Use of the NetBeans Platform for NASA Robotic Conjunction Assessment Risk Analysis

Nick Sabey
a.i. solutions, Inc.
JavaOne 2014

Note: These slides will be moved into the JavaOne template prior to presentation
NASA Robotic Conjunction Assessment Risk Analysis (CARA)

- CARA is the process and team for:
  - Determining the **risk of collision** between two orbiting objects
  - Assisting with **risk mitigation** (typically, via an orbital maneuver)
- The NASA Robotic CARA group at NASA GSFC provides this support to **all operational** NASA robotic (unmanned) missions
  - Started in January 2005
  - Over **65 missions** in total
  - Over 1,000 close approach messages received per day
  - Maneuver recommendations result in as many as 30 realized maneuvers each year
NASA Robotic Conjunction Assessment Risk Analysis (CARA)
NASA Robotic Conjunction Assessment Risk Analysis (CARA)

- The Joint Space Operations Center (JSpOC) is a USAF operational unit that is responsible for maintaining the locations of all objects in space
  - For NASA, the JSpOC identifies close approaches between those objects and provides data to CARA to enable the collision risk assessment
  - The JSpOC operates in a secure environment
    - i.e. only accredited or in-house software; accreditation can take more than 6 months
- CARA has an analyst/developer resident at the JSpOC to access data which cannot be sent to NASA
  - Able to develop NASA-specific products and services
    - i.e. development of space weather risk trade space, orbit quality data product
  - Symbiotic relationship between NASA and USAF/JSpOC
    - i.e. developed automation and improvements to legacy software components
NetBeans for CARA JSpOC Support

• JSpOC development & support provided by a single developer
  – Scope of project would not have been possible without the use of a Rich Client Platform (RCP)
  – NetBeans was intuitive, and provided excellent support
• Initially developed using NetBeans 7.3.1
  – Provided significant UI features with little to no code
  – Scalability from plugin architecture
  – Platform API reduced coding time substantially
    • Nodes, Module, Lookup, and FileSystems APIs used extensively
  – Reused existing platform modules developed for other a.i. solutions projects
  – Interfacing with legacy and supporting software very easy
Upgrading to Java 8 / JavaFX 8

• No software changes required to update to JDK8
  – i.e. nothing broke
• We are adding some of the new JDK 8 features to our application as part of the upgrade:
  – Stream API and Lambdas
    • Reduction in code / simplified syntax
    • Simplifies our use of collections
    • In the process of adding substantially more parallelism
  – JavaFX 8
    • No prior access to JavaFX on our system
    • JavaFX has been approved along with JDK 8
    • Experimenting with both 3D and 2D visualizations
JFX 8 OD Quality Visualization Demo*

*Note: This will be an interactive demo, with all data being simulated