

ATV Flight Application Software Command Checker

Ada-Europe 2012
Stockholm
June 14



Jørgen Bundgaard
jb@ada-dk.org

Automated Transfer Vehicle Flight Application Software Command Checker (FCC)

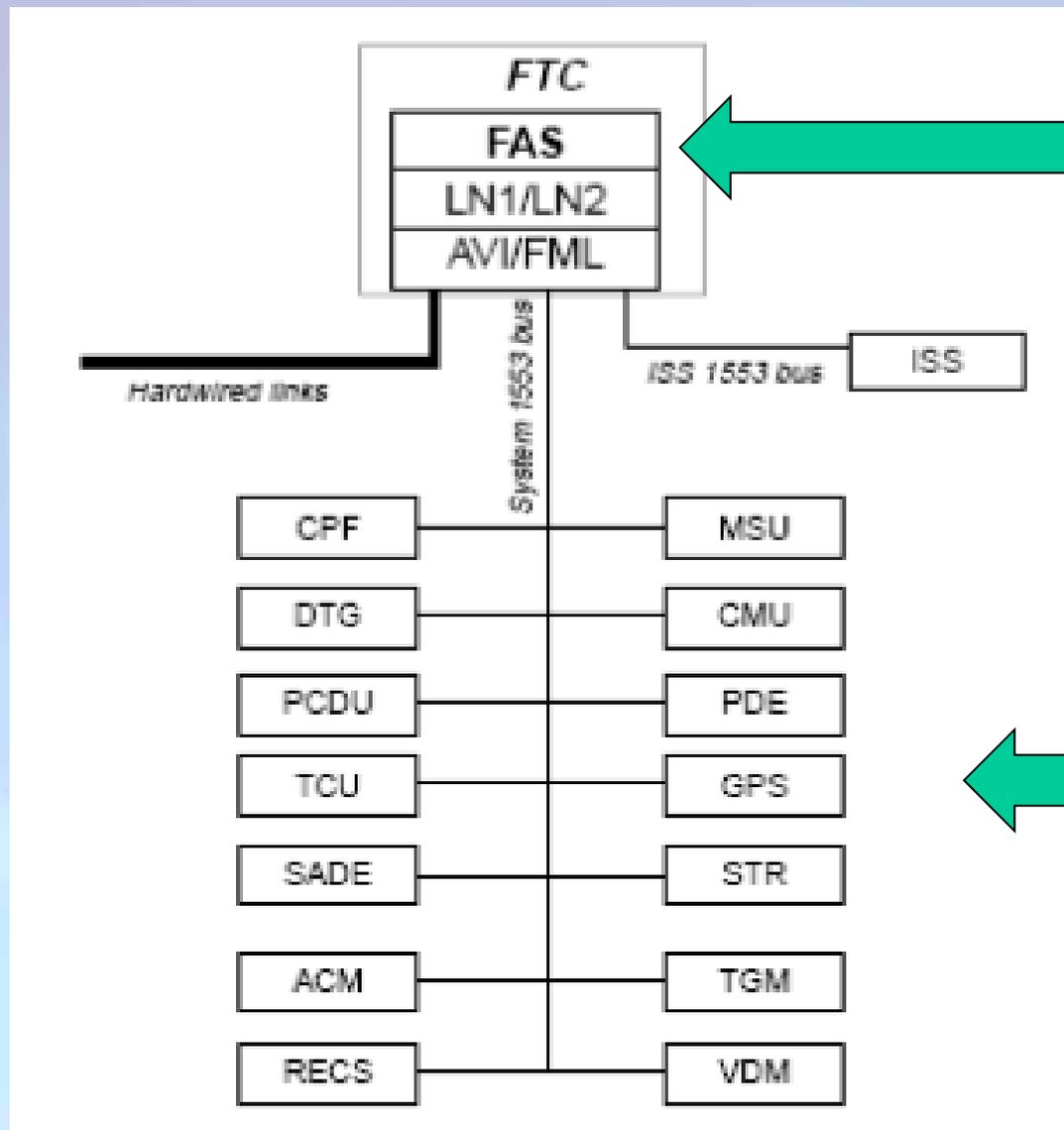
A very large on-board spacecraft software application contained, despite that it was programmed in Ada 83, a coding pattern, which was unprotected against out-of-range errors in telecommand parameters.

Not only was the pattern used over eight hundred times in the code, but the design of the software application prevented the problem to be fixed by fixing the pattern.

This is the story about the solution.

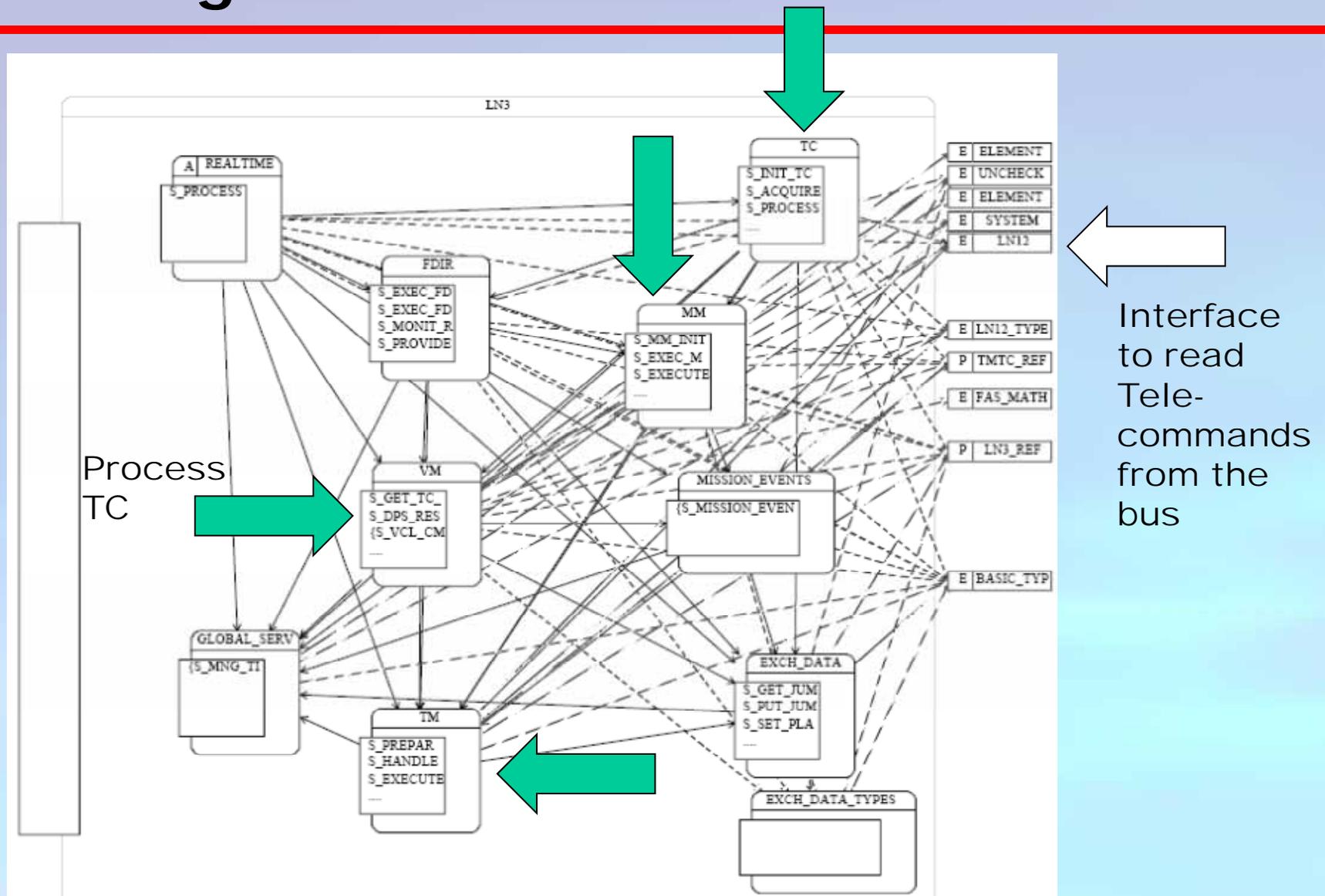


ATV Flight Application Software (FAS)



Sensors
and
actuators

Processing of TeleCommands in the FAS



What was the problem? Why is the FCC needed?

```
type T_CMD_ID is ( CMD1, CMD2, ..., CMDn ); -- enumeration type
...
type T_LLC_TYPE is
  record
    CMD : T_INT32;
    ...
  end record;
...
LLC : T_LLC_TYPE := ... obtain Low-Level telecommand from the bus ...
...
case T_CMD_ID'VAL( LLC.CMD ) is
  when CMD1 => ... process CMD1 command ...
  when CMD2 => ...
  ...
  when CMDn => ...
end case;
```

← UNSAFE

How could it happen?

- **FAS Technical Specification, section 2.3.5 Security requirements:**

“For data coming from ground or ISS, it is assumed that they will respect format and structure that will be defined in the FAS external interface documentation (see [AD10] and [AD11]). The ALB SW shall perform only integrity check (through checksum calculation) of the transmitted data. Check of IEEE format availability (integer or floating point), and check of data availability (boundary and conformity to defined IF) is under ground or ISS responsibility.”

How did EADS-ST (the developer) react?

The “Red Team” analysed the problem :

- More than 3700 different enumeration types used in FAS
- Implementing parameter bound checks in the onboard FAS will have major impact on FAS design, code and validation
- Telecommand parameters cannot just be checked where used first time because they are also part of HW setup onboard procedures which cannot be stopped in the middle
- The FAS schedule will be impacted
- The recommendation is to implement the checks on ground

How did ESA (the customer) react?

Rovsing was contracted to develop a FAS Command Checker (FCC) with the following requirement highlights:

- The FCC shall run on a separate computer on ground at ATV CC, run on a SunSPARC/Solaris platform and be coded in Ada83
- The FCC shall receive telecommand packets from M&C via TCP/IP, check the packet for compatibility with the FAS, and return a corresponding packet evaluation report to M&C (OK or not)
- The maximum response time for the FCC shall be 10 ms
- The FCC shall be developed as Category B software
- The FCC shall protect the FAS *as coded*, i.e. in particular **not** as specified in the Technical Specification, and **not** as defined by the Mission Data Base (MDB) ... !

The FCC Project was established

Customer:

ESA

Contractor:

Rovsing A/S

User:

ATV Control Centre

ATV developer:

EADS-Space Transportation

Development phase:

Sep 2004 – Dec 2005

Maintenance phase:

Jan 2006 – launch in 2008

The Technical Solution

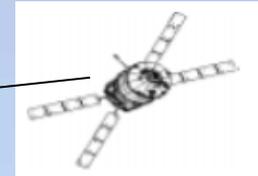
FCC in the ATV Control Centre

FCC architecture

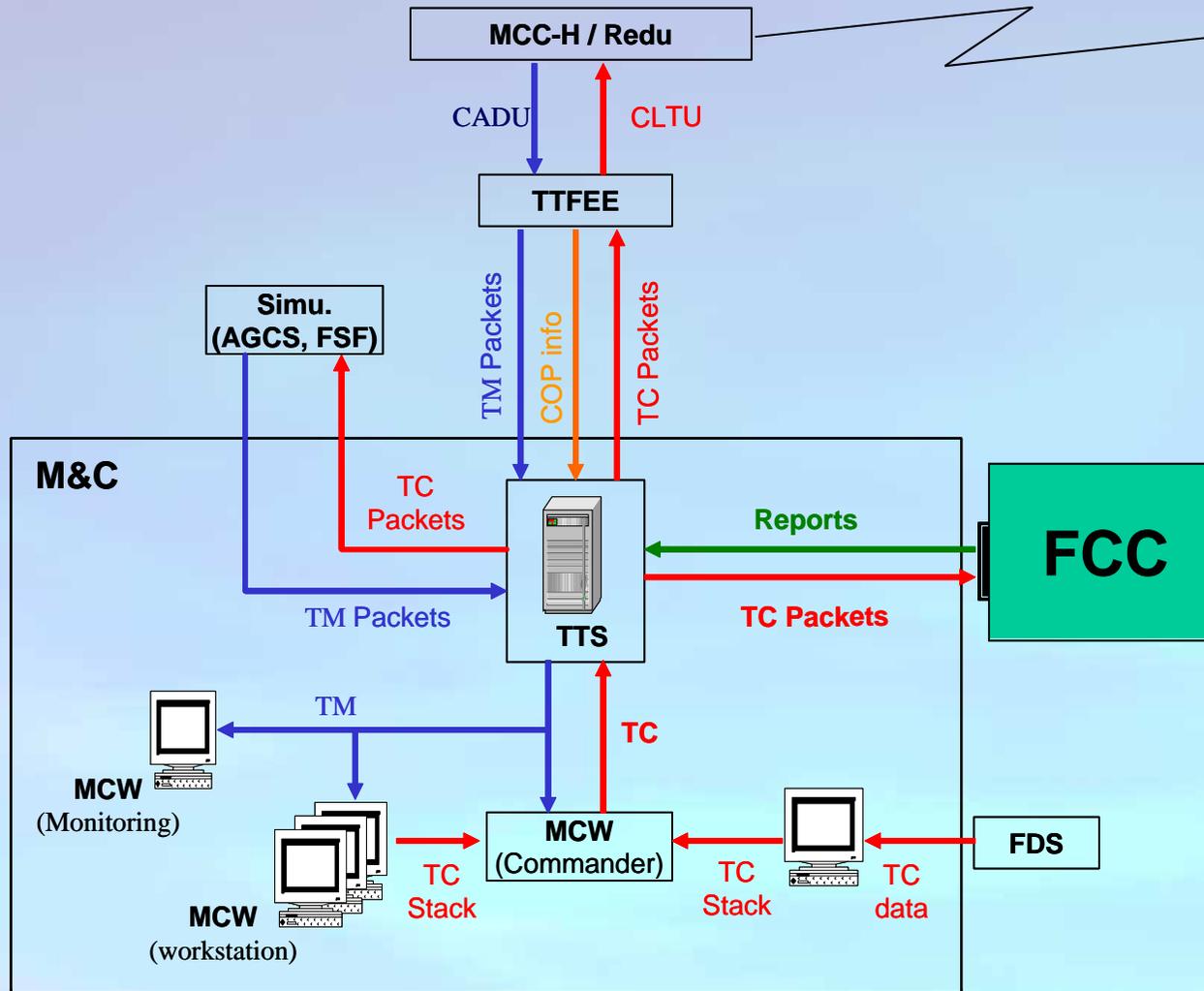
Protecting the FAS *as coded*

Testing the FCC

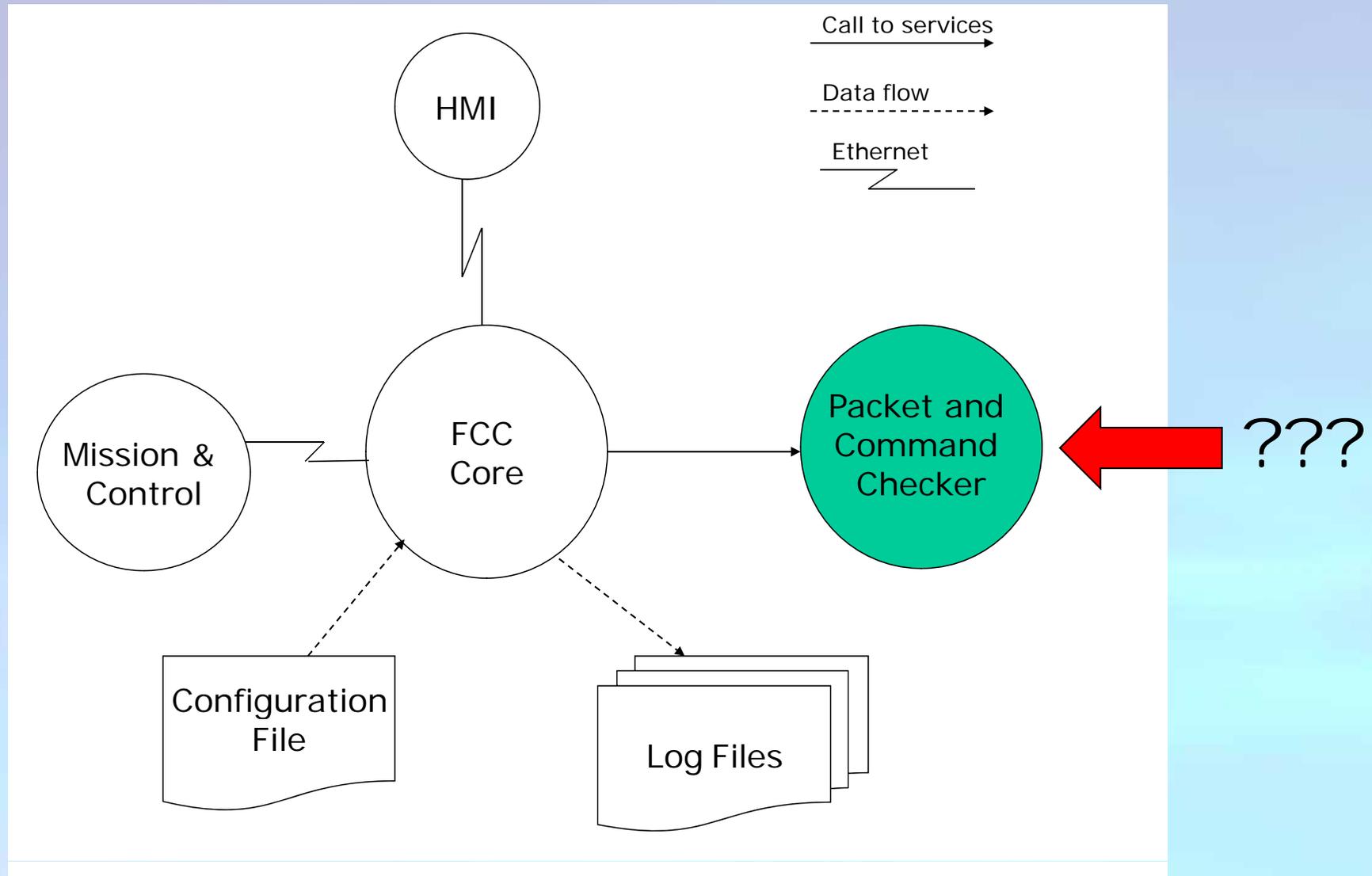
FCC integrated at the ATV Control Centre



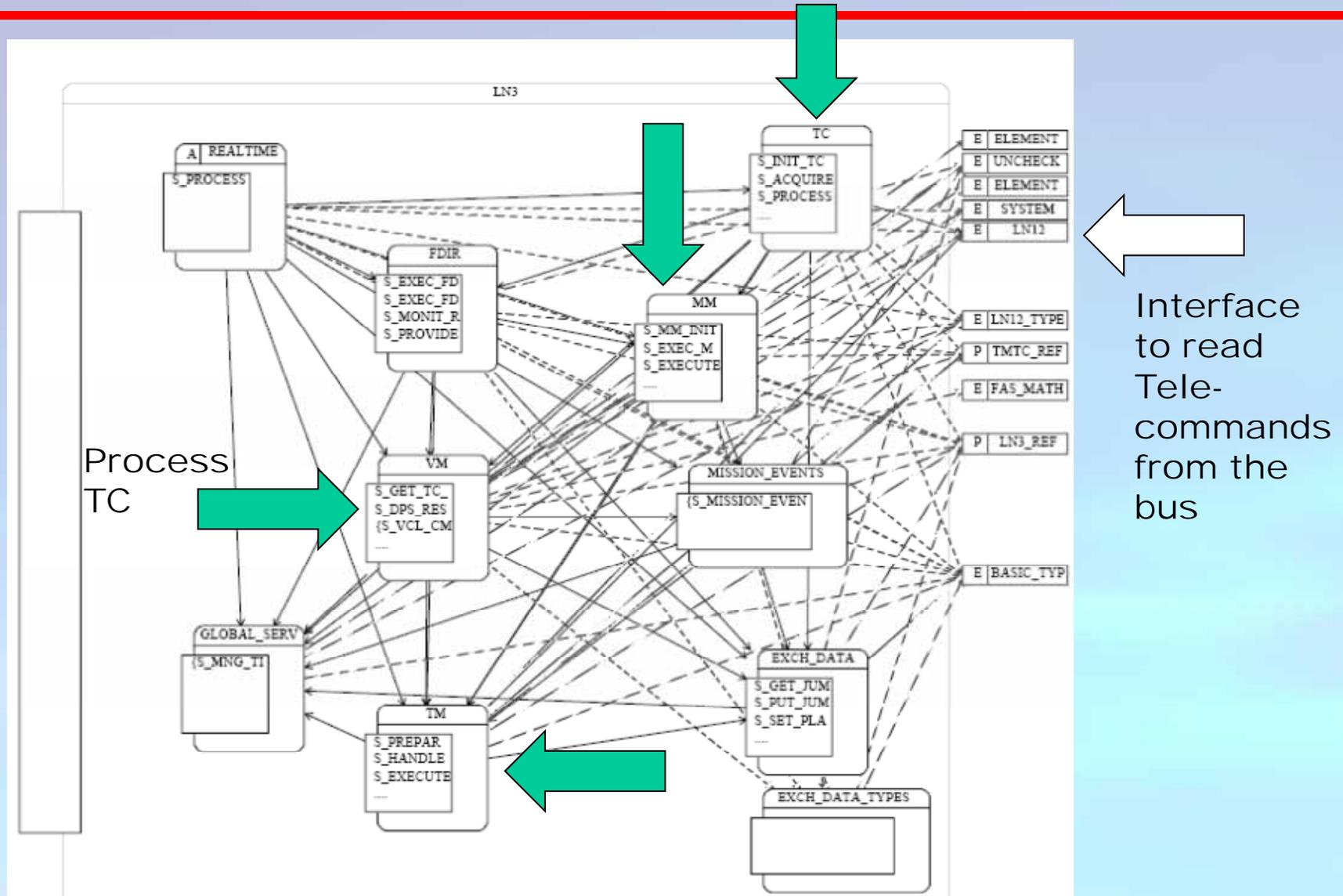
ATV CC



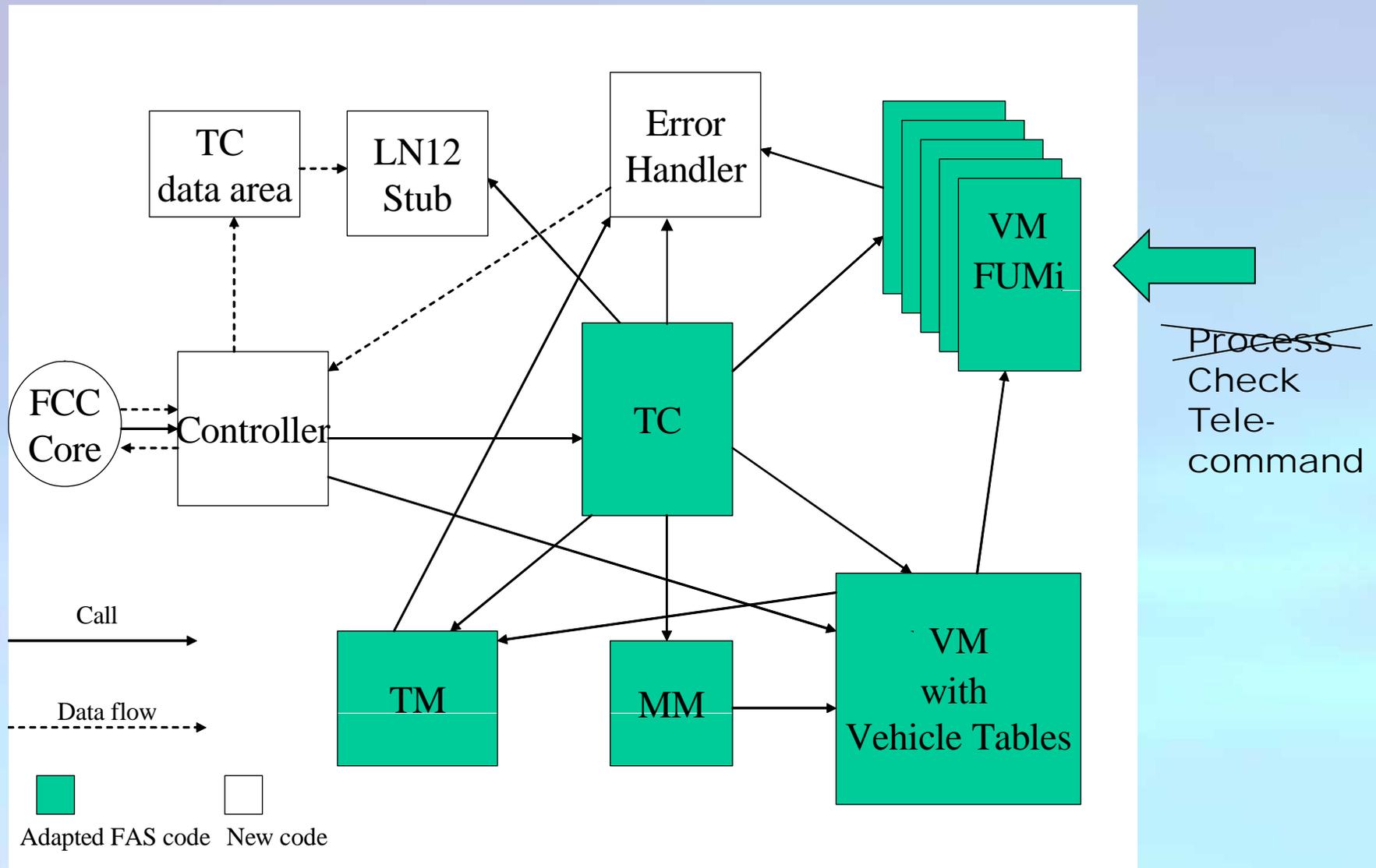
FCC Architecture



Processing of TeleCommands in the FAS



Packet and Command Checker – reuse of FAS code



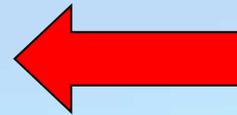
How to protect the FAS as *coded*?

... LLC is an input parameter to this subprogram ...

begin

```
-- Default value for execution report is LLC_EXECUTED  
EXEC_RPT := TMTC_REF.LLC_EXECUTED;
```

```
case (T_GYRA_LLC_ID'VAL(LLC.PARAM_ID)) is
```



UNSAFE

```
-----  
-- GYRA_LLC_ANG_RATE  
-----
```

```
when GYRA_LLC_ANG_RATE =>
```

```
  L_GYRA_LLC_ANG_RATE_PTR :=
```

```
    S_ADDRESS_TO_GYRA_LLC_ANG_RATE(LLC.PARAM_VALUE.DATA'ADDRESS);
```

```
  GYRA_DATA.S_PUT_GYRA_LLC_ANG_RATE (VALUE => L_GYRA_LLC_ANG_RATE_PTR.all);
```

```
when => ...
```

```
end case;
```

```
...
```

Adapted FAS code is reused in the FCC

```
begin
-- Default value for execution report is LLC_EXECUTED
EXEC_RPT := TMT_REF.LLC_EXECUTED;

--!FCC Insert
-- Check that LLC.PARAM_ID is in the positional range of type T_GYRA_LLC_ID
FCC_CHECKER.S_CHECK ( PARAM => "LLC ID",
                      VALUE => LLC.PARAM_ID,
                      MIN    => T_GYRA_LLC_ID'POS(T_GYRA_LLC_ID'FIRST),
                      MAX    => T_GYRA_LLC_ID'POS(T_GYRA_LLC_ID'LAST),
                      AREA   => "GYRA_EXECUTE_LLC.S_EXEC_GYRA_LLC 1",
                      ENUM   => "T_GYRA_LLC_ID",
                      IS_VALID => IS_VALID_PARAM );

if IS_VALID_PARAM then
--!FCC End Insert

case (T_GYRA_LLC_ID'VAL(LLC.PARAM_ID)) is
-----
-- GYRA_LLC_ANG_RATE
-----
when GYRA_LLC_ANG_RATE =>

--!FCC Remove
--fcc  L_GYRA_LLC_ANG_RATE_PTR :=
--fcc      S_ADDRESS_TO_GYRA_LLC_ANG_RATE(LLC.PARAM_VALUE.DATA'ADDRESS);
--fcc  GYRA_DATA.S_PUT_GYRA_LLC_ANG_RATE (VALUE => L_GYRA_LLC_ANG_RATE_PTR.all);
--!FCC End Remove
  --!FCC Insert
  null;
  --!FCC End Insert
```

insert-tag

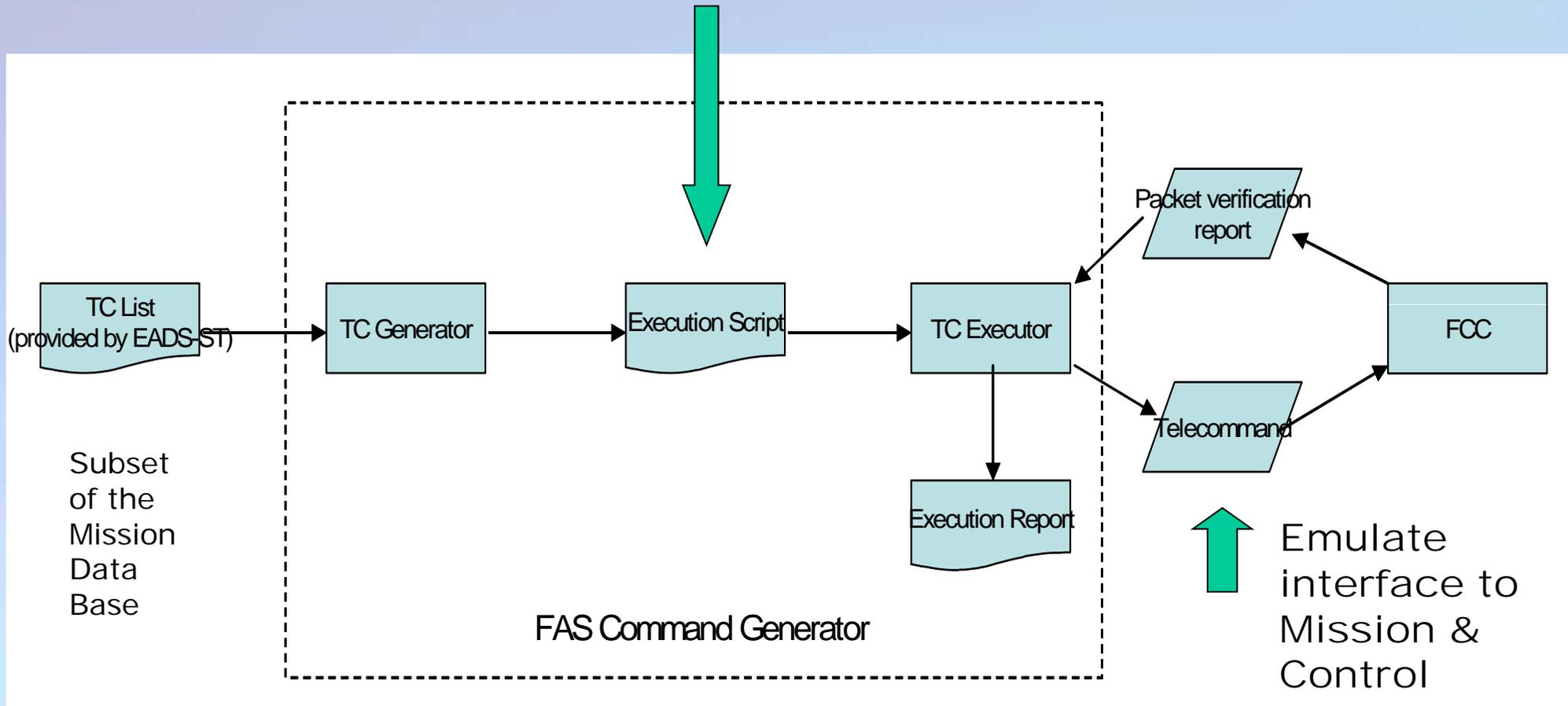
CHECK
using
FAS
types

SAFE

remove-tag

Testing the FCC

500.000 automatically generated test cases



The Big Surprise ... !

130 FAS-Mission Database (MDB) discrepancies, causing thousands of test cases to fail, were discovered, e.g.:

- FAS expects values in the range 1 .. 4, MDB allows range is 0 .. 5
- Wrong values in MDB make FAS perform wrong table lookup
- MDB describes parameters that are never used by the FAS
- Parameters needed by the FAS are not described in the MDB
- MDB counts in bits while FAS counts in bytes ...!

We have detected and reported many genuine bugs in the FAS and/or in the MDB which would have caused severe problems if the ATV had been flying with them.

Software tools used in the FCC project

- Atlassian JIRA for tracking Discrepancies, NCRs, Action Items
- IBM Rational Rose for UML design
- GNU CVS for version and revision control
- GNU C/C++ compilers
- Gnat Ada 95 compiler
- Aonix Object Ada 83 compiler
- New Jersey University Standard ML compiler
- TrollTech QT GUI builder
- IBM Rational Test Realtime for test coverage measurement
- STI Understand for Ada for source browsing and metrics
- EADS Ada tool for checking coding standard
- ComponentSoftware CSdiff for comparing FAS versions

Selected Code Metrics

- Calls to FCC_CHECKER.S_CHECK: 803
- "--!FCC insert" in adapted FAS: 1.173
- "--!FCC remove" in adapted FAS: 1.326

- Ada source lines in adapted FAS: 236.573
- Ada source lines in FCC core: 12.304
- C source lines in FCC core: 1.337
- C++ source lines in HMI : 2.831

Verification and Validation

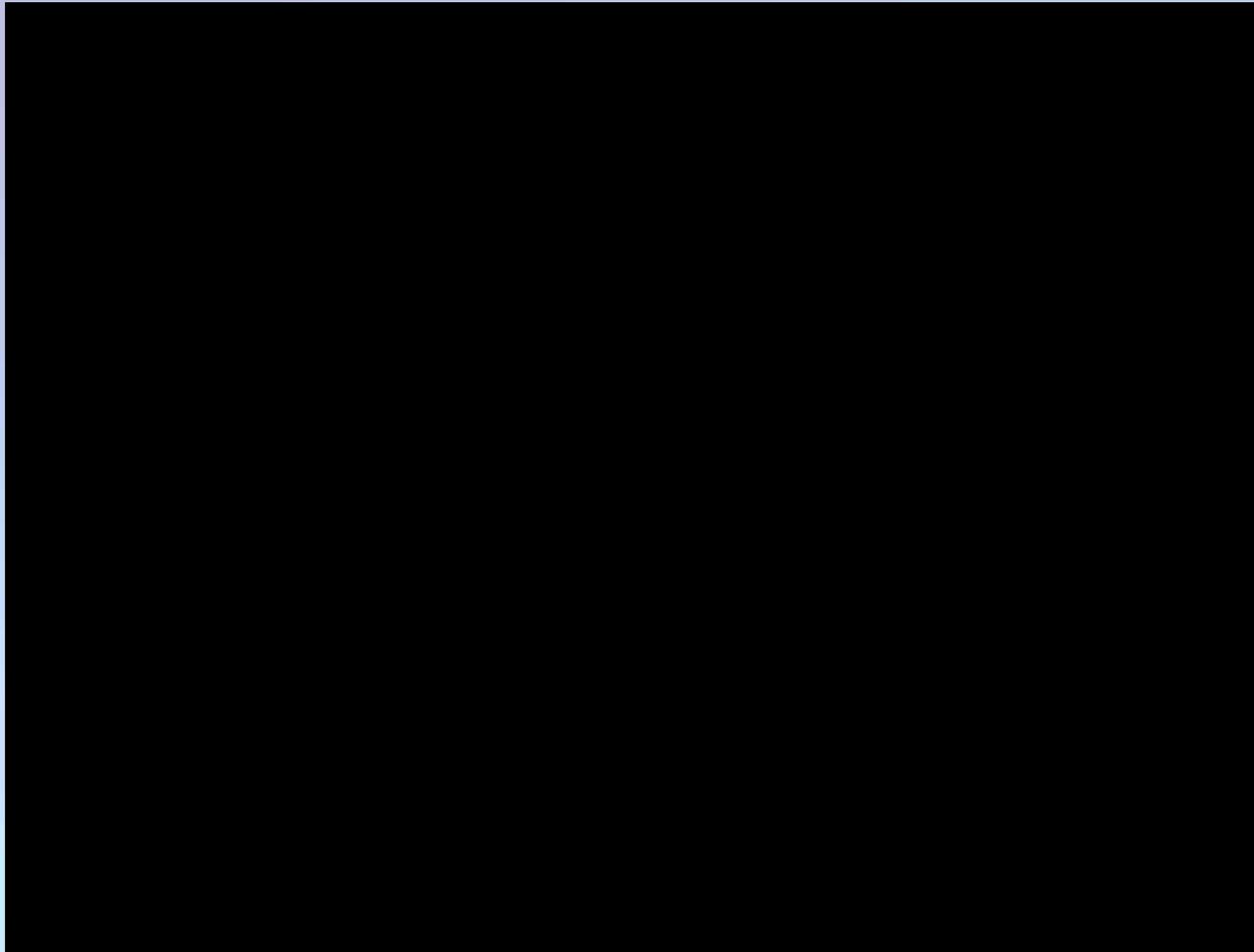
- ESA standard model for Category B software
- Extensive documentation, tracing, and compliance matrixes
- Unit test, 100% MC/DC
- Integration test, at Ada package level
- Validation test, requirements based
- 5700 different telecommands: boundary value tests for all telecommand parameters of enumeration / integer subtypes
- Continuous integration and test
- Assertions
- Coding standard compliance analysis via tools
- Manual inspection of the code

Lessons learned

- Errors in the requirements can be very expensive to fix later
- The MDB needed a completely unexpected amount of debugging. The FAS and the MDB were inconsistent on far more points than initially expected. About 130 “discrepancies” were raised
- It is very expensive, time consuming, and frustrating to try to plan, estimate, and staff a development project of this size before a viable technical solution has been identified
- It is very easy to underestimate the cost of developing category B software. In particular the cost of documentation and verification
- It pays off to have good personal relations and contacts at the “factory floor” level. The involved parties were able to collaborate very efficiently and effectively

ATV successfully docked - April, 3rd 2008





Questions?

The Jules Verne ATV module docking with the International Space Station (ISS)

ATV FAS Command Checker (FCC)

- An extension of FAS on ground
- Runs on a computer at the ATV Control Centre
- Checks telecommands for compatibility with the ATV FAS onboard software before they are sent to the ATV by the operators
- Because:
Incompatible telecommands can potentially crash the ATV onboard software

