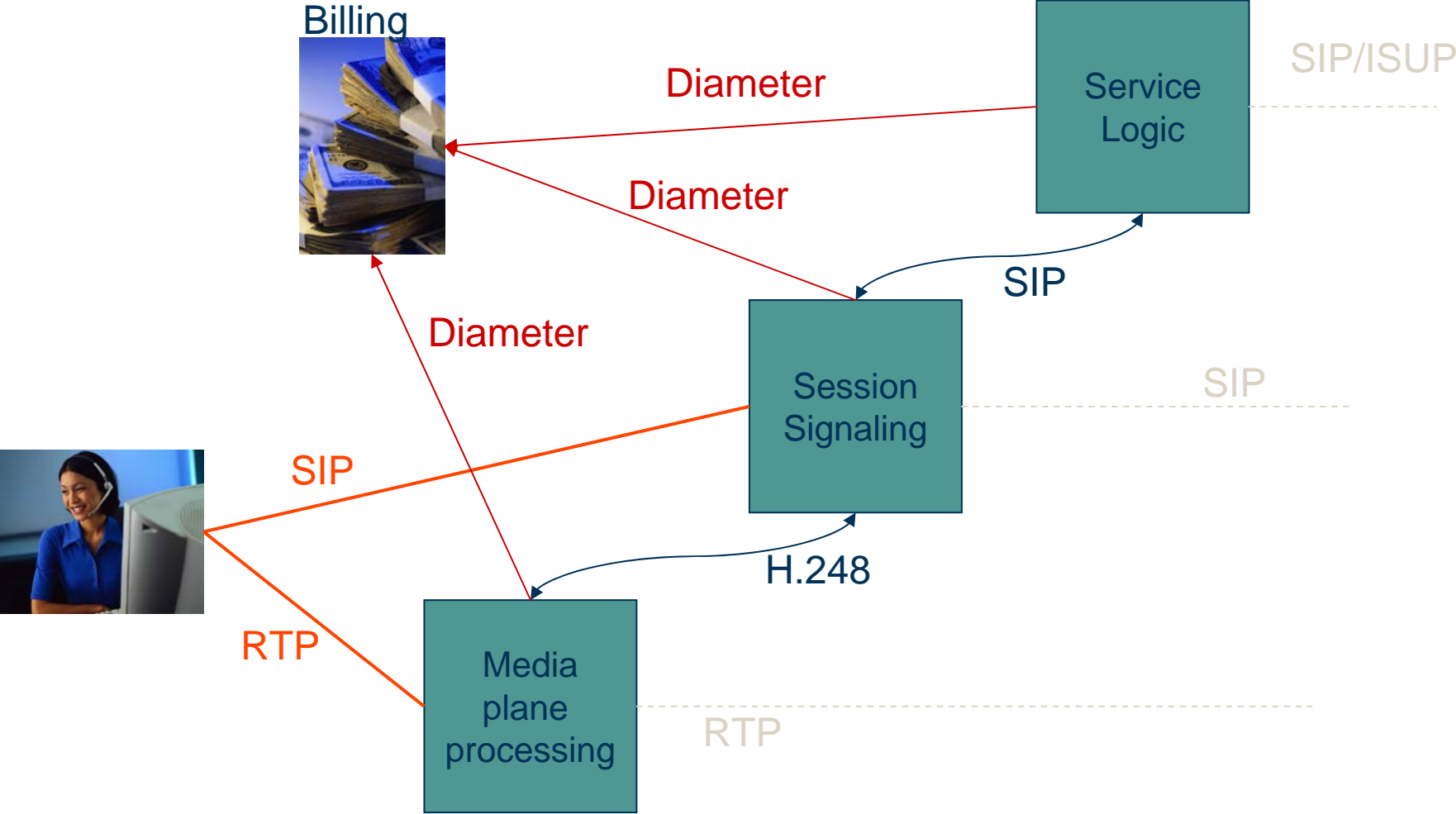


# Exploiting Multicore at Ericsson

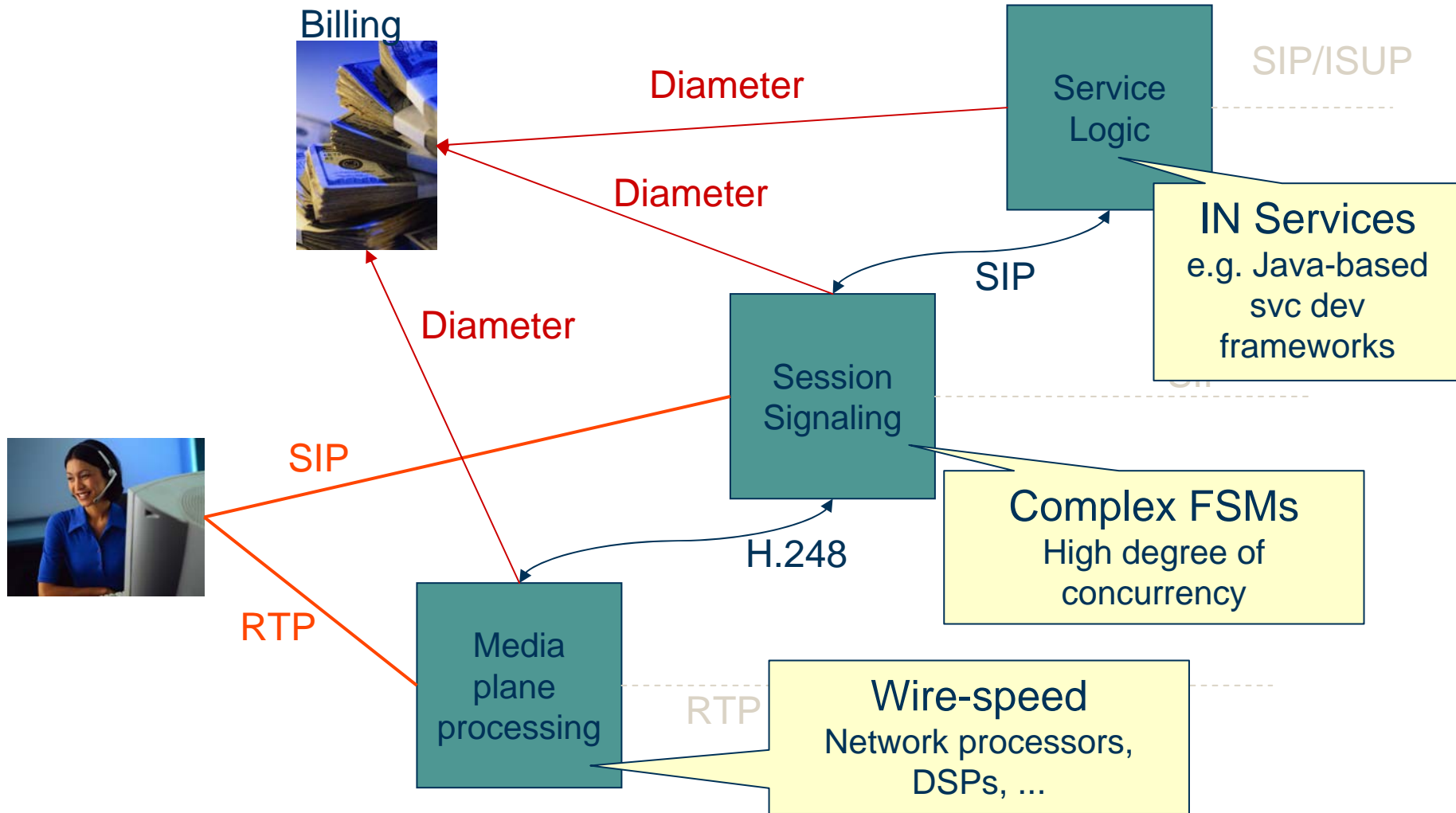
Ulf Wiger

Sr Software Architect  
IMS Gateways, Ericsson

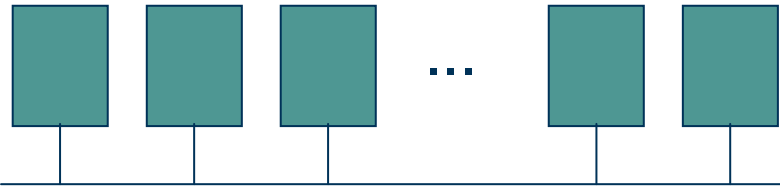
# Modern Telecoms



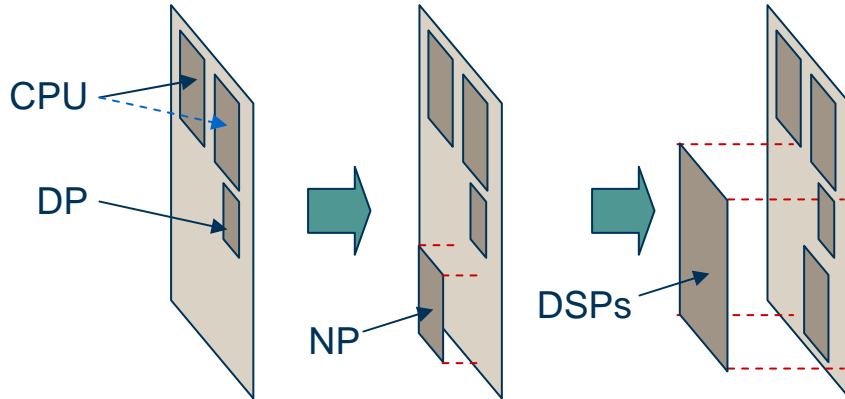
# Modern Telecoms



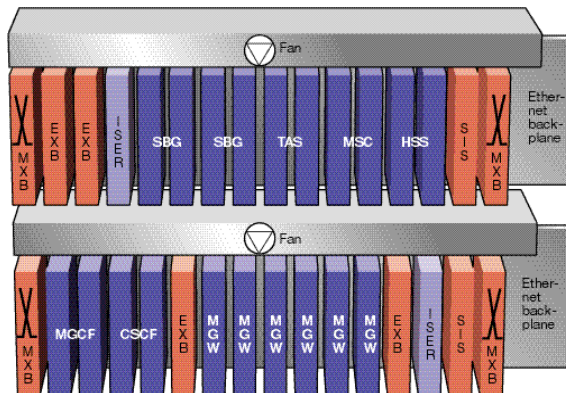
# Different Execution Platforms



Highly scalable processing clusters (C++/Erlang/3PP)



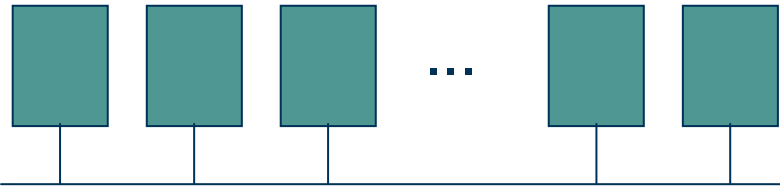
Compact "system on a blade"  
Tight space and power budget



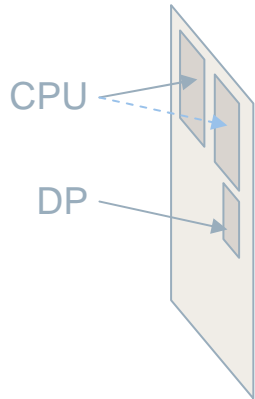
Subracks of dedicated blades  
(e.g. "Integrated Site", or IS)

[http://www.ericsson.com/ericsson/corpinfo/publications/review/2005\\_01/files/2005014.pdf](http://www.ericsson.com/ericsson/corpinfo/publications/review/2005_01/files/2005014.pdf)

# Different Execution Platforms



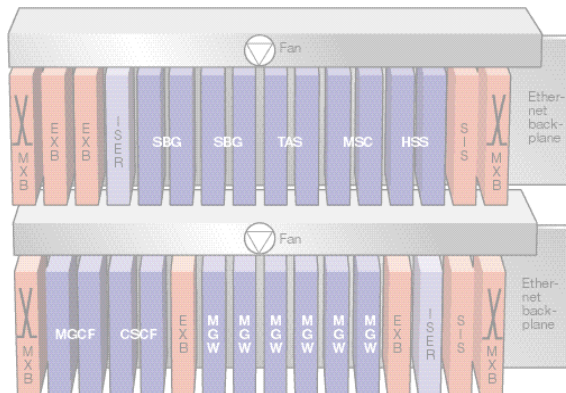
Highly scalable processing clusters (C++/Erlang/3PP)



Multicore can help reduce footprint

- E.g. through virtualization
- Software already written for distributed processing

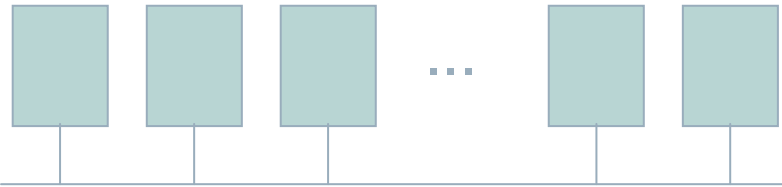
on a blade”  
over budget



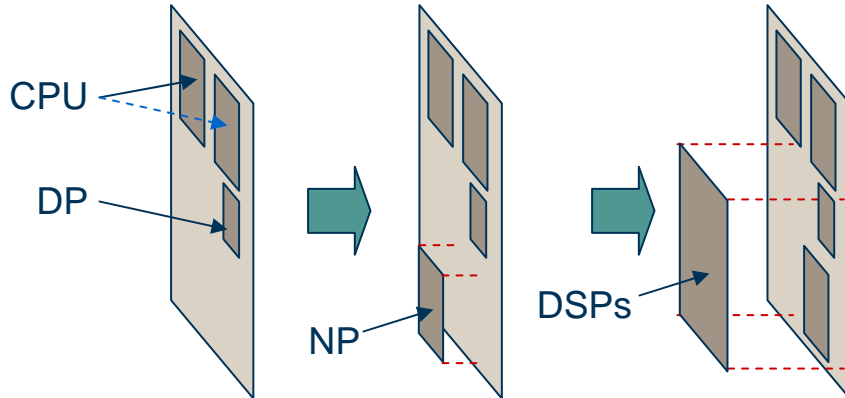
Subracks of dedicated blades (e.g. "Integrated Site", or IS)

[http://www.ericsson.com/ericsson/corpinfo/publications/review/2005\\_01/files/2005014.pdf](http://www.ericsson.com/ericsson/corpinfo/publications/review/2005_01/files/2005014.pdf)

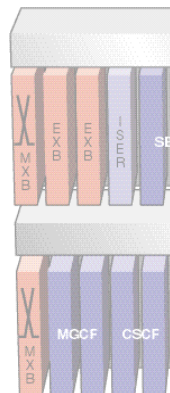
# Different Execution Platforms



Highly scalable processing clusters (C++/Erlang/3PP)



Compact "system on a blade"  
Tight space and power budget



Multicore can help through

- Packing more umph in one slot
- Replacing different special-purpose processors, reducing cost and footprint

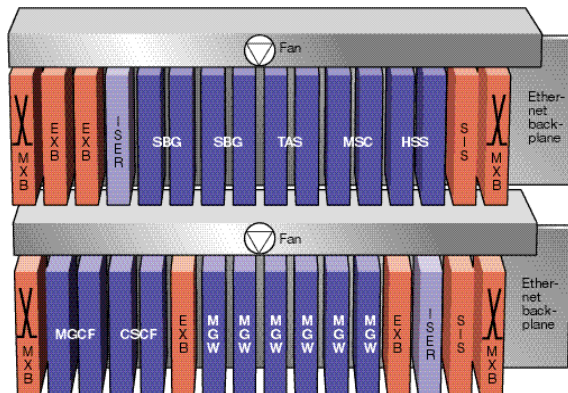
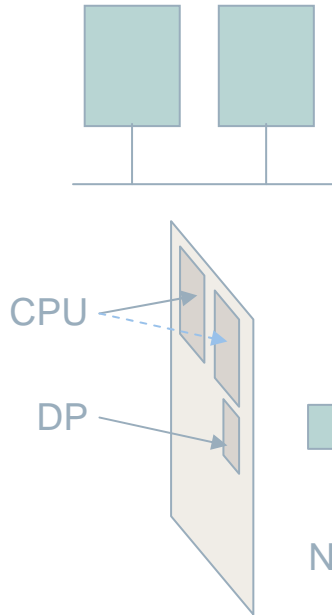
icated blades  
Site", or IS)

[ew/2005\\_01/files/2005014.pdf](http://www.ericsson.com/ew/2005_01/files/2005014.pdf)

# Different Execution Platforms

Multicore can help through

- Making each "blade system" more compact
- Better performance/slot  
=> better price/performance
- (Chaining subracks is always tricky – inter-subrack links a bottleneck)

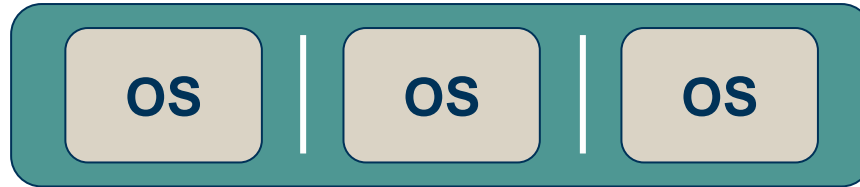


Subracks of dedicated blades  
(e.g. "Integrated Site", or IS)

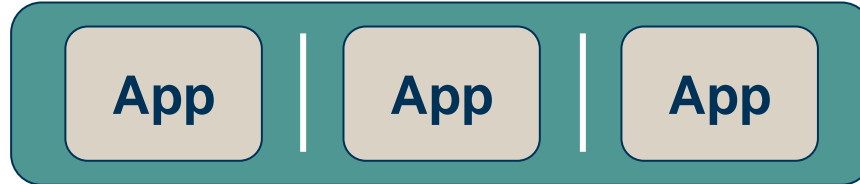
[http://www.ericsson.com/ericsson/corpinfo/publications/review/2005\\_01/files/2005014.pdf](http://www.ericsson.com/ericsson/corpinfo/publications/review/2005_01/files/2005014.pdf)

# Different levels of parallelism

Virtual  
Container:



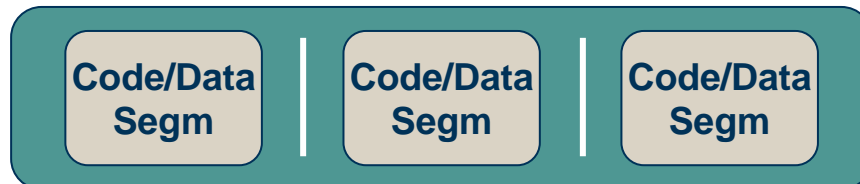
Operating  
System:



Application:

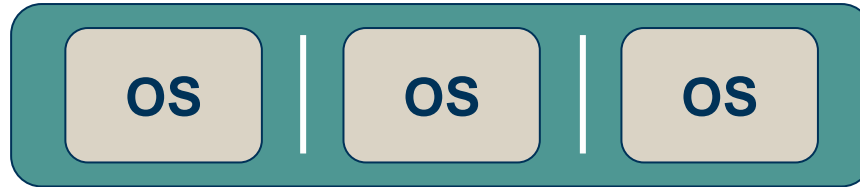


Thread:



# Different levels of parallelism

Virtual  
Container:



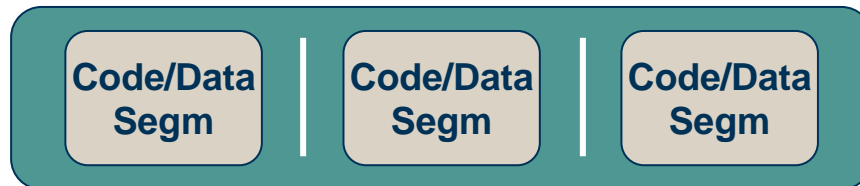
Operating  
System:



Application:



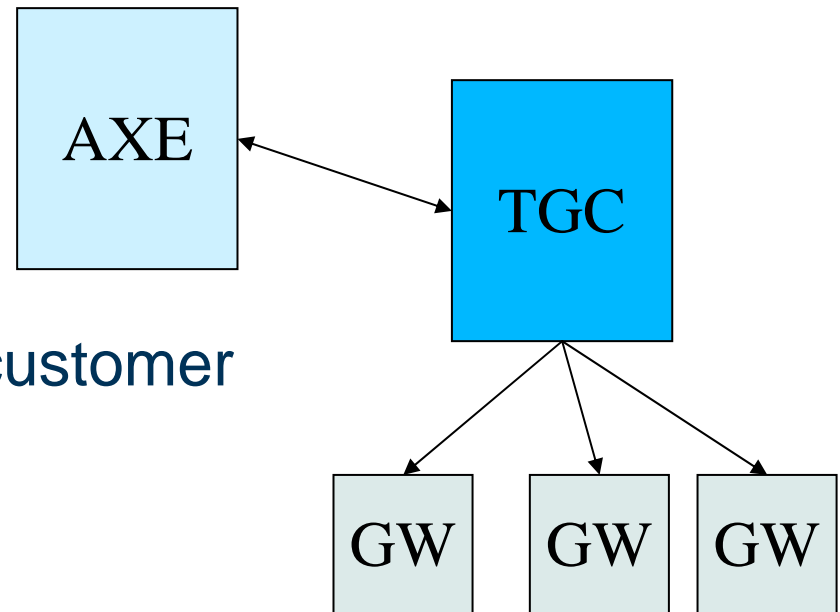
Thread:



} Difficult(?)

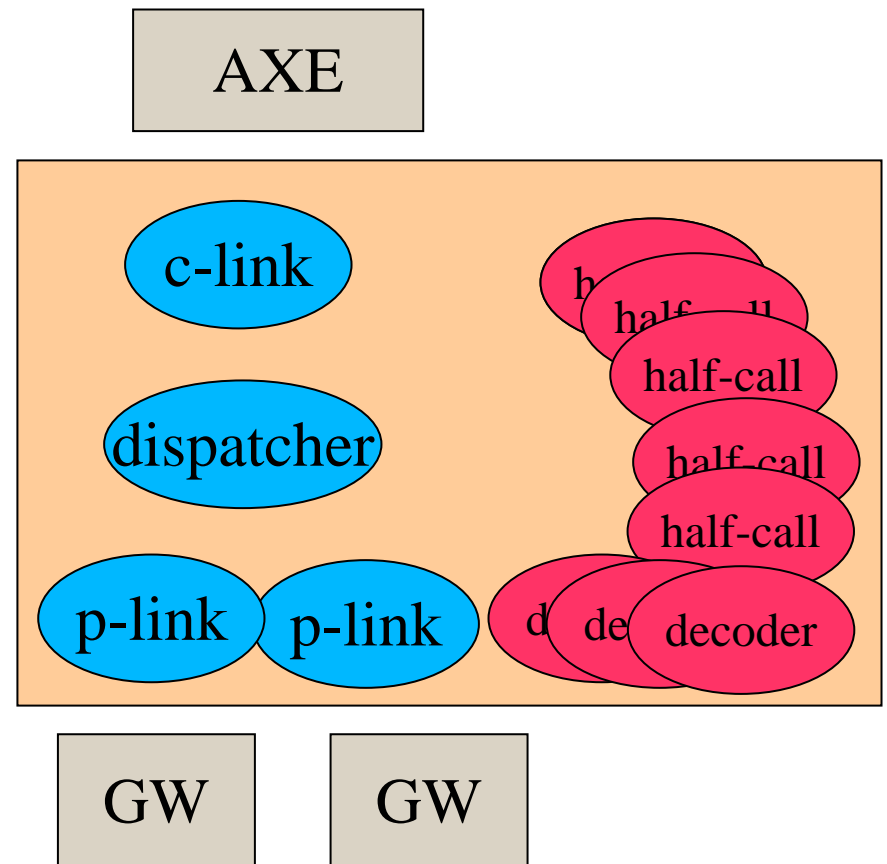
# Case study: The Ericsson TGC

- **Telephony Gateway Controller**
- Formerly known as
- the Hybrid
  - reported (by customer!) to have ISP of 0.999999999
- the Mediation Logic
- Runs on AXD 301 and IS
- Developed in Erlang
- Multicore version shipped to customer



# The TGC (internal)

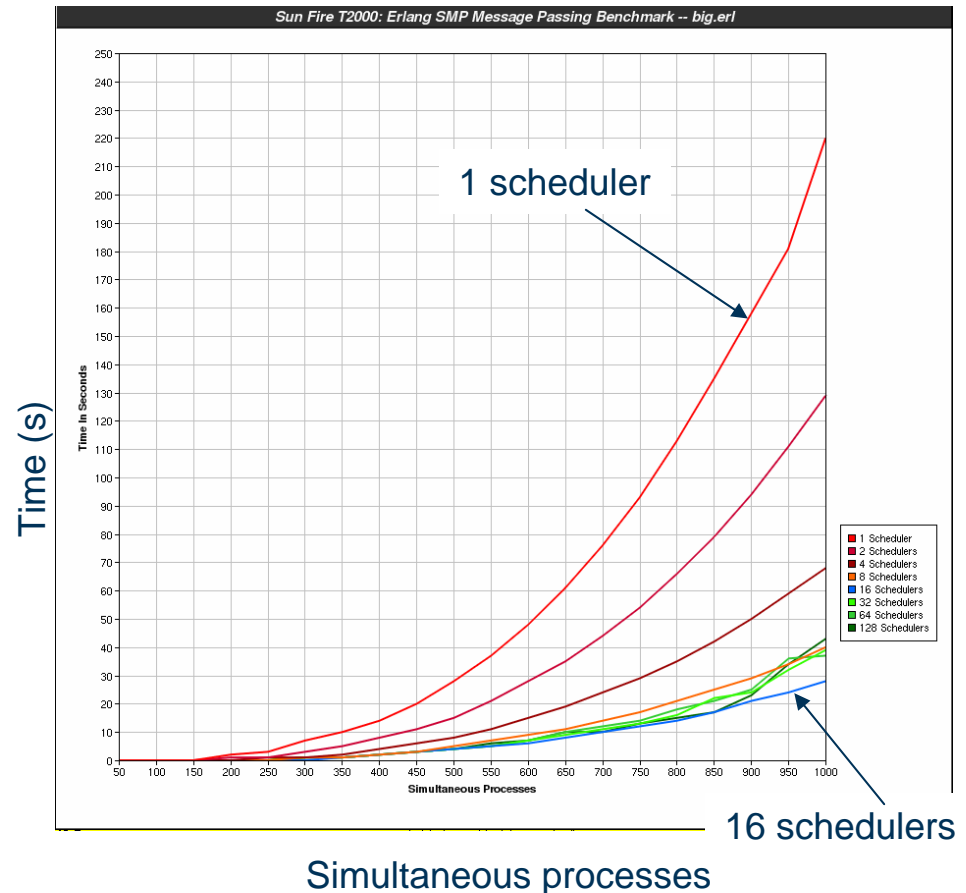
- Process model
  - Servers
  - Workers
- ~100 servers
- ~10 workers per call
- 10,000 workers is perfectly normal



# Erlang on multicore

- Erlang programs are meant for distributed processing
  - Erlang processes do not share data
  - Message passing is distribution transparent
- SMP prototype -97, serious implementation in -05.
- Mid -06 we ran a benchmark mimicking call handling (axdmark) with a prototype SMP emulator. Observed speedup/core: 0.95
- TGC released on multicore in Q207

## "Big bang" benchmark on Sunfire T2000



<http://www.franklinmint.fm/blog/archives/000792.html>

# TGC Results (top)

```
Tasks: 50 total, 2 running, 48 sleeping, 0 stopped, 0 zombie
Cpu0  : 62.5% us, 3.7% sy, 0.0% ni, 32.4% id, 0.0% wa, 0.0% hi, 1.3% si
Cpu1  : 36.1% us, 2.7% sy, 0.0% ni, 60.9% id, 0.0% wa, 0.0% hi, 0.3% si
Mem:   4092764k total, 459352k used, 3633412k free, 8196k buffers
Swap:   0k total, 0k used, 0k free, 215796k cached
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
1975	homer	25	0	2295m	192m	2144	S	99.9	4.8	179:40.46	beam.smp
1	root	16	0	664	244	208	S	0.0	0.0	0:01.50	init
2	root	RT	0	0	0	0	S	0.0	0.0	0:00.02	migration/0
3	root	34	19	0	0	0	S	0.0	0.0	0:00.00	ksoftirqd/0
4	root	RT	0	0	0	0	S	0.0	0.0	0:00.01	migration/1
5	root	34	19	0	0	0	S	0.0	0.0	0:00.00	ksoftirqd/1

# TGC Results (dtop)

```
ppbl_bs13-R3A@blade_ size 2345(131M, cpu% 107, procs 10371, runq 0 15:15:53
memory[kB]:  proc  58223, atom  1768, bin  170, code  29772, ets  39215
```

pid	name	current	msgq	mem	cpu
<0.5872.491	prfTarg	(prfPrc:pinf/2)	0	2036	22
<0.18323.47	(erlang:apply/2)	(gcpServ:recvl/3)	0	17	10
<0.18436.47	(erlang:apply/2)	(gcpServ:recvl/3)	0	24	5
<0.1813.0>	sysProc	(gen_server:loop/6)	0	981	2
<0.27384.47	(pthTcpNetHandler:init/1)	(gen_server:loop/6)	0	587	1
<0.18350.47	(erlang:apply/2)	(gcpTransportProxy:	0	8	1
<0.1935.0>	ccpcServer_n	(gen_server:loop/6)	0	587	0
<0.18526.47	(erlang:apply/2)	(gcpTransportProxy:	0	6	0
<0.1923.0>	sbm	(gen_server:loop/6)	0	1719	0
<0.3603.0>	(erlang:apply/2)	(gcpServ:recvl/3)	0	5	0

# TGC results (performance)

Traffic scenario	IS/GCP 1slot/board	IS/GEP Dual core One core running 2slots/board	IS/GEP Dual core Two cores running 2slots/board	AXD CPB5	AXD CPB6
POTS-POTS /AGW	X call/sec	2.3X call/sec One core used	4.3X call/sec OTP R11_3 beta+patches	0.4X call/sec	2.1X call/sec
ISUP-ISUP /Inter MGW	3.6X call/sec	7.7X call/sec One core used	13X call/sec OTP R11_3 beta+patches	1.55X call/sec	7.6X call/sec
ISUP-ISUP /Intra MGW	5.5X call/sec		26X call/sec	3.17X call/sec	14X call/sec

# TGC Experience

- Porting effort: negligible (for the application)
- Porting effort: modest (for the platform)
- Architecture dependency: low (for the application)
- Results: excellent
- Future: bright
  
- “Funky languages” (Hagersten)  
can sometimes save the day

**ERICSSON**



**TAKING YOU FORWARD**