

CEDES

Cost Efficient Dependable Electronic Systems



A Process Membership Service
using TT/ET Communication
Scheduling

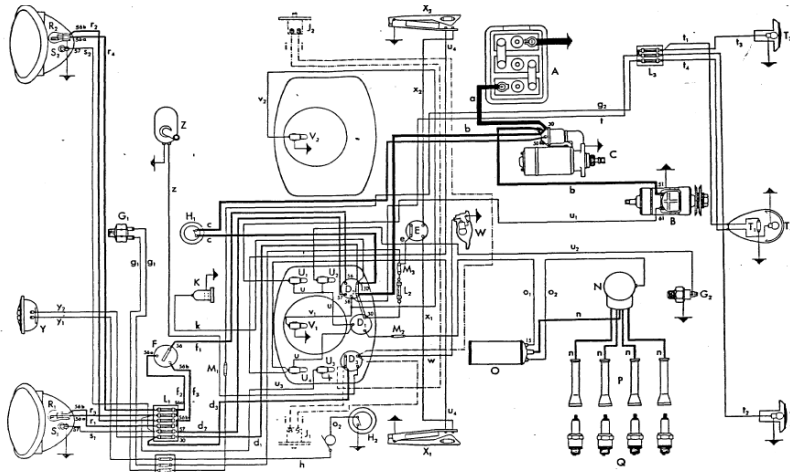
Carl Bergenhem, SP Elektronik

carl.bergenhem@sp.se

010-516 55 53



Example: Electrical plan for VW ca 1950



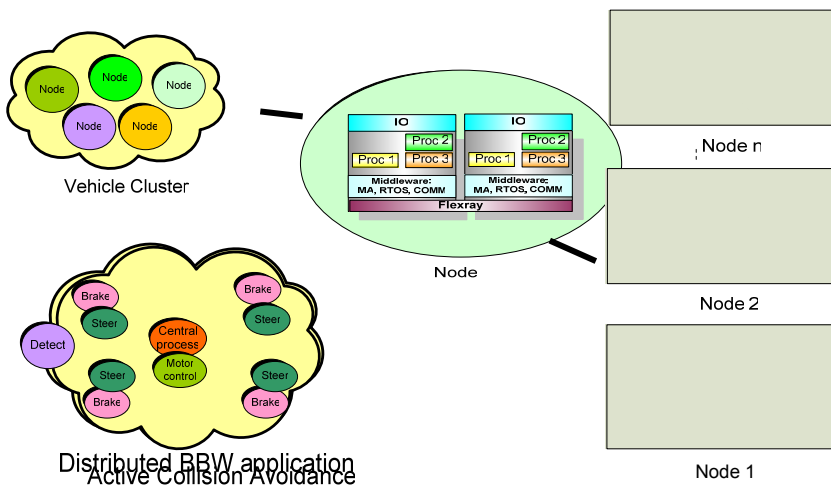
KABELSCHÜSSEL

e. schwarz-weiß-grün 1.0 mm ²	b. braun 0.75 mm ²	v ₁ . gelb-schwarz 1.5 mm ²	f. grau 1.0 mm ²	v ₂ . blau 0.5 mm ²	w. grau-grün 0.5 mm ²
f. weiß-schwarz 2.5 mm ²	i. grau-grün 0.75 mm ²	v ₃ . gelb 1.5 mm ²	f ₁ . grau-rot 0.5 mm ²	v ₃ . blau-grün 0.5 mm ²	z. schwarz-weiß 1.0 mm ²
f ₂ . weiß 2.5 mm ²	k. rot 0.5 mm ²	v ₄ . weiß-schwarz 1.5 mm ²	f ₂ . grau-schwarz 0.5 mm ²	v ₄ . blau-rot 0.5 mm ²	x. schwarz-grün 1.0 mm ²
f ₃ . gelb 2.5 mm ²	n. schwarz 0.85 mm ²	v ₅ . weiß 1.5 mm ²	f ₃ . grau 0.5 mm ²	v ₅ . blau-schwarz 0.5 mm ²	y. braun 1.0 mm ²
m. schwarz-rot 0.75 mm ²	o. schwarz 0.75 mm ²	v ₆ . grau-schwarz 0.5 mm ²	f ₄ . schwarz-rot 0.75 mm ²	v ₆ . schwarz 0.5 mm ²	y ₁ . schwarz-grün 1.0 mm ²

CEDES
Cost Efficient Dependable Electronic Systems

IVSS
Intelligent Vehicle Safety Systems

Abstraction levels



CEDES
Cost Efficient Dependable Electronic Systems

IVSS
Intelligent Vehicle Safety Systems

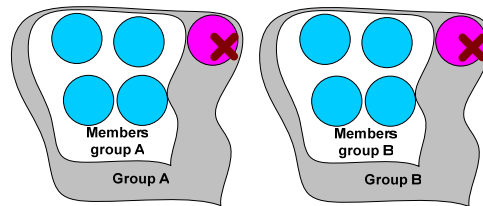
Membership Agreement

• There shall be agreement in the system on the current status of..

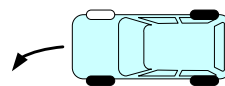
- Nodes
- Processes
- Entities!

• Applications are distributed => One membership group per distributed task

• A group may contain both correct and incorrect entities during run-time



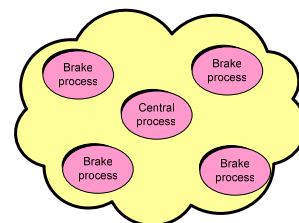
Process Membership - I



• Membership for Distributed Applications

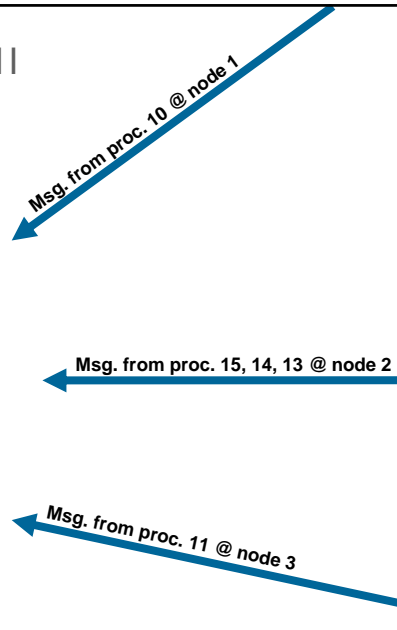
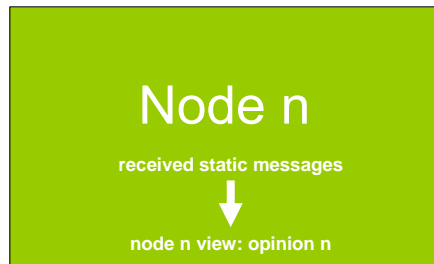
- Distributed task consists of several cooperating processes
- Which processes are live and which are not?
- Preferably a platform service
- Service must be timely & correct

• E.g. Four-wheel brake mode with only three live wheels?



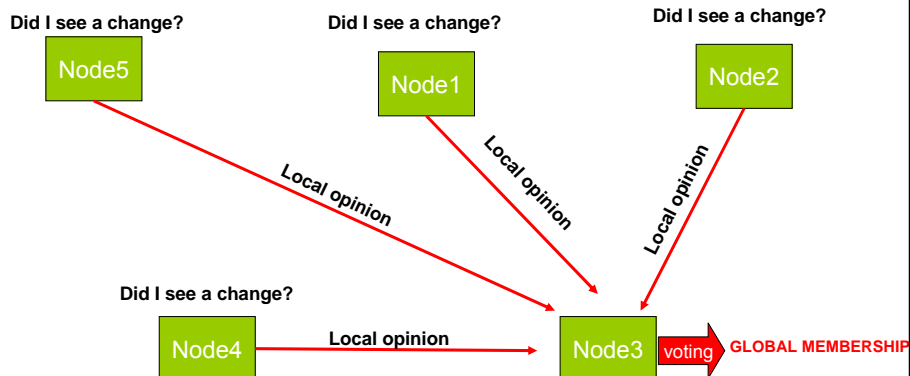
Distributed BBW application

Process Membership - II



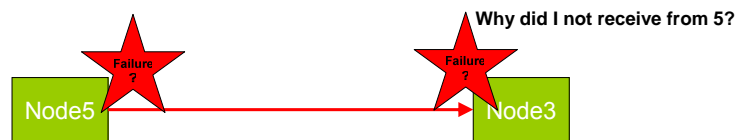
Process Membership - III

- Need to offer a consistent view of (process) status
- Send opinion in dynamic segment (if there was a change)
- Distributed vote for new membership status



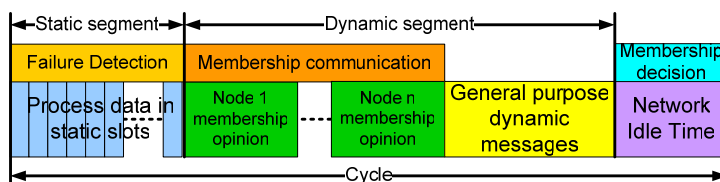
Why do we need a membership protocol at all?

- Self-assessment will do?
 - Message from 5 not received at 3
 - ▶ Was it node 5 that failed or its outgoing link?
 - ▶ Did node 3 suffer in incoming link failure?
 - ▶ Which node should be blamed and shut down?
 - Self-assessment not possible with asymmetric failures (e.g. in incoming link)
 - ▶ All nodes must (eventually) have same view
 - ▶ Need a protocol to solve this
 - ▶ Communication over >1 cycle

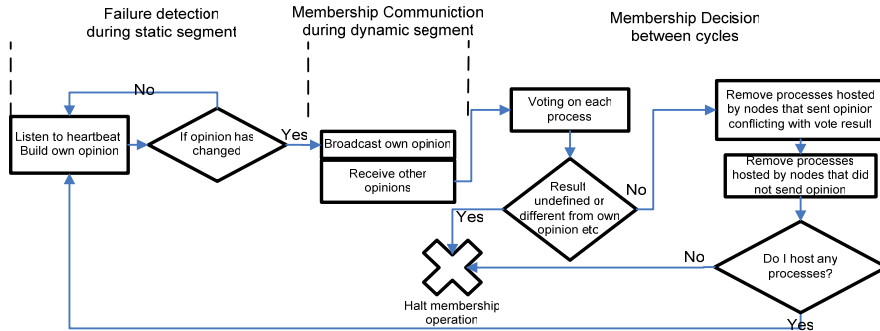


Protocol operation - I

- Failure detection (heartbeat) via TT comm.
 - ▶ Statically scheduled messages
 - ▶ Compare received messages with what is expected
 - ▶ Silence of a slot implies failure of the associated process
- On demand consensus via ET comm.
 - ▶ If failure is detected, run membership protocol
 - ▶ Broadcast opinion to all nodes
- Consistent decision leads to new membership

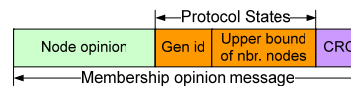


Protocol operation - II (Simplified)



Messages

- Heartbeat message
 - ▶ Static slot
 - ▶ Payload or failure report
 - ▶ Flags for protocol states
 - Membership status,
 - Join-request,
 - Run protocol request
- Membership message
 - ▶ Dynamic slot
 - ▶ The nodes opinion: One bit per process
 - ▶ Protocol states: Gen id and "upper bound of nbr. nodes"



Decision function – Voting on opinions

- Distributed vote on opinions from all nodes

- Must be consistent, handle tie situation
- Need to know opinions to expect
 - ▶ The number of live nodes
- Result about a process may be:
 - ▶ 1: Member
 - ▶ 0: Not member
 - ▶ U: Undefined

- Must be enough opinions for eit decision else undefined

Expected opinions: 4	Proc 12	Proc 11	Proc 10	Proc 9	Proc ...	Proc 2	Proc 1
Opinion N1	0	1	0	1	...	1	0
Opinion N2	0	1	0	1	...	1	0
Opinion N3	0	1	0	1	...	0	1
Opinion N4	0	1	1	0	...	0	1
Opinion N5	0	1	1	0	...	0	1

Candidate Membership	0	1	0	1	...	0	0
	0	1	0	1	...	0	0

Overhead Example

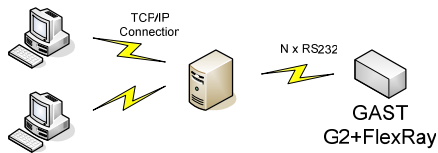
- Assume 10 nodes and 128 processes in the system
- Cycle length of 20 ms
- Observe that this example accounts for worst case sending of opinions in dynamic segment
- During a period of no change, there will be virtually no overhead

Network technology	Line speed	Frame overhead	Max Payload	Required frames	Total overhead	Ratio to line spd.	Notes
TTCAN	500 kbit/s	4-6 bytes	8 bytes	3	135,7 kbit/s	27,2%	Disregard bit-stuffing
FlexRay	10 Mbit/s	8 bytes	254 bytes	1	96,7 kbit/s	0,97%	
Ethernet	100 Mbit/s	26 bytes	1500 bytes	1	167,0 kbit/s	0,17%	Assume that there is a time triggered scheme

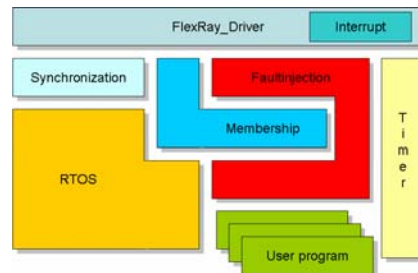
- Other schemes:

- CANEly: Enhanced layer, with membership, for std. CAN
- Overhead: 2-14% (32 nodes, ...)

Experimental system setup



- Cluster with 5 nodes
- Node with dual CPU: MPC+HCS12
- MFR4200 FlexRay chip
- Remote control of experiments
 - ▶ "Server" with serial connection to each node in cluster
- Layers of software in a node



Evaluation of protocol

- Evaluated protocol with fault injection
 - ▶ Specific situations: *scenarios*
 - ▶ Type of failure, time and place
 - ▶ Tested ca 40 different scenarios => OK!



Implementation Results

- Working experimental platform for current and future needs
- Buffer handling and processing takes time!
 - ▶ Current protocol not adapted for this limitation
 - ▶ Delay must added between heartbeat messages to be in accordance with protocol
- Max. cycle length (of single cycle) is 16ms

Selection of cluster parameters:	Value:
Cycle length (total, static, dynamic, NIT)	8,2ms (=7,04 + 0,648 + 0,512)
Number of static slots	128
Static slot duration	55 µs
Static slot payload (maximum)	25 bytes
Worst case consensus latency	16,4 ms
Worst case overhead (dynamic segment)	46,9 kbit/s

Membership simulator

- Test and visualise the membership protocol
- Simulation to slot-level precision
 - ▶ I.e. assume no clock drift.
- Input: Scenario-file
 - ▶ Same format as for the test-system
- Can test
 - ▶ different configurations
 - node, process, message
 - ▶ different failure scenarios
 - ▶ New algorithm ideas

```

C:\y:\min\forking\programming\membership\sim\members13\Debug\members13.exe
N3 opinion matrix
node 0 1 2 3 4 5 6 7 8 9 gen request-YES
OK OK RS OK OK OK OK OK OK OK H
OK OK RS OK OK OK OK OK OK OK H
OK OK RS OK OK OK OK OK OK OK H
OK OK RS OK OK OK OK OK OK OK H
OK OK RS OK OK OK OK OK OK OK H
N3 membership Status: OK
memb: H H H H H H H H
memb_alg: HD1 maxagreement0
st: H H H H H H H H my u15 my gen:0
PROP: H H H H H H H H
memb_alg: HD1 maxagreement0
PROP: H H H H H H H H Request-NOI
N3 membership Status: OK
memb: H H H H H H H H
memb_alg: HD1 maxagreement0
st: H H H H H H H H my u15 my gen:0
PROP: H H H H H H H H
memb_alg: HD1 maxagreement0
PROP: H H H H H H H H Request-NOI
N3 membership Status: OK
memb: H H H H H H H H
=====
Message: 2 Cycle: 1 Slot: 0 Total slots 36 STA phase
proc01 new status OK
proc01 new status OK
proc01 new status OK
proc01 new status OK
proc01 new status OK
Proc: proc00 rcvd gen#6 evsn#0 STAI statusOK hHEATING req#NO
Proc: proc00 rcvd gen#6 evsn#0 STAI statusOK hHEATING req#NO
Proc: proc00 rcvd gen#6 evsn#0 STAI statusOK hHEATING req#NO
Proc: proc00 rcvd gen#6 evsn#0 STAI statusOK hHEATING req#NO
Proc: proc00 rcvd gen#6 evsn#0 STAI statusOK hHEATING req#NO
=====
Message: 3 Cycle: 1 Slot: 1 Total slots 37 STA phase
Proc: proc01 rcvd gen#7 evsn#0 STAI statusOK hHEATING req#NO
    
```

Todo

- More experiments and measurements
 - ▶ A “real” application
- Formal verification of protocol
 - ▶ UPPAAL
- Handling of transient failures
 - ▶ Threshold and counter for each process
 - ▶ Classification of failure (null frame, CRC errors, etc.) -> Penalty
- Fail reporting and
 - ▶ Faster consensus
- Use redundancy capability: Dual bus etc.
 - ▶ Several signals per frame
- Adapt protocol theory to
 - ▶ Overcome delay requirements of buffer handling
 - ▶ Allow for a longer system cycle than one cycle