7 DS2 Containerisation Module (CONT)

7.1 DS2 Containerisation Module (CONT)

Owner(s): ICE
DOA Task: T6.4 (was in T6.1)
Tier: Tier 1
Nature: System
Results: K6.1

This task will package the primary outcomes of the projects in a single containerised environment (K8 based) where all modules/concepts/information/methodologies of WP3-5 along with those of T6.1/2/3 will be implemented with the adaptation to IDT being performed within those WPs. This will enable the users to pick-and-mix which final tools may be of use to them. As a modular environment based on the ZDMP Projects (Project Manager ICE) both existing and third-party tools should be accessible.

7.1.1 Introduction

Purpose: To allow easy and automated packaging and deployment of modules on the IDT Kubernetes runtime subcomponent environment. The containerisation module leverages on custom Helm Chart descriptors to automatically convert them into full Kubernetes Helm Charts representing the module, based on standard base templates located in the DS2 Portal Marketplace. The Helm Charts are then deployed on the IDT Module.

Description: The Containerisation module is a core module to the IDT Broker module that enables deployment of all the DS2 modules in the IDT Broker Kubernetes sub-component. The Containerisation module uses Helm Chart standard base templates describing a DS2 module. Those templates are provisioned by the IDT Broker module and provide the standard for DS2 module deployment in IDT Broker. Base templates are stored in the DS2 Portal Marketplace. Then, when uploading a DS2 module by module developers, to the DS2 Portal Marketplace, a custom Helm Chart descriptor that includes values for those base templates needs to be provided with the module. The Containerisation module will use the descriptor together with the base templates to create the Helm Chart for the DS2 module during deployment time on the IDT Broker. The Containerisation module can work in two different modes:

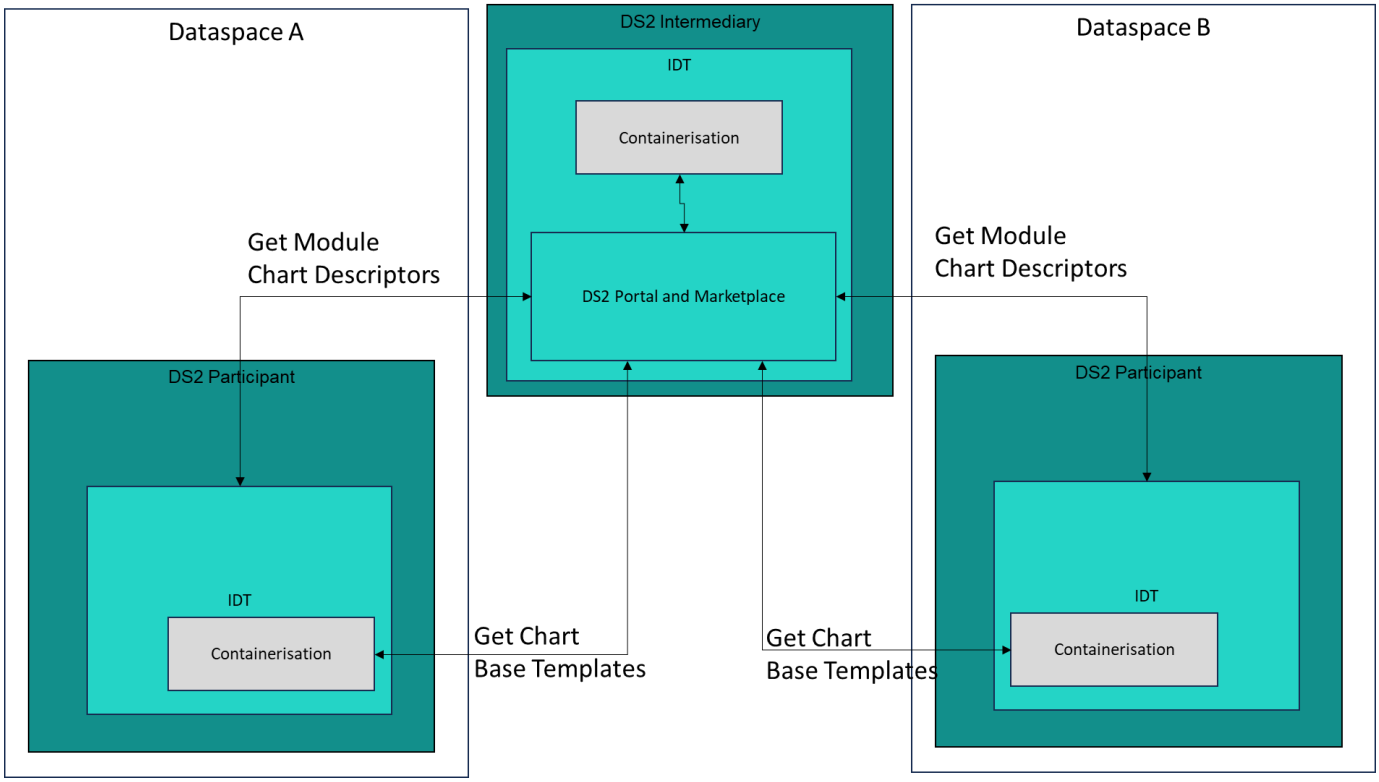
- The standard DS2 working mode: developers upload module Helm Chart descriptor to the DS2 Portal Marketplace. Participants use the IDT Broker Kubernetes UI to deploy the descriptor on the IDT. The Containerisation module is triggered when detecting the deployment of that descriptor, retrieves the base templates from the DS2 Portal Marketplace, creates the full Helm Chart and deploys it on the IDT Kubernetes Runtime sub-component.
- The GitOps way: automatic deployment of the Helm Chart descriptor is triggered by the Source controller sub-component upon detecting a change on the descriptor in the DS2 Portal Marketplace. Then as in the previous mode, the Containerisation module, create the full Helm Chart and deploys it on the IDT. This could be the deployment mode of the DS2 Portal.

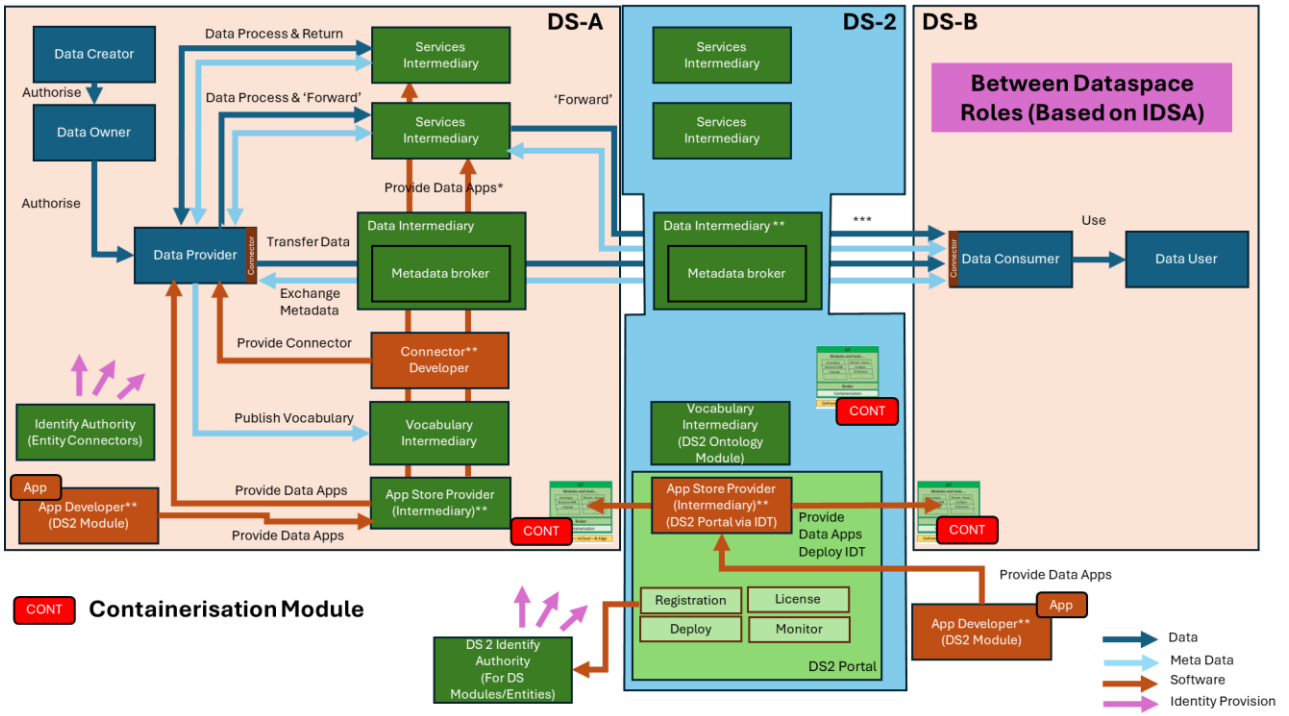
In both cases, the only difference is how the Helm Chart descriptor is deployed on the IDT either by the participant manually deploying the descriptor or being automatically deployed by the Source Controller sub-component.

7.1.2 Where this component fits

7.1.2.1 Big Picture

The Containerisation module is a core module of the IDT Broker module used in a participant IDT and required to deploy the other modules on the IDT. It will also be used by DS2 service intermediaries, since those services are also DS2 modules being run in an IDT.





Where	Status
Within and across a single Dataspace	N/A
Deployed and used by a single participant	Yes: core IDT module, pre-installed with the IDT installation
Across Dataspaces without intermediary	N/A
Across Dataspaces with Intermediary	Yes: intermediary services are also DS2 modules, thus, deployed on IDT using the Containerisation module
Other Comments	N/A

7.1.2.2 Within and across a single Dataspace (where applicable)

N//A

7.1.2.3 Deployed and used by a single participant

The Containerisation module is a core module of the IDT Broker module, pre-installed during IDT installation. It will then be used to automatically package and deploy the DS2 modules in the IDT with the right format.

7.1.2.4 Across Dataspaces without intermediary (where applicable)

N/A.

7.1.2.5 Dataspace Intermediary with intermediary(where applicable)

Intermediary services are also DS2 modules. Those are also deployed and run on IDT and the deployment mode is using the Containerisation module. In some cases, for the DS2

Portal, the automatic deployment mode of the Containerisation module might also be used.

7.1.3 Component Definition

The figure below represents a high-level flow diagram of the different steps in order to install a DS2 module on the IDT using the Containerisation module.

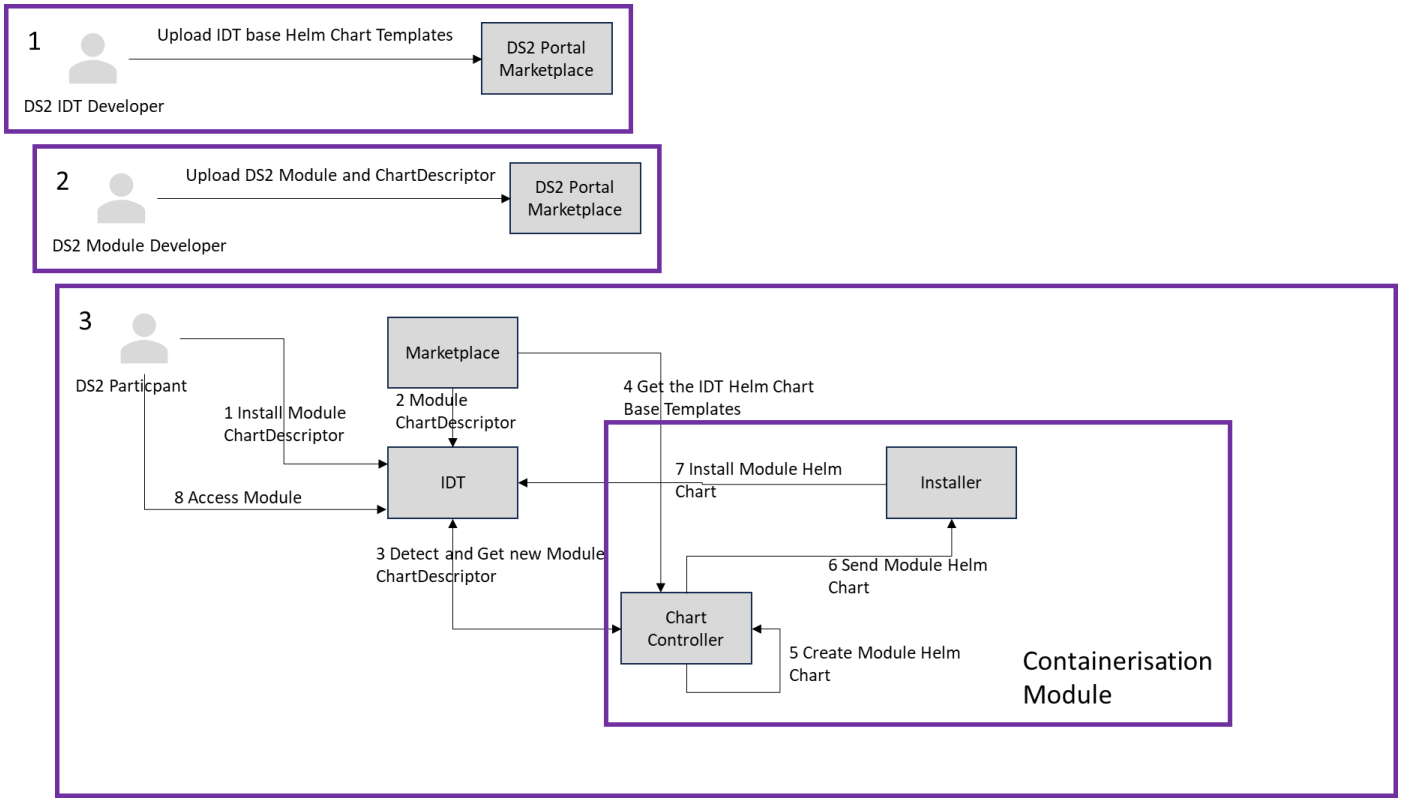


Figure 1: Flow Diagram for the DS2 Containerisation Module

The figure below represents the actors, internal structure, primary sub-components, primary DS2 module interfaces, and primary other interfaces of the module.

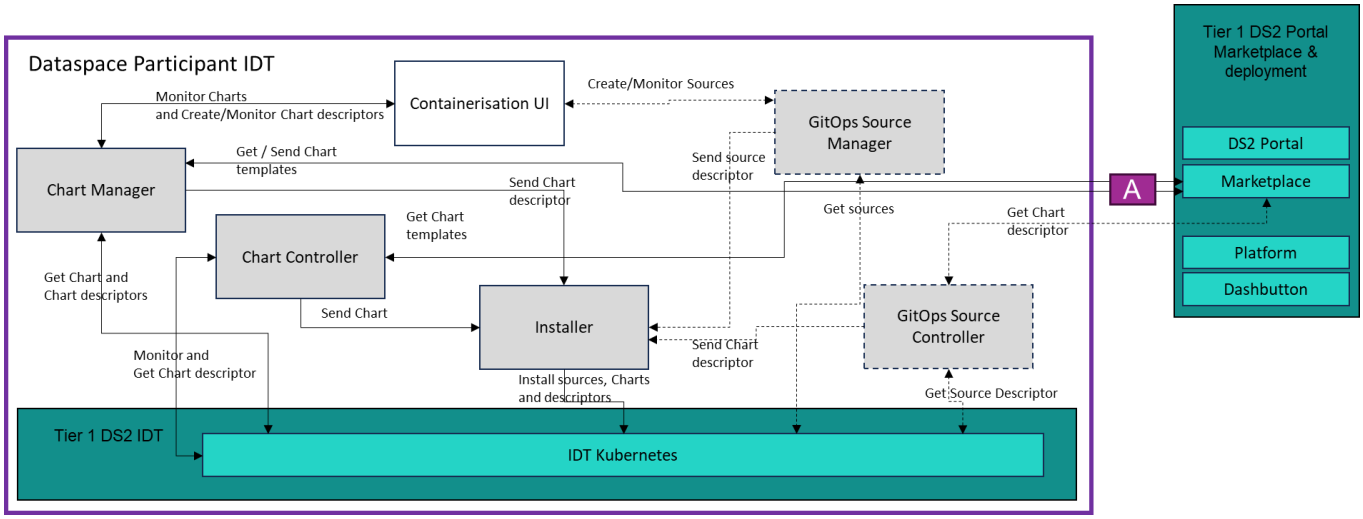


Figure 2: Schema for the DS2 Containerisation Module

This module has the following subcomponent and other functions:

- **ChartController:** The ChartController is a Kubernetes controller, following the Kubernetes controller pattern which keeps track of a new Kubernetes custom resource definition - the “HelmChartDescriptor”. When changes are detected on a descriptor, ie. addition, update, the Controller connects to a configured location ie. GitHub repository, to download the corresponding Helm Chart base templates. Then, together with the HelmChartDescriptor, the Chart Controller will create a full Helm Chart describing the module. This Helm Chart will be deployed into the IDT Kubernetes Runtime subcomponent using the Installer component.
- **ChartManager:** The ChartManager is mainly used to monitor the Helm Charts and HelmChartDescriptors deployed in the system. It will query the IDT Module’s Kubernetes subcomponent to retrieve current Charts and descriptors. The Chart Manager can also be used to create a HelmChartDescriptor using some input parameters and install it via the Installer component. Once installed, the ChartController will detect the new ChartDescriptor and will convert it to a Chart deploying it back into the IDT Module’s Kubernetes subcomponent.
- **Installer**: This is the component responsible for installing Helm Charts and HelmChartDescriptors in the IDT Kubernetes subcomponent. It will receive the corresponding Charts and HelmChartDescriptors and will apply them in the IDT Kubernetes subcomponent. The Installer also takes care of installing new Sources created by the Source Manager component.
- **Containerisation UI:** This is the main module UI that allows users to monitor current existing Charts, ChartDescriptors and Sources in the system. Users will have an overview of what is installed in the system and its current status regarding to those specific resources. The UI can also be used to create, update or delete ChartDescriptors via the ChartManager and Sources via the Source Manager.
- **GitOps Source Controller:** The Source Controller, similar to the ChartController, is a Kubernetes controller that keeps track of the custom resource definition Source. A Source mainly represents a reference to a repository where ChartDescriptors are stored. The Source Controller monitors the status of the Source and reacts to changes by reflecting those changes in the IDT Kubernetes subcomponent. The Source Controller is an optional subcomponent, and users can just install the ChartDescriptors using the IDT or via Kubernetes standard kubectl.
- **(DS2) GitOps Source Manager:** The Source Manager, similar to the ChartManager is mainly used to monitor the Source in the system and is customised to DS2. It can also be used to create, update, and delete new sources that will be installed via the Installer component. As the Source Controller, this is an optional component.
- **Tier 1 Service Stack for Marketplace and deployment** and **API:** The full stack will be implemented as generically described elsewhere in this document. Exceptions: This module runs in the IDT and uses the IDT Kubernetes subcomponent for Chart and ChartDescriptor installations. The DS2 Portal Marketplace component and its repository system is used to store the Chart base templates. Since the DS2 Portal is also a DS2 module, it is deployed and run on the IDT, so Containerisation module can also be used for the DS2 Portal and other intermediary services.

7.1.4 Technical Foundations and Background

The Containerisation module is brand new development for DS2 based and evolving from existing background on the use of Helm Charts Subcharts and ICE’s Helm plugin making use of it. This subchart and plugin concept is further evolved into an actual Kubernetes controller using Kubernetes controller standard frameworks, and delivered as a full web

app. Two main controllers will be developed most likely using the GoLang frameworks. The Source Controller is expected to be an extension of current GitOps Toolkit framework. The UI will be developed using one of the modern frontend frameworks such as Angula, React or Vue. ICE vision is to include this in a commercial product still pending of branding or commercial name but most likely to be called icEECharts or HelmChartsEE (Enterprise Edition).

Subcomponent/Component	Owner	License
N/A		

7.1.5 Interaction of the Component

The following table specifies the primary input/output controls/data to blocks which are not part of the module

With Module/Feature	Receives from /Gives To	What
Portal	Gives To	When published on the portal, information (technical, how tos etc etc) will be provided according to the general model
Portal Marketplace	Receives From	The ChartDescriptors and the Chart base templates
T6.4 IDT	Gives To	Sends the Charts and ChartDescriptors (install in IDT Kubernetes)
T6.3 IDT	Receives From	Receives information on current Charts and ChartDescriptors

7.1.6 Technical Risks

Risk	Description	Contingency Plan
Integrate with DS2 trust system	The containerisation module needs login information for the Portal Marketplace repositories	This can be done integrating with the DS2 trust system, or establishing specific mechanisms to this integration, such as specific repository access tokens for participants, since this is not part of Data exchange, thus it should not need DS2 trust system
Complexity of generating base templates	The generation of base templates could not be as general as initially expected depending on module needs	Generate several groups of templates according to the different modules

7.1.7 Security

Security Issue	Description	Need
In-Dataspace	The containerisation module is deployed at a participant so there is no specific DS Security risk. The UI will integrate with the DashButton to provide the local security if used	N/A
Tier-1 Portal	The containerisation module needs login information for the Portal Marketplace repositories	Get token in order to pull eeChart templates and Docker images from repository

7.1.8 Data Governance

Data Governance Issue	Description	Need
N/A	No issues since the Containerisation module does not use participant data at all, only takes care of deploying the module from ChartDescriptor to Chart	N/A

7.1.9 Requirements and Functionality

This module will be used in the following usecases:

City Scape ✓
Green Deal ✓
Agriculture ✓
Inter-Sector ✓

Their requirements and functions/extensions to achieve them relative to this module, specifically extracted from the use case are as per the table below noting that in many cases further discussion might need to take place between pilot partner and module partner to determine if a fit or the scope of the precise fit.

In respect to all use cases the DS2 Containerisation module is a system module which is an intrinsic part of the project design and is too low-level to have been mentioned in any use case.

WHERE	WHAT	WHY	Run/Design Time	Priority
	Use Case 1: City Scape			
N/A			R & D	M
	Use Case 2: Green Deal			
N/A			R & D	M
	Use Case 1: Agriculture			
N/A			R & D	M

7.1.10 Workflows

The following sub-sections describe the sequence diagrams of the Module

7.1.10.1 Create and Install Module Helm Chart from Descriptor

This feature provides the capability to get a Descriptor already installed in the IDT Kubernetes and with the Chart Templates, convert it into a Module Helm Chart and install it back into the IDT.

The main steps/functionalities are as follows:

- Get the Descriptor
- Get the Chart Templates
- Create the Module Helm Chart

- Install the Module Helm Chart

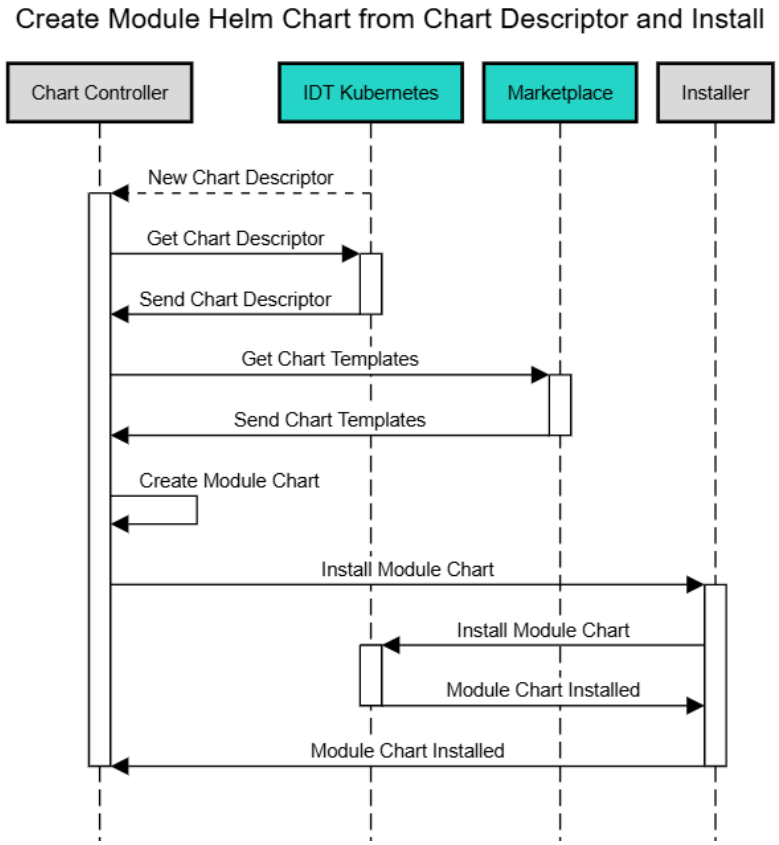


Figure 3: Create and Install Module Helm Chart from Descriptor sequence diagram

7.1.10.2 Create and Install Chart Descriptor

This feature provides the capability to create a Chart Descriptor using the UI and install it into the IDT Kubernetes. This will trigger the conversion of the Descriptor into a Helm Chart and install it as per the previous workflow.

The main steps/functionalities are as follows:

- Get the Chart Templates
- Create the Values for the Chart Descriptor
- Send the Values to the Chart Manager
- Create the Chart Descriptor
- Install the Chart Descriptor

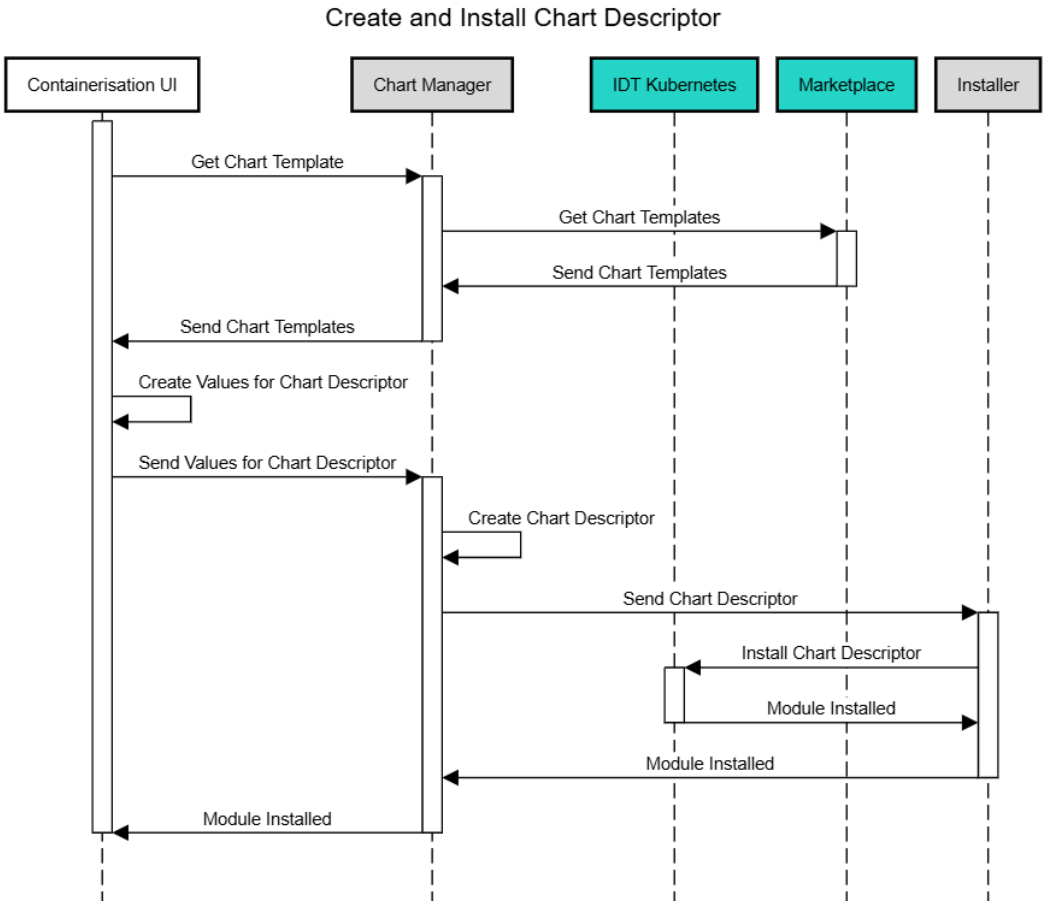


Figure 4: Create and Install Chart Descriptor sequence diagram

7.1.10.3 Create and Install Source

This feature provides the capability to create a Source Descriptor using the UI and install it into the IDT Kubernetes. This will trigger the Source Controller to update the sources to monitor.

The main steps/functionalities are as follows:

- Get the Source Template
- Create the Values for the Source Descriptor
- Send the Values to the Source Manager
- Create the Source Descriptor
- Install the Source Descriptor

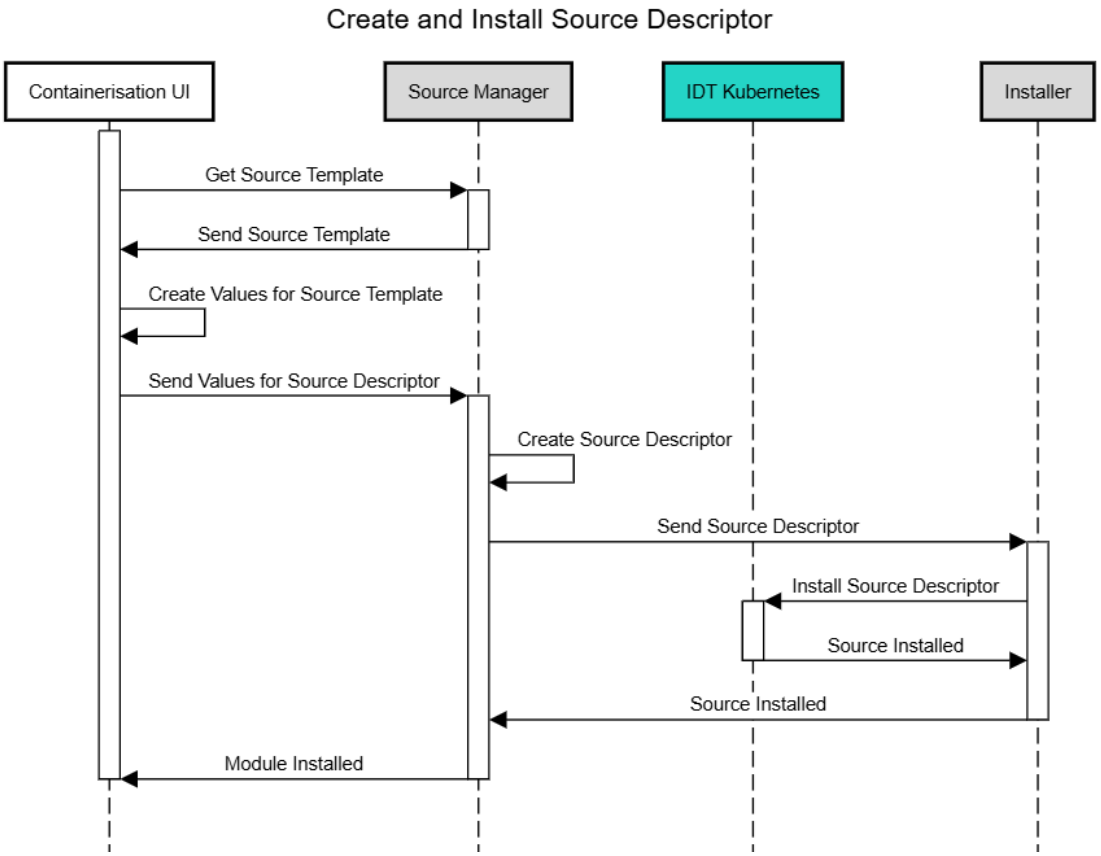


Figure 5: Create and Install Source Descriptor diagram

7.1.10.4 Update Source in the Source Controller

This feature provides the capability to update a Source in the Source Controller. This registers a new git repository in the Source Controller then triggers the Source Controller to pull the Chart Descriptor in that repository and install it via the Installer in the IDT Kubernetes. This triggers the Create and Install Module Helm Chart workflow.

The main steps/functionalities are as follows:

- Get the Source Descriptor
- Register new git repository in the Source Controller
- Get the Chart Descriptor in the git repository
- Install the Chart Descriptor in the IDT Kubernetes

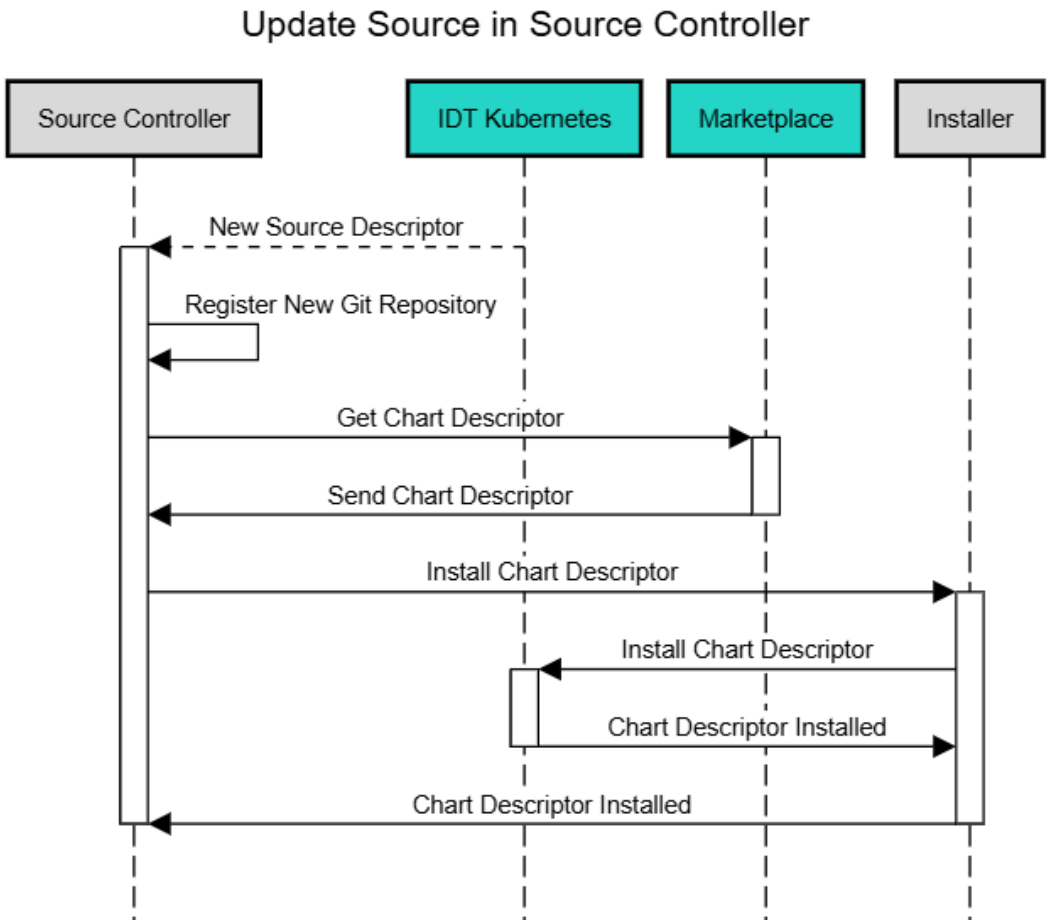


Figure 6: Create and Install Source Descriptor sequence diagram

7.1.10.5 Monitor Charts and Descriptors

This feature provides the capability to monitor the installed Charts and Descriptors and display information about those resources in the Containerisation UI.

The main steps/functionalities are as follows:

- Get the Charts and Descriptors Information
- Display the Charts and Descriptors Information in the Containerisation UI

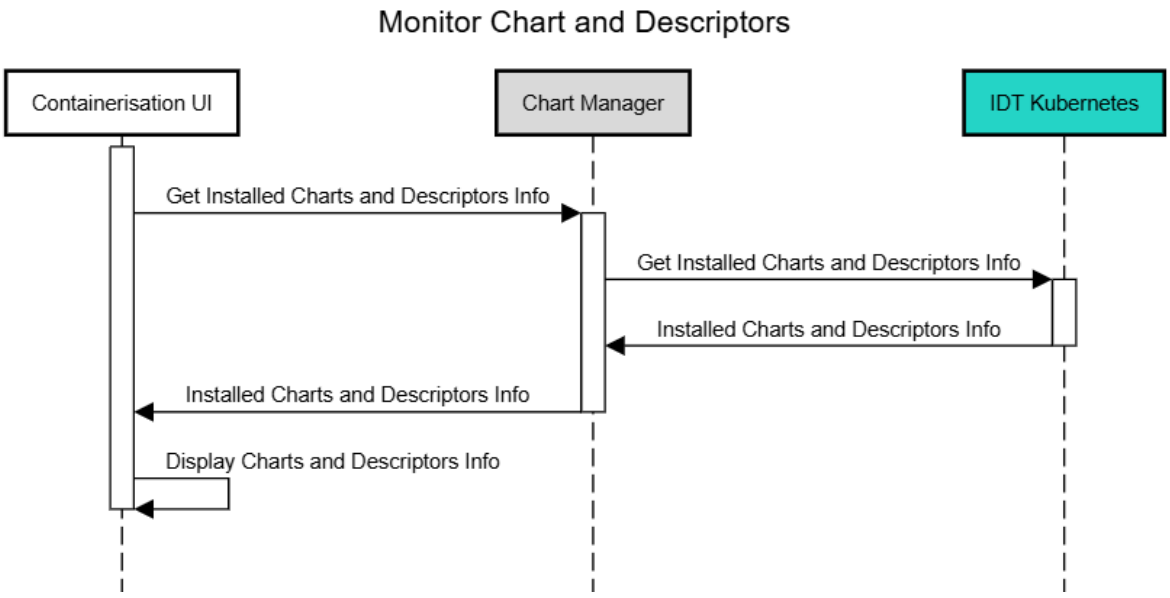


Figure 7: Monitor Charts and Descriptors sequence diagram

7.1.10.6 Monitor Sources

This feature provides the capability to monitor the installed Sources and display information about those resources in the Containerisation UI.

The main steps/functionalities are as follows:

- Get the Sources Information
- Display the Sources Information in the Containerisation UI

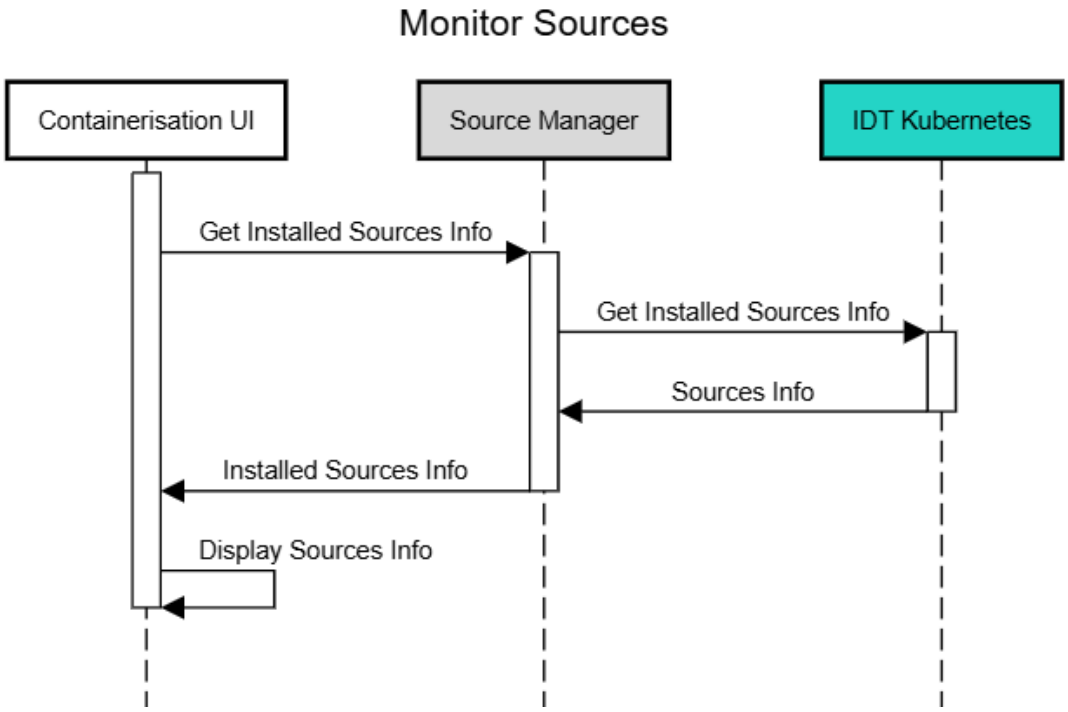


Figure 8: Monitor Sources sequence diagram

7.1.11 Role, Resourcing, and Milestones

Sub-component	Main Activity	M18	M24	M30	M36
ChartController	Full component development				
ChartManager	Full component development				
Installer	Full component development				
Containerisation UI	Full component development				
GitOps Source Controller	Investigate the possibility of reusing open-source software “The GitOps Toolkit” for this or full component development				
GitOps Source Manager	Full component development				
Dashbutton	Integration				
Table Total/DOA Task Total/Resilience	**Comments:** The Source Controller and Manager are optional				

7.1.12 Open Issues

The following table summarise open issues/uncertainties that need to be resolved during the next stages or implementation.

Issue	Description	Next Steps	Lead or Related Component
Source Controller	The source controller could be based on open source software The GitOps Toolkit	Research on The GitOps Toolkit to check if that's an option	None