A Method for Rigorous Development of Fault-Tolerant Systems

**student** Ilya Lopatkin
**supervisor** Prof Alexander Romanovsky

**Dependability of critical systems**

- Formal methods to remove faults (bugs)
- Fault tolerance to live through runtime

**Problems**
- Lack of formal experience in industry
- Lack of tools
- Man efforts due to flexible notations

**Solution**
- Development guidance
- Refinement strategy
- Set of modelling principles
- Set of modelling patterns

**Problems**
- Significant amount of FT reqs (40%)
- Cross-cutting FT requirements
- Belated consideration in formal models

**Solution**
- Separate FT viewpoint
- Orthogonal to Event-B model
- Early development
- Additional restrictions on model

**Motivations**
- Normal
  - Door1
  - Door2
- Trapped
  - Door1_closing
  - Door2_closing
  - Trapped_closing_door1
- Degraded
  - Broken

**Fault Tolerance / Mode Views**
- Formally defined
- Modecharts for interleaving semantics
- Readable by engineers and analysts

**Event-B**
- State-based
- Interleaving semantics
- Top-down refinement

**Fault Tolerance / Mode Views**
- Abstract model
- Abstract view
- Model i
- Proof obligations
- View i1
- Model i+1
- View i2
- Detailed model
- Detailed view

**Patterns**
- FT behaviour pattern
- Behavioural split template

**Evaluation**
- Two case studies
  - Sluice gate system
  - Attitude and Orbit Control System
- Qualitative criteria
  - Proof efforts
  - Modelling guidance
  - Early modelling of FT

**Modelling principles**
- Implementable causality
- Reactive systems
- Refinement of functionality
- Error refinement

**Patterns**
- Safestop pattern
- Availability variable
- Availability invariant pattern
- FT behaviour pattern
- Behaviour restriction pattern
- Control cycle pattern

**Pattern examples**