30th Annual INCOSE
international symposium

Virtual Event
July 20 - 22, 2020

A system design methodology

Top-down functional composition

Johan Bredin, SAAB Aeronautics
johan.bredin@saabgroup.com

This presentation includes

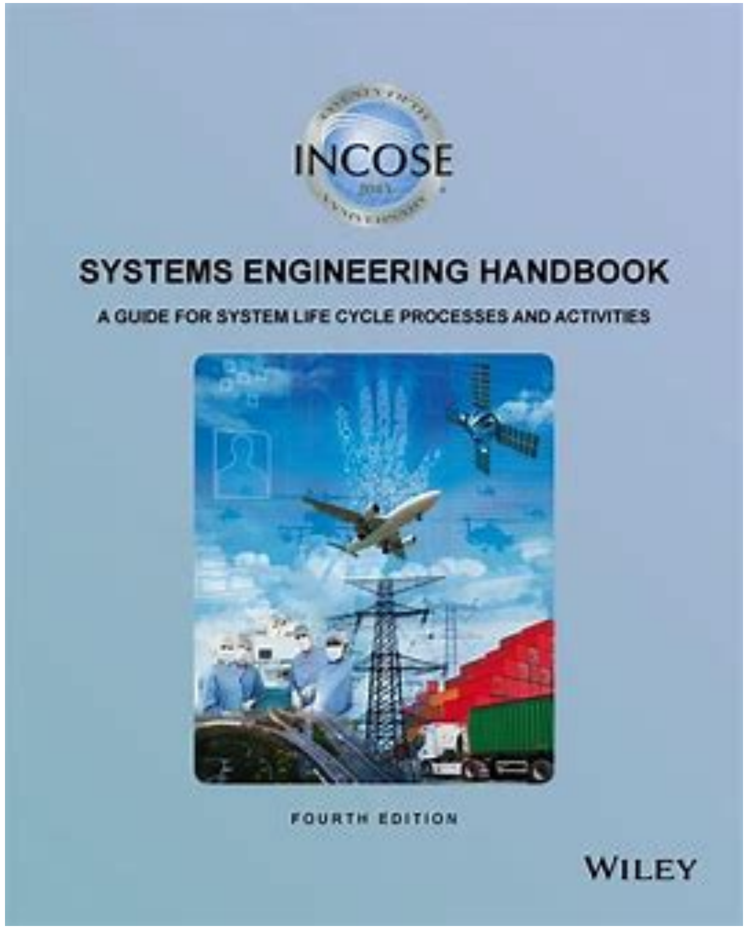
- A critical view on the top-down decomposition methodology

- A new top-down composition design methodology, including:
  - Executable design- and integration models
  - Very early and continuous design integration
  - Very early and continuous verification of

My background

- 15 years as a software developer
  - Mainly code generation from UML moc
  - Manual coding
  - Most of the time: safety critical software f avionics applications
- 6 years as a MBSE methodology developer / support person
  - Systems engineering aware

The handbook


INCOSE
SYSTEMS ENGINEERING HANDBOOK
A GUIDE FOR SYSTEM LIFE CYCLE PROCESSES AND ACTIVITIES
FOURTH EDITION
WILEY

Problem understanding the requirements, architecture and design processes

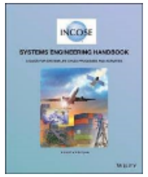
Part One : Decomposition

- **Where I argue that a system design approach based on functional decomposition can not produce good requirements.**
- This is a problem because requirements is the foundation on which systems are built.

- Warning: this may feel a little bit uncomfortable.

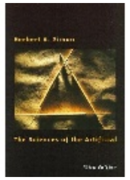
Legend

- INCOSE SE Handbook, 4th edition



- Engineering Systems (Buede & Miller, 2016)

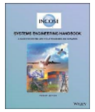
- The sciences of the artificial (Simon,1996)



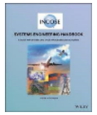
- Yours truly

Two premises that need to be true

- P1 : *All requirements sets is complete*



- P2 : *All requirements sets is design-agnostic*



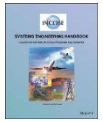
  - P2 means that one shall not make design choices while writing requirements

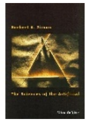
Questioning Premise P1

- P1 : *All requirements sets is complete*

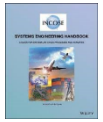
Recursively applied requirements process

- P3 : Complex/complicated systems require system internal hierarchies



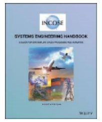


- P4 : System hierarchies require a recursively applied requirements process

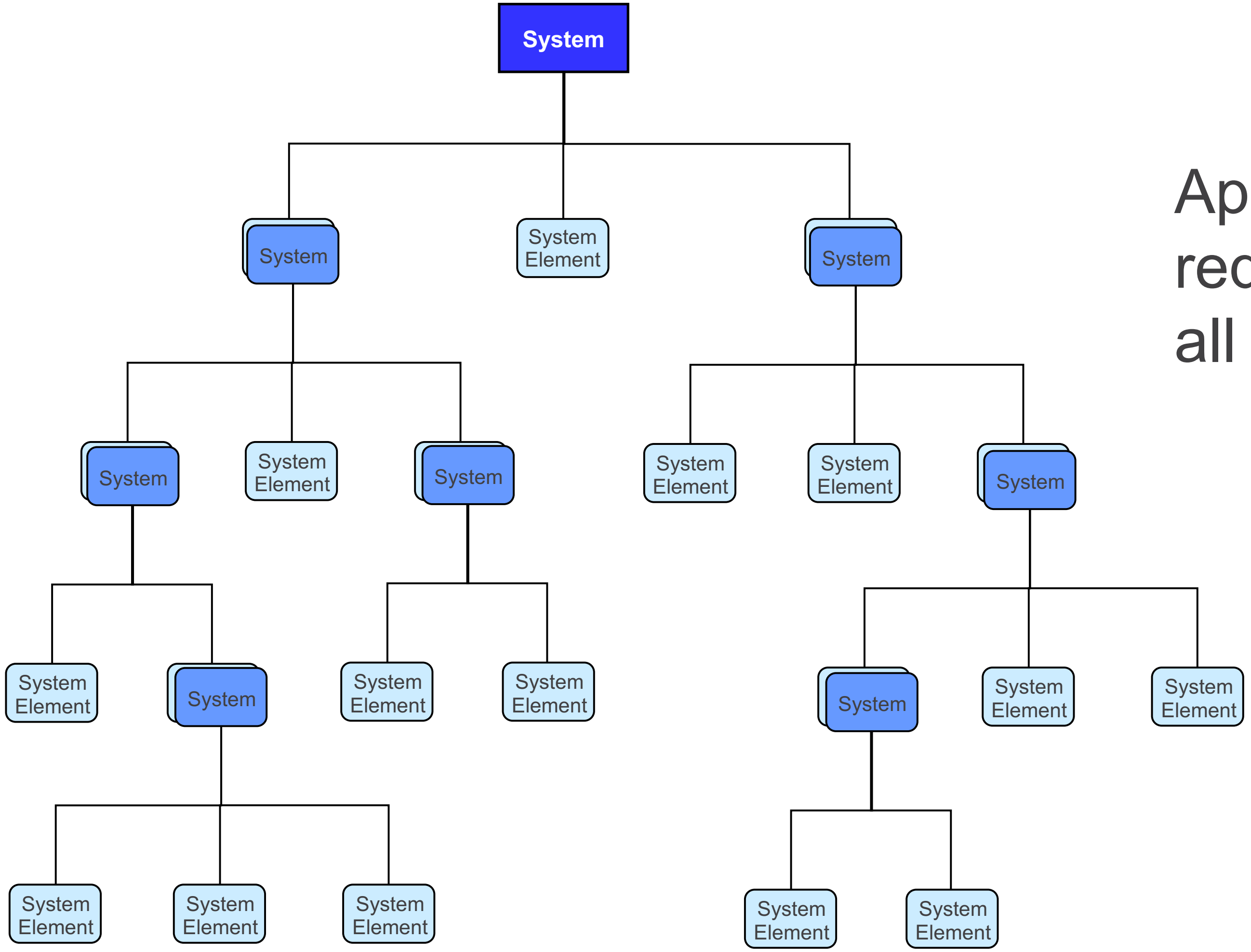


- P5 : The system under consideration is complex/complicated

{P3, P4, P5} => C1 :The system requirements process shall be recursively applied.



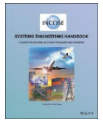
System hierarchy

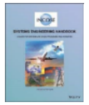


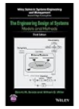
Apply the same system requirements process for all systems / system elements

Transformation of inputs into outputs

- C1: The system requirements process shall be recursively applied.
- P6 : A system function implies transformation of inputs into outputs







Functional decomposition

- P8: Inputs and outputs from higher level system nodes must be conserved if the methodology ***functional decomposition*** is used.

”This decomposition process must conserve all of the inputs to and all of

Back to the future

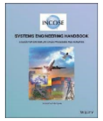
Wiley Series in Systems Engineering and Management
The Engineering Design of Systems
Models and Methods
Third Edition
Dennis M. Buede and William D. Miller
WILEY

- P8: Inputs and outputs from higher level system nodes must be conserved if the methodology ***functional decomposition*** is used.

- C2: A system requirements set, on all system levels, includes transformation of inputs into outputs

Back to the future – failure of P1

- P1 : *All requirements sets is complete*
- C3 : The transformation of the (to be) realized systems inputs to the (to be) realized systems outputs must be specified in the top-level requirements set.



Questioning Premise P2

- P2 : *All requirements sets is design-agnostic*

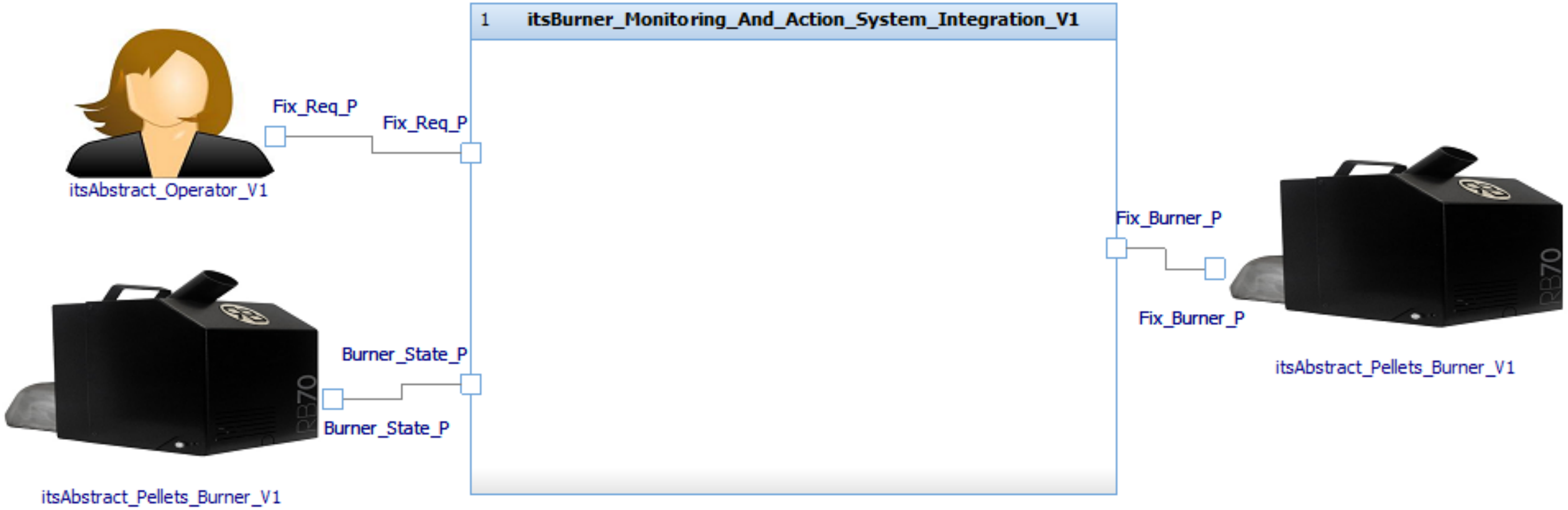
To down select is to design

- P9 : To down select between valid design solutions is part of the design process

- P10 : To choose inputs and outputs equals a down select

- {P9, P10} => C4 : An inputs and outputs choice is part of the design

To down select is to design


1
itsBurner_Monitoring_And_Action_System_Integration_V1
Fix_Req_P
Fix_Req_P
itsAbstract_Operator_V1
Burner_State_P
Burner_State_P
itsAbstract_Pellets_Burner_V1
Fix_Burner_P
Fix_Burner_P
itsAbstract_Pellets_Burner_V1

Assume:

- 4 different ways of communicating a fix request,
- 4 ways of getting to know the burner state and
- 4 ways of fixing the pellets burner;

=> 64 valid input/output sets

We do not want to be forced to select a specific input/output set when writing top-level system requirements.

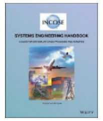
Requirements is not design-agnostic

- C2: A system requirements set, on all system levels, includes transformation of inputs into outputs

- C4 : An inputs and outputs choice is part of the design

- P8: Inputs and outputs from higher level systems must be conserved if the methodology ***functional decomposition***

Crash and burn – failure of P2



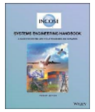
- P2 : *All requirements sets is design-agnostic*

- C5 : A requirements set is, in the general case, not design-agnostic when applying ***functional decomposition.***

- *C5 => - P2*
- *P2 can not be valid in a **functional decomposition*** context.

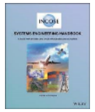
Two premises that can’t to be true

- P1 : *All requirements sets is complete*



P1 can be true in theory, but not in practice

- P2 : *All requirements sets is design-agnostic*



P2 can not be true

Be a good engineer and…

- Write a requirements set that is both complete and design-agnostic

- Is that possible?

Part Two: Composition

- Where I argue that a system design approach based on functional ***composition*** can produce good requirements.

- Relax, this is the feel-good part.

Change – Yes we can!

- P8: Inputs and outputs from higher level systems must be conserved if the methodology ***functional decomposition*** is used.

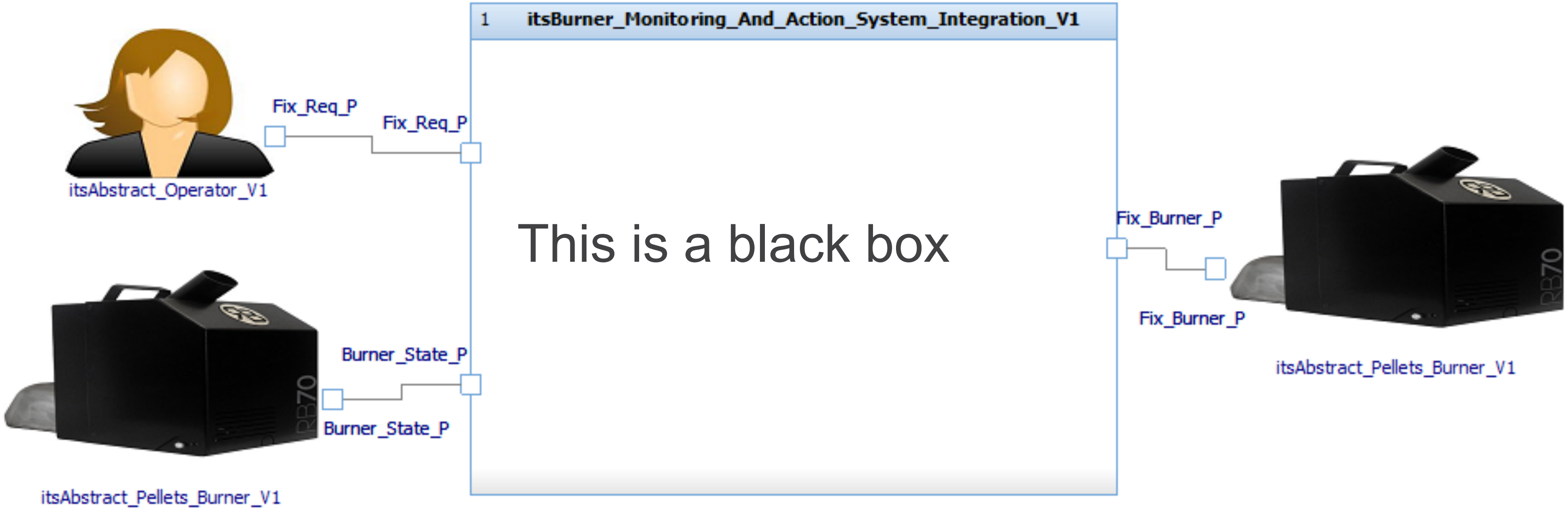
- It was the input and output conservation rule that got us into trouble.

Relaxations

- Relaxation: **Inputs and outputs from higher level system nodes do not need to be conserved.**

- Relaxation: **A black box description of a system and a white box description of the same system do not need to have identical inputs and outputs.**

Top-level system black box

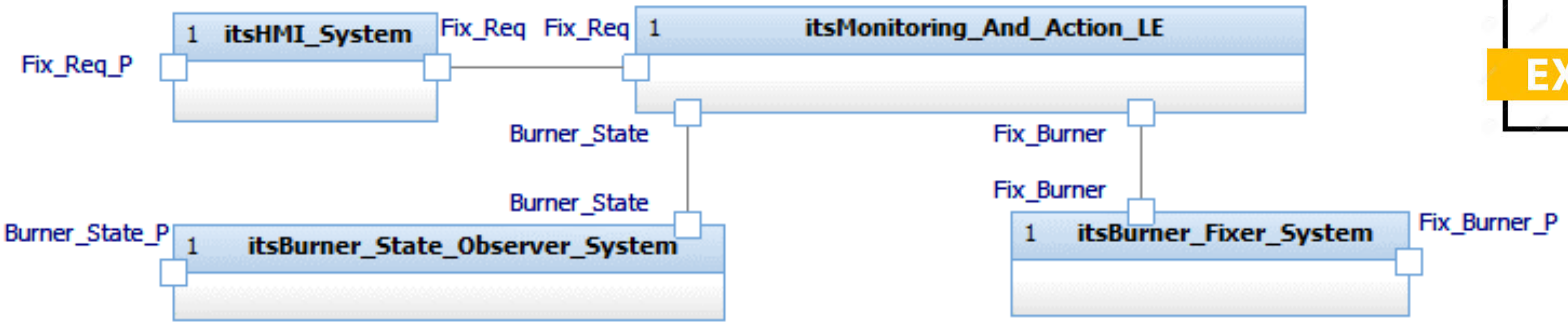

1
itsBurner_Monitoring_And_Action_System_Integration_V1
Fix_Req_P
Fix_Req_P
itsAbstract_Operator_V1
Burner_State_P
Burner_State_P
itsAbstract_Pellets_Burner_V1
Fix_Burner_P
Fix_Burner_P
itsAbstract_Pellets_Burner_V1
RB70
RB70

This is a black box

Requirements example

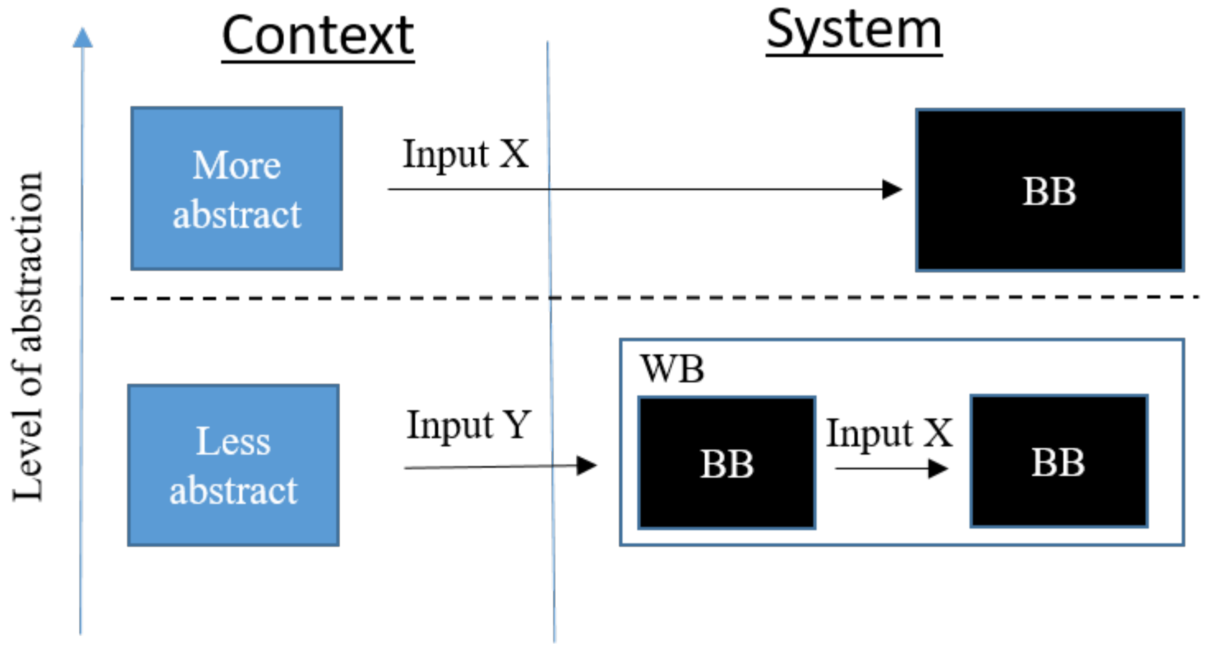
- **Pellets burner manual fix function**
  - – R-001 If <fix pellets burner request> event shall a <pellets burner fix> action be performed
  - – R-002 The <pellets burner manual fix function> shall be executed within a timeframe of 15 seconds

Decomposition – no change in the context abstraction



An integrated system with three subsystems and a logical element leaf node

Abstraction leap rule

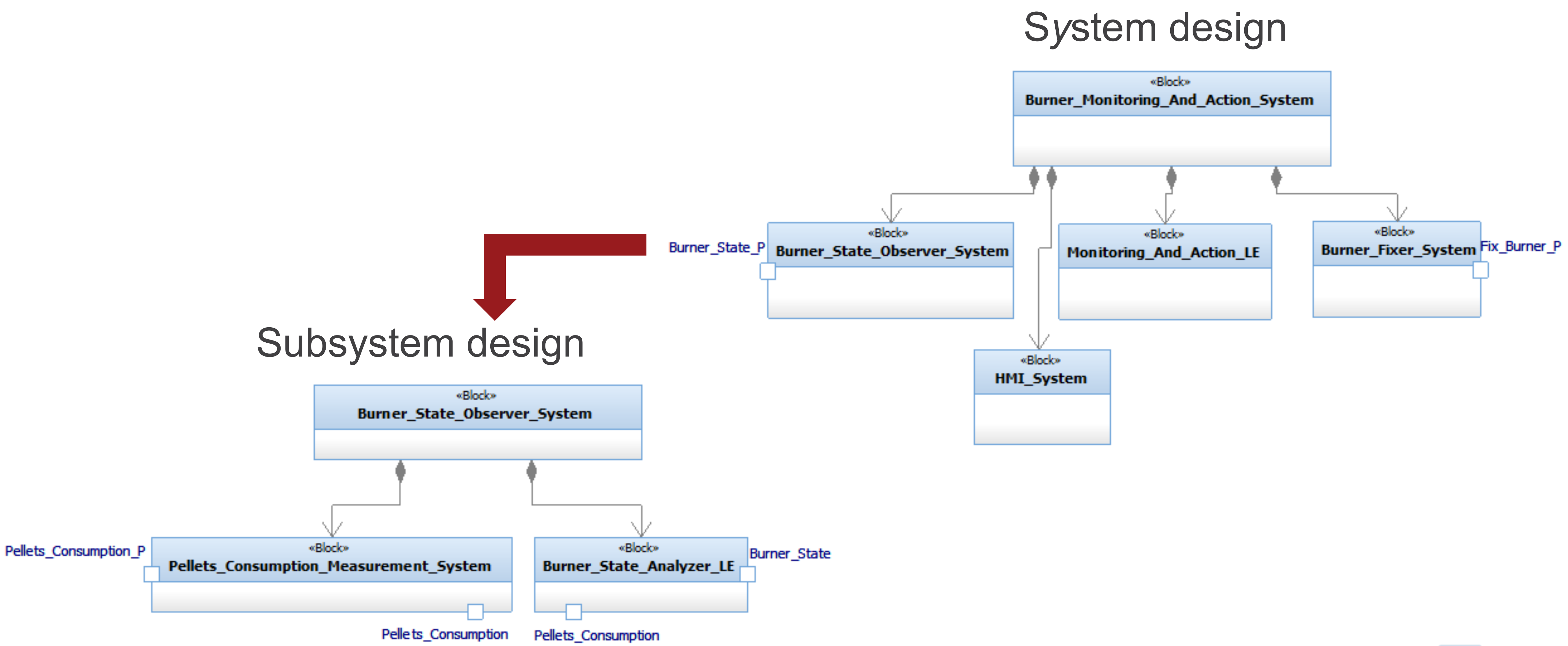


As an example:
Context entity = Pellets burner
Input X = Burner state
Input Y = Pellets consumption

Rules – from abstract to concrete

- If a set of inputs X on the black box description is exchanged for a set of inputs Y on the white box description there shall be a transformation of Y to X in the white box design so that X is used as an internal input.
- If a set of outputs X on the black box description is exchanged for a set of

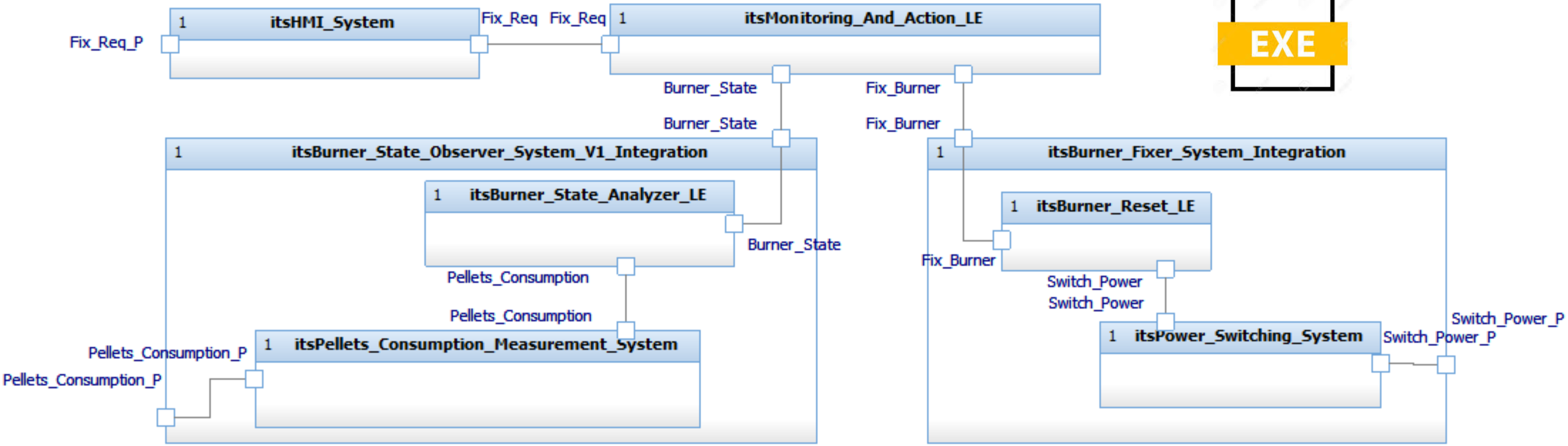
Composition – a new level of abstraction



System design

Subsystem design

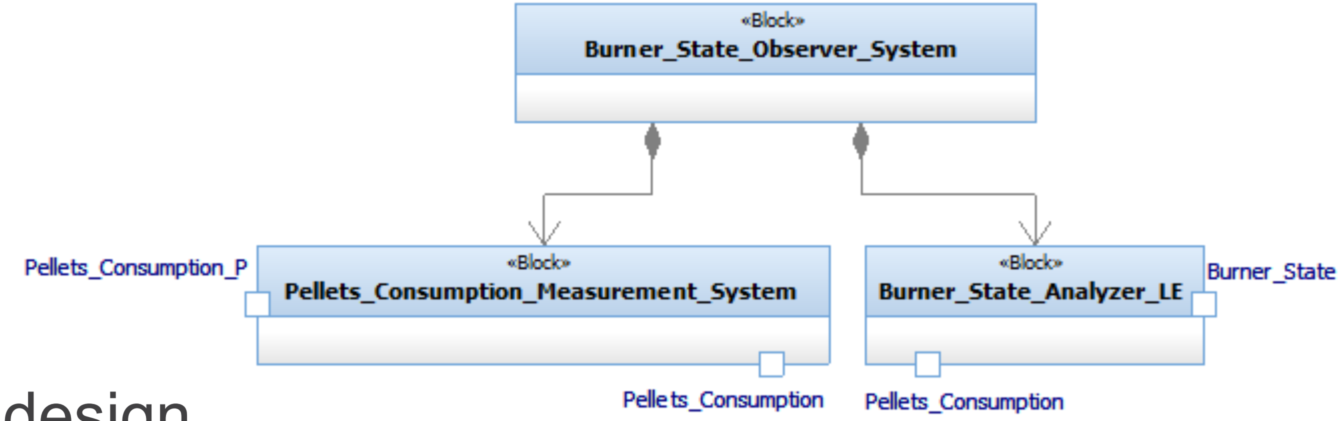
Composition – a new level of abstraction



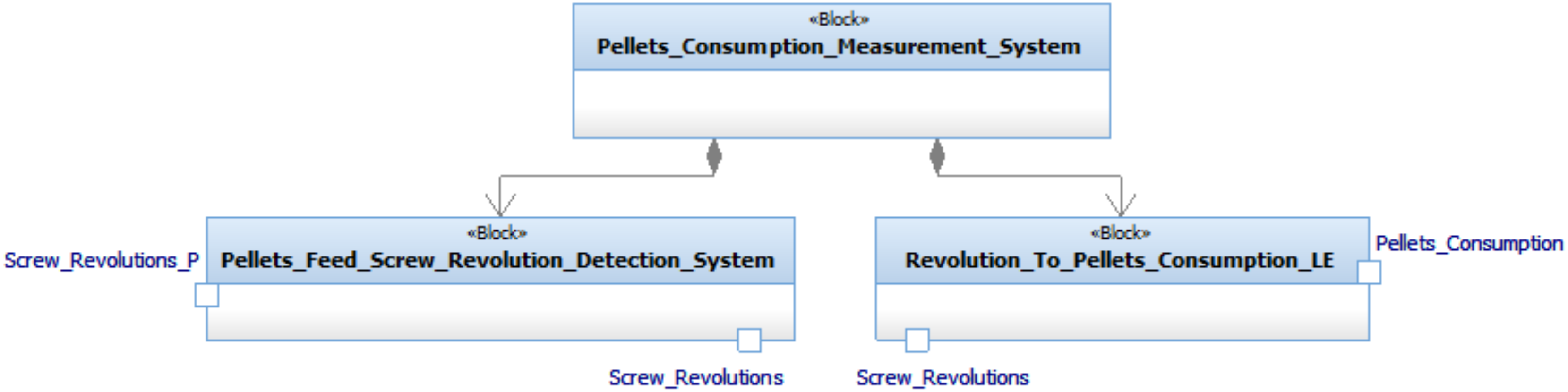
A new version of the integrated system

Moving towards a realizable system

System design

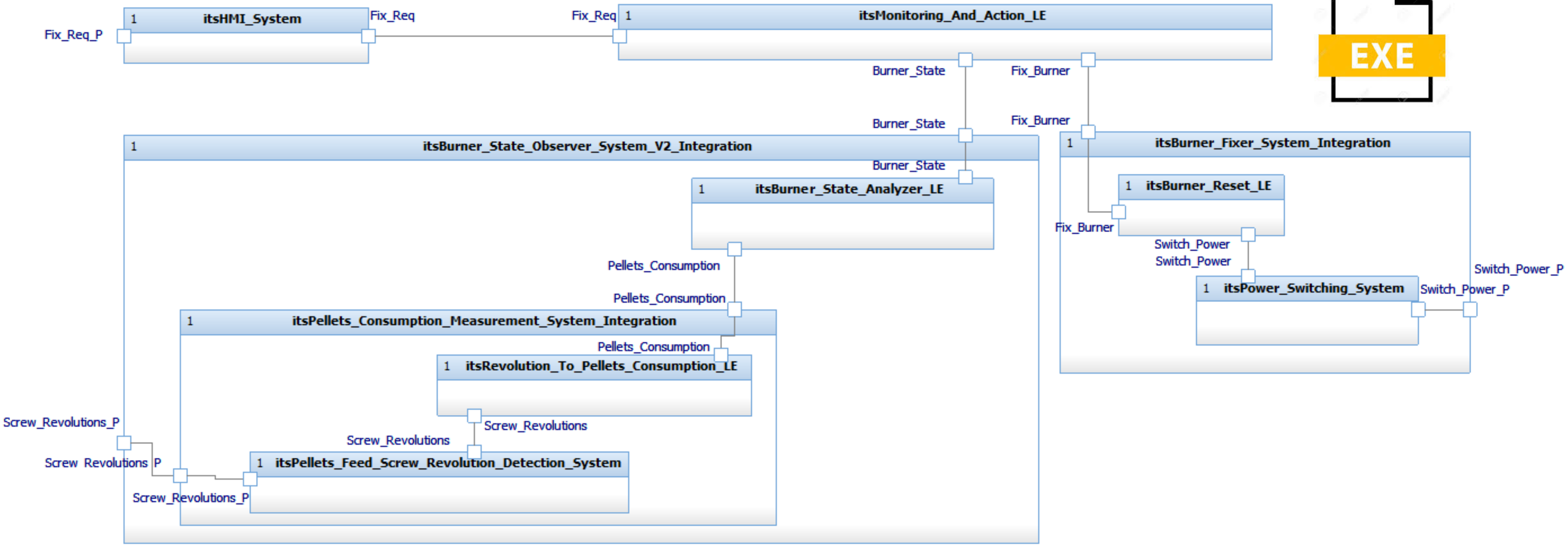


Subsystem design

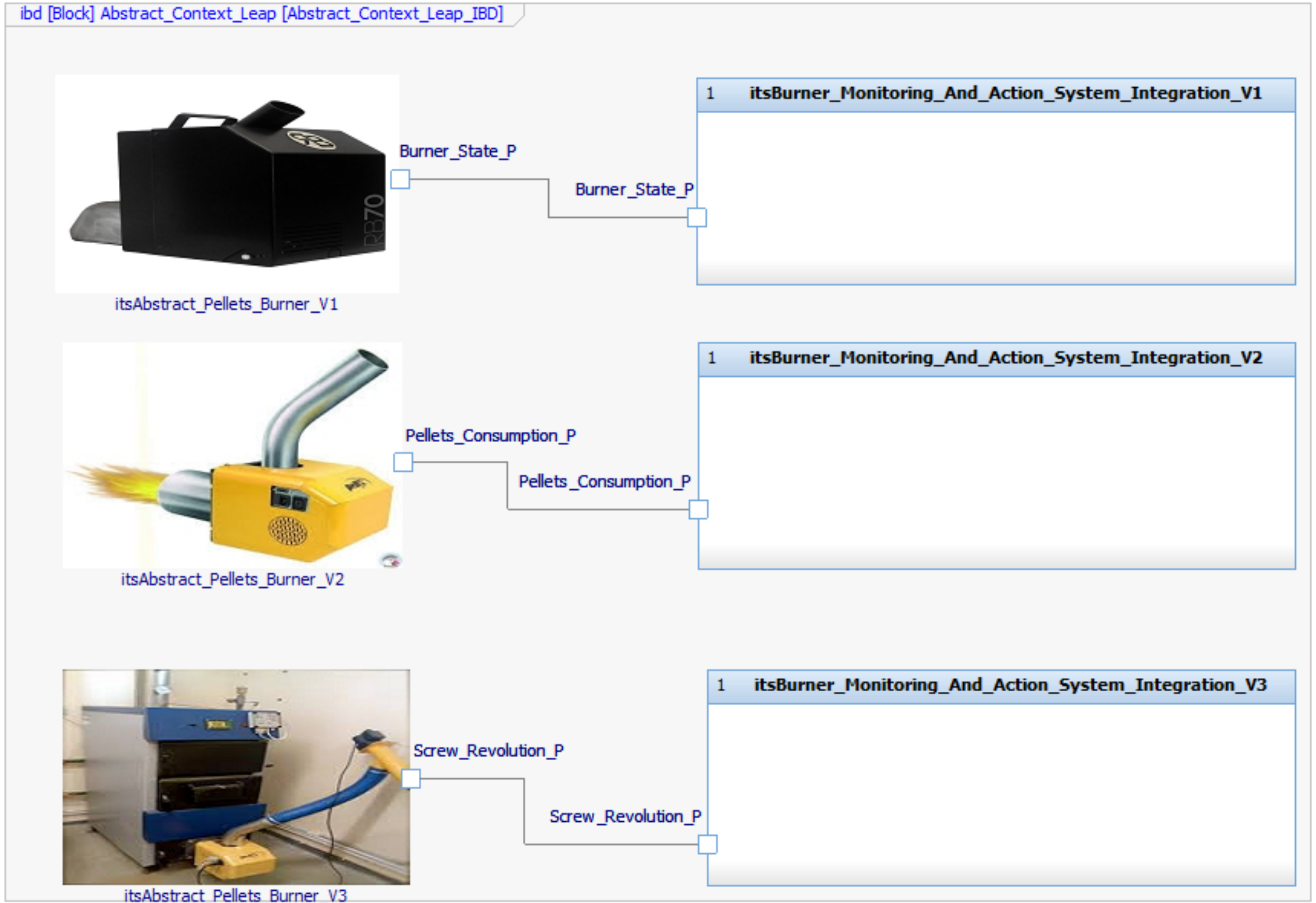


«Block»
Pellets_Consumption_Measurement_System
Screw_Revolutions_P
«Block»
Pellets_Feed_Screw_Revolution_Detection_System
«Block»
Revolution_To_Pellets_Consumption_LE
Pellets_Consumption
Screw_Revolutions
Screw_Revolutions

Moving towards a realizable system



Different versions of the integrated system


ibd [Block] Abstract_Context_Leap [Abstract_Context_Leap_IBD]
Burner_State_P
Burner_State_P
1 itsBurner_Monitoring_And_Action_System_Integration_V1
itsAbstract_Pellets_Burner_V1
Pellets_Consumption_P
Pellets_Consumption_P
1 itsBurner_Monitoring_And_Action_System_Integration_V2
itsAbstract_Pellets_Burner_V2
Screw_Revolution_P
Screw_Revolution_P
1 itsBurner_Monitoring_And_Action_System_Integration_V3
itsAbstract_Pellets_Burner_V3

Burner state

↓

Pellets consumption

↓

Screw revolutions

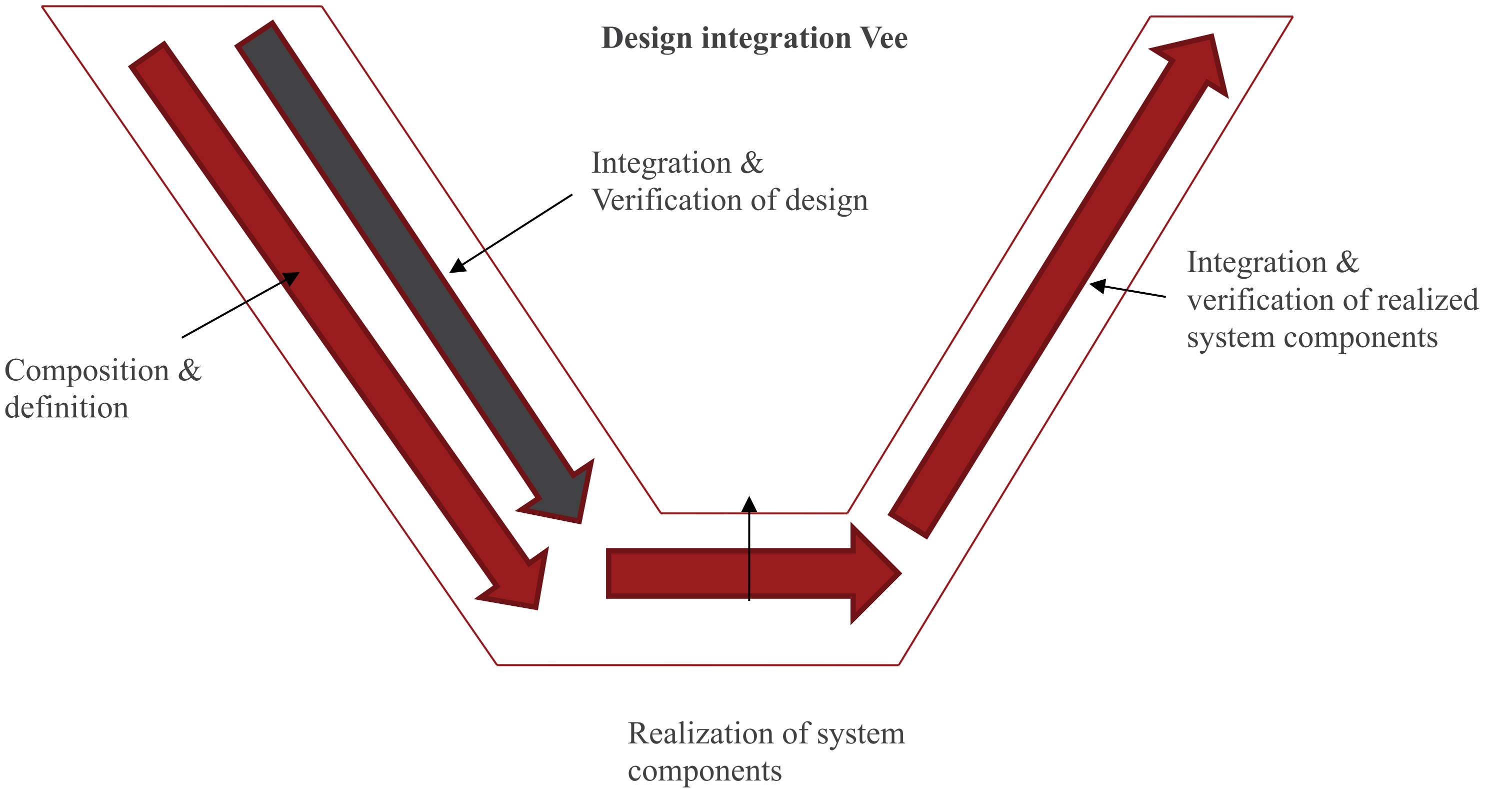
Continuous Design Integration

Executable system models

- Functional composition makes it possible to fully specify the mapping of inputs to outputs on all system levels.
- The system design can be made executable (also in practice).

- An executable and composable design enables a **new development model.**

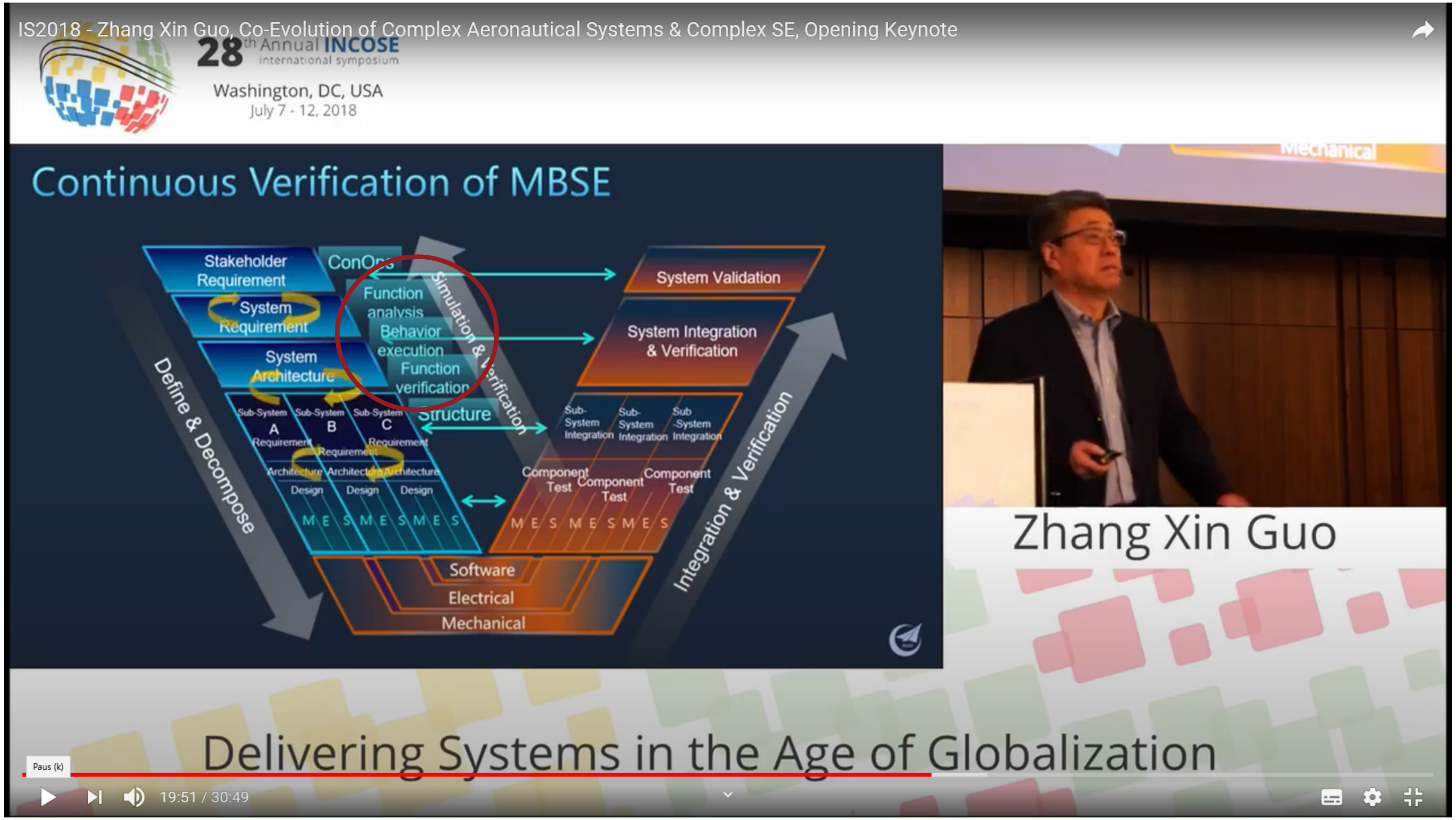
Design integration Vee - model


Design integration Vee
Integration & Verification of design
Integration & verification of realized system components
Composition & definition
Realization of system components

Key findings

- A top-down composition design methodology, including:
  - Executable design models and integration models
  - Very early and continuous design integration
  - Very early and continuous verification of design
  - Very early and continuous validation of

The MBSE dream



IS2018 - Zhang Xin Guo, Co-Evolution of Complex Aeronautical Systems & Complex SE, Opening Keynote

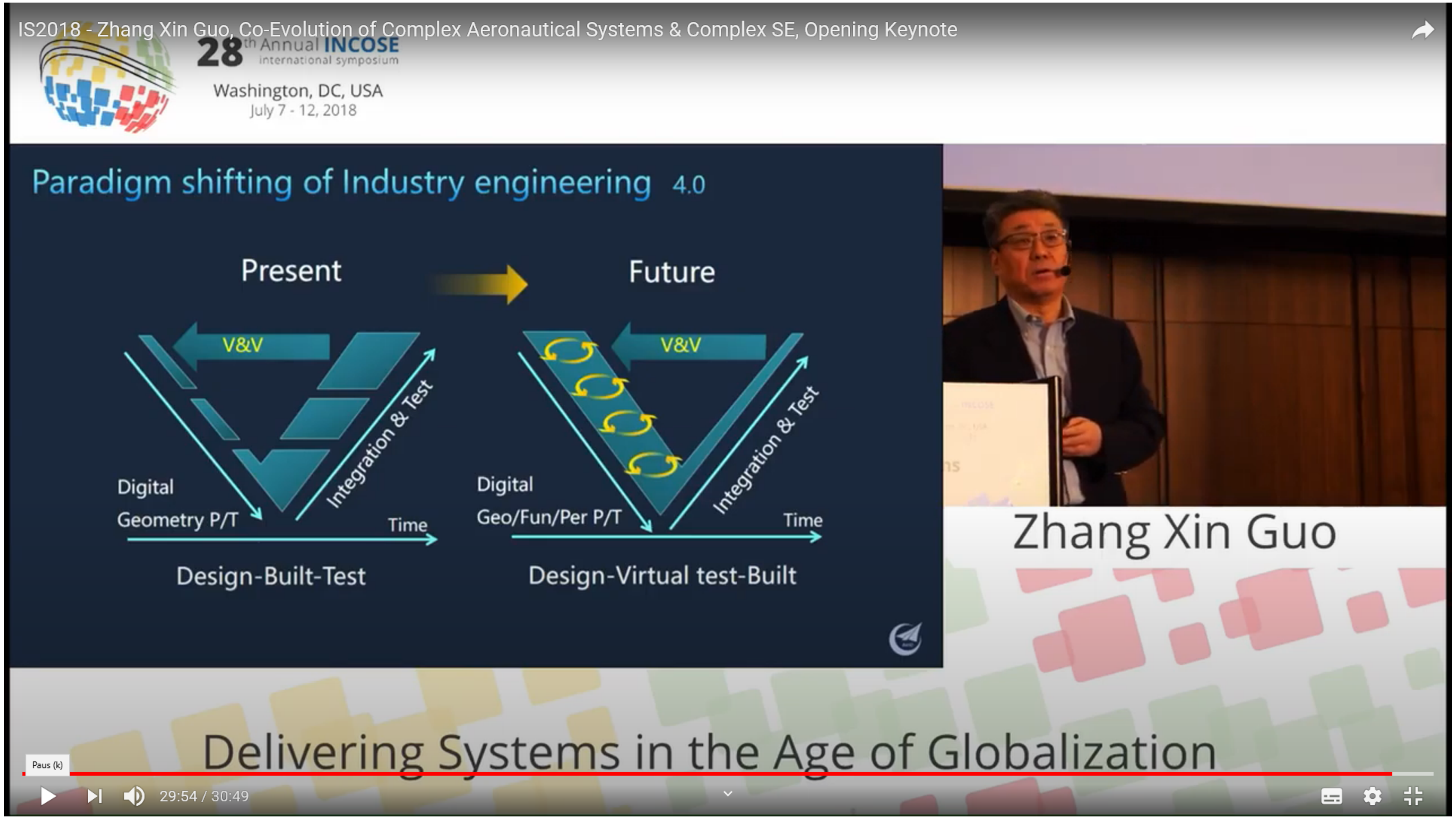
28th Annual INCOSE international symposium
Washington, DC, USA
July 7 - 12, 2018

Continuous Verification of MBSE

Zhang Xin Guo

Delivering Systems in the Age of Globalization

The MBSE dream



IS2018 - Zhang Xin Guo, Co-Evolution of Complex Aeronautical Systems & Complex SE, Opening Keynote

28th Annual INCOSE international symposium

Washington, DC, USA
July 7 - 12, 2018

Paradigm shifting of Industry engineering 4.0

Zhang Xin Guo

Delivering Systems in the Age of Globalization

MBSE - a new hope

- **Functional composition** and **Design integration** can be a game ch r SE & MBSE

- MBSE is, in the context of top-down composition, not just a communication

Questions?

Johan Bredin, SAAB Aeronautics
johan.bredin@saabgroup.com

Something to think about

SE4MBSE