Evolution of Seqgen – A Spacecraft Sequence Simulator

Barbara Streiffert
Taifun O’Reilly
Jet Propulsion Laboratory,
California Institute of Technology
Pasadena, CA

Barbara.Streiffert@jpl.nasa.gov
Taifun.Oreilly@jpl.nasa.gov

SpaceOps 2008 Heidelberg, Germany May 12 – 16, 2008
Agenda

• Background
• Original Seqgen
• Seqgen Core and Adaptation
• Seqgen Execution
• New Capabilities
• Changes to Seqgen
• Mission Formulation Example
• Future Seqgen
• Conclusion
Background

• Seqgen is:
  • A multi-mission software application
  • Used by most JPL missions for validation of commands to be sent to the spacecraft
  • Written in C++
  • Runs on Unix platforms (originally – now it runs on Linux too)
• A Discrete Event Simulator
  • Commands are the discrete events
Seqgen:

- Simulates spacecraft flight and Deep Space Network ground events
- Uses a ‘little interpreted language’ for defining spacecraft activities and spacecraft command models
- Performs hardware constraint checking
- Tracks resource usage over time
- Produces predicts
- Has graphical user interface for activity building and editing
- Displays activity timeline and resources
- Has two halves – a core (project independent) and adaptation (project dependent)
Seqgen Core and Adaptation

Project Specific (Adaptation):
- Activities
- Flight Rules
- Mission Rules
- Commands
- Sequences
- Project Spacecraft Models

Project Independent (Core):
- XML Readers/Writers
- Flight Rule Checking Infrastructure
- Activity Infrastructure & Expansion
- Sequence Infrastructure & Expansion
- Command Handling & Checking
- Resource Checking Infrastructure
- Constraint Propagation Infrastructure
Prior to Mars Exploration Rovers (MER) and Virtual Machine Language* (VML) Missions

- Deterministic activity execution
- Commands are instantaneous
- All sequences and blocks expanded to a list of absolute time tagged commands
- Expansion from activities to commands completes before any command modeling begins

Note: VML is a on-board command and activity sequencing engine
Seqgen Past Modeling Behavior

Time Tagged Spacecraft Activities and Commands
1..n
- Command
- Activity
- Loop
- Command
- End loop
- Activity
- Command
- Command

Activity Blueprint
0..n
- Command
- If
- Command
- Else
- Command
- End if

Expansions
Modeler
Spacecraft Events

Uplink

DSN View Periods

DSN Allocations

May 12, 2008
Space Ops 2008 Heidelberg, Germany
Challenges in Modeling Current Flight Behavior

- Commands take time
  - Passage of command execution time needed to be modeled
  - Command completion time needed to be modeled
- Newer spacecraft permit multiple processes at one time
- Activity and command execution by the Spacecraft Sequencing Engine is multi-threaded
  - Note: Originally Seqgen was single threaded
- Newer spacecraft create event driven activities and commands
- Newer spacecraft react to events that happen on the spacecraft
Current Seqgen Capabilities

- Ground expanded activities are expanded into a list of steps containing commands, spawns, calls, waits etc.
- Seqgen supports multi-threaded event driven sequencing
- Commands can take time (sequence engine waits for command to finish)
- Commands can Spawn/Call activities
- Commands and activities can wait for conditions
- Activities can be suspended, resumed or killed
- Modeling information from commands can be fed back to the parent activities
- Seqgen can generate predicts without a sequence of commands
- Seqgen supports both deterministic and event driven activities and commands in a single deployment
  - Seqgen can be run using the time-ordered capability
  - Seqgen can be run using the event driven capability
  - Seqgen can be run using both time-ordered and event driven capability
Seqgen Current Modeling Behavior

- Ground Expanded Activities
- Ground Expanded Blocks
- DSN View Periods
- DSN Allocations

Expansion

Modeler

Spacecraft Events

On-board Expanded Sequence
- call
- while
- command
- wait
- End while
- delay
- command
- spawn

On-board Expanded Activity
- wait
- If
- spawn
- else
- command
- End if

Uplink
Current Seqgen Command Execution

Engine 1
- command
- spawn
- command
- call
- command
- delay
- Command a

Engine 6
- command
- delay
- command
- spawn
- Command c

Command c
- delay
- power_on = false
- Wait GV::power_on==true

Engine 3
- command
- delay
- Command d
- delay
- GV::power_on = true
- Block_spawn

Engine 8
- Command a
- delay
- GV::power_on = false
- Block_spawn
- wait

Command x
- command
- pause
- Command x
- abort
- Resume 2
- GV::power_on = true
- delay

Space Ops 2008 Heidelberg, Germany
May 12, 2008
Capabilities Support Earlier Mission Phases

• Seqgen models parallel, asynchronous events with conditions that vary over time, for example
  • Available station tracking
  • Shadow periods with seasonal variations
• Activities are scheduled on the fly when conditions are meet
• Seqgen can be used to support trade studies, for example
  • To determine battery size,
  • To determine data capacity
  • To determine station tracking required for maximum data return
• Seqgen allows migration from low to high fidelity models as conditions and project decisions are refined
  • Seqgen models from high-level activities to detailed commands
Mission Formulation Example

• **Requirements**
  - Schedule instrument activity when
    • the instrument is not in shadow,
    • the energy \( \geq \text{min}_\text{energy} \)
    • data below threshold
  - Downlink data when
    • not performing science
    • tracking station available
    • energy \( > \text{min}_\text{energy} \)
    • data available
  - Priority – Science over Downlink
  - Battery
    • charge while not in shadow at an input rate
    • Used by telecom when downlinking
    • Used by instrument when taking data

• **Conditions**
  - 4 hrs in shadow followed by 2 hrs of light
  - 3 hrs station tracking with 6 hrs non-tracking
One-Day Adaptation

- Created Activities Running in Parallel
  - Activity to simulate days and nights
  - Activity to simulate station tracking periods
  - Activity to wait for science opportunities
  - Activity to wait for down link opportunities

- Modeled Output:
  - Data downlinked
  - Number of missed science opportunities
    - due to maximum data volume
    - due to Data taken
    - less than minimum energy
    - due to lack of data
  - Number of downlink passes misses due to less than minimum energy
Results of Mission Formulation Example

Activity Summary

- Telecom Activity killed due to science opportunity
- Science activity killed due to max data volume

Tracking

Downlink

Instrument

Shadow

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21
Seqgen’s Future

- Seqgen has been recently ported to Linux
- Seqgen has been upgraded to use standard C++ libraries
- Seqgen has been refactored to be a modeling engine
  - Eclipse plug-in components for user input, file input and communications with other tools have been added
  - Editing features are being performed by another tool, the MPS Editor, written in Java and built on the Eclipse Rich Client Platform
  - Timeline viewing capabilities will be started later this year
- Seqgen has the ability to operate as a web service plug-in
- Peripherals for viewing spacecraft events are planned
- Adaptation debugger is also planned
Conclusion

Seqgen:
• Verifies and Validates Sequences of Commands
• Models spacecraft events
• Can be deterministic and/or event driven
• Allows adaptation to mission specific capabilities
• Determines resource consumption
• Adapts to Mission Formulation as well as Operations Mission Phases
• Permits use by simple or complex missions
• Adapts to different mission types (i.e., orbiter, flyby, in-situ)

SEQGEN IS A SOFTWARE APPLICATION FOR ALL SEASONS