Big Data as a Service from an Urban Information System

Alexandre Sorokine
Computational Sciences and Engineering Division
Oak Ridge National Laboratory
(865) 574-4966
sorokina@ornl.gov

Rajasekar Karthik
Computational Sciences and Engineering Division
Oak Ridge National Laboratory
(865) 576-1610
karthikr@ornl.gov

Anthony King
Climate Change Science Institute
Oak Ridge National Laboratory
(865) 241-3888
kingaw@ornl.gov

Bhaduri Budhendra
Computational Sciences and Engineering Division
Oak Ridge National Laboratory
(865) 241-9272
bhaduribl@ornl.gov

ABSTRACT

Big Data has already proven itself as a valuable tool that lets geographers and urban researchers utilize large data resources to generate new insights. However, wider adoption of Big Data techniques in these areas is impeded by a number of difficulties in both knowledge discovery and data and science production. Typically users face such problems as disparate and scattered data, data management, spatial searching, insufficient computational capacity for data-driven analysis and modelling, and the lack of tools to quickly visualize and summarize large data and analysis results. Here we propose an architecture for an Urban Information System (UrbIS) that mitigates these problems by utilizing the Big Data as a Service (BDaaS) concept. With technological roots in High-performance Computing (HPC), BDaaS is based on the idea of outsourcing computations to different computing paradigms, scalable to super-computers. UrbIS aims to incorporate federated metadata search, integrated modeling and analysis, and geovisualization into a single seamless workflow. The system is under active development and is built around various emerging technologies that include hybrid and NoSQL databases, massively parallel systems, GPU computing, and WebGL-based geographic visualization. UrbIS is designed to facilitate the use of Big Data across multiple cities to better understand how urban areas impact the environment and how climate change and other environmental change impact urban areas.

5th ACM SIGSPATIAL Workshop on Analytics for Big Geospatial Data (BigSpatial'16)

ACM ISBN 978-1-4503-4581-1. . . $15.00

DOI: http://dx.doi.org/10.1145/3006386.3006391

CCS Concepts

•Information systems → Geographic information systems; •Computer systems organization → Cloud computing; •Applied computing → *Cartography;*

Keywords

urban informatics; environmental change impact; big data as a service; high-performance geocomputing

1. INTRODUCTION

As of 2014, 54% of the world population was living in cities and by 2050 this number is projected to increase to 66% [8]. Global urbanization and challenges to the sustainability of cities under changing climate make understanding of urban dynamics a burning issue. Understanding the interaction between cities and their environment requires insights from a very large number of variables including both social and physical factors. The data needed for understanding urban dynamics are not only large but also very diverse, coming from a variety of disparate sources and multiple scientific communities. Moreover, these data have to be prepared, analyzed, and presented to the user in an organized and comprehensible way.

Big Data and high-performance computing have proven themselves valuable tools in urban research [1]. However, adoption of these technologies by the wider scientific community is limited by multiple factors such as the difficulty of finding appropriate data, moving and maintaining large amounts of data, and implementing modeling and analytical algorithms on high-performance and high-throughput computing systems. Here we propose an architecture for an Urban Information System (UrbIS) that aims to assist urban and climate researchers by shifting the burden of many technical and maintenance tasks from a users' infrastructure into the cloud. The architecture is based on the concept of Big Data as a Service (BDaaS) and aims to automate common pieces of work such as finding the right data sets, data wrangling, running models and analytical methods on the high-performance platforms, and presenting computation results. Proposed architecture targets researchers, planners, and analysts with the need to process large and diverse urban datasets using advanced and high-performance models and analytical techniques. At the end we want to reduce the cost of entry to cloud-based and high-performance geoprocessing for this group of users.

The paper is structured as follows: after this introduction, we describe our view of the problem and characterize the challenges with which researchers in the area of urban dynamics are faced. The section 2 "Problem Statement" is followed by the detailed description of the new architecture in the section 3 "Proposed Solution". The proposed architecture is compared to the other existing solutions in the section 4 "Earlier Studies". The section 5 "Implementation Details" discusses underlying software that is used to build UrbIS. Preliminary results and development experience are briefly covered in two final sections 6 and 7.

2. PROBLEM STATEMENT

Understanding urban dynamics in the context of climate change requires integration of a large variety of information sources ranging from demographic, social and economic data, including operation of the city services and public health, to environmental and geophysical observations, and global climate models. Nowadays the lack of data is not the problem, but the ability to seamlessly integrate disparate datasets in a uniform analytical workflow remains an elusive goal. With regards to urban information, the size of the datasets is not the only, or even the primary, challenge. Most of the data sources in this research domain form an "ecosystem of small data" rather than large but uniform datasets in typical Big Data domains like messages in social networks or sensor readings. Almost all urban and related data have geographic and temporal registration that can be used for integration of multiple datasets. However, these datasets are created by different scientific and practitioners' communities, governments and government agencies and sometimes volunteer groups. Heterogeneity is the rule. These data need to be aligned not only in geographic and temporal space but also converted from multiple formats, measurement units, and aligned in terms of variable and observation semantics.

In our experience, the typical Big Data workflow involves the following tasks:

1. Finding the datasets that contain the necessary data.
2. Retrieving and converting the data into forms suitable for processing by standard and in-house analytical and modeling software.
3. Processing the data in most cases using high-performance and massively parallel platforms.
4. Retrieving the results of modeling and analysis and presenting them graphically in a geographic context.

Results of the modeling and analysis can then be used to refine the study, a process that may involve repeating of any of these steps like incorporating additional data sources into the analysis, rerunning simulations and analytical routines with different input parameters, developing new representations for the results in combination with various data, and/or using alternative cartographic methods. Fig. 1 shows workflow steps and screenshots of the UrbIS prototype that demonstrates implementation of this workflow.

The goal of the first step of the workflow is to finding the right data. Typically this requires searching through in-house and public data archives and repositories. The amount of data available on the web under permissive licenses is growing rapidly. However, web searches for data are less effective than searching for web pages. General-purpose search engines like Google or Bing are not particularly well suited for finding data. In most cases they are limited to indexing textual annotations of the datasets and are not able to filter results by spatial or temporal constraints, attributes, and other metadata content. This problem can be addressed by using specialized metadata search engines. Most of the data archives and repositories containing data relevant to urban areas and environment do have capabilities for searching through their metadata holdings. For example, the ORNL Distributed Data Archive (ORNL DAAC, https://daac.ornl.gov) provides access through a THREDDS (http://www.unidata.ucar.edu/software/thredds/current/tds/) protocol that permits complex queries over standard metadata. Similar facilities are available in other data archive like NASA Data Portal (https://data.nasa.gov) and even on the data portal of some cities (*e.g.*, https://data.nashville.gov).

Found datasets have to be retrieved from the data portals and archives and made available locally. The size of the datasets typically ranges from a few kilobytes for per-state nation-wide tables to hundreds of terabytes for global climate model ensembles and historical archives of satellite imagery. Moving large data across the Internet is still rather time consuming and not always reliable or in the extreme cases is not possible without physical transfer of the storage media. Integrity of the transferred data has to be verified especially when the datasets are distributed in error-prone formats. Retrieving of small datasets is not a problem either in terms of connection bandwidth or in terms of storage space. However, for both large and small datasets maintenance of local copies (whethe reformatted or to avoid redownloading) is challenging. The datasets have to be catalogued, verified for completeness and integrity, and regularly checked with multiple vendors for updates and corrections.

After the data have been retrieved and stored locally, they have to be prepared for use with analytical and modeling software. As in most cases with disparate datasets this step requires more effort than a simple format conversion. The interdisciplinary character of the urban research often results in the need to merge datasets from different scientific communities. Such datasets may have significant semantic dissimilarities like conflicting definitions of the attributes, incompatible grids, or inconsistent spatial regions.

Processing of big volumes of data requires large computational resources that can be obtained in high-performance and high-throughput computing environments. There are multiple types of high-performance platforms that can be effectively used in urban dynamics research for solving different types of problems. Commodity clusters are common but usually limited in terms of disk throughput that is important for Big Data loads. More advanced systems like massively-parallel ORNL supercomputers TITAN and Eos or UIUC ROGER utilize parallel file systems Lustre or IBM GPFS that help to mitigate I/O bottlenecks. Machine learning algorithms are well-suited for GPU processing that is available on some of the leadership supercomputers. There are also specialized platforms like Cray Urika optimized for

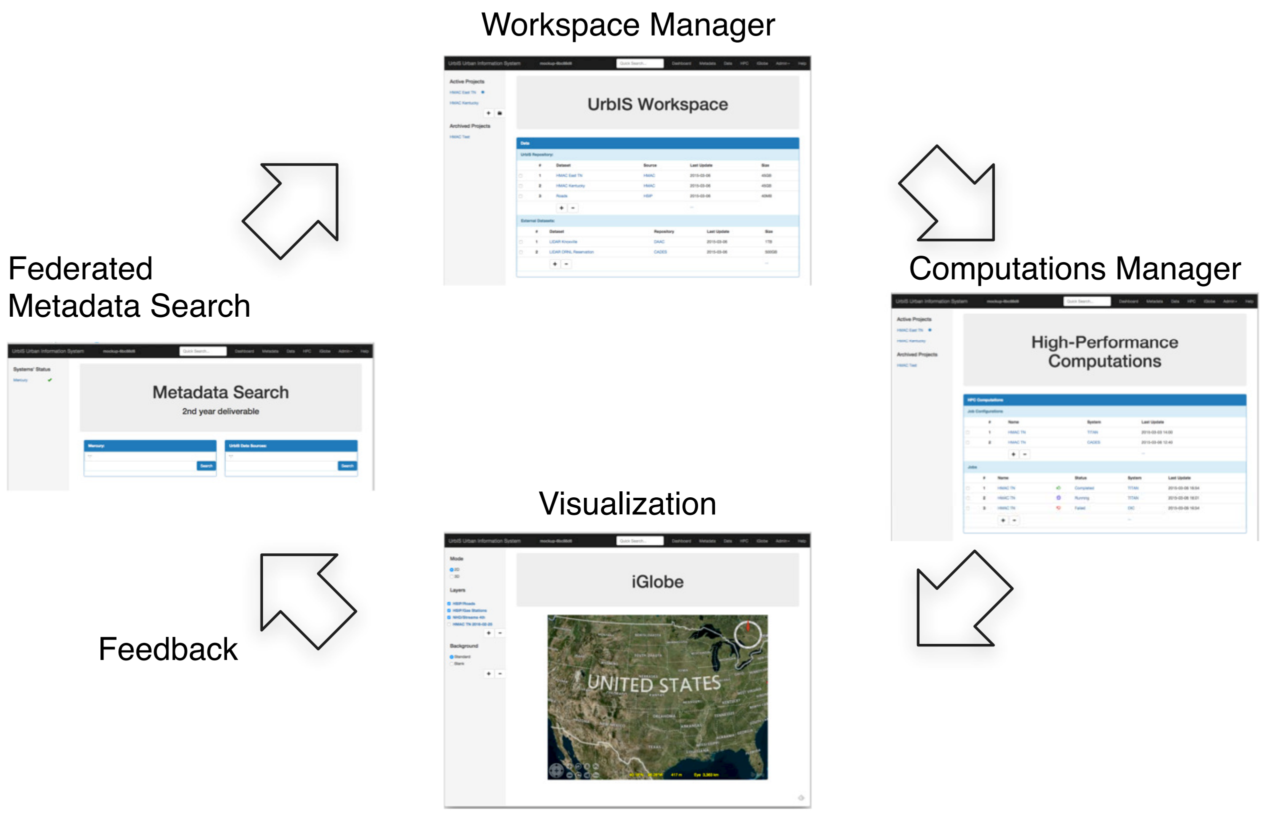

Figure 1: UrbIS Workflow

graph computations and suitable for solving hard to parallelize network problems. In a more recent development, cloud technologies are being adapted for high-performance computations. The heterogeneity of Big Data and high-performance computing platforms require a diverse set of skills from the researchers who want to utilize such resources. Additional burden results from the need to maintain code and executables on multiple architectures and environments.

As the final step of the workflow (Fig. 1), computation results have to be presented in graphical form in a geographic context. Most urban data, within or among cities, requires visualization with tools available in geographic information systems (GIS). Thus there is always a need to translate the data generated by the analytical or modeling software into formats understood by GIS.

Greater adoption of Big Data and high-performance computing technologies in urban dynamics research can improve the quality and efficiency of the studies. However, a number of existing technical shortcomings prevent these innovations from faster adoption. Here we propose a solution to help scientists to overcome these problems using emerging web and cloud technologies.

3. PROPOSED SOLUTION

Cloud technologies have achieved dominant positions in a number of application domains thanks to their ability to offload from the users to the service providers many routine tasks like systems maintenance, backups, or disaster recovery. The benefits of using service-oriented architecture for data processing and analysis are rather obvious from the user's perspective: the burden of maintaining of infrastructure and data can be outsourced to the provider, eventually leading to significant costs saving resulting from the economy of scale. Big Data-as-a-Service is already being used in commercial application for analytics in social networks and health data [14, 9]. In this project we apply the concept of BDaaS to advance urban dynamics research. The architecture that is being implemented is shown on Fig. 2. Our aim is to facilitate the workflow detailed in the previous section (Fig. 1) and overcome the limitations of existing technologies.

UrbIS core functionalities can be grouped into dataset search, data management, high-performance computations, and visualization. Significant portions of UrbIS infrastructure are devoted to data storage and management. UrbIS has facilities for searching through the metadata of external and in-house data archives, retrieval of the data from external sources, storing of the data with restricted access for internal use, format conversion, and serving visualization data.

Discovery of the datasets is supported by the federated metadata search based on ORNL Mercury [6]. Mercury indexes metadata from a large number of external data repositories including ORNL DAAC (https://daac.ornl.gov) and is based the open-source SolR search engine (http://lucene.apache.org/solr/). In addition Mercury also contains indexes of internal UrbIS data holdings. The interface for the metadata search is unified for both internal and external data sources and is shown on the screenshot "Federated Metadata Search" on Fig. 1.

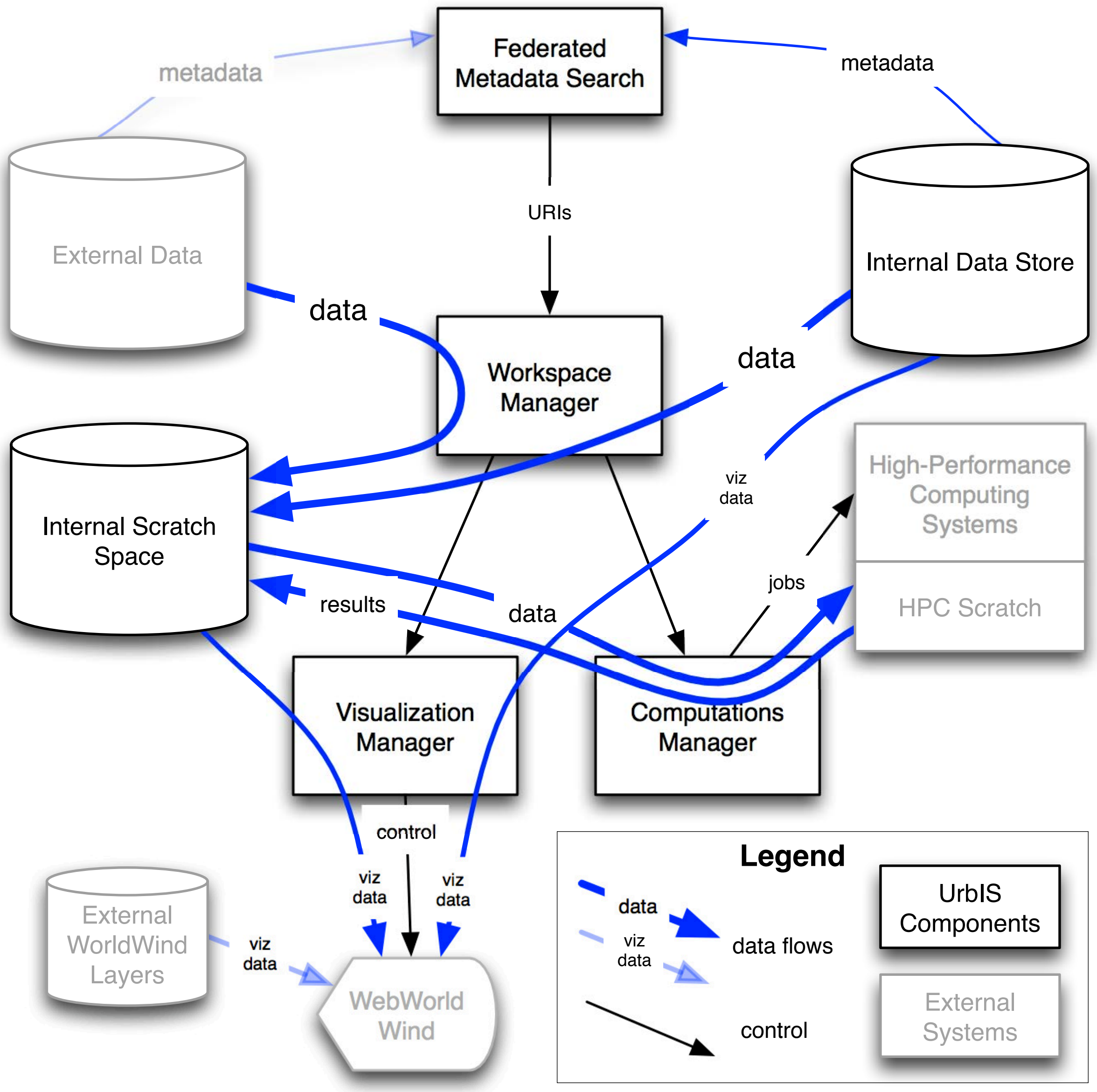

Figure 2: UrbIS Architecture

After the necessary datasets have been found and selected by the user their URIs are handed from the search engine to the workspace manager (Fig. 2). Next, the user specifies the region of interest (ROI) and the system starts downloading necessary fragments of the data into the internal scratch space (Fig. 2). The data preparation is controlled from the UrbIS Workspace page (screenshot "Workspace Manager" on Fig. 1). The data can be downloaded either from external or internal sources. For external sources OpenDAP, WCS and WFS protocols are supported. The copies of the data stored in the scratch space are used for further processing and can be reused again to reproduce the same results.

To process the data, the user chooses from a list of supported analytical methods, specifies the necessary inputs and then submits the task for processing (screenshot "Computations Manager" on Fig. 1). Smaller tasks like data conversion or reprojection can be performed right on UrbIS infrastructure itself. Tasks requiring intensive computations are shipped to high-performance platforms. UrbIS is able to utilize a number of HPC resources including massively-parallel supercomputers, commodity clusters and cloud-based systems. The computations manager (Fig. 2) pulls the input data from the internal scratch space and converts them into the formats used by the analytical or modeling software. Then if the task is going to be processed on a high-performance system its input data will be transfered to the scratch space of that high-performance system. Computations manager can submit the jobs directly to the batch system or generate scripts for manual submission. After the job completes the results are transferred back to the internal scratch space and associated with the user's workspace. From there the results and input data are available for further processing and/or visualization.

We use WebWorldWind (https://webworldwind.org) to visualize the results of modeling and analysis in the geographic context (screenshot "Visualization" on Fig. 1). Visualization manager (Fig. 2) stores the styles and other content of the visuals. WebWorldWind is a digital globe developed by NASA and is intended for scientific and engineering use. WebWorldWind is capable of visualizing geodata using several standard protocols like GeoJSON and WMS. In addition, WebWorldWind supports user-customizable background layers such as roads, cities, major infrastructure elements, pseudocolor satellite imagery, and others. This information is typically necessary to place the users' data and results of analysis in a geographic context. All UrbIS functions including visualization are available from any standard-compliant web browser without the need to install any plugins or desktop applications.

3.1 Use Cases

UrbIS is being developed as an evolving system which progress is guided by daily interaction with the scientists working on several real-world use cases in the area of urban dynamics research:

- finding and characterizing urban heat islands in the south-east U.S. [15],
- using Hierarchical Mode Association Clustering to define urban typologies [16],
- urban mobility modeling for understanding of the changes in energy use with the advent of new technologies like Plug-in Hybrid Electric Vehicles, alternative fuels, and automated vehicles [2].

Operation of UrbIS can be illustrated using the example of finding and characterizing urban heat islands for a specific area. In this case the user starts with creating a project in the UrbIS workspace manager and specifying the region of interest. Then he or she searches for suitable input data using federated metadata search engine in UrbIS. The user finds several candidate datasets that include, for example, temperature records from the Daymet weather data (https://daymet.ornl.gov), locations of the urban areas, and corresponding socio-economic data from the U.S. Census Bureau. At the next step the user picks up the heat island detection algorithm from the list of available analytical routines and specifies the inputs and other parameters. The systems retrieves the requested subset of the Daymet data from external THREDDS data server and creates a local copy of all the needed data in UrbIS workspace. Then depending upon the size of the data the computation can be performed either on UrbIS cloud infrastructure or submitted to a high-performance system. After the computations completes UrbIS notifies the user and retrieves the results. The results of the computation can be demonstrated in a geographic context using UrbIS visualization component. The input data and the results are stored in UrbIS internal data store and can be reused in other computations.

4. EARLIER STUDIES

We have used our experience in several other projects and systems both as users and developers to develop the UrbIS architecture. One of the early UrbIS predecessors is iGlobe — a desktop application for analyzing general circulation model (GCM) climate simulations. iGlobe was developed at ORNL and is based on NASA WorldWind Java SDK [5]. It supports analysis of GCM data in a single workflow including time series analysis and visualization of the results in a geographic context. In addition, iGlobe facilitates data retrieval and performs some of the data processing tasks on a dedicated server rather than the user's desktop. UrbIS brings iGlobe architecture up to date by employing such modern technologies as cloud, web applications, and high-performance computing.

Two other technologies that we have examined against our needs are NASA Earth Exchange (NEX) and Google Earth Engine. Comparison of UrbIS with these two systems is depicted in Table 1. All three systems utilize cloud technologies and can perform a number of geoprocessing tasks. However, there are significant difference in the design goals of the systems and how these goals are achieved. NEX users can access a multipetabyte collection of NASA and contributed data. Processing capabilities are available via sandboxed virtual machines (VM) that provide users with access to standard software packages. User's data can be uploaded to their VMs. Google Earth Engine uses a collection of publicly available datasets that is stored on Google servers and is available to the Earth Engine users. Data access and processing is controlled via an API and computations are performed on the Google Compute Engine infrastructure. Google earth Engine can be programmed using an interactive development environment that supports javascript. Results of the analysis and computations can be visualized on Google Maps. Unlike NEX and Earth Engine, UrbIS

	NASA Earth Exchange	Google Earth Engine [7]	ORNL UrbIS
URL	https://nex.nasa.gov/nex	https://earthengine.google.com	internal
metadata search	search through catalog of local data holdings	search through catalog of local data holdings	federated metadata search
Data	holdings of NASA datasets, ability to upload your datasets inside a VM	ready-to-use public datasets, users' datasets can be uploaded via Google Tables	ready-to-use in-house and government data, caching of external data
Processing	VM in a sandbox with standard software packages	API for Javascript and Python, Javascript IDE	packaged data preprocessing scripts and analytical tools
High-performance computing	NASA supercomputers	Google Compute Engine	ORNL supercomputers and cloud
Geovisualization	standard software packages in the VM	Google Maps in the browser	WebWorldWind in the browser
Subject area	Earth Sciences	Global environmental projects	Urban dynamics

Table 1: Comparison with other Systems

targets a more narrow research area (urban dynamics and climate change impacts) with emphasis on data-driven analysis and high-performance leadership computing. Another difference is the use of the federated metadata search engine to unify searches over internal and large numbers of external data repositories. Internal UrbIS data holdings store limited access and privacy-sensitive data that are necessary for characterization of many aspects of urban areas.

5. IMPLEMENTATION DETAILS

This section outlines the technologies that we are using to develop UrbIS.

5.1 Middleware

Our middleware is powered by Node.js, an emerging server-side computing paradigm environment for developing seamlessly and scalable network applications. In contrast to traditional multi-thread programming model, it uses an event-driven and non-blocking I/O model to deliver the power of multithreading, but with low overhead and a lock free design (as compared to the former) [12]. Detailed comparisons can be found in [12]. Due to its lightweight yet scalable nature, it is suitable for data-driven applications running across distributed nodes. It is built on top of Google Chrome V8 engine and the programming languages used is C and JavaScript. Another noteworthy point is that one language — JavaScript is being used to power various components of both client and server-side application in UrbIS.

5.2 Workflow Management

Workflow Management plays a key role in an information system like UrbIS for data-driven analysis, and data and science production. The need for such a component is further strengthened by the federal agencies mission goal to ensure data reliability and reproducibility. To achieve this goal, we have leveraged Flow-Based Programming (FBP) paradigm with assistance from a Node.js module called NoFlo using this paradigm. It achieves the separation of the control flow of UrbIS from the actual business logic using Object-Oriented Programming (OOP) paradigms. Independent components in UrbIS are split into logical sections, and are brought together in a graph, which has been especially useful in processing and management in Big Data applications. Each graph can consist of sub-graphs. With reusability in mind, these sub-graphs are being created so they can be reused by other flows. In comparison to other workflow tools like Kepler, Taverna, and Swift, we choose NoFlo due to the above benefits and also as it fits our JavaScript-based ecosystem much better [13]. Workflow in NoFlo is fully customizable, where each task is executed asynchronously, following the principle of Node.js. Workflows can be created and edited by either a web-based graph editor or NoFlo graph file format. In UrbIS, we have designed our workflow management to support data changes over time, i.e. workflow can be re-executed when there is change in the underlying datasets, old and new results compared and visualized over time.

For data services between our middleware and database, we have used Model-View-Controller (MVC) approach. We have adopted a framework called Sails.js (http://sailsjs.org) that is well suited for a scalable infrastructure like UrbIS for providing data-driven APIs. Sails.js provides a powerful, yet simple Object-relational mapping (ORM) component to various databases. Using the data mapper patterns, JavaScript classes can be abstracted and mapped to wide variety of databases with persistence. Inheritance and association relationships between the data models and decoupling of database schema and the data models can be infused in a simple and clear fashion from the beginning. It also follows the complimentarity principle of automation where the REST APIs for data access and manipulation can be auto-generated, reducing the redundant tasks for jumpstarting an enterprise-level application. Module interoperability with other Node.js based modules, agnostic front-end compatibility, support for emerging standards like WebSockets, and declarative and reusable security policies (encouraging encapsulation) are some of the other powerful features Sails.js offers.

For geoprocessing and spatial analysis operations, we are using wide collections of technologies inside our Node.js environment. Turf.js aids in spatial data aggregation, measurement, transformation and interpolation. Built on top of Node.js modules such as GeoJSON, shapefile etc, our custom modules add enhanced support for spatial data types and data structure conversions such as from immutable JavaScript arrays to GeoJSON and shapefile. For additional data transformation support, we have utilized modules such as proj4, geolib etc. [3].

Grunt (http://gruntjs.com) is the Continuous Integration (CI) tool used for automation of repetitive tasks for debugging, testing, and production build. With a plugin-based ecosystem, we are developing both in-house plugins and using community contributed plugins for our mundane CI tasks.

5.3 Front End

Development of responsive web and mobile applications is in the heart of UrbIS design, and as such we have incorporated Bootstrap.js, one of the leading JavaScript modules that simplifies achieving this purpose. We are using it for our system webpages to confirm to industry standards and usage guidelines across a wide variety of devices from phones to tablets to desktops.

For security and authentication to our system, we are using the role-based security access control approach via Passport.js module. It is a highly popular module due to its broad support strategies for wide variety of existing systems be it LDAP, SAML, OAuth, OpenID or social media such as Facebook, Twitter etc. This feature will be useful so that external users would like to access our system, without the need for registration. Last, but not the least, in our Node.js environment, we have employed a multi-transport async library called Winston for logging purposes.

Harnessing "Graphics Processing Unit" (GPU) for graphics computations and rendering, WebGL has taken the next leap on Computer Graphics on clients (browser) to deliver very rich and interactive experience to the user. Prior to WebGL, plug-ins or native applications were required for the same visualization experience, notably 3D [12, 4]. In this section, we discuss few a WebGL-based libraries that are being used and planned for use in the system.

To simplify the use of WebGL, we are also adopting another WebGL based library called Three.js due to its support for various common utilities such as textures, lighting, shaders etc. [12]. Typed Arrays is a powerful data type introduced in ECMAScript 6 which can be used by WebGL with memory efficient and safe access to values of the array, when compared to standard JavaScript Arrays [10].

5.4 Database Backbone

PostgreSQL is a widely used Relational Database Management System (RDBMS). We choose PostgreSQL for three reasons:

- Existing installations: Many of our internal systems for settlement mapping, mobility science, and critical infrastructure applications use PostgreSQL as the backend. For seamless systems-interoperability and ability for information to freely flow across systems, we also have chosen PostgreSQL.

- PostGIS: Be it any RDBMS or NoSQL solutions, PostGIS is one of the well-supported tool that supports spatial database out-of-the-box.

- Evolving data sets and formats: prior to 9.4, adherence to strict schema structure and limited support for JSON made it unfit for the emerging data trends [11]. These are two of the biggest reasons for NoSQL solutions in getting wide attention. But with 9.4+, these two challenges were addressed with two data types hstore (a schemaless key-value), and jsonb (binary representation of JSON). While the former made it possible for one database to be used for both relational and non-relation data models, the latter made it seamless ingestion of evolving dataset formats such as document data [11]. By bringing these powerful flexible features of NoSQL to PostgreSQL, it has made PostgreSQL highly attractive for continued use and for handling Big Data storage. PostgreSQL/PostGIS plays a major contributor to the development goals of UrbIS.

Inside UrbIS we use PostgreSQL with PostGIS for storing bulky vector and raster data and for some geoprocessing operations like clipping and spatial queries. In addition to PostgreSQL we are evaluating the use of cloud-based geospatial databases and data processing engines such as GeoMesa (www.geomesa.org) and GeoTrellis (geotrellis.io). These systems can directly use cloud-based data stores such as Spark (http://spark.apache.org/) or Cassandra (http://cassandra.apache.org) that may become available on UrbIS cloud infrastructure.

6. PRELIMINARY RESULTS

Currently UrbIS is under active development and some parts of the system are available for testing for internal users. In its early development stages the system is primarily used as a development platform that provides researchers with simplified access to a number of datasets and on-line services like metadata search and WMS servers. The system has already demonstrates the advantages of having a single catalog of metadata and sharing large datasets among multiple research groups. In the near future the system will be deployed to more advanced cloud infrastructure and made accessible to a larger number of users.

7. CONCLUSIONS AND FUTURE WORK

We have presented our approach to building of an urban information system (UrbIS) around the concept of Big Data as-a-service (BDaaS). With UrbIS we aim to leverage cloud and web technologies coupled with high-performance computing to advance and accelerate urban dynamics research. Our goal after completing the initial implementation of UrbIS will be to identify performance and usability bottlenecks in the system by testing it against a variety of use cases defined by practicing urban scientists. The system will be optimized for more effective use of cloud-based resources and services. We also plan to exposes most of UrbIS functionality via web APIs using standard-based protocols like OGC Catalog Service (http://www.opengeospatial.org/standards/cat) and OGC Web Processing Service (http://www.opengeospatial.org/standards/wps).

8. ACKNOWLEDGMENTS

This manuscript has been authored by employees of UT-Battelle, LLC, under contract DE-AC05-00OR22725 with the U.S. Department of Energy. Accordingly, the United States Government retains and the publisher, by accepting the article for publication, acknowledges that the United States Government retains a non-exclusive, paid-up, irrevocable, world-wide license to publish or reproduce the published form of this manuscript, or allow others to do so, for United States Government purposes.

The authors would like to acknowledge the financial support for this research from the US Government for Oak Ridge National Laboratory's Laboratory Directed Research and Development (LDRD) project number 9969.

9. REFERENCES

[1] M. Batty. Big data, smart cities and city planning. *Dialogues in Human Geography*, 3(3):274–279, 2013.

[2] B. Bhaduri, D. Patlolla, R. R. Vatsavai, A. Cheriyadat, W. Lu, and R. Karthik. Emerging trends in monitoring landscapes and energy infrastructures with big spatial data. *SIGSPATIAL Special*, 6(3):35–45, 2015.

[3] H. Borges, M. T. Valente, A. Hora, and J. Coelho. On the popularity of GitHub applications: A preliminary note. *arXiv:1507.00604 [cs]*, 2015.

[4] D. Cantor and B. Jones. *WebGL Beginner's Guide*. Packt Publishing, 2012.

[5] V. Chandola, R. R. Vatsavai, and B. Bhaduri. iGlobe: an interactive visualization and analysis framework for geospatial data. In *Proceedings of the 2nd International Conference on Computing for Geospatial Research &amp; Applications*, COM.Geo '11, pages 21:1–21:6. ACM, 2011.

[6] R. Devarakonda, G. Palanisamy, B. E. Wilson, and J. M. Green. Mercury: reusable metadata management, data discovery and access system. *Earth Science Informatics*, 3(1):87–94, 2010.

[7] Google Earth Engine Team. *Google Earth Engine: A planetary-scale geo-spatial analysis platform.* Google Inc.,, 2015. Published: https://earthengine.google.com.

[8] G. K. Heilig. *World urbanization prospects the 2014 revision.* United Nations, 2015.

[9] IBM and Twitter turn tweets into business insights. http://www-03.ibm.com/press/us/en/pressrelease/46330.wss.

[10] R. Karthik. SAME4hpc: A promising approach in building a scalable and mobile environment for high-performance computing. In *Proceedings of the Third ACM SIGSPATIAL International Workshop on Mobile Geographic Information Systems*, MobiGIS '14, pages 68–71. ACM, 2014.

[11] R. Karthik and P. Chowdhury. A comparison of data storage technologies for remote sensing cyberinfrastructures. 2016.

[12] R. Karthik and W. Lu. Scaling an urban emergency evacuation framework: Challenges and practices. In *Workshop on Big Data and Urban Informatics (BDUIC)*, 2014.

[13] B. Ludäscher, I. Altintas, C. Berkley, D. Higgins, E. Jaeger, M. Jones, E. A. Lee, J. Tao, and Y. Zhao. Scientific workflow management and the kepler system. *Concurrency and Computation: Practice and Experience*, 18(10):1039–1065, 2006.

[14] B. Marr. Big Data-As-A-Service is next big thing. *Forbes*, 2015.

[15] T. R. Oke. City size and the urban heat island. *Atmospheric Environment (1967)*, 7(8):769–779, 1973.

[16] S. Surendran Nair, B. L. Preston, A. W. King, and R. Mei. Using landscape typologies to model socioecological systems: Application to agriculture of the united states gulf coast. *Environmental Modelling & Software*, 79:85–95, 2016.