AEROSYSTEMS

VERS UNE CONCEPTION INTÉGRALE À BASE DE MODÈLES ?

—

Sébastien RATISSEAU – System Engineer

06/18/2019

SAFRAN

Table of contents

SAFRAN

1

INTRODUCTION OF SAFRAN ELECTRICAL POWER (SEP)

1. SEP ACTIVITIES

SAFRAN

SEP activities

EPGDS products

Primary Electrical Power Distribution

Secondary Electrical Power Distribution

Main & Auxiliary Power Generation

Power Conversion

Battery

SAFRAN

2

PROJECT A: SYSTEM MODEL EMBEDDED IN A FLIGHT SIMULATOR FOR PILOT TRAINING

1. WHAT WAS THE CUSTOMER EXPECTATION?
2. HOW TO REPRESENT THE SYSTEM?
3. HOW TO ENSURE THE MODEL REPRESENTATIVITY?
4. WHAT HAVE WE LEARNED FROM THIS PROJECT?

SAFRAN

What was the customer expectation?

Design a flight simulator

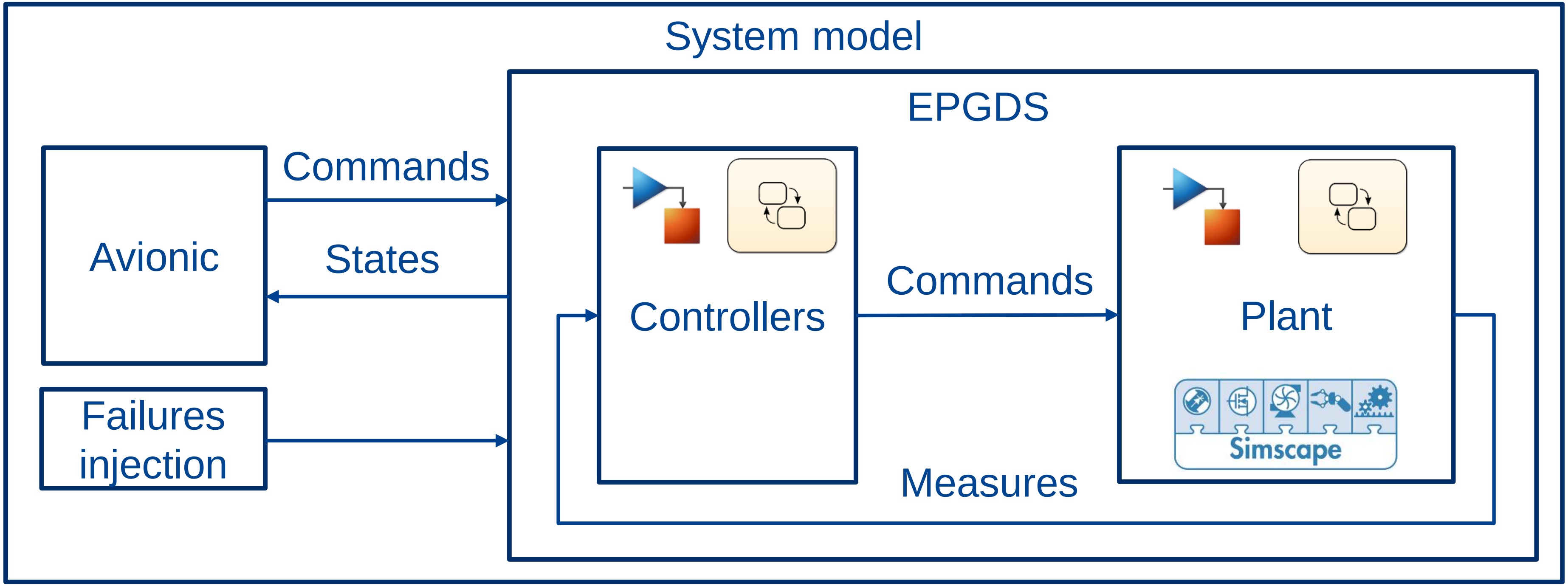
- **Develop a real-time solution which simulate the whole EPGDS**

System level requirements	> 1,500
Electrical loads	> 500
ECU	> 50
Physical interfaces	> 5,000
Communication	> 120,000 signals sent through 30 communication buses

- **MATLAB/Simulink R2012b used to create the model**
  - Model contains core partner's elements
  - Use of encrypted models to protect intellectual properties

- **National Instrument solutions for the real-time bench**
  - LabVIEW for standalone application deployment
  - VeriStand for the bench

How to represent the system?



How to represent the system?

Primary Electrical Power Distribution

Secondary Electrical Power Distribution

Main & Auxiliary Power Generation

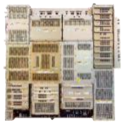
Power Conversion

Battery

How to represent the system?

PPDS

Primary Electrical Power Distribution



Main & Auxiliary Power Generation

Power Conversion

Battery

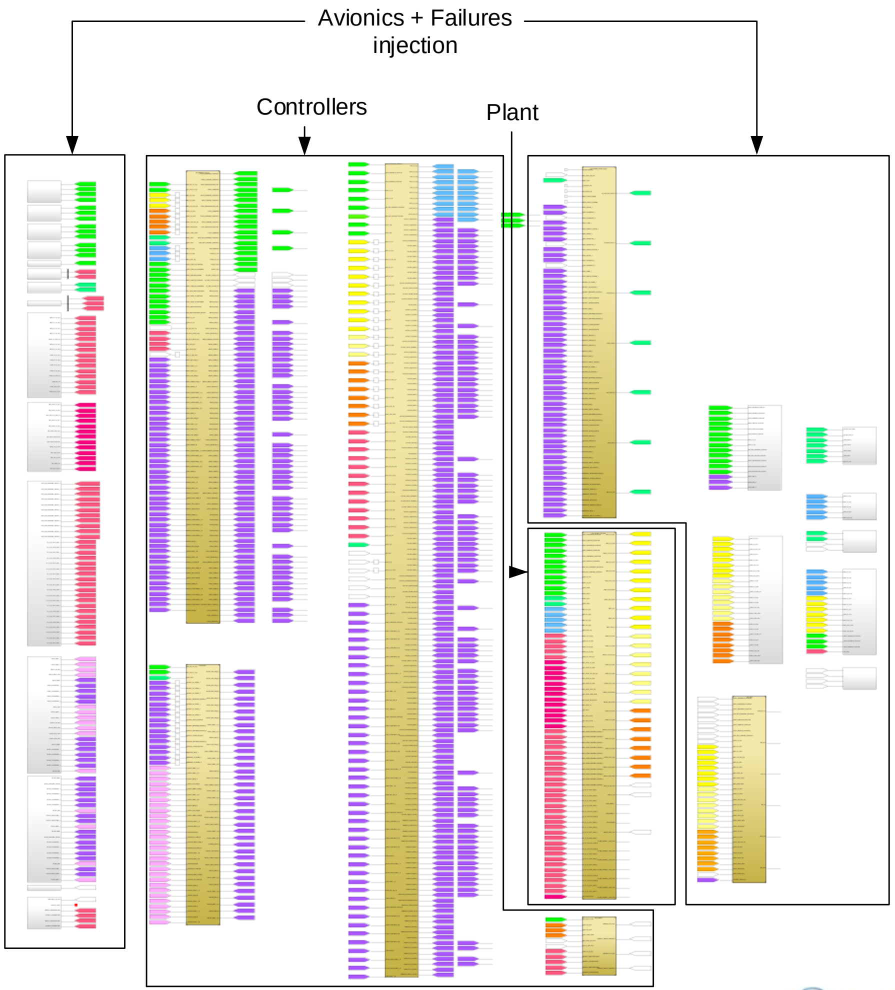
SPDS

Secondary Electrical Power Distribution

How to represent the system?
PPDS model

Model complexity
- 134k blocks
- 15 referenced models
- 300 actuators inside the plant

Functional interfaces
- 600 inputs
  - 5% from avionic
  - 95% of failures injection
- 1000 outputs
  - 25% system state communication
  - 75% of electrical power availability information



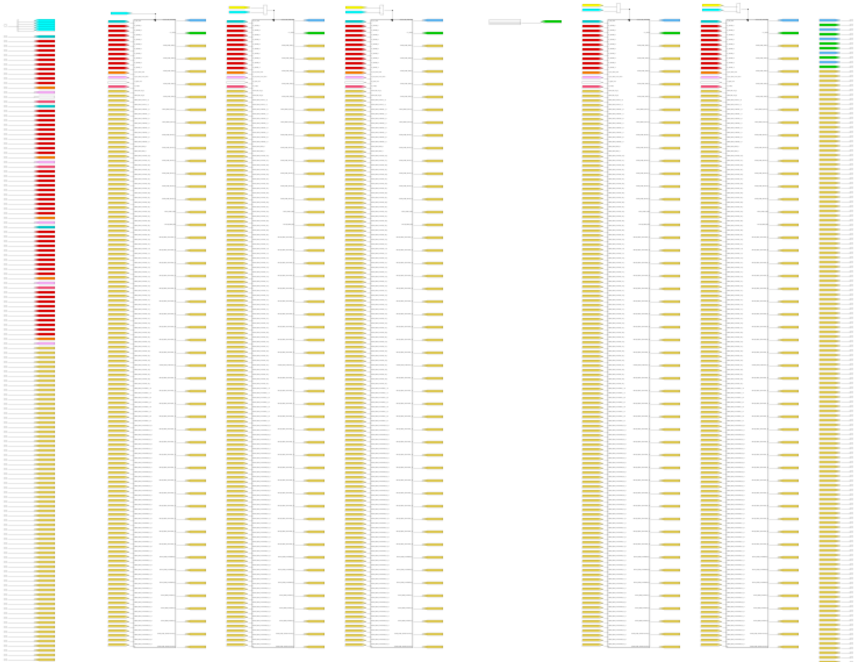
How to represent the system?

SPDS model

- **Model complexity**
  - 750k blocks

- **Functional interfaces**
  - 4,800 inputs
    - 5% of electrical power availability information from PPDS
    - 95% of commands through communication buses
  - 1,800 outputs
    - 5% of electrical power availability information
    - 95% system state communication

- **Model is auto-generated**



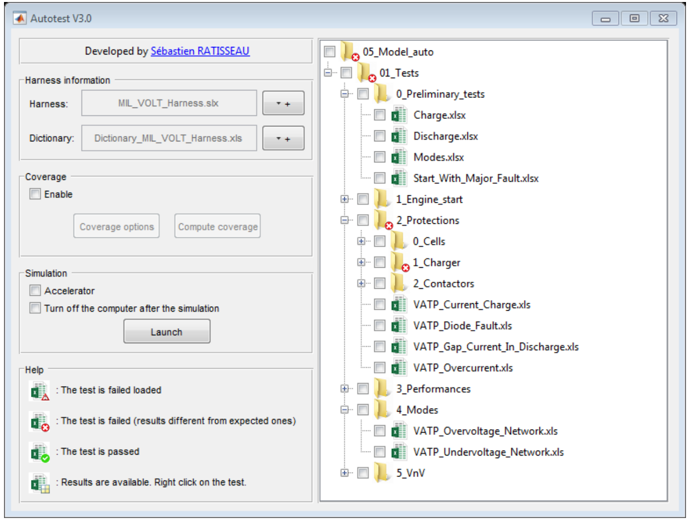
How to ensure the model representativity?

Simulation

- **Traceability with the specification thanks to Simulink V&V toolbox**

- **MIL automated tests from a custom MATLAB application**
  - Tests are defined in Excel with a custom format
  - Automated verification of expected results with simulation’s outputs

- **Non regression tool**
  - Use of Simulink Project
  - Run automated tests after each integration phase



How to ensure the model representativity?

Standalone application and real-time bench deployment

- **System real-time tests with a Windows application (Simulink Coder)**
  - Help for system analysis and debug
  - Good communication tool used inside the company and with the customer

- **Same HMI and compiled models are used on the real-time bench**

What have we learned from this project?

- **Modular architecture**
  - Team work is easier (referenced models + Simulink Project + source control)
  - Possibility to use the model in MIL and in real-time without additional effort
  - SimScape real-time usage feedback

- **Development time reduction process**
  - Model
    - 1.5 FTE during 1 year for the POC
    - 2 FTE during 2 years for the updates
  - Real-time means
    - 2 FTE during 2 years for the development
  - 2.5 months required for the first bench integration → 12 hours at the end of the project

- **Real-time bench has been used 24/7 by the customer for 2 years**
  - Global bench behavior is consistent with real system

SAFRAN

3

PROJECT B: BMS SOFTWARE DESIGN

1. WHY GENERATING C CODE?
2. WHAT’S THE IMPACT ON THE MODEL ARCHITECTURE?
3. HOW TO VALIDATE THE CODE GENERATION TOOLCHAIN?
4. WHAT HAVE WE LEARNED FROM THIS PROJECT?

Why generating C code?

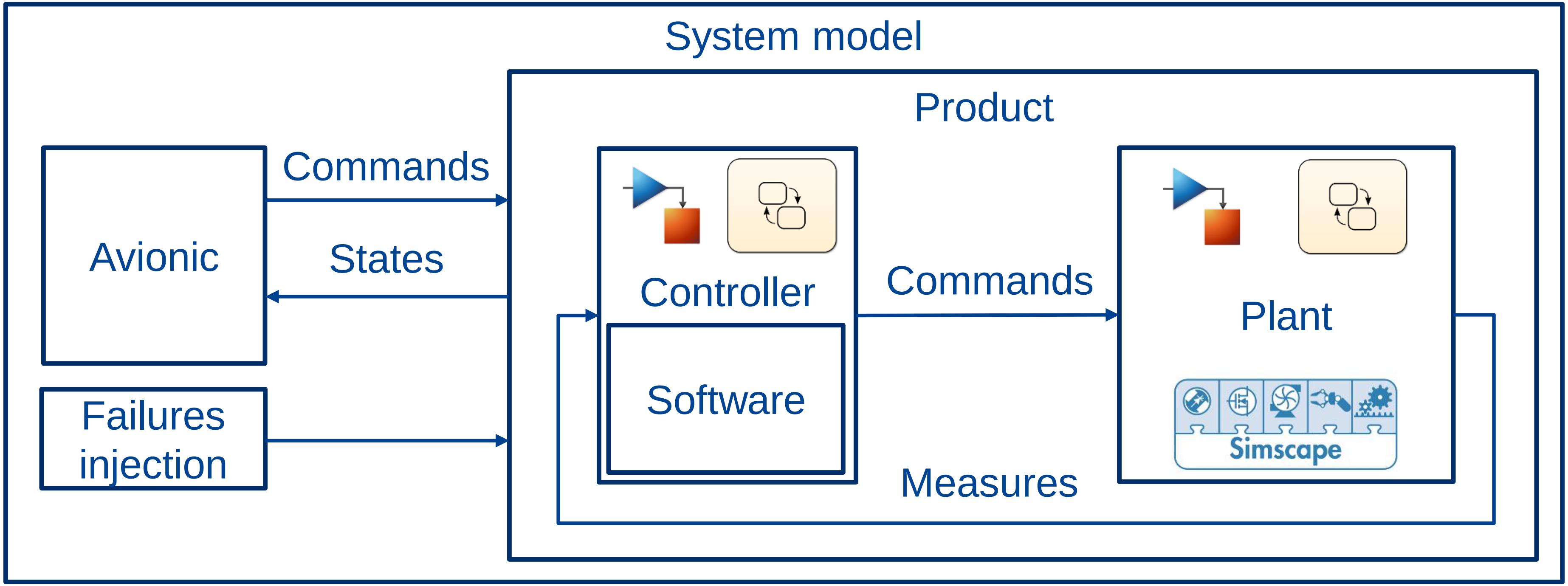
- **Develop a high voltage battery**
  - Safety Of Flight expected
  - R&T project

- **Reduce time development**
  - > Use the system model to define the embedded logics
    - Modular architecture reuse from project A
  - > Whole applicative software auto-generated from Simulink (Embedded Coder)
    - DO-178C inspired process

- **Upgrade of MATLAB version to R2017b**

SAFRAN

What's the impact on the model architecture?



What’s the impact on the model architecture?

Software model

Software

- **Embeds all applicative software functions**
  - Input signals conditioning (analogic measures, communication frames …)
  - Internal logics (battery management, states estimators, …)
  - Output signals conditioning (communication frames …)

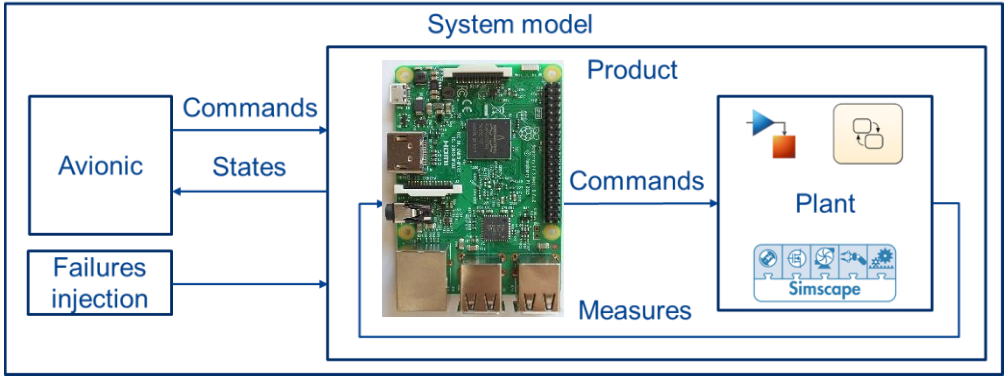
- **All applicative software functions can be validated in simulation with the system model**

- **Model is not linked to the controller hardware**
  - Auto-generated C code could be used with another controllers if they have enough hardware resources

How to validate the code generation toolchain?

Does the C code behave as the model?

- **MIL**
  - Automatic tests (nominal + dysfunctional)
    - Defined at system level
  - Model coverage measure
    - Target: 100% of Condition and Decision
- **HIL**
  - 2 modes
    - Manual
      - For integration and specific debug tests
    - Automatic
      - Run the same tests as the MIL
- **Automatic comparison of MIL/HIL tests**
  - Unexpected controller behavior risk is highly minimized



SAFRAN

What have we learned from this project?

- **New architecture used both for system simulation and C code generation**
  - Software model is validated in the system context before being auto generated
- **Risk of an unexpected behavior of the generated code is highly reduced**
  - Thanks to automatic tests in MIL + HIL
- **New features could be tested quickly thanks to rapid prototyping method**
  - 1 minute required to generate and deploy the software on the custom target

4

ROAD MAP

Road map

- **Develop a unique design standard for all projects**
  - Custom checks inside Model Advisor

- **Use the model for architecture design (System Composer or other tool)**
  - Make team communication easier
  - Problems can be anticipated since the beginning of the project
    - Time saving for some activities (interface definition etc)

POWERED
BY TRUST