SIRIUS-DV : CNES NEW FLIGHT DYNAMICS ALGORITHMS

Iván LLAMAS, Yannick TANGUY, Michel LACOTTE, Jean-Jacques WASBAUER

15/03/2016
SUMMARY

- INTRODUCTION
- THE SCENARIO
- SCENARIO PROCESSINGS
- PRODUCTIVE PROPAGATOR
- DEVELOPMENT PROCESS

07/03/2016
INTRODUCTION (I) : MAIN GUIDELINES

- New approach in the development of a FDS
  - Clear division between computation layer and FD supporting services
  - 2 different contracts

- Development driven by CNES
  - Architecture definition
  - Decomposition and reuse of functions

- FD services
  - A service can invoke other sub-services, and it only knows their interfaces
  - Stateless
  - Simple interface: Inputs (common arguments + specific parameter) + Response

- Why?
  - Flexibility
  - Easy to evolve
  - Limited coupling between computation layer and CC solutions
INTRODUCTION (II) : BUILDING A FDS

FDS

FD Applications

Widgets IHMs

Framework FDS

FD algorithms

PATRIUS

DataStore

Visualization

Infrastructure

Operations Services

Mission

Automation

Code generator

ISIS product line

SIRIUS product line

Final product for a mission
INTRODUCTION (III) : ASSEMBLING A FD SERVICE
SCENARIO (I) : THE CORE DATA

- Complex and structure data, without associated operations
- Describes the state and the evolution model of one satellite, over the whole lifetime
- Evolution in parallel (not independent) of the defined domains:
  - Trajectory
  - Attitude
  - Mass, inertias
  - ...
- Composed by:
  - Activities: lowest level, per domain (trajectory, attitude)
  - Feeds: Temporal axis, per domain
  - Blends: Synthesis of the feeds, best global vision
- Its structure is adapted/defined for every mission
SCENARIO (II) : EXAMPLE

TIME

Trajectory Blend

Fragment 1

Numerical Activity 1

PV Ephemeris Activity

Numerical Activity 2

Fragment 2

Reference Orbit Activity

Keplerian Activity

Trajectory Feed 1

Trajectory Feed 2

Impulsive Man Feed

Continuous Man Feed

Man1

Man2

Man3

Maneuver Group
SCENARIO PROCESSINGS (I) : GENERAL CONCEPTS

- Simulation of the states of the scenario

- Several levels:
  - Blend processing
    » By fragments
    » By priority
    » By user selection
  - Feed processing
  - Activity processing

- Divided by the domains of the scenario
  - Trajectory, attitude, MCI, maneuvers, tanks, thrusters and solar arrays
SCENARIO PROCESSINGS (II) : TYPES

● Trajectory processing: Position/velocity at T
  ✦ Blend : Searches the fragment and invokes the feed processing
  ✦ Feed : Searches the activity (error if not found) and invokes the activity processing
  ✦ Activity : Depends on type (numerical, keplerian, ephemeris,…)

● Attitude processing: Attitude (and derivatives) at T
  ✦ Blend : Searches the feed by priority
  ✦ Rest as trajectory, but no error in feed processing

● Maneuver processing : Force/delta-V produced by the maneuver at T
  ✦ Blend : Searches the maneuver by user choice.
  ✦ Feed : Searches the activity (no error) and invokes the activity processing

● MCI processing : Mass, center of gravity and inertia at T
● Tank processing : Tank state (propellant mass, Pressure, Temp) at T
● Thruster processing : Flow rate, force, throughput of thruster at T
  ✦ Similar logic as for trajectory
In charge of producing output ephemeris (list of time-stamped data):
- At dates, as defined by the input list of DateDescriptor
  - Fixed dates
  - Given interval
  - Interval of a phenomenon
- At events, as defined by the input list of EventDescriptor

The output data are defined by OutputParametersDescriptor
- Reference to scenario descriptors (blend/feed) needed in the computation
- Dedicated « part » in charge of filling the data

Algorithm:
- The output parameters define the data from scenario that are needed.
- The scenario processings are in charge of simulating (propagation) the scenario state with the required descriptors (feed/blend) at desired date.
- The part is invoked with this scenario state to compute and fill the output parameter
PRODUCTIVE PROPAGATOR (II)
DEVELOPMENT PROCESS

- Agile/SCRUM with 4-weeks sprints
- CNES builds the model (data, interfaces, requirements).
- The templates of the code are generated (using a code-generator).
- The code is implemented, tested and documented by GMV.
- CNES checks if everything is OK and closes the story.
Thank you for your attention 😊

For further details:
Iván LLAMAS
Flight Dynamics and Operations BU - GMV
e-mail : illamas@gmv.com