Odolnost proti poruchám

Přednášky z Distribuovaných systémů

Základní koncepce

Spolehlivost zahrnuje:

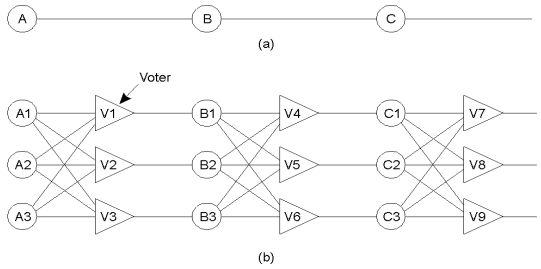
- Dostupnost
- Spolehlivost
- Bezpečnost
- Udržovatelnost

Modely chyb

Type of failure	Description
Crash failure	A server halts, but is working correctly until it halts
Omission failure *Receive omission* *Send omission*	A server fails to respond to incoming requests A server fails to receive incoming messages A server fails to send messages
Timing failure	A server's response lies outside the specified time interval
Response failure *Value failure* *State transition failure*	The server's response is incorrect The value of the response is wrong The server deviates from the correct flow of control
Arbitrary failure	A server may produce arbitrary responses at arbitrary times

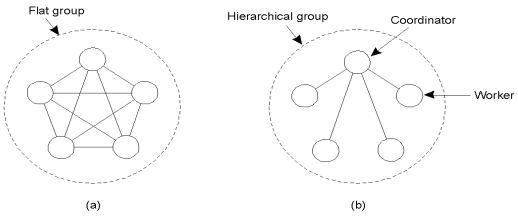
Odlišné typy chyb.

Maskování chyb a redundance



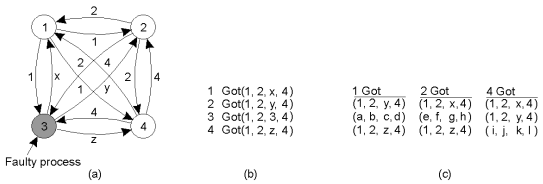
Trojnásobná modulární redundance.

Jednotné a hierarchické skupiny



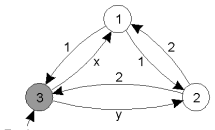
a) Komunikace v jednotné (prosté) skupině.
b) Komunikace v jednoduché hierarchické skupině.

Dohoda v systémech s poruchami (1)



Problém Byzantinských generálů pro tři věrné a jednoho zrádce.
a) Generálové oznamují sílu svých baterií (v jednotkách 1000 vojáků).
b) Vektory vytvořené generály na základě (a).
c) Vektory které každý generál obdrží ve kroku 3.

Dohoda v systémech s poruchami (2)



(a)

1 Got(1, 2, x)
2 Got(1, 2, y)
3 Got(1, 2, 3)

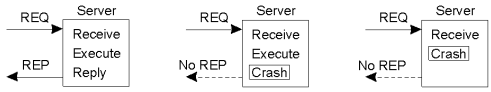
(b)

1 Got	2 Got
(1, 2, y)	(1, 2, x)
(a, b, c)	(d, e, f)

(c)

Totéž jako na předchozím obrázku, ale nyní s 2 věrnými generály a jedním zrádcem.

Ztracené požadavky, zhroucení systému (1)



Server při komunikaci klient-server:

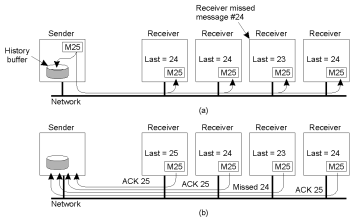
a) Normální případ
b) Výpadek po provedení
c) Výpadek před provedením

Zhroucení systému (2)

Client	Server					
	Strategy M -> P			Strategy P -> M		
Reissue strategy	MPC	MC(P)	C(MP)	PMC	PC(M)	C(PM)
Always	DUP	OK	OK	DUP	DUP	OK
Never	OK	ZERO	ZERO	OK	OK	ZERO
Only when ACKed	DUP	OK	ZERO	DUP	OK	ZERO
Only when not ACKed	OK	ZERO	OK	OK	DUP	OK

Různé strategií chování klienta a serveru při výpadku serveru.

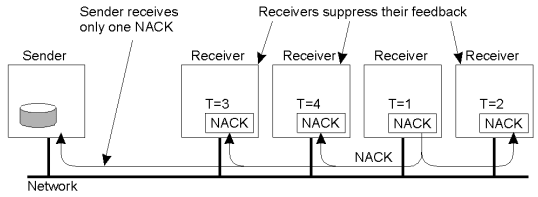
Základní schémata spolehlivé skupinové komunikace



Jednoduché řešení k zajištění spolehlivého skupinového přenosu jestliže jsou všichni příjemci známi a za předpokladu že nechybují:

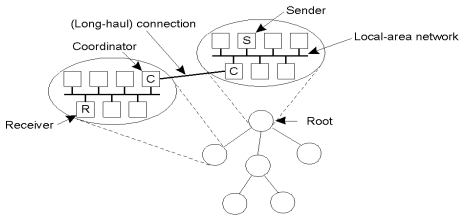
a) Přenos zprávy
b) Zpětná vazba

Nehierarchické řízení odezvy



Několik příjemců má naplánováno opakované odeslání požadavku, ale první odeslání vede k potlačení ostatních.

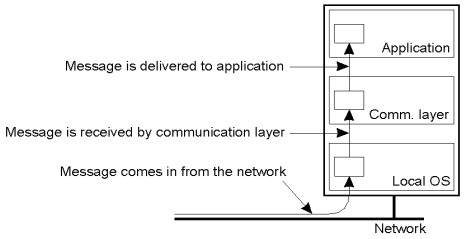
Hierarchické řízení odezvy



Podstata hierarchického spolehlivého skupinového vysílání.

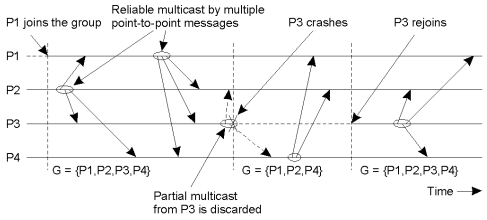
a) Každý lokální koordinátor forwarduje zprávu svým potomkům.
b) Lokální koordinátor zpracovává požadavky na opakované vysílání.

Virtuální synchronnost (1)



Logická organizace distribuovaného systému s odlišením mezi přijetím zprávy a doručením zprávy.

Virtuální synchronnost (2)



Princip virtuálně synchronního skupinového vysílání.

Uspořádání zpráv (1)

Process P1	Process P2	Process P3
sends m1	receives m1	receives m2
sends m2	receives m2	receives m1

Komunikace tří procesů z jedné skupiny. Uspořádání událostí v jednotlivých procesech.

Uspořádání zpráv (2)

Process P1	Process P2	Process P3	Process P4
sends m1	receives m1	receives m3	sends m3
sends m2	receives m3	receives m1	sends m4
	receives m2	receives m2	
	receives m4	receives m4	

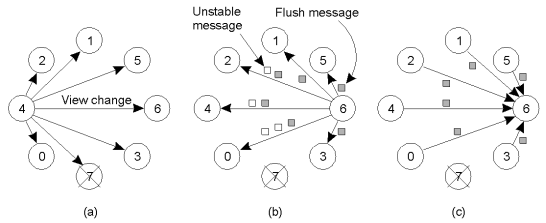
Možné pořadí doručení zpráv pro čtyři procesy v jedné skupině ode dvou různých vysílačů v případě FIFO skupinového vysílání.

Realizace virtuální synchronnosti (1)

Multicast	Basic Message Ordering	Total-ordered Delivery?
Reliable multicast	None	No
FIFO multicast	FIFO-ordered delivery	No
Causal multicast	Causal-ordered delivery	No
Atomic multicast	None	Yes
FIFO atomic multicast	FIFO-ordered delivery	Yes
Causal atomic multicast	Causal-ordered delivery	Yes

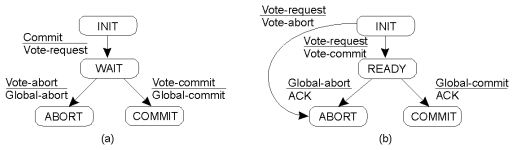
Šest různých verzí virtuálně synchronního spolehlivého skupinového vysílání.

Realizace virtuální synchronnosti (2)



a) Process 4 oznamuje, že process 7 zhavaroval, posílá změnu pohledu
b) Process 6 odesílá všechny své nestabilní zprávy, následované zprávou flush
c) Process 6 zakládá nový pohled jakmile přijme od kohokoliv zprávu flush

Dvoufázové ukončení (1)



a) Konečný automat pro koordinaci dvoufázového ukončování.
b) Konečný automat pro účastníka.

Dvoufázové ukončení (2)

State of Q	Action by P
COMMIT	Make transition to COMMIT
ABORT	Make transition to ABORT
INIT	Make transition to ABORT
READY	Contact another participant

Akce prováděné účastníkem *P* nacházejícím se v ve stavu *READY* a zkontaktovaným s jiným účastníkem *Q*.

Dvoufázové ukončení (3)

actions by coordinator:

```
while START _2PC to local log;
multicast VOTE_REQUEST to all participants;
while not all votes have been collected {
    wait for any incoming vote;
    if timeout {
        while GLOBAL_ABORT to local log;
        multicast  GLOBAL_ABORT to all participants;
        exit;
    }
    record vote;
}
if all participants sent VOTE_COMMIT and coordinator votes COMMIT{
    write GLOBAL_COMMIT to local log;
    multicast GLOBAL_COMMIT to all participants;
} else {
    write GLOBAL_ABORT  to local log;
    multicast GLOBAL_ABORT to all participants;
}
```

Náčrt kroků prováděných koordinátorem ve dvoufázovém protokolu.

Dvoufázové ukončení (4)

Kroky prováděné účastníkem ve dvoufázovém protokolu

actions by participant:

```
write INIT to local log;
wait for VOTE_REQUEST from coordinator;
if timeout {
    write VOTE_ABORT to local log;
    exit;
}
if participant votes COMMIT {
    write VOTE_COMMIT to local log;
    send VOTE_COMMIT to coordinator;
    wait for DECISION from coordinator;
    if timeout {
        multicast DECISION_REQUEST to other participants;
        wait until DECISION is received; /* remain blocked */
        write DECISION to local log;
    }
    if DECISION == GLOBAL_COMMIT
        write GLOBAL_COMMIT to local log;
    else if DECISION == GLOBAL_ABORT
        write GLOBAL_ABORT to local log;
} else {
    write VOTE_ABORT to local log;
    send  VOTE ABORT to coordinator;
}
```

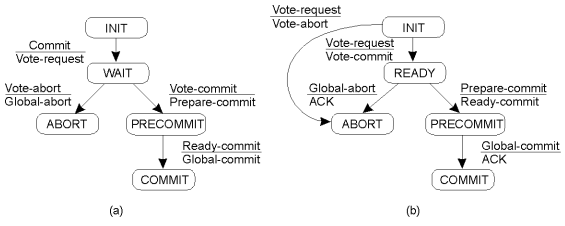

Dvoufázové ukončení (5)

actions for handling decision requests: /* executed by separate thread */

```
while true {
    wait until any incoming DECISION_REQUEST is received; /* remain blocked */
    read most recently recorded STATE from the local log;
    if STATE == GLOBAL_COMMIT
        send GLOBAL_COMMIT to requesting participant;
    else if STATE == INIT or STATE == GLOBAL_ABORT
        send GLOBAL_ABORT to requesting participant;
    else
        skip;  /* participant remains blocked */
```

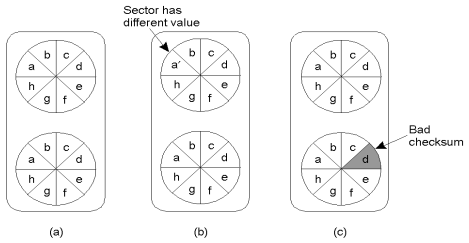
Kroky prováděné při zpracování Steps taken for handling incoming požadavku rozhodnutí o výsledku transakce.

Třífázové ukončení



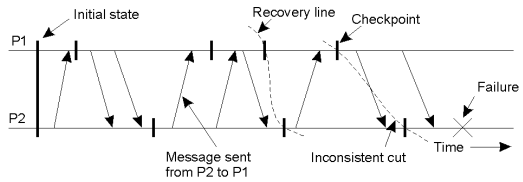
a) Konečný automat koordinátora 3PC
b) Konečný automat účastníka

Obnova trvalé paměti



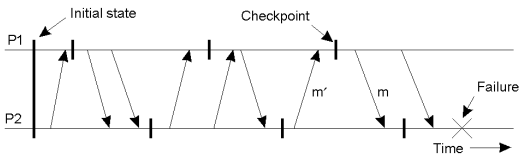
a) Stabilní paměť
b) Výpadek po aktualizaci 1 mechaniky
c) Špatný sektor

Kontrolní body



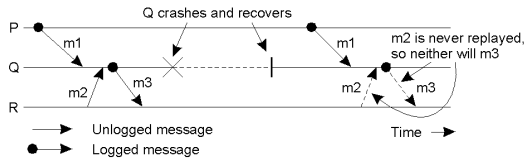
Postup obnovy.

Nezávislé kontrolní body



Účinek dominového efektu.

Logování zpráv



Nesprávné opakování zpráv po zotavení, vede do osiřelému procesu.