EU-CIRCLE

A pan-European framework for strengthening Critical Infrastructure resilience to climate change

D7.3 Administration and User Manuals For SimICI System V1

Contractual Delivery Date: 31/01/2017	**Actual Delivery Date**: 30/08/2017
Type: Report	**Version**: v1.0

Dissemination Level: Public Deliverable

Statement

This document accompanies the first software release of the SimICI reference testbed and provides guidance to Administrators and Users of the Simulated Interconnected (Critical) Infrastructure (SimICI) software system. It should be noted that this document was previously suspended pending the definition of extensions to support the ANDI methodology. The ANDI extensions mean that there is a significant difference between the first release and the planned end state of SimICI. This deliverable is therefore released as a contractual obligation and will be superseded in full by the end of the project.

EU-CIRCLE is a project that has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 653824. Please see http://www.eu-circle.eu/for more information.

Preparation Slip			
	Name	Partner	Date
From	David Prior	XUV	25/08/2017
Reviewer	STWS	STWS	30/08/2017
Reviewer			
For delivery	David Prior	XUV	30/08/2017

Document Log			
Issue	Date	Comment	Author / Organization
V0.1	05/12/2016	Document Structure and Indexes	D. Prior / XUV
V0.2	16/01/2017	Base Content Completed	D. Prior / XUV
V0.3	06/02/2017	Document Suspended for MTR	D. Prior / XUV
V0.4	13/03/2017	Document Reopened post MTR	D. Prior / XUV
V0.5	21/04/2017	Additional Content Added	D. Prior / XUV
V0.6	28/04/2017	Document Suspended for Development of ANDI Methodology	D. Prior / XUV
V0.7	30/06/2017	Internal Review for ANDI Changes	R. Vaughan / XUV, D. Prior / XUV
V0.8	14/07/2017	Document Reopened to meet Obligations for Deliverables	D. Prior / XUV
V0.9	25/08/2017	Released for Internal Review	R. Vaughan / XUV, D. Prior / XUV
V1.0	30/08/2017	Final Version	D. Prior / XUV

List of Abbreviations	
Term	Description
2D	Two Dimensional
3D	Three Dimensional
ANDI	Assets, Networks, Dependencies, and Interconnections
AOI	Area of Interest
API	Application Programming Interface
CI	Critical Infrastructure
CIRP	Critical Infrastructure Resilience Platform
CRS	Coordinate Reference System (Alternative term for SRS, see below)
D	Deliverable
EPSG	European Petroleum Survey Group
GIS	Geographic Information System
GML	Geography Markup Language
GPS	Global Positioning System
HTML	HyperText Markup Language
HTTP(S)	HyperText Transfer Protocol (Secure)
ITRS	International Terrestrial Reference System
JSON	JavaScript Object Notation
KML	Keyhole Markup Language
NoN	Network of Networks
OGC	Open Geospatial Consortium
REST	REpresentational State Transfer
SDK	Software Development Kit
SimICI	Simulated Network of Interconnected Critical Infrastructures
SRS	Spatial Reference System
URI	Uniform Resource Identifier (Like a URL but does not have to reference an actual thing)
URL	Uniform Resource Locator (An address string that references a thing on the Internet)

UI	User Interface
VDS	Virtual Data Set
WebDAV	Web Distributed Authoring and Versioning
WCS	Web Coverage Service
WGS	World Geodetic System
WFS(-T)	Web Feature Service (-Transactional)
WMS(-T)	Web Mapping Service (-Transactional)
WPS	Web Processing Service
XML	eXtensible Markup Language
XUV	Xuvasi Ltd

Executive Summary

EU-CIRCLE‘s scope is to derive an innovative framework supporting resilience of the interconnected European Critical Infrastructure to climate pressures as the increasingly dependent, interdependent and interconnected nature of CI networks exposes previously unseen risks, new vulnerabilities, and opportunities for disruption of those networks.

This document accompanies the first release of the integrated SimICI environment that provides an innovative collaborative environment, delivered through standard web application interfaces, within which technical and non-technical users may collaborate in order to explore scenarios; to investigate, in conjunction with CIRP, and test intervention strategies, and to define additional analysis prototypes that will help support the resilience of the interconnected European Critical Infrastructure.

SimICI has been designed and implemented as a ‘single-window’ web-based application that provides a series of intuitive interfaces through which the EU-CIRCLE ‘virtual city’ dataset and associated hazards and propagation models may be manipulated. SimICI integrates geospatial data services, a workflow engine supporting impact and intervention strategy assessment, and the ability to rapidly prototype new analyses for the EU-CIRCLE CIRP platform. SimICI is, and will continue to be, populated with the data comprising the EU-CIRCLE ‘virtual city’ dataset and other assets, hazards, impacts, and propagation models as arise from the EU-CIRCLE project as a whole.

The approach to SimICI has been to leverage open-source software and open standards, data formats, and protocols wherever possible. This obviates third-party licence costs, maximises the potential to engage with the open-source community, and provides maximum flexibility and extensibility in the capability provided, within and beyond the extent of the EU-CIRCLE project.

SimICI has not only been designed as an integrated capability in itself but also, and in keeping with the testbed objective for SimICI within EU-CIRCLE, as a capability that can be readily integrated with the CIRP platform as and when required.

This document provides the Administration and User Manuals for SimICI version 1.0 as described in our previous deliverable D7.1.

Contents

EXECUTIVE SUMMARY **4**

CONTENTS **5**

FIGURES **6**

1 INTRODUCTION **7**

1.1 SimICI Endpoints 7

1.2 SimICI GIS Feature Data Pipeline 7

1.3 Document Structure 7

2 ADMINISTRATOR / DEVELOPER GUIDE **8**

2.1 Uploading a New Feature Shapefile 9

2.2 Publishing a New Feature Layer from a Shapefile 11

2.3 Confirming Publication and Validation 15

2.4 Additional Administrator / Developer Actions 21

3 USER GUIDE **22**

3.1 The SimICI Web Application 22

3.2 The SimICI Engine 25

3.3 The Impact! Client 35

3.4 Putting it all Together 36

4 IN CLOSING **40**

4.1 Coda to D7.3 40

Figures

Figure 1: SimICI GIS Feature Data Pipeline 7
Figure 2: Copy New CI Shapefile to SimICI 9
Figure 3: Make New CI Shapefile Readable 10
Figure 4: Layer Manager for Feature Data 11
Figure 5: Select Datastore for the New Layer 12
Figure 6: Publishing the New Layer from the Shapefile 12
Figure 7: Set the Descriptive Tags for the New Layer 13
Figure 8: Set and Compute the CRS for the New Layer 14
Figure 9: Search for the New Layer - Confirm Publication 15
Figure 10: Validate the New Layer with Layer Preview 15
Figure 11: Validating the New Layer with Internal Map Preview 16
Figure 12: Portal Export of New Layer as a Shapefile 17
Figure 13: View Portal Exported Shapefile in QGIS 18
Figure 14: API Call to Export New Layer as Shapefile 18
Figure 15: View API Exported Shapefile in QGIS 19
Figure 16: Other Formats Example - QGIS View of GeoJSON Export 20
Figure 17: Exploitation of the New Layer in SimICI Mapterface 21
Figure 18: SimICI Web Application - Frontpage 22
Figure 19: SimICI Mapterface View 23
Figure 20: SimICI 3D View 24
Figure 21: Viewing the Rethymno VDS Oil Network 25
Figure 22: The SimICI Engine Cascade Designer 26
Figure 23: Adding a Node to a Cascade 27
Figure 24: Connecting Node Inputs and Outputs 28
Figure 25: A Valid Connection 28
Figure 26: Sending Node Updates to the SimICI Mapterface 29
Figure 27: Adding a Process Node to Calculate Metrics 30
Figure 28: Connecting the Process across the Network 31
Figure 29: Sending Process Updates to the SimICI Mapterface 32
Figure 30: Allowing External Influence at Run Time 33
Figure 31: Saving and Running a Cascade 33
Figure 32: Status of the Running Cascade 34
Figure 33: A Killed Cascade can be Restarted or Edited 34
Figure 34: The Impact! Client 35
Figure 35: Applying an Impact 36
Figure 36: Cascade Status with the Impact Applied 37
Figure 37: Propagation of Impact to SimICI Mapterface 37
Figure 38: Applying a Second, Cumulative Impact 38
Figure 39: Cascade Status with Both Impacts Applied 39
Figure 40: Propagation of Cumulative Impacts to SimICI Mapterface 39

1 Introduction

This document comprises deliverable D7.3 of the EU-CIRCLE project; providing the Administration and User Manuals for the SimICI System at version 1.0. For a description of the SimICI system, please refer to previous EU-CIRCLE deliverable D7.1.

1.1 SimICI Endpoints

The SimICI System is deployed as a set of web applications encapsulated by the single server URL: http://simici.eu-circle.eu. Specific endpoints within the SimICI System, as accessed from any modern web browser, are:

- Main SimICI web application: http://simici.eu-circle.eu
- SimICI Engine web application: http://simici.eu-circle.eu:3000
- SimICI Impact! client web application: http://simici.eu-circle.eu:3005
- SimICI Geoserver Portal: http://simici.eu-circle.eu:8080/geoserver

Absent a single central authentication service for the EU-CIRCLE project, each web application is protected by it's own authentication mechanism. User accounts for each application are currently controlled by Xuvasi Ltd but are expected to be automated by the end of the EU-CIRCLE project.

1.2 SimICI GIS Feature Data Pipeline

System administration in SimICI, as opposed to deployment and configuration, is primarily concerned with the management of the GIS Feature Data Pipeline. Or, in other words, the publication in useful form of the virtual city dataset.

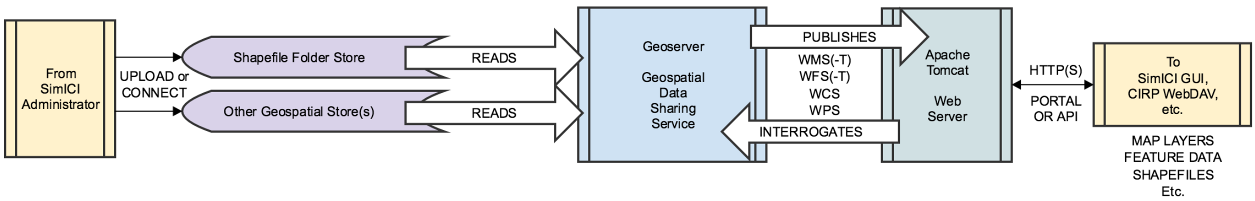


Figure 1: SimICI GIS Feature Data Pipeline

Figure 1 illustrates this pipeline and should be referenced when reading the Administration Guide.

1.3 Document Structure

The remainder of this document is structured as follows:

- Section 2 provides the Administrator / Developer Guide
- Section 3 provides the User Guide
- Section 4 provides an end note with regard to this report.

2 Administrator / Developer Guide

As noted previously, Administration in SimICI is primarily concerned with the management of the GIS Feature Data Pipeline in relation to the EU-CIRCLE VDS.

In EU-CIRCLE, features (representing assets and networks) are typically provided in Shapefile format. Per the standard Wikipedia definition[1], the Shapefile format is a digital vector storage format for storing geometric location and associated attribute information. While the term 'Shapefile' is commonly used, it is misleading since the format consists of a collection of files with a common filename prefix, stored in the same system location. The format itself comprises three mandatory files which may be supplemented with additional associated files as:

Mandatory files

- .shp — shape format; the feature geometry itself
- .shx — shape index format; a positional index of the feature geometry to allow seeking forwards and backwards quickly
- .dbf — attribute format; columnar attributes for each shape, in dBase IV format

Other files

- .prj — projection format; the coordinate system and projection information, a plain text file describing the projection using well-known text format
- .sbn and .sbx — a spatial index of the features
- .fbn and .fbx — a spatial index of the features that are read-only
- .ain and .aih — an attribute index of the active fields in a table
- .ixs — a geocoding index for read-write datasets
- .mxs — a geocoding index for read-write datasets (ODB format)
- .atx — an attribute index for the .dbf file in the form of *shapefile*.*columnname*.atx (ArcGIS 8 and later)
- .shp.xml — geospatial metadata in XML format, such as ISO 19115 or other XML schema
- .cpg — used to specify the code page (only for .dbf) for identifying the character encoding to be used
- .qix — an alternative quadtree spatial index used by MapServer and GDAL/OGR software

As may be expected, therefore, the process of publishing a new feature set from SimICI is concerned with ingesting the feature source Shapefile, which may have been produced on an entirely different system, and publishing it such that it may be consumed for use within and across the EU-CIRCLE project and elsewhere.

As Figure 1 illustrates, the key enabler to this process is the open-source Geoserver which supports interoperability through the publication of of data from any major spatial data source using open standards. The remainder of this section discusses the administration of this process within SimICI.

[1] https://en.wikipedia.org/wiki/Shapefile

2.1 Uploading a New Feature Shapefile

The first step in the Administration process is to upload the Shapefile containing the new feature data to the SimICI Geoserver. The SimICI Geoserver is deployed on a Linux server which supports any file upload mechanism that is capable of leveraging Unix secure copy and it's enablers.

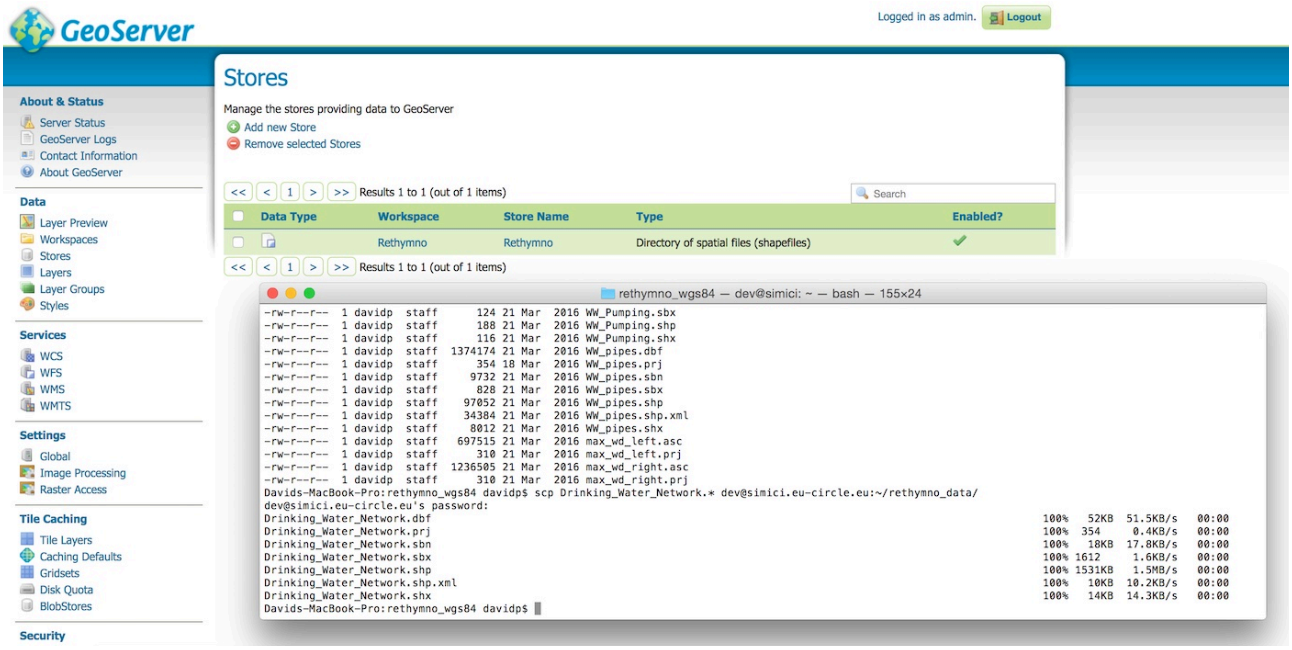


Figure 2: Copy New CI Shapefile to SimICI

Figure 2, above, shows the upload of the Rethymno VDS Drinking Water Network features from a Mac client to the SimICI Geoserver host. Host system usernames and passwords for the Administration process are managed by Xuvasi on behalf of the EU-CIRCLE project.

In this upload, the set of files comprising the Shapefile formatted data for the new feature are uploaded to the SimICI Geoserver. On the SimICI Geoserver, the folder 'HOME/rethymno_data' (for each host username) is symbolically linked to the primary Geoserver data folder. The use of the symbolic link minimizes path lengths and therefore reduces the typing burden on the Administrator.

The effect of this upload is to place the new Shapefile onto the SimICI Geoserver host and into the right location for the SimICI Geoserver to access it. Despite this, however, the newly uploaded file is not fully accessible to the SimICI Geoserver and some permissions must be applied to address this.

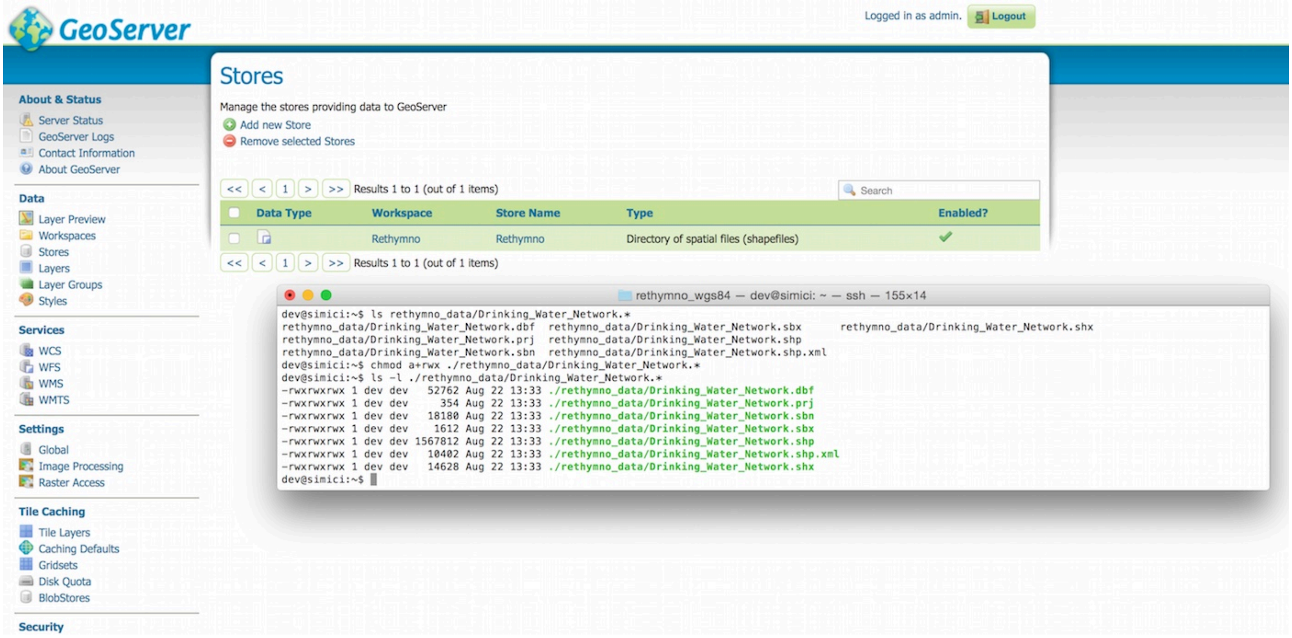


Figure 3: Make New CI Shapefile Readable

Figure 3, above, illustrates the process for applying all-access permissions to the newly uploaded Shapefile components. The Administrator must remotely connect to the SimICI Geoserver and, at the terminal prompt, perform a permissions modification that makes the Shapefile components readable, writable, and executable by all. That modification is encapsulated in the Unix command:

chmod a+rwx ./rethymno_data/SHAPEFILE_FILENAME_PREFIX.*

The command is executed from the HOME folder (the default location after a Unix user connects to a remote server) and leverages the symbolic link noted previously.

Once executed, a second Unix command can be executed to confirm that the permissions have been successfully modified:

ls –l ./rethymno_data/SHAPEFILE_FILENAME_PREFIX.*

The report string '-rwxrwxrwx' at the left hand side of the resulting listing indicates that each file has been modified to be fully accessible by all system users. This, by definition, includes the SimICI Geoserver application and, therefore, the new feature data may now be published for exploitation in EU-CIRCLE and elsewhere.

Once the modification has been confirmed, the Administrator may disconnect from the SimICI Geoserver. There are no further system level interactions as all subsequent steps in the Administration process are performed through the SimICI Geoserver UI.

2.2 Publishing a New Feature Layer from a Shapefile

The SimICI Geoserver operates against a pyramidal hierarchy comprising a:

- Workspace:
  - A container which organizes other items
  - Uniquely identified by a name and a namespace (as a URI)
  - Eg: The Rethymno workspace in SimICI organizes the Rethymno VDS
- Containing one or more Stores:
  - Connect to a data source containing raster or vector data
  - Parented to a workspace and identified by a name
  - Eg: The Rethymno store in SimICI exposes the Rethymno VDS shapefile folder source
- Containing one or more Layers:
  - Represents a collection of geographic features
  - Is drawn from a store and inherits that store's workspace
  - Eg: The Rethymno:photoVolt layer in SimICI exposes the photovoltaic assets (held in shapefile format in the Rethymno data folder) of the Rethymno VDS
- Containing one or more Features:
  - Representing a physical geographic feature and its associated attributes
  - Typically grouped by geometry type (eg: point, line, polygon) within Layer
  - Eg: Feature ID 'photoVolt.1' in the Rethymno:photoVolt layer represents the first photovoltaic asset in the Rethymno VDS with a geometry type of point.

Figure 4, below, illustrates accessing the Layers perspective in order to publish the newly uploaded Shapefile from the SimICI Geoserver portal. Once logged in at the SimICI Geoserver endpoint, click on 'Layers' in the left-hand menu bar and then on 'Add a new layer' in the main page.

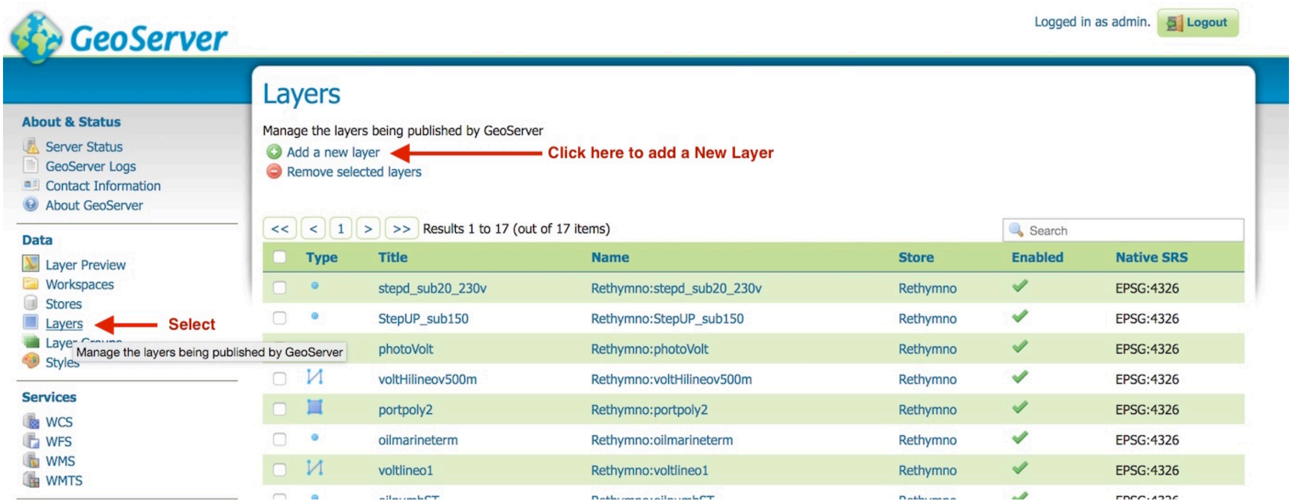


Figure 4: Layer Manager for Feature Data

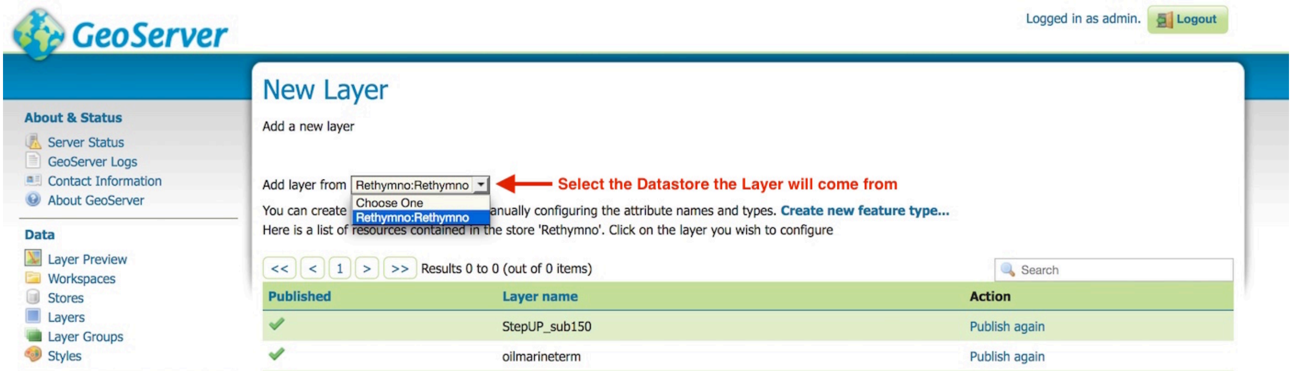


Figure 5: Select Datastore for the New Layer

To add a new layer, select the Store that the new layer will be drawn from. For the SimICI Geoserver, at release version 1, the Rethymno workspace and Store have been predefined. All new layers, for the Rethymno VDS at least, are to be published from this workspace and store.

Select 'Rethymno:Rethymno' as the source for the new layer as shown in Figure 5, above.

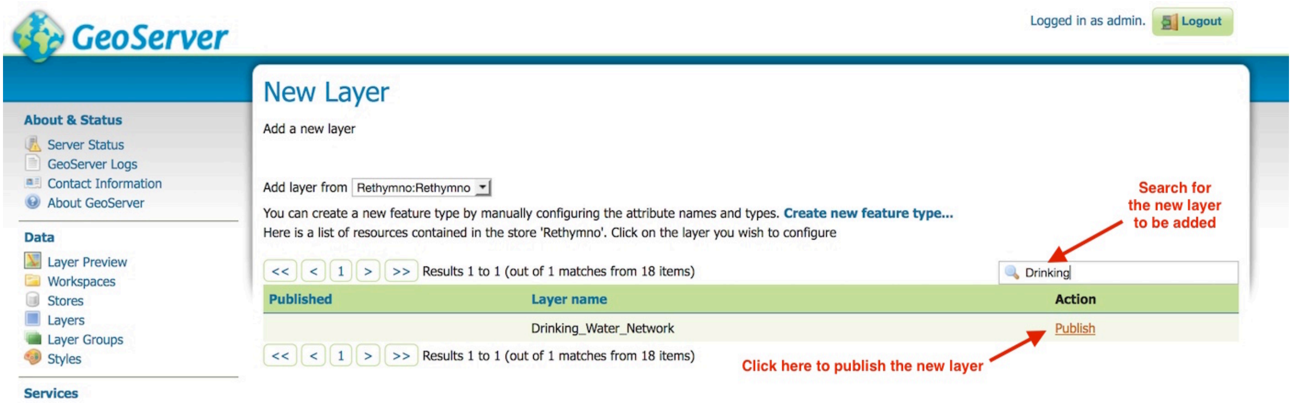


Figure 6: Publishing the New Layer from the Shapefile

Once the store is selected, a list of layers from that store will appear in the lower half of the page. To search for the layer contained in the new shapefile that was uploaded, type all or part of the SHAPEFILE_FILENAME_PREFIX into the search box. Press the enter key and the layer contained in the shapefile will be shown; with an available action of 'Publish'.

Clicking on 'Publish' will open the Edit Layer screen such that the new layer can be defined. The contents of the Edit Layer screen will vary depending on the nature of the data source used to derive the layer. For the purposes of SimICI version 1.0, the screen will allow definition of the layer based on the contents of the uploaded Shapefile.

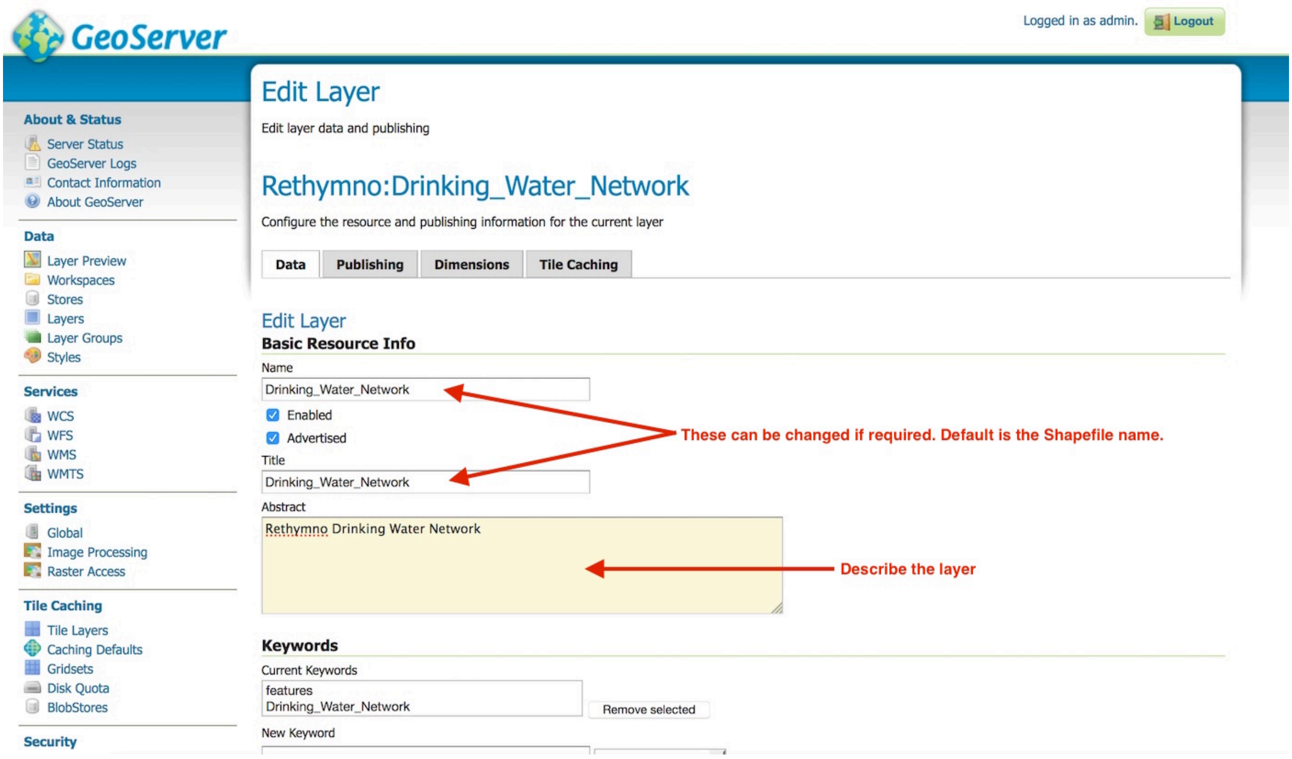


Figure 7: Set the Descriptive Tags for the New Layer

The first thing to do in the Edit Layer screen is to set the descriptive tags for the layer being generated from the contents of the uploaded Shapefile. The Name and Title tags will default to the SHAPEFILE_FILENAME_PREFIX but may be changed if required (for example, to make the tags more intuitive or relevant to the content of the layer).

An abstract may also be entered. This is intended to provide a summary or other salient details pertaining to the layer being published. Abstracts are useful where, for example, a Geoserver instance contains many layers and the abstract can be used to help a user decide which layer best suits their purposes. Abstracts are accessible via the Geoserver API: meaning that they can be presented to users in downstream software applications. It is not mandatory to enter an abstract but it is suggested that time taken at the publication stage will make layers more useful in operation.

Further down the Edit Layer screen, the Administrator must both declare (set) and compute the Co-ordinate Reference System (CRS – also known as Spatial Reference System (SRS)) for the features contained within the new layer. It is imperative that this operation is successful as the operation of SimICI – and, by extension, operations based on SimICI data - assumes the common and consistent use of the EPSG:4326 SRS. EPSG:4326 is maintained by the European Petroleum Survey Group (EPSG) and is a that refers to latitude and longitude coordinates on the WGS84 reference ellipsoid.

World Geodetic System 1984 (WGS84) is an Earth-centred, Earth-fixed terrestrial reference system and geodetic datum. WGS84 is based on a consistent set of constants and model parameters that describe the Earth's size, shape, gravity, and geomagnetic fields. WGS84 is the standard US Department of Defense definition of a global reference system for geospatial information and is the reference system for the Global Positioning System (GPS). WGS84 is compatible with the International Terrestrial Reference System (ITRS).

In short, successful re-projection of feature data to WGS84 in SimICI serves to minimize the amount of geometry management that must be performed by downstream consumers of SimICI generated data. In many cases, the use of the WGS84 datum means that zero geometry management is required.

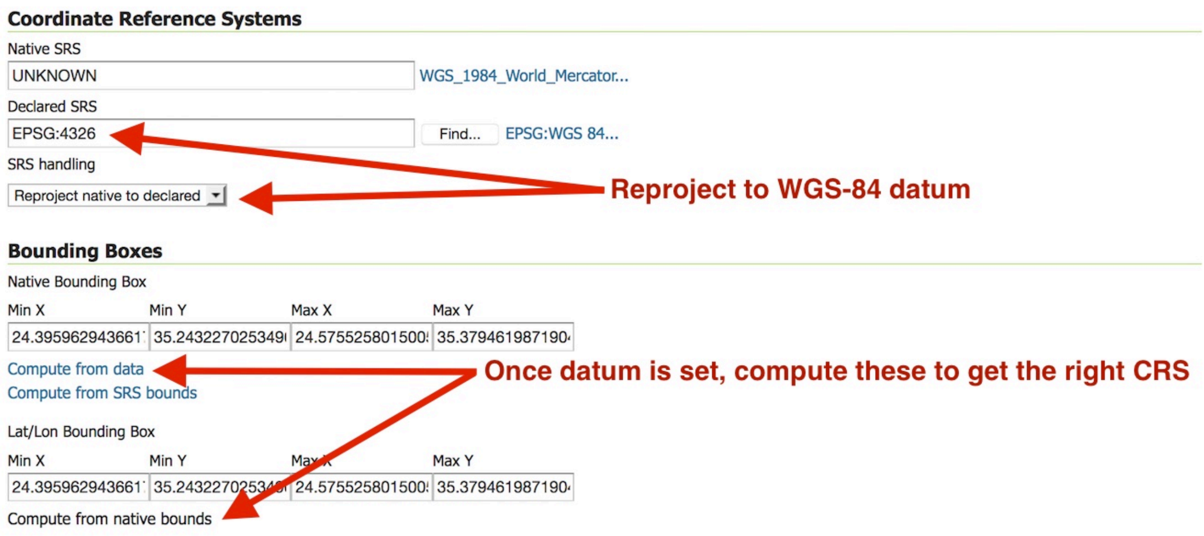


Figure 8: Set and Compute the CRS for the New Layer

When setting and computing the CRS for the new layer, the SimICI Geoserver will endeavour to detect the native SRS of the uploaded Shapefile. Typically, this will be done by reference to the SHAPEFILE_FILENAME_PREFIX.prj file uploaded as part of the Shapefile format. It should be noted, however, that a .prj file may not be provided. In this case, either a .prj file will need to be manually generated (typically through editing of an existing .prj file for the same data store) or the Administrator must go back to the source of the Shapefile and request the .prj file be generated from the source's GIS system. It is possible to try to force the projection to EPSG:4326 **without** a .prj file but the results may be unpredictable.

Once the native projection is available (denoted by the blue text to the right of the 'Native SRS' box – despite the box stating UNKNOWN), set the 'Declared SRS' to 'EPSG:4326'. Then set the 'SRS Handling' option to 'Reproject native to declared'. This will instruct that the SimICI Geoserver reprojects feature geometry to the WGS84 datum.

In the 'Bounding Boxes' section, under the 'Native Bounding Box' entries, click 'Compute from data'. The 'Native Bounding Box' fields will then fill with computed entries. For the 'Lat/Lon Bounding Box', click 'Compute from Native Bounds'. The 'Lat/Lon Bounding Box' fields will again fill with computed entries. The two sets of computed entries should be the same and should be formed as decimals. If the computed entries are not formed as decimals then, typically, something has gone wrong in the re-projection process and the native SRS of the uploaded Shapefile should be checked and confirmed with the source of the data.

If all is well, scroll down to the end of the page and click 'Save'. If the save is successful, the SimICI Geoserver UI will return to the Layers screen.

2.3 Confirming Publication and Validation

Once published, the new layer must be confirmed and validated before use. This step ensures that the Shapefile was uploaded correctly, that it contained viable data, and that the relevant spatial (re-) projections have been successful.

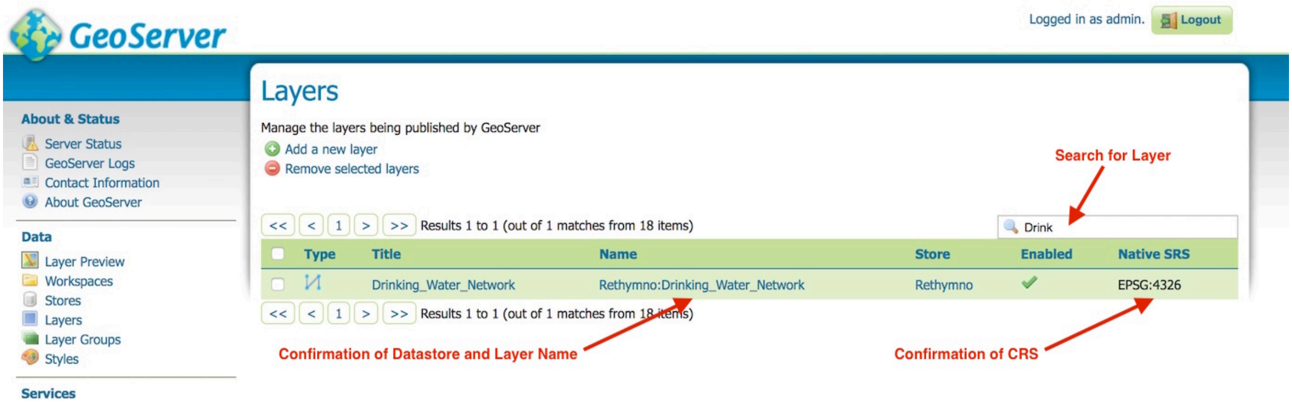


Figure 9: Search for the New Layer - Confirm Publication

The first step is to confirm publication. In the Layers screen, enter all or part of the newly published Layer's name and press 'enter'. The list will collapse to only show the new Layer. This is shown in Figure 9, above, where various other information about the layer can be confirmed. In Figure 9, the geometry type is confirmed as a (poly)line, the position of the layer in the Rethymno workspace can be seen, and the SRS in use can be confirmed.

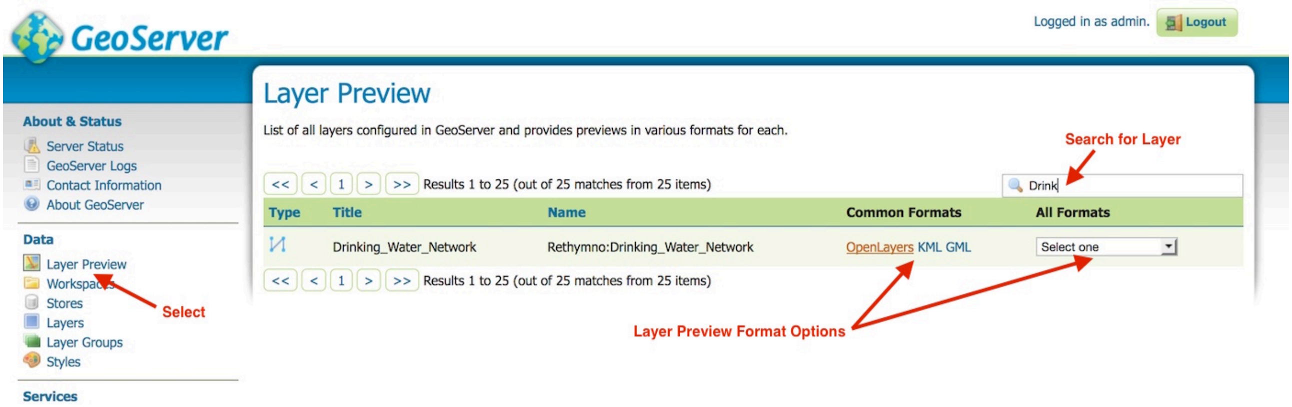


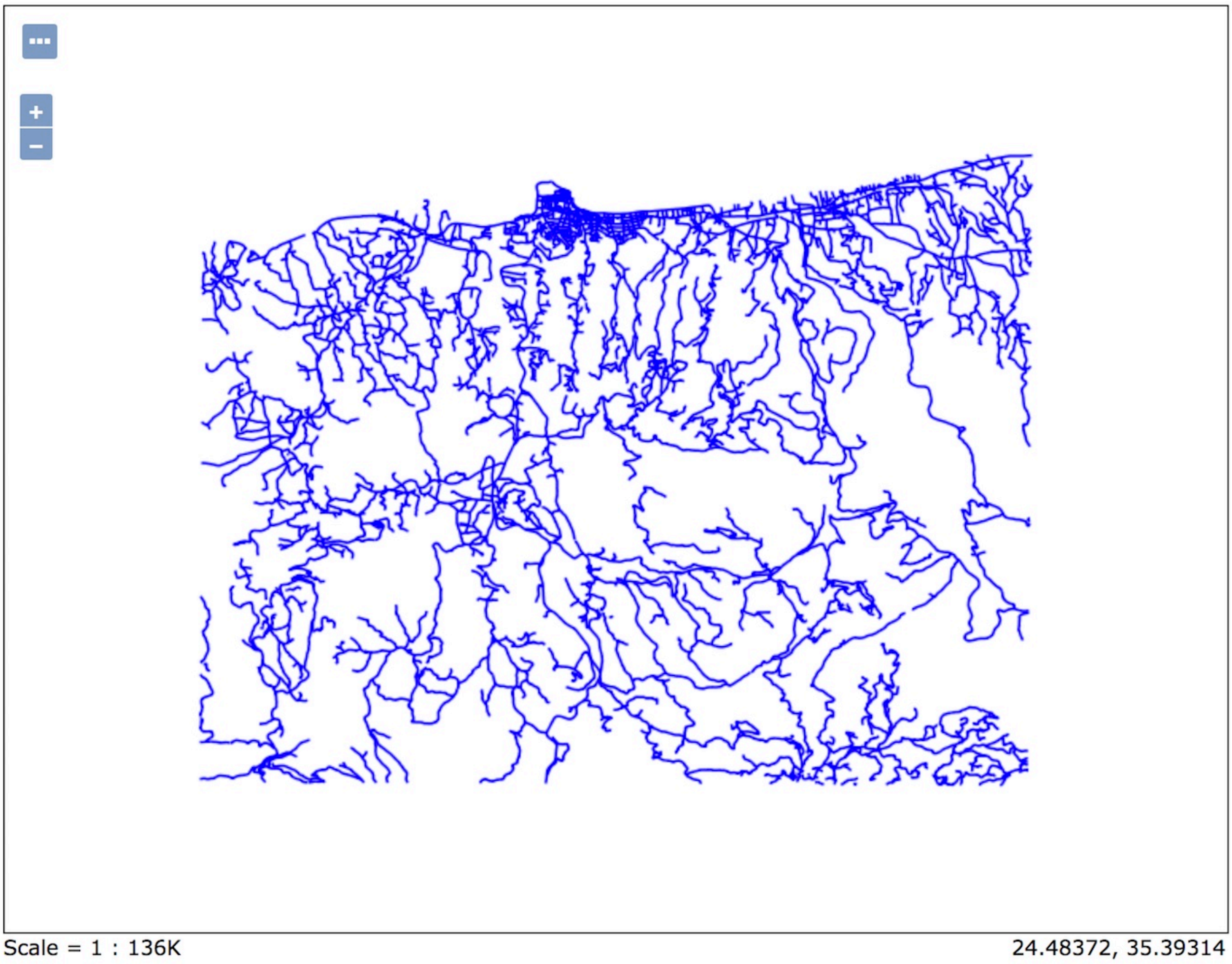
Figure 10: Validate the New Layer with Layer Preview

Selecting 'Layer Preview' from the left hand menu allows the content of the new Layer to be validated. This guide labours the point here in order to illustrate the extent of layer services and data retrieval provided by the SimICI Geoserver. In practice, assuming familiarity with the ground truth that the data represents, validation through the internal viewer is sufficient.

Figure 10, above, shows the 'Layer Preview' screen filtered, through search as before, to the newly uploaded Layer. At the right hand side of the Layer entry are links to Layer Previews in three common formats plus a drop down selector containing all formats as enabled in the SimICI Geoserver instance.

The three common formats are OpenLayers, Keyhole Markup Language (KML – Google Earth, for example), and Geography Markup Language (GML – an eXtensible Markup Language (XML) format maintained by the Open Geospatial Consortium (OGC)).

The OpenLayers format essentially generates a very lightweight mapping client with a single layer – the one to be previewed - on display. Click on 'OpenLayers' and a screen similar to that in Figure 11, below, will open. This screen renders the new Layer for visual confirmation and, if that rendering is successful, confirms that the new Layer has been published correctly.



Drinking_Water_Network

fid	Id	Shape_Leng
Drinking_Water_Network.42	0	106.643893638
Drinking_Water_Network.43	0	157.243591769

Figure 11: Validating the New Layer with Internal Map Preview

The OpenLayers preview map can be zoomed (use the +/- buttons in the top left) and panned (click, hold, and drag) to explore the extents of the layer. Clicking within the map will populate the table below the map with details of any specific features, including associated attributes, at the clicked location. Again, this confirms that publication has been successful and that the original Shapefile data is available from the SimICI Geoserver.

For the exploitation of SimICI data, however, the primary requirement is that geospatial data created, held, and managed within SimICI should be available for use within the EU-CIRCLE CIRP platform. As noted in D7.1 previously, the data integration mechanism between SimICI and CIRP requires the publication of a Shapefile from SimICI to a Web Distributed Authoring and Versioning (WebDAV) location that CIRP can access[2]. It is therefore necessary to validate the output of a Shapefile from the SimICI Geoserver.

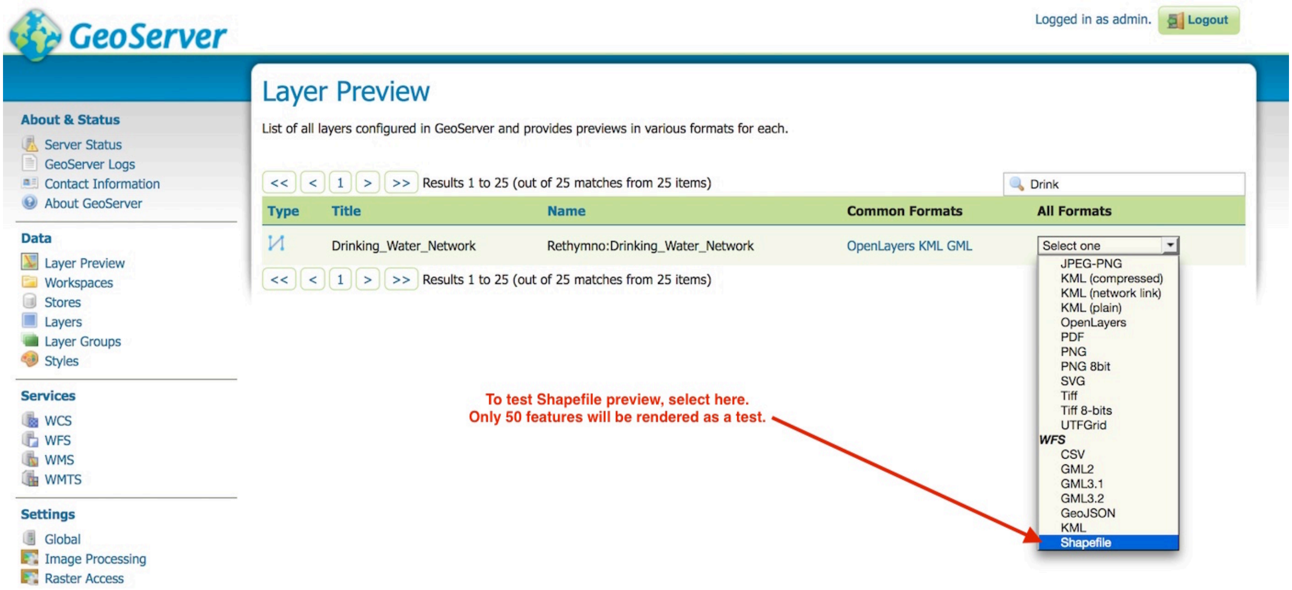


Figure 12: Portal Export of New Layer as a Shapefile

In the Layer Preview screen, dropping down the 'All Formats' selector exposes the option to export the layer through the Web Feature Service (WFS) Shapefile format mechanism. It should be noted that, as a preview function, only the first 50 features of any layer will be exported through this preview function.

Selecting the Shapefile option will cause the SimICI Geoserver to render the first 50 features of the Layer into Shapefile format, zip the resulting files into a single file, and then instruct the Administrator's web browser to open the file. As with any similar call, the web browser will respond with a dialog box asking whether the Administrator wishes to open or save the file.

Save the downloaded zip file to a useful location and unzip it. The SHAPEFILE_FILENAME_PREFIX.shp file extracted from the downloaded zip file can now be opened in a GIS software application in order to verify it's contents.

[2] The obvious question here is: why not simply put the Shapefile into CIRP in the first instance and 'cut out the middle-man'? The simple answer is that the EU-CIRCLE VDS is being provided as an open data output and, therefore, should be able to be used, leveraged, and exploited by systems other than CIRP. Putting all the EU-CIRCLE data into Shapefiles on a WebDAV would meet the brief but is not particularly organised or useful. For reasons of open-standards and open-source supports, the use of Geoserver within SimICI not only meets the open data brief but also provides a mechanism that allows the widest possible manual and programmatic use of CI data generated by the EU-CIRCLE project.

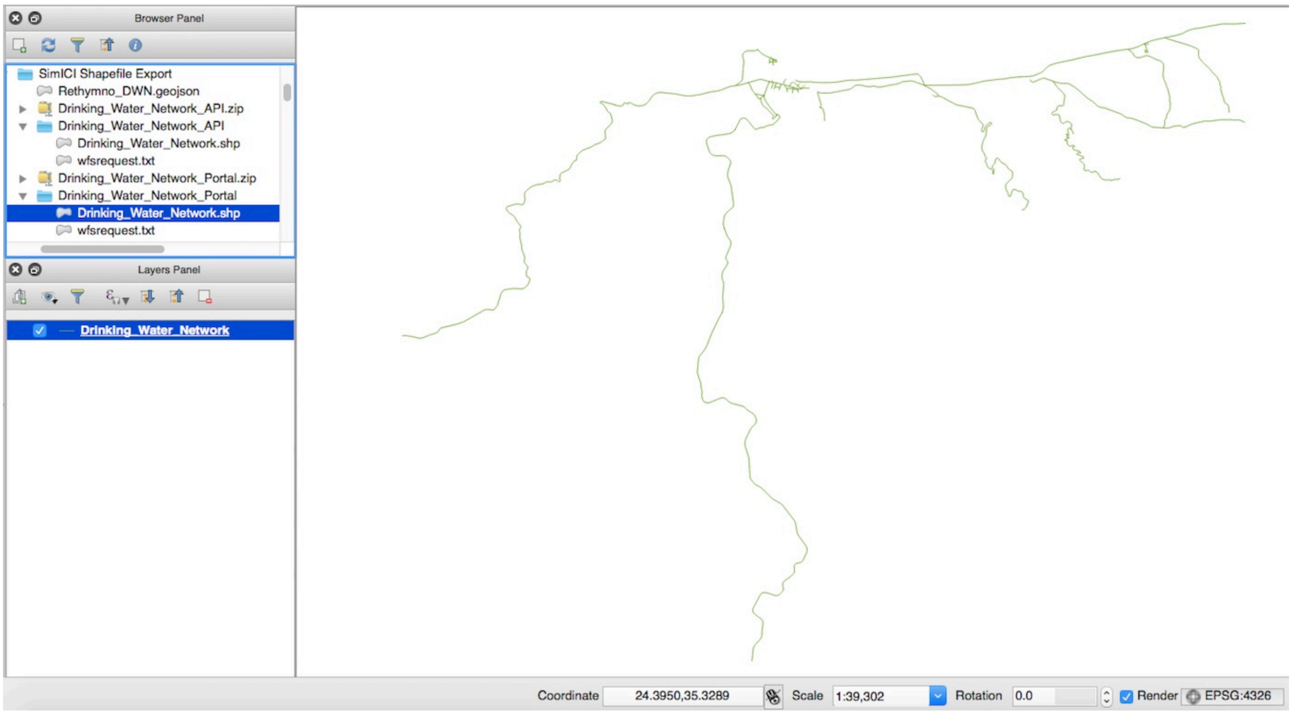


Figure 13: View Portal Exported Shapefile in QGIS

Figure 13, above, shows the data subset for the previewed layer as rendered from the preview Shapefile in the QGIS GIS software application. Note that this differs from the extent shown in Figure 11. This is because a Shapefile preview contains only a subset of the features within any given layer.

The previewed Shapefile, however, looks to be correctly projected so a full Shapefile validation is the next logical step. Figure 14, below, illustrates the method to do this.

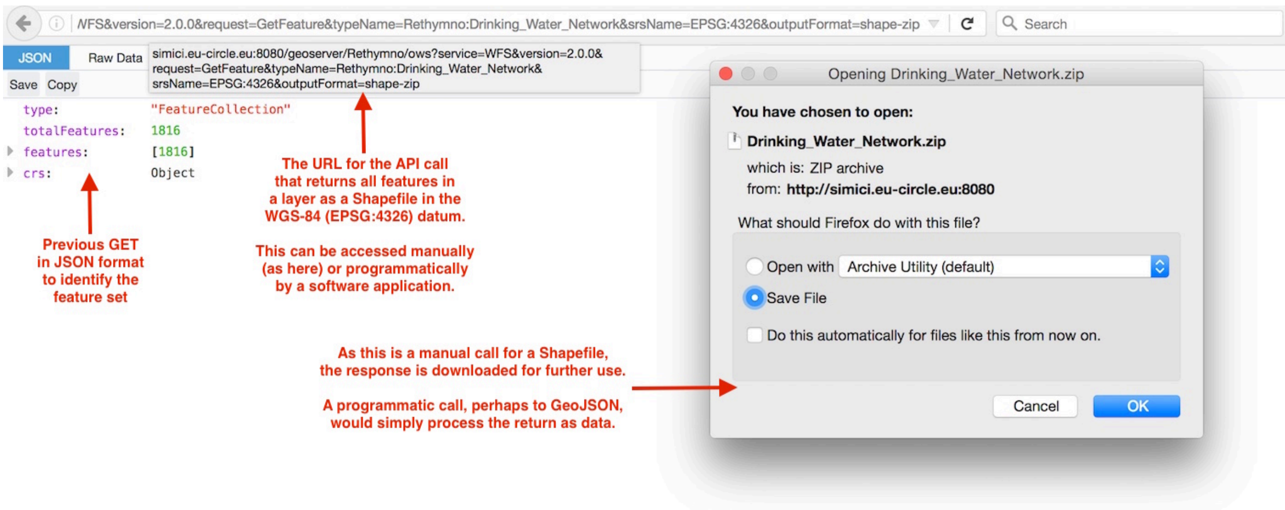


Figure 14: API Call to Export New Layer as Shapefile

The key difference between the previous validation test and this is that this test is taking place from a standard external web browser and **not** from within the SimICI Geoserver portal UI. This method is, essentially, a manual interaction with the SimICI Geoserver API and serves to illustrate not only the accessibility of the data held within SimICI but also the programmatic techniques through which that data may be accessed for exploitation downstream of SimICI.

In Figure 14, a standard HTTP GET call is made to the URL: http://simici.eu-circle.eu:8080/geoserver/Rethymno/ows?service=WFS&version=2.0.0&request=GetFeature&typeName=Rethymno:Drinking_Water_Network&srsName=EPSG:4326&outputFormat=shape-zip

That URL needs to be broken down to constituent parts for more clarity:

1. http://simici.eu-circle.eu:8080/geoserver - Go to the SimICI server
2. /Rethymno/ows - Use the Rethymno workspace
3. ?service=WFS&version=2.0.0 - Talk to the WFSv2 service
4. &request=GetFeature - Ask the service to return feature data
5. &typeName=Rethymno:Drinking_Water_Network - For the named Layer
6. &srsName=EPSG:4326 - Using the WGS84 datum
7. &outputFormat=shape-zip - In zipped Shapefile format

Again, as this is a manual request, the returned Shapefile should be downloaded, saved, and unzipped for validation in a GIS software application. Figure 15 shows the result from the above rendered in the QGIS GSI software application. This time, with all data retrieved, the extent matches that seen in Figure 11.

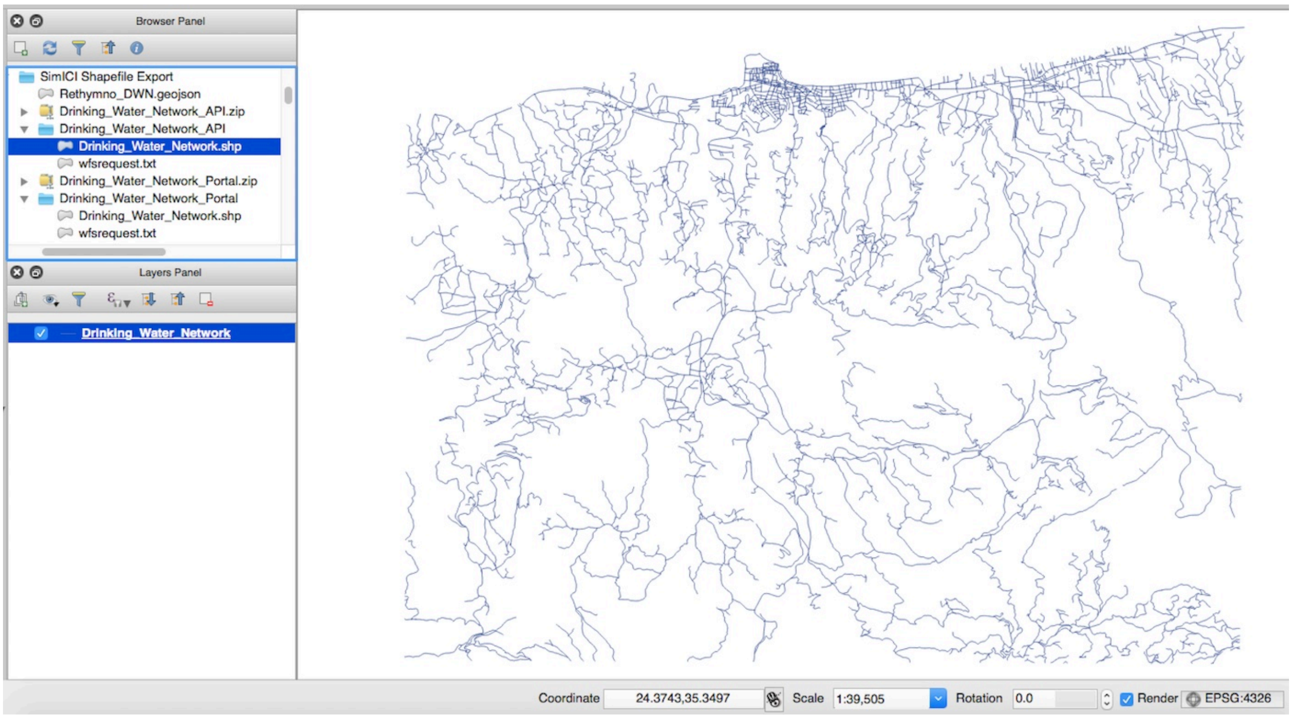


Figure 15: View API Exported Shapefile in QGIS

Figure 14 references the use of GeoJSON as a format: firstly as a pre-call to the above that identifies that there are 1,816 features in the Layer under validation and, secondly, as a potential format for use in programmatic interactions with the SimICI Geoserver.

In order to obtain GeoJSON data from the SimICI Geoserver, the API call is structured in exactly the same manner as that shown, and broken down, previously. The only difference is a parameter change from:

&outputFormat=shape-zip - In zipped Shapefile format

to:

&outputFormat=application/json - In GeoJSON text format

Making the modified request manually allows for the GeoJSON data to be downloaded for validation in a GIS application as with the Shapefile downloads previously. Figure 16, below, shows the GeoJSON data rendered in the QGIS GIS application software. It can be clearly seen that the render is identical to that shown in Figure 15, above.

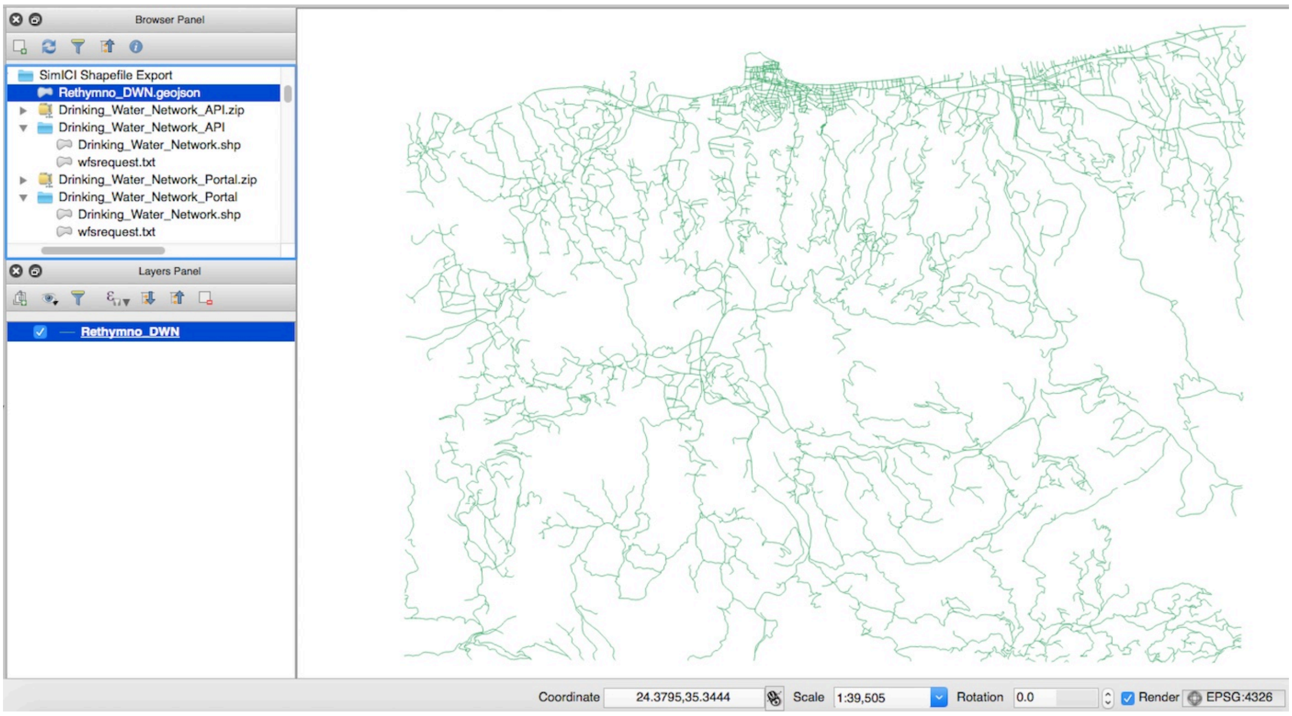


Figure 16: Other Formats Example - QGIS View of GeoJSON Export

For programmatic exploitation outside of EU-CIRCLE (or even within EU-CIRCLE but outside of CIRP), API calls – similar to the GeoJSON example above - can be made to the SimICI Geoserver and the resulting responses processed by handlers in the exploiting application software.

Indeed, it should be noted that it is exactly this mechanism which is employed by the main SimICI web application to permit user interaction with the EU-CIRCLE VDS.

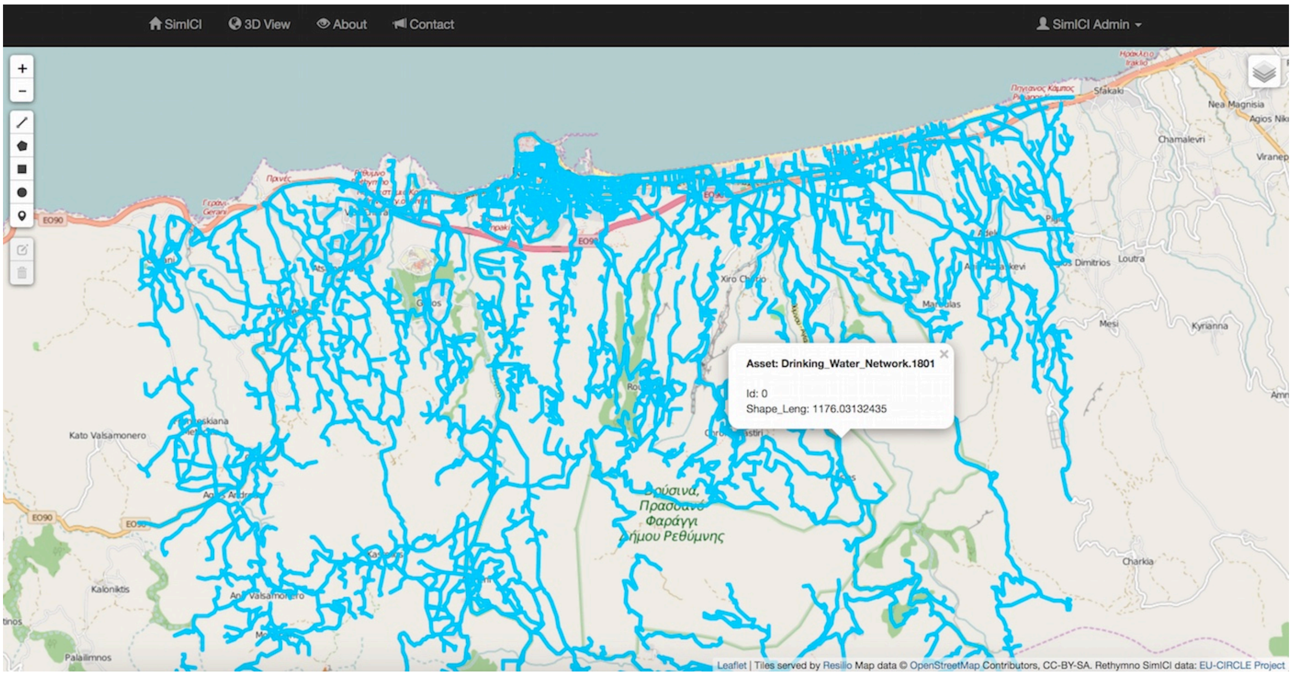


Figure 17: Exploitation of the New Layer in SimICI Mapterface

The final validation of the publication process is to confirm that the new Layer appears in the main screen, the Mapterface, of the SimICI web application. The SimICI web application looks up the list of layers available from the Rethymno store in the Rethymno workspace on the SimICI Geoserver and automatically populates the layer picker with those layers.

The default action of the SimICI Mapterface is to automatically render every layer it finds. Turning off all layers except for the newly uploaded layer, as shown in Figure 17 above, will provide the opportunity to confirm that the new layer is correctly projected onto the base map in the SimICI Mapterface.

2.4 Additional Administrator / Developer Actions

For full exploitation of the new Layer in SimICI – for example, through the stimulus and exercise of assets contained within the Layer in the SimICI Engine and Impact! Client (refer to the User Guide for more details of these components) – it is necessary to develop SimICI Engine nodes representing the asset class.

A SimICI Engine node is a software representation of an asset class that defines its attributes, inputs, and outputs. In addition, a SimICI Engine node encapsulates the known behavior of the asset class when one or more of its inputs is varied. Varying an input creates stimulus to a behavior which, in turn, varies one or more outputs from the node. When nodes are chained together in the SimICI Engine, a stimulus applied to one node has an obvious cascading affect on the node(s) downstream from it. This provides support for cascading effects both within and across CI networks.

SimICI Engine nodes are written against a Software Development Kit (SDK) providing a structured format for node software in the Java programming language. The SDK is available on request from Xuvasi Ltd.

3 User Guide

The User Guide for the SimICI System, Version 1.0, is concerned with the operational use of the SimICI software[3], leveraging data made available by an Administrator through the SimICI Geoserver. As discussed in the Adminstration Guide previously, the SimICI Geoserver contains the EU-CIRCLE Virtual Data Set (VDS).

The SimICI applications and processes described in this section represent one exploitation path for the EU-CIRCLE VDS. The data level integration of the EU-CIRCLE VDS with EU-CIRCLE CIRP represents another exploitation path. Through the use of open standards, loose-coupling via APIs, and open data formats, it is fully anticipated that other client applications will look to exploit the EU-CIRCLE VDS.

For now, however, this User Guide is focused on the SimICI System as released at Version 1.0, demonstrated, and described in EU-CIRCLE deliverable D7.1 previously.

3.1 The SimICI Web Application

Accessing the main SimICI web application presents the user with the SimICI front page as shown in Figure 18, below.

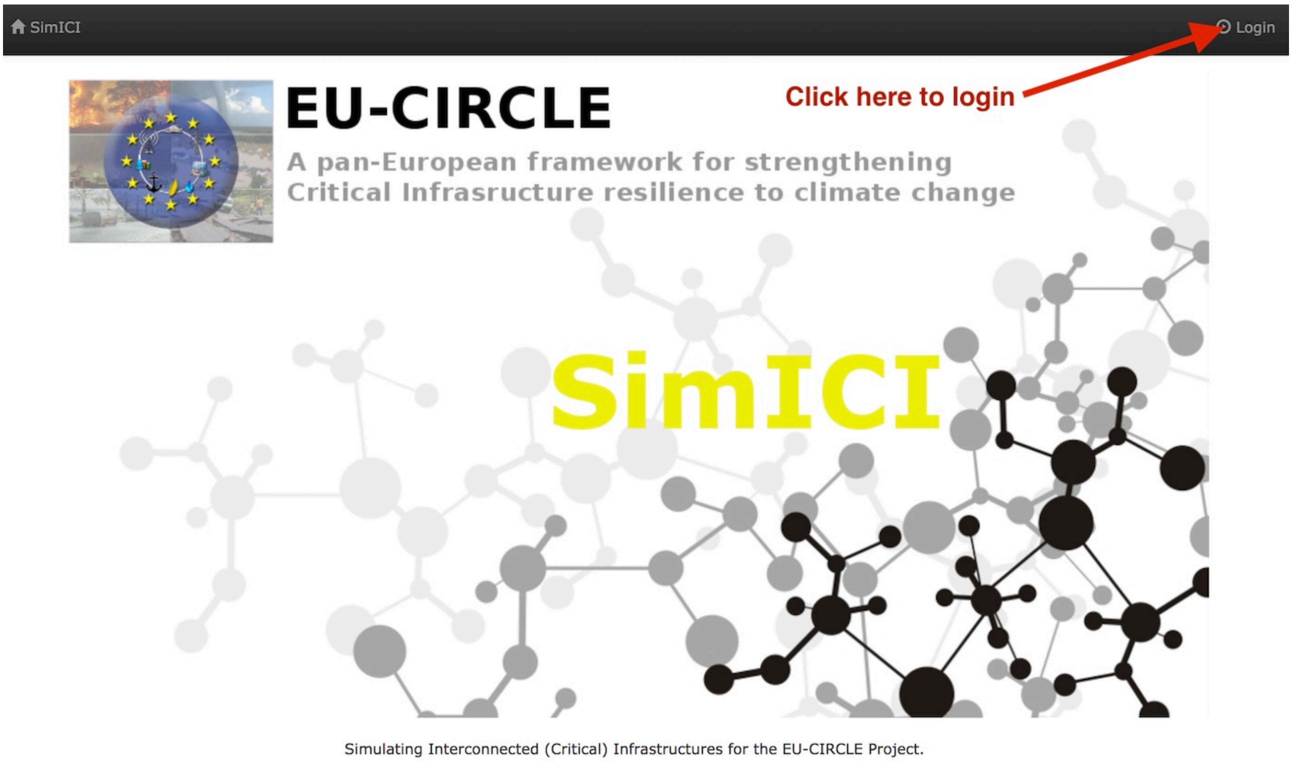


Figure 18: SimICI Web Application - Frontpage

Clicking the 'Login' link at the top-right opens a user authentication screen (email and password entry) for the user to login to the SimICI web application. User accounts are currently controlled by Xuvasi Ltd on

[3] Endpoints for the different web applications within the SimICI system are declared in Section 1.1 of this document.

behalf of the EU-CIRCLE project but this function is expected to be automated before the end of the project. A successful login results in the main SimICI application page as shown in Figure 19, below.

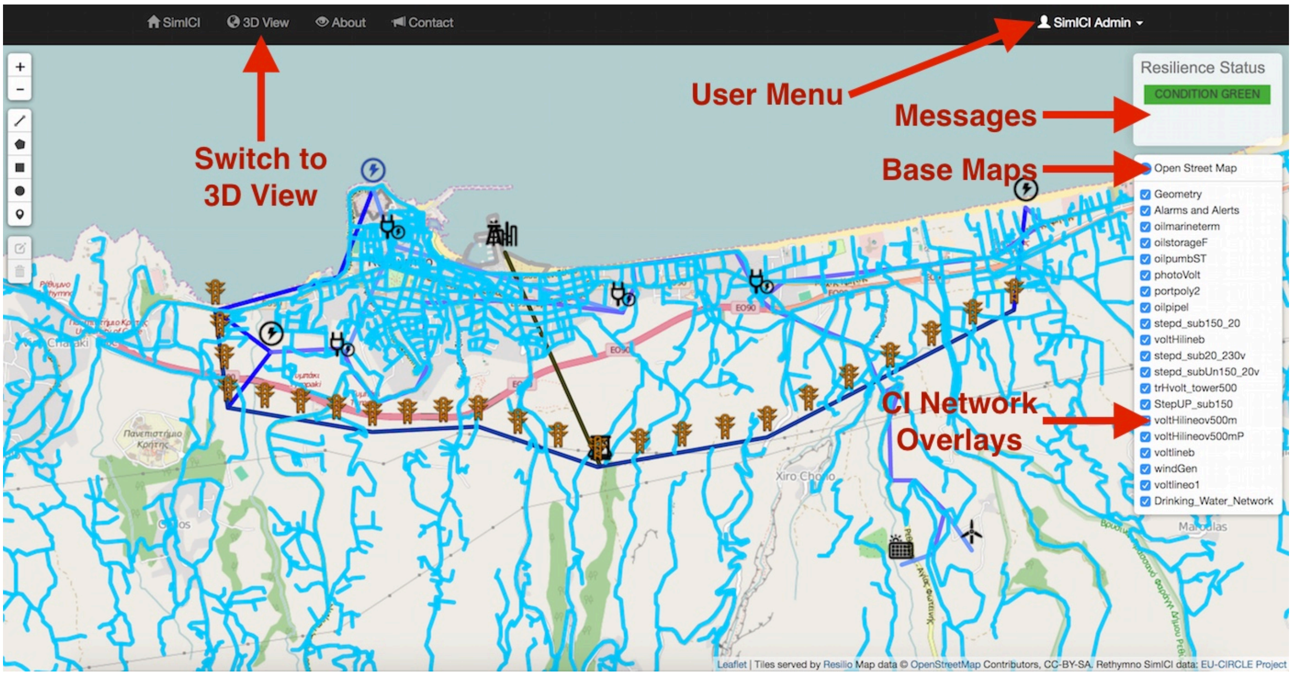


Figure 19: SimICI Mapterface View

The SimICI main page is known as the 'Mapterface': a user interface (UI) in which all interaction and experience is related to and from a 2D map background. The SimICI Mapterface can be centred and zoomed anywhere in the world and that perspective can then be stored as the default Area of Interest (AOI) for any given user. This means that different users can focus on different parts of the world within a single system. The view in Figure 19 is for the Rethymno AOI: the location of the EU-CIRCLE VDS.

The Mapterface is constructed from a base map (providing standard cartography) and a set of overlays representing the layers contained within the EU-CIRCLE VDS. The geographic elements of the Mapterface are supplemented with a top menu, a toolbar, a layer switcher, and a message box. In addition, a context menu is available through a right-click anywhere on the map. The context menu provides shortcuts to commonly used functions such as zooming, reporting co-ordinates at a chosen location, centring the map on a chosen location, and setting the current view as the user's default view.

Per Figure 19, the layer switcher element allows for overlays to be switched on or off. Please note that there is only a single base map defined (derived from the crowd sourced, open-source Open Street Map dataset) and this cannot be switched off. As and when other base map types (eg: satellite imagery) become available and are integrated into SimICI, base map switching will be enabled.

The messages box shown in Figure 19 is linked to the 'Alarms and Alerts' overlay layer. This is used to deliver messages to the user as and when a SimICI view is affected by a change in a SimICI Engine process. Section 3.4 illustrates the use of this element.

The User Menu seen in Figure 19 currently provides the sole option of logging out of SimICI.

The top menu of the SimICI application also contains a link to '3D View'. This is an experimental option that is intended to foster discussion of the utility of a 3D visualisation of CI assets and networks. The visualisation this view provides is shown, centred on Nicosia in Cyprus, in Figure 20 below.

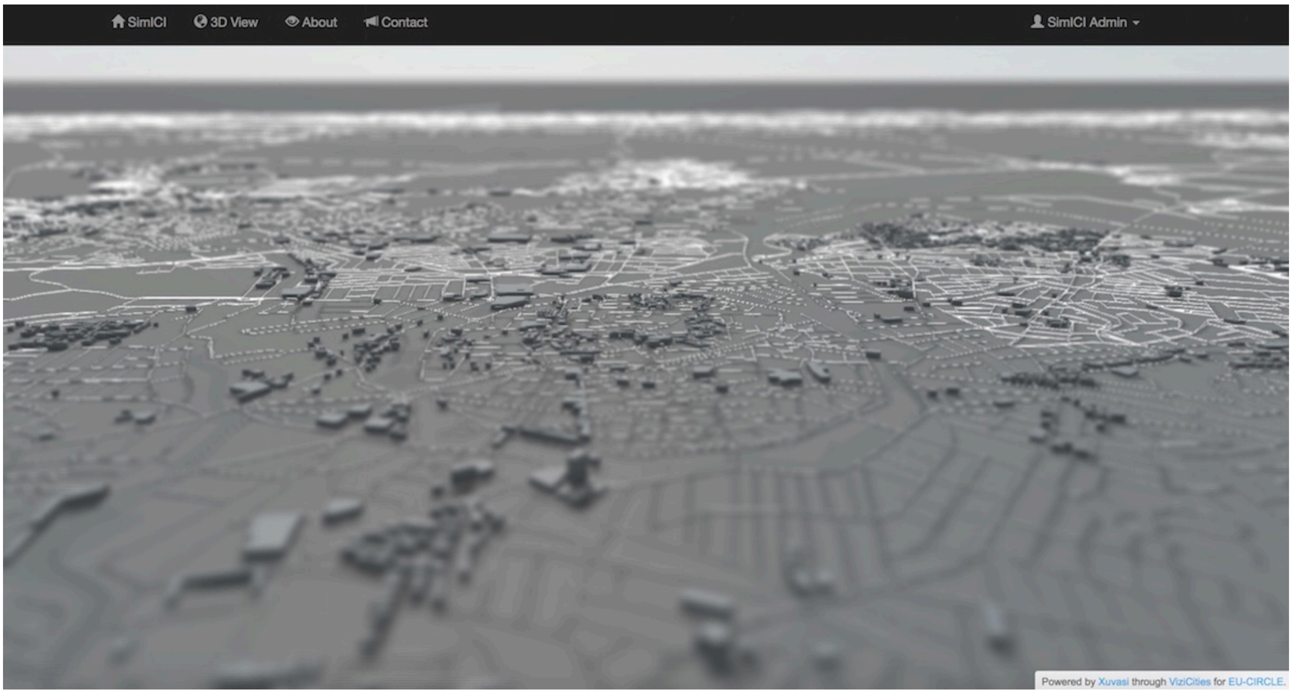


Figure 20: SimICI 3D View

The key difference between Figures 19 and 20, other than the perspective, is that Figure 20 shows the incorporation of height extrusions as part of the geographic experience. This may have the advantage of providing additional context to the user but is, as yet, unproven. The presentation of a 3D view is also computationally expensive (given that this view is being rendered in a web browser and not in a native application).

In short, the jury is out with regard to the operational usefulness of providing a 3D view in SimICI and, therefore, we would welcome any comments in this regard.

For the purposes of this guide, the focus is on the primary 2D Mapterface as a scenario is developed concerning the Oil Network described in the EU-CIRCLE VDS. A view of that Oil Network, located in Rethymno in Crete, is illustrated in Figure 21, below.

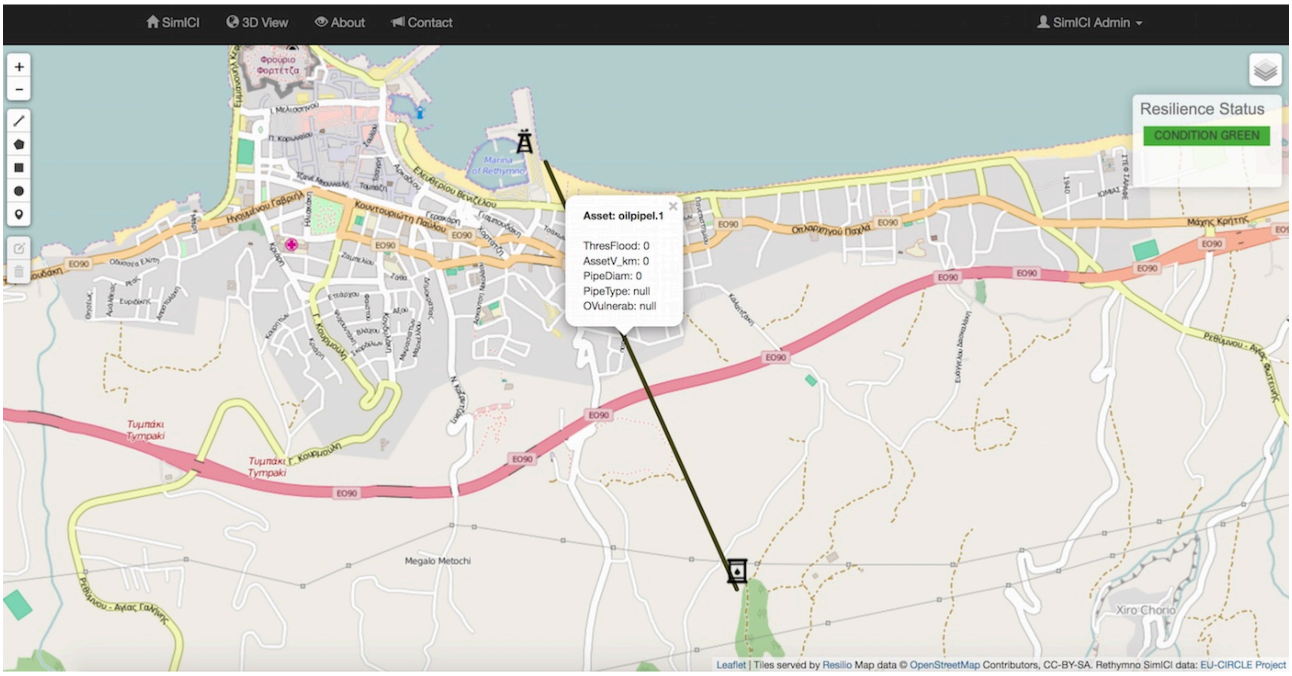


Figure 21: Viewing the Rethymno VDS Oil Network

The view in Figure 21 is achieved by using the layer switcher element (the small white box in the top right of Figure 21: move the mouse over the box to access the layers as it expands and collapses automatically to reduce on-screen clutter) to turn off all layers except those providing the assets of the VDS Oil Network.

The Mapterface is then zoomed in to just the area in which the Oil Network is present.

As can be seen in Figure 21, the VDS Oil Network contains a Pumping Station (located at Rethymno port and presumably serviced by incoming tanker vessels), an Oil Storage Facility (located inland behind the main urban areas of Rethymno), and pipeline that connects the two.

Although a very basic network, this serves to support the development of a scenario exploring cascading effects due to damage or other affects at points within that network. As a key output of the scenario we also want to determine how resilient the Oil Network is to different types of impact.

3.2 The SimICI Engine

In order to turn a pretty standard map into an active simulation that can be used to explore impacts, sensitivities, and resilience, it is necessary to use the SimICI Engine to build and execute a model of the Oil Network.

This is not, however, a requirement for a SimICI user to know how to write software as, using the SimICI Engine and the provisions made therein by the SimICI Administrator, the required model can be built visually using a drag-and-drop approach.

Logging in at the SimICI Engine endpoint provides access to the SimICI Engine Designer User Interface (UI) as shown in Figure 22, below.

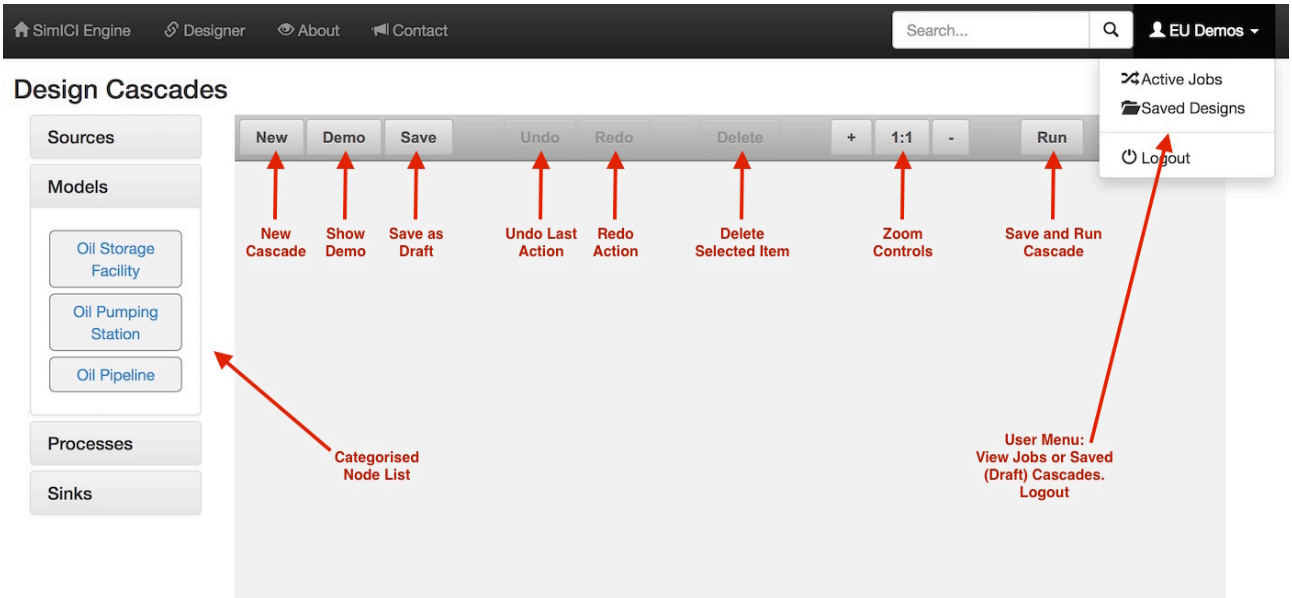


Figure 22: The SimICI Engine Cascade Designer

Per Figure 22, the SimICI Engine UI contains a number of application elements as:

- Designer Workspace:
  - The shaded grey area occupying the majority of Figure 22
  - The Designer is where nodes are dropped and connected together
  - The set of nodes and connections is known as a 'cascade'
- Categorised Node List:
  - The accordion element at the left hand side of the screen
  - A SimICI Engine Source node is a location from which to acquire data
  - A SimICI Engine Model node is a representation of an asset in a CI network
  - A SimICI Engine Process node is a calculation you want to perform on a CI network
  - A SimICI Engine Sink node is a location to which you want to send data
  - SimICI Engine nodes are dragged from the list to the main designer workspace for use
- Designer Menu:
  - The darker grey menu bar at the top of the workspace
  - Provides tools and functions relative to the Designer activity:
    - Create a new cascade (clear the workspace and start afresh)
    - Show a demonstration of a cascade (builds a sample)
    - Save as a draft (saves the current cascade without running it)
    - Undo the last thing you did

        - Redo the thing you just undid
        - Delete the currently selected node or connection from the cascade
        - Zoom the workspace in and out or return to actual size (1:1)
        - Run the current cascade, saving it first
- SimICI Engine menu:
    - The black menu bar at the top of the screen
    - The User Menu drop down provides:
        - Active Jobs – a list of running and previously run cascades
        - Saved Designs – a list of designs saved as drafts
        - Logout of the SimICI Engine

NB: if you log out of the SimICI Engine any cascades you have started will remain running.

The Designer is used to build a software model of the CI network of interest: in this case, the Oil Network from the VDS. To get started, we drag the relevant models from the node list to the workspace.

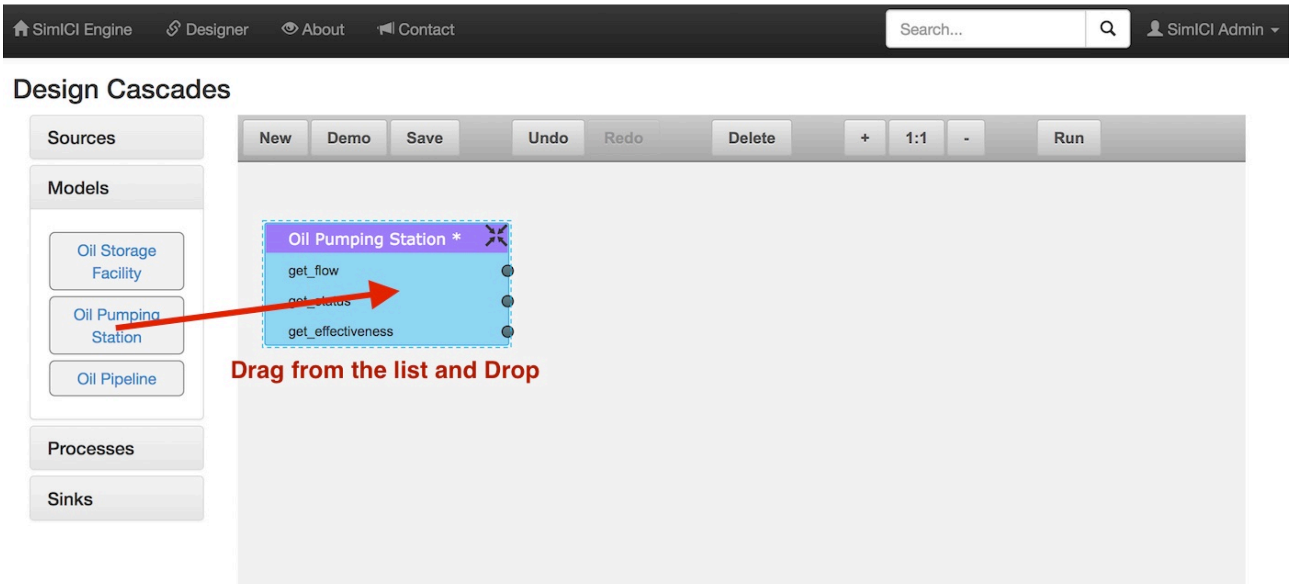


Figure 23: Adding a Node to a Cascade

Figure 23 illustrates the start of this process. Clicking on the 'Models' tab in the node list accordion reveals three models available for use[4]. Clicking on 'Oil Pumping Station' and, with the mouse button held down, dragging it into the workspace creates an instance of that model in the cascade.

[4] The definition and compilation of models for use in the SimICI Engine is an Administrator task.

Once dropped, the model expands and draws itself to show any available inputs and outputs (the grey dots on the right and/or left side of the model instance in the workspace). Clicking, dragging and dropping an 'Oil Pipeline' model into the workspace creates a second model, also with available inputs and outputs.

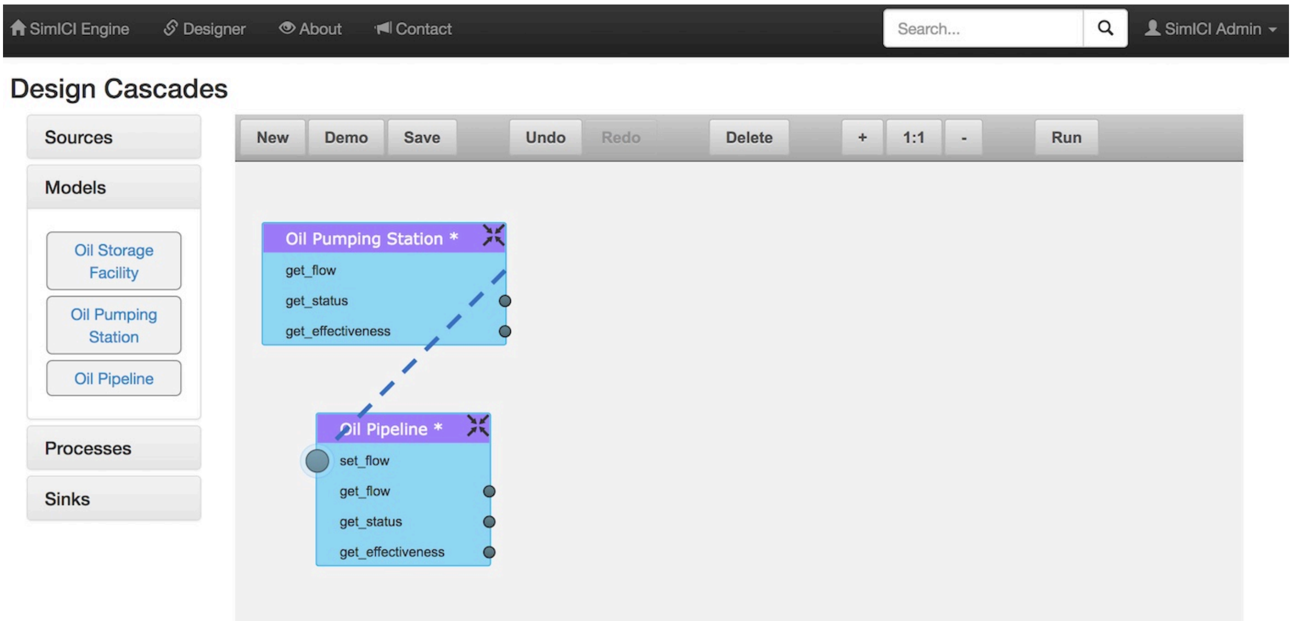


Figure 24: Connecting Node Inputs and Outputs

As Figure 24, above, shows, dragging the get_flow output from the Oil Pumping Station onto the set_flow input of the Oil Pipeline indicates, with a dashed blue line, that a connection can be made between that output and input. Releasing the mouse button at this point results in a valid connection as illustrated in Figure 25, below.

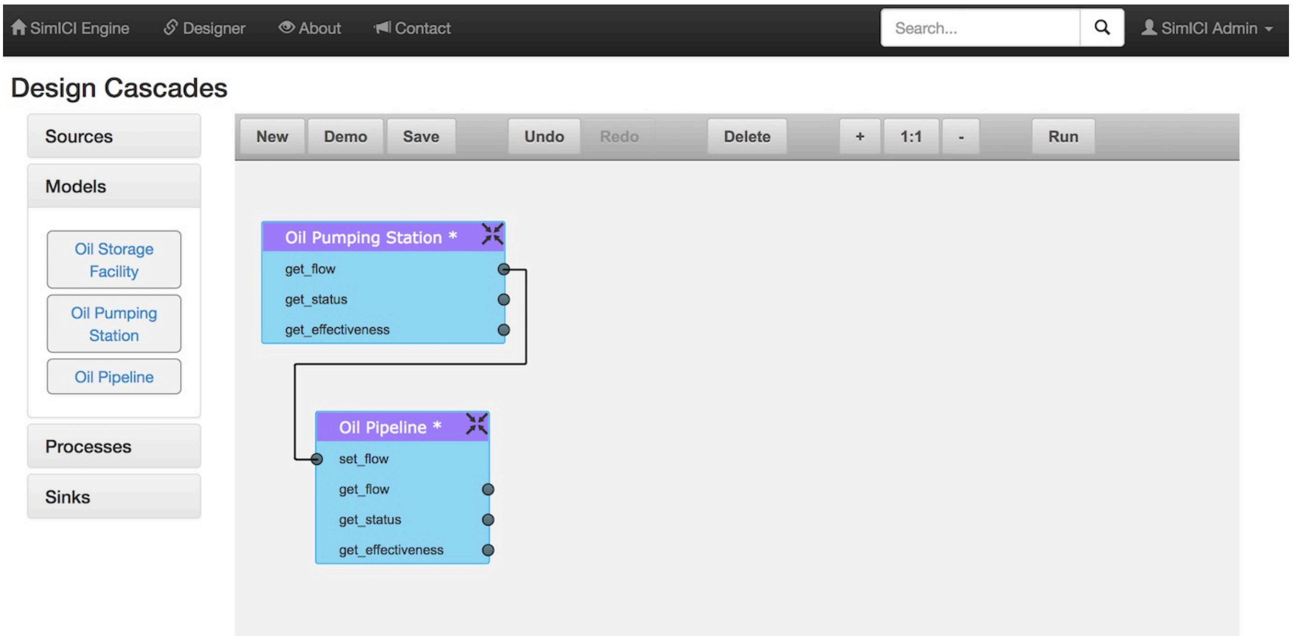


Figure 25: A Valid Connection

As might be surmised from Figure 25, a connection between the output flow of the Oil Pumping Station and the input flow of the Oil Pipeline has now been made. This is as would be expected: what comes out of the Pumping Station must go into the pipeline: at least in the network we have to hand.

The connection in the designer, however, is more than a simple visual representation: it actually represents a real-world flow. As with the real-world, the flow can be varied: perhaps the Oil Pumping Station has had a breakdown or is running slowly due to reduced available power? The effect of this reduction in flow would propagate from the Oil Pumping Station into the Oil Pipeline, reducing flow to assets downstream of the Oil Pumping Station with attendant effects.

As with the real-world, so with the SimICI Engine. If an impact is applied to the Oil Pumping Station model, it will stimulate the model to dynamically recalculate its output. The result will be a change in the flow rate from the Oil Pumping Station model which, of course, is an input to the Oil Pipeline. The change in the input to the Oil Pipeline stimulates a dynamic recalculation of the flow rate of the pipeline, which then propagates through the model of the network. It is this propagation effect – with effects propagating dynamically from model to model – that gives the design its name: a cascade.

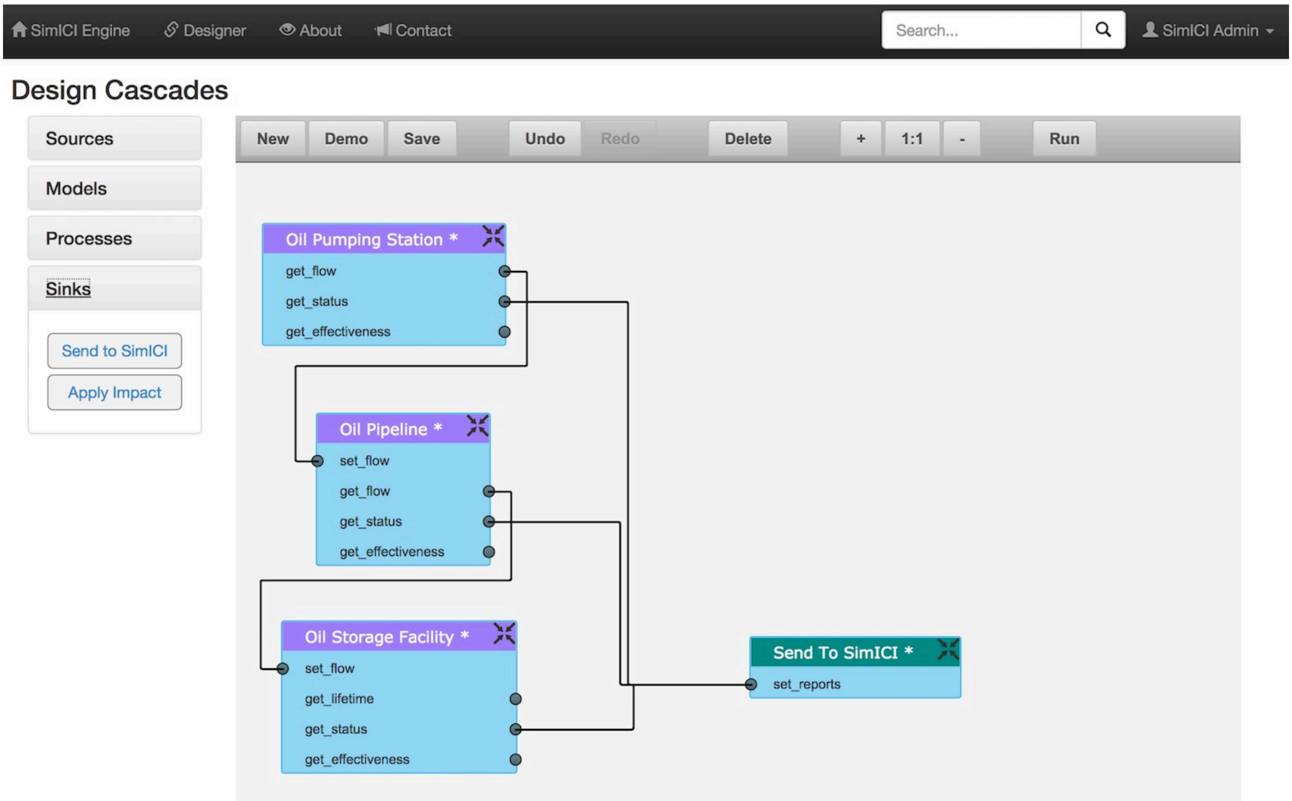


Figure 26: Sending Node Updates to the SimICI Mapterface

Figure 26, above, illustrates the completed model of the VDS Oil Network in the SimICI Engine. All of the assets are modelled and the relevant outputs and inputs connected. Picking up on the previous discussion, the causal mesh of connections and interdependencies in which the assets exist can be clearly seen.

Additionally, Figure 26 contains a new Sink node to 'Send to SimICI'. The node is preconfigured to send messages from a cascade in the SimICI Engine to the main SimICI web application. Use of the node requires simply that it is dragged into the workspace and then connected to messages that are output from one or more nodes in the cascade.

A Sink node such as this simply takes a standard text message, perhaps representing a change in status or condition for a node, and relays it to the main SimICI web application. The SimICI web application then displays the message in the Message Box (assuming the Alarms and Alerts layer is visible) so that the user is aware of changes in the model.

This provides for the ability to generate and route intangible outputs from a model within a cascade: this indicates that outputs do not have to be directly related to the physical aspects of a model but that, stimulated by changes in the overall cascade, outputs can also be related to process or other metrics. In short, it is as if your real-world assets could tell you when they were experiencing an issue!

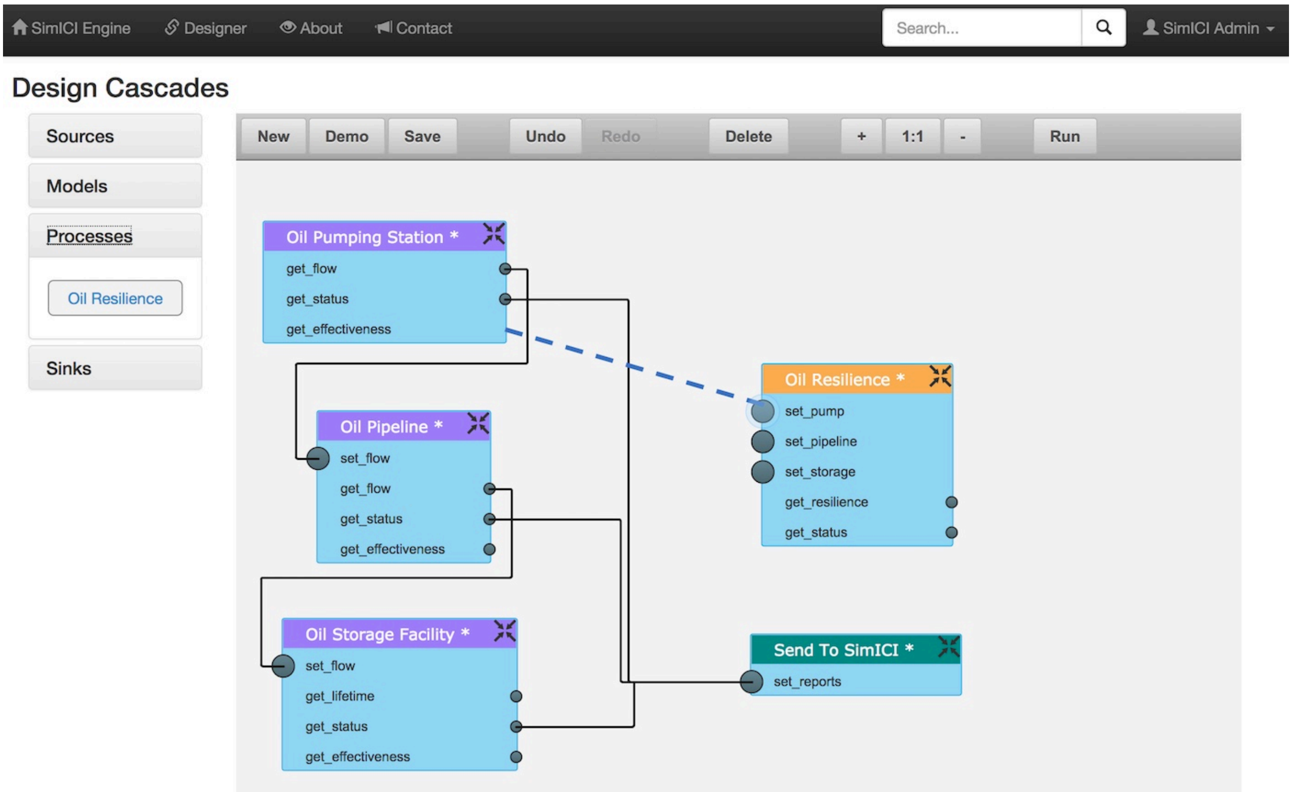


Figure 27: Adding a Process Node to Calculate Metrics

Figure 27 extends the idea of intangible outputs. Each of the model nodes in the Oil Network cascade emits an output representing its effectiveness. Effectiveness, obviously, is not directly related to the physical aspects of an asset but, rather, to the cumulative operational state of an asset considering asset type, context, etc. If effectiveness, as an abstract attribute of an asset, can be collected from all assets within a network then a collected view of network effectiveness – or resilience, capacity, etc. – can be calculated in real time and in response to changes occurring at any point in the network.

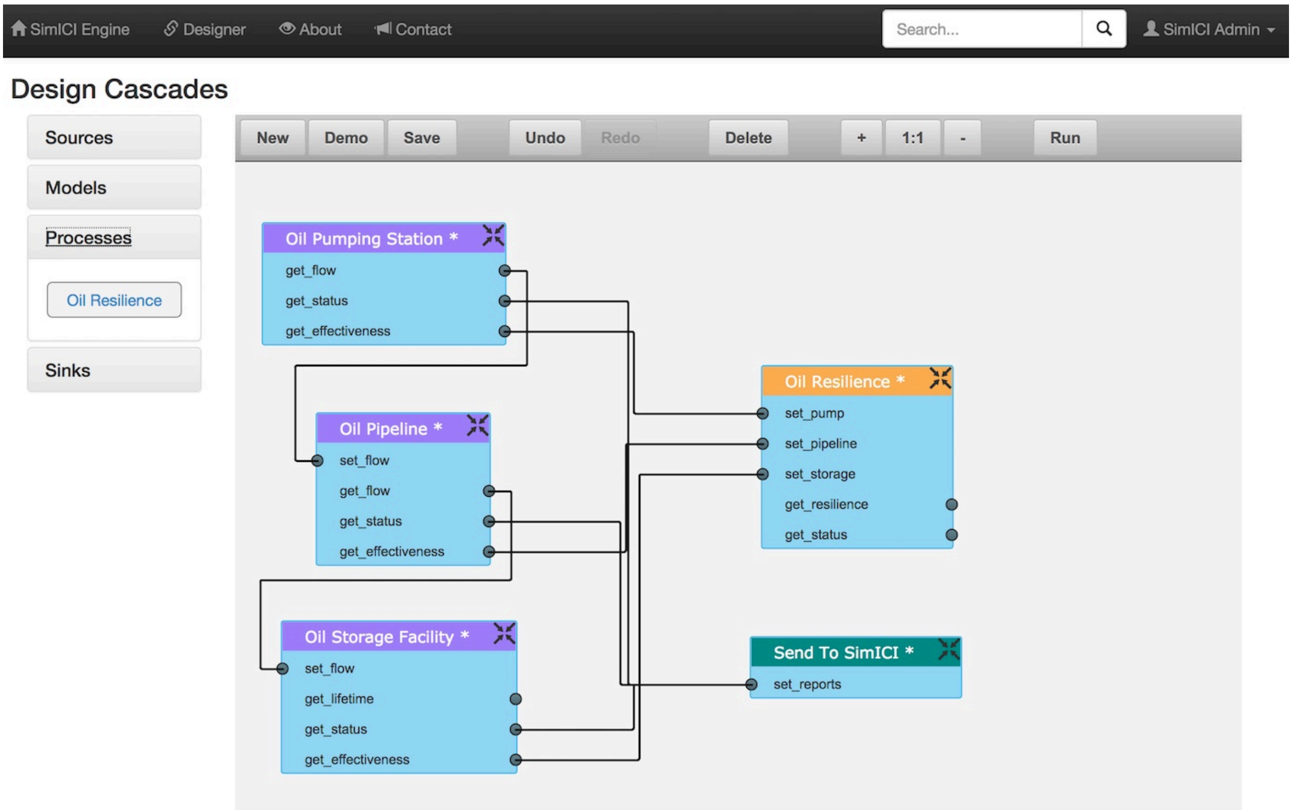


Figure 28: Connecting the Process across the Network

Figure 28 illustrates the connection of the Oil Resilience process node (per Figure 27) across all of the assets in the VDS Oil Network. This design means that any change in effectiveness, at any point in the network, will stimulate a dynamic recalculation of the 'resilience' of that network[5].

Careful consideration of the inputs, outputs, and behaviours of a model node will provide the opportunity to connect a variety of process nodes to key points in the set of models used in a cascade. Strong collaboration with the Administrator will allow such considerations to flow into the definition and compilation of model nodes for use within the SimICI Engine which, in turn, are able to provide the relevant stimuli for a wide range of process nodes.

Using this approach, there is no reason why the same cascade could not be employed to calculate a variety of observations from the network modelled in the cascade. Such observations might include financial, operational, efficiency, resilience, and more.

To complete the core design, connecting the get_status output from the Oil Resilience node to the input of the Send to SimICI node ensures that any changes in the calculated resilience of the Oil Network will be routed to the main SimICI web application and displayed to the operator. This is illustrated in Figure 29, below.

[5] The calculation employed in this example 'Oil Resilience' node is a simple compound multiplier and is by no means representative of a true resilience algorithm.

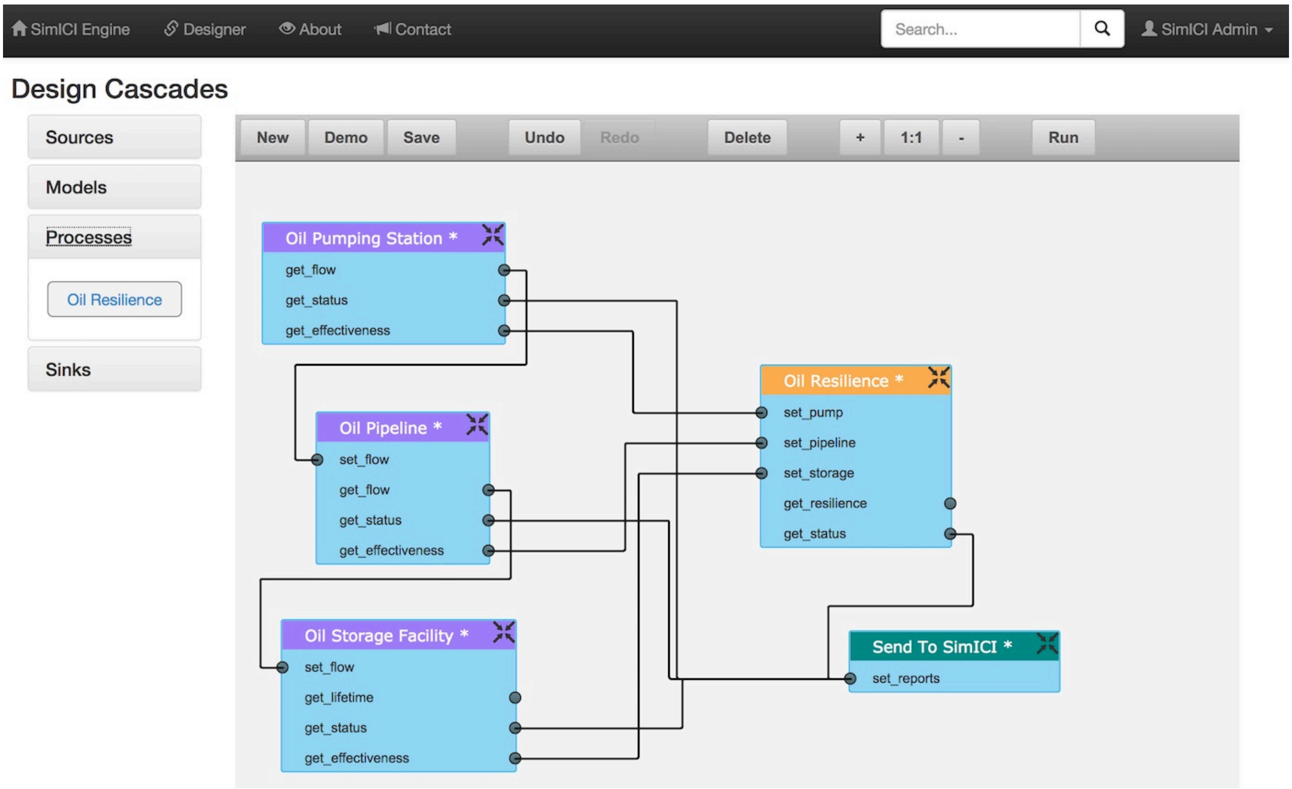


Figure 29: Sending Process Updates to the SimICI Mapterface

Figure 29 illustrates the completed cascade representing the assets and required observations of the VDS Oil Network. The challenge now is how to influence and impact the models contained within the cascade such that they dynamically respond to affects and events.

In a future version of SimICI, a source node will be provided that will read impact data from a data source output from EU-CIRCLE CIRP analyses. That impact data will then be applied to impacted nodes within a cascade representing a real world network. This will allow projected impacts – typically damage – to be applied in SimICI and propagated within and across CI networks.

In the absence of that source node, however, and in support of a more experimental approach to resilience analysis, the SimICI system includes a web application known as the Impact! client.

This application connects, dynamically and in real time, to a running cascade in the SimICI Engine. It reads the set of models from the cascade and provides the opportunity to directly influence any of those nodes **while** the cascade is running. Impacts applied in this way work in exactly the same way as impacts read in from a CIRP output: they stimulate the node they are impacting which generates an effect that propagates across the cascade.

In order to use this approach, however, we must provide a way for the Impact! client to communicate with the cascade. Figure 30, below, illustrates the addition of two nodes to support this.

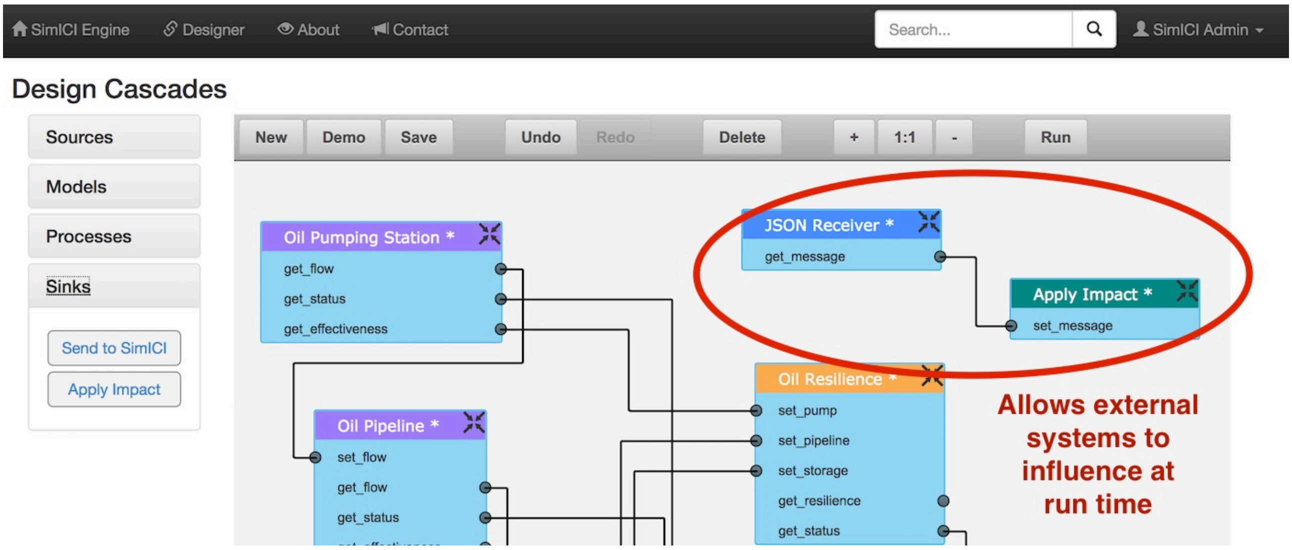


Figure 30: Allowing External Influence at Run Time

In Figure 30, we add a source node 'JSON Receiver' that provides a way for a message to be received by the cascade. We also add a sink node 'Apply Impact' that will determine the content of the received message and apply the relevant impact to the right impacted node.

Connecting the two new nodes together enables the cascade to be influenced by an external application while it is running. This means that impacts and influences can be applied to the cascade and be responded to in real time by the cascade. The application here extends beyond traditional what-if analysis and provides a simulation capability, directly reflecting a client's real world network, that can be employed in workshops or other consultant-led activities.

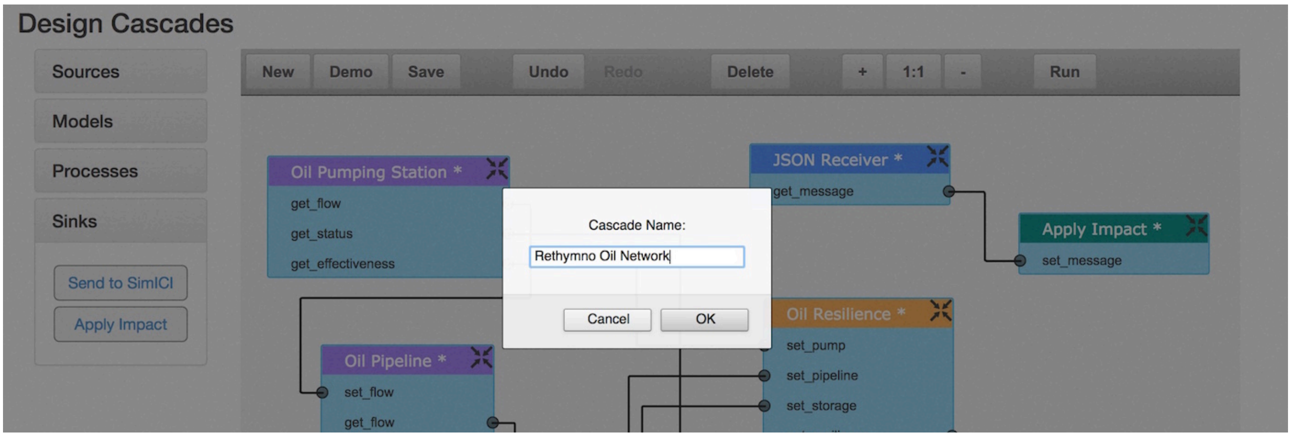


Figure 31: Saving and Running a Cascade

With the cascade design complete, pressing the 'Run' button prompts for a name for the cascade. Entering a name will trigger a validation of the cascade design which, if successful, leads to the dynamic assembly, compilation, and execution of the cascade.

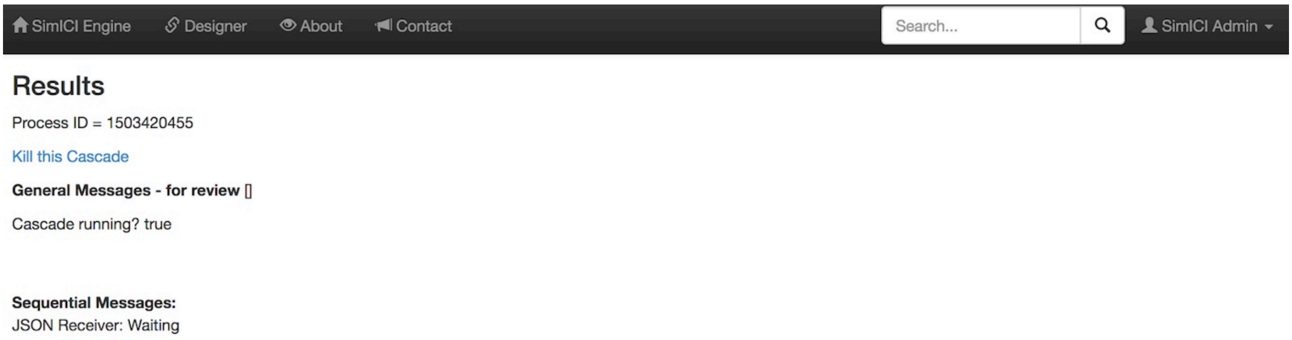


Figure 32: Status of the Running Cascade

Once a cascade has been successfully executed, the SimICI Engine UI changes to show the Results of that cascade. The cascade results screen, shown in Figure 32 above, reports generally useful information about the execution of the cascade: including a log of messages received by the cascade from an external source.

Any cascade will continue to execute until it is killed. Clicking on 'Kill this Cascade' in the results screen (see Figure 32) will terminate the cascade. The SimICI Engine UI will then switch to the 'Active Jobs' screen, displaying a list of all cascades that have been previously run and their respective status. This is shown in Figure 33, below.

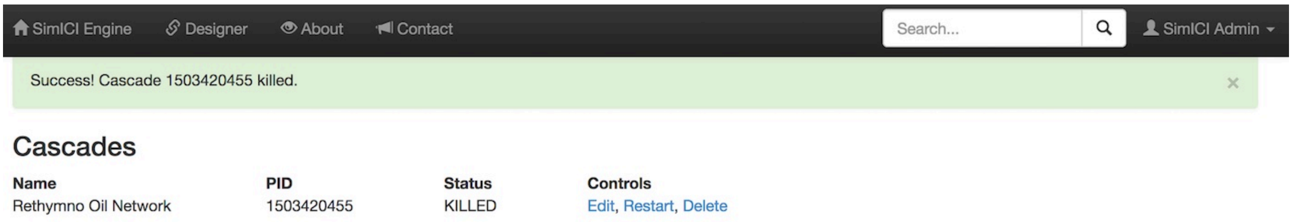


Figure 33: A Killed Cascade can be Restarted or Edited

As indicated in Figure 33, a cascade that has been killed can be edited, in which case the cascade is opened in the designer workspace for modification and subsequent rerunning[6], or restarted directly from the Active Jobs list. When a cascade is rerun, the results screen becomes available from the Active Jobs list – along with a 'Kill' option as a shortcut to stopping any given cascade.

Finally, as also shown in Figure 33, a cascade can be deleted from the Active Jobs list. If deleted, all references to the cascade – including the design - will be removed from the SimICI engine.

[6] Note that an edited cascade can be saved under a different name. This allows for cascades to be subclassed if required.

3.3 The Impact! Client

As mentioned previously, SimICI includes a web application known as the Impact! client. This application is provided as a means of dynamically connecting to a running cascade and applying affects and influences to it in order to stimulate the cascade into action.

Logging in to the Impact! client[7] endpoint presents the Impact! main screen as shown in Figure 34, below.

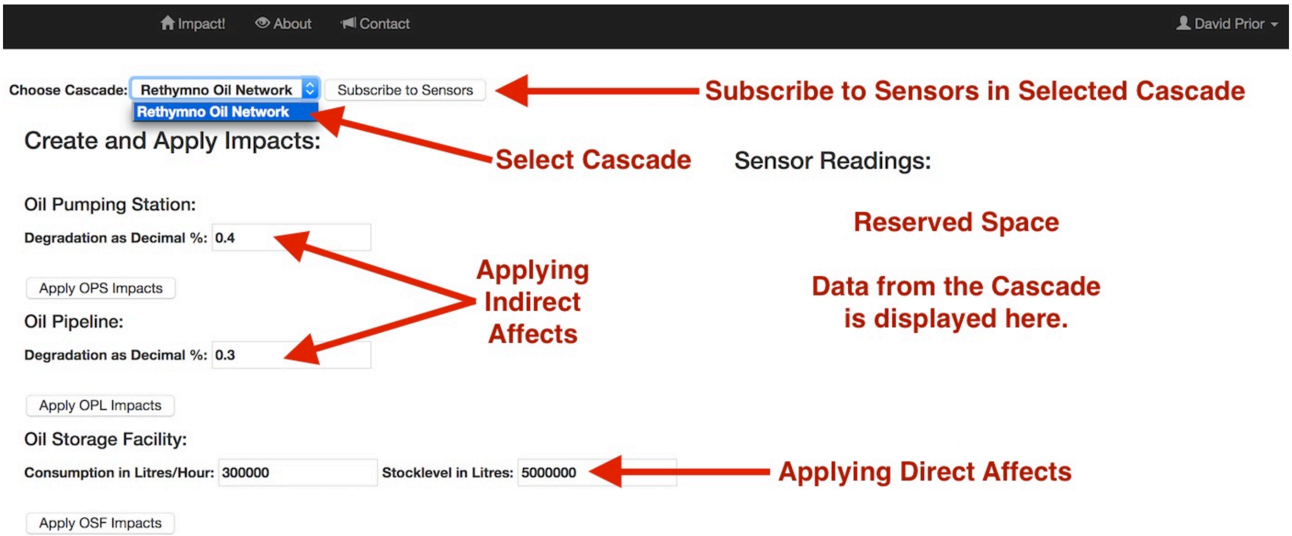


Figure 34: The Impact! Client

The Impact! client identifies available cascades within the SimICI Engine and provides a selector from which the user chooses the cascade they want to influence. With the cascade selected, a press of the 'Subscribe to Sensors' button will read out the available nodes and associated impact points from the selected cascade as well as subscribe to all of the internal sensors available from all of the nodes within that cascade.

As shown in Figure 34, the available impact points for our Oil Network cascade are listed on the left hand side of the screen while space is reserved on the right of the screen for the sensor data outputs to be received from the cascade.

It should be noted that different types of affect can be applied. For the Oil Pumping Station and the Oil Pipeline, indirect affects are available. These are, essentially, calculated or estimated inputs that have a more generic impact (eg: degradation is not a known physical input to an asset but can be estimated based on the current context in which that asset is operating).

For the Oil Storage Facility, on the other hand, direct impacts – correlated with actual or forecast data – can be applied. Here, for example, there is the potential to vary demand and reserves in order to determine the useful lifetime of the storage. Factoring those impacts with variable input flow (replenishment rates) due to impacts elsewhere in the network then helps with resilience analysis under different conditions.

[7] As previously, user accounts are currently controlled by Xuvasi Ltd on behalf of the EU-CIRCLE project.

3.4 Putting it all Together

At this point in our scenario, we have:

1. A 2D map displaying the geography of the real world assets in the Oil Network
2. A cascade representing the Oil Network, and its causal mesh, as running software with the capability of sending updates to the 2D Map
3. The ability to directly affect Oil Network assets from an Impact! client connected to the cascade

In order to exercise our cascade, perhaps for reasons of a resilience workshop with a client looking to make a balance of investment decision, we need to apply an impact.

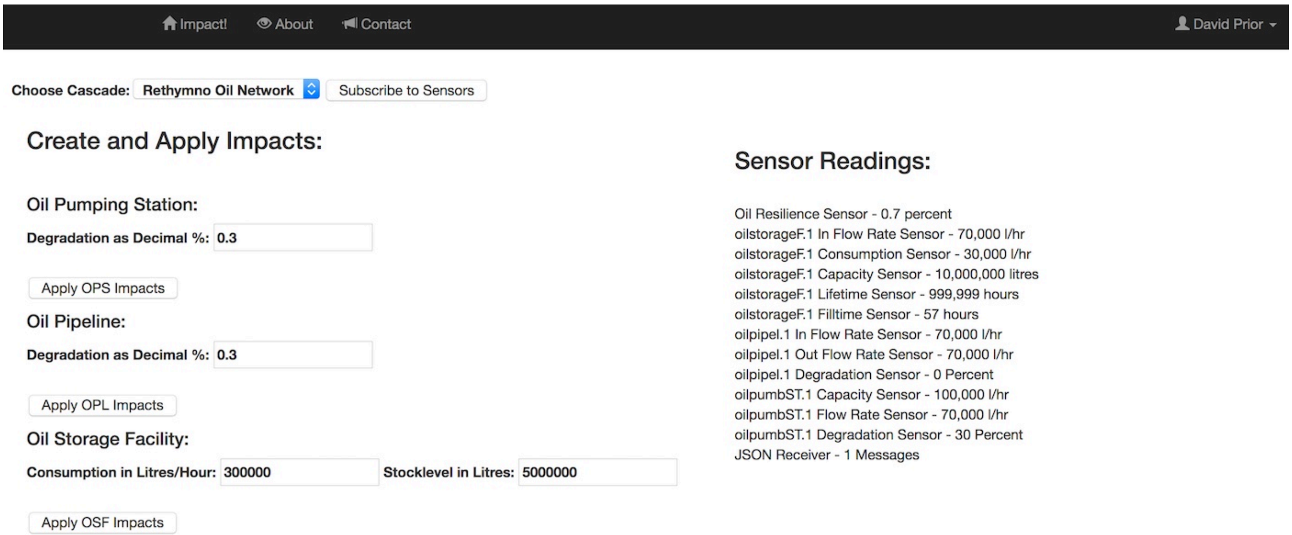


Figure 35: Applying an Impact

In Figure 35, we apply an indirect impact to the Oil Pumping Station that assumes degradation of the overall capabilities of that asset by 30%. As soon as we pressed the 'Apply OPS Impacts' button, the impact was sent to the Oil Pumping Station model in the cascade. That stimulus triggered a recalculation of the flow rate from the Oil Pumping Station as a result of the degradation which, in turn, propagated through the cascade to determine the cascading effect on Oil Pipeline capacity and, subsequently, the fill rate of the Oil Storage Facility.

The propagation of the impact through the cascade changed the majority of the run time sensors inside the cascade: causing each sensor to report data back to the Impact! client (the right hand side of Figure 35). The whole process – from application of impact through to sensor data being presented on the screen – took almost less time than releasing the mouse button that triggered it.

We can confirm that the impact was applied to the cascade and that the cascade responded to the Impact! client by reference to the results screen for the cascade in the SimICI Engine. This is shown in Figure 36, below.

Figure 36: Cascade Status with the Impact Applied

As expected, the results screen for the cascade shows receipt of the impact message from the Impact! client: declaring a degradation of 30% needs to be applied to the asset 'oilpumbST.1' (the asset identifier for the Oil Pumping Station as drawn from the EU-CIRCLE VDS). On receipt of that message, the cascade was stimulated and reacted to the impact on one of its models. This triggered a dynamic response in the causal mesh described by the cascade.

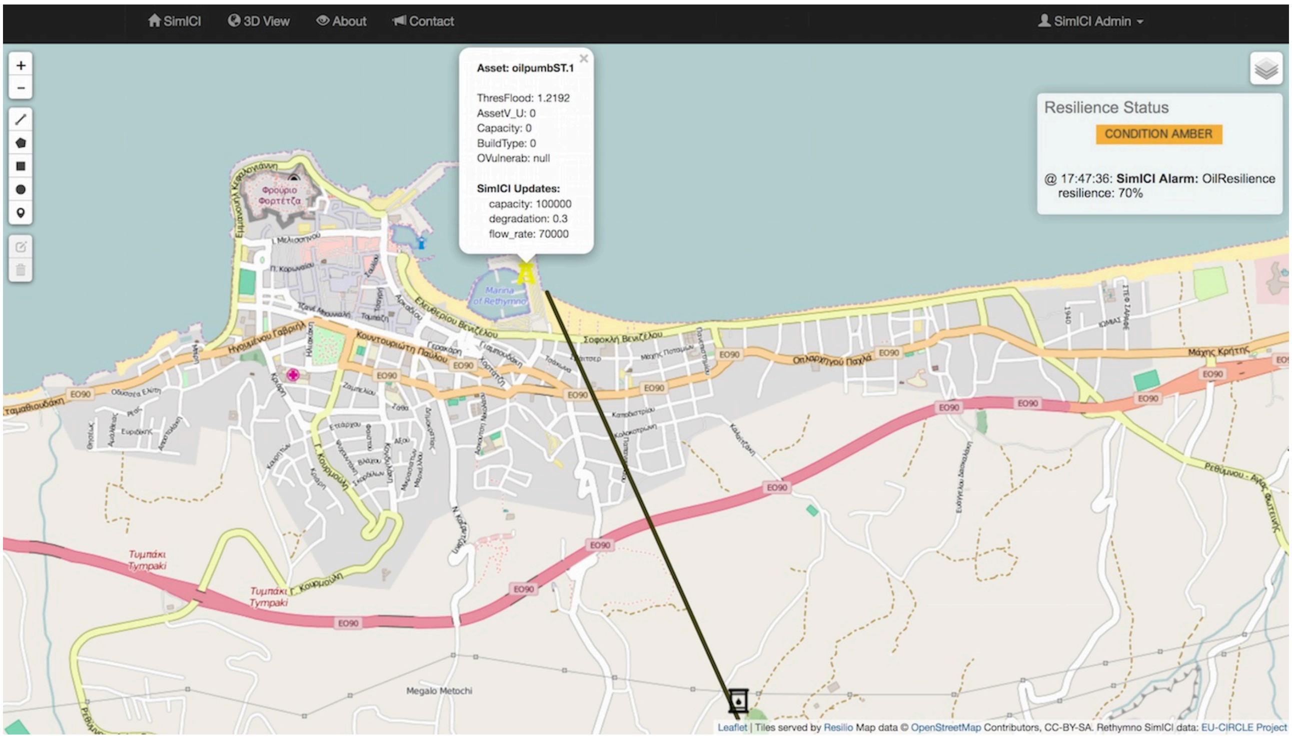


Figure 37: Propagation of Impact to SimICI Mapterface

Back in the 2D Mapterface, the degraded asset has been visually highlighted and a SimICI Alarm message has been displayed in the message box. The map icon representing the asset has been highlighted in response to the status update that has been sent from the model in the cascade to SimICI (the connections described in Figure 26) while the Alarm message has been routed from the Oil Resilience node in the cascade to SimICI (the connections described in Figures 28 and 29) as a result of recalculating the resilience of the Oil Network due to the impact received.

As we can now dynamically create and apply impacts at will, lets impact another asset and have that impact accumulate on top of the existing affect from the degradation of the Oil Pumping Station.

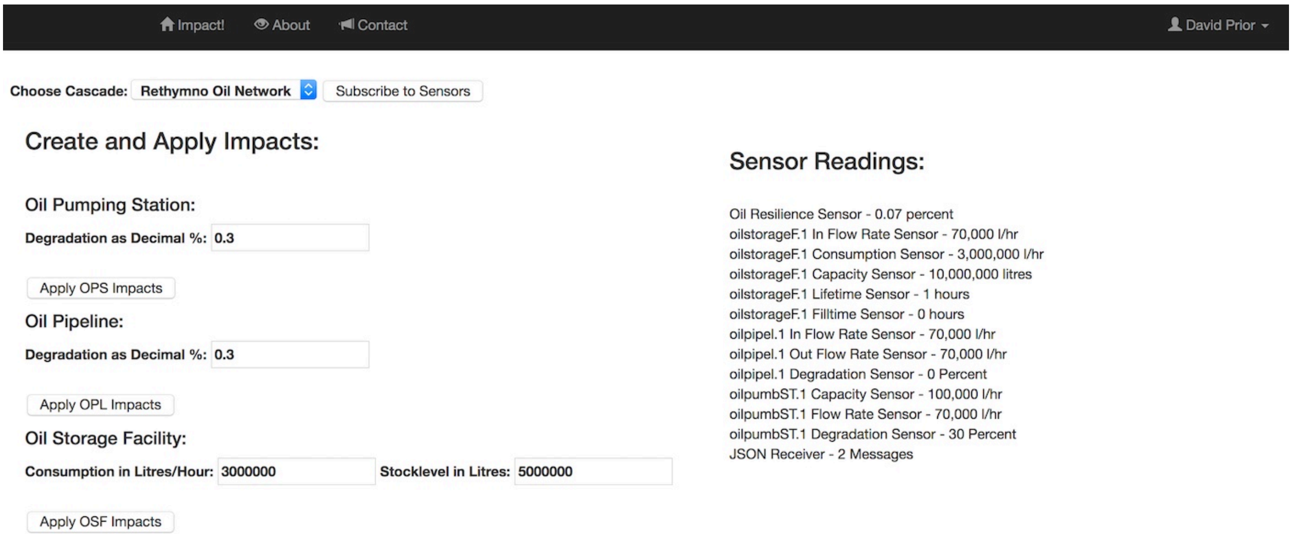


Figure 38: Applying a Second, Cumulative Impact

This time, as shown in Figure 38, we have applied a direct impact to the Oil Storage Facility and have increased consumption from it a hundredfold: from the default 30,000 litres per hour to a massive 3,000,000 litres/hour. This, for example, may occur where other Oil Storage Facilities have been removed from the network due to a pipe failure elsewhere and all demand has been switched to the single tank in our cascade.

As previously, the sensors inside the cascade have reported their changes; showing not only the change in consumption rates but also the dramatic impact on the performance of the Oil Storage Facility. Not only does the facility now only have a single hour lifetime (against current stock levels): it will never be able refill! In consequence, you should also note that the (basic multiplier) Oil Resilience metric has collapsed from 70% due to the previous impact just 7% as a result of the cumulative effect of both impacts.

It should be noted that, while this Guide is detailing a scenario solely concerned with the VDS Oil Network, exactly the same techniques can be applied to scenarios containing multiple CI networks in Network of Networks (NoN) form. In such scenarios, impacts can be applied at any point in any network and the propagation of effects both within and across the NoN can be observed due to the definition of interdependencies in the cascade.

As previously, the application of the second impact can be observed through reference to the cascade results screen in the SimICI Engine. Figure 39, below, shows the receipt of the second impact to be applied to the asset 'oilstorageF.1'.

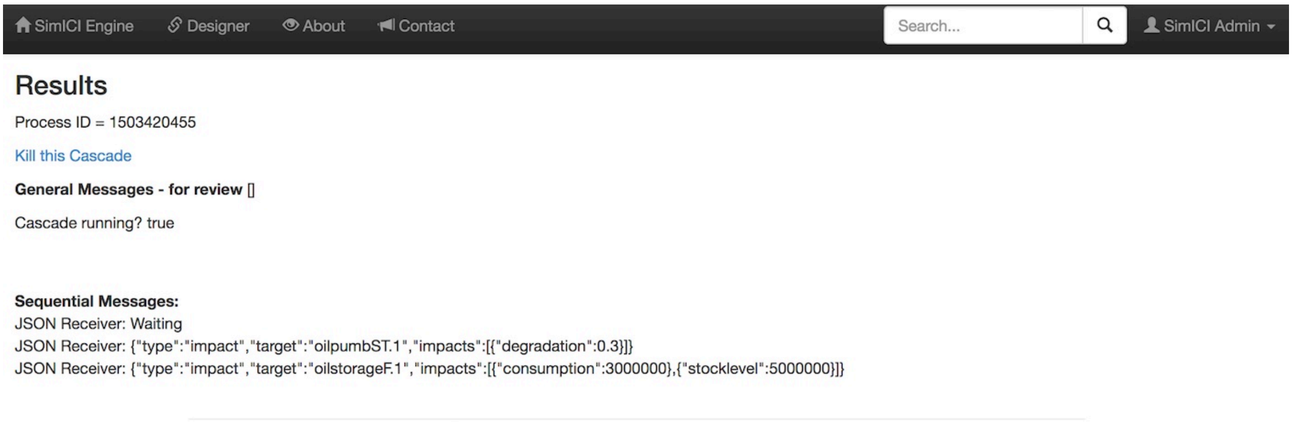


Figure 39: Cascade Status with Both Impacts Applied

And, as previously, the cumulative effects are also visible in the SimICI Mapterface for the VDS Oil Network.

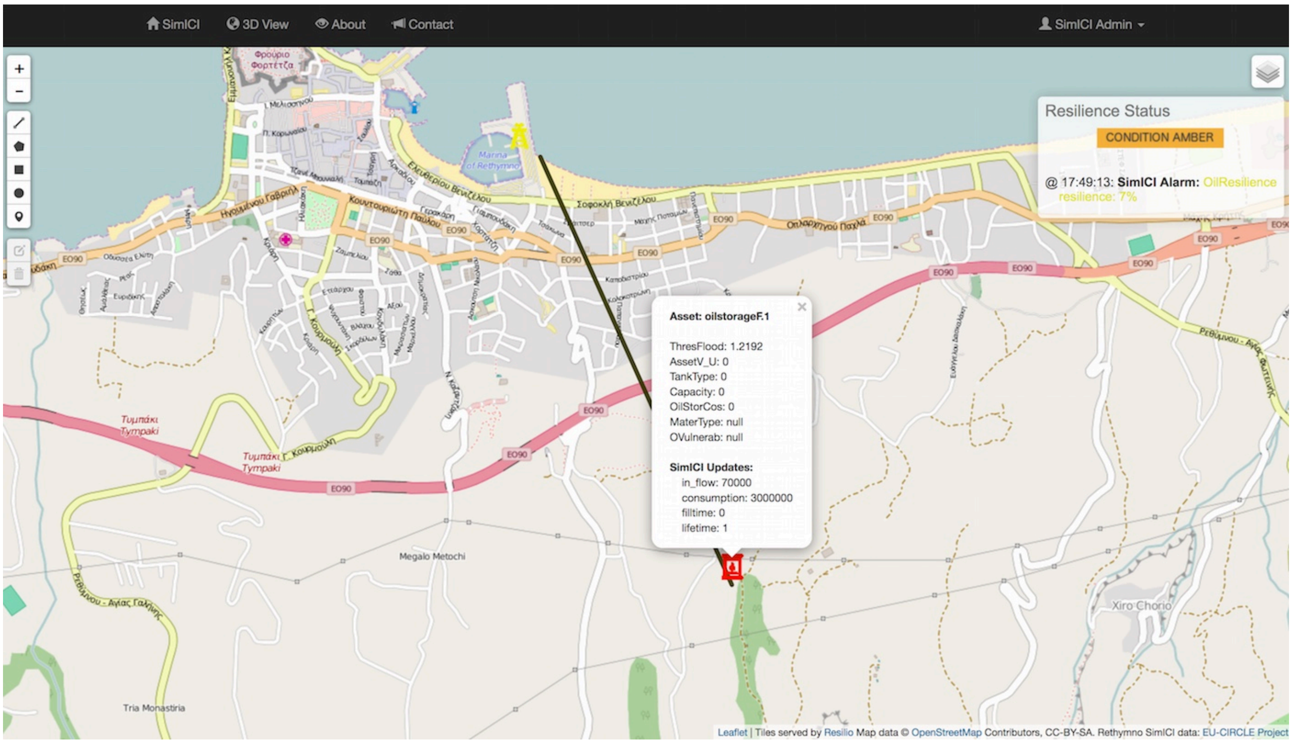


Figure 40: Propagation of Cumulative Impacts to SimICI Mapterface

It should be noted that at no time has the underlying data from the EU-CIRCLE VDS been modified by any of the scenario activities described previously. All of this has been driven from a snapshot of the VDS data which has then been manipulated through the application of impacts against what is, essentially, a simulation of the real world network.

4 In Closing

SimICI Version 1.0 presents a powerful collection of software that can be exploited in support of the overall objectives of the EU-CIRCLE project.

- The SimICI Geoserver provides structured management in support of the EU-CIRCLE VDS which, with support for open standards and protocols, allows for the exploitation of the VDS both within and outside of the EU-CIRCLE project.
- The main SimICI application provides an intuitive interface to work with geospatial data representing real world assets and networks and, moreover, within which the impact of damages and other events can be visualised in workshop or similar environments.
- The SimICI Engine provides a mechanism that allows for a software representation of assets and networks – and the causal mesh arising from such – with support for the derivation of a wide variety of metrics and other observations from the the application of impacts to that representation.
- The SimICI Impact! client provides a mechanism through which consultant-led activities may explore and determine the impact of damages and other events on a real world environment: providing clients with insight into current and future CI provisions.

4.1 Coda to D7.3

As a result of changes to the requirements for the SimICI environment, resulting from the conceptualisation and definition of the ANDI methodology and the Asset Class Repository service, SimICI Version 1.0 will undergo an extensive redesign with associated redevelopment.

As such, this deliverable D7.3, while relevant and complete in relation to SimICI Version 1.0, will be redundant before the end of the EU-CIRCLE project.

** ENDS **