Software and System Health Management with R2U2

Johann Schumann†

t†SGT, Inc., NASA Ames Research Center

Johann.M.Schumann@nasa.gov

NASA ARC-Boeing 2019
Abstract

R2U2 (Realizable, Responsive, Unobtrusive Unit) is a hardware-supported tool and framework for the real-time system and software health management of cyber-physical systems. R2U2 continuously monitors properties about safety, performance, and security of the vehicle and can perform diagnostic reasoning. Efficient observers for past-time and future-time Metric Temporal Logic, reasoners for Bayesian Networks, and model-based prognostics algorithms are major components of R2U2. Their combination makes it possible to design powerful models for system runtime monitoring, diagnostics, software health management, prognostics, and security monitoring.

The R2U2 monitoring engine is designed for minimal runtime overhead and is available as Simulink block or as a software component for integration into the flight software stack, and enables R2U2 to monitor complex cyber-physical systems without any instrumentation of the flight software.

In this presentation, we give an overview of R2U2 architecture and reasoning algorithms, present its features, and give a life demo of the tool.
Overview

\[ R_1 : \text{start-talk} \rightarrow \Diamond [40,50 \text{min}]” \text{LAST SLIDE}” \]

R2U2

- **RESPONSIVENESS**: respond in “real time”
- **REALIZABILITY**: plug-and-play
- **UNOBTRUSIVENESS**: do not “mess up” the flight software
- **UNIT**

R2U2 is a Run-time monitoring and V&V tool that combines *Metric Temporal Logic* observers, *Bayesian Network* reasoners, and *model-based Prognostics* for high expressiveness.
Why all the R2U2 Ingredients?

Different health management approaches focus on specific properties

- Boolean Logic (assertions)
  - Limit Checker (cFS), Monitors
- Model-Based Diagnostics and Monitoring
  - HyDE
  - TEAMS
  - Prognostics
- Temporal
  - Timed Failure Propagation Graph (TFPG)
- Probabilistic
  - Bayesian Networks
Why all the R2U2 Ingredients?

Different health management approaches focus on specific properties

- Boolean Logic (assertions)
  - Limit Checker (cFS), Monitors
- Model-Based Diagnostics and Monitoring
  - HyDE
  - TEAMS
  - Prognostics
- Temporal
  - Timed Failure Propagation Graph (TFPG)
- Probabilistic
  - Bayesian Networks

Our R2U2 framework provides powerful mechanisms to enable temporal, probabilistic diagnostic models integrating advanced prognostics models.
Overview of the Talk

- R2U2 Architecture
- Metric Temporal Logic
- Bayesian Networks
- Prognostics
- R2U2 Simulink Integration and Demo
- R2U2 for Security Monitoring
- R2U2 Hardware Variants
- Summary
R2U2 Architecture

- R2U2 as Simulink Block
- R2U2 as software app for cFS/cFE Core Flight System
- R2U2 on parallel co-processor (Epiphany)
- R2U2 as stand-alone software
- R2U2 on Field Programmable Gate Array (FPGA)
Future Time Metric Temporal Logic (MTL) reasons about bounded timelines:

- Atomic propositions: \( p, q \)
- Operators: \( \neg, \land, \lor, \rightarrow, \leftrightarrow, \circ, [2,6], \diamondsuit, \triangledown, \bigtriangledown, \Rightarrow \)

\[ [2,6]p \quad \text{ALWAYS}[2,6] \]  
\[ [2,6]p \quad \text{EVENTUALLY}[0,7] \]  
\[ p \triangledown q \quad \text{UNTIL} \]

- Past Time: similar with Historically, Once, Since
Observer Pairs

For each future time MTL formula, we create two observers:

- **asynchronous** observers return \( \{ T, F \} \)
  - results are not instantaneous
  - observer has considerable complexity and needs local memory

- **synchronous** observer returns \( \{ T, F, \text{maybe} \} \) at each timestamp
  - results immediately available
  - observer has low complexity
  - three-valued logic can be useful for reasoning

We do not translate MTL formulas into finite state machines; rather we use synchronisation queues [TACAS’2014]
Example: Safety Rule

After receiving a command for takeoff, the UAS must reach an altitude of 600ft within 60 seconds.

\[ \square((\text{cmd} == \text{takeoff}) \rightarrow \Diamond_{[0,60s]}(alt \geq 600 \text{ ft})) \]
Example: UAS Monitoring I (Past Time)

If the battery current is high for the last 10 seconds, we expect that during that time, we have a consecutive climb for at least 3 seconds

\[ \square_{[10s]}(I_{\text{batt}} > 28A) \rightarrow \Diamond_{[10s]} \square_{[3s]}(V_z > 5\text{ft/s}) \]
Example: UAS Monitoring II (Past Time)

When in the mode “guided” or “auto”, after a change of the target heading, the NAV heading needs to be aligned within 5 timestamps. Short glitches in the alignment should not count.

\[(\text{mode} \equiv \text{guided} \lor \text{mode} \equiv \text{auto}) \rightarrow (\Diamond[2]\text{heading\_achieved} \lor \Diamond[3]\text{nav\_bearing\_changed})\]
DEMO: R2U2 Visualizer

The R2U2 Visualizer runs under Matlab and Simulink and enables the interactive exploration and validation of R2U2 Temporal Formulas.
Ingredient II: Bayesian Reasoning

Our Bayesian Networks contain
- unobservable health nodes $H$
- unobservable behavior nodes
- observable sensor nodes $S$

During operation
- inputs are clamped at observable $S$ nodes
- posteriors of the health nodes $H$ reflect the most likely health status of the component

Efficient implementation using Arithmetic Circuits

We use SamIam for modeling: http://reasoning.cs.ucla.edu/samiam/
Example 1: Sensor Monitoring

UAS with Baro, Speed (Pitot), LIDAR, and GPS sensor must fly safely over mountaineous terrain. Sensor failures must be recognized.
Example 1: Sensor Monitoring: Simulation Results

**Baro Failure**

- GPS alt [m]
- $\alpha$ [°]
- Timestamp [x2s]

**GPS Failure**

- GPS alt [m]
- Timestamp [x2s]
Ingredient III: Model-based Prognostics

\[ \text{start-climb} \rightarrow \square_{[10\text{min}]}\text{battery-OK} \]

- **Not Good:** valuation only after 10 minutes or ”maybe”

\[ \text{start-climb} \rightarrow (RUL > 10\text{min}) \]

- **Good:** model-based prognostics. Result available *now*

R2U2 uses electro-chemical model for LiPo batteries and UKF-based algorithm
Signal Processing

R2U2 generates customized C code for filtering and signal processing from a compact specification

- FIR low-pass filter of order $n$
- Moving average with window size $n$
- FFT of length $n$
- Rate filter (for scalar or angle)
- Linear interpolation
- Regular expression for strings
- Prognostics
- ... (easy extensible)
DEMO: R2U2 Simulink Integration
Security Monitoring for Autonomous Vehicles

- Autonomous vehicles are an easy target for malicious attacks
- Multiple attack surfaces via:
  - malicious commands sent to the UAS,
  - compromised flight software
  - GPS spoofing
- Threat detection and diagnosis has to be performed on-board

R2U2 performs system behavior monitoring and reasoning to detect malicious attacks
Example: Malicious Commands cause Oscillation

Detect attack via dangerous GCS commands that cause UAS oscillation

- received commands change gains of control loops
- we use FFT to detect oscillations in low $O_L$ and high $O_H$ frequencies
- monitoring by:
  $□(C \land \Diamond [0,500] (□[0,200]O_H))$
  $□(C \land \Diamond [0,1200] (□[0,300]O_L))$
Example: GPS Jamming and Spoofing

- Autonomous vehicles heavily rely on GPS for navigation
- Attack 1: Jamming comms with a powerful transmitter
- Attack 2: Spoofing: a valid GPS signal is overlaid with an attack signal corresponding to slightly different coordinates

The UAS is flying to where the bad guy wants it.
Experiment: Detection of GPS Spoofing

R2U2 can detect GPS spoofing by monitoring multiple signals of the FSW and correlating GPS, flight path, and inertial navigation.

- actual and desired latitude
- Filtering output of mixed signals (North)
- Filtering output of mixed signals (East)
- Bayesian Network can be used for further disambiguation
Unobtrusive Monitoring

The flight software must not be instrumented or modified. Still (global) variables of interest must be monitored while the software is running.

- mapping of variables of interest to *shared* memory buffer
  - special memory region
  - frame buffer
- accessible from R2U2 via Direct Memory Access (DMA)
- fast hardware synchronization

- no modifications or instrumentation of code
- minimal changes in timing
Scalability: Parallelization of R2U2

Three levels of parallelization

1. Parallel pipeline of
   - AT: signal preprocessing
   - TL: temporal reasoning
   - BN: Bayesian reasoning

2. Parallel execution of AT filters

3. Parallel evaluation of TL formulas

Different sets of intermediate registers enable us to run AT, TL, and BN in parallel without having to synchronize the shared data.
Introduction

R2U2 Architecture

R2U2: Demo

Applications

Hardware Variants

R2U2 and the "V"

1. Fret
   Safety requirements specification

2. Cocosim
   Safety requirement verification on Simulink models

3. IKOS
   Static code analyzer for C/C++ with low false positive rate

4. MARGInS
   Validation testing to identify unusual behaviors

Code

Integration Testing

Unit Testing

Software Development

System/SW Design

Requirement Engineering

Acceptance Testing

R2U2

IKOS

Static code analyzer for C/C++ with low false positive rate
Selected Publications

Team

- Johannes Geist
- Timmy Mbaya
- Thomas Reinbacher
- Julian Rhein
- Kristin Rozier
- Johann Schumann

PoC: johann.m.schumann@nasa.gov or johann.schumann@gmail.com