Housekeeping Telemetry Viewer: 10 years of operations

Jakob Livschitz
ESA/TEC-SWT

Lars Fiedler
EUMETSAT/LEO

ESA UNCLASSIFIED - For Official Use

What is HKTV?

HKTV stands for House-keeping Telemetry Viewer.

In essence, HKTV is a **packet**[1] viewer and analyzer.

[1]What is packet? A chunk of data with some properties: length, type (TM/TC), APID. Example: CCSDS packet.

Motivation

HKTV development has been initiated by EUMETSAT exactly 10 years ago as an in-house project to enable the analysis of IASI[2] data collected during AIT at Satellite level:

- telemetry
- telecommands
- science data.

For:

- reporting, trending, investigation, and
- to preserve knowledge in long term program.

[2]Part of EUMETSAT Polar System, on board Metop-A,B,and C satellite.

History

HKTV in 2009:

- Monolithic Java application (only standalone processing on one PC)
- Java 5 and 6 support
- GUI based on Java Swing
- Proprietary archive format
- Plotting based on JFreeChart
- Reporting based on JFreeReport

HKTV past and current use

HKTV in 2009-2019 supports:

- IASI on Metop
- MHS on Metop
- IASI-NG on Metop-SG
- METIMAGE on Metop-SG
- Metop-SG (all instruments)

Good tools survive…
with little maintenance

HKTV in 2019:

- Monolithic Java application (main target platform: MacOS)
- Distributed processing support via message bus (ActiveMQ)
- PUS support
- Java 8-13 support
- GUI based on Java Swing
- Archive format: proprietary and MongoDB
- Plotting based on JFreeChart
- Reporting based on JFreeReport
- Synoptic displays based on SVG/HTML/JavaScript and JavaFX

Concepts

HKTV follows the following concepts:

- **One Click** – satisfy the user, not the requirement
- **AGILE**
- **KISS** (keep it small and simple): if functionality is needed in >90% of use cases, it goes into the kernel, otherwise it's implemented as an extension (dedicated java package per instrument). This keeps the kernel very simple.
- **YAGNI** (you aren't gonna need it): never process anything unless really needed. Opaque packets remain the only data always available. Everything else, e.g. parameters are extracted only upon request. Plots painted only if visible.
- **CI/CD** with automated testing

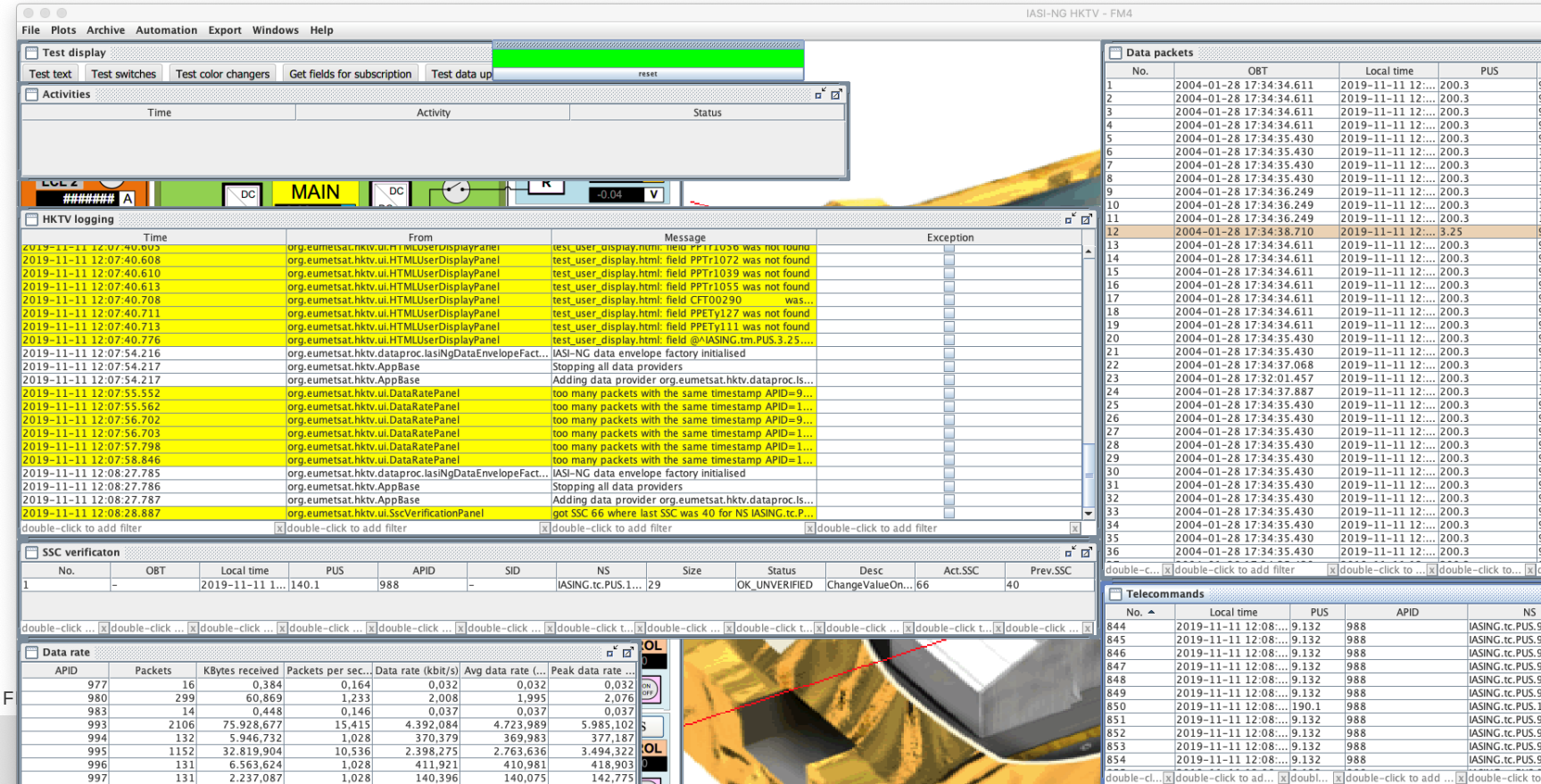
Architecture

HKTV is built around a **data bus**. Every CCSDS packet is pushed through this bus. The bus is implemented in ActiveMQ, meaning that you can have both local and remote clients.

Once you want to add an additional component (a window, data processing, specific storage, etc.) just subscribe to the data bus.

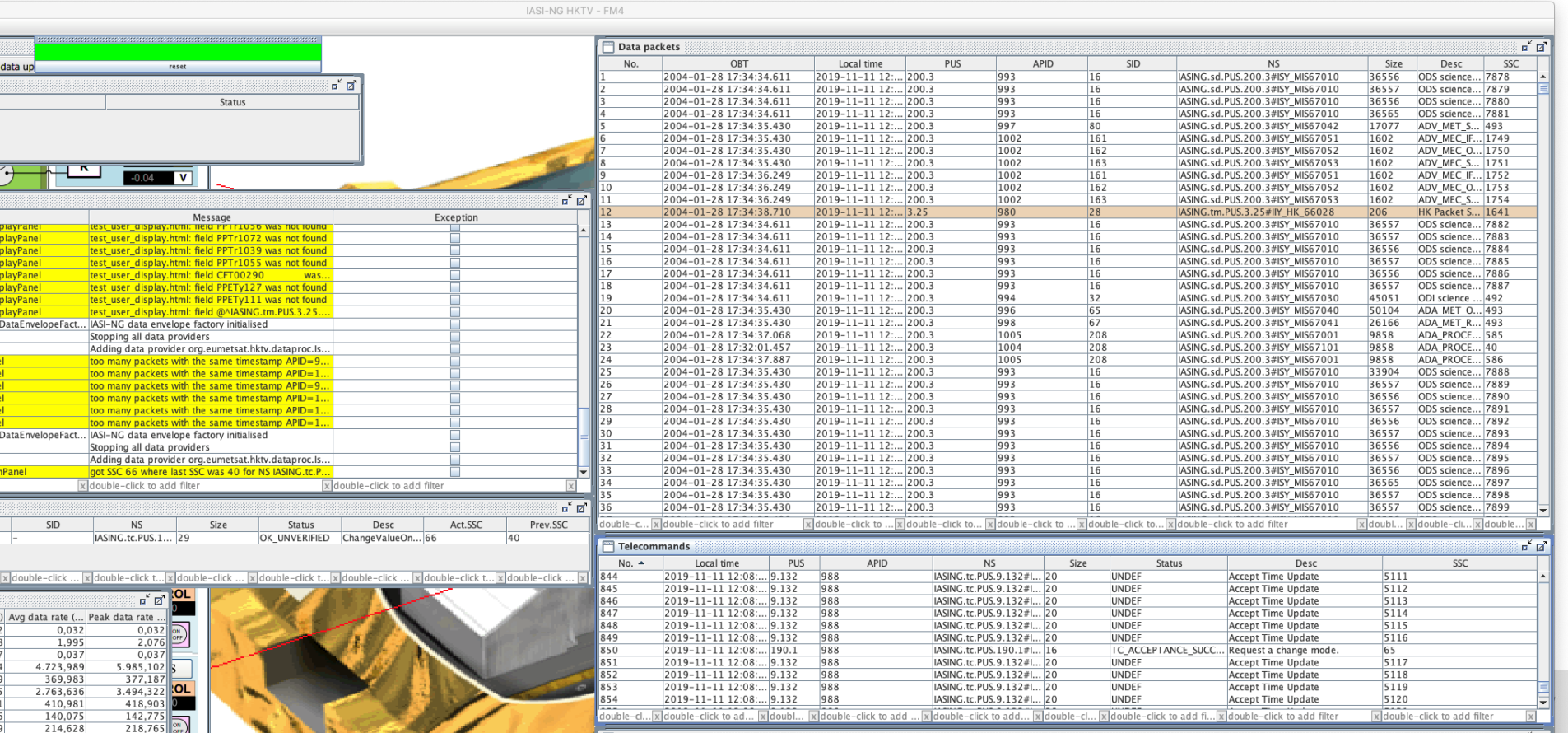
Each data consumer runs in a separate thread.

Main window

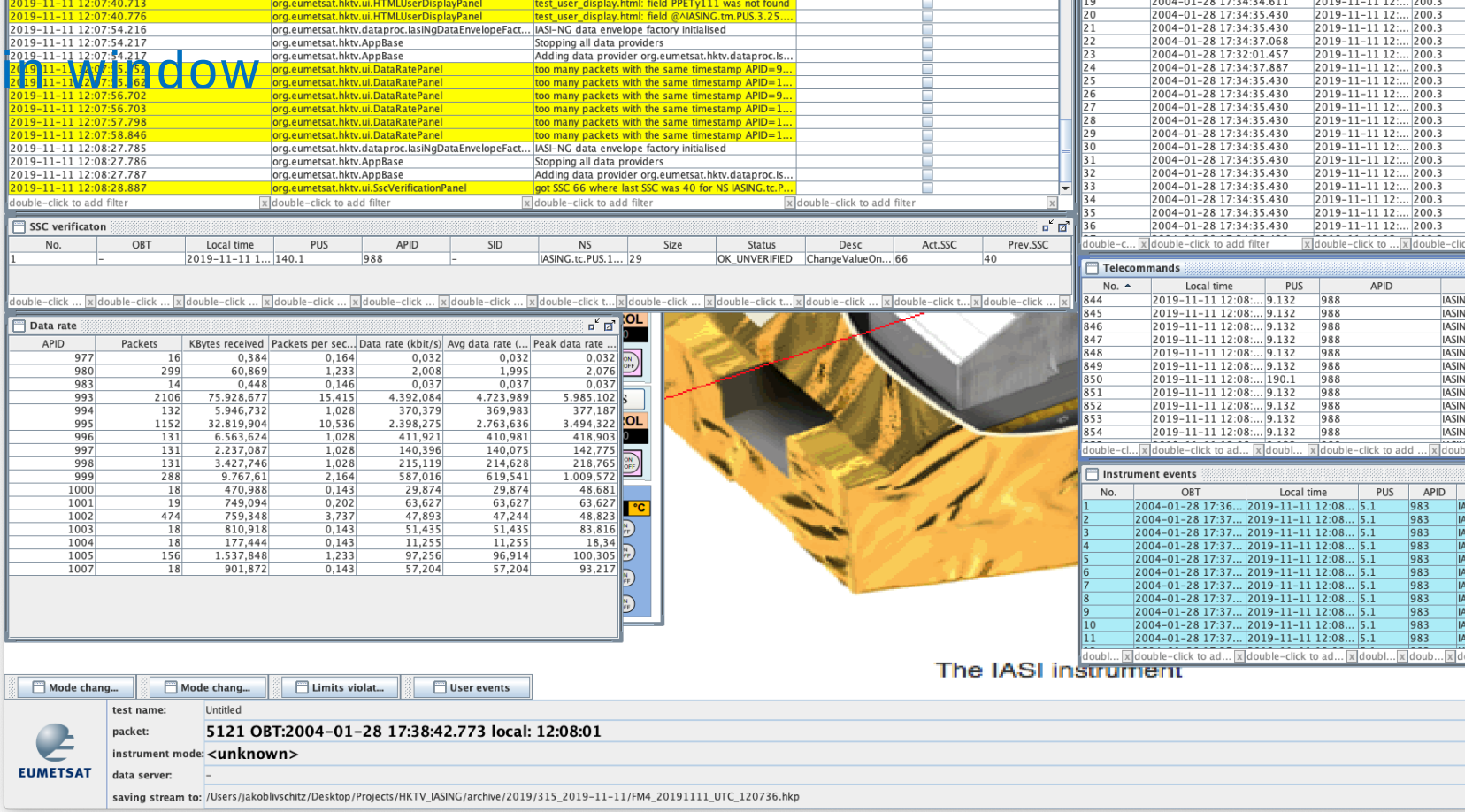


ESA UNCLASSIF

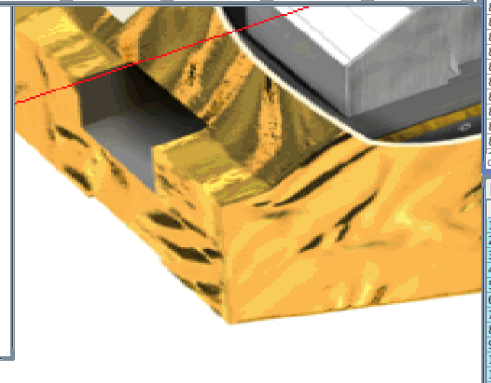
Main window


IASI-NG HKTV - FM4
data up
reset
Status
-0.04
V
Message
Exception
test_user_display.html: field PPTr1072 was not found
test_user_display.html: field PPTr1039 was not found
test_user_display.html: field PPTr1055 was not found
test_user_display.html: field CFT00290 was...
test_user_display.html: field PPETy127 was not found
test_user_display.html: field PPETy111 was not found
test_user_display.html: field @^IASING.tm.PUS.3.25....
IASI-NG data envelope factory initialised
Stopping all data providers
Adding data provider org.eumetsat.hktv.dataproc.Is...
too many packets with the same timestamp APID=9...
too many packets with the same timestamp APID=1...
too many packets with the same timestamp APID=9...
too many packets with the same timestamp APID=1...
too many packets with the same timestamp APID=1...
got SSC 66 where last SSC was 40 for NS IASING.tc.P...
double-click to add filter
SID
NS
Size
Status
Desc
Act.SSC
Prev.SSC
IASING.tc.PUS.1...
29
OK_UNVERIFIED
ChangeValueOn...
66
40
Avg data rate (...
Peak data rate ...
0,032
0,032
1,995
2,076
0,037
0,037
4.723,989
5.985,102
369,983
377,187
2.763,636
3.494,322
410,981
418,903
140,075
142,775
214,628
218,765
Data packets
No.
OBT
Local time
PUS
APID
SID
NS
Size
Desc
SSC
2004-01-28 17:34:34.611
2019-11-11 12:...
200.3
993
16
IASING.sd.PUS.200.3#ISY_MIS67010
36556
ODS science...
7878
2004-01-28 17:34:34.611
2019-11-11 12:...
200.3
993
16
IASING.sd.PUS.200.3#ISY_MIS67010
36557
ODS science...
7879
2004-01-28 17:34:34.611
2019-11-11 12:...
200.3
993
16
IASING.sd.PUS.200.3#ISY_MIS67010
36556
ODS science...
7880
2004-01-28 17:34:34.611
2019-11-11 12:...
200.3
993
16
IASING.sd.PUS.200.3#ISY_MIS67010
36565
ODS science...
7881
2004-01-28 17:34:35.430
2019-11-11 12:...
200.3
997
80
IASING.sd.PUS.200.3#ISY_MIS67042
17077
ADV_MET_S...
493
2004-01-28 17:34:35.430
2019-11-11 12:...
200.3
1002
161
IASING.sd.PUS.200.3#ISY_MIS67051
1602
ADV_MEC_IF...
1749
2004-01-28 17:34:35.430
2019-11-11 12:...
200.3
1002
162
IASING.sd.PUS.200.3#ISY_MIS67052
1602
ADV_MEC_O...
1750
2004-01-28 17:34:35.430
2019-11-11 12:...
200.3
1002
163
IASING.sd.PUS.200.3#ISY_MIS67053
1602
ADV_MEC_S...
1751
2004-01-28 17:34:36.249
2019-11-11 12:...
200.3
1002
161
IASING.sd.PUS.200.3#ISY_MIS67051
1602
ADV_MEC_IF...
1752
2004-01-28 17:34:36.249
2019-11-11 12:...
200.3
1002
162
IASING.sd.PUS.200.3#ISY_MIS67052
1602
ADV_MEC_O...
1753
2004-01-28 17:34:36.249
2019-11-11 12:...
200.3
1002
163
IASING.sd.PUS.200.3#ISY_MIS67053
1602
ADV_MEC_S...
1754
2004-01-28 17:34:38.710
2019-11-11 12:...
3.25
980
28
IASING.tm.PUS.3.25#IIY_HK_66028
206
HK Packet S...
1641
2004-01-28 17:34:34.611
2019-11-11 12:...
200.3
993
16
IASING.sd.PUS.200.3#ISY_MIS67010
36557
ODS science...
7882
2004-01-28 17:34:34.611
2019-11-11 12:...
200.3
993
16
IASING.sd.PUS.200.3#ISY_MIS67010
36557
ODS science...
7883
2004-01-28 17:34:34.611
2019-11-11 12:...
200.3
993
16
IASING.sd.PUS.200.3#ISY_MIS67010
36556
ODS science...
7884
2004-01-28 17:34:34.611
2019-11-11 12:...
200.3
993
16
IASING.sd.PUS.200.3#ISY_MIS67010
36557
ODS science...
7885
2004-01-28 17:34:34.611
2019-11-11 12:...
200.3
993
16
IASING.sd.PUS.200.3#ISY_MIS67010
36556
ODS science...
7886
2004-01-28 17:34:34.611
2019-11-11 12:...
200.3
993
16
IASING.sd.PUS.200.3#ISY_MIS67010
36557
ODS science...
7887
2004-01-28 17:34:34.611
2019-11-11 12:...
200.3
994
32
IASING.sd.PUS.200.3#ISY_MIS67030
45051
ODI science ...
492
2004-01-28 17:34:35.430
2019-11-11 12:...
200.3
996
65
IASING.sd.PUS.200.3#ISY_MIS67040
50104
ADA_MET_O...
493
2004-01-28 17:34:35.430
2019-11-11 12:...
200.3
998
67
IASING.sd.PUS.200.3#ISY_MIS67041
26166
ADA_MET_R...
493
2004-01-28 17:34:37.068
2019-11-11 12:...
200.3
1005
208
IASING.sd.PUS.200.3#ISY_MIS67001
9858
ADA_PROCE...
585
2004-01-28 17:32:01.457
2019-11-11 12:...
200.3
1004
208
IASING.sd.PUS.200.3#ISY_MIS7101
9858
ADA_PROCE...
40
2004-01-28 17:34:37.887
2019-11-11 12:...
200.3
1005
208
IASING.sd.PUS.200.3#ISY_MIS67001
9858
ADA_PROCE...
586
2004-01-28 17:34:35.430
2019-11-11 12:...
200.3
993
16
IASING.sd.PUS.200.3#ISY_MIS67010
33904
ODS science...
7888
2004-01-28 17:34:35.430
2019-11-11 12:...
200.3
993
16
IASING.sd.PUS.200.3#ISY_MIS67010
36557
ODS science...
7889
2004-01-28 17:34:35.430
2019-11-11 12:...
200.3
993
16
IASING.sd.PUS.200.3#ISY_MIS67010
36556
ODS science...
7890
2004-01-28 17:34:35.430
2019-11-11 12:...
200.3
993
16
IASING.sd.PUS.200.3#ISY_MIS67010
36557
ODS science...
7891
2004-01-28 17:34:35.430
2019-11-11 12:...
200.3
993
16
IASING.sd.PUS.200.3#ISY_MIS67010
36556
ODS science...
7892
2004-01-28 17:34:35.430
2019-11-11 12:...
200.3
993
16
IASING.sd.PUS.200.3#ISY_MIS67010
36557
ODS science...
7893
2004-01-28 17:34:35.430
2019-11-11 12:...
200.3
993
16
IASING.sd.PUS.200.3#ISY_MIS67010
36556
ODS science...
7894
2004-01-28 17:34:35.430
2019-11-11 12:...
200.3
993
16
IASING.sd.PUS.200.3#ISY_MIS67010
36557
ODS science...
7895
2004-01-28 17:34:35.430
2019-11-11 12:...
200.3
993
16
IASING.sd.PUS.200.3#ISY_MIS67010
36556
ODS science...
7896
2004-01-28 17:34:35.430
2019-11-11 12:...
200.3
993
16
IASING.sd.PUS.200.3#ISY_MIS67010
36565
ODS science...
7897
2004-01-28 17:34:35.430
2019-11-11 12:...
200.3
993
16
IASING.sd.PUS.200.3#ISY_MIS67010
36557
ODS science...
7898
2004-01-28 17:34:35.430
2019-11-11 12:...
200.3
993
16
IASING.sd.PUS.200.3#ISY_MIS67010
36557
ODS science...
7899
double-click to add filter
Telecommands
No.
Local time
PUS
APID
NS
Size
Status
Desc
SSC
844
2019-11-11 12:08:...
9.132
988
IASING.tc.PUS.9.132#I...
20
UNDEF
Accept Time Update
5111
845
2019-11-11 12:08:...
9.132
988
IASING.tc.PUS.9.132#I...
20
UNDEF
Accept Time Update
5112
846
2019-11-11 12:08:...
9.132
988
IASING.tc.PUS.9.132#I...
20
UNDEF
Accept Time Update
5113
847
2019-11-11 12:08:...
9.132
988
IASING.tc.PUS.9.132#I...
20
UNDEF
Accept Time Update
5114
848
2019-11-11 12:08:...
9.132
988
IASING.tc.PUS.9.132#I...
20
UNDEF
Accept Time Update
5115
849
2019-11-11 12:08:...
9.132
988
IASING.tc.PUS.9.132#I...
20
UNDEF
Accept Time Update
5116
850
2019-11-11 12:08:...
190.1
988
IASING.tc.PUS.190.1#I...
16
TC_ACCEPTANCE_SUCC...
Request a change mode.
65
851
2019-11-11 12:08:...
9.132
988
IASING.tc.PUS.9.132#I...
20
UNDEF
Accept Time Update
5117
852
2019-11-11 12:08:...
9.132
988
IASING.tc.PUS.9.132#I...
20
UNDEF
Accept Time Update
5118
853
2019-11-11 12:08:...
9.132
988
IASING.tc.PUS.9.132#I...
20
UNDEF
Accept Time Update
5119
854
2019-11-11 12:08:...
9.132
988
IASING.tc.PUS.9.132#I...
20
UNDEF
Accept Time Update
5120
double-click to add filter

Main Window

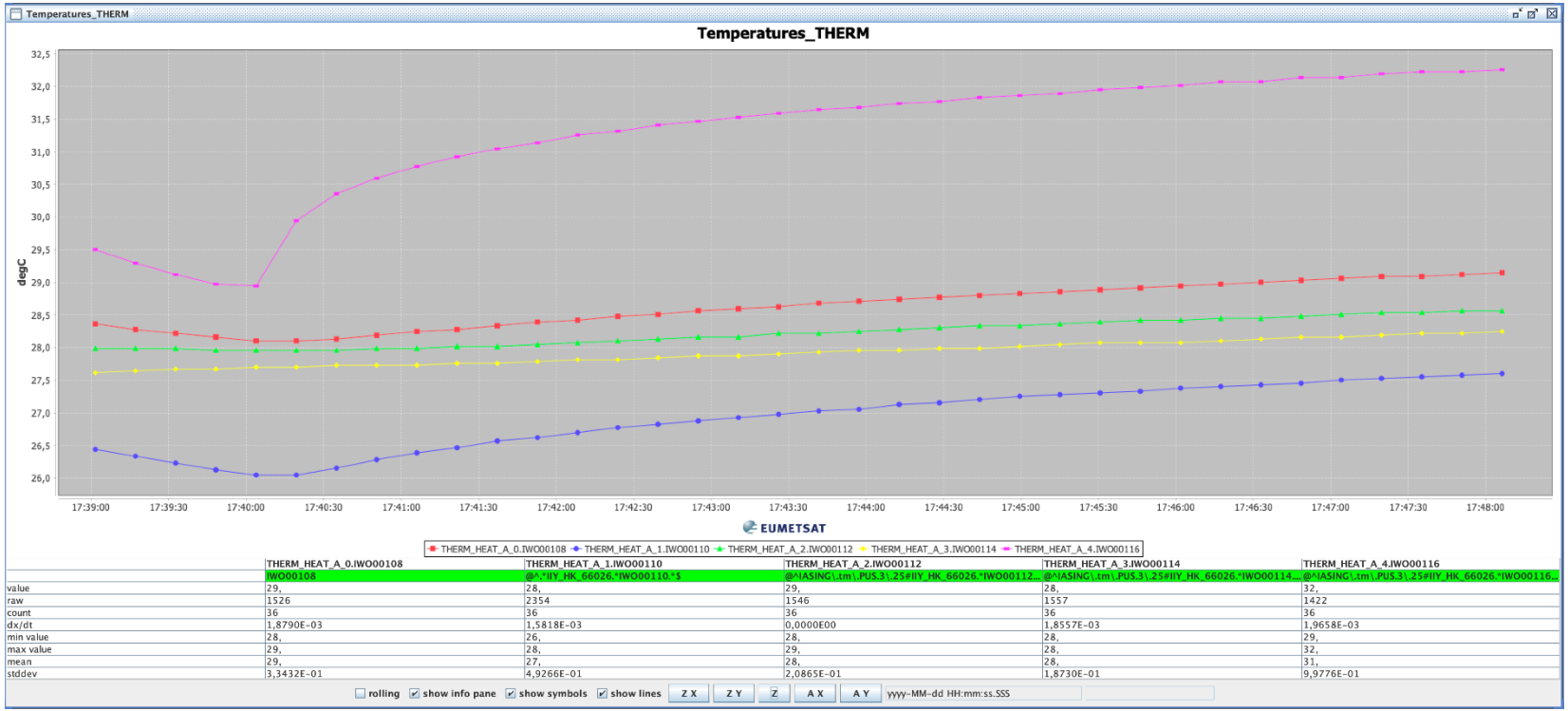

SSC verificaton
Data rate
Telecommands
Instrument events
The IASI instrument
Mode chang...
Mode chang...
Limits violat...
User events
test name: Untitled
packet: 5121 OBT:2004-01-28 17:38:42.773 local: 12:08:01
instrument mode: <unknown>
data server: -
saving stream to: /Users/jakoblivschitz/Desktop/Projects/HKTV_IASING/archive/2019/315_2019-11-11/FM4_20191111_UTC_120736.hkp
EUMETSAT

Main window



The IASI instrument

HKTV Time Series:



	THERM_HEAT_A_0.IWO00108	THERM_HEAT_A_1.IWO00110	THERM_HEAT_A_2.IWO00112	THERM_HEAT_A_3.IWO00114	THERM_HEAT_A_4.IWO00116
	IWO00108	@^.*IIY_HK_66026.*IWO00110.*$	@^IASING\.tm\.PUS.3\.25#IIY_HK_66026.*IWO00112...	@^IASING\.tm\.PUS.3\.25#IIY_HK_66026.*IWO00114...	@^IASING\.tm\.PUS.3\.25#IIY_HK_66026.*IWO00116...
value	29,	28,	29,	28,	32,
raw	1526	2354	1546	1557	1422
count	36	36	36	36	36
dx/dt	1,8790E-03	1,5818E-03	0,0000E00	1,8557E-03	1,9658E-03
min value	28,	26,	28,	28,	29,
max value	29,	28,	29,	28,	32,
mean	29,	27,	28,	28,	31,
stddev	3,3432E-01	4,9266E-01	2,0865E-01	1,8730E-01	9,9776E-01

HKTV core components

- **DataProvider** – pushes packets into the system
- **PacketSaver** – saves received packets
- **PacketFormatDetector** - detects packet format
- **StreamConsistencyChecker** – checks input stream (e.g. OBT or SSC jumps)
- **TimeCorrelation** - OBT <-> Java time
- **Archive** – stores data
- **Automation** – infrastructure to run scripts
- **LimitsChecker**
- **SynopticsDisplay** (e.g. redundancies switch)
- **TimeSeriesDisplay** (e.g. for temperatures)

Input data providers

HKTV implements a bunch of input data providers. As HKTV message bus requires data packets on input, some preprocessing is usually required to extract/generate the packet from the input data. Supported data sources:

- Local files with CCSDS packets, CADU stream, MMFU SCOE dumps, UDMS dumps, IRIG dumps, etc.
- Remote TCP connection (incoming/outgoing): CCSDS packet with some optional headers
- Data loading from CCS5 MySQL database

Data loaders are light-weight components (100-200 lines) extracting the data from the input source and putting onto the bus, so it's very easy to add a new one.

Data savers

HKTV supports several output data formats:

- proprietary HKP-2 format (portable house-keeping data format)
- MHS SD archive data format
- CCSDS packet stream
- Easily extensible to add other data formats

Packet format detector

Default implementation determines packet format based on PUS service/subservice and fixed values as specified in the MIB database.

Tailored mission-specific implementations can be provided overriding the virtual method (e.g. for performance reasons).

Archive

HKTV implements archive in MongoDB.

The archive stores:

- opaque data (bytestream of CCSDS packets received)
- decomposed data (each mnemonic stored separately: high resources need!)

What exactly is stored is configurable: opaque packets, CCSDS packet headers, serialized packets, decomposed packets (individual parameters).

Automation

HKTV implements automation in Groovy language.

From the Groovy environment access is granted to all HKTV components.

Groovy execution is mirrored line-by-line into as-run report.

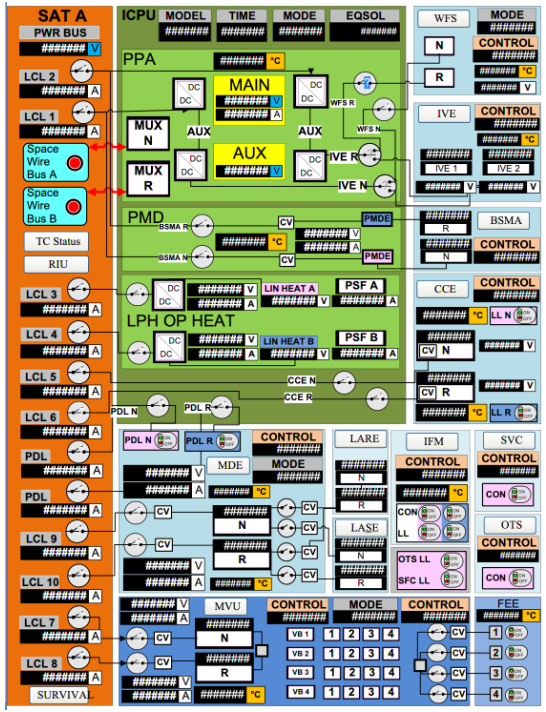
Example to load a set of CADU files:

```
logger.info("starting CADUs data feed")
x = new CaduStreamDataProvider()
app.setDataProvider(x)
path = "/data/REC_2019_09_13_14_11_59_TMRATE_FORMAL_RUN_SMDA/"
x.loadCadu(new File(path + "0.dat"))
x.loadCadu(new File(path + "1.dat"))
x.loadCadu(new File(path + "2.dat"))
x.loadCadu(new File(path + "3.dat"))
logger.info("CADU data feed finished")
```

Synoptics displays

HKTV supports synoptics displays specified in SVG or HTML.

The displays receive the full data feed, including parameter status. JavaScript is used for additional data processing, special effects, etc.



Limits checking

Fully SCOS2000 compatible.

In addition, different limit sets can be chosen (e.g. for ambient, thermal vacuum, etc.)

Data fetching concept

Each monitoring parameter is represented internally in DataField class: can be PhysicalDataField or SyntheticDataField.

To retrieve data, a Fetcher shall be obtained using DataField.getFetcher(). Normally it is a BitStreamFetcher. A fetcher is a highly-optimized class going directly into bit stream and getting the necessary piece of data.

But it's also flexible due to a proxying approach, e.g.

- PatchableFetcher(Fetcher): real-time data patching
- ConditionalFetcher(Fetcher): conditional field
- EnumFetcher(Fetcher): fetch number and apply textual calibration

And this is absolutely transparent to the users.

Performance

Performance depends on the deployment configuration, e.g. archive options, plotting options, synoptic displays, etc.

Reference configuration loading Metop-SG CADU stream processes >5000 CCSDS packets per second on a 14-core Mac Pro (Xeon 2.5GHz). Deployment options: no archive, 4 plots, no synoptic displays.

Switching on MongoDB archive on local machine reduces performance by ~30-90% depending on settings (proprietary HKTV archive has no performance impact).

Build time (gradle): ~2 minutes.

Performance

HKTV memory footprint:

- 15000 IASI-NG packets: 1.3GB
- 365000 MetOp-SG packets: 5.6GB

Scalability

Each data consumer runs in a separate thread. Thus performance on a single machine is limited by the number of cores.

The slowest data consumer defines the length of backlog, since packets have to be kept in memory as long as they have not been processed.

It is possible to connect to the message bus remotely via TCP, thus some processing can be made on a remote machine.

Mission database

MDB is represented by a set of XML files and includes definitions of:

- TM, TC and Science data packets description
- Transfer functions definitions
- Limits definitions
- Plots definitions
- User displays definitions
- Reporting can also be customized, but functioning defaults are provided

Tailoring to a new mission

Tailoring means:

- Creation of a new application class (inherit from AppBase and implement abstract methods), tailor creation of windows and menus
- Provision of mission database (manual or automatic conversion; convertors from SCOS MIB, RangeDB XLS and EGS-CC CDM are available)
- Cosmetic: background, logo, etc.

If needed: implementation of instrument-specific data processing

HKTV in action

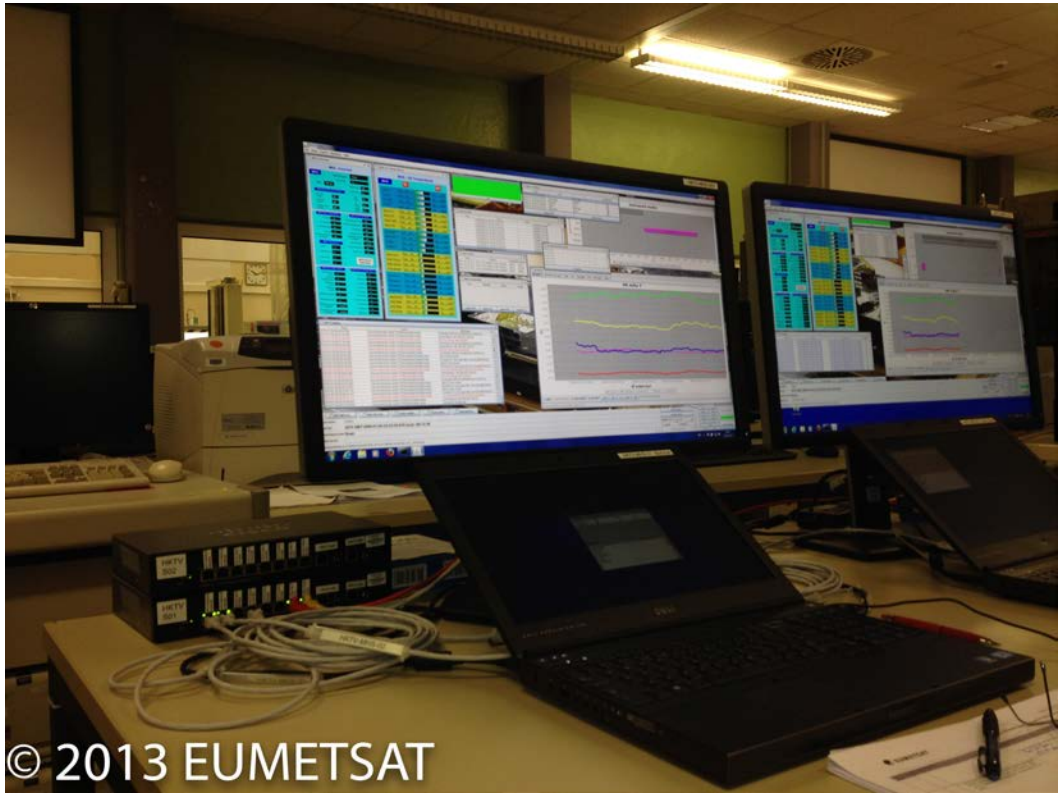
MHS bench:

- 2 DELL laptops
- 2 monitors
- 2 Cisco switches
- 1 printer

HKTV in action

MHS bench:

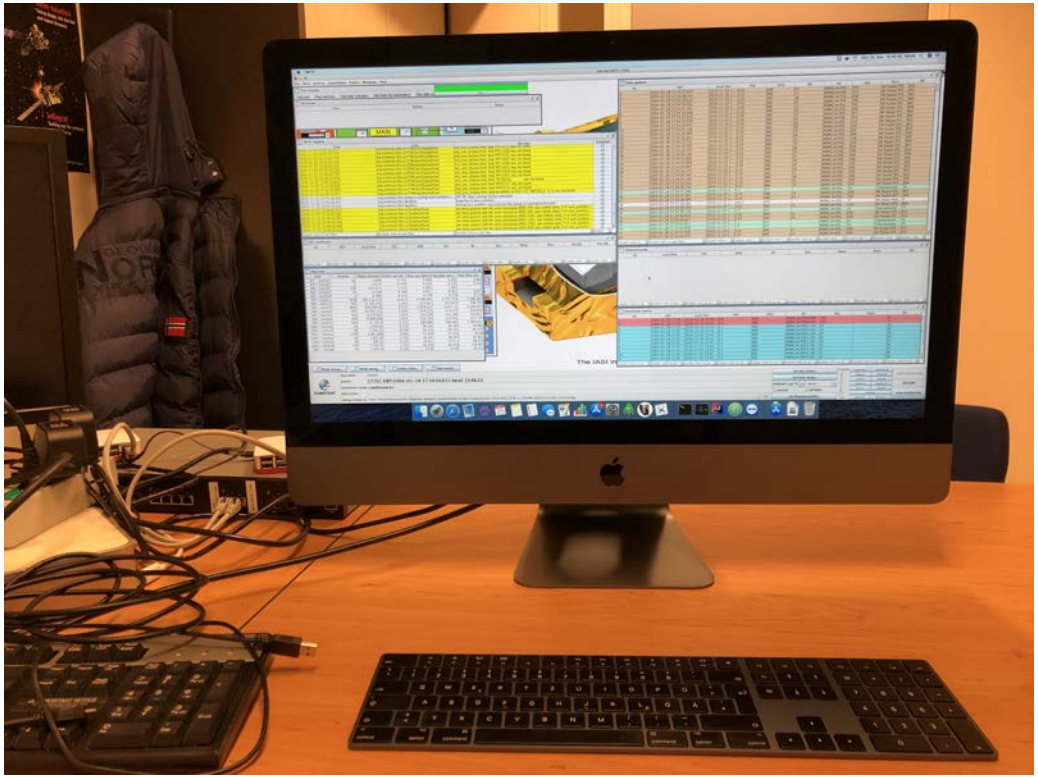
- 2 DELL laptops
- 2 monitors
- 2 Cisco switches
- 1 printer



HKTV in action

IASI-NG bench:

- Mac Pro
- Cisco switch
- CCS5 simulator (2 Raspberry PIs)



Looking back

- Java proved to be the right choice due to its API stability and multiplatform support. HKTV required almost no adaptation moving from Java 5 to Java 13.
- Java Swing proved to be the right choice for UI. Other technology, e.g. SWT, JavaFX or Web-based have not remained that stable over 10 years and would have required significant adaptations.
- Monolithic application is appropriate for such project (~50K LOC). Clean package and component separation is essential. Other approaches e.g. OSGi or service based would have often required reengineering and adaptations.
- Data bus concept proved to be extremely flexible and extensible. Numerous data sources and consumers added without any changes being necessary.
- Build system migrated from ant to gradle: minor change.