*algorithms*

*Article*

# An Aspect-Oriented Approach to Time-Constrained Strategies in Smart City IoT Applications

**Vyas O'Neill and Ben Soh ***

Department of Computer Science and Information Technology, La Trobe University, Melbourne 3083, Australia; v.oneill@latrobe.edu.au
* Correspondence: b.soh@latrobe.edu.au

**Abstract:** The Internet of Things (IoT) is growing rapidly in various domains, including smart city applications. In many cases, IoT data in smart city applications have time constraints in which they are relevant and acceptable to the task at hand—a window of validity (WoV). Existing algorithms, such as ex post facto adjustment, data offloading, fog computing, and blockchain applications, generally focus on managing the time-validity of data. In this paper, we consider that the functional components of the IoT devices' decision-making strategies themselves may also be defined in terms of a WoV. We propose an aspect-oriented mechanism to supervise the execution of the IoT device's strategy, manage the WoV constraints, and resolve invalidated functional components through communication in the multi-agent system. The applicability of our proposed approach is considered with respect to the improved cost, service life, and environmental outcomes for IoT devices in a smart cities context.

**Keywords:** IoT sensors; IoT devices; smart cities; smart environment; multi-agent systems; aspect-oriented software design

check for updates

**Citation:** O'Neill, V.; Soh, B. An Aspect-Oriented Approach to Time-Constrained Strategies in Smart City IoT Applications. *Algorithms* **2023**, *16*, 454. https://doi.org/10.3390/a16100454

Academic Editors: Sotirios Kontogiannis and Myrto Konstantinidou

Received: 1 September 2023
Revised: 21 September 2023
Accepted: 22 September 2023
Published: 23 September 2023

## 1. Introduction

The Internet of Things (IoT) is rapidly gaining traction in various problem domains, including smart cities applications [1–3]. From an engineering perspective, IoT networks can be modelled as heterogeneous multi-agent systems (MASs) [4,5], often with memory, time, power, bandwidth, and processing power constraints [6,7].

The smart city is "derived from the adoption and application of mobile computing systems through practical data management networks amongst all components and layers of the city itself" [8]. A smart city is an "urbanization region that collects data using several digital and physical devices . . . to manage revenues, resources and assets . . . utilized to boost performance throughout the city" [3].

In [9], the authors identify social, environmental, and economic sustainable development as prerequisites for smart cities. IoT-based and related disruptive technologies are proposed as a way of achieving these goals [9]. They consider smart cities in terms of dimensions, digital transformation, sustainability, and resiliency [9]. However, they note that existing research has been focused on "megacities in developed nations [9]" and that there is room for broader studies which consider smart city applications in a wider range of contexts [9].

In a survey of smart city IoT applications [2], the authors enumerate several components of smart cities in which IoT technologies take part: smart transportation, smart agriculture, smart energy, smart infrastructure, smart city services, smart homes, smart health, and smart industry.

Each of these smart city components is supported by a complex infrastructure, from the sensing layer, comprising resource-constrained edge devices, through to the business layer, comprising data analytics supported by applications and cloud services [2]. To achieve these goals, the temporal aspect of data and agent behavior in such systems is critical: smart city applications must respond to changes in the environment and dynamically reconfigure themselves to provide optimal service.

In [10], the authors propose to use enterprise architecture (EA) and service-oriented architecture to support the development of smart cities. The EA approach considers four perspectives: the "business view, IT view, governance view, and security view [10]". Service-oriented architecture (SOA) "offers flexible integration and service reusability [11]" through a service-based modular architecture [11].

Other research has considered the potential for applying machine-learning strategies to improve smart city outcomes [12], the relationship between smart city policies and sustainability [13] and security and privacy risks [14].

Each of these considerations can be applied in a smart city in the form of a strategy, implemented in an executable form and disseminated to the IoT nodes in the smart city.

The distributed and real-time nature of the data in such IoT applications means that, often, these data are transient [15]. Some of the data are superfluous in a given time period, while other data "may remain useful for only a specific period of time after . . . originally queried and cached" [16]. In [16], we proposed that, generally, data in the IoT have a window of validity (WoV) which can be defined according to the relevance of the data at a particular time and whether an acceptable copy of the data is available.

In this paper, we consider the research question of how the problem of time-constrained strategy dissemination can be resolved with minimal resource impact. We propose that, in smart cities applications, besides the WoV of data, the strategies employed in the decision-making algorithms themselves may be time-constrained by a WoV. Our paper proposes a method by which the software engineering problem of how to efficiently manage the dissemination and evolution of such strategies in the IoT/smart cities context can be resolved, ensuring the timeliness and applicability of both the data and the decision-making strategies acting on them.

We propose to extend the aspect-oriented approach we took for managing time-constrained data in smart city IoT systems [16] to the management of time-constrained strategies. Our research has broad applicability across the smart city components [2], providing a method for improving the efficiency and responsiveness of the strategies, cost, service life, and environmental outcomes in the smart cities context. The proposed approach could be applied to a wide range of data and strategies in a smart cities context, including geospatial data, IoT sensor data, health data, transport data, energy usage data, demographic data, hazard and climate data, and human information and reporting data, among others, which, in turn, drive the software implementations of smart city management strategies such as smart health, transportation, and environmental management [9].

In Section 2, we survey existing approaches to time-constrained data and strategies in smart cities. In Section 3, we present our proposed aspect-oriented mechanism which builds on our research in [16], extending and applying it to the management of time-constrained of strategies. In Section 4, we discuss the benefits of applying our proposed model to the management of time-constrained strategies in IoT systems in a smart cities context.

## 2. Existing Approaches

In this section, we review several existing approaches to the general problem of time constraints in smart city IoT systems and discuss their limitations with respect to time-constrained strategies. We then discuss several potential approaches to software updates in IoT systems, highlighting their shortcomings with respect to time-constrained strategies. Then, we review our prior research on aspect-oriented approaches to the WoV of data.

### *2.1. Time-Constrained Data in Smart Cities*

Due to their resource constraints, IoT applications in smart cities contexts often operate such that their data are time-constrained into a window of validity (WoV) [11,16–18]. In many smart city applications, such as smart transportation, the correctness and applicability of the data may be strongly impacted by the ability to communicate it upstream within the WoV [19].

In [16], we proposed to model the WoV of IoT device data according to two parameters: whether the data are relevant to the decision-making algorithm at a particular time, and whether an acceptable version of the data is cached locally. When the decision-making algorithm which runs on the IoT data attempts to access data which is not in its WoV, the supporting software infrastructure is engaged either to fetch the data for the first time or to update an invalidated, locally cached copy [16] (Figure 1).

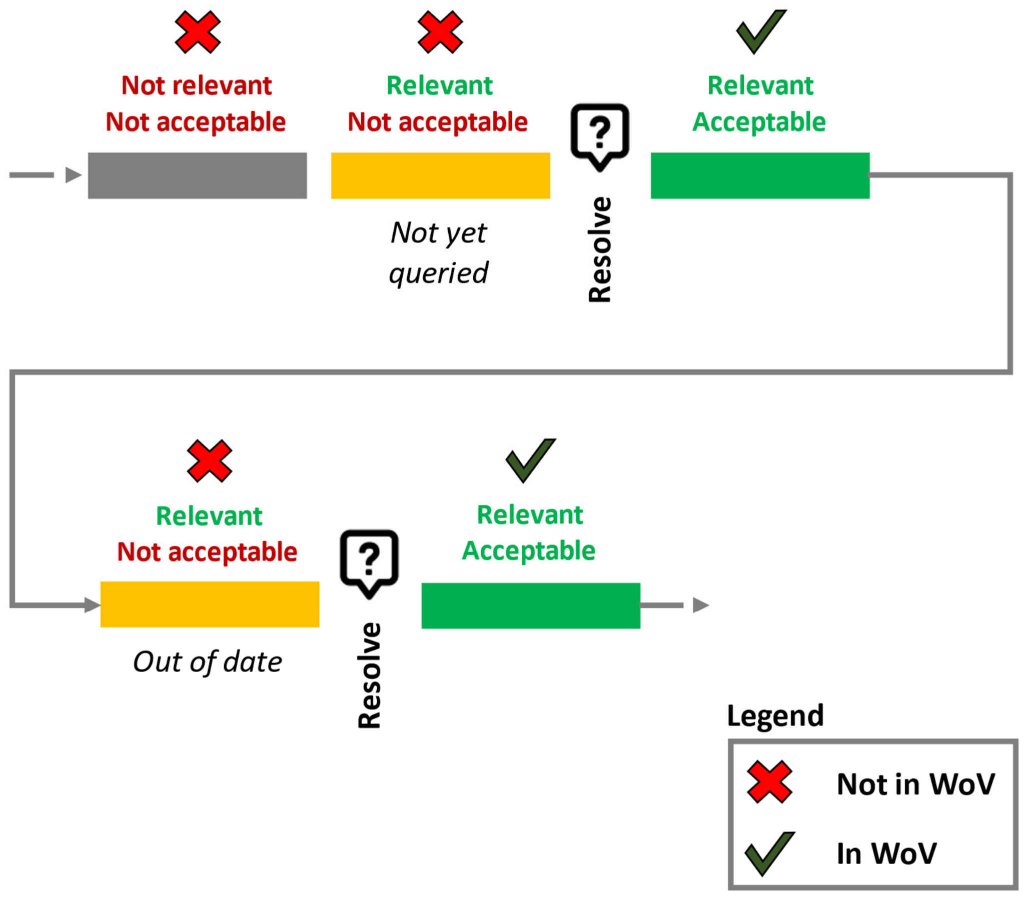

**Figure 1.** Example execution flow in our model for managing time validity of data in IoT systems according to their window of validity (WoV).

There are various practical approaches used to resolve data in IoT systems when they are outside their WoV. In [19], the authors propose an algorithmic framework to adjust, ex post facto, for temporal and spatial discrepancies caused by communications losses in smart transportation data to improve the quality and precision of the data. This approach is suitable where the data is collected into a dataset for later analysis but is less useful where decisions must be made based on the real-time values of IoT sensors.

Others have proposed offloading the data from less capable edge devices to more capable mobile devices such as unmanned aerial vehicles (UAV) [11,20]. In theory, this approach would improve the real-time data gathering and processing in the IoT network; however, in practice, the deployment of swarms of autonomous vehicles may not be feasible or available with sufficient coverage at scale in a smart city. Other potential obstacles to this approach include the high level of heterogeneity typical of IoT devices and their limited capacity to communicate with complex, moving platforms such as UAVs.

### 2.2. Fog Computing

One method which seeks to achieve the same offloading benefits without the associated complexity of mobile physical platforms is fog computing [21–24]. The exact definition of fog computing is yet to be formally set, but, generally, it can be described as "a paradigm that extends the cloud and integrates edge and IoT, while providing a new, horizontally scalable highly virtualized layer that distributes computing, control and networking capabilities" [22]. In [24], the authors describe the relationship between fog and cloud computing as a "computational paradigm that aims to bring the benefits of the cloud closer to end devices. . . highly integrated with the cloud, but with the processing also being carried out at the edge of the network, enabling the execution of applications that [previously] were not possible due to the high latency that existed between the devices and the cloud".

An example of a smart city architecture using fog nodes as an intermediary between the smart city cloud and deployed IoT devices is shown in Figure 2. This architecture creates a middleware tier called the 'fog' between the high-cost, high-capability cloud, and the constrained edge device. Thus, some of the real-time decision-making logic may be carried out in the fog node, close to the edge devices, while complex, data-intensive applications can run in the cloud with an acceptable level of latency.

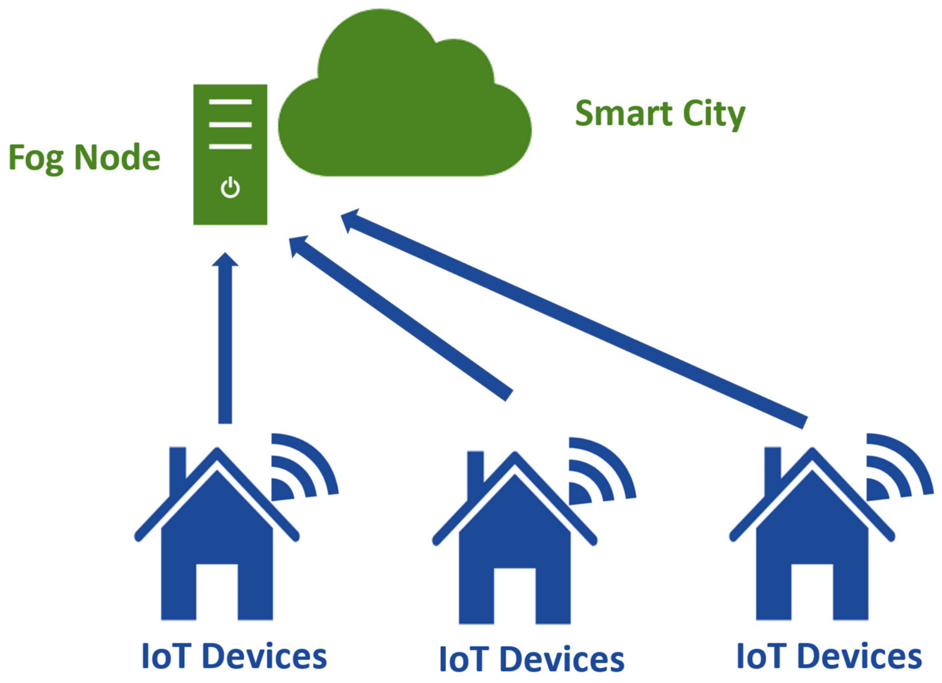

**Figure 2.** Smart city architecture with fog computing.

The essential characteristics of fog computing include the contextual location awareness, low latency, geographical distribution, heterogeneity, interoperability and federation, real-time interactions, scalability, and agility of federated, fog-node clusters [25]. Many smart cities applications of IoT technology aim for a close temporal connection between the data gathered by IoT sensors and their use in the environment [23]. To this end, in [23], the authors present a scalable model to "process data close to sensors and distributed in the fog while data are stored in the cloud" [23]. Thus, fog computing presents a promising approach to managing the WoV problem for data in smart city applications of IoT.

In [21], the authors propose a framework named PIAF (Processing Intelligent Agent running on Fog infrastructure) for maximizing the distribution of time-critical data streams processing on the fog infrastructure using MAS. The PIAF workflow comprises the following steps: the selection of the time-critical data stream, splitting the data stream into sub-tasks, prioritization, allocation, execution, tracking, and storage [21].

However, a limitation of this approach is that there "is no unique architecture that can be deployed to represent all industries' structures and behaviors" [21]. We, thus, find that, while fog computing provides a promising approach to handling time-constrained data, there is still the question of how to manage the domain-specific behavioral aspect—the decision-making strategy which functions on these data—as it is distributed across the IoT devices.

### *2.3. Time-Constrained Strategies*

A time-constrained strategy can be defined as some algorithm, implemented as executable instructions, which operates on data in order to make decisions in an IoT system, where parts of the algorithm have relevance and acceptability constraints according to some WoV.

Just as there are various IoT smart city applications in which the data are time-constrained, we propose that there are, similarly, various applications where the decision-making strategy itself is time-constrained and may not be applicable to the problem domain outside some WoV.

In our considerations of future research directions in [16], we suggested one such example in the smart agriculture context, where certain IoT devices measure various crop parameters, with a decision-making strategy implemented on other devices (such as fog nodes) to control the irrigation of crops. In our example, while the data that drive the decision-making strategy (e.g., meteorological parameters, soil measurements, etc.) are time-constrained, the decision-making strategy itself may also change in accordance with externalities such as seasonal changes and even policy changes (e.g., the water authority may make a substantial change to the water quota, necessitating a new strategy for water allocation).

Supposing that the irrigation strategy is implemented in a fog node physically located in each field, some mechanism must be incorporated in the software engineering design of the fog node to enable it to manage the time validity of the components of its decision-making process. It must recognize when its local strategy is outside its WoV and resolve this either through interaction with the high-level cloud infrastructure or with other fog-level agents in the MAS.

Another potential practical application of time-constrained strategies could be to improve energy efficiency in the smart city infrastructure. For example, a smart city could potentially have thousands of streetlights which activate according to a lighting strategy. Rather than simply having the lights turn on and off at particular times of the day, the lighting strategy could be updated regularly, taking into account the dawn and dusk times, moon phase, individual light locations/conditions, and expected weather conditions. Thus, the lights could be activated optimally, providing better safety for pedestrians and motor vehicles while also reducing unnecessary activation time, thereby improving energy usage and cost at scale.

The software architectures required to support such applications would make use of the orchestration capabilities of the cloud to disseminate up-to-date strategies to the fog nodes [24]. In the context of IoT software architecture, it is desirable that any such mechanism be easily implementable in the resource-constrained environments typical of IoT devices and fog computing [16].

### *2.4. Software Updates in an IoT Context*

There are several existing approaches to the implementation of updating functionality in IoT devices. The Internet Engineering Task Force (IETF) has established a working group on Software Updates for Internet of Things (SUIT) which aims to create a standard for the secure updating of IoT firmware [26]. However, the SUIT architecture is primarily concerned with the secure dissemination process and does not specifically address time constraints arising from the WoV of the decision-making process [26].

Another approach to updating IoT devices makes use of distributed ledger technologies and smart contracts to run executable code over the blockchain without the intervention of a trusted third party [27–29]. Smart contracts have the advantage of being able to incorporate a time constraint for the WoV into the execution of the contract; however, they do not provide a standardized software engineering mechanism for managing the associated WoV in the code of the IoT device itself. Another disadvantage of blockchain-based approaches is the relatively high computational overhead which may present a barrier to implementation in constrained IoT devices [27,30,31].

A recent approach which does consider the time constraint of data is the Age of Information (AoI) approach which seeks to optimize the rate at which information is sent from IoT devices with respect to the devices' power and resource usage [32]. AoI is a "metric that captures how frequently the information status at a destination node . . . needs to be updated through status update transmissions from a source node" [32]. The AoI approach in [32] uses a scheme where different processes in an IoT agent are prioritized according to their importance and AoI values, considered as an optimization problem.

In [33], the authors approach the AoI from the perspective of an energy-constrained edge-caching node, similar in concept to a fog-computing node in that it processes and caches IoT data close to the edge of the network. The authors propose a time-slotted model which splits the activity of such a node into two phases: a data delivery phase, in which data are transmitted from the edge-caching node to the consumers, and a status updating phase, in which the edge-caching node updates its local copy of the IoT sensor data [33].

### *2.5. Aspect-Oriented Approach to the WoV Problem for Data*

Aspect-oriented software design (AOSD) is a programming paradigm that localizes cross-cutting concerns into separate modules, called aspects [34]. An aspect-oriented program, then, contains "a standard object-oriented model, with the addition of aspects—an organizational construct which co-locates code pertaining to a cross-cutting concern, even when the execution takes place in different classes" [16].

A mechanism called an aspect weaver is used to match particular signatures in the object-oriented model, such as calls to a method or database, and inject relevant cross-cutting code from the aspect into the execution flow of the program [35].

In [16], we proposed an aspect-oriented mechanism for software architecture in IoT systems with WoV constraints on the data, with the goal of simplifying the architecture of IoT systems which rely on time-constrained data. In [16], we proposed a bespoke aspect weaver responsible for maintaining a table which maps data members in the object-oriented software architecture to the real-time status of their WoVs [16].

At runtime, the aspect weaver intercepts calls to data members in the IoT device's object-oriented code. When a call is made to a time-constrained data member, the aspect weaver checks if the data member's value is up-to-date according to its WoV [16]. If so, the regular execution flow of the object-oriented program is continued, accessing the data member's value in memory as per a regular object-oriented program [16]. However, where a data member's WoV is invalidated, the aspect weaver uses network communications with other agents in the MAS to acquire an up-to-date value, which is then injected into the object-oriented program's memory [16].

The aspect-oriented approach has the advantage of addressing both components of the WoV: relevance, through the on-demand resolution of invalidated data, and acceptability, which may be implemented through some metric such as AoI. Critically, because the aspect code is out-of-band from the regular execution flow of the program, it can be applied 'aftermarket' to existing IoT device code without extensive changes to the existing codebase.

## 3. Proposed Aspect-Oriented Approach to Time-Constrained Strategies

In this section, we first discuss some of the limitations of the existing approaches. Then, we present our proposed aspect-oriented approach to time-constrained strategies and the resolution of missing or invalidated data through an MAS. Finally, we present our proposed aspect-oriented approach to the problem of time-constrained strategies.

### *3.1. Limitations of the Existing Approaches and Motivation*

The AoI-based approaches are perhaps the most promising candidates presently in the literature for incorporating time constraints in IoT data. However, they are generally focused on the time validity of IoT data flowing in the direction from the IoT device to the consumer, rather than addressing the WoV of executable strategies by which the edge devices themselves make decisions.

Thus, there is a need for an approach to updating functionality which fully integrates time constraints on the functional decomposition of the IoT device's strategy. Additionally, IoT systems run on complex, heterogeneous software stacks.

Time constraints on sections of executable code are not inherently integrated into existing programming languages as native constructs. Integrating such changes in an object-oriented environment would necessitate a fundamental change in approach to how these systems are coded, severely limiting the feasibility of practical implementations unless some mechanism can be provided to retroactively apply the WoV constraints to existing executable code.

To this end, we propose that an aspect-oriented approach is ideal for this software architectural problem.

### *3.2. Resolution of Invalidated Data through the MAS*

To achieve the inter-agent resolution of invalidated data, we propose to use the MAS framework we developed previously in other research: the Intelligence Transfer Model (ITM) [16,36]. The ITM is a software architectural model which aims to improve the fault tolerance and reliability in the MAS through the transfer of 'intelligent activities' which encapsulate the situational awareness, decision-making logic, and agency of the agents in the MAS [36]. Intelligent activities may comprise both data and executable code, as well as unresolved references to other intelligent activities which may exist locally at an agent, or require resolution from the MAS [36].

The ITM achieves improved fault tolerance and reliability by first decomposing the graph of a complex executable function into its constituent steps (method calls, attempts to access data variables, etc.), separating the knowledge of how to execute some task from the physical capability of an agent to actually execute it [36]. The ITM then distributes this knowledge redundantly in the MAS, even to agents who do not have the capacity to physically execute the task [36].

Under this task-based redundancy model, knowledge of the individual components of a task is replicated amongst agents in the MAS, regardless of their ability to actually perform the task, as opposed to an agent-based redundancy model in which entire agents must be replicated [36]. Thus, when new agents join the MAS that do have the required physical capability, they can learn the knowledge of how to execute the task from the existing agents in the MAS [36].

In [36], we developed a candidate implementation of the ITM in which the transfer of intelligent activities was achieved through this real-time sharing of parts of the object model between agents in the MAS and their on-demand injection into the control flow of the program at runtime. We propose that this method of communicating task components between agents can be combined with the WoV mechanism we proposed in [16] to achieve a WoV treatment of the executable components of strategies in IoT devices.

### *3.3. Resolution of Invalidated Strategies through the MAS*

A time-constrained strategy is essentially a set of executable instructions and data which potentially make use of other functionality and data which may or may not be acceptable and available at the local agent.

These executable instructions need not necessarily be a complete, self-contained program. Rather, the strategies may be modular units of code (such as classes) which can be serialized and transferred between agents in the MAS according to the Strategy design pattern [37,38].

At any given time, different components of the algorithm, represented as calls to methods in the Strategy object, may be in differing WoV states (Figure 3).

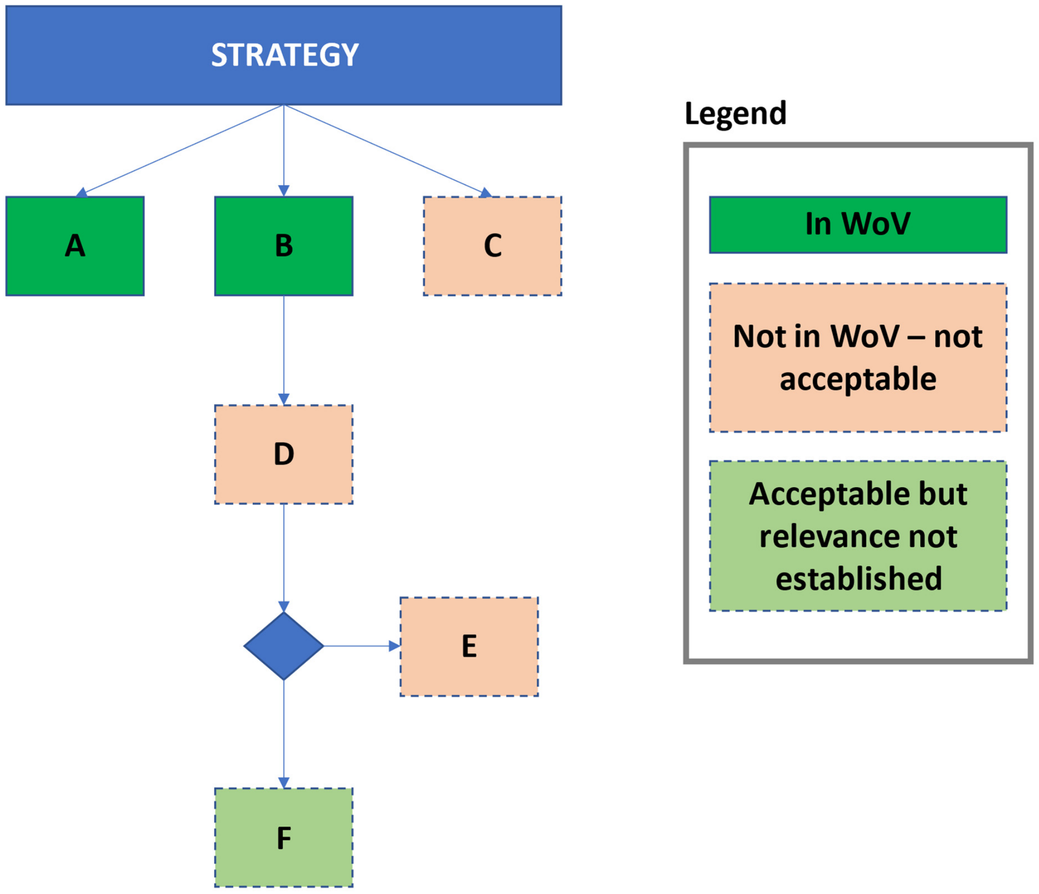
STRATEGY
A
B
C
D
E
F
Legend
In WoV
Not in WoV – not acceptable
Acceptable but relevance not established

**Figure 3.** Example functional decomposition of a strategy showing the WoV status of different components (A–F) in the algorithm at some time *t*.

The example in Figure 3 shows the functional decomposition of a strategy comprising six executable components (A–F) at some time *t*, of which only A and B are within their WoVs (i.e., both relevant and acceptable) at *t*. No acceptable versions of components C, D, and E are available, perhaps due to their age. Thus, some resolution action will be required to update components C, D, and E when the strategy is executed.

At time *t*, an acceptable version of component F exists on the device and could be called without further resolution action being necessary. However, at time *t*, there are no components within their WoV which call F. Thus, the relevance of F to the strategy cannot be determined until the intermediary D is resolved, as, conceivably, an up-to-date version of D may not require F. Should an up-to-date version of D be acquired which references F, component F will immediately enter its WoV without any further resolution action being needed.

### 3.4. Applying the Aspect-Oriented Approach to Time-Constrained Strategies

In developing the software architecture to support time-constrained strategies, we add an additional dimension to the ITM model, integrating it with the aspect-oriented approach we took in [16] for time-constrained data. Thus, in our proposed model, additional out-of-band metadata are maintained by the aspect weaver on the WoV status of each functional component of the strategy.

There are various approaches which may be used to achieve this, the most general being a simple mapping table, with more complex potential solutions including modifying the in-memory representation of the Strategy object to include a WoV field according to the particular constructs of the programming language in use.

Code in the Strategy object may call functions which may or may not yet exist locally, or be within a valid WoV, on the local agent. The aspect-oriented mechanism in the device's infrastructure supervises the execution of the device's main program, intervening to update and acquire, on demand, those executable components of the strategy which are not within a valid WoV. Where calls are made to components of the strategy whose WoVs are invalidated, the aspect weaver may acquire up-to-date versions of the functionality from other agents in the MAS and inject them directly into the object model of the executing program.

Figure 4 shows the architecture of a strategy-based IoT system implementing a WoV resolution component. The Strategy object contains the functional code for implementing the strategy on the IoT device. When a call is made to a functional component, the aspect weaver suspends the normal flow of execution and runs the code in the resolver. The resolver handles the invalidation of the WoV, acquiring the necessary code from other fog nodes or from the cloud. On completion of this process, the control flow is returned to the Strategy object, with the necessary updates injected into the codebase.

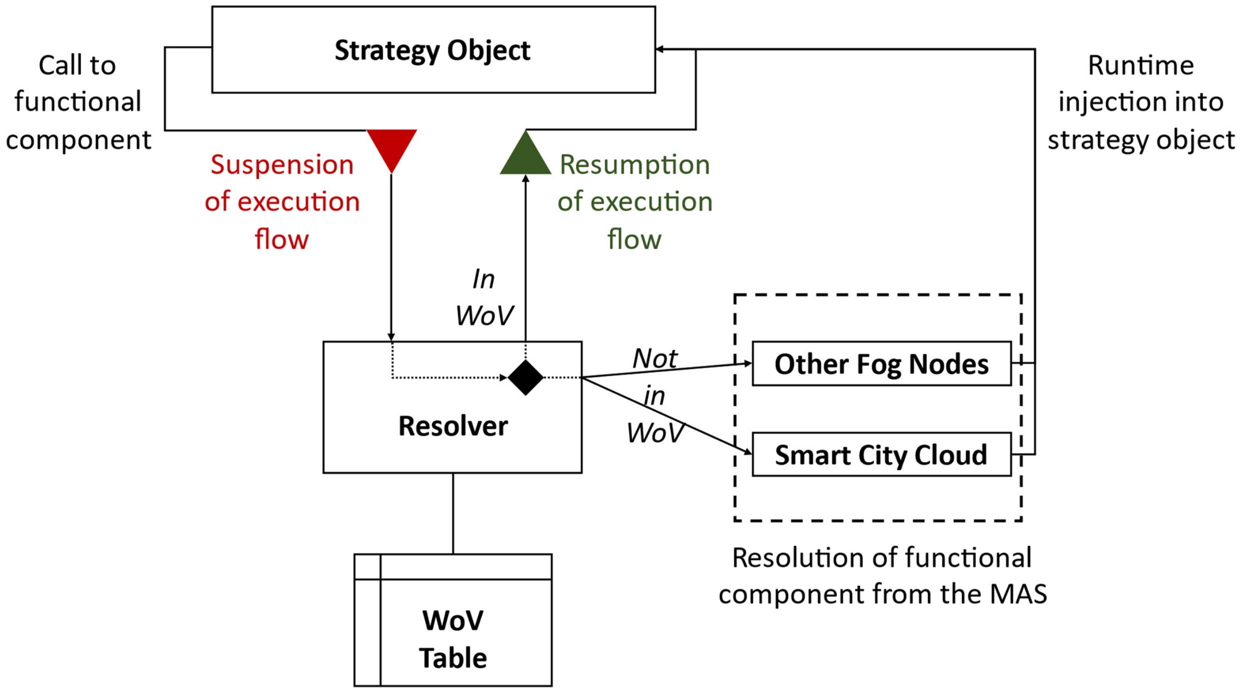
Strategy Object
Call to functional component
Runtime injection into strategy object
Suspension of execution flow
Resumption of execution flow
In WoV
Not in WoV
Other Fog Nodes
Smart City Cloud
Resolver
Resolution of functional component from the MAS
WoV Table

**Figure 4.** Proposed aspect-oriented mechanism for managing the resolution of time-constrained strategies.

We propose that the ITM's loosely coupled architecture for transferring intelligent activities is ideal for time-constrained strategy applications because it enables a complex strategy to be broken down into its constituent functional components, each of which may be shared and updated between agents in the MAS.

The resulting efficiencies achieved by sharing only those functional components of an algorithm that are time-sensitive, when scaled up across an entire smart city, have positive implications for such metrics as energy consumption, communications overhead, and latency. In the next section, we discuss some of these potential benefits in the context of a smart city.

## 4. Benefits of Applying Our Approach in the Context of a Smart City

In this research, we have proposed a method for managing time-constrained strategies at the granular level of functional decomposition. We also applied technological methods we developed in previous research [16,36] to enable the updating of IoT device functionality at runtime at this granular level of functionality. In the context of smart cities, there are several areas where this approach improves upon existing methods.

Firstly, the act of updating the IoT device's strategy and core functionality is in and of itself non-trivial in the context of the resource-constrained and heterogeneous nature of IoT software. Extant research efforts such as SUIT [26] focus on creating standards for updating device firmware. However, this approach is accompanied by complex security, maintenance, and scale considerations, as well as a need to manage the downtime of devices and services.

Our approach to updating the device functionality is far more flexible, as we achieve this goal through the runtime injection of the necessary functional components directly into the object model using an out-of-band, aspect-oriented mechanism as shown in Figure 4. The disadvantage of our approach in comparison to firmware updates is that the dynamic nature of our approach does not provide for persistence. Thus, where IoT devices are reset, they must re-acquire any updates to their functionality at runtime. The specific characteristics of the smart city application will dictate where this is reasonable and where a persistent firmware update is more appropriate.

Another area of improvement is in the decentralization of the update process and the potential for agents to learn strategies from each other. In the context and scale of a smart city, decentralized approaches provide benefits in terms of reliability, fault tolerance, availability, latency, and survivability. While decentralization is not exclusive to our approach, it is key to it insofar as agents may acquire up-to-date versions of strategy components from any other agent in the system. This capacity for the inter-agent sharing of strategies and their components lays a foundation for various artificial intelligence applications in which intelligent agents may communicate, co-operate, and teach each other better strategies for accomplishing tasks (Figure 5).

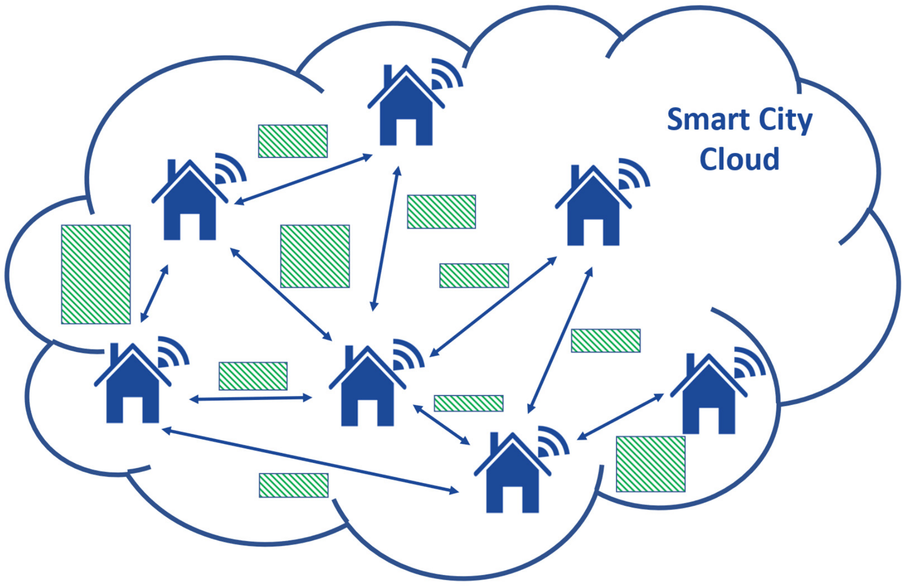

**Figure 5.** A smart city implementing an aspect-oriented mechanism which facilitates the synthesis of local strategies through inter-agent sharing of functional components in an MAS.

The practical implementation of our proposed mechanism in a smart city should also include a plan for its security. Generally, security is an essential component of the design of any form of software update in a deployed/production environment. This consideration is compounded by the physical accessibility of devices across a smart city and the corresponding challenge of physically securing both the devices and the access points to the smart city network which may run over the public internet. Additionally, the large number of IoT devices on the network and their heterogeneity make spoofing and the connection of unauthorized hosts a feasible method of attack.

In the case of our proposed model, executable code is acquired from agents in a decentralized system. Thus, ideally, this code should be wrapped in a trust and verification mechanism prior to usage in the field. Security and trust in IoT systems are ongoing areas of research [39–41] which are outside the scope of this work. However, the ITM, which we propose to use as the basis for MAS communication in our approach, is designed to integrate into the OSI seven-layer model at the application layer. Trust, encryption, and other security technologies can, thus, be easily integrated into the Presentation and Session layers in our model.

Another key area in which our approach improves outcomes in the smart city context is in terms of communications traffic, and, by extension, power consumption. In contrast with a full firmware update, which may require the transmission of a full image over the network, our approach transmits only those components of a strategy for which the WoV constraints determine an update is necessary (Figure 6).

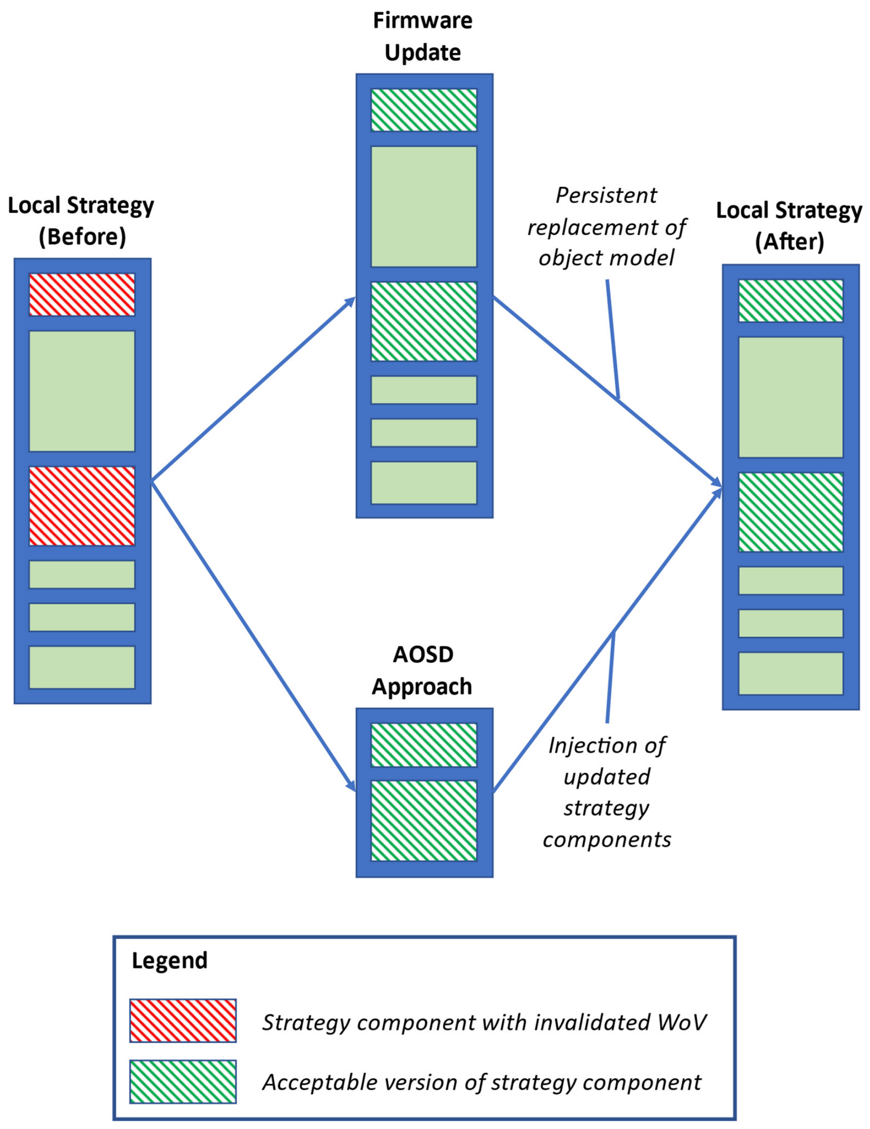
Firmware Update
Local Strategy (Before)
Persistent replacement of object model
Local Strategy (After)
AOSD Approach
Injection of updated strategy components
Legend
Strategy component with invalidated WoV
Acceptable version of strategy component

**Figure 6.** Comparison of the firmware-update and aspect-oriented approach to managing the time validity of strategies, showing the lower overhead in the aspect-oriented approach to reach the same goal state.

A reduction in communications usage between IoT devices improves both their overall energy usage and service life [42,43]. Thus, by reducing the communications overhead associated with strategy updates at the scale of an entire smart city, both cost and environmental savings can be achieved.

## 5. Conclusions and Future Work

### *5.1. Contributions to Theory*

In this paper, we have proposed a novel approach for managing time-constrained strategies in IoT systems in a smart cities context, building on previous research in which we proposed a mechanism for the management of time-constrained data in IoT systems. This research aligns with the broad smart city research goals to improve resource usage, sustainability, and tailored service provided to the residents of smart cities.

Our approach uses an aspect-oriented mechanism to supervise the execution of a strategy, maintain an account of the window of validity (WoV) of different functional components, and resolve components whose WoV is invalidated.

### *5.2. Contributions to Practitioners*

In this paper, we presented two scenarios in which our proposed approach could be applied to improve outcomes in smart agriculture and in optimizing smart city energy usage. However, the proposed approach has general applicability to the design and management of IoT devices in various smart cities contexts with the potential to improve cost, service life, and environmental outcomes through the economical use of communications for strategy updates.

Another key practical advantage of the aspect-oriented approach is its ability to localize the code considerations for the management of the WoV of strategies in an out-of-band manner, independent of the functional implementation of the strategy itself. Considering the time constraint to be an independent dimension of the software design process, the aspect-oriented approach enables a greatly simplified software architecture to be developed. Additionally, it enables the independent development and maintenance of both the strategy itself, and the time-constraint validation process, facilitating the future evolution of each separately.

### *5.3. Limitations and Future Work*

This study has considered the potential applications of an aspect-oriented approach to managing the dissemination of strategies in a smart cities context. Although, in earlier research, we have demonstrated the mechanisms upon which our proposal is based, access to a functioning smart city or simulator was not available during this study, and, as such, we were not able to test our approach on a live smart city system.

Therefore, future work may experiment with a practical implementation, perhaps based in a simulator. Another further area of inquiry will be to investigate how the WoV constraints may be seamlessly integrated into programming languages and compilers. Other potential future avenues of research include the trust and security platform on which our proposed approach may sit, and the issue of persistence on the edge device.

**Author Contributions:** V.O. and B.S. contributed equally to the manuscript. All authors have read and agreed to the published version of the manuscript.

**Funding:** V.O. was supported in this research by an Australian Government Research Training Program Scholarship.

**Data Availability Statement:** No new data were created or analyzed in this study. Data sharing is not applicable to this article.

**Conflicts of Interest:** The authors declare no conflict of interest.

## References

1. Arasteh, H.; Hosseinnezhad, V.; Loia, V.; Tommasetti, A.; Troisi, O.; Shafie-khah, M.; Siano, P. IoT-based Smart Cities: A Survey. In Proceedings of the 2016 IEEE 16th International Conference on Environment and Electrical Engineering (EEEIC), Florence, Italy, 7–10 June 2016. [CrossRef]
2. Syed, A.; Sierra-Sosa, D.; Kumar, A.; Elmaghraby, E. IoT in Smart Cities: A Survey of Technologies, Practices and Challenges. *Smart Cities* **2021**, *4*, 429–475. [CrossRef]
3. Alam, T. Cloud-based IoT Applications and Their Roles in Smart Cities. *Smart Cities* **2021**, *4*, 1196–1219. [CrossRef]
4. Lee, S.; Bae, M.; Kim, H. Future of IoT Networks: A Survey. *Appl. Sci.* **2017**, *7*, 1072. [CrossRef]
5. Gheysari, M.; Tehrani, M. The Role of Multi-Agent Systems in IoT. In *Multi-Agent Systems*; Gupta, S., Banerjee, I., Bhattacharyya, S., Eds.; Springer: Singapore, 2022. [CrossRef]
6. Sehgal, A.; Perelman, V.; Kuryla, S.; Schowalder, J. Management of Resource Constrained Devices in the Internet of Things. *IEEE Commun. Mag.* **2012**, *50*, 144–149. [CrossRef]
7. Sabri, C.; Lobna, K.; Azzouz, S. Comparison of IoT Constrained Devices Operating Systems: A Survey. In Proceedings of the 14th IEEE/ACS Conference on Computer Systems and Applications, Hammamet, Tunisia, 30 October–3 November 2017. [CrossRef]
8. Kirimtat, A.; Krejcar, O.; Kertesz, A.; Tasgetiren, M. Future Trends and Current State of Smart City Concepts: A Survey. *IEEE Access* **2020**, *8*, 86448–86467. [CrossRef]
9. Reis, J.; Marques, P.A.; Marques, P.C. Where Are Smart Cities Heading? A Meta-Review and Guidelines for Future Research. *Appl. Sci.* **2022**, *12*, 8328. [CrossRef]
10. Prasetyo, Y.; Lubis, M. Smart City Architecture Development Methodology (SCADM): Meta-Analysis Using SOA-EA and SoS Approach. *Sage Open* **2020**, *10*, 2158244020919528. [CrossRef]
11. Niknejad, N.; Ismail, W.; Ghani, I.; Nazari, B.; Bahari, M.; Hussin, A. Understanding Service-Oriented Architecture (SOA): A Systematic Literature Review and Directions for Further Investigation. *Inf. Syst.* **2020**, *91*, 101491. [CrossRef]
12. Band, S.; Ardabili, S.; Sookhak, M.; Chronopoulos, A.; Elnaffar, S.; Moslehpour, M.; Csaba, M.; Torok, B.; Pai, H.-T.; Mosavi, A. When Smart Cities Get Smarter via Machine Learning: An In-Depth Literature Review. *IEEE Access* **2022**, *10*, 60985–61015. [CrossRef]
13. Yigitcanlar, T.; Kamruzzaman, M.; Foth, M.; Sabatini, J.; da Costa, E.; Ioppolo, G. Can Cities Become Smart Without Being Sustainable? A Systematic Review of the Literature. *Sustain. Cities Soc.* **2019**, *45*, 348–365. [CrossRef]
14. Ismagilova, E.; Hughes, L.; Rana, N.; Dwivedi, Y. Security, Privacy and Risks Within Smart Cities: Literature Review and Development of a Smart City Interaction Framework. *Inf. Syst. Front.* **2022**, *24*, 393–414. [CrossRef] [PubMed]
15. Vural, S.; Navaratnam, P.; Wang, N.; Wang, C.; Dong, L.; Tafazolli, R. In-Network Caching of Internet-of-Things Data. In Proceedings of the IEEE International Conference on Communications, Sydney, Australia, 10–14 June 2014. [CrossRef]
16. O'Neill, V.; Soh, B. Applying Aspect-Oriented Design Methodology to Manage Time-Validity of Information in Internet-of-Things Systems. In Proceedings of the IEEE World AI IoT Congress, Seattle, WA, USA, 6–9 June 2022; pp. 749–752. [CrossRef]
17. Samir, M.; Sharafeddine, S.; Assi, C.; Ngyuen, T.; Ghrayeb, A. UAV Trajectory Planning for Data Collection from Time-Constrained IoT Devices. *IEEE Trans. Wirel. Commun.* **2019**, *19*, 34–46. [CrossRef]
18. Jiang, D. The Construction of Smart City Information System Based on the Internet of Things and Cloud Computing. *Comput. Commun.* **2020**, *150*, 158–166. [CrossRef]
19. Liu, Y.; Weng, X.; Wan, J.; Yue, X.; Song, H.; Vasilakos, A. Exploring Data Validity in Transportation Systems for Smart Cities. *IEEE Commun. Mag.* **2017**, *55*, 26–33. [CrossRef]
20. Mazaffari, M.; Saad, W.; Bennis, M.; Nam, Y.-H.; Debbah, M. A Tutorial on UAVs for Wireless Networks: Applications, Challenges, and Open Problems. *IEEE Commun. Surv. Tutor.* **2019**, *21*, 2334–2360. [CrossRef]
21. Gharbi, I.; Barkaoui, K.; Samir, B. An Intelligent Agent-Based Industrial IoT Framework for Time-Critical Data Stream Processing. In *Mobile, Secure and Programmable Networking*; Bouzefrane, S., Laurent, M., Boumerdassi, S., Renault, E., Eds.; Springer Nature: Cham, Switzerland, 2021. [CrossRef]
22. Tange, K.; De Donno, M.; Fafoutis, X.; Dragoni, N. A Systematic Survey of Industrial Internet of Things Security: Requirements and Fog Computing Opportunities. *IEEE Commun. Surv. Tutor.* **2020**, *22*, 2489–2520. [CrossRef]
23. Meslin, A.; Rodriguez, N.; Endler, M. Scalable Mobile Sensing for Smart Cities: The MUSANet Experience. *IEEE Internet Things J.* **2020**, *7*, 5205–5209. [CrossRef]
24. Costa, B.; Bachiega, J.; Reboucas de Carvalho, L.; Araujo, A. Orchestration in Fog Computing: A Comprehensive Survey. *ACM Comput. Surv.* **2022**, *55*, 1–34. [CrossRef]
25. Iorga, M.; Feldman, L.; Barton, R.; Martin, M.; Goren, N.; Mahmoudi, C. *Fog Computing Conceptual Model*; NIST Special Publication 500-325; NIST: Gaithersburg, MD, USA, 2018. [CrossRef]
26. Moran, B.; Tschofenig, H.; Brown, D.; Meriac, M. *A Firmware Update Architecture for Internet of Things*; RFC 9019; IETF: Wilmington, DE, USA, 2021. [CrossRef]
27. Hernandez-Ramos, J.; Baldini, G.; Matheu, S.; Skarmeta, A. Updating IoT Devices: Challenges and Potential Approaches. In Proceedings of the Global Internet of Things Summit, Dublin, Ireland, 3–5 June 2020. [CrossRef]
28. Khan, S.; Loukil, F.; Ghedira-Guegan, C.; Benkhelifa, E.; Bani-Hani, A. Blockchain Smart Contracts: Applications, Challenges, and Future Trends. *Peer-Peer Netw. Appl.* **2021**, *14*, 2901–2925. [CrossRef]

29. Zheng, Z.; Xie, S.; Dai, H.-N.; Chen, W.; Chen, X.; Weng, J.; Imran, M. An Overview on Smart Contracts: Challenges, Advances and Platforms. *Future Gener. Comput. Syst.* **2020**, *105*, 475–491. [CrossRef]
30. Pennino, D.; Pizzonia, M.; Vitaletti, A.; Zecchini, M. Blockchain as IoT Economy Enabler: A Review of Architectural Aspects. *J. Sens. Actuator Netw.* **2022**, *11*, 20. [CrossRef]
31. Imran, M.; Yao, B.; Ali, W.; Akhunzada, A.; Azhar, M.; Junaid, M.; Iqbal, U. Research Perspectives and Challenges of Blockhain for Data-Intensive and Resource-Constrained Devices. *IEEE Access* **2022**, *10*, 38104–38122. [CrossRef]
32. Abd-Elmagid, M.; Pappas, N.; Dhillon, H. On the Role of Age of Information in the Internet of Things. *IEEE Commun. Mag.* **2019**, *57*, 72–77. [CrossRef]
33. Xu, C.; Wang, X.; Yang, H.; Sun, H.; Quek, T. AoI and Energy Consumption Oriented Dynamic Status Updating in Caching Enabled IoT Networks. In Proceedings of the IEEE Conference on Computer Communications Workshops, Toronto, ON, Canada, 6–9 July 2020. [CrossRef]
34. Wedyan, F.; Freihat, R.; Hammad, M. Visualization of Aspect-Oriented Programs Using City Transportation Metaphor. *Clust. Comput.* **2022**, *25*, 3993–4008. [CrossRef]
35. Cerny, T. Aspect-Oriented Challenges in System Integration with Microservices, SOA and IoT. *Enterp. Inf. Syst.* **2018**, *13*, 467–489. [CrossRef]
36. O'Neill, V.; Soh, B. Improving Fault Tolerance and Reliability of Heterogeneous Multi-Agent IoT Systems Using Intelligence Transfer. *Electronics* **2022**, *11*, 2724. [CrossRef]
37. Gamma, E.; Helm, R.; Johnson, R.; Vlissides, J. *Design Patterns: Elements of Reusable Object-Oriented Software*, 1st ed.; Addison-Wesley: Boston, MA, USA, 1994.
38. El Maghawry, N.; Dawood, A. Aspect Oriented GoF Design Patterns. In Proceedings of the 7th International Conference on Informatics and Systems, Cairo, Egypt, 28–30 March 2010.
39. Aldowah, H.; Ul Rehman, S.; Umar, I. Trust in IoT Systems: A Vision on the Current Issues, Challenges and Recommended Solutions. In *Advances on Smart and Soft Computing*; Saeed, F., Al-Hadhrami, T., Mohammed, F., Mohammed, E., Eds.; Springer: Singapore, 2020. [CrossRef]
40. Hallappanavar, V.; Birje, M. A Reliable Trust Computing Mechanism in Fog Computing. *Int. J. Cloud Appl. Comput.* **2021**, *11*, 1–20. [CrossRef]
41. Fortino, G.; Fotia, K.; Messina, F.; Rosaci, D.; Sarne, G. Trust and Reputation in the Internet of Things: State-of-the-Art and Research Challenges. *IEEE Access* **2020**, *8*, 60117–60125. [CrossRef]
42. Ikpehai, A.; Adebisi, B.; Anoh, K. Effects of Traffic Characteristics on Energy Consumption of IoT End Devices in Smart City. In Proceedings of the Global Information Infrastructure and Networking Symposium, Thessaloniki, Greece, 23–25 October 2018. [CrossRef]
43. Tahiliani, V.; Digalwar, M. Green IoT Systems: An Energy Efficient Perspective. In Proceedings of the 11th International Conference on Contemporary Computing, Noida, India, 2–4 August 2018. [CrossRef]