Verteilte Systeme

Prof. Dr. Martin Becke

CaDS - HAW Hamburg

Version 0.9

Inhalt

Monitoring

Idee

- Management der Komplexität
- Unterstützung der Fehlertoleranz
- Automatisierung und Abstraktion (Auch für Fachfremde)

Monitoring

Schlüsselfunktionen

- Performance-Optimierung
- Fehlerdiagnose
- Systemintegrität

Monitoring

Metrikarten

- Performance-Metriken
- Ressourcen-Metriken
- Fehlermetriken

Monitoring

Verteilte Metrikarten

- Netzwerk-Metriken
- Datenkonsistenz-Metriken
- Lastverteilungs-Metriken
- Skalierbarkeits-Metriken

Monitoring

Metriken

- Erfassen
- Speichern
- Analysieren
- Visualisieren

Monitoring

Metriken nutzen

- Logging (Basis System)
- Tracing (Basis Transaktion)
- Herausforderung Synchronisation

Monitoring

Ebenen

- System-Monitoring
- Netzwerk-Monitoring
- Datenfluss-Monitoring

Monitoring

Aktuelle Themen

- Automatisiertes Monitoring
- KI-gesteuerte Analyse
- Verteilte Tracing-Technologien
- Cloud-basiertes Monitoring

Monitoring

Modelle

- Benchmarks und Performance
- Anomalie- und Fehlererkennung
- Vorhersagende Analytik
- Formale Methoden

Monitoring

Weitere Anforderungen : Banken

- Datenschutz und Datensicherheit
- Auditierung
- Authentifizierung und Zugriffskontrolle auch für das Monitoring

Monitoring

Self-Healing

- Fehler oder Defekte erkennen und automatisch beheben
- Ohne menschliches Eingreifen.
- Einfluss auf Fehlertoleranz

Monitoring

Entwickler

- Wichtig für Fehlerbehebung
- Mit in die Architektur einplanen!

Inhalt

Debugging

Distributed Debugging

- Speziell dafür entwickelt
- Funktionen für das Setzen von Breakpoints, das Durchlaufen von Code und das Überwachen des Systemzustands
- Nicht selten ein Bundle

Debugging

Minimum

- Tracing-Tools
- Log-Aggregation-Tools
- Fehlerüberwachung und Crash-Reporting-Tools
- Hoher Verbund zum Monitoring

Debugging

Beispiel Tracing : Jaeger

- Trace-Erzeugung
- Context Propagation
- Span-Sammlung und Speicherung
- Trace-Analyse und Visualisierung

Der Einsatz von Jaeger erfordert eine gewisse Anstrengung

Debugging

Techniken

- Snapshot : Zustand eines Systems für die Analyse festhalten
- Replay-Techniken : Wiederholung durch Aufzeichnung und Snapshot

Der Einsatz von Jaeger erfordert eine gewisse Anstrengung

Debugging

Werkzeuge für Techniken

- Record and Replay Frameworks (Beispiel : Mozilla's rr)
- Distributed Snapshot Frameworks (Beispiel : Flockport)
- Event Logging und Distributed Tracing Tools (Beispiel : Zipkin)

Der Einsatz von Jaeger erfordert eine gewisse Anstrengung

Debugging

Fehlerinjektion

- Absichtlich Fehler
- Beispiele Gremlin oder Chaos Monkey
- Nicht selten Live Betrieb

Debugging

Fehlerinjektion

- Hardware-Fehlerinjektion
- Software-Fehlerinjektion
- Netzwerk-Fehlerinjektion

Debugging

Häufigste Problem-Sektoren

- Performance
- Konfiguration
- Netzwerk

Debugging

Werkzeuge für Problem-Sektoren

- Profiling-Werkzeuge (Performance)
- Konfigurationsmanagement-Werkzeuge wie Ansible oder Puppet (Kontrolle über Config)
- Netzwerksimulationswerkzeuge

Inhalt

Deployment

Idee

- Ist ein Prozess
- Transfer von Entwicklung auf Produktionssystem
- Verschiedene Deployment-Strategie denkbar

Deployment

Deployment-Strategien Beispiele

- Rolling Deployment
- Blue-Green Deployment
- Canary Deployment

Deployment

Sicherheitsrelevante Anwendung

- Extensive Tests
- Redundanz
- Notfallpläne
- Sicherheitsüberlegungen

Deployment

CD/CI

- Continuous Delivery (CD)
  Fokus Bereitstellung Produktion
- Continuous Integration (CI)
  Fokus Entwicklung

Deployment

Beispiele Werkzeuge

- Jenkins
- Ansible
- GitLab CI/CD
- Spinnaker
- Terraform
- Chef
- Beispiele aus der Cloud :
  AWS CloudFormation und Google Cloud Deployment Manager

Deployment

Infrastruktur als Code (IaC)

- Hoch automatisierte und konsistente Methode zur Bereitstellung und Verwaltung von Infrastruktur
- Infrastrukturen auf die gleiche Weise behandeln wie Anwendungscode
- Versionen verwalten, Tests durchführen, Wiederverwendung und Modularität fördern und kontinuierliche Integration und Bereitstellung (CI/CD)

Inhalt

Fazit

Bauen wir VS mit

- Lastausgleich oder Redundanz
- Mechanismen und Fehlerbehebung und Wiederherstellung
- Effektives Transaktionsmanagement
- Angemessenen Synchronisationselementen
- Ausreichenden Sicherheitsmechanismen
- Intelligenten Ressourcenmanagement
- Angemessene Kommunikation
- Kapslung der wesentlichen Anforderungen in einer Middleware
- ...