tgm
Die Schule der Technik

JUST DO IT
HÖHERE ABTEILUNG FÜR INFORMATIONSTECHNOLOGIE

Systemtechnik Theorie

xHIT 2024/25, Gruppe A

Protokoll Zlabinger 5DHIT

Theorieprotokoll

Christof Zlabinger 5DHIT

3. Oktober 2024

Bewertung:		Version:	1.0
Betreuer:	Michael Borko	Begonnen:	05.09.2024
		Beendet:	XX.XX.XXXX

Inhaltsverzeichnis

1 Verschluesselung **3**

2 BUS **3**

3 Konsistenz in Datenbanken **3**
3.1 Konsistenz . . . 3
3.1.1 Einheitlich . . . 3
3.1.2 Regeln (Constraints) . . . 3
3.1.3 Deterministisch . . . 3

4 Transaktionskonzepte **3**
4.1 Isolation Levels (psql) . . . 4
4.2 Durability . . . 4
4.3 Geschwindigkeit . . . 4

5 Build prozess **5**
5.1 Prozess . . . 5

6 CAP-Theorie **5**
6.1 CAP . . . 5

7 Resources **6**
7.1 Pico . . . 6

1 Verschluesselung

Symetrisch mittels einem Key
Asymetrische als verstaerkung mittels public/private key
Checksums z.B.: Sha256, Sha512, md5, …
Bei veraenderung komplett anderer Hash
Es kann trozedem sein dass 2 verschiedene daten gleichen Hash haben
Bei Sha256 groessere wahrscheinlichkeit als bei Sha512 weil weniger bits

2 BUS

SPI I2C BUS
BUS controller um collisions zu vermeiden
Wenn collisions -> Beide teilnehmer warten eine zufaellige Zeit

3 Konsistenz in Datenbanken

3.1 Konsistenz

3.1.1 Einheitlich

Datentyp Der Datentyp gibt an wie die Daten zu interpretieren sind und wie gross dieser ist. Fuer Konsistenz ist es wichtig dass die Interpretation gleich oder mapbar ist.

SQL subsprachen Data Definition Language Data Manipulation Language Data Query Language

3.1.2 Regeln (Constraints)

z.B: NotNull, Unique Helfen bei Konsistenz koesten aber Zeit (Alles ueber 0.5s lade Zeit ist lange)
CRUD = Create Read Update Delete

3.1.3 Deterministisch

Computer sind deterministisch gebaut/programmiert wesshalb es eigendlich nicht moeglich ist zufaellige Zahlen zu generieren. Desshalb wird PRNG (Pseudo Randon Number Generator) genutzt. Dieser verwendet Sensoren, deren Stand nicht vorhergesehen werden kann, wie z.B.: CPU temperatur, Fan speed, ...

4 Transaktionskonzepte

ACID = Atomacity Consistency Isolation Durability

Atomar Atomar bedeutet, dass eine Transaktion entweder ganz durchgefuehrt wird oder garnicht.

CAP CAP = Consistency Availability Partitiontolerant Es koennen hoechstens zwei dieser Ziele gleichzeitig erfuellt sein. Bei ACID muessen alle fier Ziele erfuellt sein.

Damit Daten konsitent sind muessen sie am Anfang und am Ende konsitent sein aber mussen bzw. koennen in der Mitte nicht konsitent sein.

4.1 Isolation Levels (psql)

Isolation Level	Diry Read	Nonrepeatable Read	Phantom Read	Serialization Anomaly
Read uncommited (Deadlocks moeglich)	Allowed, but not in PG	Possible	Possible	Possible
Read commited	Not possible	Possible	Possible	Possible
Repeatable read	Not possible	Not possible	Allowed, but not in PG	Possible
Serializable	Not possible	Not possible	Not possible	Not Possible

Es gibt einen Unterschied zwischen Lesen zum anzeigen und Lesen zum schreiben. Beim Lesen zum schreiben muessen die gelesenen Daten gesperrt werden.

4.2 Durability

Dauerhaftes Spechern der Daten. Wird sichergestellt dadurch dass die veraenderten Daten welche im RAM sind gespechert werden, dann auf die DB uebernommen werden und dann ueberprueft wird ob diese korrekt gespeichert werden koennen. Falls dies nicht der fall ist wird ein rollback durchgefuehrt.

4.3 Geschwindigkeit

ACID ist am sichersten aber ist auch am langsamsten. BASE = Basically Available Soft-state Eventually consistant

BA: Atomates speichern der Anfang in einer Queue und deren sequenzielle ausfuehrung. S: Nach einer gewissen Zeit in der queue werden die Anfragen getimeouted. E: Wenn alle Anfragen bearbeitet wurden ist die Datenbank konsitent.

Jede Anfrage kann von BASE mit der Hilfe von ACID bearbeitet werden.

5 Build prozess

.h/.hpp files (Header files) beinhalten die Struktur des dazugehoerigen .c/.cpp files. Der inhalt der header files wird vom pre processor in das C/C++ file kopiert. Da C nicht objekt orientiert ist gibt es kein method overloading.

5.1 Prozess

Das .h/.hpp file wird vom pre processor in das .c/.cpp file kopiert. Der compiler verwandelt dieses dann in .o/.obj files. Der Assembler verwaltet das startup.a/.s file woraus der Linker dann ausfuehrbaren code macht welche mittels des Flash tools geflashed werden.

6 CAP-Theorie

Server = Diener

RPC, RMI, gRPC sind remote prozedure call implementationen welche fuer mittelware verwendet werden koennen. MVC = Model View Controller MVVM wird fuer web based implementations verwendet. DAO = Data Acess Object welches ueberprueft ob bestimmte Daten korrekt sind.

6.1 CAP

Consistency, Availability wurden bereits gemacht Partition tolerance bedeutet das sub-systeme eigenstaendig funktionieren und die Daten nach einem Ausfall wieder miteinander syncen. Es sind immer nur 2 der 3 CAP Ziele gleichzeitig moeglich.

7 Timer, Delay, FPU

7.1 Cpu Cycles

Ein delay kann mittels zaehlen von CPU Cycles hergestellt werden. CPU clock speed / f/2

7.2 Timer

Timers haben ein Limit wie klein der delay sein kann wodurch ein offset entstehen kann der dann haendisch ausgebessert werden muss.

8 MCP2515

CAN Controller CAN = Controler Area Network

9 Resources

9.1 Pico

Pico sdk: https://datasheets.raspberrypi.com/pico/raspberry-pi-pico-c-sdk.pdf CAN Controller: https://elearning.tgm.ac
$Stand - Alone - CAN - Controller - with - SPI -$ 20001801$J.pdf$