Synthesis of software architecture for the control of embedded electrical generation and distribution system for aircraft under safety constraints: The case of simple failures

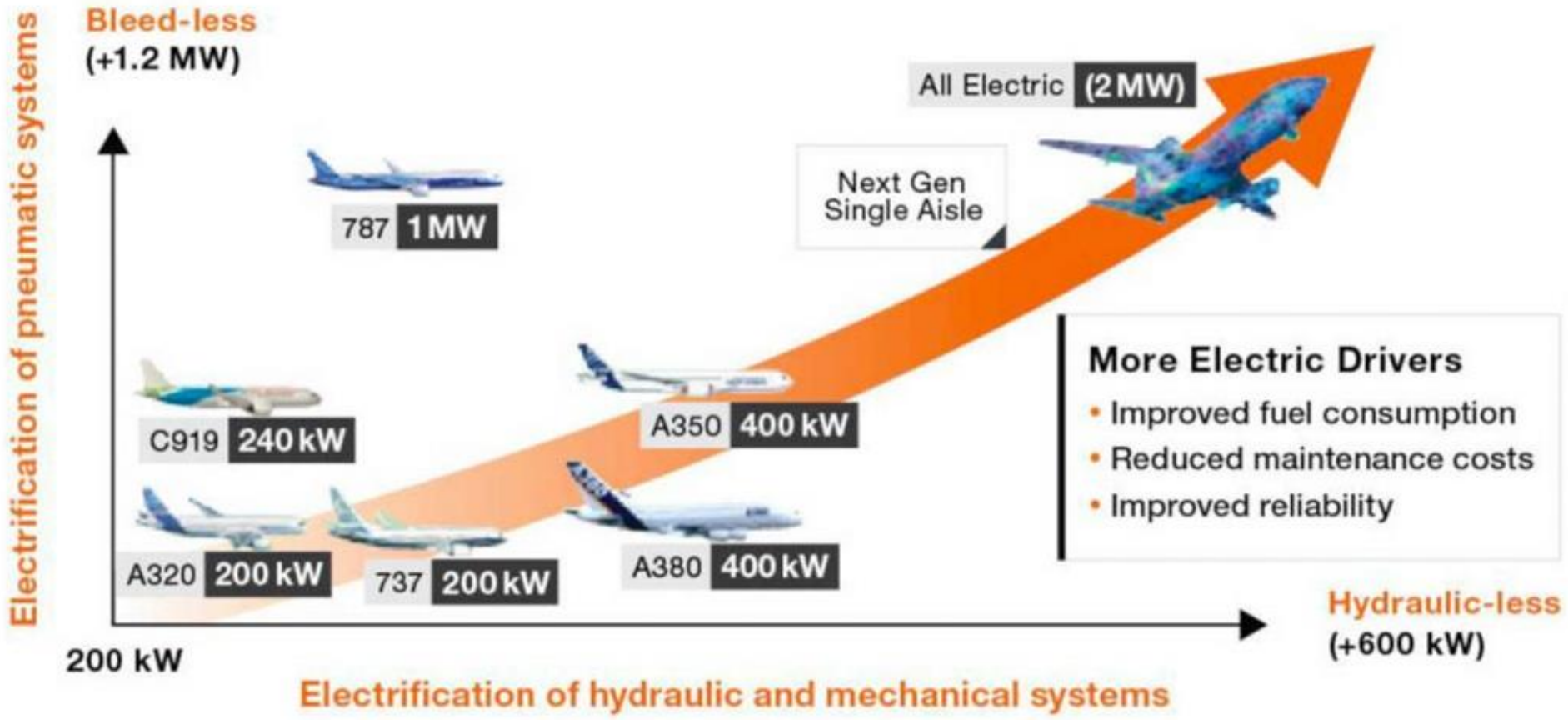
Laurent ZIMMER
Dassault Aviation
Direction de la Prospective
laurent.zimmer@dassault-aviation.com

Pierre-Alain YVARS
ISAE-Supméca
Laboratoire Quartz – EA 7393
pierre-alain.yvars@supmeca.fr

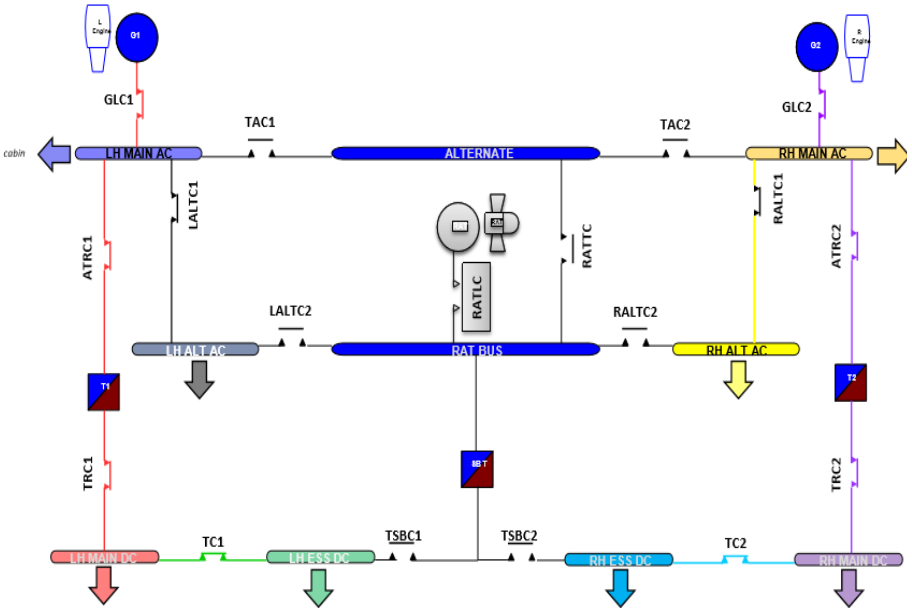
OUTLINE

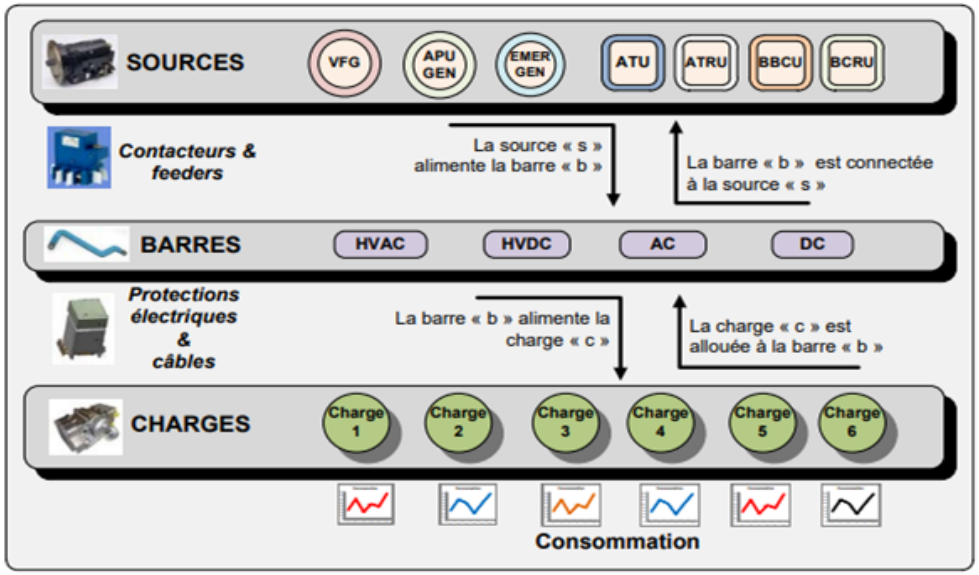
- Context
- Problem
- Synthesis approach
- Modeling the problem
- Solving and results
- Ongoing and future works

Context


Bleed-less
(+1.2 MW)
Electrification of pneumatic systems
All Electric (2 MW)
Next Gen
Single Aisle
787 1 MW
C919 240 kW
A350 400 kW
A320 200 kW
737 200 kW
A380 400 kW
More Electric Drivers
• Improved fuel consumption
• Reduced maintenance costs
• Improved reliability
Hydraulic-less
(+600 kW)
200 kW
Electrification of hydraulic and mechanical systems

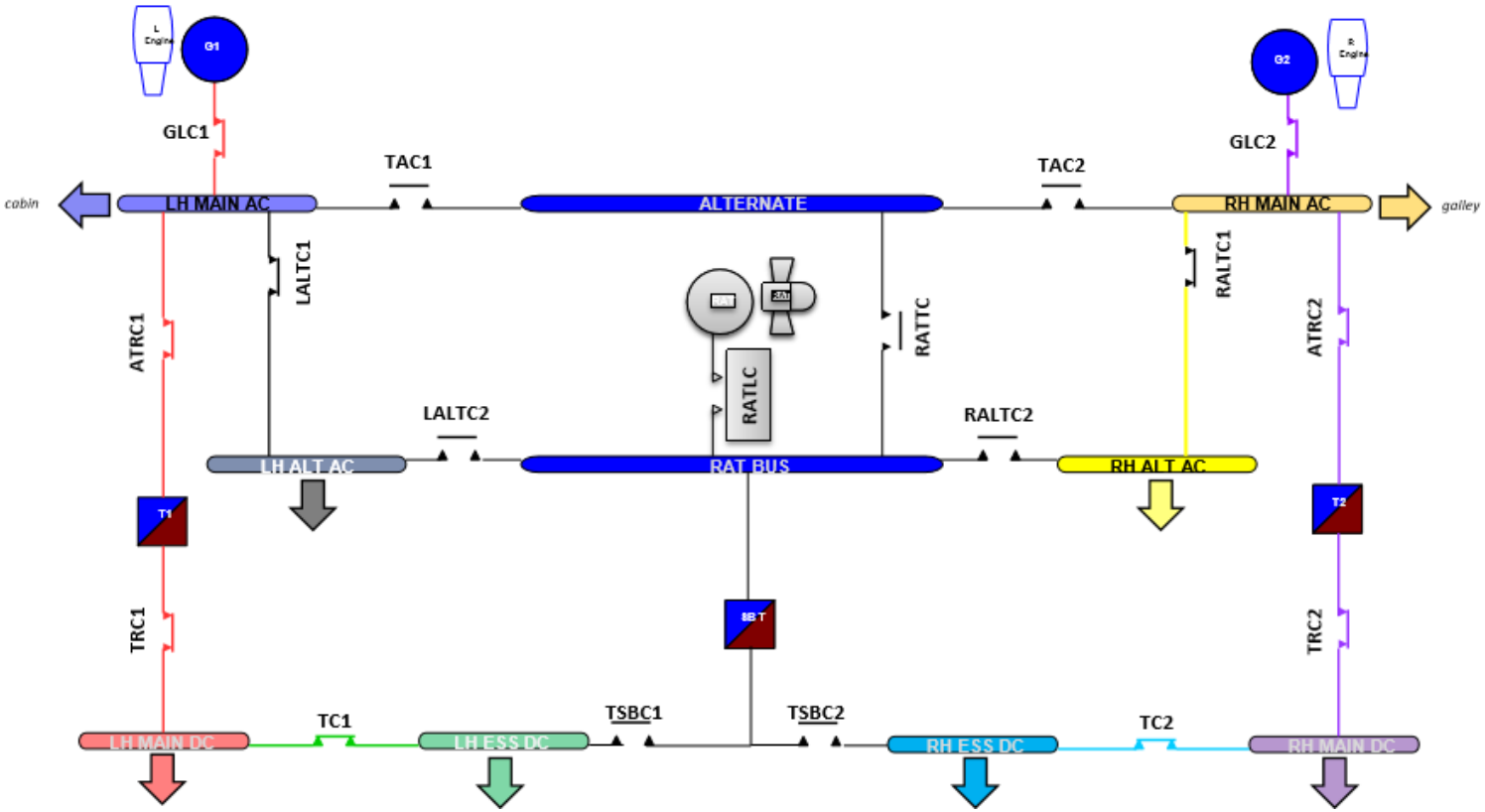
Designing aircraft electrical Power Distribution System (PDS)





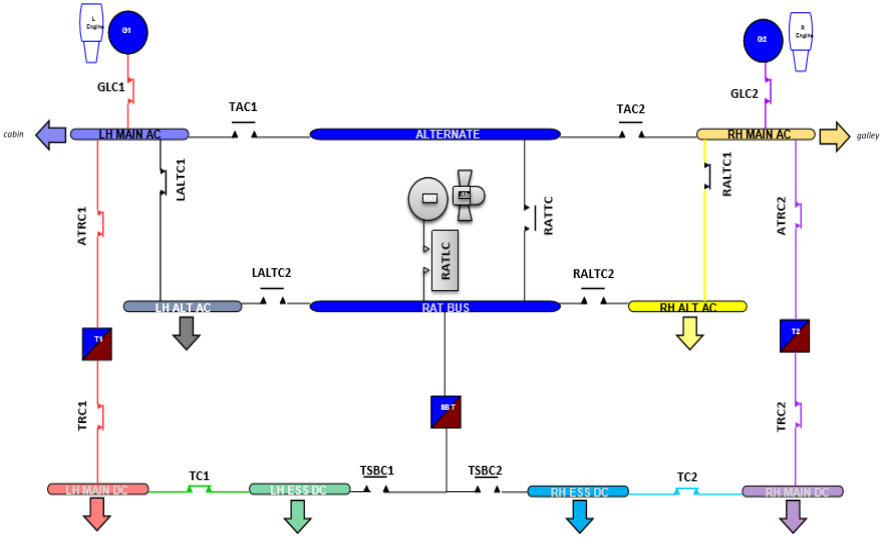
- Designing the electrical part of the PDS
- Path generation
- Allocation bar-buses to Laods
- **Deployment of safety functions on calculators**

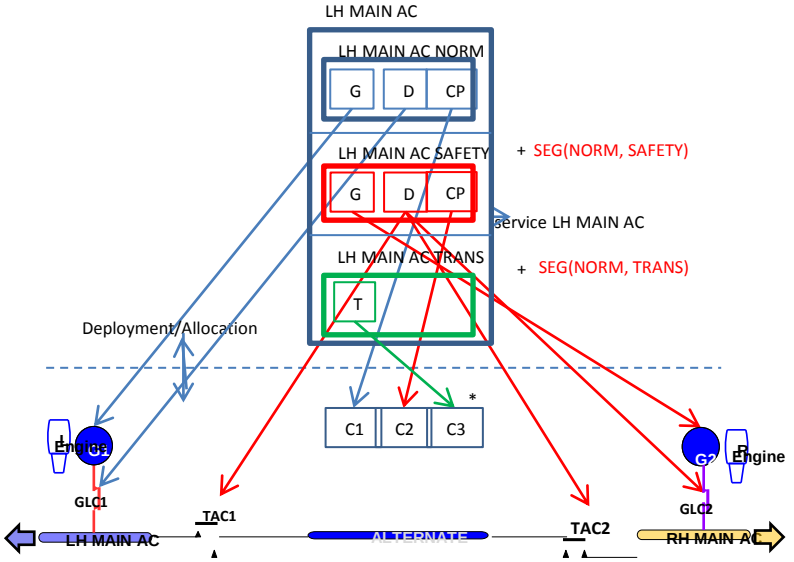
Electrical System Description



- Three generators (GLC1, GLC2, RAT)
- Eight bus bars (LH MAIN AC, LH ALT AC, LH MAIN DC, LH ESS DC, RH MAIN AC, RH ALT AC, RH MAIN DC, RH ESS DC)
- Three converters (T1, T2, SBT);
- Seventeen power contactors

Control System Description





- Processing programms running on calculators (C1, C2, …, Cn)
- Calculator = Power Supply Unit + microcontroller

Safety requirements

(R) In the case of a single failure occurring on equipment all bus bars must continue to be powered after reconfiguration of the system.

Safety requirements

(R) In the case of a single failure occurring on equipment all bus bars must continue to be powered after reconfiguration of the system.

The safety requirements are issued from a preliminary Safety Analysis of systems leading to :

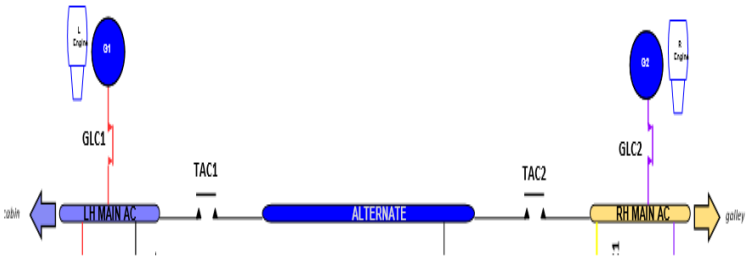
- Duplication … of channels
- Material segregation of channels

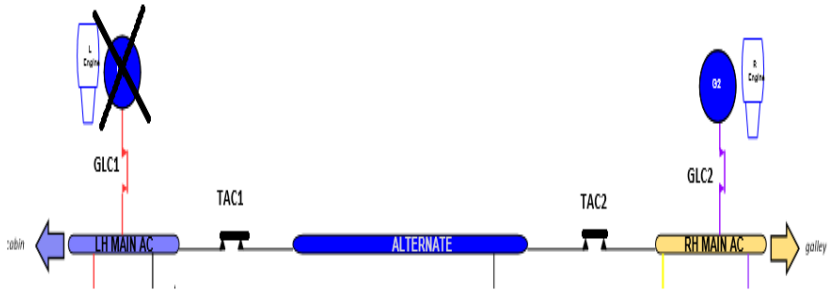
Safety requirements

(R) In the case of a single failure occurring on equipment all bus bars must continue to be powered after reconfiguration of the system.

The safety requirements are issued from a preliminary Safety Analysis of systems leading to :

- Duplication … of channels
- Material segregation of channels





Problem description

With a fixed electrical architecture:

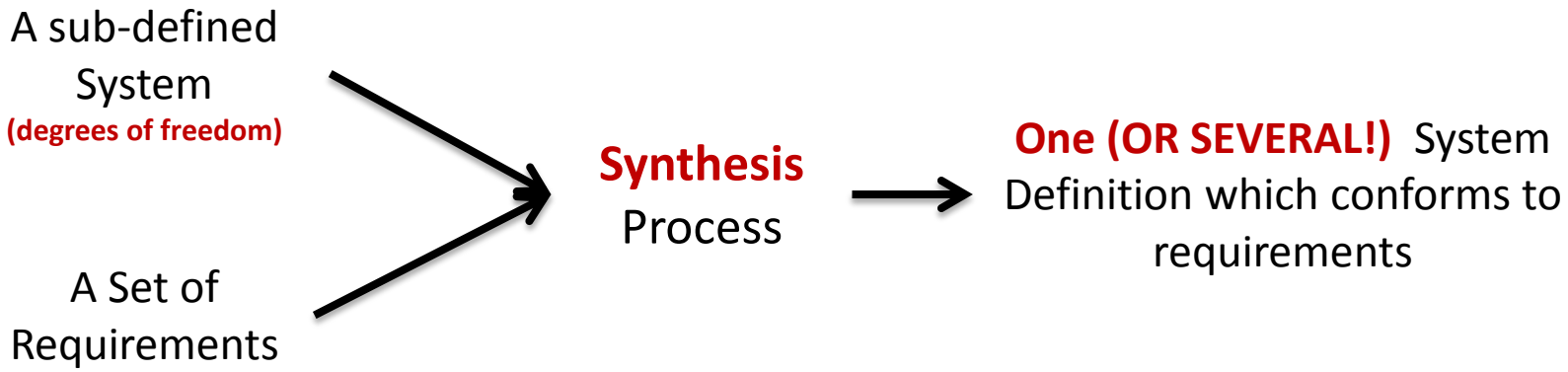
- What is the necessary and sufficient number of calculators ?
- How to allocate calculators to the control and transition programs ?
- How to allocate the contactor commands to the calculator ports ?

With:
- R1: a contactor is controlled by one and only one contactor command of a calculator
- R2: a contactor command of a calculator controls one and only one contactor

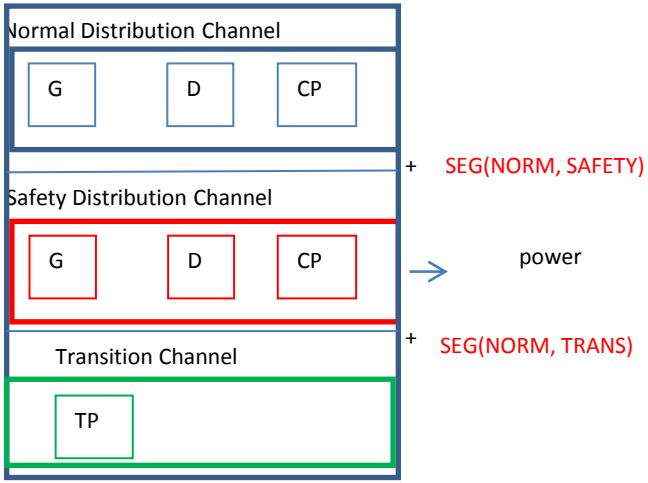
Synthesis problem
MBSS

MBSE HAS TO BE REVISED TO ADRESS SYNTHESIS PROBLEMS:

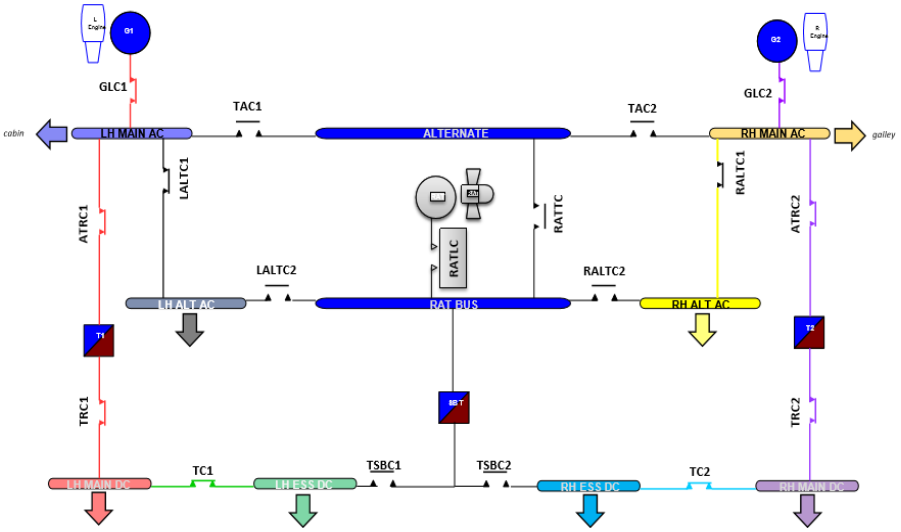
- Modeling languages to capture definitions of problem
- Synthesis software tools to fix, select, allocate …



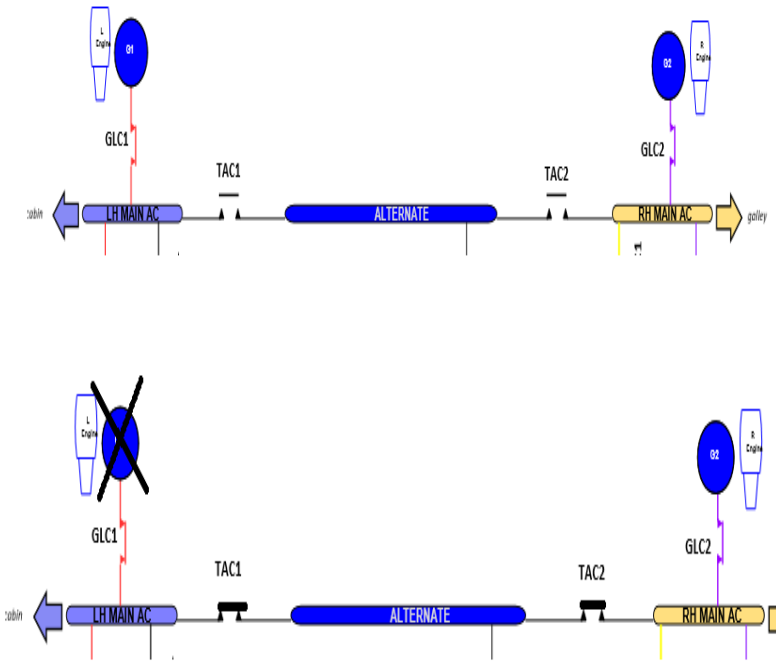
Safety Pattern example

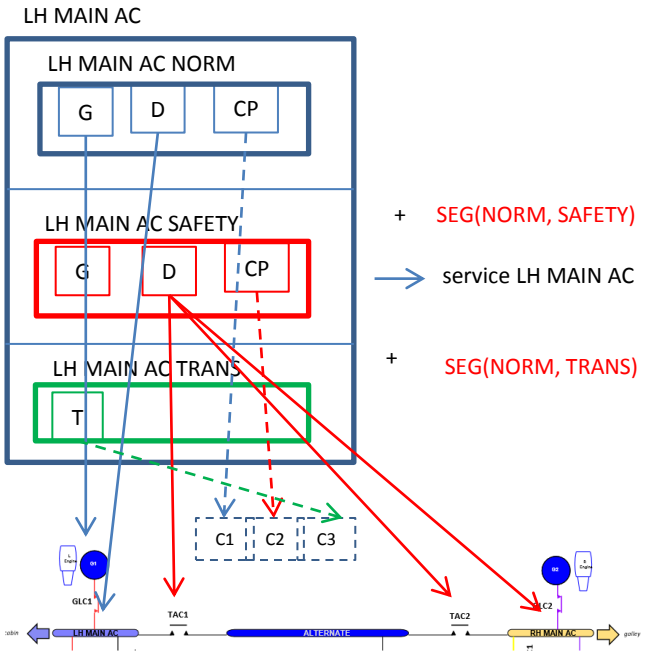


G: generation function CP: control program
D: distribution function TP: transition program



Safety Pattern usage





The DEPS language
A DSML for system synthesis

- **Declarative** DSM Language for problem specification and MBSS studies (EBNF)
- **Object-oriented** Knowledge Representation (*Models* are classes, *elements* are instances)
  - class-instance model
  - inheritance, composition, association, polymorphism
  - some attributes can be sub-defined (*variables*)
- **Formal properties** encapsulated inside or between Models
  - equations, inequalities between algebraic expressions (IEEE 754)
  - data catalogs
- **Ontology for engineers**
  - quantities, dimensions, units
- **Applications :** IMA architecture, battery system sizing, …
- DEPS is supported by the ***DEPS Link*** non profit organization

www.depslink.com

The DEPS Studio IDE

An Integrated Modelling and Solving Environment

A SYNTHESIS TOOL CHAIN: a

an integrated problem solving tool chain to address design problems:

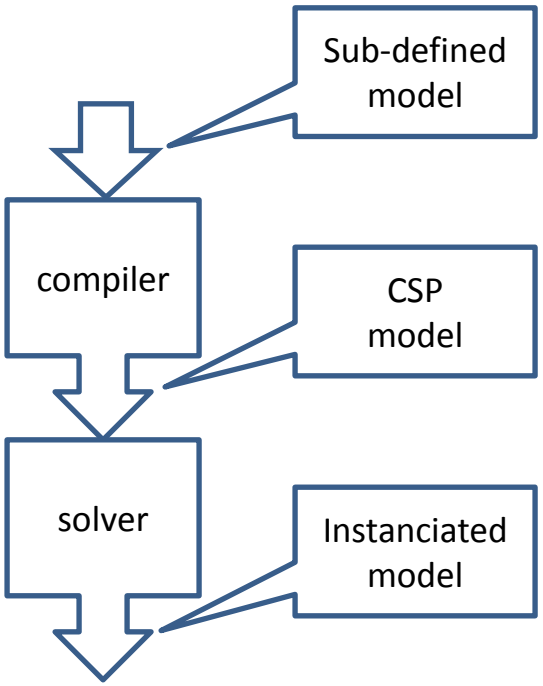
- sizing, deployment, configuration, architecture synthesis

DEPS COMPILER

- Ahead-of-time with static type checking
- generation of sub-defined model instances with constraints

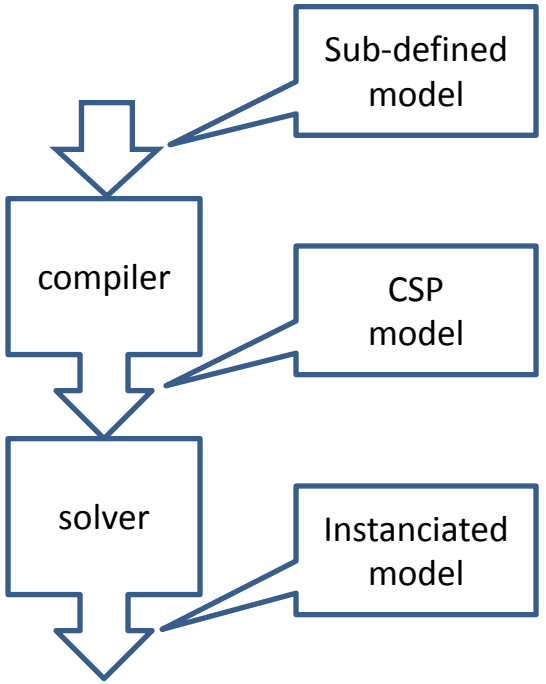
DEPS SOLVER

- constraint programming paradigm
- Purpose-built
- Mixed (integer/real) solving capabilities



Modelling and solving process with DEPS Studio

- **Modelling the problem with DEPS language**
  - PDS functions, channels, programs, contactors, …
  - Microcontrollers
  - Safety requirements
- **Compiling the problem**
- **Solving the problem**
  - Generation of one or several solutions (zero is possible too)



Modelling subdefined components

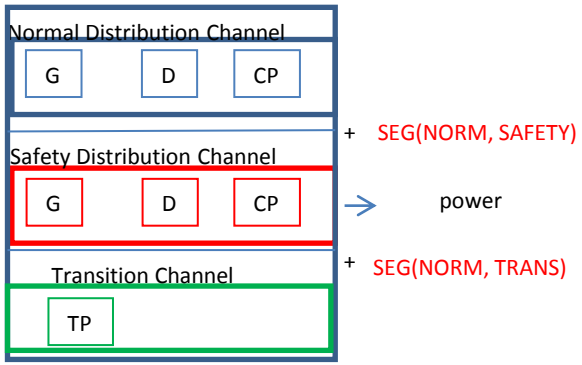
QuantityKind Integer
Type : integer ;
Min : -maxint;
Max : +maxint;
Dim : U ;
End

Quantity CpuIndex
Kind : Integer ;
Min : 1 ;
Max : 4 ;
Unit : u ;
End

Model Contactor ()
Constants
Variables
ProcIndex : CpuIndex;
Elements
Properties
End

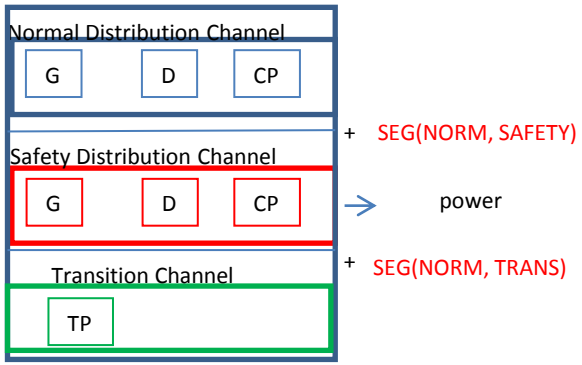
Model Processing (contSet)
Constants
Variables
ProcIndex : CpuIndex;
Elements
contSet: ContactorSet;
Collections
Properties
End

Modelling Pattern



G: generation function CP: control program
D: distribution function TP: transition program

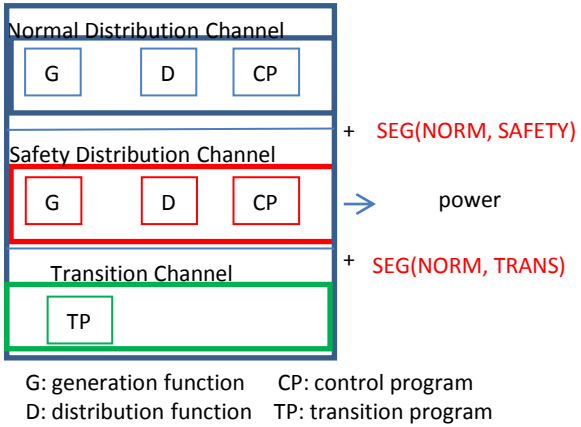
Modelling Pattern



G: generation function CP: control program
D: distribution function TP: transition program

Model S1 () extends ThreeChannelSystem[ContactorSet]
Constants
Variables
Elements
ch1: ChS1Norm(ContSet);
ch2: ChS1Safety(ContSet);
ch3: TRANSChannel();
seg1 : SEG(ch1, ch2);
seg2 : SEG(ch3, ch1);
Properties
End

Modelling Pattern

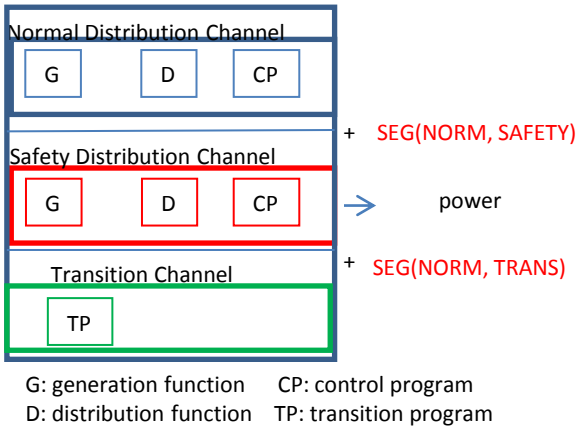


Model S1 () extends ThreeChannelSystem[ContactorSet]
Constants
Variables
Elements
ch1: ChS1Norm(ContSet);
ch2: ChS1Safety(ContSet);
ch3: TRANSChannel();
seg1 : SEG(ch1, ch2);
seg2 : SEG(ch3, ch1);
Properties
End

Model ChS1Norm () **extends** GDChannel[ContactorSet]
Constants
Variables
Elements
GenF: S1NormGenFunction();
DistF: S1NormDistFunction (ContSet);
ProcF: S1NormProcFunction(ContSet);
Properties
End

Model S1NormProcFunction()
extendsOneContactorProcFunction[ContactorSet]
Constants
Variables
Elements
Properties
L1.ProcIndex = ContSet.GLC1.ProcIndex;
End

Modelling Pattern



Model SEG(ProcF1, ProcF2) **extends** SEG[ProcFunction[ContactorSet], ProcFunction[ContactorSet]]
Constants
Variables
Elements
ProcF1 : OneContactorProcFunction[ContactorSet];
ProcF2 : ThreeContactorProcFunction[ContactorSet];
Properties
ProcF1.L1.ProcIndex <> ProcF2.L1.ProcIndex;
ProcF1.L1.ProcIndex <> ProcF2.L2.ProcIndex;
ProcF1.L1.ProcIndex <> ProcF2.L3.ProcIndex;
End

Model S1 () extends ThreeChannelSystem[ContactorSet]
Constants
Variables
Elements
ch1: ChS1Norm(ContSet);
ch2: ChS1Safety(ContSet);
ch3: TRANSChannel();
seg1 : SEG(ch1, ch2);
seg2 : SEG(ch3, ch1);
Properties
End

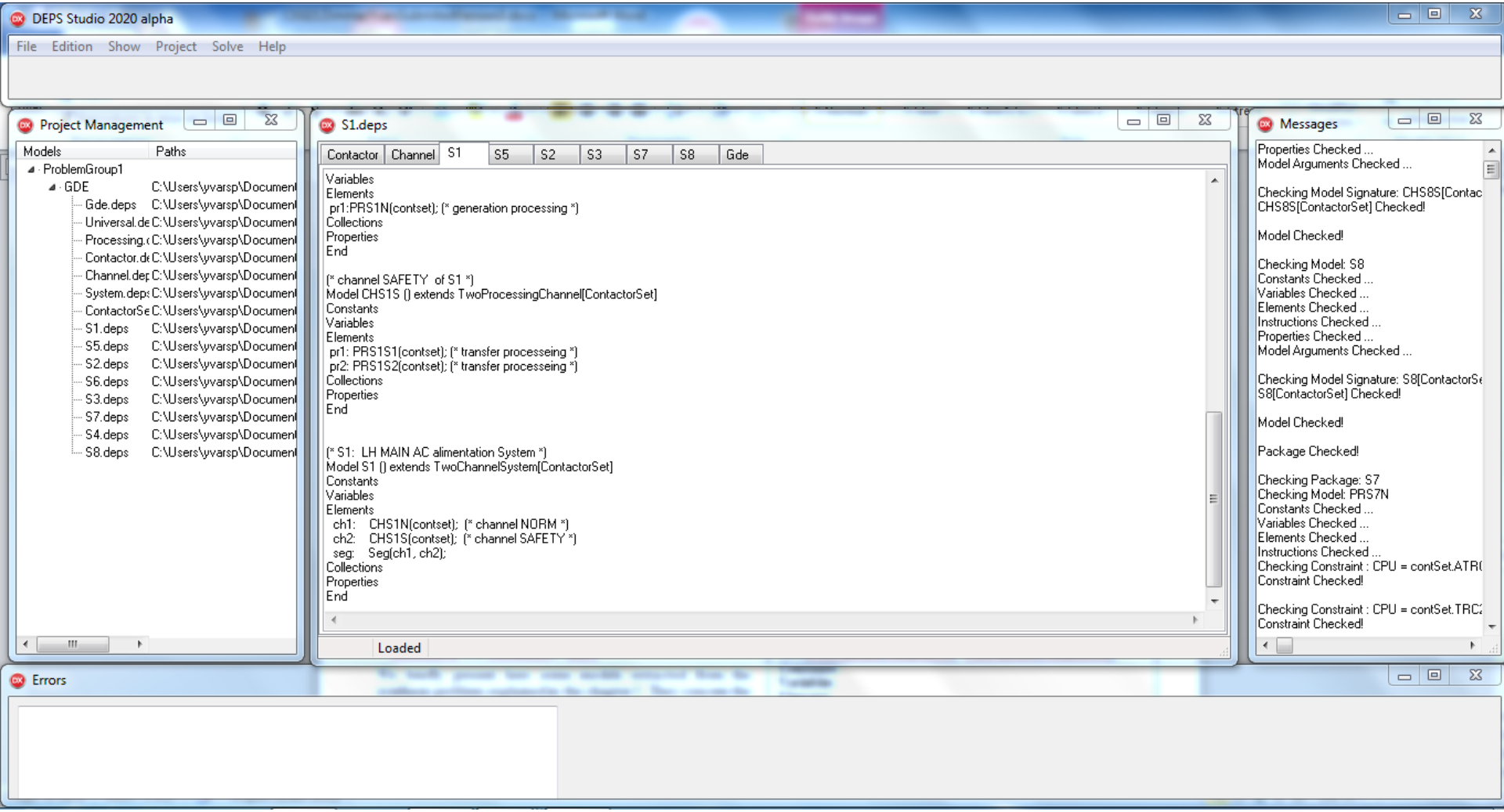
Model ChS1Norm () **extends** GDChannel[ContactorSet]
Constants
Variables
Elements
GenF: S1NormGenFunction();
DistF: S1NormDistFunction (ContSet);
ProcF: S1NormProcFunction(ContSet);
Properties
End

Model S1NormProcFunction()
extendsOneContactorProcFunction[ContactorSet]
Constants
Variables
Elements
Properties
L1.ProcIndex = ContSet.GLC1.ProcIndex;
End

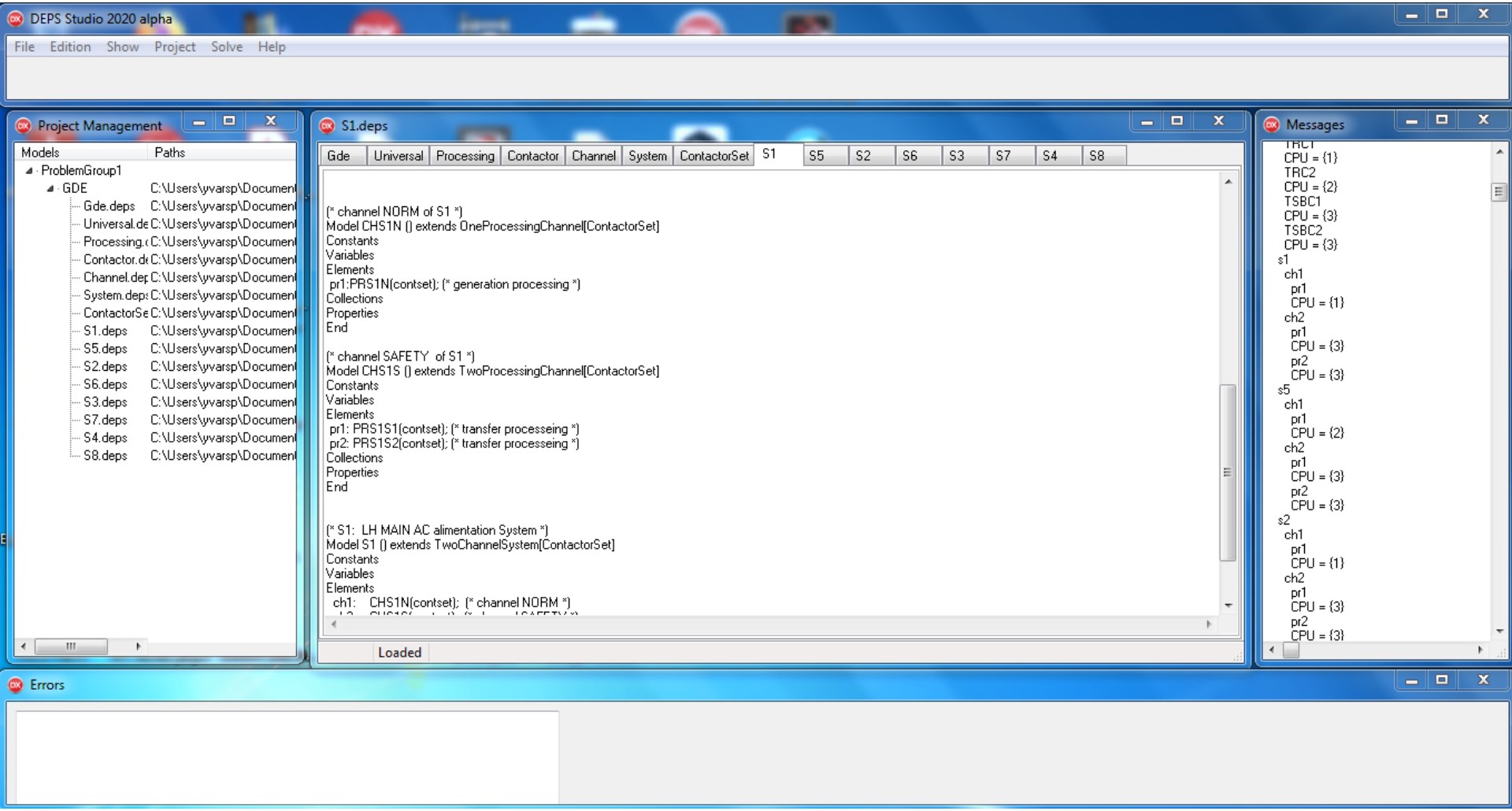
Modeling the problem

Problem FailSafeProblem
Constants
Variables
Elements
TheContactors: ContactorSet();
F1: S1(TheContactors); F2: S2(TheContactors);
F3: S3(TheContactors); F4: S4(TheContactors);
F5: S5(TheContactors); F6: S6(TheContactors);
F7: S7(TheContactors); F8: S8(TheContactors);
seg1: SEG(F1.ch1, F3.ch1);
seg2: SEG(F2.ch1, F4.ch1);
seg3: SEG(F5.ch1, F7.ch1);
seg4: SEG(F6.ch1, F8.ch1);
Properties
End

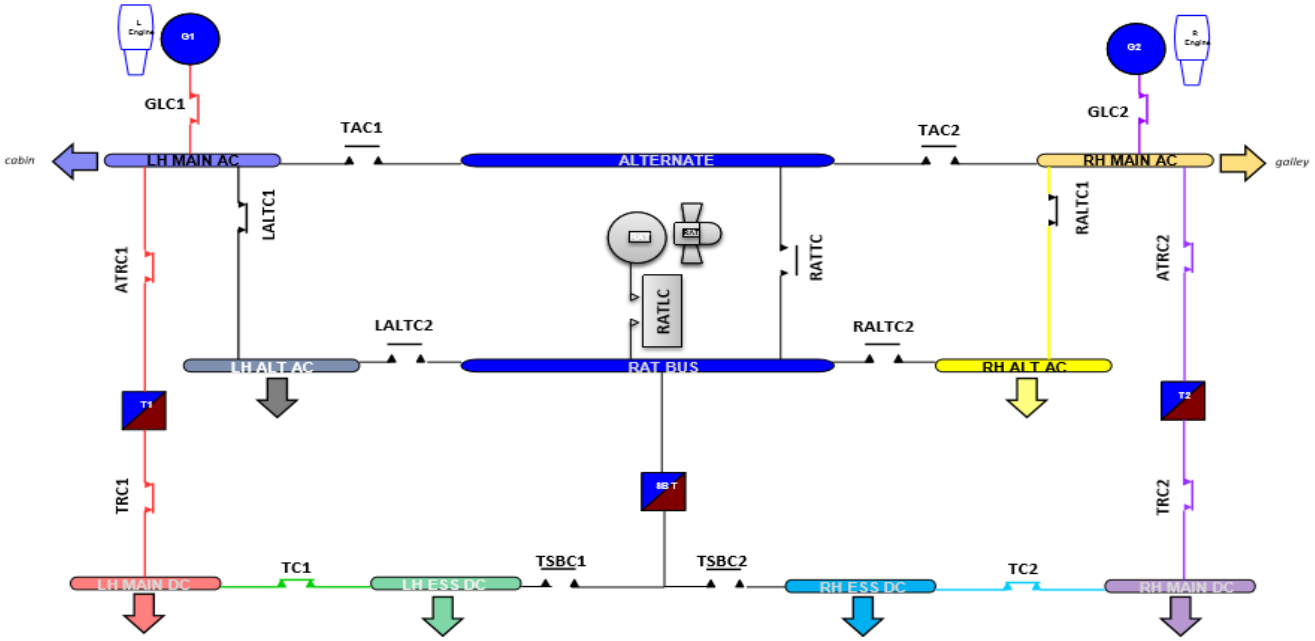
DEPS Studio
Compiling the models



DEPS Studio
Solving and Results



Numerical Results



Contactors	Calculator Index
GLC1	1
ATRC1	1
TRC1	1
LALTC1	4
GLC2	2
ATRC2	2
TRC2	2
RALTC1	3
RALTC	1
RATTC	1
TAC1	3
TAC2	4
LALTC2	3
RALTC2	4
TC1	2
TSBC1	3
TSBC2	3
TC2	1

Bus bar system	NORM calculator Index	SAFETY calculator Index	TRANS calculator Index
LHMAIN AC	1	3, 4, 2	3
LHMAIN DC	1, 1	4, 3, 3, 2	2
RHMAIN AC	2	4,3,1	3
RHMAIN DC	2, 2	3, 4, 3, 1	1
LHALTAC	4	3, 1, 3	1
LHESSDC	2	4, 3, 3	1
RHALTAC	3	4, 1, 4	1
RHESSDC	1	3, 4, 3	2

Ongoing and Future Work

- **A synthesis approach**
- **Abstraction, genericity, reusability**
- **Integrated modeling and solving process with DEPS Studio**
- **Solutions correct by construction**

Ongoing and Future Work

- **A synthesis approach**
- **Abstraction, genericity, reusability**
- **Integrated modeling and solving process with DEPS Studio**
- **Solutions correct by construction**

- **Taking into account additional safety requirements :**
  - double faults,
  - ultimate rescue mode.
- **Taking into account additional components :**
  - fault sensors.

Ongoing and Future Work

- **A synthesis approach**
- **Abstraction, genericity, reusability**
- **Integrated modeling and solving process with DEPS Studio**
- **Solutions correct by construction**

- **Taking into account additional safety requirements :**
  - double faults,
  - ultimate rescue mode.
- **Taking into account additional components :**
  - fault sensors.

➔ The evolution of the models will follow those of the DEPS language in particular on the possibilities of manipulation of collections of objects in the future versions of DEPS and DEPS studio

THANKS FOR YOUR ATTENTION

QUESTIONS ?