VSDB

Space Based Computing

Ausarbeitung

Daniel Dimitrijevic Thomas Traxler

13. Dezember 2013

5AHITT

Inhaltsverzeichnis

1 Erklärung

1.1 Wofür steht Space-Based Computing?

Space-Based Computing (fortan SBC) hat seine Ursprünge im parallel programming, und stellt dabei hauptsächlich ein Datenorientiertes Modell zur Koordination dar. Wie aus dem Namen eindeutig hervorgeht handelt es sich hierbei um ein Modell welches auf 'Spaces' basiert. Ein Space ist hierbei gleichbedeutend mit einem (logischen) Ort auf welchem Daten von mehreren Komponenten geteilt verwendet werden. Diese Komponenten können im einfachsten Fall 'write', 'read' und 'take' Aktionen ausführen. Ein Write steht hierfür für das zur Verfügung Stellen eines neuen Datenteil an alle anderen Komponenten in diesem Space, ein Read für das Lesen von Daten ohne diese zu entfernen und ein Take für das Lesen und entfernen einer Datei aus dem Space, auch destructive read genannt. Im einfachsten Fall hat ein Space nun das Datenmodell eines Tuples.
Eine Hauptanforderung die an einen Space hier nun gestellt wird ist das persistente Aufbewahren aller Daten die sich in ihm befinden, auch bei Systemausfällen. Wie das SBC modelliert und implementiert wird kann sehr stark variieren, je denn gegebenen Anforderungen und Wünschen entsprechend. Was hierbei vor allem variiert ist die Zahl der Geräte auf dem der Space implementiert ist und die Zahl der Geräte die auf diesen Space zugreifen, diese Zahlen sind prinzipiell, wenn nicht durch den Anwendungsfall anders umgesetzt, voneinander unabhängig und befinden sich jeweils im Bereich von 1 bis n.
Das SBC-System stellt hierbei eine logische zentrale Einheit dar welche nicht spezifiziert wo sie genau überall Physikalisch vorhanden ist. Der Ort wo eine Datei schlussendlich wirklich physikalisch abgespeichert wird kann dabei nach verschiedensten Methoden ausgewählt werden, in der simpelsten Form wird das selbe Prinzip wie bei Tuples angewendet oder es werden andere Prinzipien verwendet wie FIFO, LIFO, keys, geo-coordinates oder noch kompliziertere, auch das ist vom gegebenen Anwendungsfall abhängig.

1.1.1 Unterschied zwischen SBC und Cloudcomputing

Um nun der Verwechslung von SBC und Cloudcomputing vor zu Beugen sie hier nun gesagt, zu aller erst behandelt SBC lediglich die Daten und Cloudcomputing stellt noch viele weitere Ressourcen (wie zB. Rechenleistung) zur Verfügung. Auch zu Cloud-Storrage kann man abgrenzen, Cloud-Storrage behandelt nicht nur die interne Abspeicherung der Daten sondern auch vor allem wie diese dem Benutzer zur Verfügung gestellt und abstrahiert werden, bei SBC geht es hauptsächlich um die interene Verteilung und Kommunikation der Komponenten untereinander und stellt dabei einen Datenkanal zwischen

den Komponenten des Spaces dar beziehungsweise zur Verfügung.

1.2 Einsatzbereiche

SBC wird vor allem immer dort verwendet wo Daten verteilt abgespeichert werden sollen. Dies kann auf verschiedenste Arte durchgeführt werden um so 'Bottlenecks' zu schließen. SBC bietet durch die Verteilung des Spaces auf mehrere Komponenten grundlegend immer mehr zur Verfügung stehenden Speicherplatz, wodurch man je nach Implementierung verschiedene Vorteile gewinnen kann. Man kann diesen Speicherplatz beispielsweise direkt logisch zur Verfügung stellen und so den logisch zur Verfügung stehenden Speicherplatz erhöhen, man kann die notwendige Kommunikation auf einer Komponente verringern (durch Aufteilung) und man kann für Redundanz sorgen. Die Vorteile lassen sich hierbei auch annähernd beliebig Kombinieren wobei zu beachten ist, das 2 Kombinierte Vorteile auch bei bester Umsetzung so gut wie immer einen dritten Vorteil verringern oder noch andere Nachteile nach sich ziehen. Redundanz schlägt sich Beispielsweise immer auf den zur Verfügung stehenden Speicherplatz nieder, was durch erhöhte Kommunikation zwischen den Komponenten jedoch wieder verringert werden kann.
Ein Beispiel für einen weiteren Vorteil wäre zum Beispiel die Möglichkeit an jeder Komponente Zugang zum ganzen Space zu haben, was in einem Master-Client System erhöhte Kommunikation zum Master mit sich bringt, außerdem müssen die Clients in diesem Fall auch durchgehend die Möglichkeit haben zumindest einen Master ansprechen zu können. Dieser Vorteil stellt sich automatisch ein wenn man auf ein Peer-to-Peer System wechselt, wobei auch oftmals stark belastete Master-Komponenten nicht mehr notwendig sind aber in der Regel wieder mehr Kommunikation zwischen den Komponenten notwendig ist.

2 Grundlegende Prinzipien

2.1 Space-Based Computing Paradigmen

2.1.1 Tuple Spaces

2.2 Mapping

2.3 EAI

3 Im genaueren betrachtet

4 Namhafte Implementierungen

5 Conclusio

6 Quellen