Verteilte Systeme – Übung

Evaluation von Systemen

Sommersemester 2022

Laura Lawniczak, Tobias Distler

Friedrich-Alexander-Universität Erlangen-Nürnberg
Lehrstuhl Informatik 4 (Verteilte Systeme und Betriebssysteme)
www4.cs.fau.de

Lehrstuhl für Verteilte Systeme
und Betriebssysteme

FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG

TECHNISCHE FAKULTÄT

Überblick

Evaluation von Systemen

Evaluation von Systemen

- Analyse des eigenen Systems
  - Leistungsfähigkeit
  - Antwortzeit
  - Durchsatz
  - Ressourcenverbrauch
  - Dienstgüte-Garantien
  - ...
- Vergleich mit anderen Systemen
  - Wie verhalten sich die unterschiedlichen Systeme in bestimmten Situationen?
  - Wo liegen die jeweiligen Stärken und Schwächen?
  - Ab welchen Punkten ist das eine bzw. das andere System besser?
  - ...

Unterscheidung

- Simulation
  - Messungen an einem Simulator, der das gewünschte Verhalten so gut wie möglich imitiert
  - \+ Oftmals einfach zu realisieren
  - \- Ergebnisse spiegeln eventuell nicht exakt die Realität wider
- Evaluation
  - Messungen an einem konkreten System (bzw. Prototyp)
  - \- Im Allgemeinen aufwändiger zu realisieren
  - \+ Ergebnisse entstammen einem realistischen Szenario

→ Evaluationen besitzen mehr Aussagekraft als Simulationen

Mögliche Probleme

- Nicht bzw. schwer zu evaluierende Merkmale
  - Eingeschränkte Quantifizierungsmöglichkeiten
  - Merkmal ist nicht isoliert messbar
  - ...
- Fehlende Vergleichsmöglichkeiten
  - Eigene Variante ist konkurrenzlos [Eher selten der Fall.]
  - Andere Varianten besitzen abweichenden Fokus
  - ...
- Beispiel: Effizienz vs. Fehlertoleranz
  - Aussagen über das Ausmaß von Fehlertoleranz können oft nicht durch Messergebnisse gestützt werden, stattdessen: oberflächliche Beschreibung (z. B. Anzahl und Art tolerierbarer Fehler)
  - Fehlertoleranz ist (fast) immer mit Effizienzeinbußen verbunden

  → Der durch den Einsatz fehlertoleranter Systeme erreichbare Gewinn lässt sich schlechter evaluieren als die damit verbundenen Verluste

Vorgehensweise

- Vorbereitung
  - Konzipierung der Evaluationsszenarien
  - Dokumentation der Evaluationsszenarien, -umgebung
  - Formulierung einer Erwartungshaltung
- Durchführung
  - Abarbeitung der vorbereiteten Szenarien
  - Sammlung der Messergebnisse
- Nachbereitung
  - Aufbereitung der Ergebnisse (z. B. in Diagrammen)
  - Beschreibung der Ergebnisse (textuell)
  - Interpretation der Resultate
  - Abgleich der Resultate mit der Erwartungshaltung

Messungen

- Mögliche Fehlerquellen
  - Existenz einer Aufwärmphase mit atypischen Systemeigenschaften
  - Verfälschung von Messungen durch unbeabsichtigtes Caching
  - Erhöhte Netzwerklatenzen aufgrund außergewöhnlicher Lastsituationen
  - Verzögerungen durch Log- bzw. Debug-Ausgaben
  - Beeinflussung des Systems durch die Messung selbst
  - ...

- Maßnahmen zur Kompensation
  - Messungen später beginnen (nicht bereits ab dem Zeitpunkt 0)
  - Messungen mehrfach durchführen
  - Verwendung von externen Messgeräten/-programmen
  - Geschickte Wahl der Messgrößen, z. B. CPU-Zyklen statt Zeit
  - Passende Wahl der Analysegrößen bei der Nachbereitung, z. B. Median vs. arithmetisches Mittel

Zeitmessung in Java

- Verfügbare Methoden (`java.lang.System`)
  - Aktuelle Zeit in Millisekunden auf Basis der Systemzeit

    ```
    public static long currentTimeMillis();
    ```

  - Aktuelle Zeit in Nanosekunden auf Basis präziser(er) Zähler des Betriebssystems

    ```
    public static long nanoTime();
    ```

- Hinweise
  - Beide Methoden verwenden die Zeitmessung des Betriebssystems
  - Methoden brauchen selbst Zeit zur Ausführung

→ Die versprochene Granularität wird (eventuell) nicht erreicht!

> "This method provides nanosecond precision, but **not necessarily nanosecond resolution** [...] - no guarantees are made except that the resolution is at least as good as that of currentTimeMillis()."
>
> "Differences in **successive calls that span greater than approximately 292 years** (2^{63} nanoseconds) will **not correctly compute elapsed time** due to numerical overflow."

[Auszug aus der Java-Dokumentation zu `System.nanoTime()`]