

# **WEATHER DATA INTEGRATION AND ASSIMILATION SYSTEM**

Gihan Chanuka Karunaratne

178004U

Degree of Master of Science

Department of Computer Science and Engineering

University of Moratuwa

Sri Lanka

January 2021

# **WEATHER DATA INTEGRATION AND ASSIMILATION SYSTEM**

Herath Mudiyansele Gihan Chanuka Karunaratne

178004U

Thesis submitted in partial fulfillment of the requirements for the degree Master of  
Science in Computer Science and Engineering

Department of Computer Science and Engineering

University of Moratuwa

Sri Lanka

January 2021

# Declaration

I declare that this is my own work and this thesis does not incorporate without acknowledgement any material previously submitted for a Degree or Diploma in any other University or institute of higher learning and to the best of my knowledge and belief it does not contain any material previously published or written by another person except where the acknowledgement is made in the text.

Also, I hereby grant to University of Moratuwa the non-exclusive right to reproduce and distribute my thesis/dissertation, in whole or in part in print, electronic or other medium. I retain the right to use this content in whole or part in future works (such as articles or books).

Signature: *UOM Verified Signature*

Date: 02/01/2021

The above candidate has carried out research for the Masters thesis under our supervision.

Name of the supervisor:

Dr. HMN Dilum Bandara

Signature of the supervisor: *UOM Verified Signature*

Date: 03/01/2021

# Abstract

Numerical Weather Models (NWMs) utilize data collected via diverse sources such as automated weather stations, radars, air balloons, and satellite images. Before using such multimodal data in a NWM, it is necessary to transcode data into a format ingested by the NWM. Moreover, the data integration system's response time needs to be relatively low to forecast and monitor time-sensitive weather events like hurricanes, storms, and flash floods that require rapid and frequent execution of NWMs. The resulting weather data also need to be accessed by many researchers and third-party applications such as logistic and agricultural insurance firms. Existing weather data integration systems are based on monolithic or client-server architectures; hence, unable to benefit from novel computational models such as cloud computing and containerized applications. Moreover, most of these softwares are proprietary or closed-source, making it difficult to customize them for an island like Sri Lanka with different weather seasons. Therefore, in this research, we propose Weather Data Integration and Assimilation System (WDIAS) that utilizes microservices to achieve scalability, high availability, and low-cost operation based on cloud computing. The use of stateless microservices also enables WDIAS to add new features on the fly with rollover capabilities. Moreover, WDIAS provides a modular framework to integrate data from different sources, export into different formats, and add new functionality by adding extension modules. We demonstrate the utility of WDIAS using a cloud-based experimental setup and weather-related synthetic workloads.

**Keywords:** Cloud computing, data assimilation, data integration, microservice, weather

# Dedication

I dedicate this thesis work to teachers, lectures, my family and specially colleagues at Center for Urban Water, Sri Lanka (CUnW-SL). A special feeling of gratitude to my loving parents, Nandawathi Dissanayake and H.M.K. Karunaratne whose put count- less sweats to support me throughout my entire life. My wife Jayani Kumarasinghe and my son Sasmita Karunaratne who missed a lots of wonderful moments to give me freedom to work on this research.

Nevertheless I could not forget all of my friends at CUnW-SL who have supported and be with me during this period of time. Also, special thanks to Dr. Dilum Bandara and Prof. Srikantha Herath for giving me this wonderful opportunity to explore this new domain.

# Acknowledgements

I would like to thank the members of my evaluation committee who were more than generous with their expertise and their precious time. Special thanks to Dr Dilum Bandara, my research supervisor for helping me with his precious time, reading, encouragement and been patient with pushing me to the next level. Also, I am, thankful to Dr. Dilika Peris, and Dr. Indika Perera for serve as my evaluation panel without any hesitation and Dr. Srikantha Herath for giving more knowledge on weather domain and support as my external supervisor.

I would like to thank the Department of Computer Science and Engineering at the University of Moratuwa for allowing me to conduct my research and provide all the assistance requested. Special thanks go to the academic and non-academic staff of the department for their continued support. I also thank the Center for Urban Water, Sri Lanka (CUrW-SL) for providing the domain expertise, access to resources, and financial support during the research.

Finally, I would like to thank the lectures, the evaluation panel and the colleagues who helped me with this project. Their enthusiasm and willingness to provide feedback support me to go far beyond my limits and complete my research with an enjoyable experience.

# Contents

<b>Declaration</b>	<b>i</b>
<b>Abstract</b>	<b>ii</b>
<b>Dedication</b>	<b>iii</b>
<b>Acknowledgements</b>	<b>iii</b>
<b>List of Figures</b>	<b>vii</b>
<b>List of Tables</b>	<b>ix</b>
<b>List of Abbreviations</b>	<b>x</b>
<b>1 Introduction</b>	<b>1</b>
1.1 Motivation . . . . .	2
1.2 Problem Statement . . . . .	3
1.3 Objectives . . . . .	3
1.4 Outline . . . . .	4
<b>2 Literature Review</b>	<b>5</b>
2.1 Delft-FEWS flow forecasting system . . . . .	5
2.2 Linked Environments for Atmospheric Discovery . . . . .	12
2.3 Data Integration and Analysis System . . . . .	17
2.4 Meteorological Assimilation Data Ingest System . . . . .	18
2.5 Meta Scientific Modeling . . . . .	20

2.6	Summary . . . . .	22
<b>3</b>	<b>Research Methodology</b>	<b>24</b>
3.1	High-level Design . . . . .	24
3.2	Architectural Decisions . . . . .	27
3.3	Microservice Architecture . . . . .	32
3.3.1	Smart Endpoints and Dumb Pipes Microservice Pattern . . . . .	33
3.3.2	Database per Microservice Pattern . . . . .	33
3.3.3	Distributed Transactions over Microservices Pattern . . . . .	34
3.3.4	The Scale Cube Concept . . . . .	34
3.3.5	WDIAS Microservices . . . . .	35
3.3.6	WDIAS Application Programming Interface . . . . .	38
3.4	WDIAS Database Structure . . . . .	39
3.4.1	Key Attributes of Timeseries . . . . .	40
3.4.2	Timeseries Metadata Storage . . . . .	43
3.4.3	In-memory Database . . . . .	44
3.4.4	Timeseries Database . . . . .	45
3.4.5	Network Common Data Form . . . . .	45
3.4.6	Document-oriented Database . . . . .	46
3.5	Data Preprocessing . . . . .	46
3.5.1	Interpolation . . . . .	47
3.5.2	Transformation . . . . .	49
3.5.3	Validation . . . . .	49
3.5.4	Extension API for Data Preprocessing . . . . .	50
3.5.5	Extension Scheduler . . . . .	53
3.6	Query Timeseries . . . . .	53
3.7	Summary . . . . .	54
<b>4</b>	<b>Performance Analysis</b>	<b>56</b>
4.1	Test Plan . . . . .	56
4.1.1	List of Test Plans . . . . .	59



4.1.2	Performance Metrics . . . . .	59
4.2	Workload Generation . . . . .	60
4.2.1	Experimental Setup . . . . .	63
4.2.2	Configure Experimental Setup on Cloud . . . . .	68
4.2.3	Performance Tuning . . . . .	69
4.3	Performance Test Observations . . . . .	71
4.3.1	Load Testing with Hourly Resolution Data . . . . .	71
4.3.2	Load Testing with 30-minute Resolution Data . . . . .	73
4.3.3	Load Testing with 15-minute Resolution Data . . . . .	75
4.3.4	Load Testing with 15-minute Resolution Data with Auto Pod Scaling . . . . .	80
4.3.5	Query Module Load Test . . . . .	86
4.4	WDIAS Performance Evaluation . . . . .	88
<b>5</b>	<b>Summary</b>	<b>91</b>
5.1	Conclusions . . . . .	91
5.2	Research Limitations . . . . .	94
5.3	Future Work . . . . .	96
	<b>References</b>	<b>97</b>
	<b>Appendix A WDIAS REST API</b>	<b>103</b>
	<b>Appendix B WDIAS Query API</b>	<b>105</b>

# List of Figures

2.1	Schematic structure of a flood forecasting system including Delft-FEWS and communication among other operational systems . . . . .	7
2.2	Architecture of Delft-FEWS . . . . .	9
2.3	Delft-FEWS integration with external models . . . . .	10
2.4	Layered architecture of LEAD . . . . .	13
2.5	LEAD system framework . . . . .	14
2.6	LEAD's service-oriented architecture . . . . .	15
2.7	DIASs common base application platform. . . . .	18
2.8	MADIS data flow . . . . .	19
2.9	The top-level design and overall architecture of MSM system . . . . .	22
3.1	Modules of a weather data system. . . . .	26
3.2	Kubernetes (K8s) architecture. . . . .	29
3.3	WDIAS architecture for handling requests on demand. . . . .	36
3.4	WDIAS architecture for handling requests asynchronously. . . . .	37
3.5	Separation of WDIAS microservices. . . . .	38
3.6	WDIAS database structure. . . . .	44
3.7	A generic mathematical function model for weather data preprocessing. . . . .	47
3.8	Serial interpolation. . . . .	48
3.9	Spatial interpolation. . . . .	49
3.10	Time interval aggregation. . . . .	50
3.11	Timeseries data validation. . . . .	50
4.1	WDIAS load testing plan with changing the request size and RPS. . . . .	58

4.2	Experimental setup with Apache JMeter. . . . .	64
4.3	WDIAS load testing throughput shaping timer RPS configurations. . .	65
4.4	Amazon Elastic Kubernetes Service (Amazon EKS) node setup. . . .	70
4.5	Response time vs threads while load testing with hourly data. . . . .	72
4.6	Latency against server hits while load testing with hourly data. . . . .	73
4.7	Response time vs threads while load testing with 30-minute of data. .	75
4.8	Latency against server hits while load testing with 30 minute of data. .	76
4.9	Response time vs active threads while load testing with 15-minute of data. . . . .	77
4.10	Latency against server hits while load testing with 15-minute of data. .	78
4.11	Transaction throughput vs thread while load testing with 15-minute of data. . . . .	79
4.12	Latency against server hits while load testing with 15-minute of data with enabled auto-scaling. . . . .	81
4.13	Transaction throughput vs threads while load testing with 15-minute of data with enabled auto-scaling. . . . .	82
4.14	Load testing with auto-scaling resource usage of import and export modules over time. . . . .	83
4.15	Load testing with auto-scaling number of pods of import and export modules over time. . . . .	84
4.16	Load testing with auto-scaling resource usage of database adapters over time. . . . .	85
4.17	Response latency over time while load testing query test over 5 minutes.	87
4.18	Response latency times vs threads load testing query test over 5 minutes.	88

# List of Tables

3.1	Comparison of features among existing weather data assimilation and integration systems and WDIAS. . . . .	31
4.1	Amazon EKS nodes . . . . .	68
4.2	Throughput and latency of load test with 60-minute data . . . . .	71
4.3	Throughput and latency of load test with 30-minute data . . . . .	74
4.4	Throughput and latency of load test with 15-minute data . . . . .	76
4.5	Throughput and latency of load test with 15-minute data while enabled K8s auto-scaling . . . . .	80
4.6	Throughput and latency of query test cases with 15-minute data . . . . .	86

# List of Abbreviations

Amazon EKS	Amazon Elastic Kubernetes Service
API	Application Programming Interface
CSV	Comma-separated Values
CUrW-SL	Center for Urban Water, Sri Lanka
Delft-FEWS	Deltares FEWS
DIAS	Data Integration and Analysis System
ESB	Enterprise Service Bus
GRIB	General Regularly-distributed Information in Binary form
JSON	JavaScript Object Notation
K8s	Kubernetes
LEAD	Linked Environments for Atmospheric Discovery
MADIS	Meteorological Assimilation Data Ingest System
Microservice	Microservice Architecture
MSM	Meta Scientific Modeling
netCDF	Network Common Data Form
NWM	Numerical Weather Model
RDBMS	Relational Database Management System
REST	Representational State Transfer
RPS	Requests Per Second
SOA	Service Oriented Architecture
WDIAS	Weather Data Integration and Assimilation System
WRF	Weather Research and Forecast