BumperBot Simulation Testing Guide

Hands-On Tutorial: Test the Three AI Agent Packages

Author: RoboShire Team
Version: 1.0
Date: October 23, 2025
Estimated Time: 2-3 hours

RoboShire GUI Testing Guide

Test the Three AI Agent Packages Using Only the GUI

Version: 1.0 Date: October 24, 2025 Estimated Time: 1-2 hours Difficulty: Beginner-Friendly (GUI Only - No Terminal!)

Table of Contents

1. Introduction

What You'll Do

This guide shows you how to test all three AI agent packages (Sarah's, Marcus's, and Elena's) using ONLY the RoboShire GUI - no terminal commands needed!

The Three Packages

Package Locations: - workspace/src/sarah_bumperbot_pkg/ - Sarah's beginner-friendly package - workspace/src/marcus_bumperbot_pkg/ - Marcus's optimized package - workspace/src/elena_bumperbot_pkg/ - Elena's enterprise package

What Makes Each Package Special

Package	Skill Level	Key Features	Node Count
Sarah's	Beginner	Simple, educational code	8 nodes
Marcus's	Intermediate	Optimized, multi-launch	7 nodes
Elena's	Expert	Lifecycle, safety, Docker	4 nodes

2. Launch RoboShire

Step 1: Start the Application

Windows: 1. Navigate to: d:\ROS2_PROJECT\ 2. Double-click: launch_roboshire.bat - OR activate virtual environment and run: python -m roboshire

What You'll See:

```
RoboShire v2.2.0 - Visual IDE for ROS2
Loading...
```

Step 2: Main Window Overview

Once RoboShire opens, you'll see:

Top Menu Bar: - File | Robot | Code | Build | Run | Tools | View | Help

Left Panel (Tabs): - Robot Design - Node Graph - System Architecture - Project Files

Right Panel (Tabs): - Properties - Build & Deploy - Logs & Monitor - Visualization

Bottom Status Bar: - Project status - Build status - Running nodes count

Step 3: Verify RoboShire is Ready

Check these indicators: - [ ] Window title says "RoboShire - Visual IDE for ROS2" - [ ] No error dialogs appear - [ ] All tabs are visible - [ ] Status bar shows "Ready"

3. Open/Import Agent Packages

The three AI agent packages already exist in your workspace. You need to tell RoboShire about them.

Method 1: Open Existing Project (Recommended)

Step 1: Open Project 1. Click File Open Project... - Keyboard shortcut: Ctrl+O 2. Navigate to: d:\ROS2_PROJECT\workspace\src\sarah_bumperbot_pkg\ 3. Look for a project file (.roboshire or package.xml) 4. Click Open

What Happens: - RoboShire loads the package - Project Files tree populates - Package name appears in status bar

Repeat for Marcus and Elena: - You can have multiple projects open by using File New Project Window - OR test them one at a time

Method 2: Import from Workspace

Step 1: Workspace Manager 1. Click Tools Multi-Package Manager 2. A dialog opens showing all workspace packages

Step 2: Select Packages 3. Check the boxes for: - [ ] sarah_bumperbot_pkg - [ ] marcus_bumperbot_pkg - [ ] elena_bumperbot_pkg 4. Click Load Selected Packages

What You'll See: - Package list in System Architecture tab - Build order calculated - Dependencies shown

4. Build the Packages

Now let's build each package using the GUI!

Sarah's Package: Build Process

Step 1: Select Package to Build 1. Click Build Build (or press Ctrl+B) 2. A dialog appears: "Select Package to Build" 3. Select: sarah_bumperbot_pkg 4. Click OK

Step 2: Watch Build Progress - RoboShire automatically switches to Build & Deploy tab - Build Output panel shows: ``` Starting >>> sarah_bumperbot_pkg [Processing: sarah_bumperbot_pkg] Finished <<< sarah_bumperbot_pkg [10.5s]

Summary: 1 package finished [10.6s] ```

Success Indicators: - [ ] Green checkmark appears - [ ] Status bar says "Build successful!" - [ ] No red error messages - [ ] Build time shown (~10-30 seconds)

If Build Fails: - Red X appears - Error messages in Build Output - See Troubleshooting section

Marcus's Package: Build Process

Repeat the same steps: 1. Build Build (Ctrl+B) 2. Select: marcus_bumperbot_pkg 3. Watch build complete (~8-15 seconds)

Expected: Slightly faster than Sarah's (fewer nodes)

Elena's Package: Build Process

Repeat again: 1. Build Build (Ctrl+B) 2. Select: elena_bumperbot_pkg 3. Watch build complete (~15-20 seconds)

Expected: Longer build (more features, tests)

Build All Packages at Once

Alternative Method: 1. Build Build (Ctrl+B) 2. Select: All Packages 3. Wait for all 3 to build sequentially

Build Order: RoboShire automatically determines dependency order:

```
1. sarah_bumperbot_pkg
2. marcus_bumperbot_pkg
3. elena_bumperbot_pkg
```

Verify Builds

Check System Architecture Tab: 1. Click System Architecture tab (left panel) 2. You should see: - sarah_bumperbot_pkg (Built) - marcus_bumperbot_pkg (Built) - elena_bumperbot_pkg (Built)

Check Build & Deploy Tab: 1. Click Build & Deploy tab (right panel) 2. Build Output shows summary: === Build Summary === Total packages: 3 Succeeded: 3 Failed: 0

5. Run and Test Nodes

Now let's launch nodes from each package!

Sarah's Package: Launch Nodes

Step 1: Launch Main System 1. Click Run Run (or press Ctrl+R) 2. Dialog: "Select Package" - Choose: sarah_bumperbot_pkg - Click OK 3. Dialog: "Select Executable/Node" - Choose: velocity_controller (or bumperbot_bringup if available) - Click OK

Step 2: Watch Node Start - Switches to Logs & Monitor tab - Log Viewer shows: [INFO] Starting velocity_controller node [INFO] Publishing to /cmd_vel at 10 Hz [INFO] Node running successfully

Step 3: Monitor Node Status - Node Status Widget shows: - velocity_controller (Running) - State: ACTIVE - Uptime: 00:00:15

Step 4: Launch More Nodes Repeat to launch: - odometry_calculator - imu_reader - ekf_localization (sensor fusion)

What You're Testing: - All 8 nodes start without errors - Nodes communicate (check Topics tab) - Velocity commands appear in logs

Marcus's Package: Launch Nodes

Launch Marcus's System: 1. Run Run (Ctrl+R) 2. Select: marcus_bumperbot_pkg 3. Select: velocity_controller (optimized version)

What's Different: - Unified TF broadcaster (1 node instead of 3) - System monitor shows performance metrics - Lower CPU usage visible in logs

Check Performance: 1. Click Performance Profiler tab 2. View metrics: - CPU usage: ~15% (vs Sarah's ~20%) - Memory: ~100 MB - Update rate: 50 Hz

Elena's Package: Lifecycle Nodes

Elena's package uses Lifecycle nodes - they have states!

Step 1: Launch with Lifecycle Manager 1. Run Run (Ctrl+R) 2. Select: elena_bumperbot_pkg 3. Select: velocity_controller (lifecycle version)

Step 2: Check Lifecycle State - Switch to Lifecycle Manager tab (Build & Deploy panel) - You'll see: Node: velocity_controller State: UNCONFIGURED

Step 3: Activate the Node 1. Select node in Lifecycle Manager 2. Click Configure button - State changes: UNCONFIGURED INACTIVE 3. Click Activate button - State changes: INACTIVE ACTIVE 4. Node is now running!

Lifecycle States: - UNCONFIGURED (gray) - Not set up - INACTIVE (yellow) - Configured but not publishing - ACTIVE (green) - Fully running - FINALIZED (red) - Shut down

Test Lifecycle Transitions: 1. Click Deactivate - stops publishing but keeps node alive 2. Click Activate - resumes publishing 3. Click Shutdown - fully stops node

6. Monitor and Visualize

Monitor Topics (All Packages)

Step 1: Open Topic Inspector 1. Click Tools Topic Inspector - OR click Topic Inspector tab 2. You'll see live topics: /cmd_vel (geometry_msgs/Twist) /odom (nav_msgs/Odometry) /odom/filtered (nav_msgs/Odometry) /imu/data (sensor_msgs/Imu) /joint_states (sensor_msgs/JointState)

Step 2: Inspect a Topic 1. Double-click /cmd_vel 2. A window shows live data: yaml linear: x: 0.2 # Robot moving forward y: 0.0 z: 0.0 angular: x: 0.0 y: 0.0 z: 0.0 # Not turning

Step 3: Compare Raw vs Filtered Odometry 1. Open Topic Inspector for /odom (raw) 2. Open Topic Inspector for /odom/filtered (EKF) 3. Compare position values: - Raw: More noise (jumpy values) - Filtered: Smoother (less variation)

Visualize TF Tree

Step 1: Open TF Tree Visualizer 1. Click Tools TF Tree Visualizer - OR click Visualization tab TF Tree

Step 2: View Transform Hierarchy You'll see a tree diagram:

```
odom
  base_link
       left_wheel
       right_wheel
       imu_link
```

Compare TF Broadcasters: - Sarah's: 3 separate TF broadcaster nodes - Marcus's: 1 unified TF broadcaster node (more efficient!) - Elena's: 1 unified broadcaster with lifecycle management

Monitor System Performance

Step 1: Open Performance Profiler 1. Click Performance Profiler tab 2. You'll see real-time metrics:

For Sarah's Package: | Node | CPU % | Memory | Frequency | |------|-------|--------|-----------| | velocity_controller | 0.8% | 8 MB | 10 Hz | | odometry_calculator | 2.1% | 12 MB | 50 Hz | | imu_reader | 1.5% | 10 MB | 100 Hz | | TOTAL | ~20% | ~125 MB | - |

For Marcus's Package: | Node | CPU % | Memory | Frequency | |------|-------|--------|-----------| | velocity_controller | 0.5% | 8 MB | 10 Hz | | odometry_calculator | 2.0% | 12 MB | 50 Hz | | tf_broadcaster (unified) | 0.8% | 8 MB | 100 Hz | | TOTAL | ~15% | ~100 MB | - |

Marcus's Advantage: 25% less CPU, 20% less memory!

View Logs

Step 1: Open Log Viewer 1. Click Logs & Monitor tab 2. Select Log Viewer sub-tab

Step 2: Filter by Node 1. Dropdown menu: Select velocity_controller 2. See only logs from that node

Step 3: Filter by Level - Show only: INFO, WARN, ERROR, or FATAL - Hide DEBUG messages for cleaner view

Look For: - INFO: Normal operation - WARN: Safety limits, timeouts (Elena's package) - ERROR: Node crashes, missing dependencies

7. Compare Results

Feature Comparison Table

Fill this out as you test each package:

Feature	Sarah	Marcus	Elena
Build Time	___s	___s	___s
Build Success	/	/	/
Nodes Launched	___/8	___/7	___/4
CPU Usage	___%	___%	___%
Memory Usage	___MB	___MB	___MB
Lifecycle Support	No	No	Yes
Safety Features	Basic	Moderate	Advanced
Monitoring	None	Yes	Yes
Your Rating	___/10	___/10	___/10

Performance Comparison

CPU Usage (Lower is Better):

```
Sarah:   ~20%
Marcus:  ~15%
Elena:   ~14%
```

Memory Usage (Lower is Better):

```
Sarah:   ~125 MB
Marcus:  ~100 MB
Elena:   ~110 MB
```

Feature Count (More is Better):

```
Sarah:   8 features
Marcus:  12 features
Elena:   20+ features
```

Which Package Would You Choose?

For Learning ROS2: - [ ] Sarah's (simplest, most educational)

For Prototyping: - [ ] Marcus's (optimized, flexible)

For Production: - [ ] Elena's (enterprise-grade, safety-critical)

Your Preference:_____

Reason:________

8. Troubleshooting

Build Errors

Error: "No packages found in workspace"

Solution in GUI: 1. Tools Multi-Package Manager 2. Click Scan Workspace 3. Select the three packages 4. Click Load Selected

Error: "Build failed - package not found"

Solution in GUI: 1. Check System Architecture tab 2. Verify package appears in tree 3. Right-click package Refresh Dependencies 4. Try building again

Error: "robot_localization not found"

Solution in GUI: 1. Tools Package Manager 2. Search for: robot_localization 3. Click Install 4. Wait for installation 5. Rebuild package

Runtime Errors

Error: "Node failed to start"

Check in GUI: 1. Logs & Monitor tab Log Viewer 2. Look for ERROR messages 3. Check Node Status widget - is node gray (stopped)?

Common Causes: - Node already running (shows in Node Status) - Missing dependencies (check logs) - Wrong executable name

Solution: 1. Run Stop All Nodes 2. Wait 5 seconds 3. Try launching again

Error: "No topics visible"

Check in GUI: 1. Topic Inspector tab 2. Click Refresh button 3. Verify nodes are running in Node Status

If still no topics: 1. Stop all nodes 2. Rebuild package 3. Launch nodes again

Lifecycle Errors (Elena's Package Only)

Error: "Node in UNCONFIGURED state"

Solution in GUI: 1. Open Lifecycle Manager tab 2. Select the node 3. Click Configure button 4. Then click Activate button

State Transition Order:

```
UNCONFIGURED  [Configure]  INACTIVE  [Activate]  ACTIVE
```

Error: "Transition failed"

Check in GUI: 1. Log Viewer - look for ERROR during transition 2. Common causes: - Missing configuration file - Invalid parameters - Hardware not ready

Solution: 1. Click Cleanup button (returns to UNCONFIGURED) 2. Check configuration files in Project Files tab 3. Try Configure Activate sequence again

Performance Issues

High CPU Usage

Check in GUI: 1. Performance Profiler tab 2. Identify which node is using most CPU 3. Check node logs for infinite loops

Solution: - Restart high-CPU node - Reduce update rate in node properties

GUI Freezing

Possible Causes: - Too many nodes running - Build in progress - Large log files

Solution in GUI: 1. Run Stop All Nodes 2. View Clear All Logs 3. Close unused tabs 4. Restart RoboShire if needed

9. Testing Checklist

Pre-Flight Check

Before starting, verify: - [ ] RoboShire launches without errors - [ ] All three packages visible in workspace - [ ] Build menu is accessible - [ ] Run menu is accessible

Sarah's Package Checklist

- [ ] Package builds successfully
- [ ] Build time recorded: _____ seconds
- [ ] All 8 nodes launched without errors
- [ ] Topics visible in Topic Inspector
- [ ] TF tree shows 4 transforms
- [ ] Odometry data streaming
- [ ] CPU usage recorded: _____ %
- [ ] Memory usage recorded: _____ MB
- [ ] Rating: _____ / 10

Marcus's Package Checklist

- [ ] Package builds successfully
- [ ] Build time recorded: _____ seconds
- [ ] All 7 nodes launched without errors
- [ ] Unified TF broadcaster confirmed (1 node vs 3)
- [ ] System monitor shows diagnostics
- [ ] Performance profiler active
- [ ] CPU usage lower than Sarah's: _____ %
- [ ] Memory usage lower than Sarah's: _____ MB
- [ ] Multi-launch files tested
- [ ] Rating: _____ / 10

Elena's Package Checklist

- [ ] Package builds successfully
- [ ] Build time recorded: _____ seconds
- [ ] All 4 lifecycle nodes launched
- [ ] Lifecycle Manager tab functional
- [ ] Configure transition successful
- [ ] Activate transition successful
- [ ] Deactivate/Activate cycle tested
- [ ] Watchdog monitoring visible in logs
- [ ] Safety limits tested (if applicable)
- [ ] CPU usage recorded: _____ %
- [ ] Memory usage recorded: _____ MB
- [ ] Rating: _____ / 10

10. Summary

What You Accomplished

Using only the RoboShire GUI, you:

1. Opened three complete ROS2 packages
2. Built all packages using the Build menu
3. Launched and monitored ROS2 nodes
4. Inspected topics and data streams
5. Visualized TF transformations
6. Managed lifecycle states (Elena's package)
7. Compared performance metrics
8. Tested three different robotics architectures

No terminal commands needed!

Key Features You Used

Build & Deploy: - Build menu (Ctrl+B) - Package selection dialog - Build output viewer - Multi-package manager

Run & Monitor: - Run menu (Ctrl+R) - Node status widget - Log viewer - Performance profiler

Visualization: - Topic inspector - TF tree visualizer - System architecture diagram

Advanced: - Lifecycle manager (Elena's package) - Multi-package builds - Real-time monitoring

What You Learned

Technical Skills: - Building ROS2 packages via GUI - Launching and stopping nodes - Monitoring system performance - Managing lifecycle states - Inspecting topics and transforms

Robotics Concepts: - Differential drive robots - Sensor fusion with EKF - Transform frames (TF2) - Lifecycle node architecture - System monitoring

Next Steps

Continue Your Learning: 1. Modify node properties using Property Editor 2. Create new nodes using Node Graph Editor 3. Test with actual robot hardware 4. Add new sensors and features 5. Deploy to production

Advanced GUI Features to Explore: - Behavior Tree Editor - Nav2 Integration Wizard - Launch File Editor - Code Editor (view/edit node code) - URDF Validator

11. Certificate of Completion

RoboShire GUI Testing Certificate

I, ___, have successfully tested all three AI agent packages using only the RoboShire graphical interface on ____ (date).

Packages Tested: - [x] Sarah's Beginner Package (Rating: _ /10) - [x] Marcus's Intermediate Package (Rating: /10) - [x] Elena's Enterprise Package (Rating: __ /10)

GUI Skills Acquired: - [x] Building ROS2 packages via Build menu - [x] Launching nodes via Run menu - [x] Monitoring topics with Topic Inspector - [x] Managing lifecycle states - [x] Visualizing TF trees - [x] Performance profiling

Favorite Package:___

Reason:_____

Signature:_____

12. Feedback

What was easy to do in the GUI:

What was difficult or confusing:

Missing GUI features you wanted:

Overall GUI experience (1-10): _____

End of GUI Testing Guide

Total Time: ~1-2 hours (no terminal needed!) Difficulty: Beginner-Friendly Interface: 100% Graphical

Thank you for testing RoboShire!

Version 1.0 | October 24, 2025 | RoboShire Team