A DSML Based Approach for Simulating On-board Equipments in Space Applications

Bálint Sódor
Gábor Tróznai, Sándor Szalai

Wigner Research Centre for Physics, Hungarian Academy of Sciences, HUNGARY
sodor.balint@wigner.mta.hu
troznai@wigner.mta.hu

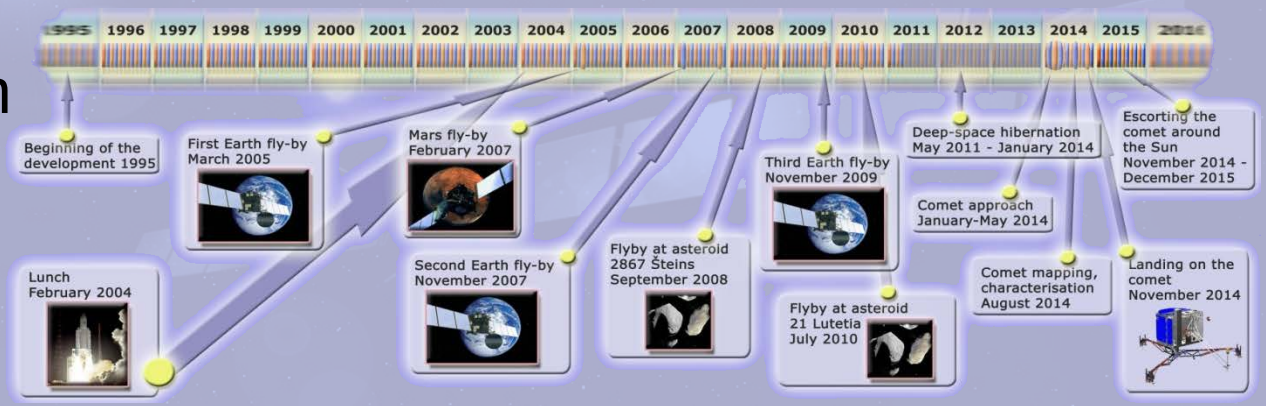
SGF Ltd., HUNGARY
szalai@sgf.hu

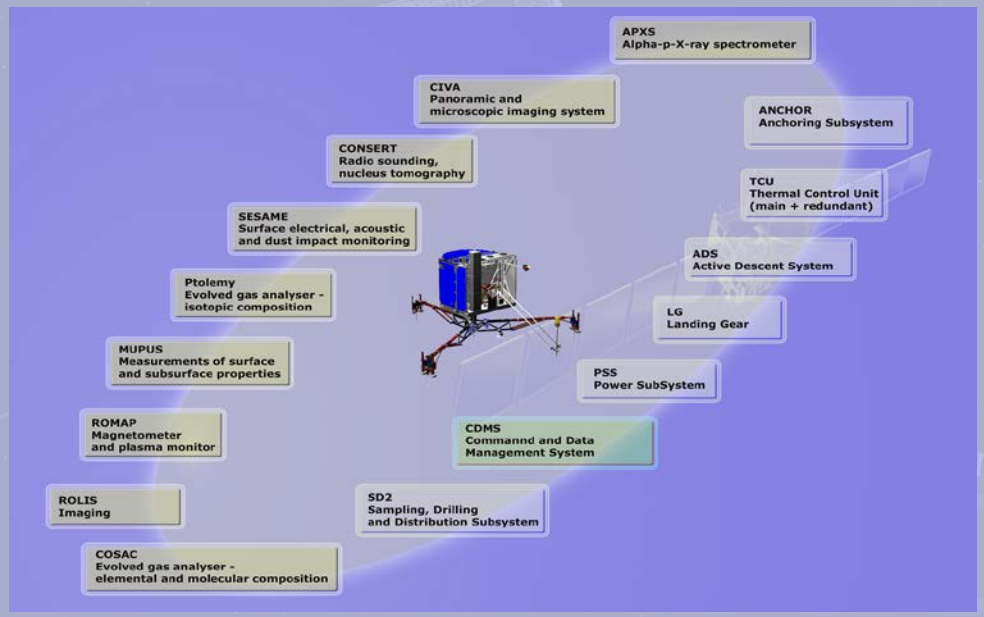
Introduction – About us

- Wigner institute of the Hungarian Academy of Sciences and Space and Ground Facilities Ltd.
- Few decades on the field of space system research
  - Development of EGSEs and on-board equipments
  - From our point of view space systems we are currently focusing on can be characterized as the following
    - Distributed reactive autonomous
    - Embedded, fault tolerant and highly reliable
    - Limited resources
- Participated in many missions

Introduction – LSS 1

- ESA approved the ROSETTA mission in 1993
- Comet observation
  - Orbiter + Lander
  - Long-term mission

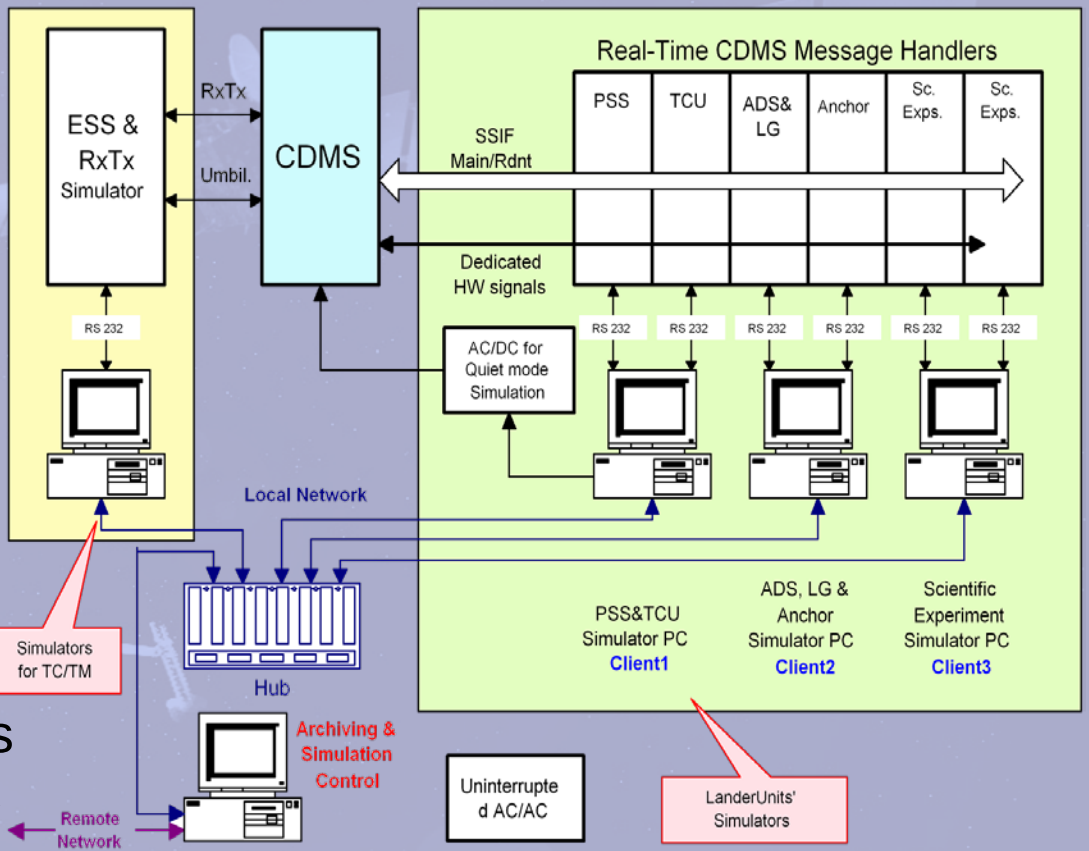




- Philae
  - Our task: CDMS
    - Central computer of a distributed system
      - 6 subsystems, 9 scientific modules
    - Long term mission, reduced maintainability
      - Training operation staff
      - Robustness, fault tolerance
    - Limited resources

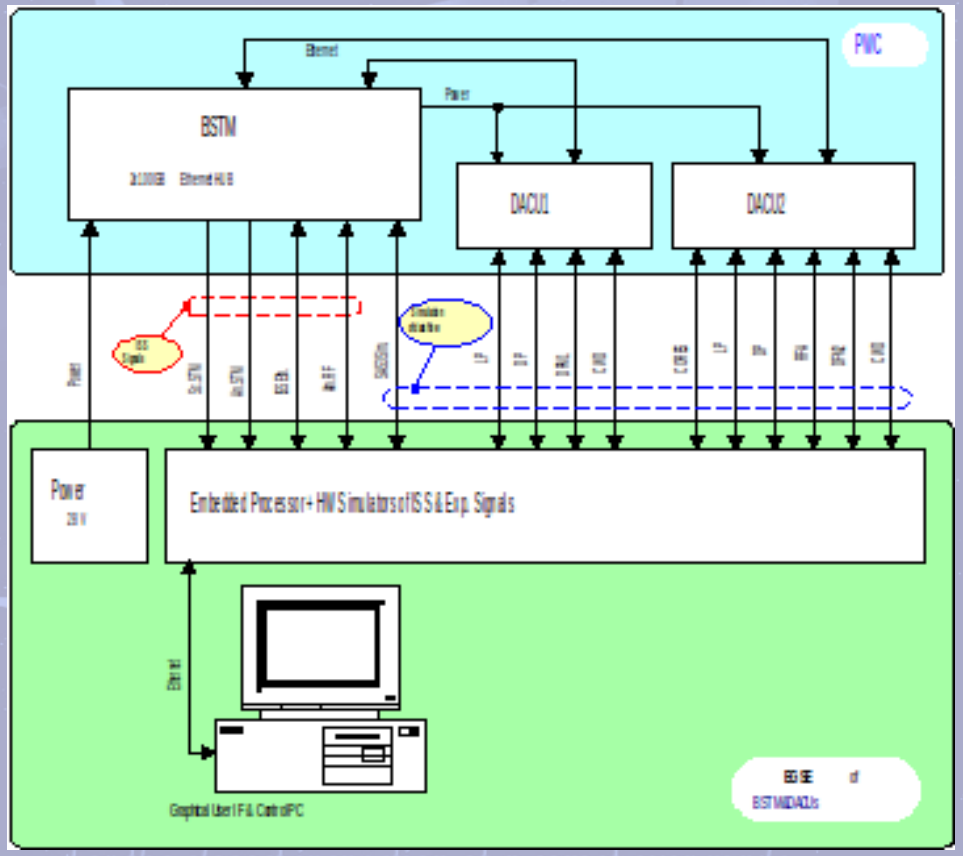
Introduction – LSS 2

- High level of autonomy and long lifetime introduces new requirements
  - Knowledge preservation
  - Training of the operator staff
  - Testing schedules
  - SW/HW modules
  - Event reproduction
  - …
- Solution: Lander Software Simulator (LSS)
- Parallel simulation environment
  - Real CDMS
  - Other on-board equipments
    - Behavior simulated on PC
      - Based on XML models
    - Communication: via Real-time Message Handlers (embedded processors)



Introduction – Obstanovka

- PWC: ISS Russian segment
- 11 scientific subsystems
- Our responsibility: Distributed acquisition and control system
  - 3 embedded units (HW-SW)
  - EGSE and control interface
- Continuously changing specification of the scientific equipments and requirements
- Running test-cases without the full set of subsystems

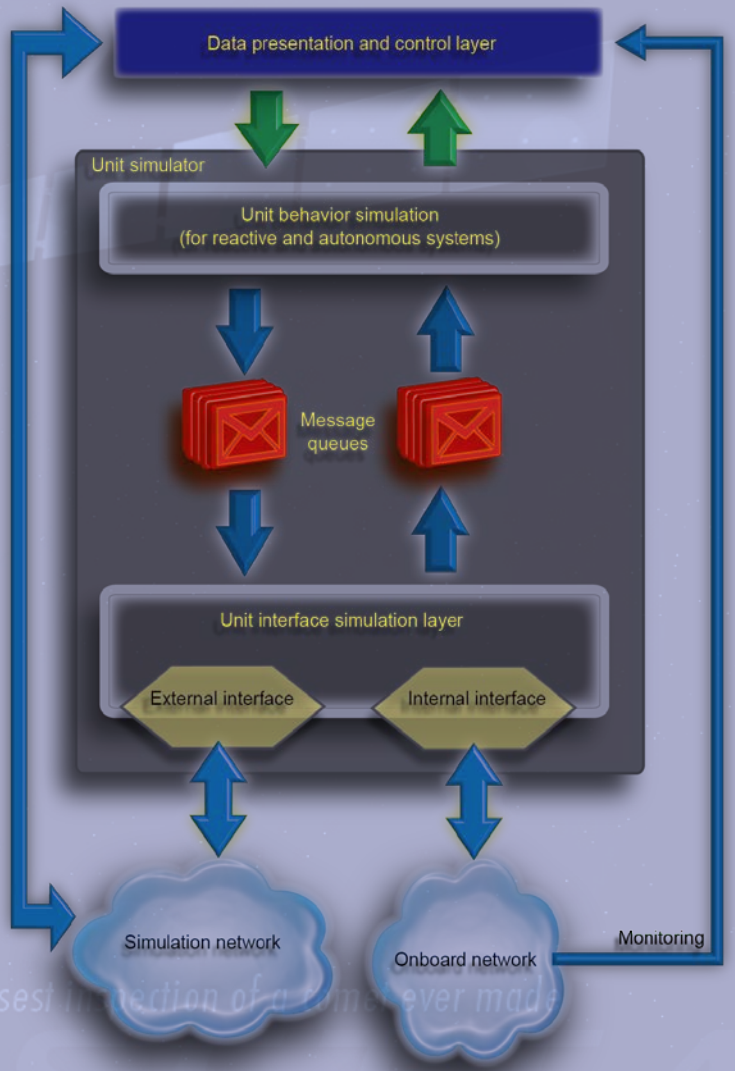


Identified problems

- On-board systems
  - Distributed systems developed parallel
  - Long term missions often operates “out of the reach”
    - Autonomous and reactive behavior
    - Limited resources (computational capacity, storage, communication, power, …)
    - Robustness and fault tolerance has to be ensured
    - Changing specifications over time
- Requirements
  - Modular and scalable testing environment for the whole life-cycle of the mission
  - Rapid integration of concepts and specifications
  - Simulation even in early phase of life-cycle where HW not available

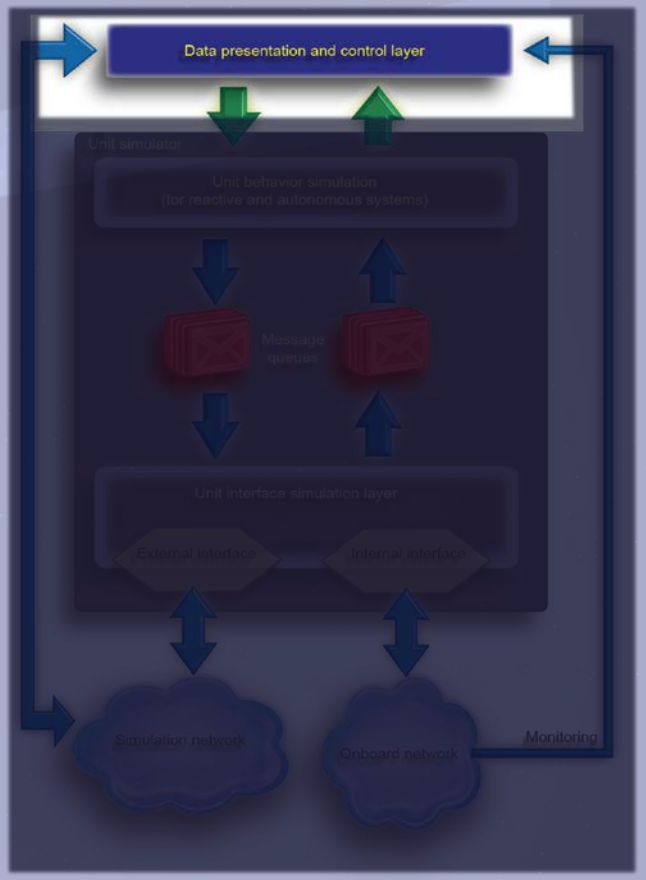
Simulation framework

- Based on the above problems we elaborated a conceptual simulation framework
- Architecture: five different layers
  - Each layer has its own well-defined functionality
  - Each layer has a unified and fixed interface to its neighbors
    - Fixed data structures
    - Internal implementation of a layer is flexible
- Desired benefits of using layer based architecture:
  - Simplifies the implementation by reducing the number of aspects
  - Simulation of a layer and the real implementation of it is interchangeable
- The core of the simulation is based on a well defined modeling language (DSML)



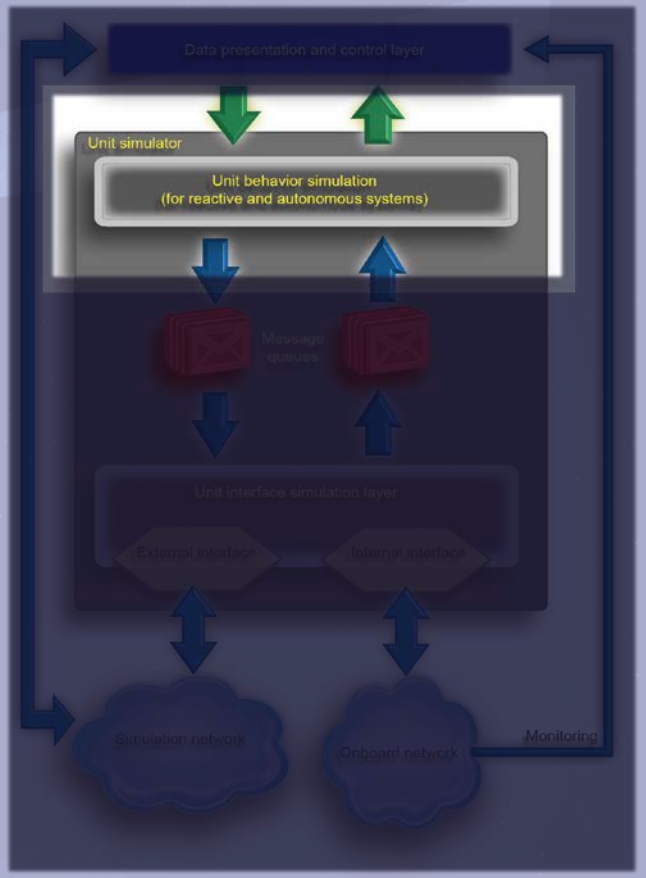
Our framework – Presentation and control

- Front-end of the simulator
- Provides access to internal states of the units and communication entities
  - Interface monitoring + HK in case of using real unit implementation
  - Fully available internal states in case of simulated unit
- Controls the simulation
  - Allows direct user controls
  - Automated test/simulation sequences by pre-stored scenarios
- Connected to:
  - Unit interface via control network
  - Unit interface monitoring via 'on-board' network
  - In case of simulating the unit: direct access to the simulator layer



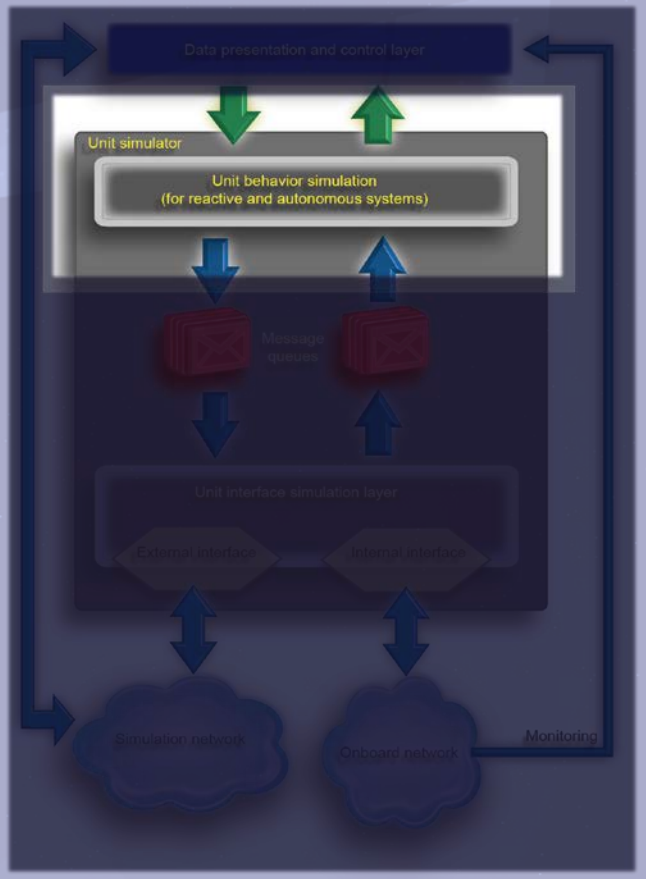
Our framework – Behavior simulation

- Encapsulates the unit behavior dependent methods
- Platform independently performs the simulation based on the functional model of the on-board unit
  - Independent of on-board communication interfaces
  - Independent of target platform
  - Independent of timing
  - Pre-defined level of abstraction
  - DSML
- The only layer which depends on the functionality of the real experiment / subsystem
- Available component:
  - Discrete event simulation (DES)



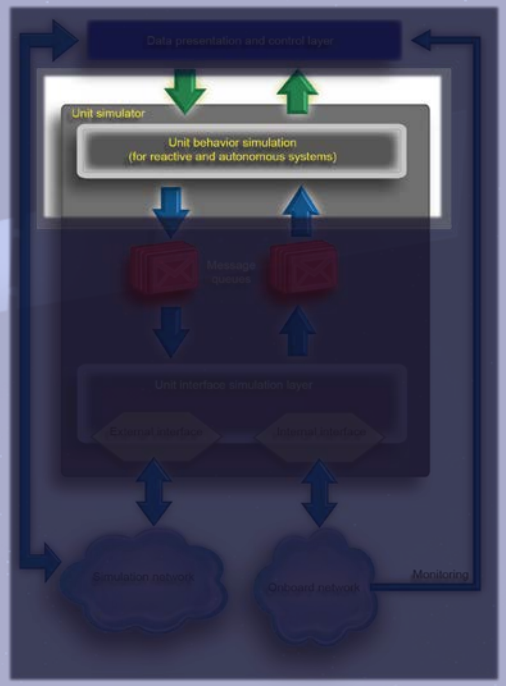
Our framework – Behavior simulation – DSML 1

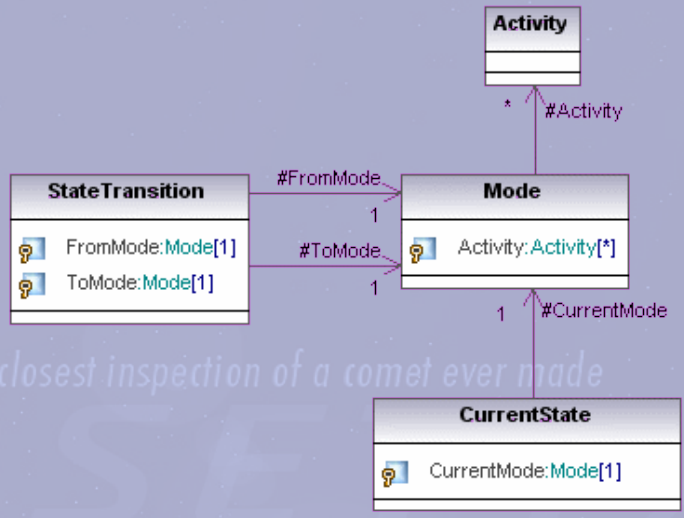
- Well-defined and suitable modeling language
  - Well-defined language syntax
  - Formalized semantic definition for the language elements
  - Enables using of formal methods for model checking and automation
    - Validation & verification
    - Test sequence generation
  - Dedicated for the modeling of on-board systems on the proper level of abstraction



Our framework – Behavior simulation – DSML 2

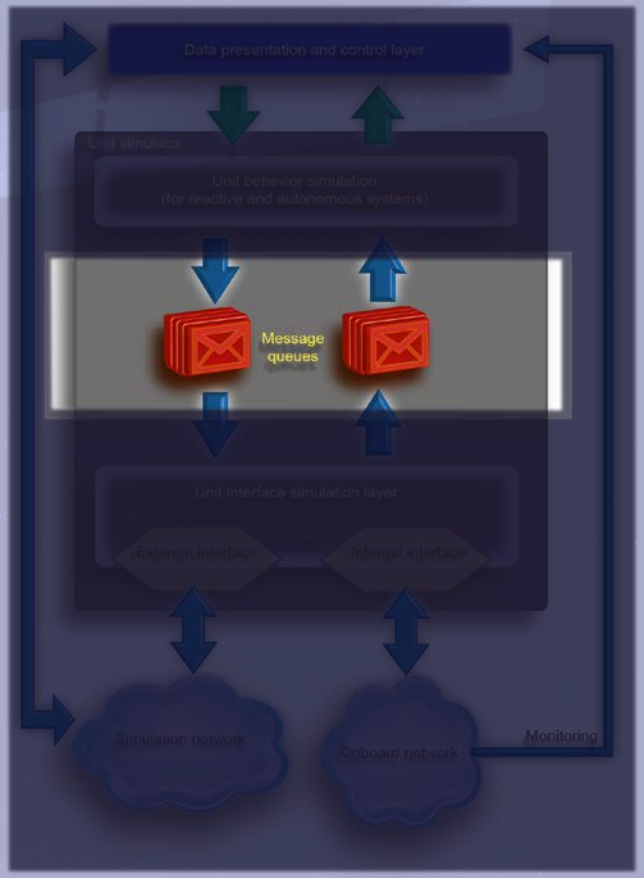
- Dedicated for the modeling of on-board systems on the proper level of abstraction
  - Functional simulation of the unit:
    - State based description of reactive systems
    - Inter state behavior description
  - Interface level description
    - Focus on communication sequences
- Graphical notation
- OCL constraints to meet the restrictions
  - Diversification between nominal and non-nominal states
  - Fault tolerance evaluation





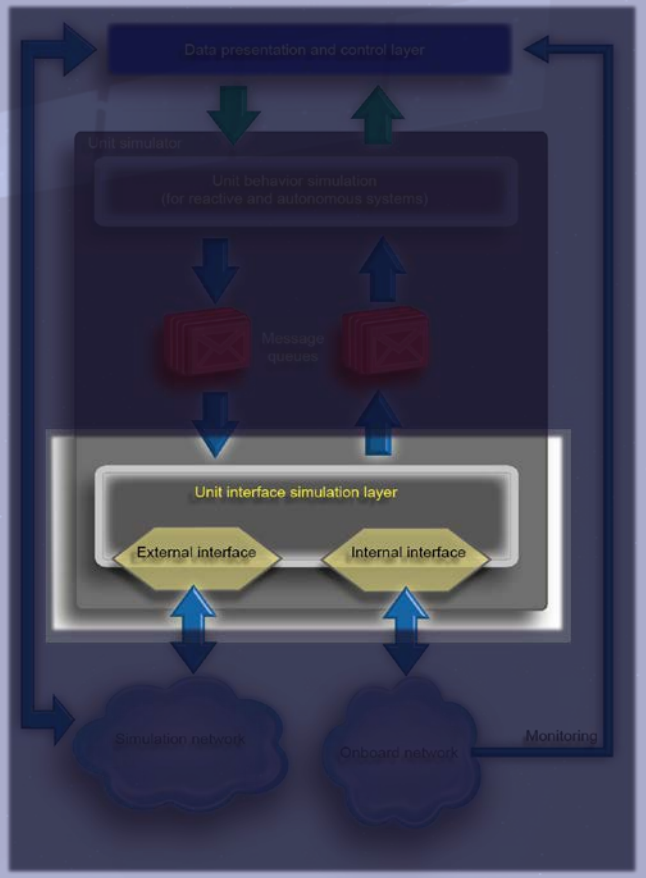
Our framework – Message broker

- Derived from message queues of the DES concept
- Detach the proper timing requirements from both the upper and underlying layers
    - Logical timing: Maintains the causality order of the message instances
    - Real-time operation: Performs the platform dependent timing of the communication
    - Except the low level timing constrains of the physical interface
- Possible message transformation



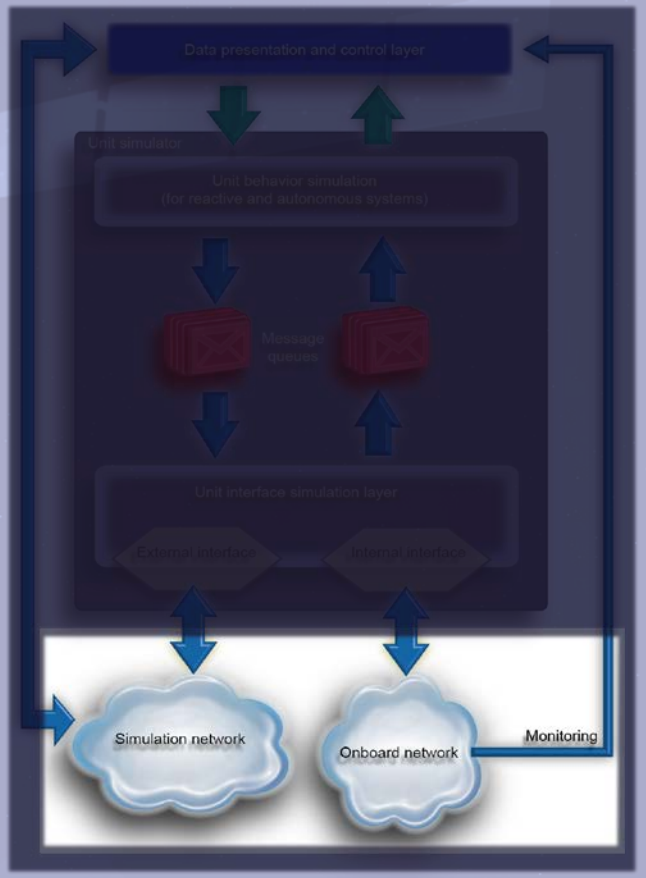
Our framework – Interface simulation

- Upper part:
  - Message broker client
  - Transforms the unified internal structures into real communication dependent message entities
- Lower part:
  - Different on-board equipments are connected via the internal interface
    - Logical
      - General data communication network (like Ethernet)
      - Transparent
    - Physical
      - Real on-board communication interface
      - Commercial interfaces like CAN, SpaceWire, MIL-STD-1553, …
      - Dedicated interface (unique control signals) should be supported
  - Environment related data can be injected via the external interface
    - Control and monitoring: simulation network



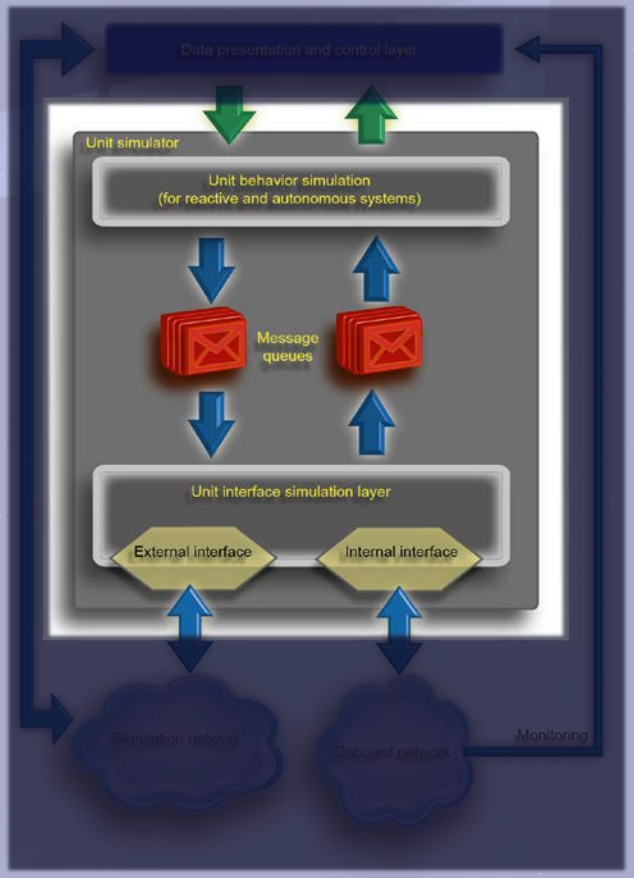
Our framework – Physical layer

- Two implementation of the on-board network:
  - Physical
    - Connects the different on-board equipments
    - Contains exactly the same data and control
      busses and dedicated signals as the on-board system
    - Identical signal and timing conditions as the on-board system
  - Logical
    - Provides a transparent tunnel over a common communication infrastructure (like Internet)
- Simulation network
  - Connects the units to the control and presentation layer



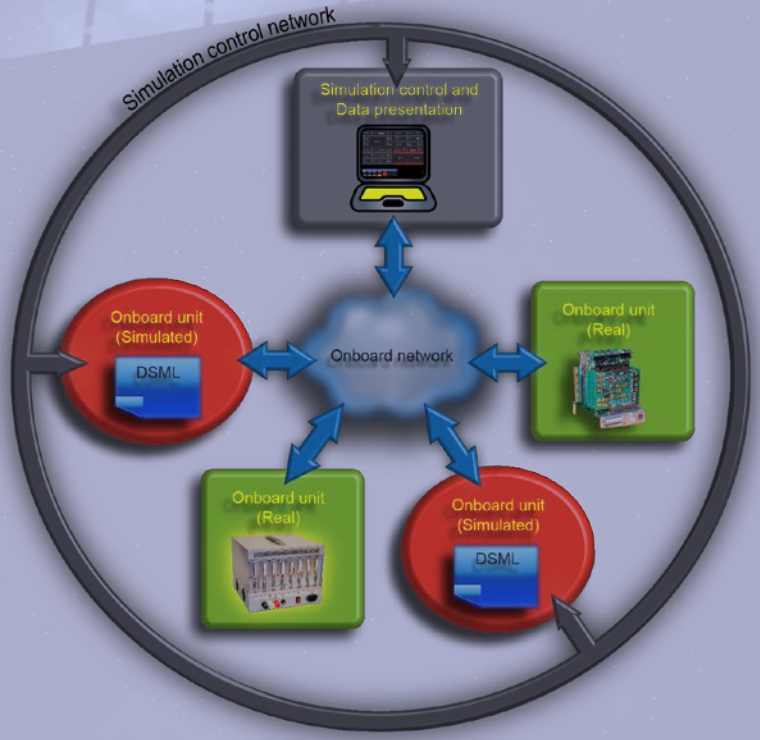
Use-cases 1

- The simulation layer along with the message broker and interface simulation layer can be used as a stand alone unit simulator
- From bottom-up the layers can be interchangeable with the real implementation as the development proceeds
- Early phase:
  - No HW available
  - Transparent interface layer over a public network
  - Formalizing the specifications by modeling
  - Outcome:
    - Check the fundamental concepts in early phase
    - Collaboration of different on-board units
    - Measuring the load on the communication interface



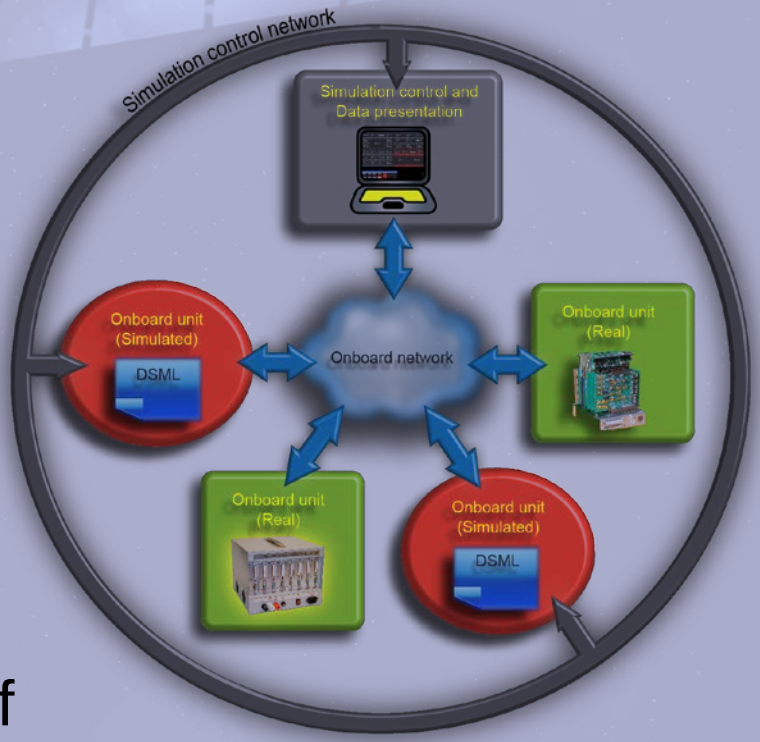
Use-cases 2

- Mid phase:
  - The on-board communication infrastructure is already defined
  - The interface simulator can be selected or has been implemented
  - On-board equipments interface model can be presented based on the earlier implemented behavior model
  - Hybrid simulation can be performed:
    - Both interface models and real unit implementations are present
    - Collaboration testing
    - Unit testing



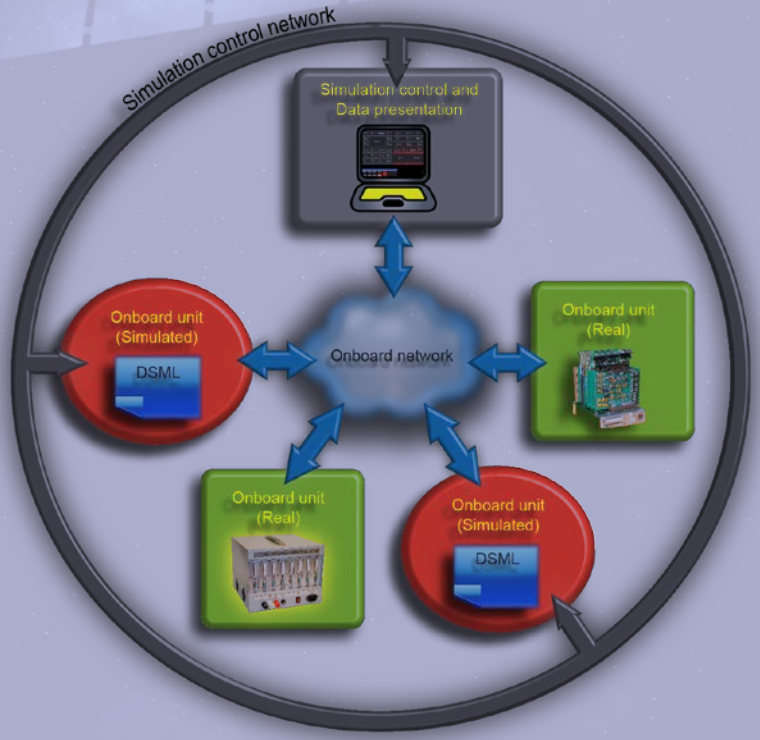
Use-cases 3

- Later phase:
  - The whole system can be simulated based either on models or real implementations of the units
  - Test sequences can be evaluated
  - Scientific scenarios can be tested
  - Teaching of the operational staff



Visionary use-cases

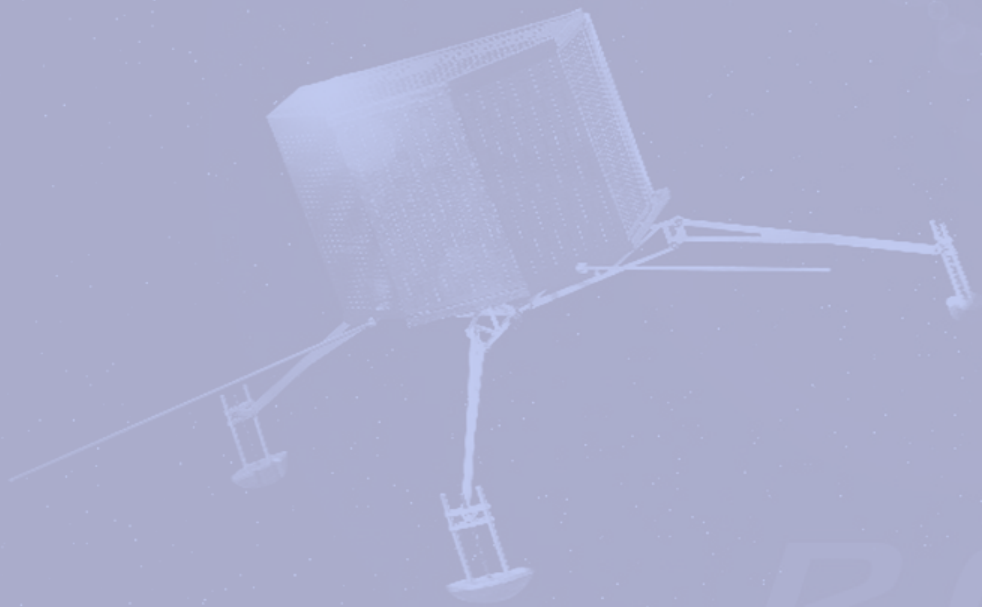
- Automatic verification of the implemented unit against its formal specification
- Automatic code generation based on the formal specification



Conclusion

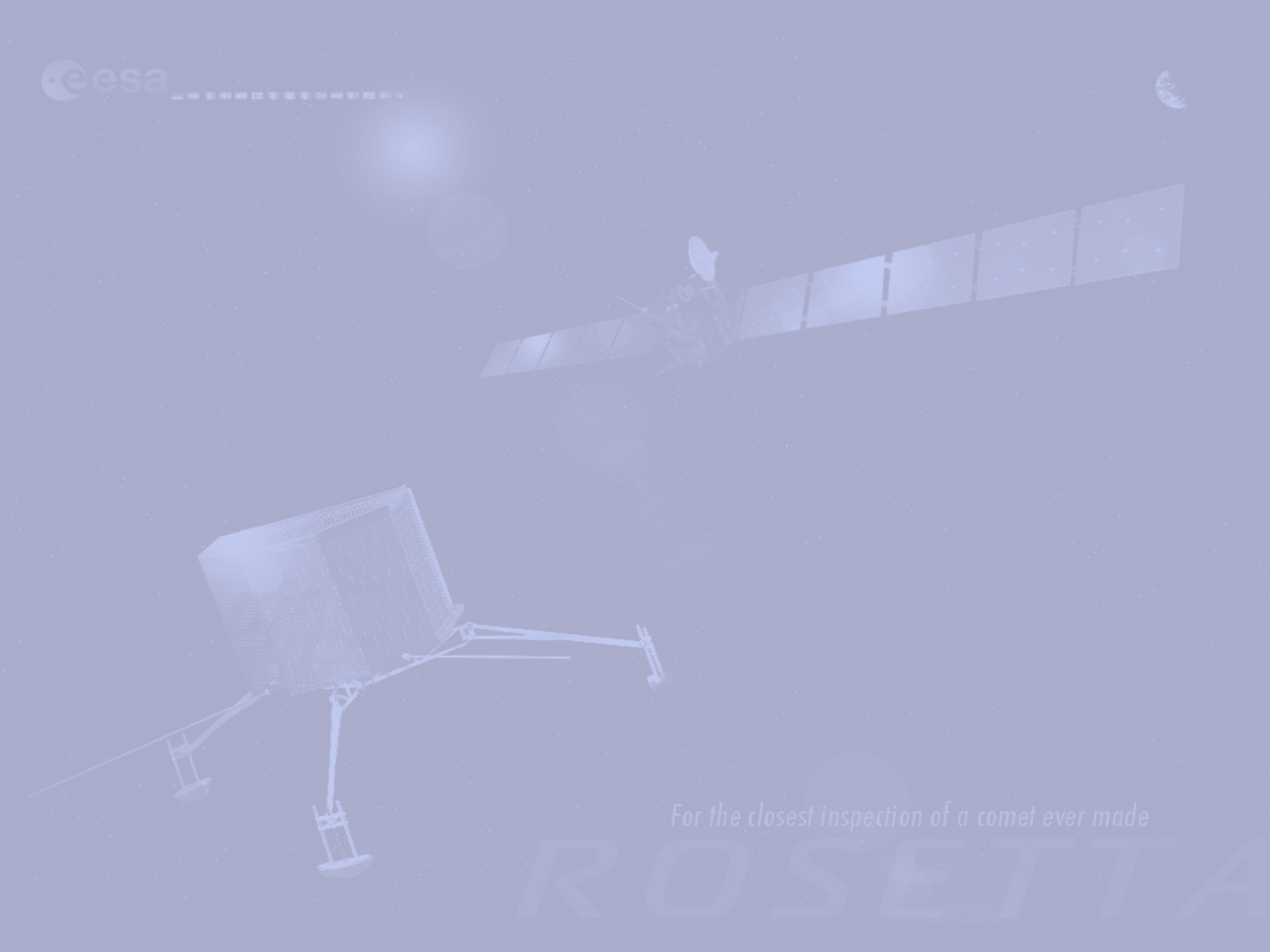
- Modular and scalable simulation framework has been presented
- Covers the whole lifecycle of an on-board equipment
- Already implemented parts
  - DSML and functional simulation
- A lot of work ahead

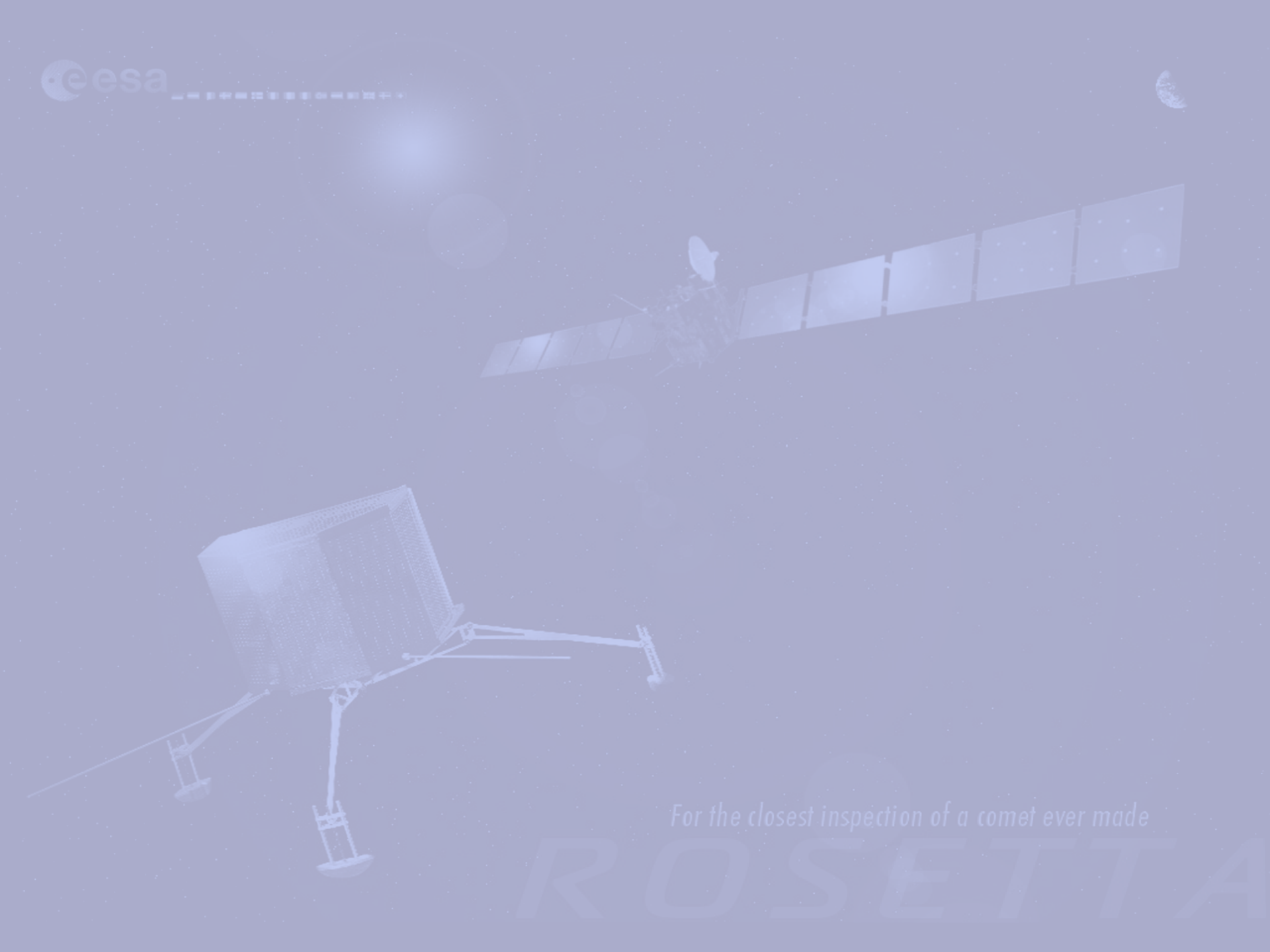
Thank you for your attention !

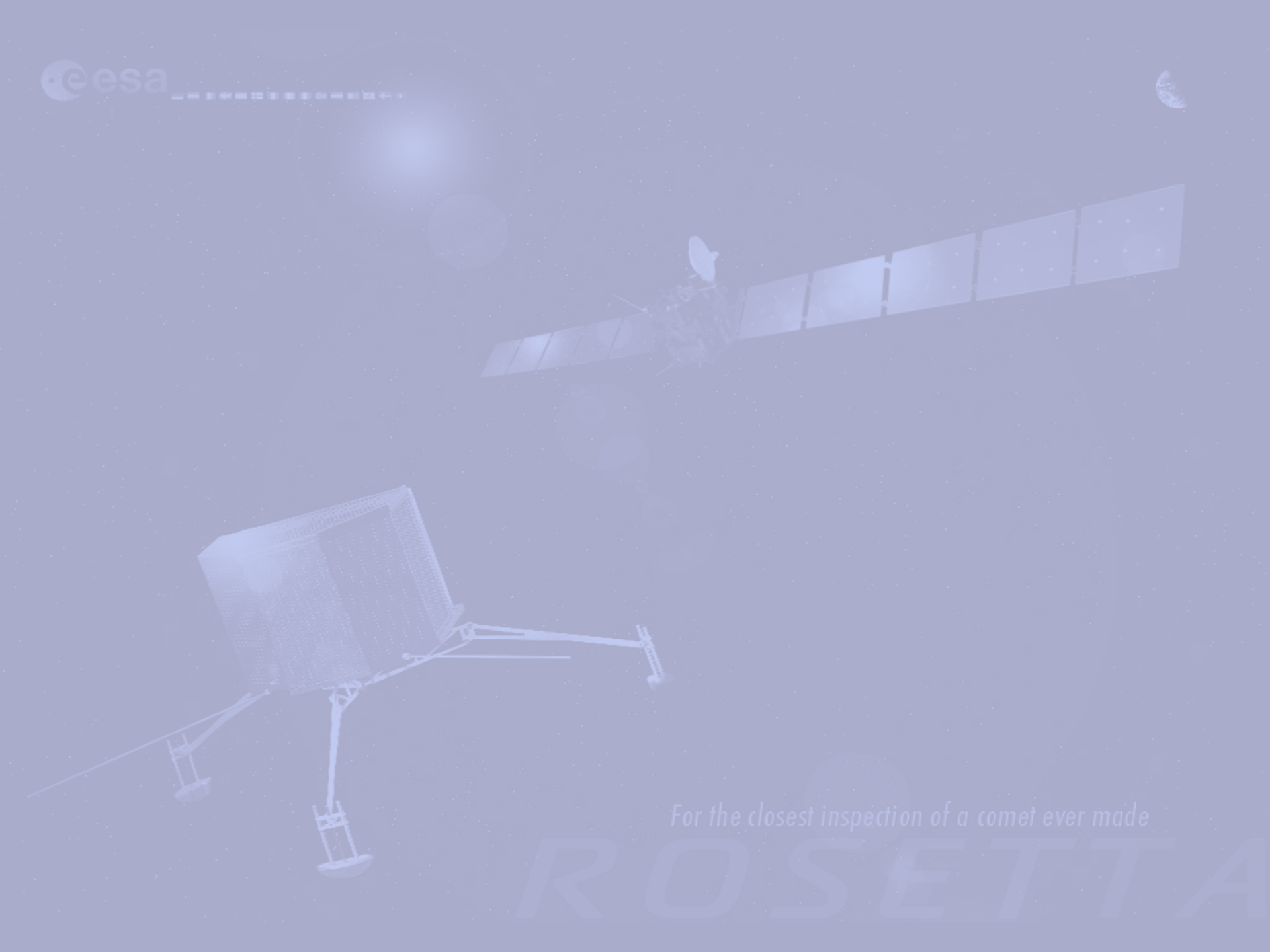


For the closest inspection of a comet ever made

ROSETTA


esa
For the closest inspection of a comet ever made
ROSETTA


esa
For the closest inspection of a comet ever made
ROSETTA


esa
For the closest inspection of a comet ever made
ROSETTA