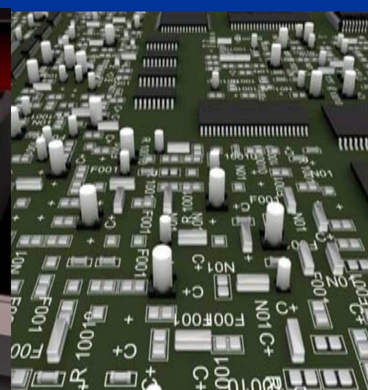
Automating the Configuration of the FlexRay Communication Cycle

Nicolas Navet

Nicolas.navet@realtimeatwork.com

http://www.realtimeatwork.com



27/11/2008

INRIA

RTaW RealTime-at-Work

Better technical solutions for real-time systems

FlexRay configuration

- Extremely complex problem:
  - Mixed of TT and ET scheduling
  - Tightly linked with task scheduling
  - Large number of parameters (>50)
  - AUTOSAR constraints (COM, FXR Interface, etc)
  - ...
- Design objectives should be first clearly identified:
  - Minimum bandwidth to use cheap components (2.5 Mbit/s, 5MBit/s ?)
  - Enable incremental design ?
  - Carry-over of ECUs ?
- No chance to solve the pb optimally – too many free variables, sub-problems alone are NP-hard

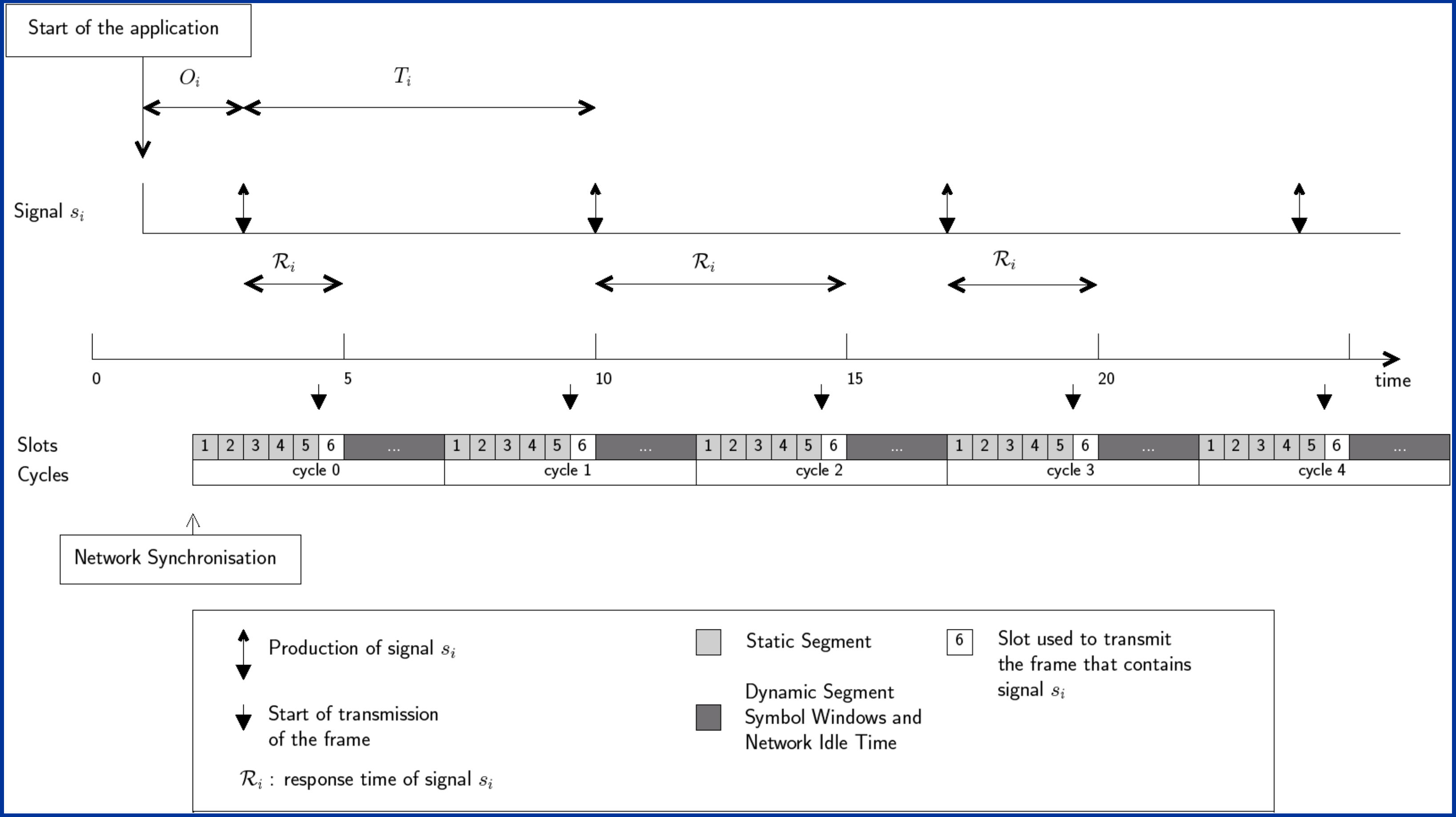
Outline

1. Configuring the FlexRay communication cycle
    1. System model
    2. Objectives of the configuration step
    3. Identifying sub-problems and solutions
2. Verifying signal timing constraints
3. Our approach to configuration : NETCAR-FlexConf
4. Experimentations
    a. Performance on a typical case-study
    b. Comparison with CAN and Multi-CANs

RTaW RealTime-at-Work

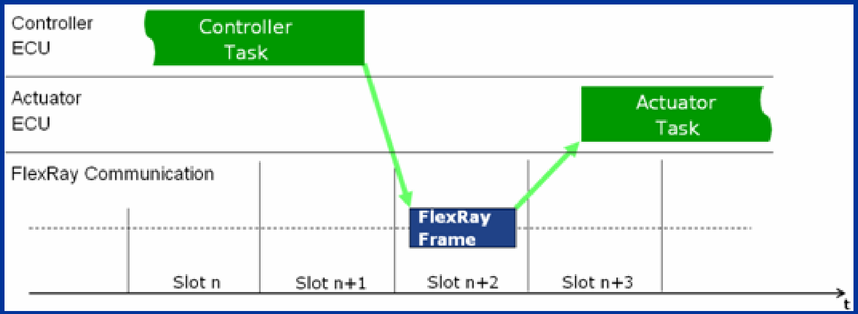
Configuring the FlexRay communication cycle

System model (1/2)



System model (2/2)

- Tasks run either synchronously or asynchronously wrt the communication cycle:
  1. Fully asynchronously : signals produced at arbitrary points in time
  2. Weakly synchronously : task startup triggered by the networks but task periods are arbitrary
  3. Synchronously : task periods multiple of the cycle length



Picture from [1]

Objectives of the configuration step

1. Respect design constraints (e.g., cycle length)
2. Ensure signal's freshness constraints
3. Preserve system's extensibility:
   - Use as few slots as possible
   - Use the slots at the right positions:
     - ST vs DYN segment (size, occupation)
     - future 2.5ms signals in the ST Segment
   - Build the frames at the right instants (CPU load)
4. Maximize robustness against transmission errors for redundant frames (i.e., replicas)

Sub-problems

- Assumptions here: cycle communication length, frame data payload, slot size are decided

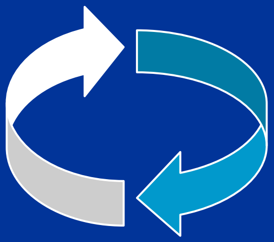
a. Set the relative size of ST and DYN segment

b. Frame packing : build frames from signals

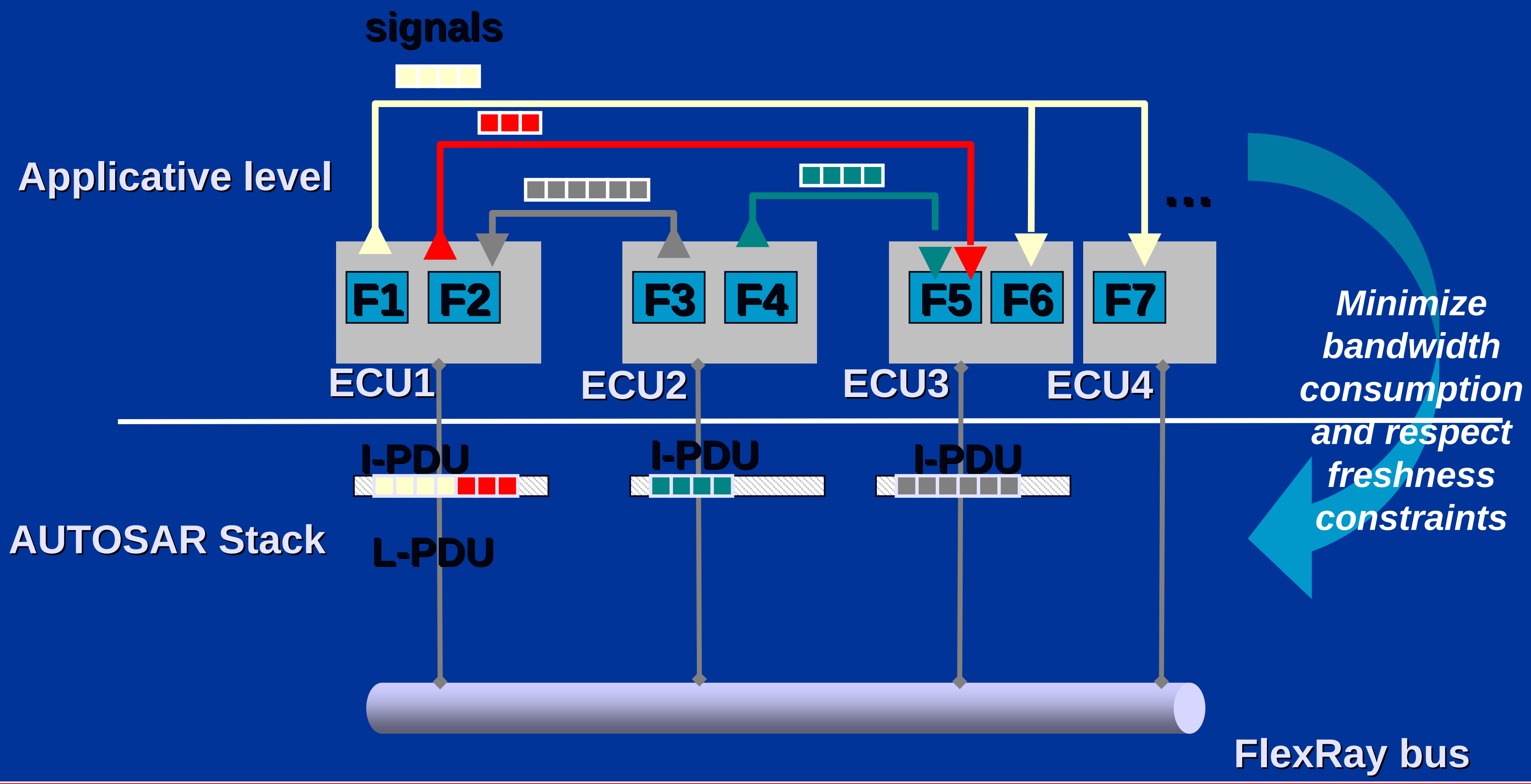
c. Slot allocation : allocate the slots to the ECUs

d. Frame scheduling: schedule the frame transmissions for the 64 communication cycles

- Issue: sub-problems are interdependent but good sub-optimal solutions are feasible

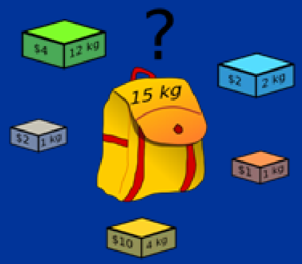


Frame packing : Packing signals into I-PDU and, if network independence is needed, I-PDU into L-PDU



Frame-packing from an algorithmic point of view

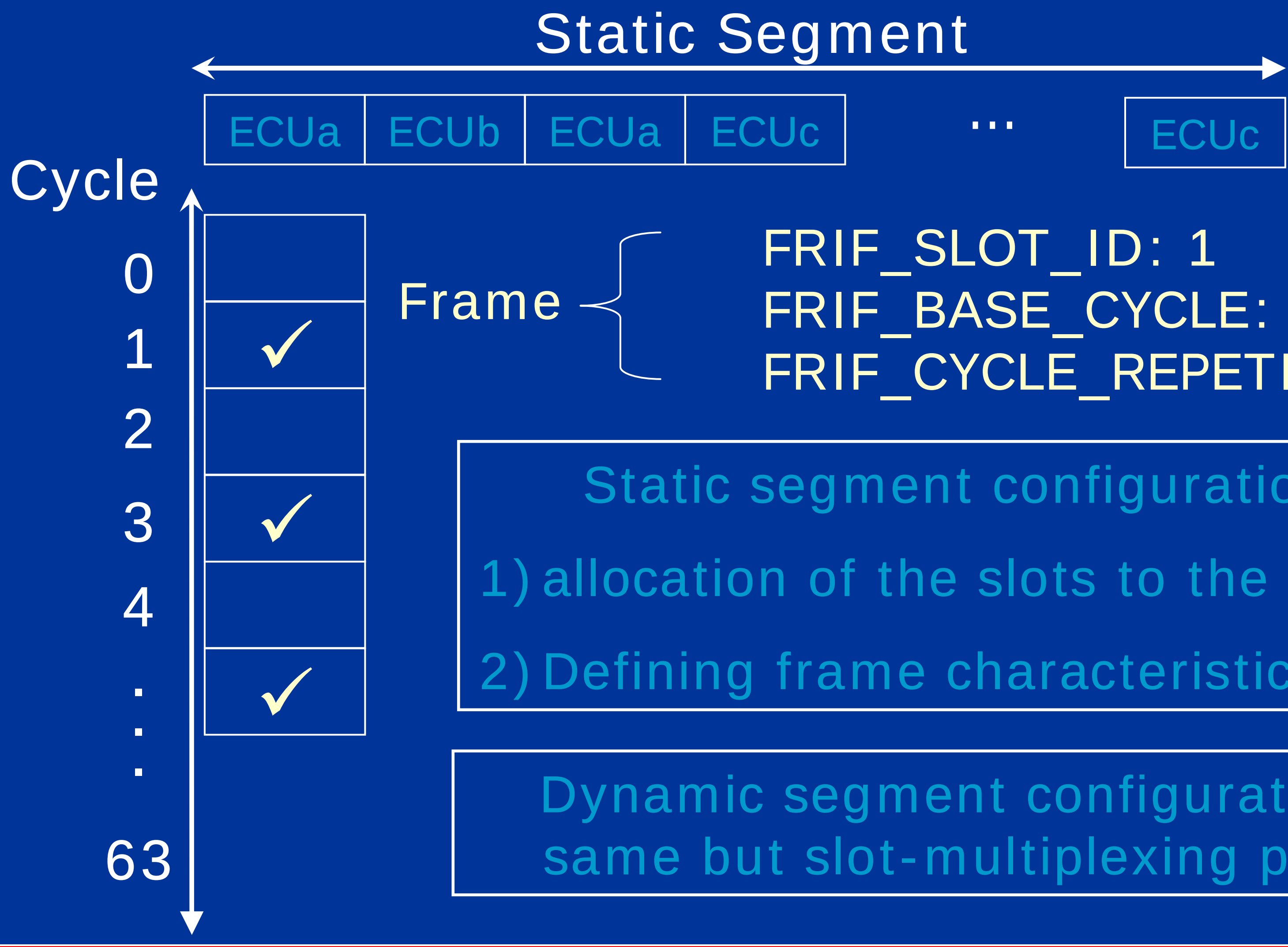
- The bad news: problem is NP-hard (bin-packing)



- The good news: there are efficient heuristics
  - Rate Monotonic is a good starting point
  - Better heuristics can be found in ref[5]
  - GA or local search techniques might provide further improvements
- What is missing: performance guarantees for the heuristics (e.g., factor 2 from the best solution)

RTaW RealTime-at-Work

Building the communication schedule



Static segment configuration :

1) allocation of the slots to the ECUs

2) Defining frame characteristics

Dynamic segment configuration :
same but slot-multiplexing possible

Building the static communication schedule: "Best Slot First" (BSF) heuristic – see ref[9]

- Step 1: For each slot and each ECU, compute the "maximum" number of signals the slot can transmit:
  - A heuristic is used to build the set of frames for each slot and each ECU
  - Only solutions that meet timing constraints are considered
- Step 2: Keep the (slot,ECU) couple that maximizes the number of signals transmitted
- Repeat until there is no frame or no slot left

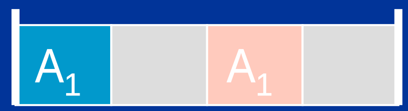
Dynamic segment – some hints

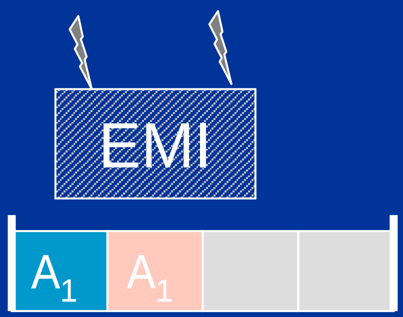
- Context:
  - Use of slot multiplexing
  - No other timing constraints than a minimum transmission frequency
  - Frame-packing is done
- There is a simple bandwidth-optimal policy to build the schedule from the frames (see ref[9]):
  - Rank the whole set of frames by increasing periods
  - Insert the frames one after the other at the first possible (slot,base cycle)
  - Use a new slot when all previous have been filled up

Relative length of the static and dynamic segments

- 2.5ms signals sent in the static segment impose some constraints …
- Proposal : share the available bandwidth between segments according to a parameter chosen by the user (e.g., ST=70% and DYN=30%)

Maximizing the efficiency of redundant transmissions

Question: 

or 

??

Fail-silent producer nodes : if a frame is received, the content is correct

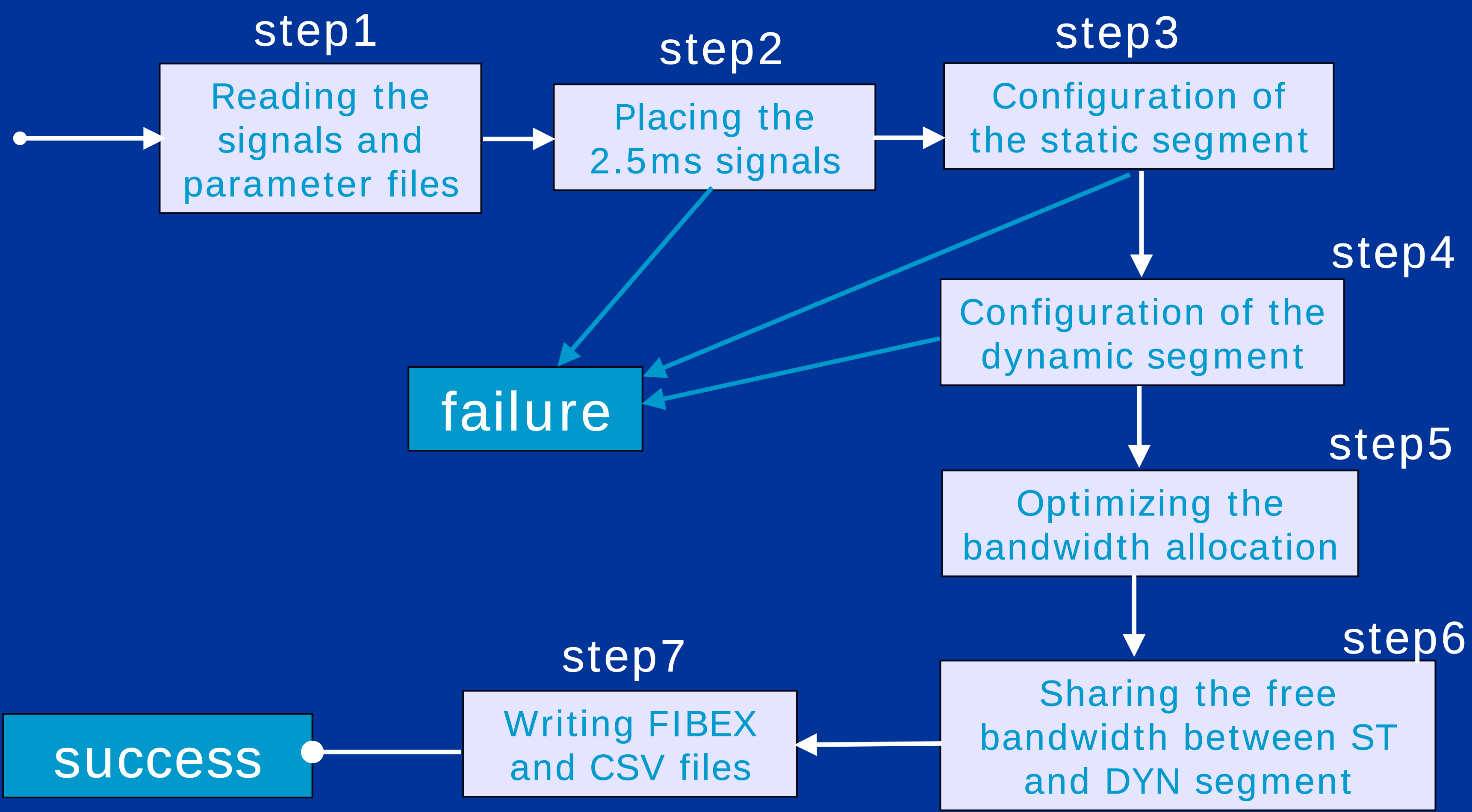
- Fail-silent nodes : one frame is enough → distribute evenly

- Non fail-silent nodes : all frames are needed → group together

➢ Simple design guidelines providing large robustness improvements – see ref[6]

RTaW RealTime-at-Work

Our approach to configuration – implemented in NETCAR-FlexConf



Verifying signal freshness constraints

Verifying signal freshness constraints

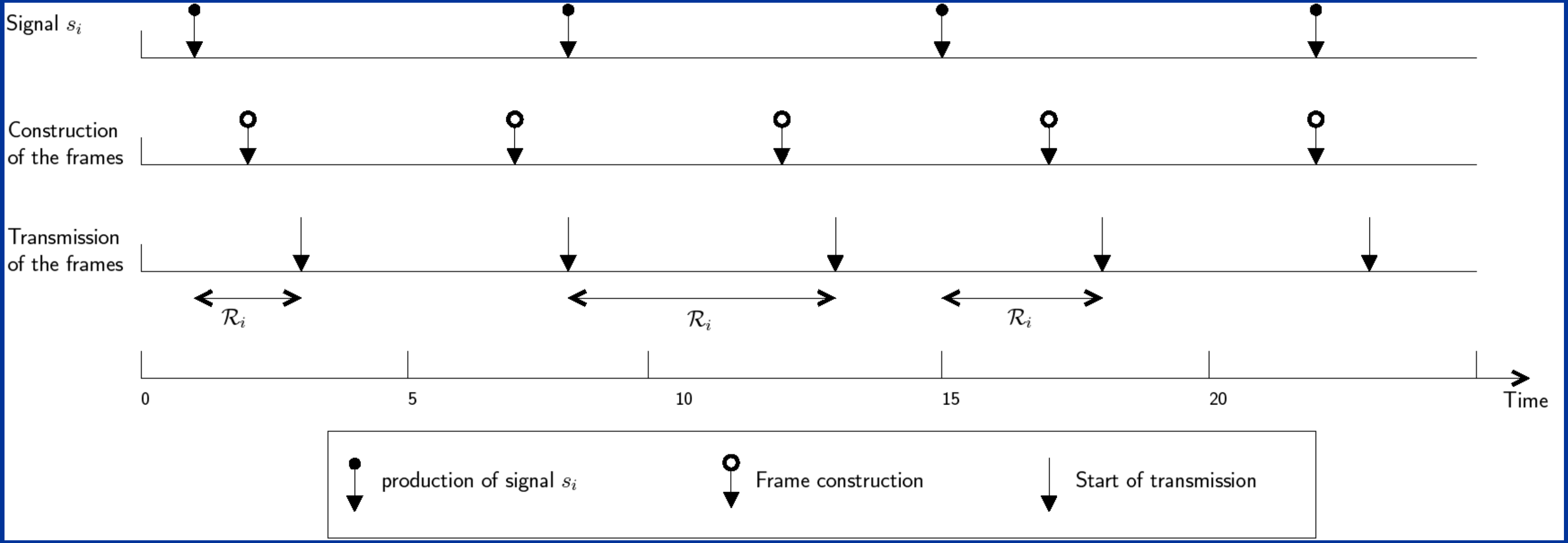
- Configuration here means communication schedule

a. Configuration not needed : non-schedulability test based on the minimum number of slots required for the ST and DYN segment (necessary but not sufficient)

b. Configuration needed : exact signal worst-case response time computation

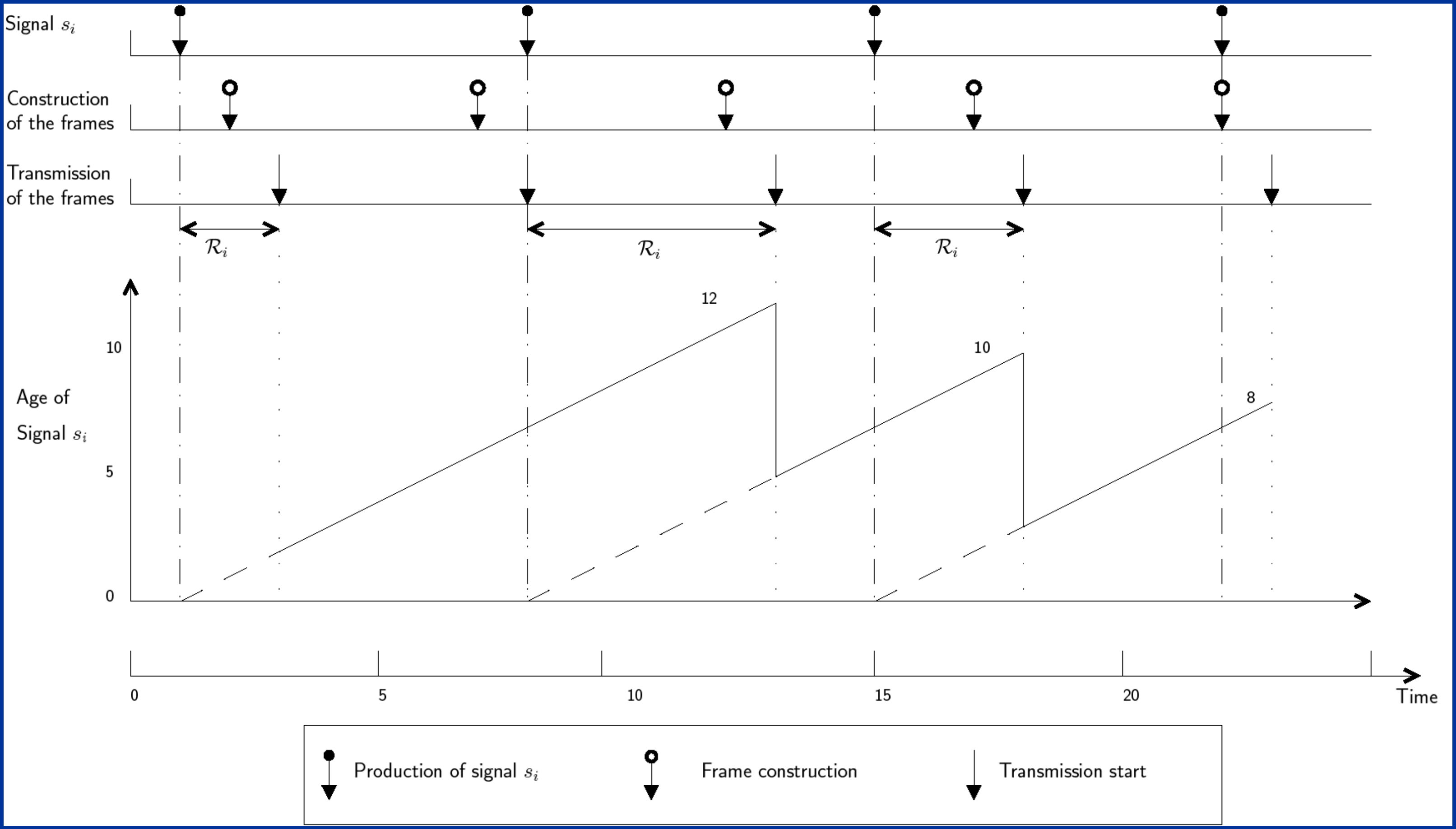
Response time of a signal

- Response time made of
  1. time between signal production and frame construction
  2. time between frame construction and reception by the receiving stations



➢ Impact of the FlexRay Job List!

Most meaningful : age of a signal on the receiver end



Asynchronous case:

max. age = production period + worst-case response time

Experimentations

1. Experimental setup
2. Typical application
3. FlexRay VS (multi)-CAN with/without offsets

Experimental setup

- Communication cycle : 5ms
- Data rate: 2.5 Mbit/s (45 slots), 5 Mbit/s (86 slots) and 10 Mbit/s (155 slots)
- Frame data payload (ST and Dyn) : 16 bytes
- Frame construction points : start of the static segment + start of the dynamic segment
- « Slot multiplexing » in DYN segment

Application under study

- Asynchronism tasks / communication cycle
- 356 signals sent by 14 ECU
- Signal sizes range from 1 to 64 bits
- Production period: 10ms to 1s
- Useful load: 60kbit/s
- 2 ECU transmit only aperiodic signals
- All aperiodic signals sent in the dynamic segment
- Transmission period for aperiodic signals: 320ms
- No 2.5ms frames
- Max. signal response time: 110% period

Results obtained with NETCAR-FlexConf: static segment

Set of FlexRay frames

ECU	Payload (bits)	Slot	BaseCycle	Repetition	#signaux
ECU1	128	31	1	2	33
ECU1	126	31	2	4	22
ECU1	90	31	4	16	6
ECU2	47	72	1	1	9
ECU3	126	78	1	8	51
ECU3	128	78	2	64	11
ECU3	24	78	3	64	2
ECU4	128	30	1	2	24
ECU4	121	30	2	4	29
ECU4	16	30	4	64	1
ECU5	56	73	1	1	10
ECU6	115	29	1	2	28
ECU6	48	29	2	64	2
ECU7	114	74	1	16	12
ECU8	52	71	1	16	8
ECU9	117	77	1	32	20
ECU9	32	77	2	64	1
ECU10	96	75	1	8	14
ECU11	8	70	1	16	1
ECU14	87	76	1	64	17

Observations:

a) 12 slots -> minimum possible

b) Configuration algorithm efficient

Dynamic segment: one slot used

Free slots left: 40 DYN vs 90 ST = 30/70% as requested

Experimentations at higher load levels

- Goal:
  - Assessing the limits of FlexRay
  - Comparison with CAN 500Kbit/s and multi-CAN solutions
- Set of signals: up to 10x the initial load (duplication)
- CAN set of frames:
  - Same frame-packing algorithm as for FlexRay
  - CAN Priorities are assigned according to Rate-Monotonic
  - CAN frame response time / offset assignement strategy computed with NETCAR-Analyzer

RTaW RealTime-at-Work

Performances at higher loads

Useful load (signals)	**FlexRay 2.5Mbit/s**		**FlexRay 10Mbit/s**		**1x CAN 500Kbit/s**	
Load 1x ($\approx$ 60kbit/s)		free slots		free slots	network load	31%
	ST	23	ST	100	R without offsets	15.3
	DYN	9	DYN	43	R with offsets	7.8
Load 2x ($\approx$ 120kbit/s)		free slots		free slots	network load	57%
	ST	21	ST	98	R without offsets	49.6
	DYN	9	DYN	43	R with offsets	14.9
Load 3x ($\approx$ 180kbit/s)		free slots		free slots	network load	85%
	ST	19	ST	96	R without offsets	148.5
	DYN	7	DYN	41	R with offsets	79.7
Load 4x ($\approx$ 240kbit/s)		free slots		free slots	non-schedulable 2x CAN 500 OK	
	ST	19	ST	96		
	DYN	7	DYN	40		
Load 5x ($\approx$ 300kbit/s)		free slots		free slots	non-schedulable 2x CAN 500 depending on the overlap	
	ST	15	ST	92		
	DYN	6	DYN	40		
Load 10x ($\approx$ 600kbit/s)		free slots		free slots	non-schedulable with two CAN buses	
	ST	**3**	ST	**84**		
	DYN	**0**	DYN	**36**		

Conclusion

- Configuring FlexRay communication cycle is a complex problem but:
  - Design choices drastically reduce the search space
  - There are efficient algorithms / guidelines / tools to build the pdu, the frames, the communication schedule, verify timing constraints, define the FlexRay Job List, maximize dependability if needed

- From our experiments:
  - FlexRay is very robust to network load increase
  - FlexRay 2.5 MBit/s might be a solution up to 10x a “regular” CAN set of signals
  - 2x CAN 500Kbit/s solutions with offsets are suited up to at most 300kbit/s of useful data (5x) but not at higher loads

References

References (1/2)

FLEXRAY – protocol and use by carmarkers

[1] B. Schätz, C. Kühnel, M. Gonschorek, “The FlexRay Protocol”, to appear in the Automotive embedded Handbook, N. Navet, F. Simonot-Lion editors, CRC Press/Taylor and Francis, 2008.

[2] Vector Informatik GmbH, interview of Mr. Peteratzinger (BMW), Mr. Steiner (BMW), “Use of XCP on FlexRay at BMW”, published in “Collection of professional articles”, 09/2006. Available at www.vector-worldwide.com/articles

[3] A. Schedl, “Goals and Architecture of FlexRay at BMW”, slides presented at the Vector FlexRay Symposium, March 6 2007.

[4] J. Broy (Porsche A.G.), K.D. Müller-Glaser, “The impact of time-triggered communication in automotive embedded systems”, IEEE SIES’2007, July 2007.

FRAME PACKING

[5] R. Saket, N. Navet, "Frame Packing Algorithms for Automotive Applications", Journal of Embedded Computing, vol. 2, n° 1, pp93-102, 2006.

DEPENDABILITY

[6] B. Gaujal, N. Navet, "Maximizing the Robustness of TDMA Networks with Applications to TTP/C", Real-Time Systems, Kluwer Academic Publishers, vol 31, n°1-3, pp5-31, December 2005.

References (2/2)

CONFIGURATION OF THE STATIC SEGMENT

[6] S. Ding, N. Murakami, H. Tomiyama, H. Takada, “A GA-based scheduling method for FlexRay systems”, EMSOFT, 2005.

[7] A. Hamann, R. Ernst, “TDMA Time Slot and Turn Optimization with Evolutionary Search Techniques”, Proceedings of the Design, Automation and Test in Europe Conference, Volume 1, p312–317, 2005.

[8] E. Wandeler, L. Thiele, “Optimal TDMA time slot and cycle length allocation for hard real-time systems”, Proceedings of the 2006 conference on Asia South Pacific design automation.

[9] M. Grenier, L. Havet, N. Navet, “Configuring the communication on FlexRay: the case of the static segment”, extended version of a paper published at ERTS’2008, available at http://www.realtimeatwork.com

CONFIGURATION OF THE DYNAMIC SEGMENT

[10] T. Pop, P. Pop, P. Eles, Z. Peng, A. Andrei, “Timing Analysis of the FlexRay Communication Protocol”, ECRTS 2006.

[11] T. Pop, P. Pop, P. Eles, Z. Peng, “Bus Access Optimisation for FlexRay-based Distributed Embedded Systems”, DATE 2007.

INTERFERENCE OF SCS TASKS ON FPS TASKS

[12] T. Pop, P. Pop, P. Eles, Z. Peng, “Optimization of Hierarchically Scheduled Heterogeneous Embedded Systems”, RTCSA’2005.

Questions / feedback ?

Please get in touch at:

nicolas.navet@realtimeatwork.com

http://www.realtimeatwork.com