

“Advancing Traffic Efficiency and Safety
through Software Technology, Phase 2 (ATESST2)”

Safety & Dependability

Yiannis Papadopoulos - University of Hull

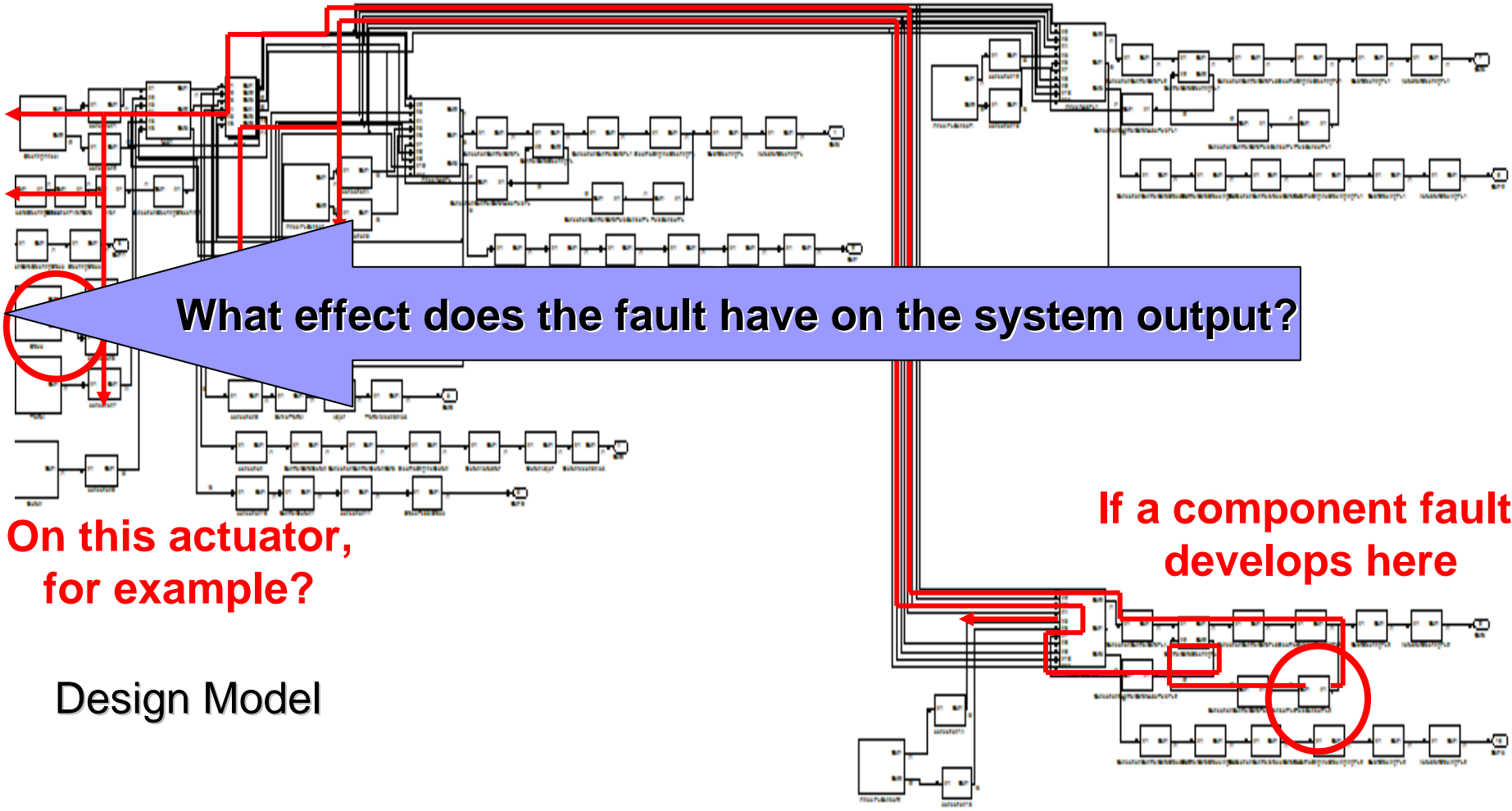
ATESST2 Final Workshop
June 21 2010



Motivation for Dependability Analysis

- Increasing safety concerns:
 - Computer controlled safety critical systems emerge in areas such as automotive, shipping, medical applications, industrial processes, etc.
- Reliability & availability concern a broader class of systems
- New standards such as the ISO DIS 26262 mandate analyses throughout the design process to ensure system safety
- Increasing complexity of systems & reduced product development times & budgets cause difficulties in classical manual analyses

Need for Automation & Tool Support



What effect does the fault have on the system output?

On this actuator,
for example?

If a component fault
develops here

Design Model

Aims/Outcomes on Dependability

- Provide support for error modelling and ISO-26262 in EAST-ADL
- Extend HiP-HOPS, a dependability analysis and optimisation tool, to enable analysis of EAST-ADL models:
 - Top-down allocation of safety requirements in the form of Safety Integrity Levels
 - Bottom-up dependability analysis in Fault Trees & FMEAs
 - Architecture optimisation with respect to dependability, timing and cost, using Genetic Algorithms

EAST-ADL Error Modelling Concepts

- EAST-ADL error model expanded to include concepts to support hazard & dependability analysis
- Uses a parallel architecture to model failure behaviour
 - **ErrorModelTypes & Prototypes** describe architecture elements and allow hierarchical composition of error models
 - **ErrorBehaviors** describe the failure behaviour of an element
 - **InternalFaults** and **FaultFailures** describe internal and interface failures
 - **FaultFailurePorts** allow propagation of failures from one error model to another by means of **FaultFailurePropagationLinks**
 - **Hazards** and **HazardousEvents** link component errors to system failures
 - **SafetyConstraints** allow integrity constraints to be specified on errors

Bottom up Dependability Analysis

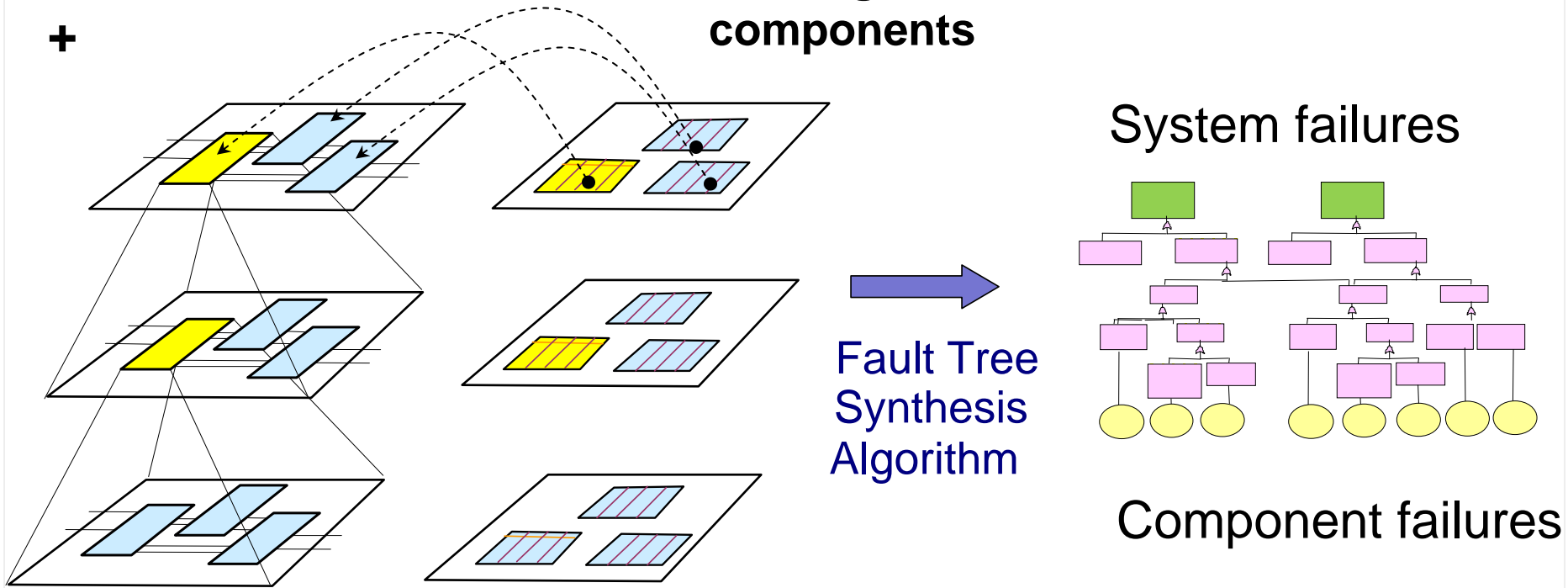
Via model-based synthesis of Fault Trees and FMEAs

EAST-ADL Model

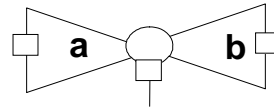
Error Modelling of components

= Global view of failure:

+



Component Error Modelling



Valve Malfunctions

control

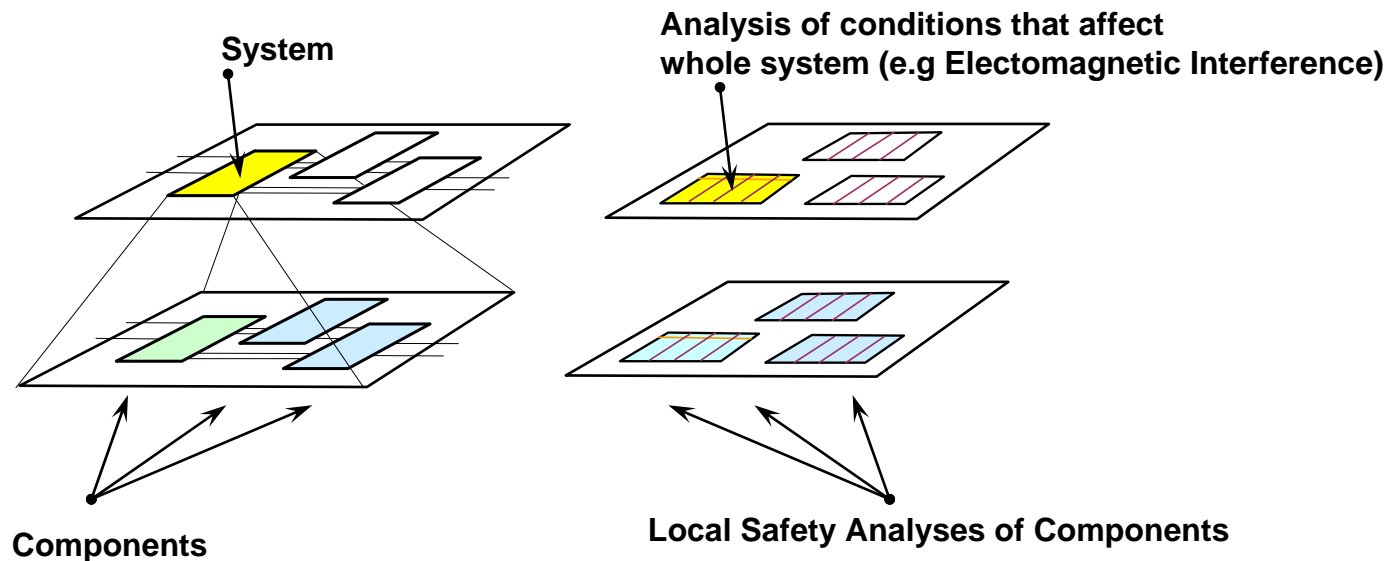
<i>Failure mode</i>	<i>Description</i>	<i>Failure rate</i>
Blocked	e.g. by debris	1e-6
partiallyBlocked	e.g. by debris	5e-5
stuckClosed	Mechanically stuck	1.5e-6
stuckOpen	Mechanically stuck	1.5e-5

Deviations of Flow at Valve Output

<i>Output Deviation</i>	<i>Description</i>	<i>Causes</i>
Omission-b	Omission of flow	Blocked or stuckClosed or Omission-a or Low-control
Commission-b	Commission of flow	stuckOpen or Commission-a or High-control
Low-b	Low flow	partiallyBlocked or Low-a
High-b	High flow	High-a
Early-b	Early flow	Early-a or Early-control
Late-b	Late flow	Late-a or Late-control

Hierarchical Failure Analysis

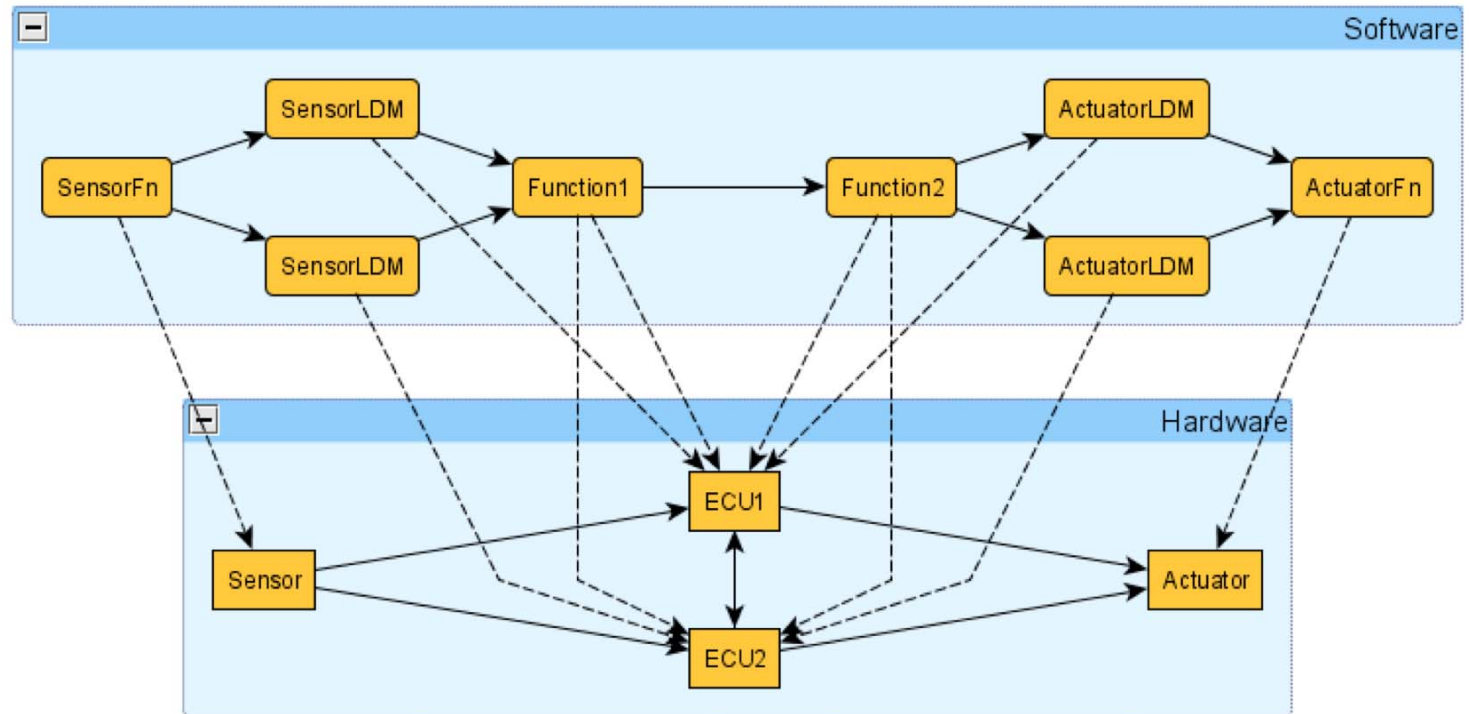
- Assessment of conditions that affect whole architectures, e.g. common cause failures is possible



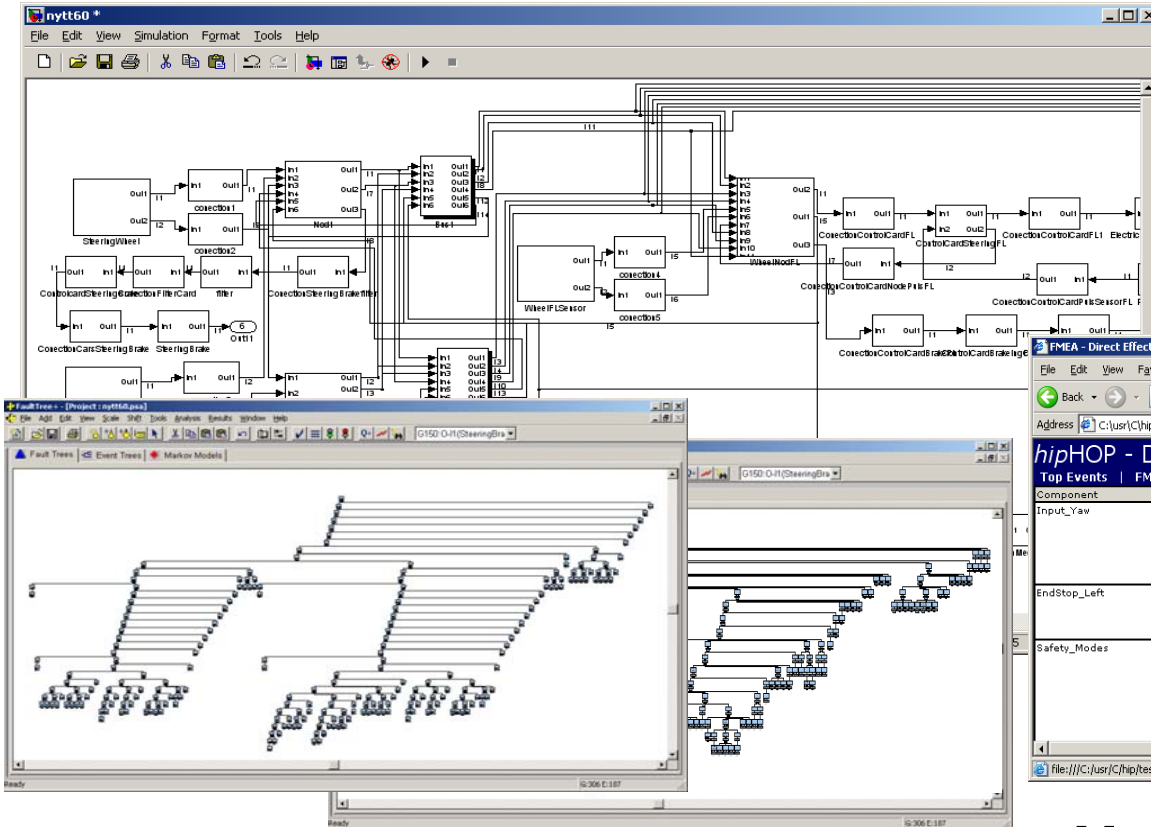
Multi-Perspective Analysis

- Analysis of systems modelled in more than one perspective, e.g. hardware and software

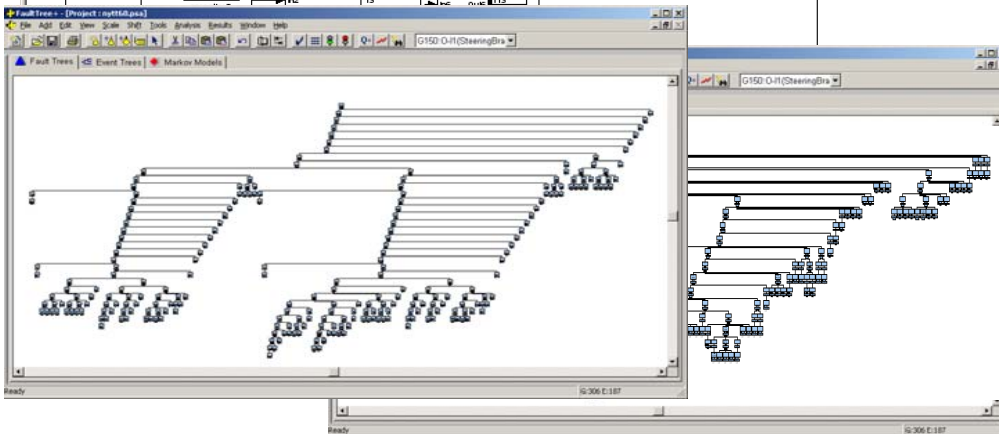
Errors can propagate via allocation relationships, e.g. software on an ECU



Dependability Analysis: Results



System Model



Fault Trees

FMEA - Direct Effects - Microsoft Internet Explorer

Address: C:\usr\C\hip\test\FTOutput\FTA1070532429\FMEA\FMEA-DirectEffects.htm

hipHOP - Direct Effects

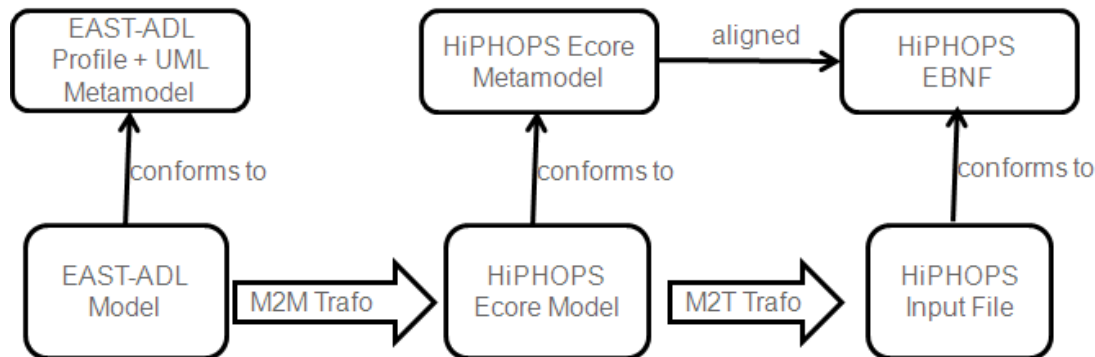
Top Events | FMEA

Component	Failure Mode	Description	Direct Effects
Input_Yaw	E48:Omission(root)(Input_Yaw)	Input of value yawrate fails	G55:O-1(Steering_Feeds)
	E14:Commission(root)(Input_Yaw)	The input of read yawrate has wrong value	G55:O-1(Steering_Feeds)
	E49:CommissionStatus(Input_Yaw)	Status not OK sent for Input_Yaw although because wrongly diagnostic a failure	G55:O-1(Steering_Feeds)
EndStop_Left	E80:Omission(EndStop_Left)	The end stop left will not be applied	G55:O-1(Steering_Feeds)
	E86:Commission(EndStop_Left)	Commission of end stop left which implies that the steering wheel can not turn passed the wrongly put stop	G215:O-1(Steering_Inp)
Safety_Modes	E28:Omission(root)(Safety_Modes)	Internal failure is not detected sending OK when not OK	G313:O-1(Steering_Degradation) G340:O-1(Steering_Degradation) G356:O-1(Steering_Degradation)
	E63:Commission(root)(Safety_Modes)	Commission of the calculated safety mode giving not OK when OK	G55:O-1(Steering_Feeds) G207:O-1(Full_Steering)

Multiple Failure Mode FMEA

Papyrus - HiP-HOPS Plugin

- Interface between EAST-ADL in Papyrus and HiP-HOPS
- Uses model transformation technology to convert EAST-ADL2 model into HiP-HOPS model in two phases
 - Semantic mapping of metamodel concepts in ATL
 - Representation transformation in Xpand



- Links new features in both EAST-ADL and HiP-HOPS

Supporting ISO 26262

- Major focus to support ISO 26262
- Features added to the EAST-ADL and HiP-HOPS to support:
 - **Hazard analysis at vehicle feature level**
 - **Automatic allocation of ASILs to components of the architecture**
- Safety methodology developed to make use of EAST-ADL and HiP-HOPS in ISO compliant processes

Hazard Analysis

- EAST-ADL supports hazard analysis at vehicle feature level
- **Item** is identified - the design element under analysis
- **Hazards** are determined - the malfunctions of the item
- **Operating Scenarios** are explored - e.g. road conditions
- **Hazardous events** are derived - the effect of a hazard in a given scenario
- **Safety requirements & goals** can then be defined

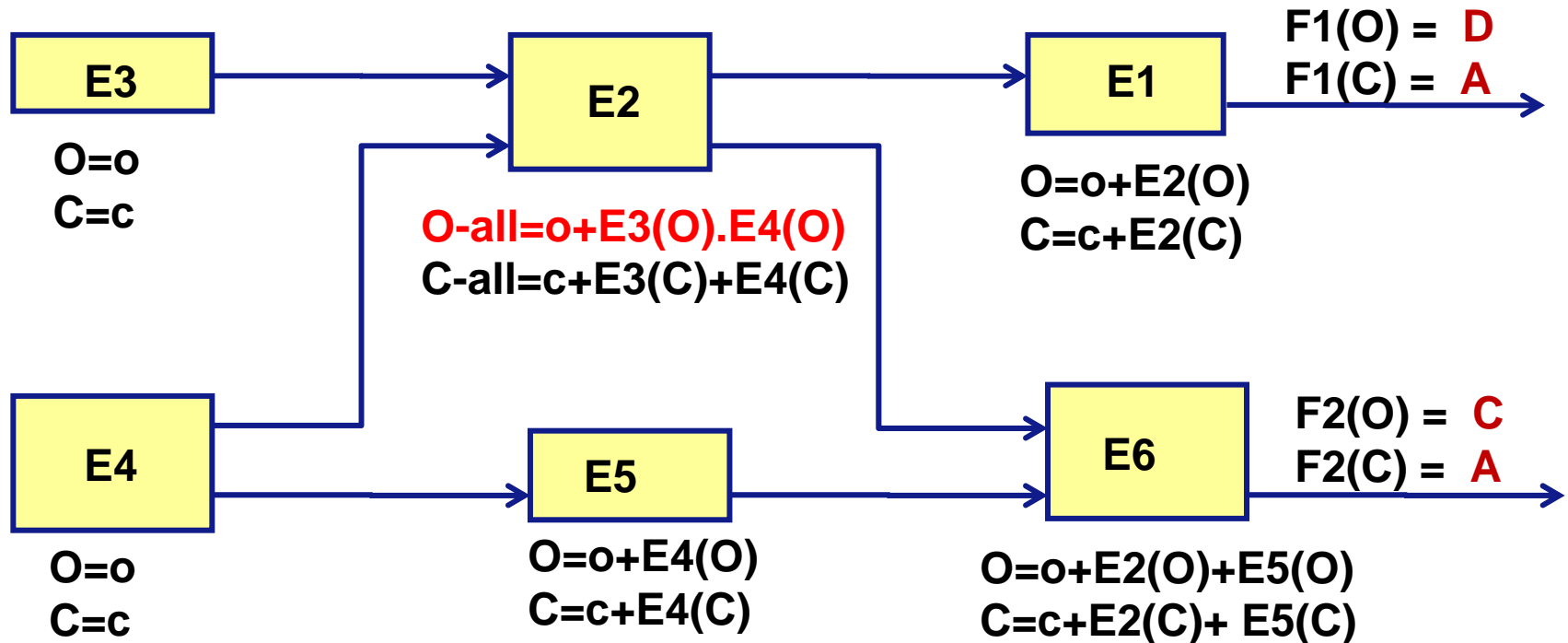
Top-down Allocation of dependability requirements as ASILs:

- ASILs are classification levels indicating safety requirements
- Five levels: **QM, A, B, C, D**
- ASILs are initially **assigned** to system-level safety functions via hazard analysis
- **Allocated** to elements of the system architecture.
- **Decomposed**: architectural elements assigned lower ASILs that combined can fulfil the ASIL of their parent function, as in fault tolerant architectures.

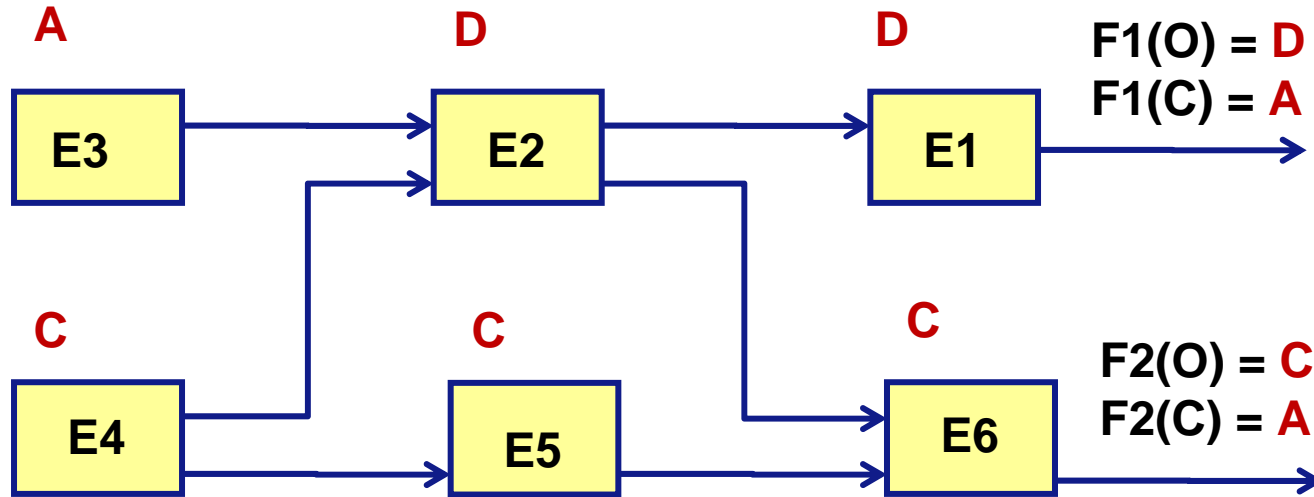
Motivation of our work on ASILS

- **ASIL allocation is a manual and unaided process.**
- **It can be automated**, making it possible to find optimal ASIL allocations in situations where multiple safety functions are delivered over complex networked architectures.
- This cannot be easily achieved in the context of a manual process.
- **In ATESST2 we developed an algorithm** that exploits EAST-ADL modelling and the fault tree synthesis capabilities of HiP-HOPS to automate ASIL allocation

ASIL allocation: Error Modelling



ASIL Allocation: Results

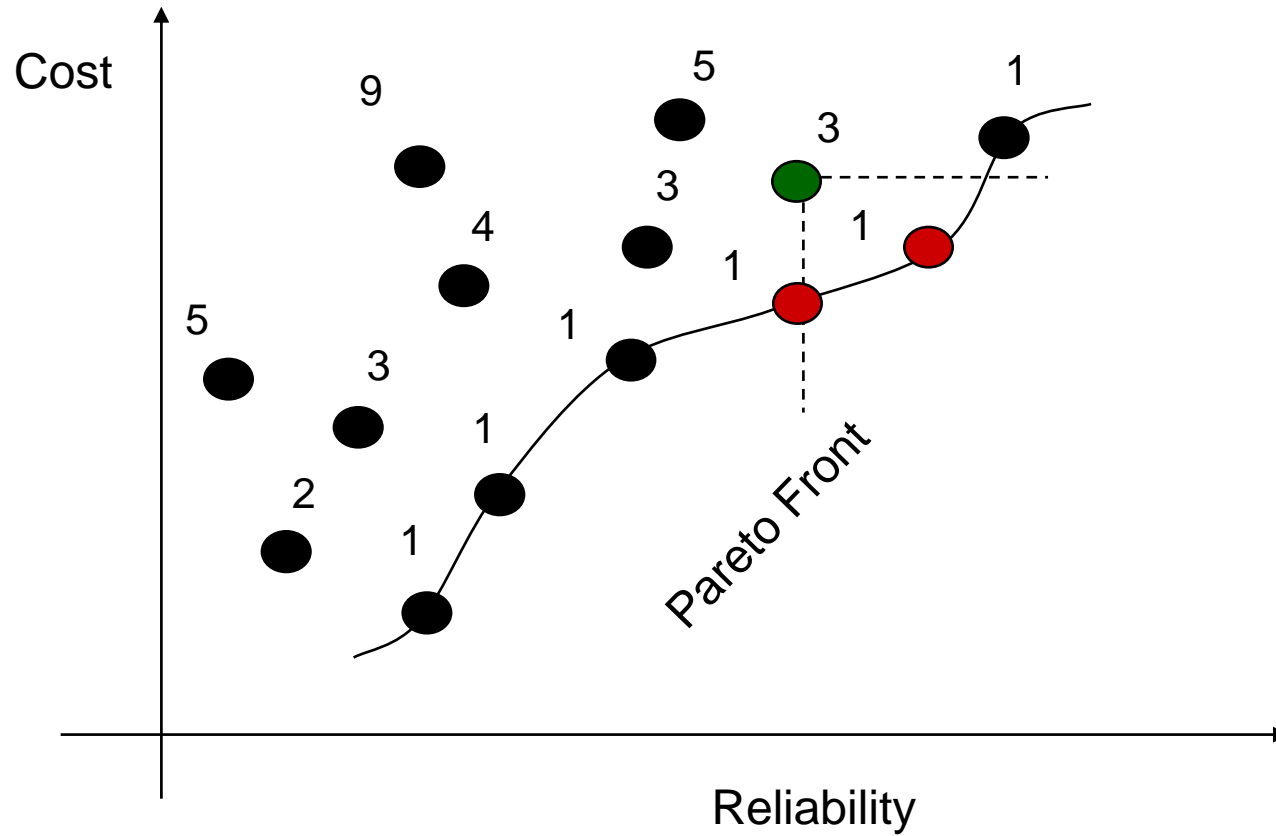


- More economical allocation
- Process is automated via HIP-HOPS
- ASILs can be allocated to components, failure modes of components, or errors at inputs/outputs
- Process can be recursively applied

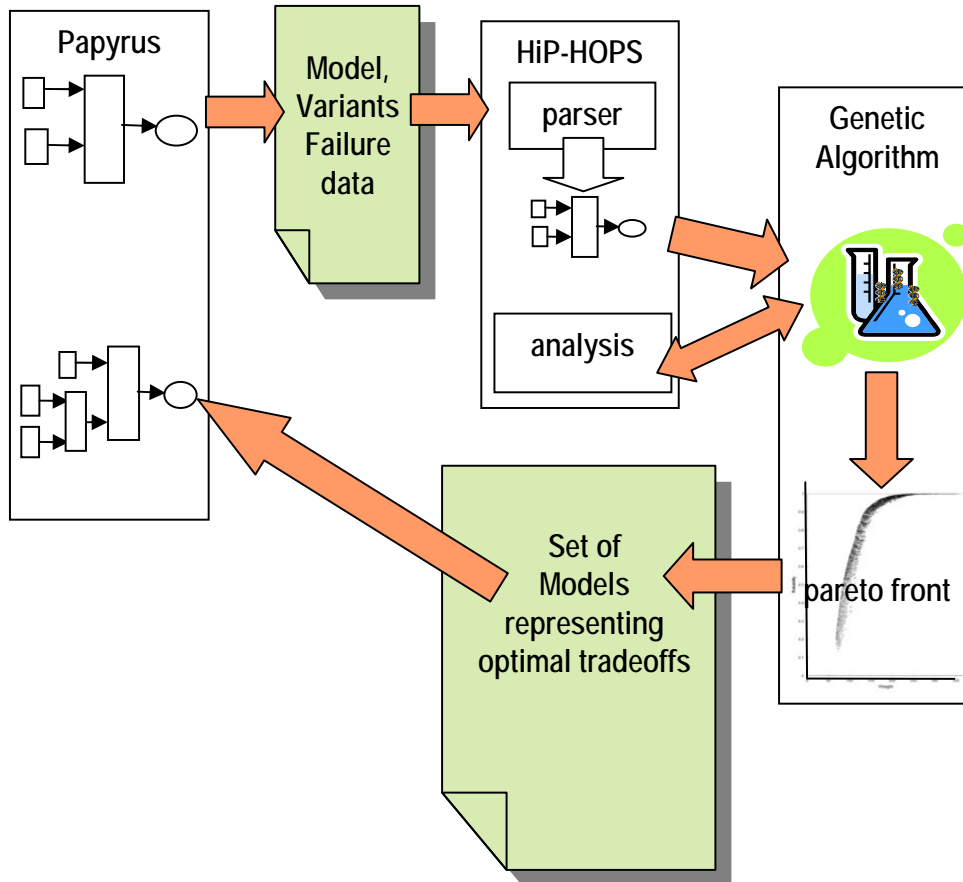
Multi-Objective Design Optimisation

- What if dependability requirements cannot be met?
- How can system dependability be improved?
Substitution of components & sub-systems, more maintenance, replication
- Which solution achieves minimal cost?
Only a few options typically evaluated
This leads to unnecessary design iterations and sub-optimal solutions
- **This is a hard multi-objective optimisation problem that can only be addressed effectively with automation**
Objectives may conflict, e.g. cost vs dependability

Pareto Fronts

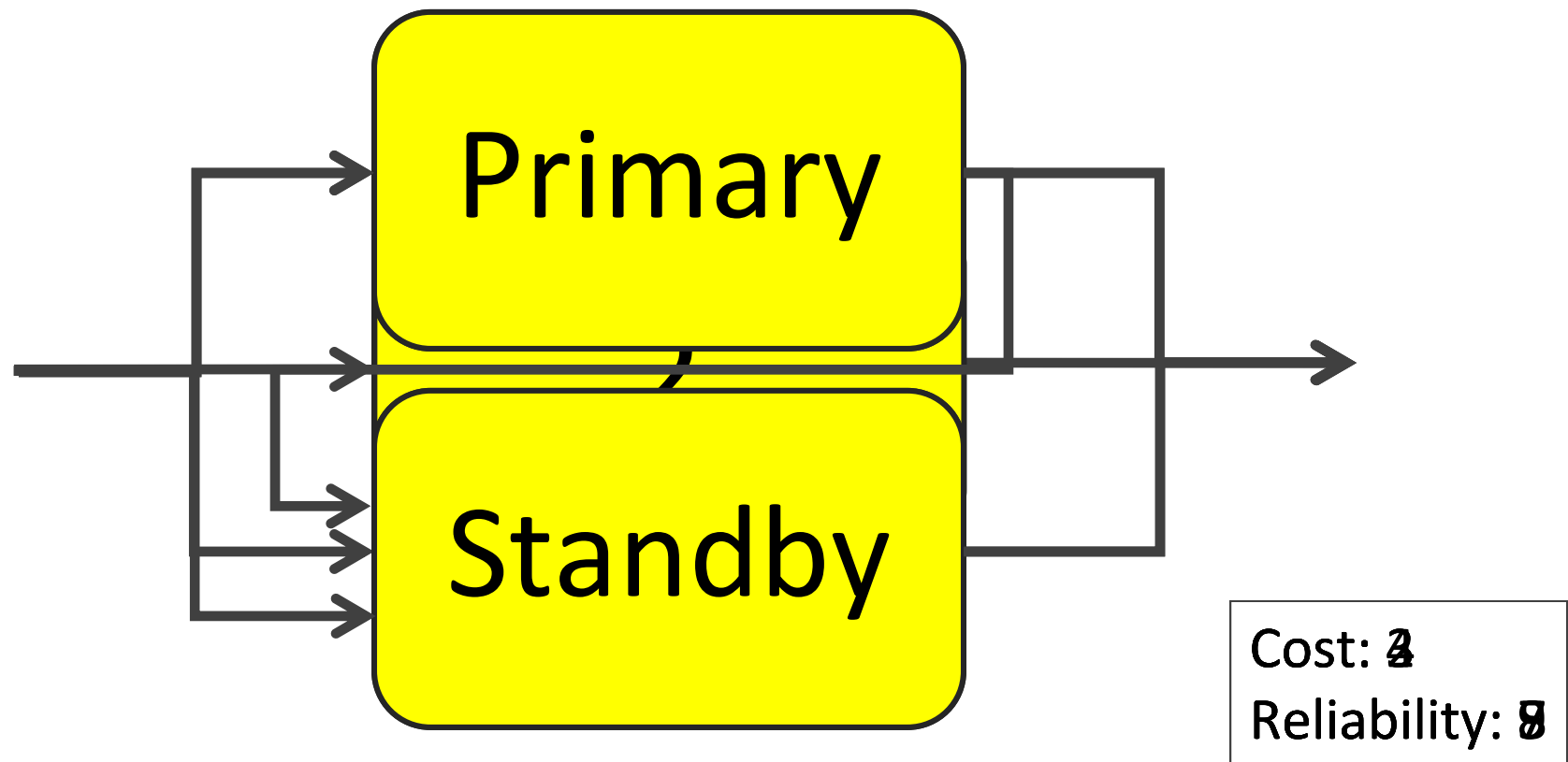


Optimisation Process



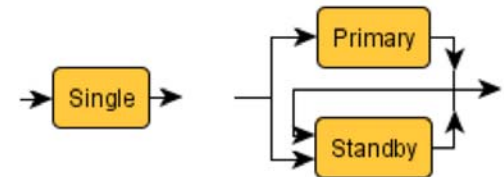
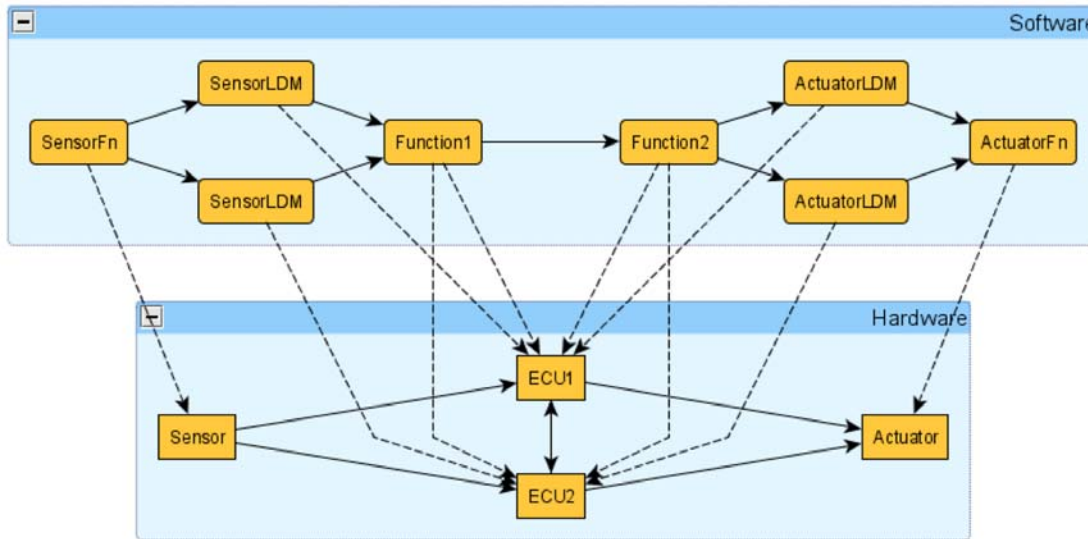
- Designer provides a system design in EAST-ADL, which can contain variants
- Error models and costs of variants are given.
- Objectives of optimisation are defined
- HiP-HOPS performs automatic optimisation via selection and application of variants in the architecture and returns a set of “Pareto Optimal Designs”

Design variations in action



Example System

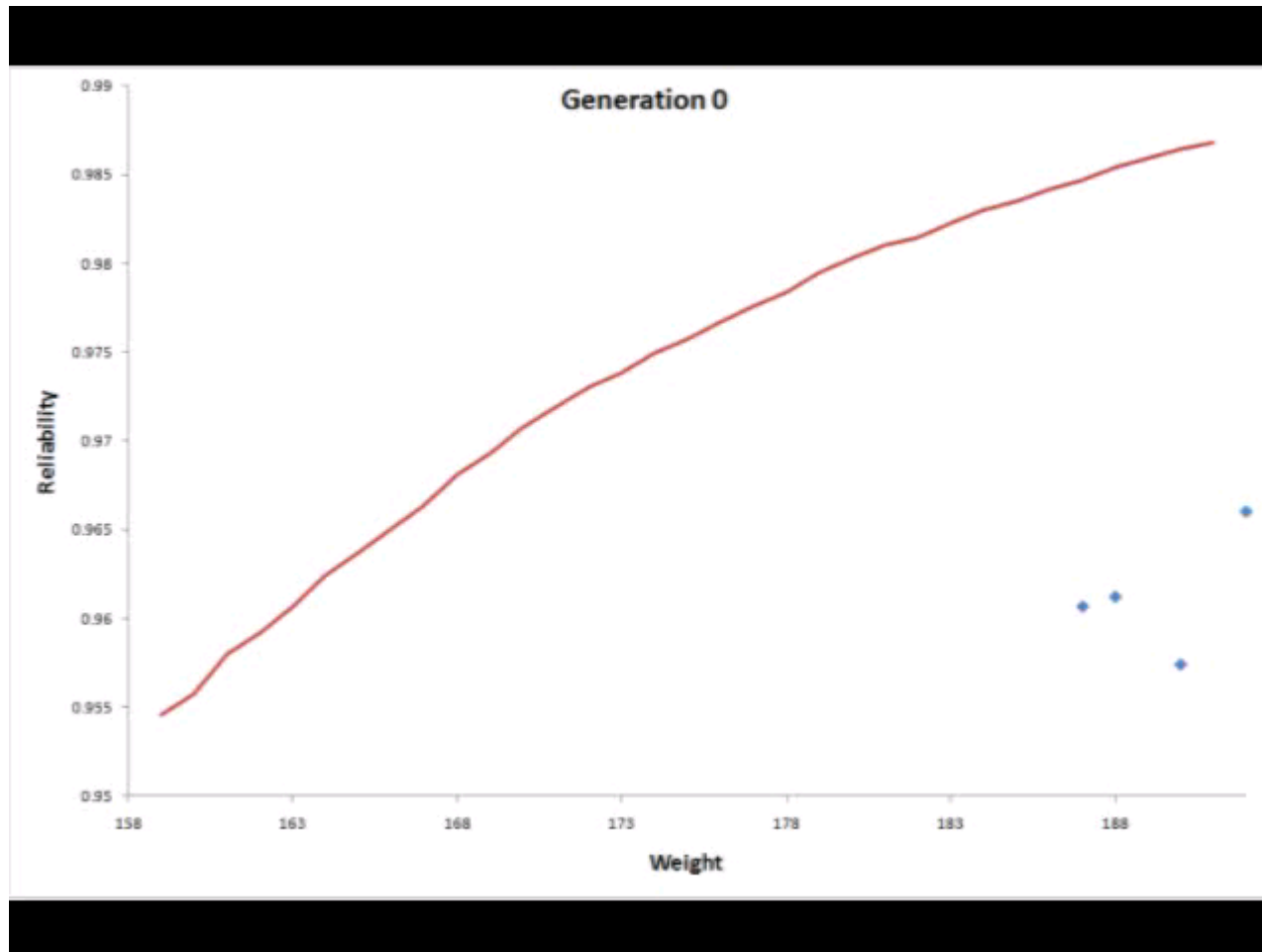
- A single end-to-end flow scenario from Sensor to Actuator
- Variants for hardware components, multiple ECU types (slow, fast), multiple implementations of software functions, possible allocation schemes



Example System

- Objectives
 - Cost: simple sum of the costs of components
 - Unavailability: calculated by HiP-HOPS
 - System slack: calculated from the results of timing analyses in MAST
- Model-transformations
 - Substitution among component variants
 - Substitution among alternative sub-architectures
 - Reallocation of SW to HW
- 28 Pareto optimal solutions, each giving a different choice of variants and allocation scheme
- Designer can view these solutions and choose most suitable

Optimisation in Action



Summary of Work on Dependability

- Support for error modelling and ISO-26262 in EAST-ADL
- Extension of HiP-HOPS to analyse EAST-ADL models:
 - Top-down allocation of safety requirements as ASILs
 - Bottom-up dependability analysis
 - Architecture optimisation
- Some functionality mature (e.g. dependability analysis – HiP-HOPS launched commercially)
- Work continued in MAENAD – forthcoming FP7 project