International Journal of Computer Science and Engineering Innovations
ISSN: 3107-9458 | Volume 2 Issue 4, 08-12, October-December 2026
DOI: https://doi.org/10.64137/31079458/IJCSEI-V2I4P102

Scholastic Research Publication

Received: 12//09/2026 Revised: 24/09/2026 Accepted: 30/09/2026 Published: 03/10/2026

Original Article

Architectures and Design Patterns for Enabling Advance Interactions of IoT Devices: Survey

[1]NURAS NASER SAEED HIZAM, [2]MADHUKAR SHELAR, [3]ARCHANA BACHHAV
[1,2]Department of Computer Science, C.M.CS College, Nashik City, University of Pune, Maharashtra State, India.
[3]Department of Computer Science, KSKW Arts, Science and Commerce College, Nashik City, Pune University, Maharashtra State, India.

ABSTRACT: *The potential advantages of the Internet of Things are practically unlimited, and IoT applications are transforming how we work and live by saving time and money and creating new chances for expansion, innovation, and knowledge development. Private and public organizations may evaluate and enhance performance as well as create new business models due to the Internet of Things. The main goal of this survey paper is to perform a survey based on previously published papers in the research of the key techniques for enabling advanced interactions across various smart device frameworks and architectures.*

KEYWORDS: *Internet of Things, Wireless Sensor Networks, Smart Space, Cloud Computing, Configuration Management.*

1. INTRODUCTION

The term "Internet of Things" (IoT) refers to an extension of the Internet in which an enormous amount of "things," such as sensors, actuators, and processors, are networked and capable of providing high-resolution data on their environment and demonstrating some sort of control over it, in addition to human users. It is still in its infancy, and numerous difficulties and research concerns must be addressed before it can be broadly embraced. As hundreds of millions of various devices are connected, many of these issues are technical, such as interoperability and scalability. However, businesses face a challenge when deciding how to participate in the IoT due to the significant social, legal, and ethical issues involved, such as data security and privacy. There are many studies that have been conducted in the field of IoT, particularly on the Graphical Interface for IoT to deal with the dynamic environment of IoT networks. To address IoT networks' dynamic properties, virtual resources are required, such as Heterogeneous Platforms, large sets of devices, Real-time operations, and limited computational capabilities.

2. OVERVIEW OF IOT

Some Internet of Things (IoT) functions may now be brought closer to physical devices due to the implementation of fog computing. A solution to handle the virtualization issues that appeared as a result of placing computation closer to the IoT "Things" layer is required. There are many studies that have been conducted in the field of IoT, precisely in the Graphical Interface for IoT, as well as platforms and software to integrate different smart devices.

Users are unable to accurately and efficiently access device services due to heterogeneous device services produced by diverse devices in varied settings. This substantially impacts the growth of the Internet of Things. In relation to the GUI interface built for users to connect smart devices to the Internet of Things network, Web technologies have shown to be scalable, and access to services via Cloud technologies is gaining fast popularity in all domestic, commercial, and industrial applications. Therefore, systems and applications for gathering and analyzing useful information utilizing web technologies like HTTP, RESTful Web services, and CoAP have been built and modelled by academics and researchers, improving the accessibility of real-world sensory data as web resources.

Researchers have found several strategies, techniques and approaches to address the aforementioned challenges. This section reviews important literature regarding the following.

- Designed patterns of Applications Programmed for software, interfaces and deployment methods used in IoT.
- Specific Architectures of the Internet of Things for Connecting Smart Devices.

3. DESIGNED PATTERNS OF APPLICATIONS PROGRAMMED FOR SOFTWARE INTERFACES AND DEPLOYMENT METHODS USED IN IOT

Design patterns offer well-known approaches to address design issues that are frequently encountered in a certain field or problem domain. For instance, stop signs, traffic lights, and roundabouts are three basic design patterns that may be used to manage traffic flow at a road intersection. Specific to certain traffic patterns and other contextual considerations, each has benefits and drawbacks.

This work is an open access and licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License
(https://creativecommons.org/licenses/by-nc-nd/4.0/)

IoT poses design challenges in a variety of contexts and at various system levels. There are many Patterns for application programming that describe ways that software and interfaces are used in IoT applications, such as:

- REST Objects: Programming language libraries are used to map REST API resources to program objects.
- Event handler: An event, which is code for an application that reacts to asynchronous events, is called Event
- Event-driven flow: a collection of application handlers functioning in an event-driven graph together with parallel and series cascade constructs.
- State Machine: a form of logic where the evaluation of the current state and a set of inputs determines the next state, which is then determined by a set of logic rules relevant to each state.
- State Externalization: A capability to transfer application state onto external resources in order to produce stateless application software.
- Rule-oriented programming: implementing state machine logic via a collection of rules or a rule language
- Abstraction of applications: Application designs are used by unidentified application software to allow for reuse.
- Application templates: elements of a conceptual application with clearly specified interfaces.
- Modular applications: applications are comprised of certain components that can be reused.
- Applications run anywhere, location-independent applications: Application modules can be executed everywhere, including on user devices, gateways, edge servers, local network servers, and cloud servers.
- Discovery and Linking: resolves resource connections in order to integrate resources into applications and sets features in application objects.
- Object Constructor: builds application software using models of metadata.

Without a framework to assess their use, how can we determine which are acceptable and what their specific benefits and drawbacks are? Various patterns and techniques are designed that focus on connecting devices using a GUI interface to connect smart devices to IoT networks. Various programming languages have been used to build those interfaces, such as HTML, JavaScript, XML, CSS, PHP, C, C++, C#.NET and Python, which are compiled with the CoAP protocol of the IoT network (Fog Network) and also HTTP for (Cloud Network), Among those who have contributed to this work are Jin Hee Park et al [01]. They stated that Angular.js and Node.js are used to implement the front-end user interface using a web browser and HTML and JavaScript. Users may browse and obtain information from the database using the data filtering and processing module, and the information is displayed on a web page. Additionally, several studies have been conducted on the use of graphical interfaces to link smart devices to the Internet of Things, and those graphical interfaces were created using programming languages that complied with the CoAP protocol (Constrained Application Protocol) used in IoT networks (Fog Networks), some other studies preferred to use the XML file to encapsulate the raw data or rich web content and send it to the endpoint. DoHyeun Kim et al. [02] proposed a new framework for modifying the activities of remote IoT devices using an easy-to-use interface. The user interface enables graphical virtual objects to be shown as resources accessible by a remote CoAP device. Common operations like drag-and-drop and property settings are performed by the VOs to construct services. An XML document with the design is maintained. To act as a link between the device's visual interface for behaviour adjustment and the remote CoAP device itself, a CoAP proxy based on the Californium framework has been created. Furthermore, in addition to the CoAP protocol, several protocols and tools are utilized to provide the endpoint service for raw data or rich web content, Such as MQTT, HTTP, to provide browsing-as-a-service and enable creating web-based representations of objects on the web. As well, Nam K Giang et al. [03]. developed BuddyThing, a platform that offers browsing as a service and enables creating web-based representations of objects. It is constructed on top of a powerful proxy architecture based on WebSockets that connects to things using a number of different protocols, including HTTP, CoAP, MQTT, or any raw data protocols. The platform hosts WoT Apps and creates them utilising the SeedHTML building method. For a resourceful backend server, a small file with a short link to JavaScript code presents rich online content like JavaScript, HTML, CSS, photos, or videos, whereas limiting items just deliver their web service endpoints. Moreover, Hyemin Lee et al. [04] introduced the OPEL Platform, an open Internet of Things platform that makes it possible for users to connect to several kinds of services using the JavaScript API set, a software platform developed with the goal of serving Internet of Things devices. The JavaScript (JS) Programming Language had been used to create the software platform, which comprises the Node.JS runtime framework. The platform offers high-level APIs for various kinds of tasks, such as controlling and interacting with sensors and devices, and it additionally includes an out-of-process model (native service daemon) built with C/C++ that can perform a number of tasks that are primarily handled by the Native Interface Layer (NIL). On the other hand, Praveen Kumar and Umesh Chandra Pati [05]. provided smart house ideas in their paper. Home appliances can be managed through a number of tools, including an interactive graphical user interface via the web. Control and monitoring have been implemented. The set of software is used for setting up and hosting the Raspberry Pi as a server using Apache2 and Node-RED in order to manage the appliances, also for building a client page for the user in Bootstrap through Cascading Style Sheets (CSS), Java, Hypertext Markup Language (HTML), and the hypertext pre-processor PHP. Pachube [06]. A platform enables programmers to link sensor data to the Web and build their own apps on top of it. A web-based service called Pachube gathers data "feeds" from many sources in order to gather and preserve knowledge about different kinds of sensing devices and the data they produce over time. Based on the idea of "triggers," it also provides processing, integration, and data visualisation capabilities in the form of a collection of apps. Amartya Sen et al [07]. This article discusses the installation of the Missouri S&T sensor

cloud, which enables interconnection and utilisation of sensor networks spread across wide geographic areas to deliver sensing as a service. The elements of the client-centric layer included the User Interface, Session Management, Membership Management, and User Repository. A graphical user interface called User Interface was created using Angular, HTML5, and CSS3. Seongju Kang et al [08]. developed a framework based on M2M that interacts autonomously between heterogeneous platforms by applying semantic web technologies through the administrative interface for the web client. The proposed framework consists of an IoT server, middleware, a web client, and things. Israr Ullah et al. [09]. The paper outlines an architecture utilising Visual Studio 2015 C# and the CoAP.NET framework for managing IoT services and virtual objects in a network of hyper-connected objects. The development of a Service and Virtual Objects Management (SVOM) System Prototype in Visual Studio 2015 C#, together with the design of IoT services and virtual object management in a hyper-connected things network, is intended to make administration duties simpler. To connect to IoT resources linked to the Intel Edison Board, utilise the CoAP.NET library. The Java-based Californium framework (Cf), which underpins the CoAP.NET implementation, is created. Microsoft SQL Server is used to store the profile data of registered IoT services and virtual objects. A simple technique to link an IoT interface with a web browser has been proposed by Kumar et al. [10]. A web browser may be used to operate any IoT device due to this integration, regardless of the operating system being utilized. The suggested system relies on browser capabilities like bookmarks, login, etc. to function. Adhitya Bhawiyuga et al. [11]. developed a proposal for an IoT-cloud computing integration platform. The suggested IoT-cloud platform software platform architecture for IoT and cloud computing integration is classified into four major components: Cloud-to-Device interface: a messaging protocol known as RESTful HTTP is being used. Authentication: Token-based authentication in the form of a JSON Web Token. Cloud-to-User Interface: RESTful web service-based Application Programming Interface (API). Kleinfeld et al [12]. presents a project that uses WebSockets, MQTT, and CoAP in order to put the concepts of device integration and real-time communication into practise. The project concentrates primarily on the compilation of data streams from online services and IoT devices using web interfaces. The project provides an interoperability platform using web technologies, including web sockets, JSON data formats, and REST APIs, and so on. Node-RED is used to power the mashup interface. As well, Cristian Gonzalez Garcia et al. [13] presented the Midgar Object Creation Specific Language (MOCSL), a visual DSL (Domain Specific Language). The presented graphic DSL was developed to assist those without programming experience in creating a variety of extremely diverse and incompatible items. It concentrated on several graphic editors for building Android and Arduino apps. The Representational State Transfer (REST) architecture was used to build the web services. The REST service from Midgar, which is based on the HTTP protocol and enhanced with cryptographic security and a protocol that implements XML files, is used to send all data from and to Smart Objects. Prehofer [14] has built a model of the Internet of Things system, which includes sensors, actuators, and RESTful APIs that can be used to find, read, and manipulate IoT resources. RESTful APIs can also be built with programming languages such as JavaScript or Python.

4. SPECIFIC ARCHITECTURES OF INTERNET OF THINGS FOR CONNECTING SMART DEVICES

4.1. SERVICE ORIENTED ARCHITECTURE (SOA) AND INTEGRATING EVENTS IN SOA FOR IOT

Service-Oriented Architecture (SOA) was developed as an architectural paradigm that enhances the service delivery capabilities of current standard systems while maintaining their core features. According to Naghmeh Niknejad et al. [15], due to its adaptability, this strategy has attracted the interest of both organisations that are academic and industrial organizations in the advancement of globally important technologies like Cloud Computing (CC) and the Internet of Things (IoT). Further research, Gupta et al. [16] explained that the SOA method might satisfy IoT needs since it can deliver monitoring of performance, security attack detection, and business intelligence. Avila, Karen, et al. [17] stated in their paper the capability of SOA in the presentation of one or more interfaces via a specific programme. Such interfaces refer to diverse network-accessible strategies. "Services" is the term used to describe these interactions. Given that interoperability is a prerequisite, SOA enables a high level of integration in Numerous systems. Internet-based application-to-application interaction is made possible via SOA, which is seen as an appropriate approach for integrating heterogeneous systems. Yuda Irawan et al. [18] in a paper explain the use of SOA and IoT concepts for smart homes via multiple platforms using the internet network and stated that the development of SOA is a technology-based business approach to combine various information sources from various code sources or platforms. Mohamed, Amin et al. [19] presented in their paper the challenges of adapting Service Oriented Architecture (SOA) in IoT. Using the SOA paradigm's technological development will assist IoT services in enabling sustainability, dependability, and heterogeneous service management. IoT services also have their own standards like RFID and CoAP, while SOA techniques are based on numerous standards like REST, SOAP, WSDL, etc.

4.2. REPRESENTATIONAL STATE TRANSFER (REST) AND WEB SERVICES FOR INTERNET OF THINGS APPLICATIONS

Roy Fielding introduced networked hypermedia systems and implemented the Representational State Transfer (REST) architectural style in 1994 [20]. REST speaks to the idea of stateful resources rather than concentrating on services. Through a URI, resources on the server side are accessible (Uniform Resource Identifier). The REST requirements emphasise generic interfaces, independent component deployment, independent component scalability, and intermediate elements that ensure security, reduce interface latency, and encapsulate legacy systems. Stefan Nastic et al [21] propose the conception of a software-defined IoT particular technique using well-defined APIs that offer an integrated perspective on accessing,

configuring, and running IoT cloud systems, The presentation layer serves as a user interface through a Web-based UI and RESTful API, a software-defined protocol based on MQTT and JSON, and a bootstrap container based on the Spring framework. Maurya, Richa et al [22] in their review paper pointed that REST APIs have a direct relationship to IoT devices since all of them enable the secure and permitted accessibility of smart devices to clients and other IT framework appliances. Every IoT device must have a REST API in order to take full advantage of the IoT, as REST manages approved and secure transactions and provides seamless data flow over network constraints. Web standards like HTTP, XML, URI, and REST are used to develop RESTful Web Services. Benomar, Zakaria, et al. [23] stated that to conceive new IoT services, it is crucial to make it possible for smart devices that connect to the Internet to disclose their capabilities and data on the internet. Web Application Programming Interfaces (APIs), also known as RESTful Web Services, are a model that can expand the range of IoT applications by integrating smart objects with the Web. Garg, H. and Dave, M. [24] proposed an architecture that has been put out; middleware is largely utilised to act as a user interface, disclose device data through REST, and conceal specifications for interacting with sensor data. By using this method, data is encrypted from beginning to finish, protecting it while it is in transit. All restrictions on IoT systems are taken into account in the suggested middleware solution. Communication and data exchange are done through a REST API. By ensuring the REST API is accessible, clients will have the ability to register their IoT devices and allow for safe access to the data the device collects; middleware successfully supports the growth of IoT.

5. CONCLUSION

This paper reviewed the state of IoT research, identified current trends, and explained the design and implementation of a novel architecture to connect smart devices to an IoT network using key technologies, highlighting open research areas and future prospects, and assembling an extensive reference list to help researchers. Through technological advancement and the Internet of Things has the ability to enhance people's lives. The purpose of this paper is to explore different frameworks and architectures of various web technologies used for controlling smart devices, to contribute to future work to develop a framework and architecture that integrate and manipulate all the smart devices. This Article covered most of the elements of designing the framework and architecture. The question at hand is whether the Internet of Things will be a long-term technology that will fail to arise or will lead to technological advancement to another paradigm.

REFERENCES

[1] J. H. Park, J. Chung, and S. Y. Chae, "Design of IoT Information System," Advanced Science and Technology Letters, pp. 80–83, July 2016, doi: 10.14257/astl.2016.135.21.

[2] M. S. Khan and D. Kim, "DIY Interface for Enhanced Service Customization of Remote IoT Devices: A CoAP Based Prototype," International Journal of Distributed Sensor Networks, vol. 11, no. 10, pp. 1–8, 2015, doi: 10.1155/2015/542319.

[3] Nam Ky Giang, M. Ha, and D.-Y. Kim, "Buddy Thing: Browsing as a Service for the Internet of Things," 2015 IEEE International Conference on Services Computing, New York, NY, USA, pp. 122-129, June 2015, doi: 10.1109/scc.2015.26.

[4] Hyemin Lee, Dongig Sin, Eunsoo Park, Injung Hwang, Gyeonghwan Hong, and Dongkun Shin, "Open Software Platform for Companion IoT Devices," 2017 IEEE International Conference on Consumer Electronics (ICCE), Las Vegas, NV, pp. 394–395, 2017, doi: 10.1109/icce.2017.7889367.

[5] P. Kumar, and U. C. Pati, "IoT Based Monitoring and Control of Appliances for Smart Home," 2016 IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT), Bangalore, India, pp. 1145-1150, May 2016, doi: 10.1109/rteict.2016.7808011.

[6] Pachube—The Internet of Things Real-Time web service and applications. Available online: http://www.appropedia.org/Pachube

[7] A. Sen, V. P. Modekurthy, R. Dalvi, and S. Madria, "A Sensor Cloud Test-Bed for Multi-Model and Multi-User Sensor Applications," 2016 IEEE Wireless Communications and Networking Conference, pp. 1–7, Apr. 2016, doi: 10.1109/wcnc.2016.7564764.

[8] S. Kang and K. Chung, "IoT Framework for Interworking and Autonomous Interaction Between Heterogeneous IoT Platforms," Lecture Notes in Computer Science, pp. 217–225, 2018, doi: 10.1007/978-3-030-05755-8_22.

[9] Ullah, M. Sohail Khan, and D. Kim, "IoT Services and Virtual Objects Management in Hyperconnected Things Network," Mobile Information Systems, vol. 2018, pp. 1–19, June 2018, doi: 10.1155/2018/2516972.

[10] K. Kumar, J. Bose, and S. Tripathi, "A Unified Web Interface for the Internet of Things," 2016 IEEE Annual India Conference (INDICON), pp. 1–6, Dec. 2016, doi: 10.1109/indicon.2016.7839142.

[11] Bhawiyuga, D. P. Kartikasari, K. Amron, O. B. Pratama, and Moch. W. Habibi, "Architectural Design of IoT-cloud Computing Integration Platform," TELKOMNIKA (Telecommunication Computing Electronics and Control), vol. 17, no. 3, pp. 1399 - 1408, June 2019, doi: 10.12928/telkomnika.v17i3.11786.

[12] R. Kleinfeld, S. Steglich, L. Radziwonowicz, and C. Doukas, "glue.things: a Mashup Platform for wiring the Internet of Things with the Internet of Services," WoT '14: Proceedings of the 5th International Workshop on Web of Things, pp. 16-21, 2014, doi: 10.1145/2684432.2684436.

[13] C. G. Garcia, D. Meana-Llorian, V. Garcia-Diaz, A. C. Jimenez, and J. P. Anzola, "Midgar: Creation of a Graphic Domain-Specific Language to Generate Smart Objects for Internet of Things Scenarios Using Model-Driven Engineering," IEEE Access, vol. 8, pp. 141872–141894, 2020, doi: 10.1109/access.2020.3012503.

[14] C. Prehofer, "Models at REST or Modelling RESTful Interfaces for the Internet of Things," pp. 251–255, Dec. 2015, doi: 10.1109/wf-iot.2015.7389061.

[15] N. Niknejad, W. Ismail, I. Ghani, B. Nazari, M. Bahari, and A. R. B. C. Hussin, "Understanding Service-Oriented Architecture (SOA): A Systematic Literature Review and Directions for Further Investigation," Information Systems, vol. 91, p. 101491, July 2020, doi: 10.1016/j.is.2020.101491.

[16] P. Gupta, T. P. Mokal, D. D. Shah, and K. V. V. Satyanarayana, "Event-Driven SOA-Based IoT Architecture," Advances in Intelligent Systems and Computing, pp. 247–258, Dec. 2017, doi: 10.1007/978-981-10-5520-1_24.
[17] K. Avila, P. Sanmartin, D. Jabba, and M. Jimeno, "Applications Based on Service-Oriented Architecture (SOA) in the Field of Home Healthcare," Sensors, vol. 17, no. 8, p. 1703, July 2017, doi: 10.3390/s17081703.
[18] Y. Irawan, Chrismondari, Yolnasdi, A. Linarta, Muhardi, and A. Febriani, "Smart Home Light Based Service Oriented Architecture and IoT," Journal of Physics: Conference Series, vol. 1845, no. 1, pp. 1-6, Mar. 2021, doi: 10.1088/1742-6596/1845/1/012070.
[19] S. Balaji, A. Salih, and C. Al-Atroshi, "Adaptability of SOA in IoT Services – An Empirical Survey," International Journal of Computer Applications, vol. 182, no. 31, pp. 25–28, Dec. 2018, doi: 10.5120/ijca2018918249.
[20] R. T. Fielding, "Architectural Styles and the Design of Network-Based Software Architectures," ics.uci.edu, 2000. https://ics.uci.edu/~fielding/pubs/dissertation/top.htm.
[21] S. Nastic, S. Sehic, D.-H. Le, H.-L. Truong, and Schahram Dustdar, "Provisioning Software-Defined IoT Cloud Systems," Aug. 2014, doi: 10.1109/ficloud.2014.52.
[22] R. Maurya, "Application of Restful APIs in IOT: A Review," International Journal for Research in Applied Science and Engineering Technology, vol. 9, no. 2, pp. 145–151, Feb. 2021, doi: 10.22214/ijraset.2021.33013.
[23] Zakaria Benomar, F. Longo, G. Merlino, and A. Puliafito, "Enabling Secure RESTful Web Services in IoT Using OpenStack," 2020 IEEE 17th International Conference on Mobile Ad Hoc and Sensor Systems (MASS), pp. 410-417, Dec. 2020, doi: 10.1109/mass50613.2020.00057.
[24] H. Garg and M. Dave, "Securing IoT Devices and SecurelyConnecting the Dots Using REST API and Middleware," 2019 4th International Conference on Internet of Things: Smart Innovation and Usages (IoT-SIU), Ghaziabad, India, pp. 1-6, 2019, doi: 10.1109/IoT-SIU.2019.8777334.