Praktikum Architektur von Informationssystemen

Sommersemester 2014 – Aufgabenblatt 3

Slobodanka Sersik <slobodanka@sersik.de>

Aufgabe 5: Redundantes MPS

Nachdem ihr Geschäft kontinuierlich gewachsen ist, müssen Sie das MPS robust gegen Ausfall gestalten. Sie entscheiden sich daher, das komplette MPS-System redundant auszulegen, und somit das Pattern „**Aktive Redundanz**“ zu implementieren. Die beteiligten Elemente sind auf dem folgenden Skizzen schematisch dargestellt:

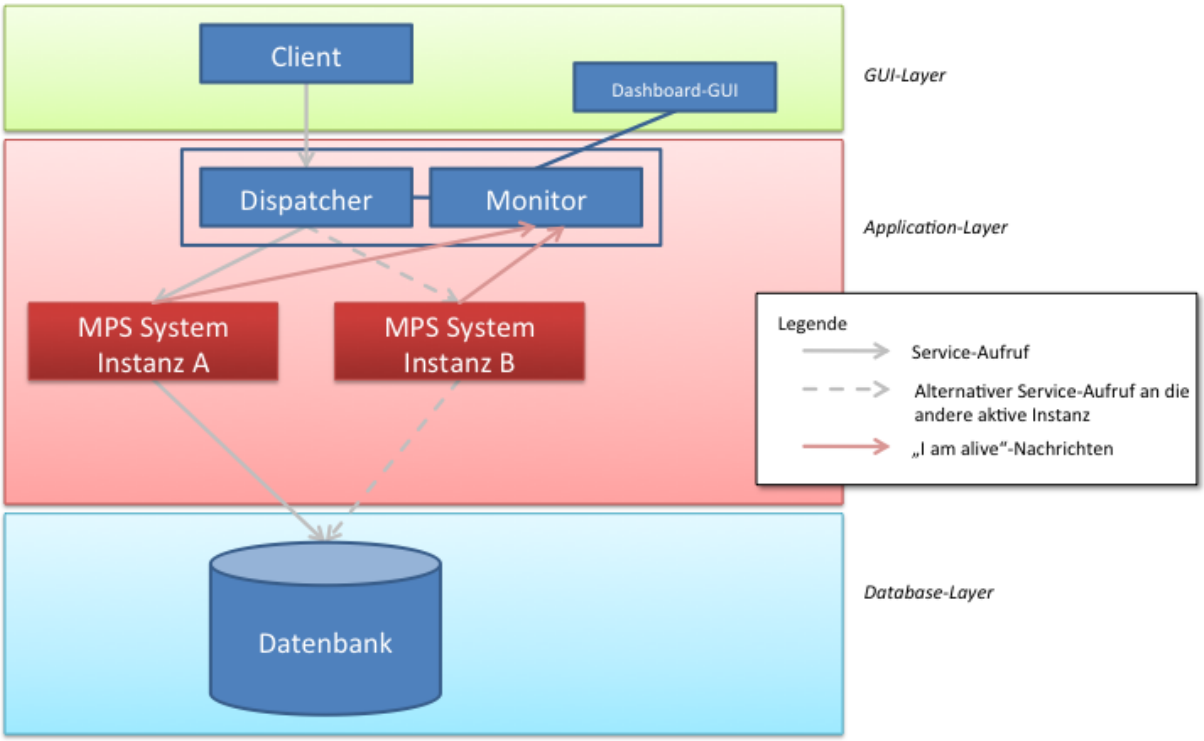

Abbildung 1: Bausteine und Verteilung

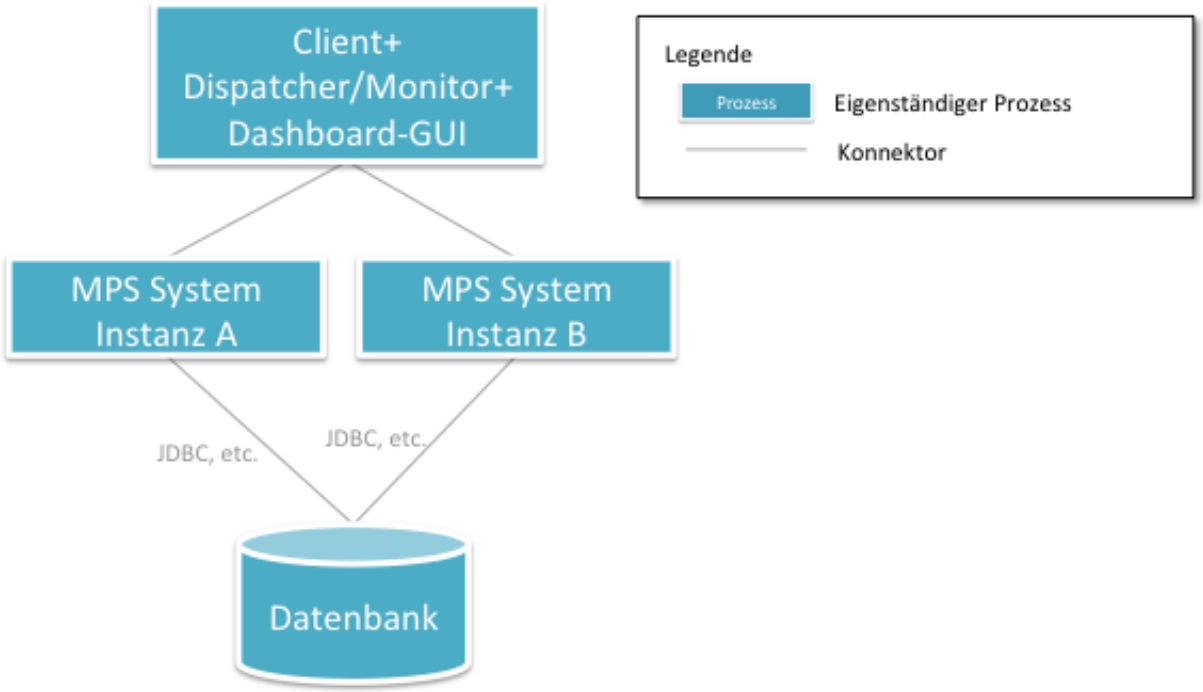

Abbildung 2: Prozesse

Der Softwareprozess, der die MPS-Logik implementiert, wird doppelt ausgelegt und je eine aktiver Serverprozess läuft auf einem separaten Rechner. Da das Datenbanksystem als sehr zuverlässig gilt, wird dieses nicht zusätzlich redundant ausgelegt, sondern nur eine einzige Instanz verwendet. Der Ablauf einer Auftragsstellung und -abwicklung durch den Klienten läuft wie folgt:

- Der Client stellt alle Anfragen (Funktionsaufrufe) an einen zentralen Dispatcherprozess.
- Der Dispatcherprozess leitet die Anfrage an **eine** der beiden aktiven Serverprozesse zur Bearbeitung weiter. Die Auswahl erfolgt „Round-Robin“.
- Eine Monitor-Komponente überwacht die aktiven Serverprozesse und teilt dem Dispatcher-Prozess den Ausfall bzw. eine Wiederverfügbarkeit eines Serverprozesses mit. Hierzu empfängt der Monitor sogenannte „*I am alive*“-Nachrichten von den aktiven Serverprozessen. Bleiben diese Nachrichten für einen bestimmten (konfigurierbaren) Zeitraum aus, geht der Monitor davon aus, dass dieser Serverprozess momentan nicht zur Verfügung steht.
- Über eine „Dashboard-GUI“ kann das MPS-Betriebsteam jederzeit den Verfügbarkeitszustand der Serverprozesse ansehen (als „Ampel-Darstellung“). Des Weiteren kann das Team auch manuell die Serverprozesse offline und wieder online schalten. Außerdem sehen sie die Anzahl der durch die jeweiligen Serverprozesse bislang verarbeiteten Serviceanfragen und die jeweilige „Uptime“ und „Downtime“.

Zusätzliche Hinweise

- **Ergänzen** Sie Ihre bisherige Fachlogik um die **Angebotsverwaltung** (Erstellung, Umwandlung in einen Auftrag, etc.)
- Zur Verbindung Client(s)+Dispatcher+Dashboard-GUI↔MPS-Serverprozesse benötigen Sie einen prozessübergreifend arbeitenden **Konnektor**.

- **Testen** Sie Ihr System sowohl durch manuelles offline/online-Schalten der Serverprozesse über die Dashboard-GUI, als auch durch „Abschießen“ und Neustarten der Serverprozesse auf den jeweiligen Rechnern. Beobachten Sie dabei die Status auf Ihrem Dashboard.
- Achten sie auf dem ACID-Prinzip Ihrer DB Transaktionen
- Nutzen Sie ggf. bestehende Redundanz-Implementierungen als Vorbild (z.B. OpenContrail, Appache ZooKeeper)

Präsentation beim Praktikumstermin

Halten Sie beim zugeordneten Praktikumstermin im Team einen Vortrag, in dem Sie folgendes zeigen:

- Die Architektur (Bausteinsicht, Laufzeitsicht, Verteilungssicht) der Dispatcher- und Monitorfunktionalität für das in der Spezifikation beschriebene Szenario als UML-Diagramme.
- Codeteile, die Ihre Dispatcher- und Monitorfunktionalität umsetzen.
- Eine Live-Demonstration Ihrer Testszenarien, inkl. offline/online-Schalten der Serverprozesse.

Die **Präsentation und der Code** sind zum Praktikumstermin per email (slobodanka@sersik.de) abzugeben.

Zusatzaufgaben (*)

a) Erweitern Sie den Dispatcher und Monitor derart, dass beliebig viele Serverprozesse verwendet werden können. Diese sollen durch das Betriebsteam über das Dashboard an- bzw. abgemeldet werden können.

b) Entwerfen und implementieren einen Load-Balancer in Ihrem Dispatcher. Dazu soll der Dispatcher für einen Serviceaufruf denjenigen Serverprozess auswählen, der auf einem Rechner mit der geringsten CPU-Last läuft. Hierzu müssen die aktiven Serverprozesse periodisch Informationen über die aktuelle Last an den Dispatcher senden. Die Last des jeweiligen Serverprozesses soll in der Dashboard-GUI dargestellt werden.