



# GeoCloud4SDI: a cloud enabled open framework for development of spatial data infrastructure at city level

Ashutosh Kumar Tripathi<sup>1</sup> · Sonam Agrawal<sup>2</sup> · Rajan Dev Gupta<sup>3</sup>

Received: 25 May 2022 / Accepted: 26 October 2022 / Published online: 11 November 2022  
© The Author(s), under exclusive licence to Springer-Verlag GmbH Germany, part of Springer Nature 2022

## Abstract

Spatial Data Infrastructure (SDI) is considered as a holistic framework for effective collection, integration, discovery, sharing and delivery on a common platform for better utilization of multi-source geospatial data by global community, thereby, resulting in increased awareness of the use of geospatial data and the cooperation between decision-makers and stakeholders. Cloud computing technology can provide convenient and on-demand network access to shared computing resources that can be rapidly provisioned and released with minimal management effort or service provider interaction, and has been widely used in different applications. However, much research could not be found related to adoption of cloud computing in development of SDI at various levels from local to country. The present research is a step forward to fill this research gap by development of an open, interoperable and efficient framework for implementation of SDI at City level (*acronym* GeoCloud4SDI) for Prayagraj city (India) and deploying the same on OpenStack private cloud. Accordingly, multi-tier client server web GIS based SDI architecture and cloud enabled SDI services and workflow using load balancing and elastic computing architecture is developed. The novel framework for GeoCloud4SDI is developed using four layers, namely, (i) physical layer, (ii) cloud services layer, (iii) geospatial services layer and (iv) client layer. The physical layer and the cloud services layer combined together enables the access of virtualized computing resources from the cloud to enhance the performance of City level SDI. Geospatial services layer provides two different functionalities, namely, (a) management, retrieval and access of spatial data and metadata, and (b) retrieval of cloud services from cloud service layer. The performance of GeoCloud4SDI is examined by cloud enabled load balancing and elastic computing framework and the results show the response time reduced up to 50.43% along with 1.95 s performance gain for 1000 concurrent requests. The developed and implemented cloud enabled framework of GeoCloud4SDI is built via Open Geospatial Consortium (OGC) compliant interoperable service standards using Open Source Software (OSS), thereby ensuring standardisation and interoperability, and can also be adopted for other cities in India and the world.

**Keywords** Spatial Data Infrastructure (SDI) · Interoperable · Web GIS Architecture · Cloud Computing · OpenStack · Open Source Software (OSS)

## Introduction

In recent years, spatial information has become one of the crucial elements in evolving a decision-making environment. It is estimated that more than 80% of the information used in decision-making has the spatial component (Klinkenberg 2003). Spatial data creation has

become easy due to adoption of advanced technologies and hence many organizations are creating and maintaining their own spatial data. Spatial applications that have the diverse data requirements usually face the difficulty of creating them at one place. Such applications require the collection and integration of spatial data from different sources. The fragmentation of spatial data custodians causes diversity in policies, standards and tools to manage and share the multi-source spatial data (Sjoukema et al. 2017, 2020; Kotsev et al. 2020). Moreover, this way of multi-source spatial data creation, collection and integration lead to many technical and non-technical issues in the decision-making. Data specifications, data models,

---

Communicated by: H. Babaie

---

✉ Ashutosh Kumar Tripathi  
ashutosh.tripathi@uniteduniversity.edu.in

Extended author information available on the last page of the article

technical standards, metadata standardization, data attribution and tools are the major technical issues identified in the spatial data integration. However, non-technical issues involve diversity in pricing, incoherent sharing policies, legal disputes and social issues.

To overcome these technical and non-technical barriers, a holistic framework is required to manage and integrate the multi-source spatial data. Spatial Data Infrastructure (SDI) is considered as a holistic framework to share and integrate the multi-source spatial data on a common platform (Rajabifard 2001; Cooper et al. 2012; Maphale and Smit 2021). According to Coleman and McLaughlin (1998), SDI encompasses the policies, technologies, standards and human resources necessary for the effective collection, management, access, delivery and utilization of spatial data in a global community. SDI increases the awareness of the use of spatial data and the cooperation between decision-makers and stakeholders. SDI can be created at different hierarchy of levels from local (i.e., city) to state to national to regional to global. The city level SDI is the base level of hierarchy and can be integrated upwards with SDIs at higher levels in hierarchy. During the past several years, high adoption and utilization of SDI is reported by different countries and spatial agencies (Niaraki et al. 2010; Masó et al. 2012; Ali and Imran 2019; Ronzhin et al. 2019; Oskouei et al. 2019). The adoption of SDI at country level refers to National Spatial Data Infrastructure (NSDI). In the early 1990's, Federal Geographic Data Committee (FGDC) initiated NSDI program for the United States (Clinton 1994). During the same time, Open Geospatial Consortium (OGC) also took up spatial data sharing initiatives and presented interoperable service standards for sharing the spatial data and metadata discovery (Yang et al. 2007; Xia et al. 2015; Agrawal et al. 2022). The most commonly used OGC services are Web Map Services (WMS) for map, Web Feature Services (WFS) and Web Coverage Services (WCS) for data service, Web Processing Service (WPS) for data processing and Catalogue Service for the Web (CSW) for metadata sharing.

With the advancement of technologies in the field of Geographic Information System (GIS) and the World-Wide-Web Consortium (W3C), many countries have taken the initiative to develop a web GIS based framework as a ubiquitous part of its NSDI (Zhang and Tsou 2009). This concept of web based NSDI ensures the standardization, interoperability and sharing of spatial datasets. This provides an interactive web based interface for spatial data discovery and access through a resource search engine which is an essential component of spatial web based infrastructure (Zhang and Tsou 2009). For this purpose, the three-tier client–server architecture consisting of data layer, application layer and client layer is commonly adopted for developing web GIS based framework for applications in the field of GIS. In fact, most of web based SDI applications are built via OGC compliant services using Open Source Software (OSS) (Willmes et al. 2014; Gupta et al. 2022).

During the recent past, several organizations have developed their own SDI at different levels from local to country level. Geodata Versatile Information Transfer environment (GeoVITE) is Academia SDI for Zurich, Switzerland which provides data visualization and download for academia (Matthys and Gkonos 2017). The Global Earth Observation System of Systems (GEOSS) of U.S. facilitates the discovery and access of Earth Observation (EO) data from metadata catalogue (Liu et al. 2011). Wang and Liu (2009) presented a TeraGrid GIScience Gateways implementation from web 2.0 technology which adopts Service Oriented Architecture (SOA) for parallel and distributed data processing. The Participatory Geographic Information Systems for Transportation (PGIST) is developed for public participation in transportation and for decision-making in Washington State of U.S. (Nyerges et al. 2006). Many other web based SDI examples can be seen in different spatial research domains such as agriculture (Chen et al. 2015), disaster assessment (Mazzetti et al. 2016), land (Xing et al. 2015, 2019), urban planning (Zhang et al. 2019a) and energy (Trumpy and Manzella 2017). These initiatives in SDI all over the world show that there is a need for the development of a robust web GIS based SDI framework using OSS and OGC compliant services for efficacious implementation of SDI from local to country level in both the developed and developing countries.

A diverse and large amount of multi-source data can be made available over SDI. However, it faces the challenges of concurrent data access, computational intensity and high workload due to big data and intensive resource demand of geoscience applications (Xia et al. 2015; Gkonos et al. 2019; Jiang et al. 2019). To improve the SDI performance in terms of processing, indexing, querying and handling concurrent requests, numerous technologies have been adopted (Knoblock et al. 2017) which includes High Performance Computing (HPC) systems like Graphics Processing Unit (GPU) (Zhang et al. 2014) and Hadoop Distributed File System (HDFS) (Shvachko et al. 2010; Tripathi et al. 2018). Several spatial analysis workload balancing methods are proposed to deal with the scheduling of service requests but they require high computational resources (Huang and Li 2009; Thulasidasan et al. 2010; Huang et al. 2011). The integration and configuration of these resources are not only costly but their customization is also very difficult (Huang et al. 2018; Mokarram and Khosravi 2021). Since the upgradation and / or customization of these computing resources by these technologies have their own advantages and limitations, outsourcing of computing resources hosted on other enterprises is required so that organizations can concentrate more on data, user and computational aspects rather than on system configuration.

Cloud computing is one such technology that has the ability to provide convenient and on-demand network access

to shared computing resources (e.g., networks, servers, storage, applications and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction (Mell and Grance 2011). Several research articles related to web based spatial data application within the cloud are available (Lee and Liang 2011; Tang and Feng 2017; Huang et al. 2018; Tripathi et al. 2020; Mete and Yomralioglu 2021). Zhang et al. (2019b) provide a cloud enabled WPS framework to solve the Earth science problems, especially for Community Multi-scale Air Quality (CMAQ) model. The developed framework used GeoBrain cloud with Infrastructure as a Service (IaaS) architecture from Apache CloudStack to create virtual machines (VMs) for the CMAQ-WS. Huang et al. (2018) proposed a Hybrid Cloud Computing (HCC) platform by using Eucalyptus as a private cloud platform and Amazon EC2 as a public cloud platform to test the dust storm forecasting. Wang et al. (2016) proposed an HPC framework after adopting the OpenStack private cloud to manage and process spatial big data. Kang and Lee (2016) performed image processing by using auto-scale enabled private cloud.

The above studies demonstrate that adoption of cloud computing is a better solution for spatial applications. In fact, virtualization technique is the key to fostering cloud computing which refers to the abstraction of computing resources such as storage, network, database and memory from the end-user (Xing and Zhan 2012). Cloud has many characteristics such as on-demand service, resource pooling, rapid elasticity, pay as per service and broad network access to meet end-users' requirements (Vouk 2008; Zhang et al. 2010; Mell and Grance 2011; Yang et al. 2011). Rapid elasticity is one of the core characteristics of the cloud that enables the user to scale up or down the computing resources based on the application requirement (Herbst et al. 2013). Thus, cloud computing provides easy access to computing resources, customization of the computing resources, distributed computing environment for the large scale spatial datasets and fault-tolerant computing platform. But much research could not be found related to adoption of cloud computing in SDI applications. Further, the private cloud, a free but having many cloud enabling tools for creating private cloud environment (Sotomayor et al. 2009), has been limitedly adopted for the deployment and testing of the spatial applications (Nurmi et al. 2009; Kang and Lee 2016; Eken and Sayar 2019). The authors, to the best of their knowledge, could not track any research on the web GIS based SDI development at city level using open standards for multi-source geospatial data and related applications under open source based private cloud computing environment.

This research is a step forward to fill the aforesaid research gap by creating and deploying the SDI framework developed at city level on the private cloud OpenStack open source cloud solution. The main aim of the present research

is to develop and implement cloud based multi-tier framework for city level SDI for Prayagraj city (acronym: GeoCloud4SDI). Accordingly, a web GIS based novel framework for an open and efficient GeoCloud4SDI is developed and implemented on the private but open source cloud through the adoption of multi-tier architecture and cloud enabled SDI services and workflow using load balancing and elastic computing architecture for spatial data discovery, access and processing.

The paper starts with a general introduction to the research background. This background helps in finding the research gaps and in framing the research objectives of the present research. The Methods section gives details of the study area, data used and details of geospatial database used in the development of SDI. The architecture of the GeoCloud4SDI is then discussed. The third section is Results and Discussion that includes the implementation, performance and functionalities of the GeoCloud4SDI. The Conclusion section presents the salient research outcomes.

## Methods

### Study area

Prayagraj city (*formally* Allahabad city) is located in the north-central region of Uttar Pradesh, India, and is a participatory of Smart City Mission of the Ministry of Urban Development (MoUD), Government of India. The city is aspiring to become a smart city in its true sense to achieve sustainable infrastructure and economic growth through inclusive development. The geographical extent of Prayagraj city is 25°23'08" N to 25°32'11" N latitude and 81°43'29" E to 81°54'11" E longitude, covering an area of 76.29 km<sup>2</sup>. The total population of the city is 1,112,544 and the city is divided into 80 wards as per Prayagraj Municipal Corporation. The map of Prayagraj city with ward boundaries is shown in Fig. 2.

### Data used

The data used include: (a) Topographic maps- Open Series Map (OSM) number- G44P11, G44P14 and G44P15 at 1:50,000 scale with World Geodetic System-1984 (WGS-84) datum and Universal Transverse Mercator (UTM) projection; (b) Ward map of Prayagraj city; (c) Census data; (d) Satellite imagery of 04–04–2019 of Sentinel-2A at 10 m spatial resolution; (e) Land Use and Land Cover (LULC) map of Prayagraj city created using Sentinel-2A data; (f) Infrastructural facility data generated using Trimble Juno 3B handheld GPS based field survey with an accuracy level of 3–5 m. The data from (a) to (d) is public. The data (e) and (f) is private and generated specifically for this research but is

also made available to the public/ decision makers for better planning of Prayagraj district.

### Geospatial database development for SDI

The applicability of any SDI depends on the accuracy and comprehensiveness of the geospatial database created for this purpose. Accordingly, a clean, accurate and comprehensive geospatial database using a common spatial framework is created for GeoCloud4SDI using the methodology

as shown in Fig. 1. The spatial and non-spatial databases are first created, followed by their proper integration through the use of common code/ identifier. The ward boundary map of Prayagraj city along with the cantonment area thus prepared is shown in Fig. 2. The datum used is WGS-84 and the projection is UTM-44 N. This map is used as base map for creation of various other thematic maps of the study area.

LULC map of Prayagraj city is generated using Sentinel-2A imagery of 10 m spatial resolution of April 04, 2019. For this purpose, Sentinel-2A band 2 (458–523 nm -blue band),

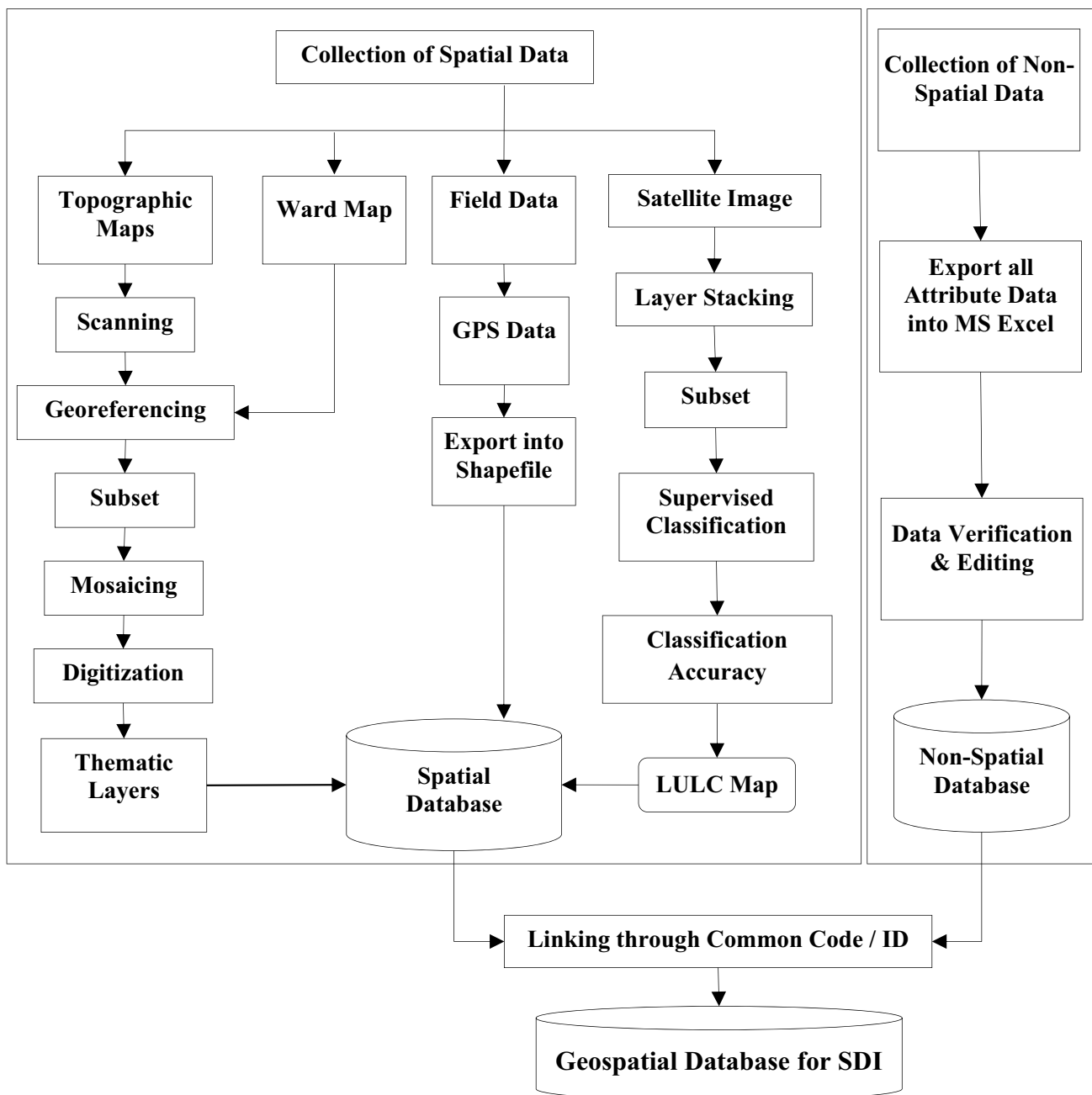
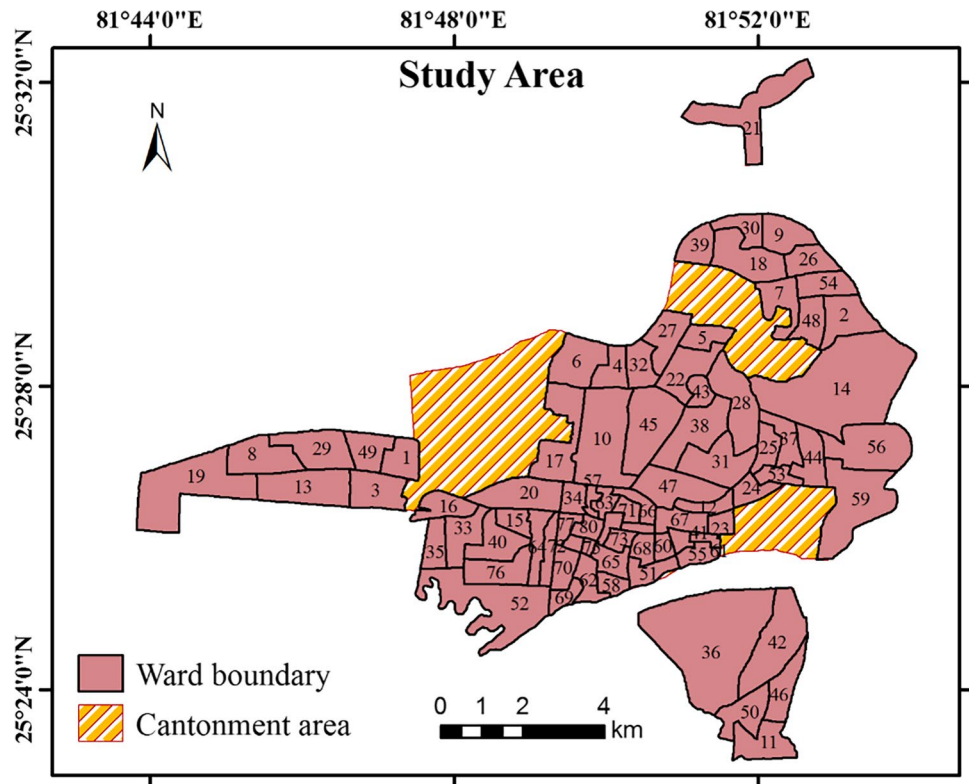


Fig. 1 Methodology adopted for geospatial database development for SDI

**Fig. 2** Prayagraj City Map with ward boundaries



band 3 (543–578 nm -green band), band 4 (650–680 nm -red band) and band 8 (785–899 nm -NIR band) are layer stacked using Erdas Imagine software. After layer stacking, the imagery of Prayagraj city was extracted using Clip operation. The image classification is then performed to classify the image in various LULC classes, namely, built up, vegetation, open land, sand, transportation and river using Maximum Likelihood Classifier (MLC), the most commonly used conventional classifier for supervised classification. The overall classification accuracy achieved is 89.06% with kappa coefficient being 87.08 which puts high confidence level in the classification. The classified LULC map of Prayagraj city is shown in Fig. 3.

Thematic maps of infrastructural facilities and places of tourist interest are then created and are shown in Fig. 4 (a) to (h). These thematic are (a) Hospital map [Fig. 4(a)], (b) Police station map [Fig. 4(b)], (c) Fire station map [Fig. 4(c)], (d) Religious place [Fig. 4(d)], (e) Public buildings [Fig. 4(e)], (f) Tourist place [Fig. 4(f)], (g) Road network [Fig. 4(g)] and (h) Railway network [Fig. 4(h)]. All these thematic maps are represented as separate thematic layers in geospatial database created for SDI development and implementation.

**Architecture of cloud enabled SDI (GeoCloud4SDI)**

The framework of high performance cloud enabled SDI, conceptualised and developed to addresses the spatial data

accessing and processing, is shown in Fig. 5. The spiral system life cycle model is used for the development of GeoCloud4SDI. It starts with the requirement specification, followed by analysis, design, implementation and testing. In the first phase, the geospatial database for SDI is created. In the second phase, it is uploaded on the GIS server. Then, OGC compliant web services for GeoCloud4SDI are added. In the final phase, GeoCloud4SDI is deployed on the OpenStack cloud. The developed GeoCloud4SDI can now be accessed anywhere in the world by any user having internet connection.

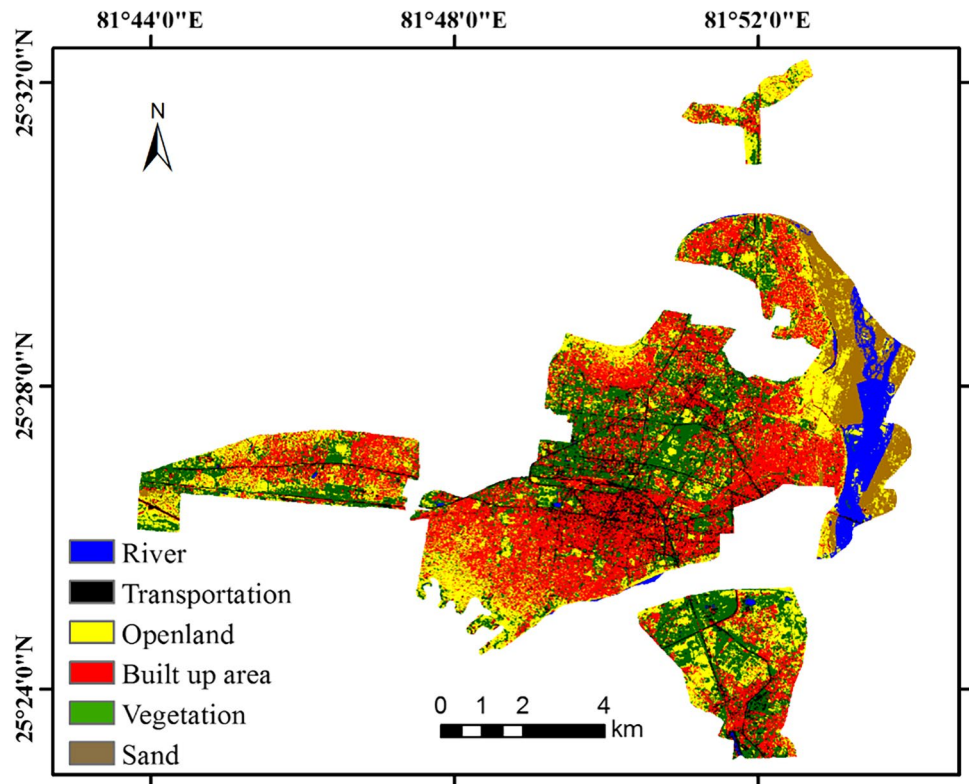
GeoCloud4SDI is based on the concept of multi-tier architecture that includes four layers, namely, (i) physical layer, (ii) cloud services layer, (iii) geospatial services layer and (iv) client layer. The physical layer and the cloud services layer combined together can also be termed as Cloud Virtual Layer. The cloud virtual layer enables the access of virtualized computing resources from the cloud. These virtualized resources are utilized to enhance the performance of any scientific application including development and implementation of City level SDI.

**Physical layer**

This layer enables the implementation of cloud infrastructure to build a data center as per the need of SDI under development using OpenStack to create IaaS environment to



**Fig. 3** LULC map of Prayagraj city (04–04–2019)



manage the network, storage and VMs. OpenStack provides VMs to users through the virtualization technique. Several types of VMs hypervisors are available to support server virtualization in the OpenStack cloud environment like Xen hypervisor (Barham et al. 2003), Kernel-based VM (KVM) (Kivity et al. 2007) and VMWare (Ali et al. 2012). Hypervisor running on this architecture is managed by KVM hypervisor. The management of these cloud services is done by web based dashboard in this study. A detailed description of the core components of the OpenStack is shown in Table 1 which are used to manage the cloud resources.

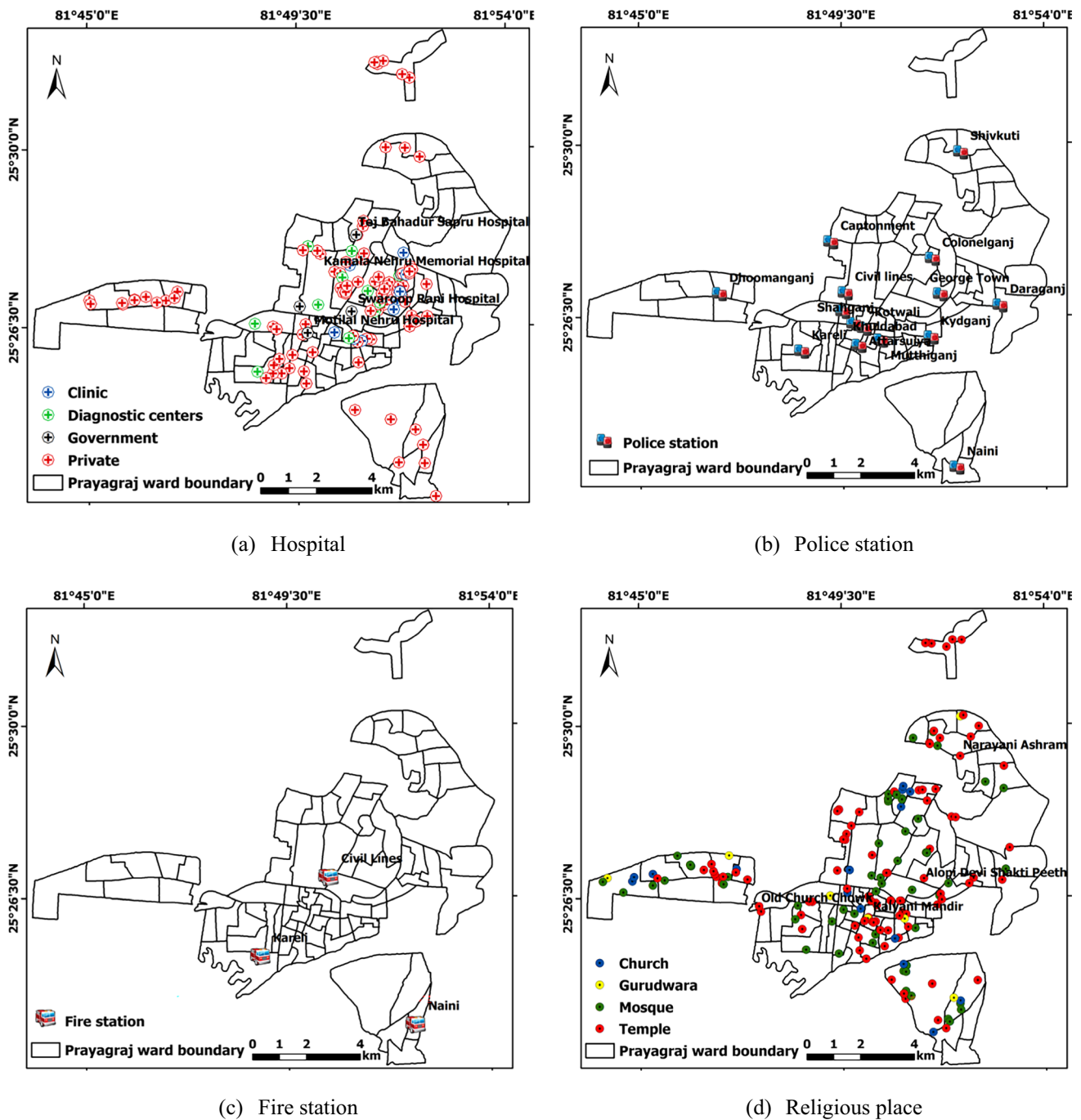
After installing and configuring the OpenStack, the setup of new VM, volume and network protocol is done. First, a new VM instance is launched using QCOW2 image format of Ubuntu 16.04 server. This new instance is then loaded and booted in cloud, and considered as a master node. This master node is used to setup the computation infrastructure for the SDI. Now, OpenStack volume for storing multi-source spatial data is created. It stores both remote sensing and facility data into geospatial database. This setup is done separately from the master node due to inherent advantages like ability to restore the data from volume when failure occurs on master node as well as to extend the volume size from present size of 1 GB to an acceptable size based on the system specifications as the data increases. Therefore, the present setup procedure makes the system fault-tolerant and also resolves the problem of storage.

Next, the setup of network configuration and firewall is performed which allows instances to communicate with each other and external network. Through this setup, developer gets the ability to have full control on the system. This setting is applied to the instances by defining the access and security rules. Further, SSH (Secure Shell) protocol, a remote access method to communicate with other networks on the range, is used to get permission to access the master instance externally. After SSH connection, login to the master node is done to setup the environment to host the SDI application.

First, the data directory is mounted to the created volume which includes all the input data (i.e., multi-source spatial data) and output data (i.e. processed data) for the proposed SDI. Next, the software environment is built to create the web based geospatial application by setting different OSS such as GeoServer (Huang and Xu 2011), GeoNetwork (Ticheler and Hielkema 2007), Apache Tomcat (Vukotic and Goodwill 2011) and Postgres/PostGIS. Apache Tomcat is used as a web server and acts as a gateway for the user to interact with the application environment. GeoServer acts as a GIS server where all the spatial data layers are published. PostgreSQL/PostGIS database is used to store all created multi-source spatial layers.

### Cloud service layer

This layer controls the access to computing resources from the physical layer and acts as a middleware between



**Fig. 4** Thematic maps of Prayagraj city showing (a) Hospital map, (b) Police station map, (c) Fire station map, (d) Religious place, (e) Public buildings, (f) Tourist place, (g) Road network and (h) Railway network

the physical and geospatial service layer to customize the computing environment for different spatial applications. The user performs resource queries to search and access the cloud resources such as VM, VM images and networking services for VM which are provisioned through OpenStack dynamically by IaaS service. For this purpose, the snapshot of master instance is created. This snapshot

presents as an OpenStack image to launch the other SDI instances as a computing node without worrying about setting up the computing environment again. These computing nodes directly share the volume data in local storage by mounting operation.

After this, the computing resources are utilized and operated for load balancing, auto-scaling and cluster

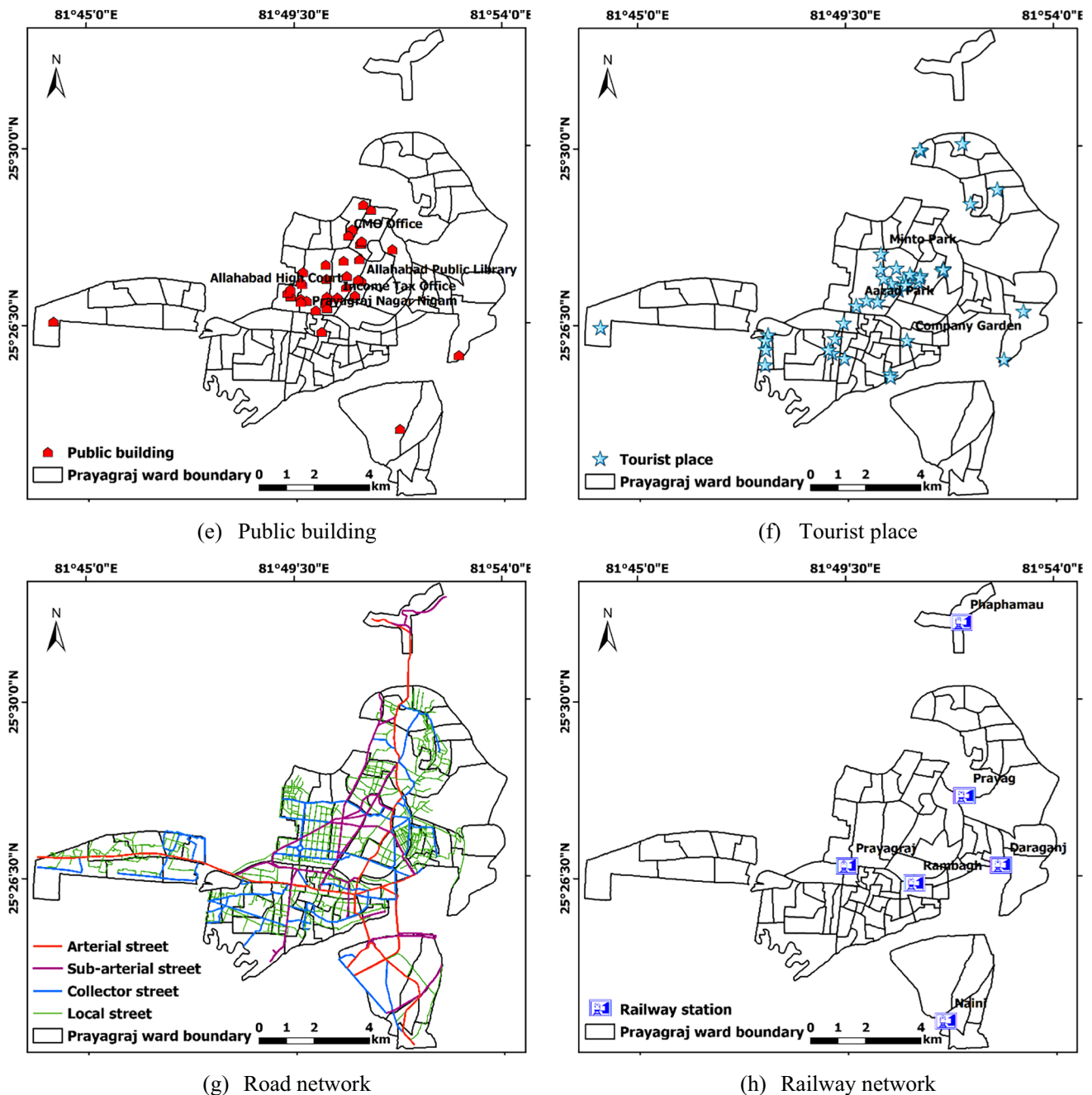


Fig. 4 (continued)

monitoring. Workload balancing involves the distribution of load into multiple computing resources. Auto-scaling is achieved through elastic computing, a fundamental feature of cloud environment. It scales the VMs based upon a predefined threshold. The current status of computing resources is monitored through cluster monitoring functionality. Such a system provides high computational environment for system and data recovery as well as also reduces the duplication of effort to maintain the system for other users or organizations.

### Geospatial services layer

This layer provides two different functionalities, namely, (a) the management, retrieval and access of spatial data and metadata, and (b) the retrieval of cloud services from cloud service layer. GeoServer and GeoNetwork are used for geospatial data and metadata management respectively. Here, both tools provide the OGC compliant services for data and metadata management. WMS and WFS services



enable interoperable data management and CSW enables catalogue services for metadata management. WPS service forwards the request for data processing to end-user.

### Client layer

This is the topmost layer of cloud enabled SDI framework and acts as an interface between users and services. In this layer, SDI application is developed through which client can make request for different services. For the development of SDI application, different programming language tools and packages like HyperText Markup Language (HTML), Java Server Pages (JSP), Cascading Style Sheets (CSS), JavaScript and OpenLayers are used. Clients can make different spatial requests such as access, discovery, processing and visualization from this layer either from desktop or mobile devices. These requests are then forwarded to the geospatial service layer. That responds with the corresponding result to the client using cloud services.

## Results and discussion

### Implementation of GeoCloud4SDI

To build the cloud environment, a high end computer system with processor of Intel(R) Core(TM) i7-2600 CPU, memory 16 GB and storage of 500 GB is used while the system configuration and specification of OpenStack are presented in Table 2.

After installing and configuring the OpenStack, a prototype model in the cloud is developed. This need to access and manage the VMs, install software, create and maintain spatial database and develop the GeoCloud4SDI geoportal. The steps of deployment of GeoCloud4SDI are done according to the proposed methodology. The basic workflow of the existing cloud enabled SDI framework includes two domains which are User domain and Cloud domain (Fig. 6).

User domain acts as Software as a Service (SaaS) to deliver a different type of infrastructure facilities information to the end-user by utilising different OGC services. Users can use these facilities for different processing purposes such as visualization, querying, access and download. Cloud domain hosts the smart facility services into the cloud. Request of processing services is accomplished from this domain through the IaaS. Resource manager has the responsibility to identify the resource requirement and allocates them for fast processing speed. The Cloud domain has two major components. First one is Cloud Load Balancer for handling large number of user requests and second one is Elasticity Manager to manage automatic provisioning of resources through scaling up or scaling down. The Load balancing and Auto-scaling utilized for SDI instances are

hosted in OpenStack cloud region. These components communicate with each other to provide distributed environment and to monitor the user workload.

### User domain services

The user domain provides the services of visualization, query and download. Advanced mapping tools and visualization techniques hosted on the cloud help users to customize their applications. Several studies integrated the mapping tools and services for spatial visualization. For example, Barik et al. (2019) developed a prototype model to compute and visualize the spatial data using the Q-GIS cloud plugin interface. Huang et al. (2018) developed a HCC platform to process the dust storm data and visualize it through GeoViz. Li et al. (2017) developed a web based visual analytic system integrated with the cloud to analyse and visualize the spatial model results. Fujioka et al. (2012) used GeoServer as a mapping service interface to visualize the marine life census result, which is maintained over the cloud. In this research, visualization tool will help to access are necessary facility datasets of Prayagraj city.

In GeoCloud4SDI, the user domain functions will respond directly to the user requests. Apache Tomcat server will receive the request and send it to GeoServer. GeoServer retrieves the data from Postgres/PostGIS database and returns the result to the user. Visualization service is provided on the GeoCloud4SDI through WMS which dynamically generates the spatially referenced image maps with the help of OpenLayers and JSP coding. Query service is also added to the GeoCloud4SDI in which the user can apply a condition on the data and view the filtered result based on this condition. Spatial query is performed through Common Query Language (CQL). The results of query is returned to the client in the form of WMS which is rendered on the client screen. Download functionality helps users to download the data and to use it in other applications or share it with users.

### Cloud domain services

**Cloud load balancer** In the cloud environment, several VM instances are created to respond to multiple incoming requests. The distribution of these requests to multiple instances increases the performance of the system. Cloud load balancer is an essential functionality of cloud computing that distributes user requests to multiple instances. Cloud load balancer used in the present study in OpenStack is described in Fig. 7. The benefits of a load balancer are to handle large number of requests, reduce the response time, improve the data access time and achieve fault-tolerant environment.

In this architecture, different instances are called from the cloud service provider end. Resource manager has the

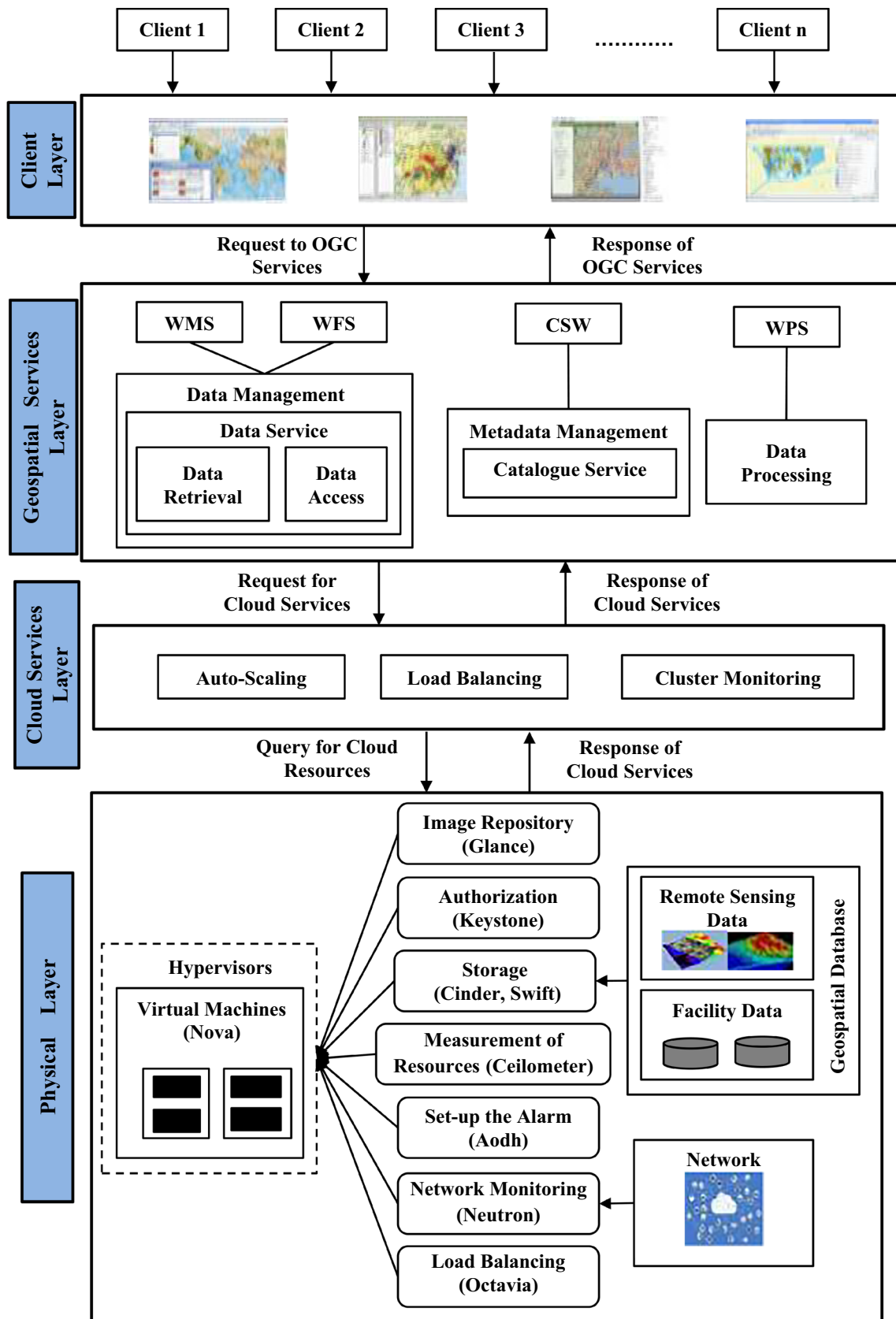


Fig. 5 Four-tier architecture developed for cloud enabled SDI

responsibility to create the VMs. For this purpose, resource manager first checks the availability of the resources based on storage capacity, images of operating systems, previously created VMs and private IPs. Then, GeoCloud4SDI instances are created as per the available resources. Now, public IP is assigned to the instances so that it can communicate with the external network. Network access policy is used to provide communication protocols like TCP, ICMP, HTTP, HTTPS, SSH to the instances. All these instances are referred to as a single access point for the GeoCloud4SDI and are managed by resource manager. These instances are called by the same IP address. Therefore, same public IP is assigned to all the instances. This public IP is the address of that system on which all the instances are running. The primary task of the resource monitor is to periodically check the current status of the instances like CPU, memory utilization and mean response time. When user requests for the OGC services, resource monitor forwards this task to the scheduler that laid it into the queue. In the ready state, these requests are distributed to the GeoCloud4SDI instances based on its current status, i.e. to those with a minimum workload.

**Cloud elastic service** Cloud elastic service provides the functionality of auto-scaling of cloud resources (Fig. 8). It helps to scale the system as per the user requirements. Elastic service is mainly utilized to perform two important tasks. The first task includes the creation of instances based upon the available computing resources and application requirements. This resource assignment is done through automatic provisioning. Another task is to automatically increase or decrease the number of instances to perform the load balancing. Elasticity monitor continuously records the status of the running instances based on the load on the server. If the alarm is triggered by the Aodh and Ceilometer component of the OpenStack, then elasticity monitor communicates with load balancer and resource manager to control the overhead on the server by provisioning or releasing the computing resources. The elasticity scheduler performs this task of resource provisioning using predefined rules of auto-scaling.

### Performance of GeoCloud4SDI

The OpenStack cloud instance is configured by setting instance type as medium, CPU cores as 1 and memory as 2 GB. Apache JMeter is used to measure the experiment results in terms of average response time. For checking the performance of developed GeoCloud4SDI, following two evaluation tests are then conducted on spatial overlay operation in GeoCloud4SDI:

- 1) Tests of resource pooling and load balancing
- 2) Testing of dynamic workload balancing

### Testing of resource pooling and load balancing

The efficacy of developed cloud enabled SDI framework, GeoCloud4SDI, is tested for improvement in the overall performance of the system in terms of computation and concurrency by resource pooling and load balancing. This evaluation is performed by launching different set of GeoCloud4SDI instances (1, 2 and 3 instances) from cloud resource pool. This set of instances is managed by load balancer that provides a single access point for multiple instances in the network to perform load balancing. The results of load balancing for GeoCloud4SDI are illustrated in Fig. 9.

The results indicate that average response time is reduced by applying resource pooling and load balancing. Moreover, as the number of instances increases, better performance is measured which can be judged from the fact that average response time for a single instance, two instances and three instances are 3.5 s, 2.34 s and 1.73 s respectively in response to 1000 concurrent requests. It is also seen that average response time of system is not constantly proportional to the number of computing resources. For example, the percentage change in performance after increasing the instances from one to two is 33% while after increasing the instances from two to three, it is only 26%.

### Testing of dynamic workload balancing

This performance evaluation shows how the implementation of GeoCloud4SDI is benefited from load balancing. This test improves the performance of the system through the elasticity characteristic of cloud. It is done by defining auto-scaling rule in terms of resource pooling and CPU utilization parameters. Two scenarios have been presented for dynamic workload balancing which are: (i) Increasing workload and (ii) Spiking workload.

To setup an auto-scaling experiment in OpenStack cloud, following rules are applied:

1. Min = 1 (i.e. one instance runs all time)
2. Max = 3 (i.e. a maximum of 3 instances can be created)
3. Min = 1 (i.e. a minimum of 1 instance should be running)
4. SD\_Threshold = 20% (i.e. if the CPU Load is below 20% in a 30 s interval, then Scale\_down() function executes that decreases the number of instances by one)
5. SU\_Threshold = 50% (i.e. if the CPU Load is above 50% in a 30 s interval, then Scale\_up() function executes that increases instance by one)

The performance measurement results for increasing workload scenario in auto-scaling are presented in Fig. 10 in which load balancer invokes the computing resources based on the above defined auto-scaling rules. It can be observed that at the starting time-interval, single instance

**Table 1** Description of relevant OpenStack cloud components used

Component	Component Function	Definition
Nova	Compute	VMs management and provisioning
Glance	Image service	Catalogue of images to create VMs
Swift	Object storage	Storage and retrieval of files and data
Neutron	Networking service	Network as a service
Cinder	Block storage service	Manages block storage volumes for VMs
Keystone	Identity service	Authorization and authentication of services
Ceilometer	Monitoring service	Measurements of cloud resources
Aodh	Alarm service	Setup the alarm parameter for the OpenStack resources
Octavia	Load balancing service	Load balancing as a service

is sufficient to respond to 10–350 concurrent users for the scaling instances pattern. Here, performance of single instance for scaling pattern is much similar to one instance performance result. When the number of concurrent users reaches more than 350, the next instance is launched because the CPU workload reaches above 50%. Now, two instances work together and the CPU workload reduces to below 50%. When the concurrent users reach above 750, the new instance is launched to balance the load and the performance pattern matches the result of three instances of load balancer. The result shows that performance of auto-scaling enabled GeoCloud4SDI is better than the load balancer enabled GeoCloud4SDI because it provides facility to save the cost of cloud resources while in load balancing, three instances work together even for lighter workloads.

The performance evaluation for spiking workload scenario in auto-scaling is presented in Fig. 11. In this test, 12 min timeline is taken within which the spiking workload performance is measured. It is observed that in the beginning, 30 concurrent users start accessing the system. When the number of users abruptly increases from 30 to 1000, the corresponding average response time grows from 21 ms to 3.42 s. Subsequently, scaling up event gets triggered. Therefore, one more instance is launched that takes approximately 90 s time to boot. Once this instance is launched, the average response time drops to 2.24 s from 3.59 s. The load of the system is still above 50%,

**Table 2** System specifications for building OpenStack cloud environment

Components	Specification
Server Operating system	Ubuntu Desktop 64 bit and 16 GB RAM
OpenStack cloud version	OpenStack Queens
Database	MariaDB
Web server	Apache
Web framework	Django

even while two instances are working together. Therefore, another instance is launched as per the auto-scaling rule number 5. Now the average response time further drops to 1.64 s from 2.08 s. Afterward, when the requests decrease to 30, scaling down event is triggered and each running instance is released in about 60 s time-interval.

### Functionalities of GeoCloud4SDI

GeoCloud4SDI is designed and developed to provide geospatial data to users. The developed interface provides several spatial functionalities to the end-users that they can directly apply onto the data. GeoCloud4SDI provides Visualization, Querying, Processing, Metadata Catalogue and Application Specific Module functionalities to the planners and decision makers. The functionalities of GeoCloud4SDI: Cloud based City level SDI developed and implemented for Prayagraj city are highlighted under subsequent heads.

### User authentication

The users are authenticated so that only an authorized person can access the data. Before accessing the data and services, the user has to first register into the GeoCloud4SDI and sign in. For the registration, users will have to fill various details like user name, email, user id, password, contact, state and pincode. The custom validation of each detail is attached to the registration so that the necessary checks can be performed.

PostgreSQL database has been used to prepare the user registration and authentication database. The user is tracked at two levels through the creation of two database tables, viz., *user\_registration* and *user\_schedule*. The *user\_registration* database includes the registration details of the users while the *user\_schedule* database includes the login access time, date and session log. When the user authentication triggers, a user session is established with an instance of a database server and the user gets the file access permission

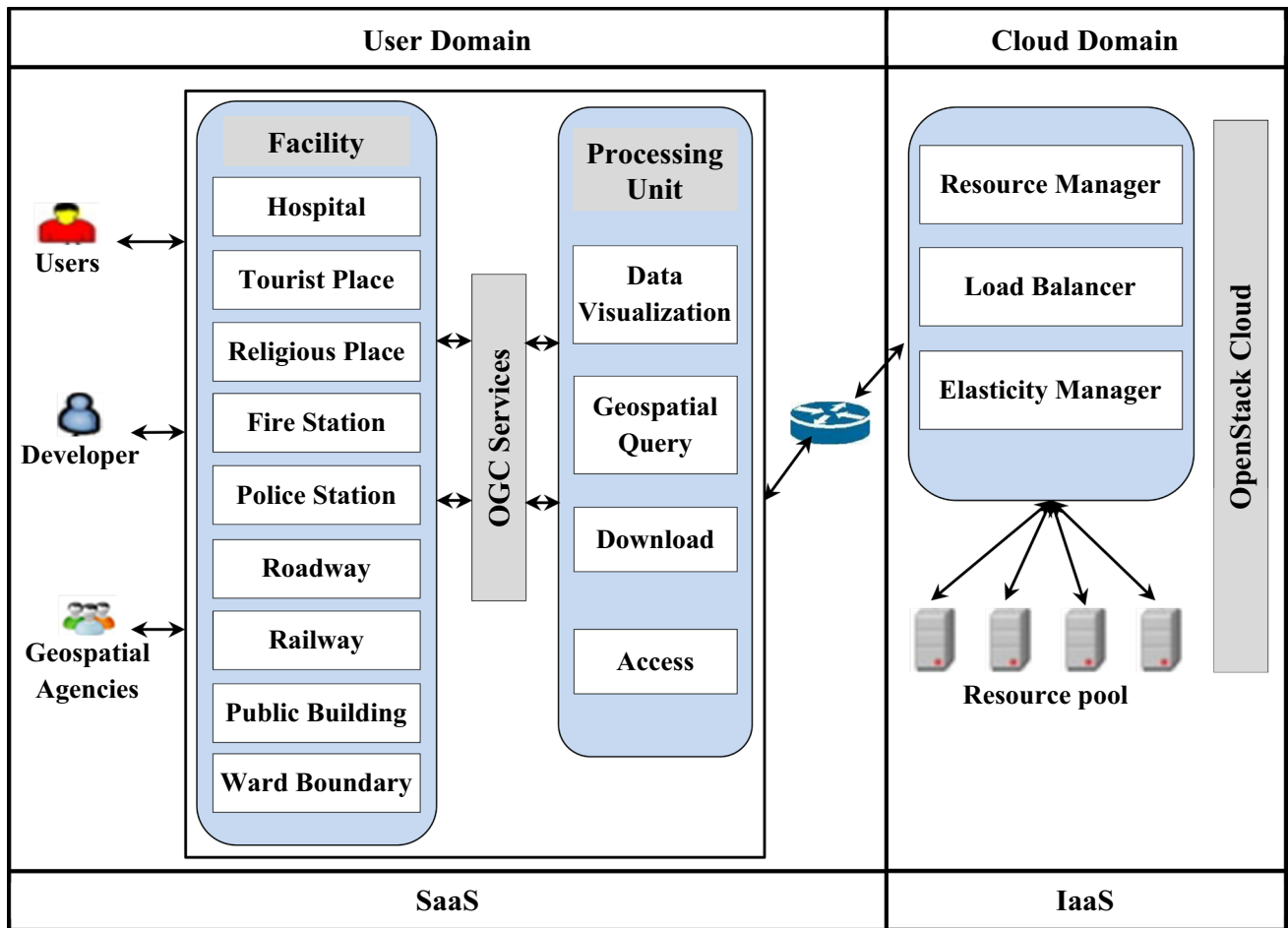
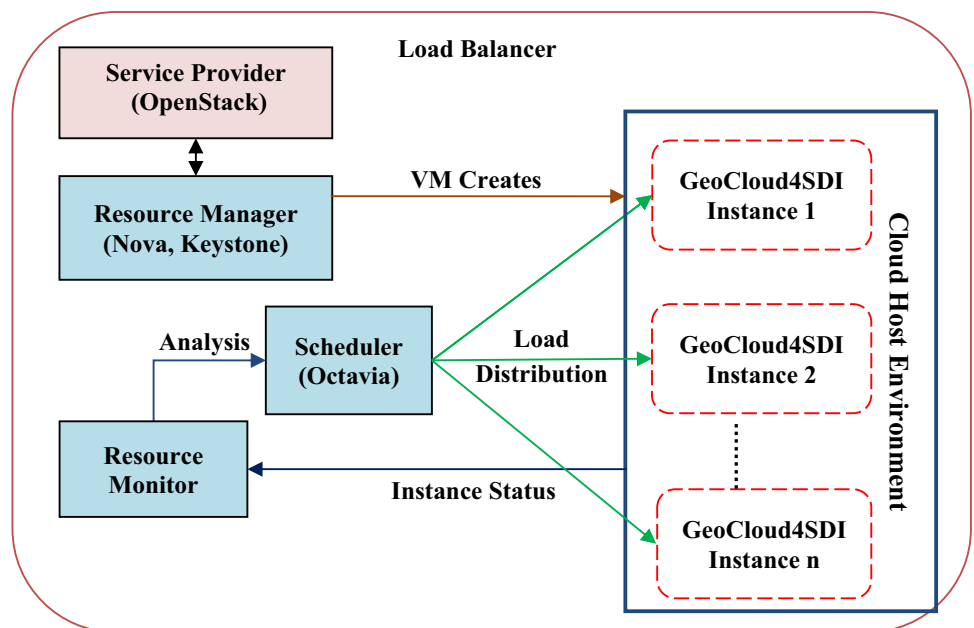


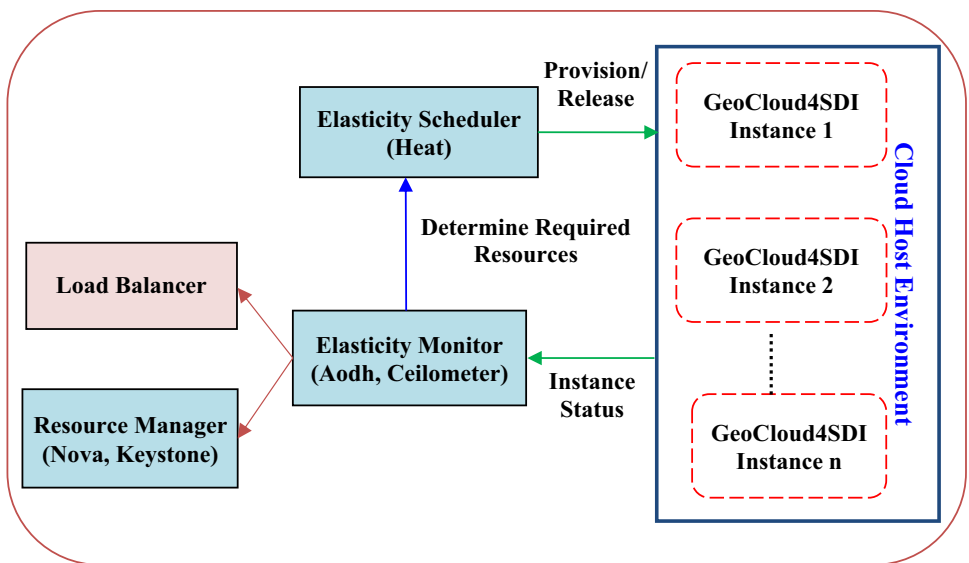
Fig. 6 Cloud enabled SDI Services and Workflow

Fig. 7 OpenStack cloud load balancer architecture

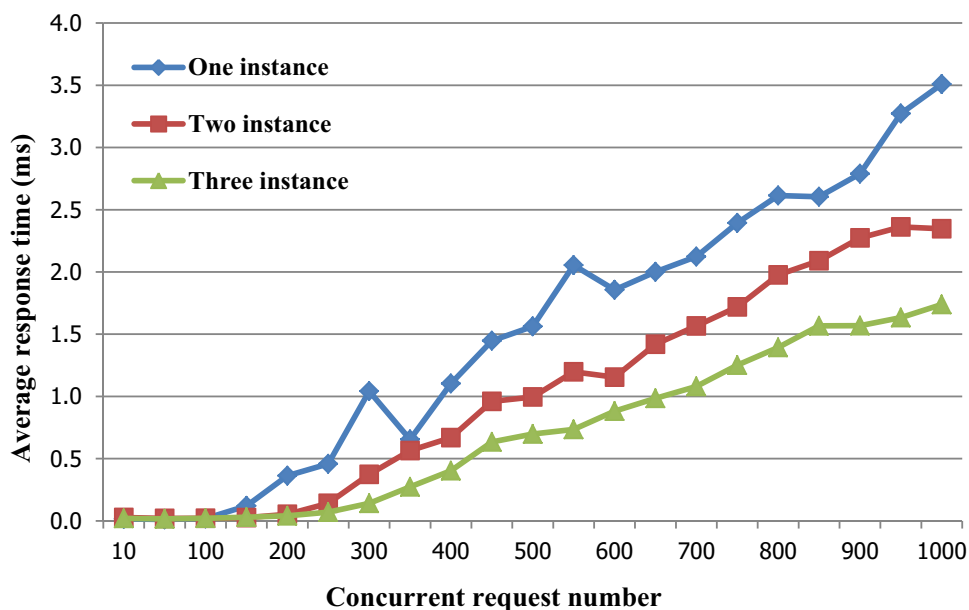




**Fig. 8** OpenStack elastic service architecture



**Fig. 9** Load balancing in Geo-Cloud4SDI



which is called as a rollback security. After the successful login, user can access the different facility data and also perform different querying operations.

**Visualization**

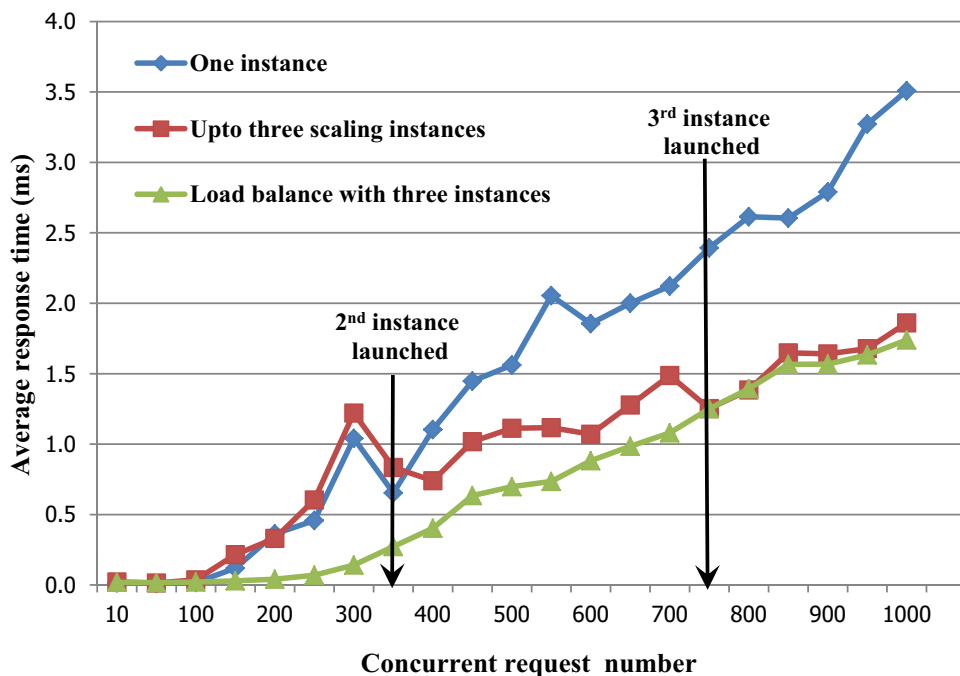
Users can view the different thematic maps of various facilities and their information from the GeoCloud4SDI. Users can select the thematic layers by clicking the visibility checkboxes. Here, the request for data extraction is made by WMS GetMap request where the service URL, server type, name of layer and tiled are set. When user clicks on a layer, at that instance, a GetMap request has been sent to the GeoServer and the corresponding data is displayed on the

screen. Here, users can set the visibility and opacity of the displayed data on the map window according to their needs. They can also perform the zoom in and zoom out operation to set the scale of data. One such visualization instance of GeoCloud4SDI interface is shown in Fig. 12.

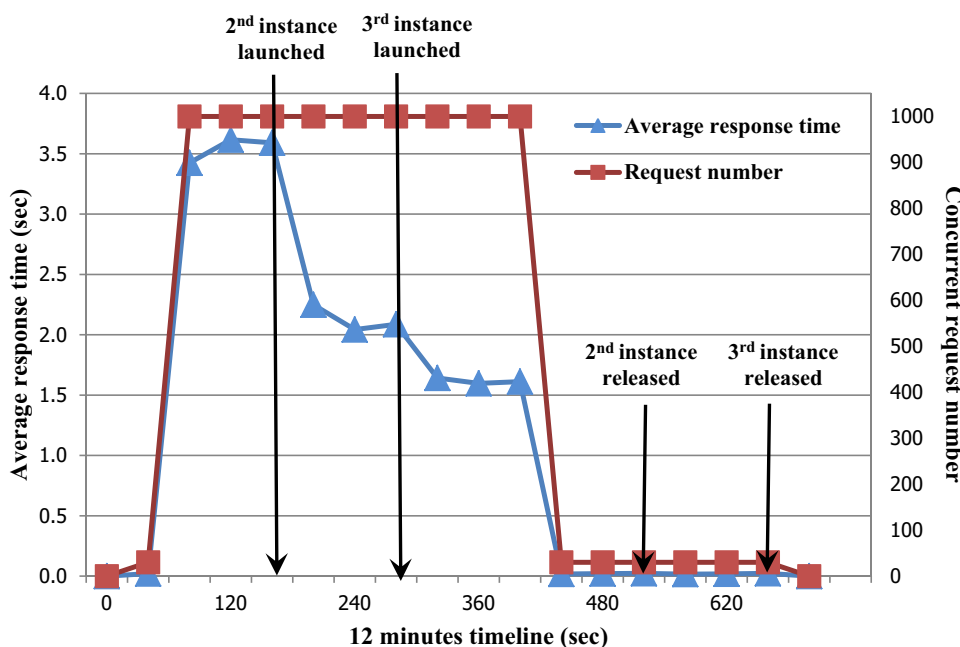
**Spatial querying**

Spatial query is the process of selecting specific features from a layer and is used for data retrieval by the users. The users can access the required area or location of a feature by its attribute. User can navigate to Querying Interface by clicking ‘Geospatial query’ link. In order to create the web based spatial querying interface, spatial query language

**Fig. 10** Auto Scaling results for increasing workload in GeoCloud4SDI



**Fig. 11** Auto Scaling results for spiking workload in GeoCloud-4SDI

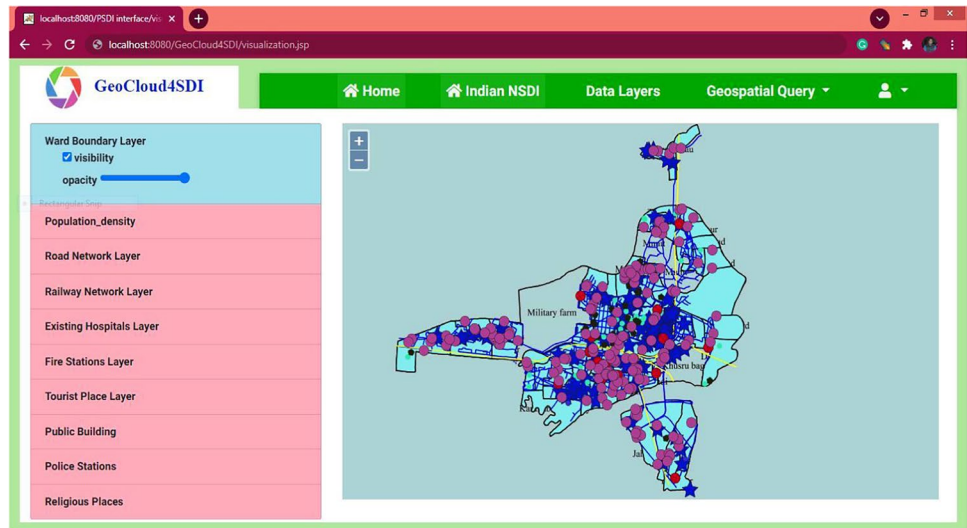


known as Common Query Language (CQL) is used which is created by OGC for catalogue web services specification. It simply uses text based syntax for filtering rather than XML based encoding. It uses attributes of the data for making spatial querying. In WMS, CQL filter is added with the GetMap request for querying, while in WFS, it is added with the Get-Feature request. In this research, two different spatial query operations are implemented: attribute value based query and condition based spatial query operation.

**Attribute based query** This is a simple query operation that includes the querying of the data based on their attribute value. For example, user wants to access the hospitals by their service type. This is implemented by performing CQL filtering operation on hospital data. The result of the hospital type wise spatial filtering for diagnostic center and its rendering on the map is shown in Fig. 13.

Here, user first selects the type of hospital using the drop-down menu on the left side of the panel and then

**Fig. 12** Visualization on Geo-Cloud4SDI



the corresponding result is displayed in the right panel of the interface. Two features are displayed on the map, first the existing hospital layer with lower visibility range and second hospital requested by user. Users can also clear the selected wards using 'Clear' button and thus performed another query.

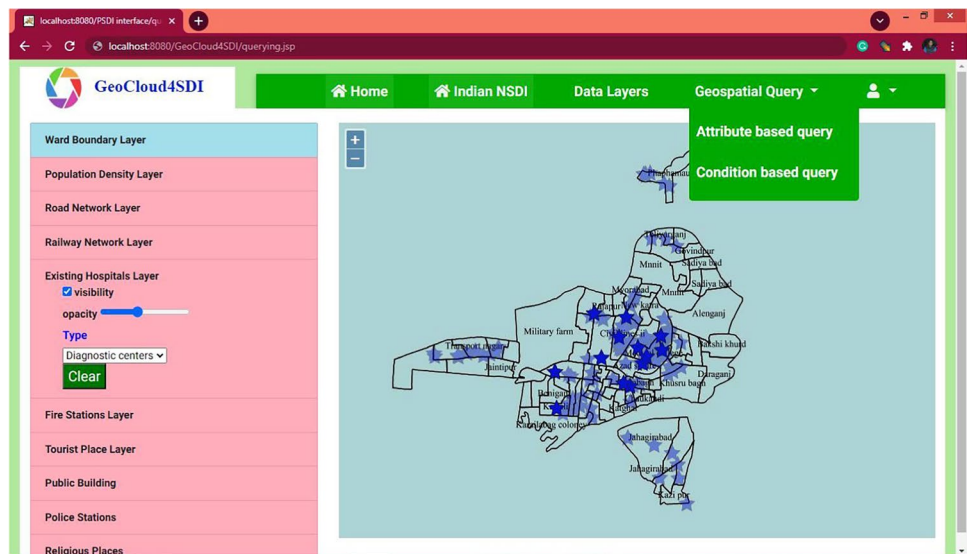
**Condition based spatial query** It is an extension of attribute value based query by adding a condition operator for feature selection. For example, user wants to access only those wards which have more than 15,000 populations. In this case, condition based CQL filtering operation is used. The result of population-wise ward filtering and its rendering on the map is shown in Fig. 14.

Here, textbox is used to give input for population and drop-down menu is used to select the operator. After giving input, 'Submit' button is pressed, which executes the request and displays the corresponding result. The result displays only those wards which are having more than 15,000 populations.

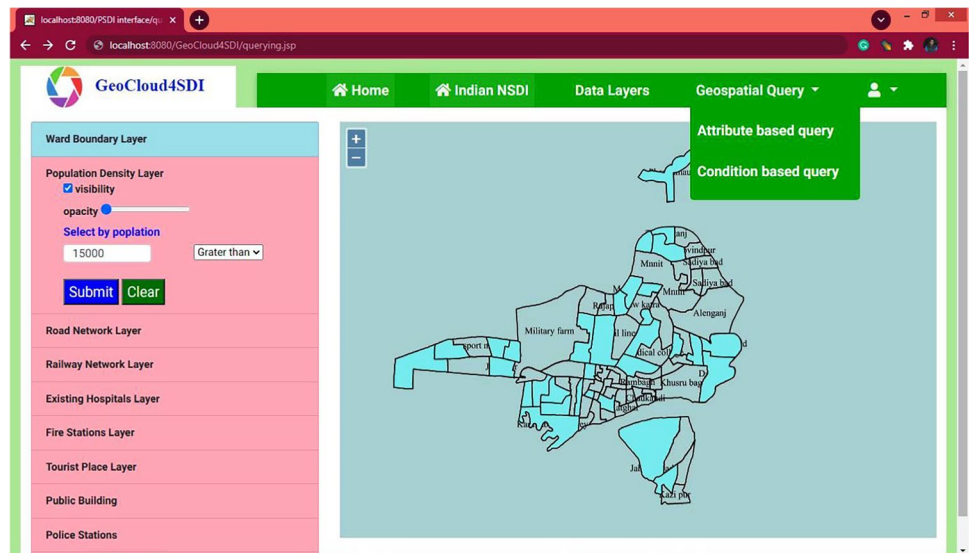
### Conclusions

SDI is an effective solution for discovery, access and utilization of geospatial data to ensure its availability to the public, reduction in duplication, improvement in the quality and reduction in cost. In the present research, GeoCloud-4SDI, an interoperable and efficient SDI at city level, is developed and implemented on open source private cloud platform through the adoption of client server web GIS

**Fig. 13** Attribute based query



**Fig. 14** Condition based spatial query



architecture and cloud enabled SDI services through the adoption of load balancing and elastic computing architecture for spatial data discovery, access and processing. The high performance cloud enabled SDI framework is conceptualised and developed based on the concept of multi-tier architecture that includes four layers, namely, physical layer, cloud services layer, geospatial services layer and client layer. GeoCloud4SDI provides an interactive web GIS based interface for geospatial data discovery and access through a resource search engine.

The performance of GeoCloud4SDI is evaluated through two tests, namely, resource pooling and load balancing; and dynamic workload balancing. The results of resource pooling and load balancing show that distribution of workload to multiple instances provides good performance. The dynamic workload balancing is evaluated through two different scenarios, viz., increasing workload and spiking workload. Both these evaluations show the effectiveness of GeoCloud4SDI in utilizing the computing resources used in the developed city level SDI when the concurrent users abruptly increase or decrease.

GeoCloud4SDI provides an effective solution to deal with the spatial challenges of data management, computing intensity and end-user access. Data management is achieved by distributing the data into multiple cloud instances through the cloud load balancer. Computing intensity is handled by the elastic load balancer that provisions or releases the computing resources as per the application requirements. End-user access, a common problem in spatial applications, is handled by auto-scaling and load balancing mechanism. It is observed that minimum response time is achieved when GeoCloud4SDI instances are increased.

This research presents the use of GeoCloud4SDI, developed on OpenStack private cloud, in public administration.

The developed framework for GeoCloud4SDI not only provides the customization of private cloud system but also provides the deployment of the application within it. The developed solution is fault tolerant because of the data isolation from the system, i.e., no data will be lost if there is a fault occur in the system due to software or network failure. Security is a major drawback of the public cloud as the data is stored on the other cloud provider enterprises, but this framework is more secure due to being deployed on the private cloud using local computational facilities. The performance evaluation results show that GeoCloud4SDI framework provides several benefits like big data management, ease and fast access to computing resources, cost reduction and global access.

GeoCloud4SDI framework developed for Prayagraj city can also be adopted for other cities in India and the world because it is open, interoperable and cloud enabled due to the use of OSS, OGC interoperable standards and OpenStack cloud that are used worldwide. Further, the development of GeoCloud4SDI will allow planners and administrators, traditionally non-GIS professionals but having basic computer skills, to be able to use geospatial information and geo-processing tools through standard web browsers for achieving sustainable development in Prayagraj city.

**Author contributions** All authors contributed to the study conception and design. Material preparation, data collection and analysis were performed by Ashutosh.Tripathi. Ashutosh.Tripathi wrote the main manuscript text. Ashutosh.Tripathi, Sonam.Agrawal and Rajan.Gupta prepared Figs. 1 and 5. Ashutosh.Tripathi prepared Figs. 2, 3, 4, 6, 7, 8, 9, 10, 11, 12, 13 and 14. All authors read and approved the final manuscript.

**Data availability** The data used to support the finding of this study are included with this article and make it available upon the reasonable request. No more additional data are required for this study.

## Declarations

**Competing interests** The authors declare no competing interests.

**Conflict of interest** The authors declare that they have no conflict of interest.

## References

- Agrawal S, Tripathi AK, Gupta RD (2022) Development and implementation of automatic metadata generation framework for SDI using OSS: a case study of Indian NSDI. *Arab J Geosci* 15:408. <https://doi.org/10.1007/s12517-022-09635-w>
- Ali A, Imran M (2019) The Evolution of National Spatial Data Infrastructure in Pakistan, Implementation Problems and the Way Forward. *Int J Spat Data Infrastruct Res* 15:110–142. <https://doi.org/10.2902/1725-0463.15.art5>
- Ali Q, Kiriansky V, Simons J, Zaroo P (2012) Performance evaluation of HPC benchmarks on VMware's ESXi server. In: Alexander M, Ambra P Da (eds) Euro-Par 2011: Parallel Processing Workshops. Euro-Par 2011. Lecture Notes in Computer Science, Euro-Par 2. Springer Singapore, pp 213–222
- Barham P, Dragovic B, Fraser K, et al (2003) Xen and the art of virtualization. In: Proceedings of the nineteenth ACM symposium on Operating systems principles, SOSP'03. ACM, Bolton Landing, New York, USA, pp 164–177
- Barik RK, Misra C, Lenka RK et al (2019) Hybrid mist-cloud systems for large scale geospatial big data analytics and processing: opportunities and challenges. *Arab J Geosci* 12:32. <https://doi.org/10.1007/s12517-018-4104-3>
- Chen N, Zhang X, Wang C (2015) Integrated open geospatial web service enabled cyber-physical information infrastructure for precision agriculture monitoring. *Comput Electron Agric* 111:78–91. <https://doi.org/10.1016/j.compag.2014.12.009>
- Clinton WJ (1994) Coordinating Geographic Data Acquisition and Access: The National Spatial Data Infrastructure. Executive Order 12906, Washington, DC
- Coleman DJ, McLaughlin J (1998) Defining global geospatial data infrastructure (GGDI): components, stakeholders and interfaces. *Geomatica* 52:129–143. <https://doi.org/10.5623/geomat-1998-0021>
- Cooper AK, Moellering H, Hjelmerger J et al (2012) A spatial data infrastructure model from the computational viewpoint. *Int J Geogr Inf Sci* 27:1133–1151. <https://doi.org/10.1080/13658816.2012.741239>
- Eken S, Sayar A (2019) A MapReduce-based big spatial data framework for solving the problem of covering a polygon with orthogonal rectangles. *Teh Vjesn - Tech Gaz* 26:36–42. <https://doi.org/10.17559/TV-20170418094421>
- Fujioka E, Berge EV, Donnelly B et al (2012) Advancing global marine biogeography research with open-source GIS software and cloud computing. *Trans GIS* 16:143–160. <https://doi.org/10.1111/j.1467-9671.2012.01310.x>
- Gkonos C, Enescu II, Hurni L (2019) Spinning the wheel of design: evaluating geoportal graphical user interface adaptations in terms of human-centred design. *Int J Cartogr* 5:23–43. <https://doi.org/10.1080/23729333.2018.1468726>
- Gupta RD, Agrawal S, Tripathi AK (2022) NSDI based innovative approach for development of open source SDI for health sector: A way forward. In: Garg PK, Tripathi NK, Kappas M, Gaur L (eds) Geospatial Data Science in Healthcare for Society 5.0. Disruptive Technologies and Digital Transformations for Society 5.0. Springer Singapore, pp 273–303. [https://doi.org/10.1007/978-981-16-9476-9\\_13](https://doi.org/10.1007/978-981-16-9476-9_13)
- Herbst NR, Kounev S, Reussner R (2013) Elasticity in cloud computing: what it is, and what it is not. In: Proceedings of the 10th International Conference on Autonomic Computing. San Jose, CA, USA, pp 23–27
- Huang H, Li Y (2009) Load balancing spatial analysis in XML/GML/SVG based WebGIS. In: 2009 International Conference on Environmental Science and Information Application Technology. IEEE, Wuhan, China, pp 538–541
- Huang H, Li Y, Gartner G, Wang Y (2011) An SVG-based method to support spatial analysis in XML/GML/SVG-based WebGIS. *Int J Geogr Inf Sci* 25:1561–1574. <https://doi.org/10.1080/13658816.2010.532133>
- Huang Q, Li J, Li Z (2018) A geospatial hybrid cloud platform based on multi-sourced computing and model resources for geosciences. *Int J Digit Earth* 11:1184–1204. <https://doi.org/10.1080/17538947.2017.1385652>
- Huang Z, Xu Z (2011) A method of using geoserver to publish economy geographical information. In: 2011 International Conference on Control, Automation and Systems Engineering, CASE 2011. IEEE, Singapore, Singapore, pp 1–4
- Jiang H, van Genderen J, Mazzetti P, et al (2019) Current status and future directions of geoportals. *Int J Digit Earth* 0:1–22. <https://doi.org/10.1080/17538947.2019.1603331>
- Kang S, Lee K (2016) Auto-scaling of Geo-based image processing in an OpenStack cloud computing environment. *Remote Sens* 8:662. <https://doi.org/10.3390/rs8080662>
- Kivity A, Kamay Y, Laor D, et al (2007) KVM: the linux virtual machine monitor. In: Proceedings of the Linux Symposium. Ottawa, Ontario, Canada, pp 225–230
- Klinkenberg B (2003) The true cost of spatial data in Canada. *Can Geogr* 47:37–49. <https://doi.org/10.1111/1541-0064.02e11>
- Knoblock CA, Joshi AR, Megotia A, et al (2017) Automatic spatio-temporal indexing to integrate and analyze the data of an organization. In: Proceedings of the 3rd ACM SIGSPATIAL Workshop on Smart Cities and Urban Analytics - UrbanGIS'17. ACM, Redondo Beach, CA, USA, pp 1–8
- Kotsev A, Minghini M, Tomas R et al (2020) From Spatial Data Infrastructures to Data Spaces—A Technological Perspective on the Evolution of European SDIs. *ISPRS Int J Geo-Information* 9:176. <https://doi.org/10.3390/ijgi9030176>
- Lee D, Liang SHL (2011) Geopot: a cloud-based geolocation data service for mobile applications. *Int J Geogr Inf Sci* 25:37–41. <https://doi.org/10.1080/13658816.2011.558017>
- Li Z, Yang C, Huang Q et al (2017) Building model as a service to support geosciences. *Comput Environ Urban Syst* 61:141–152. <https://doi.org/10.1016/j.compenvurbysys.2014.06.004>
- Liu K, Yang C, Li W, et al (2011) The GEOSS clearinghouse high performance search engine. In: Proceedings 19th International Conference on Geoinformatics, Geoinformatics 2011. IEEE, Shanghai, pp 8–11
- Maphale L, Smit JL (2021) A theoretical proposition for spatial data infrastructure on-going improvement. *ISPRS Int J Geo-Information* 10:9. <https://doi.org/10.3390/ijgi10010009>
- Masó J, Pons X, Zabala A (2012) Tuning the second-generation SDI: theoretical aspects and real use cases. *Int J Geogr Inf Sci* 26:983–1014. <https://doi.org/10.1080/13658816.2011.620570>
- Matthys C, Gkonos C (2017) Cloud-based architectures for auto-scalable web geoportals towards the cloudification of the GeoVITe swiss academic geoportal. *ISPRS Int J Geo-Inform* 6:192. <https://doi.org/10.3390/ijgi6070192>
- Mazzetti P, Roncella R, Mihon D et al (2016) Integration of data and computing infrastructures for earth science: an image mosaicking use-case. *Earth Sci Inform* 9:325–342. <https://doi.org/10.1007/s12145-016-0255-5>
- Mell P, Grance T (2011) The NIST definition of cloud computing (draft). In: Natl. Inst. Stand. Technol. Spec. Publ. 800–145, 7. <http://csrc.nist.gov/publications/nistpubs/800-145/SP800-145.pdf>. Accessed 15 March 2019



- Mete MO, Yomralioglu T (2021) Implementation of serverless cloud GIS platform for land valuation. *Int J Digit Earth* 14:836–850. <https://doi.org/10.1080/17538947.2021.1889056>
- Mokarram M, Khosravi MR (2021) A cloud computing framework for analysis of agricultural big data based on Dempster-Shafer theory. *J Supercomput* 77:2545–2565. <https://doi.org/10.1007/s11227-020-03366-z>
- Niaraki AS, Rajabifard A, Kim K, Seo J (2010) Ontology based SDI to facilitate spatially enabled society. In: *GSDI 12 World Conference-Realizing spatially enabled societies*. GSDI 12, Singapore, Singapore, pp 19–22
- Nurmi D, Wolski R, Grzegorzczak C, et al (2009) The eucalyptus open-source cloud-computing system. In: *Proceedings of the 2009 ninth IEEE/ACM international symposium on cluster computing and the grid*. IEEE, Shanghai, China, pp 124–131
- Nyerges TL, Ramsey KS, Wilson MW, et al (2006) Design considerations for an internet portal to support public participation in transportation improvement decision making. In: Balram S, Dragicvic S (eds) *Collaborative Geographic Information Systems*. IGI Global, Hershey, pp 208–236
- Oskouei AK, M. Modiri A, Alesheikh R, et al (2019) An analysis of the national spatial data infrastructure of Iran. *Surv Rev* 51:225–237. <https://doi.org/10.1080/00396265.2017.1420586>
- Rajabifard A (2001) SDI hierarchy from local to global SDI initiatives. In: *Presented at Open Seminar on SDI in Asia and the Pacific Region 7th PCGIAP meeting*. Tsukuba, Japan, pp 1–9
- Ronzhin S, Folmer E, Lemmers R et al (2019) Next generation of spatial data infrastructure: lessons from linked data implementations across europe. *Int J Spat Data Infrastruct Res* 14:83–107. <https://doi.org/10.2902/1725-0463.2019.14.art4>
- Shvachko K, Kuang H, Radia S, Chansler R (2010) The hadoop distributed file system. In: *2010 IEEE 26th Symposium on Mass Storage Systems and Technologies (MSST)*. IEEE, Incline Village, NV, USA, pp 1–10
- Sjoukema J, Bregt AK, Crompvoets J (2017) Evolving spatial data infrastructures and the role of adaptive governance. *ISPRS Int J Geo-Inform* 6:254. <https://doi.org/10.3390/ijgi6080254>
- Sjoukema J, Bregt AK, Crompvoets J, Sensing R (2020) Understanding governance dynamics: the governing. *Int J Spat Data Infrastruct Res* 15:1–35. <https://doi.org/10.2902/1725-0463.2020.15.art1>
- Sotomayor B, Montero RS, Llorente IM, Foster I (2009) Virtual infrastructure management in private and hybrid clouds. *IEEE Internet Comput* 13:14–22. <https://doi.org/10.1109/MIC.2009.119>
- Tang W, Feng W (2017) Parallel map projection of vector-based big spatial data: coupling cloud computing with graphics processing units. *Comput Environ Urban Syst* 61:187–197. <https://doi.org/10.1016/j.compenvurbsys.2014.01.001>
- Thulasidasan S, Kasiviswanathan SP, Eidenbenz S, Romero P (2010) Explicit spatial scattering for load balancing in conservatively synchronized parallel discrete-event simulations. In: *Proceedings - Workshop on Principles of Advanced and Distributed Simulation, PADS*. IEEE, Atlanta, GA, USA, pp 150–157
- Ticheler J, Hielkema JU (2007) Geonetwork opensource internationally standardized distributed spatial information management. *Osgeo J* 2:1–6
- Tripathi AK, Agrawal S, Gupta RD (2018) A comparative analysis of conventional hadoop with proposed cloud enabled hadoop framework for spatial big data processing. *ISPRS Ann Photogramm Remote Sens Spat Inf Sci IV-5:425–430*. 10.5194/isprs-annals-IV-5-425-2018
- Tripathi AK, Agrawal S, Gupta RD (2020) Cloud enabled SDI architecture: a review. *Earth Sci Inform* 13:211–231. <https://doi.org/10.1007/s12145-020-00446-9>
- Trumpy E, Manzella A (2017) Geothopica and the interactive analysis and visualization of the updated Italian National Geothermal Database. *Int J Appl Earth Obs Geoinf* 54:28–37. <https://doi.org/10.1016/j.jag.2016.09.004>
- Vouk MA (2008) Cloud computing- issues, research and implementations. *J Comput Inf Technol* 4:29–34. <https://doi.org/10.2498/cit.1001391>
- Vukotic A, Goodwill J (2011) *Introduction to Apache Tomcat*. In: *Apache Tomcat 7*. Apress, Berkeley, CA, p 294
- Wang L, Ma Y, Yan J et al (2016) pipsCloud: High performance cloud computing for remote sensing big data management and processing. *Futur Gener Comput Syst* 78:353–368. <https://doi.org/10.1016/j.future.2016.06.009>
- Wang S, Liu Y (2009) TeraGrid GIScience gateway: bridging cyber-infrastructure and GIScience. *Int J Geogr Inf Sci* 23:631–656. <https://doi.org/10.1080/13658810902754977>
- Willmes C, Kürner D, Bareth G (2014) Building research data management infrastructure using open source software. *Trans GIS* 18:496–509. <https://doi.org/10.1111/tgis.12060>
- Xia J, Yang C, Liu K et al (2015) Adopting cloud computing to optimize spatial web portals for better performance to support digital earth and other global geospatial initiatives. *Int J Digit Earth* 8:451–475. <https://doi.org/10.1080/17538947.2014.929750>
- Xing H, Chen J, Wu H, Hou D (2019) A web service-oriented geoprocessing system for supporting intelligent land cover change detection. *ISPRS Int J Geo-Inform* 8:50. <https://doi.org/10.3390/ijgi8010050>
- Xing H, Chen J, Zhou X (2015) A Geoweb-based tagging system for borderlands data acquisition. *ISPRS Int J Geo-Inform* 4:1530–1548. <https://doi.org/10.3390/ijgi4031530>
- Xing Y, Zhan Y (2012) Virtualization and cloud computing. In: Zhang Y (ed) *Future Wireless Networks and Information Systems*. Lecture Notes in Electrical Engineering, Future Wir. Springer, Berlin, Heidelberg, pp 305–312
- Yang C, Goodchild M, Huang Q et al (2011) Spatial cloud computing: how can the geospatial sciences use and help shape cloud computing? *Int J Digit Earth* 4:305–329. <https://doi.org/10.1080/17538947.2011.587547>
- Yang P, Evans J, Cole M, et al (2007) The emerging concepts and applications of the spatial web portal. *Photogramm Eng Remote Sensing* 73:691–698. <https://doi.org/10.14358/PERS.73.6.691>
- Zhang L, Geertman S, Hooimeijer P, Lin Y (2019a) The usefulness of a web-based participatory planning support system in Wuhan, China. *Comput Environ Urban Syst* 74:208–217. <https://doi.org/10.1016/j.compenvurbsys.2018.11.006>
- Zhang J, You S, Gruenwald L (2014) Large-scale spatial data processing on GPUs and GPU-accelerated clusters. *ACM SIGSPATIAL Spec* 6:27–34. <https://doi.org/10.1145/2766196.2766201>
- Zhang C, Di L, Sun Z et al (2019b) Exploring cloud-based web processing service: a case study on the implementation of CMAQ as a service. *Environ Model Softw* 113:29–41. <https://doi.org/10.1016/j.envsoft.2018.11.019>
- Zhang Q, Cheng L, Boutaba R (2010) Cloud computing: State-of-the-art and research challenges. *J Internet Serv Appl* 1:7–18. <https://doi.org/10.1007/s13174-010-0007-6>
- Zhang T, Tsou MH (2009) Developing a grid-enabled spatial web portal for internet GIServices and geospatial cyberinfrastructure. *Int J Geogr Inf Sci* 23:605–630. <https://doi.org/10.1080/13658810802698571>

**Publisher's note** Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Springer Nature or its licensor (e.g. a society or other partner) holds exclusive rights to this article under a publishing agreement with the author(s) or other rightsholder(s); author self-archiving of the accepted manuscript version of this article is solely governed by the terms of such publishing agreement and applicable law.

## Authors and Affiliations

Ashutosh Kumar Tripathi<sup>1</sup> · Sonam Agrawal<sup>2</sup> · Rajan Dev Gupta<sup>3</sup>

Sonam Agrawal  
sonam@mnnit.ac.in

Rajan Dev Gupta  
rdg@mnnit.ac.in

<sup>1</sup> Department of Computing Science and Engineering, United  
University, Prayagraj 211012, Uttar Pradesh, India

<sup>2</sup> GIS Cell, Motilal Nehru National Institute of Technology  
Allahabad, Prayagraj 211004, Uttar Pradesh, India

<sup>3</sup> Civil Engineering Department, and GIS Cell, Motilal  
Nehru National Institute of Technology Allahabad,  
Prayagraj 211004, Uttar Pradesh, India