INCOSE International Symposium
Las Vegas, NV
June 30 - July 3, 2014

Traceable Engineering of Fault-Tolerant SoSs

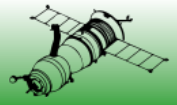
Zoe Andrews; Claire Ingram;
Richard Payne;
Alexander Romanovsky
(Newcastle University, UK)

Jon Holt; Simon Perry
(Atego, UK)

Newcastle University

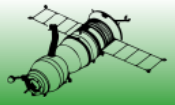
atego™

COMPASS
www.compass-research.eu

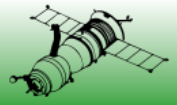

Outline

- Introduction
- Case study
- Traceable engineering of fault-tolerant SoSs
  - Requirements engineering
  - Architectural design
  - Requirements tracing
- Integration into a systems engineering process
- Conclusions and future work


Outline

- **Introduction**
- Case study
- Traceable engineering of fault-tolerant SoSs
  - Requirements engineering
  - Architectural design
  - Requirements tracing
- Integration into a systems engineering process
- Conclusions and future work


Systems of Systems

- Systems of Systems (SoSs):
  - *constituent systems (CSs)* interacting via a networked *infrastructure*
  - *reliances and responsibilities* borne by constituent systems
- Achieve a *global emergent* functionality and performance
- In the face of heterogeneity of ownership, management, stakeholders, evolution, …

Fault Tolerance in SoSs

- SoS engineers must consider faults carefully
  - CSs may withdraw from the SoS arbitrarily
  - independently-owned CSs may evolve without taking into account the needs of the SoS
  - network-based connectivity problems
- CS owners may resist possibly costly recovery processes for faults introduced by third parties

Need quality *methods* and *tools* for reasoning about faults in SoSs at the architectural level

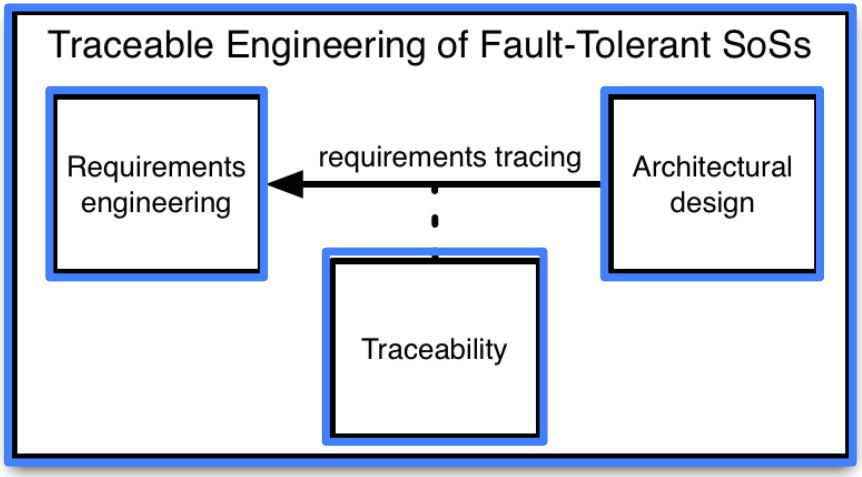
Fault Tolerance in SoSs

- Dependability definitions [ALRL2004] specialised to SoSs:
  - **SoS failure**: "*a deviation of the service provided by the SoS from expected (correct) behaviour*"
  - **SoS error**: "*the part of the SoS state that can lead to its subsequent service failure*"
  - **SoS fault**: "*the adjudged or hypothesised cause of an error*"
- The focus of our work is on **fault-tolerant SoSs**
  - preventing SoS failures from arising in the presence of faults
  - achieved by detecting errors and conducting system recovery

[ALRL2004]: Avizienis, A.; Laprie, J.-C.; Randell, B. & Landwehr, C., "Basic concepts and taxonomy of dependable and secure computing," *Dependable and Secure Computing, IEEE Transactions on* , vol.1, no.1, pp.11,33, Jan.-March 2004.

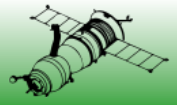
The Contribution

- Contribution: traceable engineering of fault-tolerant SoSs
  - structured approach to **capturing requirements** of fault-tolerant SoSs
  - **architectural framework** supporting disciplined and reusable development of fault-tolerant architectures
  - **traceable mapping** of fault tolerance requirements into SoS architectural designs

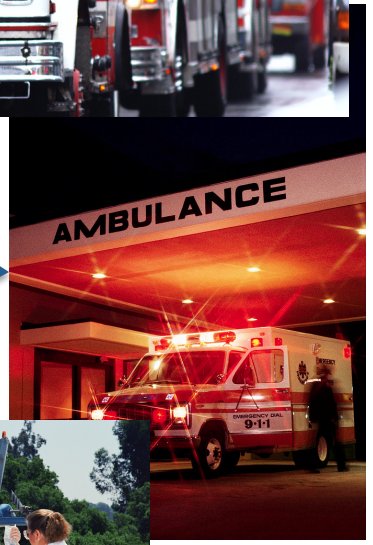


Outline

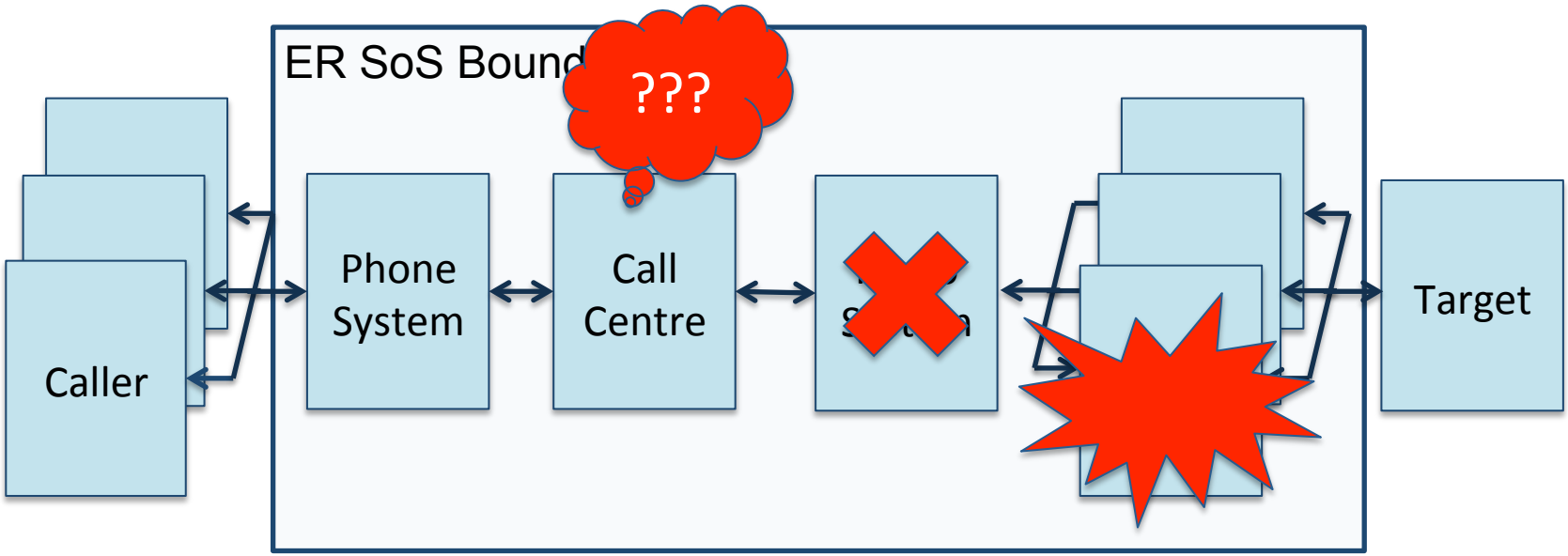
- Introduction
- **Case study**
- Traceable engineering of fault-tolerant SoSs
  - Requirements engineering
  - Architectural design
  - Requirements tracing
- Integration into a systems engineering process
- Conclusions and future work


Emergency Response Case Study



Emergency Response Case Study

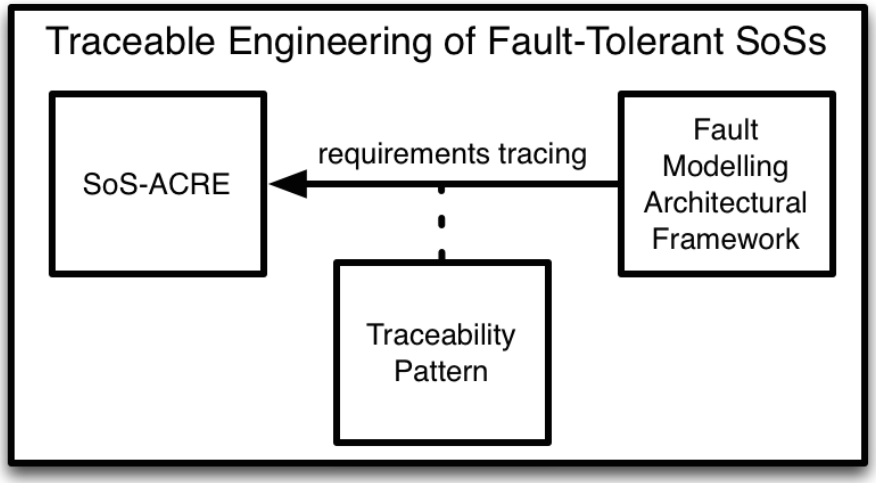


High-level requirement:

- For every call received, send an ERU with correct equipment to correct target

Case Study: Approach

- Fault tolerance requirements engineering using the SoS Approach to Context-based Requirements Engineering (SoS-ACRE)
- Architectural fault modelling using Fault Modelling Architectural Framework (FMAF)
- Establish traceability links between requirements and FMAF elements using Traceability Pattern



Emergency Response Case Study

COMPASS: Model-based SoS Engineering

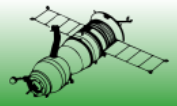
Providing and evaluating advanced model-based methods and tools for development and analysis of SoS.

Key outputs:

- **Guidelines & patterns** for SoS Requirements, Architectures and Integration
- **A modelling language (CML)** with formal semantics, developed specifically for SoS Engineering problems
- **An open tools platform** providing computer-assisted analysis of global properties, and test generation and management
- **Industry evaluation** of methods and tools based on case studies.

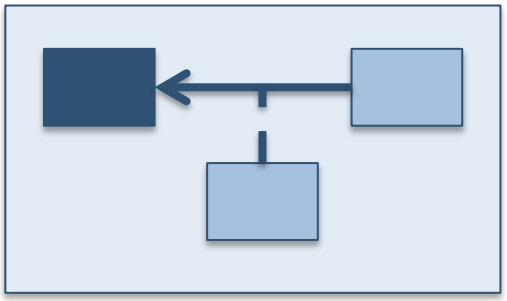
Outline

- Introduction
- Case study
- **Traceable engineering of fault-tolerant SoSs**
  - **Requirements engineering**
  - Architectural design
  - Requirements tracing
- Integration into a systems engineering process
- Conclusions and future work


Requirements Engineering

- Apply SoS-ACRE: a structured way of engineering and managing the requirements of an SoS



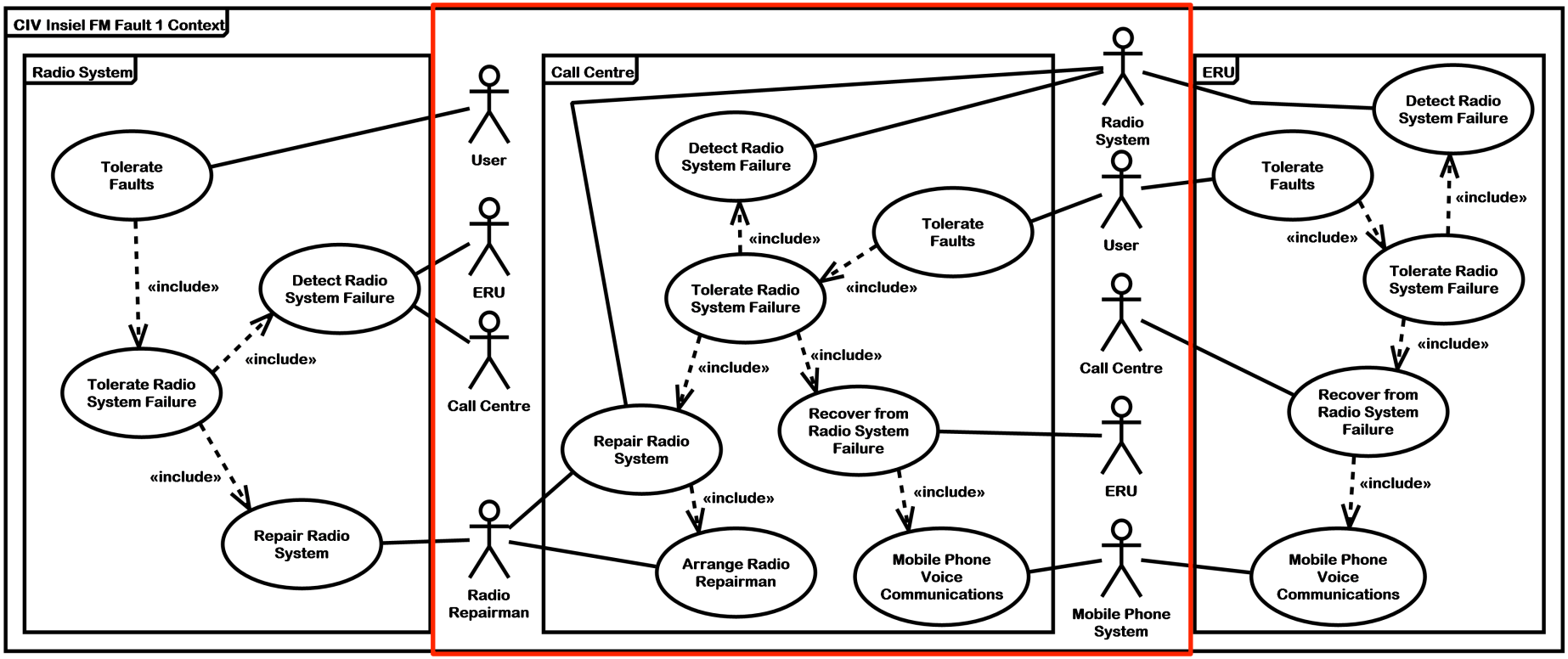
We focus on the requirements engineering processes

- Key elements of SoS-ACRE for fault tolerance:
  - define the fault tolerance requirements of the SoS (what faults)
  - examine the fault tolerance requirements in the context of the CSs
  - identify error detection and recovery scenarios

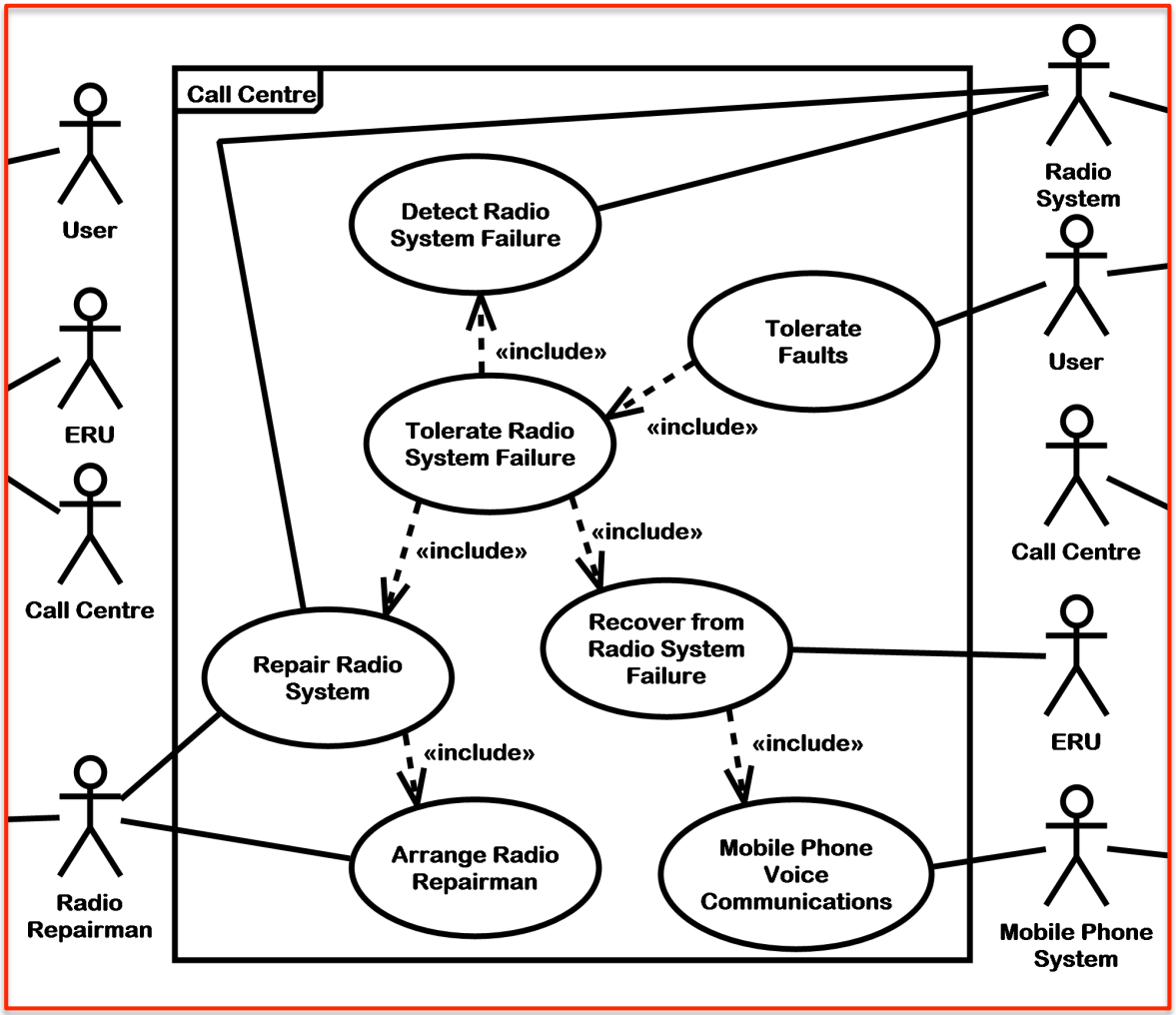
ER SoS: Context Interaction

INCOSE International Symposium
Las Vegas, NV
June 30 - July 3, 2014

Requirement: Tolerate Radio System Failure

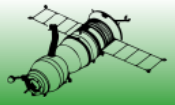


ER SoS: Context Interaction



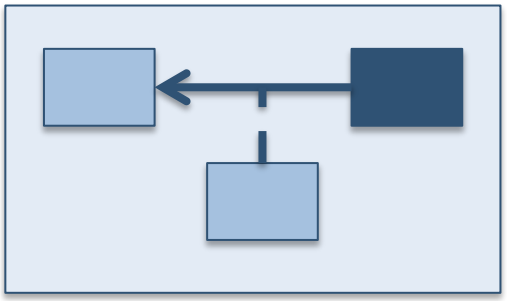
Outline

- Introduction
- Case study
- **Traceable engineering of fault-tolerant SoSs**
  - Requirements engineering
  - **Architectural design**
  - Requirements tracing
- Integration into a systems engineering process
- Conclusions and future work

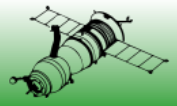




Architectural Design

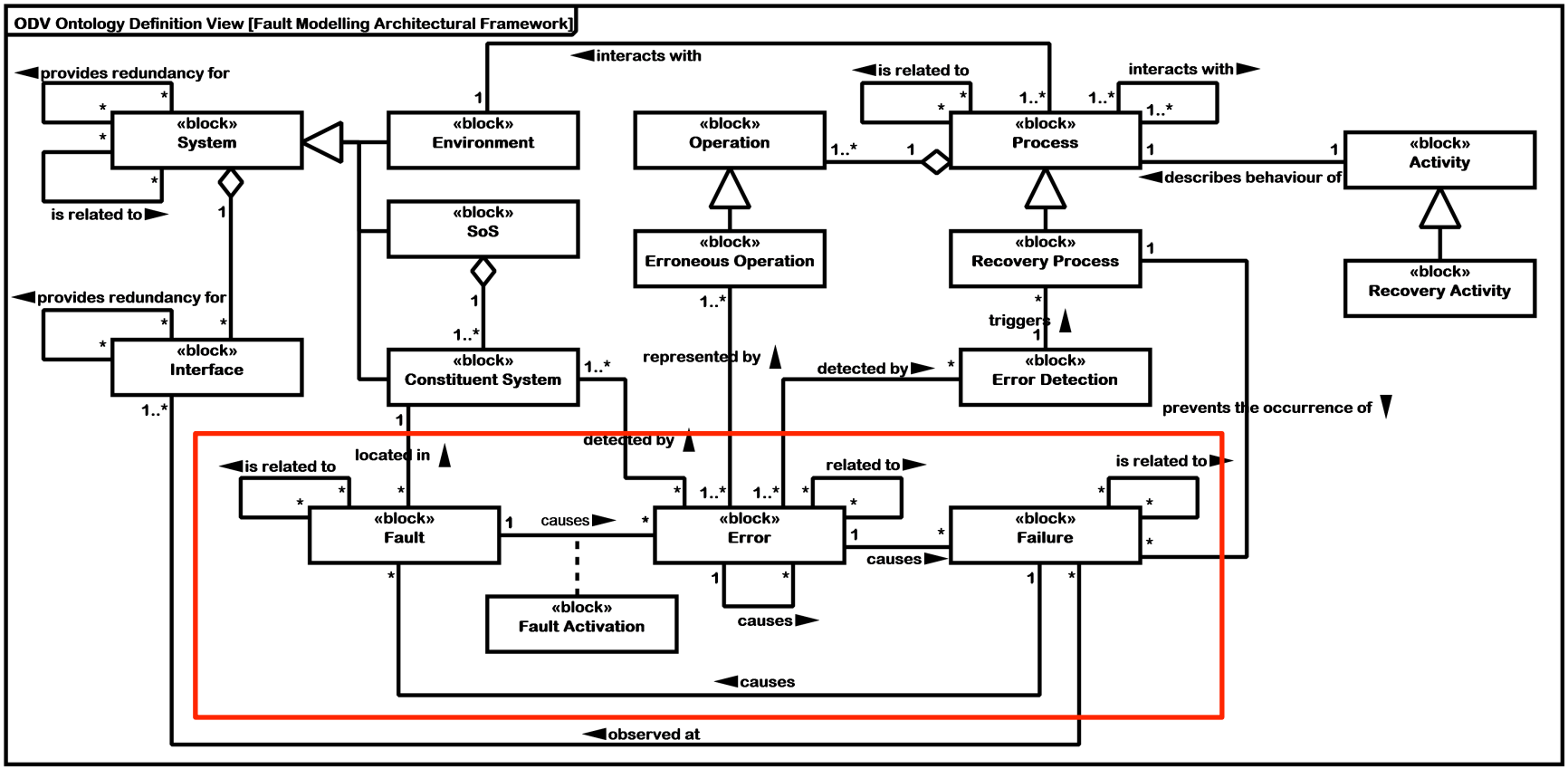
- Defined a Fault Modelling Architectural Framework (FMAF) + SysML profile



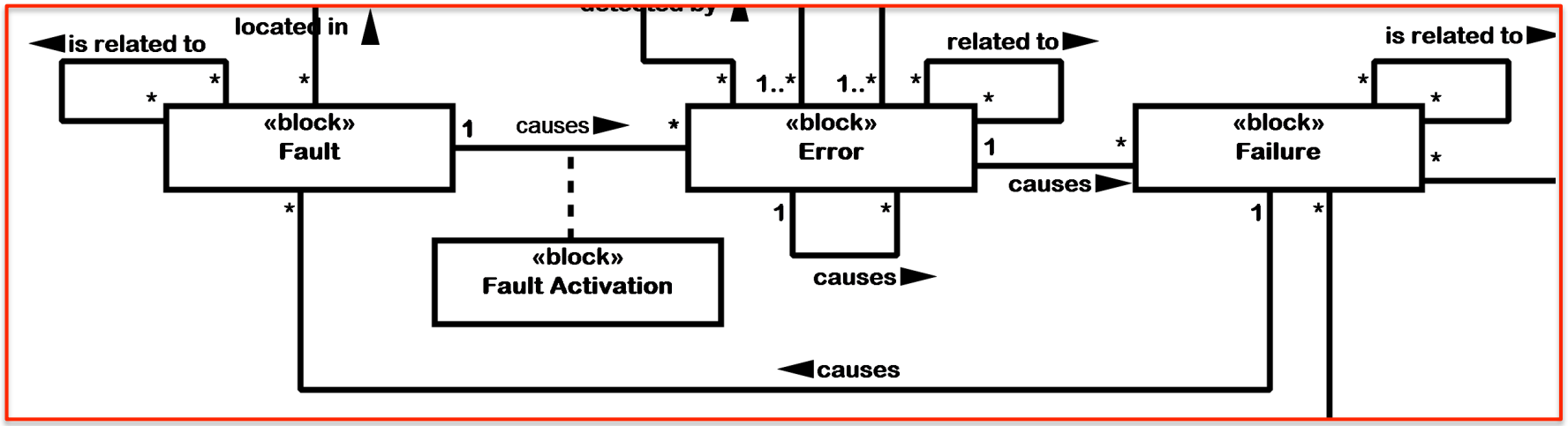
- Prompts an SoS engineer to consider the impact of faults at the early stages of design
- A coherent set of views & concepts for designing fault-tolerant SoSs


FMAF Ontology Definition

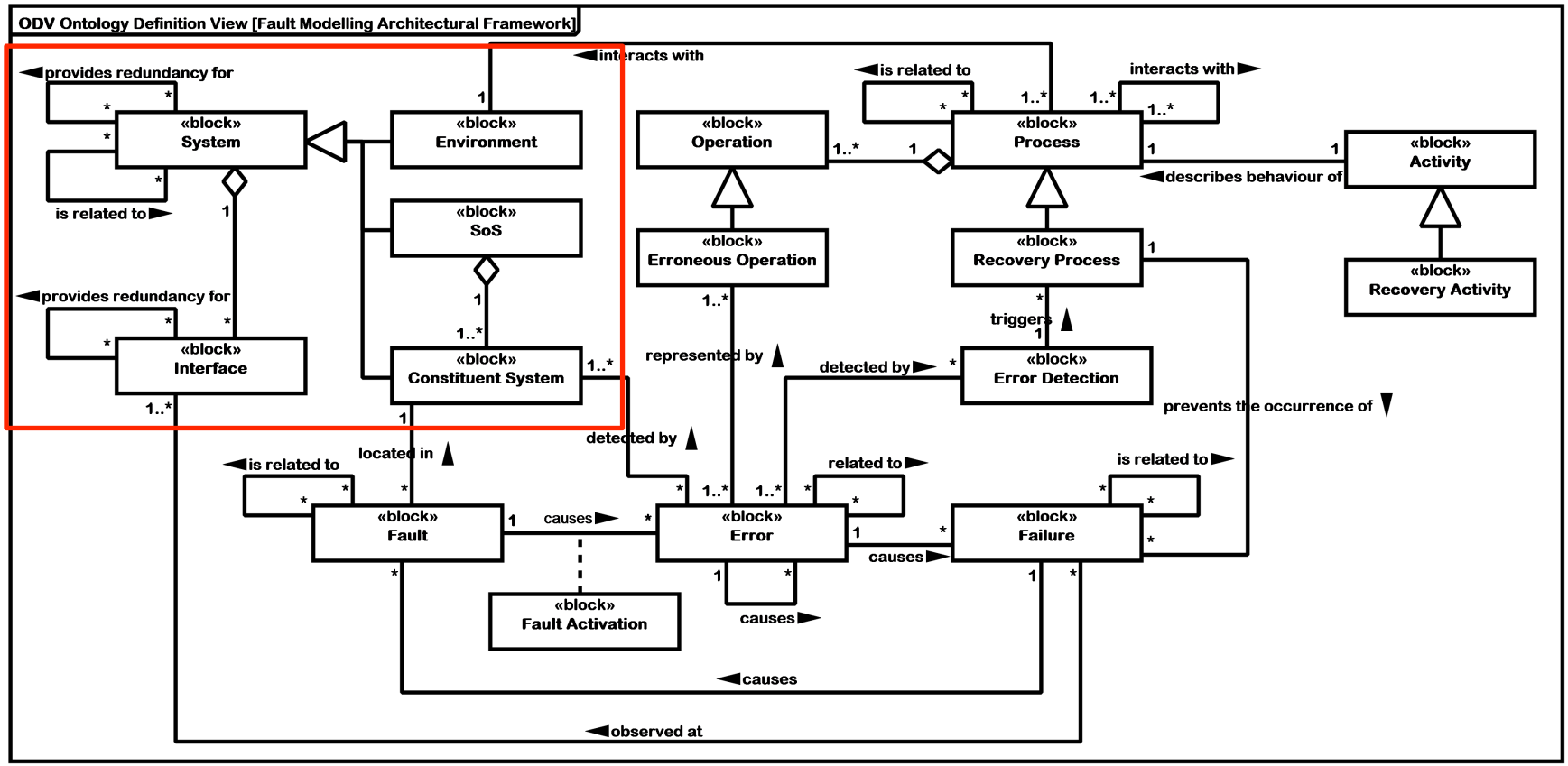


FMAF Ontology Definition

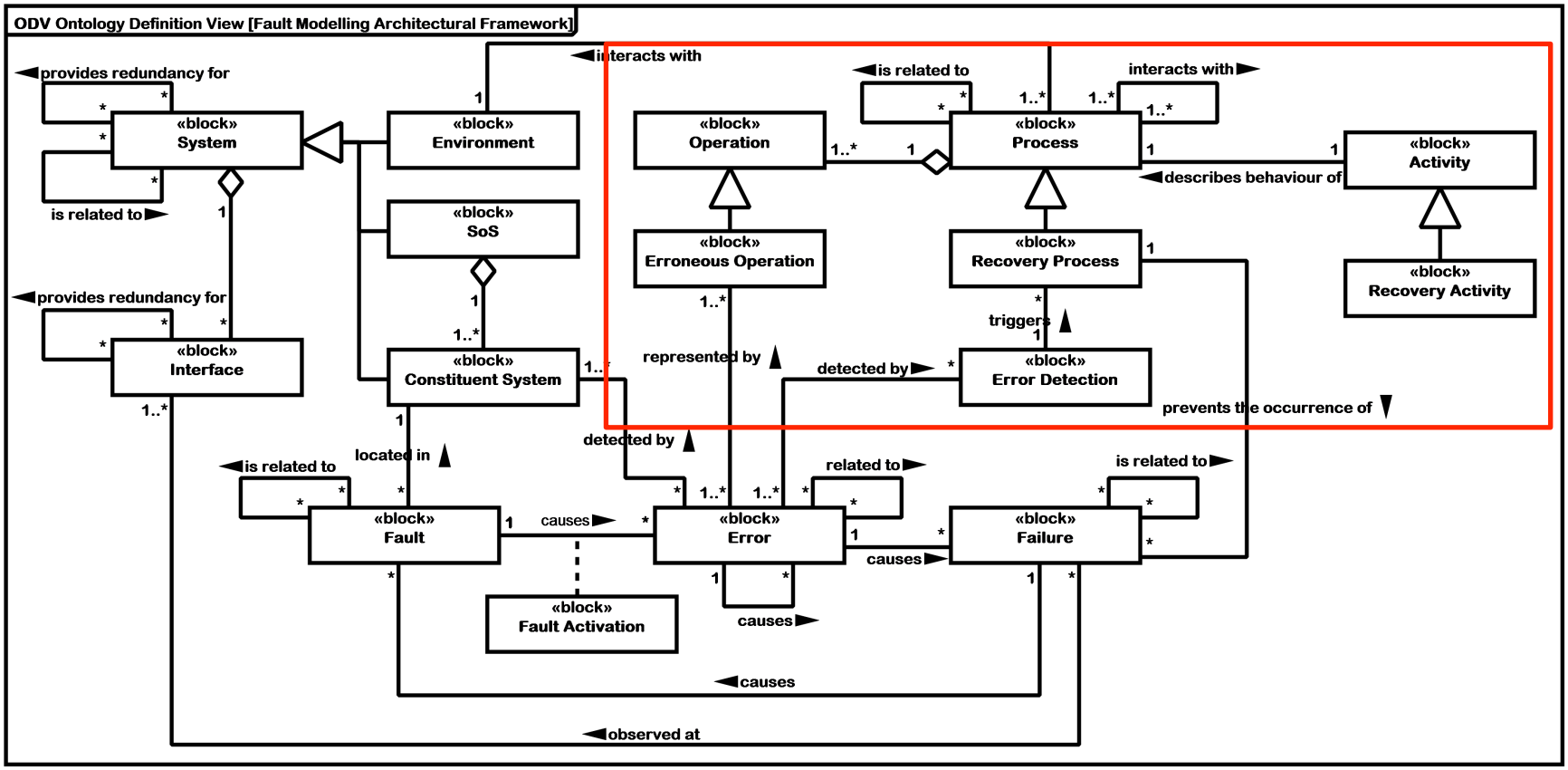


- **SoS failure**: "*a deviation of the service provided by the SoS from expected (correct) behaviour*"
- **SoS error**: "*the part of the SoS state that can lead to its subsequent service failure*"
- **SoS fault**: "*the adjudged or hypothesised cause of an error*"

FMAF Ontology Definition



FMAF Ontology Definition



FMAF Viewpoints

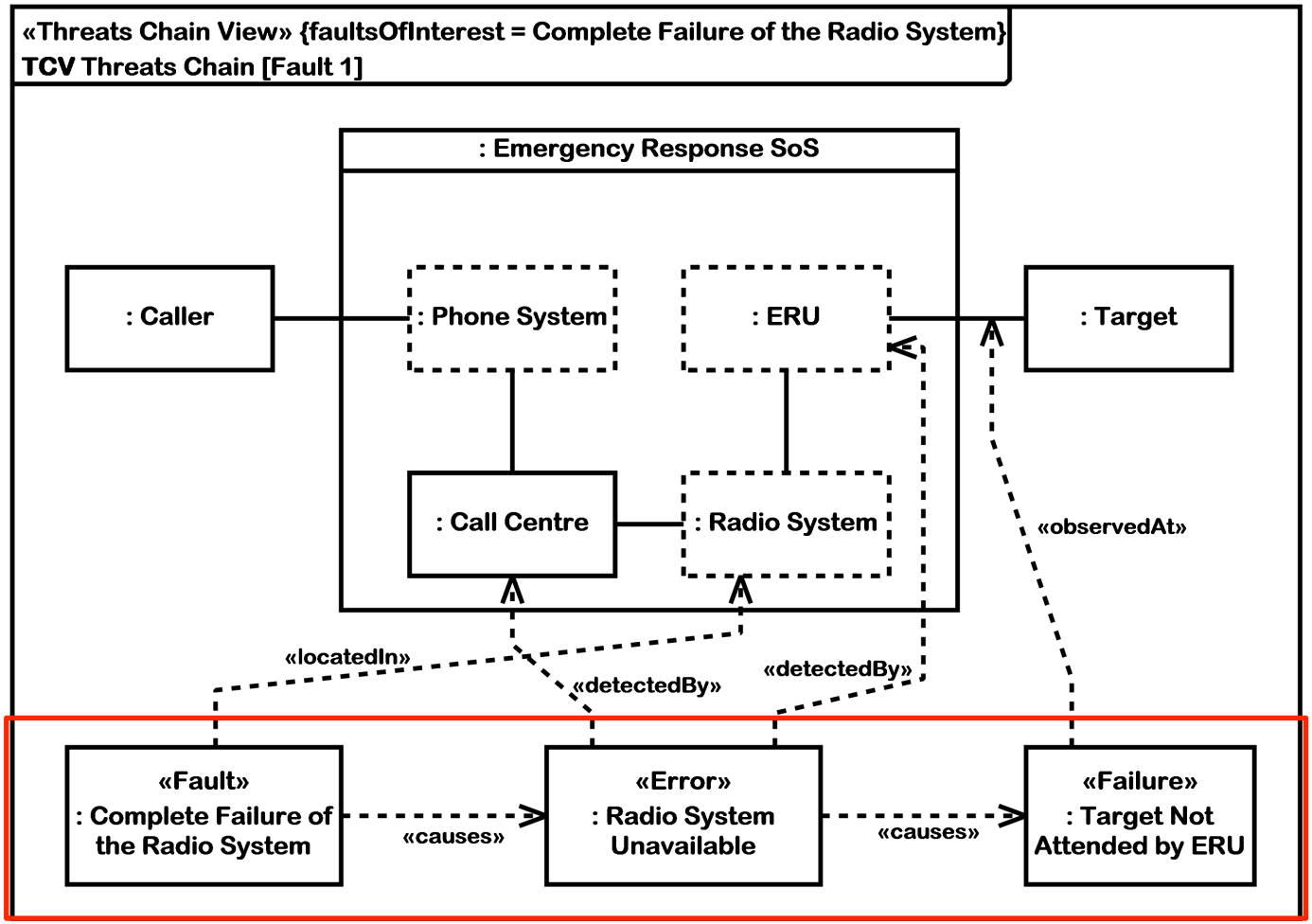
Structural Viewpoints

- Fault/Error/Failure Definition
- Threats Chain
- Fault Tolerance Structure
- Fault Tolerance Connections

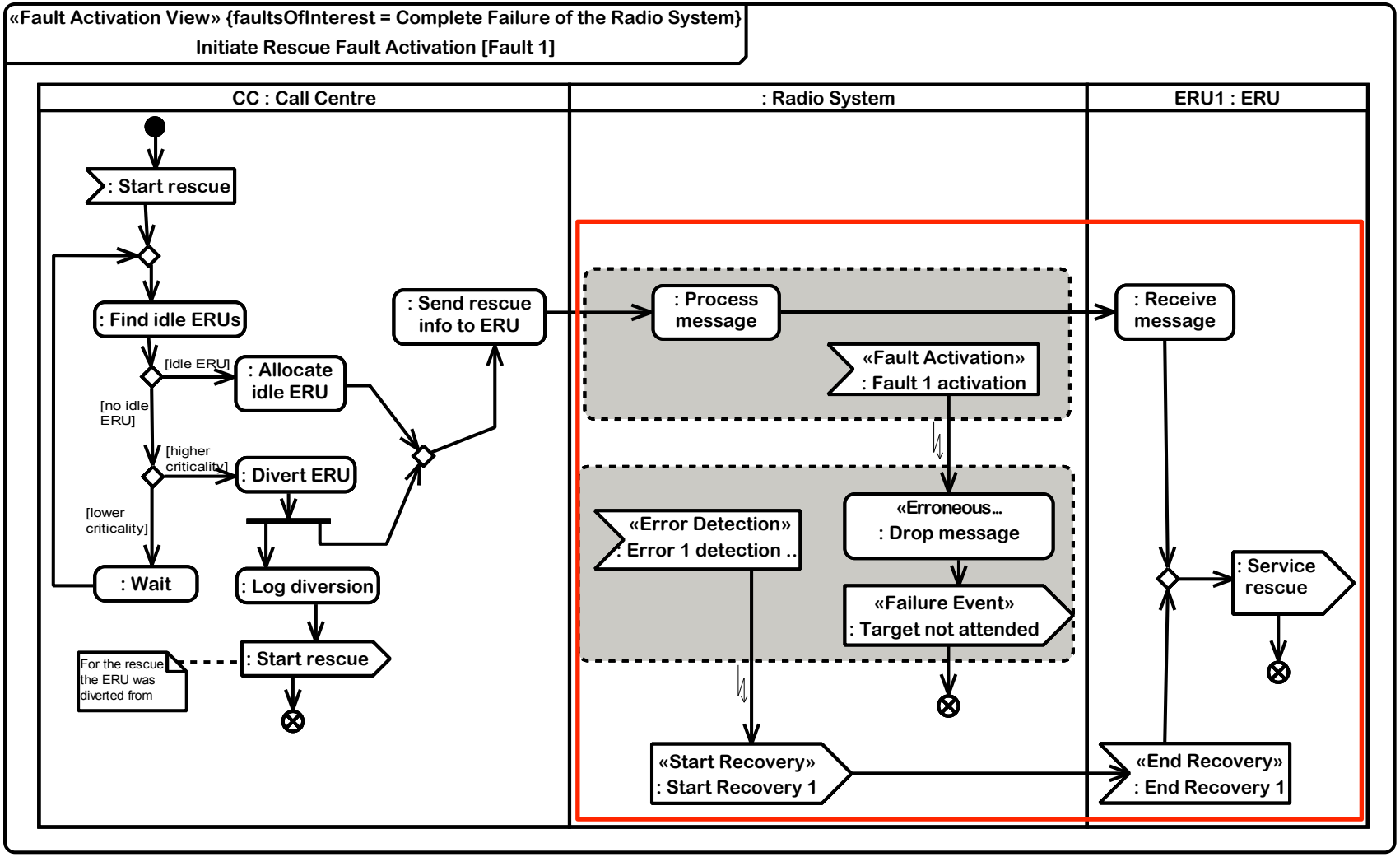
Behavioural Viewpoints

- Erroneous/Recovery Processes
- Erroneous/Recovery Scenarios
- Fault Activation
- Recovery

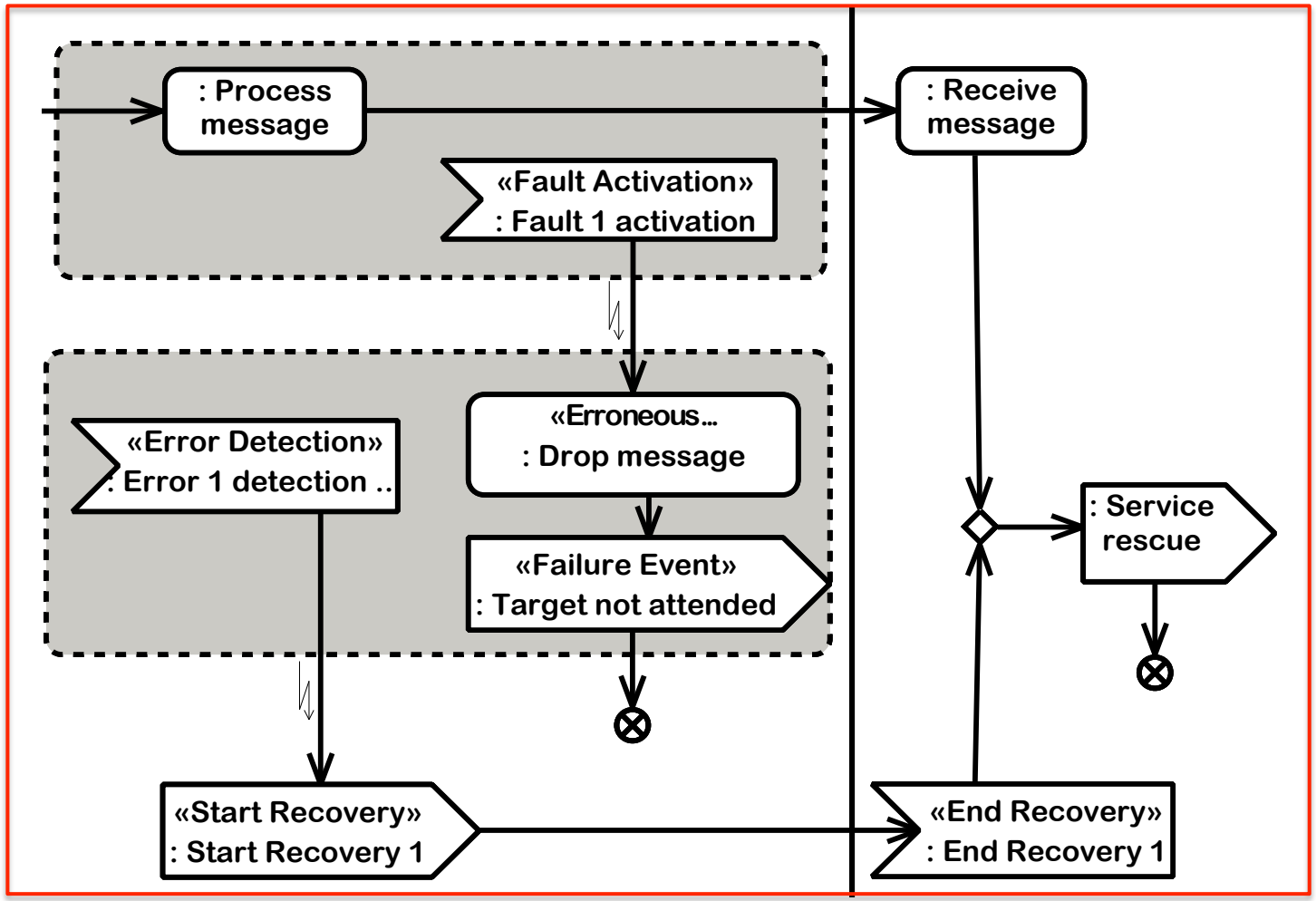
ER SoS: Threats Chain



ER SoS: Fault Activation

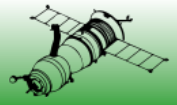


ER SoS: Fault Activation



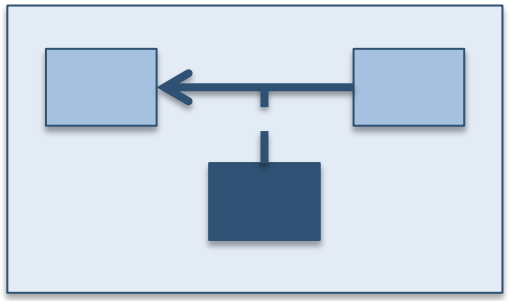
Outline

- Introduction
- Case study
- Traceable engineering of fault-tolerant SoSs
  - Requirements engineering
  - Architectural design
  - **Requirements tracing**
- Integration into a systems engineering process
- Conclusions and future work


Requirements Tracing

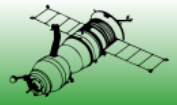
- Traceability Pattern – a structured way of defining traceability
  - in this case we will use it for fault tolerance requirements traceability



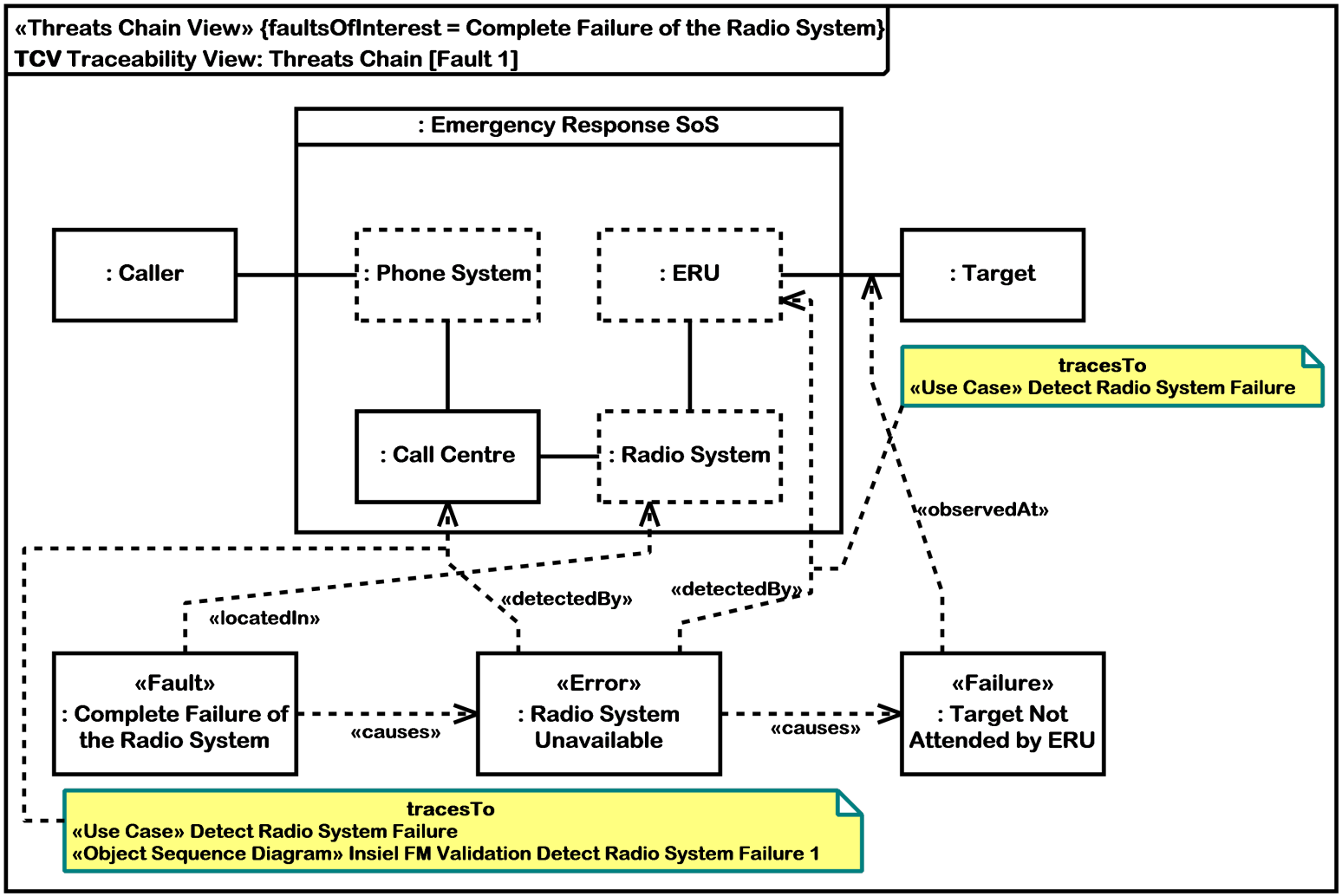
Viewpoint	Description
Relationship Identification	Identifies the set of traceability relationships that may be used
Traceability Identification	Identifies which traceable elements may partake in which traceability relationships
Traceability	Shows traces between traceable elements
Impact	Shows traceability trees for traceable elements

Traceability Pattern: Traceability Identification

From (FMAF)	To (RE)	Relationship
Fault/Error/Failure/Definition View	Requirement	tracesTo
Fault	Requirement	tracesTo
Error	Requirement	tracesTo
Failure	Requirement	tracesTo
detectedBy dependency (*Threats Chain View*)	Use case Validation View	tracesTo
Erroneous/Recovery Scenarios View	Validation View	tracesTo
Error detection interruptible region (*Fault Activation View*)	Use case Validation View	tracesTo
Recovery View	Use case Validation View	tracesTo

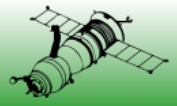




ER SoS Traceability



Outline

- Introduction
- Case study
- Traceable engineering of fault-tolerant SoSs
  - Requirements engineering
  - Architectural design
  - Requirements tracing
- **Integration into a systems engineering process**
- Conclusions and future work

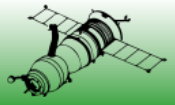

Integrating with SE

Can our approach easily be integrated into an SE architectural process?

Architectural Design Process:

1. Define the architecture
2. Analyse and evaluate the architecture
3. Document and maintain the architecture

SoS is the system, constituent system (CS) is a system element


Integrating with SE

1. Define the architecture

- Rigorous fault modelling helps engineers understand the system
- Easily incorporated into iterative process
- Identify which constituent systems need to implement detection and recovery functionality

Integrating with SE

2. Analyse and evaluate the architecture

- Supports intuitive documentation trail for evaluations based on fault-tolerant criteria
- Can identify (un)desirable tradeoffs, unacceptable risks
- Behavioural models from FMAF can model and analyse specific fault-tolerant scenarios

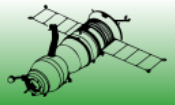
Integrating with SE

3. Documentation and maintain the architecture

- Long-term documentation requirements are supported by the traceable links to an SoS-appropriate RE process
- Links candidate and selected architectures with safety requirements

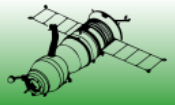
Outline

- Introduction
- Case study
- Requirements engineering
- Fault Modelling Architectural Framework
- Traceability
- Integrating into a systems engineering process
- **Conclusions and future work**


Conclusions

- A structured approach to capturing requirements of fault-tolerant SoSs
- Traceable mapping of fault tolerance requirements into SoS architectural designs
- Disciplined development of FT architectures
- SysML profile has been developed
- Integrate with Fault Analysis AF and third party industrial-standard tools
- Can be complemented with use of formal verification using CML

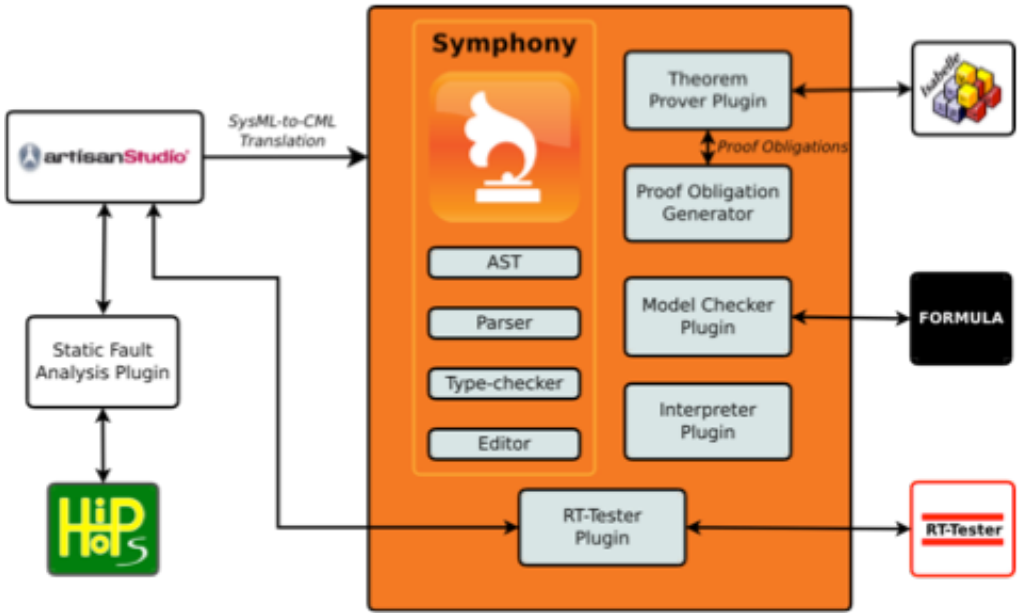

INCOSE International Symposium
Las Vegas, NV
June 30 - July 3, 2014

COMPASS SUMMER SCHOOL:
New Developments in Model-Based SoS Engineering

Radisson Blu Edwardian Vanderbilt, London, 17-18 September 2014

Training in COMPASS methods and tools:

- Architectural SoS modelling
- SysML
- The Symphony tool
- Model-based testing



More information:

http://www.compass-research.eu/summerschool.html

Newcastle University

zoe.andrews@ncl.ac.uk @ZoeAndrewsNcl
claire.ingram@ncl.ac.uk @_Claire_Ingram
richard.payne@ncl.ac.uk @riffio

COMPASS

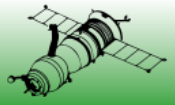
www.compass-research.eu

http://www.compass-research.eu/summerschool.html

INCOSE International Symposium
Las Vegas, NV
June 30 - July 3, 2014

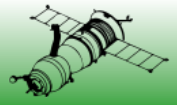
Traceable Engineering of Fault-Tolerant SoSs

Zoe Andrews (Newcastle University, UK)
Claire Ingram (Newcastle University, UK)
Richard Payne (Newcastle University, UK)
Alexander Romanovsky (Newcastle University, UK)
Jon Holt (Atego, UK)
Simon Perry (Atego, UK)


Traceability Pattern

Viewpoint	Description
Relationship Identification	Identifies the set of traceability relationships that may be used
Traceability Identification	Identifies which traceable elements may partake in which traceability relationships
Traceability	Shows traces between traceable elements
Impact	Shows traceability trees for traceable elements

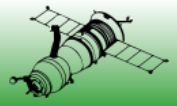




Traceability Pattern: Traceability Identification

From (FMAF)	To (RE)
Fault/Error/Failure/Definition View	Requirement
Fault	Requirement
Error	Requirement
Failure	Requirement
detectedBy dependency (*Threats Chain View*)	Use case Validation View
Erroneous/Recovery Scenarios View	Validation View
Error detection interruptible region (*Fault Activation View*)	Use case Validation View
Recovery View	Use case Validation View

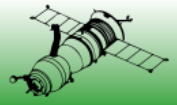
Conclusions

- Our approach can be implemented using any architectural modelling language
- Intuitive, graphical way to document design decisions and trade-offs
- Suitable for involving users
- Applicable for non-software elements
- Can be fitted to standard SE processes
- Helpful for developing reuseable FT architectures


Future work

- Developing links between our approach here and external tools for fault analysis (e.g., HiP-HOPS)
- Extending model behaviour
- Integrated with other work, especially our Fault Analysis Architectural Framework


ER SoS: Validation Interaction

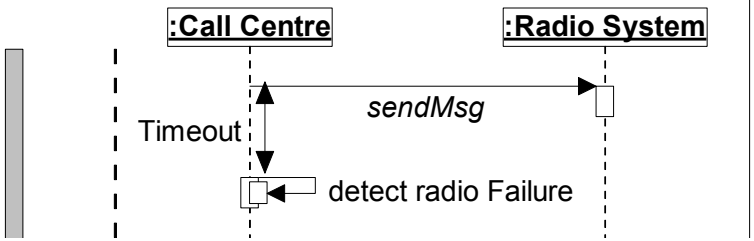
VIV Insiel FM Validation Detect Radio System Failure 1

Description

The mission details are transmitted to the ERU

The message times out

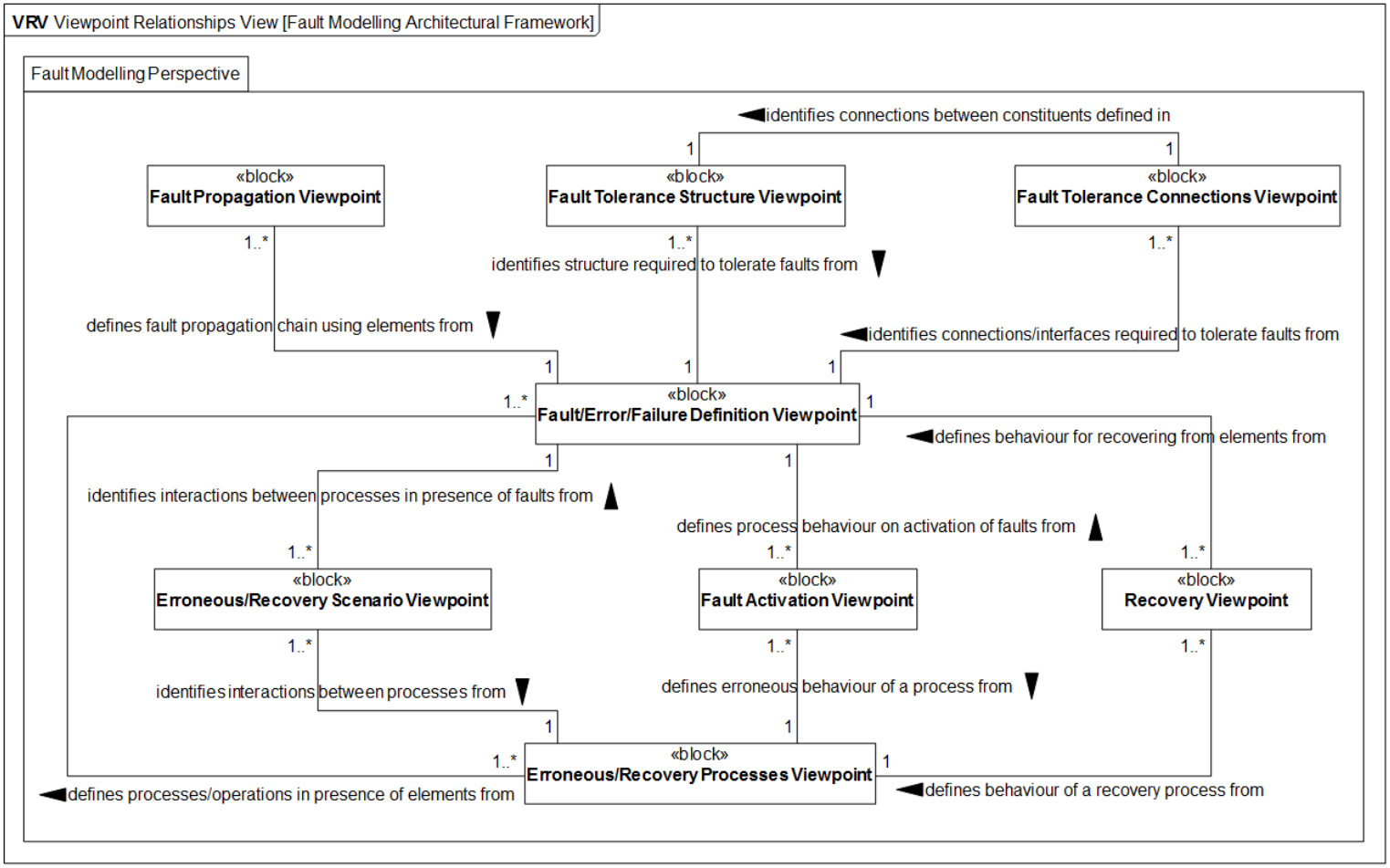
Call Centre detects a Radio System failure (no ack within timeout)



Fault Modelling Architectural Framework

- Standard dependability definitions [ALRL2004] specialised to SoS:
  - **SoS failure**: "*a deviation of the service provided by the SoS from expected (correct) behaviour*"
  - **SoS error**: "*the part of the SoS state that can lead to its subsequent service failure*"
  - **SoS fault**: "*the adjudged or hypothesised cause of an error*"
- NB: a failure of a CS can cause a fault of the SoS

FMAF Viewpoint Relationships



Outline

- Requirements engineering
  - Overview of SoS-ACRE
  - Application to fault tolerance – where the focus is (e.g. FT requirements, CIV for each fault, VIV for fault activation, error detection, recovery scenarios)
  - Application to case study (RDV, CIV, VIV)
- FMAF
  - FMAF overview (motivation, purpose etc. – use RCV to describe)
  - FMAF concepts – via ontology
  - FMAF viewpoints
  - Application to case study (in particular – include TCV, FAV?)