INTERNATIONAL JOURNAL OF ADVANCE RESEARCH, IDEAS AND INNOVATIONS IN TECHNOLOGY

**ISSN: 2454-132X**
**Impact factor: 4.295**
***(Volume 4, Issue 4)***
Available online at: www.ijariit.com

# Context-aware middleware in QRCS System

*Abhilash Javalkoti*
*abhilashj.6193@gmail.com*
*Veermata Jijabai Technological Institute, Mumbai, Maharashtra*

*Dr. Anala Pandit*
*aapandit@mc.vjti.ac.in*
*Veermata Jijabai Technological Institute, Mumbai, Maharashtra*

## ABSTRACT

***In the times of a disaster, there is a need for coordination of resources and quick response in order to minimize and rectify the damage done. Quick Response Co-ordination System (QRCS) provides a robust, mobile web application architecture to locate, allocate and track resources. It concentrates on quick communication and coordination between the disaster management team and resource handlers for better managing and mapping of resources for faster aid and rescue. A context-aware infrastructure manages the context model representing contextual information and provides appropriate information. In this paper, we introduce Context-Aware Middleware for QRCS as a context-aware infrastructure for a distributed intelligent QRCS and discuss the JSON-based context modeling and reasoning approach which will be used in the infrastructure.***

***Keywords***: *Quick Response Co-ordination System, QRCS, Context-aware, Context-broker*

## 1. INTRODUCTION

The use of information and communication technologies (ICTs) has been advocated for addressing the obstacles and improving decision-making for emergency and disaster management [10]. A number of ICT support systems and frameworks, both conceptual and application-based, have evolved to support the high time and collaboration intensive task of emergency and disaster management [10]. A number of the existing systems, ongoing research projects, supporting systems, and concepts were surveyed and classified based on their use in the four stages of comprehensive emergency management (CEM) [6]. They were further classified into monitoring, live and simulation systems.

In the vision of ubiquitous computing environment, the computer system understands their situational contexts and provides appropriate services to users [3]. To realize this vision, it is important to develop a context-aware infrastructure which can help ubiquitous agents, services, and devices become aware of their contexts because such computational entities need to adapt themselves to changing situations [3]. Context is any information that can be used to characterize the situation of an entity. An entity includes a person, a device, a location and a computing application [3]. For example, environmental attributes such as noise level, light intensity, temperature, and motion, system and device capabilities, available services, and user's intention could be context. A context-aware system uses such contexts to provide relevant information and/or services to the user depending on the user's task.

The context is the information that can be used to characterize the situation of the entities that are considered relevant to the interaction between a user and various applications, including both the user and the applications. This leads to the consideration that applications should take advantage of the contextual information, such as the user's location, to offer greater services to the user; therefore, ubiquitous computing environments must provide middleware support for context-aware applications [2]. Recently, various applications of the user's location-based services have been required such as in buddy finding services and position tracking services [1].

During a disaster, it is very important to create plans to decrease the effects of the disaster in order to preserve human life and valuable assets. It is very crucial to utilize resources available to us in an efficient and effective manner in order to lessen the impact of the disasters (natural/man-made) which occur impetuously. This appropriate utilization of resources can be achieved by proper organization and management of the resources and responsibilities. To address these issues QRCS was proposed, but QRCS as a solution had a dependency on (Admin or Control room) to make decisions. This paper introduces Context-Aware Middleware for QRCS System as a context-aware infrastructure for a distributed intelligent QRCS system and discusses the JSON-based context modeling and reasoning approach which is used in that infrastructure. Context-aware middleware aims to help devices, services, and agents to become context-aware and provide real-time context to the decision maker (Context Broker). The Context Broker based on the inference will send commands to agents and services for an appropriate response.

The remainder of the paper is organized as follows. Section 2 discusses the related researches about context-aware infrastructure and context modeling approach. Section 3 describes the concept of the Quick Response and Coordination System (hereafter referred to as QRCS) which is the background of our research. In section 4, we introduce our proposed context-

 

aware infrastructure and its components. Section 5 describes json-based context modeling and reasoning approach which is used in infrastructure. Finally, in section 6, we conclude this paper with some remarks.

## 2. RELATED WORK

Building context-aware systems is a complex and time-consuming task due to the lack of an appropriate infrastructure or the middleware-level support [4]. An appropriate infrastructure for a context-aware system should provide support for most of the tasks involved in dealing with the context.

The Gaia project developed at the University of Illinois is a distributed middleware infrastructure that provides support for context-aware agents in smart spaces [1]. UMBC developed a broker-centric agent architecture (CoBrA) to provide runtime support for context-aware systems in an Intelligent Meeting Room environment [9]. Context modeling approaches are classified by the scheme of data structures which are used to exchange the contextual information in the respective system. In various kinds of approaches, ontology-based approach such as CONNON and CoBrA system has a number of advantages of expressiveness, knowledge sharing, logic inference, knowledge reuse, and extensibility [5]. AT&T Laboratories Cambridge in U.K presented a platform for context-aware computing which enables applications to follow mobile users as they move around a building [7]. The platform is particularly suitable for the richly equipped, networked environments. Users are required to carry a small sensor tag, which identifies them to the system and locates them accurately in three dimensions. MyCampus (Sadeh et al. (2005)) is a much more complex system, in which the agents retain bases of various knowledge about their users, in what the authors call an e-Wallet [8]. There are also agents associated with public or semi-public services (e.g. printers). The e-Wallet manages issues related to security and privacy.

The EPIC project developed in 2010 used in the Haiti earthquake uses twitter for message communication [6]. It converts normal twitter messages to Hash-based syntax. It provides location, status, and damage; but it's just a reporting and monitoring tool and doesn't provide any response. The MRCCFR real-time info for first responders is a command and control system. It sends video annotations to first responders describing the situation but has only 60% accuracy [6]. In all the above-discussed disaster management systems, the application of each system is restricted to a particular area [6]. Hence the focus of our research was to build a mobile application based system and use GPS for location tracking in QRCS.

In this paper, we are trying to focus only on using Context-Broker as a middleware and use it for what it is good at agent registration, data acquiring, reasoning, autonomy, robustness and decision making. We assume that the information can be provided by the agents and that interfacing with the user can be done in the mobile apps.

## 3. QRCS

QRCS, as the name suggests, concentrates on quick communication and coordination between the crisis management team and resource persons so as to better managing of resources and faster aid. QRCS features include:
**Real-Time Notification, Crisis categorization [low, medium, high], Send and Receive Alerts, Real-time status update, Aid Traffic Control Room, Low implementation Cost, Modular and Scalable in Nature.**

**QRCS components and their roles**

QRCS consists of four mobile applications and one web application which is the server (control room). The four applications are Ground Worker App, Civilian App, Resource Head App and Asset Tracking App. All the four applications were android apps. The web application was built using node.js.

The Ground Worker App is meant for the disaster management member who is responsible for a particular area or locality. The Civilian App is for any citizen. The Resource Head App is for heads of Ambulance, Fire station, local Police, Hospitals and Public transport buses. Asset tracking App is for an ambulance, fire brigades and bus drivers for an actual location with a route on a map showing how to reach the place.

In event of any crisis situation, any person i.e. civilian or disaster management member can alert the control room using the app. If the alert is sent by a civilian then the control room notifies any disaster management member in the nearby area as to verify the situation reported.

When a disaster management member reports a disaster then the control room can verify it from other authorities or if they are convinced then the control room sets an epicenter of the crisis and marks the area of impact on its map. They generate an alert to all the members present within the area and around 1 km radius outside the marked area.

A similar alert is also generated for the resource heads, who are asked to update and specify the available resources to the control room. These resources provided by resource heads are marked as active by the control room. The disaster management member can request control room for resources (for eg: 5 buses for moving 400 people from a crisis location to a safe shelter) depending on the severity of the disaster.

The control room allocates the nearest available and active resource to the request and the asset tracking app receives an alert from the server. The asset tracking app receives the alert with destination location on the map along with the shortest path towards it. The app automatically notifies the server regarding the asset reaching a destination and the server may provide it with the new destination location. Figure 1 shows QRCS architecture.

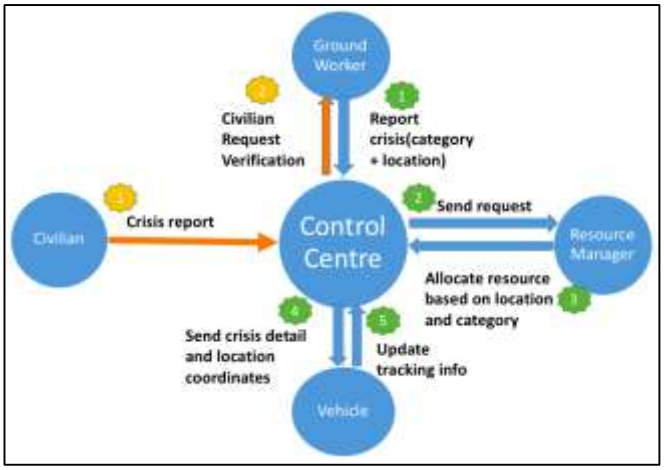


**Fig. 1: System Architecture and message flow of QRCS**

## 4. PROPOSED SYSTEM

The underlying concept of a Context Broker in QRCS is to have a separate software facility that gathers, prepares and serves context data so that a decision maker (a person or an application system) can have the benefit of this data without having to do

 

all of the work of obtaining and managing the context data as part of the application itself. It is essentially a design pattern for sourcing context data more efficiently and effectively, offloading work from the decision-making application.

The Context Broker will act as a centralized server which will collate and analyze all the information and decide on what the next course of action will be.

### 4.1 Components of the proposed system

**4.1.1 Context Broker:** A Context Broker is a centralized server which will be placed in the control room and will be the core of the system.

**4.1.2 Mobile Applications:** Mobile applications are necessary to inform the control unit in event of any crisis. This app will have the same functionalities as of QRCS – Civilian, Resource Head, Asset Tracking, and Ground Worker.

**4.1.3 Database:** The database will include a collection of records of all the people comprising the disaster management team- the control room members as well as the people responsible for providing various resources (Fire Brigade, Ambulance, and Transport Bus). The data to be stored and will include the personal details such as name, contact number, where the person works, a role he/she plays etc. It will also store information such as disaster type, location, resource allocation and all communication between various agents and the broker.

### 4.2 How the proposed system works

In event of any crisis situation, any person, civilian or disaster management member (Ground worker) can alert the control room using the app. When this alert is sent to the control room, the location is also sent using the Location Services/GPS provided by mobile phones along with the severity of crisis (high/medium/low). This will help the Context Broker identify where the crisis has occurred. In the event of a loss of Internet in the mobile app, a facility to send a text message to the control room is also present.

If the alert is sent by a civilian then the context broker will notify its ground workers (with the help of the data present in database or sourcing their location) and any worker who is relatively near the area where the crisis has been reported can verify if the reporting is true.

If it is sent by the ground worker then the context broker sets an epicenter of the crisis and marks the area of impact on its map. The context broker will then generate an alert to all the workers, resource heads, assets present within the area and around 1 km radius outside the marked area. The locations of the assets, workers are again tracked using the location tracking services in mobile phones.

The worker can request the context broker for resources. Once the crisis has been confirmed and logged in, the context broker will generate an emergency alert to be sent to all people so that they can avoid the area. The context broker then tries to analyze the severity of the crisis. Figure 2 shows the architecture of Context-Aware Middleware in QRCS and interaction among various entities.

A worker from the disaster management who is there nearby, report the severity of the crisis (LOW, MEDIUM, HIGH) (most reliable) using their Ground Worker app. If any worker is not near the crisis location or will take time to reach, track the various social networking sites such as Twitter, Facebook using the location as a keyword to try and gauge the severity (less reliable and won't always provide information). At the same time, the context broker will also generate an alert to all the resource heads who are asked to update and specify the available resources to the control room and be prepared. These resources are marked as active by the control room.

The context broker will also send an alert to all hospitals in the nearby area notifying them of the crisis and asking them to be prepared in advance in order to swiftly & efficiently provide medical aid to the injured. This can be done by first gathering a list of all hospitals present in a nearby location and then sending out alerts to each.

Once the severity and the exact nature of the crisis have become clear, the dispatching of the resources will commence. Using the services of the traffic control room, the broker will identify the most optimal route to the crisis area (based upon the approximate time taken to reach the location after considering factors like traffic) and intimate the resource heads who are closest to the crisis location with the route and how many resources to release (no of buses).

Alert the respective traffic control rooms, so as to keep the road clear and minimize the time required to reach the destination is also the responsibility of the broker. The context broker will then do a real-time tracking of the resources dispatched, to track their location, find out the time required to reach the location and notify the team present at the crisis location of the same. The context broker can also provide a new destination under certain circumstances.

The broker will also be responsible for continuous coordination with the crisis team present on location to find out if more resources are required, to get the approx. a number of injured people and inform nearby hospitals to send out ambulances for the same and prepare for their arrival.

If the crisis severity is high or if more resources are required, then the broker is responsible for coordinating with other resource heads and provide more resources as quickly as possible.

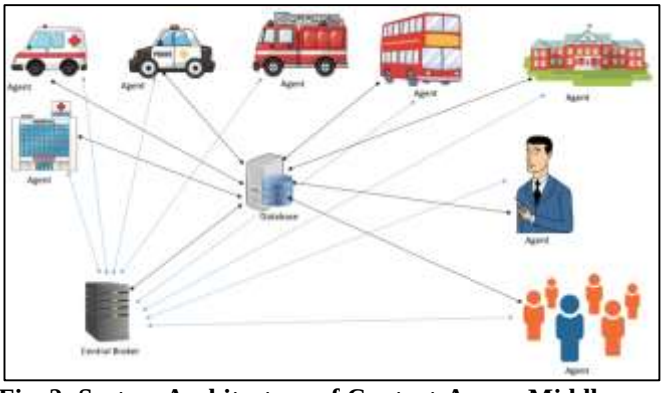

**Fig. 2: System Architecture of Context-Aware Middleware in QRCS**

**Central Broker:** The Central Broker is a Context Broker that does following activities:

i. **Resource/Asset Tracking:** Central broker is continuously tracking the location of assets such as ambulance, fire brigade, transport bus, ground workers.
ii. **Resource Requesting:** When a ground worker requests for resources, the broker checks with listed resources in the database and then contacts the respective resource heads.

 

iii. **Alert:** It alerts all nearby ground workers & resource heads during the crisis.
iv. **Info Sharing:** When a ground worker requests for contact numbers of assets and team members the broker sends the information.
v. **Agent Registration:** Broker registers all new agents.

## 5. JSON Based Communication

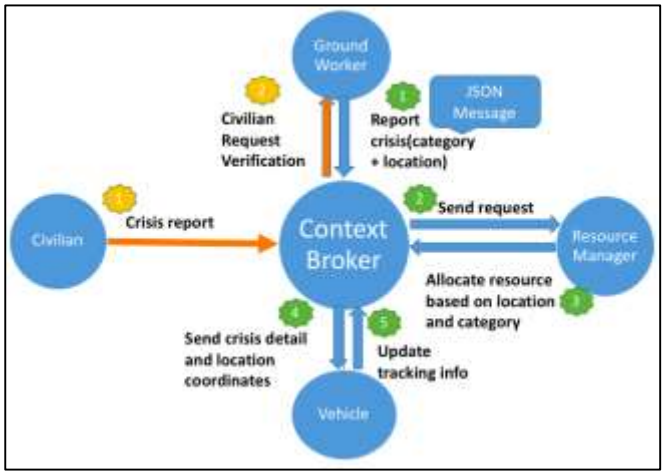


**Fig. 3: Message flow and context sharing in context-aware middleware in QRCS**

JSON (JavaScript Object Notation) is a lightweight data-interchange format. It is easy for humans to read and write. It is easy for machines to parse and generate. It is based on a subset of the JavaScript Programming Language. JSON is a text format that is completely language independent but uses conventions that are familiar to programmers of the C-family of languages, including C, C++, C#, Java, JavaScript, Perl, Python, and many others. These properties make JSON an ideal data-interchange language.

JSON will be used to create ontology rules for agents to describe contextual information and to share the context knowledge. Since QRCS includes a node.js based event driven server for handling requests and JSON for data interchange we thought to use the same. JSON becomes common data interchange for mobile and web-based agents which helps in having a uniform ontology.

First, the agents (Ground Worker, Resource Heads, Assets, and Civilian) need to register with context broker. The registration message that an agent will send is given below. Message (1) shows the registration of an asset (ambulance, fire brigade or transport bus).

Once the agents are registered, they can then send messages based on their roles. (2) Shows a Ground Worker reporting a disaster. The worker specifies disaster type, severity, and location. Then the context broker creates an alert and sends the (3) message to all Ground Workers and Resource Heads near the location of the disaster. Before sending the message, context broker will acquire the location of all agents by sending (4) message and the send (3) message to nearby agents. Figure 3 shows how the message communication will flow among the agents and context broker. Below is the JSON message structure to be used for communication among agents and context broker.

**5.1 To register an Agent: (Asset)**

Accept: application/json
Content-Type: application/json
X-Auth-Token: [TOKEN AUTHENTICATION]

```
{
 "AgentRegistration": [
  {
   "agent": [
    {
     "type": "Asset",
     "id": "FBH4452"
    }
   ],
   "attributes": [
    {
     "type": "Ambulance",
     "registration_no": "MH 02 Y7766",
     "resource_head id": "R10066"
    }
   ],
  }
 ],
 "UpdateAction": "APPEND"
}
```

**5.2 To create a request: (Alert a disaster)**

```
{
   "Disaster": [
   {
      "agent": [
       {
        "type": " Ground Worker ",
        "id": "100001"
       }
       ],
       "attributes": [
          {
           "disaster_name": "Earthquake",
           "severity": "High ",
           "location":"180'N15'W"
          }
         ]
       }
       ],
 "UpdateAction": "APPEND"
}
```

**5.3 If all went well, application will receive a response from the context broker and it will give the details of the disaster alert created to nearby agents:**

```
 {
 "Disaster": [
  {
    "attributes": [
         {
          "disaster_name": "Earthquake",
          "severity": "High ",
          "location":"180'N15'W"
         }
        ]
    },
   ],
    "agent": [
     {
      "type": " Ground Worker ",
      "id": "100001"
     }
     ],
 }
```

 

**5.4 To ask an agent to share location:**

```
{
  "Request": [
    {
      "type": "Ground Worker",
      "id":"100001"
      "request-type": "location_share",
    }
  ],
}
```

In above the Context-Broker is requesting the Ground Worker agent to share its location.

## 6. CONCLUSION

Communication, real-time information, decision making, coordination and quick emergency response are the features of an effective disaster/emergency management system. In this paper, we discussed QRCS and other disaster management systems and also propose Context-Aware Middleware in QRCS to offload decision making to a separate software. QRCS had its lacuna with dependence on control room (people belonging to disaster management) to make the decisions. This would slow the process. Hence Context Broker was proposed to ensure the decision making be independent and also the whole process to be automated. The Context-Aware Middleware along with JSON based message communication ensures quick and autonomous decision making due to the context broker's ability to source data and reason them for any contextual information which helps in triggering appropriate actions. It also helps the existing QRCS to be independent of human touch. This saves precious time during disasters. Thus Context-Aware Middleware along with other entities of the QRCS would ensure a robust, scalable and highly effective disaster management system.

## 7. REFERENCES


[1] Aekyung Moon, Hyoungsun Kim, Hyun Kim, and Soowon Lee, "Context-Aware Active Services in Ubiquitous Computing Environments."

[2] A. Ranganathan and R.H. Campbell, "A Middleware for Context-Aware Agents in Ubiquitous Computing Environments," LNCS (2672), 2003, pp. 143-161.

[3] K. Henricksen, J. Indulska, and A. Rakotonirainy, "Modeling context information in pervasive computing systems," Proceedings of the First International Conference on Pervasive Computing (LNCS 2414), August 2002.

[4] Chung-Seong Hong, Hyung-Sun Kim, Joonmyun Cho, Hyun Kyu Cho, and Hyun-Chan Lee "Context Modeling and Reasoning Approach in Context-Aware Middleware for URC System" International Journal of Mathematical, Physical and Engineering Sciences Volume1 Number 4.

[5] H. Chen, T. Finin, and A. Joshi, "An ontology for context-aware pervasive computing environments," The Knowledge Engineering Review, vol. 18, no. 3, 2003.

[6] Anjana Sara Thomas and R. Vikraman Nair, "A survey on Disaster Management Systems," International Journal of Computer Applications (0975 – 8887) Volume 128 – No.10, October 2015.

[7] A. Harter, A. Hopper, P. Steggles, A. Ward, P. Webster, "The anatomy of a Con-text-aware application", Wireless Networks Vol. 8, Issue 2/3, pp. 187-197, 2002.

[8] Sadeh, N.M., Gandon, F.L., and Kwon, O.B. "Ambient Intelligence: The MyCampus experience" Technical Report CMU-ISRI-05-123, School of Computer Science, Carnagie Mellon University 2005.

[9] Harry Chen, Tim Finin, Anupam Joshi "A Context Broker for Building Smart Meeting Rooms".

[10] Moumita Mukherjee and Jai Asundi "Considering Emergency and Disaster Management Systems from a Software Architecture Perspective" 2011.


 