Institution: University of Manchester		
Unit of Assessment: 11 (Computer Science and Informatics)		
Title of case study: Rule-based analysis for talking to spacecraft		
Period when the underpinning research was undertaken: 2002 – 2013		
Details of staff conducting the underpinning research from the submitting unit:		
Name(s):	**Job title:**	**Periods employed by HEI:**
Howard Barringer	Professor	1 Oct 1975 – 30 Sept 2011
David Rydeheard	Senior Lecturer	1 Oct 1983 – 30 Sept 2018
Period when the claimed impact occurred: September 2013 – December 2020		
Is this case study continued from a case study submitted in 2014? N		

1. Summary of the impact

Spacecraft missions represent enormous investment (billions of dollars) for NASA. It is critical to protect this investment by ensuring missions operate safely, and perform as expected. Researchers at The University of Manchester developed novel fundamental methods in runtime verification, that were implemented in two of NASA's software tools - **TraceContract** and **LogFire -** used to analyse commands (sent to spacecraft) and telemetry (received from spacecraft).These tools (i) **protect the multi-billion dollar investments of NASA**, providing automated methods to identify bugs in command sequences, and (ii) **provide human readable interpretation** of sensor readings, enabling NASA engineers to continuously and reliably monitor the health of the spacecraft. These tools were (and are) **used on a daily basis** to guarantee safe operation in high profile NASA missions:

- **LADEE** lunar mission (Sept 2013 - April 2014: overall cost USD280,000,000),
- **Curiosity Rover** on Mars (Nov 2011 - present: overall cost USD2,500,000,000)

2. Underpinning research

Identifying a problem in theoretical computer science

In 2003, whilst on sabbatical from The University of Manchester (UoM) at The National Aeronautics and Space Administration's (NASA) Ames Research Centre, Barringer met NASA scientist, Klaus Havelund, and was introduced to a problem in theoretical computer science - how to define formal languages and algorithms that can express and efficiently verify temporal properties of long sequences of structured records. This was known as the "first-order trace-checking" problem - an exponential time complexity for certain types of temporal properties, those with *high data interdependencies*. At the time no practical algorithms existed.

Addressing the real-world challenge

NASA's interest in this problem developed from the fact that long command sequences are transmitted to, and telemetry is received from, spacecraft on a daily basis. The temporal properties NASA needs to check are in regard to *safe operations*, and by nature have *high data interdependencies*, caused by the many interacting sensors and actuators on spacecraft. This problem occurs both in *live* missions (e.g. Curiosity Rover currently on Mars), and in ground simulations for *future* missions.

The challenge was therefore how to formally express complex properties that assure the safety of spacecraft operation, and provide efficient automated checks of these properties. This required fundamental developments in the field of *runtime verification,* addressing the "first-order trace-checking" problem.

Novel research:
From 2003 onwards, UoM researchers developed a series of novel specification formalisms and monitoring algorithms to address this "first-order trace-checking" problem:

- **Eagle** [1] introduced a novel temporal finite trace-monitoring logic, which for the first time enabled both recursive and parameterized expressions. This in turn enabled specification and verification of requirements in a first-order fixed-point temporal logic. However, a weakness was non-determinism of fixed-point operators (which caused some operations to be computed in polynomial or exponential time).
- **RuleR** [2] addressed this weakness, developing a *small-step trace-semantics*, as opposed to the big-step semantics present in Eagle. This enabled a non-backtracking algorithm, applied and demonstrated for the first time in [3] within the target research community of Aerospace Engineering.
- The **QEA** approach [4] developed a novel partial ordering data structure, which further reduced the time complexity compared to [1] and [2] (this also won the international Runtime Verification competition, 2014-15).

In parallel to the underpinning research [1 – 4], Barringer worked with NASA to implement the algorithms as software [5] (see section 4). NASA and UoM continue a close working relationship, with the UoM team's research being disseminated within NASA to support development of additional software and runtime verification tools.

3. References to the research

The research was published in leading journals and conferences in the field (e.g. FM, RV, and VMCAI, typically only 25% acceptance). The quality of the research in the target community (Aerospace Engineering) is demonstrated by [3]. All citations are from Google Scholar, October 2020.

[1] Barringer H., Goldberg A., Havelund K., Sen K. (2004) Rule-Based Runtime Verification. In: Steffen B., Levi G. (eds) Verification, Model Checking, and Abstract Interpretation. VMCAI 2004. *Lecture Notes in Computer Science*, **2937**. Springer, Berlin DOI/10.1007/978-3-540-24622-0_5 (445 citations)

[2] Barringer H., Rydeheard D.E., and Havelund K. (2010) Rule Systems for Run-time Monitoring: from Eagle to RuleR. *Journal of Logic and Computation*, **20** (3), 675–706, DOI: 10.1093/logcom/exn076 (179 citations)

[3] Barringer H, Groce A., Havelund K., and Smith M. (2010). Formal Analysis of Log Files. *Journal of Aerospace Computing, Information, and Communication*, **7** (11): 365-390. DOI:10.2514/1.49356 (102 citations)

[4] Barringer H., Falcone Y., Havelund K., Reger G., and Rydeheard D., (2012) Quantified Event Automata: Towards Expressive and Efficient Runtime Monitors. In: Giannakopoulou D., Méry D. (eds) FM 2012: Formal Methods. FM 2012. *Lecture Notes in Computer Science*, **7436**. Springer, Berlin, DOI: 10.1007/978-3-642-32759-9_9 (142 citations)

[5] Barringer H., and Havelund K. (2011) TraceContract: A Scala DSL for Trace Analysis. In: Butler M., Schulte W. (eds) FM 2011: Formal Methods. FM 2011. *Lecture Notes in Computer Science*, **6664**. Springer, Berlin. DOI: 10.1007/978-3-642-21437-0_7 (105 citations)

4. Details of the impact

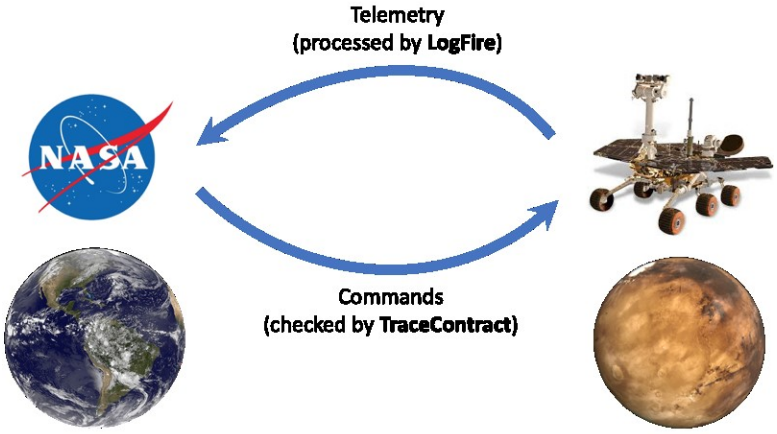
Context

NASA is an independent agency of the U.S. Federal Government. One of NASA's primary activities is the launch and monitoring of spacecraft around the solar system. These spacecraft are typically multi-billion dollar investments by the U.S. Government, and thus require an exceptionally high degree of safety within NASA's daily operations in order to protect that investment.

NASA spacecraft receive command sequences sent from NASA ground stations for their control, and transmit telemetry (sensor signals) back to NASA, giving indications of their surroundings. This requires constant work to ensure command sequencing is free of defects to avoid damage to the spacecraft, and that data can be continuously and reliably used by NASA engineers. This checking uses methodologies from the field of "runtime verification". Prior to this research, common practice for this checking at NASA's Jet Propulsion Laboratory (JPL), was to develop *ad-hoc* tools using various scripting languages, resulting in a growing collection of scripts [A]. These were hard to maintain and modify, as well as having many points of potential failure. This became a concern for long-running missions over many years.

Pathway to impact

In 2003, Barringer took a sabbatical at the NASA Ames Research Centre in California. This visit established a relationship between UoM and NASA, and enabled a programme of industry-led research problems to be discussed, developed, and addressed. Barringer worked with NASA to implement the initial research [1,2,3] as software, resulting in TraceContract (see below) [5]. Subsequent research at UoM [4] generated novel and efficient data structures which informed further software by NASA, resulting in LogFire (below) [B].



TraceContract is a domain-specific programming language (DSL) suitable for runtime verification, implemented by Barringer and NASA [5]. The DSL is implemented in Scala, but incorporates the core RuleR language [2] with temporal operators. This was used to create "*Flight Rule Checkers"*, ensuring safe operations for the transmitted command sequences, and has been deployed in NASA missions since 2013.

LogFire is a software tool implemented by NASA, which relies fundamentally on the rule-based approach introduced by RuleR [2] and enables engineers to generate abstract (human-readable) events from the (non-human readable) events in log files coming from spacecraft telemetry. This allows them to assess spacecraft health and operational status. The tool remains in use within NASA, and as confirmed in September 2020 by a senior scientist within the Jet Propulsion Laboratory (JPL) at NASA *"With its demonstrated effectiveness, this tool [Logfire] has been incorporated as a standard feature for future ground systems and will have lasting benefits to JPL operations."* [A, C].

Reach and significance of impact

In 2014, NASA had a cumulative annual budget of USD17,646,000,000, increasing to USD22,689,000,000 in 2020 [D] and therefore this research is having impact in a multi-billion dollar organisation.

Senior NASA Scientist, Klaus Havelund, confirms that "*the research of Barringer et al. has deeply influenced the development of a new generation of runtime verification technology, used in real NASA missions […] The TraceContract and Logfire tools have been an asset to our activities, and have possibly made certain mission-critical operations easier to perform than they would have been without the tools"* [A]. Specific examples of missions that have used TraceContract and LogFire within this REF period are:

The LADEE mission

The Lunar Atmosphere and Dust Environment Explorer (LADEE) was a robotic mission that orbited the moon to gather detailed information about the structure and composition of the thin lunar atmosphere, and determine whether dust is lofted into the lunar sky. LADEE launched on 7 September 2013, and lasted seven months, ending on 18 April 2014, at a cost of USD280,000,000 [A].

The **TraceContract** tool was used **on a daily basis throughout the 7-month mission** to ensure that command sequences sent to the LADEE spacecraft satisfied pre-defined flight rules, designed to ensure safe operation [A][E]. TraceContract provided automated checks of the daily command sequences *before* they were uploaded, providing assurance to the ground staff. Throughout the 7-month mission, **the tool processed over 250 command sequences, and identified safety issues in roughly half of these.** The NASA engineer responsible for command sequencing and verification, when asked how often errors were found, responded [E]:

> *"I would actually say it was often, perhaps every other command sequencing cycle."*

The Curiosity Rover mission

The Curiosity Rover mission to Mars is one of NASA's most high-profile recent missions, costing USD2,500,000,000 [A]. Since 2014, the **LogFire** tool has been used on a daily basis to analyse telemetry received from the Curiosity Rover [F]. These telemetry streams can contain millions of events [G] and can therefore be difficult to comprehend by humans, as well as interpret against the higher-level execution plans submitted to the spacecraft.

The output of **LogFire** feeds into visualisation software used to monitor the health of the rover. This is used in ground modelling and simulation work, providing essential information for NASA engineers on a daily basis. NASA Engineer Klaus Havelund has confirmed [A] that:

> *"The Curiosity Rover remains on Mars to this day, and thus the research of Barringer et al. continues to have a significant impact on NASA's activities."*

In summary, the software developed from the UoM research, has mitigated against losses (both monetary and public image) through improved methods in safety and security critical situations.

5. Sources to corroborate the impact

[A] Letter of Support from Klaus Havelund, Senior Research Scientist, NASA Jet Propulsion Laboratory, (21 November 2020)
[B] NASA Tech Briefing (August 2012) - *Internal NASA tech report on **LogFire**, citing the collaboration of Havelund and Barringer.*
Available at https://ntrs.nasa.gov/citations/20120013237 - pdf on file

[C] *NASA web page "Avionics and Flight Software" - Cached website dated 8 Sept 2020 saved as pdf [Accessed 14th October 2020]*

[D] NASA Summary budget reports:
[D (i)] 2014 Summary budget report
[D (ii)] 2020 Summary budget report

[E] Kurklu, E., and Havelund, K (2020) A Flight Rule Checker for the LADEE Lunar Spacecraft, Keynote Talk at the 17th International Colloquium on Theoretical Aspects of Computing (ICTAC 2020)., China 30 October 2020 - 4 December 2020, Available at https://www.havelund.com/Publications/frc-ladee-ictac-2020.pdf - pdf on file

[F] Publication confirming LogFIRE is used within the Curiosity Rover mission: Havelund K., and Joshi R. (2015) Experience with Rule-Based Analysis of Spacecraft Logs. In: Artho C., Ölveczky P. (eds) Formal Techniques for Safety-Critical Systems. FTSCS 2014. Communications in Computer and Information Science, vol 476. Springer, Cham. DOI:10.1007/978-3-319-17581-2_1

[G] Kauffman, S., Havelund, K., Joshi, R. and Fischmeister, S., (2018) Inferring event stream abstractions. Formal Methods in System Design, 53, 54–82, DOI:10.1007/s10703-018-0317-z