GovOps: The Missing Link for Governance in Software-defined IoT Cloud Systems

Stefan Nastic, Christian Inzinger, Hong-Linh Truong, and Schahram Dustdar

Distributed Systems Group, Vienna University of Technology, Austria
Email: {lastname}@dsg.tuwien.ac.at

Abstract. Cloud computing and the IoT are converging ever stronger, thus enabling proliferation of diverse large-scale IoT cloud systems. Such novel IoT cloud systems offer numerous advantages for the variety of stakeholders. However, due to scale, complexity and inherent geographical distribution of IoT cloud systems, governing new IoT cloud resources and capabilities poses numerous challenges in practice. In this paper, we introduce GovOps – a novel approach and a conceptual model for cloud-based, dynamic governance of software-defined IoT cloud systems at runtime. By introducing a suitable *GovOps reference model* and a dedicated *GovOps manager*, it simplifies realizing governance processes and enables performing custom governance tasks more efficiently in practice. We illustrate the feasibility of our approach and its applicability to govern large-scale software-defined IoT cloud systems, using real-world case studies in the building automation and vehicle management domains.

1 Introduction

To date, cloud computing models and techniques, such as infrastructure virtualization and management, Compute-, Storage- and Network-as-a-Service, etc., have been intensively exploited for large-scale Internet of Things (IoT) systems [9, 16, 20]. Recently, software-defined IoT cloud systems have been introduced [11] to enable easier provisioning and management of IoT cloud resources and capabilities. Generally, software-defined denotes a principle of abstracting low-level components (e.g., hardware) and enabling their management, programmatically through well-defined APIs [10]. This enables refactoring the underlying infrastructure into finer-grained resource components whose functionality can be (re)defined after they have been deployed. While IoT cloud systems introduce numerous possibilities, a plethora of challenges remain to govern and operate these new IoT cloud resources and capabilities at runtime.

Various domains, such as smart building and vehicle management, increasingly rely on IoT cloud resources and capabilities. Consequently, governance issues such as security, safety, legal boundaries, compliance, and data privacy concerns are ever-stronger being addressed [6, 7, 19], mainly due to their potential impact on the variety of involved stakeholders. However, such approaches are mostly intended for high-level business stakeholders, neglecting support,

e.g., tools and frameworks, to realize governance strategies in large-scale, geographically distributed IoT cloud systems. Approaching IoT cloud from the operations management perspective, different approaches have been presented, e.g. [4, 16, 17, 20]. For example, such approaches deal with IoT cloud infrastructure virtualization and its management, enabling utilization of cloud computation resources and operating cloud storage resources for big IoT data. However, most of these approaches do not consider high-level governance objectives such as legal issues and compliance. This increases the risk of lost requirements or causes over-regulated systems, potentially increasing costs and limiting business opportunities.

Conceptually, IoT governance usually addresses the *Internet* part of the IoT, e.g, in the context of the Future Internet services[1], while operations processes mostly deal with *Things* (e.g, in [5]) as additional resources that need to be operated. Therefore, governance objectives (law, compliance, etc.) are not easily mapped to operations processes (e.g., querying sensory data streams or adding/removing devices), so that contemporary models, which assume that business stakeholders define governance objectives, and operations managers implement and enforce them, are hardly feasible in IoT cloud systems. In practice, bridging the gap between governance and operations management of IoT cloud systems poses a significant challenge for stakeholders (e.g., operations managers), because traditional management and governance approaches are hardly applicable for IoT cloud systems, mainly due to the large number of involved stakeholders, novel requirements for shared resources and capabilities, dynamicity, geographical distribution, and the sheer scale of IoT cloud systems.

This calls for a systematic approach to govern and operate IoT cloud resources and capabilities. Extending on previously developed concepts, in this paper we introduce GovOps – a novel approach for cloud-based dynamic governance and operations management in software-defined IoT cloud systems. The main objectives of GovOps are twofold. On the one side, it aims to enable seamless integration of high-level governance objectives with concrete operations processes. On the other side, it enables performing operational governance processes for IoT cloud systems so that they are feasible in practice. We present a GovOps reference model that defines required roles, concepts, and techniques, to reduce the complexity of realizing IoT cloud governance processes. GovOps enables performing custom governance tasks more efficiently, thus reduces time, costs and potential consequences of insufficient or ineffective governance.

The remainder of this paper is structured as follows: Section 2 presents our motivating scenarios. In Section 3, we present the GovOps approach to governance and operations management in software-defined IoT cloud; Section 4 outlines the GovOp reference model; Section 5 discusses the related work; Finally, Section 6 concludes the paper and gives an outlook of our future work.

[1] http://ec.europa.eu/digital-agenda/en/internet-things

2 Scenarios: Governing software-defined IoT systems

Consider the following scenarios in the Building Automation and Vehicle Management domains that we will refer to throughout the rest of this paper. The scenarios are derived from our work conducted in the P3CL lab[2].

2.1 Scenario 1 – Fleet Management System

General description. Fleet Management System (FMS) is responsible for managing electric vehicles deployed worldwide, e.g., on different golf courses. We have identified three stakeholders who rely on the FMS to optimize their business tasks: vehicle manufacturer, distributors and golf course managers. The stakeholders have different business models. For example, as the manufacturer only leases the vehicles, he is interested in the complete fleet, e.g., regular maintenance, crashes and battery health. Golf course manages are mostly interested in vehicles security (e.g., geofencing features), preventing misuse, and safety on the golf course.

Infrastructure setup. The FMS is an IoT cloud system comprising vehicles' proprietary on-board gateways, network and cloud infrastructure. The on-board gateway is capable to host lightweight applications for: vehicle maintenance, tracking, info and club set-up. Vehicles communicate with the cloud via 3G, GPRS or Wi-Fi network to exchange telematic and diagnostic data. On the cloud we host different FMS subsystems and services to manage this data, e.g., determine vehicle status, perform remote diagnostics and batch configuration or software updates. For the legacy vehicles, not capable to host applications, a device which acts as a CAN-IP bridge is used (e.g, Teltonika FM5300[3]).

2.2 Scenario 2 – Building Automation System:

General description. Building Automation System (BAS) is responsible to monitor and control various building assets, such as HVAC, lightning, elevators and humidity control systems, as well as to handle fault events and alarms (e.g, fire or gas leakage). For safety-critical services, e.g., alarm handling, timely processing of the events and the availability of the BAS play a crucial role.

Infrastructure setup. Generally, BAS comprises a set of cloud-based services, gateways and various sensors and actuators integrated with the building's assets. Gateways which support typical BA device protocols (ModBus, BACnet, LonWorks and Fox), e.g., Niagra or Sedona[4], are used to communicate with sensors and actuators. For local processing, the gateways usually allow executing custom

[2] http://pcccl.infosys.tuwien.ac.at/
[3] http://www.teltonika.lt/en/pages/view/?id=1024
[4] http://www.tridium.com/

triggers, rules and some form of complex event processing (CEP) queries. For permanent storage and more resource-demanding processing, the gateways send streams of data to the remote cloud services.

2.3 System characteristics

We notice that both the FMS and the BAS have large-scale, geographically distributed infrastructure. Additionally, the FMS also utilizes virtualized IoT cloud infrastructure, such as virtual gateways (VGW), to support integrating legacy vehicles. Depending on a stakeholder and task-at-hand our systems have different customization requirements and non-functional requirements (e.g, regarding fault-tolerance and availability). For example in BAS, while for safety-critical services real-time delivery and processing is essential, for services such as HVAC controller, cost reduction is more important. Due to the multiplicity of the involved stakeholders, the FMS needs to allow for runtime customizations in order to exactly meet the stakeholder's functional requirements, depending on the problem at hand and availability or accessibility of the vehicles, as well as desired system's non-functional properties.

3 GovOps – A novel approach to governance and operations management in IoT cloud

The main objective of our GovOps approach (Governance and Operations) is twofold. On the one side it aims to enable seamless integration of high-level governance objectives and strategies with concrete operations processes. On the other side, it enables performing operational governance processes for IoT cloud systems in such manner they are feasible in practice.

Fig. 1 illustrates how GovOps relates to IoT cloud governance and operations. It depicts the main idea of GovOps to bring governance and operations closer together and bridge the gap between governance objectives and operations processes, by incorporating the main aspects of both IoT cloud governance and operations management. To this end, we define *GovOps principles and design process* of GovOps strategies (Section 4) that support determining what can and needs to be governed, based on the current functionality and features of an IoT cloud system, and that allow for aligning such system capabilities with regulations and standards. Additionally, we introduce a novel role, *GovOps manager* (Section 3.3) responsible to guide and manage designing GovOps strategies, because in practice it is very difficult, risky, and ultimately very costly to adhere to the traditional organizational silos, separating business stakeholders from operational managers. Therefore, GovOps integrates business rules and compliance constraints with operations capacities and best-practices, from early stages of designing governance strategies in order to counteract system over-regulation and lost governance requirements [7].

It is worth noting that GovOps does not attempt to defined a general methodology for IoT cloud governance. There are many approaches (Section 5), which

define governance models and accountability frameworks for managing governance objectives and coordinating decision making processes, and that can usually be applied within GovOps without substantial modifications.

3.1 Governance aspects

From our case studies, we have identified various business stakeholders such as building residents, building managers, governments, vehicle manufacturers and golf course managers. Typically, these stakeholders are interested in energy efficient and greener buildings, sustainability of building assets, legal and privacy issues regarding sensory data, compliance (e.g, regulatory or social), health of the fleet, security and safety issues related to the environments under their jurisdiction.

Depending on the concrete (sub)system and the involved stakeholders, governance objectives are realized via different governance strategies. Generally, we identify the following governance aspects: i) *environment-centric*, ii) *data-centric* and iii) *infrastructure-centric governance*.

Environment-centric governance deals with issues of overlapping jurisdictions in IoT cloud managed environments. For example, in our BAS, we have residents, building managers and the government that can provide governance objectives, which directly or indirectly affect an environment, e.g., a residential apartment. In this context, we need to articulate multiple governance objectives related to comfort of living, energy efficiency, safety, health and sustainability.

Data-centric governance mostly deals with implementing the governance strategies related to the privacy, quality, and provenance of sensory data. Examples include addressing legal issues, compliance, and user preferences w.r.t. such data.

Infrastructure-centric governance addresses issues about designing, installing, and deploying IoT cloud infrastructure. This mostly affects the early stages of introducing a IoT cloud system and involves feasibility studies, cost analysis, and risk management. For example, it supports deciding between introducing new hardware or visualizing the IoT cloud infrastructure.

3.2 Operations management aspects

Operations managers implement various processes to manage BAS and FMS at runtime. Generally, we distinguish following operational governance aspects: i) *configuration-centric*, ii) *topology-centric*, and iii) *stream-centric governance*.

Configuration-centric governance includes dynamic changes to the configuration models of the software-defined IoT cloud systems at runtime. Example processes include a) enabling/disabling an IoT resource or capability (e.g,

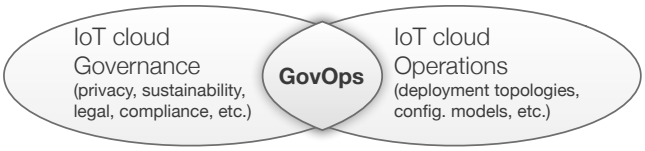

Fig. 1. GovOps in relation to IoT cloud governance and operations.

start/stop a unit), b) changing an IoT capability at runtime (e.g, communication protocol), and c) configuring an IoT resource (e.g, setting sensors poll rate).

Topology-centric governance addresses structural changes that can be performed on software-defined IoT systems at runtime. For example, a) Pushing processing logic from the application space towards the edge of the infrastructure; b) Introducing a second gateway and an elastic load balancer to optimize resource utilization, e.g., provide more bandwidth; c) Replicating a gateway, e.g., for fault-tolerance or data source history preservation.

Stream-centric governance addresses runtime operation of sensory data streams and continuous queries, e.g., to perform custom filtering, aggregation, and querying of the available data-streams. For example, to perform local filtering the processing logic is executed on physical gateways, while complex queries, spanning multiple data streams are usually executed on VGWs. Therefore, operations managers perform processes like: a) Placing custom filters (e.g., near the data source to reduce network traffic); b) Allocating queries to VGWs; and c) Splitting streams, i.e., sending events to multiple VGWs.

3.3 Integrating governance objectives with operations processes

The examples presented in Sections 3.1 and 3.2 are by no means a comprehensive list of IoT cloud governance processes. However, due to dynamicity, heterogeneity, geographical distribution and the sheer scale of IoT cloud, traditional approaches to realize these processes are hardly feasible in practice. This is mostly because such approaches implicitly make assumptions such as physical on-site presence, manually logging into gateways, understanding device specifics, etc., which are difficult, if not impossible, to meet in IoT cloud systems. Therefore, due to a lack of a systematic approach for operational governance in IoT systems, currently operations managers have to rely on ad hoc solutions to deal with the characteristics and complexity of IoT cloud systems, when performing governance processes.

Table 1 lists examples of governance objectives and according operations management processes to enforce these objectives. The first example comes from the FMS, since many of the golf courses are situated in countries with specific data regulations, e.g., the US or Australia. In order to enable monitoring of the whole fleet (as required by the manufacturer) the operations managers needs to understand the legal boundaries regarding data privacy. For example, in Australia, the OAIC[5] has issued a 32 page guidance as to what *"reasonable steps to protect personal information"* might include, that in practice need to be interpreted by operations managers. The second example contains potentially conflicting objectives supplied by stakeholders, e.g., building manager, end user, and the government, leaving it to the operations team to solve the conflicts, at runtime. The third example, hints that GNSS is usually better-suited to simultaneously work in both northern and southern high latitudes. Even for these basic processes, an operations team faces numerous difficulties, since in practice

[5] Office of the Australian Information Commissioner(OAIC), Aus. privacy regulator.

	Governance objectives	Operations processes
1.	Fulfill legal requirements w.r.t. sensory data in country X. Guaranty history preservation.	Spin-up an aggregator gateway. Replicate VGW, e.g., across different availability zones.
2.	Reduce GHG emission. User preferences regarding living comfort. Consider health regulations.	Provide a configuration directives for a IoT cloud resource (e.g, HVAC).
3.	Data quality compliance regarding location tracking services.	Choose among available services, e.g., GPS vs. GNSS (Global Navigation Satellite System) platform.

Table 1. Example governance objectives and operations processes.

there is no one-size-fits-all solution to map governance objectives to operations processes.

Therefore, GovOps proposes a novel role, *GovOps manager*, as a dedicated stakeholder responsible to bridge the gap between governance strategies and operations processes in IoT cloud systems. The main rationale behind introducing a GovOps manager is that in practice designing governance strategies needs to involve operations knowledge about the technical features of the system, e.g., physical location of devices, configuration and placement of queries, and component replication strategies. Reciprocally, defining systems configurations and deployment topologies should incorporate standards, compliance, and legal boundaries at early stages of designing operations processes. To achieve this, the GovOps manager is positioned in the middle, in the sense that they continuously interact with both business stakeholders (to identify high-level governance issues) and operations team (to determine operations capacities).

The main task of a GovOps manager is to determine suitable tradeoffs between satisfying the governance objectives and the system's capabilities, as well as to continuously analyze and refine how high-level objectives are articulated through operations processes. In this context, a key success factor is to ensure effective and continuous communication among the involved parties during the decision making process, facilitating i) openness, ii) collaboration, iii) establishment of a dedicated GovOps communication channel, along with iv) early adoption of standards and regulations. This ensures that no critical governance requirements are lost and counteracts over-regulation of IoT cloud systems.

3.4 Main principles of GovOps in IoT cloud systems

Generally, GovOps strategies manipulate the state of IoT cloud resources at runtime while considering governance objectives and regulations. The core idea of GovOps is to provide abstractions that shield stakeholders from the complexities of underlying system and diversities of various legal and compliance issues, allowing them to focus on integrating governance objectives with practically feasible operations processes. To support performing such processes in IoT cloud systems, (e.g., listed in Section 2), while considering system characteristics

(e.g., large-scale, geographical distribution and dynamicity), GovOps relies on concepts that include:

Central point of operation – Conceptually centralized interaction with the software-defined IoT cloud system to enable a unified view on the system's operations and governance capabilities (available at runtime), without worrying about low-level infrastructure details.

Automation – Allows for dynamic, on-demand governance of software-defined IoT cloud systems on a large scale (e.g, hundreds gateways) and enables governance processes to be easily repeatable, e.g., enforced across the IoT cloud, without logging into individual gateways.

Fine-grained control – Exposing the control functionality of IoT cloud resources at fine granularity allows for precise definition of governance processes (to exactly meet the requirements) and flexible customization of IoT cloud system governance capabilities.

Late-bound policies – Declarative policies that are bound later during runtime allow for designing generic and flexible governance processes in order to account for system dynamicity.

Resource autonomy – Providing a higher degree of autonomy to IoT cloud resources reduces the communication overhead, increases availability (e.g., when the connection is lost), enables local exception and faults handling, supports protocol independent interaction, and increases system scalability.

4 A reference model for GovOps in IoT cloud

4.1 GovOps model for software-defined IoT cloud systems

To realize the GovOps approach we need suitable abstractions to describe IoT cloud resources that allow IoT cloud infrastructure to be (re)defined after it has been deployed. We show in [11] how this can be done with *software-defined IoT units*. GovOps model (Fig. 2) builds on this premise and extends our previous work with fundamental aspects of operational governance processes: i) describing states of deployed IoT resources, ii) providing capabilities to manipulate these states at runtime, and iii) defining governance scopes.

Within our model, the main building blocks of GovOpsStrategies are *GovernanceCapabilities*. They represent operations which can be applied on IoT cloud resources, e.g., query current version of a software, change communication protocol, and spin-up a virtual gateway. These operations manipulate IoT cloud resources in order to put an IoT cloud system into a specific (target) state. Governance capabilities are described via well-defined software-defined APIs and they can be dynamically added to the system, e.g, to a VGW. From the technical perspective, they behave like add-ons, in the sense that they extend resources with additional operational functionality. Generally, by adopting the notion of governance capabilities, we allow for processes to be *automated to a great extent*, but also give a degree of *autonomy* to IoT cloud resources.

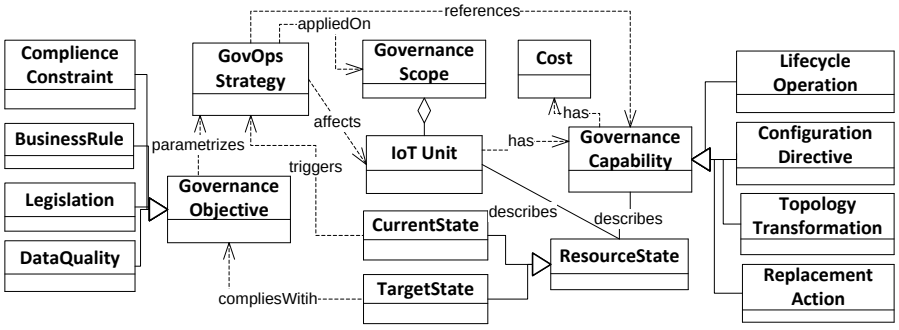

Fig. 2. Simplified UML diagram of GovOps model for IoT cloud governance.

Since the meaning of a resource state is highly task specific, we do not impose many constraints to define it. Generally, any useful information about an IoT cloud resource is considered to describe the *ResourceState*, e.g., a configuration model or monitoring data such as CPU load. Technically, there are many frameworks (e.g., Ganglia or Nagios) that can be used to (partly) describe resource states. Also configuration management solutions, such as OpsCode Chef[6], can be used to maintain and inspect configuration states. Finally, design best practices and reference architectures (e.g., AWS Reference Architectures[7]) provide a higher-level description of the desired target states of an IoT cloud system.

The *GovernanceScope* is an abstract resource, which represents a group of IoT cloud resources (e.g., gateways) that share some common properties. Therefore, our governance scopes are used to dynamically delimit IoT cloud resources on which a GovernanceCapability will have an effect. This enables writing the governance strategies in a scalable manner, since the IoT cloud resources are not individually addressed. It also allows for backwards compatible GovOps strategies, which do not directly depend on the current resource capabilities. This means that we can move a part of the problem, e.g., faults and exceptions handling, inside the governance scope. For example, if a gateway loses a capability the scope simply wont invoke it i.e., the strategy will not fail.

4.2 Design process of GovOps strategies

As described in Section 3, the GovOps manager is responsible to oversee and guide the GovOps design process and to design concrete GovOps strategies. The design process is structured along three main phases: i) identifying governance objectives and capabilities, ii) formalizing strategy, and iii) executing strategy.

Generally, the initial phase of the design process involves eliciting and formalizing governance objectives and constraints, as well as identifying required *fine-grained* governance capabilities to realize the governance strategy in the underlying IoT cloud system. GovOps does not make any assumptions or impose

[6] http://opscode.com/chef
[7] http://aws.amazon.com/architecture/

constraints on formalizing governance objectives. To support specifying governance objectives the GovOps manager can utilize various governance models and frameworks, such as the 3P [15] or COBIT [8]. However, it requires tight integration of the GovOps manager into the design process and encourages collaboration among the involved stakeholders to clearly determine risks and tradeoffs, in terms of what should and can be governed in the IoT cloud system, e.g., which capabilities are required to balance building emission regulations and residents temperature preferences. To this end, the GovOps manager gathers available governance capabilities in collaboration with the operations team, identifies missing capabilities, and determines if further action is necessary. Generally, governance capabilities are exposed via well-defined APIs. They can be built-in capabilities exposed by IoT units (e.g., start/stop), obtained from third-parties (e.g., from public repositories or in a market-like fashion), or developed in-house to exactly reflect custom governance objectives. By promoting *collaboration* and early integration of governance objectives with operations capabilities, GovOps reduces the risks of lost requirements and over-regulated systems.

After the required governance capabilities and relevant governance aspects are identified, the GovOps manager relies on the aforementioned concepts and abstractions (Section 4.1) to formally define the GovOps strategy and articulate the artifacts defined in the first phase of the design process. Governance capabilities are the main building blocks of the GovOps strategies. They are directly referenced in GovOps strategies to specify the concrete steps which need to be enforced on the underlying IoT cloud resources, e.g., defining a desired communication protocol or disabling a data stream for a specific region. Also in this context, the GovOps reference model does not make assumptions about the implementation of governance strategies, e.g., they can be realized as business processes, policies, applications, or domain specific language. Individual steps, defined in the generic strategy, invoke governance capabilities that put the IoT cloud resources into desired target state, e.g., which satisfies a set of properties. Subsequently, the generic GovOps strategy needs to be parameterized, based on the concrete constraints and rules defined by the governance objectives. Depending on the strategy implementation these can be realized as process parameters, language constraints (e.g., Object Constraint Language), or application configuration directives. By formalizing the governance strategy, GovOps enables reusability of strategies, promotes consistent implementation of established standards and best practices, and ensures operation within the system's regulatory framework.

The last phase involves identifying the system resources, i.e. the governance scopes that will be affected by the GovOps strategy and executing the strategy in the IoT cloud system. It is worth mentioning that the scopes are not directly referenced in the GovOps strategies, as the GovOps manager applies the strategies on the resource scopes instead of the actual resources. Introducing scopes at the strategy-level shields the operations team from directly referencing IoT cloud resources, thus enables designing *declarative, late-bound strategies* in a scalable manner. Furthermore, additional capabilities identified in the previous

phase will be acquired and/or provisioned at this point in the underlying IoT cloud system, whereas unused capabilities will be decommissioned in order to optimize resource consumption.

4.3 Enabling techniques & design practices

As previously discussed, IoT cloud systems need to support the dynamic addition and removal of governance capabilities at runtime. To this end, these capabilities are modeled as microservices and we employ mechanisms, such as automated provisioning and continuous delivery to manage them at runtime.

For example, to support the last phase of the GovOps process, we need to enable centrally managed governance capabilities. We rely on a centrally managed repository for maintaining governance capabilities and dynamic provisioning mechanisms, on the infrastructure level to deliver them to IoT cloud resources. To support multi-tenancy on the gateway level we can utilize proven concepts from the cloud computing domain, e.g., Linux containers that rely on light-weight virtualization technologies to execute governance capabilities in isolation, while maintaining low memory and performance overheads necessary for the constrained resources.

The dynamic nature of governance capabilities calls for an appropriate management mechanism, such as *dynamic software-defined APIs*. Such APIs offer a centralized view on the governance processes in IoT cloud systems, but in practice they depend on the available governance capabilities, which can dynamically change at runtime. In this context, providing stable and well-defined APIs plays a crucial role, but also requires rethinking existing API management solutions.

5 Related work

The IoT governance has been receiving a lot of attention recently. For example, in [19] the author evaluates various aspects of the IoT governance, such as privacy, security and safety, ethics, etc., and defines main principles of IoT governance, e.g., legitimacy and representation, transparency and openness, and accountability. In [18], the authors deal with issues of data quality management and governance. They define a responsibility assignment matrix that comprises roles, decision areas and responsibilities and can be used to define custom governance models and strategies. Traditional IT governance approaches, such as SOA governance [2,3,13] and governance frameworks like CMMI [1], the 3P model [15], and COBIT [8], provide a valuable insights and models which can be applied in GovOps processes, usually without substantial modifications. Compared to these approaches, GovOps does not attempt to define a general methodology for IoT cloud governance. Therefore, such approaches conceptually do not conflict with our approach and they can rather be seen as methodologies and techniques complementing GovOps.

Also approaches addressing operations management in IoT cloud system have recently emerged. For example, in [16, 20] the authors deal with IoT infrastructure virtualization and its management on cloud, whereas [4] utilizes the cloud

for additional computation resources. In [17] the authors focus on operating cloud storage resources for IoT data, and [12, 14] present approaches for monitoring IoT systems and enforcing QoS aspects. Such approaches provide useful concepts and techniques, which can be used to support the GovOps processes in IoT cloud systems. In [9] the authors develop an infrastructure virtualization framework, based on a content-based pub/sub model for asynchronous event exchange. In [20] the authors propose virtualizing physical sensors on the cloud and provide management and monitoring mechanisms for the virtual sensors. Such approaches provide various governance capabilities, e.g., template-based controlling of sensor groups, registering and decommissioning sensors and monitoring the QoS that can seamlessly be integrated with our GovOps approach.

The GovOps model builds on these approaches and addresses the issue of bridging the gap between governance objectives and operations processes, by introducing the GovOps manager as a dedicated stakeholder, as well as defining the suitable GovOps reference model to support early integration of governance objectives and operations processes. For high-level business stakeholders, GovOps enables continuous analysis, verification, and improvement of governance objectives and implemented strategies using a systematic approach. Furthermore, implementing the GovOps approach enables technological advantages such as greater flexibility, reduction of time-to-delivery, improved ease of operation, and shielding operations from regulatory issues.

6 Conclusion and future work

In this paper, we introduced the GovOps approach to runtime governance of software-defined IoT cloud systems. We presented the GovOps reference model that defines suitable concepts and a flexible process to design IoT cloud governance strategies. We introduced the GovOps manager, a dedicated stakeholder responsible to determine suitable tradeoffs between satisfying governance objectives and IoT cloud system capabilities, and ensure early integration of these objectives with operations processes, by continuously refining how the high-level objectives are articulated through operations processes. We showed how GovOps allows for IoT cloud governance processes to be easily and flexibly realized in practice, without worrying about the complexity and scale of the underlying IoT cloud and diversities of various legal and compliance issues.

In our ongoing work, we will develop a comprehensive framework for GovOps that implements the presented enabling techniques and required tool set in order to support GovOps managers in realizing IoT cloud governance strategies.

Acknowledgments. This work is sponsored by Pacific Controls Cloud Computing Lab (PC3L), as well as the Austrian Science Fund (FWF) under grant P23313-N23 (Audit 4 SOAs).

References

1. D. M. Ahern, A. Clouse, and R. Turner. *CMMI distilled: a practical introduction to integrated process improvement.* Addison-Wesley Professional, 2004.

2. M. Bano, D. Zowghi, and N. Ikram. Alignment between business requirements and services: the state of the practice. In *ICSSEA*, 2013.
3. A. Charfi and M. Mezini. Hybrid web service composition: business processes meet business rules. In *ICSOC*, pages 30–38. ACM, 2004.
4. B.-G. Chun, S. Ihm, P. Maniatis, M. Naik, and A. Patti. Clonecloud: elastic execution between mobile device and cloud. In *Conference on Computer systems*. ACM, 2011.
5. A. Copie, T. Fortis, V. I. Munteanu, and V. Negru. From cloud governance to iot governance. In *Advanced Information Networking and Applications Workshops*, pages 1229–1234. IEEE, 2013.
6. Don DeLoach. Internet of Things Part 4: Critical issues around governance for the Internet of Things. URL: `http://tinyurl.com/mxnq3ma`. [Online; accessed July-2014].
7. European Commission. Report on the public consultation on IoT governance. URL: `http://tinyurl.com/mx24d9o`. [Online; accessed August-2014].
8. G. Hardy. Using IT governance and COBIT to deliver value with IT and respond to legal, regulatory and compliance challenges. *Information Security technical report*, 11(1):55–61, 2006.
9. M. M. Hassan, B. Song, and E.-N. Huh. A framework of sensor-cloud integration opportunities and challenges. In *ICUIMC*, 2009.
10. B. Lantz, B. Heller, and N. McKeown. A network in a laptop: rapid prototyping for software-defined networks. In *SIGCOMM Workshop on Hot Topics in Networks*. ACM, 2010.
11. S. Nastic, S. Sehic, D.-H. Le, H.-L. Truong, and S. Dustdar. Provisioning software-defined iot systems in the cloud. In *FiCloud*, 2014.
12. M.-A. Nef, L. Perlepes, S. Karagiorgou, G. I. Stamoulis, and P. K. Kikiras. Enabling qos in the internet of things. In *CTRQ 2012*, pages 33–38, 2012.
13. M. Niemann, A. Miede, W. Johannsen, N. Repp, and R. Steinmetz. Structuring SOA governance. *International Journal of IT/Business Alignment and Governance*, 1(1):58–75, 2010.
14. V. Pereira, J. Sa Silva, and E. Monteiro. A framework for wireless sensor networks performance monitoring. In *WoWMoM*, pages 1–7. IEEE, 2012.
15. B. Sandrino-Arndt. People, portfolios and processes: The 3p model of it governance. *Information Systems Control Journal*, 2:1–5, 2008.
16. J. Soldatos, M. Serrano, and M. Hauswirth. Convergence of utility computing with the internet-of-things. In *IMIS*, pages 874–879, 2012.
17. P. Stuedi, I. Mohomed, and D. Terry. Wherestore: Location-based data storage for mobile devices interacting with the cloud. In *MCS*. ACM, 2010.
18. K. Weber, B. Otto, and H. Österle. One size does not fit all—a contingency approach to data governance. *Journal of Data and Information Quality (JDIQ)*, 1(1):4, 2009.
19. R. H. Weber. Internet of things–governance quo vadis? *Computer Law & Security Review*, 29(4):341–347, 2013.
20. M. Yuriyama and T. Kushida. Sensor-cloud infrastructure-physical sensor management with virtualized sensors on cloud computing. In *NBiS*, 2010.