# Introduction to Systems Engineering Synthesis and Feedback

Jean-Charles Chaudemar

ISAE-SUPAERO

1MAE003 - Introduction to Systems Engineering

## Outline

## Important ideas - 1/2

- Systems engineering enables the realisation of a baseline for the development of an intended complex system
- System engineer is in charge of all processes required to achieve this baseline
- Requirements are the main shape of this baseline for early design
- SE is of utmost importance to minimize requirement errors

## Important ideas - 1/2

- Systems engineering enables the realisation of a baseline for the development of an intended complex system
- System engineer is in charge of all processes required to achieve this baseline
- Requirements are the main shape of this baseline for early design
- SE is of utmost importance to minimize requirement errors

*Economical* stakes

- Tight relationship with the project management

## Important ideas - 2/2

- Modelling is a technics for studying abstract system according to specific aspects e.g. mechanical models for structural stress, mathematical models to study physics and behaviour
- Model analysis focusing on properties like safety, deadlock or liveness
- Output for requirement-related document: Users' Requirement Document, Technical Requirement Document, System Architecture Document

## Important ideas - 2/2

- Modelling is a technics for studying abstract system according to specific aspects e.g. mechanical models for structural stress, mathematical models to study physics and behaviour
- Model analysis focusing on properties like safety, deadlock or liveness
- Output for requirement-related document: Users' Requirement Document, Technical Requirement Document, System Architecture Document

Main advantages

- Re usability
- Sharing viewpoints and understanding
- Early implementation, Verification & Validation

## Event-B

- Formal language in order to state required properties and to find out new ones
- Models correct by construction thanks to theorem solvers/provers
- Refinement principle enables to tackle the complexity through progressive modelling
- Many Proof Obligation rules generated and discharged automatically

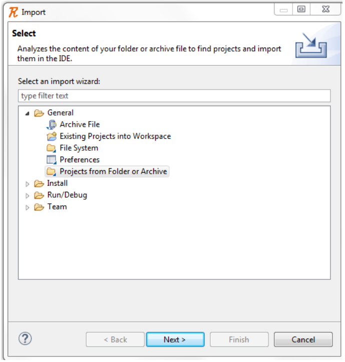
## Outline

## Tips for Rodin - 1/3

If you are not using the same PC environment, if your workspace is blind and if you don't know why, then do this:

- Create a new workspace
- Select File/Import/General/Project from Folder or Archive
- Click Next

THEN pray to get your project !!!


Import

**Select**

Analyzes the content of your folder or archive file to find projects and import them in the IDE.

Select an import wizard:

type filter text

- General
  - Archive File
  - Existing Projects into Workspace
  - File System
  - Preferences
  - Projects from Folder or Archive
- Install
- Run/Debug
- Team

## Tips for Rodin - 2/3

Reasons why an invariant PO could not be discharged automatically:

- Not automatical proof, need to try several provers
- Slow PC, so the prover would stop before achieving the proof end
- Invariants are incomplete or wrong
- Missing a guard, or incorrect guard(s)

Reasons why a guard PO (refinement) could not be discharged automatically:

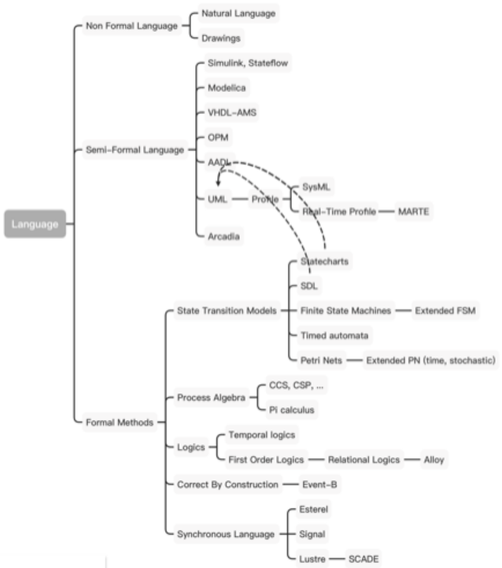
- Refined event incomplete
- Current guard(s) inconsistent with regard to the previous abstract event

## Tips for Rodin - 3/3

- Follow and respect scrupulously the rules. For instance, the deadlock freedom rule involves the guards of each guarded event, then you have to use exactly the same guards.
- Do not try to modify them in order to get a discharged PO rule.
- The DLF rule is a theorem, therefore if it is correctly written, you need to select a prover in order to discharge it.
- Do not hesitate to remove one invariant and to write it at the bottom of your invariants list. Sometimes, the position of one invariant may affect theorem provers.

## Outline

## Languages for MBSE



*Technical Report, de Saqui-Sannes, 2020*

## Requirement simulation tool

- Tool developed by a French company called Argosim
- Specify and validate formal requirement through simulation
- Offer a native graphical framework contrary to Rodin platform where plug-ins like B2UML or BRAMA give this graphical feature

## Pros & Cons

Comparison of Event-B vs. Stimulus.

*Give the Pros and Cons for Event-B*

**Pros**

## Pros & Cons

Comparison of Event-B vs. Stimulus.
*Give the Pros and Cons for Event-B*

**Pros**

- Theorem provers to validate the correctness of models by construction
- Simulation comes later after an exhaustive validation since the test does not allow to evaluate the whole space of possible values
- The formal notation sounds richer through first order predicate logic
- Refinement method gives the basis for an iterative and progressive methodology to be developped and adapted according to a specific domain

## Pros & Cons

Comparison of Event-B vs. Stimulus.
*Give the Pros and Cons for Event-B*

**Pros**

- Theorem provers to validate the correctness of models by construction
- Simulation comes later after an exhaustive validation since the test does not allow to evaluate the whole space of possible values
- The formal notation sounds richer through first order predicate logic
- Refinement method gives the basis for an iterative and progressive methodology to be developped and adapted according to a specific domain

**Cons**

- No amenable graphical editor
- Time analysis, even if it is constrained-time and not real-time
- Visualisation of results by plotting the behaviour