4-Tier Service-Oriented Architecture for Building Smart Cities

Youssef Bassil

Abstract*: Currently, the world is increasingly focusing on transforming its traditional way of living into a digital, intelligent, mobile, and futuristic new urban environment called Smart City. This new paradigm shift heavily relies on information and communication technologies and has led to the rise of web services, service-oriented architectures, digital ecosystems, intelligent transport systems, e-services, and online social collaboration. This paper proposes a 4-Tier, Distributed, Open, and Service-Oriented Architecture for building Smart Cities. It is a 4-Tier architecture comprising Presentation, Middleware, Service, and Data tiers. It exploits Distributed computing as it is made up of small computational units operating over distant machines. It is open due to its scalable and extendable architecture, and it is Service-based as it is composed of granular interoperable and heterogeneous micro services. At the core of the proposed architecture is the middleware which provides Standardization and Communication Language, Application Programming Interface, Service Registry, and Security Services. All in all, the proposed architecture could prove to be a role model for building sustainable, interoperable, scalable, agile, open, and collaborative Smart Cities for 21st century. Future research can improve upon the proposed architecture so much so that data intelligence can be integrated into the middleware allowing the system to infer, reason, and help in decision making and problem solving.

***Index Terms*: Distributed Computing, Service-Oriented Architecture, Services, Smart City**

I. INTRODUCTION

Cities around the world are in the midst of a transformation. They have embarked on new modernization plans using Smart City concepts to forge social and economic development, and to enhance lifestyle of people and citizens [1]. Cities such as Singapore and Dubai have already started in a similar direction. Each, in its own way, is committed to build a global city for the 21st century. Others, such as Kuala Lumpur in Malaysia, Seoul in South Korea, and Edinburgh in Scotland, to name a few, are re-examining their next stage of growth and development to achieve similar goals. The key to all of these cities has been in their commitment to design and implement advanced information infrastructures. Furthermore, new Smart Cities are embracing the use of socio-economic and citizen-centric sustainable development models, intelligent agent-based concepts, interoperable service-based architectures, and secure collaborative platforms [2]. As a result, Smart Cities are being created to meet new global challenges in terms of environmental protection, employment opportunities for the younger generation, and security for citizens against global terrorism and crime. In short, a Smart City is a fully integrated approach for socio-economic development where government, business, and people are linked through a comprehensive, advanced, and intelligent information infrastructure. If planned and executed well, a Smart City can deliver market leadership, distributed opportunity, enhanced public services, secure environment, and increased profitability [3]. Another way to understand the full scope of a Smart City is to examine what might be called the Smart City Ecosystem. This ecosystem depicts interrelations between people, process, and technology while defining the critical role of information and communication systems, and achieving measurable outcome of specific investments.

This paper proposes a 4-Tier, Distributed, Open, and Service-Oriented Architecture for building Smart Cities. It is a 4-Tier architecture comprising Presentation, Middleware, Service, and Data tiers. It exploits Distributed computing as it is made up of small computational units operating over distant machines. It is open due to its scalable and extendable architecture, and it is Service-based as it is composed of granular interoperable and heterogeneous micro services.

II. SMART CITIES

A city is defined as Smart City when investments in people, social capital, traditional and modern communication infrastructure power sustainable economic development and high quality of life through state-of-the-art Information and Communication Technologies (ICTs) [4] such as Internet, networking, mobile apps, and Intelligent Transport Systems (ITS) [5]. Moreover, Smart Cities are shaped by their innovation and their ability to solve problems using ICTs and Artificial Intelligence [6].

In practice, a Smart City uses different types of electronic data collection sensors to supply information which is used to manage assets and resources efficiently [7]. This includes data gathered from devices, machineries, and citizens to monitor and manage such assets as power plants, water supply, law enforcement, traffic and transportation systems, information systems, schools, libraries, hospitals, and other community services. On a more advanced level, the key in Smart Cities design is to create the intellectual, telecommunication, and IT infrastructure to allow such applications as prompt position location for trucks, buses, and ships; security monitoring systems for public roads and buildings; traffic light management during commuting hours; Tele-education, Tele-health, and Tele-monitoring services; emergency notifications, warnings, and evacuation actions;

Revised Version Manuscript Received on March 05, 2019.

Youssef Bassil, Chief Science Officer of the Lebanese Association for Computational Sciences (LACSC), Beirut, Lebanon.

meteorological observation and urban pollution monitoring; and continuous 24/7 business operations.

III. THE PROPOSED REFERENCE MODEL

This paper proposes a 4-Tier, Distributed, Open, and Service-Oriented Architecture for building Smart Cities. It is a 4-Tier architecture delivering Presentation, Middleware, Service, and Data tiers. It is based on Distributed Computing as it is made up of small computational units dispersed over remote machines. It is Open due to its scalable architecture, and it is Service-based composed of small interoperable heterogeneous commercial off-the-shelf (COTS) components. The proposed model exhibits the following properties:

- *Interoperability*: It is about connecting heterogeneous systems through middleware with standard communication language and message routing capabilities.
- *Data Size*: It is about processing, storing, and analyzing petabytes of data using Big-Data algorithms and high-performance data management frameworks.
- *Security*: It is about ensuring authentication, data security, data integrity, among other services.
- *Scalability*: It is about the capability of scaling up new components and data with the help of loosely-coupled services, non-monolithic systems, and service-oriented architecture.
- *Collaboration*: It is about system-to-system integration through the use of Application Programming Interfaces (API) and open data.

Figure 1 depicts the 4-Tier Service-Oriented Architecture of the proposed model.

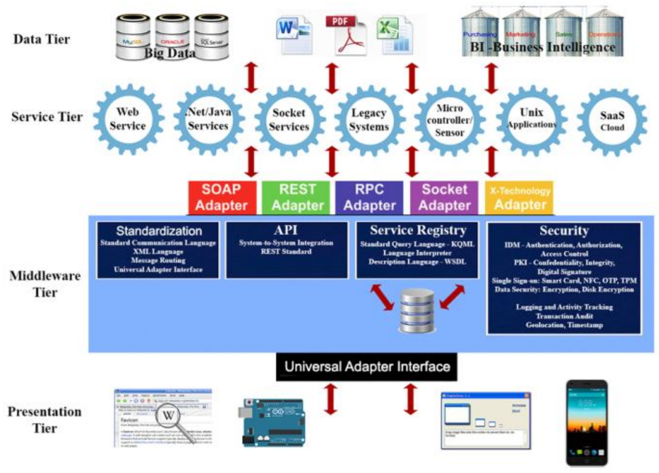

Figure 1 – The Proposed Model

A. Service-Oriented Architecture

Service-Oriented Architecture (SOA) is a model for system development based on loosely-integrated suite of services that can be used within multiple business domains [8]. SOA architectures are based on deploying independent and reusable micro services that interact through a communication protocol over a network. Early programmers realized that writing software was becoming more and more complex [9]. They needed an optimal way to reuse some of the code that they were rewriting. Service-oriented architecture (SOA) was the solution for reducing program complexity and reusing code as much as possible instead of reinventing the wheel. The basic principles of service-oriented architecture are the orchestration of vendors, products, and technologies that exhibit standardized service contract, service abstraction and autonomy, service granularity, service reusability, and service encapsulation. In practice, SOA can be implemented using different technologies and protocols including but not limited to Web Services, RMI, RPC, SOAP, and REST [10]. Today, most SOAs are built using Web Services which are software components designed to support interoperable Machine-to-Machine interaction over a network. They use several technologies to communicate over the HTTP protocol. The most popular one is the SOAP short for Simple Object Access Protocol [11]. However, recently, a new model of web services has emerged, it is known as REST (Representational State Transfer) or RESTful web services for short. REST web services exploit the plain HTTP protocol as a communication medium. The advantage over the SOAP protocol is that REST-based web services are easier to build, manage, and reuse [12].

B. The Presentation Tier

The presentation tier is the front most level of the proposed model that represents the input interface of the system. Usually web browsers, computer applications, workstations, devices, and handheld mobile peripherals are all part of this tier.

C. The Service Tier

The service tier coordinates the communication between various integrated applications, executes commands, processes data, and performs operations through COTS and other service-based systems. Services are by nature heterogeneous as they are built using different programing languages and protocols and target different operating systems. Fundamentally, a service in SOA is an exposed piece of functionality having three properties: A platform-independent interface contract, a dynamic discovery mechanism, and self-contained functionalities [13]. A platform-independent interface contract implies that a client from anywhere, regardless of his underlying platform, can consume any type of service seamlessly; dynamic discovery implies the use of registries and directory services as a look-up mechanism for clients to locate and consume services; and a service is said to be self-contained meaning that it embraces a set of functions and data that no client can have access to before establishing a direct connection to the actual service, a concept often known as encapsulation in object-oriented programming.

D. The Data Tier

The data tier is where information is stored and retrieved. It is a persistent storage of some kind of relational databases, files, data warehouses, and data silos. The service tier has access to the data tier through its COTS and other deployed service-based systems [14]. Data Warehouse is a system used for data analysis, and is considered a core component of Business Intelligence (BI) including reporting, analytical processing, data mining, text mining, event processing, business performance management, benchmarking, and predictive analytics [15].

E. The Middleware Tier

The middleware is a tier sitting between the presentation layer and the service layer while providing the following functionalities and services: 1) Standardization and Communication Language, 2) Application Programming Interface, 3) Service Registry, and 4) Security Services.

F. Standardization and Communication Language

Standardization provides a common interface through which heterogeneous services can communicate regardless of their underlying architectures, programming languages, and protocols. This is accomplished through a Standard Communication Language and a Universal Adapter Interface. In essence, the standard communication language is a common language mainly based on an open data standard such as XML to allow the transmission of data between all the interconnected services via a uniform common protocol. It harnesses the concept of adapter interfaces which consist of protocol converters that convert back and forth messages from source to target machines.

Inherently, the Standard Communication Language is based on an open data standard, namely XML, for exchanging structured information between the different services of the SOA. The language itself is based on XML syntax to format messages sent to and received from inner-system services. In practice, a client requesting an operation sends a message with the appropriate parameters to a destination service. The service then returns an XML-formatted response with the resulting data. Being based on a standard message format, the standard communication language promotes interoperability and standardization for all services regardless of their implementation, target-platform, and underlying technologies. Following is a sample request using the standard communication language. It consists of a sender application whose IP is 10.0.1.20 and ID is 15 invoking a function called "Max" with two integer parameters 20 and 30 respectively, over a service whose IP is 10.0.1.180 and ID is 81.

```
<protocol>
   <sourceIP>10.0.1.20</sourceIP>
   <destinationIP>10.0.1.180</destinationIP>
   <sourceID>15</sourceID>
   <destinationID>81</destinationID>
   <functionInvoked>Max</functionInvoked>
   <functionParams>
        <param>20</param>
        <type>int</type>
        <param>30</param>
        <type>int</type>
   </functionParams>
   <functionReturnType>int</functionReturnType>
   <stamp>5/11/2018 06:25:10PM</stamp>
   <version>1.1</version>
</protocol>
```

G. Application Programming Interface

API short for Application Programming Interface provides a big platform of exposed services that deliver reusable and simplified functionalities to users to build and integrate software applications. They reinforce data abstraction concept by hiding the underlying implementation of complex functionalities to simplify program development. Hence, relieving developers from the burden of re-inventing the wheel and reducing cost and development time. Furthermore, APIs can be used to automate repetitive tasks while allowing the customization and expansion of functions and data based on custom requirements.

H. Service Registry

The Service Registry provides a management platform to facilitate the automatic administration of services in the system, in addition to automating service discovery, integration, and deployment. Another feature of the service registry is that it provides a Standard Query Language to control, monitor, and administer the distributed services inside the SOA.

The Standard Query Language is a declarative language based on a proprietary syntax used to administer every single component connected to the SOA infrastructure. At heart, its purpose is to ease and automate the management and control of the deployed services using control commands issued by administrators via a console manager. For instance, one of these commands is the "bind/unbind" command which is used to connect new incoming web services to the system. Likewise, the "is-run" command is used to check the online/offline status of a web service. Another command is the "grant/revoke" which is used to grant or revoke security permissions to the different services of the system. Similarly, the command "replica" is used to create a replication for an existing service. The Standard Query Language is backed up by an interpreter which scans, extracts, parses, and executes management and control commands in the SOA.

The Service Registry also facilitates the discovery, self-integration, and dis-integration of services in and out of the SOA. The process usually starts when a new service needs to integrate into the present infrastructure. The Service Registry intervenes to validate the Service Description Language (SDL) of the service that is requesting integration. If validation is successful, an acknowledgement is sent to the corresponding service and a new record is created in the internal registry along with important details such as service ID, service protocol, service IP, service functions, parameters, and return data type.

I. Security

The middleware tier employs a security layer which provides all sort of protection against malwares and attacks, and ensures the correct implementation of security polices and access controls. It essentially delivers several features vital to ensure maximum protection most of the time. These features can be classified into broad categories and classes and they are spam filtering [16] used to isolate unsolicited requests; threat scanning used to detect and quarantine viruses, spywares, Trojans and backdoors [17];

firewall [18] used to block unwanted ports and Internet addresses from accessing the internal of the system; encryption [19] used to cipher all messages that travels in and out of the system; access controls used to grant and revoke permissions based on users and service identities; and reporting used to log all communication and activities during the routine operations of the system [20]. Next is a list of security services that are provided by the proposed SOA architecture:

- *Authentication Services* ensure that system entities (e.g. processes, systems, and personnel) are uniquely identified and authenticated.
- *Access Control Services* prevent the unauthorized use of information system resources. They also prevent the use of resources in unauthorized ways.
- *Integrity Services* guarantee the protection of the system through open system integrity, network integrity, and data integrity. This ensures that data are not altered or destroyed in unauthorized manners.
- *Confidentiality Services* ensure that data are not made available or disclosed to unauthorized individuals or computer processes through the use of data encryption, security association, and key management.
- *Non-Repudiation Services* include open system non-repudiation, electronic signature, and electronic hashing.
- *Availability Services* ensure that timely and regular communication services are always available and up running. Availability is intended to minimize the delay and the non-delivery of data passed through the network.
- *Encryption Services* allow the encryption of data communication using a cipher algorithm. In fact, AES (Advanced Encryption Standard) is used to encrypt the communication between all the system components; while, SSL is used to securely encrypt web requests and HTTP communication.

IV. CONCLUSIONS

This paper proposed a 4-Tier, Distributed, Open, and Service-Oriented Architecture for building Smart Cities. It is a service-based model comprising Presentation, Middleware, Service, and Data tiers. It uses small interoperable heterogeneous computational units distributed over remote machines. The middleware which is at the core of the proposed architecture delivers Standardization and Communication Language, Application Programming Interface, Service Registry, and Security Services. The proposed architecture could prove to be a role model for building sustainable, interoperable, scalable, agile, open, and collaborative Smart Cities for 21st century.

V. FUTURE WORK

The proposed Tier-4 Service-Oriented Architecture can be improved in several ways, one of which is adding data intelligence services to the middleware so as to give the system the ability not only to process raw data but also to infer, reason, and help in decision making and problem solving. Moreover, the Standard Communication Language and the Standard Management Language could be extended to provide richer functionalities allowing more control over the different interconnected services of the infrastructure.

ACKNOWLEDGMENT

This research was funded by the Lebanese Association for Computational Sciences (LACSC), Beirut, Lebanon, under the “Service-Oriented Architecture Smart Cities Research Project – SOASCRP2019”.

REFERENCES

1. Mark Deakin, Husam Al Waer, "From Intelligent to Smart Cities", Journal of Intelligent Buildings International: From Intelligent Cities to Smart Cities, vol. 3 no. 3, pp. 140–152, 2011
2. Paskaleva, K, "Enabling the smart city: The progress of e-city governance in Europe", International Journal of Innovation and Regional Development, vol. 1 no. 4, pp.405–422, 2009.
3. Peris-Ortiz, Marta; Bennett, Dag R.; Yabar, Diana Pérez-Bustamante, "Sustainable Smart Cities: Creating Spaces for Technological, Social and Business Development", Springer, ISBN 9783319408958, 2016
4. Komninos Nicos, "Intelligent cities: towards interactive and global innovation environments", International Journal of Innovation and Regional Development vol. 1 no. 4, pp. 337–355, 2009
5. Komninos Nicos, "Intelligent cities: innovation, knowledge systems and digital spaces", London: Spon Press, 2002.
6. Odendal Nancy, "Information and communication technology and local governance: understanding the difference between cities in developed and emerging economies", Computers, Environment and Urban Systems, vol. 27, no. 6, pp.585–607, 2003
7. McLaren, Duncan; Agyeman, Julian, "Sharing Cities: A Case for Truly Smart and Sustainable Cities", MIT Press. ISBN 9780262029728, 2015
8. Nicolai M. Josuttis, "SOA in Practice", O’Reilly, ISBN-10: 0596529554, 2007
9. Ian Sommerville, "Software Engineering", 7th Edition, Addison Wesley, ISBN-10: 0121313156, 2002
10. Thomas Erl, "Service-Oriented Architecture: Concepts, Technology, and Design", Prentice Hall, ISBN-13: 0131858580, 2005
11. Olaf Zimmermann, Cesare Pautasso, Gregor Hohpe, Bobby Woolf, "A Decade of Enterprise Integration Patterns", IEEE Software, vol. 33 no. 1, pp.13–19, 2016
12. Benslimane, D.; Dustdar, S.; Sheth, A., "Services Mashups: The New Generation of Web Applications", IEEE Internet Computing, vol. 10 no. 5, pp. 13–15, 2008
13. Pautasso, Cesare, "Microservices in Practice, Part 1: Reality Check and Service Design", IEEE Software, vol. 34 no. 1, pp. 91–98, 2017
14. Christoph Schroth & Till Janner, "Web 2.0 and SOA: Converging Concepts Enabling the Internet of Services", IT Professional, Nr. 3, pp. 36–41, IEEE Computer Society, 2007
15. Negash, S, "Business Intelligence", Communications of the Association of Information Systems, vol. 13, pp. 177–195, 2004
16. Mark Rhodes-Ousley, Roberta Bragg, and Keith Strassberg, "Network Security: The Complete Reference", McGraw-Hill, ISBN: 0072226978, 2003.
17. William Stallings and Lawrie Brown, "Computer Security: Principles and Practice", Prentice Hall, ISBN: 0136004245, 2007.
18. Matt Bishop, "Computer Security: Art and Science", Addison-Wesley, ISBN-10: 0201440997, 2002
19. Anderson, R., Bond, M., Clulow, J., Skorobatov, S., Cryptographic processors a survey, Proceedings of the IEEE, vol. 94 no. 2, pp. 357-369, 2006.
20. Lampson, Butler W, "Protection", Proceedings of the 5th Princeton Conference on Information Sciences and Systems, pp. 437, 1971.