11 DS2 Model Development Toolkit Module (MDT)

11.1 DS2 Model Development Toolkit Module (MDT)

Owner(s): INDRA
DOA Task: T4.2
Tier: 2
Nature: Optional
Result: Outcome

This task will research and classify different forms of anomalous behaviour / error conditions over complex data lifecycles and create classifiers to recognise them. It will investigate the most appropriate AI algorithms, Methodologies to mitigate the effect of detected degradation of data quality will be investigated, and work together with the Human-centric Tools work package when human intervention is required.

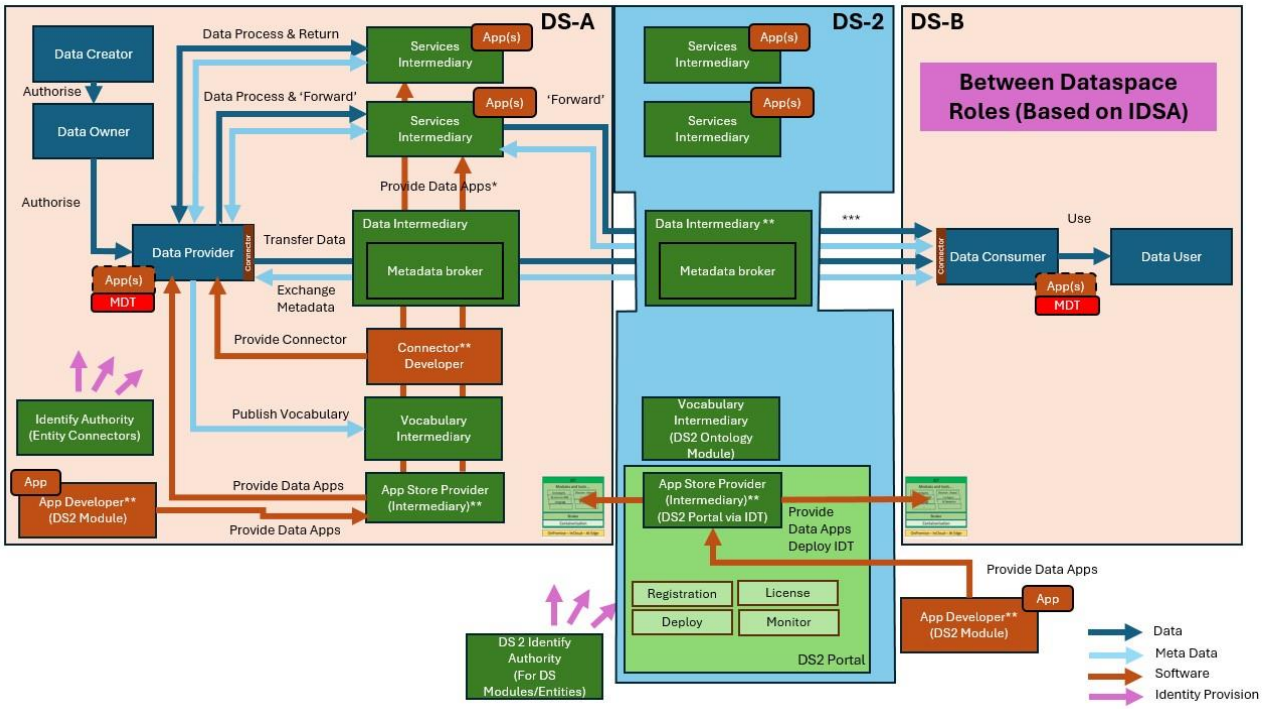
11.1.1 Introduction

Purpose: The main purpose of the DS2 Model Development Toolkit Module (MDT) is to provide a set of tools to allow the users to develop Algorithms based on the CRISP-DM standard to assist in the whole development cycle (training, test, etc.) and package the algorithms which can be deployed as executable software component.

Description: MDT provides a suite of integrated tools to develop algorithms that adhere to CRISP-DM, a widely accepted methodology for data mining projects. It includes features for understanding the business objectives and the data that are the focus of the analysis. It offers tools for preparing datasets from raw data, developing, and training models, and testing them. It allows packaging models as containers with a REST API for easy deployment and integration into other systems. Additionally, it provides monitoring capabilities to evaluate the performance of the models in runtime when deployed. The module supports the use of external libraries for data analysis and comes with preloaded libraries that are particularly useful for anomaly detection.

11.1.2 Where this component fits

11.1.2.1 Big Picture



Where	Status
Within a single Dataspace for use between participants in that Dataspace only	N/A
Deployed and used by a single participant to enable the participant in either an In-Data space or Inter- Data space scenario	Yes: The Model Development Toolkit can be used in single dataspace use cases.
Across Dataspaces without Service Intermediary	N/A
Across Dataspace with Intermediary	No.
Other Comments	N/A

11.1.2.2 Within a single Dataspace (where applicable)

N/A

11.1.2.3 Deployed and used by a single participant (where applicable)

The Model Development Toolkit is a module designed for use by data providers and data consumers to help analyse data before or after data sharing. It is intended to be part of the

participants' data management operations and can therefore be used in both single-dataspace scenarios and for data sharing between participant in different dataspaces.

The DS2 MDT allows the creation of data models that can be used in various contexts. The general process for a participant using this toolkit involves selecting the data they want to analyse, inspecting the data to better understand its characteristics, creating datasets for training, and using the provided tools to test the model with different parameters and inputs to find the best configuration. Participants can then package and deploy the data model, making it ready to be integrated into other applications and data pipelines using a REST API. The toolkit also enables monitoring the performance of the deployed data model and, when necessary, repeating the process to achieve a more accurate version of the model. This process is agnostic to data sharing and is designed to develop data analytics components that participants can integrate into their data pipelines to share data and analyse data shared by other participants or for exploit data by themselves.

11.1.2.4 Across Dataspaces without Intermediary (where applicable)

N/A

11.1.2.5 Across Dataspace with Intermediary (where applicable)

N/A

11.1.3 Component Definition

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

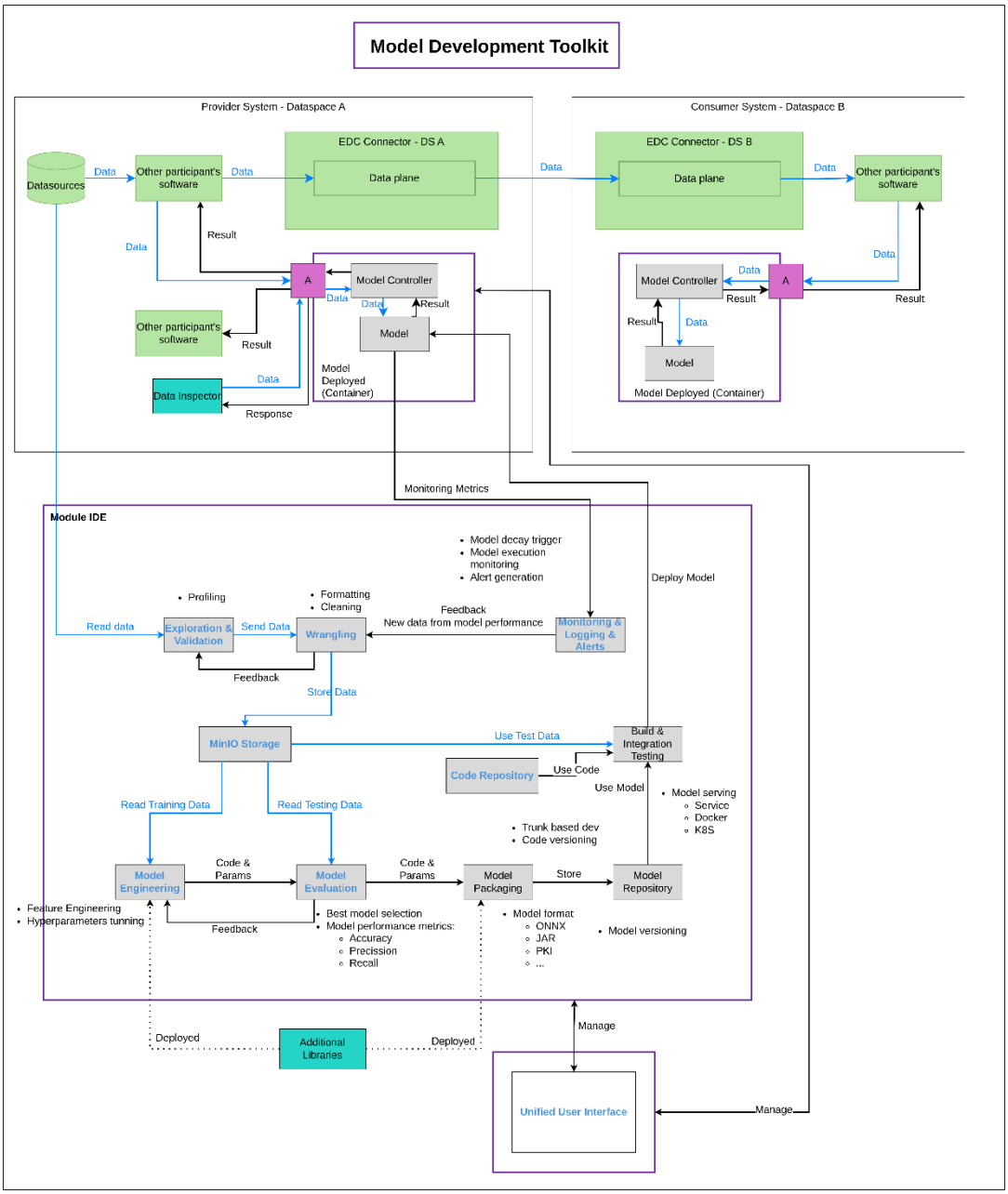


Figure 1: Schema for the DS2 Module

This module has the following subcomponent and other functions:

- **Module IDE:** This set of components provides all the functionality needed to develop data models and algorithms during design time.
    - **Exploration & Validation:** This component allows visual exploration and profiling of data by analysts. It incorporates aspects of the Onesait Platform's profiling and dashboard engine components, which will be extended to enhance profiling and tagging of input data. It supports structured data in various formats (CSV, JSON, XML, etc.) as well as non-structured data such as text, images, and video. This module read data from the repositories directly. It supports the most used technologies as SQL database, MongoDB, S3, FTP, etc. This component temporally stores the data in a staging area.

This component together with Wrangling component help to create datasets for training.

- **Wrangling:** This component enables the definition and execution of rules to clean data of malformed or undesirable values. It leverages an existing component from the Onesait Platform, based on the Open Refiner open-source project, which is useful for structured data. Additional development will be required to handle non-structured data. The Wrangling component can process data in the staging area of the Exploration & Validation module. Both modules collaborate to define datasets for training. Once the cleaning and formatting process is complete, the resulting dataset is stored in the Minio Storage component.
- **Monitoring, Logging, & Alerts:** Provides real-time monitoring and data visualization, including functionality for creating dashboards to track request flows and performance issues in model execution. It also defines alerts for detecting anomalous behaviour during model execution and is based on the Grafana software stack. Key tasks include creating monitoring dashboards, defining alerts, and integrating with the Unified User Interface.
- **MinIO Storage:** Datasets used for training and testing models will be stored in MinIO object storage. Minimal integration work is needed. The Wrangling component stores the datasets, and MinIO Storage provides them to the components that train and test the models. Any component that requires datasets can request them from MinIO Storage. For example, the Model Evaluation component will request data for validating models.
- **Model Engineering:** This is used to define and create models, based on two open-source projects. The first is Apache Zeppelin, a web-based notebook to perform interactive data analytics and visualization, supporting multiple programming languages such as Scala, Python, SQL, and more. The second is a tool for defining simple models using SQL language. Some minor enhancements will be necessary for better integration and pre-installed libraries, with additional tools required for picture and video analysis during the project execution.
- **Model Evaluation:** This component handles the training-evaluation cycle and is based on MLflow, an open-source platform for managing the end-to-end machine learning lifecycle. MLflow enables tracking experiments, packaging code into reproducible runs, and sharing and deploying models across different environments. The main task is to integrate MLflow with the Model Engineering and Model Packaging components.
- **Model Packaging:** Once a model is ready for production deployment, it will be packaged into a model container with clear interfaces for integration with other systems. This component is responsible for creating such deployable and executable modules.
- **Model Repository:** Stores model definitions, including versions and metadata about model development, such as historical information about datasets and tests performed in the training process. It also stores the necessary data for building the model, such as libraries and their versions, additional parameters, and more.
- **Code Repository:** Stores the code required to build the model as a deployable component, based on GitLab software, which provides Git repositories. Only configuration and integration tasks with the Unified User Interface are needed.

    - **Build & Integration Testing:** Defines and executes the construction of models as deployable software. It uses the code provided by the Code Repository together with the Model Repository data to build a software that can be deployed and executed.
- **Model Deployed:** As well as the containerised model, relevant code, each runtime-deployed model will include the following components:
    - **Model Controller:** Manages the lifecycle of the module at runtime and provides an interface for external applications to use the module.
    - **Model:** Executes the model definition, including all required software dependencies, runtime engines, and libraries for normal module execution.
- **Unified User Interface:** By interfacing with the other components of the module, this component enables the development of data models and algorithms during design time, as well as the visualization of alerts and monitoring data during runtime. Facilitates the usage and management of all integrated tools, providing single sign-on access to all module components. It is based on the existent Control Panel component of Onesait Platform that will be extended to support all the new capabilities introduced by this module and described in this section.
- **Data Inspector:** The analysis jobs executed by the T6.2 Data Inspector Module will have the capability to use models created and deployed with MDT. For more details, refer to the T6.2 Data Inspector description.
- **Additional Libraries:** As part of the execution of T4.2, DIGI will analyse and identify additional libraries to be included and supported for the MDT.

11.1.4 Technical Foundations and Background

Some of the components included in the Module IDE group are based on existing Onesait Platform components. Onesait Platform is an open-source modular platform, and consequently, its components, as the used by this module, also are open-source. The Onesait Platform components will be upgraded as it is stated in the previous section.

as part of the execution of T4.2, DIGI will analyse and identify additional libraries to be included in MDT.

Subcomponent/Component	Owner	License
Onesait Platform	INDRA	Apache 2.0

11.1.5 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	Received From/Gives To	What
Additional Libraries for Anomaly Detection	Received From	T4.2 provides a list of preinstalled libraries that will be available for use in the module for creating data models. DIGI identifies the libraries and INDRA integrates them in the module.
T6.2 - Data Inspector	Gives To	The T6.2 Data Inspector Module will utilize models developed by the current module during runtime to allow the use of anomalous detection algorithms (T4.2) for data quality continuous monitoring (T6.2).

11.1.6 Technical Risks

Risk	Description	Contingency Plan
Deployment of libraries with compatibility issues	Libraries often have dependencies on specific runtime engine versions and third-party libraries, which can sometimes lead to incompatibilities with other libraries present in the environment.	When integrated, the libraries will be analysed, and a set of compatible library versions will be selected for use.
Complex data models may require substantial hardware resources.	Complex data models and/or large datasets may require expensive hardware.	For demonstrations of the module, datasets and problems will be selected to mitigate this risk. In practical use cases, participants must provide hardware that matches the scale of the data they intend to analyse.

11.1.7 Security

Security Issue	Description	Need
Authentication / Authorization of IDE UI	The IDE tools require user management.	As development tools, they necessitate users be managed by the participants in the same way as other internal user accounts, such as those for databases or internal documentation.
Dataset Access	The creation and execution of data models requires access to the dataset they are analysing.	The data owner determines who can access the data.

11.1.8 Data Governance

Data Governance Issue	Description	Need
Datasets	The data analytics require access to the dataset they are analysing.	The data owner determines who can access the data.
Handling of personal data	Personal data may be contained within the datasets processed by this module.	Data owners and data users should ensure that personal data is used and storage in accordance with relevant regulations.

11.1.9 Requirements and Functionality

This module will be used in the following use cases:

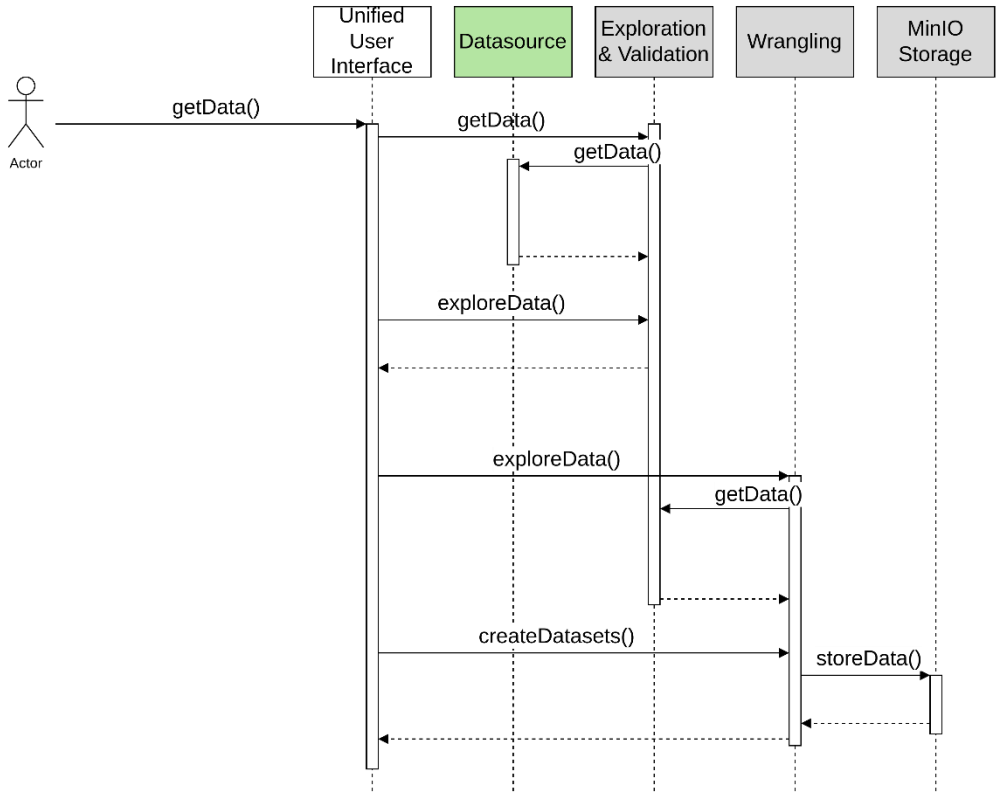
City Scape ✓
Green Deal ✓
Agriculture ✓
Inter-Sector TBD

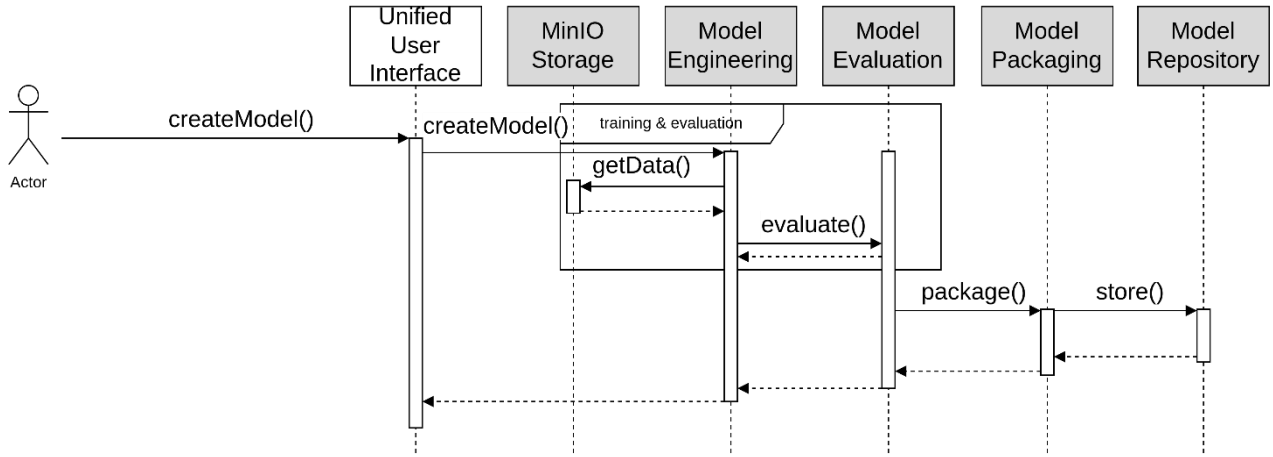
Their requirements and functions/extensions to achieve them relative to this module, specifically extracted from the use case need further discussion during the WP7 setup phase between pilot partner and module partner to determine if a fit or the scope of the precise fit.

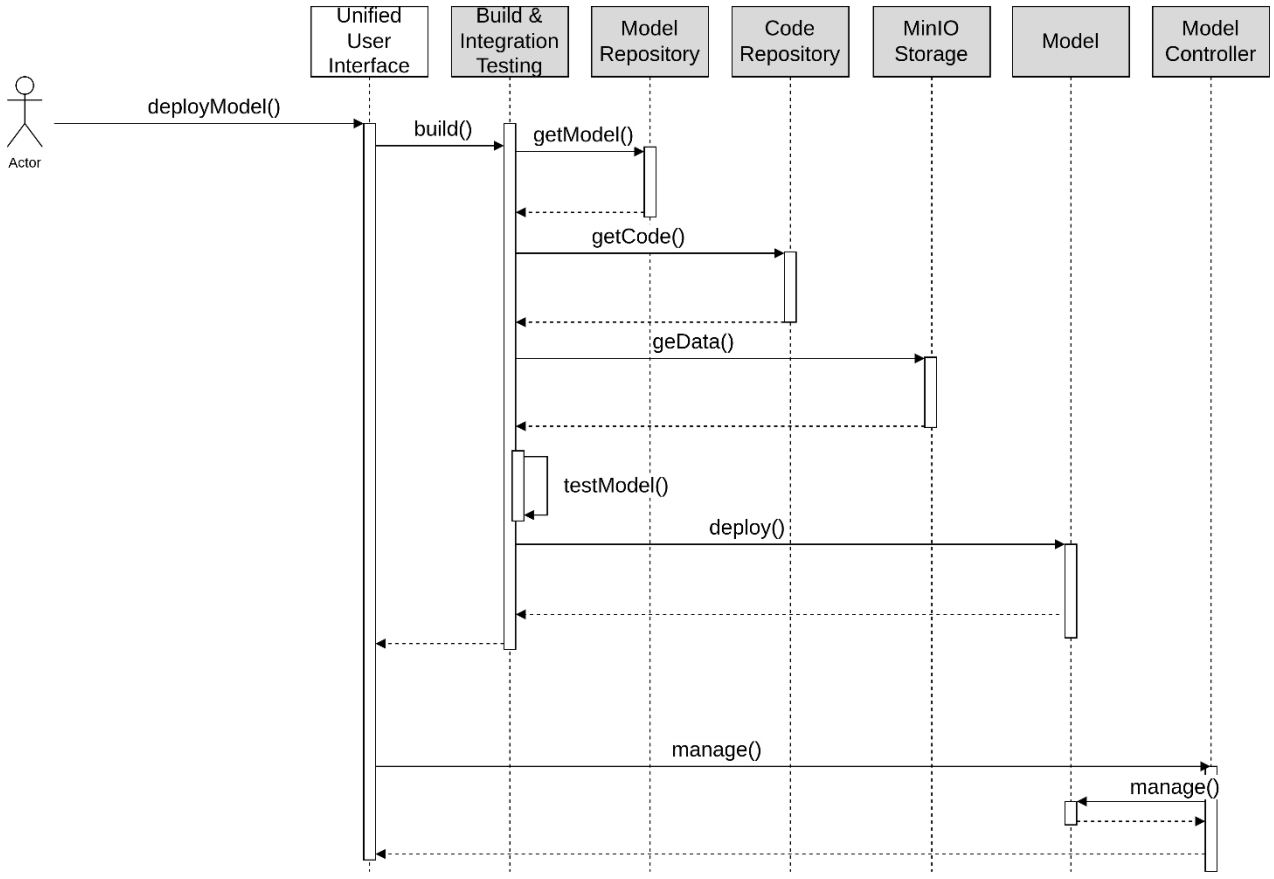
WHERE	WHAT	WHY	Run/Design Time	Priority
	Use Case 1: City Scape			
N/A				
	Use Case 2: Green Deal			
N/A				
	Use Case 3: Agriculture			
N/A				

11.1.10 Workflows

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






Unified User Interface
Build & Integration Testing
Model Repository
Code Repository
MinIO Storage
Model
Model Controller
Actor
deployModel()
build()
getModel()
getCode()
geData()
testModel()
deploy()
manage()
manage()

11.1.11 Role, Resourcing, and Milestones

Sub-component	Main Activity	M18	M24	M30	M36
Exploration & Validation	Upgrade to latest version and small adaptations				
Wrangling	Upgrade to latest version and small adaptations				
MinIO	Upgrade to latest version and small adaptations				
Model Engineering	Upgrade to latest version and small adaptations				
Model Evaluation	Upgrade to latest version and small adaptations				
Model Packaging	Develop the component				
Model Repository	Develop the component				
Code Repository	Upgrade to latest version and small adaptations				
Build & Integration Testing	Develop the component				
Monitoring Logging, & Alerts	Upgrade to latest version and small adaptations, develop new components dashboards				
Model Controller	Develop the component				
Unified User Interface	Extend current user interface to use the new features				
Data Inspector Integration	Test integration with Data Inspector				
Additional Libraries	Deploy identified libraries				
Bug Fixing and Maintenance	Maintenance after release				
Table Total/DOA Task Total/Resilience	**Comments:**				

11.1.12 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
Integrate third-party libraries	Libraries identified in T4.2 as useful for anomaly detection must be included in the toolkit.	Once libraries are identified, they will be included in the software tools.	Internal T4.2 dependency.
Containerisation of deployed models	Examine if containerisation module can be used for that even if this component is designed to package a model definition along with all the runtime and library software required to execute it and CONT is aimed at modules; This component will provide descriptors for deployment that are compatible with the CONT module.	Discussion with CONT	INDRA/ICE