1. **OBJECTIVE**

The idea of the Internet of Things has sparked countless opportunities for various heterogeneous devices to coexist in a network. IoT devices, particularly sensor devices, are frequently used in severely resource-constrained environments, necessitating the need to increase their functionality and lifespan in terms of energy consumption. The implementation of green communication and system models in IoT has been a key hard issue since the devices in IoT have limited energy sources and frequently run on batteries with a specified energy capability. In order to increase the lifespan of the entire IoT network, this article proposes a system model for G-IoT and an energy-efficient plan for IoT devices to solve the concerns with energy efficiency.

1. **INTRODUCTION**

The envisioned concept of IoT includes intelligent communication between different communicable or noncommunicable heterogeneous objects with less human intervention, fostering context awareness within a framework of smart objects. Those objects should be able to grasp the desired characteristics like Automation, Intelligence, Dynamicity, and Zero-Configuration. Some of the key challenges in IoT related research have been identified by the researchers till now which are Heterogeneity, Scalability, Interoperability, and Security and Privacy. According to a recent study by Gartner, approximately 26 billion devices will be connected to the network by the year 2020. These devices will produce a lot of electronic waste and will also consume a significant amount of energy to execute different tasks. This will eventually pose a challenge in near future to reduce the energy consumption and will also demand for new ways of developing a green communication across the network.

The vision has become sparser and promising since the promotion of green IoT(G-IoT). As a result, reduction in context of energy consumption and the deployment of green IoT, should be considered as the future challenges in IoT along with the current projected challenges. IoT devices as it actively relates to cost and availability of the IoT network. Thus, energy consumption has become a core issue in future internet and different algorithmic approaches have been initiated for different effective solutions like complementing hardware or different system-based approaches. The most highlighted energy efficient methodologies includes different power-down mechanisms, defining systems with two or multiple states using ALG-P algorithm, scheduling with deadlines or minimizing response time .The main contribution of this is to propose a system model which is apt to resolve the challenges of IoT related research and an energy efficient mechanism for heterogeneous IoT devices that conforms to the proposed IoT system model topromote G-IoT and increase the longevity of the IoT network.

1. **HISTORY**

Before this system model there was state-of-the-art reference model and architecture for IoT namely IoT-A Architectural Reference Model (ARM) and IoT6 architecture design.

The main objective of IoT-A is mainly devoted to providing a generic ARM to discourage the emergence of silo application based IoT architecture rather to encourage to provide several ways to derive an IoT architecture.

Side by side the IoT6 architecture design leverages different IPv6 functionality.

Then a distributed architecture called Distributed Internet like Architecture for Things (DIAT) has been proposed to address the interoperability and heterogeneity issue in IoT.

A conceptual system skeleton was proposed to reflect upon the flexibility and extendibility issues by following the Service Oriented Architecture (SOA) and Model-View-Controller (MVC) design pattern in IoT driven system.

Large scale IoT system has been divided into subsystems namely IoT unit to decentralize the capability of the system and thus has proposed distributed brokers to gather data or information from different sensors and actuator within the IoT unit.

A message scheduling algorithm has been proposed considering MVC design pattern to stabilize the HTTP derived message stability in IoT system to reduce missed deadline of messages.

An IoT system model and energy efficient routing protocol called Pruned Adaptive IoT Routing (PAIR) for heterogeneous IoT was proposed.

Ambient intelligence System (AMI) that possess a profound impact on reducing the energy consumption of different sensor devices and actuators in IoT network was proposed.

A middleware for home automation network and a RESTful interface for heterogeneous IoT based home appliances was proposed.

An IETF constrained RESTful environments (CoRE) for resource constrained devices or sensors in a network was used.

**3.WORKING**

The proposed system model in Fig.3.1 is comprised of conventional device hardware which represents the physical hardware for example sensors, home appliances. These devices will be connected with the embedded web server. Embedded webserver hosts RESTful web service to communicate with the cloud server for virtualization of objects. Virtual objects will be transferred to server application for service lookup and will host the virtualized object service executable applications. Application server interface will host the skeleton of the server-side code and client will not be able to access the server-side code directly rather client will communicate with the application server application through an interface. The admin can directly access the Server application directly without the use of server interface. Nowadays a lot of open-source software packages are used for different devices in the system but some of the open-source software might be malicious and the untested material may contain bugs that can severely jeopardize the whole network by breaching the security. In order to find a solution, we are going to introduce the concept of a test platform in IoT system model. In our system model, inside cloud server, in case of new service implementation, the server application will test the service package inside the application repository. The application repository will test the service package before deploying it in the environment.

Diagram

Description automatically generated

**FIG. 3.1**

**ENERGY EFFICIENT SCHEDULING ALGORITHM**

The proposed energy efficient algorithm in Fig. 3.3 is solely devoted to schedule the duty cycle of different sensors and appliances. The algorithm is framed in the proposed system model so that the model can incorporate with the scheduling algorithm and can also serve its’ true purpose. The algorithm has three core stages such as On-duty, pre-off duty and Off- duty and Fig. 3.2 illustrates the activity diagram of the scheduling algorithm.

Diagram

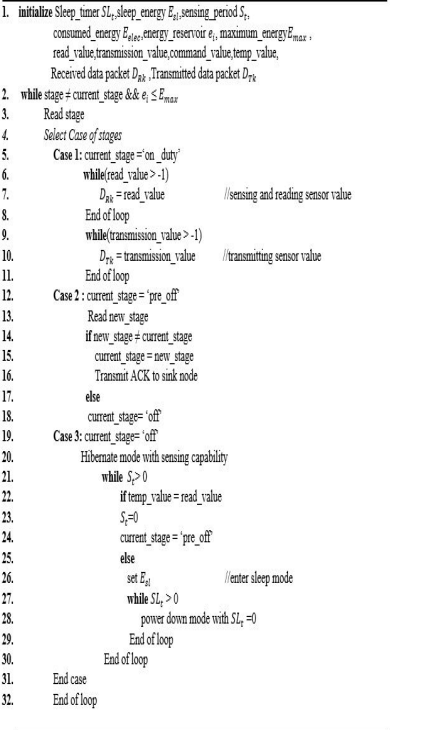
Description automatically generated

**FIG.3.2**

1) On duty: In this state device will be performing with its full-fledged capability. A device within the network will sense, receive, and transmit the data accordingly. Apart from that, usually the devices specially the sensor devices have limited processing capability to process data in a resource constrained environment. But in the proposed system model the sensors and devices will only either act as a relay node or sink node based on the devices capability and the whole processing task of the devices and sensors is performed in the virtual environment in the cloud server.

This enhancement in the network will reduce the tasks of the devices and will fuel for better data processing opportunity to ensure the QoS of any sort of services.

2. pre-off duty



**FIG.3.3**

This state will be activated after On-duty state when the device will be idle for sometimes. This state is bidirectional can switch state to both On-duty and Off-duty when it requires to do so. During Pre-off stage, a device can only receive and transmit the necessary commands from the sink node. If the sink node sends any new request, the pre-off stage will be changed to on duty to perform its tasks. In a nutshell, the devices will be activated but with a limited capability of receiving and transmitting. On the other hand, if the sink node sends command to the device to go to sleep/hibernate/power off mode the Pre-off stage will be changed to the corresponding mode.

3. Off duty

This stage holds three states to save energy in different circumstances and the energy efficiency of the entire cluster or network mainly depends on this state. This state can perform three tasks as follows,

1. Hibernate:

Hibernate state is the state where the device will have small power to only sense the environment before going into more energy efficient state. No transmission or reception of data will be occurring in this state and only sensing can trigger the next energy efficient task or can move the state of the device to previous state. In this state the device will use only the least amount of power and for the devices that have the renewable energy capability will recharge by that time which will extend the device’s life expectancy.

1. Sleep

Sleep is a power saving state that can quickly allow the device to use resume the full-power operation. The device will immediately stop working in this state but will start again when the device is required to resume again.

1. Power off

Power off is the most energy efficient state that will put the device into deep sleep. The consumption of energy is supposed to be 0 at this point since all sort of energy consumed tasks of the device will be stopped by that time. The sink node will directly trigger the device when any necessary task should be performed. The embedded webserver will be responsible for putting the device into right state. If the device is only required to sense the environment and any change in the environment that requires to be received and transmitted to sink node, the device will be in the pre-off state first and if it requires the full capability of the device the embedded webserver will send command through corresponding sink node to the device to go On duty mode.

1. **TECHNICAL SPECIFICATION**

**SOFTWARE SPECIFICATION**

Software specifications in this system model include Virtual Environment, Server Application and Interface, Application Repo, User group and application that includes Admin access application and Client application.

1. Virtual Environment

Virtual environment provides the support of virtualization of the physical objects in the cloud server. It will host the virtual objects and composite virtual objects as application(s) hosted inside the virtual machine with enhanced processing capabilities. The virtual environment also has the responsibility to mesh up multiple objects for initiating composite objects. Composite objects are made according to the service request of the user or based on situation in the system. Cyber objects are also tested in the application repo and the test result is conveyed to the server for performing the deployment in the virtual environment.

1. Server Application and Interface Application

Server can communicate to the client through the interface of can also be accessed directly by the admin for any necessary modification. The server also keeps track of the services and devices that are currently available within the IoT network and thus works as a registry. Activity in the virtual environment and physical environment will be monitored and will be notified to the authorized user. Server interface protects the server from directly accessing through the client application. Only the admin can directly access and modification of the server.

1. Application Repo Application repo or repository

It is responsible for service look up for objects and based on the search result, the deeming perfect service package will be installed to the object. Application repo hosts the test services for different applications. Before deploying an application in

real case and within the virtual environment, application repo runs a test of that applications’ capability and functionalities and look for any malicious or faulty codes in the service package according to the service request. If the test fails, the search for more capable service goes on in the service package until the test phase is successfully completed.

Application repo sends the test result of new application to the application server and this result is analysed by the server for sending request to the virtual environment to add the tested application. D. User group and application User application is the client-side application. The applications are categorized into two subsets based on the authentication policy.

1. Admin access application

Admin can directly access the application server to perform necessary modification and can monitor the overall system performance. To make necessary changes in the network or in the server the admin interface can directly access the server.

1. Client application

The client can send request(s) to the application server thorough application interface and can use the objects that are hosted as application in the virtual machine.

**HARDWARE SPECIFICATION**

Hardware specification in this system model includes Physical Devices and Embedded Web Server.

1. Physical Devices

IoT network is comprised of both IP enabled sensor and physical devices. Sensor devices have low memory and data processing capability and on the contrary some of the physical devices have better processing capability than the sensor devices. So, it is feasible to categorize all the devices through their capability. In our scenario we devise a cluster of low powered and less capable sensor devices with IP address which act as relay nodes. On the other hand, devices with better connectivity and capability act as sink nodes that communicate with the embedded devices. Relay nodes produces the data and send it to the sink node so that the sink nodes can redirect the aggregated flow of data to the embedded web server for further processing.

B. Embedded Web Server

Embedded web server will host IETF constrained RESTfull environment. This server will exploit its limited connectivity through Ethernet shield. In the REST architectural style, the data and functionalities are considered as resources and can be accessed directly using Uniform Resource Identifiers (URI).This architectural style is based on simple client/server architecture with some constraints and designed to use a stateless communication protocol. The RESTfull applications are simple, lightweight and fast and ideal choice for resource constrained environment. Primary task of this web server will be to bridge the gap between physical devices and virtual objects. HTTP protocol will be used for communication between physical devices and embedded web server since HTTP can also be used between constrained devices, servers and devices in the constrained environment or even across the Internet. This server environment will receive the device specification, for example product id/MAC id, assigning device IP address and will notify the cloud

server for creating a virtual object in the cloud virtual environment with the specifications or the semantic description of the device that includes device features and capabilities as parameter as well. Another task of the embedded web server is to schedule the physical and sensor devices. In an IoT network some of the devices remain idle most of the time and those devices only react upon request. Scheduling will let this type of devices to become energy efficient and the energy consumption will be directly proportional to device utilization.