Volume 01 Issue 01, June 2024

Programmable Urban Environments: Shaping the Future of Cities

Dr. Bhaskar Seth[1] , Ms. Nargis Banu[2], Ms. Deepika Paliwal[3]
Associate Professor, Department of Computer Application, Geetanjali Institute of T echnical Studies, Udaipur, Rajasthan[1]
Assistant Professor, Department of Computer Application, Geetanjali Institute of Technical Studies, Udaipur, Rajasthan[2,3]

sethbh.udr@gmail.com[1], nargisjaroli@gmail.com[2], n3deepika@gmail.com[3]

Abstract - The pervasiveness of mobile devices that are linked to the internet has brought about a fundamental transformation in our day-to-day lives and will have a significant impact on the architecture of urban spaces in the future. Nevertheless, putting city-wide systems into reality has a distinct set of challenges in terms of scale, administration, and community engagement. To make the notion of a programmable Internet of Things (IoT) a reality, it is vital to find a solution that is both effective and efficient to these issues. Within the scope of this article, we will conduct an analysis of these concerns and provide a novel programming style that is intended to construct scalable Internet of Things systems, with a particular emphasis on applications for smart cities. Our goal is to pave the way for the seamless integration of Internet of Things (IoT) technology into urban environments, which will ultimately lead to increased creativity, productivity, and sustainable growth.

Keywords — Ubiquitous Computing, Programming Languages, Macro programming, Internet of Things, Mobile devices, smart cities, urban environment.

I. INTRODUCTION

Our day-to-day lives have been subjected to large and deep transformations as a result of the advent of ubiquitous computing. As a result of these changes, our social interactions have been altered, our connection with memory has been altered as a result of the fact that we constantly have access to digital storage, marketing and sales strategies have been altered as a result of new channels, and even political dynamics have been altered concerning how we see and choose leaders. These transformations are being driven by the convergence of several factors, including the growth of service offerings aimed at enhancing individual experiences, mobile connectivity, and embedded technological systems. Considering the manner in which this shift is progressing, there are significant issues about the planning of our interior spaces. It is necessary for us to not only enhance the existing services in order to meet the ever-evolving requirements, but also to include novel concepts of individuality, connection, and business into the fabric of our manufactured environments. In the long run, the objective is to have sensors that are widely utilised to monitor the circumstances of urban areas and to direct the development of services, regulations, and infrastructure. The long-term impacts of instrumenting cities, on the other hand, need careful consideration. Despite the fact that the construction of physical and communication infrastructures is a demanding and costly process, the maintenance of these systems provides a far greater challenge. There is also the possibility that the existing network infrastructures may become overloaded as the number of devices that are sharing data continues to rapidly increase. With this potential obstacle in mind, the scientific community has been anticipating such challenges for a considerable amount of time and has imposed stringent constraints in order to confront them head-on. In order to successfully navigate the complexities of urban instrumentation, we need to be vigilant in order to guarantee that the technological advancements we create contribute to the development of urban areas that are inclusive, resilient, and sustainable.

In the realm of Wireless Sensor Networks (WSN), the concept of macro programming, which refers to the programming of whole networks, has been investigated for a considerable amount of time. This approach, which is shown by programmes like as Regiment and tinyDB, makes it possible to do data processing and querying over sensor networks in a distributed manner. However, despite the fact that macro programming in WSN focuses primarily on data collection and calculation, it often operates with very restricted devices, which limits the complexity of calculations that can be performed. Our proposed solution, on the other hand, goes beyond the customary limits that are often imposed by introducing a mechanism for general-purpose language extensions. This mechanism is designed

to determine the optimal network position for code execution based on the requirements and demands of the outside world. Because of this, applications may include a greater variety of computer tasks, such as sensing, processing, and actuation duties. In particular, our architecture provides unrivalled flexibility and adaptability in network development by allowing the execution of any application that is compatible with the operating system that is currently in use.

II. SCALING IOT APPLICATIONS

The two most common methods that are used to alleviate network saturation are the reduction of the amount of data that is transferred and the implementation of network off-loading solutions. Data from sensors is often sent to a central sink or cloud storage for processing and storage in the majority of today's instrumentation solutions. On the other hand, not all of the information that is obtained from the municipal instrumentation that is extensively utilised is equally useful; some of it is just temporarily significant. As an example, fine-grained pressure sensing is required in order to locate leaks in water pipes; however, due to the fact that the majority of samples are redundant, auditing does not need the saving of every sample. It is possible to utilise the network to compute valve actuation, which has the potential to reduce the quantity of data transmitted to the cloud as well as the length of time it takes for valve actuation to occur, hence reducing the strain placed on the communication medium. This method is necessary in order to expand municipal automation and instrumentation without interfere with the services that are already being provided. In a manner that is analogous, network off-loading is a strategy that similarly makes use of mobile devices to collect and transmit data, and it also reduces the distances that devices are connected to one another. Bluetooth or Wi-Fi Direct, in addition to the physical movement of devices (data mules), may be used to allow multi-hop data transmission between mobile devices, hence lowering congestion on traditional networks. This can be accomplished via the mobility of devices.

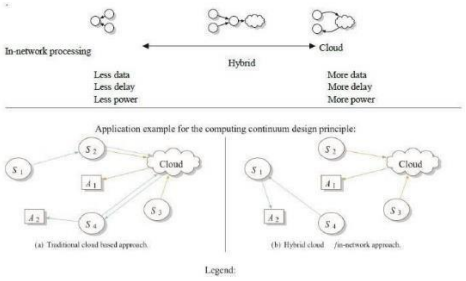

Fig. 1. As a Design Principle for the Internet of Things, the Computing Continuum

Because of the integration of a large number of sensors, actuators, and cloud services, the network is able to support the operation of three distinct applications. Initially, it is used for the purpose of leak detection; it adjusts the state of valve actuator A2 by using the data obtained from sensors S1 and S4. In the second application, pictures are processed, and the control data from sensor S2 is used to operate actuator A1. During this interim period, the third application requires just the output from sensor S3 in order to show the data from the temperature sensor on another map. When all of the data is sent to the cloud for processing and analysis before being applied to actuators, a number of drawbacks become apparent with this approach. To begin, communication congestion is caused by the transmission of unnecessary data by nodes that are not participating in particular applications (for example, sensor S2 delivering data for two different applications). Furthermore, if the data is sent over the network and then has to be analysed in the cloud, it may take a considerable amount of time for the actuators to be activated after the sensor data has been received. As a result, the appropriate placement of compute units is determined by a variety of factors, including the requirements for timeliness, redundancy, and accuracy. Consequently, it is of the utmost importance to consider the network as a programmable device from the edge to the cloud, and it is also vital to offer tools that assist this perspective when it comes to the process of developing and deploying applications.

The new technologies that are being developed need to be able to meet the significant requirements that these sorts of applications present. When it comes to the utilisation of shared infrastructure for smart city applications, for instance, concerns of security and privacy need to be addressed. When it comes to preserving isolation between the several programmes that evaluate data on the network, this is also an essential consideration. The equitable and efficient use of shared network resources is necessary for the timely transmission of data. The establishment of a registry is necessary in order to specifically identify applications and locate resources. Furthermore, robust guarantees on the behaviour of applications are necessary for city applications that have the potential to have real-world impacts, such as the destruction of property or even the endangerment of life, such as those seen in healthcare automation. In order for these technologies to effectively serve the growing field of smart city projects, they must, in a nutshell, offer the highest importance to the protection of users' privacy, the utilisation of resources, and the reliability of applications.

III. Application Development

For the purpose of illustrating the paradigm that was

presented, we have developed a skeleton structure and put the pipe leak monitoring scenario into reality. In this section, we examine the water pressure in a pipe and control a valve from a distant location. There is a possibility that we will use containers and supply Linux networking tasks since we assume that the nodes are capable of running Linux. This assumption is reasonable given the availability of devices that are capable of running Linux on embedded systems. Some examples of such devices are those manufactured by Texas Instruments and National Instruments. In addition, hobbyist platforms like as Raspberry Pi and Intel Galileo provide fully functioning computer nodes that are not only affordable but also fully functional for use in do-it-yourself Internet of Things applications. In addition to an analytics dashboard that displays information from the water network, including historical sensor readings and statistics, the system will include three concurrent applications: a third application, which will be named later, will be a leak detection application that monitors pressure readings to identify leaks and activates the valve appropriately.

An application that will be referred to as "control" will be added, which will enable the water company to manually operate the valve via the use of a web page. The flow of data that occurs throughout these programmes. In an effort to speed up the development process, we have created Scale, a domain-specific language, utilising the Haskell programming language. This programming language abstracts the programming of individual devices by including concepts of abstract sensors and actuators, which are resolved upon the deployment of the application. Through the use of Scale, it is possible to write code for a variety of applications without having to explicitly consider the complexity of device programming. The pipe control software, for instance, allows us to input specifications in order to reference network components, search for concrete sources, do calculations in order to locate potential leaks, and change the valve in the proper manner. A copy of the code for each of the three examples is included. Among the many crucial instructions that the compiler reads in order to distribute computing, scale is one of them.Queries based on the Source and Scale.

It is important to take into consideration the Scale.HandleSource and Scale.ExecuteCommand commands that are included in our framework in order to go further into the distribution process. As soon as the compiler reaches Scale.HandleSource in the application code, it immediately inserts instructions that make use of the MQTT publish-subscribe mechanism. These instructions direct the sensor node to send in sensor readings on a regular basis, and they instruct the processing node to analyse these data automatically once they are received. When all of the devices are connected to the internet, MQTT security standards, which include SSL encryption, guarantee the safety of the network. The application code is taken as input by our in-house created Scale compiler, which then generates containers for deployment on cloud servers and edge devices (including actuator processing units and sensor nodes). During the deployment process, each device receives a physical IP address. This process also involves remotely pushing the container images and launching the containers (in our instance, SSH was used for this action). This streamlined technique ensures efficient data transmission and processing by facilitating the installation and operation of Internet of Things applications throughout the network in a seamless and efficient manner.

Using this example, we demonstrate the paradigm that we suggest for the construction of Internet of Things applications. With our approach, we emphasise shifting the focus away from the cloud and the programming of individual devices and towards seeing the network as a programmable ecosystem that extends from the edge to the cloud. Determining which abstractions are the most effective in displaying this unified image of the network requires careful consideration in order to prevent the concealment of vital information and to facilitate the debugging and understanding of applications. The implementation of this method has been carried out on sensor nodes that are intended to be included into the London Living Labs project that is headquartered in Hyde Park. A range of environmental parameters, including as light pollution, water, air, and soil quality, will be monitored by these nodes with the purpose of ensuring their safety. The rollout will make use of Intel Galileo platforms for gateway nodes that are powered by the mains, while Intel Edison boards will be used for satellite nodes that are powered by batteries. Not only do they run Linux, but they also support our framework. Sandboxed experimentation with the network will be ensured by the platform, which will enable for normal sensor network operation to be retained. This will ensure that data transmission is completed promptly. By using edge virtualization in conjunction with lightweight containers and duty cycling techniques, we are able to make use of fully-fledged x86 processors while still maintaining a battery life that is satisfactory for the sensor nodes.

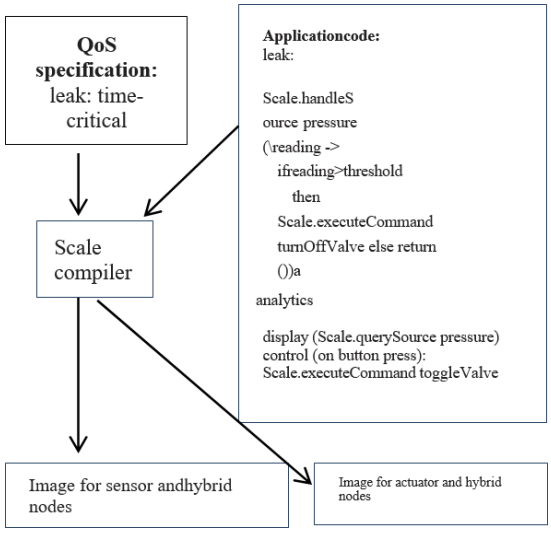

Fig.2. Procedures for the Scalar Compiler to Follow.

Through the process of defining application requirements in terms of time, redundancy, and other characteristics, we have studied the possibility of automating the selection of code deployment locations. In addition, it is possible that we may be required to provide many code implementations for various network locations, perhaps in order to maximise the amount of power that is used. When application requirements are subject to change, however, it may be challenging to ensure consistent behaviour. This is especially due to the fact that different devices may interpret data differently due to the fact that they have distinct internal representations. It is possible that actuators would suffer negative consequences as a result of an incorrect sense of values brought about by such inequalities. In order to address this issue, we are investigating formal methods that will allow us to exert a greater degree of control over this unpredictability. Specifically, we are interested in the utilisation of type systems as a means of ensuring that different programming alternatives, such as one that operates on an edge node and another that operates on a cloud virtual machine, are maintained distinct from one another and that their combination is well-formed. As a means of demonstrating the efficacy of our technique, we have used this method to create two distinct versions of a sensor programme that makes use of sensors that have differing degrees of accuracy. By using our technique, the compiler is able to effectively handle conversions and ensure that the data generated by one sensor is never mistakenly interpreted as data from another sensor. Our solution offers static guarantees against representation mismatch concerns, in contrast to the approaches that are currently in use, which often rely on common supertypes (and perhaps harmful downcasts) or tagged unions (which boost runtime and programmer overhead costs). This indicates that essential systems may be protected against the danger of data interpretation mistakes in a secure manner, hence providing further assurances of their reliability and security.

IV. CONCLUSIONS

Since Batty made the observation in 1995 that cities are "turning into constellations of computers," a significant amount of research has been conducted on the ways in which information and communication technology influences the forms and functions of metropolitan zones. However, the majority of these studies have focused on the effects of software-enabled devices rather than the crucial role that software itself plays in controlling these impacts. This is because software played a significant role in regulating these consequences. I argue that in order to fully comprehend the world and its practices, including ideas, rules, measurements, locations, equations, images, and more, as well as the various ways in which software operates within it, we need to investigate the nature, composition, political economy, and spatial dimensions of software, as well as its underlying principles. This is necessary in order to fully comprehend the world. In addition to educating people, software is responsible for shaping the interactions and outcomes that occur in the universe of many distinct technologies. Therefore, in order to have a complete understanding of urban dynamics, it is required to conduct an in-depth examination of the complicated role that software plays in the formation of contemporary urban environments. Therefore, in order to have a comprehensive understanding of this paradigm, which I have essentially referred to as "programmable urbanism," we need to carry out an in-depth analysis of the numerous ways in which the city is transformed into computer code and the ways in which this software is influencing the way people live in urban areas. We will, in essence, need to go through the complexities of the "black box" in order to get an understanding of the functions that are external to it. With the assistance of this investigation, we will be able to fill in significant information gaps and give fresh perspectives on cities that are going through significant transformations in terms of their size, governance, and organisation. By using this approach, we are able to get a more in-depth comprehension of the dynamic processes that are resulting in the formation of urban landscapes in the era of digital technology.

REFERENCES

[1] Gould, C.R., et al., New battery model and state-of-health determination through subspace parameter estimation and state-observer techniques. IEEE Transactions on Vehicular Technology, 2009. 58(8): p. 3905-3916.

[2] Maccartney, G.R., et al. A flexible wideband millimeter-wave channel sounder with local area and NLOS to LOS transition measurements. in IEEE International Conference on Communications. 2017.

[3] Maroti, M., et al., Shooter localization in urban terrain. Computer, 2004. 37(8): p. 60-61.

[4] Qian, Y., R.F. Follett, and J.M. Kimble. Soil organic carbon input from urban turfgrasses. in Soil Science Society of America Journal. 2010.

[5] Stone, E.A., et al., Insights into the nature of secondary organic aerosol in Mexico City during the MILAGRO experiment 2006. Atmospheric Environment, 2010. 44(3): p. 312-319.

[6] Raychaudhuri, D., et al. Challenge: COSMOS: A city-scale programmable testbed for experimentation with advanced wireless. in Proceedings of the Annual International Conference on Mobile Computing and Networking, MOBICOM. 2020.

[7] Tiete, J., et al., SoundCompass: A distributed MEMS microphone array-based sensor for sound source localization. Sensors (Switzerland), 2014. 14(2): p. 1918-1949.

[8] Williamson, B., Computing brains: learning algorithms and neurocomputation in the smart city. Information Communication and Society, 2017. 20(1): p. 81-99.

[9] Druml, N., et al. PRYSTINE - PRogrammable sYSTems for INtelligence in AutomobilEs. in Proceedings - 21st Euromicro Conference on Digital System Design, DSD 2018. 2018.

[10] Camillen, F., et al. Multi agent simulation of pedestrian behavior in closed spatial environments. in TIC-STH'09: 2009 IEEE Toronto International Conference - Science and Technology for Humanity. 2009.

[11] Danner, J., et al. Rapid precedent-aware pedestrian and car Classification on constrained LoT platforms. in Proceedings of the 14th CM/IEEE Symposium on Embedded Systems for Real-Time Multimedia, ESTIMedia 2016. 2016.

[12] Lehsing, C., A. Kracke, and K. Bengler. Urban Perception-A Cross-Correlation Approach to Quantify the Social Interaction in a Multiple Simulator Setting. in IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC. 2015.

[13] Xie, Y., et al., Dual-Source Detection and Identification System Based on UAV Radio Frequency Signal. IEEE Transactions on Instrumentation and Measurement, 2021. 70.

[14] Yang, H., et al., A very fast decision tree algorithm for real-time data mining of imperfect data streams in a distributed wireless sensor network. International Journal of Distributed Sensor Networks, 2012. 2012.

[15] Balz, T. and N. Haala. Improved real-time SAR simulation in Urban areas. in International Geoscience and Remote Sensing Symposium (IGARSS). 2006.

[16] Abdrabou, M. and T.A. Gulliver, Adaptive Physical Layer Authentication Using Machine Learning With Antenna Diversity. IEEE Transactions on Communications, 2022. 70(10): p. 6604-6614.

[17] Flavio Bonomi, Rodolfo Milito, Preethi Natarajan, and Jiang Zhu. Fog computing: A platform for internet of things and analytics. In Nik Bessis and Ciprian Dobre, editors, Big Data and Internet of Things: A Roadmap for Smart Environments, volume 546 of Studies in Computational Intelligence, pages 169–186. Springer International Publishing,2014.

[18] P. Gupta and P.R. Kumar. The capacity of wireless networks. Information Theory, IEEE Transactions on, 46(2):388–404, Mar2000.

[19] Roman Kolcun and Julie A McCann. Dragon: Data discovery and collection architecture for distributed IoT. In Internet of Things 2014 The 4th International Conference on the Internet of Things (IoT 2014), Cambridge, USA, Oct2

[20] Roman Kolcun and Julie A McCann. Dragon: Data discovery and collection architecture for distributed IoT. In Internet of Things 2014 The 4th International Conference on the Internet of Things (IoT 2014), Cambridge, USA, Oct2011