{mthree}

Observability

Site Reliability Engineering

Overview

In this module, we will provide an overview of the concept of observability.

Learning Objectives

- Describe the parts of observability
- Discuss the concept of application performance monitoring
  - Business transactions
  - Component monitoring
- Explain the relationship between service level indicators (SLIs), service level objects (SLOs), and service level agreements (SLAs)

{mthree}

Observability

- More than monitoring
- Three parts
  - Logs
  - Metrics
  - Traces
- Using data from a complex system to infer its internal state
  - Capacity & Performance
  - Satisfaction
  - Expectations

{mthree}

Three Parts of Observability

1. Logs
   - >>> Record of past events
2. Metrics
   - >>> Current data about the system components
3. Traces
   - >>> Capture activity for a business transaction
   - >>> Shows interactivity in complex systems

{mthree}

Log Files

- Automatically created
  - >>> Application
  - >>> Operating system
- Hold information
  - >>> User behaviour
  - >>> Events
- Used for root cause analysis
  - >>> Understand why a metric changed
  - >>> Identify where an event began

{mthree}

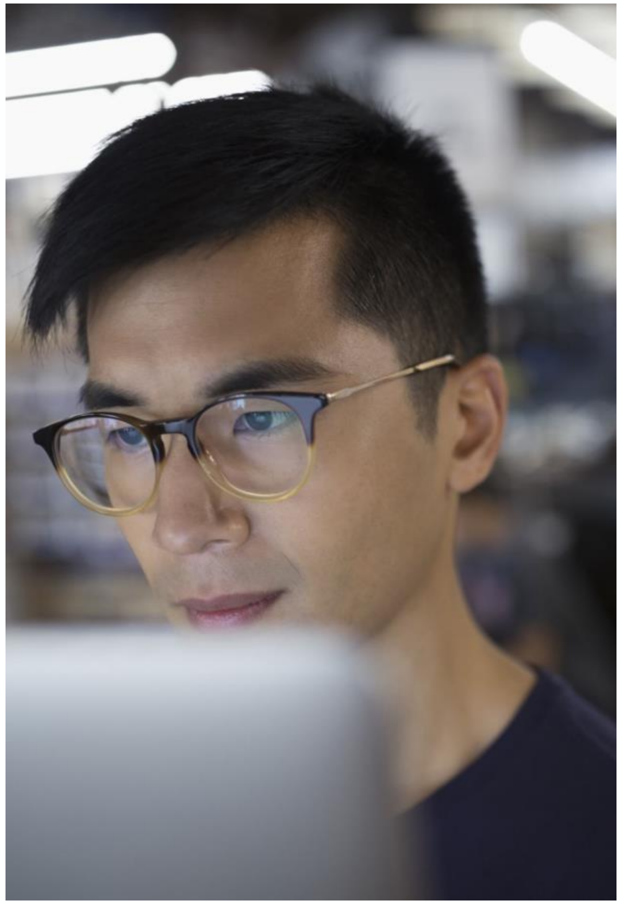
Metrics

- Quantified measurement
  - Used to understand the status of a specific process
  - Often compared to a defined baseline
    - Analyse the system's or process's status
- Trends in metric changes
  - Indication of underlying issue

{mthree}

Trace

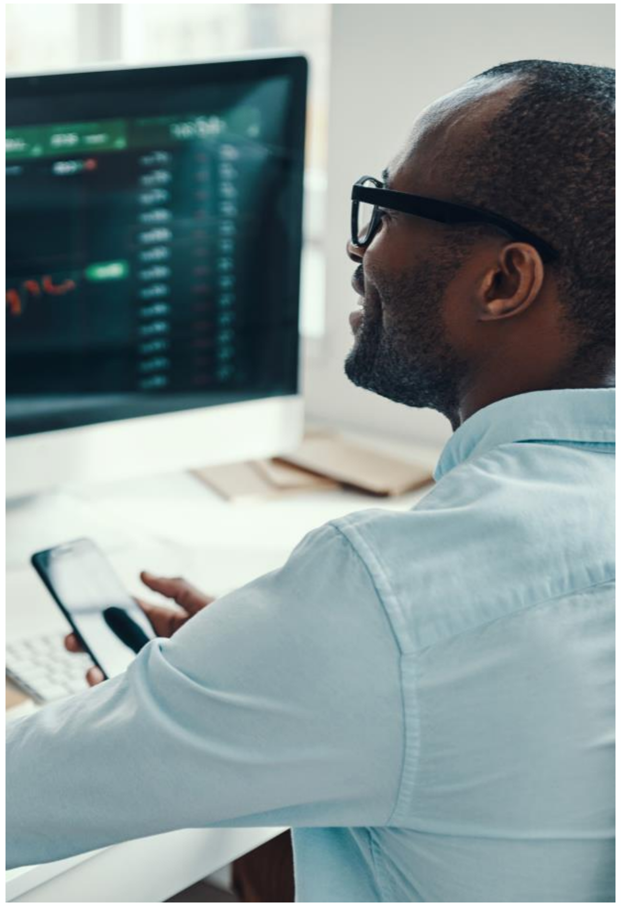
- Complete record of business request
  - Illustrates a complete transaction
  - Captures all the components and services involved
- Contains hundreds of data points that can:
  - Indicate errors
  - Diagnose security threats
  - Detect and isolate component or network issues



{mthree}

Observability Data

- Combining three parts to answer questions
  - Why performance is degrading
  - What dependency behaviors have changed
  - Why this application is failing
  - Where to look for a fix

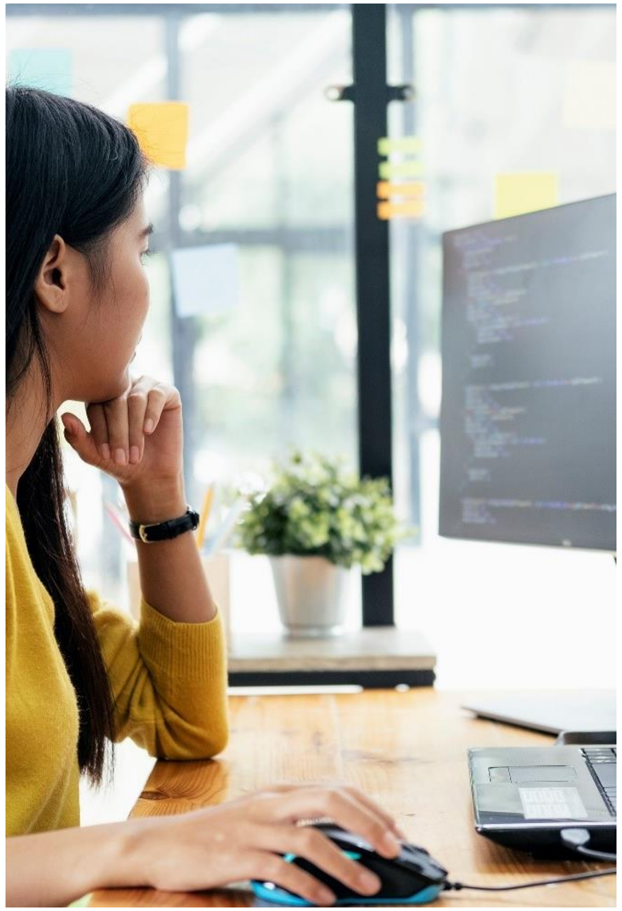


{mthree}

Expectations

↘ Focus on user expectations
  - >>> Delivering more than expected doesn't gain much
  - >>> Failing to deliver will lose a lot

↘ Observability should provide at a glance:
  - >>> Are we out of SLO?
  - >>> Should SLO be adjusted?
  - >>> Where is the problem?



{mthree}

Benefits of Observability

- Overview of complex system
- Faster feature release
- Observe impact of updates
  - Confirmation improvements
  - Identify issues

{mthree}

Application Performance Monitoring

- Purpose of application monitoring
- Components of APM

Purpose of Application Monitoring

- Essential to maintain uninterrupted business processes
- Monitor the overall application
  - Continued availability
  - Appropriate performance
- APM solution provides:
  - Connection of app performance to business outcomes
  - Isolate and fix errors
    - Before they affect the end user
  - Reduce the mean time to repair (MTTR)

{mthree}

Five Most important Elements

1. Runtime application architecture discovery
2. End-user experience monitoring
3. User-defined transaction profiling
4. Component deep-dive monitoring
5. Analytics

Runtime Application Architecture

- Analyses the hardware and software components used
  - Direct execution
  - To communicate with
- Anticipate potential problems
  - Pattern recognition
  - Performance problems

{mthree}

Real User Monitoring

- Real user monitoring enables an organisation to efficiently respond to faults and understand their impact
  - Also referred to as *end-user experience monitoring*
- Gather user-based performance data
  - How well the application is performing (from user point of view)
  - Gauge potential performance problems
- Performance through entire infrastructure
- Details on the analysed client
  - Location
  - Operating system
  - Browser

{mthree}

Two Ways to Track End User Experience

Synthetic Monitoring

- Uses probes and bots
- Simulates an end user to determine problems before the app is opened
- Used to monitor service-level agreements

Agentless monitoring

- Uses data probes
- Analyse network traffic that travels
  - Load balancers
  - Switches

{mthree}

Business Transactions

Focus on specific user interactions

- Recreating them to test and understand the conditions that lead to a performance problem
- AKA - User-defined transaction profiling

Help organizations

- Trace business transaction movement across various components
- Reveal when and where events are occurring
- Optimize performance by identifying bottlenecks

Component Monitoring

- Provides a deeper understanding of the specific elements and pathways
- Also referred to as an application component deep dive
  - Tracking all components of the IT infrastructure
- Extensive, in-depth monitoring
  - All resources and events
    - Analysis of all servers
    - Operating systems
    - Middleware
    - Application components
    - Network components

{mthree}

Analytics and Reporting

- Essential to ensuring the organization receives a good return on investment
- Translating data gathered into information that can be used
  - Define a performance baseline using historical and current data
    - Set expectations for normal app performance
  - Identify areas of improvement
    - Comparing infrastructure changes to performance changes
  - Identify, locate and resolve performance issues
    - Using historical and baseline data
  - Predict and alleviate potential future issues
    - Before the customer notices

Critical APM Metrics

- Web performance monitoring
  - Average response time for end user interactions
  - Identify if speed is affecting app performance
- System metrics impacting app performance
  - CPU usage
  - Disk read/write speeds
  - Memory demands
- Application availability and uptime
  - App is online and available to users
  - Frequently used to determine compliance with SLA
- Request rates
  - Amount of traffic received by the application
  - Identify significant increases, decreases
  - Coinciding users
- Customer satisfaction
  - Compare how customers feel about the app against defined baseline
- Error rates
  - Capture app degrading or fails at the software level
- Number of instances
  - How many server or app instances are running
  - Important for cloud application

{mthree}

Observing Toil

- How do we measure toil success?
- Where do we get the metrics?

{mthree}

Observing Toil

- Toil backlog
  - Should be reducing
  - Should not see the same toil recurring in the backlog
  - Amount of toil being reported by specific people reducing
  - Error budget not decreasing as fast = more reliable system
- Reduced fatigue in the team
- Shorter MTTR
- Toil metrics come from
  - Ticketing and job systems such as Jira and ServiceNow
  - Other systems that gather information about time at work

{mthree}

SLAs vs SLOs vs SLIs

Service level agreements

Formal financial or contractual agreements

Usually with penalties

Service level objectives

- Reflect customer expectations
- Based on customer satisfaction
- Majority of customers happy
  - Always outliers

Service level indicators

Show progress towards SLO

Indicate health of business

Summary Q & A

- Observability
  - Logs
  - Metrics
  - Traces
- Application Performance Monitoring
  - Business Transactions
  - Component Monitoring
- Service level indicators indicate service level objects which meet service level agreements