Computer-Aided Software Design for Spacecraft Guidance, Navigation and Control

Jean de Lafontaine
President
NGC Aérospatiale Ltée
NGC Aerospace Ltd

2008 IEEE Multiconference on Systems and Control
2-5 September 2008, San Antonio, Texas, USA
The European Space Agency (ESA) identified space autonomy as the next enabling technology for:

- terrestrial missions (Earth observation: environment, security)
- planetary exploration missions (Mars, Moon, asteroid, comet)
High resistance:
- will not work, will lose control of the spacecraft
- will increase development cost

ESA upper management: Let try and see

Initiated the PROBA programme in 1990’s
- PRoject for On-Board Autonomy ("Probare" = "let’s try")
- demonstrate the benefit of autonomy in space
- demonstrate new technologies, new S/C development methods
- launch the PROBA-1 spacecraft within 2 years after start of Phase B
PROBA-1: Earth-Observation Mission

- launched in October 2001
- 2-year mission
- still successfully operating after 7 years

- 1st fully autonomous ESA spacecraft
- 1st with automatic flight code generation
- 1st with variable-gain Kalman filter
- 1st with complete on-board guidance
- 1st with quaternion-based multivariable gyroless + sliding-mode controller for large-angle manoeuvres
PROBA-2: Sun-Observation Mission

- to be launched in April 2009
- same autonomy as in PROBA-1 + GNC technology experiment
- magnetic-based state estimation with unscented Kalman Filter
PROBA-3: Formation-Flight Mission

- to be launched in 2013
- Coronagraph S/C and Occulter S/C on elliptical orbit
- high-accuracy position and attitude determination & control
NGC Aerospace was or is currently the contractor for the development of the autonomous GNC system for:

- PROBA-1
- PROBA-2
- PROBA-3 (in negotiation)

Realisation of these complex on-board software would not have been possible without the use of computer-aided software development tools.
OBJECTIVE & OUTLINE

✧ OBJECTIVE
To demonstrate the need for, and the characteristics of, computer-aided software design for flight-code generation via the particular case of the PROBA flight software

✧ OUTLINE
- The Need: Trends in Spacecraft Control System Design
- The Example: PROBA
- The Process: The PROBA Software Development
- The Lessons Learned and the Benefits
- Conclusions
Navigation (NAV)
- the determination of the current dynamical state of the vehicle
- by extension: the determination/calculation of environmental variables (Sun position, Earth attitude, Earth target position)

Guidance (GDC)
- the determination of the difference between the estimated state from NAV and the desired state from the Mission Manager
- the computation of the time history of the desired state

Control (CTL)
- the computation of the required actions that will bring the estimated state coincident with the desired state in a stable manner and compliant with performance specifications
THE NEED AND THE TRENDS
The Need: Trends in S/C GNC Design

What the Users need

TREND IN MISSION DESIGN

The tools we need

TREND IN FLIGHT COMPUTERS

TREND IN GNC ALGORITHMS

The methods we need

TREND IN FLIGHT S/W DEVELOPMENT

NEW GENERATION OF MISSION AND SPACECRAFT
The Need: Trends in S/C GNC Design

What the Users need

TREND IN MISSION DESIGN

The tools we need

TREND IN FLIGHT COMPUTERS
TREND IN GNC ALGORITHMS

The methods we need

TREND IN FLIGHT S/W DEVELOPMENT

NEW GENERATION OF MISSION AND SPACECRAFT

2008 IEEE Multiconference on Systems and Control, 2-5 September 2008, San Antonio, Texas, USA
Smaller spacecraft on low-cost missions

- increased needs in environment monitoring and security
- formation flight of many S/C instead of large S/C
- availability of cheaper “piggy-back” launches
- smaller, cheaper sensors and actuators

⇒ reduction of development costs and operational costs

autonomy
Spacecraft autonomy ⇒ GNC autonomy

- cost: smaller staff at the ground station for operations
- efficiency: quick correction of in-flight anomalies
- accuracy: real-time state measurements vs predicted
- no choice: in some missions, the signal time-of-flight precludes closing the control loop via the Earth station (e.g. Mars landing)

⇒ ‘intelligent’ flight software ⇒ larger development costs
CONCLUSIONS

- lower operational costs ⇒ spacecraft autonomy
- lower development costs

but...

- intelligent on-board software
- higher software development costs

\$(one \ line \ of \ code \ in \ space) = 2 \times \$(same \ line \ of \ code \ on \ ground)
The Need: Trends in S/C GNC Design

What the Users need

- TREND IN MISSION DESIGN
- TREND IN FLIGHT COMPUTERS
- TREND IN GNC ALGORITHMS

The tools we need

The methods we need

NEW GENERATION OF MISSION AND SPACECRAFT

2008 IEEE Multiconference on Systems and Control, 2-5 September 2008, San Antonio, Texas, USA
NAVIGATION: the traditional
- sensor output feedback for attitude
- ground-based orbit determination

NAVIGATION: the trend
- state feedback
- Extended Kalman Filter, Unscented Kalman Filter
- autonomous star sensor, GPS-based orbit determination

⇒ sensor delay recovery, sensor outage compensation, measurement interpolation, sensor fusion
⇒ more complex on-board software
TRENDS IN S/C GNC ALGORITHMS-2

✧ GUIDANCE: the traditional
  - ground-based state-trajectory computation
  - uplink of polynomial coefficients for reference attitude

✧ GUIDANCE: the trend
  - on-board computation of reference attitude profile
  - on-board computation of reference trajectory profile

⇒ autonomous on-board decision
⇒ more complex on-board software
TRENDS IN S/C GNC ALGORITHMS-3

✧ CONTROL: the traditional
  • decoupling assumption: one controller per axis ⇒ SISO
  • PID controller, lead-lag controllers, flexibility filters

✧ CONTROL: the trend
  • multivariable control of coupled dynamics ⇒ MIMO
  • LQG/LQR control, robust control, adaptive control, predictive control, nonlinear control, sliding-mode control
  • nonlinear dynamic inversion, robust dynamic inversion

⇒ better performance of ‘intelligent’ algorithms
⇒ higher design complexity, higher controller complexity
⇒ more complex on-board software
CONCLUSIONS

- intelligent GNC software with...
- better performance
- better autonomy

but...
- more complex on-board software
- higher software development costs
- more demanding on-board computer resources

next
The Need: Trends in S/C GNC Design

What the Users need

TREND IN MISSION DESIGN

The tools we need

TREND IN FLIGHT COMPUTERS
TREND IN GNC ALGORITHMS

The methods we need

TREND IN FLIGHT S/W DEVELOPMENT

NEW GENERATION OF MISSION AND SPACECRAFT
TRENDS IN FLIGHT COMPUTERS-1

✧ Hardwired Control System
  • analogue link between sensors and actuators
  • analogue/hybrid computer for verification & validation
  • no in-flight reprogramming
  • limited to simple input-output relationships

✧ Microprocessor-based GNC System
  • digital link between sensors and actuators
  • digital computer for verification & validation
  • in-flight reprogramming possible
  • complexity of the software only limited by memory, computing power and ability to validate and verify the software before flight
  • 10 MIPS (2001), 40 MIPS (2006), 100 MIPS, 500 MIPS
CONCLUSIONS

• space-qualified computers are more powerful
• can cope with more complex GNC algorithms

but...

• more complex on-board software remains
• higher software development costs remain
The Need: Trends in S/C GNC Design

What the Users need
- TREND IN MISSION DESIGN

The tools we need
- TREND IN FLIGHT COMPUTERS
- TREND IN GNC ALGORITHMS

The methods we need
- TREND IN FLIGHT S/W DEVELOPMENT

NEW GENERATION OF MISSION AND SPACECRAFT
TRENDS IN FLIGHT CODE DEVELOPMENT

Evolution roughly organised into 4 generations

✧ 1st generation:
  - paper design, home-made computer tools for validation
  - hand-coding in low-level language (Assembler)
  - limited flight-code validation with flight computer

✧ 2nd generation:
  - computer-aided tools for design/validation (CASE tools)
  - hand-coding at high level (C or ADA)
  - home-made simulator for flight-code validation

✧ 3rd generation:
  - CASE tools for design/validation
  - CASE tool for automatic flight-code generation
  - home-made simulator for flight-code validation

✧ 4th generation (the PROBA generation):
  - single CASE tool from conceptual design to flight-code validation
The Need: CONCLUSIONS

What the Users need
- TREND IN MISSION DESIGN
  - low dev. cost
  - low ops. cost
  - autonomy

The tools we need
- TREND IN FLIGHT COMPUTERS
  - more powerful CPU
- TREND IN GNC ALGORITHMS
  - intelligent GNC algo
  - larger OB S/W
  - more CPU power
  - higher dev. cost
  - higher V&V cost

The methods we need
- TREND IN FLIGHT S/W DEVELOPMENT
  - Computer-aided tools to reduce development and validation costs
THE EXAMPLE
THE PROBA-1 MISSION

✧ The PROBA-1 Mission

- demonstration of autonomy in space
- Earth observation with two instruments
  - hyperspectral camera (color) @ 20m
  - high-resolution camera (black & white) @ 4m

ORBIT
615 km altitude

SPACECRAFT
95 kg, 600 X 600 X 800 mm (*a big TV*)
40 W average power (*a light bulb*)

LAUNCH & OPERATION
Launched 22 October 2001
Still operating successfully
THE PROBA-1 MISSION

TARGET

2D DETECTOR ARRAY
CCD-TYPE CAMERA
B&W CAMERA @ 4m
THE PROBA-1 MISSION

SINGLE-LINE DETECTOR ARRAY
PUSH-BROOM POINTING
(pointing to centre of the Earth)

COVERED AREA
THE PROBA-1 MISSION

SINGLE-LINE DETECTOR ARRAY
REDUCED-SPEED PUSH-BROOM

COVERED AREA

2008 IEEE Multiconference on Systems and Control, 2-5 September 2008, San Antonio, Texas, USA
THE PROBA-1 MISSION

CAMERA Line of Sight

20 km

SPECTROMETER POINTING

5 CONSECUTIVE IMAGES
THE FOUR MOTIONS TO TAKE INTO ACCOUNT

Presentation at RMC, Kingston, 8 October 2002
2008 IEEE Multiconference on Systems and Control, 2-5 September 2008, San Antonio, Texas, USA
THE FOUR MOTIONS TO TAKE INTO ACCOUNT

SPACERCRAFT TRANSLATION (measured)

EQUATORIAL PLANE

GROUND TRACE OF ORBITAL PLANE

EARTH TARGET
THE FOUR MOTIONS TO TAKE INTO ACCOUNT

SPACERACRAFT TRANSLATION
(measured)

SPACERACRAFT ROTATION
(measured & controlled)

EQUATORIAL PLANE

EARTH TARGET

GROUND TRACE OF ORBITAL PLANE

Presentation at RMC, Kingston, 8 October 2002
2008 IEEE Multiconference on Systems and Control, 2-5 September 2008, San Antonio, Texas, USA
THE FOUR MOTIONS TO TAKE INTO ACCOUNT

- **Earth Motion** (modelled)
- **Spacecraft Translation** (measured)
- **Spacecraft Rotation** (measured & controlled)
- **Equatorial Plane**
- **Ground Trace of Orbital Plane**

Presentation at RMC, Kingston, 8 October 2002

2008 IEEE Multiconference on Systems and Control, 2-5 September 2008, San Antonio, Texas, USA
THE FOUR MOTIONS TO TAKE INTO ACCOUNT

EARTH MOTION
(modelled)

SPACECRAFT TRANSLATION
(measured)

SPACECRAFT ROTATION
(measured & controlled)

SCANNING MOTION
(desired)

EQUATORIAL PLANE

GROUND TRACE OF ORBITAL PLANE
PROBA-1 SENSORS AND ACTUATORS

2008 IEEE Multiconference on Systems and Control, 2-5 September 2008, San Antonio, Texas, USA
PROBA-1 SPACECRAFT CONTROL

(GPS, STAR SENSORS)  DYNAMICS  (REACTION WHEELS)

SENSORS

ATTITUDE DETERMINATION & DELAY-RECOVERY

q_mes

(STR)

r_mes

(GPS)

r_est, v_est

FLYBY PREDICTIONS OVER 1 DAY

h_T, \phi_T, \lambda_T, \gamma_{MAX}

ATTITUDE DETERMINATION & DELAY-RECOVERY

q_est

\omega_{est}

ATTITUDE PROFILE GENERATION

q_cmd

\omega_{cmd}

v_est

r_est

ATITUDE CONTROL

q_err

\omega_{err}

\alpha_{cmd}

\alpha_{cmd}

time_FB_pre

time_FB_pre

min_angle_pre

TARGET SELECTION

time_FB_sys

TARGET TRACKING

SPACECRAFT

USER IMAGING REQUESTS

AUTOMATIC REQUEST VALIDATION

 USER IMAGING REQUESTS

h_T, \phi_T, \lambda_T, \gamma_{MAX}

h_T, \phi_T, \lambda_T, \gamma_{MAX}
PROBA-1 MODES OF OPERATION

**HARDWARE**
- GPS RECEIVER +
- STAR SENSORS REACTION WHEELS +
- MAGNETOMETER MAGNETIC TORQUER
- S/C SEPARATION

**SOFTWARE**
- NAV_ORB
- NAV_EARTH
- NAV_TARGET
- NAV_EVENTS
- GDC_IMAGING
- GDC_RELORB +
- NAV_ATT
- NAV_SUN
- NAV_ECLIPSE
- GDC_ATTERR
- GDC_SUN
- GDC_INERTIAL
- CTL_ATT
- CTL_ANGMOM +
- NAV_MAG
- CTL_MAG

**SUB-MODES**
- TERRESTRIAL MODES
  - CHRISTIS IMAGING SCANS
  - FIXED-EARTH POINTING
  - NADIR POINTING

- CELESTIAL MODES
  - SUN POINTING
  - COARSE POINTING
  - FINE POINTING

- MAGNETIC MODES
  - DETUMBLING
  - MAGNETIC ACQUISITION
  - ATTITUDE HOLD

- ACNS INITIALISATION

PROBA OPERATIONAL MODES
SOME PROBA-1 FLIGHT RESULTS
LAUNCH 22 OCTOBER 2001
B&W CCD CAMERA OPERATION

Spacecraft angular velocity [%sec]

Fixed target pointing commanded
Target Flyby
Nadir pointing commanded
THE PROCESS
The typical V-shape software development/validation process.

**PHASES**
- **0: Reqts Specification**
  - URD
- **A: Conceptual design**
  - SRD ⇒ CD
- **B: Preliminary design**
  - ADD ⇒ PD
- **C: Detail design**
  - DDD
- **D: Implementation**
  - Coding

**TESTS**
- Acceptance tests (SAT)
- System tests (SST)
- Integration tests (SIT)
- Unit tests (SUT)
More Definitions

- Algorithms:
  - mathematical description of a software function
  - at conceptual design level

- Pseudo-Code
  - mathematical description of a software module and flow logic
  - at preliminary and detailed design levels

- Models (in the context of CASE tools):
  - block-diagram description of algorithms and pseudo-code
  - e.g. Simulink™ models, SystemBuild™ models

- Code:
  - description of algorithms and pseudo-code in high-level, readable, computer language (ADA, C, C++)
Typical 2\textsuperscript{nd}-generation S/W development/validation process

PHASES
0: Reqts Specification
A: Conceptual design
B: Preliminary design
C: Detail design
D: Implementation

CASE Tool Models

GNC engineers

S/W engineers

Pseudo-coding
Hand Coding in ADA or C

TESTS
Acceptance tests
System tests
Integration tests
Unit tests
Validation in home-made test environment
Typical 4th-generation S/W development/validation process

PHASES
0: Reqs Specification
A: Conceptual design
B: Preliminary design
C: Detail design
D: Implementation

Tests
System tests
Integration tests
Unit tests

Acceptance tests
Code Generation
Validation in executable
Validation in CASE Tool test environment

GNC/SW engineers
THE LESSONS AND THE BENEFITS
MatrixX models:
- 1401 instances of 355 superblocks, 548 parameters in total
- The onboard GNC module has 128 inputs, 983 outputs
- The environment module has 33 inputs, 190 outputs

AutoCode generated software:
- The onboard GNC module has 57217/27181 lines, 1016 global variables and 249 functions.
- The environment module has 18220/9563 lines, 734 global variables and 86 functions.
- The code is very readable.
- Traditional coding and validation alone would have taken 15 persons-years (ESA estimation)
- With AutoCoding, PROBA spent <9 persons-years including requirements phase, algorithms definition and design, architecture specification, code production and validation.
PROBA-2 STATISTICS

AOCS Design (Models) 🔹 Matrix/SystemBuild Environment 🔹 Completed by NGC in 2006

SIT (Models) 🔹 Software Integration Tests (SIT) 🔹 Completed by NGC in 2006

SST (C code) 🔹 Software System-level Tests (SST) 🔹 Completed by NGC in 2007
🔹 Automatic generation of C-code
🔹 Automatic compilation and building of executable
🔹 Automatic generation of test reports

SIV (System Sim.) 🔹 Software Independent Validation (SIV) 🔹 Completed by Verhaert & NGC in 2007-08
🔹 Perform Software Acceptance Tests (SAT)

HIL (Spacecraft) 🔹 Hardware-In-the-Loop (HIL) Tests
🔹 On-going at Verhaert
🔹 Perform HW-SW integration tests & SAT

3 NGC Engineers at NGC (<5 PY)
1 NGC Eng
1 VE Eng
1 ESA Eng at S/C Contractor (<1 PY)
LESSONS LEARNED

+ Reduction in the number of documents:
  • automatic generation of document
  • models act as Architectural and Detail Design Documents

+ Reduction of human interface from models to on-board code
  • reduction in verification process
  • minimisation of human errors, discrepancies, etc.

+ Better visibility/understanding/organisation of the algorithms
  • easier to find sources of bugs
  • easier to add/delete modules
  • non-expert can easily understand

+ Simpler/faster transfer of knowledge
  • easy and quick transfer of knowledge to software engineer
  • easy to add new engineers to the project
LESSONS LEARNED

+ Automated generation of test results
  • Hundreds of cases can be automatically generated over night
  • Turn-around time from bug correction to validation is shorter
+ Dramatic reduction in level of effort required
  • PROBA-1 took less than half the LOE compared to typical mission
  • PROBA-2 took even less
  – The size of the on-board C code is not as optimal as if it had been written by humans
  – The computational efficiency is not as optimal
  – Some common algorithms (e.g. for-loop, while) available in native code (C code) are more complicated to implement in model-based form.
  ✷ One needs to learn how the code generator works in order to optimize the models for code generation
CONCLUSION

COMPUTED-AIDED SOFTWARE ENGINEERING

IS THE WAY FORWARD
THANKS TO...

- Jimmy Côté*
- Aymeric Kron*
- Steve Ulrich
- Frédéric Teston
- Pierrick Vuilleumier*
- Stefano Santandrea
- Pieter VanDer Braembussche*
- Joris Naudet*
- Dirk Bernaerts
- and thanks for your attention