Modelling Interactive Real-time Applications on Service Oriented Infrastructure

IRMOS Workshop, Dec 09

Zlatko Zlatev

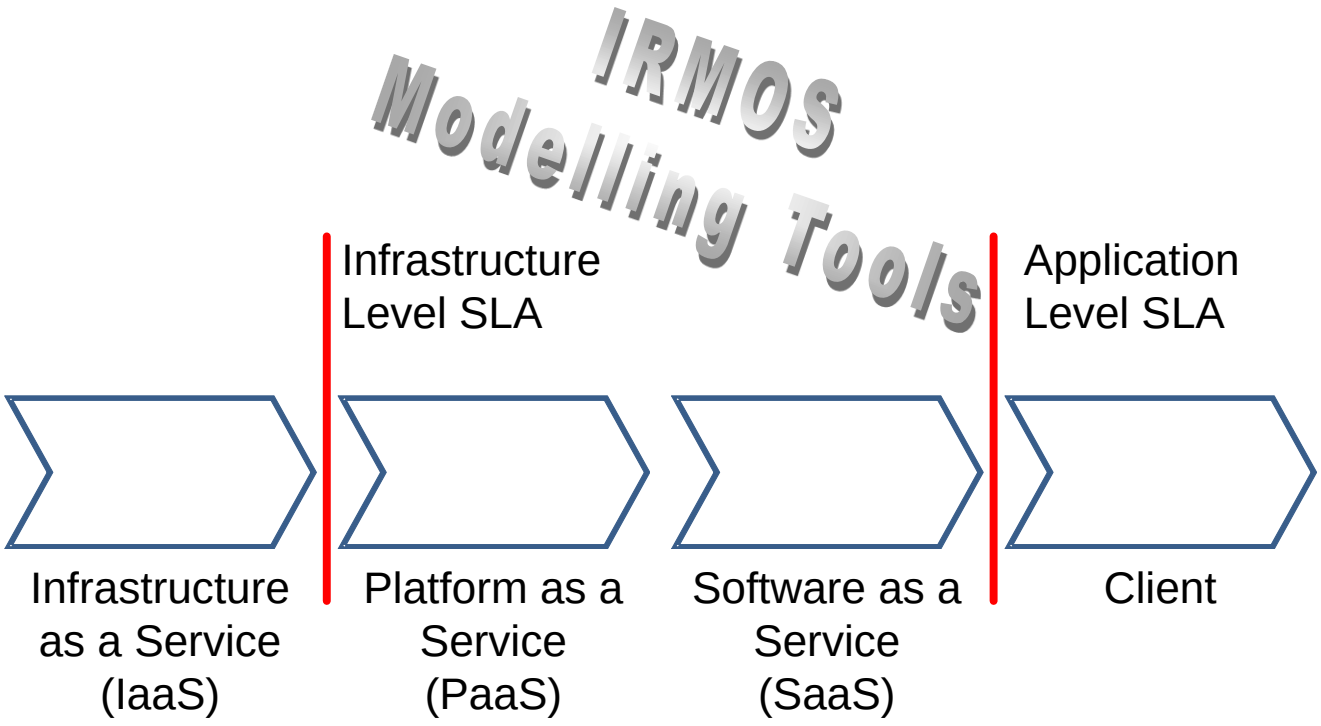
zdz@it-innovation.soton.ac.uk

Content

- Target domain and objectives.
  - Users.
  - Requirements.
- High level Interactive Real-Time application modelling method introduction.
- Application to a simple use case.
  - Post production scenario.
- Performance estimation framework.
- Work in progress and future work.

IRMOS Value Chain

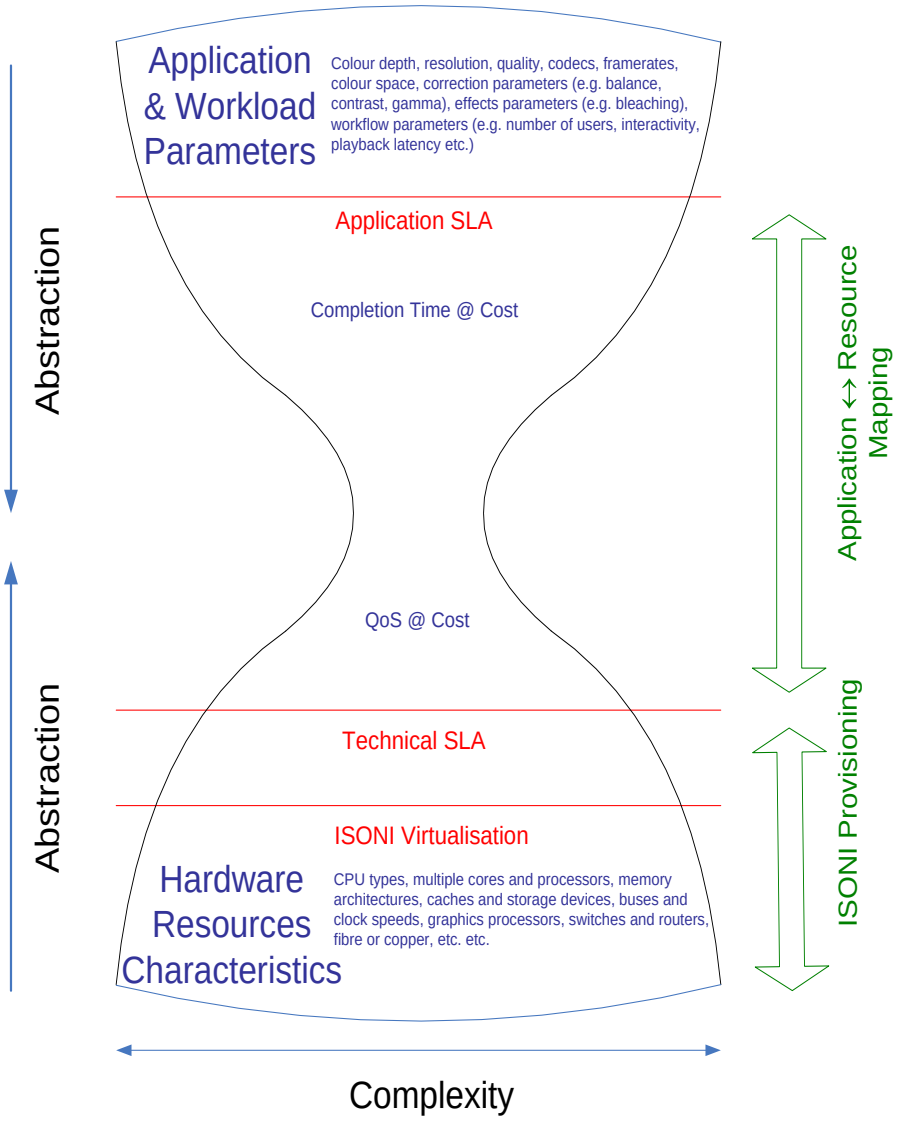
- *IRMOS aims to address the needs of* ***market players providing:***
  - ***infrastructure platforms and***
  - ***software application services****.*
- *Non-modelling experts need tools to model interactive real-time applications.*



Objectives and IRMOS approach

- The objective is to map:
  - hardware resources characteristics
  - to application and workload parameters
  - depending on user requirements for Time & Cost.

- *IRMOS Approach:*
  - ***Virtualisation*** *of resources,*
  - *Performance* ***Benchmarking Suites*** *for each target domain,*
  - ***Performance estimation tools****, based on:*
    - ***Application Benchmarking****,*
    - *User-Application* ***interaction modelling****,*
    - *Resources* ***faults and performance degradation modelling****.*



Performance Estimation Objectives (2) & Challenges

- Estimate **completion time** (Tc) **distribution** using an **application** running on a **selected resource**.

- From the **Tc distribution** (PDF or CDF) can estimate:
  - Probability of **fulfilling a deadline**,
  - **Time to reserve** the infrastructure resource for, …

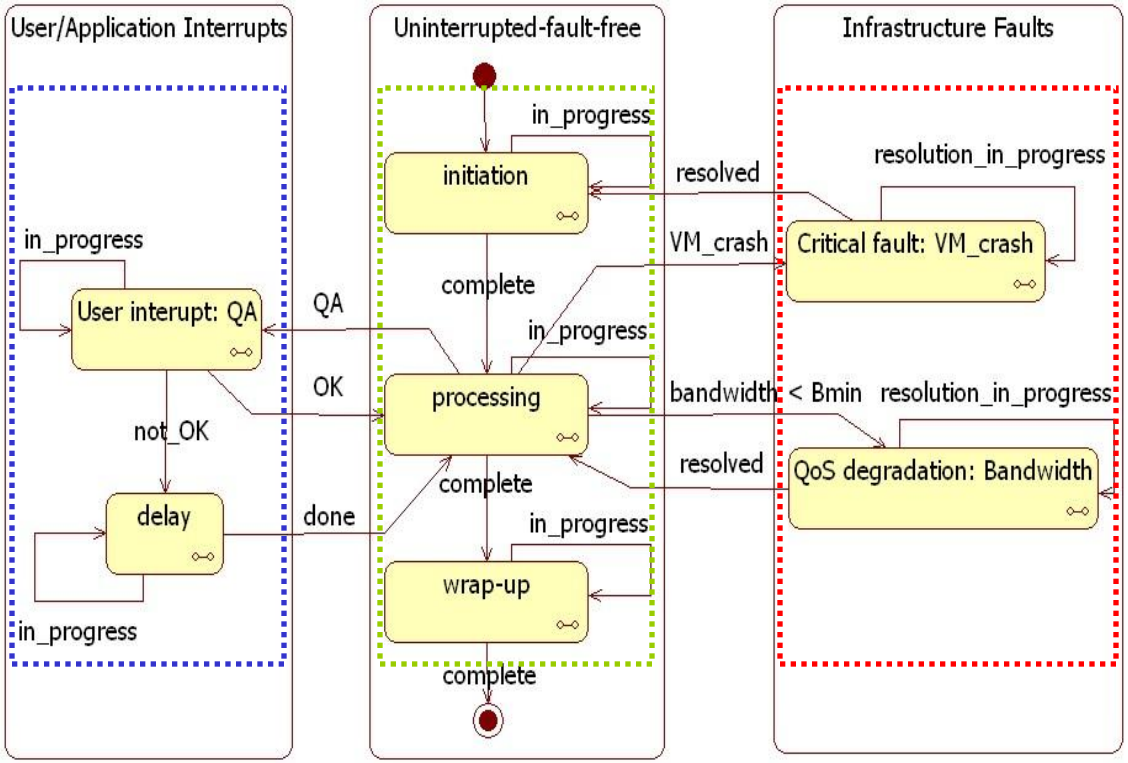
- Modelled applications are **interactive**: Start, Pause, Rewind …
  - Requires **account for interactions** that cause an **interrupt and a delay** in the normal execution of the ASC.

- Modelled applications are **real-time**:
  - We need to account for infrastructure **QoS variations** that cause **application failure events** (e.g. bandwidth below X b/s result in streaming failure).

- There are **critical infrastructure failures**, e.g. VM crash, which may come in play for **long running activities** (rendering may take days).

Application performance model template

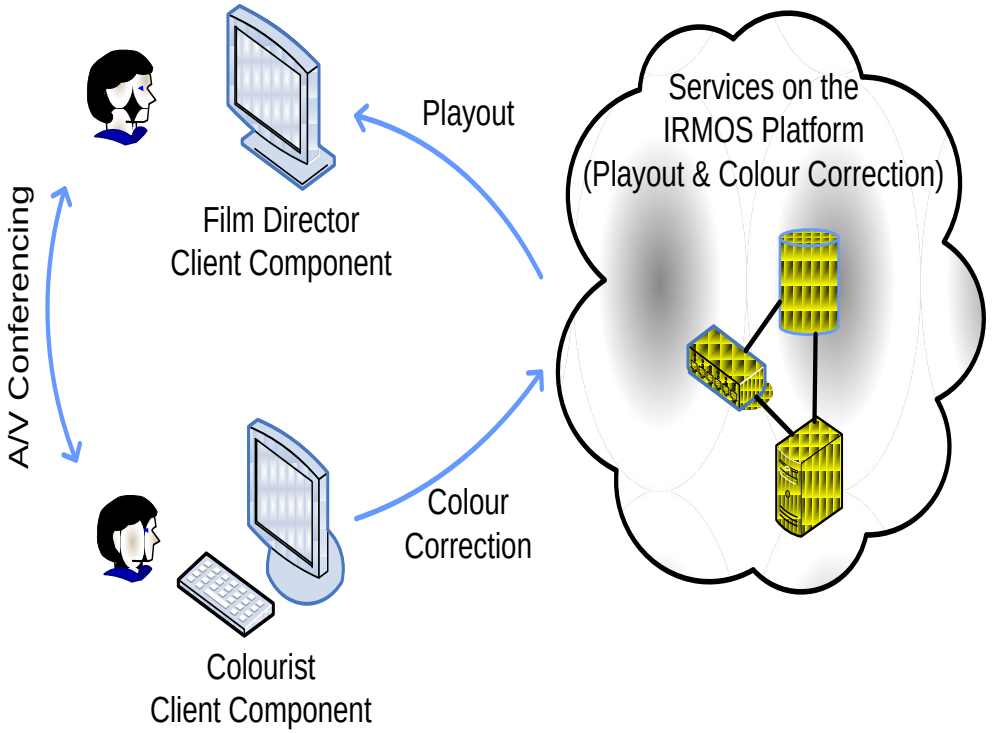
- Approach based on **Finite State Machine** (FSM) formalism:
  - a **statistical** high level performance model of the application
  - models **behaviour**, **not implementation**,
  - produces an estimate of the activity **completion time**.
- Devised a **generic FSM template** that **combines**:
  - **Uninterrupted-fault-free** operation,
  - **User-Application interaction**,
  - **Resources faults** (critical failures and QoS degradation).
- Advantages:
  - **Externally observable states**, no application internals knowledge needed.



- State **transition probabilities** estimates can be obtained **by experience**.

Mock-up Colour Correction Scenario

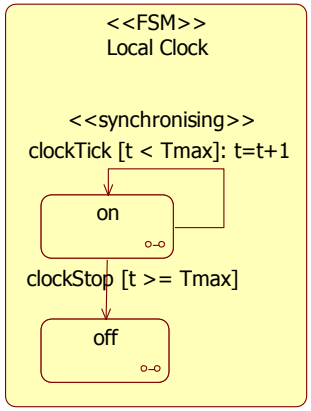
- Colour correction of digital daily film shooting:
  1. Footage streamed a **number of times** (e.g. 1, 2 or 3) to the film director for setting out the manner of colour correction.
  2. If playout link **bandwidth drops below a critical value streaming stops**. It takes some **delay** to go back to the beginning of the current scene and recommence reviewing.
  3. Colourist starts colour correction and output is streamed to the director as soon as generated.
  4. Director can ask over a conferencing link for **pausing and colour regrading** of some frames or a scene.

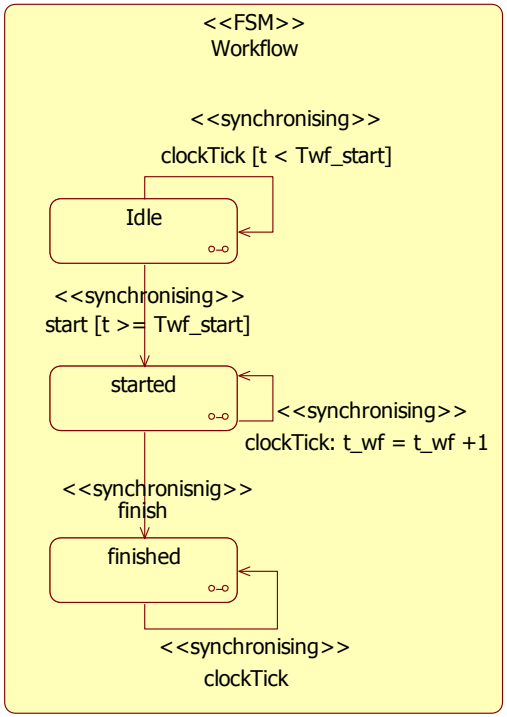


Scenario FSM model (1)

1. Create a **trivial Local Clock** on/off FSM
   - Counts the elapsed FSM execution time

2. Create a **trivial Workflow** idle/started/finished FSM
   - It is not a controller
   - It starts the first activity
   - It finishes when the last activity completes
   - The individual activities interact with each other

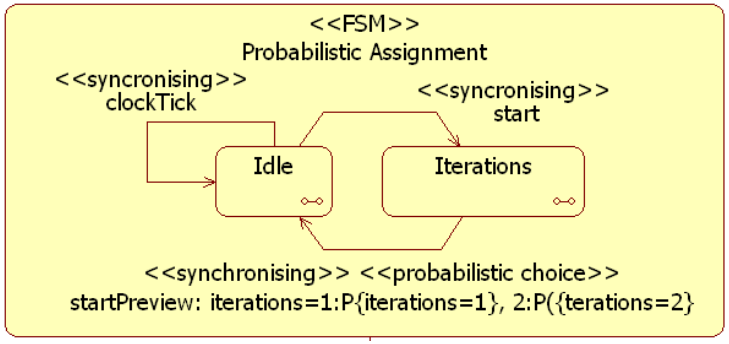




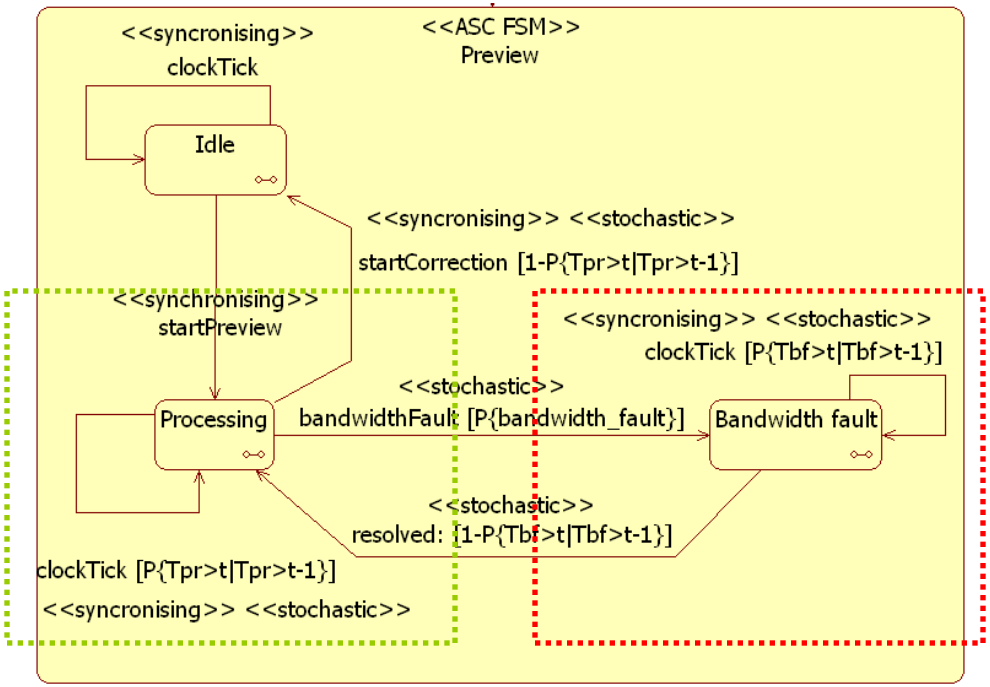
Scenario FSM model (2)

Activity 1: Footage Preview

3. Create a **trivial Probabilistic Assignment** FSM:
   - Set the footage preview iterations to 1, 2 or 3 with given probabilities.



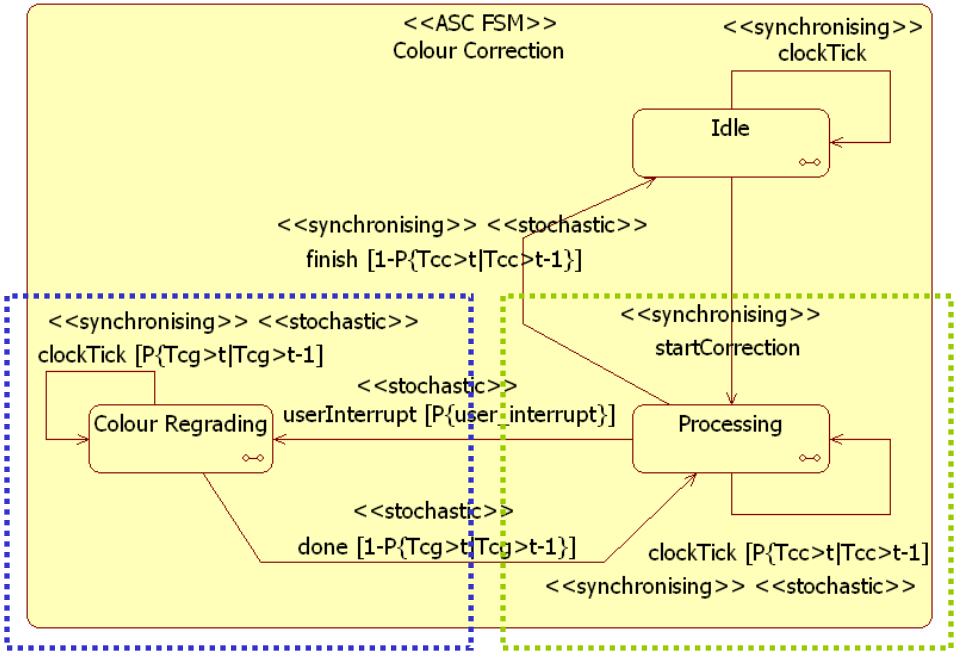
4. Create a Preview FSM **based on the generic FSM template**. States used:
   - Uninterrupted-Fault-Free Processing, i.e. Previewing the footage without playout link faults.
   - Infrastructure Bandwidth fault.
   - A **trivial Idle state**.



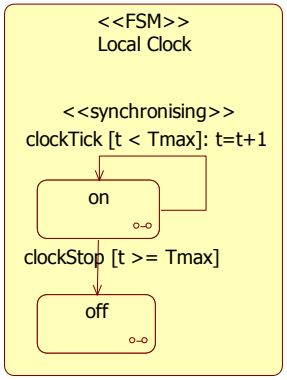
Scenario FSM model (3)

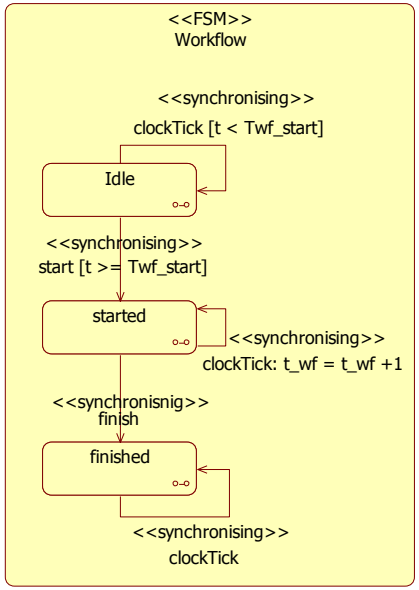
Activity 2: Colour Correction

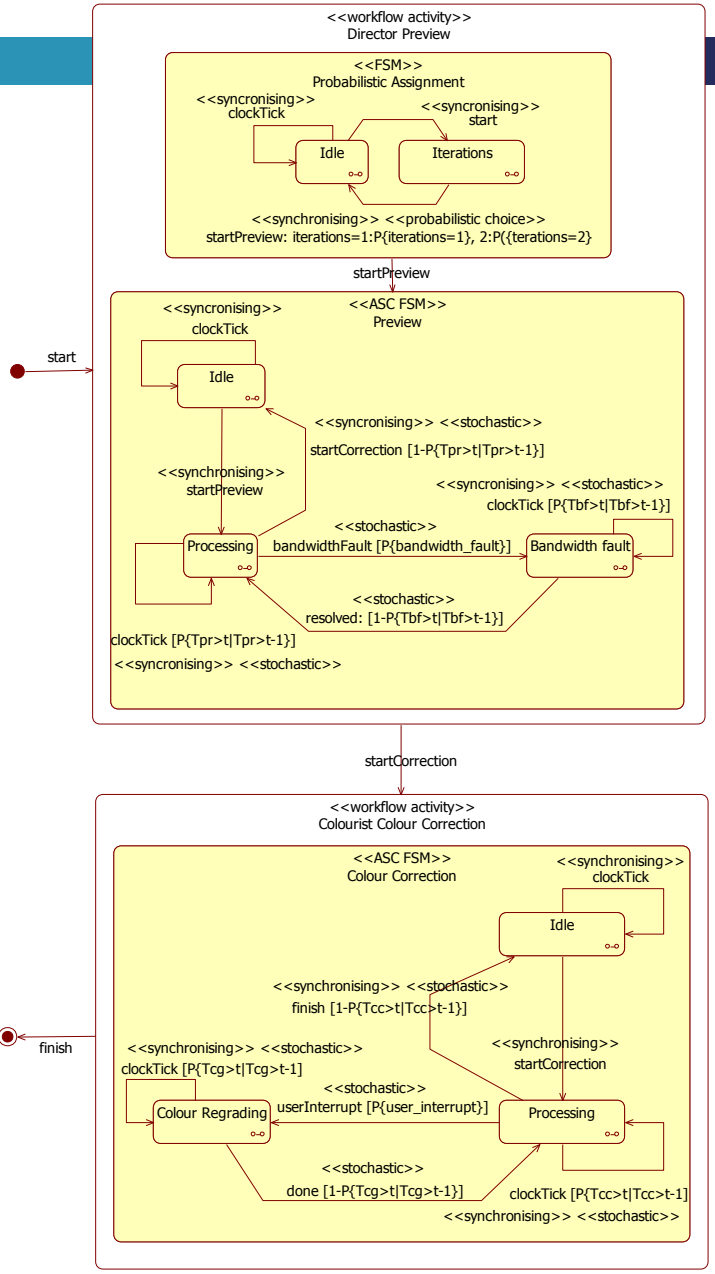
5. Create a Colour Correction FSM **based on the generic FSM template**. States used:
   - Uninterrupted-Fault-Free Processing, i.e. footage colour grading without interruption by the directir for regrading
   - User interrupt for Colour Regrading.
   - A **trivial Idle state**.



Full FSM UML model



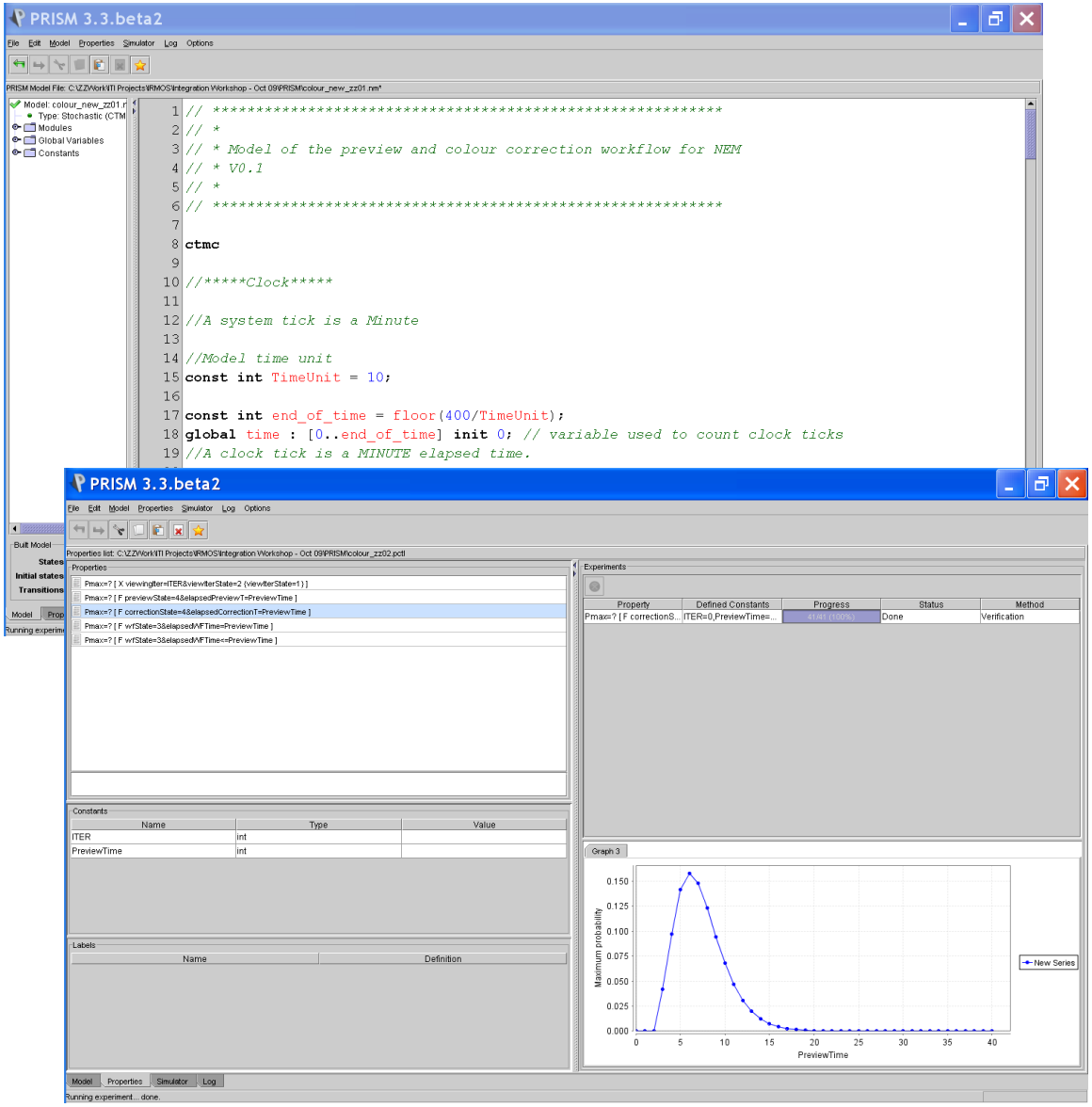




- UML state **transitions stereotypes**:
  - **<<synchronising>>** - must be triggered together with another transition with the same name
  - **<<stochastic>>** - is triggered according to some probability
  - **<<probabilistic choice>>** - after triggering a probabilistic assignment occurs
- Transitions notation: ***name [guard]: outcome***

FSM properties calculation

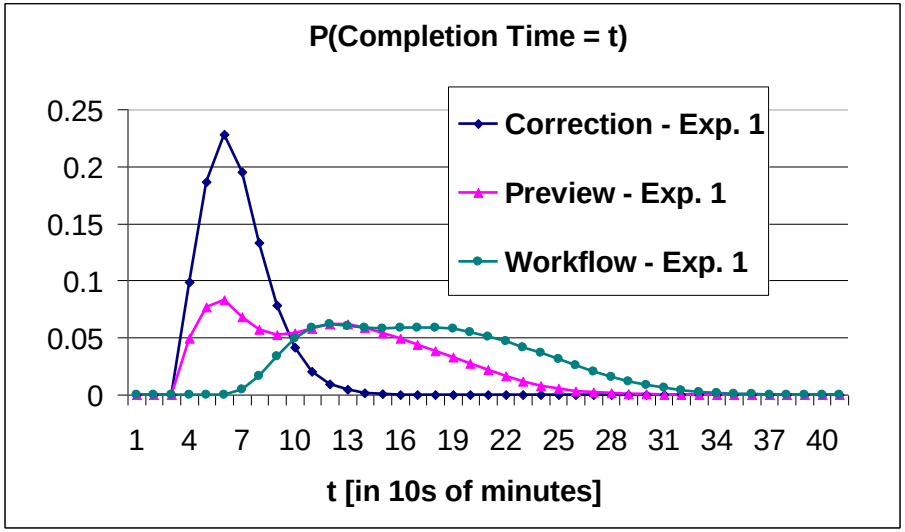
- FSM UML model manually translated to the **PRISM modelling language**.
- Model **properties**, e.g. probability of workflow completion by given time, **calculated by PRISM numerically or by Monte-Carlo simulations.**



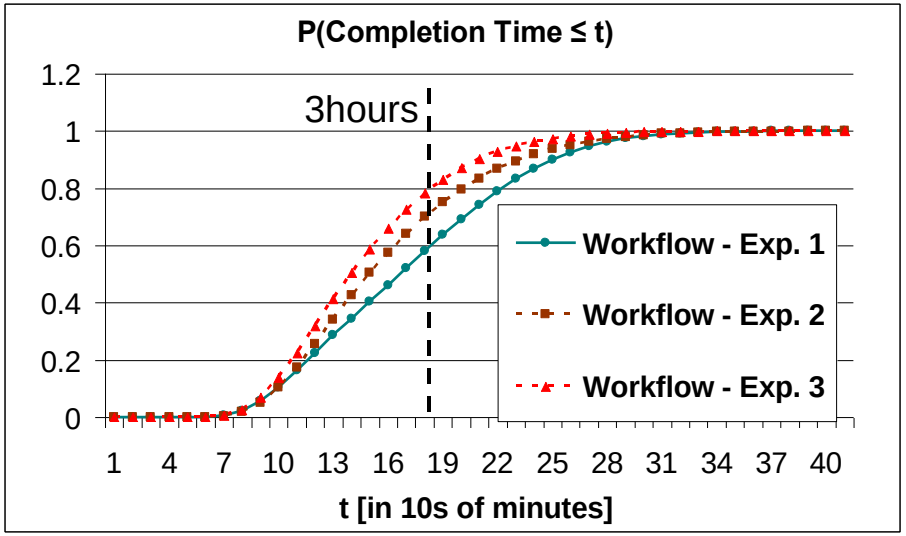
Colour Correction Modelling Experiments

- The model responds to **variation in user behaviour** and **variations in resource behaviour**.

Experiment	Footage Length [min.]	Preview Iterations Probability	Streaming Fault Rate [per hour]	Delay after Streaming Fault [min.]	Correction Interrupt Rate [per hour]	Delay after Correction Interrupt [min.]
1	30 to 50	P(1)=1/3, P(2)=1/3, P(3)=1/3	2	10	2	10
2	30 to 50	P(1)=0.6, P(2)=0.3, P(3)=0.1	2	10	3	10
3	30 to 50	P(1)=0.6, P(2)=0.3, P(3)=0.1	1	10	3	10



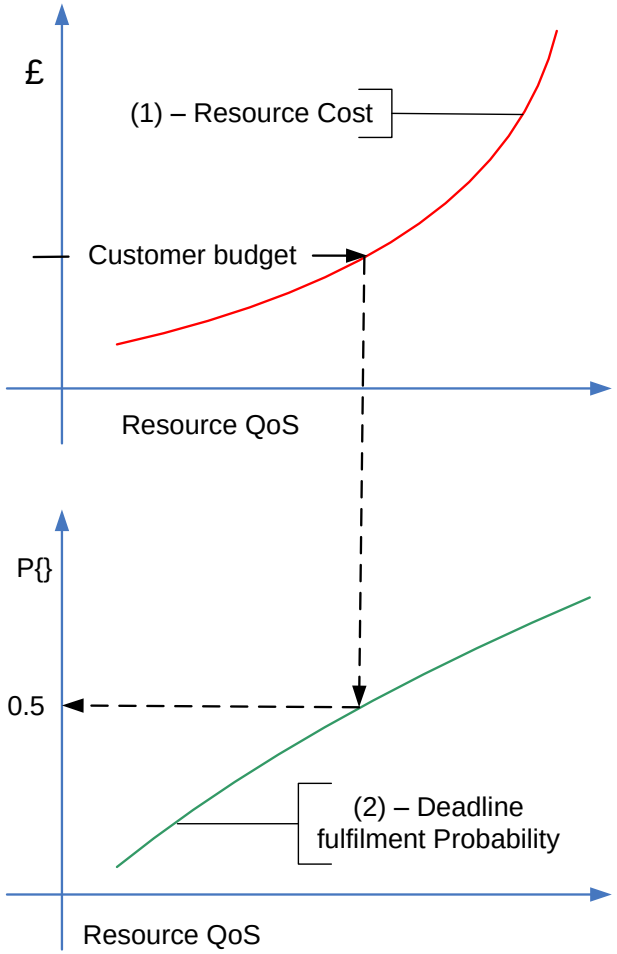
- Workflow and individual stages PDFs.



- Workflow CDFs for all experiments.

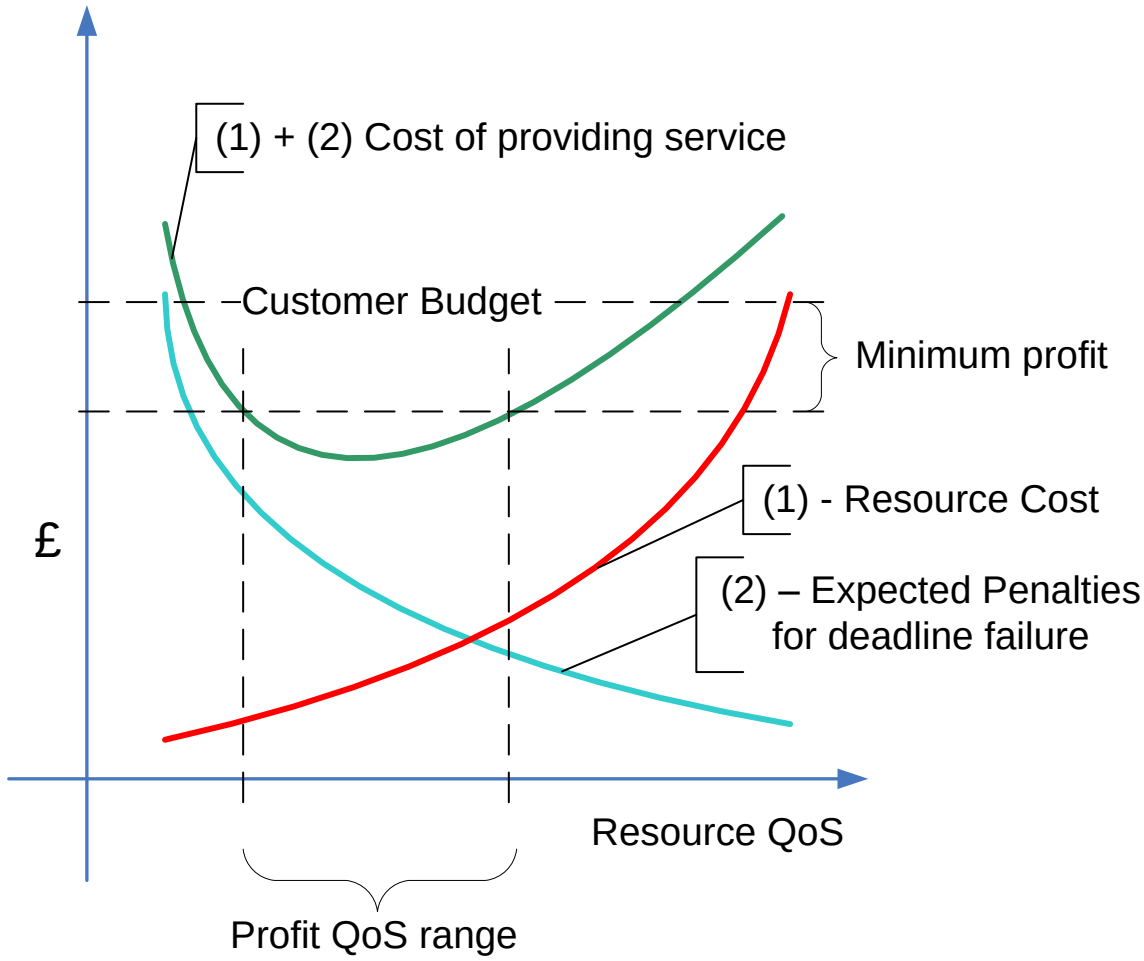
Simplistic Business Model

- Given:
  - **resource cost** for different QoS available (1),
  - customer **budget**,
  - workflow completion time **deadline** required.
- Fund:
  - the **probability of fulfilling the deadline** (2).
- **Reverse reasoning** also possible:
  - the application provider has **set QoE targets** (2).
  - wants to know **resource QoS and cost** (1).



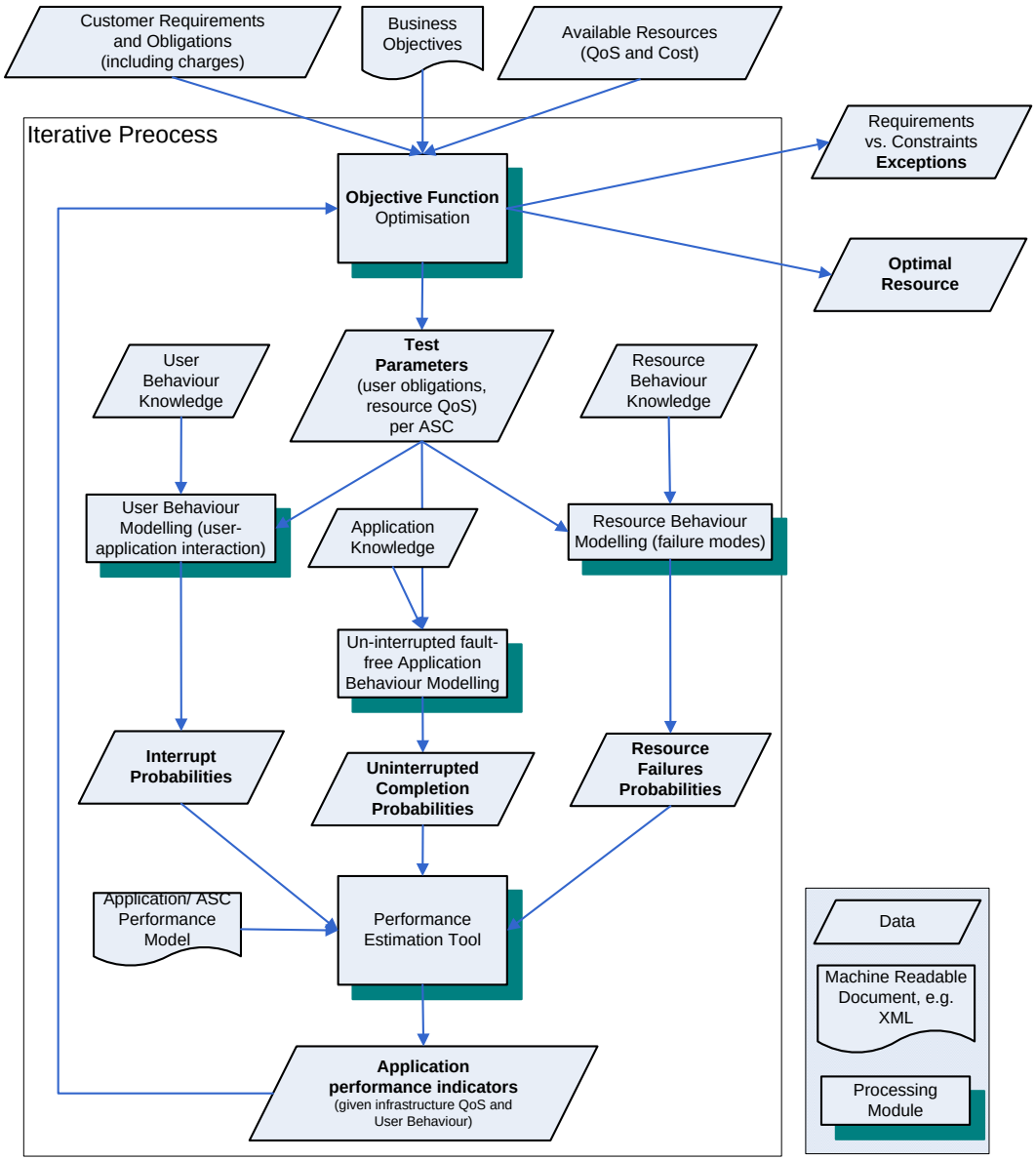
More complicated Business Model

- The application provider gives guarantees:
  - Penalties for failed deadline.
- The application provider has:
  - Set min profit margin,
  - Operational costs.

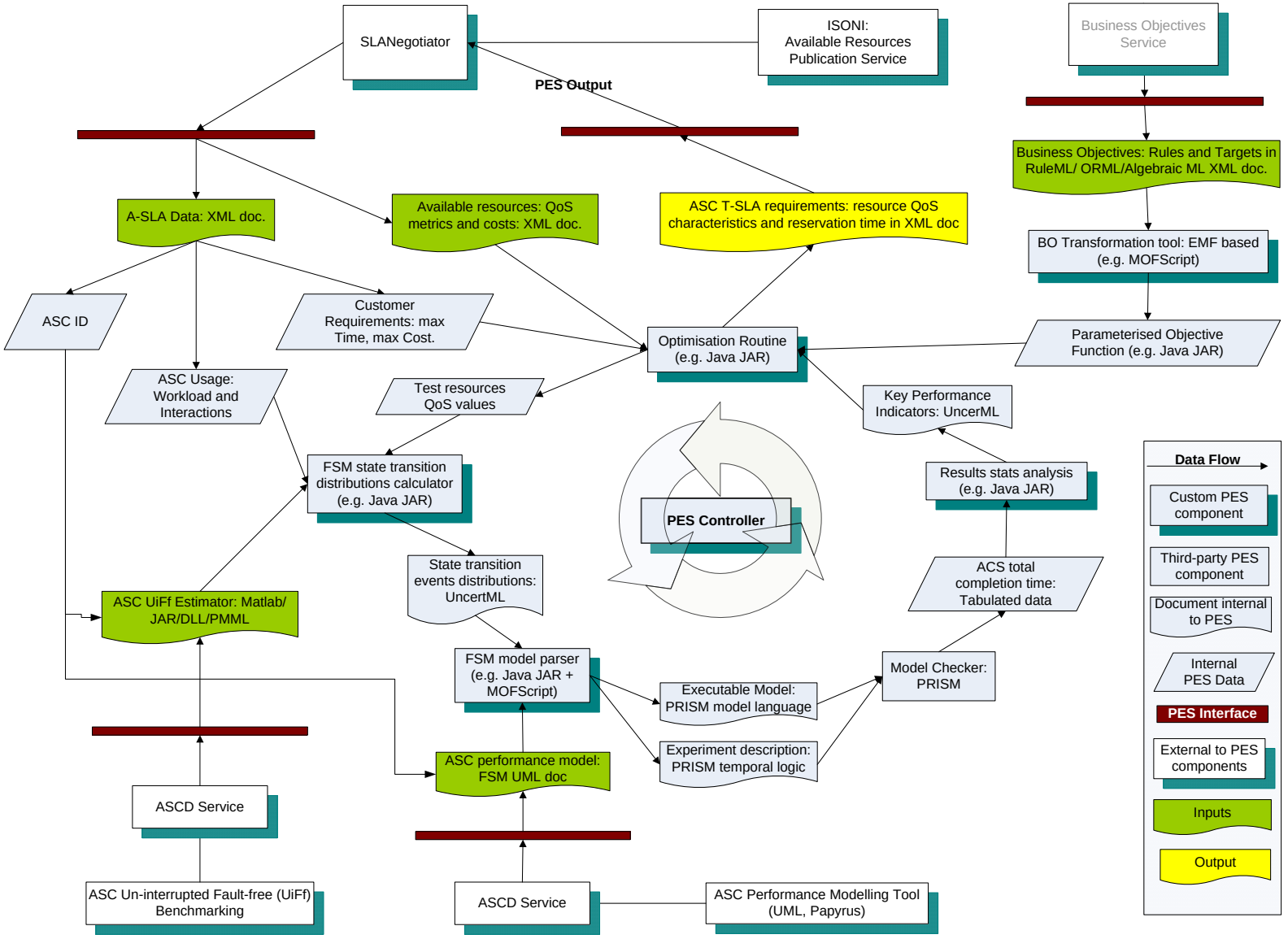


Abstract Service Engineering Process

- In the context of Business Objectives,
- for some Customer Requirements (e.g. average application execution time) and Obligations (e.g. workload average size)
- choose optimal resource allocation from a pool of resources,
- using Background User Behaviour (interrupts), Resource Behaviour (failures) and Normal Application (failures) and Normal Application (benchmarked),
- through optimisation.



Performance Modelling Framework: The IRMOS PES service



Work in progress and future work

- Tool for graphical FSM creation:
  - Specifying and creating Papyrus profiles for FSM
  - Using MOFScript for FSM UML parsing and PRISM code generation
  - Wizard for step by step FSM creation.

- Specifying business objectives:
  - Investigate Rule ML, ORML, Algebraic ML
  - Use MOFScript to parse XML business objectives and create an executable objective function.

- Methods for updating knowledge with monitoring data:
  - Bayesian updating with Markov Chain Monte Carlo (BUGS for MCMC).

Thank You!
Any Questions ?

Zlatko Zlatev
zdz@it-innovation.soton.ac.uk