Wien, Austria. His research interests include IoT, edge computing, and event-based systems. Rausch has an MS in software engineering and Internet computing from TU Wien. Contact him at trausch@dsg.tuwien.ac.at.
- Ognjen Scekcic** is a postdoctoral university assistant at the DSG, TU Wien, Austria. His research interests include social computing, collective adaptive systems, and smart cities. Scekcic has a PhD in automated incentive management for social computing (foundations, models, tools and algorithms) from TU Wien. Contact him at oscekcic@dsg.tuwien.ac.at.
- Schahram Dustdar** is a full professor of computer science, and he heads the DSG at TU Wien. His work focuses on distributed systems. Dustdar is an IEEE Fellow, a member of the Academia Europaea, an ACM Distinguished Scientist, and recipient of the IBM Faculty Award. He is on the editorial boards of *IEEE Internet Computing* and *Computer*. He's an associate editor of *IEEE Transactions on Services Computing*, *ACM Transactions on the Web*, and *ACM Transactions on Internet Technology*. He's the editor-in-chief of *Springer Computing*. Contact him at dustdar@dsg.tuwien.ac.at; <http://dsg.tuwien.ac.at/staff/sd/>.
- Marjan Gusev** is a professor at Ss. Cyril and Methodius University, Skopje, Macedonia. His research interests include IoT, cloud computing, and eHealth solutions. Gusev has a PhD in electrical sciences from the University of Ljubljana. Contact him at marjan.gusev@finki.ukim.mk.
- Bojana Koteska** is a PhD student and teaching and research assistant at Ss. Cyril and Methodius University. Her research interests include scientific and cloud computing, software quality, and distributed computing. Koteska has an MS in software engineering from Ss. Cyril and Methodius University. Contact her at bojana.koteska@finki.ukim.mk.
- Magdalena Kostoska** is an assistant professor at Ss. Cyril and Methodius University. Her research interests include cloud computing and IoT. Kostoska has a PhD in cloud computing from Ss. Cyril and Methodius University. Contact her at magdalena.kostoska@finki.ukim.mk.
- Boro Jakimovski** is an associate professor at Ss. Cyril and Methodius University. His research interests include grid computing, high-performance computing, parallel and distributed processing, and genetic algorithms. Jakimovski has a PhD in distributed systems from Ss. Cyril and Methodius University. Contact him at boro.jakimovski@finki.ukim.mk.
- Sasko Ristov** is a university assistant (postdoc) at the University of Innsbruck, Austria,

and an assistant professor at Ss. Cyril and Methodius University, Skopje, Macedonia. His research interests include performance modeling and optimization and parallel and distributed systems. Ristov has a PhD degree in computer science from Ss. Cyril and Methodius University. Contact him at sashko@dps.uibk.ac.at.

Radu Prodan is an associate professor at the University of Innsbruck, Austria. His research interests include parallel and distributed systems and software tools (performance, debugging, scheduling, fault tolerance, and energy). Prodan has a habilitation degree in computer science from the University of Innsbruck. He's an associate

editor of *IEEE Transactions on Parallel and Distributed Systems*. Contact him at radu@dps.uibk.ac.at.

myCS

Read your subscriptions through the myCS publications portal at <http://mycs.computer.org>.



CALL FOR STANDARDS AWARD NOMINATIONS

IEEE COMPUTER SOCIETY HANS KARLSSON STANDARDS AWARD



A **plaque** and **\$2,000 honorarium** is presented in recognition of **outstanding skills and dedication to diplomacy, team facilitation, and joint achievement in the development or promotion of standards** in the computer industry where individual aspirations, corporate competition, and organizational rivalry could otherwise be counter to the benefit of society.

NOMINATE A COLLEAGUE FOR THIS AWARD!

DUE: 15 OCTOBER 2017

- Requires 3 endorsements.
- Self-nominations are not accepted.
- Do not need IEEE or IEEE Computer Society membership to apply.

Submit your nomination electronically: awards.computer.org | Questions: awards@computer.org



IEEE  computer society