Jet Propulsion Laboratory
California Institute of Technology

Ranger: 3D Space Mission Visualization in a Web Browser

Marc Pomerantz

Technical Group Supervisor
Information and Data Management Group (393G)
Mission Control Systems Section
Jet Propulsion Laboratory,
California Institute of Technology,
Pasadena, CA, 91109

Contact: **Marc.I.Pomerantz@jpl.nasa.gov**

Outline

- Engineering Goals
- Past Visualization Projects
- Customer Domains
- 3D visualization in prehistoric and modern times
- *Ranger* Overview
- *Ranger* System Architecture
- Who is Using Ranger?
- *Flow* Server Architecture
- Summary

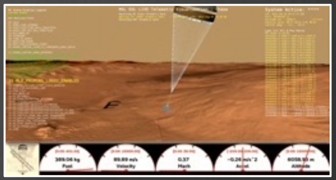
Engineering Goals

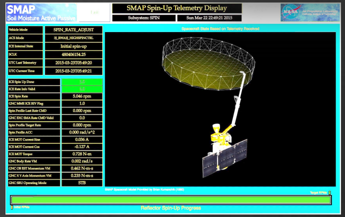
Build an easily re-useable and data driven visualization system to help mission engineers understand the state of their system and that is *engineering accurate* and can be reconfigured to support a variety of space and robotic mission scenarios

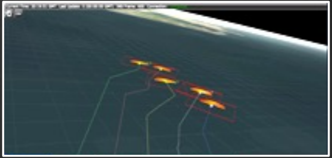
JPL
Jet Propulsion Laboratory
California Institute of Technology

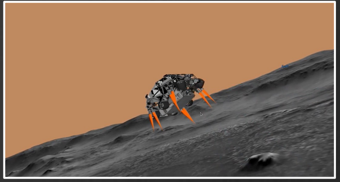
Past Visualization Projects

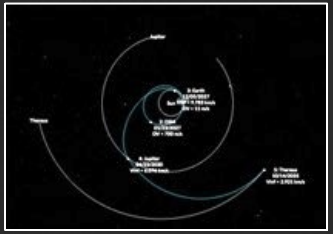
- We have supported a variety of NASA/JPL flight and engineering projects by providing real-time telemetry visualization and situational awareness displays, vehicle commanding, modeling and simulation
  - Hypercube missile defense simulation/visualization
  - Autonomous Feature and Star Tracking simulation/visualization
  - Synthetic Aperture Radar instrument coverage
  - **Galileo** telemetry display of Huygens probe health and status
  - Support of various robotic vehicle operations, simulation and proposal tasks
  - **MSL** landing night telemetry visualization
  - **LDSD** Flight Director display (health and status of vehicle). Used during I&T and operations during two test flights.
    - Hundreds of channels displayed and splashdown predict model in the loop
  - **M2020 Sampling Arm** telemetry display and robot commanding software used during system hardware and software development
  - **SMAP** boom deployment and antenna spin-up telemetry displays for operations
  - Multi-agent autonomy tasks. 3D situational awareness displays.
  - **M2020 EDL** testbed telemetry visualization (like our MSL viz). In development
  - Mission visualization for JPL formulation. In development







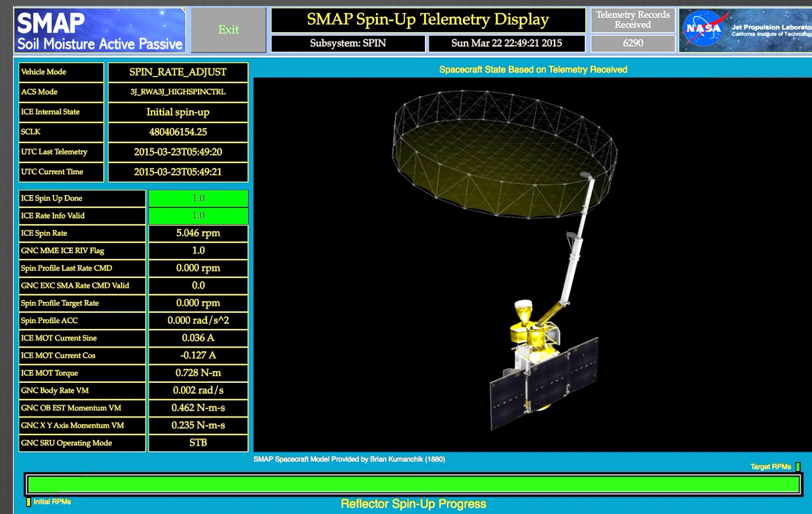




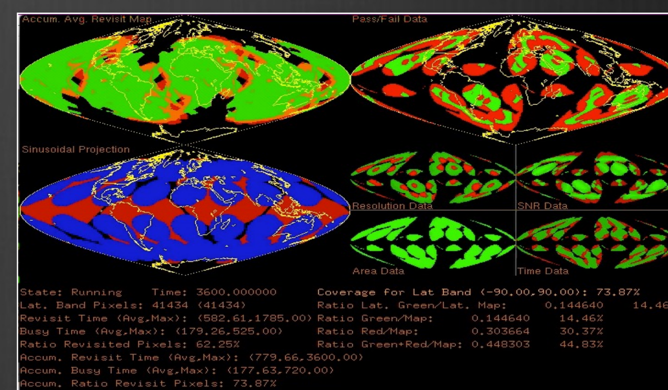
Jet Propulsion Laboratory
California Institute of Technology

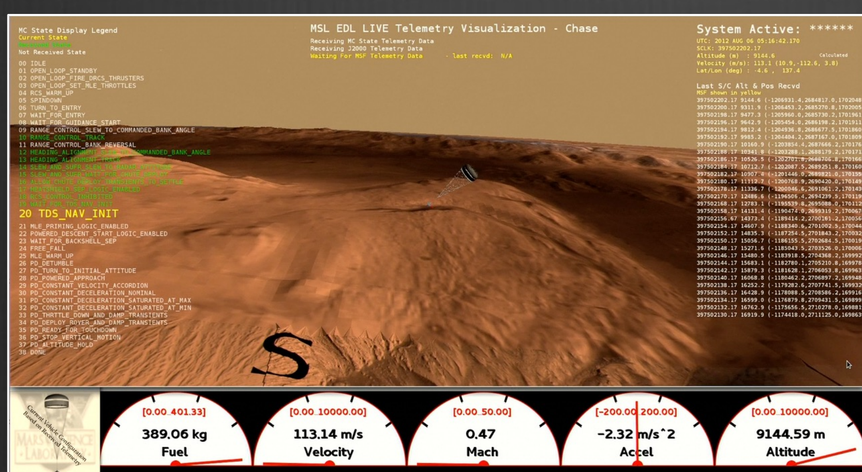
Prehistoric Times (pre 2015)

Interactive 3D Visualization software required high-end workstations and expensive graphics cards



SMAP Antenna Deployment





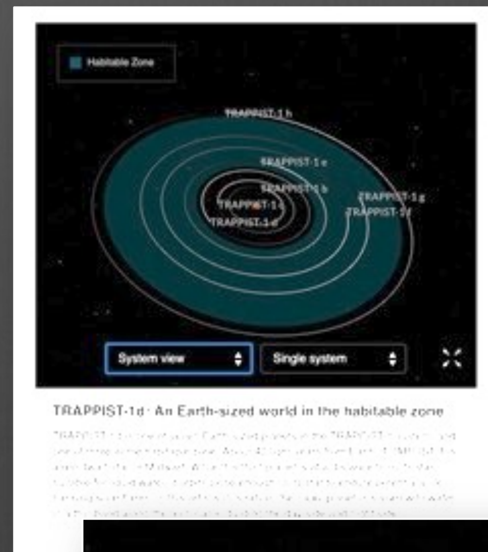
MSL Landing Night

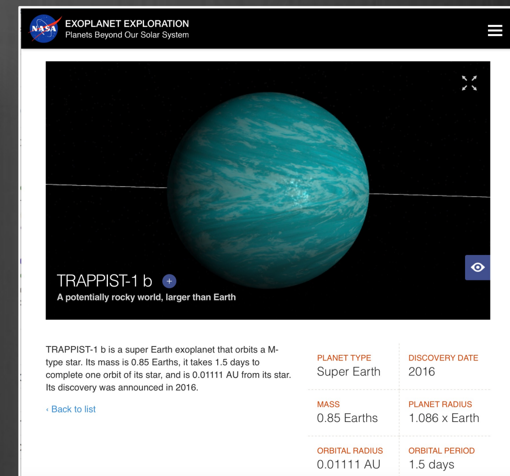
Synthetic Aperture Radar coverage

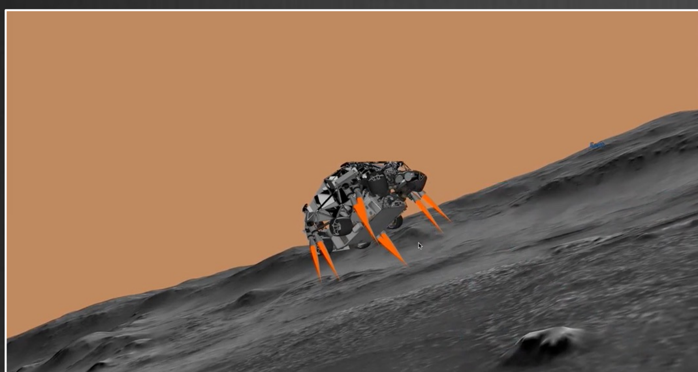
Jet Propulsion Laboratory
California Institute of Technology

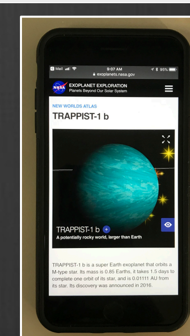
Modern Times (post 2015)

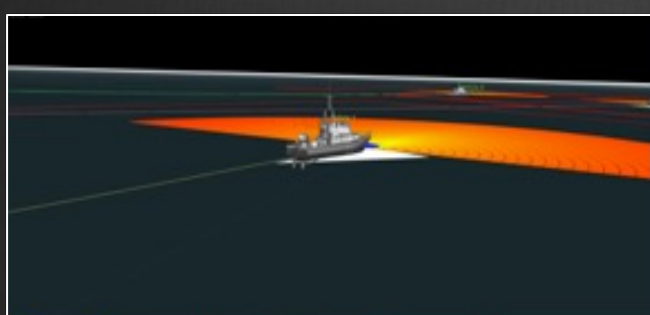
Interactive 3D Visualization software can run inside a web browser and on typical laptop and mobile devices







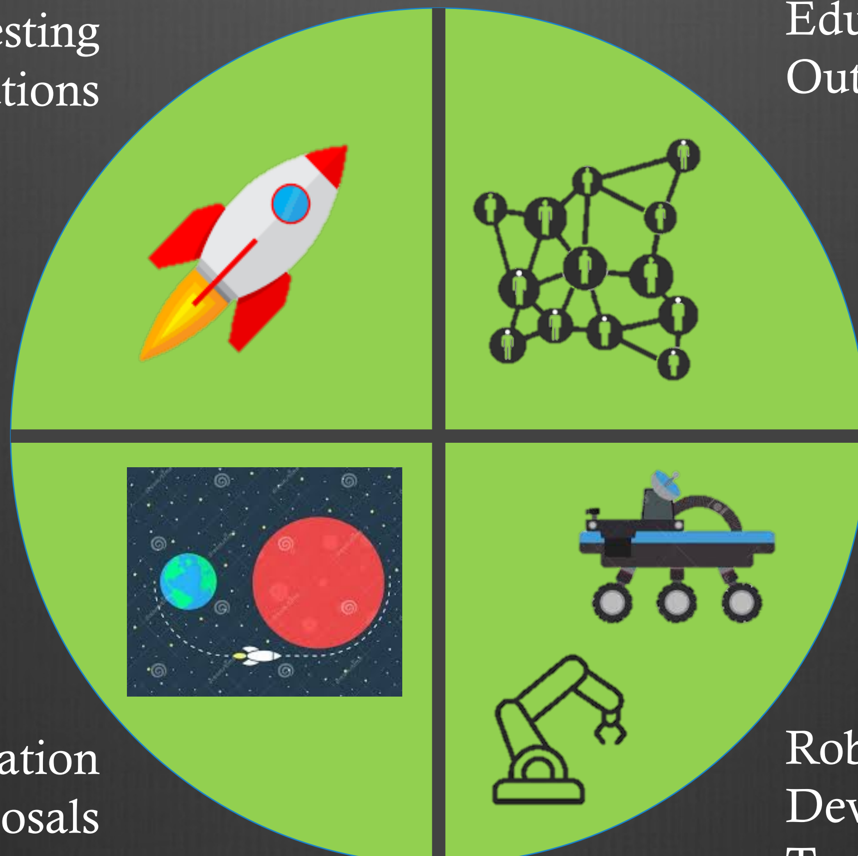




Customer Domains

Flight Mission Testing and Operations

Education and Public Outreach



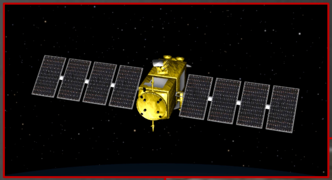
Mission Formulation and Proposals

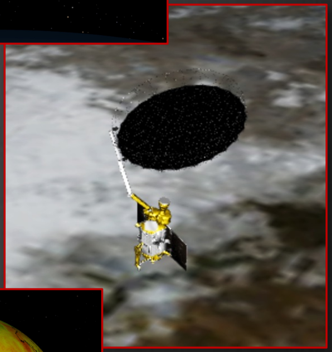
Robotic System Development and Testing

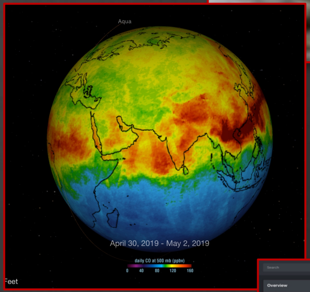
Jet Propulsion Laboratory
California Institute of Technology

Ranger Overview
Engineering Accurate Visualization

- Ranger is designed to ***accurately and correctly*** visualize a 3D representation of spacecraft and robotic vehicle missions
  - Use of supporting mission data and documentation for applications (CAD, GNC Ref. Frames, environment info, imagery, etc.)
  - All calculations double precision on the CPU
  - Frame centric system (no world coordinate system). Preserves numerical accuracy
  - Automated build and regression testing for Ranger code using Jenkins
- Visualizations runs locally on device graphics hardware in a web browser and on desktop and mobile devices
  - GPU acceleration and programming via shaders
- Can be driven from multiple data sources
  - Physics-based models, hardware/software testbeds, mission design tools, telemetry streams, ephemeris data
  - Science data or instrument coverage overlays (static or animated)
- Maintain data security by running server and browser on a secure network
  - Authorization at the web server level
  - Authentication at the network level
- ***Flow*** is Ranger's backend server that processes data to be shipped to clients
- Immersive VR support (HTC Vive)
- UI support using ReactJS or D3 for integrated dashboard widgets, controls, analytic visuals
- Over 90 Wiki pages of documentation on Github.jpl.nasa.gov, including examples and tutorials

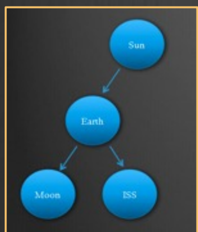






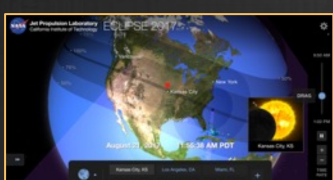
Why is Ranger Multi-Mission?

Mission Specific
Input Data



Scene Description
(json)

Visual Assets
(CAD, imagery)



User Interface
Assets
(HTML, CSS)

Data drives Ranger

The same application code can provide a completely different user experience

by changing visual assets, vehicle data, ephemerides or user interface

Ranger Core

C++/OpenGL compiled using Emscripten, to ASM.js and WebGL (faster execution speed then native JavaScript)

Ranger Core changes infrequently and is shared across applications. Application specific layer can have reusable elements but typically changes from app to app

Multi-mission
Application Specific
User Interface

JPL
Jet Propulsion Laboratory
California Institute of Technology

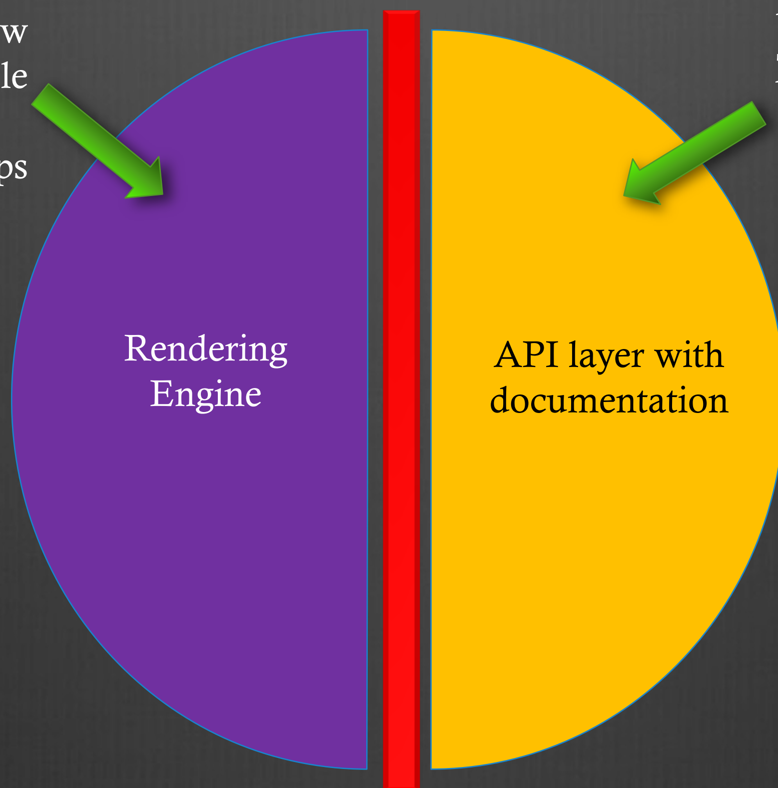
Ranger Engine Design

Abstraction Layer
(reusable software interface)

Rendering engine can be replaced as new technologies are available

No (or few) changes to user apps

Application developers use JavaScript in the API layer to build their apps



C++ and OpenGL compiles to asm.js (or WebAssembly) and WebGL 1 or WebGL 2

ASM.js is more performant than JavaScript. ~10-50% of native C++

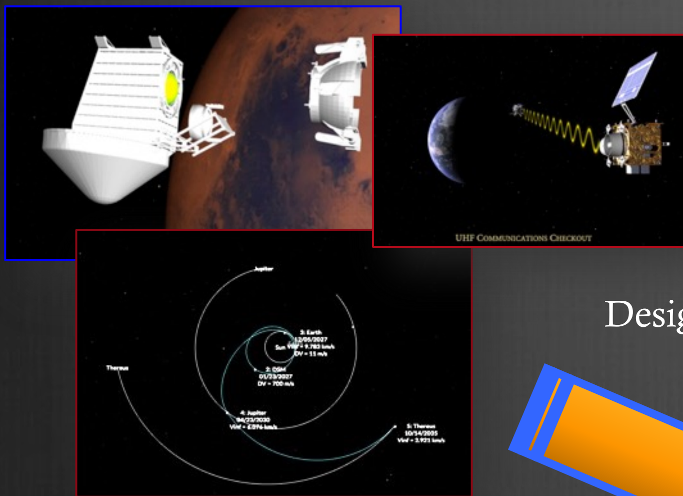
Supported in all major browsers

JavaScript & JSON

Ranger has been designed with an abstraction layer between the JavaScript Application Programmers Interface (API) and the rendering engine that makes it possible to replace the renderer as new technologies arise, with minimal changes to the API layer or documentation

Jet Propulsion Laboratory
California Institute of Technology

Ranger Support for Mission Lifecycle



Design and Fabrication

Integration and Test

Launch

Operations

Multi-dimensional view into the flight system's health, status and localization. Predict models in the loop

Formulation

Help develop innovative, cost-effective, mission concepts that satisfy mission requirements and communicate designs to engineers and stakeholders

("In The Loop" visualization during trade studies)

Future Use:

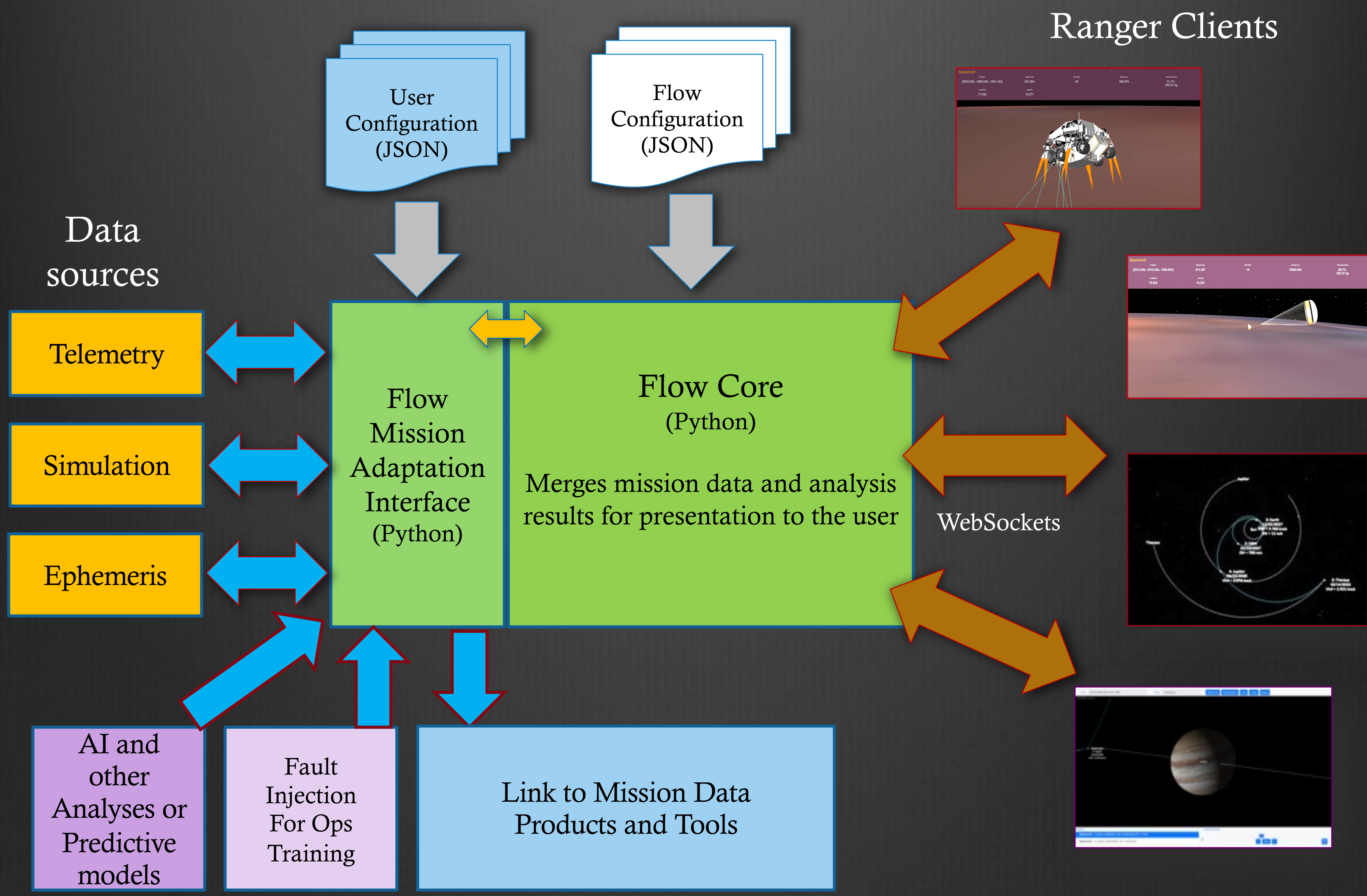
Virtual Operations for Training

Train new ops personnel without requiring access to actual spacecraft or vehicles

Jet Propulsion Laboratory
California Institute of Technology

Typical Ranger System Architecture

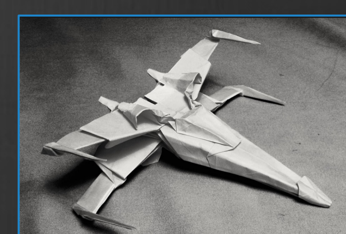
Multi-mission interface for Mission Information Analysis, Introspection and Control

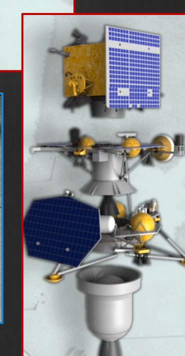


Jet Propulsion Laboratory
California Institute of Technology

Who is Using the Ranger System?

- Recent and currently supported projects:
  - Engineering Projects
    - 2017-2019 JPL Formulation Application (SCOPE) (JPL Section 312)
    - 2017-2019 JPL Swarms of autonomous boats (JPL Section 347)
    - 2019 M2020 EDL visualization
  - JPL Education and Public Outreach
    - FY16 Exoplanet - https://exoplanets.nasa.gov/alien-worlds/strange-new-worlds/
    - FY17 Eclipse - https://eyes.jpl.nasa.gov/eyes-on-eclipse-web-app.html
    - FY18 Eyes on Exoplanets - https://exoplanets.nasa.gov/eyes-on-exoplanets/
    - FY19 Earth-Now - https://climate.nasa.gov/earth-now/
  - Not only for visualizations of physical systems and situational awareness
    - Visualize cyber-physical/virtual systems
    - Exploded view
    - Unfold the origami spacecraft to display internals
    - Display alarm states in context of the flight system
    - Flight rule violations
  - Future?
    - Europa Clipper avionics displays
    - Psyche
    - Machine Learning integrated w/visualization for mission ops





Jet Propulsion Laboratory
California Institute of Technology

Flow

Ranger's Telemetry-to-Visualization Interface

Jet Propulsion Laboratory
California Institute of Technology

Flow System Goals

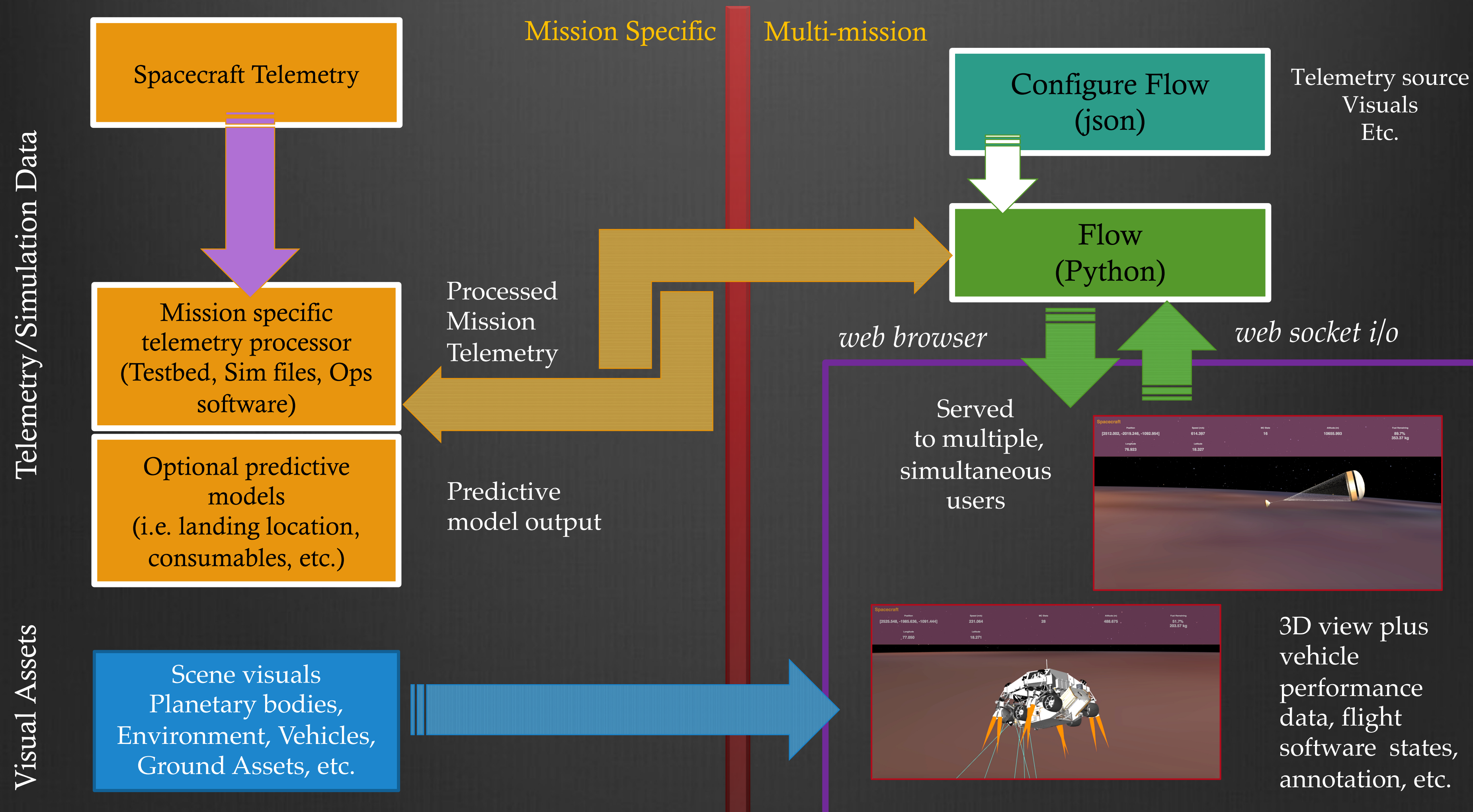
Provide an easy to configure, multi-mission,
data interface to drive Ranger visualizations

- Support multiple simultaneous visualization application clients
- Support data coming from multiple, simultaneous data sources
- Support various data types (telemetry, simulation, ephemeris, alarm states, etc.)
- Support mission-provided or third-party analysis (predictive models, AI, etc.)
- Shorten development time for data source developers by providing an easy to use Python API
- Exception handling, logging, connect/re-connect for clients
- Connect to existing, non-Python data sources (Matlab, Simulation, Ops tools, telemetry, etc.) using typical Unix inter-process communication (sockets, pipes, shared memory, pub/sub messaging, etc.)

Telemetry Visualization System Architecture

NASA Mars 2020 Mission Example using Ranger and Flow

To drive a real-time mission visualization, telemetry data is processed as needed and streamed into Ranger via a web server on a secure network. The Multi-Mission components of Ranger are reused without changes, from mission to mission



Summary

- Ranger applications have been deployed for multiple customers since 2016
  - Visualizations can be driven by a variety of data sources (telemetry, ephemeris, simulation)
  - Applications can be embedded in existing web pages
- Mission configurations use the Ranger core software and require:
  - Mission specific CAD and environment models
  - Mappings between telemetry items and changes to Ranger app visuals
  - Mission specific UI/UX as needed. UI elements, graphs/plots, ReactJS/D3
- Not only for visualizations of physical systems and situational awareness
  - Visualize cyber-physical/virtual systems.
    - Unfold the origami spacecraft to display internals and data
      - Display telemetry alarm states in context of the flight system or subsystem
      - Flight rule violations
  - Train operations staff by using playback data with injected anomalies
- Predictive models in the loop with mission telemetry to compare predicts vs. actuals

Backups

Jet Propulsion Laboratory
California Institute of Technology

Ranger System Architecture

- C++11 / Javascript / HTML5
  - CMake and Emscripten build C++ into a JavaScript module
  - Webpack assembled Javascript assets into a bundled web app
- C++ to Javascript and ASM.js (or WebAssembly) via Emscripten
  - ASM.js performance from C++ code is 10-50% of native code performance
  - WebAssembly advertised performance is near native code performance
    - Now supported in all major browsers
    - low-level assembly-like language with a JavaScript API
- JSON Scene description
  - Describing the scene as data allows many scenes to be built with zero JavaScript knowledge
- JSON Scene management API
  - Allow interacting with the engine from anywhere using a web socket
- Continuous Integration use Jenkins
- Art Pipeline – Blender to Ranger
- Regression Test suite

Ranger System Architecture
More Details

Jet Propulsion Laboratory
California Institute of Technology

- C++ responsible for core renderer and heavy computation
  - OpenGL ES 2.0 Renderer
  - GLSL shaders. i.e. Lighting or atmosphere effects
  - Ephemerides calculation
  - Computational utilities
  - *Note:* Supports future linking with C/C++-based modules such as NAIF Spice Library, predictive models or science data processing into Ranger
- JavaScript library supports the scene API
  - Main event loop
  - User interaction
  - Camera processing
  - Scene graph evaluation
- HTML5 provides user interaction layer

C++ to Javascript via Emscripten

- Emscripten is a Javascript backend for the LLVM compiler
- C, fortran, C++, ruby, rust, and many more languages are available as LLVM front ends
- Emscripten can compile to ASM.js or WebAssembly
- ASM.js performance from C++ code is 10-50% of native code performance
- Performance is substantially better than pure JavaScript for heavy computations