{mthree}

Monitoring

Site Reliability Engineering

Overview

In this module, we will provide a high-level overview of why we need monitoring and the ways in which we use it.

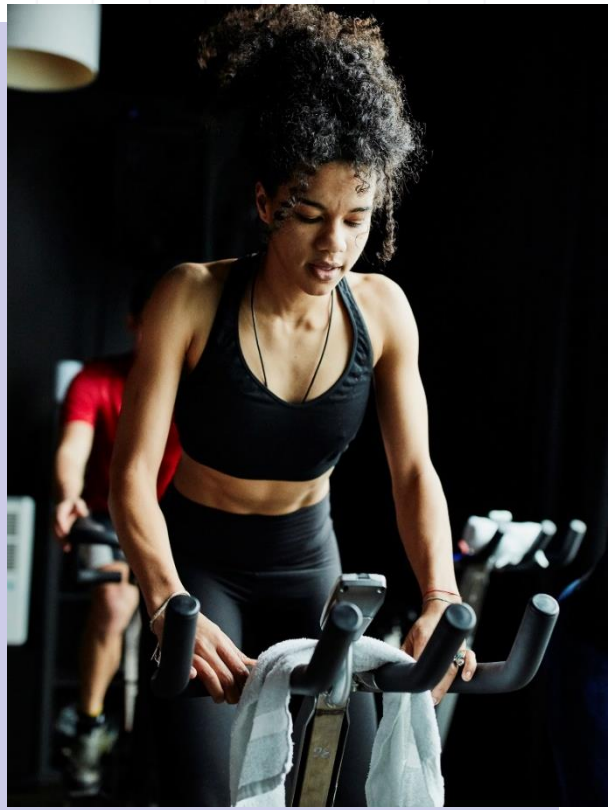
Learning Objectives

- Explain what it means to monitor
- Describe why monitoring is needed and its benefits
- Describe the different layers of monitoring and types
- Explain what distributed monitoring is
- Identify the different methods systems use

What is Monitoring?

- Think of something you do every day.
  - >>> Going for a run or to the gym
  - >>> Walking or driving to a new place
  - >>> Cooking a meal or cake
  - >>> Catching a bus or train

- Do you monitor anything while performing these tasks?



{mthree}

Yes, You Do

- Running or gym
  - >>> Monitor heart rate or blood pressure
  - >>> See if it is getting to the desired target
- Walking or driving to a new
  - >>> Check road signs or names to make sure you are going the right way
- Cooking a meal or cake
  - >>> Set timers
  - >>> Check weights and measures
  - >>> Check that it is cooked inside
- Catching a bus or train
  - >>> Check the platform and arrival time
  - >>> Do I have enough time to get a coffee?

{mthree}

What is Monitoring?

"The collecting, processing, aggregating and displaying of real-time quantitative data about a system."

Rob Ewaschuk, Monitoring Distributed Systems

{mthree}

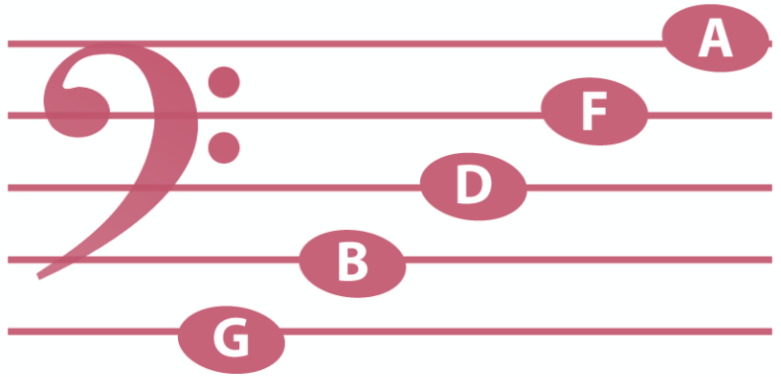
Why Monitoring?

- We can make executive decisions from the data
  - >>> Increase resource
  - >>> Add more storage
  - >>> Design a better process
  - >>> Fix site and system flaws
- Report findings in a timely fashion
  - >>> Dashboards to display useful charts, data and diagrams
  - >>> Alerts to email, SMS or chat
  - >>> Escalate issues quickly
  - >>> Determine if an issue is really occurring
    - ~ e.g., Did my CPU just hit 100%? Is it still there?
- Perform trend analysis through historical data
  - >>> Helps with planning and delivery of reliable systems
  - >>> Foresee potential demand through external events such as political events, or disasters
  - >>> Link to machine learning and AI systems to predict future trends
- Compare behaviours between different versions of applications or system components

{mthree}

Baselines

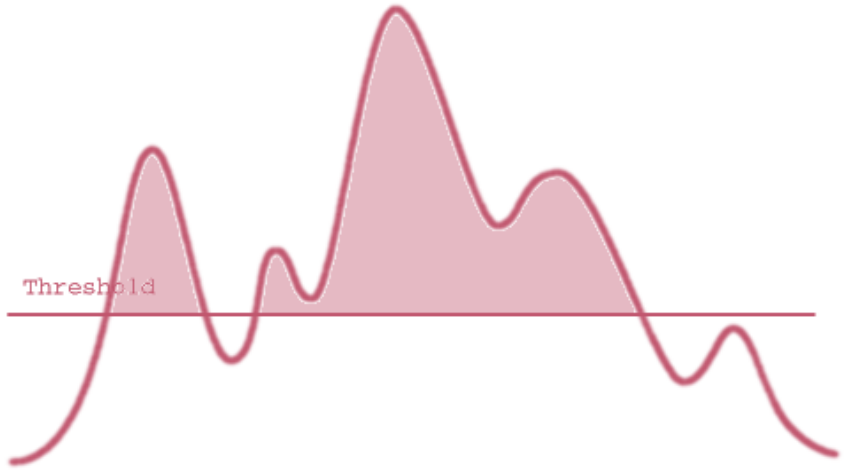
- Give us the starting point of monitoring
  - >>> Normal activity
  - >>> Increased activity
  - >>> Low activity
- Obtaining values for production on new projects
  - >>> Testing as part of Dev
    - ~ Capacity and Performance
    - ~ Playing potential scenarios to see what results you get
    - ~ Recording the outcomes as thresholds
- Using values from previous projects
  - >>> Where we have "like" usage



{mthree}

Thresholds

- Baselines help us set thresholds
- Setting the limits and boundaries for
  - >>> Perfect operation
  - >>> Forewarning of potential problem
  - >>> Critical when the issue is impacting the client or system



{mthree}

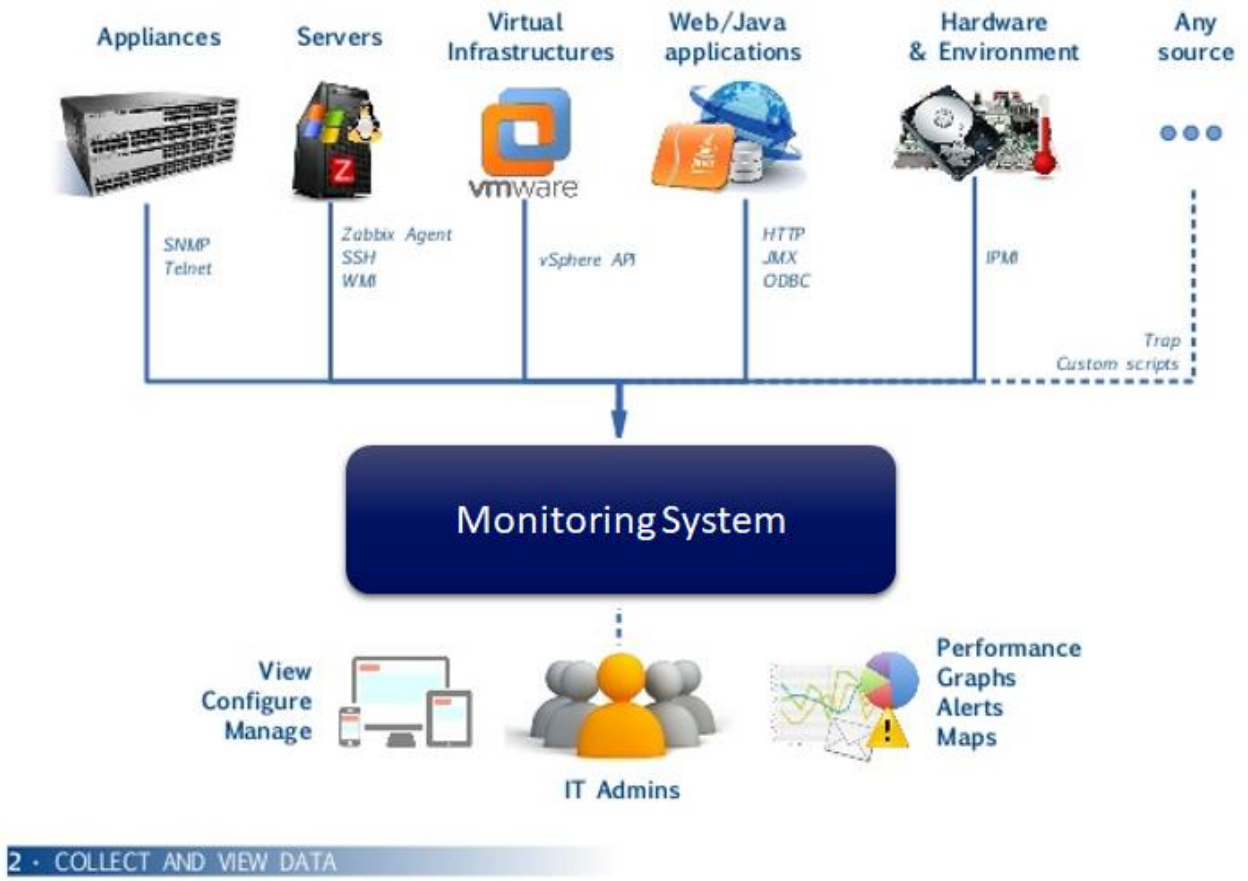
Benefits of Monitoring

Business	Technical/Project	Both
• Reputation • Keeping the user happy • Audit compliance • Discover user trends	• Pre-empt failure • Security threat detection • Early problem detection • Notification and visualization • Performance analysis	• Historical trend analysis and future prediction • Meeting service agreements and objectives • Planning and budgeting

{mthree}

Classic Monitoring

- System resources
  - >>> Hardware components
  - >>> Operating system
- Network components
  - >>> Switches
  - >>> Load balancers
  - >>> Firewalls
  - >>> Routers
- Applications
  - >>> Internals
- Metrics, rules and thresholds
- Reactive



{mthree}

Modern Systems

- Centralized dashboards, collection and alerting
- Gathers
  - >>> Metric data
  - >>> Log file information
  - >>> Either by requesting or receiving
  - >>> Application and infrastructure
- Queries, calculations & trend analysis
- Extensible and pluggable
  - >>> Additional information gathering and calculations
- Low impact agents
  - >>> Gathering data from resources being monitored

{mthree}

Aggregate Metrics

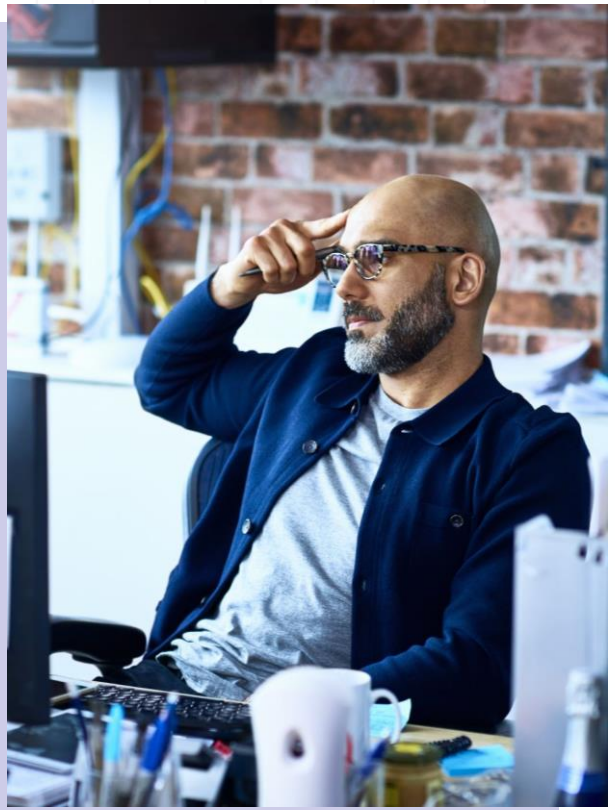
- Aggregates must be
  - >>> Meaningful
  - >>> Relevant
- What time period is relevant?
  - >>> Is an hour too short – will we trigger an alert
  - >>> What's our objective to the client

- Am I measuring in the right place?
  - >>> User latency may be hard to identify
  - >>> Where can I measure latency?
    - ~ Inbound proxy/load balancers
    - ~ Do I record at all network connected points and take the sum/average?
  - >>> What if I cannot pin point a particular request?
    - ~ Do I take the average of all traffic for the period?

{mthree}

Monitoring Production

- Is it enough just to monitor production environment?
- What if the customer was expecting the update today?
- Is our pipeline functioning correctly?
  >>> Are all the agents on line and available
- Is the QA environment set to the correct versions?
  >>> Do we have the correct infrastructure set up?
  >>> Are the tests up to date?
- Are the correct versions in our software repository?
- Is connectivity between Jira and Jenkins responding



{mthree}

Monitoring Automation

- Encourage re-use
- Links to documentation to help resolve issues on alerts
- Segregation of duty
  - >>> System monitoring code
  - >>> Application monitoring code
- System monitoring is common
  - >>> Configurable attributes for common use
  - >>> Infrastructure and platforms responsible for code
- Application monitoring is specific
  - >>> Enable developer to include monitoring in their SCM
  - >>> Code linked to central monitoring service
  - >>> Developer responsible for application monitoring

{mthree}

Rules for Effective Monitoring Management

- As simple as possible
- Avoid piling up the requirements
  - >>> Leads to complex monitoring systems
  - >>> Complexity introduces
    - ~ Differing latency thresholds
    - ~ Different percentiles on different kinds of metrics
    - ~ Specific dashboard components for each type of cause
  - >>> Complexity increases with time
    - ~ Monitoring system becomes fragile, difficult to change, increase in toil
- Design with simplicity in mind
  - >>> Rules to catch real incidents – simple, predictable, reliable
  - >>> Rarely used data collection and aggregation should be removed
  - >>> Rarely used features on dashboards should be removed
- SRE toil reduction methods should be applied

{mthree}

Summary

- SREs to be familiar with a service's monitoring system and features
- SREs require monitoring to define users experience of service
- Need to know
  - >>> Where to look
  - >>> How to identify abnormal behaviour
  - >>> How to find the information they need during an emergency
- Combine some source of metrics and logging in your monitoring strategy
  - >>> Exact mix is highly context-dependent
  - >>> Collect metrics that serve a particular purpose
    - ~ Better capacity planning
    - ~ Assist in debugging
    - ~ Directly notify you of problems

{mthree}

{mthree}

References

- Mushero, S. (2019, August 2). Push vs. Pull Monitoring Configs. Retrieved from https://steve-mushero.medium.com/push-vs-pull-configs-for-monitoring-c541eaf9e927
- Ewaschuck, R. (2016). Monitoring Distributed Systems. In Beyer, B. et al., Site Reliability Engineering. From https://www.oreilly.com/library/view/site-reliability-engineering/9781491929117/ch06.html

{mthree}