Patching Flight Software on Mars

Ed Benowitz / Mark Maimone
MSL Flight Software Developers
NASA/Jet Propulsion Laboratory
Caltech

Copyright 2015 California Institute of Technology. Government sponsorship acknowledged.
Agenda

- Spirit/Opportunity Rover Mission Overview
- Curiosity Rover Mission Overview
- Flight Software Architecture Overview
- Options for Changing Software in Flight
- Goals for Patching
- General Patching Approaches
- Trades
- How a FSW Image is Loaded
- Patch Scenarios
- Lessons learned
NASA’s Mars Exploration Rovers were designed to survive 90 days.

But Spirit explored Mars for 6 years!

A stalled wheel motor led to a serendipitous discovery proving Mars had freestanding pools of water.

FSW patches helped keep Spirit going even with failed actuators.
Opportunity is now in its 12th year of Mars Exploration!

Opportunity holds the distance record for roving across a planetary surface (over 42km)

There have been 6 major FSW updates and tens of hot or cold patches.
Spirit and Opportunity Rover Mission Overview

• Size
  – About the size of a golf cart -- 5 feet long (not including the arm), 6 feet wide and 5 feet tall

• Arm reach
  – About 3 feet

• Weight:
  – 185 kilograms (400 pounds)

• Mission:
  – Search for and characterize a wide range of rocks and soils that hold clues to past water activity on Mars.
Curiosity Rover Mission Overview

• Size
  – About the size of a small car -- 10 feet long (not including the arm), 9 feet wide and 7 feet tall or about the height of a basketball player

• Arm reach
  – About 7 feet

• Weight:
  – 900 kilograms (2,000 pounds)

• Mission:
  – To search areas of Mars for past or present conditions favorable for life, and conditions capable of preserving a record of life
Flight Software Overview

- MER and MSL rovers all run the VxWorks operating system
  - Supports executing shell commands
    - Change variables
    - Dynamic code loading and execution
- MSL has over 100 flight software modules, most run as their own task
- Communication between tasks
  - Inter-process communication (IPC) messages
    - Send a message
    - Wait for a reply
    - Proceed
Options for Changing Software in Flight

- Uplink and install an entire flight software image
- Patch an existing flight software image
  - Modify or extend the existing onboard flight software image
- Extend FSW with a new software component
  - Dynamically load new code
  - Add a new .o and add new commands, telemetry, and data products using a FSW-provided API
Why Patch?

- Patching can:
  - Add entirely new functionality to flight software
    - Could include new commands and telemetry
    - May be necessary to work around newly discovered hardware behavior
    - Increase science return
  - Fix a flight software bug
    - Patching potentially a faster process than a full FSW image
      - A full FSW load typically requires that Validation and Verification (V&V) tests run against all FSW capabilities, even ones that are not changing
General Patching Approaches

- **Hot patch**
  - Changes code in RAM while we are running
  - VxWorks also allows dynamic loading of .o files
- **Cold patch**
  - Modify the flight software image stored in a non-volatile storage area.
  - The new code image would be used on a subsequent boot
  - Does not change the currently running code
- **Trades**
  - Risk
    - Simpler uninstall for a hot patch
      - Reboot would restore flight software back into a clean version.
  - Testing effort
  - Uplink bandwidth
General Patching Approaches

• Patches may need to
  – Add code to an existing function
  – Replace a buggy function with a new function
  – Remove code from an existing function
  – Add new global variables
  – Change or assign references to existing global variables, e.g. function pointers
  – Add new commands and telemetry

• And just in case the patch doesn’t work
  – Have a plan for un-installing the patch
• Trade: Should flight software provide commands for patching?
  – MER and MSL had some support for patches
    • MER: Could apply patches to a copy of FSW in RAM and write to non-volatile storage
    • MSL has a command for adding a new FSW component module
      – But hot patches that change existing code require back-door methods
• Issues with a general built in patch capability
  – Trades FSW development resources against Ops resources
  – A small code change can result in a large difference in the FSW binary
    • If memory locations move, the size of a diff file can be large
  – Both missions had small amounts of RAM
    • MSL could not add padding inside binaries to allow for reduced diff file sizes
    • MER did have room to add padding
Memory Start

Green – No change
Red – At least one bit changed

Memory End
How MSL Boots

• The boot loader loads the FSW image from the currently selected NOR bank
  – Copy from NOR to RAM
  – Execute the image image from RAM

• If the image fails to load, the boot loader uses another NOR bank
How MER boots normally

- MER had a Rad6K flight computer
  - FSW images were stored in EEPROM
  - Multiple, different FSW images could be stored
How A Full New Image is Installed

• The ground uplinks a new binary image to the file system
• Upon command, the new image is saved
  – Flight software loads the image into RAM
  – Flight software burns the NOR zone with the image

• MER
  – In addition to the above, MER also allowed patching the RAM image before burning to its nonvolatile storage in EEPROM
• A bug affecting both Cruise and Surface operations required the removal of several lines of code
  – Cruise: The spacecraft is always on
    • Hot patch approach: Patch the code in RAM only
      – Replace the original code with no-ops
        » A memory assignment
  – Surface: The rover has a wakeup/shutdown cycle
    • Cold patch approach: Change the image in NOR
      – Load the original image into RAM
      – Modify the RAM image to add the no-ops.
      – Save the newly modified image into NOR
      – The bug fixes are retained across reboots
MSL Patch Scenario: Replacing a function

- Add code to a function to fix a bug
  - The function referenced global variables
- Implemented as a hot patch.
- Use a replacement function
  - Create a new .o file to be dynamically loaded by Vxworks containing
    - Replacement function
    - Pointers to the global variables
- Installation
  - Load the .o
  - Find the address of the old function
  - Poke an instruction into the old function to jump to the new function
  - Find the address of the global variables, and assign them to the new global variable pointers
- Install performed on every boot by a sequence
MSL Patch Scenario:
Adding new functionality

- MSL can add new functionality
  - Special commands can install a new FSW component .o containing new code, commands, telemetry

- Registration to add new functionality
  - The new .o is dynamically loaded during the boot process
  - The new code may register new commands, telemetry, data products
  - The new code can hook into the existing wakeup and shutdown process
    - Functions are registered.
  - The registration is performed once
    - Registration information is stored in non-volatile memory
Lessons Learned

- We had to use back-door methods to change small parts of the existing flight code on MSL
  - Even though explicit commands were provided to add new functionality and load an entire new flight software image
- Onboard patching of small code changes was frequent enough to consider adding a patch command in future missions.
  - Running hot patch setup sequences has become a standard part of our nominal sequencing process
Thanks

- MSL flight software team
  - Led by Ben Cichy
  - Special thanks to Danny Lam

- MER flight software team
  - Led by Glenn Reeves
QUESTIONS?
BACKUP
Mars facts

• Size:
  – Half the size of Earth, but same land area
• Weight:
  – 1/10th of what the Earth weighs
• Gravity:
  – 38% as strong as on Earth
• Average Temperature:
  – -81 degrees Fahrenheit
• Atmosphere:
  – Mostly carbon dioxide, with some water vapor
• Moons:
  – 2, Phobos and Deimos
MSL Flight Computer

- Single Board Computer
  - RAD 750
- On-board memory includes 128 MB of volatile DRAM
- 4 GB of NAND non-volatile memory on a separate card
- Both with error detection and correction
- Runs on two flight computers
  - Prime and backup
Cameras

- **Cameras**
  - **Four Pairs of Engineering Hazard Avoidance Cameras (Hazcams):**
    - Mounted on the lower portion of the front and rear of the rover, these black-and-white cameras use visible light to capture three-dimensional (3-D) imagery.
  - **Two Pairs of Engineering Navigation Cameras (Navcams):**
    - Mounted on the mast (the rover "neck and head"), these black-and-white cameras use visible light to gather panoramic, three-dimensional (3D) imagery. The navigation camera unit is a stereo pair of cameras, each with a 45-degree field of view
  - **Four Science Cameras:**
    - MastCam (one pair), will take color images, three-dimensional stereo images, and color video footage
    - The Mars Hand Lens Imager is the equivalent of a geologist's hand lens
    - Chemcam
Arm and Hand
• Curiosity drilled on Mars
Instruments on Curiosity

- ChemCam
  - Will fire a laser and analyze the elemental composition of vaporized materials from areas smaller than 1 millimeter on the surface of Martian rocks and soils.

- Chemin:
  - The Chemistry and Mineralogy instrument, or CheMin for short, will identify and measure the abundances of various minerals on Mars.

- SAM
  - The Sample Analysis at Mars features chemical equipment found in many scientific laboratories on Earth. SAM will search for compounds of the element carbon.

- REMS
  - A weather monitoring station

- RAD
  - Measures radiation
Communication

- Curiosity can communicate directly to Earth
- Typically Curiosity talks to satellites orbiting Mars
  - These orbiters then forward data to Earth
Photos from Mars
References

