BULGARIAN ACADEMY OF SCIENCES

CYBERNETICS AND INFORMATION TECHNOLOGIES • Volume **24**, No 4
Sofia • 2024 Print ISSN: 1311-9702; Online ISSN: 1314-4081
DOI: 10.2478/cait-2024-0036

A Cost-Benefit Model for Feasible IoT Edge Resources Scalability to Improve Real-Time Processing Performance

Maen M. Al Assaf[1], *Mohammad Qatawneh*[1], *AlaaAldin AlRadhi*[2]

[1]*King Abdullah II School for Information Technology, University of Jordan, Amman, 11942, Jordan*
[2]*Seneca Polytechnic College, Toronto, Ontario, Canada*
E-mails: m_alassaf@ju.edu.jo mohd.qat@ju.edu.jo Aladdin.alradhi@senecapolytechnic.ca

Abstract: *Edge computing systems have emerged to facilitate real-time processing for delay-sensitive tasks in Internet of Things* (*IoT*) *Systems. As the volume of generated data and the real-time tasks increase, more pressure on edge servers is created. This eventually reduces the ability of edge servers to meet the processing deadlines for such delay-sensitive tasks, degrading users' satisfaction and revenues. At some point, scaling up the edge servers' processing resources might be needed to maintain user satisfaction. However, enterprises need to know if the cost of that scalability will be feasible in generating the required return on the investment and reducing the forgone revenues. This paper introduces a cost-benefit model that values the cost of edge processing resources scalability and the benefit of maintaining user satisfaction. We simulated our cost-benefit model to show its ability to decide whether the scalability will be feasible using different scenarios.*

Keywords: *IoT edge computing, Real-time processing, Enterprise feasibility study, User satisfaction, Resource scalability.*

1. Introduction

Edge computing has emerged recently as a supportive innovation for Internet of Things (IoT) systems and Wireless Sensor Networks (WSN) to provide them with assistance [1]. It facilitates fast real-time processing for delay-sensitive tasks requiring quick decisions to be issued before a certain deadline without relying on cloud resources [2, 31]. The use of IoT systems has increased recently in several domains and many IoT devices can easily join the system and continuously generate a huge amount of data [20]. This increased the volume of tasks that require real-time processing. IoT systems are used in domains such as health care, smart cities, smart industries, agriculture, smart grids systems, smart homes, retail systems, remote monitoring systems, financial systems, and others that require real-time data analysis and decisions at the edge to be produced before specific deadlines [1, 2, 31].

An IoT system mainly consists of IoT devices considered resource-constrained in terms of processing power and energy sources that rely on batteries [5]. These devices are connected to both, edge servers and the cloud servers, where the edge

servers process the real-time (delay-sensitive) tasks on the data collected from the IoT devices and take decisions without refereeing to the cloud [2]. Such real-time tasks must be processed before specific deadlines. Edge servers are located much closer to the IoT devices which helps them in achieving these goals. Since IoT devices are resource-constrained [6], they usually tend to be low-cost [5], so their scalability is much easier. As the IoT devices scale up, more volume of data will be generated.

In many cases, IoT devices do not have enough processing resources to process delay-sensitive tasks locally on the same devices, so they are offloaded and processed on the edge servers to meet their deadlines [6]. The remaining delay-tolerant tasks are offloaded to the cloud servers and their processing will take longer.

In the domains mentioned above, there exist enterprises that provide services for their customers (users) that require the use of IoT systems. Many enterprises have adopted IoT systems in their business models and eco-systems [8]. Even if the users' devices are not directly connected to the edge, users expect that the delay-sensitive tasks incorporated into the service should meet their processing deadlines. As more tasks are categorized as delay-sensitive tasks, more pressure will be placed on the edge servers. Edge servers have limited processing resources in comparison to cloud servers and might not have the same reactive scalability. Such an increase in delay-sensitive tasks offloaded to the edge servers will degrade their ability to meet the users' anticipated processing deadlines. In case users don't find that analysis and decisions performed on the edge were processed within the acceptable time frame, the enterprise may lose its users' satisfaction as user experience will be negatively affected [7]. Such degradation in user satisfaction will decrease the customer base of the enterprises that are running their service using the IoT system. This will reduce its revenues, as the customers will tend to become willing to spend less on the service or to completely abandon the service as sales are highly related to service latencies [25]. Some offloading algorithms may offload extra tasks attached to the provided service to the cloud server for processing in order to relax the edge servers and to make them able to meet the deadlines or to save power [6, 7]. However, if such tasks are delay-sensitive, it makes no sense for that offloading as these tasks will certainly miss their deadlines. This leads us to think about edge servers' resource scalability in case users' satisfaction is negatively affected. Such scalability involves increasing the processing resources at the edge servers and could be vertical on the existing servers or horizontal by adding more servers. Indeed, scalability in IT infrastructures has many benefits in maintaining business continuity [49].

However, as the edge servers' scalability will incur an upfront investment that the enterprises have to pay, it will be important to investigate if the scalability is really feasible. This can be achieved by building and running a cost-benefit model that reaches a decision on the feasibility of the edge resources' scalability by comparing the cost of the edge resources that need to be installed and their operations with the benefit of saving the amount of revenue that will be forgone due to the degraded user satisfaction.

In this paper, we introduce a cost-benefit model that consists of five phases to assess the need for scalability and to decide if it will have a feasible return on investment in case the benefit of saving the forgone user's revenues will surplus the

scalability cost. Some research addressed the issue of increasing user satisfaction in edge systems for delay-sensitive IoT applications [44]. To the best knowledge, our study is the first that addresses the limitations of available processing resources at the edge servers and helps to make scalability decisions that match with the benefit of maintaining user satisfaction.

One may say that the need for edge resources scalability is a fact when they become unable to meet the deadlines. However, it is important to evaluate the volume of the needed resources and if it will be feasible at the moment to implement them, especially in many cases, users may still prefer to keep using the service even if they encounter extra latencies. This depends on many factors such as the competitive advantage, the unavailability of substitutes, and the absence of competitors [27]. Our research has many motivations that are related to the abovementioned issues:

- IoT devices can scale up easily as many of such devices can join.
- Volume of data generated from IoT devices can increase rapidly.
- Offloading algorithms may not help in case of a large number of delay-sensitive tasks that are required to be processed at the edge.
- User satisfaction and revenues may decrease in case the IoT system is not properly performing decisions at the edge within the proper time frames.
- Users might keep using the service even if the performance is degraded due to the absence of competitors or the degradation has not reached a specific limit. This creates the necessity to value if the scalability provides a feasible return on investment.

The remaining parts of the paper illustrate related work, describe the system design, present our cost-benefit model, provide a simulation, and conclude our research.

2. Related work

This section provides an overview on concepts and current research related to our topic.

2.1. IoT and edge computing

The Internet of Things (IoT) has emerged and developed in recent years to connect things and objects through the Internet [9]. By IoT, the internet is not only used to connect general-purpose computer devices and servers, but it also connects special-purpose and mobile devices that can generate huge amounts of data from things of different business domains [10]. Devices such as sensors, actuators, wearable devices, and others. It uses internet protocols to make the object devices connected and visible through the internet. The number of these devices reached billions in recent years [15]. The IoT devices are used to generate data for a specific domain in which they are used to enrich data needed for processing. Data are processed locally or sent to the cloud systems to be processed. Processed data generate much informative knowledge as data analytics in many cases is used [16]. IoT and edge systems have created much need for cybersecurity as there exist many security weaknesses in any IoT system due to their constraints in resources especially as more

devices join [11, 34, 50]. Much cyber-crime evidence can be collected from such devices [33]. Key distribution is a major challenge in IoT systems security due to the need for authentication and identification of nodes connected using a shared communication medium, and there exist many key distribution schemes for IoT systems [17, 18]. Many distributed, centralized, flat, and hierarchical authentication techniques are available for IoT systems that researchers can consider [19]. Predicting malicious nodes that can join the IoT system using a trust management system is highly important [48]. Many researchers recently proposed solutions based on blockchain technologies to address IoT system security challenges and maintain information privacy and integrity by leveraging blockchain immutability and consensus abilities [14, 32]. Several lightweight cryptographic algorithms were proposed to meet the requirement of an IoT resource-constrained environment [47]. In [35], the authors reviewed the different IoT system's security challenges, authentication protocols, and potential attacks on the different IoT systems layers.

Many IoT systems are used for systems in which there exist real-time requirements where certain deadlines must be met. So, solutions used to reduce any delay in the system are highly important [12]. Many domains require IoT real-time decision making such as smart cities, Smart surveillance, health systems, water quality, and others. Real-time Air pollution and Gas leakage monitoring systems using IoT have become recently important to save the environmental effects [30, 40]. This requires solutions for enabling IoT systems to make feasible automatic real-time decisions on real-time data[13]. Edge computing emerged to integrate with IoT systems to provide fast processing for real-time services and data. In such systems, edge servers are installed near the IoT devices and their accessibility is fast in comparison to cloud servers that require a longer time to be reached [20].

2.2. Tasks offloading

IoT devices collect a huge amount of data that needs to be processed [2]. Some tasks are delay-tolerant where it will be acceptable if the processing takes longer than the expected duration, and others are delay-sensitive (real-time) where processing must be made before a specific deadline [6, 7]. Edge computing helps in processing delay-sensitive tasks to meet their anticipated deadlines, as they are located near the IoT devices, and hence, no need to travel a long distance through the network to process these tasks in the cloud servers which will take longer time [2]. Edge servers usually have limited resources in comparison with the cloud servers' resources [2, 23, 28]. So, they are only used for processing such delay-sensitive tasks where the delay-tolerant tasks are processed in the cloud servers. The offloading mechanism helps in classifying the delay-sensitive from the delay-tolerant tasks and if they must be moved to the edge of the cloud servers for processing. Several offloading algorithms makes load balancing for the offloaded tasks to ensure that they are equally distributed among the processing servers [7]. Some research joined offloading and load balancing to improve the system utility [26]. Many other offloading algorithms are used to reduce the power consumption in the IoT devices as these devices rely on batteries as a source of power [5, 6, 24]. Other researchers have addressed the problem of the huge traffic generated over the networks connecting IoT systems and

the limited bandwidth [23]. Offloading algorithms faces several challenges such as network dynamics changes, dynamic use behavior, and edge and cloud dynamics [46].

2.3. IoT in business

IoT systems have become important pilers in today's business as they create many opportunities for digital innovation and transformation for enterprises and help corporates to improve their productivity, revenues, and the business added value [36]. As mentioned, many business domains are using IoT systems in their operations. To leverage the IoT systems benefits for enterprises, there should be an IoT ecosystem, architecture, and business model to properly select and deploy the IoT systems and services as suggested by [38]. Having a proper ecosystem involves having developers for IoT hardware, software, applications, and networking besides the users/customers [38]. For enterprises, the IoT system will include many IoT devices such as sensors and actuators designed to perform the required tasks [10]. These devices are connected through the internet and generate huge amounts of data needed for processing and providing particular services for customers. Artificial Intelligence (AI), business intelligence, and analytics tools are used for processing such data to forecast many business issues and to provide appropriate decisions that increase user satisfaction. Many of such tools are real-time and are executed near the source of the data [39]. Even small and medium-sized businesses employ artificial intelligence and predictive analytics with IoT systems in their manufacturing process [42]. IoT systems in business can leverage business process management technologies [21]. Many challenges exist when deploying IoT systems for businesses such as readability, security, interoperability, and scalability [43]. In the Information Technology world, user satisfaction and loyalty are always linked to the performance of the Information system and many factors such as responsiveness, processing speed, reliability, perceived usefulness, and support [22, 37, 45]. There exists a user-perceived performance for the application where the user expects the response time of a request to happen within the expected time frame to stay satisfied and in the flow [4]. For those systems that use AI, accuracy is an important factor that influences user satisfaction [41]. User satisfaction in healthcare information systems is very sensitive and includes key factors such as privacy and ease of use [3]. In real-time and delay-sensitive applications, the increased latencies will lead to a low user experience and satisfaction [44]. Edge computing in IoT systems improves user experience for real-time and delay-sensitive applications as it reduces processing time and the need for processing at the cloud [24].

3. System design

In this section, we describe the IoT system for which we are providing our solution. In addition, we provide models for different phases that our cost-benefit model goes through including real-time (delay-sensitive) task processing, user satisfaction, the required scalability cost, and the present value of the future forgone revenues. This will give details on all the parameters that will be used in the cost-benefit model that

we will discuss in the next section. In addition, we illustrate and discuss some assumptions and limitations that we have in our research.

3.1. Modeling the IoT system

For a particular enterprise or organization that provides a particular service for their customers (users) in a particular business domain that requires the use of IoT systems, there exists an application (process) installed on the users' devices. The users' devices are usually general purpose computers such as laptops, desktops, or smart mobile devices. Through the application, the user can request and receive services from the cloud system in the domain of services provided by the enterprise by communicating with the cloud servers. In case IoT devices are involved in providing the service, the system will be expanded to include also edge server and IoT devices to collect and process the specific required data and tasks. Any typical IoT system consists of several special-purpose IoT edge devices where each device can collect a specific type of data or perform a specific type of action. They are deployed depending on the domain of service provided by a specific enterprise where they serve.

The IoT system also consists of edge servers that are implemented near the IoT devices to process delay-sensitive tasks that require immediate processing without being offloaded to the cloud. Such tasks perform real-time decisions at the edge that must be processed before a specific deadline. In addition, the system consists of a cloud system where the users communicate through their applications to request and receive a service. In many other systems, end users may also communicate with the edge servers immediately [7]. Hence, we mainly focus on the delay-sensitive tasks to be executed for the IoT devices rather than the tasks that are processed on the end users' devices. In case the service requires many real-time decisions to be taken on the data collected from the IoT devices, the role of edge servers becomes more significant as more delay-sensitive tasks will be processed on them. As they process the delay-sensitive tasks, there exit deadline in terms of processing time that needs to be met on the processing of each task. Failure to meet the processing deadline causes user dissatisfaction and a reduction in enterprise revenue in case many users leave the service. Fig. 1 summarizes the model.

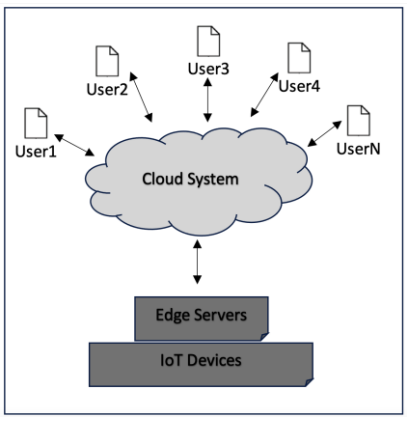

Fig. 1. Modeling the IoT system

3.2. Modeling real-time tasks

To enable our cost-benefit model to find if the delay-sensitive tasks that are running on the edge servers are meeting their expected deadlines, there should be parameters that will be used for that concern. Here we define and describe the parameters related to different delay-sensitive tasks executed on the edge servers, their processing time, and their deadlines. In [4], system response time represents the duration between the time of submitting a request and the time when the result shows up. Users have user-perceived time for each particular application, which determines the limit of the response time. This affects user satisfaction in case the user-perceived performance is not met. Similarly, for any enterprise that provides a service for its users, the users expect that processing is performed within an acceptable time frame and the responsiveness of the system is maintained. In case the processing latency shows frequent increases that span longer than a particular period, users will start complaining and thinking about abandoning the service. Let T_{sats} be the maximum Time duration in which users can tolerate the frequent degradation of the overall edge server performance in processing the delay-sensitive tasks and meeting the expected deadlines. Such a decrease in performance will show many delays in decisions that are taken in the edge. In case the service keeps these frequent delays, users will choose to pay less for the service or to abandon the service and the enterprise will notice a significant reduction in their revenues. In case the overall performance is not maintained for a duration that exceeds T_{sats}, our cost-benefit model is triggered to determine the feasibility of any potential edge servers' scalability. The duration of T_{sats} varies among different service domains and its value is setup by the enterprise itself where each enterprise can determine the duration of time in which it has reached a significant level of revenue losses based on its accounting system and the enterprise cannot accommodate more losses. For example, in many healthcare systems that uses IoT devices that are directly close to the patients, there exists not much room for delay tolerance in processing tasks that do the decisions related to patients [29], so, T_{sats} value will be small. Other domains like remote monitoring of agricultural systems may tolerate delays that may repeat frequently during certain periods. In some domains, T_{sats} might be hours, days, weeks, or months depending on how much it is acceptable to tolerate the reduction in performance. For the delay-sensitive tasks that are processed on the edge servers, let Tsk be the set of all of them where: $Tsk = \{Tsk_1, Tsk_2, Tsk_3, \ldots, Tsk_N\}$ for N tasks that are load-balanced on the edge servers. Let (P_{avg}) be the set of the average Processing time that the tasks took to be processed during the last (T_{sats}) period, where $P_{avg} = \{P_{avg_1}, P_{avg_2}, P_{avg_3}, \ldots, P_{avg_N}\}$ for each of the N tasks. P_{avg} is calculated continuously for the recent T_{sats} duration. This requires keeping a history of the processing time (how much time the task took to be processed) for each time the task was scheduled for processing during the recent T_{sats} duration. The history is cleared for those entries that precede the recent T_{sats} duration. In addition, let D be the set of the anticipated deadline for the tasks where each task processing time should not exceed its corresponding deadline duration where $D = \{D_1, D_2, D_3, \ldots, D_N\}$ for each of the N tasks. As we will be using a weighted average to value the average processing time that the tasks took during the recent T_{sats} and their corresponding deadlines, we will set weights for the tasks according to their

deadlines where tasks with a shorter deadline will have higher weight. Our motivation behind that is the fact that tasks with shorter deadlines tend to be more urgent. Lets W be the set of Weights for the tasks where $W = \{W_1, W_2, W_3, \ldots, W_N\}$ for each of the N tasks.

3.3. Modelling user satisfaction

After exceeding a particular T_{sats} where the processing at the edge servers is exceeding the anticipated deadlines, user satisfaction might tend to decrease and users will start to pay less for the service or to completely abandon the service. In both cases, there exists initial REVenues (REV_{before}) which is the total revenue that the enterprise has collected from its users during the T_{sats} period that precedes the recent T_{sats} in which user satisfaction has degraded. We assume that REV_{before} reflects a good level of revenues that financially satisfies the enterprise. After a period of T_{sats}, the enterprise will find that the revenues gained during the last T_{sats} period will be less than REV_{before}. Let REV_{after} be the total REVenue that the enterprise has collected during the recent T_{sats} where the enterprise should have lost much of its users or lost their willing to pay. In case REV_{after} is much less than REV_{before}, then the scalability option needs to be decided. The difference between REV_{after} and REV_{before} reflects the forgone revenues that the enterprise recently has lost which indicates that a potential scalability might be needed.

3.4. Modeling required scalability volume and cost

In case user satisfaction has decreased, and the enterprise has encountered significant revenue loses for a duration of time T_{sats} that is determined by the enterprise, scaling up the edge servers resources might be a good decision. However, to enable our cost-benefit model to instruct the enterprise either to scale up the resources or not, the model has to value the upfront investment and the operational costs for scalability. Adding more resources will incur an immediate payment for buying new resources as well as their continuous operational costs. Estimating such values helps the model to determine if such investment will have a feasible Return On Investment (ROI) after a particular period determined by the enterprise. Let $Scal_{Req}$ be the volume of extra resources that need to be added to improve the processing performance at the edge servers and to enable them to meet the expected deadlines for delay-sensitive tasks that are executed on their side. Let Inv_{Exist} be the original investment in money that was paid for the existing edge servers' resources before the scalability. Let $Scal_{Inv}$ be the upfront Investment that needs to be paid to scale up the existing system for the amount $Scal_{Req}$ of resources.

3.5. Modelling net present value of the actual future forgone revenues

Scaling up the edge servers will incur an upfront investment that has to be paid immediately to add the required volume of resources as mentioned in the previous subsection. One of the objectives of our cost-benefit model is to compare the upfront investment $Scal_{Inv}$ with the net present value of the projected (anticipated) actual forgone revenues NPV_{Afr} that the enterprise will lose in the next few future financial periods in case of not scaling up the system. To calculate that net present value, the

enterprise has to determine the number of future Financial Periods (F_{Prd}) in which it wishes to have its Return On Investment (ROI) and the Discount Rate (D_{Rate}) needed for that calculation. In addition, the enterprise has to calculate a projected value of the Forgone Revenues (F_{Rev}) that it will lose on each of the given future financial periods due to the absence of scalability. Also, the enterprise has to calculate a projected value for the OPerational Costs (OP_{Cost}) that will be incurred on each given future financial period in case the system is scaled up. The set of differences between the forgone revenues and the operational costs in each period will reflect the Actual Forgone Revenues (AF_{Rev}) of that period.

Table 1 summarizes all the parameters modeled in the Sections 3.2, 3.3 3.4, and 3.5.

Table 1. Parameters summery

Parameter	Description	Parameter	Description
T_{sats}	Maximum duration of time in which the degradation in system performance is tolerated by users	Inv_{Exist}	The investment in money on the existing resources that were deployed before the need for scaling up the system
Tsk	Delay-sensitive tasks that are processed on the edge	$Scal_{Inv}$	The upfront investment in money that is needed to deploy the needed scalability
*P*avg	Average Processing Time for each Task in Tsk during the Previous T_{sats}	NPV_{AFR}	Net present value for the future actual forgone revenues
D	Anticipated processing time deadline duration of time for each task of Tsk that has to be met every time the task is processed	AF_{Rev}	The set of actual forgone revenues for the given future financial periods
W	The weight of each task in Tsk is based on its processing deadline	F_{Rev}	The value of forgone revenues that the enterprise will lose on a particular future financial period
REV_{before}	Total revenues collected during the T_{sats} period that precedes the recent T_{sats}	OP_{Cost}	The extra operational cost that will be incurred on each future financial period due to the scalability
REV_{after}	Total revenues collected during the recent T_{sats} period	F_{Prd}	The number of future financial periods where the enterprises wish to achieve its ROI
$Scal_{Req}$	The volume of resources that need to be added for scalability	D_{Rate}	Discount rate used for calculating the net present value

3.6. Assumptions and limitations

Here we discuss some of our model assumptions and limitations that we have in our research.

- Our main objective in this research is to present a cost-benefit model for feasible edge resources scalability and simulate that model using realistic values to see how it is responding. Real-world enterprise evaluation will be a future work.
- We evaluate the edge servers' needed processing resources as a volume without going into the hardware details and the exact resources that have to be added. Similarly, we do when we evaluate the cost of the scalability by increasing the existing volume of resources to make edge servers able to meet the processing

deadlines. However, dealing with exact hardware needed in the scalability is possible in our cost-benefit model when inputting real data. The scalability might be vertical or horizontal depending on the actual required hardware.

- We assume that delay-sensitive tasks offloaded to the edge servers are well load-balanced. This makes the servers to have similar workload and facilitates the process of evaluating the required volume of resources for scalability.
- We assume that the IoT system is used by any enterprise in any business domain.

4. Cost-benefit model for edge resources scalability based on user satisfaction.

Our cost-benefit model passes through five phases that will eventually provide an instructive decision for the enterprise whether the scalability of the edge servers processing resources at the moment is feasible. The first phase will calculate the average processing time that the delay-sensitive tasks took recently at the edge servers and compare that with their corresponding anticipated average deadlines to find out how the system is performing. The second phase will calculate the change in user satisfaction in case it was recently affected negatively due to the degradation in the system performance. The third phase calculates the needed volumes and cost for the scalability. The fourth phase calculates the anticipated future losses in revenues in case the system continues without scalability. The fifth phase decides whether the scalability will be feasible in case it finds that the system is not performing well, the user satisfaction is affected, and the scalability investment will recover the future actual losses in revenue.

4.1. Phase 1: Monitoring the edger servers' performance

In this phase, the model will monitor if the edge servers' resources are meeting the processing deadlines for the processed tasks.

For a given N number of delay-sensitive tasks (Tsk) where Tsk = {Tsk_1, Tsk_2, Tsk_3, …, Tsk_N} that will be processed on the edge servers. In addition, for a given set of deadlines Duration (D) for the N tasks where $D = \{D_1, D_2, D_3, \ldots, D_N\}$ in which each task is anticipated to be processed before its corresponding deadline, the model assigns Weights (W) for each of the N tasks where $W = \{W_1, W_2, W_3, \ldots, W_N\}$ based on its deadline values. For weight calculation, the model finds out the number of distinct deadline values x in D and assign the weights from 1 up to x where the tasks with the lowest deadline value receives the weigh x that reflects the highest weight value. As the tasks' deadline values increase, their urgency become less and they receive lower weight values until reaching to 1. Tasks with the same deadline values will receive the same weight value. The model then calculates weighted average for deadlines W_D in the next equation,

$$W_D = \frac{\sum_{i=1}^{N}(W_i \times D_i)}{\sum_{i=1}^{N} W_i}. \quad (1)$$

For the recent (T_{sats}) duration, the model will keep recording in a history (a temporary storage) the processing time that the tasks took for all the times they

were scheduled for processing at the edge servers. Then, it calculate the average Processing time (P_{avg}) where $P_{\text{avg}} = \{P_{\text{avg}_1}, P_{\text{avg}_2}, P_{\text{avg}_3}, \ldots, P_{\text{avg}_N}\}$ for each of the N tasks during the recent T_{sats} duration. This process continues as the time passes and the T_{sats} window frame moves. The model then keeps calculating the Weighted average for the average Processing times W_{Pavg} that is calculated in the equation

$$W_{P_{\text{avg}}} = \frac{\sum_{i=1}^{N}\left(W_i \times P_{\text{avg}_i}\right)}{\sum_{i=1}^{N} W_i}. \quad (2)$$

In that equation, Phase 1 of the model will decide either there exists a degradation in the system performance or not (P_{deg}), and hence, the model should continue to the next phase or to stop. This depends on whether the average processing time of the delay-sensitive tasks tends to exceed the anticipated deadlines. The model used weighted average in this calculation as the tasks with early deadlines have to be given more significance when using the average in such a comparison.

$$P_{\text{deg}} = \begin{cases} 1 & \text{if} \quad W_D < W_{P_{\text{avg}}}, \\ 0 & \text{otherwise.} \end{cases} \quad (3)$$

In case $P_{\text{deg}} = 1$, this indicates that the edge servers processing resources are not performing well and the scalability option needs to be future investigated in the next phases of the model. In case $P_{\text{deg}} = 0$, this means that the edge servers are performing well and they are meeting in average the processing deadlines. Hence, no edge resource scalability will be needed. In case the scalability is not needed, and the enterprise is losing its users satisfaction and revenues, the reason will be beyond the edge resources performance and might be from many other reasons, such as losing the competitive advantage or the emergence of other substitutes.

4.2. Phase 2: Monitoring user satisfaction

In case the model has found that the edge servers' performance is degraded where they are unable to meet the delay-sensitive tasks processing deadlines, it will trigger the second phase which monitors if the user satisfaction was affected where significant number of users became paying less for the service or abandoned the service. As mentioned previously, T_{sats} reflects the duration of time in which the enterprise will lose significant amount of revenues as the users will not be able to accept the decrease in performance. The enterprise should be able to determine this duration as well as the total revenues in the duration before $\text{REV}_{\text{before}}$ and the new total of the revenues after $\text{REV}_{\text{after}}$ for the recent T_{sats}. In Equation (4), Phase2 of the model will decide either there exists a degradation in the user satisfaction (Rev_{deg}) in case there exists decreased revenues, and hence, the model should continue to the next phase or to stop.

$$\text{Rev}_{\text{deg}} = \begin{cases} 1 & \text{if} \quad \text{REV}_{\text{before}} < \text{REV}_{\text{after}}, \\ 0 & \text{otherwise.} \end{cases} \quad (4)$$

In case $\text{Rev}_{\text{deg}} = 1$ in that equation, this indicates that users are not satisfied, and the model should continue further to the next phase. In case $\text{Rev}_{\text{deg}} = 0$ in Equation (4), the model stops as the scalability is not necessary at the moment. This reflects the case where the users are still paying well for the service, and they are willing to accept using the service even if there exists a degraded performance. As mentioned previously, this depends on many business factors such the domain of the service and the competitive environment.

4.3. Phase 3: Calculating the scalability cost

After determining if the degradation of the edge servers' performance in processing delay-sensitive tasks has caused a reduction in user satisfaction and revenues, the model calculates the required resources volume and cost for scalability. The ratio of variation between the Weighted average for the deadlines (W_D) that was calculated in Equation (1) and the Weighted average for the average Processing times ($W_{P_{avg}}$) that was calculated in Equation (2) reflects the volume of the needed resources. Adding this volume will make the resources able to process the tasks before or within their deadlines, and hence, $W_{P_{avg}}$ will become less than or equals to W_D. The next equation shows the calculation for the volume of the required Scalability ($Scal_{Req}$) in percentage (%) that is needed to be added on the top of the existing resources,

$$Scal_{Req} = \left(\left(\frac{W_{P_{avg}}}{W_D}\right) - 1\right) \times 100\%. \quad (5)$$

From this equation, the model can then calculate the upfront investment in money ($Scal_{Inv}$) that is needed to deploy the scalability after knowing the cost of the investment that the enterprise has paid previously on the existing edge servers processing resources (Inv_{Exist}). This information is known and prompted by the enterprise accounting system systems to the model. Then it will be able to calculate the required money for scalability ($Scal_{Inv}$) by the next equation

$$Scal_{inv} = \left(Scal_{Req} \times Inv_{Exist}\right) \times (1 + \in). \quad (6)$$

where $\in$ is the inflation rate since the original investment prompt by the enterprise accounting system if applicable.

4.4. Phase 4: Anticipating actual future forgone revenue

In the previous phase, the upfront investment of scalability was calculated. To decide if the scalability will be feasible, the upfront investment should be compared with actual future decrease in revenues that might span for several financial periods in case of choosing not scale up the system. Hence, the model in this phase asks the enterprise accounting system to prompt the number of future Financial Periods (F_{Prd}) in which the enterprise is looking to achieve the scalability Return On Investment (ROI), to provide a set of projected values to calculate the Actual Forgone Revenues (AF_{Rev}) that are anticipated to be lost in the F_{Prd} future periods, and to provide a Discount Rate (D_{Rate}). These values will be used in the last phase to determine the feasibility for scaling up the resources by calculating the Net Present Value of the Actual Forgone Revenues (NPV_{AFR}) for the future F_{Prd} to be compared with the upfront investment that will be paid for the scalability. To calculate the set of the actual Forgone Revenues (AF_{Rev}) that are anticipated to be lost in the upcoming F_{Prd} future periods in case no scalability takes place, enterprise will enter the value of the projected Forgone Revenues (F_{Rev}) for each of the upcoming F_{Prd} period as well as the expected extra OPerational Costs (OP_{Cost}) that will incur each period in case the system was scaled up. Their absolute difference represents that actual forgone revenues on each period in case of no scalability assuming that forgone revenues are always higher than the operational costs. In fact, the enterprise has enough information from previous the phases to determine F_{Rev} and OP_{Cost} for each F_{Prd}. The next equation shows AF_{Rev} set calculation,

(7) $$\mathrm{AF}_{\mathrm{Rev}} = \left\{ \left| F_{\mathrm{Rev}_i} - \mathrm{OP}_{\mathrm{Cost}_i} \right|, \ldots, \left| F_{\mathrm{Rev}_{F_{\mathrm{Prd}}}} - \mathrm{OP}_{\mathrm{Cost}_{F_{\mathrm{Prd}}}} \right| \right\} \quad \forall\, i \in \{1, \ldots, F_{\mathrm{Prd}}\}.$$

4.5. Phase 5: Scalability decision

The final phase is to determine whether the scalability is feasible. This is achieved by comparing the Net Present Value of the Actual future Forgone Revenues ($\mathrm{NPV}_{\mathrm{AFR}}$) expected in the next F_{Prd} financial period to the prompted Discount Rate (D_{Rate}) with the upfront scalability investment. The $\mathrm{NPV}_{\mathrm{AFR}}$ is calculated on the set of forgone revenues calculated in Equation (7). The next equation shows the $\mathrm{NPV}_{\mathrm{AFR}}$ calculation,

(8) $$\mathrm{NPV}_{\mathrm{AFR}} = \sum_{i=1}^{F_{\mathrm{Prd}}} \frac{\mathrm{AF}_{\mathrm{Rev}_i}}{(1+D_{\mathrm{Rate}})^{F_{\mathrm{Prd}}}}.$$

After calculating the required investment for scalability ($\mathrm{Scal}_{\mathrm{Inv}}$) in Equation (6) and the net present value of the future actual forgone revenues ($\mathrm{NPV}_{\mathrm{AFR}}$) in Equation (8), the model can now decide if the scalability will be feasible and will achieve the anticipated Return On its Investment ROI by the equation

(9) $$\mathrm{Scal}_{\mathrm{Decision}} = \begin{cases} 1 & \text{if } \mathrm{Scal}_{\mathrm{Inv}} < \mathrm{NPV}_{\mathrm{AFR}}, \\ 0 & \text{otherwise.} \end{cases}$$

Hence, if $\mathrm{Scal}_{\mathrm{Decision}} = 1$, this indicates that the scalability will be feasible and will achieve its anticipated ROI when scaling up the resources for extra $\mathrm{Scal}_{\mathrm{Req}}$ volume. If $\mathrm{Scal}_{\mathrm{Decision}} = 0$, this indicates that the scalability investment that will be paid immediately will exceed its return in improving user satisfaction. Indeed, all enterprises look for their user satisfaction, however, in case the model states that scalability is not feasible while the users are dissatisfied, the enterprise must find other ways to compensate their users such as lowering the cost of the service or offering them more incentives.

5. Cost-benefit model simulation

In this section, we simulate the cost-benefit model using three scenarios. The first scenario is when the edge servers are meeting the processing deadlines for the real time tasks, and hence, no scalability is needed. The Second Scenario is when there exists a degradation in the processing performance and user satisfaction where the scalability does not recover its investment cost. In the third scenario, we simulate the case when there exists a degradation in the processing performance and user satisfaction, and the scalability will be feasible and able to recover its investment cost.

5.1. Simulator design

We built a simulator using Python to simulate the cost-benefit model using a randomly generated data set that reflects real-world scenarios. In the generated data set, there exists 1000 delay sensitive tasks that have random values of pre-defined processing deadlines in milliseconds. They are executed a random number of times during an arbitrary value of T_{stats} that equals to 48 hours where the processing time for each execution is recorded in milliseconds. Each task is assigned a weight based on its deadline. For the different scenarios, the simulated edge servers processing resources are set to have different processing capacities that vary in their capabilities

to meet the processing deadlines for the simulated tasks. For the other parameters that are related to user satisfaction and revenues, required scalability costs, and the net present value of the actual future forgone revenues, we use realistic arbitrary values that represent the different scenarios.

5.2. Scenario 1: Edge servers are meeting processing deadlines; scalability is not needed

Due to the existence of sufficient edge processing resources that are set in the simulator, the edge servers are meeting the processing deadlines for all the 1000 delay-sensitive tasks and for all the times they were executed during the previous T_{stats} of 48 hours. Hence, no scalability is needed as P_{deg} will equal to 0. Fig. 2 shows the average Processing Time and the Deadlines (in Milliseconds) for the tasks. The tasks deadlines were met in all the times they were executed. Due to the space limitation, Fig. 2 illustrates the average processing time for each task.

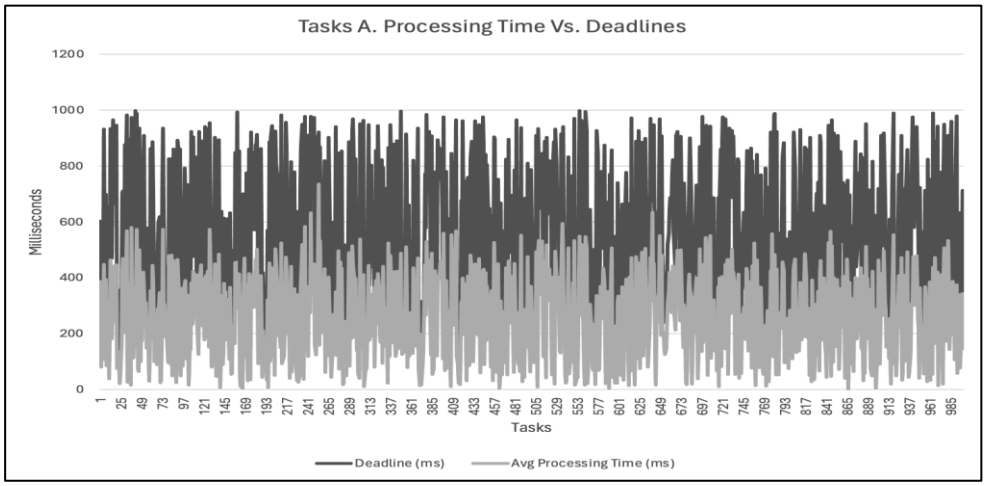

Fig. 2. Average processing time vs. deadlines (ms). Tasks processed within their deadlines

The Weighted average for the average Processing times ($W_{P_{\text{avg}}}$) and the Weighted average for the average Deadlines (W_D) were calculated and illustrated in Table 2. It is clear the $W_{P_{\text{avg}}}$ is less than W_D, and hence, the model will stop as there exists no degradation in the system performance.

Table 2. Scenario 1: $W_{P_{\text{avg}}}$ and W_D values

Parameter	Value
W_D	347.45 ms
$W_{P_{\text{avg}}}$	174.52 ms

5.3. Scenario 2: Edge servers' performance degradation with non-feasible scalability

We re-executed our simulator after sitting up the processing resources at the edge servers to be insufficient for meeting the simulated tasks processing deadlines. We simulated the same 1000 delay-sensitive tasks and for all the times they were executed during the previous T_{stats} of 48 hours. P_{deg} in this case will equals to 1 and

hence, scalability is needed if it will be feasible. So, we need to proceed to the second phase of the cost-benefit model. Fig. 3 shows the average processing time W_{Pavg} and the average deadlines (W_D) (in ms) for the tasks. Most of the tasks have missed their processing deadlines in most of the times in which they were executed. Due to the space limitation, Fig. 3 illustrates the average processing time for each task.

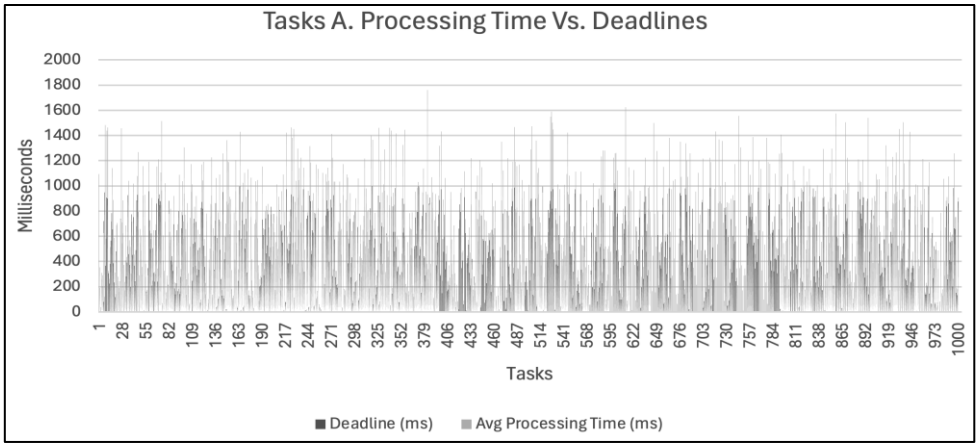

Fig. 3. Average processing time vs. deadlines (in ms). Most of tasks missed their processing deadlines

The weighted average for the average processing times $W_{P_{avg}}$ and the Weighted average for the average Deadlines W_D were calculated and illustrated in Table 3.

Table 3. Scenario 2: $W_{P_{avg}}$ and W_D values

Parameter	Value
W_D	325.94 ms
$W_{P_{avg}}$	424.47 ms

The model will prompt the values of the REV_{before} and the REV_{after} for the recent T_{sats} duration of 48 hours. Assume arbitrary values for REV_{before} and REV_{after} of \$1K and \$0.9K, respectively. This shows a slight reduction in user satisfaction where REV_{deg} equals to 1. This indicates the need for the scalability, and the cost-benefit model moves to the third phase for calculating the required resources and the scalability cost. The $Scal_{Req}$ will equal to 30.2% based on the W_D and the $W_{P_{avg}}$ values in Table 3. To calculate the cost for the scalability $Scal_{Inv}$, the model will prompt the initial investment that was paid previously for the existing edge processing resources Inv_{Exist} and the inflation rate since the original investment $\in$. Assume that Inv_{Exist} equals to \$90K and $\in$ equals to 5%. Based on the model, $Scal_{Inv}$ in this case will equals \$28.5K. Table 4 summarizes user satisfaction and scalability cost parameters.

The model moves to Phase 4 for anticipating the actual future forgone revenue that will be used for calculating its net present value. In this phase, the enterprise will need to prompt several parameters as mentioned in the model. Assume the values for these different parameters as shown in Table 5.

Table 4. Scenario 2: Parameters values for user satisfaction and scalability investment

Parameter	Value
REV_{before}	\$1K
REV_{after}	\$0.9K
$Scal_{Req}$	30%
Inv_{Exist}	\$90K
$\in$	5%
$Scal_{Inv}$	\$28.5K

Table 5. Scenario 2: Parameters values for actual future forgone revenue

Parameter	Value
F_{Prd}	Three periods
D_{Rate}	5%
F_{Rev} for 3 periods	{\$8.1K, \$7.2K, \$9.9K}
OP_{Cost} for 3 periods for $Scal_{Req}$ volume	{\$1.2K, \$1.2K, \$1.2K}

The model calculates the set of the AF_{Rev} for the F_{Prd} periods and then moves to Phase 5 to calculate their NPV_{AFR}. Their calculations are shows in Table 6.

Table 6. Scenario 2: AF_{Rev} and NPV_{AFR} calculations

Parameter	Value
AF_{Rev} for three periods	{\$6.9K, \$6K, \$8.7K}
NPV_{AFR}	\$18.65K

Phase 5 of the model will now value if the scalability is feasible. Since the NPV_{AFR} is less than the $Scal_{Inv}$, then $Scal_{Decision}$ will equal to 0 which indicates that it will not be feasible to scale up the edge processing resources for the volume of $Scal_{Req}$ resources.

5.4. Scenario 3: Edge servers performance degradation with feasible scalability

In this scenario, we will use the parameters from Scenario 2 except some changes. After the cost-benefit model has verified that there is a degradation in system performance through phase1, there is a reduction in user satisfaction in Phase 2, and calculated the required volume and cost for scalability (i.e., $Scal_{Req}$ = 30.2% and $Scal_{Inv}$ = \$28.5K).The model will proceeds to Phase 4 to prompt values for parameters needed for calculating the actual future forgone revenue that will be used for calculating its net present value. Assume the values shown in Table 7.

Table 7. Scenario 3: Parameters values for actual future forgone revenue

Parameter	Value
F_{Prd}	Three periods
D_{Rate}	5%
F_{Rev} for three periods	{\$16.5K, \$14.4K, \$18.8K}
OP_{Cost} for three periods for $Scal_{Req}$ volume	{\$1.2K, \$1.2K, \$1.2K}

The model calculates the set of the AF_{Rev} for the F_{Prd} periods and then moves to Phase 5 to calculate their NPV_{AFR}. Their calculations are shows in Table 8.

Table 8. Scenario 3: AF_{Rev} and NPV_{AFR} calculations

Parameter	Value
AF_{Rev} for three periods	{\$15.3K, \$13.2K, \$17.6K}
NPV_{AFR}	\$39.82K

As the enterprise's accounting system anticipates a significant loses in revenues. These loses will incur in the next three financial periods due to the high reduction in user satisfaction in case of no scalability takes place. In this case, the model will recommend for the enterprise to scaleup the edge resources for a volume of $Scal_{Req}$ of resources as the $Scal_{Decision}$ value will equal to 1. So, the scalability will be feasible and will achieve its return on investment during the upcoming F_{Prd} periods.

6. Conclusion

In this paper, we introduced a cost-benefit model that decides the feasibility of edge servers processing resources scalability. In IoT edge systems, there exists many delay-sensitive tasks that must be processed within specific deadlines to issue real-time decisions on the collected data. In case the edge servers are not being able to meet the anticipated processing deadlines, the enterprise might lose their users satisfaction and hence, will have decreased revenues. For that reason, we built and simulated a cost benefit model of five phases that decides the feasibility of scaling up the edge servers processing resources in case the scalability investment cost will achieve its return on investment within particular financial periods.

References

1. Li, X., et al. Edge Computing-Enabled Wireless Sensor Networks for Multiple Data Collection Tasks in Smart Agriculture. – Journal of Sensors, 2020, pp. 1-9.
2. Cao, K., et al. An Overview on Edge Computing Research. – IEEE Access, 2020, pp. 85714-85728.
3. Kitsios, F., et al. e-Service Evaluation: User Satisfaction Measurement and Implications in Health Sector. – Computer Standards & Interfaces, Vol. **63**, 2019, pp. 16-26.
4. Doherty, R. A., P. Sorenson. Keeping Users in the Flow: Mapping System Responsiveness with User Experience. – Procedia Manufacturing, Vol. **3**, 2015, pp. 4384-4391.
5. Capra, M., et al. Edge Computing: A Survey on the Hardware Requirements in the Internet of Things World. – Future Internet, Vol. **11**, 2019, No 4, 100.
6. Babar, M., M. S. Khan. ScalEdge: A Framework for Scalable Edge Computing in Internet of Things – Based Smart Systems. – International Journal of Distributed Sensor Networks, Vol. **17**, 2021, No 7.
7. Zhang, Z., et al. A New Task Offloading Algorithm in Edge Computing. – EURASIP Journal on Wireless Communications and Networking, Vol. **2021**, No 1, 2021, 17.
8. Lee, I. The Internet of Things for Enterprises: An Ecosystem, Architecture, and IoT Service Business Model. – Internet of Things, Vol. **7**, 2019, 100078.
9. Ray, P. P. A Survey on Internet of Things Architectures. – Journal of King Saud University-Computer and Information Sciences, Vol. **30**, 2018, No 3, pp. 291-319.
10. Twilion, T. G. What are IoT Devices? 2022. **https://www.twilio.com/en-us/blog/what-are-iot-devices**
11. Ande, R., et al. Internet of Things: Evolution and Technologies from a Security Perspective. – Sustainable Cities and Society, Vol. **54**, 2020, 101728.
12. Hussein, M., et al. Design and Implementation of IoT Platform for Real Time Systems. – In: Proc. of International Conference on Advanced Machine Learning Technologies and Applications (AMLTA'2018). Springer International Publishing, 2018.
13. Zyrianoff, I., et al. Scalability of Real-Time Iot-Based Applications for Smart Cities. – In: Proc. of IEEE Symposium on Computers and Communications (ISCC'18), IEEE, 2018.
14. Qatawneh, M. Use of Blockchain in the Internet of Things: A Survey, 2023. **https://arxiv.org/ftp/arxiv/papers/2303/2303.06035.pdf**

15. P r e m s a n k a r, G., et al. Edge Computing for the Internet of Things: A Case Study. – IEEE Internet of Things Journal, Vol. **5**, 2018, No 2, pp. 1275-1284.
16. S i o w, E., T. T i r o p a n i s, W. H a l l. Analytics for the Internet of Things: A Survey. – ACM Computing Surveys (CSUR), Vol. **51**, 2018, No 4, pp. 1-36.
17. A b u A l g h a n a m, O., et al. A New Hierarchical Architecture and Protocol for Key Distribution in the Context of IoT-Based Smart Cities. – J. Inf. Secur. Appl., Vol. **67**, 2022, 103173.
18. A b u A l g h a n a m, O., et al. A Survey of Key Distribution in the Context of Internet of Things. – Journal of Theoretical and Applied Information Technology, Vol. **97**, 2019, No 22.
19. S a a d e h, M., A. S l e i t, M. Q a t a w n e h, W. A l m o b a i d e e n. Authentication Techniques for the Internet of Things: A Survey. – In: Proc. of Cybersecurity and Cyberforensics Conference, 2016.
20. C a o, K., et al. An Overview on Edge Computing Research. – IEEE Access, Vol. **8**, 2020.
21. J a n i e s c h, C., et al. The Internet of Things Meets Business Process Management: A Manifesto. – IEEE Systems, Man, and Cybernetics Magazine, Vol. **6**, 2020, No 4, pp. 34-44.
22. L i, J., et al. Budget-Aware User Satisfaction Maximization on Service Provisioning in Mobile Edge Computing. – IEEE Transactions on Mobile Computing, 2022.
23. W u, J., et al. Deep Reinforcement Learning for Scheduling in an Edge Computing-Based Industrial Internet of Things. – Wireless Communications and Mobile Computing, 2021, pp. 1-12.
24. J i a n g, C., et al. Energy Aware Edge Computing: A Survey. – Computer Communications, Vol. **151**, 2020. pp. 556-580.
25. Gigaspaces, 2023, Amazon Found Every 100 ms of Latency Cost them 1% in Sales. **https://www.gigaspaces.com/blog/amazon-found-every-100ms-of-latency-cost-them-1-in-sales**
26. D a i, Y., et al. Joint Load Balancing and Offloading in Vehicular Edge Computing and Networks. – IEEE Internet of Things Journal, Vol. **6**, 2018, No 3, pp. 4377-4387.
27. T w i n, A. Competitive Advantage Definition with Types and Examples. 2023. **https://www.investopedia.com/terms/c/competitive_advantage.asp**
28. A v a n, A., et al. A State-of-the-Art Review of Task Scheduling for Edge Computing: A Delay-Sensitive Application Perspective. – Electronics, Vol. **12**, 2023, No 12, 2599.
29. M o c n e j, J., et al. Impact of Edge Computing Paradigm on Energy Consumption in IoT. – IFAC-PapersOnLine, Vol. **51**, 2018, No 6, pp. 162-167.
30. S a r i t h a, S. V. Reliability Analysis of an IoT-Based Air Pollution Monitoring System Using Machine Learning Algorithm-BDBN. – Cybernetics and Information Technologies, Vol. **23**, 2023, No 4, pp. 233-250.
31. B a n g u i, H., et al. Moving to the Edge-Cloud-of-Things: Recent Advances and Future Research Directions. – Electronics, Vol. **7**, 2018, No 11, 309.
32. Q a t a w n e h, M., et al. Challenges of Blockchain Technology in Context Internet of Things: A Survey. – Int. Journal of Computer Applications, Vol. **175**, 2020, No 16.
33. Q a t a w n e h, M., et al. DFIM: A New Digital Forensics Investigation Model for Internet of Things. –Journal of Theoretical and Applied Information Technology, Vol. **97**, 2019, No 24.
34. Z h a n g, J., B. C h e n, Y. Z h a o, X. C h e n g, F. H u. Data Security and Privacy-Preserving in Edge Computing Paradigm: Survey and Open Issues. – In: IEEE Access, Vol. **6**, 2018.
35. S u d h a, K. S., N. J e y a n t h i. A Review on Privacy Requirements and Application Layer Security in Internet of Things (IoT). – Cybernetics and Information Technologies, Vol. **21**, 2021, No 3.
36. I a n s i t i, M., K. R. L a k h a n i. Digital Ubiquity: How Connections, Sensors, and Data are Revolutionizing Business. – Harvard Business Review, November 2014.
37. K u r n i a w a n, I., et al. The Effect of the Information System Quality, Service Quality, and User Satisfaction on Academic Information System User Loyalty. – International Journal of Scientific and Technology Research, Vol. **10**, 2021, No 5, pp. 350-355.
38. L e e, I. The Internet of Things for Enterprises: An Ecosystem, Architecture, and IoT Service Business Model. – Internet of Things, Vol. **7**, 2019, 100078.
39. L e e, I., K. L e e. The Internet of Things (IoT): Applications, Investments, and Challenges for Enterprises. – Business Horizons, Vol. **58**, 2015, Issue 4, pp. 431-440.
40. M a l l i k, A., et al. IoT Utilized Gas-Leakage Monitoring System with Adaptive Controls Applicable to Dual Fuel Powered Naval Vessels/Ships: Development & Implementation. – Cybernetics and Information Technologies, Vol. **20**, 2020, No 1, pp. 138-155.

41. H s u, C.-L., J. C.-C. R. L i n. Understanding the User Satisfaction and Loyalty of Customer Service Chatbots. – Journal of Retailing and Consumer Services, Vol. **71**, 2023, 103211.
42. H a n s e n, E. B., S. B ø g h. Artificial Intelligence and Internet of Things in Small and Medium-Sized Enterprises: A Survey. – Journal of Manufacturing Systems, Vol. **58**, 2021.
43. K h a n n a, A., S. K a u r. Internet of Things (IoT), Applications and Challenges: A Comprehensive Review. – Wireless Personal Communications, Vol. **114**, 2020, pp. 1687-1762.
44. L i, J., et al. Maximizing the Quality of User Experience of Using Services in Edge Computing for Delay-Sensitive IoT Applications. – In: Proc. of 23rd International ACM Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems, 2020.
45. K a l a n k e s h, L. R., et al. Factors Influencing User Satisfaction with Information Systems: A Systematic Review. – Galen Medical Journal, Vol. **9**, 2020, e1686.
46. S a e i k, F., et al. Task Offloading in Edge and Cloud Computing: A Survey on Mathematical, Artificial Intelligence and Control Theory Solutions. – Computer Networks, Vol. **195**, 2021.
47. S u r y a t e j a, P. S., K. R a o, V e n k a t a. A Survey on Lightweight Cryptographic Algorithms in IoT. – Cybernetics and Information Technologies, Vol. **24**, 2024, No 1, pp. 21-34.
48. S u b r a m a n i a n, A., et al. Linear Regression Trust Management System for IoT Systems. – Cybernetics and Information Technologies, Vol. **21**, 2021, No 4, pp. 15-27.
49. Cloud 7 IT Services, Inc., 2023. Scalability in IT Infrastructure: Procurement for Future Growth. **https://www.linkedin.com/pulse/scalability-infrastructure-procurement-future-growth/**
50. P u t h a l, D., et al. Fog Computing Security Challenges and Future Directions (Energy and Security). – In: IEEE Consumer Electronics Magazine, Vol. **8**, May 2019, No 3, pp. 92-96.

Received: 01.08.2024; Second version: 17.09.2024; Accepted: 20.09.2024