World Journal of Research and Review (WJRR)
ISSN:2455-3956, Volume-3, Issue-1, July 2016 Pages 28-32

Development of an Intelligent System for IoT using Web Services and Cyber Physical Approaches

Ms. Twinkle Jagani, Ms. Snehal Dilip Sawant,
Ms. Komal Vijay Sonawane, Ms. Anagha Chaudhari

Abstract— **The Internet of Things (IoT) is changing the way we perceive information. It has inspired solutions for a variety of everyday problems. With the advent of IoT, the internet will house several "intelligent "objects capable of making their own decisions and communicate with each other in an efficient manner. Cyber-Physical Systems (CPSs) represent a new paradigm of future intelligent systems. They consist of loosely coupled subsystems which interact with mechanisms of Service oriented Architecture (SoA). One of the most important goals for many organizations is to satisfy their clients' service level agreements with respect to the response time and throughput. Web services are one of the popular technologies to achieve SOA solutions.Web service is a very important candidate technology to achieve SOA requirements that allows the service providers to publish their services to many service consumers.**

Index Terms— **Web service, Performance, Design Patterns, Cyber Physical System(CPSs), Service oriented architecture (SOA), decision making, clustering, Internet of Things (IoT), consensus decision making.**

I. INTRODUCTION

Today, computing capabilities are not just integrated in computers or industrial machines, but also in smart phones, televisions, watches, cars, or even light bulbs. The interconnection of all these different devices with each other and with humans leads to the Internet of Things (IoT). The term Internet of things (IoT), was coined by Ashton in the year 1999[2]. A Cyber-Physical System (CPS) is an interconnection of different control units that collaborate. Web service is a very important candidate technology to achieve SOA requirements that allows the service providers to publish their services to many service consumers[4]. Web service technology provides powerful solutions to represent, discover and expose communications with the other business entities.

Ms. Twinkle Jagani, Student, Department of Information Technology, Savitribai Phule Pune University/Pune, India,
Ms. Snehal Dilip Sawant, Student, Department of Information Technology, Savitribai Phule Pune University/Pune, India,
Ms. Komal Vijay Sonawane, Student, Department of Information Technology, Savitribai Phule Pune University/Pune, India,
Ms. Anagha Chaudhari, Assistant Professor ,Department of Information Technology, Savitribai Phule Pune University/Pune, India,

To form an intelligent system, interaction and coordination with other entities is needed. Communication networks at this level are less focused on hard real-time properties, but more on offering and using services. A lookup service is a web service published by service providers to provide data to their service consumers. The data provided by these services tends to be stable and of a size that it can be loaded to the server provider's server memory. Moreover, it is expected that these data are popular and frequently requested to be used by a variety of applications. Data about cities, currency exchange rates and weather data are examples of lookup data that are needed by many applications [5].

II. SYSTEM ARCHITECTURE

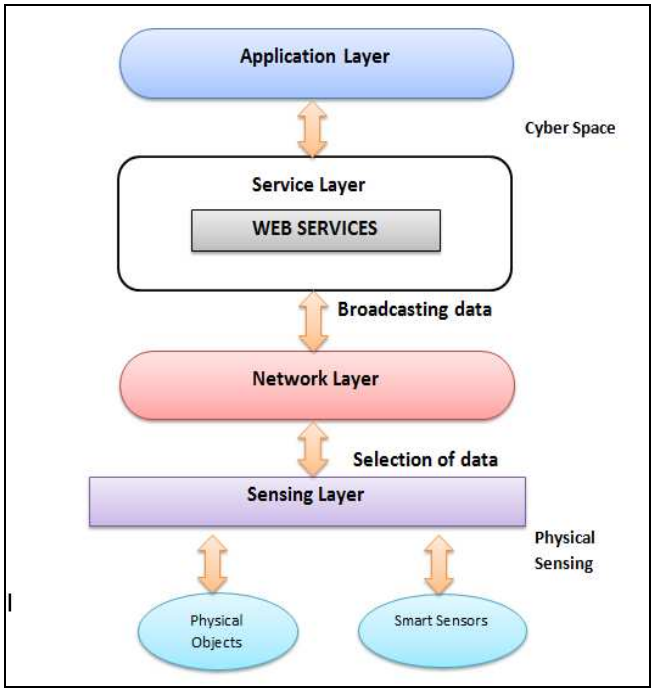

Fig1. System Architecture

Initially data is collected through the physical objects using smart sensors and passed to the sensing layer. Sensing layer does the job of selecting the sensible or required data which is further broadcasted by the network layer to the service layer. In the service layer using web services the data is processed and particular service is provided to the application layer. Application layer is the final layer where user can interact

with the system. This whole procedure occurs in Cyber Physical Space.

III. PROPOSED SYSTEM

SOA services coupled with decision making capability helps in efficient processing of information. This proposed solution provides a general framework for quick decision making in a rapidly changing IoT environment with a SOA based framework. Thus the SOA framework combined with the decision making capability serves as an effective solution for information processing over the internet of things [2]. The implementation can further be analyzed with the help of the following section:

1. IDENTIFYING SERVICES FOR IMPLEMENTATION
2. CREATING A SERVICE COMPOSITION
3. CLUSTERING OF SERVICES
4. CLUSTER CONTROLLED DECISION MAKING

(1) IDENTIFYING SERVICES FOR IMPLEMENTATION

This forms the first and primary step in developing a SOA framework. Based on the application layer requirements, services have been identified. Each service should be created to handle specific requirements from the application layer. The services handling these data are created with the specifications that apply to each service i.e. the acceptable temperature and pressure ranges are defined within each service. Each service acts as a standalone unit. These services could be deployed and made available to any user.

(2) CREATING A SERVICE COMPOSITION

In an IoT environment an array of services could be required for processing a single application. Hence appropriate service aggregation is necessary for efficient implementation. The services could be grouped in order to provide useful information to a higher level application. For example, consider a scenario where a higher level service takes in an input parameter and uses two other lower level services to fetch specific information. Later on all these information could be aggregated to generate an appropriate output.

(3) CLUSTERING OF SERVICES

Clustering enables proper organization of the services developed. It provides better performance in terms of service selection and discovery. Clusters are created based on homogeneity, i.e. similar services or services handling similar streams of data from an IoT network are placed in a cluster. Several algorithms are available for clustering.

(4) CLUSTER CONTROLLED DECISION MAKING

The final process of decision making is achieved when all the participating services converge to provide a common solution. As mentioned earlier co-operation among the various services is essential for a consensus decision making. Thus within each service a local consensus is calculated which is then combined in an iterative manner leading to a global consensus. With consensus decision making there is higher trust and reliability among the decisions offered.

IV. CLUSTERING

In IoT data is generated in large volumes. With continuous ranges of values captured from the various sensors, there needs to be efficient means of handling such large volumes of data. The nature of the data captured could be noisy, corrupted, heterogeneous or high dimensional in nature. Under such circumstances a cluster based approach could prove to be an ideal solution [2].

Fig2. Cluster of various devices**.**

Cluster analysis or clustering is the task of grouping a set of objects in such a way that objects in the same group (called a cluster) are more similar (in some sense or another) to each other than to those in other groups (clusters) [1]. Clustering is a data mining (machine learning) technique used to place data elements into related groups without advance knowledge of the group definitions [3]. Based on specific parameters, the incoming data could be grouped into specific clusters. Clusters are usually used in load balancing and ensuring data persistence. In the proposed paper, clusters could be used as a reference framework for grouping data into specific sets. The concept of clusters in reference to this paper can further be explained with the help of ADDM method. This method defines the notion of clusters created for this implementation [2]. **ADDM** expands into **alternating direction method of multipliers.** It is based on the concept of decomposition-coordination. It is very useful to solve large problem by efficient consensus optimization. Let us assume there are 'N'inter-connected smart objects. Let ***fi(x)*** denotes the **functionality of the smart object**. The goal of each cluster should be to reach a common acceptable solution with respect to the other clusters. So this can be summarized as

$$\min(x)\, f(x) = \sum_{i=1}^{n} fi(x)$$

This represents the minimum factor by which all the clusters agree.
ADDM provides good convergence results, i.e. good accuracy with reasonable amount of time.

V.COMMUNICATION PATTERNS

Model-based development assisted by patterns is quite common in the area of protocol engineering. For general communication protocols there are patterns for areas such as architecture, interaction, control, or management. They

provide general solutions such as reliable communication over unreliable media, segmentation, or handling of duplicated data. Coordination describes what messages are transported and how systems use them to coordinate tasks between each other This section is interested in general communication and interaction tasks especially in CPS communication [4].

.

A. EXISTING PROTOCOLS

There are a number of protocols and middle wares that are currently in use or in development for CPSs. Some of them are: Web services, Universal Description, Discovery, and Integration (UDDI), OPC Unified Architecture (OPC-UA), and the data centric approach Data Distribution Service (DDS). This section refers to the goals and usage areas of these protocols. It reviews their intentions and goals, as well as their basic functionality [4].

1) WEB SERVICES & SERVICE-ORIENTED ARCHITECTURE:

A SOA describes a set of black-box components, that provides services via an Application Programming Interface (API) which is accessed using communication services. The term web service describes such a communication service that is based on web technologies. The World Wide Web Consortium (W3C) defines web services as "set of components which can be invoked, and whose interface descriptions can be published and discovered". Depending on the used implementation, the API may be self-descriptive. Web services are mainly used as middleware for calculation and for the query of data [4].

2) UNIVERSAL DESCRIPTION, DISCOVERY, AND INTEGRATION:

UDDI is a discovery mechanism for web services. Service providers register their APIs at registries. Other entities that need to consume services are able to lookup services in a registry and automatically connect to them via the published API. The communication of the registration and discovery process is based on web services. The architecture contains server(s), client(s), and registry(s). The discovery process is based on a Publish-Subscribe pattern. A server registers a service by publishing its API to a registry. A client queries the registry and receives information about the servers that deliver the requested service. Afterwards the client can subscribe the service and is able to interact with it [4].

3) OPC UNIFIED ARCHITECTURE:

OPC-UA is a communication middleware for manufacturing systems. Its main goal is the seamless exchange of data between different entities. The data structure is based on an address space containing several nodes. Each node represents a resource, like a piece of data or a functionality. The nodes are provided by servers that are deployed at sensors or controllers. Furthermore the architecture can be expanded hierarchically, so that multiple servers can combine their address spaces. Clients can access these nodes. A discovery mechanism similar to UDDI allows to discover OPC-UA endpoints during runtime. Semantic representations describe the provided data and enable interoperability. The main communication pattern is Request-Response. It is used to query the address space as well as to access node data or to call procedures.

4) DATA DISTRIBUTION SERVICE:

DDS is a vendor independent data-centric middleware for data distribution in loosely coupled systems. It uses a flexible Publish-Subscribe architecture and provides QoS requirements. The data model is based on topics, instances, and samples. A topic is a type of data, an instance is an instance of a topic, and a sample is a certain state of an instance. Publishers provide topics and guarantee a QoS, such as a sample rate or a historical storage. Subscribers subscribe to filtered topics and require a QoS, such as time limits. The middleware acts transparent, i.e. subscribers and publishers are not aware of each other. Instead, when data is provided by the publisher, the middleware transmits the corresponding data to the right set of subscribers. QoS policies define what requirements are required, but the subscriber is not interested on how this is actually achieved or where the data originated.

B. PATTERNS

Each of the reviewed protocols fulfills the tasks to solve certain problems. The comparison of these tasks and the implementations of the different protocols shows similarities. Some of the protocols solve common tasks and also use similar strategies to fulfill them. This section lists these tasks and groups them to pattern classes.

1. DISCOVERY:

Discovery allows protocols to match communicating entities on the fly, i.e. without prior static configuration. This task is included in the protocols UDDI and OPC-UA. The main idea is to provide a directory where entities can register themselves and their offered services. Requesters can find matching entities by querying the directory for required services. The requester receives the network address of the service-providing entity and can interact with it. The architecture and exchanged messages are similar on both protocols as they are based on a client-server architecture and use a separate entity for the necessary service directory.

2. REQUEST-RESPONSE:

Request-Response based data transmission is used to access data that is provided by a resource on demand. The pattern requires a receiver entity to actively request a resource at a needed time rate from a provider entity.

3. PUBLISH-SUBSCRIBE:

Publish-Subscribe allows entities to subscribe to topics and to specify QoS requirements. Publishers publish data to topics, that is automatically transferred to entities that

subscribed to these topics before. The main difference to the client-server based Request-Response approach is that the architecture is not based on a single server but acts P2P. Only the topic of a data is relevant but not the source of it. DDS and—in the future—OPC-UA support this kind of communication.

VI. DECISION MAKING

The decision making layer makes use of semantic information and knowledge abstracted from the lower layers to reason, plan and select the most suitable outcome. Hence decision making can be viewed as a form of "selecting". There are several ways to model a selection process, like, ***Markovian decision process***, ***multibandit armed problem*** or using a ***multi agent based learning*** [2].

VII. SOA DESIGN PATTERNS

Proven solutions to common problems faced in the design of Service-Oriented Architecture. Describes how to solve a problem that can be used in many different situations. There are some design patters, originally inspired from object oriented programming, that can be used to improve the performance in terms of response time and throughput of web services [6]. There are few design patterns mentioned below:

1 .SINGLETON DESIGN PATTERN
2. CACHING DESIGN PATTERN
3. OBSERVER DESIGN PATTERN
1. SINGLETON DESIGN PATTERN

Applying singleton design pattern on the service consumer side limits the number of the created lookup service instances to only one instance, by creating a singleton instance of the service [5].

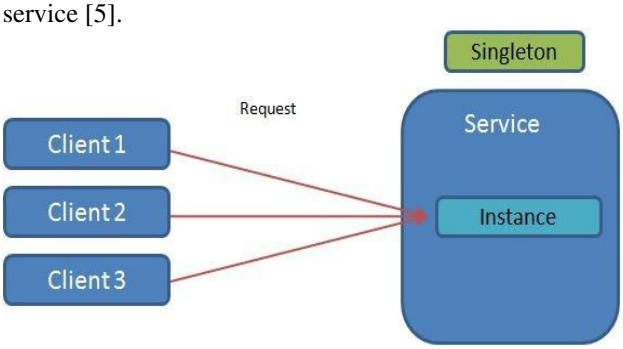

Fig3. Singleton design pattern

2. CACHING DESIGN PATTERN

The caching design pattern keeps the frequently used data in a temporary place because it tends to be repeatedly used by the service consumers and it is expensive to fetch each time [5]. For all subsequent requests, the data is retrieved directly from the cache instead of from the file system or from the database. This redesign of the service provider's lookup services by applying caching pattern is totally transparent to the service consumers [5].

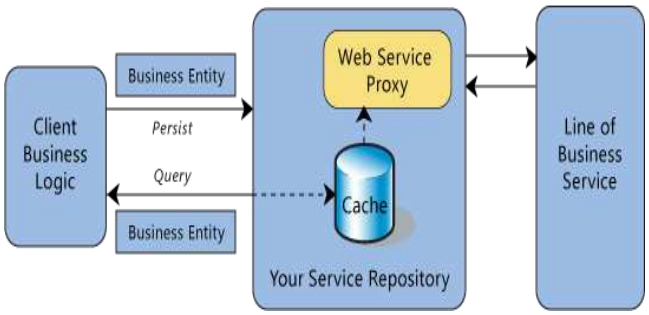

Fig4. Caching design pattern

4. OBSERVER DESIGN PATTERN

The observer design pattern is a behavioral design pattern that defines a way to keep some objects to be notified of a specific change. The Observer defines a one-to-many relationship so that when one object changes state, the others are notified and updated automatically [6].

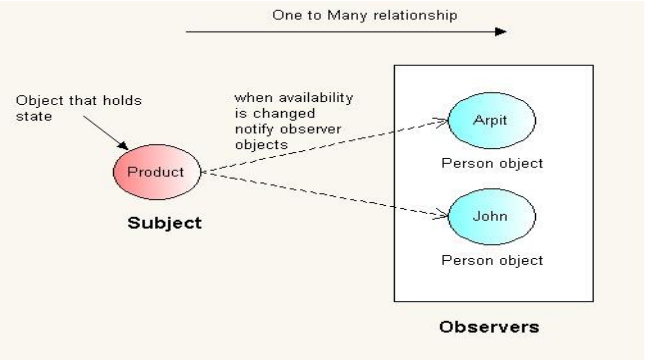

Fig5. Observer design pattern

There are two modes in which service consumer can send request to target web service provider they are described below:

SEQUENTIAL MODE

Web service is dedicated to one service consumer, who is submitting multiple requests to the target web service in a sequential manner. This mode measures the benefit of applying the caching pattern over the singleton pattern [5].

CONCURRENT MODE

Concurrent mode provides more dynamism than the sequential mode by simulating concurrent service consumers requests. Furthermore, each service consumer can submit multiple requests [5].

VIII. CONCLUSION

CPS provides an efficient platform to develop solutions for IoT. With the help of web services physical world and cyber world can be connected together through internet. CPS is similar to IoT sharing same basic architecture. This paper aims at providing basic architecture for IoT, CPSs solutions with the help of web services.

.

REFERENCES

[1] Aditya G. Bhor, Komal K. Rahane, Poonam P.Rajurkar, Neha S. Pathak "Development of an Excellent System for Information Retrieval Corresponding to Vivid Comparisons of Distinct Methodologies Incorporating DR and Clustering Schemes".

[2] Sandhya Balasubramaniam, R.Jagannath "A Service Oriented IoT using cluster controlled decision making". 978-1-4799-8047-5/15/$31.00_c 2015 IEEE

[3] Poonam P. Rajurkar, Aditya G. Bhor, Komal K. Rahane, Neha S. Pathak, "Efficient Information Retrieval through Comparison of Dimensionality Reduction Techniques with Clustering Approach ", *International Journal of Computer Applications (0975 – 8887), Volume 129 – No.4, November2015.*

[4] Dominik Henneke1, Mohammad Elattar1, and J¨urgen Jasperneite "Communication Patterns for Cyber-Physical-Systems", 978-1-4673-7929-8/15/$31.00 c 2015 IEEE.

[5] Yehia Elshater, Patrick Martin "Using Design Patterns to Improve Web Service Performance" 978-1-4673-7281 -7/15 $31.00 © 2015 IEEE DOI 10.1109/SCC.2015.106

[6] Akram KAMOUN, Mohamed HADJ KACEM and Ahmed HADJ KACEM, "Feature Model for Modeling Compound SOA Design Patterns ", 978-1-4799-7100-8/14 , 2014 IEEE

[7] http://internetofthingsagenda.techtarget.com/definition/Internet-of-Things-IoT

[8] http://home.deib.polimi.it/matteucc/Clustering/tutorial_html/

[9] http://cyberphysicalsystems.org/