Observability of Software Systems: Challenges and Opportunities

Prof. Wahab Hamou-Lhadj
Concordia University
Montréal, QC, Canada
wahab.hamou-lhadj@concordia.ca

Keynote Presentation

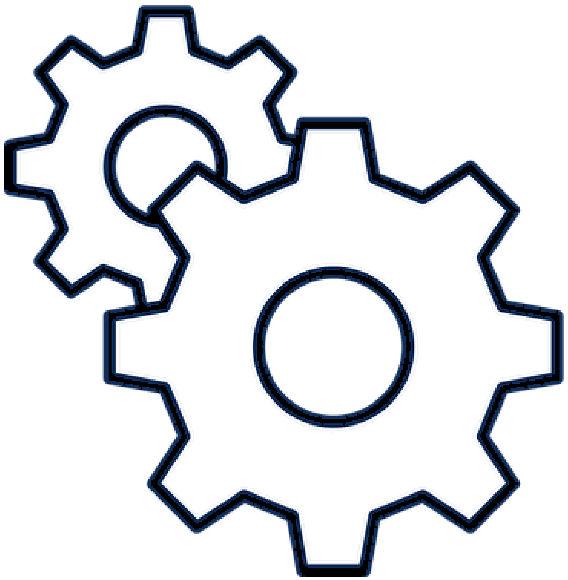
3rd International Conference on Embedded & Distributed Systems, EDiS'2022
Oran, Algeria
November 2-3, 2022

User vs. Operational Data

- **User data** describes information about users.
    - E.g. social media data, user preferences, geo-location data, images, etc.
    - Applications include marketing campaigns, fraud detection, image recognition, etc.

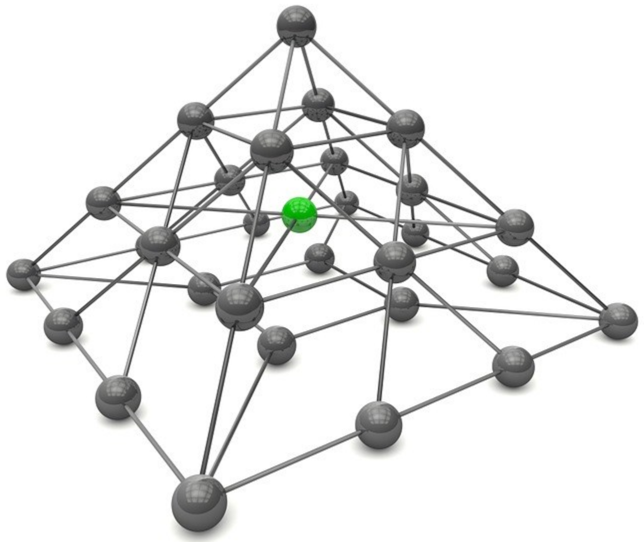
User vs. Operational Data

- **Operational (machine) data** describes information about a system (or a machine)
- It is collected automatically from devices, IT platforms, applications with no direct user intervention.
  - Useful for diagnosing service problems, ensuring reliability, detecting security threats, improving operations, and so on.



Operational Data for Software-Intensive Systems

- The proper functioning of software-intensive systems **relies heavily on operational data** to diagnose and prevent problems.
- New trends in SW dev. make this challenging:
  - Highly distributed and parallel systems
  - Micro-service architectures
  - Virtualisation and containerization
  - Device connectivity and IoT
  - Cyber physical systems
  - Intelligent and autonomous systems
  - Agile, DevOps, and continuous delivery processes

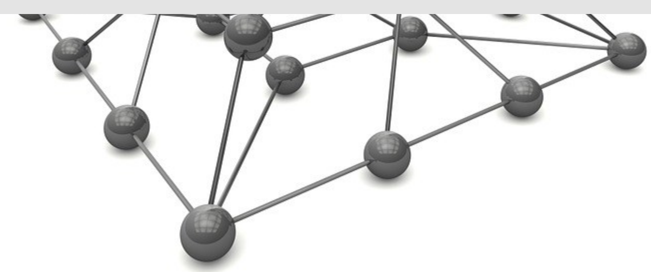


Operational Data for Software-Intensive Systems

- The proper functioning of software-intensive systems **<u>relies heavily on operational data</u>** to diagnose and prevent problems.

We need better runtime system analysis and fault diagnosis and prediction methods that provide full visibility of a system's internal states.

- Micro-service architectures
- Virtualisation and containerization
- Device connectivity and IoT
- Cyber physical systems
- Intelligent and autonomous systems
- Agile, DevOps, and continuous delivery processes



Software Observability

- In control theory:
  - **Observability** is "a measure of how well internal states of a system can be inferred from knowledge of its external outputs" [Wikipedia]

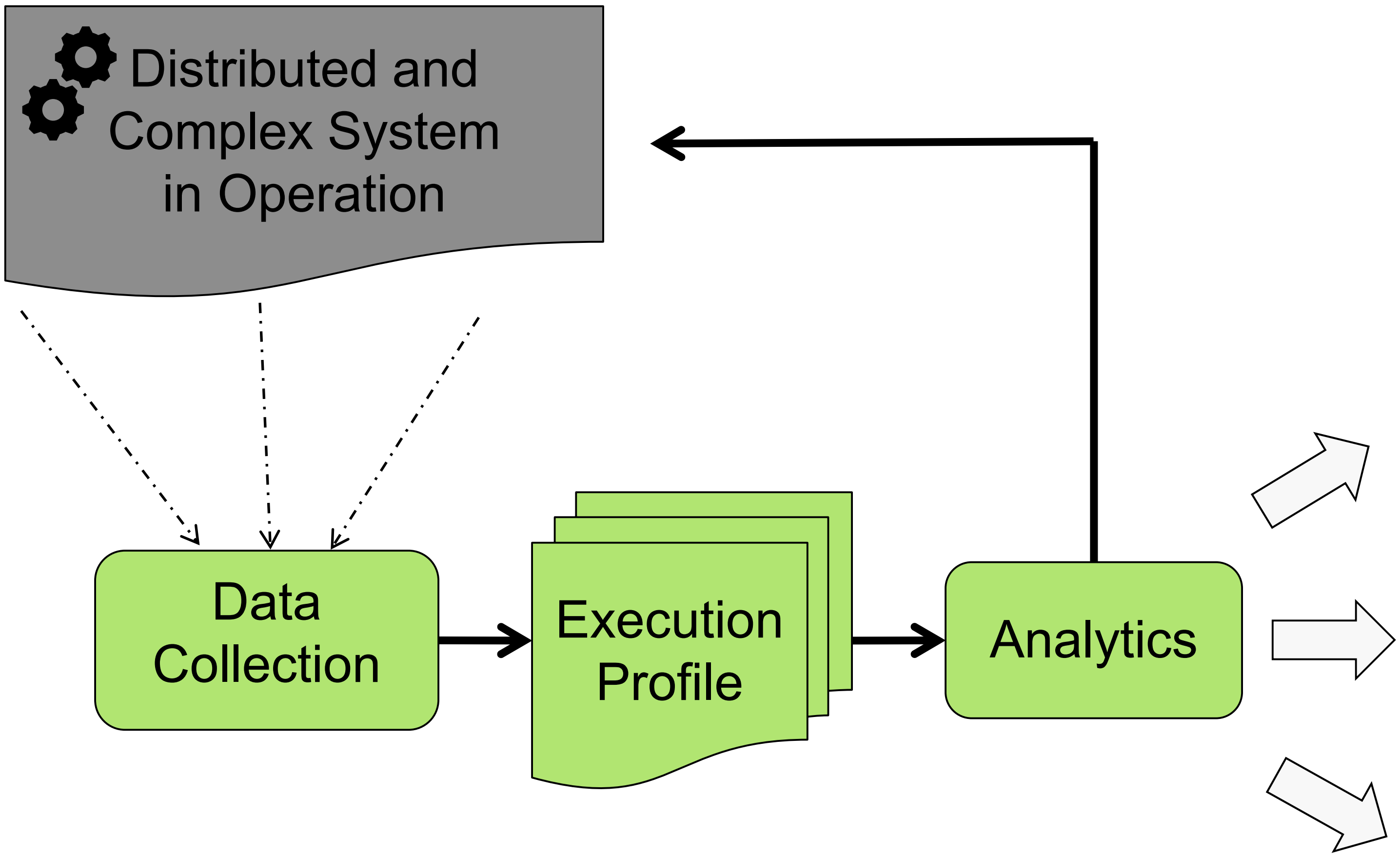
- Software Observability:
  - A set of end-to-end techniques and processes that allow us **to reason** about **what a software system is doing and why** by analyzing its external outputs.

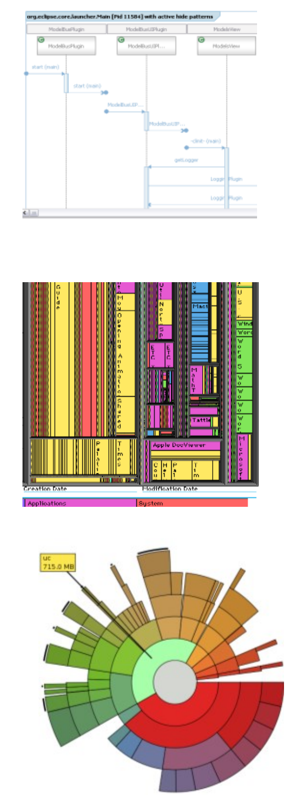
Monitoring vs Observability

- **Monitoring:**
  - Tracks known metrics and raises alerts when thresholds are not met (e.g., 4 golden signals of Google SRE: latency, traffic, errors, and saturation)
  - Answers the question: “how is the system doing?”
  - Helps diagnose known problems

- **Observability:**
  - Answers the question: “what is the system doing and why?”
  - Enables to reason about the system by observing its outputs
  - Helps diagnose known and unknown problems

Building Blocks





Offline and/or real-time analytics

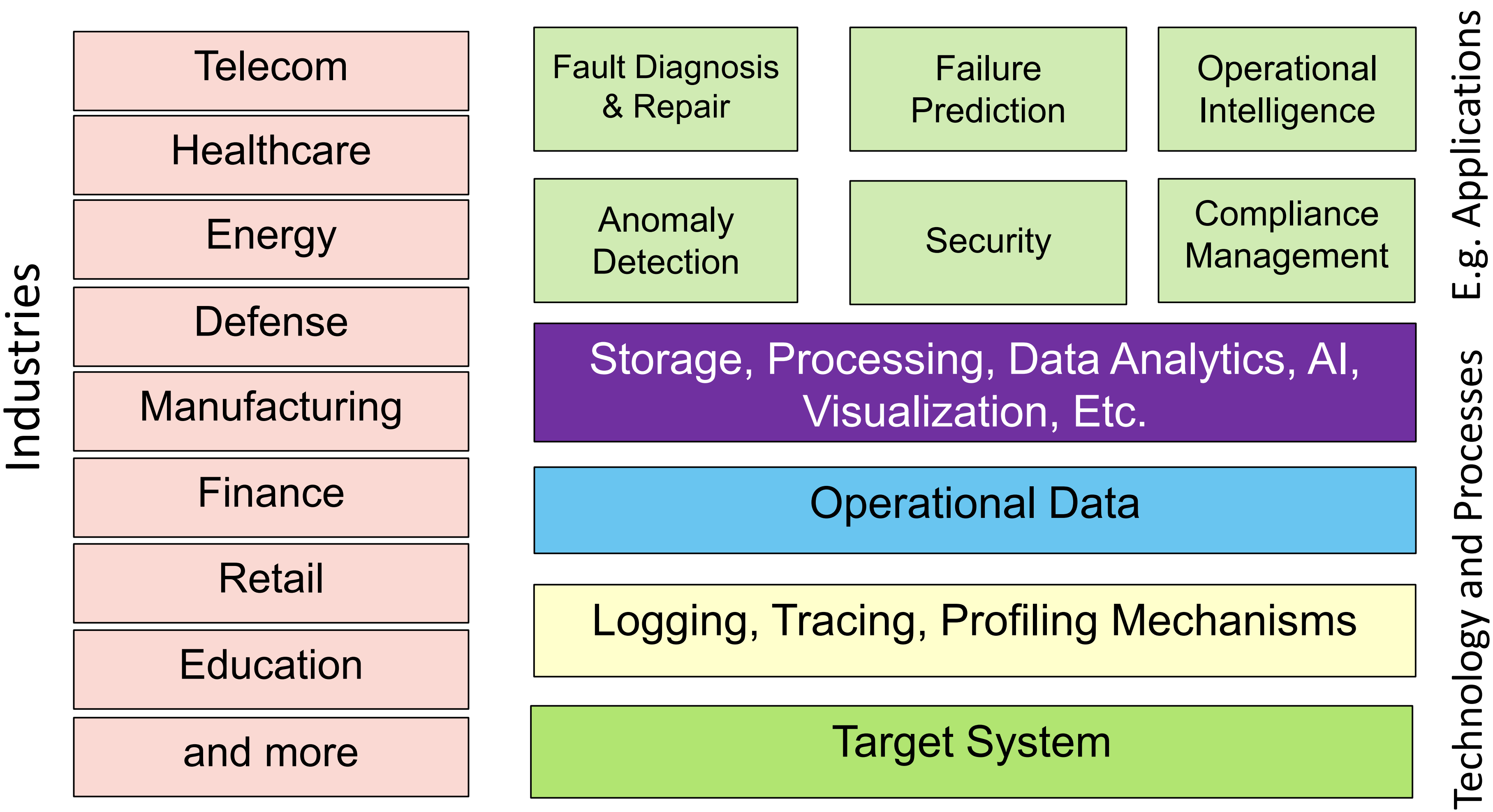
Prof. Wahab Hamou-Lhadj (wahab.hamou-lhadj@concordia.ca) - EDiS 2022

Operational Data

- **Logs:**
  - Records of events generated from logging statements inserted in the code to track system execution, errors, failures, etc.
  - Different types of logs: system logs, application logs, event logs, etc.
- **Traces:**
  - Records of events showing execution flow of a service or a (distributed) system with causal relationship
  - Require additional instrumentation mechanisms
- **Profiling Metrics:**
  - Aggregate measurements over a period of time (e.g., CPU usage, number of user requests, etc.)

Prof. Wahab Hamou-Lhadj (wahab.hamou-lhadj@concordia.ca) - EDiS 2022

Scope of Observability



Emergence of AI for IT Operations

- AIOps is the application of AI to enhance IT operations
- An important enabler for digital transformation
- Building Blocks:
  - Data collection and aggregation
  - Pattern recognition
  - Predictive analytics
  - Visualization
- Applications:
  - Fault detection and prediction
  - Root cause analysis
  - Security
  - Regulatory compliance
  - Operational intelligence

Beyond Software Systems

- Using machine data analytics to drive operational efficiency (a Splunk success story)
- Dubai airport uses machine data to increase airport capacity
- Machine data sources:
  - Flight schedules,
  - Wi-Fi network data
  - Metal detector data
  - Baggage system
  - Sensor data (doors, faucets, etc.)

Source: https://www.splunk.com/en_us/customers/success-stories/dubai-airports.html

Our Past and Current Projects

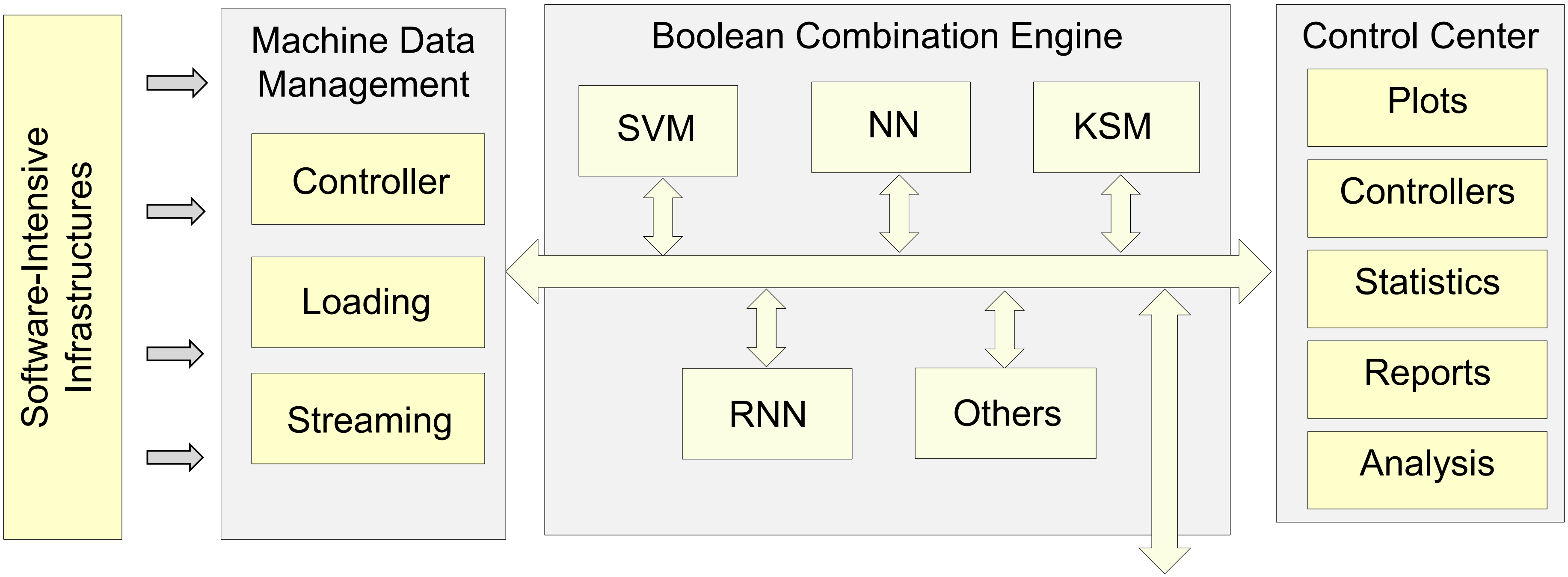
- Md Shariful Islam, "On the use of Software Tracing and Boolean Combination of Ensemble Classifiers to Support Software Reliability and Security Tasks," Ph.D. Dissertation, 2021.
- Korosh K. Sabor, "Automatic Bug Triaging Techniques Using Machine Learning and Stack Traces," Ph.D. Dissertation, 2020.
- Neda E. Koopaei, "Machine Learning and Deep Learning Based Approaches for Detecting Duplicate Bug Reports with Stack Traces," Ph.D. Dissertation, 2019.
- Fazilat Hojaji, "Techniques to Compact Model Execution Traces in Model Driven Approach," Ph.D. Dissertation, 2019.
- Heidar Pirzadeh, "Trace Abstraction Framework and Techniques," Ph.D. Dissertation, 2012.
- Luay Alawneh, "Techniques to Facilitate the Understanding of Inter-process Communication Traces," Ph.D. Dissertation, 2012.

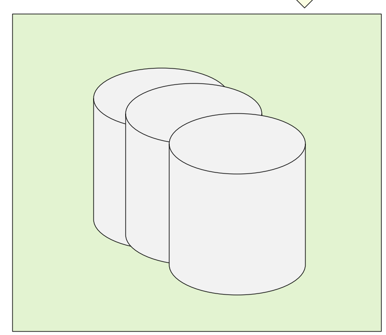
http://www.ece.concordia.ca/~abdelw/publications.html

Software Tracing and Boolean Combination of Ensemble Classifiers to Support Software Reliability and Security Tasks

- PhD Thesis of Shariful Islam in collaboration with Postdoc Wael Khreich
- Contributions:
  - WPIBC: A weighted pruning ensemble of homogeneous classifiers (HMMs) applied to anomaly detection
  - EnHMM: Ensemble HMMs and stack traces to predict the reassignment of bug report fields
  - MASKED: A MapReduce solution for the Kappa-pruned ensemble-based anomaly detection system

TotalADS: Total Anomaly Detection System Architecture





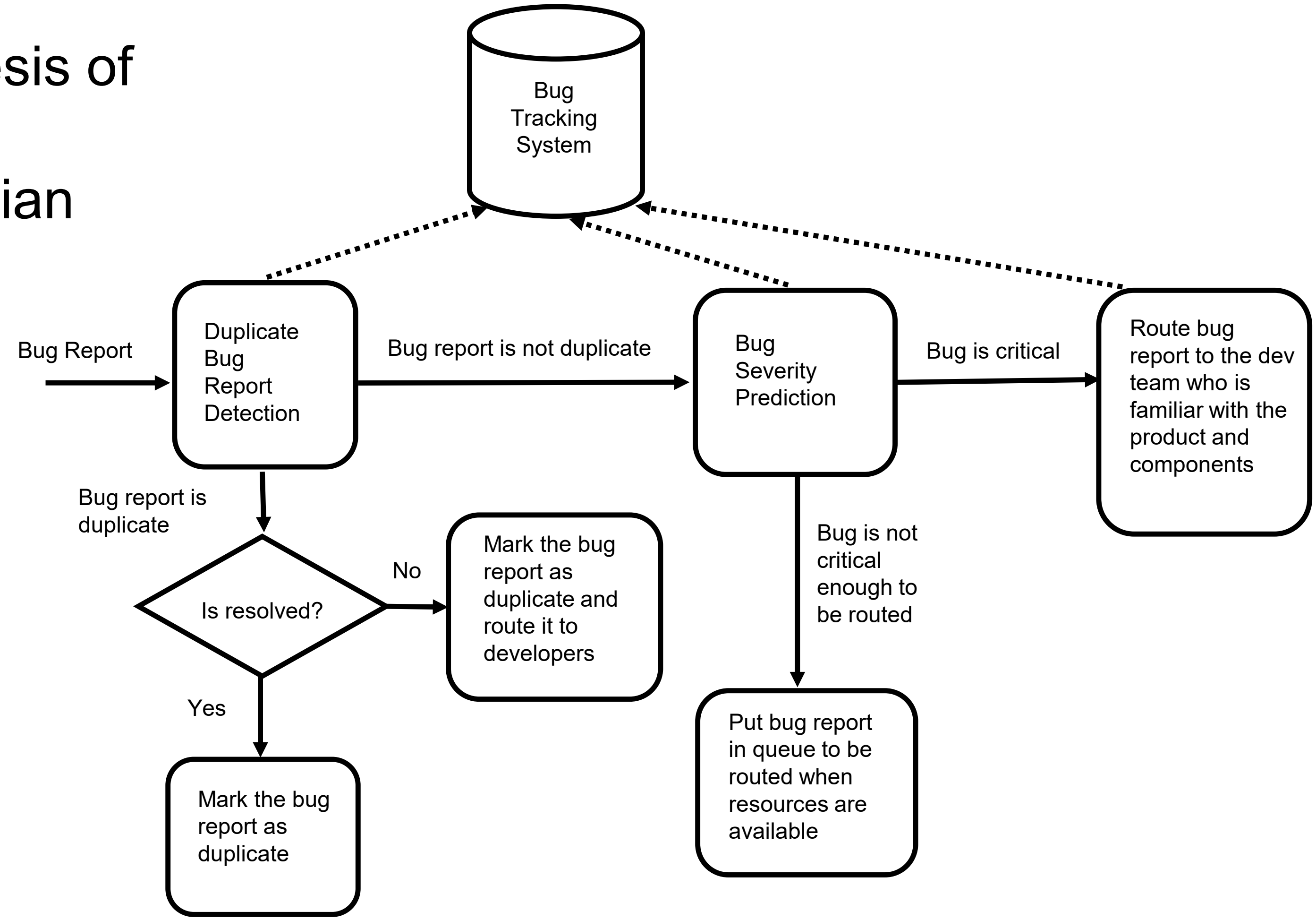
Data Centers
Radio Stations
Smart Grids
IoT Devices

IBM CASCON PEOPLE'S CHOICE AWARD

S. S. Murtaza, A. Hamou-Lhadj, W. Khreich, M. Couture, "TotalADS: Automated Software Anomaly Detection System," In Proc. of the 14th IEEE International Working Conference on Source Code Analysis and Manipulation (SCAM), 2014

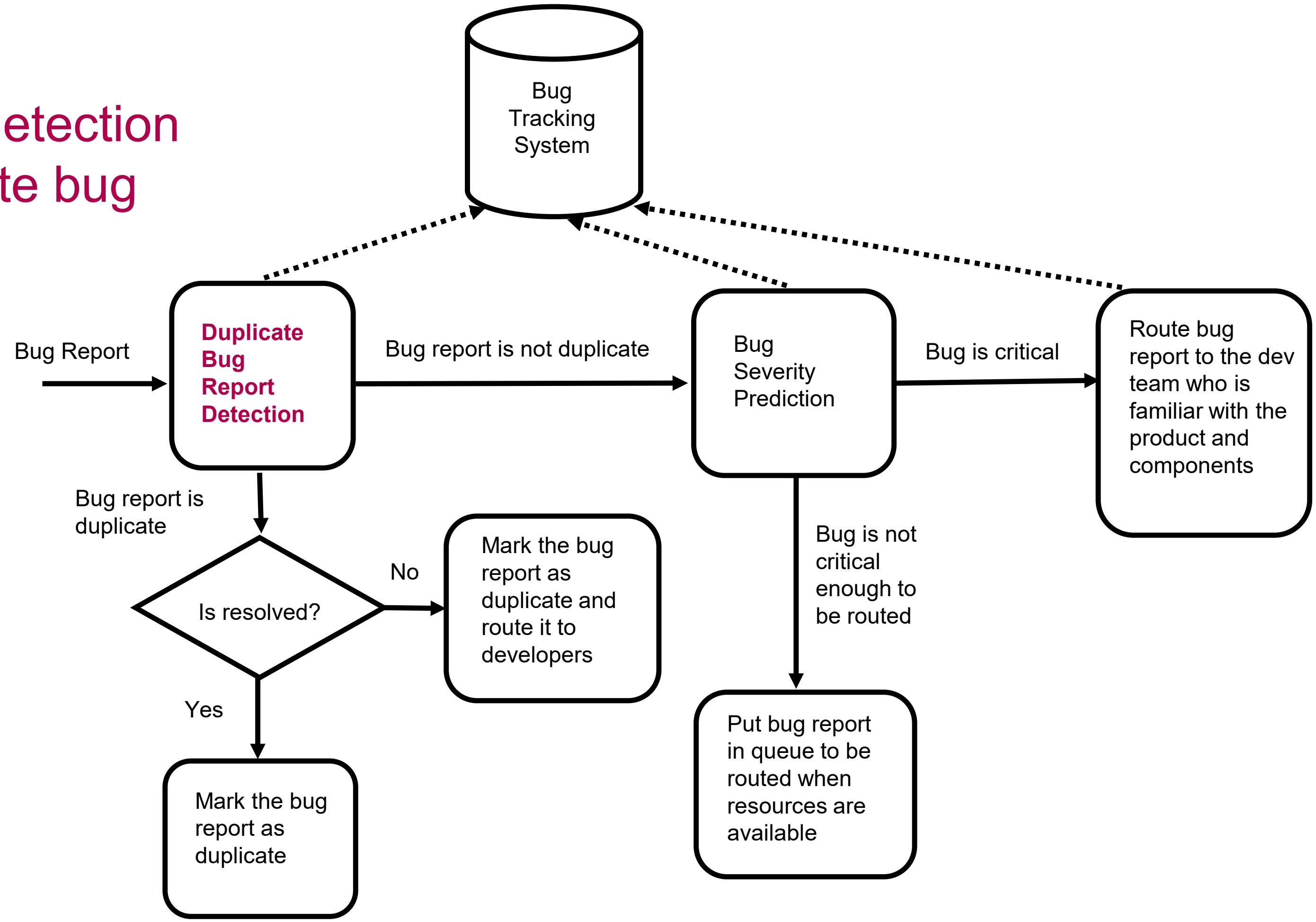
Automatic Crash/Bug Triaging Techniques Using Machine Learning and Stack Traces

- PhD Thesis of Korosh Koochekian Sabor



Automatic Crash Triaging Techniques Using Machine Learning and Stack Traces

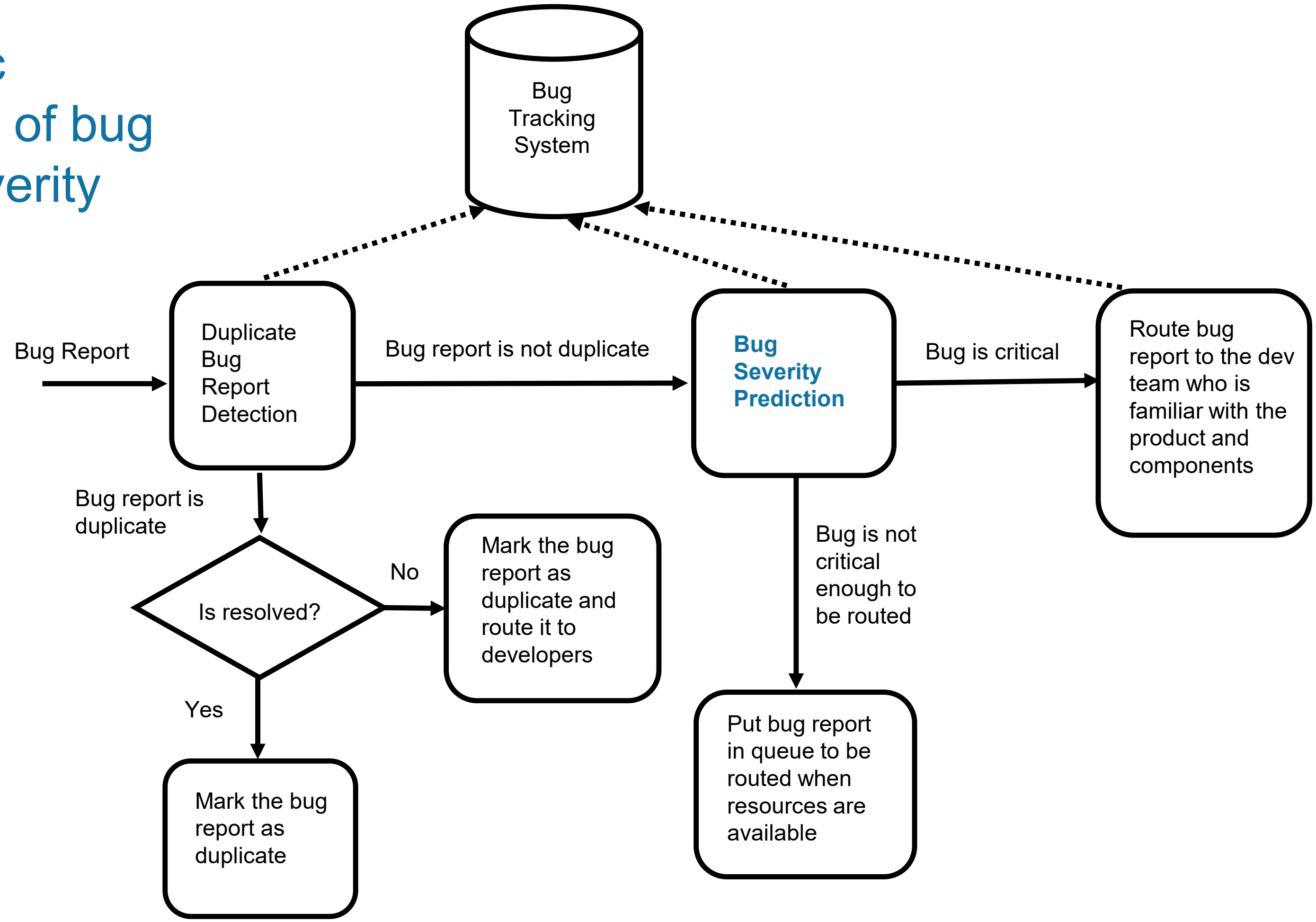
- DURFEX: Efficient detection of duplicate bug reports



Prof. Wahab Hamou-Lhadj (wahab.hamou-lhadj@concordia.ca) - EDiS 2022

Automatic Crash Triaging Techniques Using Machine Learning and Stack Traces

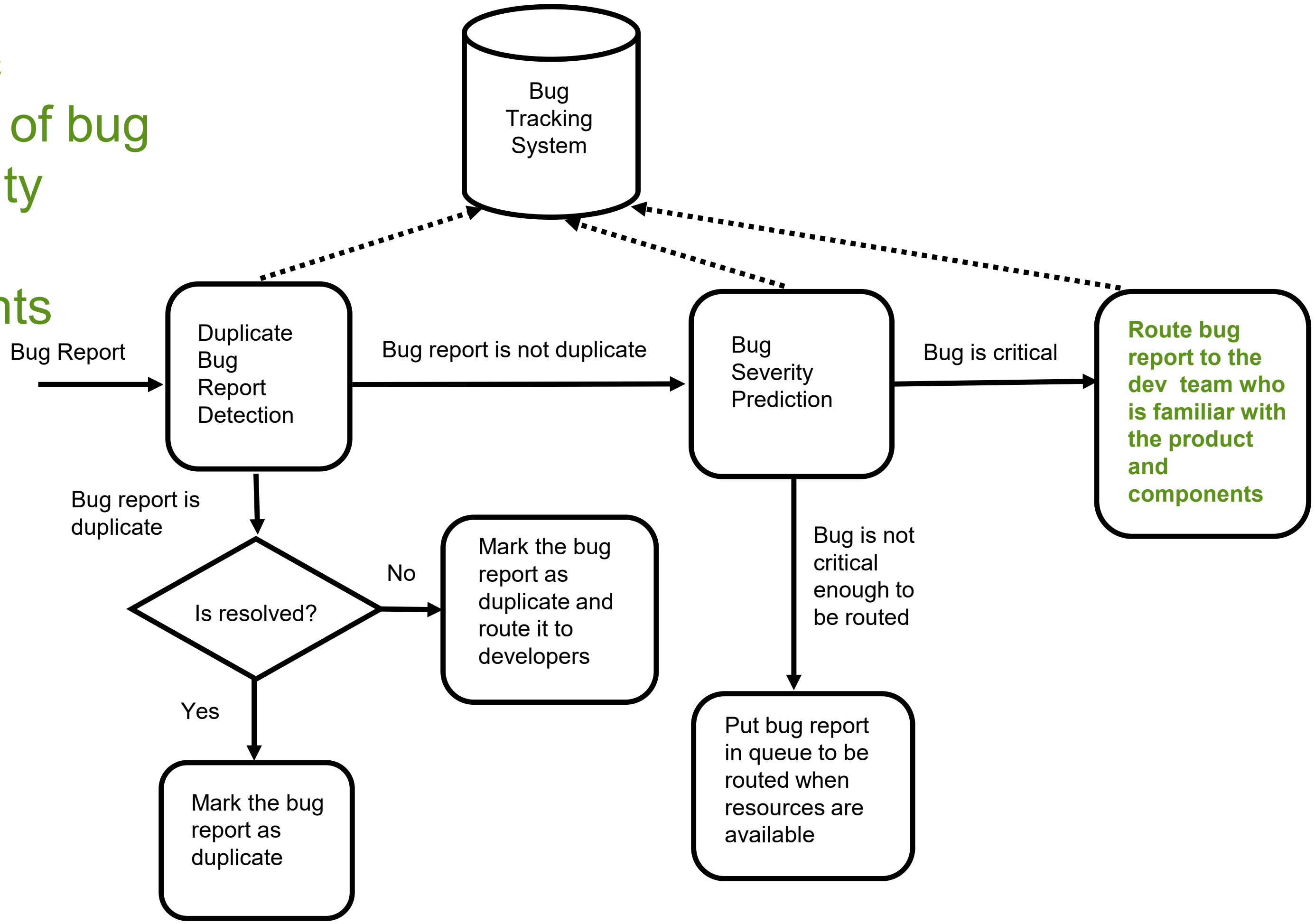
- Automatic prediction of bug report severity



Prof. Wahab Hamou-Lhadj (wahab.hamou-lhadj@concordia.ca) - EDiS 2022

Automatic Crash Triaging Techniques Using Machine Learning and Stack Traces

- Automatic prediction of bug report faulty products components



Characteristics of Logs and Traces

- **Velocity:** the data (in some cases) must be processed in real time
- **Volume:** mountain ranges of historical data
- **Variety:** captured data can be structured or unstructured
- **Veracity:** captured data must be cleaned
- **Value:** not all captured data is useful

Challenges

- **Standards and Best Practices:**
  - Lack of guidelines and best practices for logging, tracing, and profiling
  - Lack of standards for representing logs, traces, and metrics (not the OpenTelemetry initiative)
- **Data Characteristics**
  - Mainly unstructured data
  - Size is a problem
  - Not all data is useful
  - High velocity

Prof. Wahab Hamou-Lhadj (wahab.hamou-lhadj@concordia.ca) - EDiS 2022

Challenges

- **Analytics and Tools:**
  - Mainly descriptive analytics
  - Predictive analytics not fully explored
  - Mainly offline analysis techniques
  - Lack of usable end-to-end observability tools

- **Cost and Management Aspects**
  - Cost vs. benefits not well understood
  - No clear alignment of observability with other initiatives
  - Roles and responsibilities are not well defined

Challenges

- **Analytics and Tools:**
  - Mainly descriptive analytics

There is a need for systematic and engineering approaches to software observability that promote best practices throughout the entire software development lifecycle

- **Cost and Management Aspects**
  - Cost vs. benefits not well understood
  - No clear alignment of observability with other initiatives
  - Roles and responsibilities are not well defined

Observability By Design

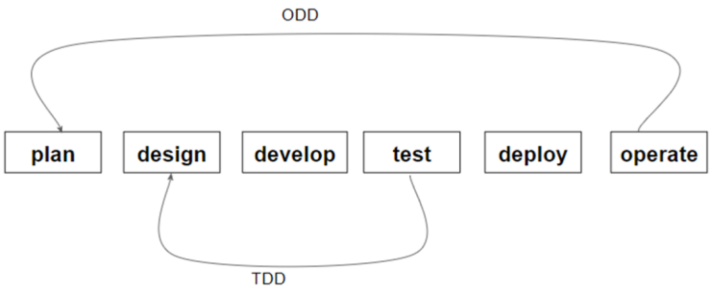
- Bringing observability **to early stages** of the software development lifecycle.

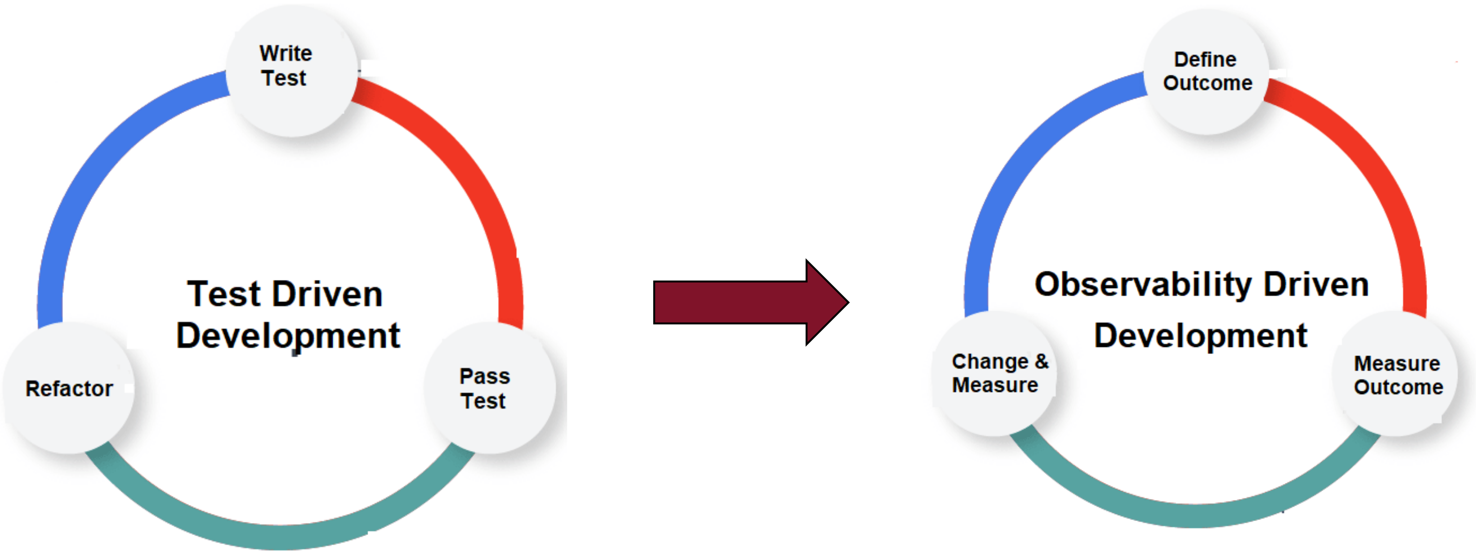
- Defining a set of **observability patterns, best practices, and reusable solutions** to be used as guiding principles for developers.

- A **systematic approach** to tracing, logging and profiling of software systems that considers different phases of the software process.

Observability-Driven Development (ODD)

- Leveraging tools and hands-on developers to observe system state and behavior
  - Interrogating the system, not just setting and measuring thresholds and metrics for it





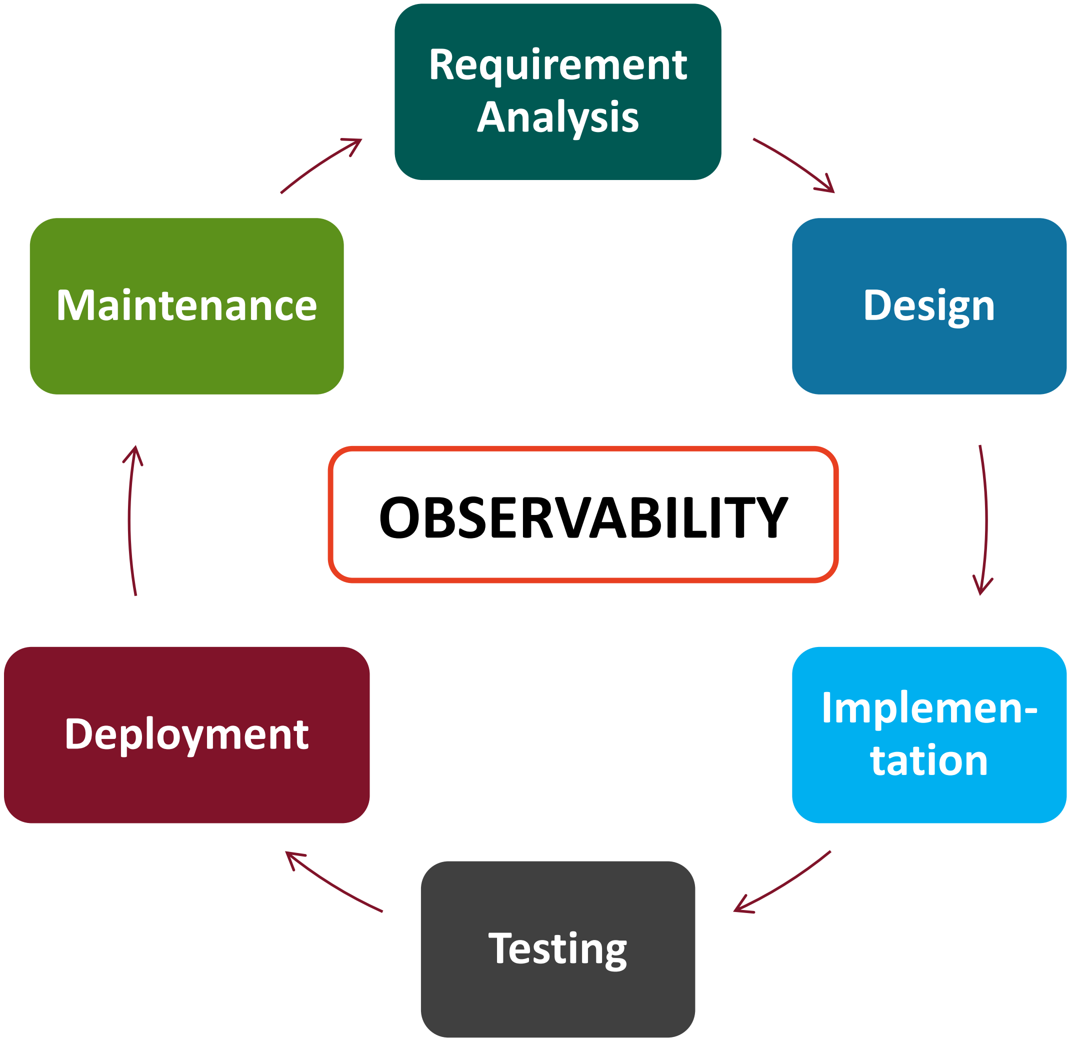
Prof. Wahab Hamou-Lhadj (wahab.hamou-lhadj@concordia.ca) - EDiS 2022

From Telemetry to OpenTelemetry

- Observability is often equated with telemetry
  - "If you have metrics, logs, and traces, then you have Observability"
- Observability is the process of deriving value from telemetry
  - Telemetry is important but not sufficient
- We also need tools to analyze and visualize the telemetry
  - OpenTelemetry

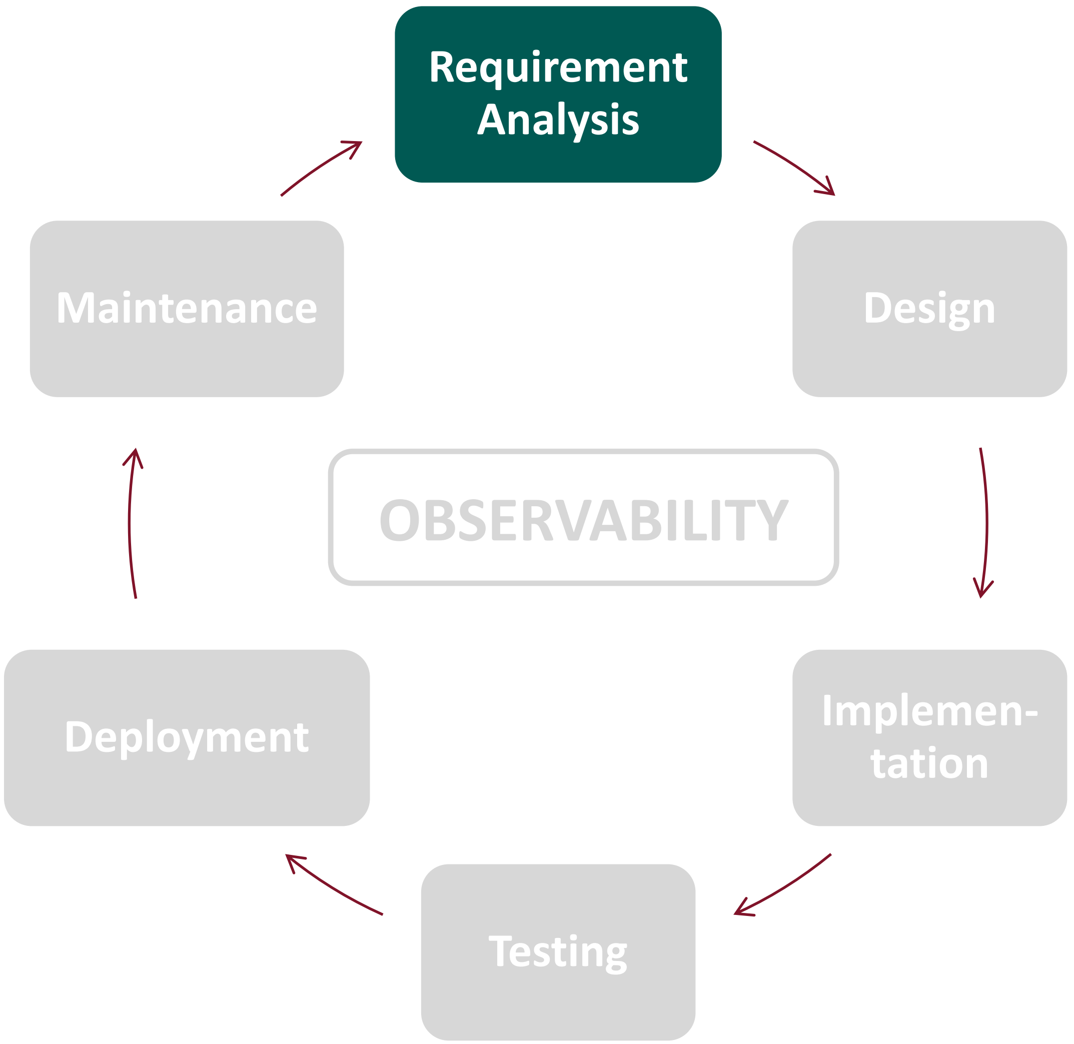
Observability By Design and SDLC

- Bringing observability **<u>to early stages</u>** of the software development lifecycle
- **<u>Cost of observability</u>** can be assessed during project planning



Observability By Design and SDLC

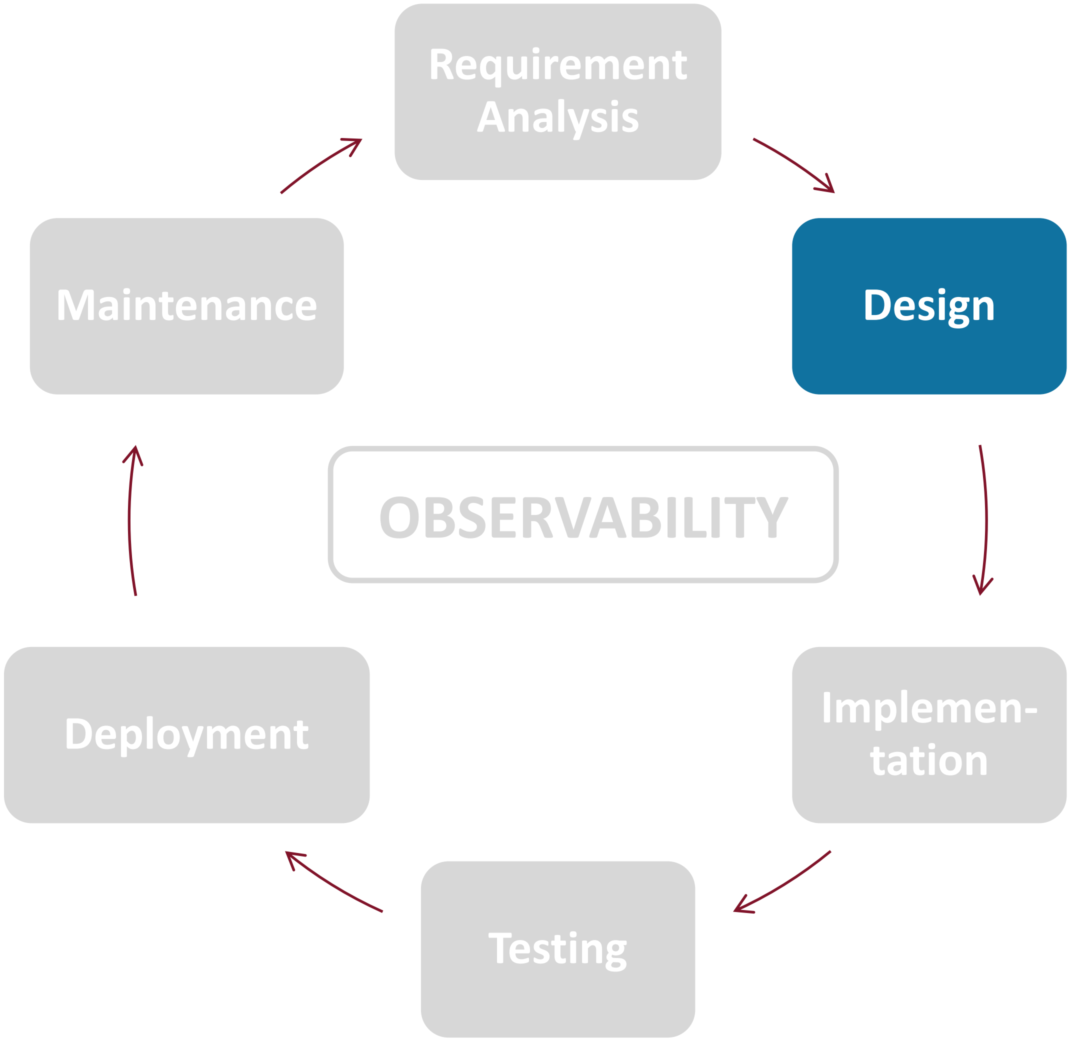
- Observability as a non-functional requirement
- What aspects of system functional requirements should be observable and how?



Prof. Wahab Hamou-Lhadj (wahab.hamou-lhadj@concordia.ca) - EDiS 2022

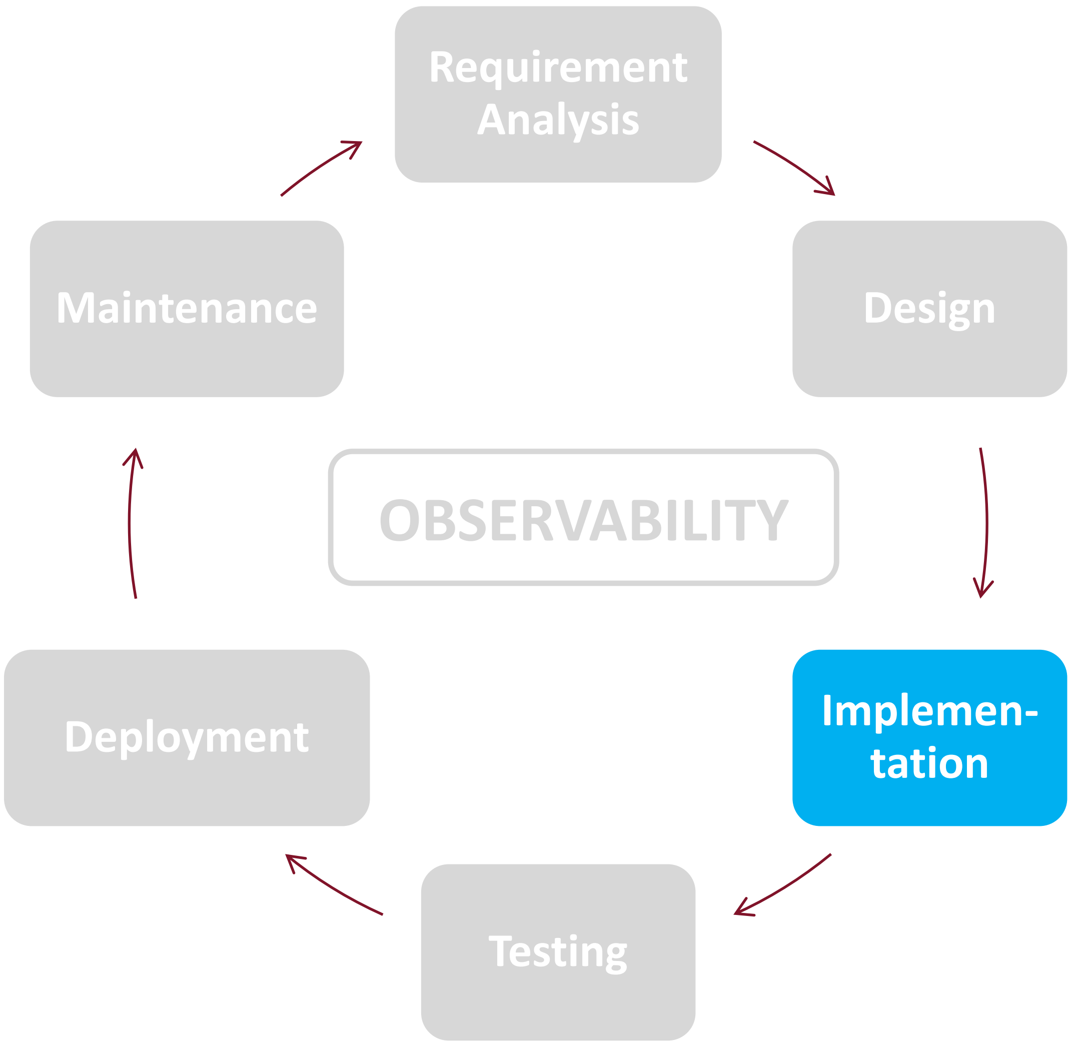
Observability By Design and SDLC

- Support of observability at the architectural level
- Detailed design for observability
- Observability patterns and best practices



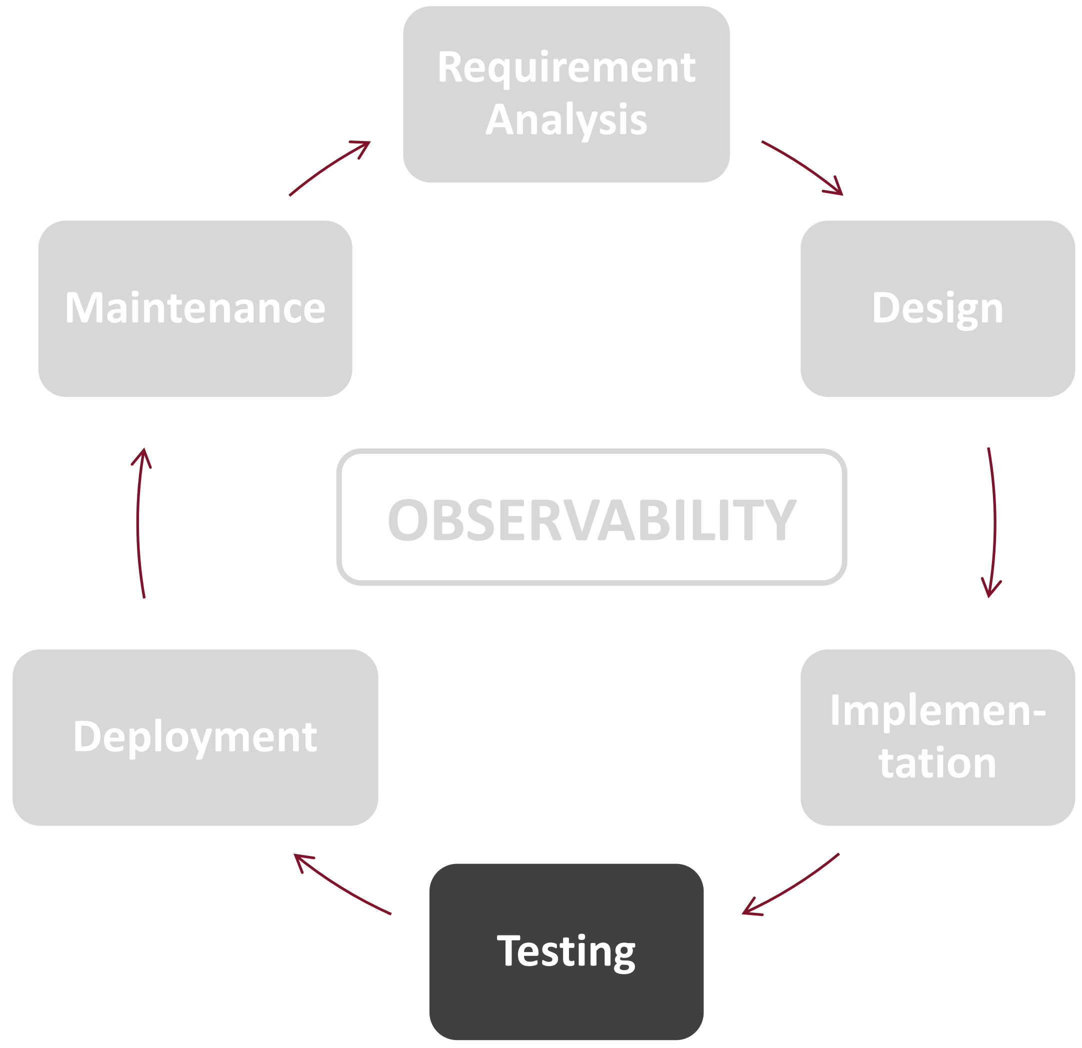
Observability By Design and SDLC

- What, where, and how to log and/or trace?
- Use of libraries and frameworks
- Patterns and best practices



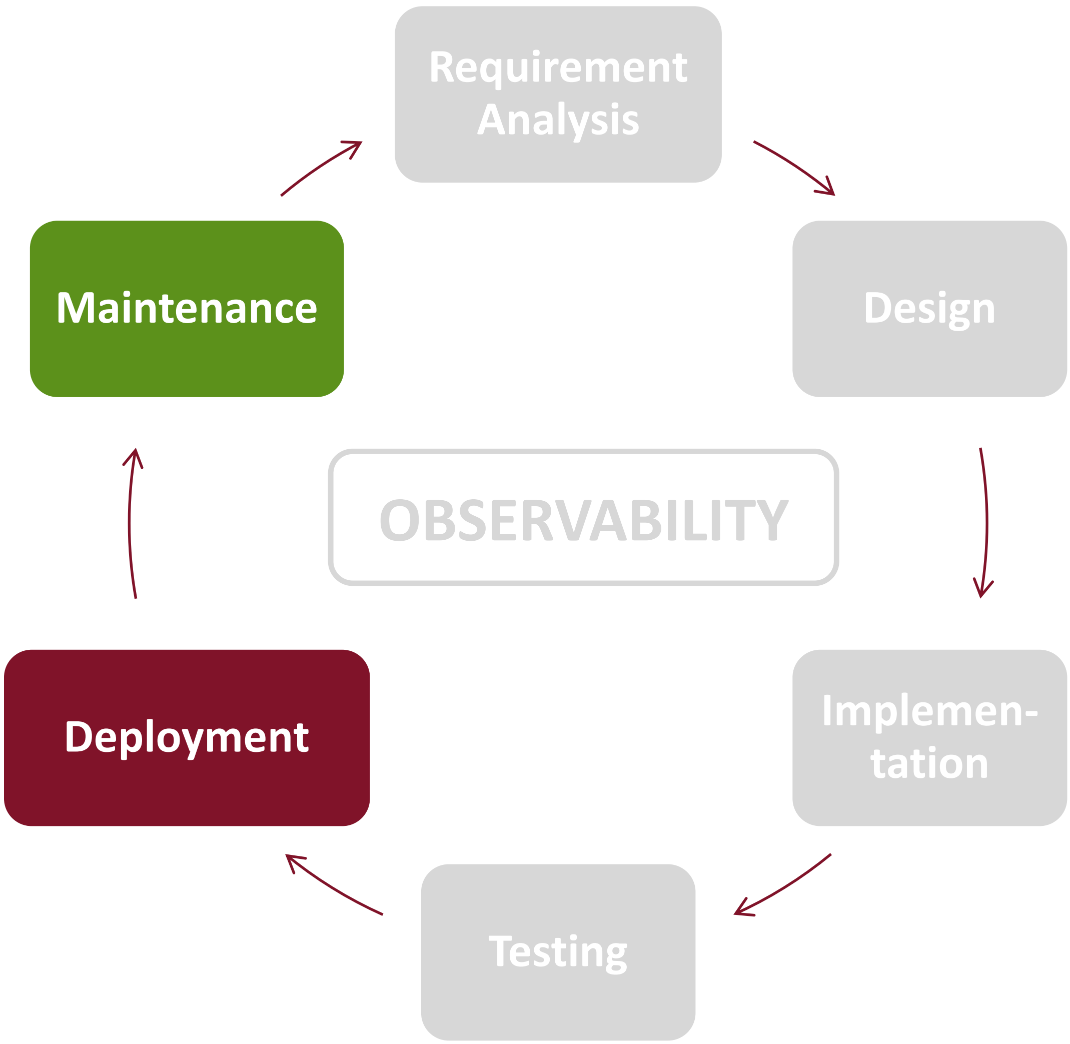
Observability By Design and SDLC

- Testing and inspection strategies for logging/tracing code



Observability By Design and SDLC

- Deployment, configuration, and maintenance aspects of observability code such as updates, performance analysis, testing, persistence, etc.



A Governance Framework for Observability By Design

Goals and objectives, Strategic alignment, KPIs,

Governance

People	Process	Technology
Training Roles & responsibilities (observability specialists)	Process maps Process compliance	AI Big data Tools & platforms

Continuous Improvement	Best Practices	Maturity Level Assessment

Prof. Wahab Hamou-Lhadj (wahab.hamou-lhadj@concordia.ca) - EDiS 2022

Observability Culture

- Observability in action!
- Before and after a problem
- Data-driven decision making
- Educate teams
- Encourage standard tools/techniques
  - Log formatting
  - Metric conventions
- Practice, share success stories, and feedback
- Measure your progress and observer your observability culture!

Conclusion

- Complex systems require sound mechanisms to ensure that they operate as intended and to detect/predict problems.
  - I presented SW system observability as one such mechanism.
  - Observability relies on processing and analyzing operational data

- The current practice is ad hoc and to take full advantage of operational data, we need to move towards systematic approaches for observability.
  - Observability By Design with its governing framework is one possible solution

Contact Information

Wahab Hamou-Lhadj, PhD, ing.

Professor

Dept. of Electrical and Computer Engineering

Gina Cody School of Engineering and Computer Science

Concordia University

wahab.hamou-lhadj@concordia.ca

http://www.ece.concordia.ca/~abdelw