SystemX
INSTITUT DE RECHERCHE TECHNOLOGIQUE

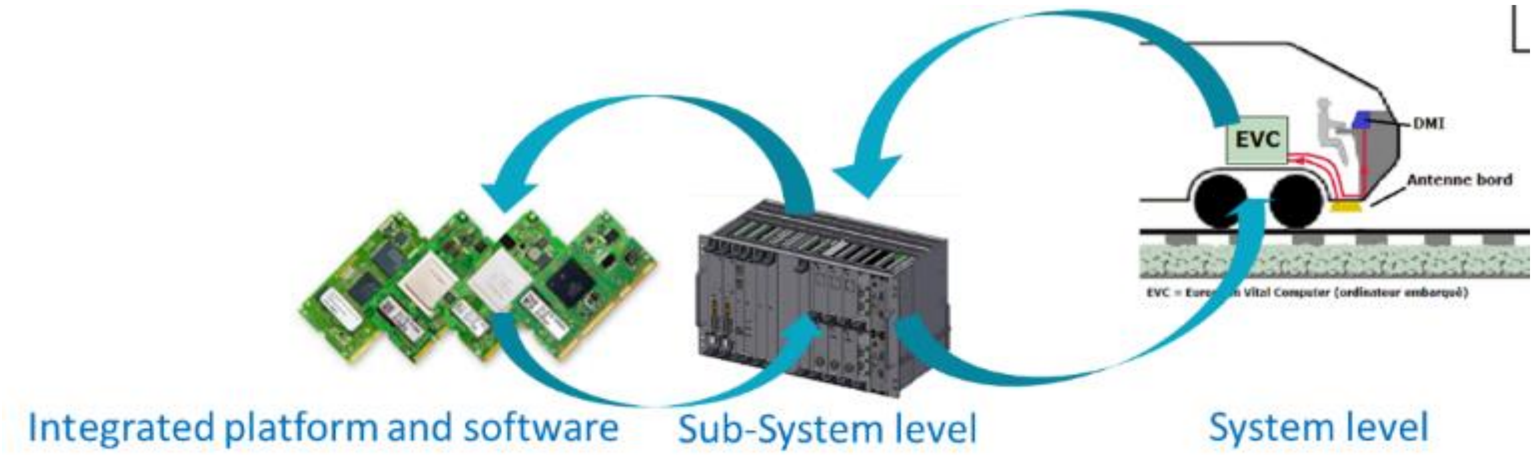
Accélérateur
de la transformation
numérique

Engineering Railway Systems with an Architecture-Centric Process Supported by AADL and ALISA: an Experience Report

Paolo Crisafulli, Dominique Blouin, Francoise Caron, Cristian Maxim

ERTS 2020 – 30/1/2020

Context: ETCS on-board and EVC

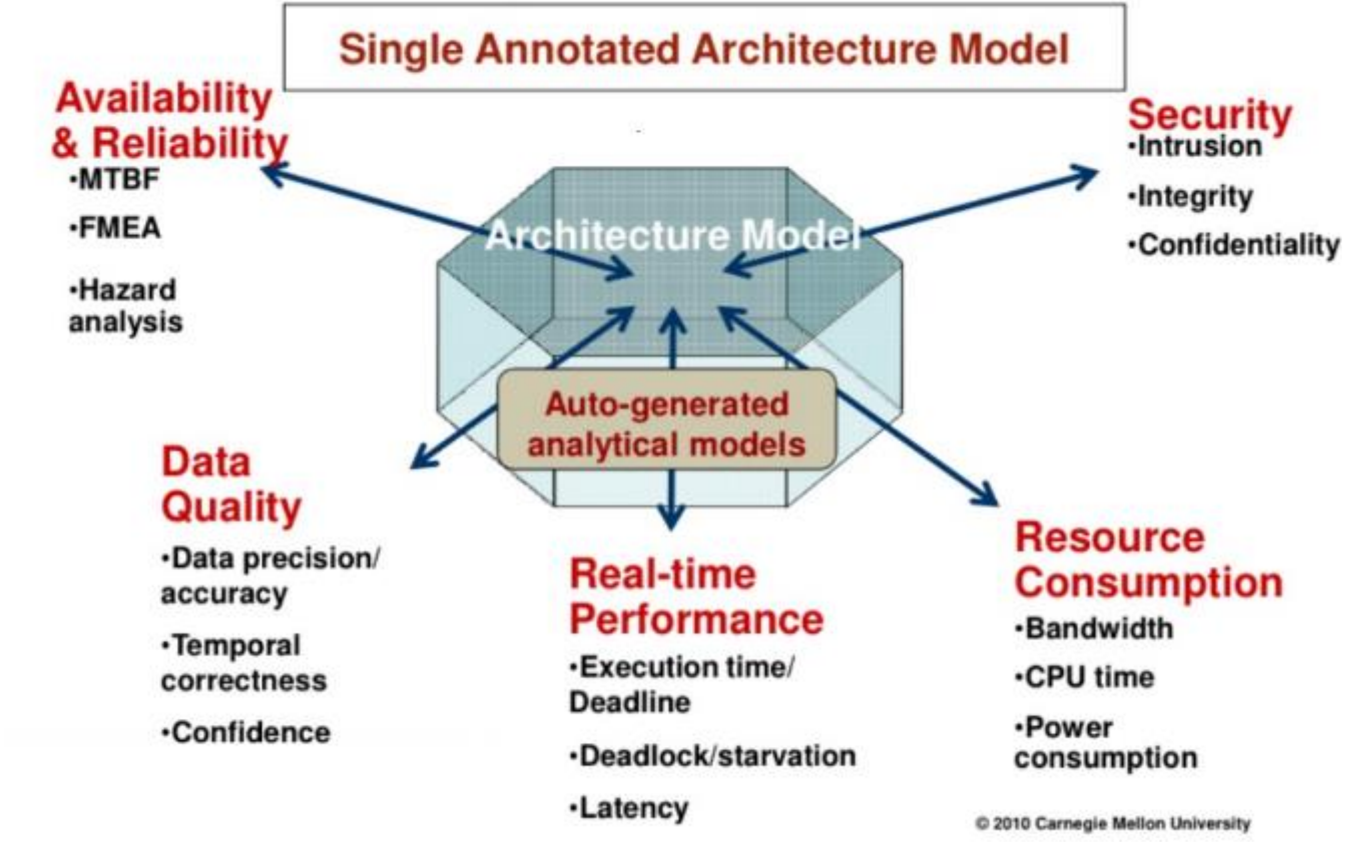

EVC
DMI
Antenne bord
EVC = Euro Vital Computer (ordinateur embarqué)
Integrated platform and software
Sub-System level
System level

ERTMS: European Rail Traffic Management System

ETCS: European Train Control System

EVC: European Vital Computer

AADL - Architecture Analysis and Design Language



Single Annotated Architecture Model

Architecture Model

Auto-generated analytical models

Availability & Reliability

- MTBF
- FMEA
- Hazard analysis

Security

- Intrusion
- Integrity
- Confidentiality

Data Quality

- Data precision/ accuracy
- Temporal correctness
- Confidence

Real-time Performance

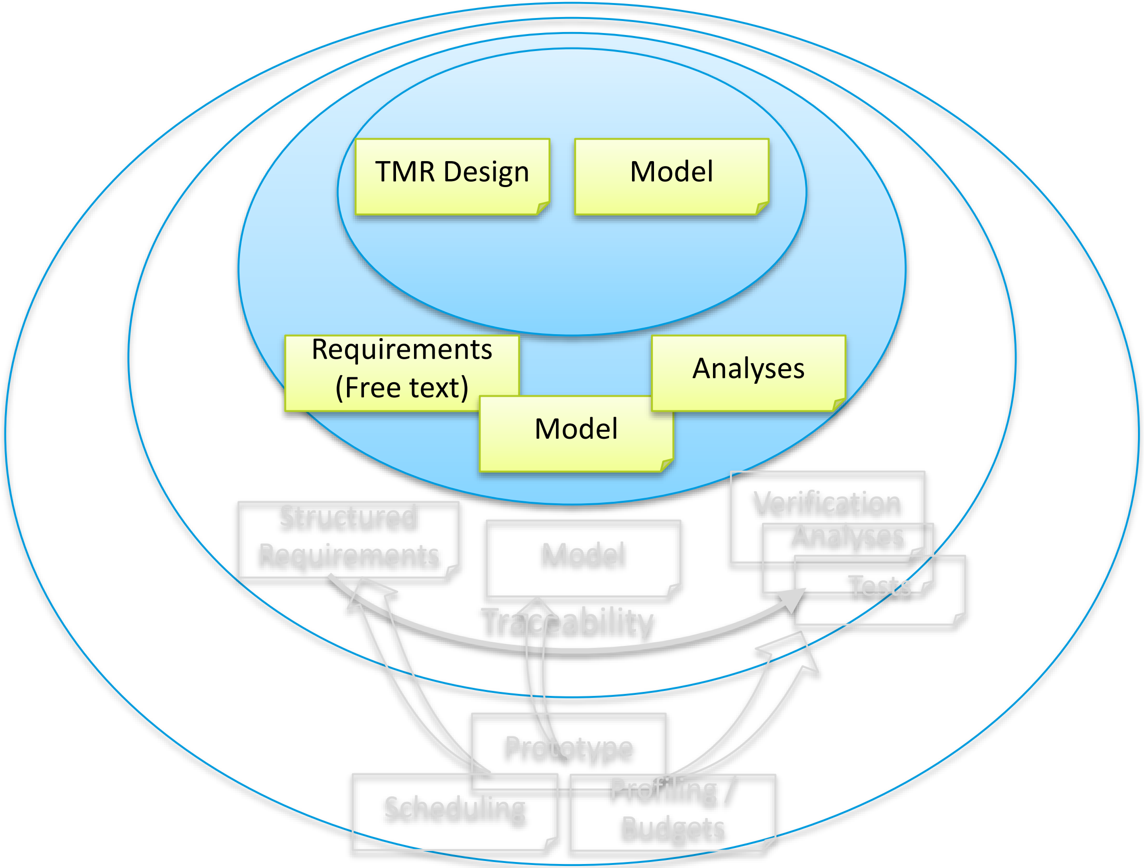
- Execution time/ Deadline
- Deadlock/starvation
- Latency

Resource Consumption

- Bandwidth
- CPU time
- Power consumption

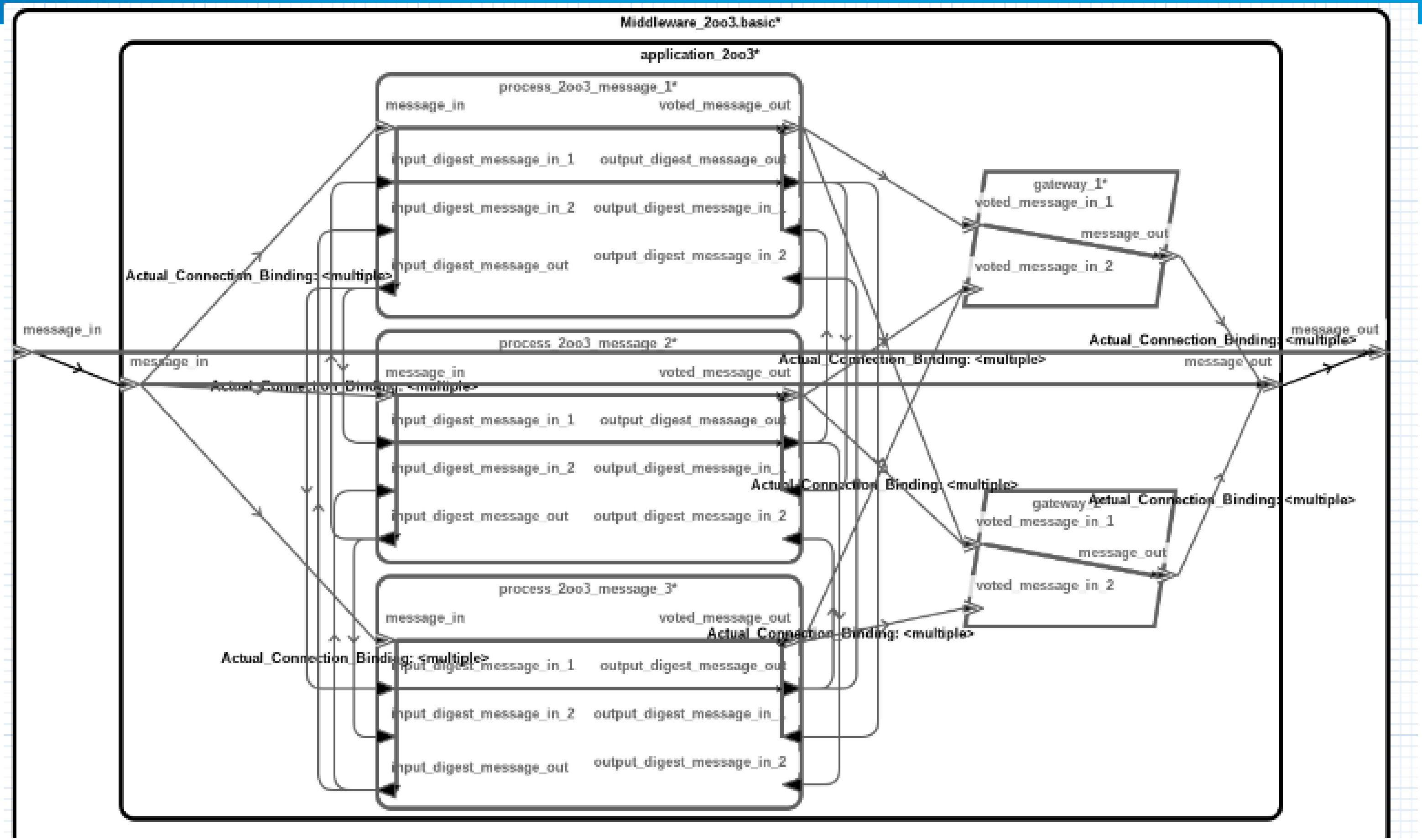
© 2010 Carnegie Mellon University

Our journey with AADL

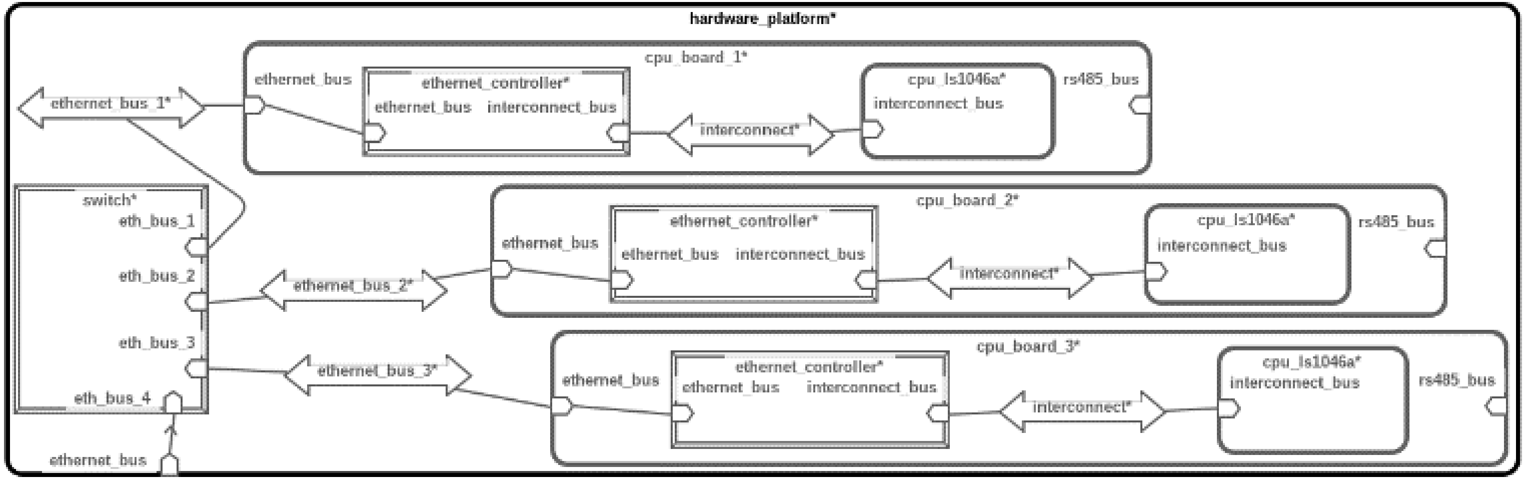


- **Expressivity: TMR**
- **Performance Analyses: RT**
- Traceability and requirements verification
- Prototyping
- Model refinement

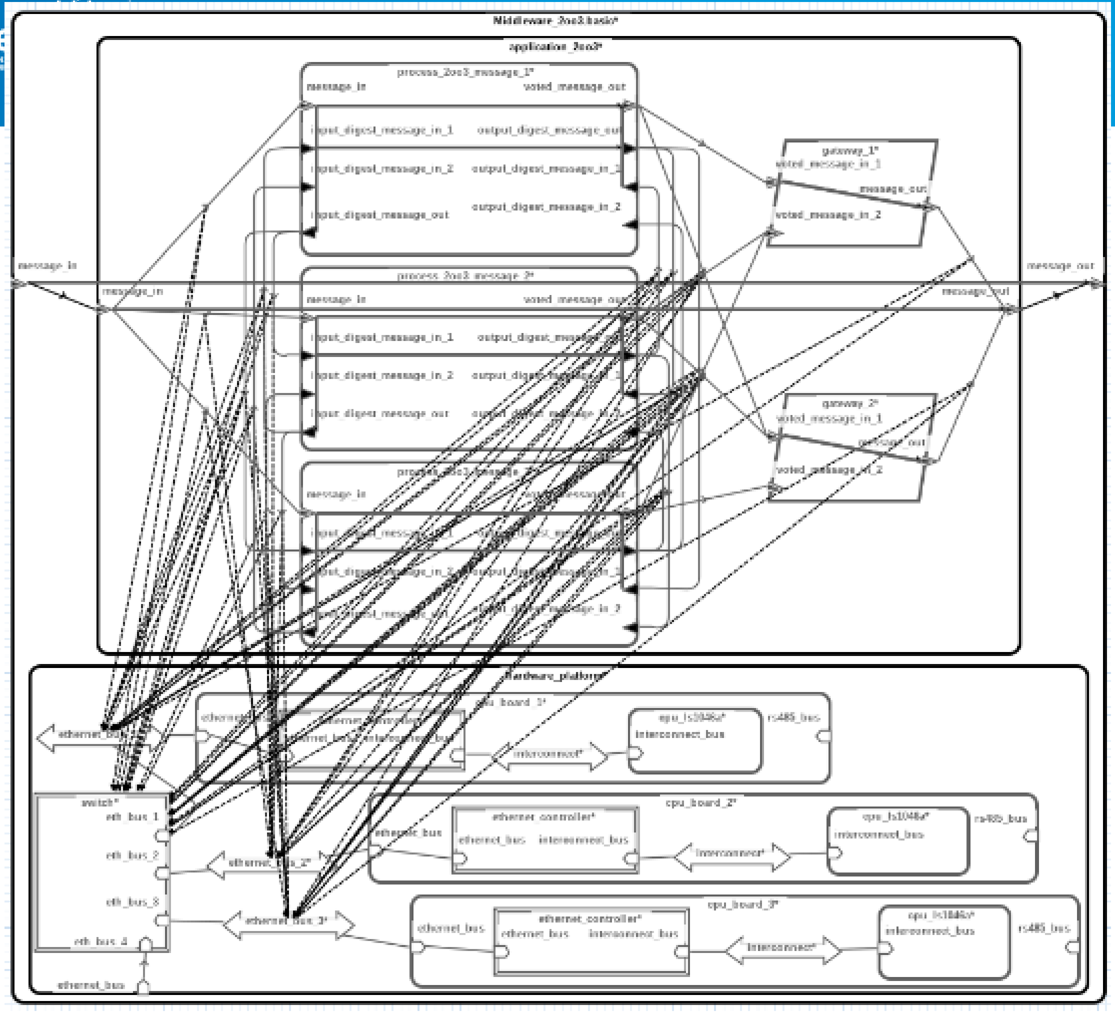
AADL Model: Software


Middleware_2oo3.basic*
application_2oo3*
process_2oo3_message_1*
message_in
voted_message_out
input_digest_message_in_1
output_digest_message_out
input_digest_message_in_2
output_digest_message_in_
input_digest_message_out
output_digest_message_in_2
Actual_Connection_Binding: <multiple>
message_in
message_in
process_2oo3_message_2*
message_in
voted_message_out
Actual_Connection_Binding: <multiple>
input_digest_message_in_1
output_digest_message_out
input_digest_message_in_2
output_digest_message_in_
input_digest_message_out
output_digest_message_in_2
process_2oo3_message_3*
message_in
voted_message_out
Actual_Connection_Binding: <multiple>
input_digest_message_in_1
output_digest_message_out
input_digest_message_in_2
output_digest_message_in_
input_digest_message_out
output_digest_message_in_2
gateway_1*
voted_message_in_1
message_out
voted_message_in_2
Actual_Connection_Binding: <multiple>
message_out
Actual_Connection_Binding: <multiple>
message_out
Actual_Connection_Binding: <multiple>
gateway_1*
Actual_Connection_Binding: <multiple>
voted_message_in_1
message_out
voted_message_in_2
Actual_Connection_Binding: <multiple>

AADL Model: Hardware



AADL Model: Bindings

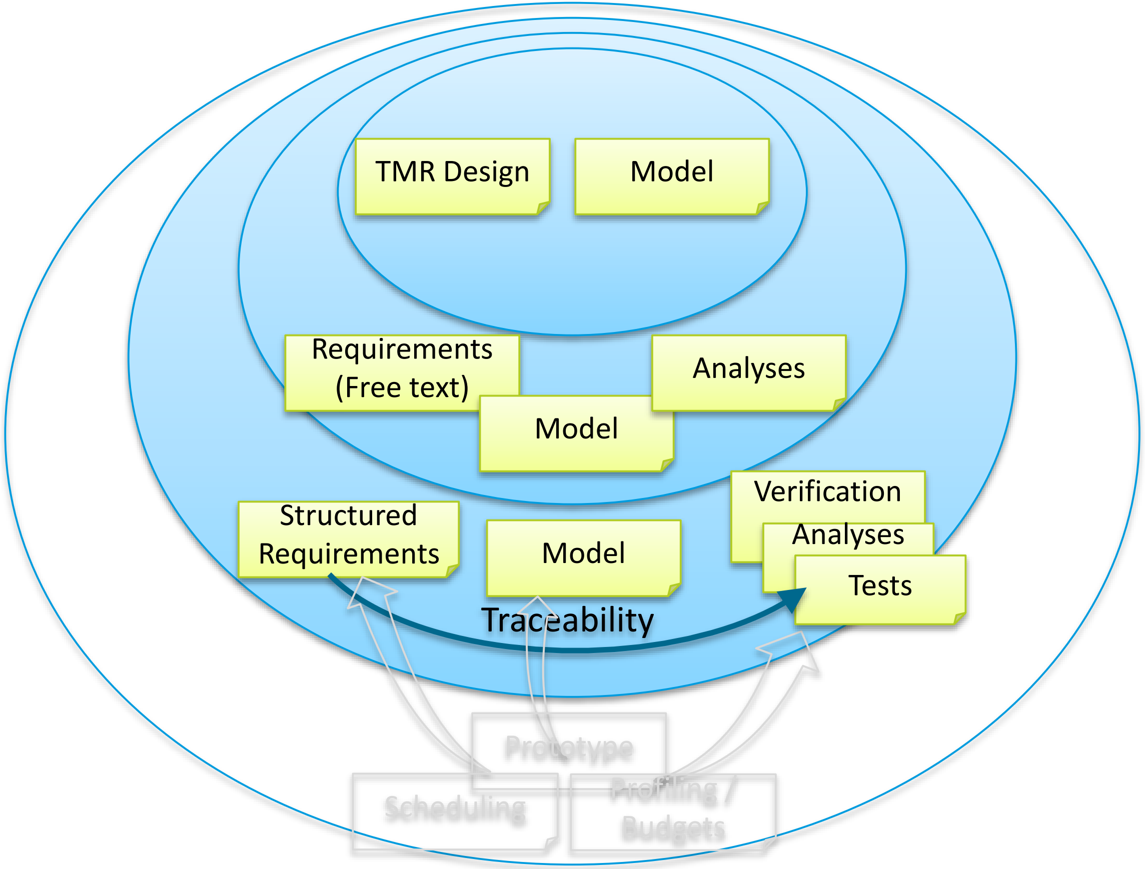


Some requirements for the EVC

Focus on performance requirements verification:

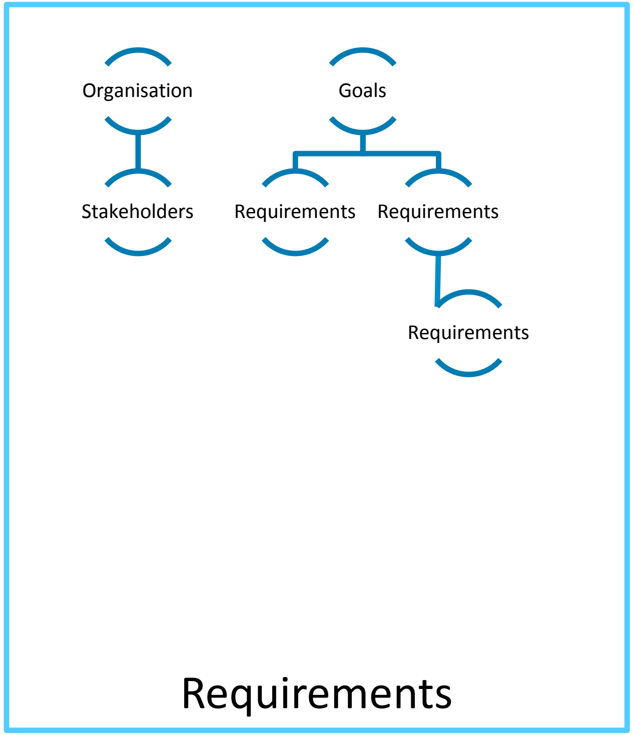
- **Latency < 300 ms**
- **Incoming messages <= 1000 msg/s**
- **Safety and availability:**
    - THR of 0.67 x 10-9 dangerous failures/hour
    - 2oo3 (aka TMR) design
        - Verify some 2oo3 design constraints
            - Same threads shall run on each board
            - Boards shall be of the same model
- **Design rules (reusable good practices)**
    - All input and output ports, physical or logical, shall be connected.
    - All threads shall be periodic

Our journey with AADL

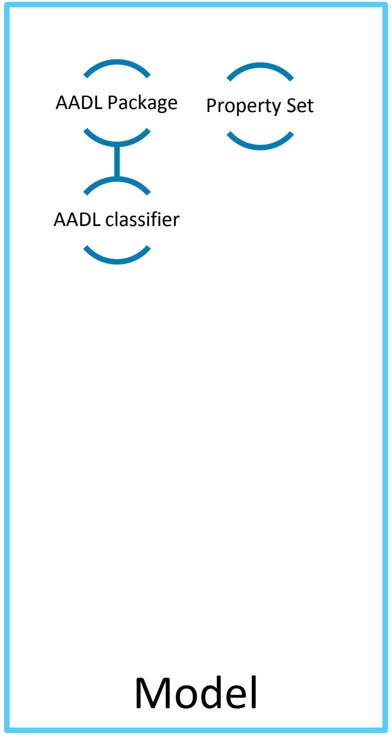


- **Expressivity**
- **Performance Analyses**
- **Requirements traceability and verification**
- Prototyping
- Model refinement

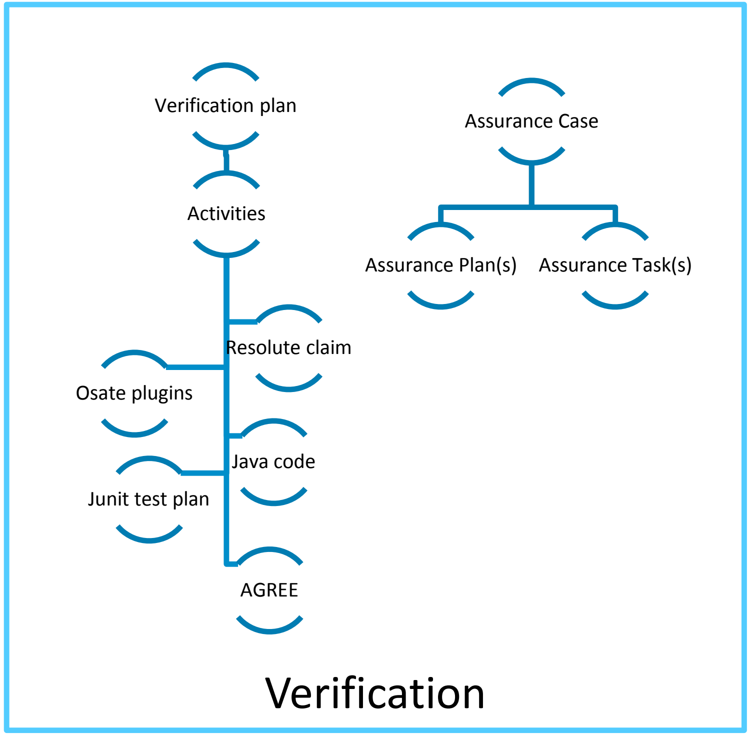
ALISA - Concepts and organisation



Requirements

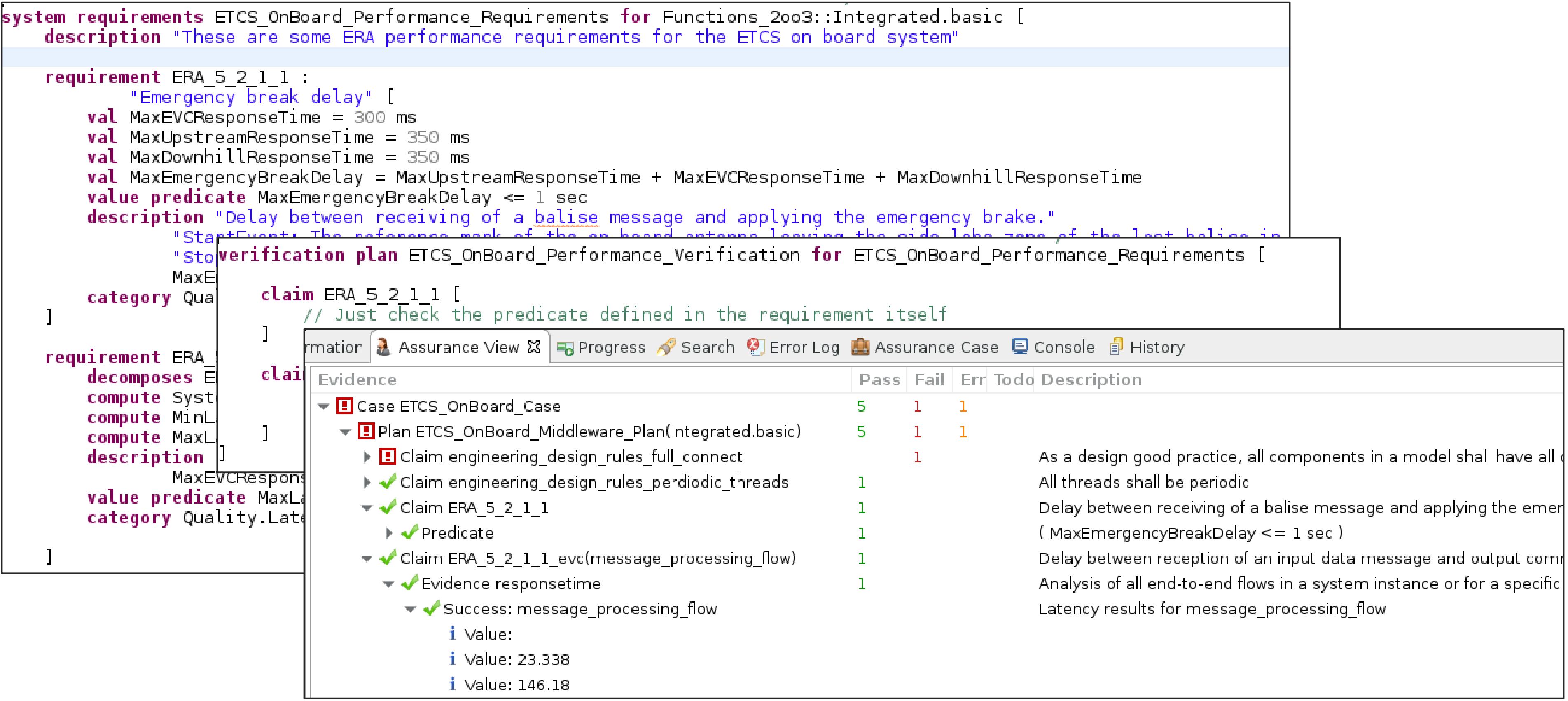


Model



Verification

EVC Requirements - ALISA

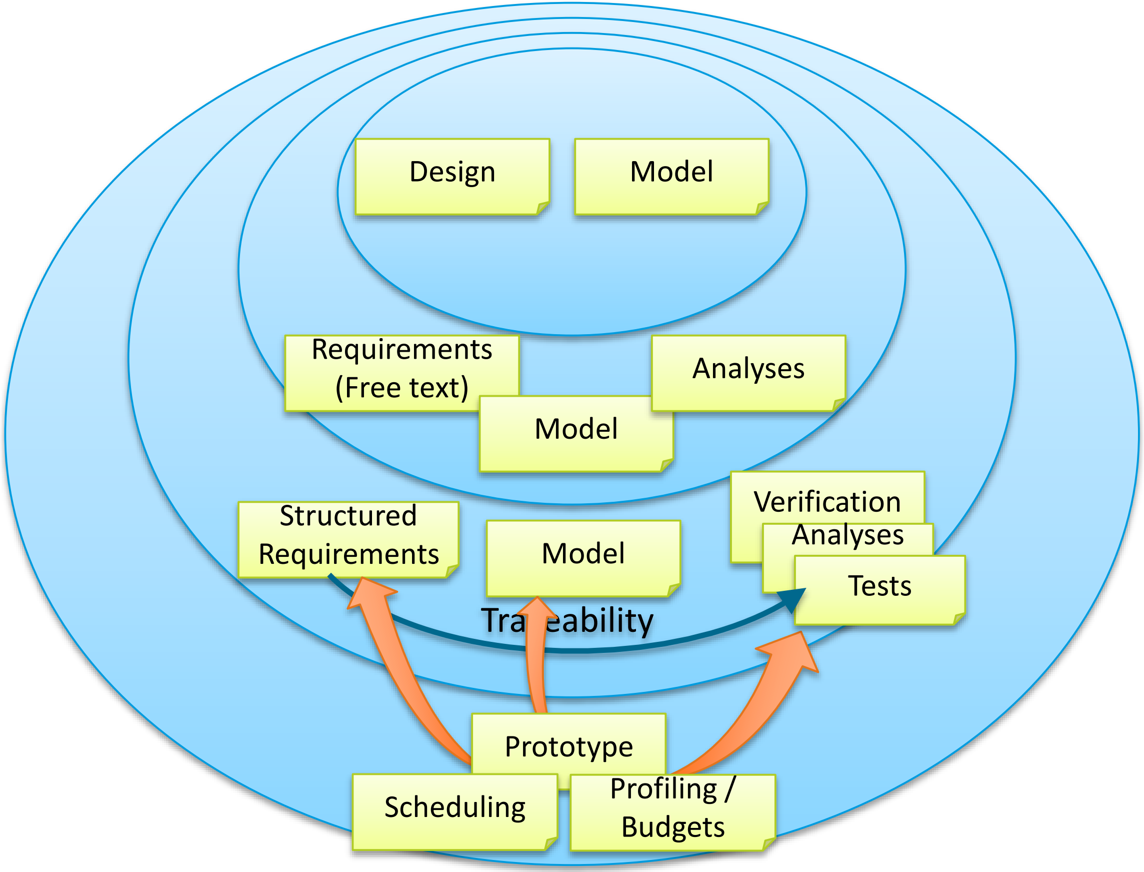


```
system requirements ETCS_OnBoard_Performance_Requirements for Functions_2oo3::Integrated.basic [
    description "These are some ERA performance requirements for the ETCS on board system"

    requirement ERA_5_2_1_1 :
            "Emergency break delay" [
        val MaxEVCResponseTime = 300 ms
        val MaxUpstreamResponseTime = 350 ms
        val MaxDownhillResponseTime = 350 ms
        val MaxEmergencyBreakDelay = MaxUpstreamResponseTime + MaxEVCResponseTime + MaxDownhillResponseTime
        value predicate MaxEmergencyBreakDelay <= 1 sec
        description "Delay between receiving of a balise message and applying the emergency brake."
                "StartEvent: The reference mark of the on board antenna leaving the side lobe zone of the last balise in
                "Sto
                MaxE
        category Qua
    ]

    requirement ERA_
        decomposes E
        compute Syst
        compute MinL
        compute MaxL
        description
                MaxEVCRespons
        value predicate MaxL
        category Quality.Late

    ]
```

```
verification plan ETCS_OnBoard_Performance_Verification for ETCS_OnBoard_Performance_Requirements [

    claim ERA_5_2_1_1 [
        // Just check the predicate defined in the requirement itself
    ]

    clai

    ]
]
```

Evidence	Pass	Fail	Err	Todo	Description
Case ETCS_OnBoard_Case	5	1	1		
Plan ETCS_OnBoard_Middleware_Plan(Integrated.basic)	5	1	1		
Claim engineering_design_rules_full_connect		1			As a design good practice, all components in a model shall have all c
Claim engineering_design_rules_perdiodic_threads	1				All threads shall be periodic
Claim ERA_5_2_1_1	1				Delay between receiving of a balise message and applying the emer
Predicate	1				(MaxEmergencyBreakDelay <= 1 sec)
Claim ERA_5_2_1_1_evc(message_processing_flow)	1				Delay between reception of an input data message and output comr
Evidence responsetime	1				Analysis of all end-to-end flows in a system instance or for a specific
Success: message_processing_flow					Latency results for message_processing_flow
i Value:					
i Value: 23.338					
i Value: 146.18					

Iterative incremental approach with AADL



- **Expressivity**
- **Performance Analyses**
- **Requirements traceability and verification**
- **Prototyping**
- **Model refinement**

Towards an agile engineering process

- **This tooling works fine for standalone development**
- **How do we scale in requirements and team size ?**
- **Incremental development (versions history)**
  - Non regression
  - Keep track of verification results and KPIs
- **(Re)use the continuous integration paradigm**
- **Define ALISA requirements for major design and implementation choices**
- **KPI Charts**

Building blocks

Osate/ALISA/AADL Inspector

AADL parsing, analysis and verification platform

Git/Repo

Versioning system for the comprehensive source of all artifacts:

- Requirements
- Models and Code
- Verification activities
- Dockerfiles

Jenkins

Continuous integration

Triggers verification check on any change to the artifacts

Docker

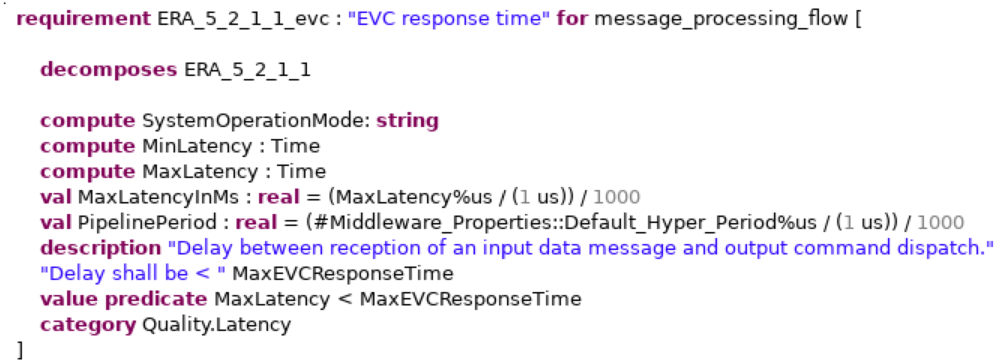
Container platform

Configuration management of the development, build and test environments

Non regression: Verify requirements, design choices, implementation choices

Requirements

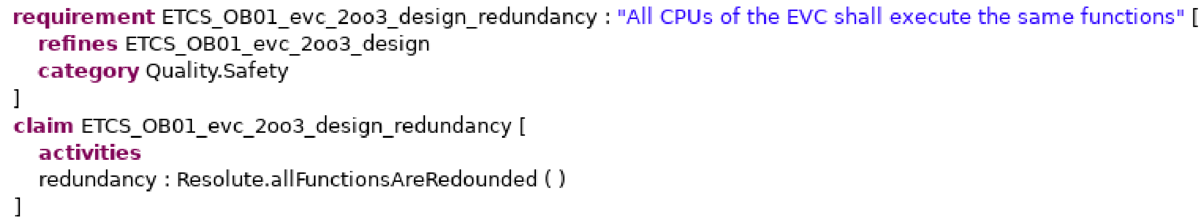
EVC response time shall be < 300 ms



```
requirement ERA_5_2_1_1_evc : "EVC response time" for message_processing_flow [

   decomposes ERA_5_2_1_1

   compute SystemOperationMode: string
   compute MinLatency : Time
   compute MaxLatency : Time
   val MaxLatencyInMs : real = (MaxLatency%us / (1 us)) / 1000
   val PipelinePeriod : real = (#Middleware_Properties::Default_Hyper_Period%us / (1 us)) / 1000
   description "Delay between reception of an input data message and output command dispatch."
   "Delay shall be < " MaxEVCResponseTime
   value predicate MaxLatency < MaxEVCResponseTime
   category Quality.Latency
]
```

Design

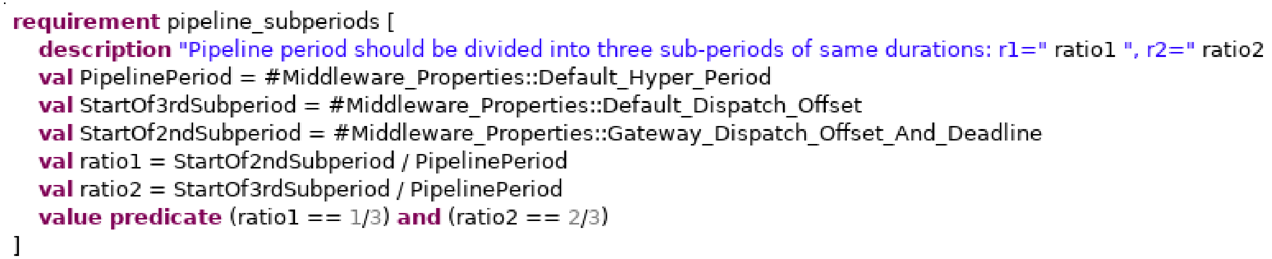
TMR: all functions shall be redounded



```
requirement ETCS_OB01_evc_2oo3_design_redundancy : "All CPUs of the EVC shall execute the same functions" [
   refines ETCS_OB01_evc_2oo3_design
   category Quality.Safety
]
claim ETCS_OB01_evc_2oo3_design_redundancy [
   activities
   redundancy : Resolute.allFunctionsAreRedounded ( )
]
```

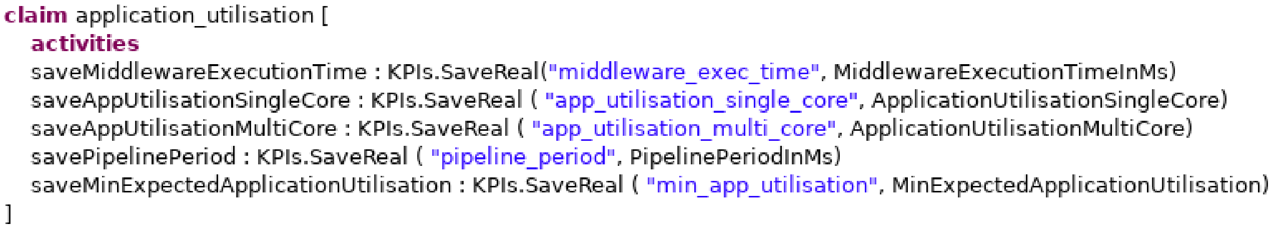
Implementation

Divide pipeline period into 3 subperiods

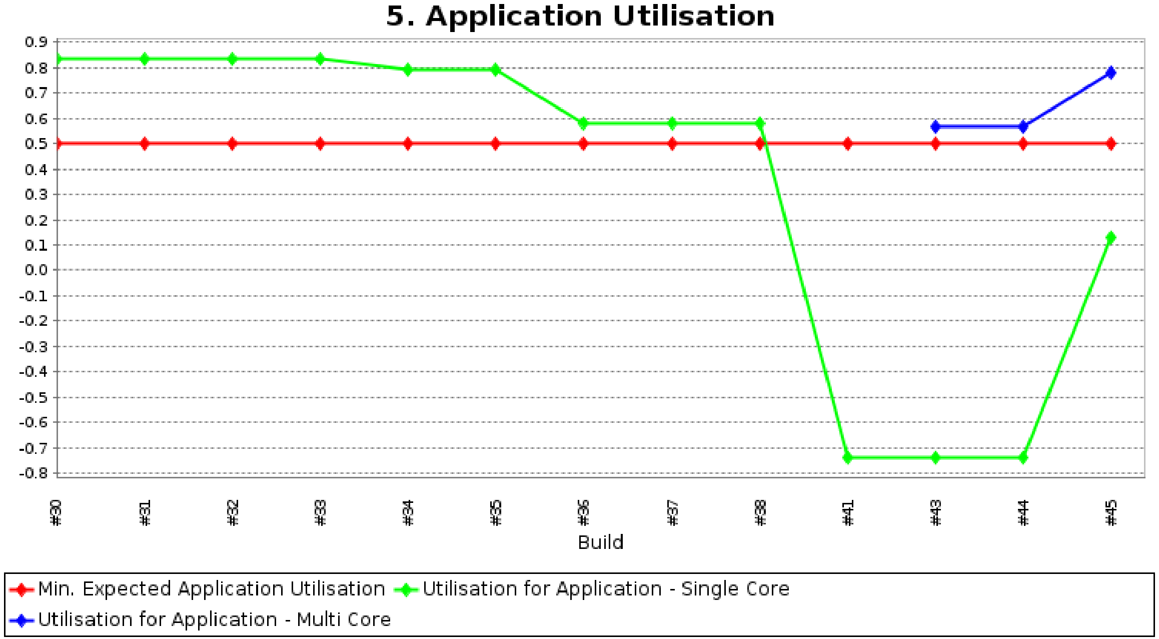


```
requirement pipeline_subperiods [
   description "Pipeline period should be divided into three sub-periods of same durations: r1=" ratio1 ", r2=" ratio2
   val PipelinePeriod = #Middleware_Properties::Default_Hyper_Period
   val StartOf3rdSubperiod = #Middleware_Properties::Default_Dispatch_Offset
   val StartOf2ndSubperiod = #Middleware_Properties::Gateway_Dispatch_Offset_And_Deadline
   val ratio1 = StartOf2ndSubperiod / PipelinePeriod
   val ratio2 = StartOf3rdSubperiod / PipelinePeriod
   value predicate (ratio1 == 1/3) and (ratio2 == 2/3)
]
```

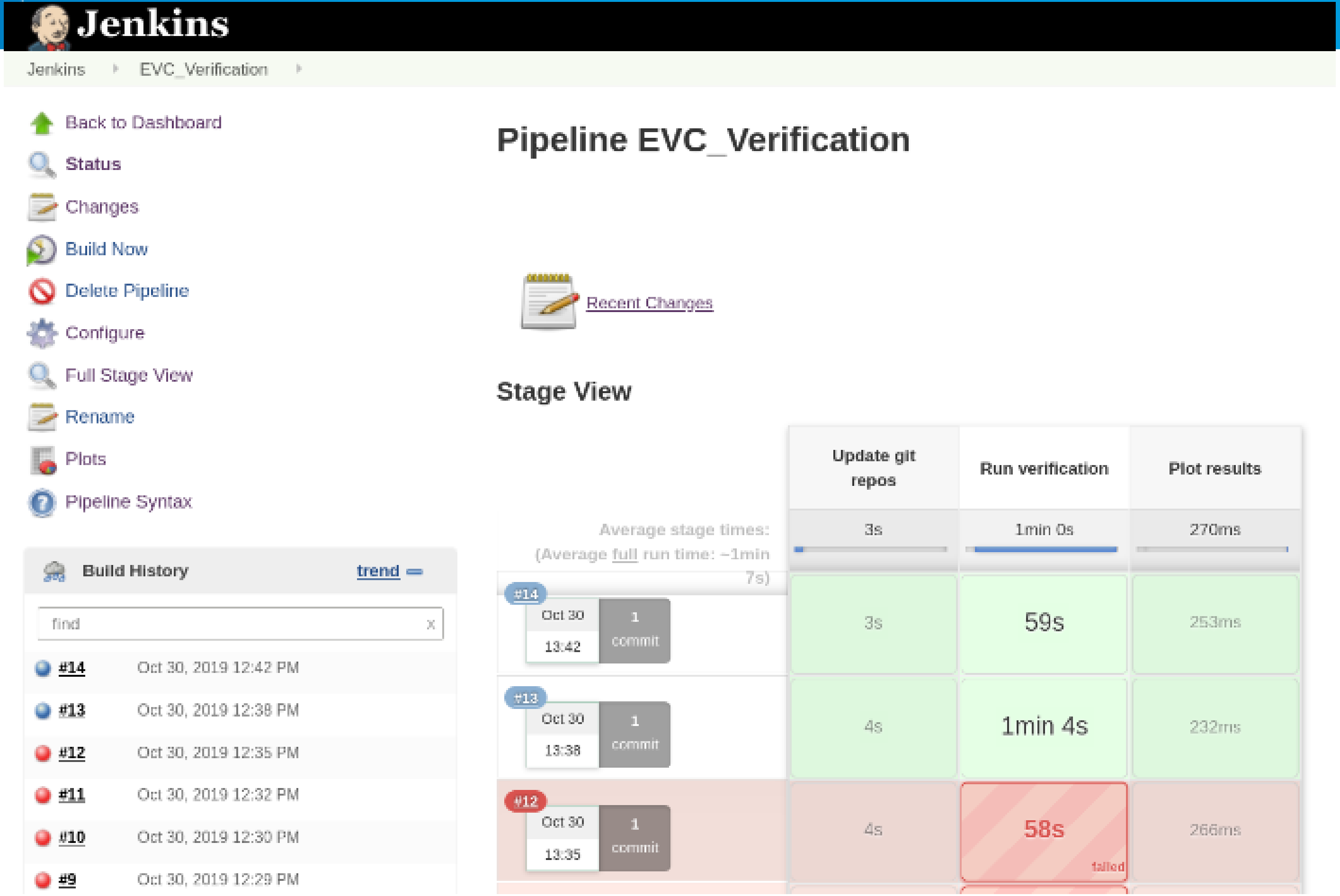
Keep track of the KPIs



```
claim application_utilisation [
  activities
  saveMiddlewareExecutionTime : KPIs.SaveReal("middleware_exec_time", MiddlewareExecutionTimeInMs)
  saveAppUtilisationSingleCore : KPIs.SaveReal ( "app_utilisation_single_core", ApplicationUtilisationSingleCore)
  saveAppUtilisationMultiCore : KPIs.SaveReal ( "app_utilisation_multi_core", ApplicationUtilisationMultiCore)
  savePipelinePeriod : KPIs.SaveReal ( "pipeline_period", PipelinePeriodInMs)
  saveMinExpectedApplicationUtilisation : KPIs.SaveReal ( "min_app_utilisation", MinExpectedApplicationUtilisation)
]
```



How it looks: build history



Jenkins

Jenkins ▸ EVC_Verification ▸

- Back to Dashboard
- Status
- Changes
- Build Now
- Delete Pipeline
- Configure
- Full Stage View
- Rename
- Plots
- Pipeline Syntax

Pipeline EVC_Verification

Recent Changes

Stage View

	Update git repos	Run verification	Plot results
Average stage times: (Average full run time: ~1min 7s)	3s	1min 0s	270ms
#14 Oct 30 13:42 1 commit	3s	59s	253ms
#13 Oct 30 13:38 1 commit	4s	1min 4s	232ms
#12 Oct 30 13:35 1 commit	4s	58s failed	266ms

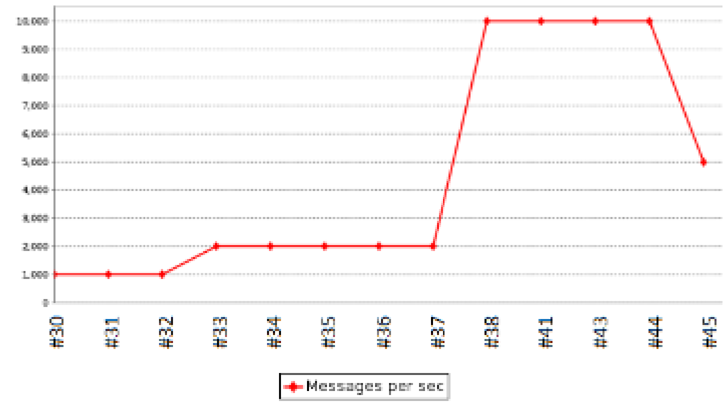
Build History trend —

find x

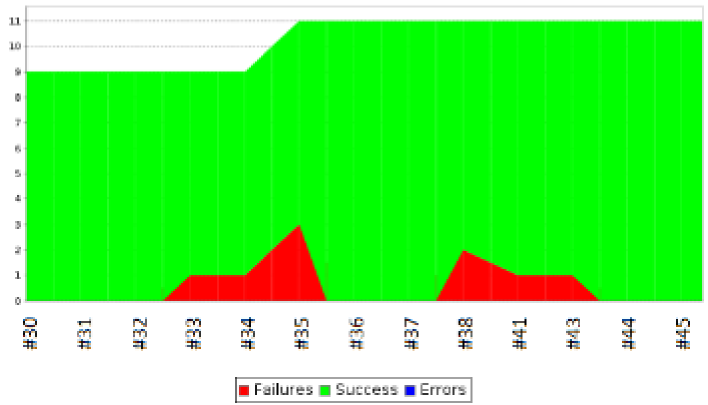
#14	Oct 30, 2019 12:42 PM
#13	Oct 30, 2019 12:38 PM
#12	Oct 30, 2019 12:35 PM
#11	Oct 30, 2019 12:32 PM
#10	Oct 30, 2019 12:30 PM
#9	Oct 30, 2019 12:29 PM

How it looks: verification and performance history

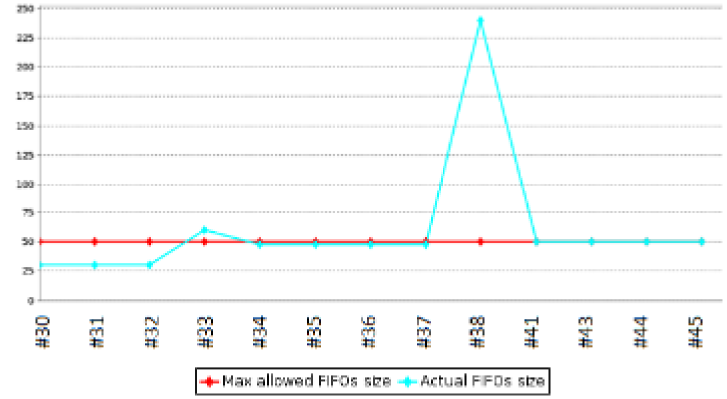
2. Input Messages per Second



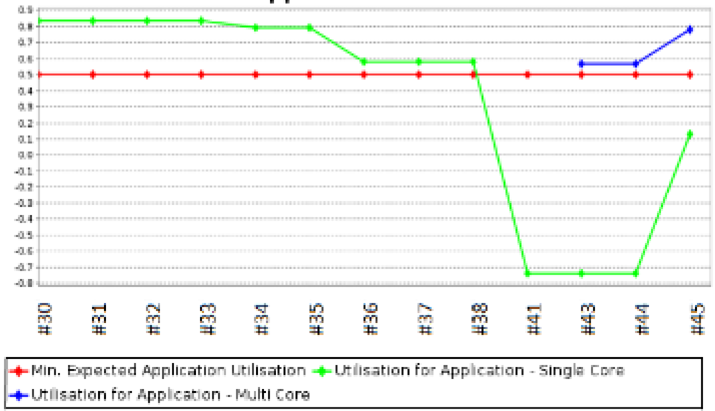
1. Verification results



3. FIFOs size



5. Application Utilisation



Conclusions

- A showcase of how AADL and ALISA can support an agile architecture-centric engineering process for a typical embedded system in the railway domain:
  - The continuous verification maintains the design within the solution space shaped by the set of requirements
  - The KPIs computation and charting qualify, in terms of performance, its evolution and alternatives over time.
- ALISA is currently still under stabilization, hence its usage cannot be recommended for an engineering team facing hard delivery deadlines.
- Nevertheless, this experiment illustrates where the AADL ecosystem of companion languages and development environments is standing, opening the way to agile engineering of highly constrained systems, such as critical systems requiring a certification process.
- Additional work: link to the overall system engineering process, SysCon 2020

Thank you!