

Dassault-Aviation AADL
experimentations feed-back
(part 2)

Serge Bruillot

Überlingen, 11th of July 2006

Dassault-Aviation Experimentation

- Experimentation#1 : AADL description of a simplified Mission System
 - general architecture description
 - general architecture refinement
- Experimentation#2 : Behavioral analysis of a system or how to translate :
 - an AADL description of a system into ...
 - a Temporal Petri Net model
- ASSERT involvement (MA3S subproject)



Today's presentation

Simplified Mission System description

(Experimentation#1)

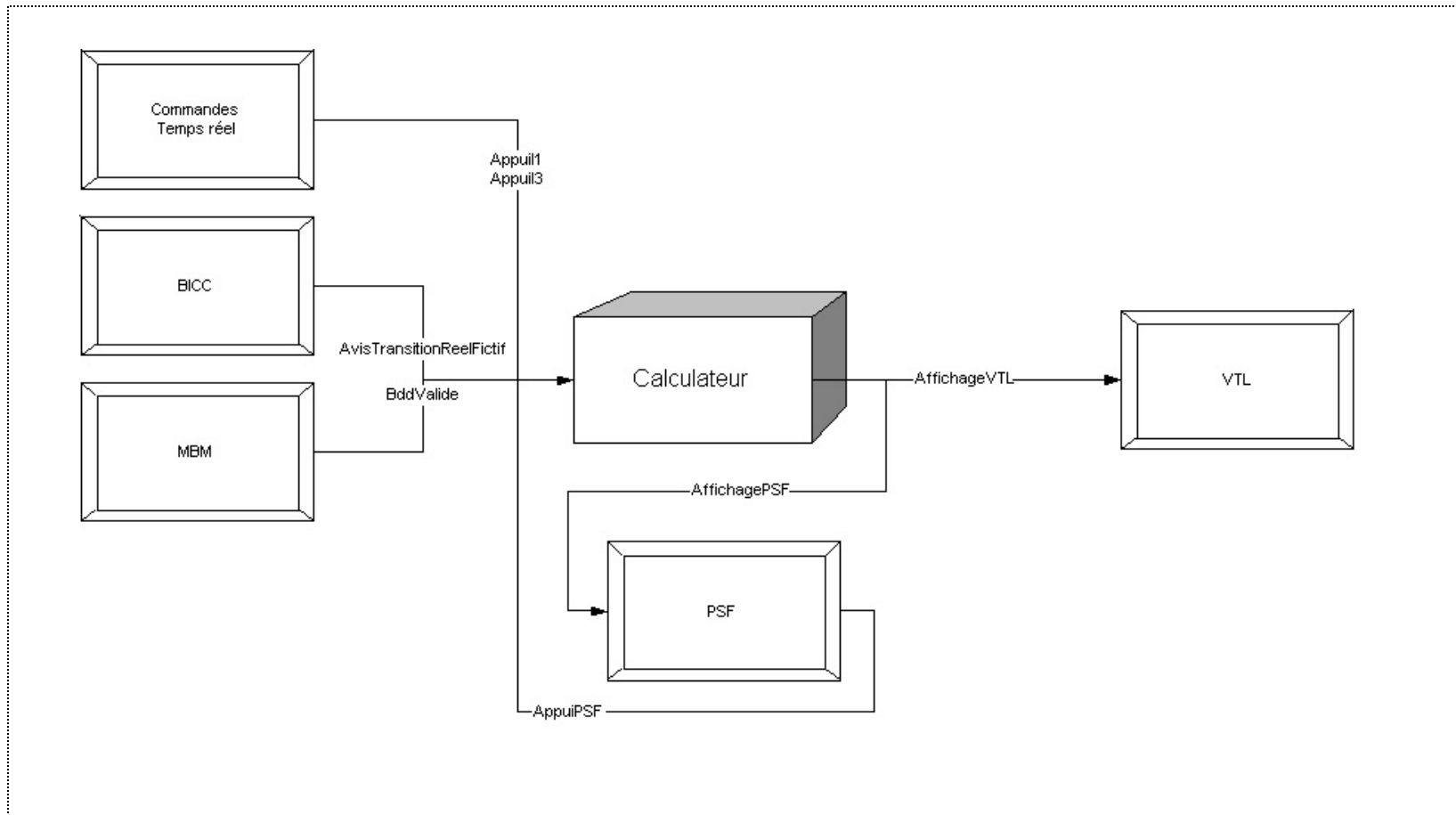
Experimentation#1 motivations

- Check the AADL capability to describe a Mission System, at different development levels
- Simulate the refinement of MS architecture component, as provided by a sub-contractor
- Establish a preliminary AADL modeling method
- Provide a feed-back to SAE AADL Task Group

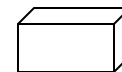
How experimentation#1 was done

- 1st set of AADL models developed by a SW engineer, with notions of avionics architecture design :
 - Mission System HW description
 - Mission Application description
- 2nd set of AADL models developed by a SW engineer, middleware and digital technology expert :
 - Mission System HW description re arrangement
 - Mission System component (HW & SW) refinement

The real system : a simplified Mission System

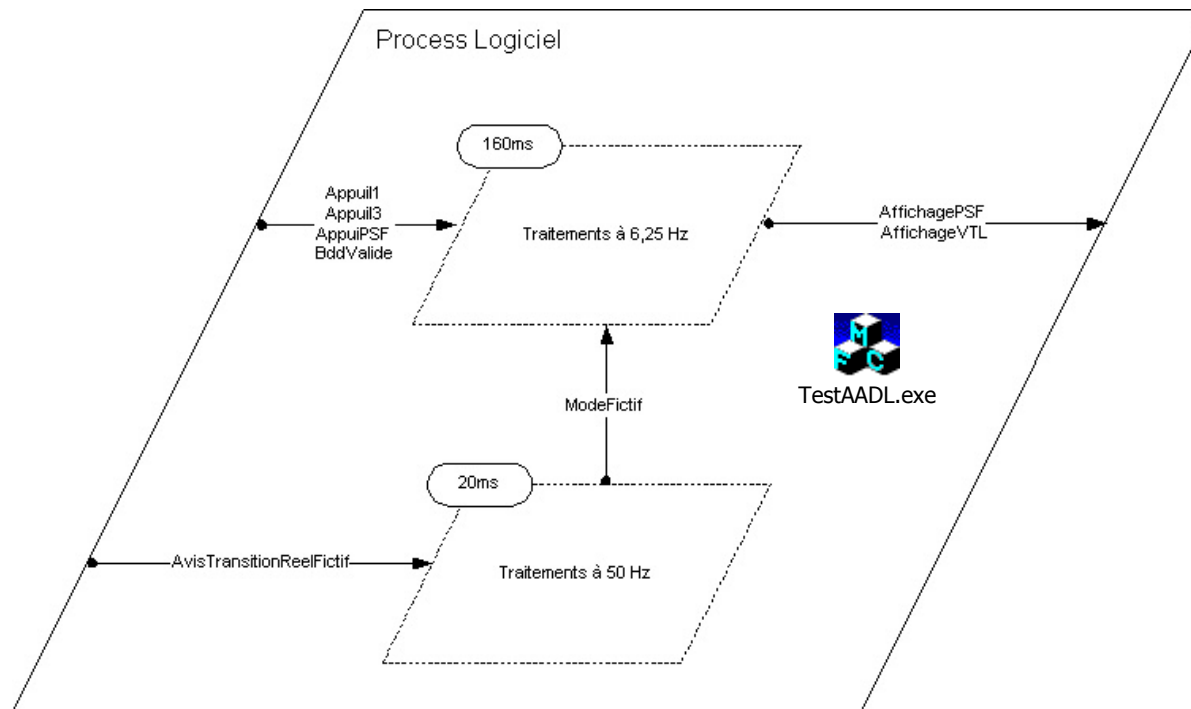


COTS units



Open Execution Platform

... controlled by a simplified Mission Application



Experimentation#1 - part1

Mission System HW description

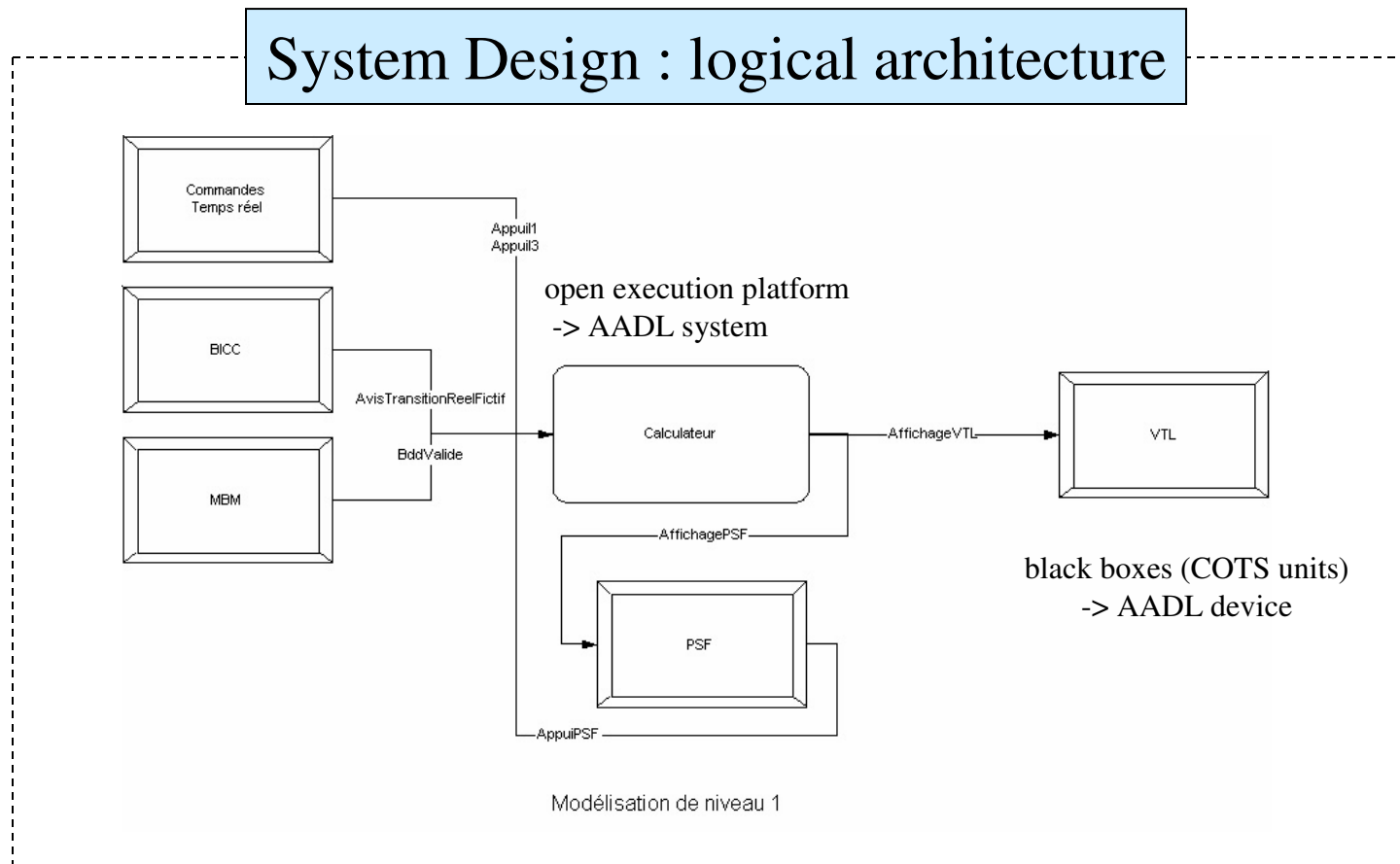
Mission Application description

Modeling process

- Step by step modeling
- models developed with OSATE tool
- textual modeling

MS AADL modeling step 1

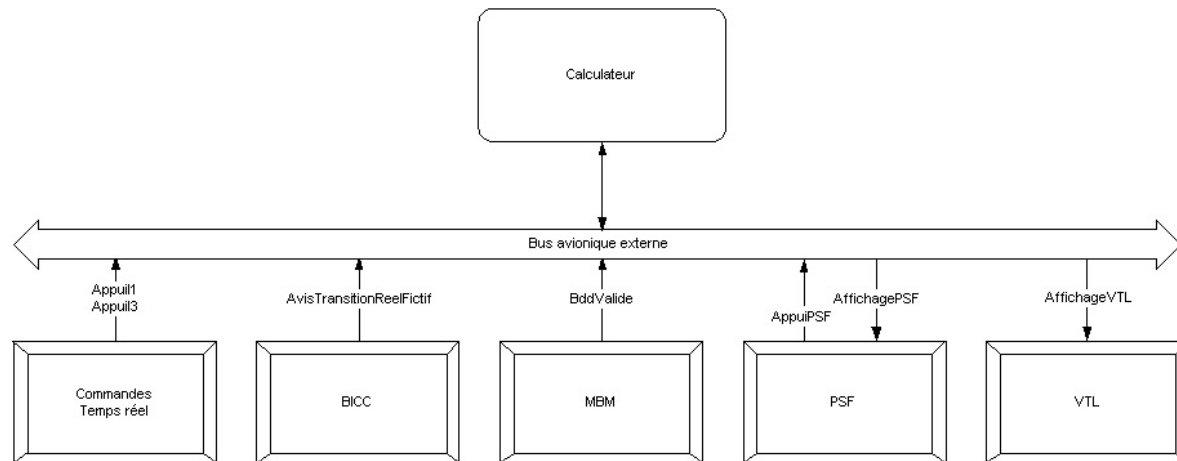
Global static description of the system



MS AADL modeling step 2

*Description of the information exchange solution :
avionics digital bus*

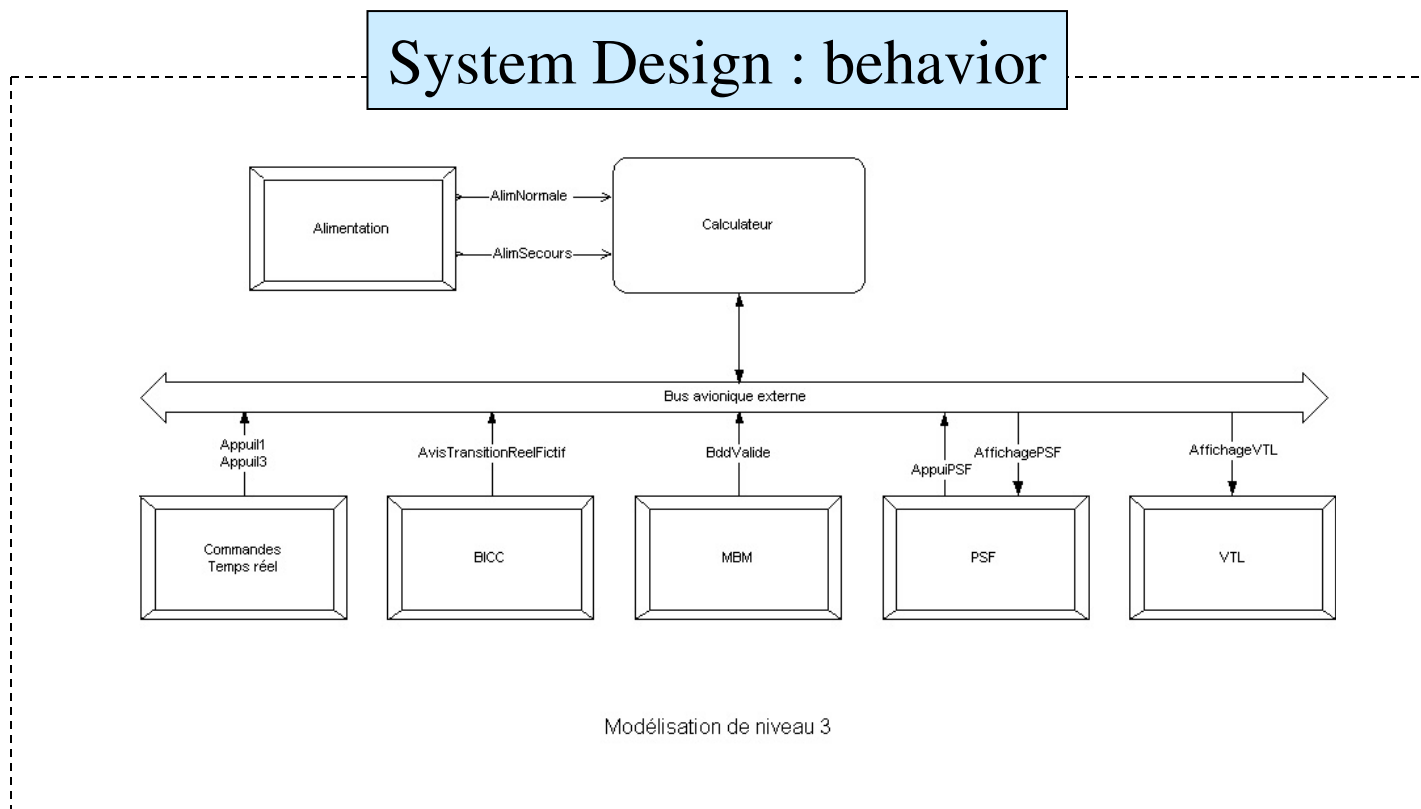
System Design : physical architecture



Modélisation de niveau 2

MS AADL modeling step 3

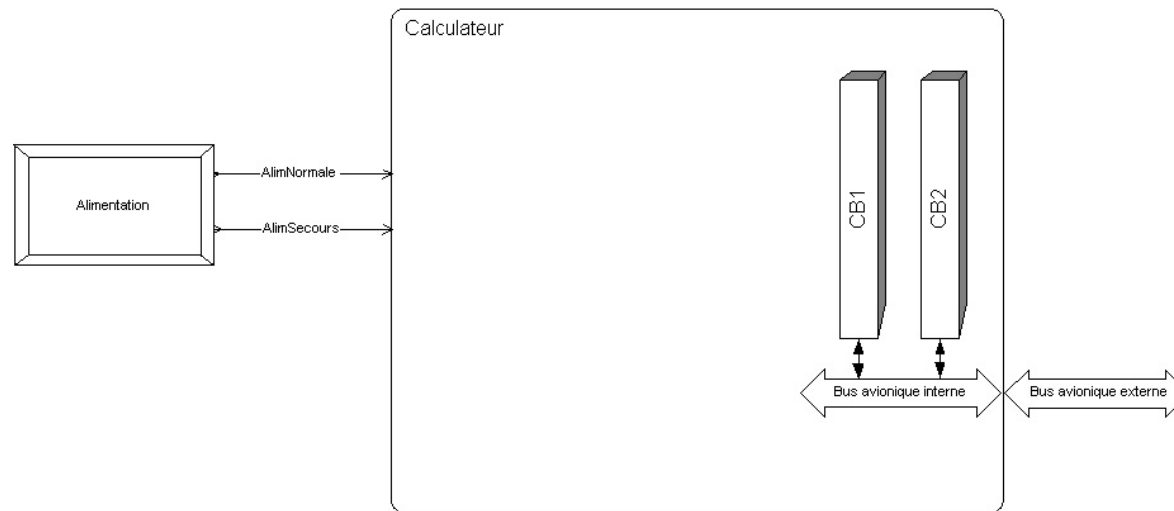
*Description of the computer power supply management
and computer modes of operations*



MS AADL modeling step 4

Description of the IMA computer - avionics bus connection

Subsystem integration : physical architecture

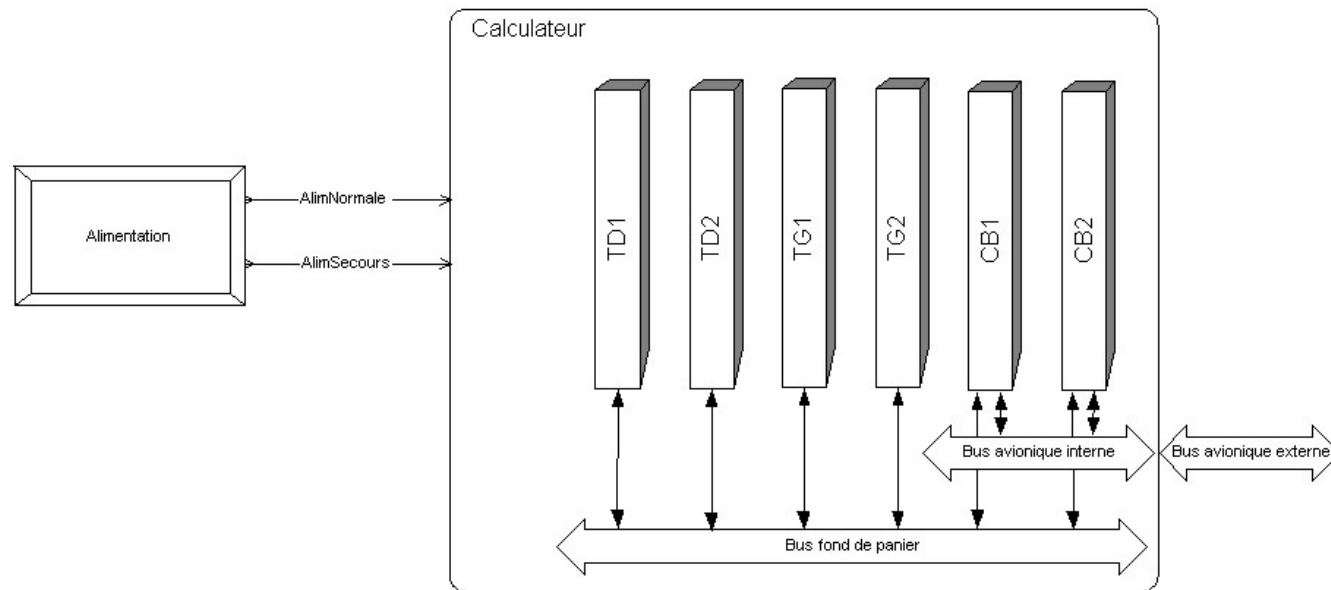


Modélisation de niveau 4

MS AADL modeling step 5

Description of the IMA computer - processing modules and behavior

Subsystem integration : physical architecture



Modélisation de niveau 5

Experimentation#1 - part2

Mission System HW description re arrangement
Mission System component (HW & SW) refinement

Experimentation goals (1/2)

- Feasibility as system designer and integrator of a cooperative AADL Mission System description
 - emulation of a third-party description, in charge of:
 - the data processing module development
 - the User Application integration
 - refinement of the previous MS model
 - some adaptations realized
 - validation of a first set of AADL modeling rules

Experimentation goals (2/2)

- Evaluation of AADL capability to:
 - describe the Data Processing Module as an Execution Platform:
 - HW description
 - Middleware abstraction
 - describe the User Application SW:
 - SW architecture description
 - I/O
 - precise the required parameters for UA integration verification:
 - schedulability analysis
 - sizing analysis

MS AADL model expected benefits (1/2)

- To represent the MS at different development steps:
 - Requirement Analysis / Component Specifications
 - Component Design / Product Realization
- To describe several design levels of the MS:
 - Functional Structure
 - Logical and Physical Architectures

MS AADL model expected benefits (2/2)

- To organize multi-partners concurrent engineering:
 - several actors:
 - E2E functional chain specification responsible
 - MS architect
 - HW and / or SW components providers
 - SW integrator
 - MS integrator
 - a unique description for:
 - MS specification consolidation and integration preparation:
 - temporal behavior, functional E2E chain delays verification
 - networks sizing and messages ordering
 - Components design and development

Model restructuring

- Previous MS model:
 - one file for all the system components
 - valid description but not compliant with the requirements:
 - parallel and independent work
 - on different system level representations
 - for the different components
- New Model proposed structure:
 - hierarchical representation, MS component based: some kind of “component tree” view
 - “heritage links” between component IFs and component implementation

Component tree creation (1/2)

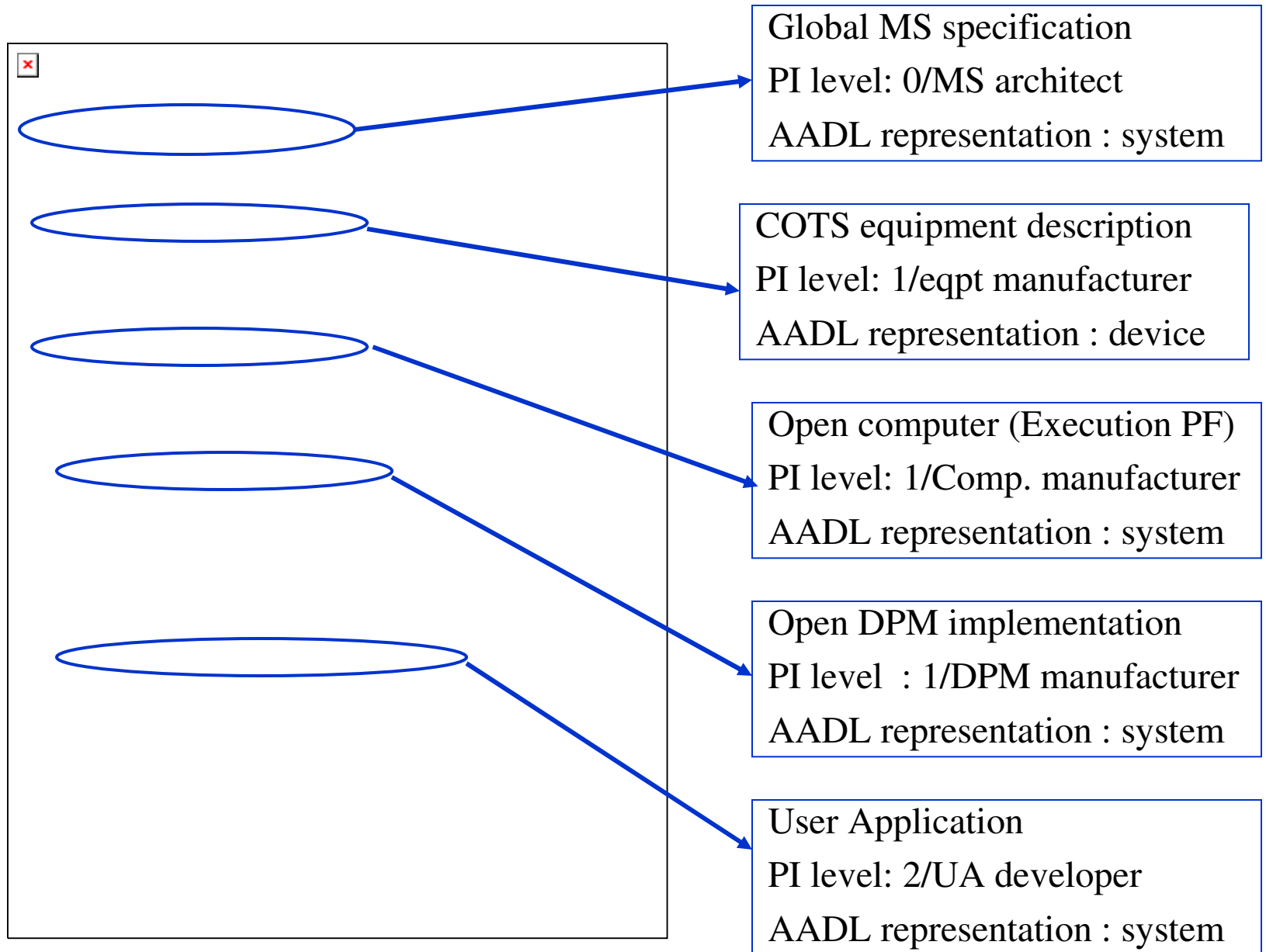
- One MS component: one file
- Files arranged in AADL MS model directory:
 - to provide the hierarchical view, compliant with:
 - the different integration levels
 - industrial work share & actors' roles
- Use of AADL packages:
 - defining naming spaces: MS.Computer.DPM

```
package SysCompleet::SysCalc::SysTD
  public
  system mon_module_TD
  ...
end mon_module_TD
```
 - allowing component both descriptions:
 - public part, external to the the package (interfaces)
 - private part, internal to the package (implementation)

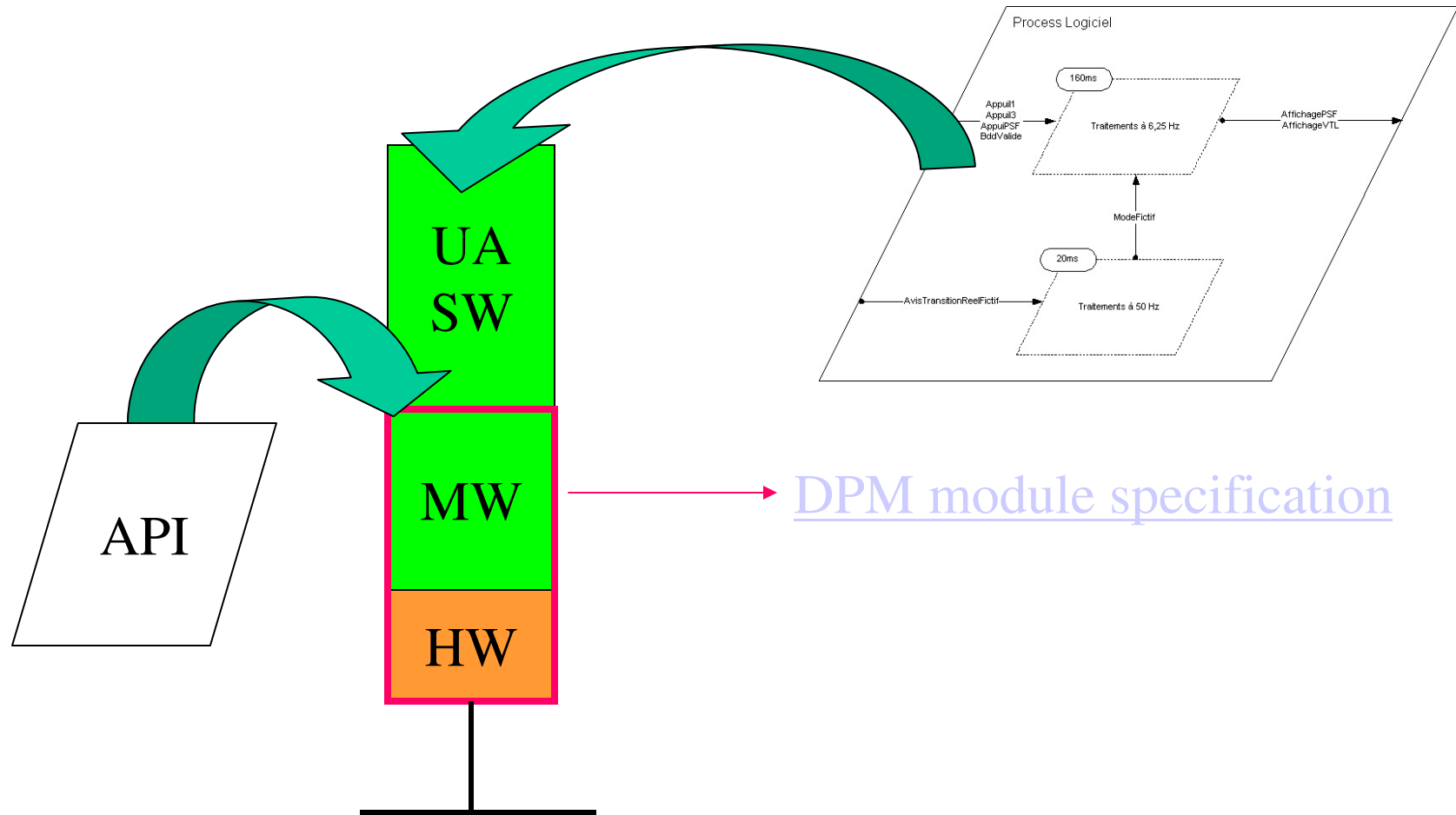
Component tree creation (2/2)

- Component description:
 - use of *devices* for COTS equipment (black boxes)
 - use of *systems* when the implementation description is required (e.g. open modular computers)
- For component described as *systems*:
 - 1st part: MS.base.aadl --> logical view (communication VC, processing generic resources, ...)
 - 2nd part : MS.derive.aadl --> one possible implementation (Ethernet MSN, PowerPC μ P, ...)

Simplified MS component tree



DPM description



[DPM module specification](#)

Modular Computer back-plane bus

Some system verifications (1/2)

- E2E chain delay computation:
 - based on system *flows* identification & analysis

flows

```
data_stream : end to end flow Calc.data_stream_out ->  
raw_data -> TD.data_computation_stream -> computed_data ->  
CB.data_stream_in;
```

- requires access to :
 - bus latencies and I/O
 - activation delays & computation time
 - times of exchanges : **ONLY available, at thread level, for AADL systems (not for devices)**

Some system verifications (2/2)

- Bus load analysis based on:
 - exchanged data volumes (at *port* level)
 - data exchange frequency:
 - use of a new property *dataFrequency*
 - define an associated unit *DataPerSecond*
- ```
{ AASM_Props::dataFrequency => 6.25 DataPerSec; };
```
- exchange support : must be the same (parser criteria)

# Experimentation feed-back (1/2)

- Bus access difficult to describe:
  - data port description performed at sub-system level:
    - data port in component IF : *in out data port*
  - bus access port performed in component interface:
    - *requires bus access*
  - connection between different data ports (to choose the bus):
  - connection between bus access port and a given bus
    - *Actual\_Connection\_Binding*

Description to be repeated at each component level  
and inside each subsystem on which a flow is projected

# Experimentation feed-back (2/2)

- Scheduler mechanism described at *processor* component level, instead at OS level
  - no property associated to the *processor* such as :
    - frequency clock, cache size, computing capacity
  - but additional properties offered at thread level for :
    - scheduling characteristics:
      - *Period, Dispatch\_Protocol, Deadline*
    - execution performances:
      - *compute\_execution\_time, compute\_Deadline*
- New property definition (*property\_sets*)
  - may be useful (lack in the std, specific verification issue)
  - should be limited because specific to a given user

# Conclusion

- Cooperative work demonstrated by the modular model approach based on *AADL packages*
- Improvement on device description proposed for E2E system functional chain timing analysis
- How to reduce connections definition effort ?  
Hidden behind OSATE now available graphical editor ?  
To be able to make reference to hierarchical elements ?

# Follow-up presentations for next meeting(s)

- Experimentation#2 : Behavioral analysis of a system or how to translate :
  - an AADL description of a system into ...
  - a Temporal Petri Net model
- Experimentation#1 forecast evolutions:
  - use templates for computers set with internal redundancy
  - use of ARINC653 MW

# Modelisation1 step1 - global MS

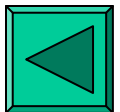
```

-- Definition du systeme complet -----

system systeme_complet
end systeme_complet;

system implementation systeme_complet.impl
 subcomponents
 BIC: device bic;
 CTR: device ctr;
 MBM: device mbm;
 PSF: device psf;
 VTL: device vtl;
 CALCULATEUR: system calculateur.impl;
 connections
 -- En entree du calculateur
 bic_calculateur: port group BIC.bic_coupleur_out -> CALCULATEUR.from_bic;
 ctr_calculateur: port group CTR.ctr_coupleur_out -> CALCULATEUR.from_ctr;
 mbm_calculateur: port group MBM.mbm_coupleur_out -> CALCULATEUR.from_mbm;
 psf_calculateur: port group PSF.psf_coupleur_out -> CALCULATEUR.from_psf;
 -- En sortie du calculateur
 calculateur_psf: port group CALCULATEUR.to_psf -> PSF.psf_coupleur_in;
 calculateur_vtl: port group CALCULATEUR.to_vtl -> VTL.vtl_coupleur_in;
 end systeme_complet.impl;

```



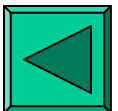
# Modelisation1 step1 - COTS units

```

-- Definition des equipements -----

-- MBM
device mbm
 features
 -- Ports d'E/S
 mbm_coupleur_out: port group mbm_infos_bus_out;
end mbm;
port group mbm_infos_bus_out
 features
 bdd_valide: out data port;
end mbm_infos_bus_out;
port group mbm_infos_bus_out_inv
 inverse of mbm_infos_bus_out
end mbm_infos_bus_out_inv;

```



# Modelisation1 step1 - Computer

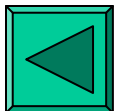
```

-- Definition du calculateur -----

-- Calculateur
system calculateur
 features
 -- Ports d'entree
 from_bic: port group bic_infos_bus_out_inv;
 from_ctr: port group ctr_infos_bus_out_inv;
 from_mbm: port group mbm_infos_bus_out_inv;
 from_psf: port group psf_infos_bus_out_inv;
 -- Ports de sortie
 to_psf: port group psf_infos_bus_in_inv;
 to_vtl: port group vtl_infos_bus_in_inv;
end calculateur;

system implementation calculateur.impl
end calculateur.impl;

```



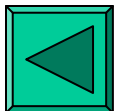
# Modelisation1 step 2 - global MS

```

-- Definition du systeme complet -----

system systeme_complet
end systeme_complet;

system implementation systeme_complet.impl
subcomponents
 BIC: device bic;
 CTR: device ctr;
 MBM: device mbm;
 PSF: device psf;
 VTL: device vtl;
 BUS_AVION_EXT: bus bus_avionique_externe;
 CALCULATEUR: system calculateur.impl;
connections
 -- Connexions au bus avionique externe
 bus access BUS_AVION_EXT -> BIC.bus_avion_ext_connector;
 bus access BUS_AVION_EXT -> CTR.bus_avion_ext_connector;
 bus access BUS_AVION_EXT -> MBM.bus_avion_ext_connector;
 bus access BUS_AVION_EXT -> PSF.bus_avion_ext_connector;
 bus access BUS_AVION_EXT -> VTL.bus_avion_ext_connector;
 bus access BUS_AVION_EXT -> CALCULATEUR.bus_avion_ext_connector;
 -- En entree du calculateur
 bic_calculateur: port group BIC.bic_coupleur_out -> CALCULATEUR.from_bic
 {Actual_Connection_Binding => reference BUS_AVION_EXT;};
 ctr_calculateur: port group CTR.ctr_coupleur_out -> CALCULATEUR.from_ctr
 {Actual_Connection_Binding => reference BUS_AVION_EXT;};
 mbm_calculateur: port group MBM.mbm_coupleur_out -> CALCULATEUR.from_mbm
 {Actual_Connection_Binding => reference BUS_AVION_EXT;};
 psf_calculateur: port group PSF.psf_coupleur_out -> CALCULATEUR.from_psf
 {Actual_Connection_Binding => reference BUS_AVION_EXT;};
 -- En sortie du calculateur
 calculateur_psf: port group CALCULATEUR.to_psf -> PSF.psf_coupleur_in
 {Actual_Connection_Binding => reference BUS_AVION_EXT;};
 calculateur_vtl: port group CALCULATEUR.to_vtl -> VTL.vtl_coupleur_in
 {Actual_Connection_Binding => reference BUS_AVION_EXT;};
end systeme_complet.impl;
```



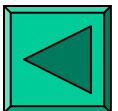
# Modelisation1 step 2- Avionics Bus

```

-- Definition des bus -----

-- Bus avionique externe au calculateur
bus bus_avionique_externe
 properties
 -- Ce bus permet de raccorder des equipements
 Allowed_Access_Protocol => Device_Access;
end bus_avionique_externe;

```



# Modelisation1 step 2 - COTS units

```

-- Definition des equipements -----

-- MBM
device mbm
 features
 -- Acces au bus avionique externe
 bus_avion_ext_connector: requires bus access bus_avionique_externe;
 -- Ports d'E/S
 mbm_coupleur_out: port group mbm_infos_bus_out;
end mbm;
port group mbm_infos_bus_out
 features
 bdd_valide: out data port;
end mbm_infos_bus_out;
port group mbm_infos_bus_out_inv
 inverse of mbm_infos_bus_out
end mbm_infos_bus_out_inv;
```



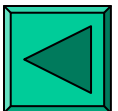
# Modelisation1 step 2 - Computer

```

-- Definition du calculateur -----

system calculateur
 features
 -- Acces au bus avionique externe
 bus_avion_ext_connector: requires bus access bus_avionique_externe;
 -- Ports d'E/S
 from_bic: port group bic_infos_bus_out_inv;
 from_ctr: port group ctr_infos_bus_out_inv;
 from_mbm: port group mbm_infos_bus_out_inv;
 from_psf: port group psf_infos_bus_out_inv;
 to_psf: port group psf_infos_bus_in_inv;
 to_vtl: port group vtl_infos_bus_in_inv;
end calculateur;

system implementation calculateur.impl
end calculateur.impl;
```



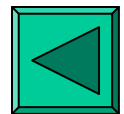
# Modelisation1 step 3 - global MS

```

-- Definition du systeme complet -----

system systeme_complet
end systeme_complet;

system implementation systeme_complet.impl
 subcomponents
 BIC: device bic;
 CTR: device ctr;
 MBM: device mbm;
 PSF: device psf;
 VTL: device vtl;
 ALIM: device alimentation;
 BUS_AVION_EXT: bus bus_avionique_externe;
 CALCULATEUR: system calculateur.impl;
 connections
 -- Connexions au bus avionique externe
 bus access BUS_AVION_EXT -> BIC.bus_avion_ext_connector;
 bus access BUS_AVION_EXT -> CTR.bus_avion_ext_connector;
 bus access BUS_AVION_EXT -> MBM.bus_avion_ext_connector;
 bus access BUS_AVION_EXT -> PSF.bus_avion_ext_connector;
 bus access BUS_AVION_EXT -> VTL.bus_avion_ext_connector;
 bus access BUS_AVION_EXT -> CALCULATEUR.bus_avion_ext_connector;
 -- En entree du calculateur
 bic_calculateur: port group BIC.bic_coupleur_out -> CALCULATEUR.from_bic
 {Actual_Connection_Binding => reference BUS_AVION_EXT;};
 ctr_calculateur: port group CTR.ctr_coupleur_out -> CALCULATEUR.from_ctr
 {Actual_Connection_Binding => reference BUS_AVION_EXT;};
 mbm_calculateur: port group MBM.mbm_coupleur_out -> CALCULATEUR.from_mbm
 {Actual_Connection_Binding => reference BUS_AVION_EXT;};
 psf_calculateur: port group PSF.psf_coupleur_out -> CALCULATEUR.from_psf
 {Actual_Connection_Binding => reference BUS_AVION_EXT;};
 -- En sortie du calculateur
 calculateur_psf: port group CALCULATEUR.to_psf -> PSF.psf_coupleur_in
 {Actual_Connection_Binding => reference BUS_AVION_EXT;};
 calculateur_vtl: port group CALCULATEUR.to_vtl -> VTL.vtl_coupleur_in
 {Actual_Connection_Binding => reference BUS_AVION_EXT;};
 -- Alimentation du calculateur
 alim_nor_calculateur: event port ALIM.alimentation_normale -> CALCULATEUR.alimentation_normale;
 alim_sec_calculateur: event port ALIM.alimentation_secours -> CALCULATEUR.alimentation_secours;
 end systeme_complet.impl;
```



# Modelisation1 step 3 - PSMU

```

-- Definition des equipements -----

-- Alimentation
 device alimentation
 features
 alimentation_normale: out event port;
 alimentation_secours: out event port;
 end alimentation;
```



# Modelisation1 step 3 - Computer Mgt

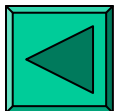
```

-- Definition de la gestion des modes et transitions -----

system gestion_systeme
 features
 -- Evenements provoquant des changements d'etats
 alimentation_normale: in event port;
 alimentation_secours: in event port;
end gestion_systeme;

system implementation gestion_systeme.impl
 modes
 -- Definition des modes
 mode_nominal: initial mode; -- 6 modules du calculateur sont alimentes
 mode_degrade: mode; -- 3 modules du calculateur sont alimentes
 -- Definition des transitions entre les modes
 mode_nominal-[alimentation_secours]->mode_degrade;
 mode_degrade-[alimentation_normale]->mode_nominal;
end gestion_systeme.impl;

```



# Modelisation1 step 4 - Computer

```

-- Definition du calculateur -----

system calculateur extends gestion_systeme
 features
 -- Acces au bus avionique externe
 bus_avion_ext_connector: requires bus access bus_avionique_externe;
 -- Ports d'E/S (sur le bus avionique externe)
 from_bic: port group bic_infos_bus_out_inv;
 from_ctr: port group ctr_infos_bus_out_inv;
 from_mbm: port group mbm_infos_bus_out_inv;
 from_psf: port group psf_infos_bus_out_inv;
 to_psf: port group psf_infos_bus_in_inv;
 to_vtl: port group vtl_infos_bus_in_inv;
 -- Ports d'E/S (sur le bus avionique interne)
 to_cb: port group calc_2_cb;
 from_cb: port group cb_2_calc;
end calculateur;

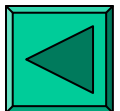
system implementation calculateur.impl extends gestion_systeme.impl
 subcomponents
 CB1: system coupleur_bus;
 CB2: system coupleur_bus;
 BUS_AVION_INT: bus bus_avionique_interne;
 connections
 -- Connexions au bus avionique interne
 bus access BUS_AVION_INT -> CB1.bus_avion_int_connector;
 bus access BUS_AVION_INT -> CB2.bus_avion_int_connector;
 -- Connexions bus avionique interne et externe
 port group from_bic -> to_cb.from_bic;
 port group from_ctr -> to_cb.from_ctr;
 port group from_mbm -> to_cb.from_mbm;
 port group from_psf -> to_cb.from_psf;
 port group from_cb.to_psf -> to_psf;
 port group from_cb.to_vtl -> to_vtl;
```

# Modelisation1 step 4 - Computer

```
-- Connexions bus avionique interne et coupleur bus 1
calc_cb1: port group to_cb -> CB1.from_calc
 {Actual_Connection_Binding => reference BUS_AVION_INT;};
cb1_calc: port group CB1.to_calc -> from_cb
 {Actual_Connection_Binding => reference BUS_AVION_INT;};
-- Connexions bus avionique interne et coupleur bus 2
calc_cb2: port group to_cb -> CB2.from_calc
 {Actual_Connection_Binding => reference BUS_AVION_INT;};
cb2_calc: port group CB2.to_calc -> from_cb
 {Actual_Connection_Binding => reference BUS_AVION_INT;};
end calculateur.impl;

port group calc_2_cb -- Sorties du calculateur vers les coupleurs bus
features
 from_bic: port group bic_infos_bus_out_inv;
 from_ctr: port group ctr_infos_bus_out_inv;
 from_mbm: port group mbm_infos_bus_out_inv;
 from_psf: port group psf_infos_bus_out_inv;
end calc_2_cb;
port group calc_2_cb_inv
 inverse of calc_2_cb
end calc_2_cb_inv;
```

---



# Modelisation1 step 4 - BC module

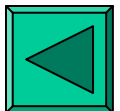
```

-- Definition des coupleurs bus -----

system coupleur_bus
 features
 -- Acces au bus avionique interne
 bus_avion_int_connector: requires bus access bus_avionique_interne;
 -- Ports d'E/S (sur le bus avionique interne)
 to_calc: port group cb_2_calc;
 from_calc: port group calc_2_cb;
end coupleur_bus;

port group cb_2_calc -- Sorties des coupleurs bus vers le calculateur
 features
 to_psf: port group psf_infos_bus_in_inv;
 to_vtl: port group vtl_infos_bus_in_inv;
end cb_2_calc;
port group cb_2_calc_inv
 inverse of cb_2_calc
end cb_2_calc_inv;

```



# Modelisation1 step 5 - BC module

```

-- Definition des coupleurs bus -----

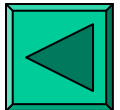
system coupleur_bus
 features
 -- Acces au bus avionique interne
 bus_avion_int_connector: requires bus access bus_avionique_interne;
 -- Acces au bus fond de panier
 bus_fond_panier_connector: requires bus access bus_fond_panier;
 -- Ports d'E/S (sur le bus avionique interne)
 to_calc: port group cb_2_calc;
 from_calc: port group calc_2_cb;
 -- Ports d'E/S (sur le bus fond de panier)
 from_td: port group td_2_cb_inv;
 from_tg: port group tg_2_cb_inv;
 to_td: port group cb_2_td;
end coupleur_bus;

system implementation coupleur_bus.impl
 connections
 port group from_tg.to_vtl -> to_calc.to_vtl; -- Les CB servent ici de passerelle
 port group from_td.to_psf -> to_calc.to_psf;
 port group from_calc.from_bic -> to_td.from_bic;
 port group from_calc.from_ctr -> to_td.from_ctr;
 port group from_calc.from_mbm -> to_td.from_mbm;
 port group from_calc.from_psf -> to_td.from_psf;
end coupleur_bus.impl;

port group cb_2_calc -- Sorties des coupleurs bus vers le calculateur
 features
 to_psf: port group psf_infos_bus_in_inv;
 to_vtl: port group vtl_infos_bus_in_inv;
end cb_2_calc;
port group cb_2_calc_inv
 inverse of cb_2_calc
end cb_2_calc_inv;

port group cb_2_td -- Sorties des coupleurs bus vers les TD
 features
 from_bic: port group bic_infos_bus_out;
 from_ctr: port group ctr_infos_bus_out;
 from_mbm: port group mbm_infos_bus_out;
 from_psf: port group psf_infos_bus_out;
end cb_2_td;
port group cb_2_td_inv
 inverse of cb_2_td
end cb_2_td_inv;

```



# Modelisation1 step 5 - DP module

```

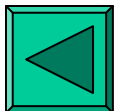
-- Definition des modules de traitements de donnees -----

system module_td
 features
 -- Acces au bus fond de panier
 bus_fond_panier_connector: requires bus access bus_fond_panier;
 -- Ports d'E/S (sur le bus fond de panier)
 from_cb: port group cb_2_td_inv;
 to_cb: port group td_2_cb;
 to_tg: port group td_2_tg;
end module_td;

port group td_2_tg -- Sorties des TD vers les TG
 features
 to_tg: port group vtl_infos_bus_in_inv;
end td_2_tg;
port group td_2_tg_inv
 inverse of td_2_tg
end td_2_tg_inv;

port group td_2_cb -- Sorties des TD vers les coupleurs bus
 features
 to_psf: port group psf_infos_bus_in_inv;
end td_2_cb;
port group td_2_cb_inv
 inverse of td_2_cb
end td_2_cb_inv;

```



# Modelisation1 step 5 - GP module

```

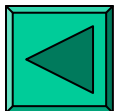
-- Definition des modules de traitements graphiques -----

system module_tg
 features
 -- Acces au bus fond de panier
 bus_fond_panier_connector: requires bus access bus_fond_panier;
 -- Ports d'E/S (sur le bus fond de panier)
 to_cb: port group tg_2_cb;
 from_td: port group td_2_tg_inv;
end module_tg;

system implementation module_tg.impl
 connections
 port group from_td -> to_cb; -- Les TG servent ici de passerelle
end module_tg.impl;

port group tg_2_cb -- Sorties des TG vers les CB
 features
 to_vtl: port group vtl_infos_bus_in_inv;
end tg_2_cb;
port group tg_2_cb_inv
 inverse of tg_2_cb
end tg_2_cb_inv;

```



# Modelisation1 step 5 - Computer backplane bus

```

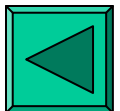
-- Definition des bus -----

-- Bus avionique externe au calculateur
bus bus_avionique_externe
 properties
 -- Ce bus permet de se connecter a des equipements
 Allowed_Access_Protocol => Device_Access;
end bus_avionique_externe;

-- Bus avionique interne au calculateur permettant de vehiculer les
-- informations du bus avion vers les coupleurs bus
bus bus_avionique_interne
 features
 bus_avion_connector: requires bus access bus_avionique_externe;
end bus_avionique_interne;

-- Bus fond de panier (interne au calculateur)
bus bus_fond_panier
end bus_fond_panier;

```



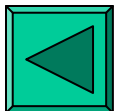
# Modelisation1 step 5 - Computer mgt

```
system implementation calculateur.impl extends gestion_systeme.impl
subcomponents
 -- En mode nominal, tous les modules sont alimentes (donc operationnels)
 -- En mode degrade, seuls TD2,TG2 et CB2 sont alimentes (donc operationnels)
 TD1: system module_td in modes (mode_nominal);
 TD2: system module_td in modes (mode_nominal,mode_degrade);
 TG1: system module_tg.impl in modes (mode_nominal);
 TG2: system module_tg.impl in modes (mode_nominal,mode_degrade);
 CB1: system coupleur_bus.impl in modes (mode_nominal);
 CB2: system coupleur_bus.impl in modes (mode_nominal,mode_degrade);
 BUS_AVION_INT: bus bus_avionique_interne;
 BUS_FOND_PANIER: bus bus_fond_panier;
connections
-- Connexions des coupleurs bus au bus avionique interne
 bus access BUS_AVION_INT -> CB1.bus_avion_int_connector in modes (mode_nominal);
 bus access BUS_AVION_INT -> CB2.bus_avion_int_connector in modes (mode_nominal,mode_degrade);
-- Connexions des 6 modules au bus fond de panier
 bus access BUS_FOND_PANIER -> TD1.bus_fond_panier_connector in modes (mode_nominal);
 bus access BUS_FOND_PANIER -> TD2.bus_fond_panier_connector in modes (mode_nominal,mode_degrade);
 bus access BUS_FOND_PANIER -> TG1.bus_fond_panier_connector in modes (mode_nominal);
 bus access BUS_FOND_PANIER -> TG2.bus_fond_panier_connector in modes (mode_nominal,mode_degrade);
 bus access BUS_FOND_PANIER -> CB1.bus_fond_panier_connector in modes (mode_nominal);
 bus access BUS_FOND_PANIER -> CB2.bus_fond_panier_connector in modes (mode_nominal,mode_degrade);
-- Connexions bus avionique interne et externe
 port group from_bic -> to_cb.from_bic;
 port group from_ctr -> to_cb.from_ctr;
 port group from_mbm -> to_cb.from_mbm;
 port group from_psf -> to_cb.from_psf;
 port group from_cb.to_psf -> to_psf;
 port group from_cb.to_vtl -> to_vtl;
-- Echange d'infos sur le bus avionique interne
-- Connexions bus avionique interne et coupleur bus 1
 calc_cb1: port group to_cb -> CB1.from_calc
 {Actual_Connection_Binding => reference BUS_AVION_INT;}
 in modes (mode_nominal);
 cb1_calc: port group CB1.to_calc -> from_cb
 {Actual_Connection_Binding => reference BUS_AVION_INT;}
 in modes (mode_nominal);
-- Connexions bus avionique interne et coupleur bus 2
 calc_cb2: port group to_cb -> CB2.from_calc
 {Actual_Connection_Binding => reference BUS_AVION_INT;}
 in modes (mode_degrade);
 cb2_calc: port group CB2.to_calc -> from_cb
 {Actual_Connection_Binding => reference BUS_AVION_INT;}
 in modes (mode_degrade);
```

# Modelisation1 step 5 - Computer mgt

```
-- Connexions bus avionique interne et coupleur bus 1
calc_cb1: port group to_cb -> CB1.from_calc
 {Actual_Connection_Binding => reference BUS_AVION_INT;}
 in modes (mode_nominal);
cb1_calc: port group CB1.to_calc -> from_cb
 {Actual_Connection_Binding => reference BUS_AVION_INT;}
 in modes (mode_nominal);
-- Connexions bus avionique interne et coupleur bus 2
calc_cb2: port group to_cb -> CB2.from_calc
 {Actual_Connection_Binding => reference BUS_AVION_INT;}
 in modes (mode_degrade);
cb2_calc: port group CB2.to_calc -> from_cb
 {Actual_Connection_Binding => reference BUS_AVION_INT;}
 in modes (mode_degrade);
-- Echange d'infos sur le bus fond de panier
-- Connexions TG1 et CB1
tg1_cb1: port group TG1.to_cb -> CB1.from_tg
 {Actual_Connection_Binding => reference BUS_FOND_PANIER;}
 in modes (mode_nominal);
-- Connexions TG2 et CB2
tg2_cb2: port group TG2.to_cb -> CB2.from_tg
 {Actual_Connection_Binding => reference BUS_FOND_PANIER;}
 in modes (mode_degrade);
-- Connexions TD1 et TG1
td1_tg1: port group TD1.to_tg -> TG1.from_td
 {Actual_Connection_Binding => reference BUS_FOND_PANIER;}
 in modes (mode_nominal);
-- Connexions TD2 et TG2
td2_tg2: port group TD2.to_tg -> TG2.from_td
 {Actual_Connection_Binding => reference BUS_FOND_PANIER;}
 in modes (mode_degrade);
-- Connexions CB1 et TD1
cb1_td1: port group CB1.to_td -> TD1.from_cb
 {Actual_Connection_Binding => reference BUS_FOND_PANIER;}
 in modes (mode_nominal);
td1_cb1: port group TD1.to_cb -> CB1.from_td
 {Actual_Connection_Binding => reference BUS_FOND_PANIER;}
 in modes (mode_nominal);
-- Connexions CB2 et TD2
cb2_td2: port group CB2.to_td -> TD2.from_cb
 {Actual_Connection_Binding => reference BUS_FOND_PANIER;}
 in modes (mode_degrade);
td2_cb2: port group TD2.to_cb -> CB2.from_td
 {Actual_Connection_Binding => reference BUS_FOND_PANIER;}
 in modes (mode_degrade);

end calculateur.impl;
```



# Model#1 part2 - DPM description

## SysCompleat.SysCalc.SysTD.Basic.aadl

```
package SysCompleat::SysCalc::SysTD::Basic
public
 bus TD_Memory_Bus extends SysCompleat::SysCalc::SysTD::HW::TD_Memory_Bus_PowerPC
 end TD_Memory_Bus;

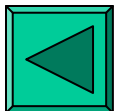
 bus TD_Device_Bus extends SysCompleat::SysCalc::SysTD::HW::TD_Device_Bus_PowerPC
 end TD_Device_Bus;

 memory TD_Ram extends SysCompleat::SysCalc::SysTD::HW::TD_Ram_PowerPC
 end TD_Ram;

 processor TD_Proc extends SysCompleat::SysCalc::SysTD::HW::TD_PowerPC
 end TD_Proc;

system module
end module;

system implementation module.impl
 subcomponents
 Proc : processor TD_Proc;
 Mem : memory TD_Ram;
 Dev_Bus : bus TD_Device_Bus;
 Mem_Bus : bus TD_Memory_Bus;
 -- prise en compte des caracteristiques SE + LdB
 LdB : process SysCompleat::SysCalc::SysTD::LdBProc::LdB_Process.impl;
 end module.impl;
end SysCompleat::SysCalc::SysTD::Basic;
```



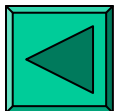
# Model#1 part2 - DPM description

## SysCompleet.SysCalc.SysTD.aadl

```
package SysCompleet::SysCalc::SysTD
public
 system module_td extends SysCompleet::SysCalc::SysTD::Basic::module
 features
 -- Acces au bus fond de panier
 bus_fond_panier_connector: requires bus access SysCompleet::SysCalc::BusFdPan::bus_fond_panier;
 -- Ports d'E/S (sur le bus fond de panier)
 from_cb: port group SysCompleet::PortGrp::cb_2_td_inv;
 to_cb: port group SysCompleet::PortGrp::td_2_cb;
 to_tg: port group SysCompleet::PortGrp::td_2_tg;
 flows
 data_stream : flow path from_cb -> to_tg;
 result_stream : flow path from_cb -> to_tg;
 end module_td;

 system implementation module_td.impl extends SysCompleet::SysCalc::SysTD::Basic::module.impl
 subcomponents
 Prog : process SysCompleet::SysCalc::SysTD::ProcAasm::Compute_Process.impl;
 connections
 fp1 : port group from_cb -> Prog.from_cb;
 fp2 : port group Prog.to_cb -> to_cb;
 fp3 : port group Prog.to_tg -> to_tg;
 flows
 data_stream : flow path from_cb -> Prog.data_stream;
 result_stream : flow path Prog.calc_stream -> to_tg;
 end module_td.impl;

end SysCompleet::SysCalc::SysTD;
```



# Model#1 part2 - DPM HW implement<sup>ation</sup>#1

## SysComple<sup>t</sup>.SysCalc.SysTD.HW.aadl

```
package SysComplet::SysCalc::SysTD::HW
public
 -- PowerPC target
 bus TD_Memory_Bus_PowerPC
 properties
 Allowed_Access_Protocol => Memory_Access;
 end TD_Memory_Bus_PowerPC;

 bus TD_Device_Bus_PowerPC
 properties
 Allowed_Access_Protocol => Device_Access;
 end TD_Device_Bus_PowerPC;

 memory TD_Ram_PowerPC
 features
 memory_access : requires bus access TD_Memory_Bus_PowerPC;
 DMA_ctrl : requires bus access TD_Device_Bus_PowerPC;
 end TD_Ram_PowerPC;

 processor TD_PowerPC
 features
 -- Again needs a bus to connect it to the RAM
 memory_ctrl : requires bus access TD_Memory_Bus_PowerPC;
 -- Needs access to bus that connects all the devices to the processor
 south_bridge : requires bus access TD_Device_Bus_PowerPC;
 properties
 Process_Swap_Execution_Time => 5 ns .. 10 ns;
 Thread_Swap_Execution_Time => 5 ns .. 10 ns;
 Scheduling_Protocol => RMS ;
 Clock_Jitter => 20 us;
 Clock_Period => 100 us;
 end TD_PowerPC;
```

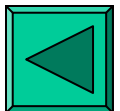
# Model#1 part2 - DPM HW implement<sup>ation</sup>#2

```
-- PENTIUM target
bus TD_Memory_Bus_PENTIUM
 properties
 Allowed_Access_Protocol => Memory_Access;
end TD_Memory_Bus_PENTIUM;

bus TD_Device_Bus_PENTIUM
 properties
 Allowed_Access_Protocol => Device_Access;
end TD_Device_Bus_PENTIUM;

memory TD_Ram_PENTIUM
 features
 memory_access : requires bus access TD_Memory_Bus_PENTIUM;
 DMA_ctrl : requires bus access TD_Device_Bus_PENTIUM;
end TD_Ram_PENTIUM;

processor TD_PENTIUM
 features
 -- Again needs a bus to connect it to the RAM
 memory_ctrl : requires bus access TD_Memory_Bus_PENTIUM;
 -- Needs access to bus that connects all the devices to the processor
 south_bridge : requires bus access TD_Device_Bus_PENTIUM;
 properties
 Process_Swap_Execution_Time => 5 ns .. 10 ns;
 Thread_Swap_Execution_Time => 5 ns .. 10 ns;
 Scheduling_Protocol => RMS ;
 Clock_Jitter => 20 us;
 Clock_Period => 100 us;
end TD_PENTIUM;
end SysComple::SysCalc::SysTD::HW;
```



# Model#1 part2 - DPM MW Process description<sup>ion</sup>

## SysCompleter.SysCalc.SysTD.LdBProc.aadl

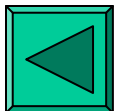
```
package SysCompleter::SysCalc::SysTD::LdBProc
public
 process LdB_Process
 end LdB_Process;

 process implementation LdB_Process.impl
 subcomponents
 LdB : thread SysCompleter::SysCalc::SysTD::LdBThread::LdB.impl;
 end LdB_Process.impl;
end SysCompleter::SysCalc::SysTD::LdBProc;
```

## SysCompleter.SysCalc.SysTD.LdBThread.aadl

```
package SysCompleter::SysCalc::SysTD::LdBThread
public
 thread LdB
 end LdB;

 thread implementation LdB.impl
 calls {
 --resource consuming
 S_LdBRTn : subprogram SysCompleter::SysCalc::SysTD::LdBCode::LdBRTn;
 };
 properties
 SysCompleterProperties::ThreadPriority => value(SysCompleterProperties::MaxThreadPriority);
 Period => 2 ms;
 Compute_Execution_Time => 1 ms .. 2 ms; -- implicate
 Compute_Deadline => 2 ms;
 Dispatch_Protocol => Periodic;
 end LdB.impl;
end SysCompleter::SysCalc::SysTD::LdBThread;
```



# Model#1 part2 - DPM MW code description

## SysCompleat.SysCalc.SysTD.LdBCode.aad1

```
package SysCompleat::SysCalc::SysTD::LdBCode
public
 subprogram LdBRtn
 properties
 Compute_Execution_Time => 25 ms .. 30 ms in binding (SysCompleat::SysCalc::SysTD::HW::TD_PowerPC);
 Compute_Execution_Time => 15 ms .. 20 ms in binding (SysCompleat::SysCalc::SysTD::HW::TD_PENTIUM);
 end LdBRtn;
 end SysCompleat::SysCalc::SysTD::LdBCode;
```

# Model#1 part2 - DPM MW code description

## SysCompleat.SubRtnLdB.aadl

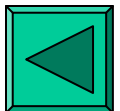
```
package SysCompleat::SubRtnLdB
public
 -- services LdB
 subprogram IntReceiveMsg
 features
 DataValue : in out parameter SysCompleat::Types::Integer;
 -- MessageId : in parameter SysCompleat::Types::Integer;
 -- MessageSize : in parameter SysCompleat::Types::Integer;
 -- Buffer : in parameter SysCompleat::Types::Integer;
 properties
 -- exemples de proprietes pour le calcul des temps de propagation
 Subprogram_Execution_Time => 4 Us .. 20 Us in binding(SysCompleat::SysCalc::SysTD::HW::TD_PowerPC);
 Subprogram_Execution_Time => 2 Us .. 10 Us in binding(SysCompleat::SysCalc::SysTD::HW::TD_PENTIUM);
 Compute_Execution_Time => 1 Us .. 10 Us;
 Compute_Deadline => 15 Us;
 end IntReceiveMsg;

 subprogram IntSendMsg
 features
 DataValue : in out parameter SysCompleat::Types::Integer;
 -- MessageId : in parameter SysCompleat::Types::Integer;
 -- MessageSize : in parameter SysCompleat::Types::Integer;
 -- Buffer : out parameter SysCompleat::Types::Integer;
 properties
 -- exemples de proprietes pour le calcul des temps de propagation
 Subprogram_Execution_Time => 4 Us .. 20 Us in binding(SysCompleat::SysCalc::SysTD::HW::TD_PowerPC);
 Subprogram_Execution_Time => 2 Us .. 10 Us in binding(SysCompleat::SysCalc::SysTD::HW::TD_PENTIUM);
 Compute_Execution_Time => 1 Us .. 10 Us;
 Compute_Deadline => 15 Us;
 end IntSendMsg;
```

# Model#1 part2 - DPM MW code description

```
subprogram FloatReceiveMsg
 features
 DataValue : in out parameter SysCompleet::Types::Float;
 -- MessageId : in parameter SysCompleet::Types::Integer;
 -- MessageSize : in parameter SysCompleet::Types::Integer;
 -- Buffer : in parameter SysCompleet::Types::Float;
 properties
 -- exemples de proprietes pour le calcul des temps de propagation
 Subprogram_Execution_Time => 4 Us .. 20 Us in binding(SysCompleet::SysCalc::SysTD::HW::TD_PowerPC);
 Subprogram_Execution_Time => 2 Us .. 10 Us in binding(SysCompleet::SysCalc::SysTD::HW::TD_PENTIUM);
 Compute_Execution_Time => 1 Us .. 10 Us;
 Compute_Deadline => 15 Us;
end FloatReceiveMsg;

subprogram FloatSendMsg
 features
 DataValue : in out parameter SysCompleet::Types::Float;
 -- MessageId : in parameter SysCompleet::Types::Integer;
 -- MessageSize : in parameter SysCompleet::Types::Integer;
 -- Buffer : out parameter SysCompleet::Types::Float;
 properties
 -- exemples de proprietes pour le calcul des temps de propagation
 Subprogram_Execution_Time => 4 Us .. 20 Us in binding(SysCompleet::SysCalc::SysTD::HW::TD_PowerPC);
 Subprogram_Execution_Time => 2 Us .. 10 Us in binding(SysCompleet::SysCalc::SysTD::HW::TD_PENTIUM);
 Compute_Execution_Time => 1 Us .. 10 Us;
 Compute_Deadline => 15 Us;
end FloatSendMsg;
end SysCompleet::SubRtnLdB;
```



# Model#1 part2 - AASM UA Process description

## SysCompleat.SysCalc.SysTD.ProcAasm.aadl

```
package SysCompleat::SysCalc::SysTD::ProcAasm
public
 process Compute_Process
 features
 -- entree est utilisee pour isoler Process du system TD
 from_cb: port group SysCompleat::PortGrp::cb_2_td_inv; -- public
 -- couplages caches a l'exterieur
 from_cb_cache: port group SysCompleat::PortGrp::cb_2_td_inv; -- privee
 -- from_cb protocol expanded
 from_cb_bic : port group SysCompleat::PortGrp::bic_infos_bus_out;
 from_cb_ctr: port group SysCompleat::PortGrp::ctr_infos_bus_out;
 from_cb_mbm: port group SysCompleat::PortGrp::mbm_infos_bus_out;
 from_cb_psf: port group SysCompleat::PortGrp::psf_infos_bus_out;
 -- to_cb
 to_cb: port group SysCompleat::PortGrp::td_2_cb;
 to_cb_psf: port group SysCompleat::PortGrp::psf_infos_bus_in_inv;
 -- to_tg
 to_tg: port group SysCompleat::PortGrp::td_2_tg;
 to_tg_vtl: port group SysCompleat::PortGrp::vtl_infos_bus_in_inv;
 flows
 data_stream : flow path from_cb -> to_tg;
 result_stream : flow path from_cb -> to_tg;
 end Compute_Process;
```

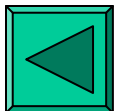
# Model#1 part2 - AASM UA Process description

```
process implementation Compute_Process.impl
 subcomponents
 T_Low_Frequency : thread
 SysCompleter::SysCalc::SysTD::ProcAasm::ThreadLF::Sampling_160ms_Thread.impl;
 T_High_Frequency : thread
 SysCompleter::SysCalc::SysTD::ProcAasm::ThreadHF::Sampling_20ms_Thread.impl;
 connections
 -- couplage entree est utilisee pour isoler Process du system TD
 port group from_cb -> from_cb_cache;
 -- couplages caches a l'exterieur
 fp1 : port group from_cb_cache.from_ctr -> from_cb_ctr;
 fp2 : data port from_cb_ctr.appui_I1 ->> T_Low_Frequency.Appui1;
 data port from_cb_ctr.appui_I3 ->> T_Low_Frequency.Appui3;

 port group from_cb_cache.from_mbm -> from_cb_mbm;
 data port from_cb_mbm.bdd_valide ->> T_Low_Frequency.BddValide;

 port group from_cb_cache.from_psf -> from_cb_psf;
 data port from_cb_psf.appui_psf ->> T_Low_Frequency.AppuiPSF;

 port group from_cb_cache.from_bic -> from_cb_bic;
 data port from_cb_bic.avis_transition_reel_fictif ->> T_High_Frequency.AvisTransitionReelFictif;
 -- for to_cb
 port group to_cb.to_psf -> to_cb_psf;
 data port T_Low_Frequency.AffichagePSF ->> to_cb_psf.affichage_psf;
 -- for to_tg
 port group to_tg.to_tg -> to_tg_vtl;
 data port T_Low_Frequency.AffichageVTL ->> to_tg_vtl.affichage_vtl;
 -- inter thread
 data port T_High_Frequency.ModeFictif ->> T_Low_Frequency.ModeFictif;
 flows
 data_stream : flow path from_cb -> T_Low_Frequency.data_stream;
 result_stream : flow path T_Low_Frequency.result_stream -> to_tg;
end Compute_Process.impl;
```



# Model#1 part2-AASM UA threadHF description

## SysCompleet.SysCalc.SysTD.ProcAasm.ThreadHF.aadl

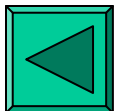
```
package SysCompleet::SysCalc::SysTD::ProcAasm::ThreadHF
public
 thread Sampling_20ms_Thread
 features
 AvisTransitionReelFictif : in data port SysCompleet::Types::Integer;
 ModeFictif : out data port SysCompleet::Types::Integer;
 end Sampling_20ms_Thread;

 thread implementation Sampling_20ms_Thread.impl
 calls {
 -- receiving msg
 R_AvisTransitionReelFictif : subprogram SysCompleet::SubRtnLdB::IntReceiveMsg;
 -- computing
 C_HF : subprogram SysCompleet::AppliAasm::Code::calcul_HF;
 -- sending msg
 S_ModeFictif : subprogram SysCompleet::SubRtnLdB::IntSendMsg;
 };
 connections
 parameter AvisTransitionReelFictif -> R_AvisTransitionReelFictif.DataValue;

 parameter R_AvisTransitionReelFictif.DataValue -> C_HF.AvisTransitionReelFictif;

 parameter C_HF.ModeFictif -> S_ModeFictif.DataValue;

 parameter S_ModeFictif.DataValue -> ModeFictif;
 properties
 SysCompleetProperties::ThreadPriority => value(SysCompleetProperties::MaxThreadPriority);
 Period => 20 ms;
 Compute_Execution_Time => 5 ms .. 10 ms; -- implicite
 Compute_Deadline => 13 ms;
 Dispatch_Protocol => Periodic;
 end Sampling_20ms_Thread.impl;
end SysCompleet::SysCalc::SysTD::ProcAasm::ThreadHF;
```



# Model#1 part2-AASM UA threadLF description

## SysCompleter.SysCalc.SysTD.ProcAasm.ThreadLF.aadl

```
package SysCompleter::SysCalc::SysTD::ProcAasm::ThreadLF
public
 thread Sampling_160ms_Thread
 features
 Appuis1 : in data port SysCompleter::Types::Integer;
 Appuis3 : in data port SysCompleter::Types::Integer;
 AppuiPSF : in data port SysCompleter::Types::Integer;
 BddValide : in data port SysCompleter::Types::Integer;
 ModeFictif : in data port SysCompleter::Types::Integer;
 AffichagePSF : out data port SysCompleter::Types::Integer;
 AffichageVTL : out data port SysCompleter::Types::Float;
 flows
 data_stream : flow sink Appuis1;
 result_stream : flow source AffichageVTL;
 end Sampling_160ms_Thread;

 thread implementation Sampling_160ms_Thread.impl
 calls {
 --receiving msg
 R_Appuis1 : subprogram SysCompleter::SubRtnLdB::IntReceiveMsg;
 R_Appuis3 : subprogram SysCompleter::SubRtnLdB::IntReceiveMsg;
 R_AppuiPSF : subprogram SysCompleter::SubRtnLdB::IntReceiveMsg;
 R_BddValide : subprogram SysCompleter::SubRtnLdB::IntReceiveMsg;
 R_ModeFictif : subprogram SysCompleter::SubRtnLdB::IntReceiveMsg;
 -- computing
 C_LF : subprogram SysCompleter::AppliAasm::Code::calcule_LF;
 -- sending msg
 S_AffichagePSF : subprogram SysCompleter::SubRtnLdB::IntSendMsg;
 S_AffichageVTL : subprogram SysCompleter::SubRtnLdB::FloatSendMsg;
 };
 end;
```

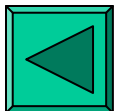
# Model#1 part2-AASM UA threadLF description

```
connections
 fp1 : parameter Appuis1 -> R_Appuis1.DataValue;
 parameter Appuis3 -> R_Appuis3.DataValue;
 parameter AppuiPSF -> R_AppuiPSF.DataValue;
 parameter BddValide -> R_BddValide.DataValue;
 parameter ModeFictif -> R_ModeFictif.DataValue;

 fp2 : parameter R_Appuis1.DataValue -> C_LF.Appuis1;
 parameter R_Appuis3.DataValue -> C_LF.Appuis3;
 parameter R_AppuiPSF.DataValue -> C_LF.AppuiPSF;
 parameter R_BddValide.DataValue -> C_LF.BddValide;
 parameter R_ModeFictif.DataValue -> C_LF.ModeFictif;

 parameter C_LF.AffichagePSF -> S_AffichagePSF.DataValue;
 fp3 : parameter C_LF.AffichageVTL -> S_AffichageVTL.DataValue;

 parameter S_AffichagePSF.DataValue -> AffichagePSF;
 fp4 : parameter S_AffichageVTL.DataValue -> AffichageVTL;
flows
 data_stream : flow sink Appuis1;
 result_stream : flow source AffichageVTL;
properties
 SysCompletProperties::ThreadPriority => value(SysCompletProperties::MinThreadPriority);
 Source_Text => "aasm_lf.exe";
 Source_Language => c;
 Source_Code_Size => 4 MB;
 Source_Data_Size => 64 kB;
 Source_Stack_Size => 64 kB;
 Source_Heap_Size => 64 kB;
 Period => 160 ms;
 Compute_Execution_Time => 50 ms .. 100 ms; -- implicite
 Compute_Deadline => 110 ms;
 Dispatch_Protocol => Periodic;
end Sampling_160ms_Thread.impl;
end SysComplet::SysCalc::SysTD::ProcAasm::ThreadLF;
```

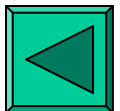


# Model#1 part2 - AASM UA code description<sup>ion</sup>

## SysCompleet .AppliAasm .Code .aadl

```
package SysCompleet::AppliAasm::Code
public
 subprogram calcule_HF
 features
 AvisTransitionReelFictif : in parameter SysCompleet::Types::Integer;
 ModeFictif : out parameter SysCompleet::Types::Integer;
 properties
 Compute_Execution_Time => 9 ms .. 10 ms in binding (SysCompleet::SysCalc::SysTD::HW::TD_PowerPC);
 Compute_Execution_Time => 1 ms .. 8 ms in binding (SysCompleet::SysCalc::SysTD::HW::TD_PENTIUM);
 end calcule_HF;

 subprogram calcule_LF
 features
 Appuis1 : in parameter SysCompleet::Types::Integer;
 Appuis3 : in parameter SysCompleet::Types::Integer;
 AppuiPSF : in parameter SysCompleet::Types::Integer;
 BddValide : in parameter SysCompleet::Types::Integer;
 ModeFictif : in parameter SysCompleet::Types::Integer;
 AffichagePSF : out parameter SysCompleet::Types::Integer;
 AffichageVTL : out parameter SysCompleet::Types::Float;
 properties
 Compute_Execution_Time => 25 ms .. 30 ms in binding (SysCompleet::SysCalc::SysTD::HW::TD_PowerPC);
 Compute_Execution_Time => 15 ms .. 20 ms in binding (SysCompleet::SysCalc::SysTD::HW::TD_PENTIUM);
 end calcule_LF;
 end SysCompleet::AppliAasm::Code;
```

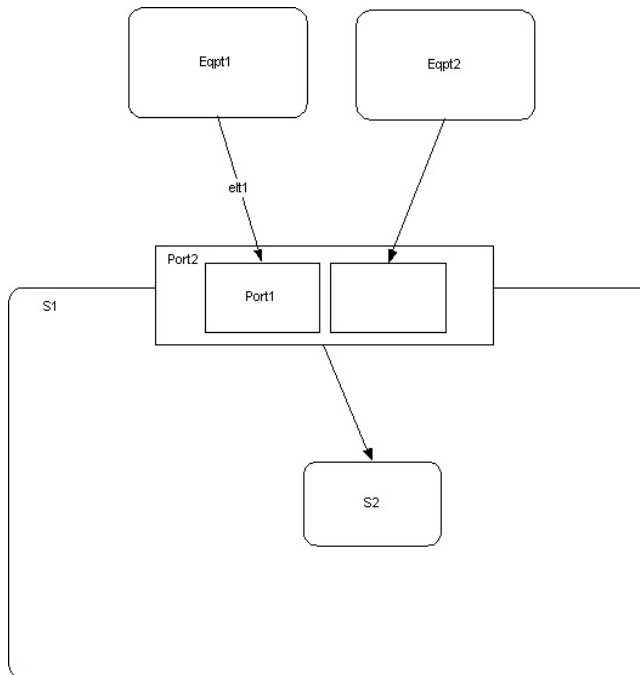


# AADL DA Experimentation

Spare slides

# Difficulty met

*It seems impossible with current AADL standard to make a reference to hierarchical elements*



- For connexion ports, it is interesting to group elementary infos in larger connexion ports, themselves grouped in a global connexion port
- to limit the number of connexions in a system, it should be better to make a reference to port 2 rather than to port 1
- but info1 access from port2, using the notation port2.port1.info1 access is forbidden ; only port1.info1 is allowed

# Conclusion

- AADL language is relatively easy to learn
- model development time is quite quick
- it seems impossible to make reference to hierarchical elements
- model complexity increases rapidly (due to the connections definition)
- although based on a formal language, a system description could be very subjective
- the availability of an AADL toolset capable of helping the modeler will be appreciate (for system definition, analysis & verification )