MOOS-IvP in an Enterprise Linux (EL) Environment

Scott R. Sideleau
NUWC Newport
Code 2534
scott.sideleau@navy.mil

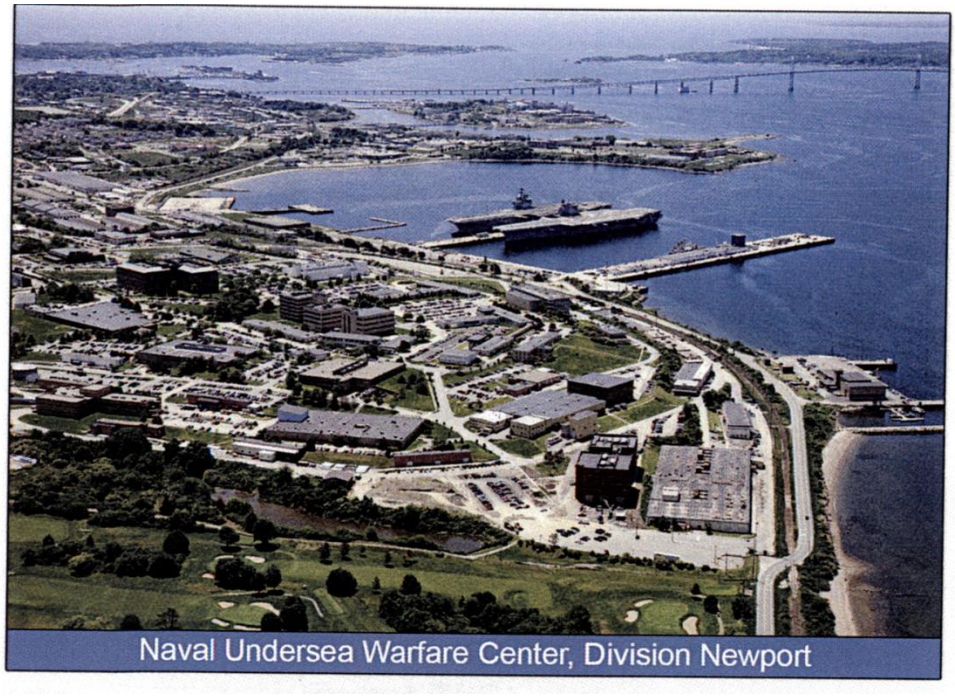
Christopher W. Gagner
NUWC Newport
Code 2534
christopher.gagner@navy.mil

Overview

- About NUWC Newport
  - NUWC Newport
  - Command & Control Asset Pool (C2AP)
  - Iver2 UUV by OceanServer
- Why Enterprise Linux (EL)?
- Backporting essential packages for:
  - MOOS-IvP
  - Gobysoft
- Automation
  - for Development Workstations
  - for Robotic Nodes
- Future Work

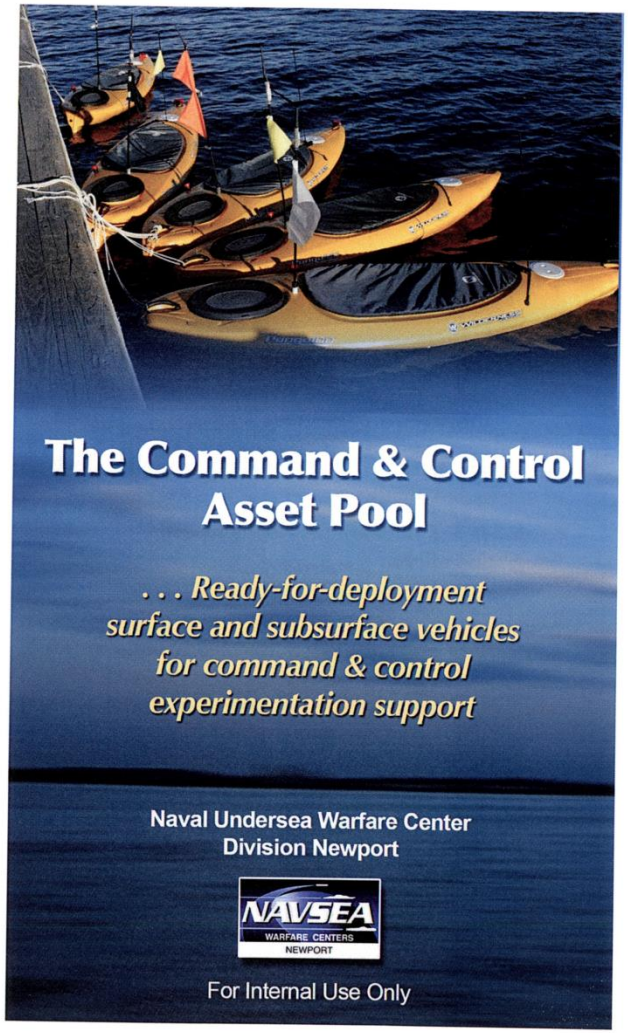
NUWC Newport

- RDT&E engineering and Fleet support facility for the US Navy
- Employing ~4k scientists and engineers (government and contractor support)
- Focus on all aspects of undersea warfare
  - Submarines
  - Offensive and Defensive Weapons Systems
  - Maritime UxVs

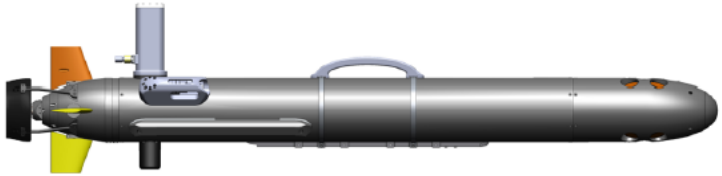

Naval Undersea Warfare Center, Division Newport

Command & Control Asset Pool

- Cross-departmental team with:
  - MOOS-IvP behavior development expertise
  - MOOS App development expertise
  - Modeling & Simulation expertise
  - Operational expertise from participation in several US Navy and NATO exercises around the globe
- Maritime robotics platforms:
  - 3x “classic” Iver2 UUVs
  - 1x Lightweight NSW Iver2 UUV
  - 1x Iver3 UUV
  - 3x Scout ASCs



Iver2 UUV by OceanServer

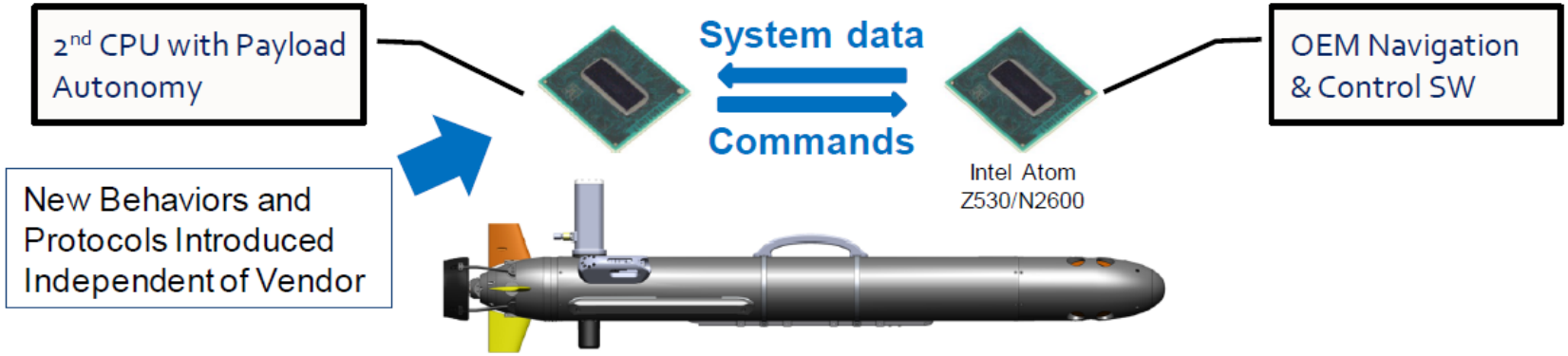


- ✓ **Commercial Product with Transition to Fleet**
  - Developed with private industry internal R&D funds
  - Vehicle components and payload predominantly COTS
  - Continuing upgrades and component integration driven and funded by market (non-Navy) demand
  - Special functions and payloads via Navy development
- ✓ **Low-Cost**
  - Economical procurement and operation
  - Multiple platforms operated at reduced risk
  - Surrogate for high-cost or unavailable platforms
  - $53K-$63K base; $110K-$250K SOF loadout
- ✓ **High Operability and Robustness**
  - Rapid training cycle
  - Flexible launch/recovery
  - On-scene maintenance and repair

RESULT = HIGH OPTEMPO

Enabling C2 Development

- Payload autonomy on secondary CPU to streamline process independent of vendor and reduce development risk
- Fully exposed and documented APIs for all sensors and vehicle control/status parameters
- 2-way exchange of data and commands



Rapid, low-cost development and introduction of new functions independent of vendor

Why use Enterprise Linux (EL)?

- Requirement for DoN systems to run “approved” software
  - Including “approved” operating systems
  - DON Application and Database Management System (DADMS)
- Support is available from the distributor
  - See your license ($$$) terms for details
- Really good at supporting offline systems
  - Red Hat Package Manager (RPM) is powerful
  - Easy to backport packages with SRC.RPMs
- Free versions (CentOS) available, too!

What should I choose?

- Development Environments
  - Red Hat Enterprise Linux 6.6 (Workstation)
  - CentOS 6.6
- Robotic Nodes
  - CentOS 6.6
- Other options:
  - SUSE Enterprise Linux
  - Ubuntu Long Term Service (LTS) releases
    - Not yet listed in DADMS!

We'll focus on Red Hat / CentOS throughout this brief…

What about dependencies?

- Most are available with little or no work by you!
  - But, you can backport anything from Fedora Core (FC) that you don't find readily available.
  - Don't jump straight to third-party source code!
- What repositories do I need to enable?
  - MOOS 14.7.1+
    - Enterprise Packages for Enterprise Linux (EPEL)
    - rhel-6-workstation-rpms
  - Gobysoft 2.0.6+
    - rhel-6-workstation-optional-rpms
- Other recommended:
  - rhel-6-workstation-fastrack-rpms
  - RPM Fusion (similar to Ubuntu's nonfree / licensed)
  - RPM Forge (great for media tools: vlc, mplayer, etc)

What specific packages?

- MOOS-IvP
  - from our **custom** repository:
    - fltk-devel (>= 1.3.2)
  - from the **EPEL** repository:
    - ~~cmake28~~
  - from **rhel6-workstation-rpms**:
    - “Development Tools” group
    - boost-devel
    - freeglut-devel
    - libtiff-devel
    - mesa-libGL-devel
    - mesa-libGLU-devel
    - proj-devel
    - sqlite-devel
    - xterm

More on FLTK 1.3.2+

- rpmfind.net is your friend!
  - Look for recent Fedora Core (FC) releases
  - You want the SRC.RPM version so that you can build an EL6 backport

- Backporting is easy!
  - wget -c http://rpmfind.net/path/to/fltk-1.3.*.rpm
  - rpmbuild --rebuild fltk-1.3.*.rpm
  - rpm -i fltk-1.3.*.rpm
    - Or, better yet, make your own Yum repo!

What about Gobysoft?

- from **EPEL** repository:
  - cryptopp-devel
  - protobuf-devel
  - wt-devel
  - zeromq-devel
- from **rhel-6-workstation-optional-rpms** repository:
  - xerces-c-devel
- from **rhel-6-workstation-rpms** repsitory:
  - gmp-devel

Any other recommendations?

- from **EPEL** repository:
  - ntfs-3g
- from **rhel6-workstation-fastrack-rpms** repository:
  - pciutils
- from **rhel-6-workstation-optional-rpms** repository:
  - elfutils-libelf-devel
- from **rhel-6-workstation-rpms** repository:
  - binutils-devel
  - ncurses-devel
  - sqlite-devel
  - zlib-devel
  - git
  - hmaccalc
  - man
  - minicom
  - ntp
  - ntpdate
  - openssh-clients
  - pkgconfig
  - rng-tools
  - setserial
  - usbutils
  - vim-enhanced
  - wget

What if I use an Iver UUV?

- Thanks for asking!

- OceanServer now has a CentOS 6.6 image available
  - You'll need to specifically ask about it for your Backseat Driver
  - You can mention specifically "the one Scott from NUWC built"

- Derived from "minimal" install of CentOS 6.6
  - Doesn't include FLTK (because it's a headless robot)

Automation

- Sample *.sh scripts for automating building of development machines available

- Disk image of Backseat Driver for Iver2 available
  - Packages used already described!
  - Roll your own images?

Future Work

- We want to hear from you, the user community!
  - Use more CMake automation?
  - Different philosophies to embrace for third-party dependencies?

- Should we setup an online survey to find out more from the user community about what is “hard” in getting started?