UDC 004.77
DOI https://doi.org/10.32782/2663-5941/2025.3.1/15

Chukov O.O.
National Technical University of Ukraine
"Igor Sikorsky Kyiv Polytechnic Institute"

Redko I.V.
National Technical University of Ukraine
"Igor Sikorsky Kyiv Polytechnic Institute"

CONTINUOUS DEPLOYMENT OF OTA UPDATES IN IOT SOLUTIONS

In this article, the challenges of applying continuous deployment in IoT solutions are examined in the context of OTA firmware updates. The traditional continuous deployment approach used in software development is not directly suitable for IoT environments due to limited device resources, unstable connectivity, and the need for careful, staged rollouts. The article highlights that most OTA updates in IoT solutions are still managed with manual intervention by humans (experts), which slows down the process and introduces risks related to human mistakes.

To address these issues and apply continuous deployment approach to OTA updates in IoT, the article proposes a solution based on adding to OTA deployment workflow the Data-Processing and Decision Making unit, which automates all the routine work, putting an expert aside to only decide on the most critical decisions and perform the most critical operations. This unit is responsible for making rollout decisions at every stage, based on initial commands received from an expert, and data received from previous rollout iterations OTA and Device Operability status reports. The proposed solution addresses challenges of applying continuous deployment to IoT, like deployment scale, latency in feedback loops, rollback complexity, and brings to the IoT domain the benefits of continuous deployment, like reduced costs, accelerated feedback loop, and improved quality. The proposed solution is device-agnostic, presented as a high-level architecture diagram, and therefore doesn't contain details on protocols to be used and ways of implementation.

The article also highlights directions for future research, including formalising the parameters used across the deployment workflow, defining the rules and thresholds for rollout decisions, and validating the solution through real-world testing or simulations. These steps are important to ensure the applicability of the approach across different IoT systems and to bring the concept from design to practical implementation.

Key words: *Internet of Things, IoT, over the air update, OTA, firmware update, continuous deployment, device management.*

Formulation of the problem. The Internet of Things (IoT) is a new technology that connects electronic devices and sensors through the Internet to make our lives easier. IoT uses smart devices and the Internet to create solutions for different problems in business, government, and public and private sectors around the world. The increasing adoption of IoT underscores the need for efficient software management, as device functionalities must be updated and improved over time.

One of the most critical aspects of maintaining IoT devices is ensuring that they receive timely and secure software updates to provide security enhancements, bug fixes, performance optimisations, and feature extensions. Over-the-air (OTA) is the only practical approach for providing updates to IoT devices due to the large number of distributed devices, many of which operate in remote or restricted environments, often with no physical access to them.

Continuous deployment (CD) is one of the commonly used techniques for the delivery of software updates, with code changes to an application being released automatically into the production environment. While CD is widely used in cloud computing and web applications, its application to IoT remains complex due to the constraints of embedded systems and heterogeneous network environments. Implementing continuous deployment for OTA updates in IoT solutions requires careful planning, robust security measures, and an efficient rollout strategy.

This article examines the applicability of the CD technique for OTA updates in IoT networks. The article explores key challenges associated with CD in IoT, followed by a proposed solution workflow diagram, aiming to resolve or minimise the impact of those challenges. By integrating Continuous Deployment with IoT update mechanisms, organisations can achieve greater efficiency, enhanced security, and improved user experiences in managing their connected devices.

Analysis of recent research and publications. The Internet of Things (IoT) is an emerging paradigm that enables the communication between electronic devices and sensors through the Internet in order to facilitate our lives. IoT uses smart devices and the internet to provide innovative solutions to various challenges and issues related to various business, governmental, and public/private industries across the world [1, p. 1].

A great transformation can be observed in our daily routine life along with the increasing involvement of IoT devices and technology. One such development of IoT is the concept of Smart Home Systems (SHS) and appliances that consist of internet-based devices, automation systems for homes, and reliable energy management systems. Besides, another important achievement of IoT is Smart Health Sensing system (SHSS), which incorporates small intelligent equipment and devices to support the health of the human being. IoT has brought up some new advancements to make transportation more efficient, comfortable and reliable. Intelligent sensors and drone devices are now controlling the traffic at different signalised intersections across major cities [1, p. 2].

IoT solutions are widely used across many business sectors and public services to improve efficiency and connectivity. Among all IoT projects, the top 5 by market share occupy Smart City (22%), Connected Industry (17%), Connected Building (12%), Connected Car (11%) and Smart Energy (10%) [2]. IoT projects are spread across the world, with the Americas making up most of those projects (45%), followed by Europe (35%), and Asia (16%).

Nowadays, IoT environments are characterised by the presence of a large number of devices, often massively deployed in an area of interest to enable an IoT application. However, because of device constraints (e.g, heterogeneity, resource limitations, etc.), IoT networks are experiencing many challenges, among the most relevant of which are the following [3, p. 39–42]:

Heterogeneity. The main objective of IoT is to create a common way to abstract heterogeneous devices and achieve the optimal exploitation of their functionality.

Scalability. It is one of the most important challenges of IoT, which means how to deal with the sustainable growth of the Internet in an efficient manner.

Security and privacy. Constant need of protecting data and systems from external attacks and internal misuse, while ensuring confidentiality, trust, and personal data protection.

Routing. This raises the task of selecting the best path between the source and the destination to complete the communication process successfully.

Interoperability. This concept can be defined as the ability to create systems or devices that efficiently cooperate with each other.

Networking. Traffic and protocols that have a significant impact on the behaviour of the network require a reliable and stable network connection.

Given the complexity of the listed challenges, efficient management of IoT networks becomes critical. To address these issues, various IoT network management solutions have been developed, offering tools for device monitoring, firmware and configuration updates, data collection, and system integration.

An overview of IoT network management frameworks, including Azure (from Microsoft), IBM

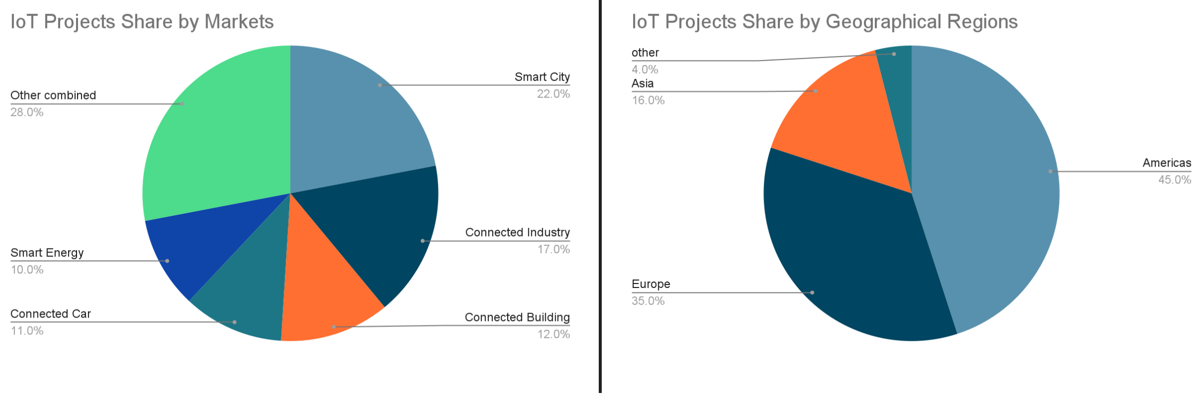


Fig. 1. IoT Projects Shares by Markets and Geographical Regions

IoT (from IBM), Artik Cloud (from SAMSUNG), Mbed (from Arm), Google Cloud Platform (from Google), AWS IoT (from Amazon), with a summary of their capabilities to perform data collection, device monitoring, network and device configuration, and communication protocols, is provided in Table 1 [4, p. 4165].

The firmware update mechanism is declared as part of the Device Management and is included in all of the listed IoT cloud platforms. A detailed overview of AWS IoT Device Management solution for automated deployment of updates to IoT devices is provided in Fig. 2 [5].

In the provided flow, the AWS IoT Device Management takes care of scheduling, retrying, and reporting the status of remote operations, like OTA update or device configuration. However, the update process is designed from the perspective of one device or small device fleets, and does not fully address the challenges of large-scale deployments. Issues like scalability to bigger device fleets, consisting of heterogeneous devices with differences

Table 1

IoT cloud platforms and their features

IoT cloud platform	Protocols for data collection	Management protocol	Device management	Device monitoring	Communication technologies	Resource constrained devices
Azure (Microsoft)	MQTT	LWM2M	✓	✓	–	✓
IBM IoT (IBM)	MQTT, HTTP	LWM2M	✓	–	–	✓
Artik Cloud (Samsung)	REST/ HTTP, websockets, MQTT, CoAP	LWM2M	✓	–	–	✓
Mbed (Arm)	HTTP, HTTPS, CoAP, MQTT	LWM2M	✓	–	BLE, Thread, 6LowPAN, Wi-Fi, LoRa	✓
Google Cloud Platform (Google)	MQTT	–	✓	✓	–	✓
AWS IoT (Amazon)	MQTT	–	✓	✓	–	✓

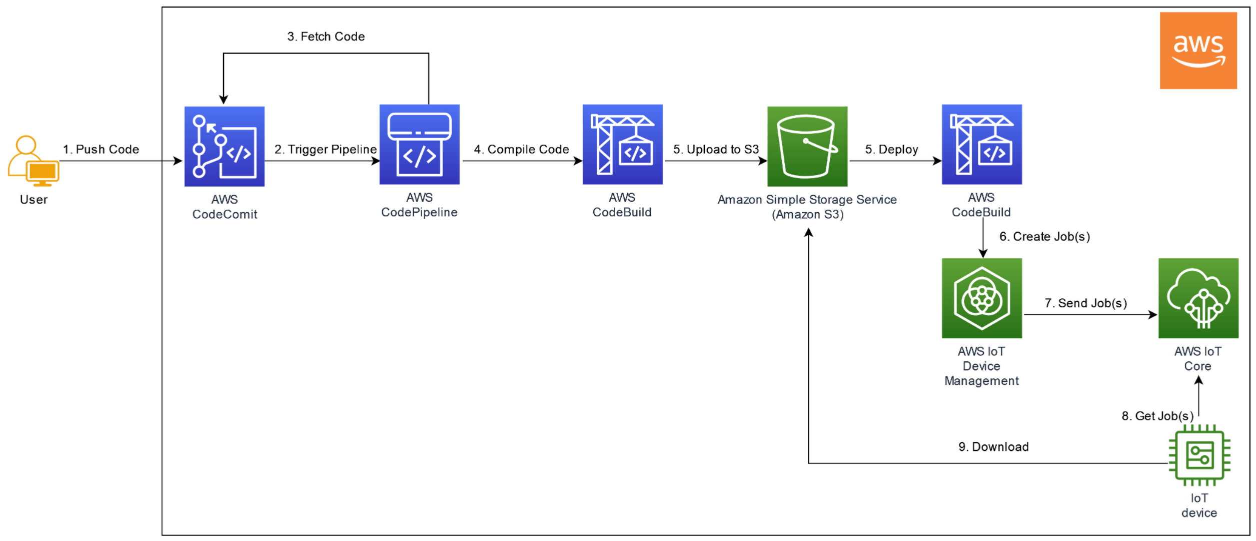


Fig. 2. AWS IoT Device Management Solution for IoT Devices Update

in hardware, operating in different environments, etc., remain on the customer side to implement. This leaves a gap that needs to be addressed through future research and the unification of the process of large-scale deployment of OTA updates.

To address the identified gap in IoT, it is important to explore software deployment techniques, which have proven effectiveness in non-IoT solutions, like Continuous Deployment. Continuous deployment (often abbreviated to CD) is a software development strategy where code changes to an application are automatically released into the production environment (Susnjara & Smalley, 2024). Not to be confused with continuous delivery, which shares the same acronym (CD), but is a software development practice where developers build software to release it into production at any time [6].

Traditionally, when we talk about continuous deployment, we think of deploying Web or Mobile apps [7]. Moreover, continuous deployment is often considered a must-have approach in software-based solutions. A list of benefits of continuous deployment that supports this statement is provided in Table 2 [6].

Task statement. IoT solutions cannot be referred to as software-based solutions in the way that is described in a previous section. IoT are embedded systems with embedded software, which is usually called firmware, so IoT are firmware-based solutions. The classical continuous deployment approach cannot be directly applied to IoT solutions due to the limitations and unique challenges it faces:

All this makes continuous deployment a desired but rather very challenging approach for IoT solutions. The task of firmware update in IoT solutions is still mostly done manually (or semi-manually), with a human (expert) to initiate the process and control it during its progress.

Taking into account the big scale of IoT networks with thousands or even millions of end devices, the deployment of OTA updates cannot be done at once. Updates are deployed iteratively, starting with a smaller pool of devices to get the update and

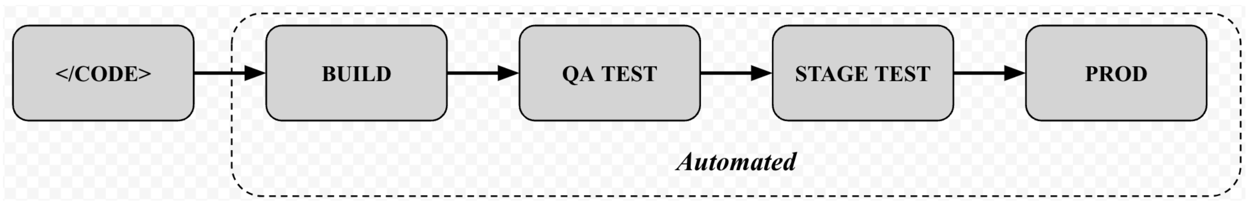


Fig. 3. High-level flow of releasing updates with the continuous deployment approach

Table 2

Benefits of continuous deployment

Accelerated feedback loop	Continuous deployment accelerates the feedback loop by allowing developers to release code changes frequently. This capability reduces the time that it takes to receive feedback from users and stakeholders.
Faster time to market	Continuous deployment helps deliver updates and software releases quickly. Once new updates pass predefined tests, the system automatically pushes them to the software's end users.
Better team collaboration	Continuous deployment frees up developers to focus more on writing code and performing tests rather than on manual deployment procedures. It also supports team collaboration and communication by providing a single view across all applications and environments.
Improved quality	Automated testing – the most critical dependency for continuous deployment – occurs at each stage of the deployment pipeline lifecycle. This capability improves the overall quality of the deployment experience. For instance, automated testing can debug errors before they reach production.
Enhanced customer experience	Automated testing allows development teams to quickly and consistently deploy new features and improvements for enhanced customer experience.
Reduced costs	Automating the deployment eliminates bottlenecks and reduces manual tasks. This process helps businesses save costs by reducing downtime.

Table 3

Challenges of continuous deployment in IoT solutions

Connectivity issues	IoT devices often operate in environments with unreliable or limited internet connectivity.
Resource constraints	Many IoT devices have limited computational power, memory, and storage.
Latency in feedback loops	Monitoring the effects of a deployment and gathering feedback from devices can be slow.
Rollback complexity	Rolling back a faulty update is more challenging for IoT devices due to potential device inaccessibility or constraints.
Deployment Scale	IoT ecosystems often involve thousands or millions of devices.

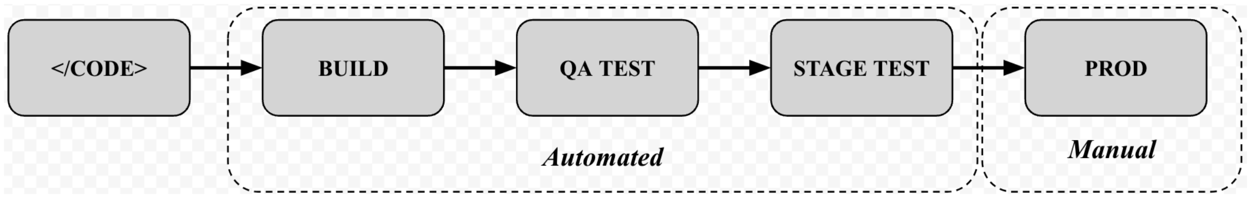


Fig. 4. High-level flow of releasing updates in IoT solutions

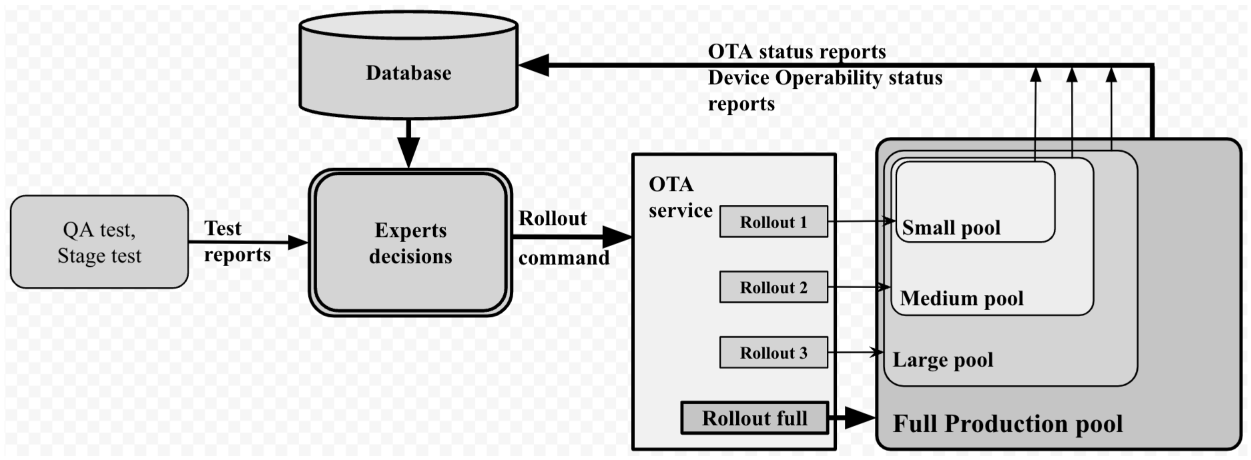


Fig. 5. Iterative deployment of OTA updates in IoT solutions

provide OTA and Device Operability status reports, which confirm the update is safe, and deployment can proceed to a larger pool of devices. A human (an expert) is set at the center of decision making for each iteration, with the tasks to launch the intial iteration, analyse output results from status reports, launch the next iteration if results satisfy criteria, pause the deployment if results are not clear and don't allow to decide on the next step, or break the deployment if results doesn't match criteria cor continuation.

The diagram of the iterative deployment of OTA update in the IoT networks with a human (an expert) at the centre is provided in the figure above.

Relying on an expert as the central decision-maker introduces delays in OTA deployment, limits scalability, and increases the risk of human error. Automating the deployment with the continuous deployment approach enables faster and more reliable updates, with other benefits provided by CD.

Outline of the main material of the study. To address the challenges of continuous deployment of OTA updates in IoT solutions, it is proposed to extend the iterative deployment flow by adding the Data Processing and Decision-Making (DPDM) unit, which makes all the decisions automatically.

Proposed concept puts DPDM into the center of the iterative deployment process, instead of a human (an expert), which still might be a part of the flow but only as a provider of additional input information (e.g., when to initiate update process, set desired schedule, define device's pool to target, etc.).

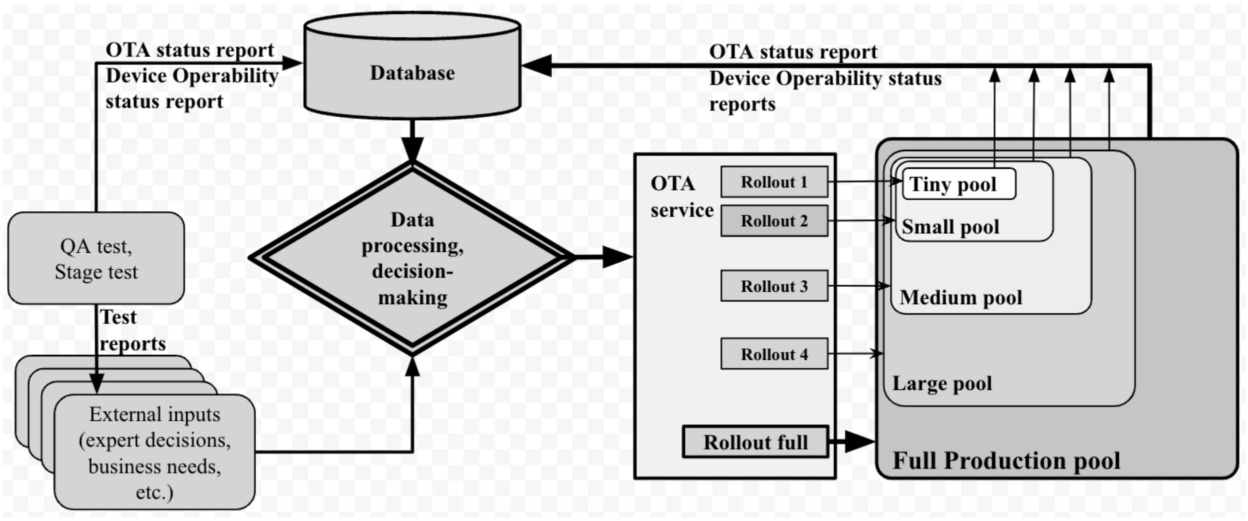


Fig. 6. Iterative deployment of OTA updates in IoT solutions with CD approach

The new flow could be divided into four major phases to explain the process of automated IoT updates.

On **Phase 1, the initiation of the deployment of the OTA update** happens. This could be done either automatically by triggering from the autorun service, or manually by a human (expert) to trigger the process. A set of input parameters should be passed to the DPDM unit to start the deployment, like:

- Quality Assurance (QA) test report, to ensure the update is stable, functional, and free of critical bugs, preventing failures or disruptions after deployment
- Stage test report, after verification of the new firmware in a production-like environment, ensuring it works correctly with real configurations and systems;
- OTA and Device Operability status reports generated during QA and Stage tests, provide initial information about IoT device's stability; this is the first set of data that is being processed by DPDM and taken into account while making the decision to initiate deployment
- Other inputs from a human (an expert) also might be provided, like:
  - Date and time to initiate deployment – this allows the start of deployment at the desired time, satisfying business needs, e.g., to complete by the announced date of update
  - Schedule of the rollout – this allows to avoid rollout stages execution during non-desired time slots, e.g., during peak load of IoT devices
  - Pool of devices to target – this allows deployment of updates to a certain desired pool, e.g., in a specific geographical region or owned by a specific customer
  - Priority criteria for picking up devices on each rollout stage – this allows to prioritise specific devices to be taken at first, e.g., from the Beta pool or devices equally distributed across regions

On **Phase 2, rollout of the OTA update to an initial pool (on the first iteration) and larger pools (on each next iteration)** happens. Based on input data received on Phase 1, DPDM communicates to the OTA service, sets exact parameters for the rollout, and awaits results from the next phases.

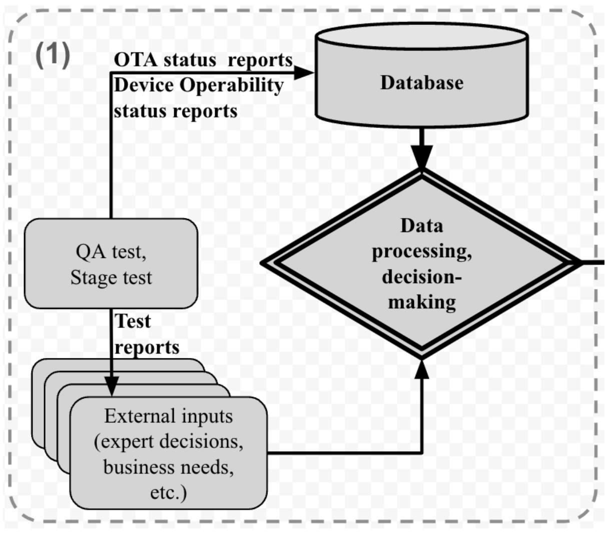
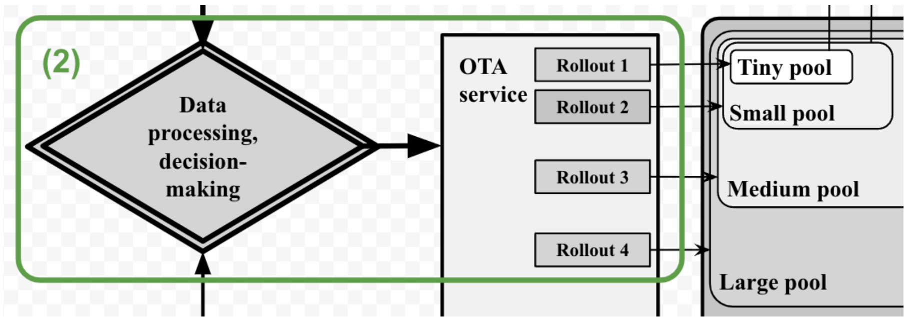
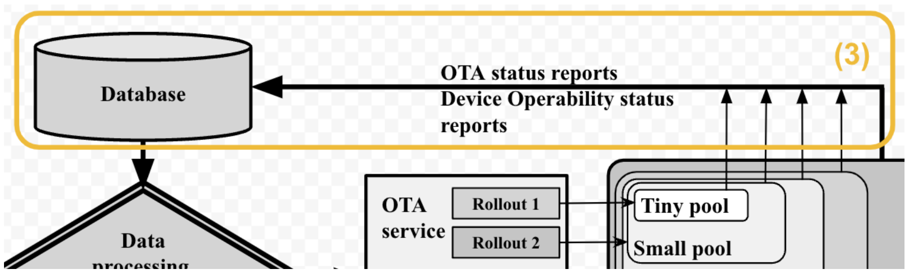
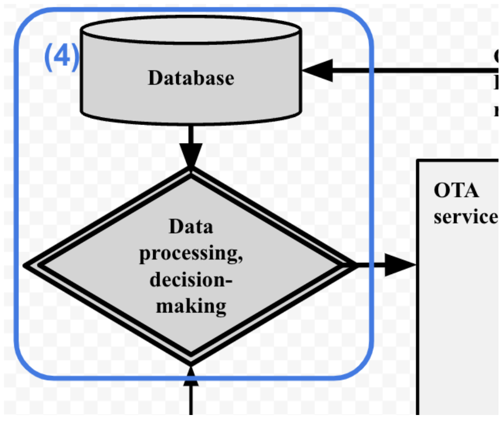
Parameters passed by the DPDM to the OTA service contain (but are not limited to) the following:

- Firmware version to be deployed – DPDM passes the version number, binary files, or link to files in the shared storage
- Pool of devices to be targeted – list of devices' unique identifiers, e.g., MAC addresses or Device Serial Numbers (DSN)
- Schedule, by which the rollout stage should be executed, e.g., exact time to start, or time range to execute
- Rollout severity – indicates whether to put this rollout stage higher in the queue of previously requested but not yet executed rollouts
- Other parameters might also be used, depending on implementation

The mechanism of the OTA updates heavily depends on many factors, like device type (manufacturer, model), used operating system (Linux, RTOS, bare metal software), communication channels (WiFi, Cellular, LoRa etc.), protocol used (MQTT, CoAP, HTTP etc.). In-depth reviews of OTA mechanisms and their use-cases are provided

Table 4

Phases of iterative deployment of OTA updates in IoT solution with CD approach

Phase	Diagram snippet
(1) Initiation of the deployment of OTA update.	 (1) OTA status reports Device Operability status reports Database Data processing, decision-making QA test, Stage test Test reports External inputs (expert decisions, business needs, etc.)
(2) Rollout of the OTA update to an initial pool (on iteration #1) and bigger pools (on each next iteration).	 (2) Data processing, decision-making OTA service Rollout 1 Rollout 2 Rollout 3 Rollout 4 Tiny pool Small pool Medium pool Large pool
(3) Collecting OTA and Device Operability status reports, putting them into the database, pre-processing.	 (3) Database OTA status reports Device Operability status reports Data processing OTA service Rollout 1 Rollout 2 Tiny pool Small pool
(4) Processing of the collected data from the previous iteration, making the decision on initiation of the next iteration (back to phase (2))	 (4) Database Data processing, decision-making OTA service

in related articles [8, 9]. This part is not covered in the article, as the proposed solution is considered device-agnostic.

On **Phase 3, the collection of OTA status reports and Device Operability status reports** happens. Status reports provide feedback on device's current firmware version, health state, performance, and connectivity.

The OTA status report contains (but is not limited to) the following information:

- OTA success state – it indicates whether OTA has been performed successfully or failed; if failed at which stage
- OTA duration – how long it took for the device to perform OTA; this value is important especially for IoT devices which cannot operate during ongoing OTA, so this indicates for how long the device was out of regular operation
- OTA attempts device has spent on conducting the OTA
- Reason for refusing OTA update, e.g., low battery level, bad connection conditions, low temperature, etc.
- Other important parameters, required for monitoring in each specific business case, like energy consumed, traffic consumed, program flash occupation, etc.

The Device Operability status report content heavily depends on the device types and business cases they are used for. For instance, the IoT security camera with the motion detection and video streaming functionality in its Device Operability status report contains (but is not limited to) the following information:

- Motion detection rate – number of motion events detected during a day; this is important for understanding if a new firmware has changed motion detection performance
- Video streaming quality – a set of metrics indicating the quality of video, like resolution, bitrate, video codec used for encoding, etc.
- Network losses during streaming – indicates the quality of the network during streaming, like video packet loss, failed video streams, etc.
- Energy consumption for video streaming – indicates the power consumed for 1 min of video streaming; this is important for battery-powered cameras
- Online availability – indicates for how long the device stayed online and has been in an offline state

On **Phase 4, Processing of the collected data and making a decision on the initiation of the next iteration** happens. To make a decision on each subsequent rollout stage, DPDM retrieves data collected on the previous stage from the database, verifies it against a set of predefined criteria, and makes a decision on the next stage. If the criteria are not met or the results cannot be analysed by DPDM, it may reiterate the rollout stage on a smaller pool or request a human (expert) interaction.

Criteria that the results are verified against, contain (but are not limited to) the following:

- Devices outage rate is within the acceptable range – indicates the percentage of devices that stopped operating after rollout
- OTA success rate is within the acceptable range – indicates the percentage of devices that conducted OTA update successfully, and within an expected time range
- OTA metrics are within the desired range – data collected from the OTA status reports, after aggregation (e.g., to represent average value), remains within the expected range, which indicates OTA is conducted as expected
- Device Operability metrics are within the desired range – data collected from the Device Operability status reports from the updated device, after aggregation (e.g,. to represent average value), remains within the expected range, which indicates devices operate as expected

Discussion and Future Research. The proposed integration of a Data Processing and Decision-Making (DPDM) unit into the OTA update deployment workflow for IoT systems addresses several key limitations of manual, expert-driven deployment processes. By automating decision-making based on predefined criteria and real-time feedback from devices, the approach reduces human error, improves scalability, and enables faster rollout iterations. The continuous deployment approach becomes applicable to the IoT solutions by putting most of the work to the DPDM unit, but still maintains flexibility by allowing expert inputs at key stages such as deployment initiation and rollout scheduling.

The proposed solution directly addresses the key challenges associated with continuous deployment in IoT environments, mitigating them or lowering their impact.

The proposed solution was designed to be platform-agnostic, with no reference to specific IoT device types, their operating systems, supported OTA mechanisms, communication channels (e.g., Wi-Fi, cellular, LoRa), or protocols used (e.g., MQTT,

Table 5

Addressing challenges of continuous deployment in IoT solutions by proposed solution based on DPDM

Connectivity issues	Mitigated through staged rollouts and DPDM-driven retries or pauses, allowing the system to adapt to unreliable network conditions without compromising the entire deployment
Resource constraints	Considered during the planning and configuration of rollout stages, enabling the DPDM to avoid overloading limited devices by managing update timing and batch sizes
Latency in feedback loops	Significantly reduced by automating data collection and analysis via the DPDM, enabling quicker decision-making compared to human-centered evaluation
Rollback complexity	Handled by incorporating device health metrics and operational status checks before proceeding to wider rollouts, ensuring issues are detected early and contained within smaller pools
Deployment Scale	Addressed by the iterative, data-driven rollout mechanism, which allows scalable orchestration of updates across thousands or millions of devices with minimal human oversight and improved consistency

CoAP, HTTP). Also, the article provided examples of the parameters exchanged throughout the deployment workflow across its phases. One direction of future research would be to continue investigating the platform-agnostic direction and defining and formalising all parameters exchanged throughout the deployment workflow. Identifying these parameters, their data types, dependencies, and relevance to various deployment scenarios will be critical for improving interoperability, ensuring consistency, and enabling cross-platform automation.

Another important direction of future research is the definition and formalisation of the rules and thresholds the DPDM uses to evaluate collected data and make decisions on the next rollout stages. Without well-defined rules, the decision-making process risks becoming inconsistent, potentially leading to slow deployments and unnecessary rollbacks. Future research should focus on developing a flexible rule engine capable of handling different rules and thresholds, with both static and dynamic criteria, and allowing adjustment for adapting to different IoT solutions and business needs.

To validate the feasibility and performance of the proposed solution, future research might also include an approbation phase. This could begin by selecting a real-world IoT solution, such as an environmental sensor network of security cameras, as a reference case. The proposed architecture should then be adapted to the specific characteristics of this system, including its device types, OTA mechanisms, and communication infrastructure. Furthermore, the solution can be tested through simulation environments or a small-scale physical IoT network, built using development boards and test devices to mimic the large-scale network. This approbation will provide valuable information about the solution behaviour under real constraints, help adjust thresholds and criteria, and ensure the practical applicability of the proposed deployment approach.

Conclusions. This article explores the application of the Continuous Deployment (CD) approach to the delivery of over-the-air (OTA) firmware updates in Internet of Things (IoT) solutions. While CD is widely used in cloud and web development, its use in IoT is limited by specific challenges such as connectivity issues, resource constraints, and deployment scale. The article proposes a new solution based on an automated Data Processing and Decision-Making (DPDM) unit that replaces manual expert intervention in the OTA rollout process. The proposed framework divides the update process into four phases and provides a workflow for automating update decisions based on real-time device data. This approach improves update speed, scalability, and reliability, and helps make CD a practical and efficient method for managing IoT firmware updates.

Bibliography:

1. Kumar S., Tiwari P., & Zymbler, M. Internet of Things is a revolutionary approach for future technology enhancement: a review. *Journal of Big Data volume* 6, *Article number* 111. 2019. DOI: 10.1186/s40537-019-0268-2

2. Lueth K.L. The Top 10 IoT Segments in 2018 – based on 1,600 real IoT projects, 2018. URL: https://surl.li/usvmpo

3. Ali Z.H., Ali H.A., Badawy M.M. Internet of Things (IoT): Definitions, Challenges, and Recent Research Directions. *International Journal of Computer Applications.* 128 (1) : 975-8887. 2015. P. 37–47.

4. Aboubakar M., Kellil M., Roux P. A review of IoT network management: Current status and perspectives. *Journal of King Saud University – Computer and Information Sciences* 34. 2022. P. 4163–4176. DOI: 10.1016/j.jksuci.2021.03.006

5. Rehan S., Syed A.. Automate application deployment to IoT devices using AWS IoT Device Management. 2023. URL: https://surli.cc/lxrpfb

6. Susnjara S., Smalley, I. What is continuous deployment? 2024. URL: https://www.ibm.com/topics/continuous-deployment

7. Swirski A. What you need to know for CI/CD for the IoT. 2022. URL: https://beetlebox.org/what-you-need-to-know-for-ci-cd-for-the-iot/

8. Bauwens J., Ruckebusch P., Giannoulis S., Moerman I., De Poorter E. Over-the-Air Software Updates in the Internet of Things: An Overview of Key Principles. *IEEE Communications Magazine.* 58. 2020. P. 35–41. DOI: 10.1109/MCOM.001.1900125.

9. Anurag T., Cory B., Poonam K. CoAP and MQTT Based Models to Deliver Software and Security Updates to IoT Devices over the Air. *2019 International Conference on Internet of Things* (*iThings*) *and IEEE Green Computing and Communications* (*GreenCom*) *and IEEE Cyber, Physical and Social Computing* (*CPSCom*) *and IEEE Smart Data* (*SmartData*), *Atlanta, GA, USA.* 2019. P. 1065–1070. DOI: 10.1109/iThings/GreenCom/CPSCom/SmartData.2019.00183.

Чуков О.О., Редько І.В. БЕЗПЕРЕРВНЕ ВПРОВАДЖЕННЯ OTA-ОНОВЛЕНЬ У IOT РІШЕННЯХ

У цій статті розглядаються виклики застосування безперервного впровадження (Continious Deployment) в IoT-рішеннях у контексті OTA-оновлень прошивки. Традиційний підхід CD, який використовується у розробці програмного забезпечення, не є безпосередньо придатним для середовища IoT через обмежені ресурси пристроїв, нестабільне з'єднання та необхідність обережного поетапного впровадження. У статті підкреслюється, що більшість OTA-оновлень в IoT-системах досі керуються вручну експертами, що уповільнює процес і створює ризики, пов'язані з людським фактором.

Щоб подолати ці проблеми та застосувати підхід CD до OTA-оновлень в IoT, у статті пропонується рішення, засноване на додаванні до процесу OTA-оновлення модуля Обробки Даних та Прийняття Рішень (Data-Processing and Decision Making, DPDM). Цей модуль автоматизує всю рутинну роботу, залишаючи експерту лише найважливіші рішення та критичні дії. Модуль приймає рішення щодо впровадження на кожному етапі, опираючись на початкові команди експерта та дані, отримані з попередніх ітерацій впровадження, включно зі звітами про статус OTA та працездатність пристроїв.

Запропоноване рішення вирішує низку проблем, пов'язаних із застосуванням CD в IoT, таких як масштаб впровадження, затримки у зворотному зв'язку, складність відкату, і водночас приносить переваги CD у сферу IoT – зниження витрат, пришвидшення циклу зворотного зв'язку та підвищення якості. Запропоноване рішення не залежить від типу пристроїв, подане у вигляді архітектурної схеми високого рівня і не містить конкретизації щодо використаних протоколів та способів реалізації.

Також у статті окреслено напрямі подальших досліджень, зокрема формалізацію параметрів, що використовуються в процесі розгортання, визначення правил і порогових значень для прийняття рішень щодо впровадження, а також валідацію рішення шляхом реального тестування або моделювання. Ці кроки є важливими для апробації застосовуванності підходу в різних IoT-системах і для подальшого переходу від концепції до практичного впровадження.

Ключові слова: *Інтернет речей, IoT, оновлення по повітрю, OTA, оновлення прошивки, неперервне впровадження, управіння пристроями.*