MODEL-BASED SYSTEMS ENGINEERING IN SPACE ROBOTICS: THE ADE EXPERIENCE

Iulia Dragomir

GMV Aerospace and Defence, Spain

MBSE 2020, Noordwijk, Netherlands

This work is supported by the European Union's Horizon 2020 research and innovation programme under grant agreement #821988 (ADE)

OUTLINE

- The ADE Project
- ADE Capabilities
- Design and Validation Approach
- Conclusion

THE ADE PROJECT

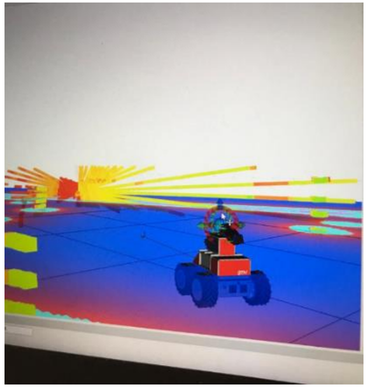
INTRODUCTION

- **ADE**: **Autonomous Decision Making in very long traverses** (https://www.h2020-ade.eu/, 2019-2021)
  - 10th operational grant (OG10) of the PERASPERA Programme (https://www.h2020-peraspera.eu/) dedicated to space robotics technologies
  - Focus on the development of key technologies for future autonomous space robotics exploration
  - Showcase of the system capabilities in a terrestrial demonstrator inspired from the Mars Sample Fetching Rover (MSR-SFR) mission
  - Spin-off to nuclear plant decommissioning activities
  - Development done by a Consortium of 14 partners from 7 countries, coordinated by GMV

SherpaTT (DFKI) in the Moroccan desert

Fuerteventura: analogue Mars representative environment

Foxizirc (GMV) in a nuclear plant representative environment



Nuclear plant map reconstruction

ON-BOARD AUTONOMY

- **The capability of a system to work without human interaction**
- There are multiple capabilities for a robot associated to autonomy such as:
  - **Autonomous mission planning**:
    - The robot is commanded via high-level goals
    - Examples: go to point (x,y) and take image, pick sample at (u,v) and take it to (z,t)
  - **Autonomous science detection**:
    - The robot detects serendipitous events and changes the current mission plan in order to gather science without abandoning the other activities
    - Example: detect a rock with specific characteristics and take pictures of it at a closer position
  - **Autonomous locomotion**:
    - The robot navigates a terrain in a safe way, reaching different objectives
  - **Autonomous device mobility**:
    - The robot performs complex operations in a safe manner with the devices that are available/connected to it
    - Example: use a robotic arm to pick a sample while avoiding collisions with the robot or the environment
  - **Autonomous safe operation in adverse conditions**:
    - The robot detects the presence of a harmful situation and reacts correspondingly
    - Example: detect that an on-board system is non-responsive and restart/reconfigure it to use a back-up system

ADE CHALLENGES

- Simulation of a **fully autonomous** MSR-SFR mission
- What is **expected**:
  - Autonomous long range navigation with high reliability (kms/sol)
  - Consistent data detection while avoiding un-detection of interesting data along mission path
  - Autonomous decision making capabilities in presence of conflicts (new goals to achieve, unexpected events, etc.)
  - Safety first in presence of failures or any unexpected events
  - Higher technology readiness level to prepare its reuse in future missions
  - Flexible-purpose robotic system design that could be used in terrestrial applications

ADE CAPABILTIES

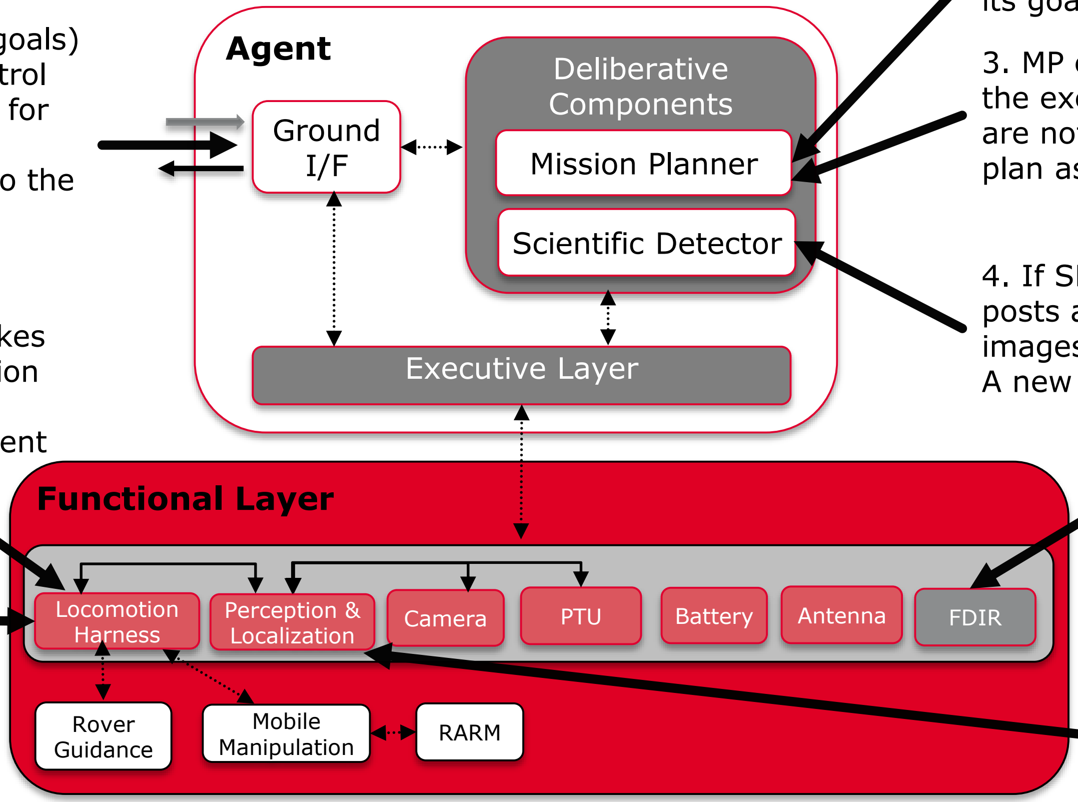
ON-BOARD AUTONOMY FOR PLANETARY EXPLORATION

1. High level commands (goals) are sent from Ground Control Station through I/F. Goals for detection are sent to SD, remaining goals are sent to the MP.

2. MP produces a plan to fulfil all its goals.

3. MP executes the plan and monitors the execution. If the plan constraints are not satisfied, it generates a new plan as appropriate.

4. If SD detects an event of interest, posts a goal to the MP to take images/analyze the detected object. A new plan is generated by MP.



5. Locomotion Harness takes us to the desired destination point (RG), with coupled rover-robotic arm movement (MM)

6. And performs with the robotic arm (MM) complex (pick/drop) operations

7. Visual perception and localization supports LH (rover, robotic arm) and SD capabilities

8. FDIR ensures continuous safe execution in adverse conditions.

AI MODELS

- **Planning Domain Definition Language (PDDL) model**:
  - Provides an abstract representation of the world to reason on
  - Used by the **Mission Planner** to transform high-level goals (e.g., move sample from one location to another) to low-level operations (e.g., move, pick, move, drop)
    - Provides the **autonomous mission planning capabilities**
    - Component agnostic to the PDDL model using computational logic (LTL, SAS+) as internal representation
    - Model reusable for other missions working on the same representation
- **Neural Network models**:
  - Provides simplified reasoning functionalities with respect to well specified tasks
    - Many models (classes) are obtained via training, one per each task
    - Models reusable for other missions that require the classification of the same events of interest
    - Subject to false positives, intensive training and parametrization needed
  - Used by **Scientific Detector** to detect novelty and classify the detected events
    - Provides the **autonomous opportunistic science capabilities**
  - Used by **Soil Traversability** (offline component) to provide an estimate of the soil traficability (e.g., no sand) and the impact on the rover

EXECUTION MODELS

- **Discrete time/discretized control models**
  - Describes the execution of components in terms of ticks/periods
  - Used by the **Agent** to control the consistent execution of the entire on-board software
    - Main controller embedding a hierarchy of control loops based on the Sense-Plan-Act paradigm
    - Configurable component with automated code generation applicable to any mission
  - Used by **Rover Guidance** to move the rover in a safe manner from one location to another
    - Provides the **autonomous locomotion capabilities**
    - Reusable control model/design principles, but tailored to the rover configuration (e.g., 4 wheel vs 6 wheel rover)
  - Used by **Mobile Manipulation** to reach samples with a coordinated approach of rover and robotic arm movements
    - Provides the **autonomous device mobility capabilities**
    - Reusable control model/design principles, but tailored to the rover model
- **Timed model**
  - Describes the execution of components in continuous real-time
  - Used by **FDIR** to detect hazardous situations and reconfigure the system
    - Enables the **autonomous safe operation in harmful conditions**
    - Dependent on the mission scenario
    - Obtained from a **formal specification** in **BIP** of the **system** and the **properties** to satisfy

ROBOTICS MODELS

- **Geometrical model**
  - Provides the static configuration of the rover
  - Consists of **Computer-Aided Design** (CAD) **model** and **kinematics model**
  - Used by **Mobile Manipulation** for collision avoidance
  - Used by **Visual Perception and Localization** to build the relevant environment for the mission goals
    - Supports the Scientific Detector, Rover Guidance and Mobile Manipulation
    - Provides an environment model and the rover position in the environment
  - Used by **Ground Control Station** to illustrate the rover and its operations in the environment for assessment purposes
    - Commands and monitors the system and assesses the results
  - Used by **Rover Simulator** to illustrate the rover and its operations in the environment for virtual execution and debugging purposes of the entire system

- **Dynamics model**
  - Provides the actuator behavior (based on physical laws)
  - Used by **Rover Simulator** to compute the rover reaction to the commands from the control system
    - A dynamics model of the physical environment interacting with the rover is also embedded

ENVIRONMENT MODELS

- **Orbital Map**
  - Representation of the global environment in which the rover performs its mission
  - Used by Rover Guidance to classify the difficulty of the terrain for traverses
- **Digital Elevation Map**
  - Representation of the local environment (e.g. 8m around the rover)
  - Used by Rover Guidance and Mobile Manipulation for their capabilities
- **3D environment model**
  - Global graphical reconstructed representation of the environment
  - Used by **Ground Truth** for the rover position validation

JOANNEUM RESEARCH

DESIGN AND VALIDATION APPROACH

DESIGN CHALLENGES

- A **complete framework** providing all the desired capabilities for on-board autonomy
  - Using at least one model for each component
  - Integrates on a **plethora of different technologies and formalisms**
- **Developing** a system that uses the different features is a complex task:
  - Correct interfaces between the different components
  - Mix of paradigms and computational models
  - Non-deterministic and deterministic behaviors (e.g., deliberative vs control)
  - Real-time safe behavior
  - Avoid deadlocks (can be costly to correct from ground)
  - Deployment on different HW components
- To tame the complexity of such development one **needs**:
  - A system design process, e.g., the waterfall model
  - Models which describe the system architecture (SW and HW) and interactions, possibly refined towards implementations
  - A model-driven engineering tool encompassing the features above, while automating some static checks, code generation, etc.

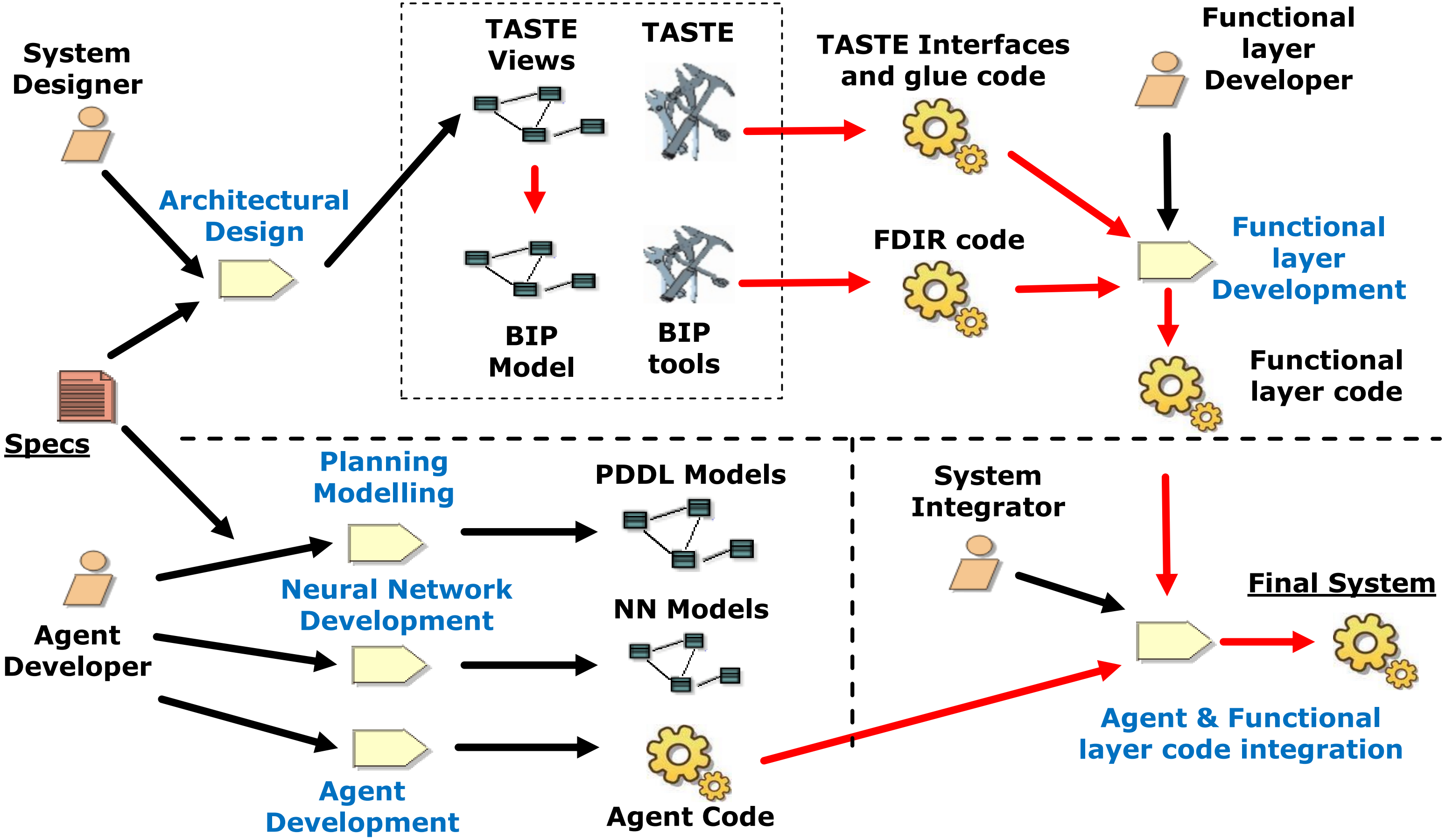
DESIGN AND VALIDATION CHOICES

- Use the **V development model**
- Perform a complete but **high-level system design with UML**
  - Identify the dependencies and interfaces between the components
  - Readable by most of the partners
- **Design** the **integrated system with TASTE toolset**
  - Define and type the interfaces
  - Update the components design according to the computational model
  - Group components based on common functionalities to have a smooth integrated design
- Design, implement and validate the **components independently**
- **Generate** the **executable(s)**
- **Validate** the system with **extensive testing**
  - Several testing levels, from unit testing to field trials
  - No formal V&V, the system too complex

THE TASTE TOOLSET

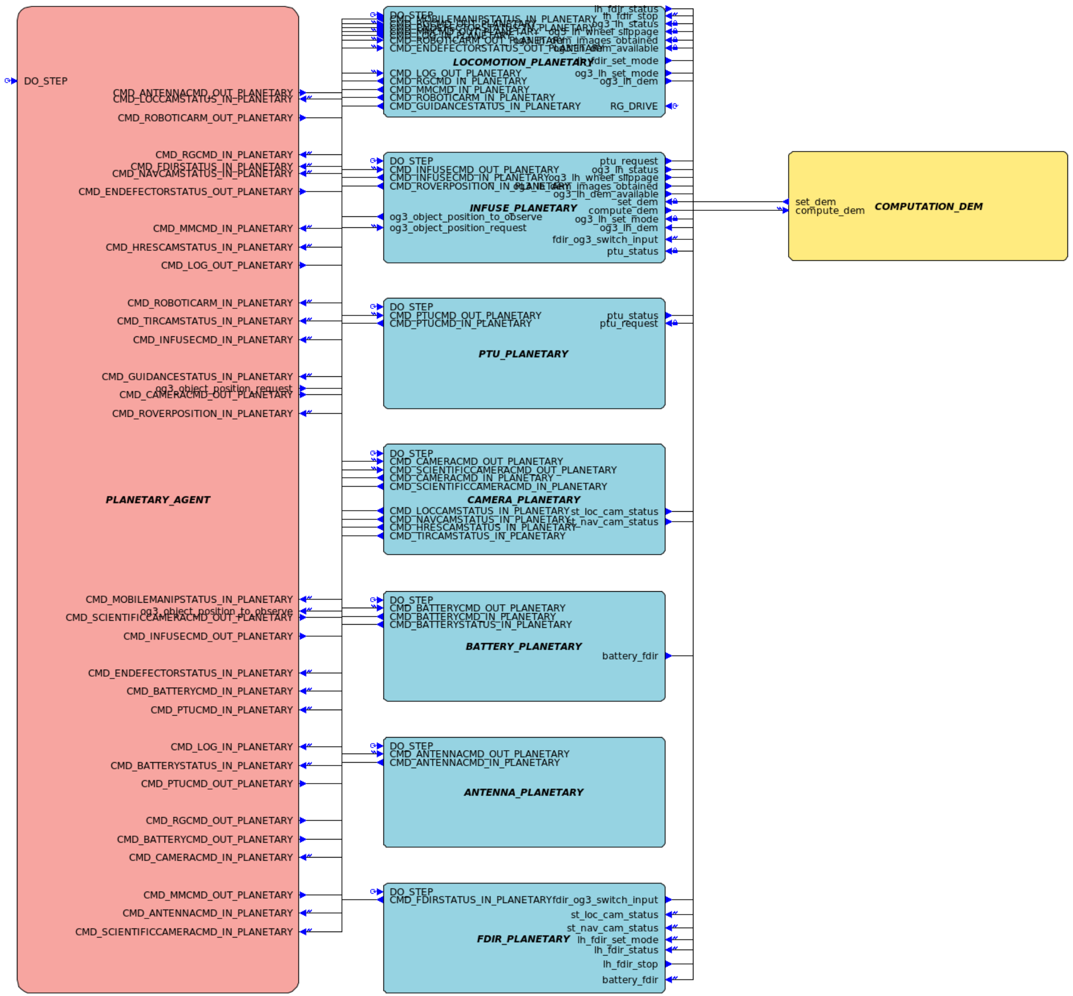
- Targets the **model-based development of heterogeneous, reactive, discrete embedded systems**
- A **TASTE design** consists of datatypes, architecture and behaviour modelling, deployment and tasks scheduling
- Some **features**:
  - Well-formedness and typing analysis of a design wrt the used modelling formalisms
  - Real-time scheduling analysis (e.g., Cheddar)
  - Code generation and deployment (in C/C++,Ada) wrt an execution platform
  - Simulation, debugging and testing
- Some **inconveniences**:
  - **Lack of a formal specification** of the modelling, programming and computational models used (opaque to the system designer)
    - Leads to interpretation errors that are detected and handled during simulation/testing
  - **Tied to the Ravenscar profile**
    - Important for space systems, but
    - cumbersome for the design of robotics systems since tasks cannot be dynamically generated
  - **Inefficient memory assignment** (for sporadic interfaces the maximum heap is assigned)
  - **Limited usability** for complex systems (e.g., for the graphical editors)

ADE DESIGN METHODOLOGY (PARTIAL)



TASTE DESIGN

- 9 TASTE functions
- 58 interfaces:
  - 9 cyclic
  - 8 protected
  - 41 sporadic
- Deployment on an Intel i7 and Arm A53



CONCLUSION

SUMMARY

- The **ADE framework for on-board autonomy**
  - Commanding a robot from ground via high-level goals transformed on-board into operations
  - Plan the daily operations from the goals requested by ground
  - Opportunistic science while performing other activities
  - Complex Mars-representative rover operations (traverse, coupled traverse-robotic arm movement, pick/drop) in safe conditions
- The **ADE approach**
  - Model-based development approach, however code-centric
  - Validation by extensive testing, no formal V&V
- **Validation** in two scenarios
  - Planetary scenario inspired from the MSR-SFR mission
  - Nuclear scenario inspired from nuclear plant decommissioning
- Both the framework and approach are **generic** enough to be tailored to different types of robotic missions (orbital, deep space probe, etc.), but also to robotics terrestrial applications

CONCLUSION (1/2)

- **Very complex design process**
  - No known means to simplify it
  - Adding new components to the system (on-board, on-ground) increases the complexity
- **Requires knowledge in multiple technologies**
  - E.g., domain and theory, computational model, programming language
  - Very difficult to find system designers/integrators with such capabilities
- Preferable **automation** of as many steps as possible by e.g., code generation
  - These steps should be correct in the formal sense
- The **tool/technology used** for system design **drives the implementation of the system and of different components**
  - Transformations at semantic level or artificial constructions are sometimes required to be able to integrate components

CONCLUSION (2/2)

- (Space) Robotics is an **inter-disciplinary field**: robotics engineer, software engineer, systems engineer, …
  - **A model has different meanings** for different communities, e.g., physical/kinematics model, software architecture
  - The **lack of unified language** is challenging in such domains
- Current system design practices are not tailored for robotics system development
  - **Model-based design** is **a viable solution**, but …
  - … the systems are complex, so to well-define an integrated model and its interfaces is a difficult task
  - … the system designer has to deal with many legacy models and code
  - … there is not one model, each component has one in a possibly different computational model
    - Multi-view modelling and views consistency could be considered for heterogeneous systems
  - … new technologies emerge, e.g., neural networks, that are not incorporated in a modelling process or have a different design process
- **Tools** are needed **for consistent system design** process taming the growing complexity and the heterogeneity of technologies used
- Yet, different communities embrace the use of model-based design and apply it at different levels
- It is **possible to build valid complex systems** even without end-to-end off-the-shelf solutions

www.gmv.com

THANK YOU