Flight Software Telemetry

Dr. Christopher Landauer
The Aerospace Corporation

chris.landauer@aero.org

Computers and Software Division
19 November 2014

Summary

The author recommends that flight software in space systems be instrumented as thoroughly as hardware, if not more so.

- Outline
  - *What is Software Telemetry?*
    - Why is it needed?
  - *How can it be done?*
    - What can be measured?
    - How should it be reported?
  - *Analyses, Models, Anomalies*
  - *Conclusions*

chris.landauer@aero.org
Software Systems Analysis Dept./ Engineering and Technology Group

What is Software Telemetry?

- Measurement of performance aspects of software in flight systems
  - *Intended to*
    - Tell us how well it is working
    - Help us predict problems before they occur
  - *Both data and processes*
    - Especially timing

- This talk is about
  - *What we think needs to be measured*
  - *How the measurements can be usefully reported*
  - *What kinds of anomalies can occur*
  - *What the ground systems can do about the data*

chris.landauer@aero.org
Software Systems Analysis Dept./ Engineering and Technology Group

Why Is It Important Now?

- Software is taking on more and more of the functionality of satellite control
  - *Software has led to an increasing number of system failures*
    - 1996 Ariane 5 was a legacy Ariane 4 error (unchecked overflow)
    - 1999 Milstar was a critical filter coefficient error (unchecked manual input)

- Telemetry is historically about hardware health and status measurements, so we can track system health and performance and the inevitable deterioration thereof

- There are, of course, some fundamental differences between software and hardware
  - *Software is not subject to the laws of physics*
    - Correct behavior is much harder to specify
  - *Software does not change gradually*
    - Slightly wrong software usually produces completely wrong results
      - *(or worse, no results at all)*

chris.landauer@aero.org
Software Systems Analysis Dept./ Engineering and Technology Group

What Can We Learn?

- We expect to be able to learn the same things we learn from hardware telemetry
  - *Nominal Behavior*
    - Over a very wide range of operating modes
    - Over long periods of time
  - *Expected Trajectories*
    - Change models
    - Timing models
  - *Deviations and Anomalies*
    - Trends and fluctuations

- We expect to be able to learn more things also
  - *Code usage density (Especially unused code)*
  - *When software collects a value from a hardware register*
    - Comparison to the reported hardware value is useful
  - *Buffer fill and empty trajectories*
  - *Slack time in schedule*

chris.landauer@aero.org
Software Systems Analysis Dept./ Engineering and Technology Group

What Needs To Be Measured?

- Necessary and helpful measurements
  - *Module Usage*
    - How often called?
    - GNU gprof(1) style profiling
  - *Branch point distribution*
    - Statistics of decisions
  - *Timing*
    - How long for straight segments?
    - How long for function calls?
    - Scheduling margins
  - *All values collected by software from hardware*
    - Verifying the size of the resulting value
  - *Buffer fill values*

- Many other measurands can be considered

chris.landauer@aero.org
Software Systems Analysis Dept./ Engineering and Technology Group

How Can Measurements Be Usefully Reported?

- Reporting measurements usefully
  - *Wide field displays*
    - Code usage density
    - Compare to expectations
  - *Long time displays*
    - Multiple summaries
    - Other time series analyses
  - *Real-time dynamic values*
    - Buffer fill fractions
    - Schedule and margin
    - Time series and derived model predictions
  - *Fault Management process progress*
    - Detect, Identify, Address, Fix or announce Error

- Many other reports can be considered

chris.landauer@aero.org
Software Systems Analysis Dept./ Engineering and Technology Group

What Kinds of Anomalies Can Occur?

- Software anomalies
  - *Unused code*
    - Inefficiencies of wasted memory and upload bandwidth
    - Potentially damaging because not usually adequately tested
  - *Buffer overflow*
    - Buffer fill increase, predicted overflow
  - *Schedule overrun*
    - Segment timing increase, predicted overrun

- Some faults are software faults
  - *Hardware fault exacerbated by software response*
  - *Software assertion error*
  - *Function input out of bounds*

- Many other anomalies can be considered

chris.landauer@aero.org
Software Systems Analysis Dept./ Engineering and Technology Group

What Can The Ground Systems Do?

Ground System Analyses

- All data arrives as time-stamped sequences of measurands
  - *of various (known) reporting period lengths or repeat interval times*

- Trajectory analysis
  - *Long term temporal models*
    - Trends, extremes, and boundaries
    - Outlier detection
    - Kalman filters for noise reduction
  - *Periodicities*
  - *Accumulation points*
  - *Diurnal and other periodic effects (eclipses, resonances)*
  - *Grammatical Inference for sequence structure identification*
    - Compare to top down models for anomaly and attack detection

chris.landauer@aero.org
Software Systems Analysis Dept./ Engineering and Technology Group

What Can The Ground Systems Do? (cont.)

Ground System Analyses

- Correlations among different trajectories
  - *Some may be weakly predictive of others*
  - *Fault management based on inconsistencies*
    - E.g., measurands should be correlated, but they aren't

- Trajectories (sequences) of values are points in a very high-dimensional space
  - *Manifold discovery and dimension reduction in trajectory spaces*
  - *Discover operational constraints, notice when they are (about to be) violated*

chris.landauer@aero.org
Software Systems Analysis Dept./ Engineering and Technology Group

Conclusions

- The author recommends serious consideration of software telemetry
  - *For the basic operating system / real-time executive*
  - *For the flight software applications*
  - *For the mission data handling*
- There are many methods ready to use
  - *and many more in the research and development pipeline*

chris.landauer@aero.org
Software Systems Analysis Dept./ Engineering and Technology Group