Syllabus for the Course «Observability Technology»

Course volume: 2 ECTS credits (72 hours)
Class workload: 15 hours of lectures, 15 hours of practical classesж independent work(SDS – Self-Directed Study): 42 hours
Semester: 4
Form of midterm assessment: credit
The discipline is implemented in the Master's degree program
"AI-Augmented Digital Systems Engineering"
as part of the professional track
DevOps & SRE (Site Reliability Engineering (engineering of service reliability).

1. General characteristics of the discipline
The discipline is aimed at developing engineering competencies in the field of analysis, diagnostics, and reliability assurance of modern distributed software systems.
Modern digital services are characterized by:

- microservice architecture;
- container infrastructure;
- dynamic scaling;
- high frequency of releases;
- complex distributed interactions of components.

In such conditions, the key factor in the stability of the system is ensuring observability - the ability of the system to provide data that allows its current state to be analyzed and the causes of emerging problems to be identified.
The discipline considers observability as an independent engineering field, including:

- collection of telemetry from software systems;
- monitoring and analysis of metrics;
- centralized logging;
- distributed query tracing;
- incident diagnostics;
- analysis of operational sustainability of digital services.

Large language models (LLM) as a tool for telemetry analysis, operational problem diagnosis and engineering decision support.

2. The place of the discipline in the structure of the educational program
The course is implemented in the 4th semester of the master's program and is the final element of the DevOps & SRE engineering track.
The course logically continues the disciplines:

- "Containerization and Kubernetes " , where skills in deploying and managing container infrastructures are developed ;

03.01 RPD Containerization and Ku …

- "Continuous Integration and Software Delivery (CI/CD) and GitOps " explores the architecture of automation for software development and delivery processes.

03.02 RPD Technologies continuously…
The discipline completes the engineering trajectory of DevOps training :
Infrastructure → Delivery → Observability → Reliability

Thus, the course provides the transition:
from automation of software development and delivery
→ to management of operation and reliability of digital systems.
Mastering the discipline develops the competencies of specialists in the field of:

- DevOps Engineering
- Site Reliability Engineering (SRE)
- Platform Engineering
- Cloud Infrastructure Engineering.

3. Objectives of mastering the discipline

The objectives of mastering the discipline are:

- formation of a systemic understanding of the principles of observability of information systems;
- mastering methods of analyzing telemetry of distributed software systems;
- development of skills in diagnosing operational problems of digital services;
- mastering engineering practices to ensure the reliability of software systems;
- formation of Site Reliability Engineering competencies ;
- development of skills in applying LLM for telemetry analysis and engineering diagnostics.

4. Objectives of the discipline

Within the framework of the discipline, the following tasks are solved:
to study the architecture of observability systems for software systems;
to master methods of collecting telemetry from information systems;
to study methods for monitoring software system metrics;;
to master centralized logging technologies;
to study of distributed query tracing methods;
to master methods of system performance analysis;
to study of service reliability engineering (SRE) practices;
to master methods for diagnosing operational incidents;
development of skills in analyzing operational data;
formation of a culture of using LLM as an engineering analysis tool.

5. Planned learning outcomes

As a result of mastering the discipline, the student must:
Know:

- principles of observability of information systems;
- architecture of monitoring systems;
- methods for collecting telemetry from distributed systems;
- mechanisms for logging and analyzing events;
- distributed query tracing methods;
- digital service reliability indicators (SLI, SLO);
- principles of service reliability engineering (SRE);
- methods for analyzing operational incidents.

Be able to:

- design the observability architecture of software systems;

- analyze service metrics and telemetry;
- diagnose performance issues;
- analyze event logs of distributed systems;
- interpret microservice interaction traces;
- formulate indicators of reliability of digital services;
- Apply LLM to analyze telemetry and diagnose problems.

Be proficient in:

- telemetry monitoring and analysis tools;
- methods for diagnosing operational incidents;
- practices of software systems performance analysis;
- skills in engineering interpretation of telemetry;
- a method for critically examining the results obtained using LLM.

6. Methodological concept of the discipline

The discipline considers the observability of software systems as an integral element of the engineering architecture of modern digital services.

The training is based on an engineering trajectory:

Telemetry → Monitoring → Tracing → Incident Analysis → Reliability Engineering

Each thematic block includes:

1. systems analysis of an engineering problem;
2. architectural design of the solution;
3. practical implementation;
4. engineering reflection;
5. solution analysis using LLM.

6.1. Engineering model of software systems observability

Within the course, observability is considered as a set of interconnected subsystems:

Metrics (metrics)

are quantitative indicators of the state of the system.

Logs (event logs)

- event information about the operation of system components.

Traces (traces)

are data about the sequence of execution of distributed queries.

The integration of these telemetry sources provides the ability to diagnose complex operational problems.

6.2. Relationship between observability and service reliability engineering

Observability is the foundation of Site practices Reliability Engineering.

The course covers:

Service Level Indicators (SLI)

Service Level Objectives (SLO)

Error Budget;

engineering methods for analyzing operational incidents.

7. The role and contribution of LLM in the educational process

Large language models are used as a tool to support students' engineering activities. The LLM serves a number of functions.
LLM as a telemetry analysis tool. The model is used to interpret metrics, logs, and traces of software systems.
LLM as an engineering consultant. The LLM is used to explain the principles of observability technologies and analyze monitoring architectures.
LLM as an incident diagnostic tool. The model is used to analyze operational data and identify potential causes of problems.
The principle of mandatory verification. All conclusions reached using the LLM must be verified by students through engineering analysis and practical experiments.

8. Educational technologies
The following educational technologies are used within the framework of the discipline:
- lectures in the format of engineering master classes;
- practical training in operational data analysis;
- digital laboratory work;
- architectural analysis of observability cases;
- engineering discussions;
- AI- assisted engineering analysis of telemetry;
- project activities.

The educational process is implemented in the AI- augmented model learning , which involves the active use of intelligent tools for analyzing engineering solutions.

9. Final certification
The form of interim assessment is a credit.
The assessment is carried out in the form of an engineering defense of a practical assignment or mini-project related to the analysis of the observability of a software system.
During the defense, the student must:
- present the observability architecture of the system;
- analyze telemetry data (metrics, logs or traces);
- justify the identified causes of system problems;
- propose measures to improve service reliability;
- answer the teacher's questions.

Evaluation criteria
Evaluation is carried out according to the following criteria:
- engineering correctness of the analysis;
- depth of telemetry interpretation;
- validity of conclusions;
- ability to explain architectural solutions;
- correctness of application of LLM.

Thematic schedule for the course

Week	Content	Lectures (hours)	Practice (hours)	SDS (h)
1	**Lecture.** Introduction to Information Systems Observability. The Evolution of Information Systems Operation. DevOps and Service Reliability Engineering Approaches (Site) Reliability Engineering (SRE). Monitoring and observability : differences and relationships. Primary sources of telemetry: metrics , event logs , and traces . Seminar **.** Analyzing the architecture of a microservice application and identifying observability points. **Applying LLM.** Analyzing the system architecture using Large Language Models (LLM) . Students describe the architecture of a microservice system and receive recommendations for selecting monitoring and telemetry parameters. **Independent work (SDS).** Preparing a brief overview of modern information system observability platforms.	2	2	4
2	**Lecture.** Telemetry of distributed information systems (telemetry) in distributed Telemetry collection architecture . Types of metrics and events. Telemetry standards. **Digital lab.** Instrumenting software code for collecting metrics using telemetry standards (e.g., OpenTelemetry). **Applying LLM.** Using LLM to generate and analyze instrumentation code . Students apply the model to explain telemetry configuration and integration of monitoring libraries. **Independent work (SDS).** Analyzing telemetry tool documentation and preparing a brief report.	1	2	4
3	**Lecture.** Metrics and monitoring systems. Application and infrastructure metrics. Time -series data Data). Monitoring system architecture. **Practical session.** Configuring microservice application monitoring and visualizing metrics.	2	2	4

Week	Content	Lectures (hours)	Practice (hours)	SDS (h)
	Applying LLM. Interpreting information system metrics using LLM. Students analyze time series of system metrics and formulate hypotheses about the causes of performance changes. **Independent work (SDS).** Analysis of a set of test system metrics and drawing conclusions.			
4	**Lecture.** Logging and Event Analysis and log Analysis). Structured logging. Centralized logging systems. Correlation of events in distributed systems. **Practical exercise.** Centralizing microservice application event logs and analyzing errors. **Using LLM.** Semantic analysis of event logs. LLM is used to classify log messages and identify recurring error types. **Independent work (SDS).** Analysis of test system event logs and report preparation.	1	1	4
5	**Lecture.** Tracing distributed systems (distributed) tracing). Problems of analyzing microservice interactions. Analysis of service call chains and execution delays. **Digital lab.** Setting up distributed service tracing and analyzing request processing delays. **Using LLM.** Interpreting distributed system traces. LLM is used to analyze service interaction graphs and identify system bottlenecks . **Independent work (SDS).** Analysis of user request traces.	2	2	4
6	**Lecture. Observability of container platforms and** container orchestration systems (Kubernetes) Observability). Container and cluster metrics. Monitoring container services. **Hands-on lab.** Analyzing the state of a Kubernetes cluster and monitoring containerized applications.	1	2	4

Week	Content	Lectures (hours)	Practice (hours)	SDS (h)
	Using LLM. Diagnosing container infrastructure issues using LLM. The model is used to interpret container management system messages and identify the causes of errors. **Independent study (SDS). Exploring** Kubernetes monitoring tools .			
7	**Lecture.** Reliability of digital services and reliability engineering practices (Site Reliability Engineering). Indicators level service (Service Level Indicators, SLI) and goals level Service Level Objectives (SLO). Error budget (error budget). **Seminar.** Developing reliability metrics for a digital service. **Applying LLM.** Using LLM to support SLI and SLO development. Students analyze the service architecture and generate reliability metrics using the LLM assistant. **Independent work (SDS).** Preparing a service reliability model.	2	1	4
8	**Lecture.** Performance analysis of information systems (performance Analysis). Identifying system bottlenecks Resource planning (capacity) Planning). **Practical lesson.** Performance analysis of a microservice application. **Application of LLM.** Diagnosing system performance issues using LLM. The model helps formulate hypotheses about possible causes of performance degradation. **Independent work (SDS).** Preparing a performance analysis report.	2	1	4
9	**Lecture.** Incident Management management) and analysis of failure causes (root cause Postmortem analysis of incidents. Practical **exercise.** Diagnosing a microservice application failure.	1	1	5

Week	Content	Lectures (hours)	Practice (hours)	SDS (h)
	Using LLM. Preparing a postmortem analysis using LLM. The model is used to structure telemetry data and describe the causes of an incident. **Independent work (SDS).** Preparing a postmortem report.			
10	**Lecture.** Observability and intelligent exploitation of information systems (AIOps). Automatic anomaly detection (anomaly detection) and self-healing systems (self-healing) Systems). **Practical session.** Integrating observability tools into continuous integration and delivery pipelines (CI/CD). **Applying LLM.** Analyzing information system telemetry using LLM and discussing operational automation opportunities. **Independent work (SDS).** Preparing a final analytical assignment.	1	1	5