Model-based SoS Engineering with the COMPASS project
• Systems of systems (SoS)
• The COMPASS Project
  – The COMPASS SoS Engineering Approach
  – Requirements Engineering
  – Architectural Modelling
  – Formal Modelling
Outline

• **Systems of systems (SoS)**
  • The COMPASS Project
    – The COMPASS SoS Engineering Approach
    – Requirements Engineering
    – Architectural Modelling
    – Formal Modelling
Defining systems of systems

A system composed of other constituents, each of which is an independent system in its own right

- Operationally & managerially independent constituents
- Geographically distributed
- Continuously evolving
- Exhibiting emergent behaviour

“emergent”: global behaviour produced by the whole SoS, can’t be produced by a single constituent alone
Key challenges for modelling SoSs:

• Difficulty of analysing compositions/emergent behaviour

• Wide range of properties to model: functionality, concurrency, communication, inheritance, time, sharing, mobility...

• Lack of full disclosure between constituents: imprecise & uncertain information about “black boxes”

• Heterogeneity – different modelling paradigms in one SoS
Outline

• Systems of systems (SoS)

• The COMPASS Project
  – The COMPASS SoS Engineering Approach
  – Requirements Engineering
  – Architectural Modelling
  – Formal Modelling
COMPASS

• Comprehensive Modelling for Advanced Systems of Systems
• EU Framework 7; October 2011 - October 2014
Audio-visual/Home Automation:
- Multiple content sources, DRMs,
- Multiple devices
- Mobile and concurrent systems

*Bang & Olufsen (Denmark): Can we ensure consistent “user experience” as devices, content, DRM, etc., change?*

Traffic Management:
- Wide variety of constituent systems: some legacy, some new
- Harsh physical environment
- Complex integration of new systems and architectures
- Faults & Fault Tolerance

*West Consulting (Netherlands): How can we add new/evolved constituent systems, and be sure that they will integrate seamlessly?*

Smart Grid:
- Many stakeholders with different needs
- Frequent changes to equipment and stakeholder needs.
- Safety cannot be guaranteed centrally

*GridManager (Denmark): Ensure continuity of service & safety in the presence of change & faults?*

Emergency Response:
- Stakeholders (patients to gov’t departments)
- Human intervention required for many interactions
- Assurance of global performance and security properties

*Insiel (Italy): Can we manage evolution to a decentralised SoS while gaining assurance of global properties?*
COMPASS focused on three major technical challenges:

• Independence and autonomy of constituent systems
  – Constituent systems evolve at the behest of their owners
  – *Response*: Collaborative SoS modelling by contractual *(rely, guarantee)* interface specification

• Complexity of confirming/refuting SoS-level properties
  – Verification of emergence
  – *Response*: verified refinement for engineering of emergent properties; simulation tools allow exploration for unanticipated behaviours

• Semantic heterogeneity (integrating models)
  – Wide range of interacting features in models (e.g. location, time, concurrency, data, communication)
  – *Response*: extensible semantic basis
Outline

• Systems of systems (SoS)

• The COMPASS Project
  – The COMPASS SoS Engineering Approach
    – Requirements Engineering
    – Architectural Modelling
    – Formal Modelling
COMPASS Technology

This is what a user of COMPASS technology sees

Semi-formal Modelling
• SysML used for SoS modelling
• Guidelines for Requirements, Architecture, Integration
• SoS modelling profiles
• Architectural patterns and extensible frameworks
COMPASS Technology

This is what a user of COMPASS technology sees

Semi-formal Modelling
- SysML used for SoS modelling
- Guidelines for Requirements, Architecture, Integration
- SoS modelling profiles
- Architectural patterns and extensible frameworks

Tool-supported Analysis
- Model-checker
- Automated proof
- Static Fault Analysis
- Test generation
- Simulation
- Model-in-Loop Test
- Exploration of design space
COMPASS Technology

This is what a user of COMPASS technology sees

**Semi-formal Modelling**
- SysML used for SoS modelling
- Guidelines for Requirements, Architecture, Integration
- SoS modelling profiles
- Architectural patterns and extensible frameworks

**Formal Modelling**
- CML allows representation of behavioural semantics of the SoS
- Supports contract specification
- Describes functionality, object-orientation, concurrency, real-time, mobility.
- Can be extended to new paradigms

**Tool-supported Analysis**
- Model-checker
- Automated proof
- Static Fault Analysis
- Test generation
- Simulation
- Model-in-Loop Test
- Exploration of design space

---

process CallCentreProc = begin
...
actions
MERGE1(r) =
(dcl e: set of ERUId @ e := findIdleERUs();
  (do
    e = {} -> DECISION2(r)
    | e <> {} ->
      (dcl e1: ERUId @ e1 :=
        allocateIdleERU(e, r); MERGE2(e1, r))
  end)

process InitiateRescue = CallCentreProc 
  [ | SEND_CHANNELS | ] RadioSystemProc 
  [ | RCV_CHANNELS | ] ERUsProc
## COMPASS SoS Technologies

<table>
<thead>
<tr>
<th>Technology</th>
<th>Method /Tool</th>
</tr>
</thead>
<tbody>
<tr>
<td>SoS-ACRE Requirement Eng.</td>
<td>M</td>
</tr>
<tr>
<td>COMPASS AF Framework</td>
<td>M</td>
</tr>
<tr>
<td>Integration Framework</td>
<td>M</td>
</tr>
<tr>
<td>Refinement Framework</td>
<td>M</td>
</tr>
<tr>
<td>Interface Pattern</td>
<td>M</td>
</tr>
<tr>
<td>Test Pattern</td>
<td>M</td>
</tr>
<tr>
<td>Contract Pattern</td>
<td>M</td>
</tr>
<tr>
<td>Traceability Pattern</td>
<td>M</td>
</tr>
<tr>
<td>Fault Modelling Architectural Framework</td>
<td>M</td>
</tr>
<tr>
<td>CML Language</td>
<td>M</td>
</tr>
<tr>
<td>Formal Refinement</td>
<td>M</td>
</tr>
<tr>
<td>Compositional Analysis</td>
<td>M</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Technology</th>
<th>Method /Tool</th>
</tr>
</thead>
<tbody>
<tr>
<td>Compositional Design</td>
<td>M</td>
</tr>
<tr>
<td>Specialised Test Strategies</td>
<td>T</td>
</tr>
<tr>
<td>Symphony Interpreter</td>
<td>T</td>
</tr>
<tr>
<td>Symphony Model Checker</td>
<td>T</td>
</tr>
<tr>
<td>Symphony Theorem Prover</td>
<td>T</td>
</tr>
<tr>
<td>Symphony RT-Tester</td>
<td>T</td>
</tr>
<tr>
<td>Symphony Test Automation</td>
<td>T</td>
</tr>
<tr>
<td>Symphony Co-Simulation Engine</td>
<td>T</td>
</tr>
<tr>
<td>Symphony Linkage to Executable</td>
<td>T</td>
</tr>
<tr>
<td>Fault Analysis Tool</td>
<td>T</td>
</tr>
</tbody>
</table>
## COMPASS SoS Technologies

<table>
<thead>
<tr>
<th>Technology</th>
<th>Method/Tool</th>
</tr>
</thead>
<tbody>
<tr>
<td>SoS-ACRE Requirement Eng.</td>
<td>M</td>
</tr>
<tr>
<td>COMPASS AF Framework</td>
<td>M</td>
</tr>
<tr>
<td>Integration Framework</td>
<td>M</td>
</tr>
<tr>
<td>Refinement Framework</td>
<td>M</td>
</tr>
<tr>
<td>Interface Pattern</td>
<td>M</td>
</tr>
<tr>
<td>Test Pattern</td>
<td>M</td>
</tr>
<tr>
<td><strong>Contract Pattern</strong></td>
<td>M</td>
</tr>
<tr>
<td>Traceability Pattern</td>
<td>M</td>
</tr>
<tr>
<td><strong>Fault Modelling Architectural Framework</strong></td>
<td>M</td>
</tr>
<tr>
<td>CML Language</td>
<td>M</td>
</tr>
<tr>
<td>Formal Refinement</td>
<td>M</td>
</tr>
<tr>
<td>Compositional Analysis</td>
<td>M</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Technology</th>
<th>Method/Tool</th>
</tr>
</thead>
<tbody>
<tr>
<td>Compositional Design</td>
<td>M</td>
</tr>
<tr>
<td>Specialised Test Strategies</td>
<td>T</td>
</tr>
<tr>
<td>Symphony Interpreter</td>
<td>T</td>
</tr>
<tr>
<td>Symphony Model Checker</td>
<td>T</td>
</tr>
<tr>
<td>Symphony Theorem Prover</td>
<td>T</td>
</tr>
<tr>
<td>Symphony RT-Tester</td>
<td>T</td>
</tr>
<tr>
<td>Symphony Test Automation</td>
<td>T</td>
</tr>
<tr>
<td>Symphony Co-Simulation Engine</td>
<td>T</td>
</tr>
<tr>
<td>Symphony Linkage to Executable</td>
<td>T</td>
</tr>
<tr>
<td>Fault Analysis Tool</td>
<td>T</td>
</tr>
</tbody>
</table>
COMPASS Approach

COMPASS APPROACH

COMPASS Ontology
COMPASS Architectural Framework
COMPASS Processes

COMPASS Ontology
System
System of Systems
Constituent System
COMPASS Approach

COMPASS Ontology

COMPASS Architectural Framework

COMPASS Processes

System of Systems

Protocol Definition Viewpoint

Interface Integration Viewpoint

Requirement Context Viewpoint

Interface Definition Viewpoint

Context Definition Viewpoint

Viewpoints use modelling elements from ontology
COMPASS Approach

COMPASS APPROACH

COMPASS Ontology

COMPASS Architectural Framework

COMPASS Processes

Processes use viewpoints from architectural framework
Outline

• Systems of systems (SoS)
• Model-based SoS Engineering

• The COMPASS Project
  – The COMPASS SoS Engineering Approach
  – Requirements Engineering
    – Architectural Modelling
    – Formal Modelling
    – Verification and Analysis
    – Fault Modelling
SoS Engineering Example

- Cross-border traffic control
- Incident in one country may require “corridor” with temporary traffic controls to be established across an international border, in a co-ordinated way
- Need to define contract to which each country adheres
- Driver perceives no difference from one country to another
Cross-border traffic control
Incident in one country may require “corridor” with temporary traffic controls to be established across an international border, in a co-ordinated way
Need to define contract to which each country adheres
Driver perceives no difference from one country to another
COMPASS developed SoS-ACRE, a requirements process for SoSs, with the following viewpoints

<table>
<thead>
<tr>
<th>Viewpoint</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source Element</td>
<td>Identifies requirements source information</td>
</tr>
<tr>
<td>Requirement Description</td>
<td>Contains a structured description of each requirement</td>
</tr>
<tr>
<td>Context Definition</td>
<td>Identifies the points of view (contexts) which will be explored in the RCVs</td>
</tr>
<tr>
<td>Requirement Context</td>
<td>Describes the requirements defined on the RDV in context</td>
</tr>
<tr>
<td>Context Interaction</td>
<td>Overview of relationships between the contexts of various CSs; combines the RCVs for each CS</td>
</tr>
<tr>
<td>Validation</td>
<td>Provides the basis for demonstrating that requirements can be validated</td>
</tr>
<tr>
<td>Validation Interaction</td>
<td>A combined view of the scenarios for related use cases in the SoS; combines the VVs of the use cases</td>
</tr>
</tbody>
</table>
**SoS-ACRE Viewpoints**

COMPASS developed SoS-ACRE, a requirements process for SoSs, with the following viewpoints:

<table>
<thead>
<tr>
<th>Viewpoint</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source Element</td>
<td>Identifies requirements source information</td>
</tr>
<tr>
<td>Requirement Description</td>
<td>Contains a structured description of each requirement</td>
</tr>
<tr>
<td>Context Definition</td>
<td>Identifies the points of view (contexts) which will be explored in the RCVs</td>
</tr>
<tr>
<td>Requirement Context</td>
<td>Describes the requirements defined on the RDV in context</td>
</tr>
<tr>
<td>Context Interaction</td>
<td><strong>Overview of relationships between the contexts of various CSs; combines the RCVs for each CS</strong></td>
</tr>
<tr>
<td>Validation</td>
<td>Provides the basis for demonstrating that requirements can be validated</td>
</tr>
<tr>
<td>Validation Interaction</td>
<td>A combined view of the scenarios for related use cases in the SoS; combines the VVs of the use cases</td>
</tr>
</tbody>
</table>
Border Traffic Requirements

- R01: Border Traffic SoS Requirements
  - Uninterrupted traffic between borders
  - Allow cross-border traffic corridor
  - Allow country to evolve traffic management independently

- R02: Uninterrupted traffic between borders
  - The SoS should allow uninterrupted traffic flow of vehicles across borders, where roads are present connecting two countries.

- R03: Allow cross-border traffic corridor
  - When an incident occurs close to the country boundary, a cross-border speed corridor must be created to ensure traffic speed decreases at a steady rate.

- R04: Allow country to evolve traffic management independently
  - The management of a given country's traffic system should be performed by that country.

- R05: Individual country must be able to manage traffic flow
  - When an incident occurs within a country's border, it must be able to take measures to ensure traffic speed reduces approaching the incident.

- R06: Cross border communication
  - There should be communication between the Traffic Management Systems of any two countries.

- R07: Must respond to request from neighbour
  - A country must respond to a request to implement traffic speed reductions on roads entering a neighbouring country.

- R08: Sensing of road traffic conditions
  - A country should be able to sense road traffic conditions and detect when measures should be taken.

- R09: Influence road traffic
  - Countries should be able to influence the speed of road traffic in its own borders.
Requirements in Context
Outline

• Systems of systems (SoS)
• Model-based SoS Engineering

• The COMPASS Project
  – The COMPASS SoS Engineering Approach
  – Requirements Engineering
  – Architectural Modelling
  – Formal Modelling
Architectural Modelling

• When defining a SoS architecture, follow COMPASS architectural approach – patterns and guidelines
• Use collections of *modelling patterns* to define SoS structure and behaviour
• In border traffic example, we define the behaviour required by each country’s TMS – using the *interface contract* pattern
• Identifying contract conformance
Architectural Modelling

- Defining connections and interfaces between systems
• Defining the functionality of the contract
Defining behaviour and ordering
Outline

• Systems of systems (SoS)
• Model-based SoS Engineering

• The COMPASS Project
  – The COMPASS SoS Engineering Approach
  – Requirements Engineering
  – Architectural Modelling
  – Formal Modelling
Formal Modelling

• CML – COMPASS Modelling Language – developed for modelling SoS
• Can model data, functionality, event ordering and communication
  – extensible
• Range of formal analysis techniques
• Tools developed for translating models from SysML into CML
Analysing the Model

```plaintext
inIncident.myId?l?t -> d : nat := calcDistance(t, nationalSpeedLimit)

- NEW_INCIDENT
  - CORRIDOR
    - RE_CHECK
      - NEIGHBOUR_REQ
```

...
Analysing the Model

```plaintext
process CountryTMS =
begin
  ...
  actions
  BEHAVIOUR= NEW_INCIDENT
    []
    NEIGHBOUR_REQ

  NEW_INCIDENT = inIncident.myId?l?t -> d : nat := calcDistance(t, nationalSpeedLimit)
    (dcl d : nat := calcDistance(t, nationalSpeedLimit) @
    CORRIDOR(l, t, d))

  CORRIDOR = l : int, t: nat, d:nat
    @ ACT_STATUS;c:Corridor :=det; ...
  ...
  @ BEHAVIOUR
End

process CountryA = CountryTMS(theAId, theBId, limitA, actCorrA)
process CountryB = CountryTMS(theBId, theAId, limitB, actCorrB)

process BorderTrafficSoS =
  CountryA [interfacer]
  CountryB
```
Analysing the Model

Symphony Tool Platform
- Analyse cross border emergent behaviour
- Simulate execution of model
- Proof obligations generated
- Theorem proving

process CountryTMS =
begin
... actions
     BEHAVIOUR= NEW_INCIDENT
         NEIGHBOUR_REQ

NEW_INCIDENT = inIncident.myId?l?t ->
   (dcl d : nat := calcDistance
   (t, nationalSpeedLimit) @
     CORRIDOR(l, t, d))

CORRIDOR = l : int, t: nat, d:nat
   @ ACT_STATUS;c:Corridor :=det; ...
   @ BEHAVIOUR
End

process CountryA = CountryTMS(theAId, theBId, limitA, actCorrA)
process CountryB = CountryTMS(theBId, theAId, limitB, actCorrB)
process BorderTrafficSoS =
    CountryA [interface]
    CountryB

Symphony
Analysing the Model

Symphony Tool Platform
- Analyse cross border emergent behaviour
- Simulate execution of model
- Proof obligations generated
- Theorem proving

```
inIncident.myId?l?t -> d : nat := calcDistance(t, nationalSpeedLimit)
```

```
NEW_INCIDENT
```

```
CORRIDOR
```

```
RE_CHECK
```

```
process CountryTMS = begin
    ...
    actions
    BEHAVIOUR= NEW_INCIDENT
        NEIGHBOUR_REQ
        NEW_INCIDENT = inIncident.myId?l?t ->
        (dcl d : nat := calcDistance(t, nationalSpeedLimit) @
         CORRIDOR(l, t, d))
        CORRIDOR = l : int, t: nat, d:nat
        @ ACT_STATUS;c:Corridor :=det; ...
    BEHAVIOUR
    End
```

```
process CountryA = CountryTMS(theAId, theBId, limitA, actCorrA)
process CountryB = CountryTMS(theBId, theAId, limitB, actCorrB)
```
Conclusions

• COMPASS has produced a large set of outputs, including methods, tools and guidelines
  – Based on an extensible approach
  – Benefits integrating both semi-formal and formal modelling
  – A unified model-based approach promotes consistency, rigour, traceability, validation and verification

• Join thecompassclub.org!
Research into model-based techniques for developing, analysing and maintaining SoSs

thecompassclub.org

www.compass-research.eu