Technical Description of HLA/NETN Including Interoperation with C2SIM and MSaaS

Tom van den Berg
TNO Applied Physics Laboratory
THE NETHERLANDS

tom.vandenberg@tno.nl

ABSTRACT

M&S as a Service (MSaaS) is a new concept of providing and consuming M&S Services. The concept includes service orientation and the provision of M&S applications via the as-a-service model of cloud computing, and has both an organizational dimension as well as a technical dimension. The concept has the potential to greatly reduce the barriers of cost and accessibility, and to result in greater utility of M&S throughout NATO and the nations.

This paper demonstrates the application of the Kubernetes technology platform in the realization of an MSaaS Capability, in particular for the key capabilities discovery, composition, and execution. This paper also shows the successful application of several simulation standards in cloud-based simulation, including HLA, NETN, WebLVC, and C2SIM.

1.0 INTRODUCTION

M&S as a Service (MSaaS) is a new concept of providing and consuming M&S Services (see [1] for more information). The concept includes service orientation and the provision of M&S applications via the as-a-service model of cloud computing, and has both an organizational dimension as well as a technical dimension. The concept has the potential to greatly reduce the barriers of cost and accessibility, and to result in greater utility of M&S throughout NATO and the nations.

The concept is described in four documents, together called the Allied Framework for MSaaS: MSaaS Operational Concept Description, MSaaS Concept of Employment, MSaaS Business Model, and MSaaS Technical Reference Architecture.

The key capabilities supported by the Allied Framework for MSaaS are described in the MSaaS Operational Concept Description. These are:

- **Discover Services:** The Allied Framework for MSaaS provides a mechanism for users to search and discover M&S services and assets (e.g., Data, Services, Models, Federations, and Scenarios).
- **Compose Services:** The Allied Framework for MSaaS provides the ability to compose discovered services to perform a given simulation use case.
- **Execute Services:** The Allied Framework for MSaaS provides the ability to deploy the composed services automatically on a cloud-based or local computing infrastructure.

The MSaaS Technical Reference Architecture defines the building blocks of an MSaaS Capability, including Portal Applications that provide these key capabilities. It defines requirements and standards associated with building blocks, but does not prescribe how these building blocks should be realized and what technology should be used. The technology choices are left to the implementor.

Over the last years many enabling technologies have emerged from the field of Information and Communications Technology (ICT) that are essential to implementing an MSaaS Capability. Figure 1 from Cloud Native Computing Foundation (CNCF, https://www.cncf.io) illustrates the breadth of cloud-related projects and technology to pick from.

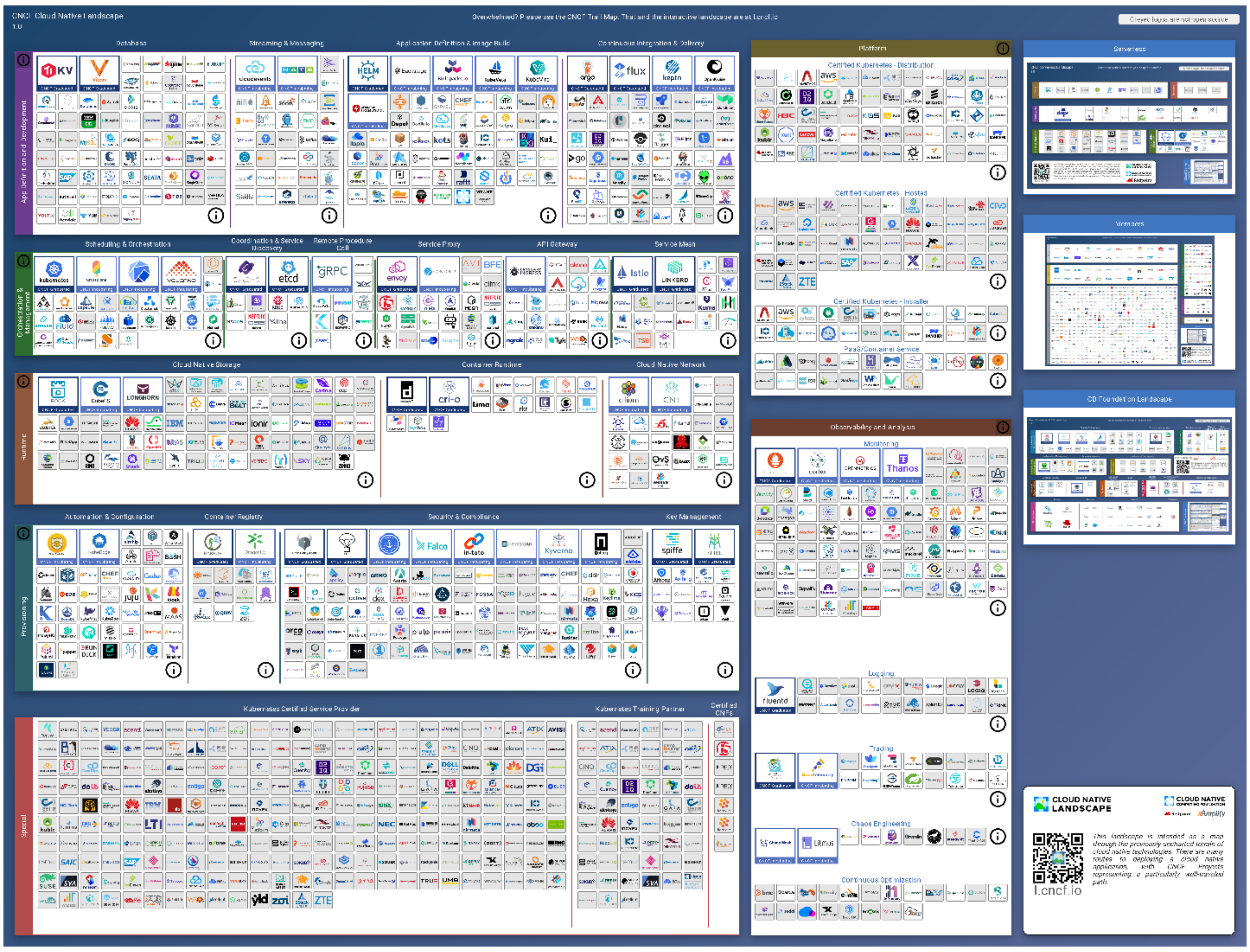

CNCF Cloud Native Landscape
Database
Streaming & Messaging
Application Definition & Image Build
Continuous Integration & Delivery
Platform
Serverless
Scheduling & Orchestration
Coordination & Service Discovery
Remote Procedure Call
Service Proxy
API Gateway
Service Mesh
Cloud Native Storage
Container Runtime
Cloud Native Network
Automation & Configuration
Container Registry
Security & Compliance
Key Management
Observability and Analysis
Kubernetes Certified Service Provider
Kubernetes Training Partner
CLOUD NATIVE LANDSCAPE
l.cncf.io

Figure 1: CNCF Cloud Native Landscape.

Kubernetes is one of the main projects in the CNCF landscape and is an open-source system for automating deployment, scaling, and management of containerized applications. The following sections introduce Kubernetes as technology platform and show how Kubernetes can be used to implement an MSaaS Capability supporting the key capabilities discovery, composition, and execution. Kubernetes is one of the best-known open source platform technologies for container orchestration and has an large ecosystem of tools and applications.

2.0 DOCUMENT OVERVIEW

This paper will guide the reader in using Kubernetes as technology platform for implementing the technical aspects of MSaaS. The structure of this paper is as follows:

- An overview of the Kubernetes technology platform.
- Presentation of the three key MSaaS capabilities using Kubernetes, namely:

    - Discovery.
    - Composition.
    - Execution.
- Perform a small exercise.
- Summary.

3.0 ABOUT KUBERNETES

Kubernetes is a container orchestration environment that can be used to realize a low-cost and feature rich MSaaS Capability, focussed on containerized applications. The original code base has its roots in Borg – a production-grade orchestration system used by Google. There is currently a large and lively community behind the Open Source development and associated ecosystem of Kubernetes [2]. Several major companies use Kubernetes to manage their containerized applications, such as Google, ING Bank, and Booking.com. Kubernetes is nicknamed "K8s" where the digit 8 refers to the eight letters between "K" and "s".

It is out of the scope of this paper to discuss Kubernetes in great depth. The following sections provide a brief overview on what Kubernetes can provide, about containers as underlying technology, and about Helm as package manager for Kubernetes. A good source of information for further reading is the Kubernetes site itself.

3.1 What Services does Kubernetes Provide?

Kubernetes provides an environment for deploying and orchestrating containerized applications in a cluster of so called "worker nodes". Kubernetes consists of several parts to manage the cluster and includes a variety of new concepts which are best explained by the Kubernetes documentation available on the Kubernetes site.

Some of the main services that Kubernetes provides are listed on the Kubernetes site [3] and are as follows:

- **Service discovery and load balancing:**

  Kubernetes can expose a container using the DNS name or using their own IP address. If traffic to a container is high, Kubernetes is able to load balance and distribute the network traffic so that the deployment is stable.

- **Storage orchestration**

  Kubernetes allows you to automatically mount a storage system of your choice, such as local storages, public cloud providers, and more.

- **Automated rollouts and rollbacks**

  You can describe the desired state for your deployed containers using Kubernetes, and it can change the actual state to the desired state at a controlled rate. For example, you can automate Kubernetes to create new containers for your deployment, remove existing containers and adopt all their resources to the new container.

- **Automatic bin packing**

  You provide Kubernetes with a cluster of nodes that it can use to run containerized tasks. You tell Kubernetes how much CPU and memory (RAM) each container needs. Kubernetes can fit containers onto your nodes to make the best use of your resources.

- **Self-healing**

  Kubernetes restarts containers that fail, replaces containers, kills containers that don't respond to your user-defined health check, and doesn't advertise them to clients until they are ready to serve.

- **Secret and configuration management**

  Kubernetes lets you store and manage sensitive information, such as passwords, OAuth tokens, and SSH keys. You can deploy and update secrets and application configuration without rebuilding your container images, and without exposing secrets in your stack configuration.

3.2 What is a Container?

Containers are process execution environments that provide isolation from other applications running on a host. A process running within a container appears to be running in its own execution environment with a dedicated filesystem, memory and process space. The operating system kernel manages resources to allow multiple such containers to run simultaneously on a single computer.

Virtual machines (VM) provide isolation similar to that provided by containers. However, each VM provides a complete operating system that needs to be booted each time it starts. A hypervisor is required to share computing resources between multiple VMs running on a single computer. In contrast, containers run within the context of a single operating system with that operating system's kernel being shared by all containers. In effect, the operating system acts as the hypervisor for containers. This results in start-up and shutdown times for containers being significantly faster than for VMs.

The typical deployment and comparison of virtualization technology is illustrated in Figure 2:

- **Traditional deployment:** legacy M&S applications, e.g., with built-in graphical front-end or with specific hardware.
- **Virtualized deployment:** virtualization of traditional/legacy M&S applications.
- **Container deployment:** containerization of either traditional/legacy or newly architected M&S applications in smaller and interoperable back-end simulation services and web-enabled front-end user interfaces.

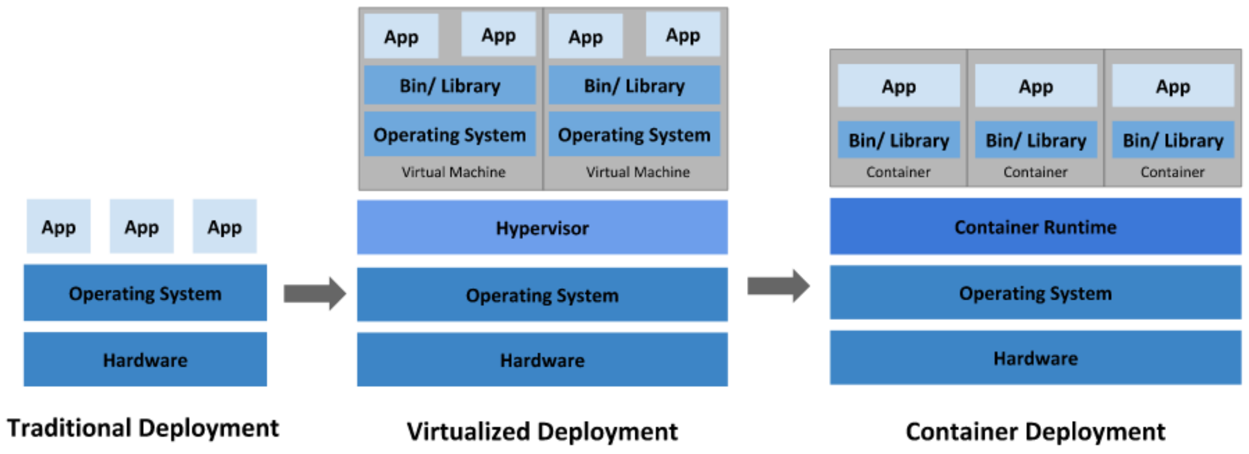


Figure 2: Deployment and virtualization (from [1]).

A common use case for containers is the packaging and deployment of an application inclusive of all its dependencies. This allows multiple applications with different dependencies to be run on a single computer. This is particularly beneficial when applications have a dependency on the same third-party product but at slightly different versions. Deploying those applications in containers allows each to have the version of the dependency they require and removes the need for the host system to have multiple, potentially conflicting, dependencies installed.

This encapsulation also allows a host computer to be minimally configured before being ready to run an application. A host need only have the necessary container runtime engine installed to be able to run containers. This is particularly useful in a cloud computing environment where computing resources can be provisioned on the fly with a minimal standard configuration and be ready to run a wide variety of applications.

3.3 Helm as Kubernetes Package Manager

Helm [4] is a tool in the Kubernetes ecosystem that automates the creation, packaging, configuration, and deployment of Kubernetes applications by combining various configuration files into a single reusable package called Helm Chart, see Figure 3. Helm itself is a command line tool, but a product like Rancher [5] provides a graphical front-end to install, upgrade, or de-install a Helm Chart in a Kubernetes cluster.

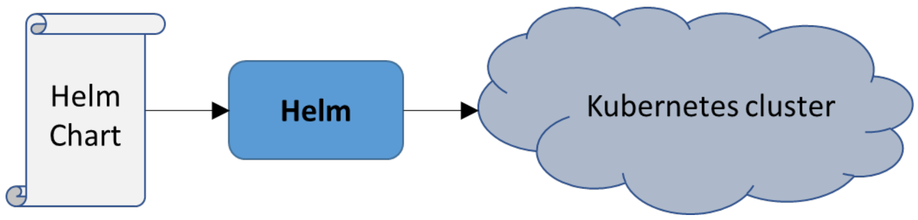


Figure 3: Helm.

A Helm Chart is a collection of files inside of a directory. This includes Kubernetes configuration files for deployments, services, secrets, and config maps that define the desired state of the application. The name of the directory is the name of the chart. Thus, a chart for the pitch-crc would be stored in a pitch-crc directory. Inside of this directory Helm expects a structure as shown in Figure 4.

```
pith-crc/
 Chart.yaml # A YAML file containing information about the chart
 LICENSE # OPTIONAL: A plain text file containing the license for the chart
 README.md # OPTIONAL: A human-readable README file
 values.yaml # The default configuration values for this chart
 values.schema.json # OPTIONAL: A JSON Schema for imposing a structure on the values.yaml file
 charts/ # A directory containing any charts upon which this chart depends.
 crds/ # Custom Resource Definitions
 templates/ # A directory of templates that, when combined with values,
 # will generate valid Kubernetes manifest files.
 templates/NOTES.txt # OPTIONAL: A plain text file containing short usage notes
```

Figure 4: Helm Chart directory structure.

The file named "values.yaml" contains default values of the chart, for instance for the Pitch CRC the network port the application should listen on and the container image version to use. Helm uses the values in this file to expand the templates in the "templates" directory to Kubernetes manifest files. The manifest files are used by Helm to deploy the application in a Kubernetes cluster.

One of the required files in a Helm Chart is named "Chart.yaml". The content of this file provides information about the chart, see Figure 5. Most properties in this file are optional; required properties are the apiVersion, the name of the chart, and the version of the chart. The content of this file is typically used for display in a catalog, or for searching in a registry. An example for the Pitch RTI is provided in Figure 6.

```
apiVersion: The chart API version (required)
name: The name of the chart (required)
version: A SemVer 2 version (required)
kubeVersion: A SemVer range of compatible Kubernetes versions (optional)
description: A single-sentence description of this project (optional)
type: The type of the chart (optional)
keywords:
  - A list of keywords about this project (optional)
home: The URL of this projects home page (optional)
sources:
  - A list of URLs to source code for this project (optional)
dependencies: # A list of the chart requirements (optional)
  - name: The name of the chart (nginx)
    version: The version of the chart ("1.2.3")
    repository: (optional) The repository URL ("https://example.com/charts") or alias ("@repo-name")
    condition: (optional) A yaml path that resolves to a boolean, used for enabling/disabling charts (e.g. subchart1.enabled )
    tags: # (optional)
      - Tags can be used to group charts for enabling/disabling together
    import-values: # (optional)
      - ImportValues holds the mapping of source values to parent key to be imported. Each item can be a string or pair of child/parent sublist items.
    alias: (optional) Alias to be used for the chart. Useful when you have to add the same chart multiple times
maintainers: # (optional)
  - name: The maintainers name (required for each maintainer)
    email: The maintainers email (optional for each maintainer)
    url: A URL for the maintainer (optional for each maintainer)
icon: A URL to an SVG or PNG image to be used as an icon (optional).
appVersion: The version of the app that this contains (optional). Needn't be SemVer. Quotes recommended.
deprecated: Whether this chart is deprecated (optional, boolean)
annotations:
  example: A list of annotations keyed by name (optional).
```

Figure 5: The Chart.yaml file.

```
apiVersion: v1
name: crc
description: Pitch Central Run-Time Infrastructure Component
icon: file://Product-Logo-pRTI-b-150x150.png
version: 1.0
appVersion: 5.5.5.0
home: https://pitchtechnologies.com
keywords:
- HLA
- Simulation
```

Figure 6: Pitch RTI Chart.yaml file.

Helm Charts may be served from a dedicated Helm Chart repository server such as ChartMuseum [6], an HTTP Server, or from a collaboration environment such as GitHub. The use of such a repository server is comparable to how package management for Linux works. The use of Helm Charts is further illustrated in the next chapter.

4.0 MSAAS KEY CAPABILITIES

The following sections discuss the use of Kubernetes as an MSaaS Capability in relation to Discovery, Composition, and Execution.

4.1 Discovery

Central to service discovery is a service registry. The MSaaS Technical Reference Architecture defines M&S Registry Services as "the capabilities to store, manage, search and retrieve data about (i.e., metadata) simulation resources stored by the M&S Repository Services, such as description of services interface and contract, information about QoS policies, and security and versioning information". And M&S Repository Services as "the capabilities to store, retrieve and manage simulation resources and associations with / references to metadata managed by M&S Registry Services."

The concept of registry and repository is shown in Figure 7. Given some event objective, a user searches a registry for suitable simulation services. The associated resources of candidate services may be downloaded from the repository where they are located in.

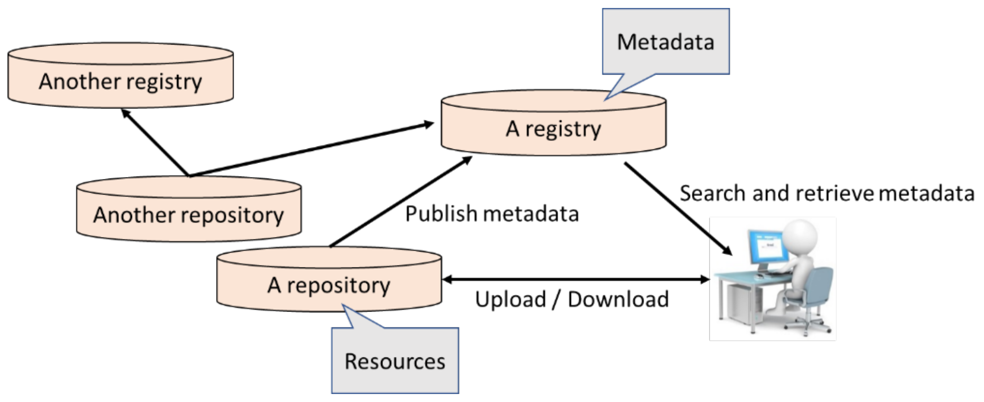


Figure 7: Registry and repository.

A recent development is that of the Artifact Hub [7], as a central place where metadata of cloud-related artifacts such as Kubernetes Helm Charts can be published and searched. The Artifact Hub does not store the artifacts itself, but harvests the metadata from the provided Helm Chart Repositories and makes the metadata available, to be searched in one place. The Artifact Hub thus functions as a registry with metadata.

The Artifact Hub is an open source project and the software can be deployed in an on-premises Kubernetes cluster, obviously using Helm. The opening page of the public Artifact Hub is shown in Figure 8, providing a simple google-like page to search for information.

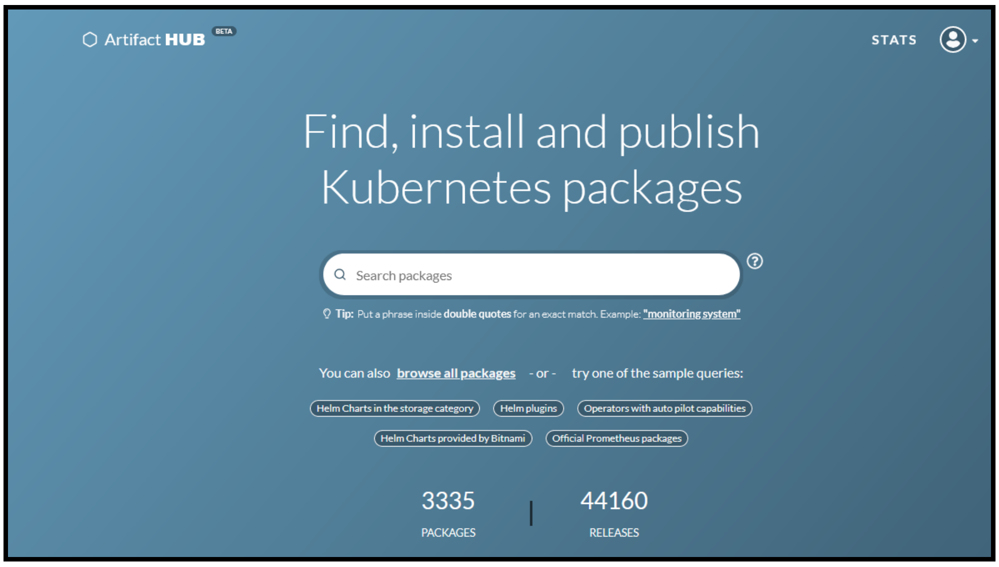
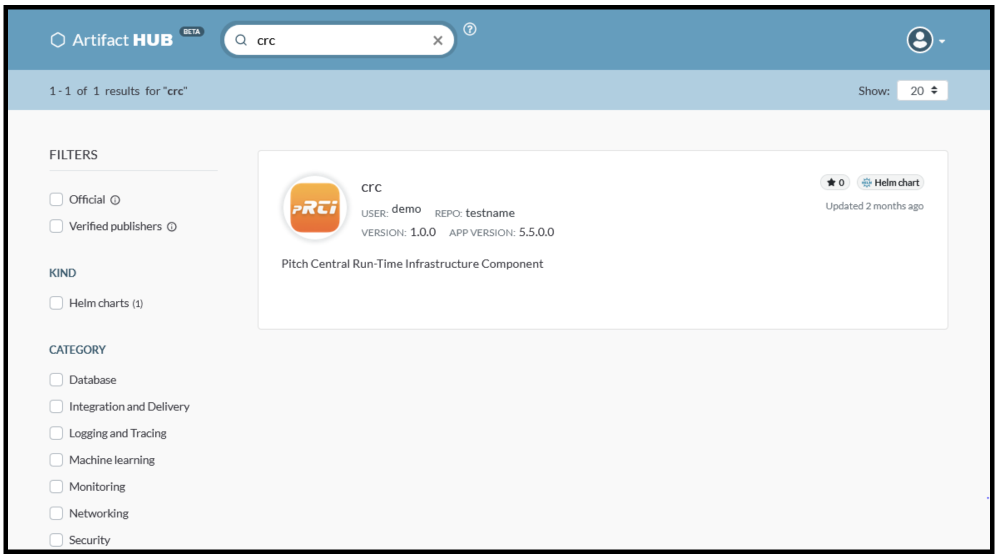

Artifact HUB BETA
STATS
Find, install and publish
Kubernetes packages
Search packages
Tip: Put a phrase inside double quotes for an exact match. Example: "monitoring system"
You can also browse all packages - or - try one of the sample queries:
Helm Charts in the storage category
Helm plugins
Operators with auto pilot capabilities
Helm Charts provided by Bitnami
Official Prometheus packages
3335
PACKAGES
44160
RELEASES

Figure 8: Artifact Hub to share and search cloud-related artifacts such as Helm Charts.

A fictitious search for “crc” yields in this example only one cloud-artifact, namely the Pitch RTI [8], as shown in Figure 9. Clicking on the discovered artifact provides more information such as type of artifact, version information, and where to get it; see Figure 10. Note that this information is all metadata. The artifact itself (a Kubernetes Helm Chart in this example) is located in the supplier’s Helm Chart repository.

The address of the Helm Chart Repository where the discovered artifact (a Helm Chart) is located in must be added to the local Helm tool or added as a new Helm Chart Repository in Rancher, so that Helm can resolve the discovered Helm Chart.

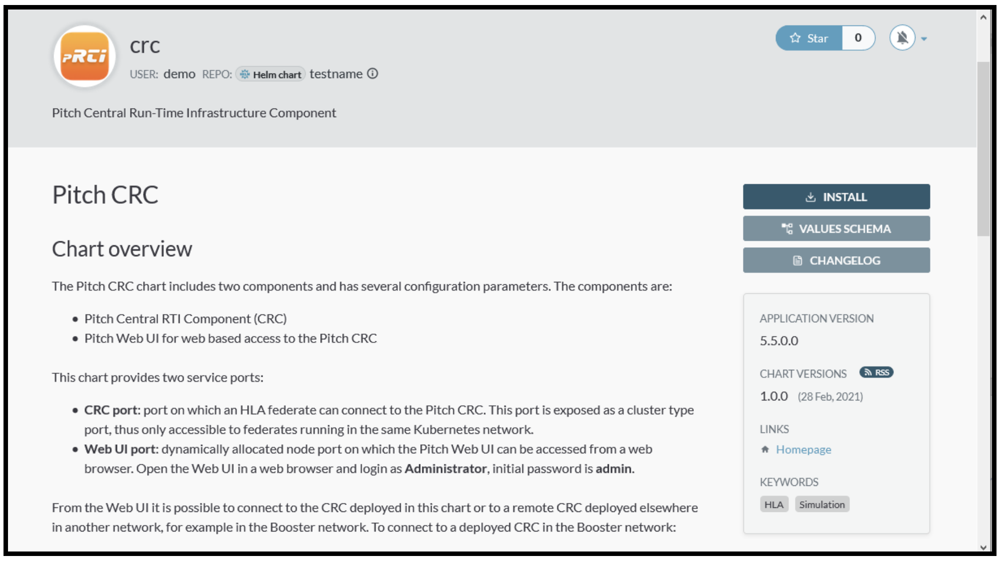


1 - 1 of 1 results for "crc"

crc
USER: demo REPO: testname
VERSION: 1.0.0 APP VERSION: 5.5.0.0
Updated 2 months ago

Pitch Central Run-Time Infrastructure Component

Figure 9: Discovered cloud-artifact for the Pitch RTI.



crc
USER: demo REPO: Helm chart testname

Pitch Central Run-Time Infrastructure Component

Pitch CRC

Chart overview

The Pitch CRC chart includes two components and has several configuration parameters. The components are:

- Pitch Central RTI Component (CRC)
- Pitch Web UI for web based access to the Pitch CRC

This chart provides two service ports:

- **CRC port:** port on which an HLA federate can connect to the Pitch CRC. This port is exposed as a cluster type port, thus only accessible to federates running in the same Kubernetes network.
- **Web UI port:** dynamically allocated node port on which the Pitch Web UI can be accessed from a web browser. Open the Web UI in a web browser and login as **Administrator**, initial password is **admin**.

From the Web UI it is possible to connect to the CRC deployed in this chart or to a remote CRC deployed elsewhere in another network, for example in the Booster network. To connect to a deployed CRC in the Booster network:

APPLICATION VERSION
5.5.0.0

CHART VERSIONS
1.0.0 (28 Feb, 2021)

LINKS
Homepage

KEYWORDS
HLA Simulation

Figure 10: Cloud-artifact metadata.

4.2 Composition

The composition of M&S Resources is in general an engineering activity where human effort is required to integrate and test these resources. MSaaS is not a magic bullet that will solve the interoperability challenges associated with composing M&S Resources, and take away the DSEEP [9] engineering activities. These activities remain to be valid as before. However, an MSaaS Capability can offer several services to support the composition of M&S Resources. For example,

- Metadata repository services to provide access to schema files for object models used in the MSaaS Capability. For instance, HLA reference FOMs (e.g. RPR-FOM and NETN-FOM versions), reference FOM modules, and related interoperability requirements;
- Component registry services to provide information about simulation components and thereby aid the selection, configuration, and integration of simulation components;
- Component repository services to provide access to the actual simulation components; and
- Component test services to verify if simulation components comply with the interoperability requirements.

Let's continue the example with Kubernetes. A simulation application is described by a Helm Chart where the chart is a collection of text files that declaratively describe what to deploy. A chart can be used to deploy a single simulation application, or something more complex such as an entire composition of simulation applications. A composition of simulation applications can be created by adding (dropping in) sub-charts to the main Helm Chart, or alternatively, by creating dependencies between Helm Charts (see Figure 11). With the Helm "package" command the chart dependencies are retrieved from the referenced Helm Chart repositories and added as sub-charts to the main chart. The Helm "install" command will automatically deploy the main chart and any contained sub-charts using Kubernetes as the orchestrator.

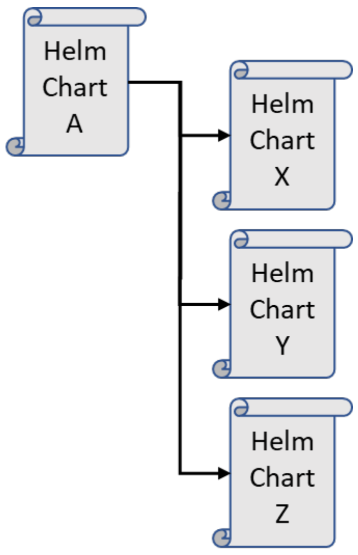


Figure 11: Kubernetes Helm Chart composition with sub-charts X, Y, and Z.

The activity of composing a new Helm Chart from existing Helm Charts described above assumes that the referenced simulation components are composable at both an engineering level and a conceptual level. Engineering composability is defined in Petty's composability Lexicon [10] and concerns the technical or implementation aspects of composing components. For instance, interfacing mechanisms, data formats, and physical timing (i.e., can components be technically and syntactically connected). The other type of composability defined by Petty is modelling (conceptual) composability. This concerns the question whether models that make up the composition can be meaningfully composed (i.e., is the combination of models semantically valid). The main activity in composing Helm Charts concerns configurability, i.e. the configuration of Helm sub-charts (by editing their "values.yaml" file) such that they can function as part of the larger composition.

The creation of the composition of Helm Charts is mostly a manual task, but the mentioned support services can help the engineer to perform this task. The composition can be uploaded to a Helm Chart Repository (typically your own one) as a new Helm Chart, where the metadata of the new chart can be made available to the public or a private Artifact Hub, see Figure 12.

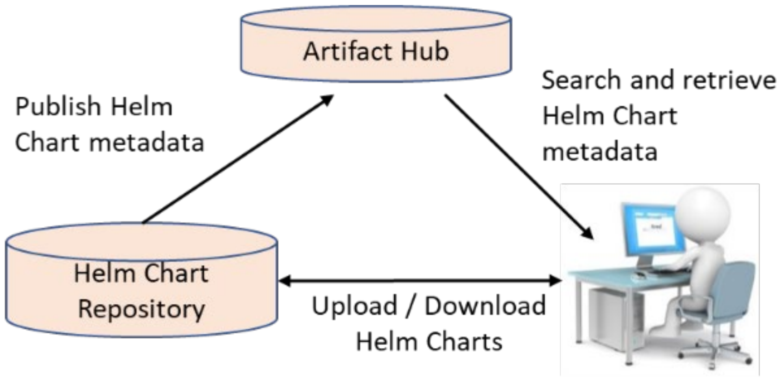

Artifact Hub
Publish Helm Chart metadata
Search and retrieve Helm Chart metadata
Helm Chart Repository
Upload / Download Helm Charts

Figure 12: Helm Chart development.

4.3 Execution

Rancher [5] is a management tool to deploy and manage Kubernetes clusters on sets of compute nodes. Rancher provides a UI to manage and control workloads in each Kubernetes cluster, and a catalog from where Helm Charts can be browsed and started in a cluster. The data that is shown in the catalog is retrieved from public or private Helm Chart repositories and/or a Git repositories, see Figure 13.

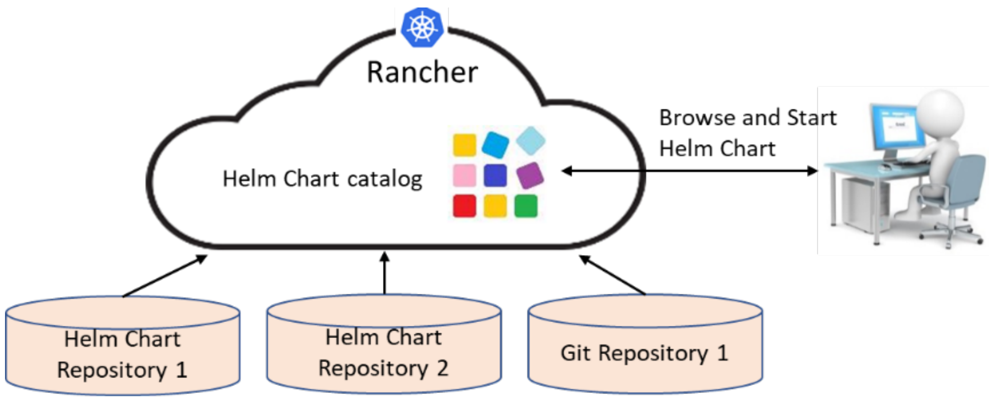
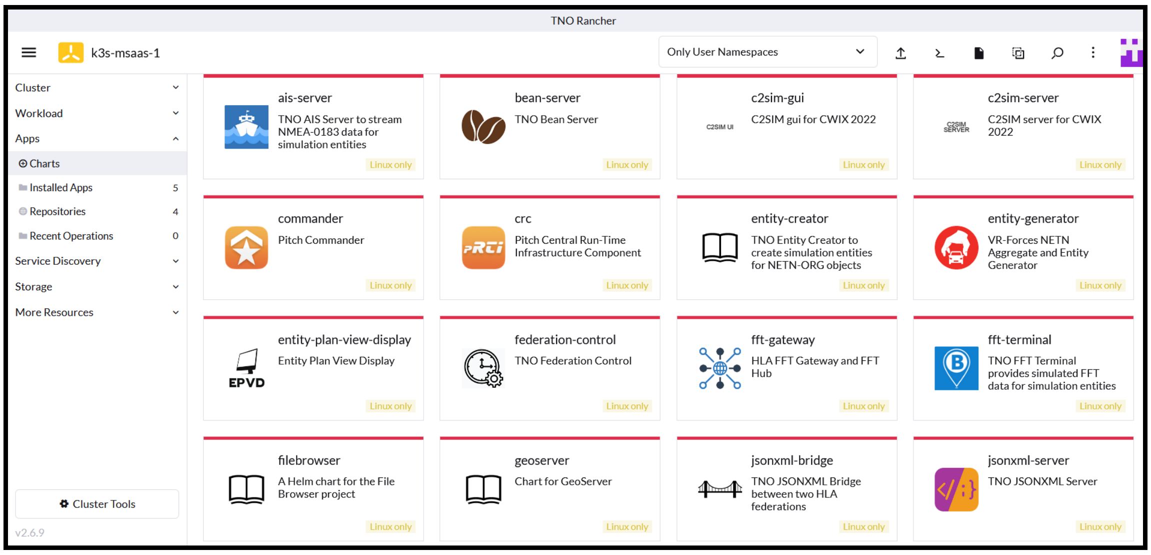


Figure 13: Rancher Helm Chart catalog to browse and start Helm Charts.

An example of a Rancher Helm Chart catalog is provided in Figure 14. From this catalog the user can start a Helm application with just a few clicks. For example, after selecting the Pitch RTI (CRC), the user runs through two steps where he can provide application deployment-specific configuration options such as shown in Figure 15. The application is deployed in the Kubernetes cluster by simply pressing the Install button. Note again that an application can be single simulation application, or a composition of simulation applications.



ais-server
TNO AIS Server to stream NMEA-0183 data for simulation entities
Linux only

bean-server
TNO Bean Server
Linux only

c2sim-gui
C2SIM gui for CWIX 2022
Linux only

c2sim-server
C2SIM server for CWIX 2022
Linux only

commander
Pitch Commander
Linux only

crc
Pitch Central Run-Time Infrastructure Component
Linux only

entity-creator
TNO Entity Creator to create simulation entities for NETN-ORG objects
Linux only

entity-generator
VR-Forces NETN Aggregate and Entity Generator
Linux only

entity-plan-view-display
Entity Plan View Display
Linux only

federation-control
TNO Federation Control
Linux only

fft-gateway
HLA FFT Gateway and FFT Hub
Linux only

fft-terminal
TNO FFT Terminal provides simulated FFT data for simulation entities
Linux only

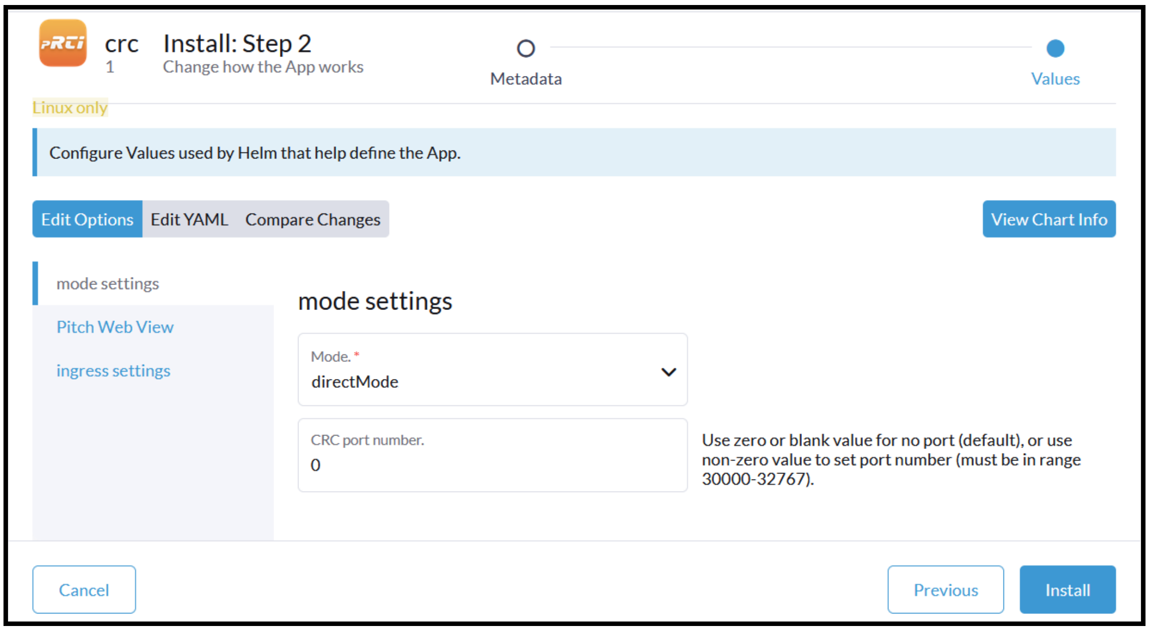
filebrowser
A Helm chart for the File Browser project
Linux only

geoserver
Chart for GeoServer
Linux only

jsonxml-bridge
TNO JSONXML Bridge between two HLA federations
Linux only

jsonxml-server
TNO JSONXML Server
Linux only

Figure 14: Rancher Helm Chart catalog with simulation applications.



Install: Step 2
Change how the App works

Linux only

Configure Values used by Helm that help define the App.

mode settings

Mode. *
directMode

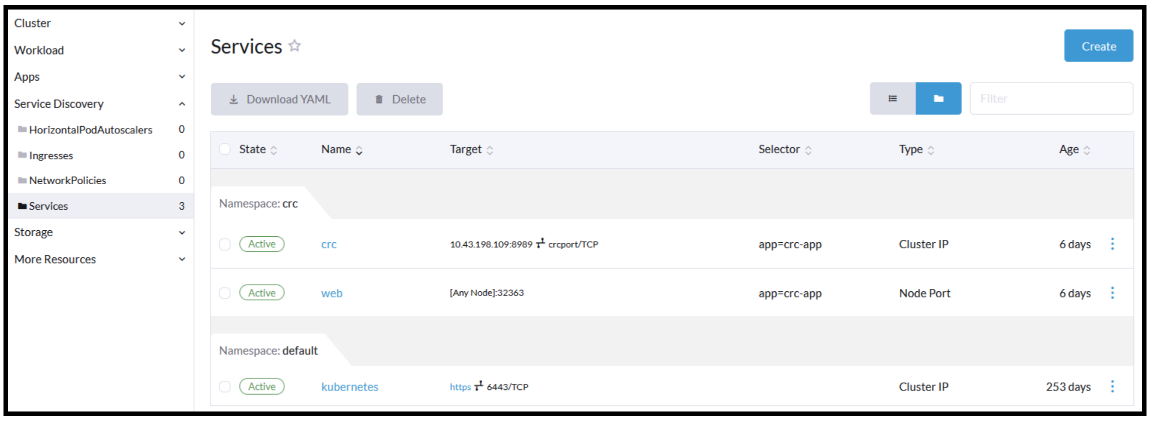
CRC port number.
0

Use zero or blank value for no port (default), or use non-zero value to set port number (must be in range 30000-32767).

Figure 15: Helm Chart configuration options.

On pressing Install the required container images are pulled on-demand from container repositories, such as the public Docker Hub or a private Docker Registry. Several Kubernetes workloads will be created for the application in the container orchestration environment, depending on the content of the Helm Chart.

Provided services (if any) and their exposed network ports can be found under the Service Discovery entry in the Rancher UI. In the example of the Pitch RTI only one workload is created providing two services (see Figure 16): one service for HLA federates to connect to (named the “crc” service), and one service for the end-user (named the “web” service). The latter service listens on port 32363 and serves the Pitch RTI Web UI shown in Figure 17.



State	Name	Target	Selector	Type	Age
Namespace: crc					
Active	crc	10.43.198.109:8989 crcport/TCP	app=crc-app	Cluster IP	6 days
Active	web	[Any Node]:32363	app=crc-app	Node Port	6 days
Namespace: default					
Active	kubernetes	https 6443/TCP		Cluster IP	253 days

Figure 16: Kubernetes services.

Pitch Web View
Server version 2.3.3
PITCH
Welcome to Pitch Web View
TNO CWIX 2023 (CRC Web View)
Any illegal use of this service will be prosecuted
to the maximum extent of the law.
Login as:
Guest
Login
Go to mobile version

Figure 17: Pitch Web UI.

The Rancher Helm Chart catalog is just one way of starting a Helm application. An application may also be started or stopped via the Helm command line tool. The latter interface enables the implementation of other (user provided) applications that can start, monitor, and control applications in a Helm Chart repository.

5.0 A SMALL EXERCISE

This chapter runs through a small simulation exercise involving a number of simulation applications that are deployed on-demand in a Kubernetes cluster. The exercise demonstrates the on-demand deployment and cloud-based execution of applications in a Kubernetes cluster, the simulation initialization pattern using C2SIM LOX INI and HLA/NETN, and the entity tasking and reporting pattern using HLA/NETN (see [11]).

Two actors are involved in the simulation exercise: the Provider managing and controlling the simulation applications in the Kubernetes cluster, and the User requesting and using the provided services for performing the exercise, see Figure 18. The Provider can choose from a range of simulation applications in the catalog. These applications are interoperable by design and can be used in conjunction for a variety of purposes. A few are used for the purpose of this exercise.

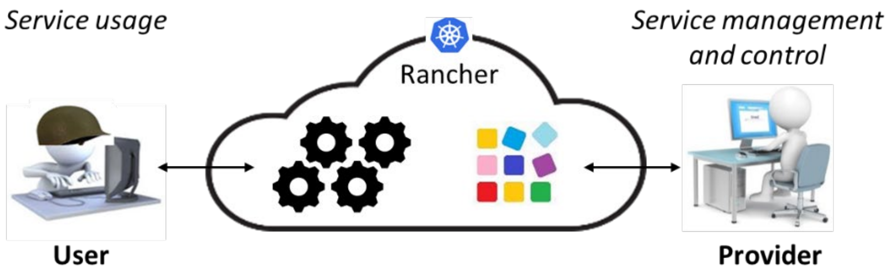


Figure 18: A small exercise involving a Provider and a User.

The Provider is in this example responsible for starting and stopping the simulation applications, and the User is responsible for the initialization of the simulation and the issueing of simulation tasks via the provided service interface (a web UI). The individual applications are started by the Provider one by one for demonstration purposes (step 1, 3, and 4 in the table below). And while the Provider is starting the simulation applications the User already submits simulation initialisation data as soon as the Simulation Control Application has been started (step 2 in the table).

The steps in the exercise are as follows:

Step	Actor	Description
	Provider	Start a Simulation Control Application (TNO Entity Plan View Display)
	User	Initialize the simulation with C2SIM LOX Initialization data
	Provider	Start a Computer Generated Forces (CGF) application (VTMaK VR-Forces)
	Provider	Start another simple CGF application (TNO Entity Creator)
	User	Issue a NETN MoveToLocation task to an entity in the simulation
	User	Issue a NETN MagicMove task to an entity in the simulation
	Provider	Terminate the applications

Once the three applications have been started the HLA federation shown in Figure 19 is created in the Kubernetes cluster. The number above each federate name refers to the step in the exercise in which the HLA federate is created and joining the federation. The arrows in the figure relate to the data flow between the HLA federate and the RTI, that is, the publication and subscription agreements of the applications. The applications all use the NETN FOM as the federation object model (see [11] for more information).

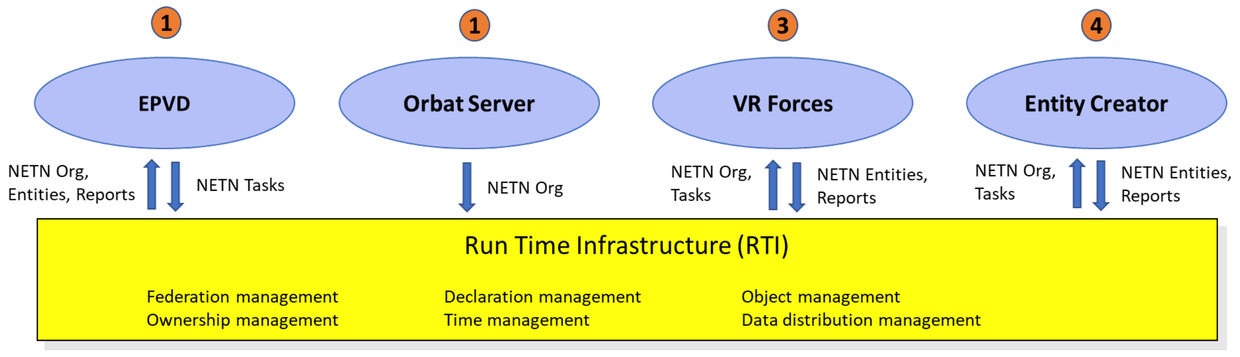


Figure 19: A small HLA federation.

The steps in the exercise are described in the next sections where we will see the following standards in action:

- C2SIM LOX (SISO-STD-019-2020 and SISO-STD-019-2020), MSDL (SISO-STD-007-2008), and NETN-ORG (AMSP-04B): for the initialization of the simulation.
- NETN-ETR (AMSP-04B): for the tasking and reporting of simulation entities.
- HLA (IEEE 1516-2010): for federating applications in a simulation environment.
- WebLVC (SISO-STD-017-2022): for the communication of simulation data within the Entity Plan View Display.

5.1 Start TNO Entity Plan View Display

The TNO Entity Plan View Display (EPVD) is a simulation control application that enables the user to initialize the simulation with MSDL or C2SIM LOX Initialization data, and to submit NETN tasks to entities in the simulation via a web UI.

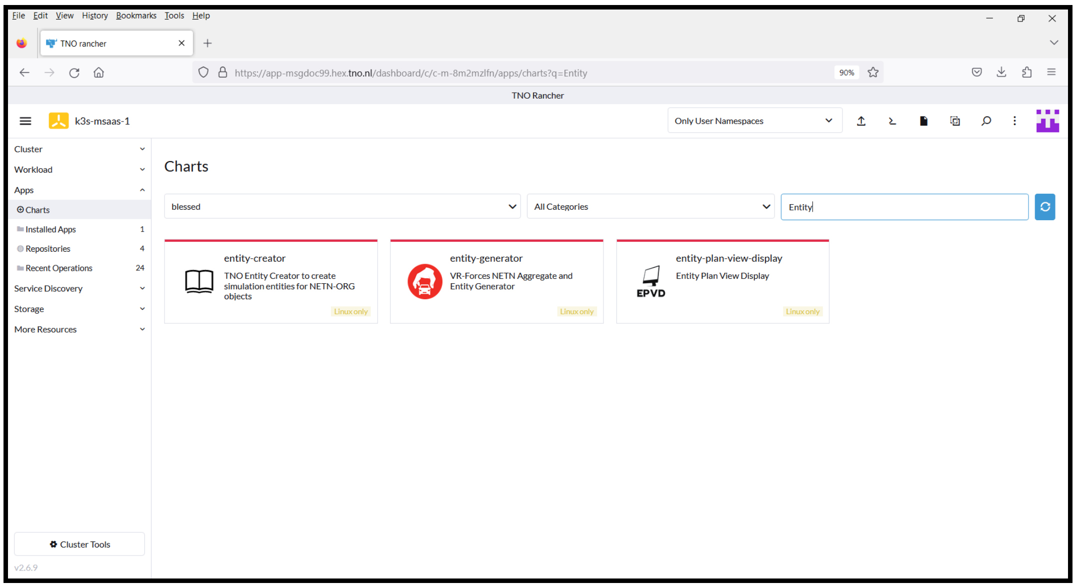
The EPVD consists of several containerized parts:

- The EPVD back-end part handles the data exchange between the EPVD front-end part and the HLA RTI. This part is a WebLVC Server that joins the federation under the federate type name EPVD.
- The EPVD front-end part provides the user interface (web UI). This part is a Node.js based application.
- An Orbat Server manages the initialization of the simulation. The Orbat Server is controlled from the EPVD front-end part and joins the federation under the federate type name ORBATSERVER.

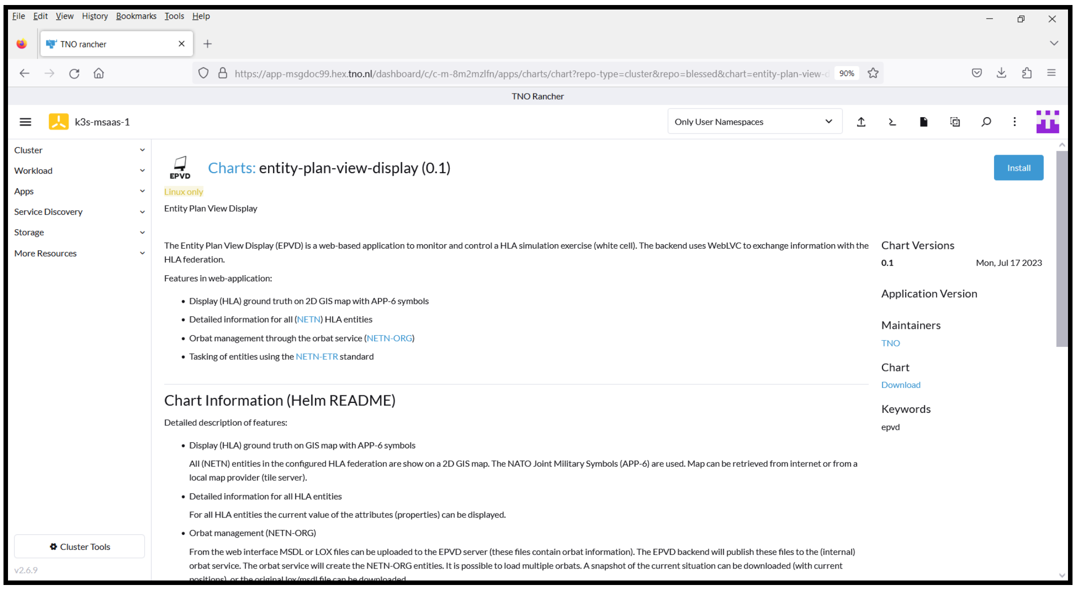
The communication between the EPVD back-end and front-end part uses WebLVC (SISO-STD-017-2022). WebLVC specifies a standard way of encoding simulation messages in JSON (JavaScript Object Notation), where JSON is a commonly used format for the exchange of data in web-based applications.

The EPVD is the first application that will be started in this exercise (besides the Pitch CRC). The tasks to start the EPVD are as follows:

1) Browse the catalog for the application name Entity Plan View Display.



2) Open the Helm Chart from the catalog.



Charts: entity-plan-view-display (0.1)

Linux only

Entity Plan View Display

The Entity Plan View Display (EPVD) is a web-based application to monitor and control a HLA simulation exercise (white cell). The backend uses WebLVC to exchange information with the HLA federation.

Features in web-application:

- Display (HLA) ground truth on 2D GIS map with APP-6 symbols
- Detailed information for all (NETN) HLA entities
- Orbat management through the orbat service (NETN-ORG)
- Tasking of entities using the NETN-ETR standard

Chart Information (Helm README)

Detailed description of features:

- Display (HLA) ground truth on GIS map with APP-6 symbols

  All (NETN) entities in the configured HLA federation are show on a 2D GIS map. The NATO Joint Military Symbols (APP-6) are used. Map can be retrieved from internet or from a local map provider (tile server).
- Detailed information for all HLA entities

  For all HLA entities the current value of the attributes (properties) can be displayed.
- Orbat management (NETN-ORG)

  From the web interface MSDL or LOX files can be uploaded to the EPVD server (these files contain orbat information). The EPVD backend will publish these files to the (internal) orbat service. The orbat service will create the NETN-ORG entities. It is possible to load multiple orbats. A snapshot of the current situation can be downloaded (with current

Chart Versions
0.1 Mon, Jul 17 2023

Application Version

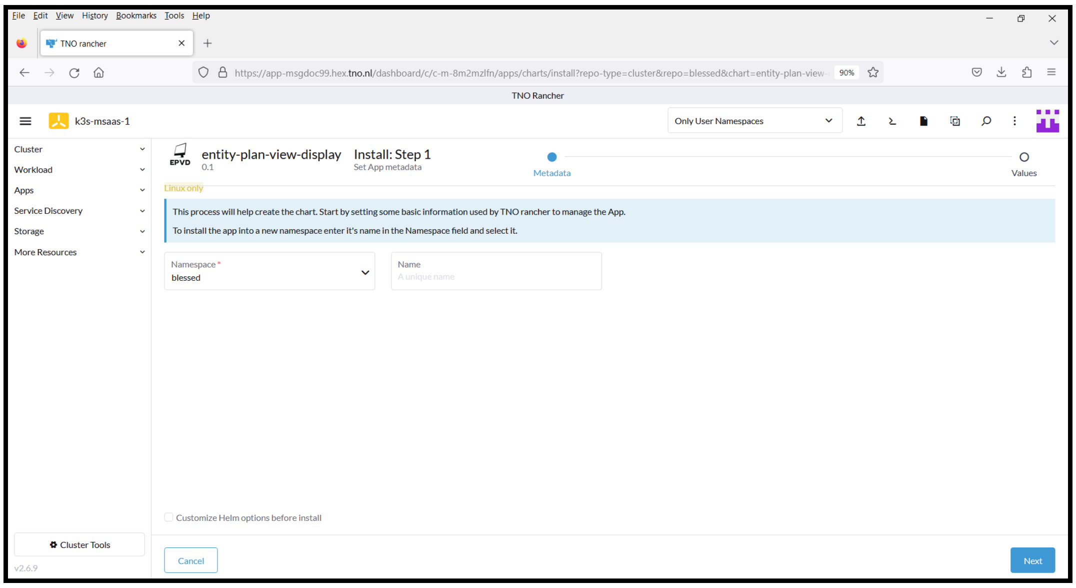
Maintainers
TNO

Chart
Download

Keywords
epvd

https://app-msgdoc99.hex.tno.nl/dashboard/c/c-m-8m2mzlfn/apps/charts/chart?repo-type=cluster&repo=blessed&chart=entity-plan-view-d
TNO Rancher
k3s-msaas-1
Only User Namespaces
Cluster
Workload
Apps
Service Discovery
Storage
More Resources
Cluster Tools
v2.6.9
Install

3) Provide the Kubernetes namespace and deployment name for the application (step 1). If the name is left empty, then Rancher will generate a name.



entity-plan-view-display 0.1

Install: Step 1
Set App metadata

Linux only

This process will help create the chart. Start by setting some basic information used by TNO rancher to manage the App.

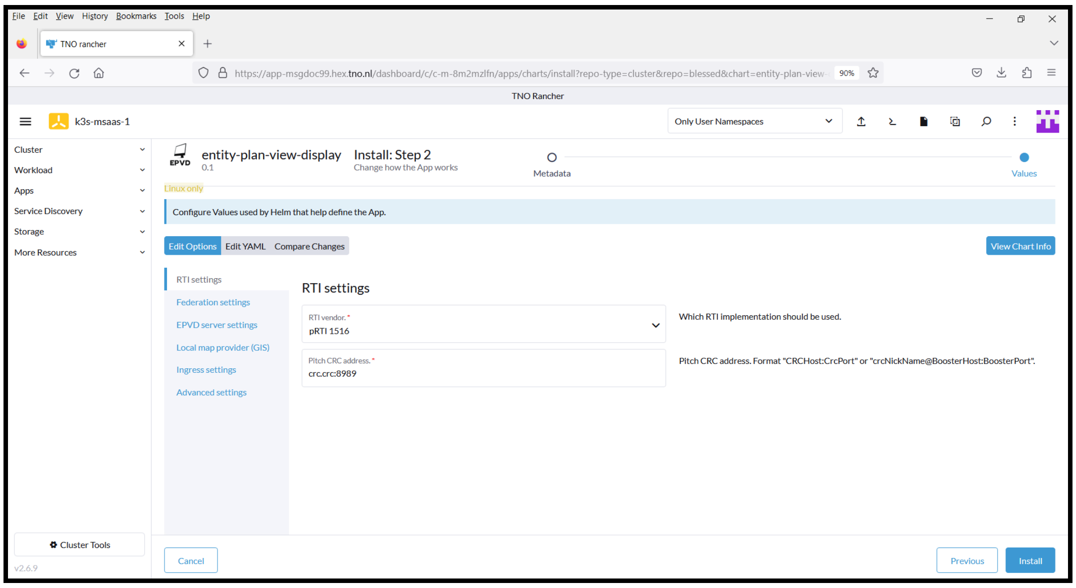
To install the app into a new namespace enter it's name in the Namespace field and select it.

Namespace *: blessed

Name: A unique name

Customize Helm options before install

4) Provide application related information (step 2), such as RTI connection information.



https://app-msgdoc99.hex.tno.nl/dashboard/c/c-m-8m2mzlfn/apps/charts/install?repo-type=cluster&repo=blessed&chart=entity-plan-view-

entity-plan-view-display 0.1 — Install: Step 2 — Change how the App works

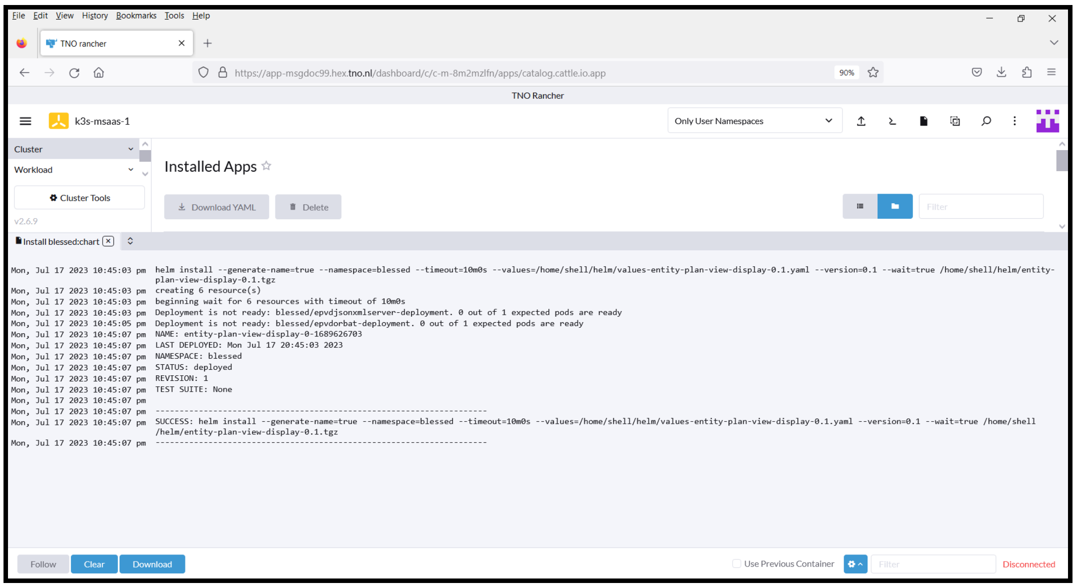
Configure Values used by Helm that help define the App.

RTI settings

RTI vendor.* pRTI 1516 — Which RTI implementation should be used.

Pitch CRC address.* crc.crc:8989 — Pitch CRC address. Format "CRCHost:CrcPort" or "crcNickName@BoosterHost:BoosterPort".

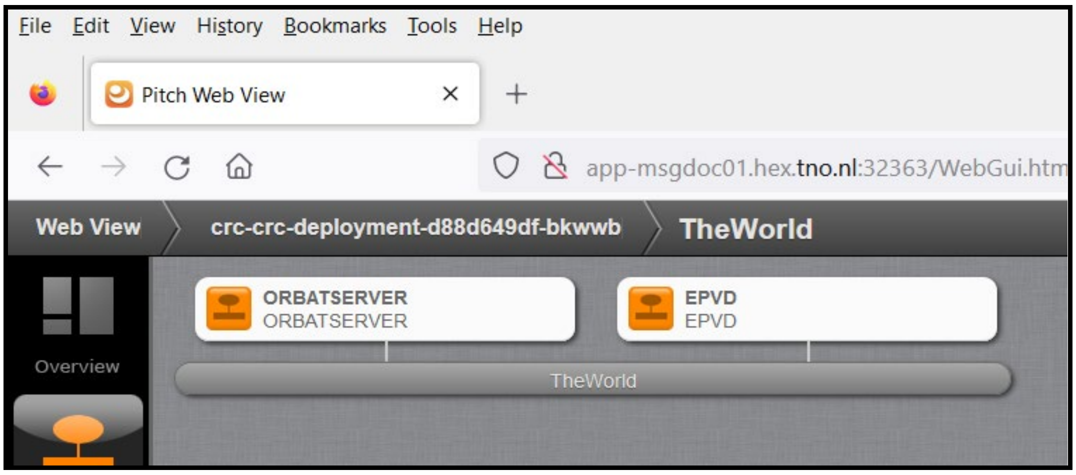
5) Press Install to deploy the application in the Kubernetes cluster.



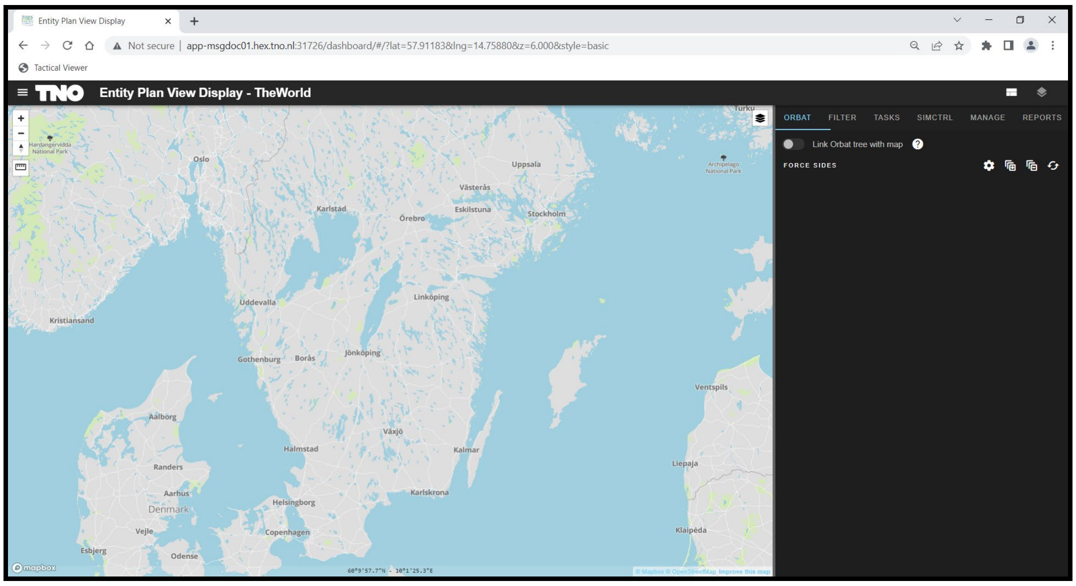
https://app-msgdoc99.hex.tno.nl/dashboard/c/c-m-8m2mzlfn/apps/catalog.cattle.io.app

```
Mon, Jul 17 2023 10:45:03 pm  helm install --generate-name=true --namespace=blessed --timeout=10m0s --values=/home/shell/helm/values-entity-plan-view-display-0.1.yaml --version=0.1 --wait=true /home/shell/helm/entity-
                              plan-view-display-0.1.tgz
Mon, Jul 17 2023 10:45:03 pm  creating 6 resource(s)
Mon, Jul 17 2023 10:45:03 pm  beginning wait for 6 resources with timeout of 10m0s
Mon, Jul 17 2023 10:45:03 pm  Deployment is not ready: blessed/epvdjsonxmlserver-deployment. 0 out of 1 expected pods are ready
Mon, Jul 17 2023 10:45:05 pm  Deployment is not ready: blessed/epvdorbat-deployment. 0 out of 1 expected pods are ready
Mon, Jul 17 2023 10:45:07 pm  NAME: entity-plan-view-display-0-1689626703
Mon, Jul 17 2023 10:45:07 pm  LAST DEPLOYED: Mon Jul 17 20:45:03 2023
Mon, Jul 17 2023 10:45:07 pm  NAMESPACE: blessed
Mon, Jul 17 2023 10:45:07 pm  STATUS: deployed
Mon, Jul 17 2023 10:45:07 pm  REVISION: 1
Mon, Jul 17 2023 10:45:07 pm  TEST SUITE: None
Mon, Jul 17 2023 10:45:07 pm
Mon, Jul 17 2023 10:45:07 pm  ---------------------------------------------------------------------
Mon, Jul 17 2023 10:45:07 pm  SUCCESS: helm install --generate-name=true --namespace=blessed --timeout=10m0s --values=/home/shell/helm/values-entity-plan-view-display-0.1.yaml --version=0.1 --wait=true /home/shell
                              /helm/entity-plan-view-display-0.1.tgz
Mon, Jul 17 2023 10:45:07 pm  ---------------------------------------------------------------------
```

6) Open the Pitch RTI Web UI to confirm that two HLA federates have joined the federation.


File Edit View History Bookmarks Tools Help
Pitch Web View
app-msgdoc01.hex.tno.nl:32363/WebGui.htm
Web View
crc-crc-deployment-d88d649df-bkwwb
TheWorld
Overview
ORBATSERVER
ORBATSERVER
EPVD
EPVD
TheWorld

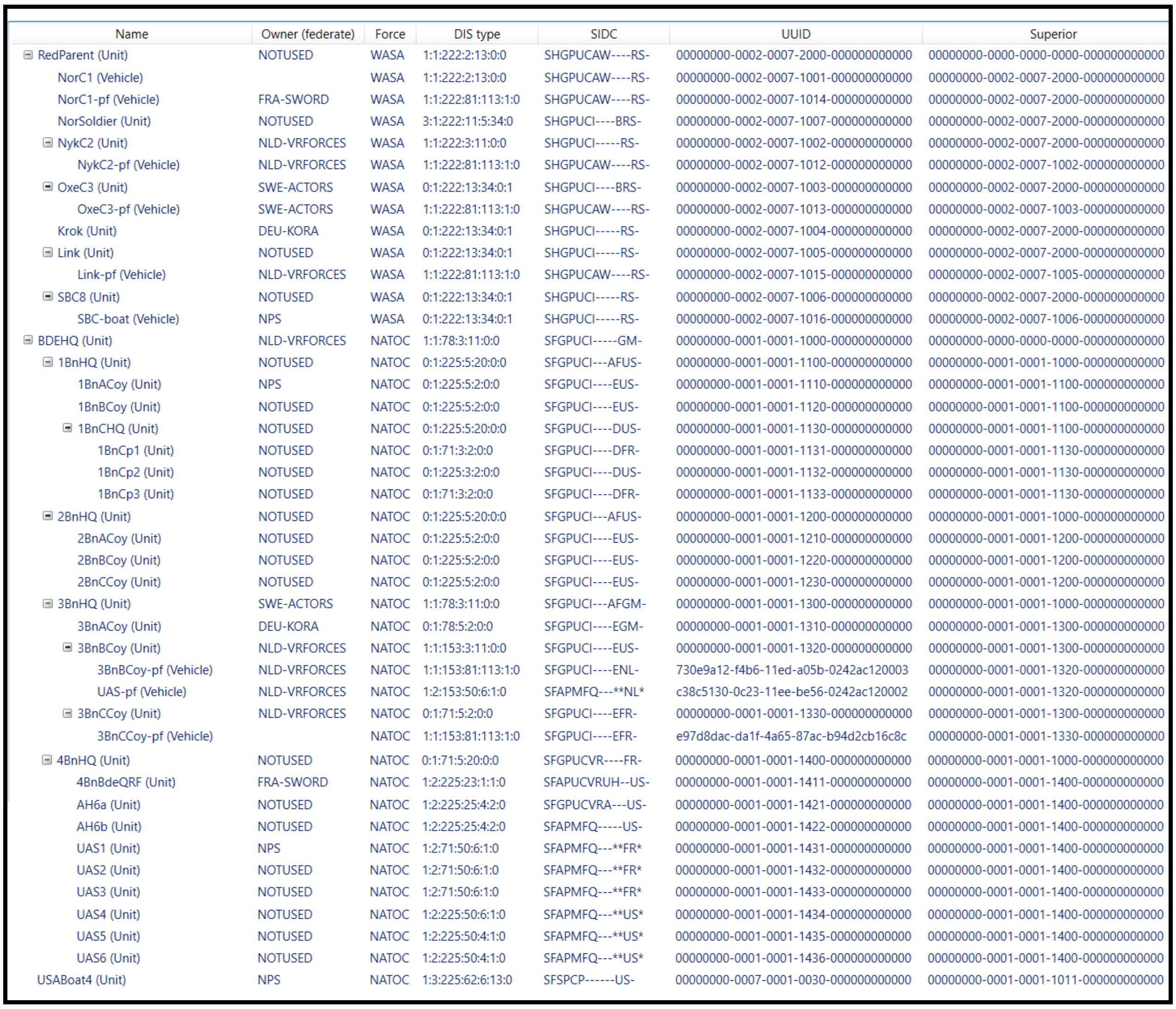
7) Open the EPVD UI to confirm that there are no entities and that there is no ORBAT data.


Entity Plan View Display
app-msgdoc01.hex.tno.nl:31726/dashboard/#/?lat=57.91183&lng=14.75880&z=6.00&style=basic
Tactical Viewer
TNO Entity Plan View Display - TheWorld
ORBAT
FILTER
TASKS
SIMCTRL
MANAGE
REPORTS
Link Orbat tree with map
FORCE SIDES

5.2 Initialize Simulation with a C2SIM LOX Initialization File

From the EPVD UI the simulation can be initialized with one or more MSDL or C2SIM LOX Initialization files. The MSDL or LOX data is published in the HLA federation as NETN-ORG object instances. The NETN Organization FOM Module (NETN-ORG) is a specification of how to represent organizations in a federated simulation. The representation is used in this small exercise for providing the initial state of simulated entities.

In this example the C2SIM LOX Initialization file “CWIX-2023 initializev9rev7.xml’ is used. A graphical view of the LOX initialisation data in the file is provided in Figure 20.



Name	Owner (federate)	Force	DIS type	SIDC	UUID	Superior
RedParent (Unit)	NOTUSED	WASA	1:1:222:2:13:0:0	SHGPUCAW----RS-	00000000-0002-0007-2000-000000000000	00000000-0000-0000-0000-000000000000
NorC1 (Vehicle)		WASA	1:1:222:2:13:0:0	SHGPUCAW----RS-	00000000-0002-0007-1001-000000000000	00000000-0002-0007-2000-000000000000
NorC1-pf (Vehicle)	FRA-SWORD	WASA	1:1:222:81:113:1:0	SHGPUCAW----RS-	00000000-0002-0007-1014-000000000000	00000000-0002-0007-2000-000000000000
NorSoldier (Unit)	NOTUSED	WASA	3:1:222:11:5:34:0	SHGPUCI----BRS-	00000000-0002-0007-1007-000000000000	00000000-0002-0007-2000-000000000000
NykC2 (Unit)	NLD-VRFORCES	WASA	1:1:222:3:11:0:0	SHGPUCI-----RS-	00000000-0002-0007-1002-000000000000	00000000-0002-0007-2000-000000000000
NykC2-pf (Vehicle)	NLD-VRFORCES	WASA	1:1:222:81:113:1:0	SHGPUCAW----RS-	00000000-0002-0007-1012-000000000000	00000000-0002-0007-1002-000000000000
OxeC3 (Unit)	SWE-ACTORS	WASA	0:1:222:13:34:0:1	SHGPUCI----BRS-	00000000-0002-0007-1003-000000000000	00000000-0002-0007-2000-000000000000
OxeC3-pf (Vehicle)	SWE-ACTORS	WASA	1:1:222:81:113:1:0	SHGPUCAW----RS-	00000000-0002-0007-1013-000000000000	00000000-0002-0007-1003-000000000000
Krok (Unit)	DEU-KORA	WASA	0:1:222:13:34:0:1	SHGPUCI-----RS-	00000000-0002-0007-1004-000000000000	00000000-0002-0007-2000-000000000000
Link (Unit)	NOTUSED	WASA	0:1:222:13:34:0:1	SHGPUCI-----RS-	00000000-0002-0007-1005-000000000000	00000000-0002-0007-2000-000000000000
Link-pf (Vehicle)	NLD-VRFORCES	WASA	1:1:222:81:113:1:0	SHGPUCAW----RS-	00000000-0002-0007-1015-000000000000	00000000-0002-0007-1005-000000000000
SBC8 (Unit)	NOTUSED	WASA	0:1:222:13:34:0:1	SHGPUCI-----RS-	00000000-0002-0007-1006-000000000000	00000000-0002-0007-2000-000000000000
SBC-boat (Vehicle)	NPS	WASA	0:1:222:13:34:0:1	SHGPUCI-----RS-	00000000-0002-0007-1016-000000000000	00000000-0002-0007-1006-000000000000
BDEHQ (Unit)	NLD-VRFORCES	NATOC	1:1:78:3:11:0:0	SFGPUCI-----GM-	00000000-0001-0001-1000-000000000000	00000000-0000-0000-0000-000000000000
1BnHQ (Unit)	NOTUSED	NATOC	0:1:225:5:20:0:0	SFGPUCI---AFUS-	00000000-0001-0001-1100-000000000000	00000000-0001-0001-1000-000000000000
1BnACoy (Unit)	NPS	NATOC	0:1:225:5:2:0:0	SFGPUCI----EUS-	00000000-0001-0001-1110-000000000000	00000000-0001-0001-1100-000000000000
1BnBCoy (Unit)	NOTUSED	NATOC	0:1:225:5:2:0:0	SFGPUCI----EUS-	00000000-0001-0001-1120-000000000000	00000000-0001-0001-1100-000000000000
1BnCHQ (Unit)	NOTUSED	NATOC	0:1:225:5:20:0:0	SFGPUCI----DUS-	00000000-0001-0001-1130-000000000000	00000000-0001-0001-1100-000000000000
1BnCp1 (Unit)	NOTUSED	NATOC	0:1:71:3:2:0:0	SFGPUCI----DFR-	00000000-0001-0001-1131-000000000000	00000000-0001-0001-1130-000000000000
1BnCp2 (Unit)	NOTUSED	NATOC	0:1:225:3:2:0:0	SFGPUCI----DUS-	00000000-0001-0001-1132-000000000000	00000000-0001-0001-1130-000000000000
1BnCp3 (Unit)	NOTUSED	NATOC	0:1:71:3:2:0:0	SFGPUCI----DFR-	00000000-0001-0001-1133-000000000000	00000000-0001-0001-1130-000000000000
2BnHQ (Unit)	NOTUSED	NATOC	0:1:225:5:20:0:0	SFGPUCI---AFUS-	00000000-0001-0001-1200-000000000000	00000000-0001-0001-1000-000000000000
2BnACoy (Unit)	NOTUSED	NATOC	0:1:225:5:2:0:0	SFGPUCI----EUS-	00000000-0001-0001-1210-000000000000	00000000-0001-0001-1200-000000000000
2BnBCoy (Unit)	NOTUSED	NATOC	0:1:225:5:2:0:0	SFGPUCI----EUS-	00000000-0001-0001-1220-000000000000	00000000-0001-0001-1200-000000000000
2BnCCoy (Unit)	NOTUSED	NATOC	0:1:225:5:2:0:0	SFGPUCI----EUS-	00000000-0001-0001-1230-000000000000	00000000-0001-0001-1200-000000000000
3BnHQ (Unit)	SWE-ACTORS	NATOC	1:1:78:3:11:0:0	SFGPUCI---AFGM-	00000000-0001-0001-1300-000000000000	00000000-0001-0001-1000-000000000000
3BnACoy (Unit)	DEU-KORA	NATOC	0:1:78:5:2:0:0	SFGPUCI----EGM-	00000000-0001-0001-1310-000000000000	00000000-0001-0001-1300-000000000000
3BnBCoy (Unit)	NLD-VRFORCES	NATOC	1:1:153:3:11:0:0	SFGPUCI----EUS-	00000000-0001-0001-1320-000000000000	00000000-0001-0001-1300-000000000000
3BnBCoy-pf (Vehicle)	NLD-VRFORCES	NATOC	1:1:153:81:113:1:0	SFGPUCI----ENL-	730e9a12-f4b6-11ed-a05b-0242ac120003	00000000-0001-0001-1320-000000000000
UAS-pf (Vehicle)	NLD-VRFORCES	NATOC	1:2:153:50:6:1:0	SFAPMFQ---**NL*	c38c5130-0c23-11ee-be56-0242ac120002	00000000-0001-0001-1320-000000000000
3BnCCoy (Unit)	NLD-VRFORCES	NATOC	0:1:71:5:2:0:0	SFGPUCI----EFR-	00000000-0001-0001-1330-000000000000	00000000-0001-0001-1300-000000000000
3BnCCoy-pf (Vehicle)		NATOC	1:1:153:81:113:1:0	SFGPUCI----EFR-	e97d8dac-da1f-4a65-87ac-b94d2cb16c8c	00000000-0001-0001-1330-000000000000
4BnHQ (Unit)	NOTUSED	NATOC	0:1:71:5:20:0:0	SFGPUCVR----FR-	00000000-0001-0001-1400-000000000000	00000000-0001-0001-1000-000000000000
4BnBdeQRF (Unit)	FRA-SWORD	NATOC	1:2:225:23:1:1:0	SFAPUCVRUH--US-	00000000-0001-0001-1411-000000000000	00000000-0001-0001-1400-000000000000
AH6a (Unit)	NOTUSED	NATOC	1:2:225:25:4:2:0	SFGPUCVRA---US-	00000000-0001-0001-1421-000000000000	00000000-0001-0001-1400-000000000000
AH6b (Unit)	NOTUSED	NATOC	1:2:225:25:4:2:0	SFAPMFQ-----US-	00000000-0001-0001-1422-000000000000	00000000-0001-0001-1400-000000000000
UAS1 (Unit)	NPS	NATOC	1:2:71:50:6:1:0	SFAPMFQ---**FR*	00000000-0001-0001-1431-000000000000	00000000-0001-0001-1400-000000000000
UAS2 (Unit)	NOTUSED	NATOC	1:2:71:50:6:1:0	SFAPMFQ---**FR*	00000000-0001-0001-1432-000000000000	00000000-0001-0001-1400-000000000000
UAS3 (Unit)	NOTUSED	NATOC	1:2:71:50:6:1:0	SFAPMFQ---**FR*	00000000-0001-0001-1433-000000000000	00000000-0001-0001-1400-000000000000
UAS4 (Unit)	NOTUSED	NATOC	1:2:225:50:6:1:0	SFAPMFQ---**US*	00000000-0001-0001-1434-000000000000	00000000-0001-0001-1400-000000000000
UAS5 (Unit)	NOTUSED	NATOC	1:2:225:50:4:1:0	SFAPMFQ---**US*	00000000-0001-0001-1435-000000000000	00000000-0001-0001-1400-000000000000
UAS6 (Unit)	NOTUSED	NATOC	1:2:225:50:4:1:0	SFAPMFQ---**US*	00000000-0001-0001-1436-000000000000	00000000-0001-0001-1400-000000000000
USABoat4 (Unit)	NPS	NATOC	1:3:225:62:6:13:0	SFSPCP------US-	00000000-0007-0001-0030-000000000000	00000000-0001-0001-1011-000000000000

Figure 20: C2SIM LOX Initialisation data.

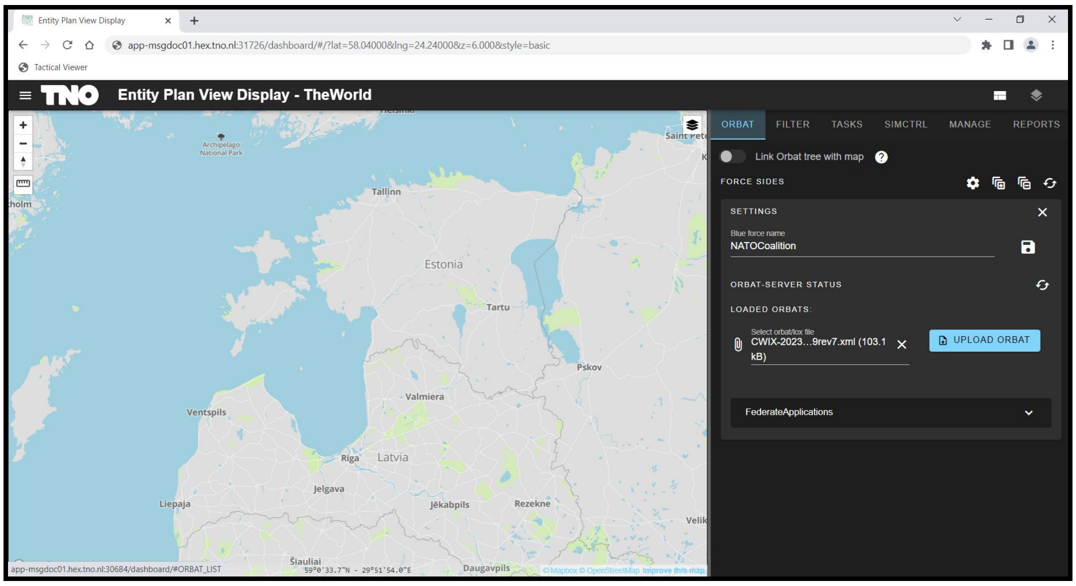
This file defines three force sides:

- NATOCoalition (friendly force side).
- WASA (enemy force side).
- Neutral (neutral force side, not shown in figure since there are no elements defined for the side).

Each force side holds several types of entities, organized in a hierarchical structure. The file also includes information on modeling responsibilities. That is, what simulation application is responsible for the modelling of the unit or equipment item defined in the C2SIM LOX Initialization file.
The tasks for initialization are as follows:

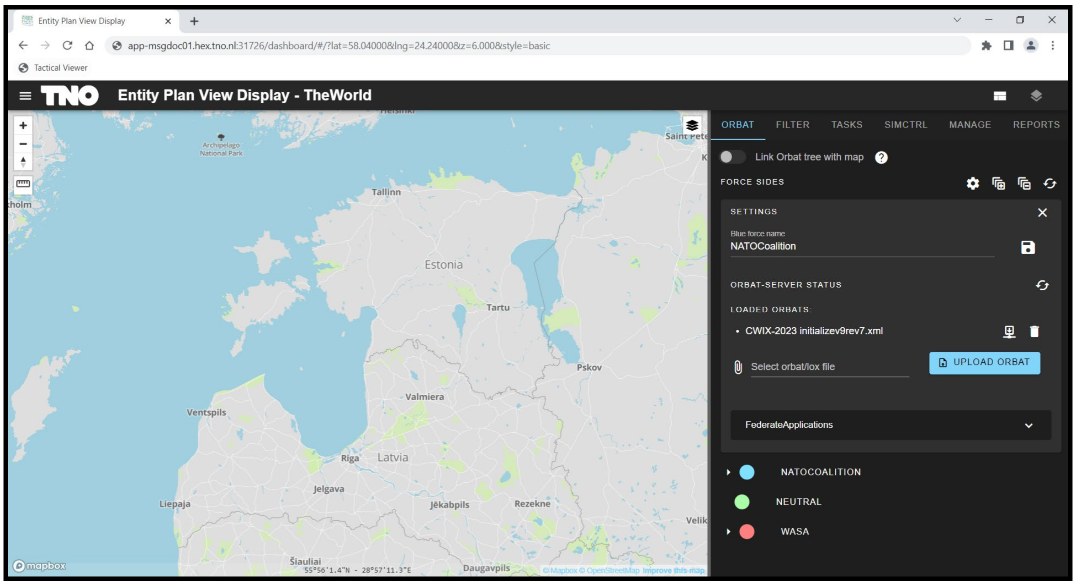
1) From the EPVD UI, set "NATOCoalition" as the name of the force that is considered as the friendly force, and browse for the file named "CWIX-2023 initializev9rev7.xml".

   The reference force can be updated in the EPVD at any time, but here the force name is set just before the simulation initialization data is uploaded.



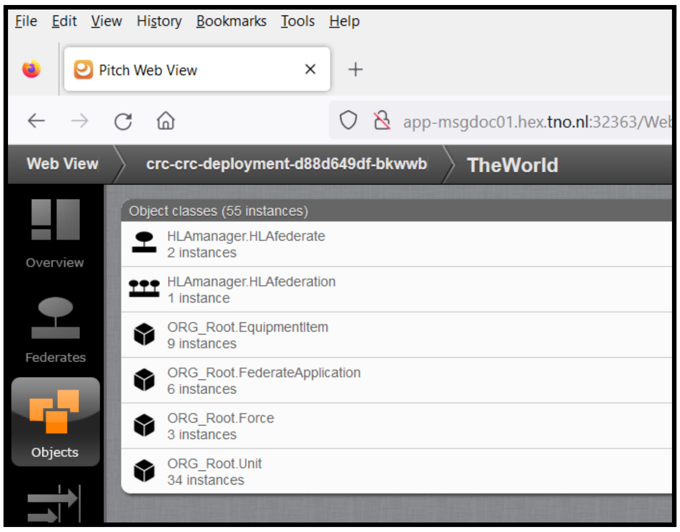
2) Press UPLOAD ORBAT to initialize the simulation with the given data.

The initialization data is published as HLA NETN-ORG object instances in the federation. The force sides are shown in the ORBAT panel of the EPVD UI. At this point the map panel does not show any entities since no CGFs have joined the federation yet.



3) Open the Pitch RTI Web UI to view what HLA NETN-ORG object instances have been created.

   In this example: 3 Force, 9 EquipmentItem, 34 Unit, and 6 FederateApplication object instances. The modeling responsibility of the equipment items and units are allocated to the 6 federate applications. The details are described by the NETN-ORG object instances.


File Edit View History Bookmarks Tools Help
Pitch Web View
app-msgdoc01.hex.tno.nl:32363/Wel
Web View
crc-crc-deployment-d88d649df-bkwwb
TheWorld
Overview
Federates
Objects

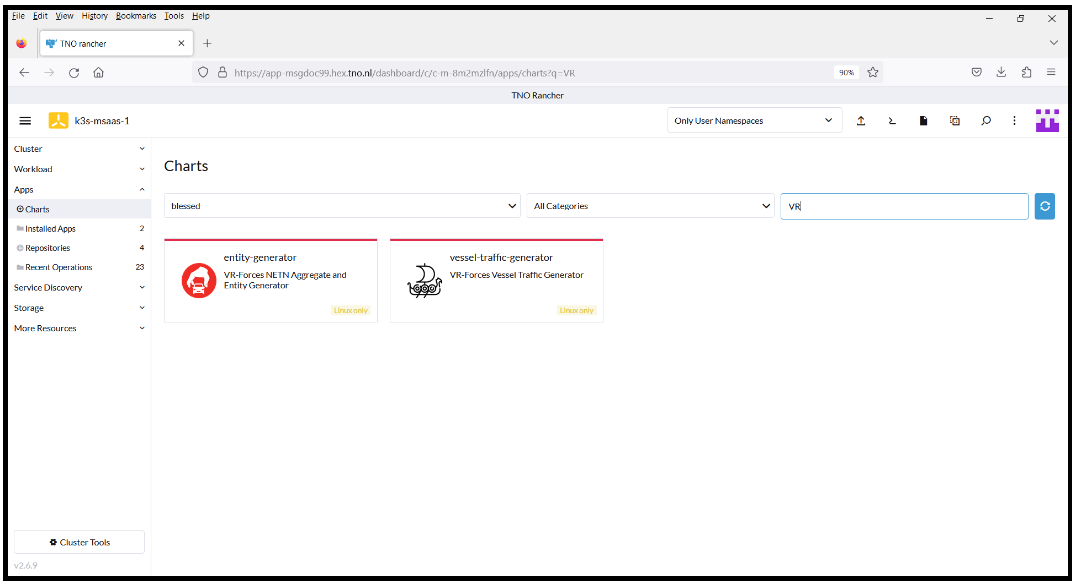
Object classes (55 instances)
HLAmanager.HLAfederate
2 instances
HLAmanager.HLAfederation
1 instance
ORG_Root.EquipmentItem
9 instances
ORG_Root.FederateApplication
6 instances
ORG_Root.Force
3 instances
ORG_Root.Unit
34 instances

5.3 START VTMAK VR FORCES

VR Forces is a Computer Generated Forces application from VTMaK [12]. VR Forces includes many simulation models for battlefield units and systems at both entity and aggregate level. VR Forces consists of two parts, namely a front-end UI and a back-end simulation engine. Both parts are HLA federates and can be enriched with plugins, providing the ability to add user-defined functionality.

In this step the VR Forces back-end is started, which has been enriched with several plugins to make it "NETN-FOM" compliant.

1) Browse the catalog for the application named VR Forces Entity Generator.



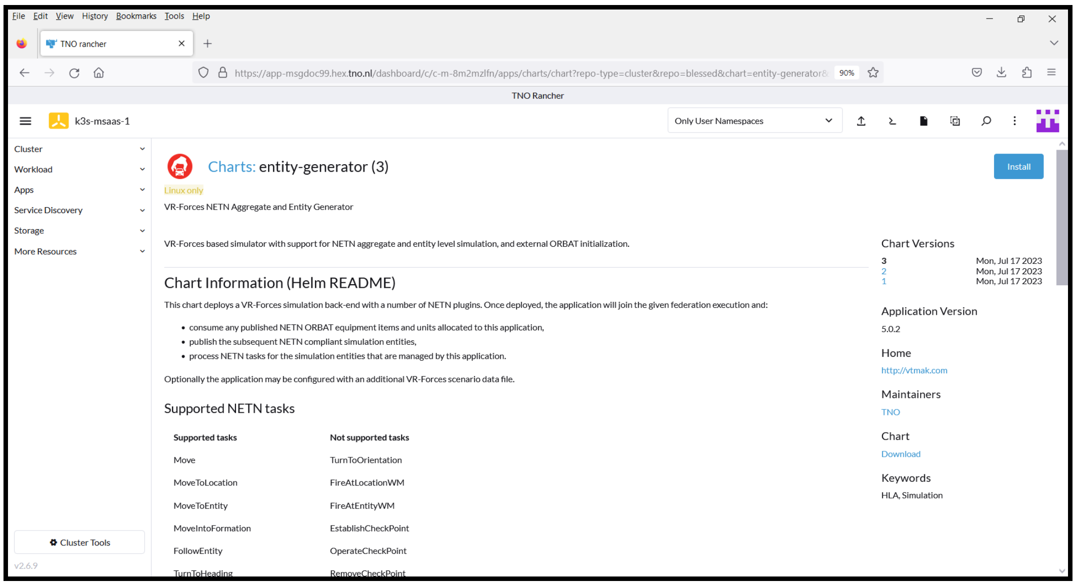
Charts

blessed | All Categories | VR

entity-generator
VR-Forces NETN Aggregate and Entity Generator
Linux only

vessel-traffic-generator
VR-Forces Vessel Traffic Generator
Linux only

2) Open the Helm Chart from the catalog.


File Edit View History Bookmarks Tools Help
TNO rancher
https://app-msgdoc99.hex.tno.nl/dashboard/c/c-m-8m2mzlfn/apps/charts/chart?repo-type=cluster&repo=blessed&chart=entity-generator&
90%
TNO Rancher
k3s-msaas-1
Only User Namespaces
Cluster
Workload
Apps
Service Discovery
Storage
More Resources
Cluster Tools
v2.6.9
Install

Charts: entity-generator (3)

Linux only

VR-Forces NETN Aggregate and Entity Generator

VR-Forces based simulator with support for NETN aggregate and entity level simulation, and external ORBAT initialization.

Chart Information (Helm README)

This chart deploys a VR-Forces simulation back-end with a number of NETN plugins. Once deployed, the application will join the given federation execution and:

- consume any published NETN ORBAT equipment items and units allocated to this application,
- publish the subsequent NETN compliant simulation entities,
- process NETN tasks for the simulation entities that are managed by this application.

Optionally the application may be configured with an additional VR-Forces scenario data file.

Supported NETN tasks

Supported tasks	Not supported tasks
Move	TurnToOrientation
MoveToLocation	FireAtLocationWM
MoveToEntity	FireAtEntityWM
MoveIntoFormation	EstablishCheckPoint
FollowEntity	OperateCheckPoint
TurnToHeading	RemoveCheckPoint

Chart Versions

3	Mon, Jul 17 2023
2	Mon, Jul 17 2023
1	Mon, Jul 17 2023

Application Version
5.0.2

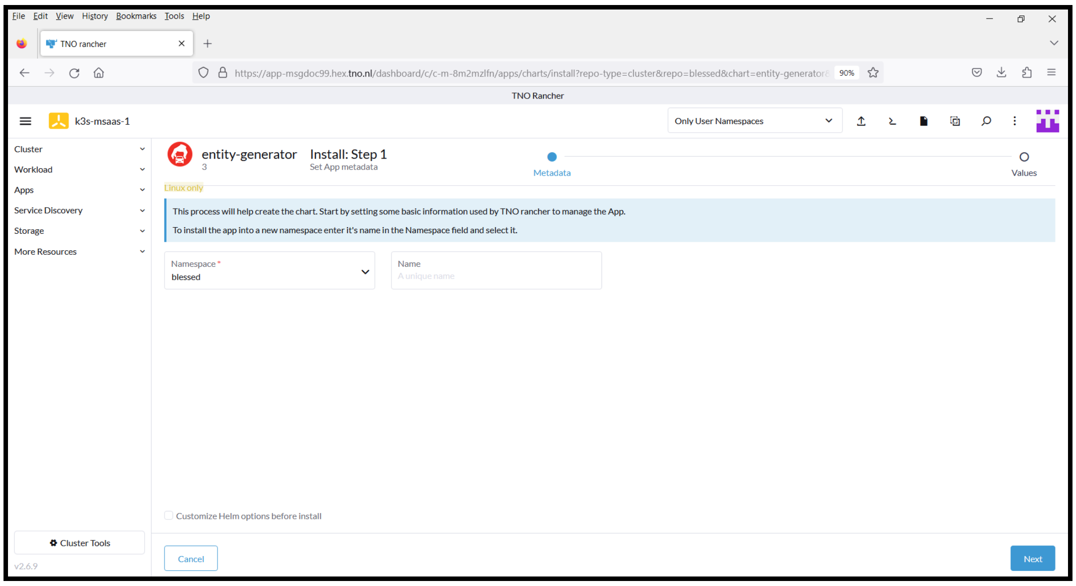
Home
http://vtmak.com

Maintainers
TNO

Chart
Download

Keywords
HLA, Simulation

3) Provide the Kubernetes namespace and deployment name for the application (step 1).


File Edit View History Bookmarks Tools Help
TNO rancher
https://app-msgdoc99.hex.tno.nl/dashboard/c/c-m-8m2mzlfn/apps/charts/install?repo-type=cluster&repo=blessed&chart=entity-generator&
90%
TNO Rancher
k3s-msaas-1
Only User Namespaces
Cluster
Workload
Apps
Service Discovery
Storage
More Resources
Cluster Tools
v2.6.9
Metadata
Values
Cancel
Next

entity-generator
3

Install: Step 1
Set App metadata

Linux only

This process will help create the chart. Start by setting some basic information used by TNO rancher to manage the App.

To install the app into a new namespace enter it's name in the Namespace field and select it.

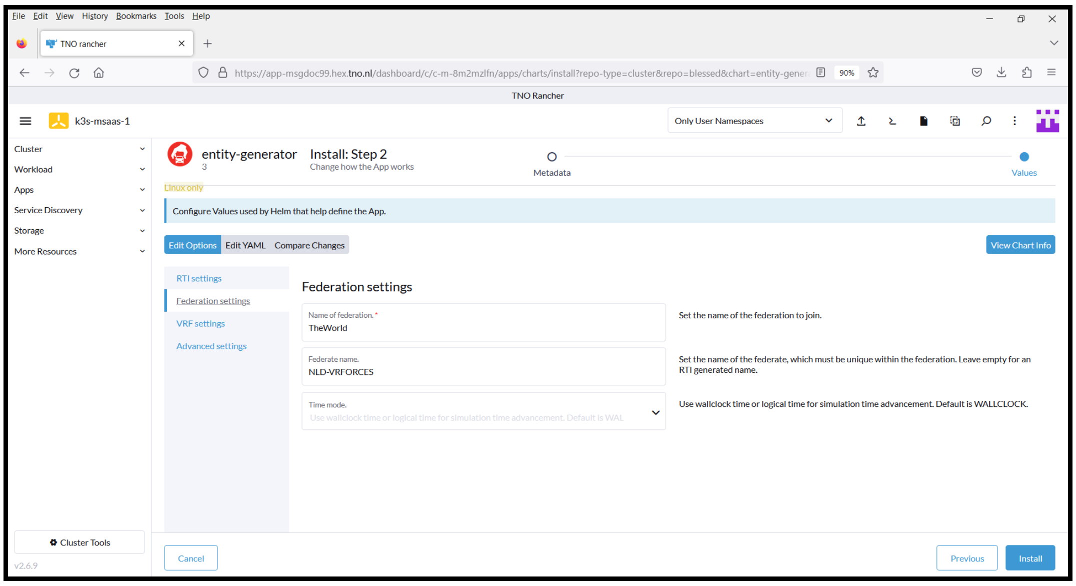
Namespace *
blessed

Name
A unique name

Customize Helm options before install

4) Provide application related information (step 2).

   In this task the name under which VR Forces must join the federation is set to "NLD-VRFORCES". VR Forces will look for published ORBAT elements that are allocated to this federate name and subsequently will create the entities for these elements.



https://app-msgdoc99.hex.tno.nl/dashboard/c/c-m-8m2mzlfn/apps/charts/install?repo-type=cluster&repo=blessed&chart=entity-gener

k3s-msaas-1

entity-generator 3 — Install: Step 2 — Change how the App works

Configure Values used by Helm that help define the App.

Federation settings

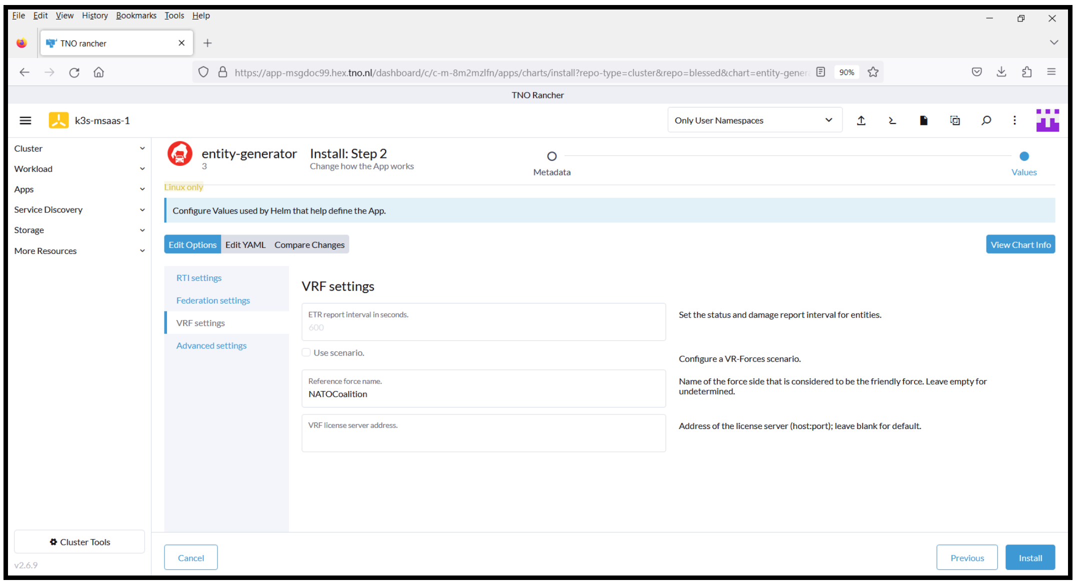
Name of federation.* TheWorld — Set the name of the federation to join.

Federate name. NLD-VRFORCES — Set the name of the federate, which must be unique within the federation. Leave empty for an RTI generated name.

Time mode. Use wallclock time or logical time for simulation time advancement. Default is WAL — Use wallclock time or logical time for simulation time advancement. Default is WALLCLOCK.

5) Provide application related information (step 2).

   In this task the name of the reference force is set to NATOCoalition. The reference force is the force side considered to be the friendly side (see also EPVD).



VRF settings

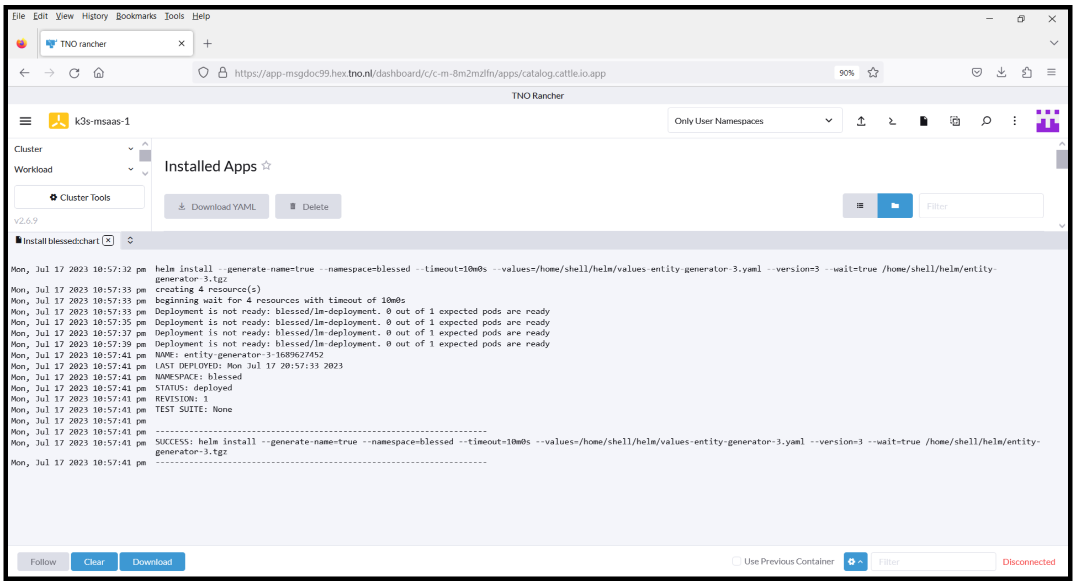
ETR report interval in seconds. 600 — Set the status and damage report interval for entities.

Use scenario. — Configure a VR-Forces scenario.

Reference force name. NATOCoalition — Name of the force side that is considered to be the friendly force. Leave empty for undetermined.

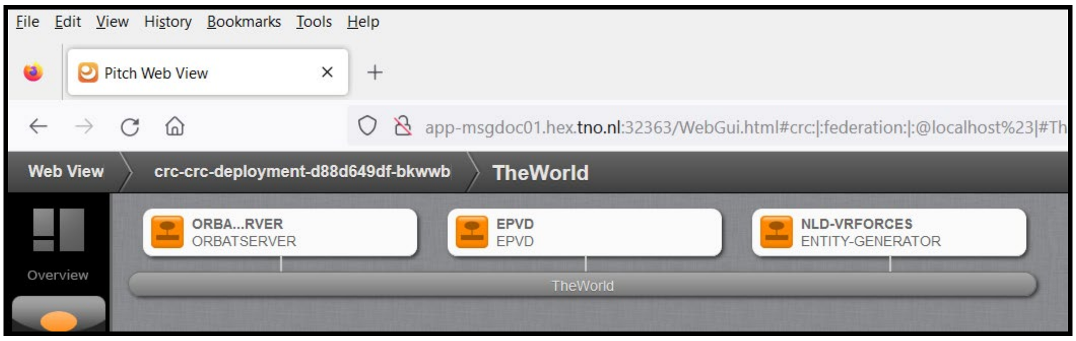
VRF license server address. — Address of the license server (host:port); leave blank for default.

6) Press Install to deploy the application in the Kubernetes cluster.

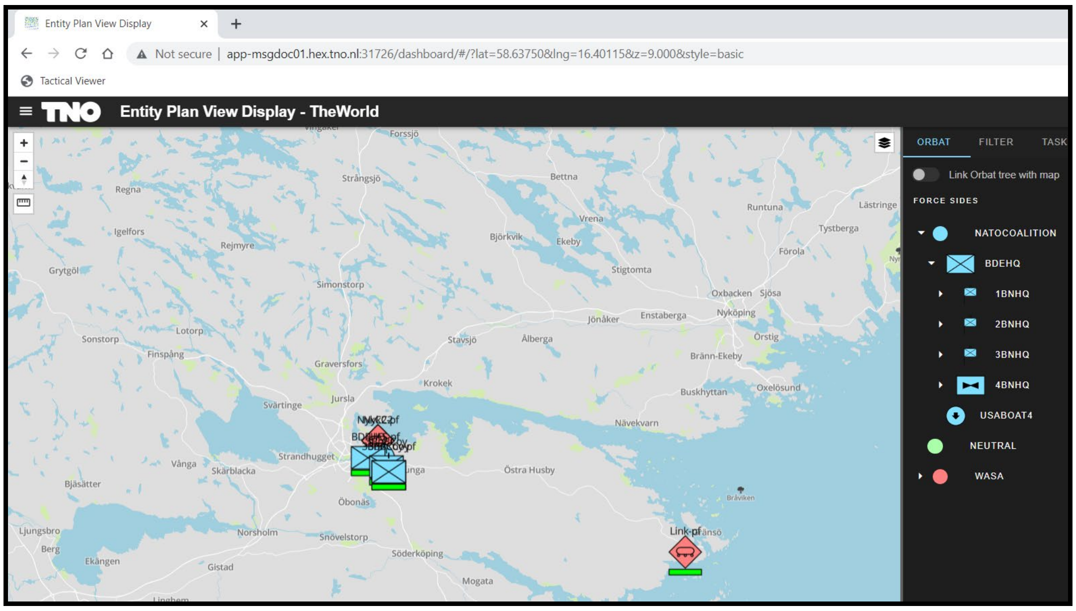


```
Mon, Jul 17 2023 10:57:32 pm  helm install --generate-name=true --namespace=blessed --timeout=10m0s --values=/home/shell/helm/values-entity-generator-3.yaml --version=3 --wait=true /home/shell/helm/entity-
                              generator-3.tgz
Mon, Jul 17 2023 10:57:33 pm  creating 4 resource(s)
Mon, Jul 17 2023 10:57:33 pm  beginning wait for 4 resources with timeout of 10m0s
Mon, Jul 17 2023 10:57:33 pm  Deployment is not ready: blessed/lm-deployment. 0 out of 1 expected pods are ready
Mon, Jul 17 2023 10:57:35 pm  Deployment is not ready: blessed/lm-deployment. 0 out of 1 expected pods are ready
Mon, Jul 17 2023 10:57:37 pm  Deployment is not ready: blessed/lm-deployment. 0 out of 1 expected pods are ready
Mon, Jul 17 2023 10:57:39 pm  Deployment is not ready: blessed/lm-deployment. 0 out of 1 expected pods are ready
Mon, Jul 17 2023 10:57:41 pm  NAME: entity-generator-3-1689627452
Mon, Jul 17 2023 10:57:41 pm  LAST DEPLOYED: Mon Jul 17 20:57:33 2023
Mon, Jul 17 2023 10:57:41 pm  NAMESPACE: blessed
Mon, Jul 17 2023 10:57:41 pm  STATUS: deployed
Mon, Jul 17 2023 10:57:41 pm  REVISION: 1
Mon, Jul 17 2023 10:57:41 pm  TEST SUITE: None
Mon, Jul 17 2023 10:57:41 pm
Mon, Jul 17 2023 10:57:41 pm  ---------------------------------------------------------------------
Mon, Jul 17 2023 10:57:41 pm  SUCCESS: helm install --generate-name=true --namespace=blessed --timeout=10m0s --values=/home/shell/helm/values-entity-generator-3.yaml --version=3 --wait=true /home/shell/helm/entity-
                              generator-3.tgz
Mon, Jul 17 2023 10:57:41 pm  ---------------------------------------------------------------------
```

7) Open the Pitch RTI Web UI to confirm that three HLA federates have now joined the federation.


File Edit View History Bookmarks Tools Help
Pitch Web View
app-msgdoc01.hex.tno.nl:32363/WebGui.html#crc:|:federation:|:@localhost%23|#Th
Web View
crc-crc-deployment-d88d649df-bkwwb
TheWorld
Overview
ORBA...RVER
ORBATSERVER
EPVD
EPVD
NLD-VRFORCES
ENTITY-GENERATOR
TheWorld

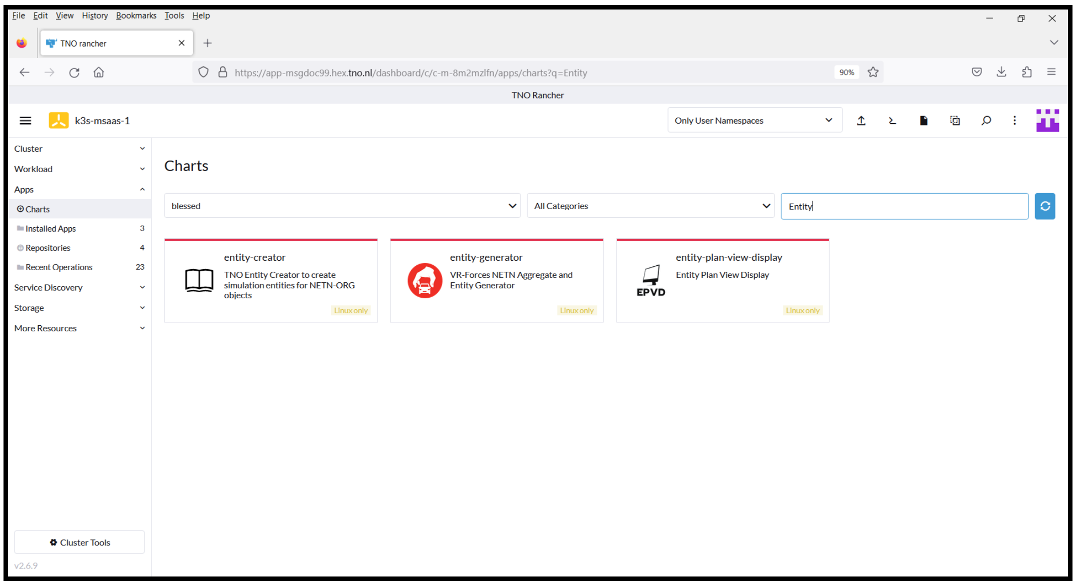
8) Open the EPVD UI to confirm that the entities allocated to VR Forces have been created.



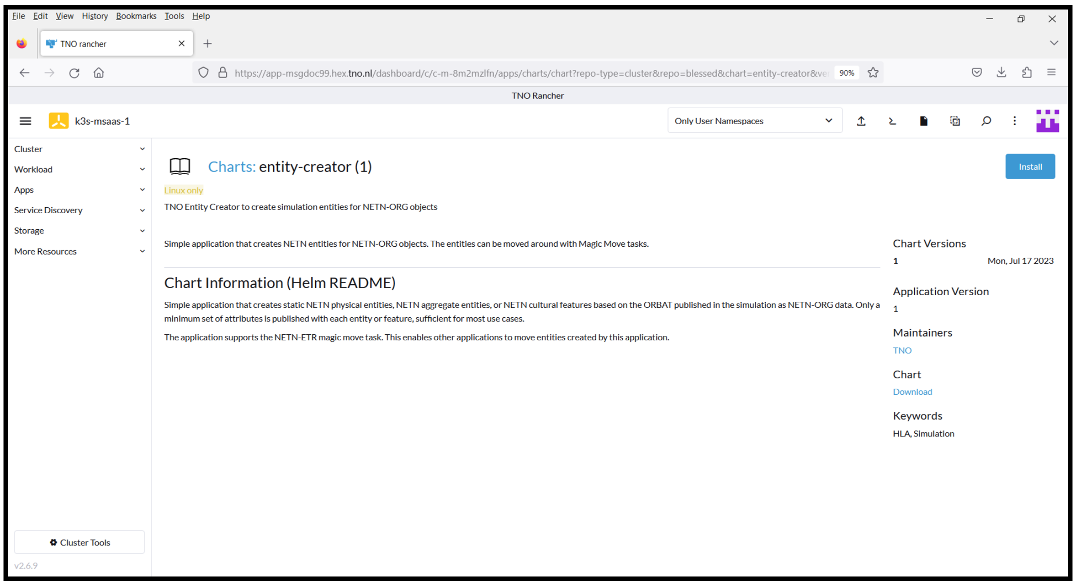
5.4 START TNO ENTITY CREATOR

The TNO Entity Creator is a simple application that merely creates entities for the assigned ORBAT elements, and can accept NETN Magic Move tasks to reposition these entities. In this step the Entity Creator is started. The tasks are similar to VR Forces, with the difference that the application shall join the federation using the name "NPS".

1) Browse the catalog for the application named Entity Creator.



2) Open the Helm Chart from the catalog.


File Edit View History Bookmarks Tools Help
TNO rancher
https://app-msgdoc99.hex.tno.nl/dashboard/c/c-m-8m2mzlfn/apps/charts/chart?repo-type=cluster&repo=blessed&chart=entity-creator&ve
90%
TNO Rancher
k3s-msaas-1
Only User Namespaces
Cluster
Workload
Apps
Service Discovery
Storage
More Resources
Cluster Tools
v2.6.9
Install

Charts: entity-creator (1)

Linux only

TNO Entity Creator to create simulation entities for NETN-ORG objects

Simple application that creates NETN entities for NETN-ORG objects. The entities can be moved around with Magic Move tasks.

Chart Information (Helm README)

Simple application that creates static NETN physical entities, NETN aggregate entities, or NETN cultural features based on the ORBAT published in the simulation as NETN-ORG data. Only a minimum set of attributes is published with each entity or feature, sufficient for most use cases.

The application supports the NETN-ETR magic move task. This enables other applications to move entities created by this application.

Chart Versions
1 Mon, Jul 17 2023

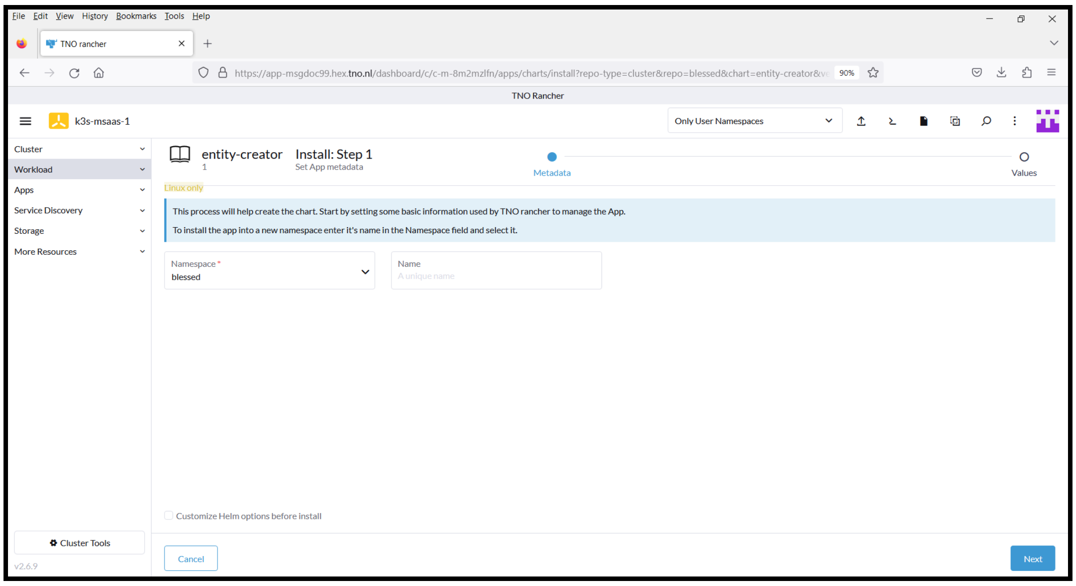
Application Version
1

Maintainers
TNO

Chart
Download

Keywords
HLA, Simulation

3) Provide the Kubernetes namespace and deployment name for the application (step 1).


File Edit View History Bookmarks Tools Help
TNO rancher
https://app-msgdoc99.hex.tno.nl/dashboard/c/c-m-8m2mzlfn/apps/charts/install?repo-type=cluster&repo=blessed&chart=entity-creator&ve
90%
TNO Rancher
k3s-msaas-1
Only User Namespaces
Cluster
Workload
Apps
Service Discovery
Storage
More Resources
Cluster Tools
v2.6.9
Metadata
Values
Cancel
Next

entity-creator 1 Install: Step 1
Set App metadata

Linux only

This process will help create the chart. Start by setting some basic information used by TNO rancher to manage the App.

To install the app into a new namespace enter it's name in the Namespace field and select it.

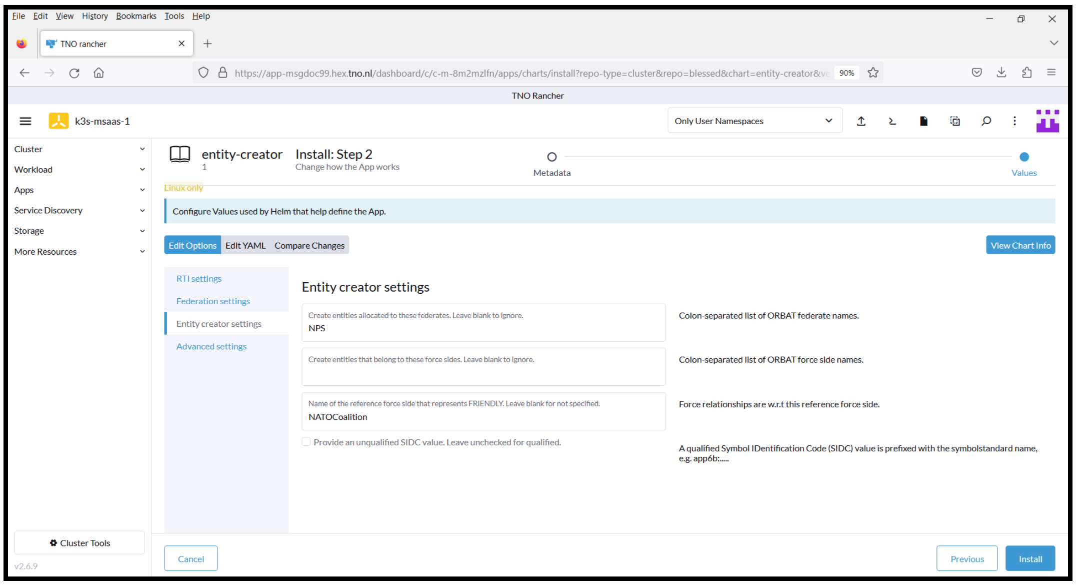
Namespace * blessed

Name A unique name

Customize Helm options before install

4) Provide application related information (step 2).

   In this task the federate name for which the Entity Creator must create entities is set to “NPS”. The Entity Creator will look for published ORBAT elements that are allocated to this federate name and subsequently create the entities for these elements. In addition, the name of the reference force is set to “NATOCoalition”. The Entity Creator itself joins under the federate type name ENTITY-CREATOR.

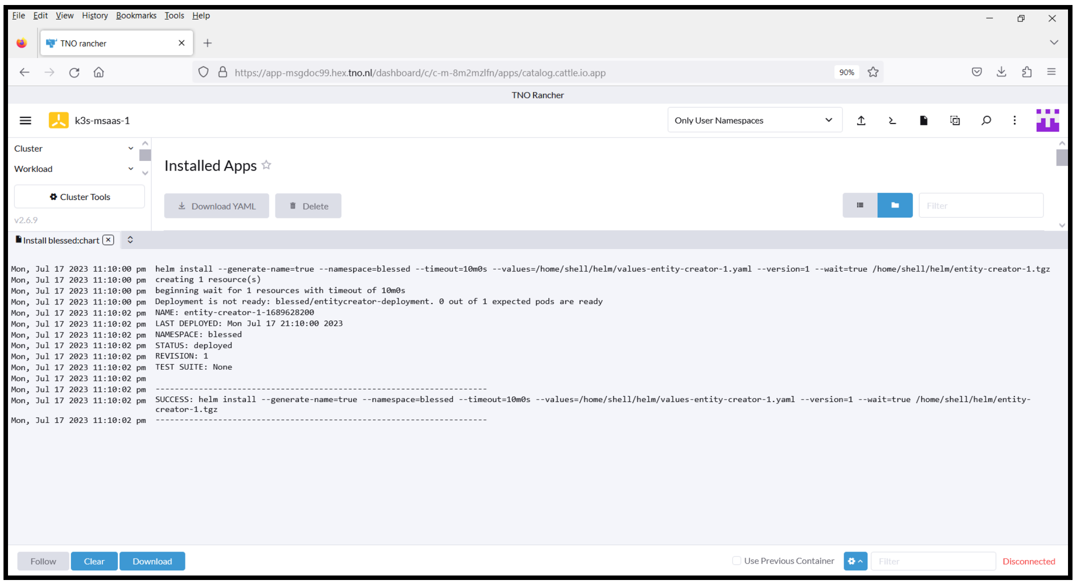


Configure Values used by Helm that help define the App.

Entity creator settings

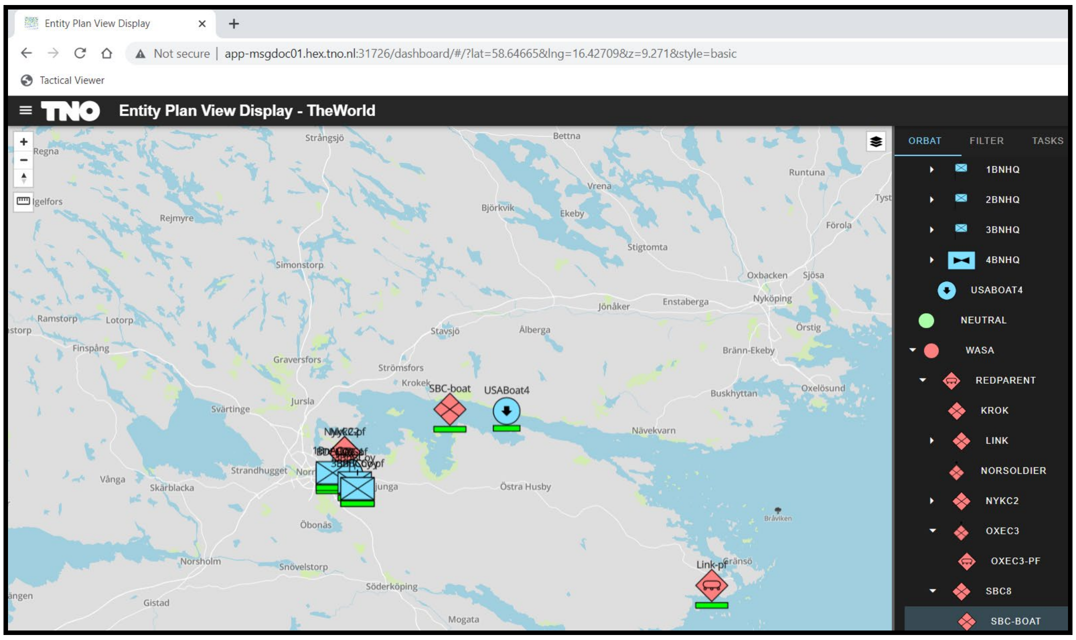
Create entities allocated to these federates. Leave blank to ignore. NPS	Colon-separated list of ORBAT federate names.
Create entities that belong to these force sides. Leave blank to ignore.	Colon-separated list of ORBAT force side names.
Name of the reference force side that represents FRIENDLY. Leave blank for not specified. NATOCoalition	Force relationships are w.r.t this reference force side.
☐ Provide an unqualified SIDC value. Leave unchecked for qualified.	A qualified Symbol IDentification Code (SIDC) value is prefixed with the symbolstandard name, e.g. app6b:.....

5) Press Install to deploy the application in the Kubernetes cluster.

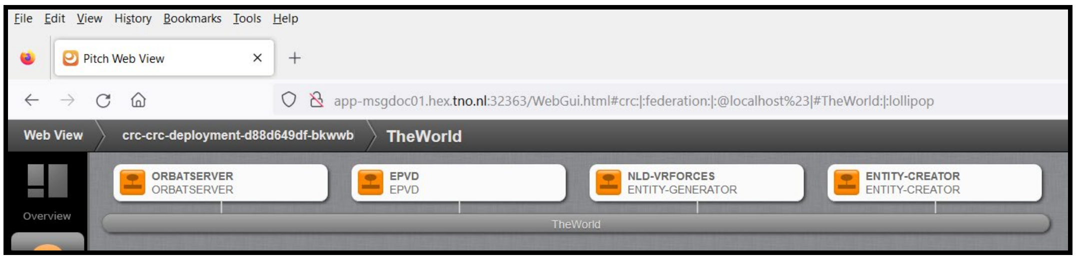


```
Mon, Jul 17 2023 11:10:00 pm  helm install --generate-name=true --namespace=blessed --timeout=10m0s --values=/home/shell/helm/values-entity-creator-1.yaml --version=1 --wait=true /home/shell/helm/entity-creator-1.tgz
Mon, Jul 17 2023 11:10:00 pm  creating 1 resource(s)
Mon, Jul 17 2023 11:10:00 pm  beginning wait for 1 resources with timeout of 10m0s
Mon, Jul 17 2023 11:10:00 pm  Deployment is not ready: blessed/entitycreator-deployment. 0 out of 1 expected pods are ready
Mon, Jul 17 2023 11:10:02 pm  NAME: entity-creator-1-1689628200
Mon, Jul 17 2023 11:10:02 pm  LAST DEPLOYED: Mon Jul 17 21:10:00 2023
Mon, Jul 17 2023 11:10:02 pm  NAMESPACE: blessed
Mon, Jul 17 2023 11:10:02 pm  STATUS: deployed
Mon, Jul 17 2023 11:10:02 pm  REVISION: 1
Mon, Jul 17 2023 11:10:02 pm  TEST SUITE: None
Mon, Jul 17 2023 11:10:02 pm
Mon, Jul 17 2023 11:10:02 pm  ---------------------------------------------------------------------
Mon, Jul 17 2023 11:10:02 pm  SUCCESS: helm install --generate-name=true --namespace=blessed --timeout=10m0s --values=/home/shell/helm/values-entity-creator-1.yaml --version=1 --wait=true /home/shell/helm/entity-
                              creator-1.tgz
Mon, Jul 17 2023 11:10:02 pm  ---------------------------------------------------------------------
```

6) Open the EPVD UI to confirm that the entities allocated to the Entity Creator have been created. For example, SBC-boat and USABoat4 have been created.



7) Open the Pitch RTI Web UI to confirm that four HLA federates have now joined the federation.


File Edit View History Bookmarks Tools Help
Pitch Web View
app-msgdoc01.hex.tno.nl:32363/WebGui.html#crc:|:federation:|:@localhost%23|#TheWorld:|:lollipop
Web View
crc-crc-deployment-d88d649df-bkwwb
TheWorld
Overview
ORBATSERVER
ORBATSERVER
EPVD
EPVD
NLD-VRFORCES
ENTITY-GENERATOR
ENTITY-CREATOR
ENTITY-CREATOR
TheWorld

5.5 Issue NETN MoveToLocation Task

At this step two CGFs and a simulation control application are deployed in the Kubernetes cluster. The simulation is initialized and the two CGFs have created their allocated entities. Both CGFs are NETN-ETR (Entity Tasking and Reporting) FOM compliant and can handle tasks defined in this FOM module. The NETN-ETR FOM module specifies common low-level tasks that can easily be interpreted and executed. The FOM module also defines a set of reports to provide status or management information. In this small exercise we use the MoveToLocation task as example.

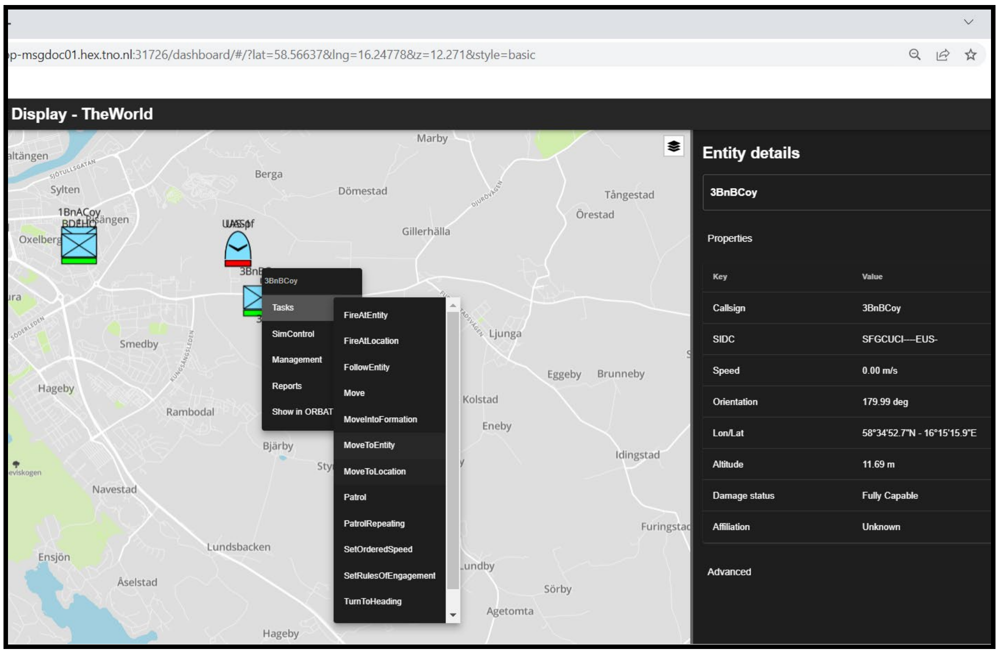
From the EPVD UI the entity named “3BnBcoy” is given the task to move to a certain location. While moving to the target position, the entity provides position reports. Note that for the EPVD UI it is transparent which CGF manages the entity. The CGF that has the modeling responsibility for the entity is required to consume and process the NETN MoveToLocation task.

The tasks are as follows:

1) From the EPVD UI right-click on the entity 3BnBcoy to open the menu with available tasks.

   The menu is constructed dynamically. The EPVD queries the CGF what tasks a discovered entity supports and constructs the menu from the returned information. Thus, another entity may have a different menu with tasks.

   The EPVD is agnostic with regards to the owner of an entity. From the C2SIM LOX Initialization file we happen to know that VR Forces has the modeling responsibility for 3BnBcoy.



Entity details

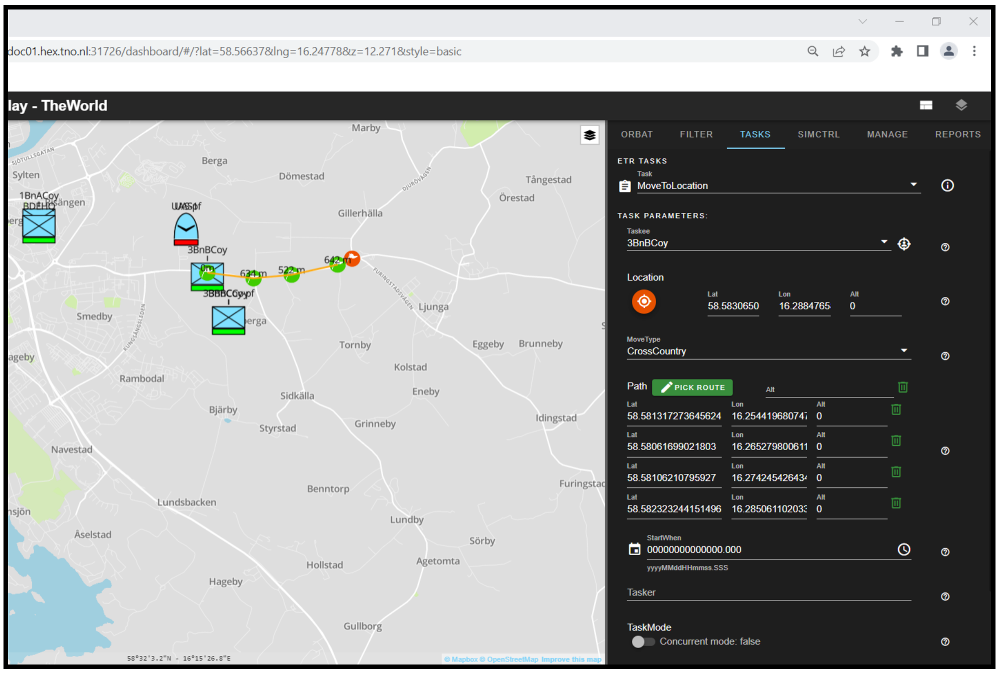
3BnBCoy

Properties

Key	Value
Callsign	3BnBCoy
SIDC	SFGCUCI----EUS-
Speed	0.00 m/s
Orientation	179.99 deg
Lon/Lat	58°34'52.7"N - 16°15'15.9"E
Altitude	11.69 m
Damage status	Fully Capable
Affiliation	Unknown

Advanced

2) Select the MoveToLocation task and enter waypoints for the route to be taken.


doc01.hex.tno.nl:31726/dashboard/#/?lat=58.56637&lng=16.24778&z=12.271&style=basic
lay - TheWorld
ORBAT
FILTER
TASKS
SIMCTRL
MANAGE
REPORTS
PICK ROUTE

ETR TASKS

Task: MoveToLocation

TASK PARAMETERS:

Taskee: 3BnBCoy

Location: Lat 58.5830650 Lon 16.2884765 Alt 0

MoveType: CrossCountry

Path

Lat	Lon	Alt
58.581317273645624	16.25441968074	0
58.58061699021803	16.26527980061	0
58.58106210795927	16.27424542643	0
58.582323244151496	16.28506110203	0

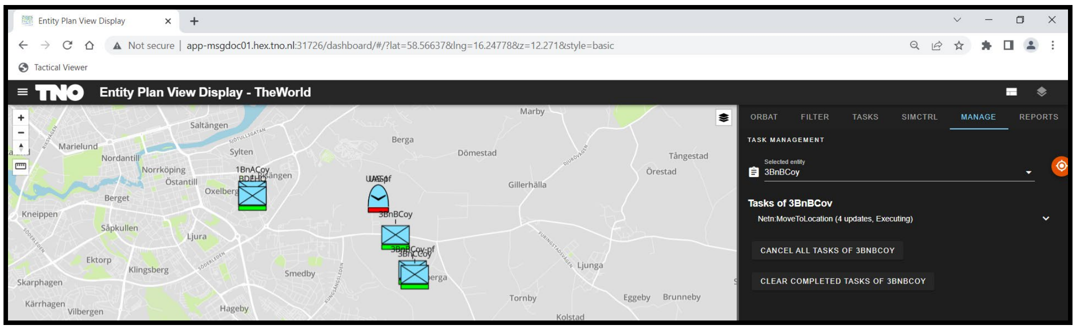
StartWhen: 00000000000000.000
yyyyMMddHHmmss.SSS

Tasker

TaskMode
Concurrent mode: false

3) Submit the task.

The CGF that manages the entity shall report back on the task status. In the example below the task status is “Executing”. If multiple tasks are submitted, they are either placed (by the CGF) in a waiting queue, or are executed concurrently, depending on the task mode and type of task.


Entity Plan View Display - TheWorld

TASK MANAGEMENT

Selected entity: 3BnBCoy

Tasks of 3BnBCov

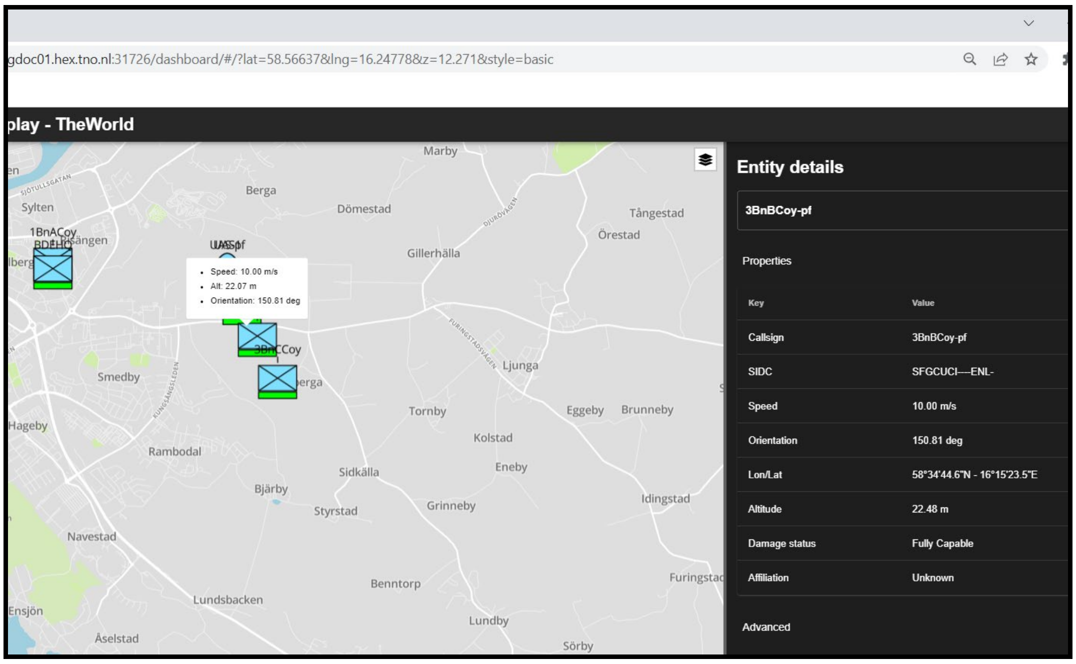
Netn:MoveToLocation (4 updates, Executing)

CANCEL ALL TASKS OF 3BNBCOY

CLEAR COMPLETED TASKS OF 3BNBCOY

4) Check that the entity is moving.

By hoovering on the entity, a pop-up appears with information amongst other speed information. The speed of the entity is 10 m/s.


- Speed: 10.00 m/s
- Alt: 22.07 m
- Orientation: 150.81 deg

Entity details

3BnBCoy-pf

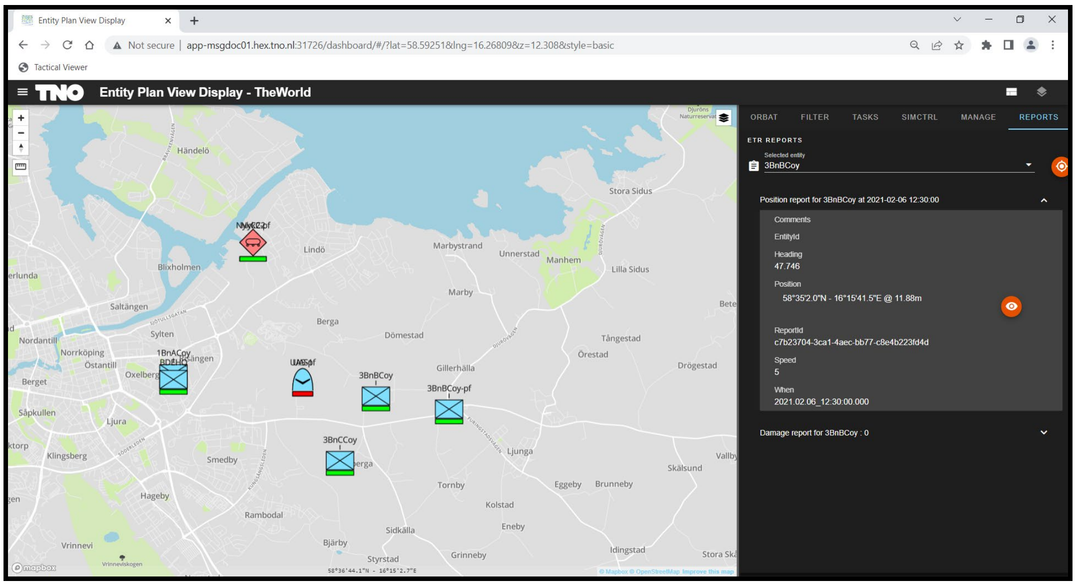
Properties

Key	Value
Callsign	3BnBCoy-pf
SIDC	SFGCUCI----ENL-
Speed	10.00 m/s
Orientation	150.81 deg
Lon/Lat	58°34'44.6"N - 16°15'23.5"E
Altitude	22.48 m
Damage status	Fully Capable
Affiliation	Unknown

Advanced

5) View position reports.

While the entity is moving the CGF may provide position reports of the entity. These (if any) can be viewed in the EPVD UI.


Entity Plan View Display
Not secure | app-msgdoc01.hex.tno.nl:31726/dashboard/#/?lat=58.59251&lng=16.26809&z=12.308&style=basic
Tactical Viewer
TNO Entity Plan View Display - TheWorld
ORBAT
FILTER
TASKS
SIMCTRL
MANAGE
REPORTS
NMCC2-pf
1BnACoy
3BnBCoy
3BnBCoy-pf
3BnCCoy
mapbox

ETR REPORTS

Selected entity
3BnBCoy

Position report for 3BnBCoy at 2021-02-06 12:30:00

Comments

EntityId

Heading
47.746

Position
58°35'2.0"N - 16°15'41.5"E @ 11.88m

ReportId
c7b23704-3ca1-4aec-bb77-c8e4b223fd4d

Speed
5

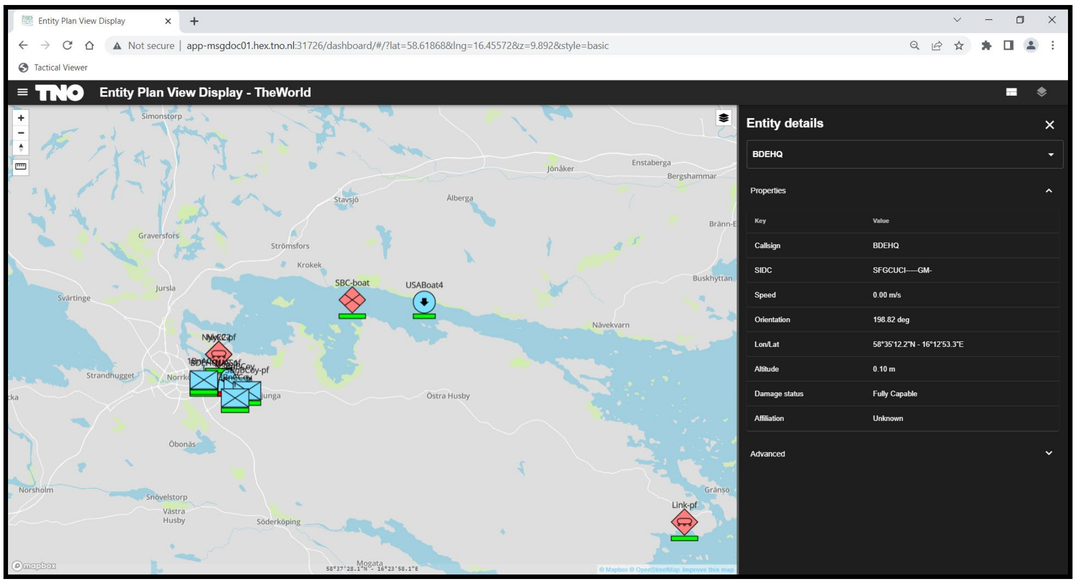
When
2021.02.06_12:30:00.000

Damage report for 3BnBCoy : 0

5.6 Issue NETN MagicMove task

While the MoveToLocation task is in progress let's reposition another entity "SBC-boat" with a magic move. This is accomplished by a drag & drop of the entity on the new position in the EPVD UI. Under the hood the EPVD sends a NETN MagicMove task for the target entity into the federation execution.

1) Select SBC-boat.


Entity Plan View Display
Not secure | app-msgdoc01.hex.tno.nl:31726/dashboard/#/?lat=58.61868&lng=16.45572&z=9.892&style=basic
Tactical Viewer
TNO Entity Plan View Display - TheWorld
SBC-boat
USABoat4
NMCC2-pf
Link-pf
mapbox

Entity details

BDEHQ

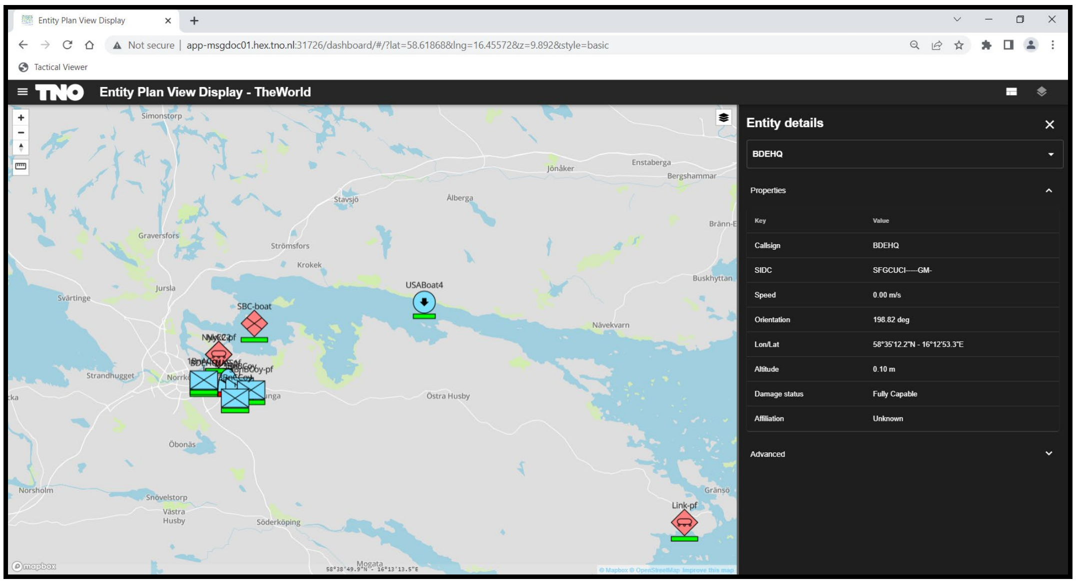
Properties

Key	Value
Callsign	BDEHQ
SIDC	SFGCUCI-----GM-
Speed	0.00 m/s
Orientation	198.82 deg
Lon/Lat	58°35'12.2"N - 16°12'53.3"E
Altitude	0.10 m
Damage status	Fully Capable
Affiliation	Unknown

Advanced

2) And drop SBC-boat on the new position.

We happen to know from the C2SIM LOX Initialization file that SBC-boat is managed by the Entity Creator. So, the NETN MagicMove task is executed by the Entity Creator application.


Entity Plan View Display
Not secure | app-msgdoc01.hex.tno.nl:31726/dashboard/#/?lat=58.61868&lng=16.45572&z=9.892&style=basic
Tactical Viewer
TNO Entity Plan View Display - TheWorld
SBC-boat
USABoat4
Link-pf

Entity details

BDEHQ

Properties

Key	Value
Callsign	BDEHQ
SIDC	SFGCUCI-----GM-
Speed	0.00 m/s
Orientation	198.82 deg
Lon/Lat	58°35'12.2"N - 16°12'53.3"E
Altitude	0.10 m
Damage status	Fully Capable
Affiliation	Unknown

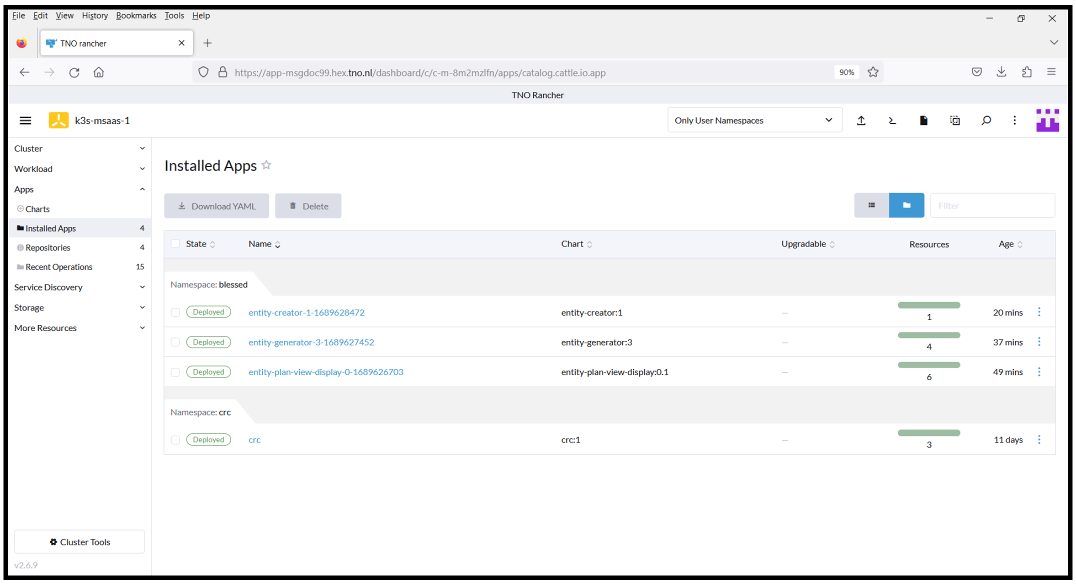
Advanced

5.7 Terminate Applications

In this final step the applications are terminated. One or more applications can be terminated from the Installed Apps section. In this case the applications are terminated one by one.

1) Terminate VR Forces.

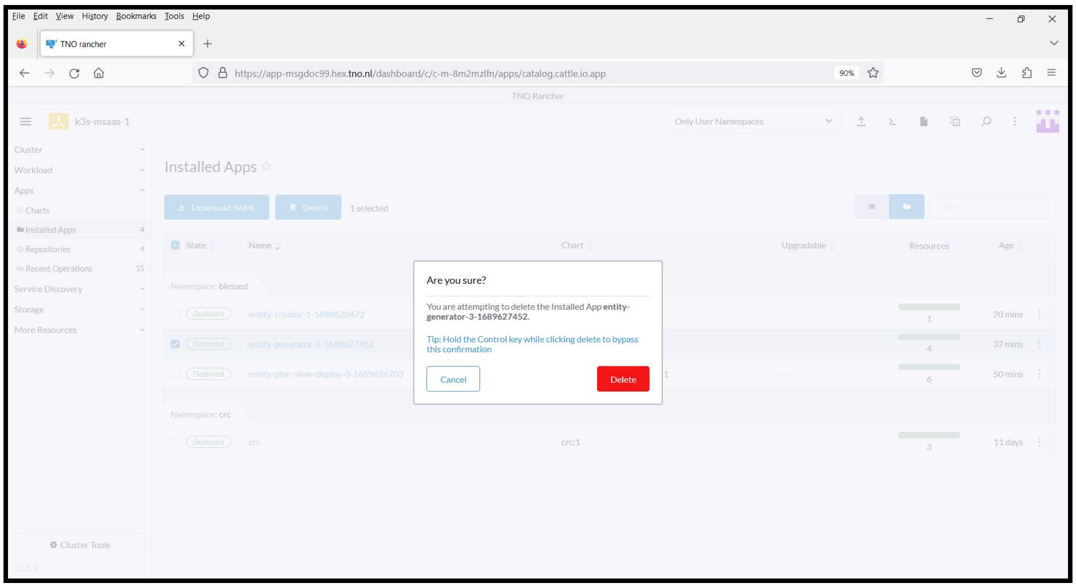
Select the Entity Generator application and press Delete.



Installed Apps

State	Name	Chart	Upgradable	Resources	Age
Namespace: blessed					
Deployed	entity-creator-1-1689628472	entity-creator:1	—	1	20 mins
Deployed	entity-generator-3-1689627452	entity-generator:3	—	4	37 mins
Deployed	entity-plan-view-display-0-1689626703	entity-plan-view-display:0.1	—	6	49 mins
Namespace: crc					
Deployed	crc	crc:1	—	3	11 days

2) Confirm the deletion by pressing Delete again.



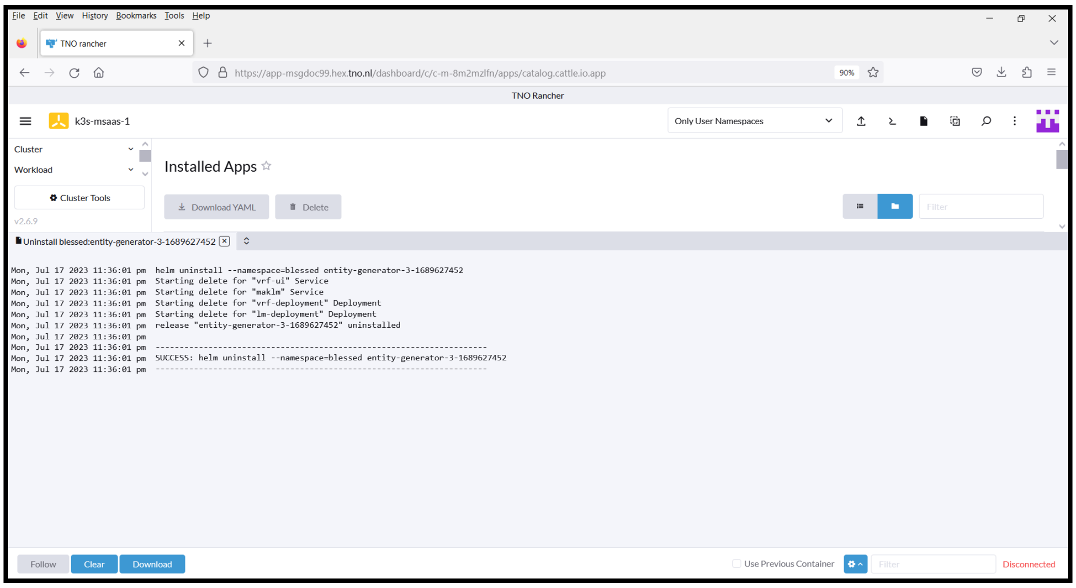
Are you sure?

You are attempting to delete the Installed App **entity-generator-3-1689627452.**

Tip: Hold the Control key while clicking delete to bypass this confirmation

3) The application is removed from the cluster.

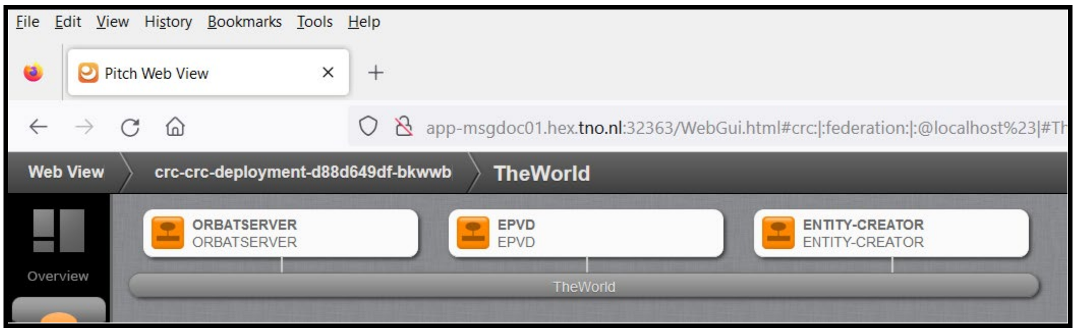
Rancher performs the Helm uninstall command to remove an application from the cluster.



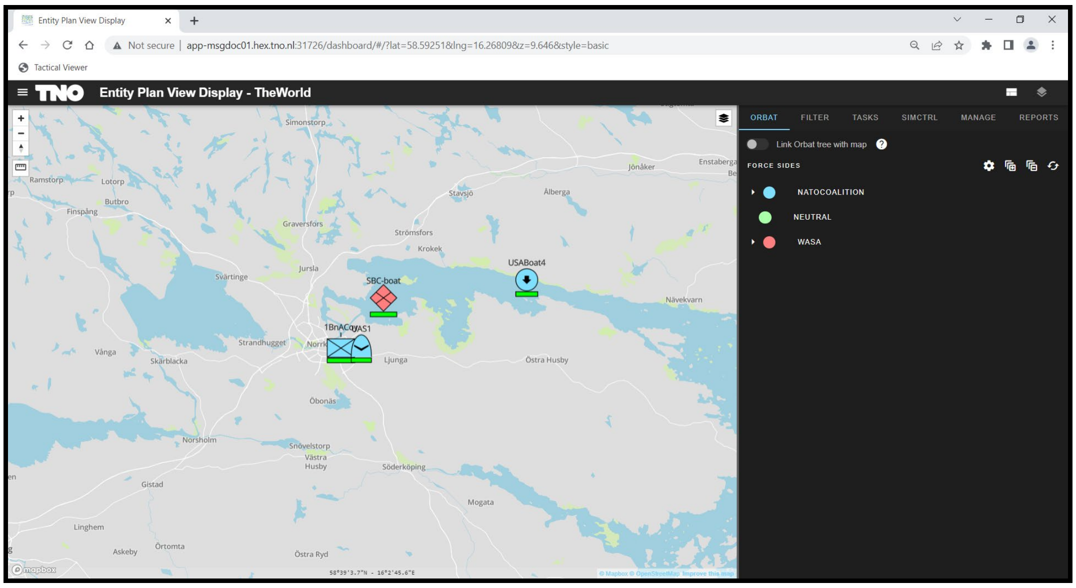
Uninstall blessed:entity-generator-3-1689627452

```
Mon, Jul 17 2023 11:36:01 pm  helm uninstall --namespace=blessed entity-generator-3-1689627452
Mon, Jul 17 2023 11:36:01 pm  Starting delete for "vrf-ui" Service
Mon, Jul 17 2023 11:36:01 pm  Starting delete for "maklm" Service
Mon, Jul 17 2023 11:36:01 pm  Starting delete for "vrf-deployment" Deployment
Mon, Jul 17 2023 11:36:01 pm  Starting delete for "lm-deployment" Deployment
Mon, Jul 17 2023 11:36:01 pm  release "entity-generator-3-1689627452" uninstalled
Mon, Jul 17 2023 11:36:01 pm
Mon, Jul 17 2023 11:36:01 pm  ---------------------------------------------------------------------
Mon, Jul 17 2023 11:36:01 pm  SUCCESS: helm uninstall --namespace=blessed entity-generator-3-1689627452
Mon, Jul 17 2023 11:36:01 pm  ---------------------------------------------------------------------
```

4) Open the Pitch RTI Web UI to confirm that VR Forces resigned from the federation.

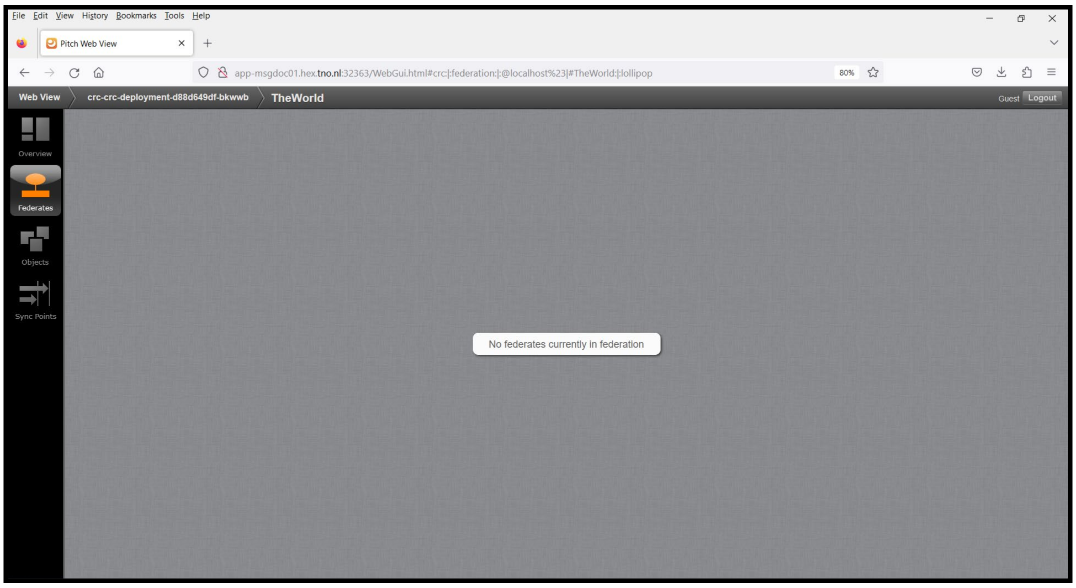

File Edit View History Bookmarks Tools Help
Pitch Web View
app-msgdoc01.hex.tno.nl:32363/WebGui.html#crc:|:federation:|:@localhost%23|#Th
Web View
crc-crc-deployment-d88d649df-bkwwb
TheWorld
ORBATSERVER
ORBATSERVER
EPVD
EPVD
ENTITY-CREATOR
ENTITY-CREATOR
Overview
TheWorld

5) Open the EPVD UI to confirm that the entities allocated to VR Forces have been removed from the simulation.



6) Terminate the other applications.

The Entity Creator and EPVD are terminated in the same way as VR Forces. The result is that all federates have resigned from the federation.


File Edit View History Bookmarks Tools Help
Pitch Web View
app-msgdoc01.hex.tno.nl:32363/WebGui.html#crc|:federation:|:@localhost%23|#TheWorld:|:lollipop
Web View
crc-crc-deployment-d88d649df-bkwwb
TheWorld
Guest
Logout
Overview
Federates
Objects
Sync Points

No federates currently in federation

6.0 SUMMARY

This paper introduced Kubernetes as technology platform and showed how Kubernetes can be used to implement an MSaaS Capability supporting the key capabilities discovery, composition, and execution. In addition, a small example exercise demonstrated the cloud-based deployment and execution of a few simulation applications. These applications, such as VR Forces, could be searched and started from a catalog. The exercise showed the successful application of the following simulation standards:

- C2SIM LOX (SISO-STD-019-2020 and SISO-STD-019-2020), MSDL (SISO-STD-007-2008), and NETN-ORG (AMSP-04B): for the initialization of the simulation.
- NETN-ETR (AMSP-04B): for the tasking and reporting of simulation entities.
- HLA (IEEE 1516-2010): for connecting applications in a simulation environment.
- WebLVC (SISO-STD-017-2022): for the communication of simulation data within the EPVD.

7.0 REFERENCES

[1] T. van den Berg, “Overview of M&S as a Service, paper number 1.6,” in MSG-211 Lecture Series Modelling and Simulation Standards in NATO Federated Mission Networking, 2023.

[2] “Kubernetes,” 2023. [Online]. Available: https://kubernetes.io

[3] “Kubernetes,” [Online]. Available: https://kubernetes.io/docs/concepts/overview/ [Accessed 2023].

[4] “Helm,” [Online]. Available: https://helm.sh

[5] “Rancher,” [Online]. Available: https://www.rancher.com/

[6] "ChartMuseum," [Online]. Available: https://github.com/helm/chartmuseum

[7] "Artifact Hub," [Online]. Available: https://artifacthub.io/

[8] "Pitch Technologies," [Online]. Available: https://pitchtechnologies.com/

[9] "Distributes Simulation Engineering and Execution Process (IEEE-1730)," IEEE, 2010.

[10] Petty, "A Composability Lexicon (03S-SIW-023)," in SISO Simulation Interoperability Workshop, 2003.

[11] T. van den Berg, "High Level Architecture and NATO Education and Training Network (NETN) Federation Object Model (FOM) overview, paper number 1.5," in MSG-211 Lecture Series on Modelling and Simulation Standards in NATO Federated Mission Networking, 2023.

[12] "VTMaK," [Online]. Available: https://www.mak.com/

[13] M. Wurster, U. Breitenbücher, M. Falkenthal and e. al., "The essential deployment metamodel: a systematic review of deployment automation technologies," SICS Softw.-Inensiv. Cyber-Phys. Syst., vol. 35, 2020.