REDHAWK

&

Tactical Open Architecture

GEON Technologies, LLC

REDHAWK

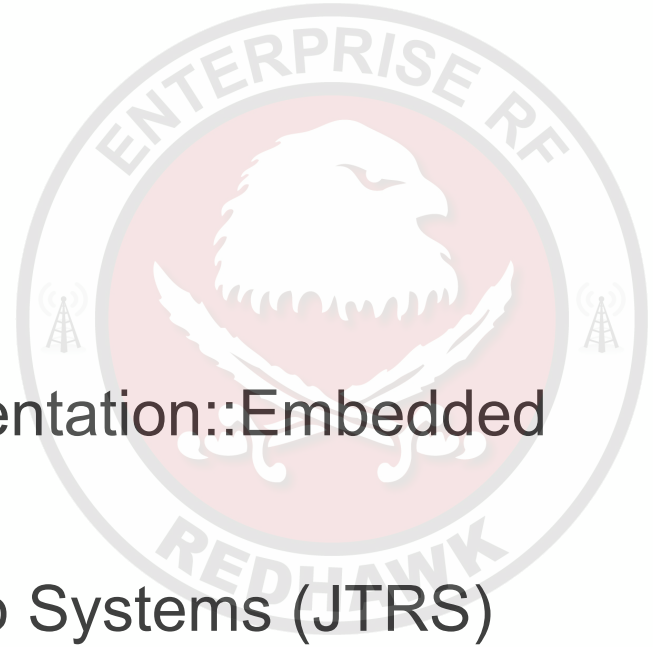
Principal Definition & Key Concepts

An open source software defined radio framework for RF sensor *systems*.

- *Inherently distributed*
- *Rich tooling for system design and capability development*
- *Designed for heterogeneous architectures*
- *Dynamic performance optimization*

REDHAWK History

Circa 2010



- Originally based on the Open Source SCA Implementation::Embedded (OSSIE); early development at Virginia Tech
- Extended by DoD based on the Joint Tactical Radio Systems (JTRS) Software Communications Architecture (SCA) 2.2.2 specification
- First major success in 2011 : 2 blade centers / 8 GHz of simultaneous spectrum on a 10Gb multicast backbone / ~100 waveforms.
- Begin publishing a *modified* SCA 2.2.2 specification in 2012
- Approved as NSA Open Source Software in 2013. Published to Github.

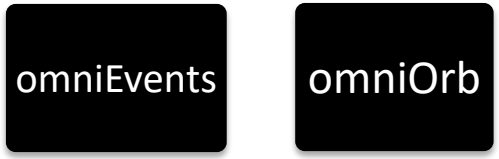
Why REDHAWK

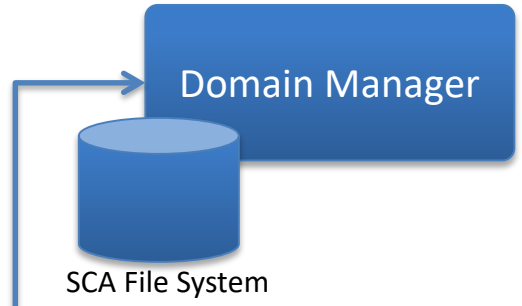
Original Motivations Explain Design …

- Few SDR frameworks are inherently *distributed*
  - Necessary to scale over the entire SWaP-C continuum.
- Few SDR frameworks are *resource aware*
  - The problem is compounded in a heterogeneous sensor architecture (Intel, ARM, GPU, FPGA, MPSoC).
- *No* SDR frameworks are specifically designed as *integration* frameworks
  - Industry has an inestimable value of legacy and evolving IP in *other* frameworks

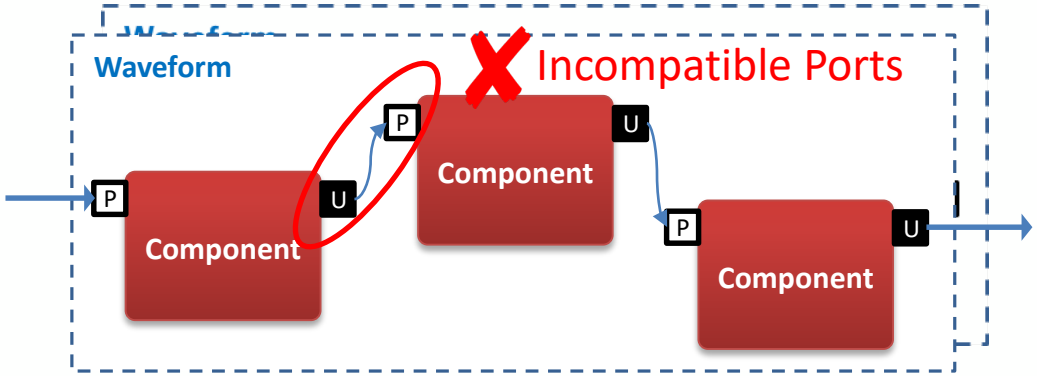
What is REDHAWK?

A Context Diagram



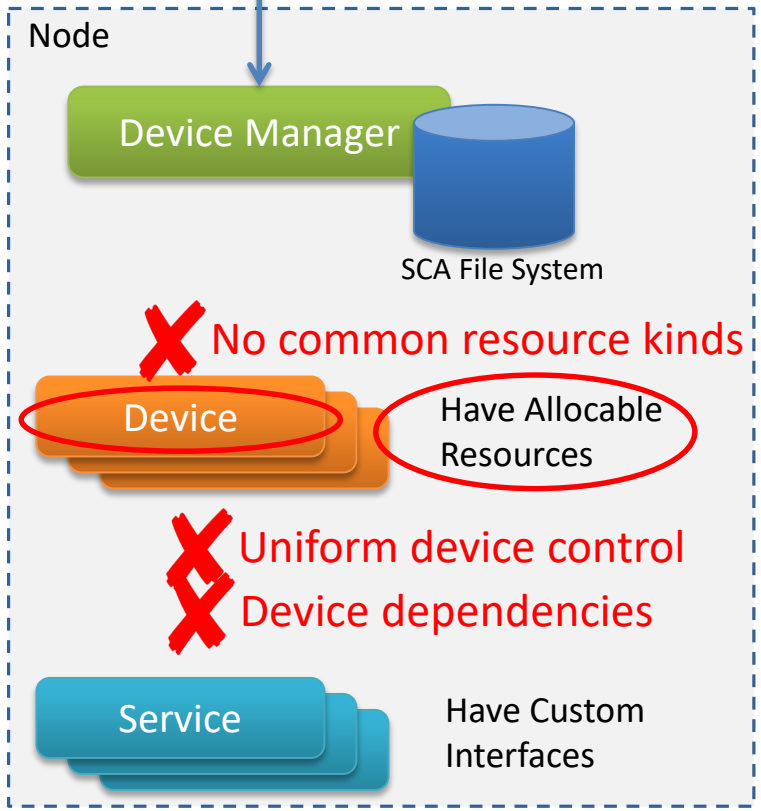


Application : One or more waveforms with the devices and resources needed to support it.



- Have *Ports*
- Have *Properties*
- Have *Connections*

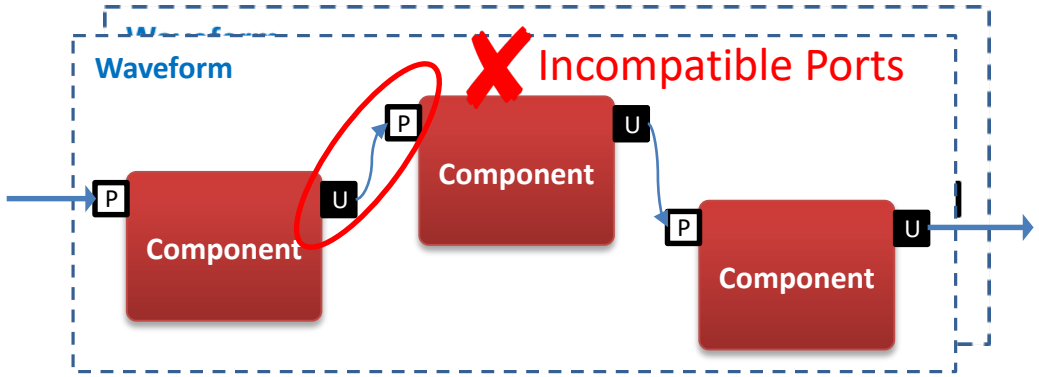
- Require *Resources*



BULKIO & FrontendInterfaces

2 Distinct But Related Problems

1. BULKIO and Messaging
   - An interface specification for *ports* that ensures portability and reuse of processing *components.*
   - BULKIO makes communication between the pieces of a distributed system work and promotes portability.



BULKIO & FrontendInterfaces

2 Distinct But Related Problems

2. FrontendInterfaces
   - Standardizes the resources kinds for frontend RF acquisition hardware
   - Ensures a device abstraction that allows an application to target any analog or digital RF hardware supporting FEI2.0

BULKIO

Inherently distributed connection logic : data and metadata

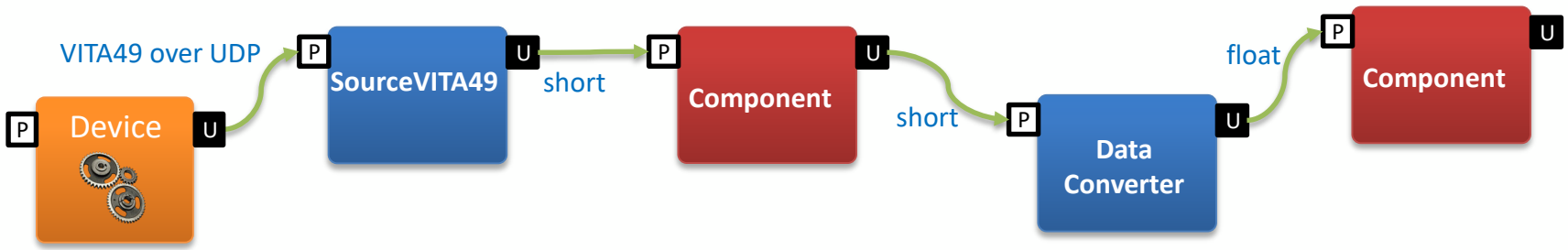
- Supported by the REDHAWK Tooling:
  - *Components* with typed *ports* can be configured using REDHAWK IDE wizards or command line tools; these will be auto-generated with work methods that allow user code to be inserted.
  - All the CORBA complexity is abstracted away

- Simple API
  - pushSRI() – metadata and keyword propagation
  - pushPacket() – data with time stamp

This API has been replace by the StreamAPI for BULKIO but demonstrates the basic mechanism clearly.

- In-band and Out-of-band Data Mechanisms
  - In-band types = short, float, octet, char, etc.
  - Out-of-band = SDDS, Vita49, raw socket, shared memory, and others possible

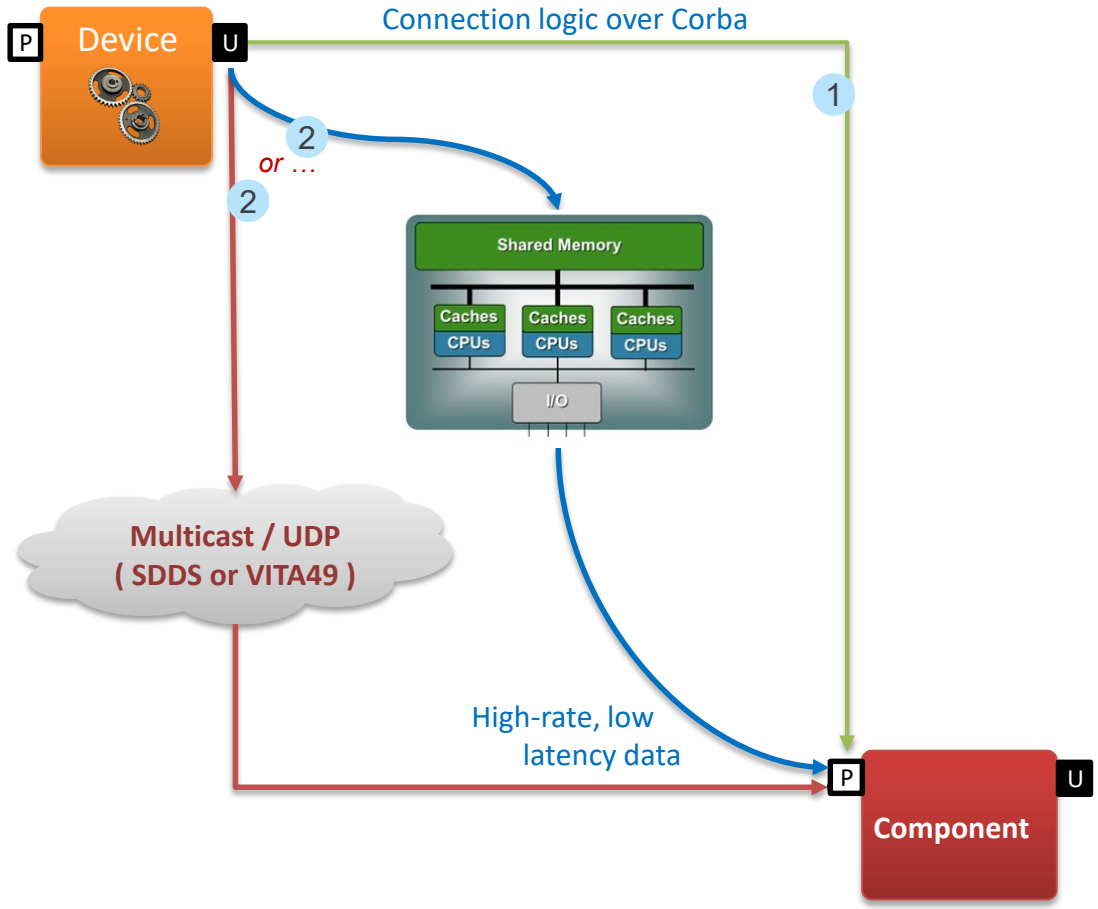
Worst Case Scenario:



BULKIO

Negotiated Performance

- Negotiated connection logic and metadata conveyance handled through CORBA interface
  - Negotiation based on run-time deployment constraints: network, CORBA, shared memory
- Data managed entirely by operating system, environment hardware, and bus / network fabric.



Frontend Interfaces

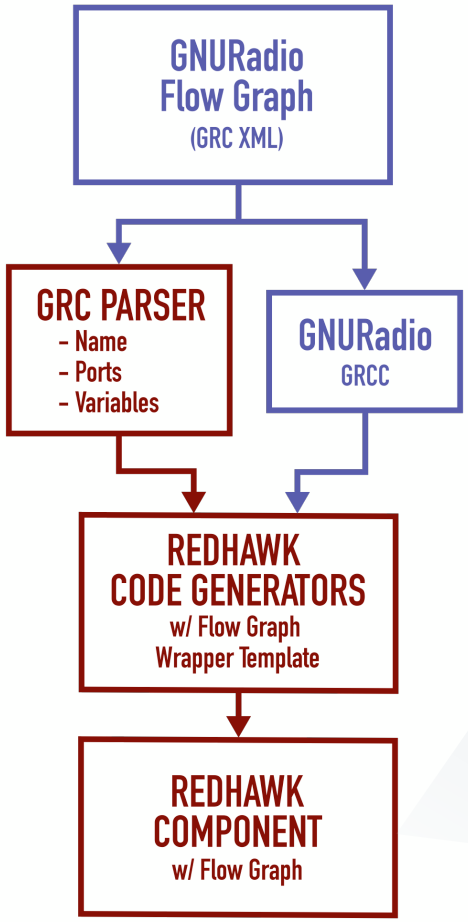
Separate Capabilities from Platforms

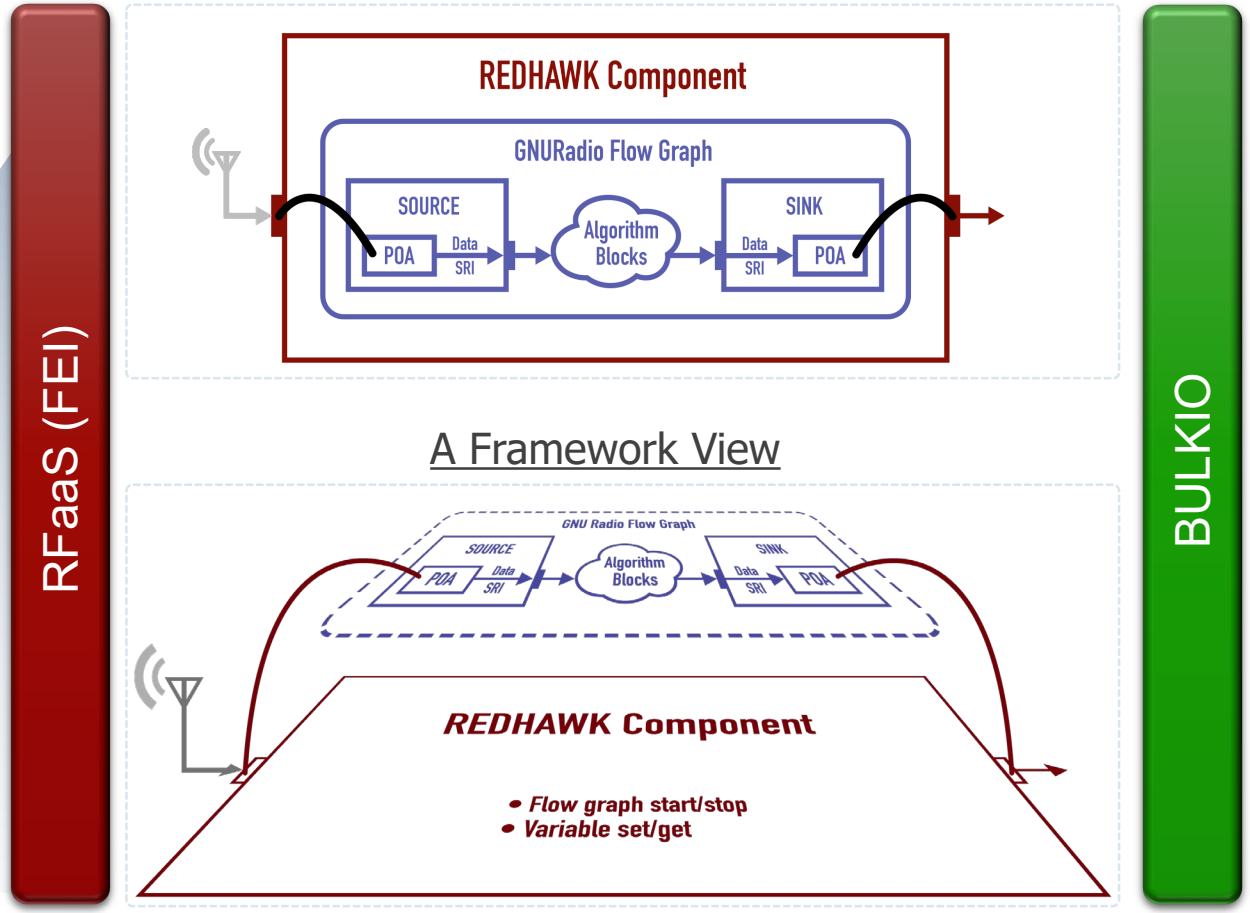
- Defined:
  - Frontend Interfaces is a specification designed to standardize the interaction between Applications and radio hardware.
  - FrontendInterfaces is designed to support the most demanding time and frequency fidelity and metadata propagation.
  - Demonstrated application portability with more than 5 years of operational duty.
  - Supports transmit and receive use-cases

- Overview:
  - Requires specific device kind and device type values
  - Creates a standard set of required and optional allocation properties; a *Property ICD* for radio hardware
  - Defines a C2 Interface for radio control – *FrontendTuner* and its extensions: AnalogTuner and DigitalTuner
  - Defines a C2 Interface for RF flow control – *RFSource*
  - Defines a metadata Interface and specification for describing an RF flow – *RFInfo*
  - Defines a metadata Interface and specification for describing geolocation- *NavData / GPSInfo*

REDHAWK Integration

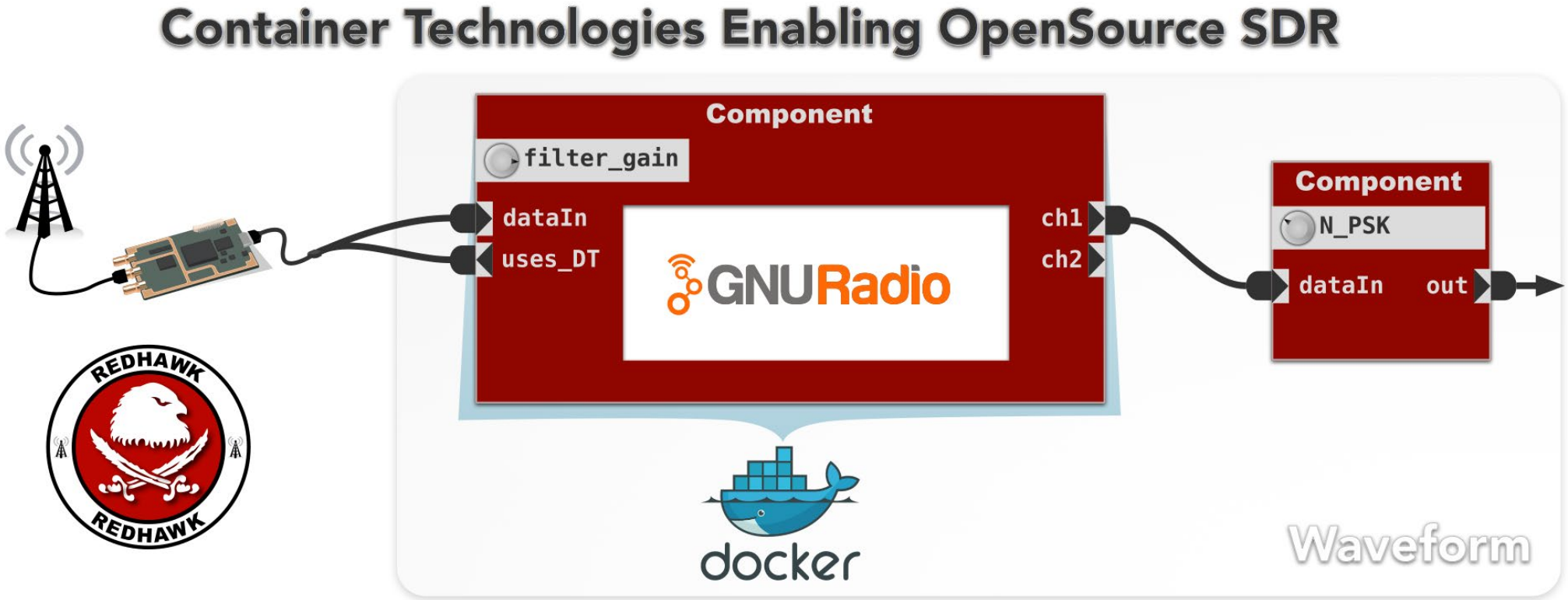
A Case Study : GNURadio





REDHAWK Integration

A Case Study : GNURadio



REDHAWK Integration

Other Examples

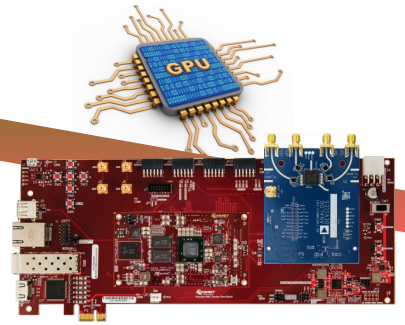
- Octave / Matlab
- XMidas
- SALVAGE
- Java / C++ libraries
- Linux shell command applications

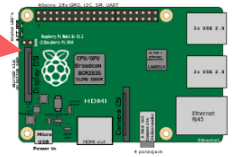
Heterogeneous By Design

From IoT to Data Center

Using

- Yocto and OpenEmbedded
- Persona Pattern : component model extended to FPGA/GPU fabric
- Extends “composable” waveform





Data Center

OpenVPX Intel / ARM

Single and Multi-core SoC & GPU

Edge Processors

Geon Technologies, LLC

Who is Using REDHAWK

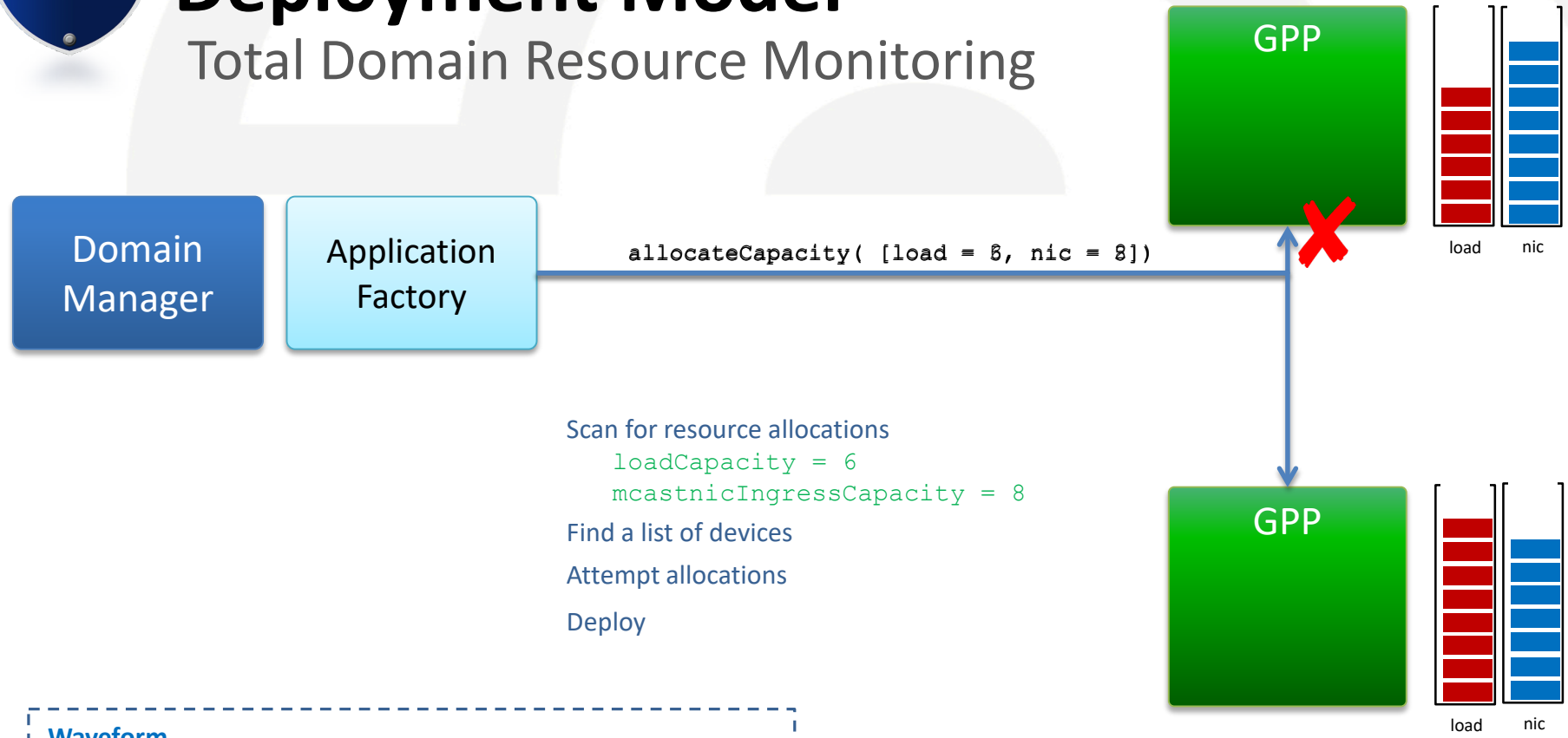
A Growing and Diverse Community

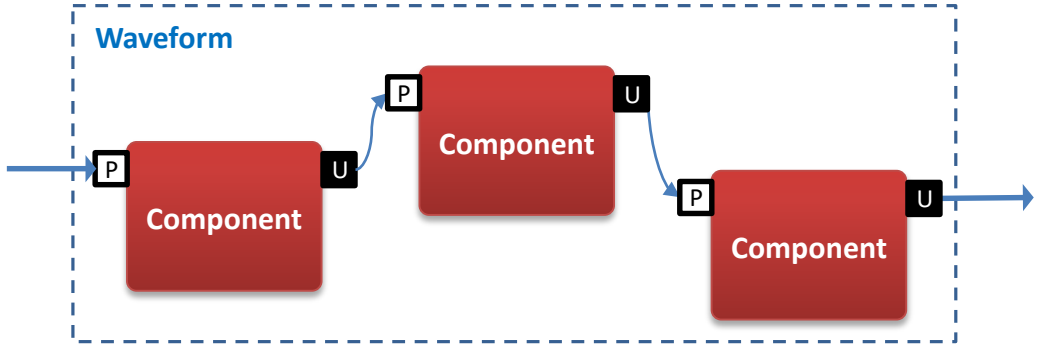
- National and tactical armed services sensor programs (and their industrial base)
- United States Intelligence Community
- Commercial Radar Simulation
- Government and commercial spectrum management organizations
- Hobbyists / Bat Ecologists
- Commercial IoT technology companies


geon
Technologies, LLC

Deployment Model

Total Domain Resource Monitoring





Integrity. Agility. Results.