Research, Applied Technology, Education and Service, Inc.

Rio Grande Valley Flood Management

Project Deliverable ID Dev

Andrew Ernest <anernest@ratesresearch.org>

Jan 07, 2023

Approval Page

Technical Review By:

Andrew N.S. Ernest, Ph.D., P.E., BCEE, D.WRE
Lead Software Developer

Final Approval For Submission:

Andrew N.S. Ernest, Ph.D., P.E., BCEE, D.WRE
President and CEO

January 7, 2023

CONTENTS

This constitutes Deliverable 1.2.1.4.3.2.3 “End-User Interface Development Report”.

CHAPTER

ONE

BACKGROUND

A beta version RGVFlood.com was released in the Fall of 2021 to demonstrate the proposed functionality of the system. This version of RGVFlood.com was based on GeoNode v3.3.2, with additional enhancements, including integration of an API for ingestion of timeseries RTHS data, a 'Flood Wizard' app for visualizing data and a H&H model availability. After operating for 6 months, it was determined that installation on a single bare-metal server was insufficient for the anticipated demand, and the base software suite upon which RGVFlood.com was constructed, limited the implementation of Continuous Integration/Continuous Delivery pathways for the already integrated extension and planned extensions - specifically those associated with H&H model execution.

In the Spring of 2022, the beta site was decommissioned to allow for spin-up of a re-incarnation of RGVFlood.com addressing the key issues identified during beta deployment:

1. Migration from a single-stack Docker container deployment, to a scalable Kubernetes cluster.
2. Reliance on an inter-pod shared, cloud-based NFS volume service to ensure expansion and accomodation of the large volumes of geospatial data to be stored, and that produces by the integrated H&H and visualization tools.
3. Reliance on a managed cloud-served database service, to ensure minimal data access bottlenecks.
4. Integration of GeoNode v4.0 to promote future growth and seamless conntinuous deployment.

CHAPTER

TWO

DEVELOPMENT INFRASTRUCTURE

The RGVFlood.com CI/CD process relies on the following RATES projects:

spyce	Spyce is a suite of tools designed to facilitate development and deployment of the :term:`Water Wizard`w ecosystem of services.
waterwizard	Water Wizard is the host platform for *Wizards* and project documentation deployment.
geonodegcp	Migration pathway for integration of RGVFlood apps, extensiond and cyber-infrastructure into the release cylce of GeoNode, specifically targeting delivery via GCP.
reonode	Development platform for RGVFlood apps and extension prior to release on RGVFlood.
reoncc	An implementation of REONode that includes add data and tools relevant to water resources management, including hydrologic extremes and ecological resilience.

2.1 spyce

Spyce is a suite of tools designed to facilitate development and deployment of the Water Wizard`w ecosystem of services. Most operations related to development, deployment, maintenance, backup/restore, etc of the :term:`REON.cc cyberinfrastructure are handled via the RATES Spyce application. The Spyce Python codebase provides the following functionality.

The initial execution establishes the configuration environment for Spyce. Configuration items are saved in a TOML file on disk, a `config` dictionary and applied to the appropriate classes to create relevant objects. Spyce allows for the execution of most functionality either on the local machine or on a pre-determined remote host.

Spyce can deploy Python applications in 3 incremental modes:

1. `app`: A stand-alone debug mode, relying on Django's built-in runserver function for testing.
2. `compose`: Using one or mode Docker Compose manifests, the app can be deployed to a host with a docker installation. Typically used to debug microservices interactivity and functionality.
3. `k8s`: Using a set of k8s manifests, the app can be deployed for production on k8s hosts.

Spyce can also deploy to and through various configurations of targets:

1. `local`: Deployed on the local machine, within the constraints of its capabilities. Most useful for documentation development and `app` mode deployments. Some local machines may have limited capabilites, such as smartphones, so the suite of functions available here may be limited.
2. `default`: Deployed to a remote host identified as the default target. Typically used for `k8s` mode deployments as GKE instructions need only to be issued from a single machine.
3. `group`: Deployment to a group of remote hosts. This functionality is in the alpha development stage. It's intended use is for constructing and deploying microservices on docker swarms for distributed computing.

Todo:

- Settings from Git Repo.
- Settings in Context data.
- CLI
- Wizards

2.2 waterwizard

Water Wizard is the host platform for *Wizards* and project documentation deployment. It is a loose template for deploying Django apps as a Docker stack, or as a k8s app.

2.3 geonodegcp

Migration pathway for integration of RGVFlood apps, extensiond and cyber-infrastructure into the release cylce of GeoNode, specifically targeting delivery via GCP.

2.4 reonode

Development platform for RGVFlood apps and extension prior to release on RGVFlood.

2.5 reoncc

An implementation of REONode that includes add data and tools relevant to water resources management, including hydrologic extremes and ecological resilience.

CHAPTER

THREE

CLOUD INFRASTRUCTURE

RGVFlood.com cloud deployment relies of conversion of it's component services to be converted to be delivered via a Microservices Architecture. Several options are available for deploying the microservices version of RGVFlood.com, including an on-premise hardware cluster, Google Cloud Platform, Amazon Web Services and Microsoft's Azure.

3.1 Google Cloud Platform

Google Cloud Platfrom (GCP) was selected as the initial platform for RGVFlood.com, based on Cost of Service, and integration with existing RATES operations.

3.2 Google App Engine

Google App Engine (GAE) allows users to deploy containerized apps on GCP, whithout have to worry about underlying server management. Other than providing a cloud testing environment for individual microservices, GAE currently has limited utility for use in the RGVFlood.com ecosystem.

3.3 Google Compute Engine

Google Compute Engine (GCE) provides access to virtual machines (VM), and serves as the infrastructure basis for the GAE environment. A GCE VM was used to support the transition from a self-contained docker-stack to deploy a fully cloud-leverage RGVFlood.com deployment.

3.4 Google Kubernetes Engine

Google Kubernetes Engine (GKE) provides the user access to the container orchestration facilities of GCP. Google's Autopilot custer configuration was selected for cluster creation, allowing for ease of horizontal (node/cpu) and vertical (memory) scaling. As a result, cluster size is reported in terms of the number of vCPU and memory rather than the traditional node count. Running the following applications to support continuous development:

- geonodegcp-app
- reonode-app
- waterwizard-app
- rgvflood-app

with nominal use, the cluster scaled to 9.75 vCPU and 38.2 GB of memory. With one vCPU being roughly equivalent to one hardware core, this is similar in capacity to a single standard bare-metal server. With the integration of user-applications (e.g. RTHS Data API and Flood Wizard), along with anticipated end-user access and demand, horizontal scaling needs are expected to quadruple at a minimum.+

3.5 CloudSQL

Rather than rely on containerized database services, the decision was made to switch to Google CloudSQL managed database services. Similar services are available though AWS and Azure. Unlike a single-stack Docker deployment, switching to a K8s with potentialy multiple replicas needing to access the database services, reliance on managed database services eliminates the need to construct and manage a separate workload specically for database services.

The first step in transtioning to k8s involved deploying the `docker-compose.yml` stack on a GCE VM. The database service was then replaced with a CloudQSL-Proxy service, allowing the containers to access the databases managed by CloudDQL and permitting the number of replicas to be scale with no collisions or impacts in performance.

3.6 Filestore

Persistent file storage is handled differently between standard Docker desktop deployments and scalable K8s clusters. Implelementing persistent storage between reboots and between containers for the K8s deplyoment involved changing from volume mounts to and NFS share. This NFS share is also mounted by as GCE VM used during the development process for debugging. It is anticipated that the volume of filestorage needed will eventually be in excess of 1TB, more once real-time forecast data is produced.

CHAPTER

FOUR

COMPONENT INTEGRATIONS

RGVFlood serves as the core platform into which value-added component services are plugged-in and provided for use by the end-user.

4.1 RTHS Data API

Originally packaged with the beta release of RGVFlood, the RTHS Data API ingests RTHS time-series data on-demand for use by the end-user. RGVFlood is also able to serve the data to other component services such as Flood Wizard, WRF-Hydro and the RAS Data Provider. The RTHS Data API wil be included in the next release of RGVFlood.com.

4.2 Flood Wizard

Flood Wizard is a Javascript Progressive Web Application developed using the Ionic framework. It was packaged with the last beta release of RGVFlood, with the folliowing features:

- Ability to view and inspect the RGVFlood delineated sub-basins and RTHS stations
- Ability to view and download the input data and modeling results from the Tier II modeling effots
- Overlay and compare stage height data from multiple RTHS stations

Flood Wizard was deployed to seed end-user discussion on the type of applications deemed most useful by end-users. Flood Wizard will be included in the next release of RGVFlood.com, incorporating user-recommended modifications and additions.

4.3 RAS Data Provider

The goal of this component is to generate HEC RAS input data for selected domains from RGVFlood.com availalble data on demand. At the very minimum, this will include discharge data extracted from hydrologic model outputs, along with pertinent RTHS data. Other data, such as topographic and hydraulic asset surveys will also be provided. The development of this component is currently on-going.

4.4 WRF-Hydro

Although not a direct integration, the outputs of the Tier I modeling effort will be ingested into RGVFlood.com for visualization and decision support application. Development of this component is pending prototype deployment of the Tier I model.

PYTHON MODULE INDEX

INDEX