Project name	Platform of Trust integration [Smart City]
GameEngine	Unreal Engine
Version	4.25
Synopsis	Platform of Trust data-product integration to 3D model
Functions	• Platform of Trust data-product data fetching • Data visualization with Widget element
Blueprint/C++ project	Blueprint
Hardware environment	PC: Windows 10, 64 bit External mouse Keyboard Visual Studio2019 C++

What plugins you need to install and active in Unreal

[kantan-charts]

[Low entry extended standard library]

[VaRest-plugin]

Plugin problems

Varest:

- In Unreal engine library VaRest cannot do JSON body keys short, so that need to be already in right order when constructing JSON.
- Pretty JSON or raw JSON format was missing right format so this process was needed to do with replace function.

Data-product blueprint model

In appendix 1.

Data-product blueprint contains forming the JSON body with the input parameters. Blueprint automatically generates Platform of Trust signature and make request to defined URL. Requested data is parsed and parsed data is saved to variable.

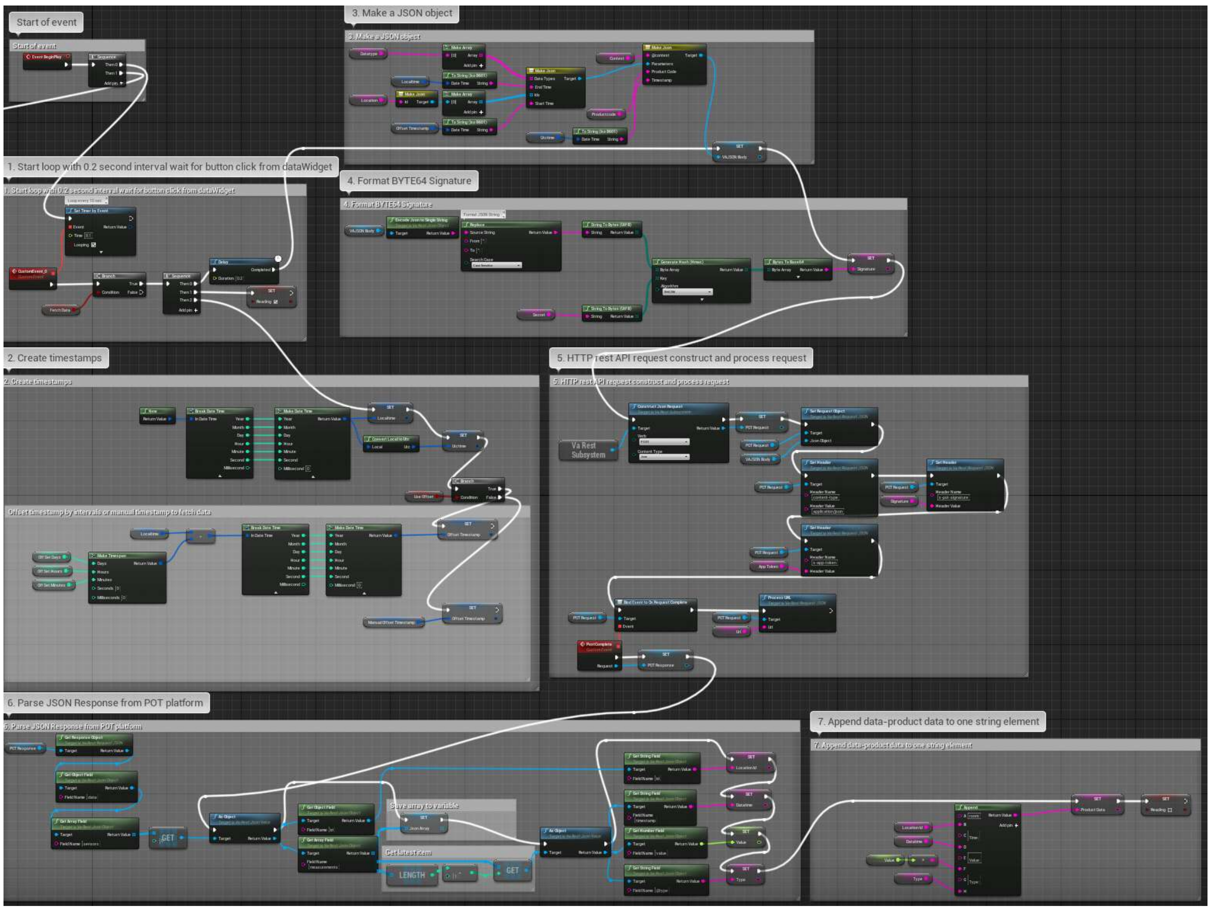

Start of event
1. Start loop with 0.2 second interval wait for button click from dataWidget
2. Create timestamps
3. Make a JSON object
4. Format BYTE64 Signature
5. HTTP rest API request construct and process request
6. Parse JSON Response from POT platform
7. Append data-product data to one string element
Offset timestamp by intervals or manual timestamp to fetch data
Save array to variable
Get latest item

Figure 1. Data-product blueprint.

Data-product blueprint contains 5 main part which are required to have for Platform of Trust API request able to work with. Whole data-product blueprint can be seen in figure 1 where 5 import blueprints are shown in own figures.

Formatting timestamps

Unreal engine can generate timestamps easily with one function, but format of the returned timestamp was not right format to use in the requested JSON object, so it needed to be break down to values for that break date time and make date time functions was able to make timestamp without milliseconds which broke the request JSON body. In figure 2 another way that timestamp was created to make able to have offset to requested start time by offsetting days, hours, minutes. There is also option to set timestamp manually.

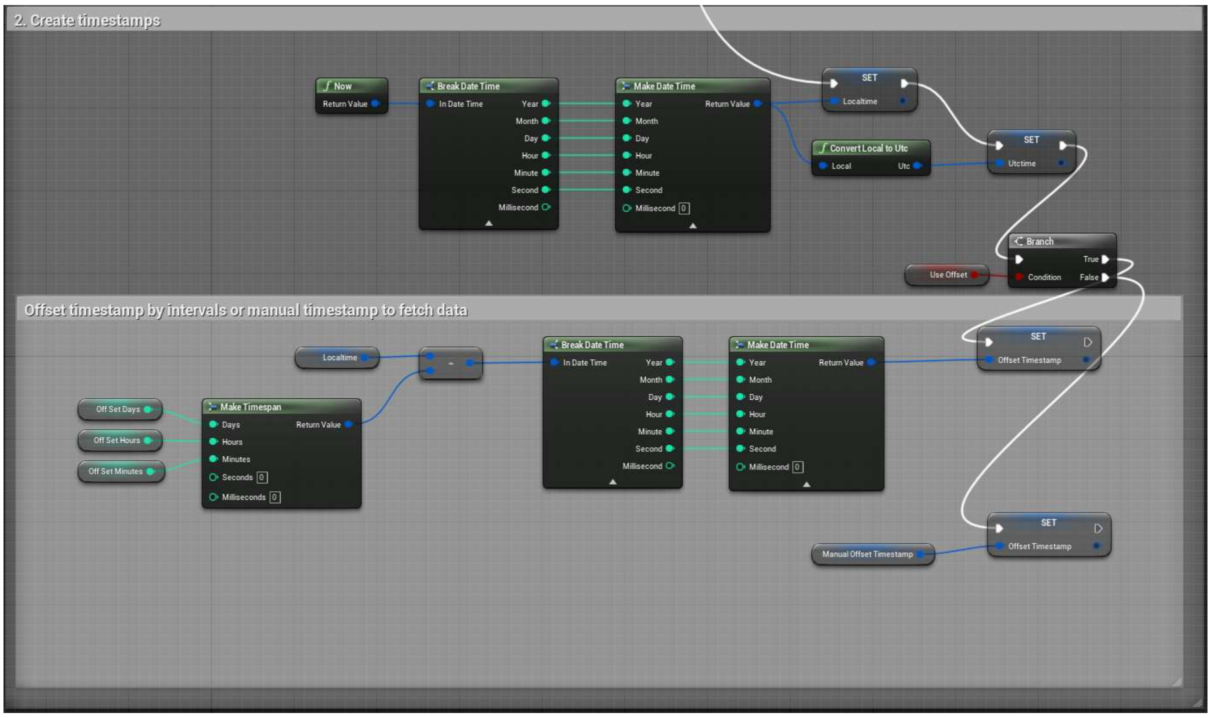


Figure 2. Format timestamps blueprint.

JSON object

Required JSON body structure is constructed with information from the API documentation fetch data product and keys shorting was manually made to match API documentation. Library functions used in this part of blueprint is VaREST and Unreal engine own library functions. Construction and order can be seen in figure 3.

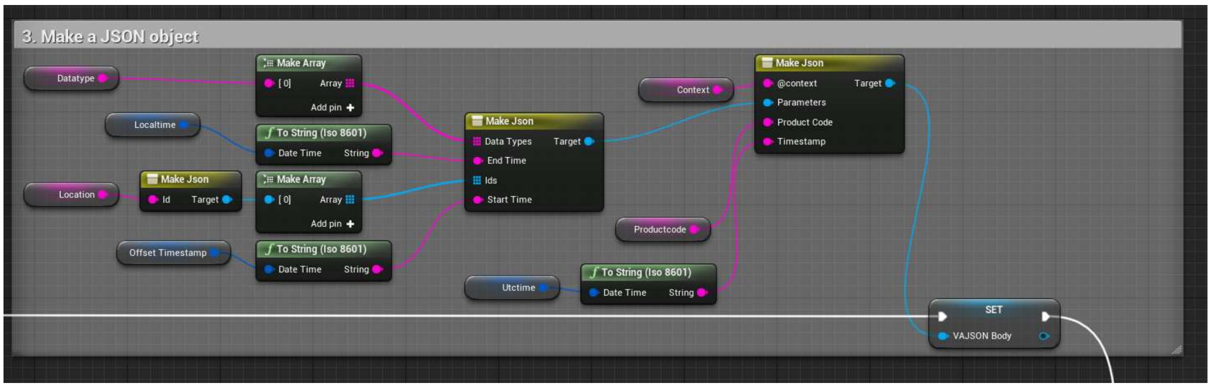


Figure 3. Make JSON body.

Creating signature string

To be able to make a signature there was need to make JSON string to have extra space which were required to have correct signature. Library functions used in this part of blueprint is LE extended standards and Unreal engine own library functions. Blueprint flow can be seen in figure 4.

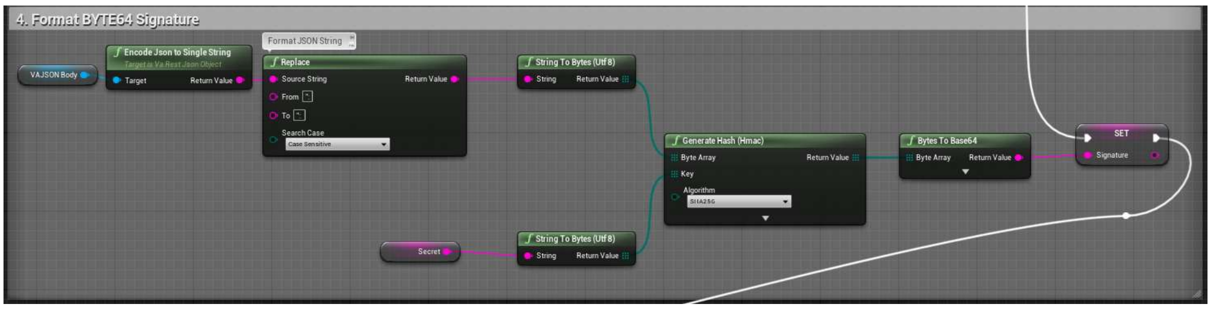


Figure 4. Create Platform of Trust signature.

HTTP request

HTTP request was made with VaREST library and constructed with information from the API documentation [fetch data product](). Most of the parameters was done in the previous steps and some are default parameters defined in the project. Request flow can be seen in the figure 5.

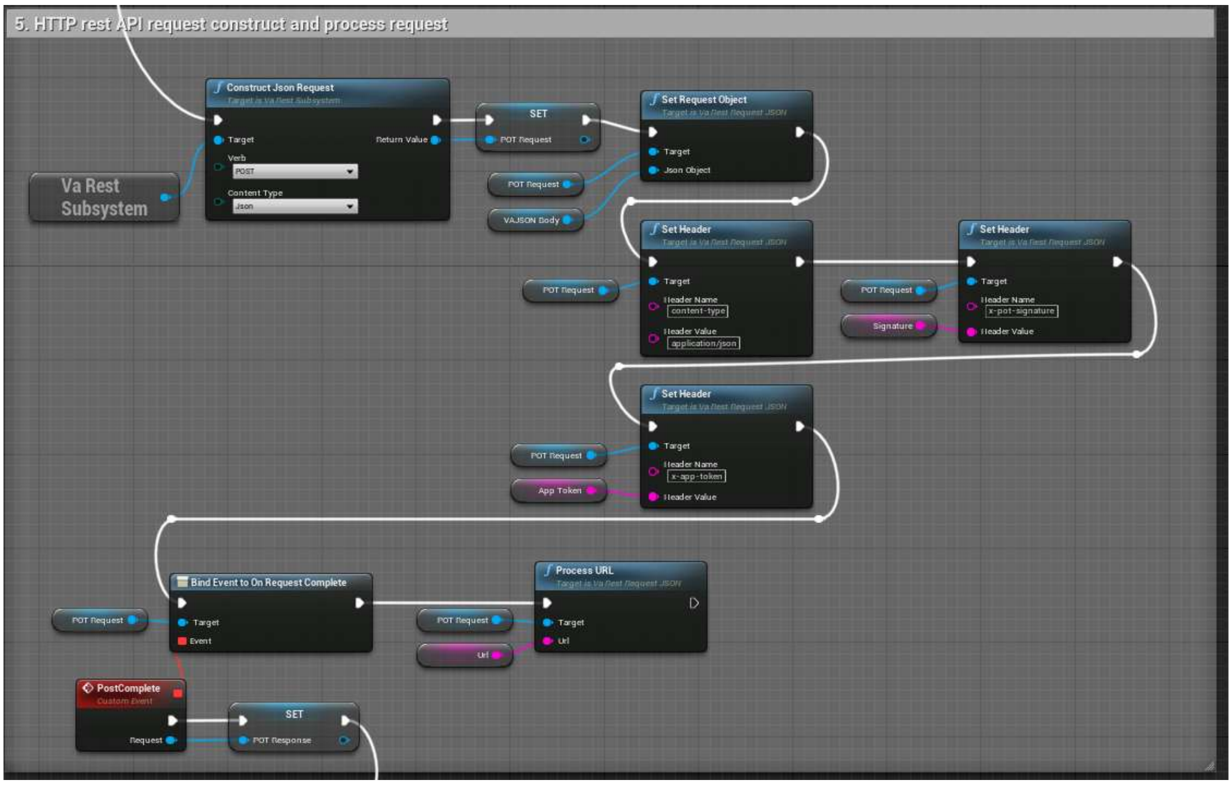


Figure 5. JSON request.

Parse response

Next step is to parse requested data response using VaRest library. There are different variables where data is saved for later usage can be seen on figure 6. Main functionalities are that blueprint:

1. Set whole requested data array as variable
2. Get latest item from requested data and set those own variables
    a. id
    b. timestamp
    c. value
    d. type

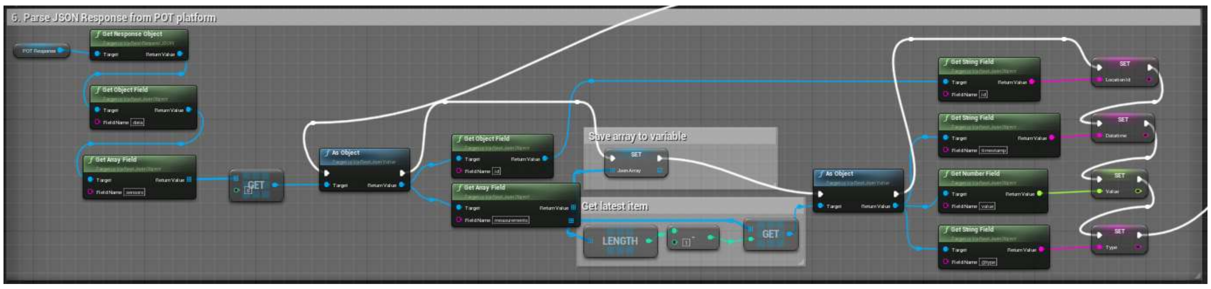


Figure 6. Parse JSON response blueprint.

Request data from product

Data request from data product is done with macro element. Figure 7 has element "Get Pot Data With Parameters" which contains parameters datatype and product code requested data product. Macro structure can be seen in figure 8. Data in this scenario is requested with 5 second delay.

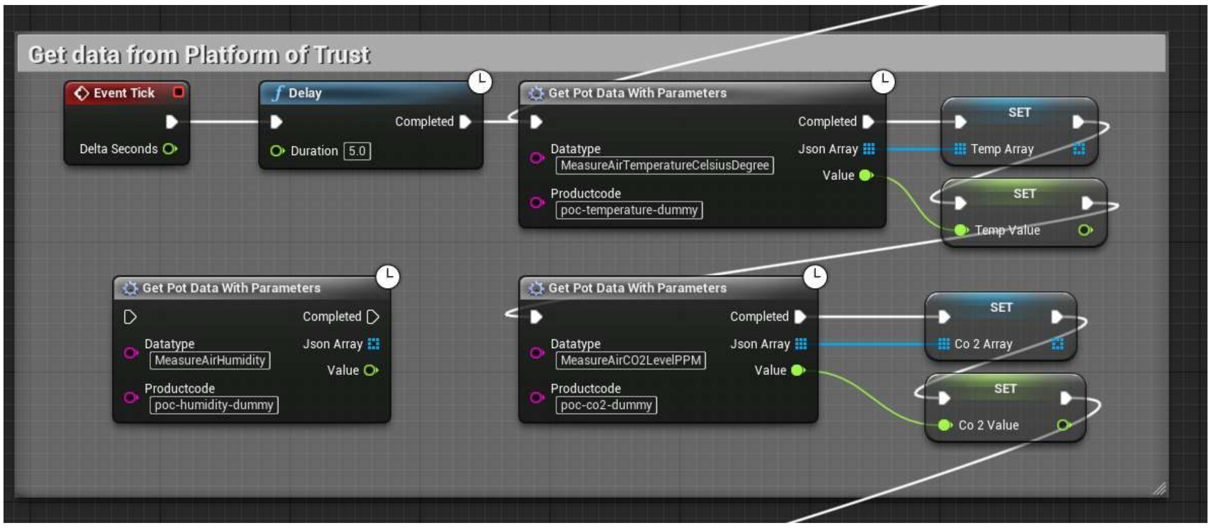


Figure 7. Get data with input parameters.

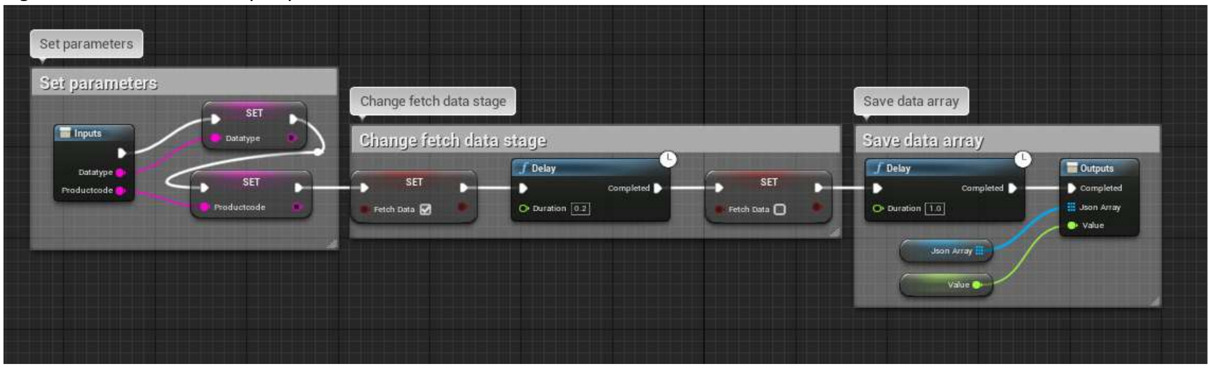


Figure 8. Get data macro.

Chart arrays

Data needs to be handled in the form that it is possible to use it in charts and for that there is in figure 9 macro called “From Chart Array” which handles inserted data array to float array which can be constructed to Line series which is utilize with Chart element in UI. Structure of macro can be seen in figure 10.

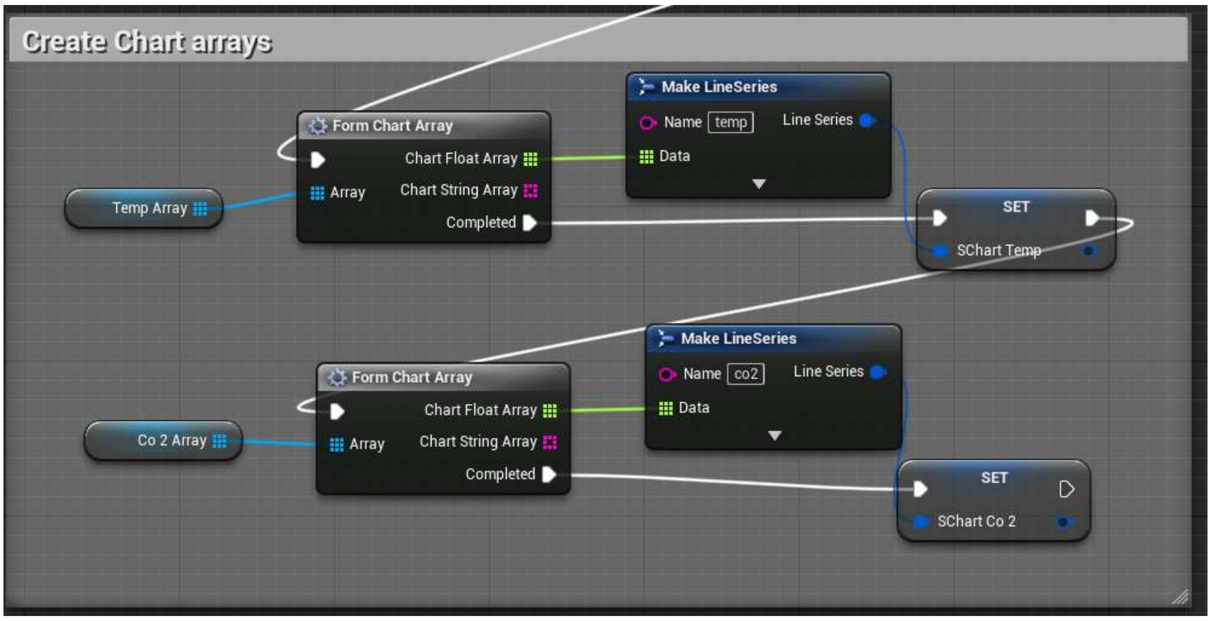


Figure 9. Create LineSeries form input array.

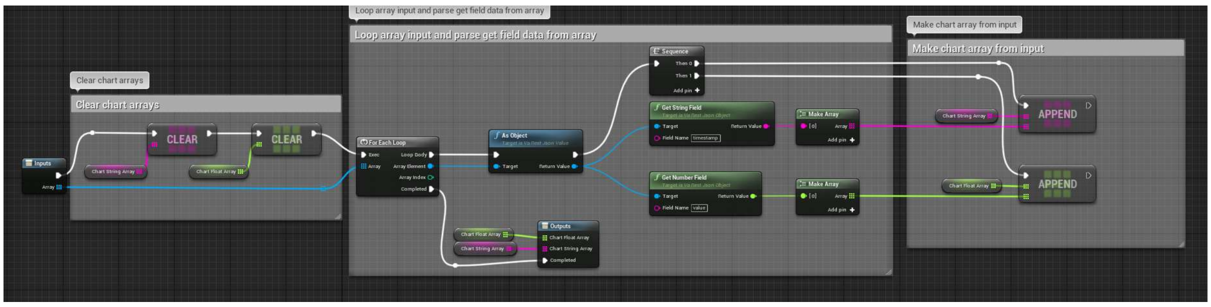


Figure 10. Create value array from data array.

Widget blueprint model

In appendix 2.

Widget blueprint figure 11 get data from the data-product and update information according of requested data. Requested data is then displayed in chart object and shown in the text element.

Widget blueprint

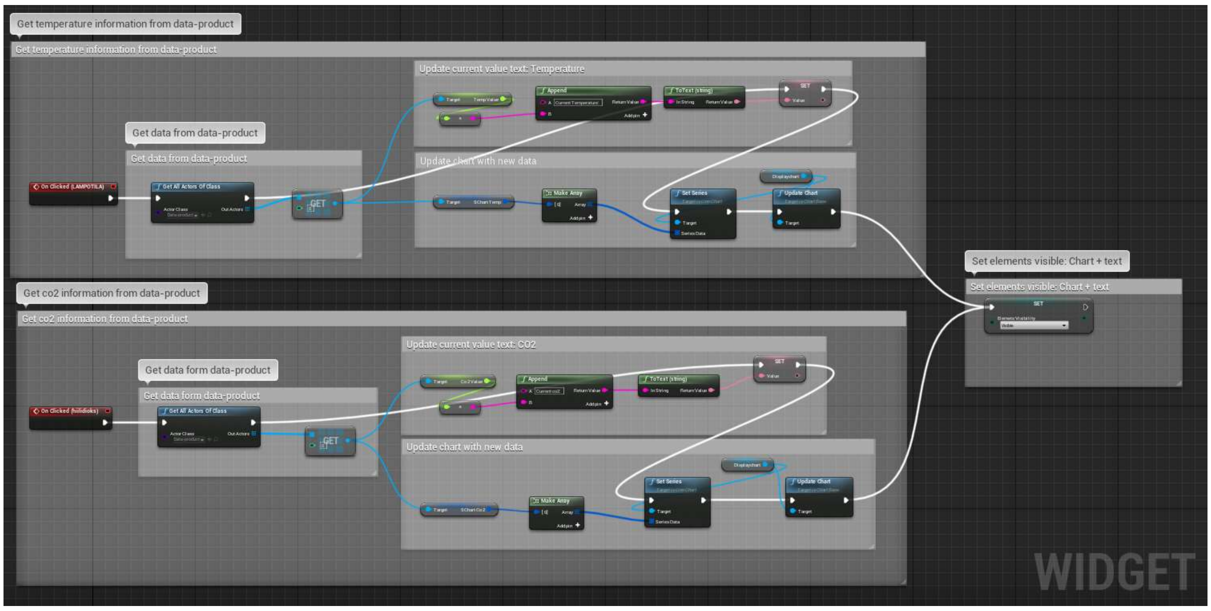

Get temperature information from data-product
Update current value text: Temperature
Get data from data-product
Update chart with new data
Set elements visible: Chart + text
Get co2 information from data-product
Update current value text: CO2
Get data form data-product
Update chart with new data
WIDGET

Figure 11. UIwidget Blueprint.

Project blueprints

Project contains two blueprints:

1. Asset blueprint at figure 12 marked as [1] is blueprint for data-product which includes data fetching form the Platform of Trust platform
2. Widget element at figure 12 marked as [2] which contains blueprint for UI element can been seen from figure 13 to display data from data-product.

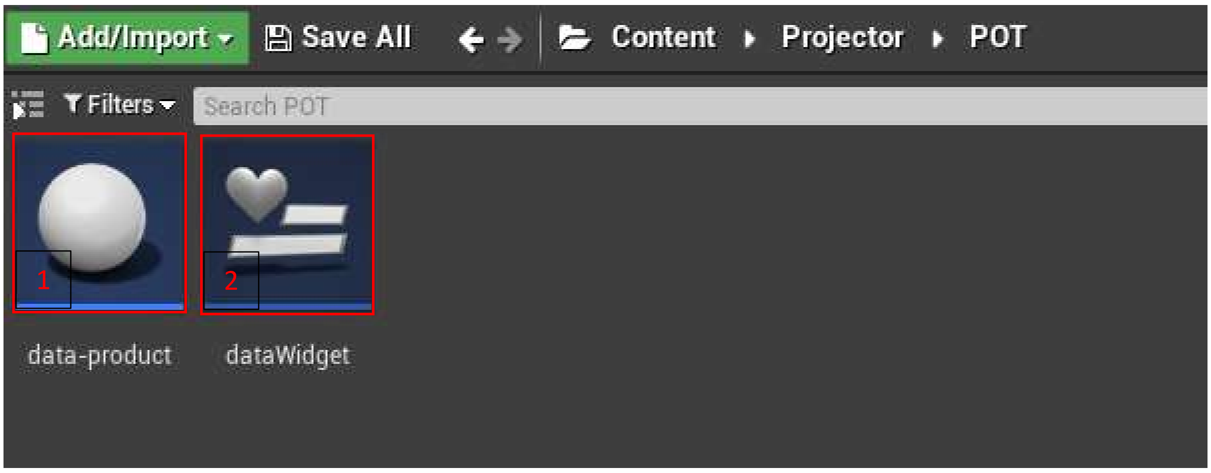


Figure 12. Project blueprints.

UI element

When data-product is reading data, it is show in the UI element [figure 14] when button is clicked. Offset by default is set to 6 hours but it can be modified from the data-product settings shown in figure 14.

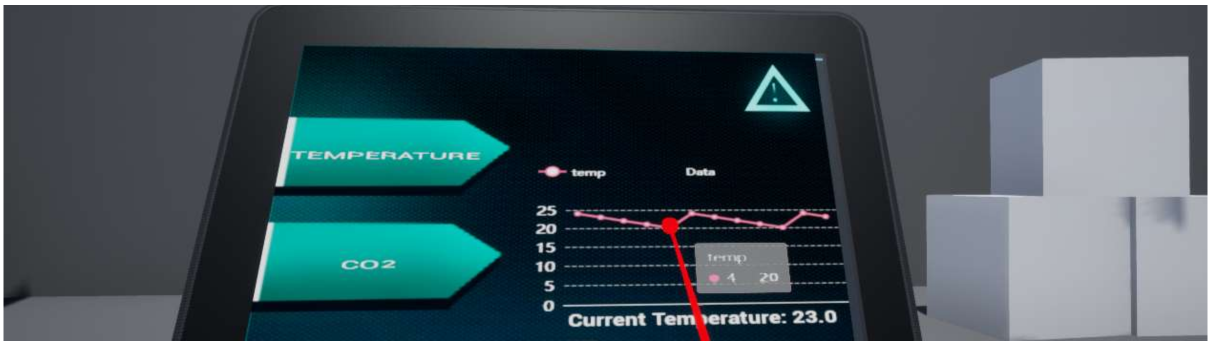
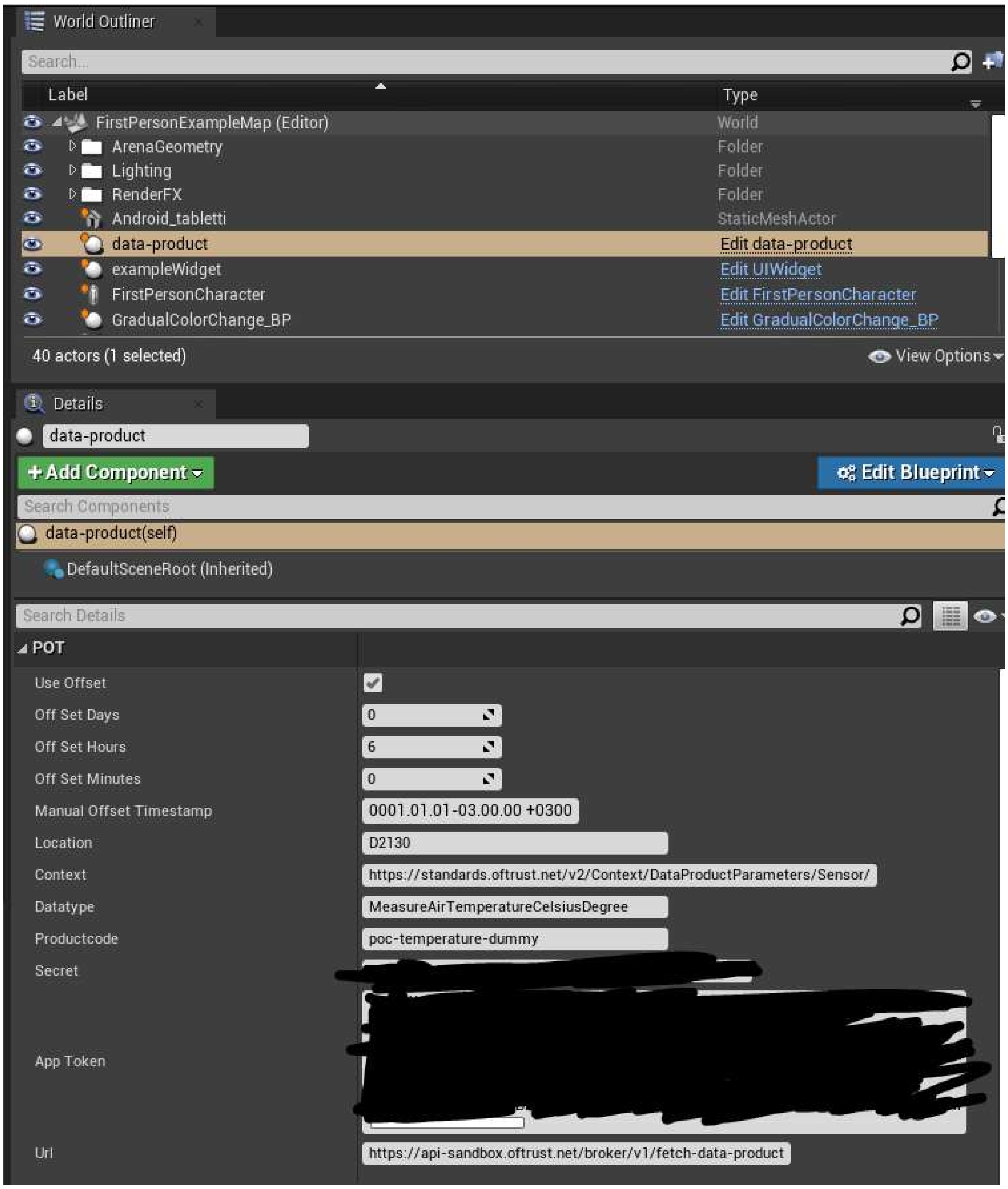


Figure 13. Widget UI in tablet.

Data-product parameters

In the data-product component parameters are shown in figure 5 which can be change are:

- Use Offset -> Boolean to set manual timestamp or offset values
- Off Set Days -> Offset Days
- Off Set Hours -> Offset Hours
- Off Set Minutes -> Offset Minutes
- Location -> data location
- Context -> URL of context
- Secret -> data-product secret
- App-token -> user app token
- URL -> API endpoint URL
- Datatype -> data-product datatype
- Productcode -> data-product code



Label	Type
FirstPersonExampleMap (Editor)	World
ArenaGeometry	Folder
Lighting	Folder
RenderFX	Folder
Android_tabletti	StaticMeshActor
data-product	Edit data-product
exampleWidget	Edit UIWidget
FirstPersonCharacter	Edit FirstPersonCharacter
GradualColorChange_BP	Edit GradualColorChange_BP

40 actors (1 selected)

Details

data-product

data-product(self)

DefaultSceneRoot (Inherited)

POT

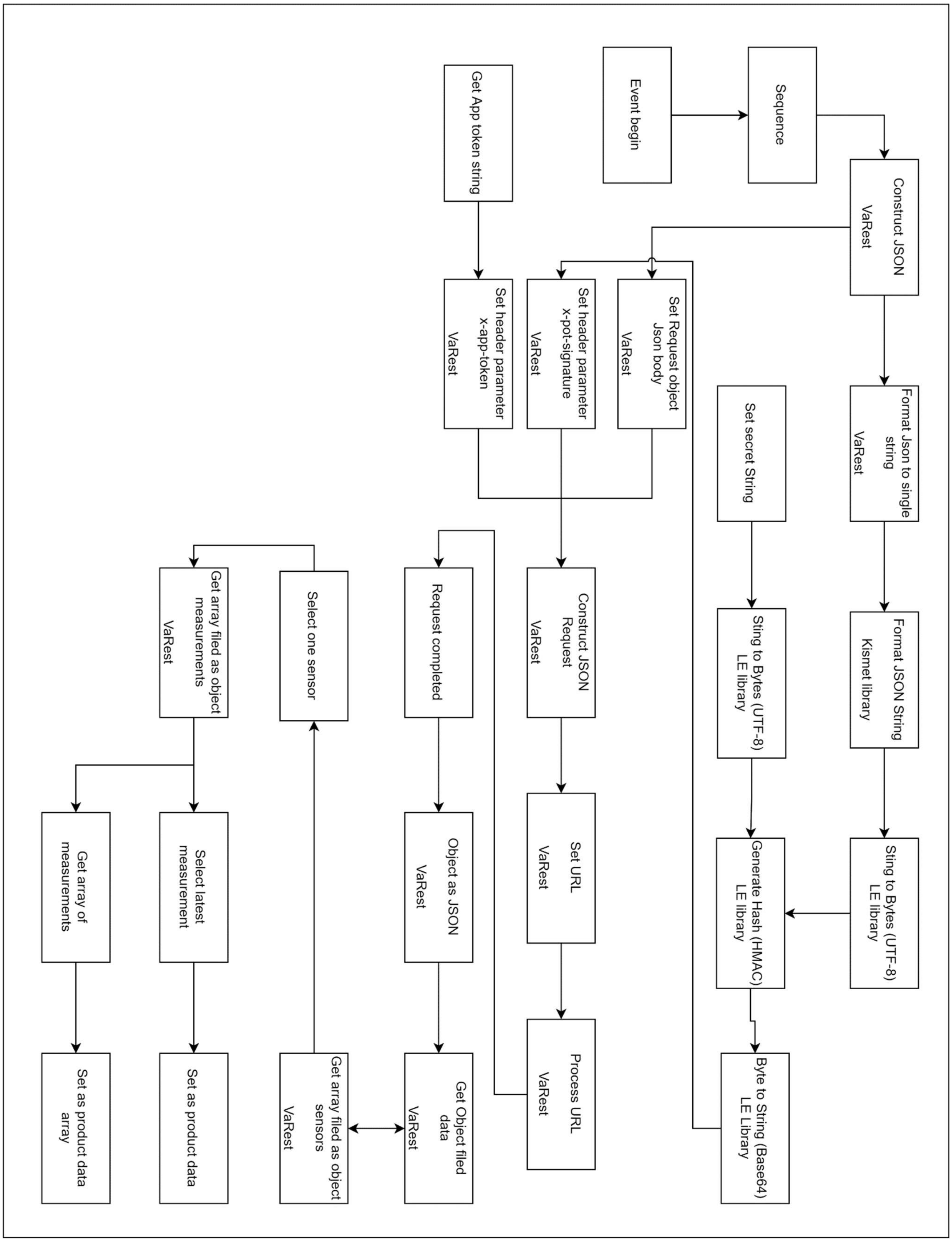
Use Offset	✓
Off Set Days	0
Off Set Hours	6
Off Set Minutes	0
Manual Offset Timestamp	0001.01.01-03.00.00 +0300
Location	D2130
Context	https://standards.oftrust.net/v2/Context/DataProductParameters/Sensor/
Datatype	MeasureAirTemperatureCelsiusDegree
Productcode	poc-temperature-dummy
Secret	[redacted]
App Token	[redacted]
Url	https://api-sandbox.oftrust.net/broker/v1/fetch-data-product

Figure 14. Data-product parameters

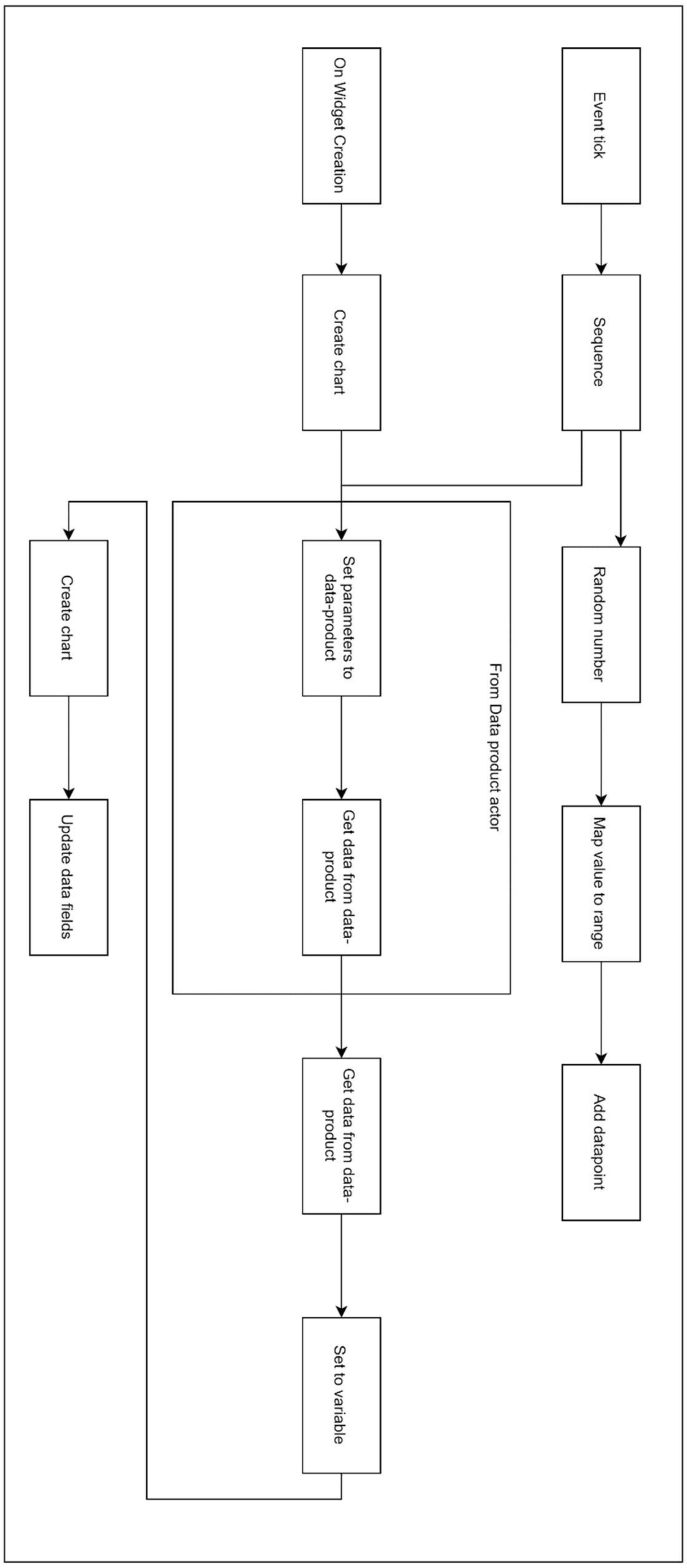
How to use it

Before data can be fetched there is need for set-up App Token and Secret which are data-product specific and make sure that product code, data type, location and URL are set to right parameters. After that data fetching is handled by the data-product blueprint.

Appendix 1. Data-Product Skeleton model


Event begin
Sequence
Construct JSON VaRest
Format Json to single string VaRest
Format JSON String Kismet library
Sting to Bytes (UTF-8) LE library
Set secret String
Sting to Bytes (UTF-8) LE library
Generate Hash (HMAC) LE library
Byte to String (Base64) LE Library
Get App token string
Set Request object Json body VaRest
Set header parameter x-pot-signature VaRest
Set header parameter x-app-token VaRest
Construct JSON Request VaRest
Set URL VaRest
Process URL VaRest
Request completed
Object as JSON VaRest
Get Object filed data VaRest
Get array filed as object sensors VaRest
Select one sensor
Get array filed as object measurements VaRest
Select latest measurement
Set as product data
Get array of measurements
Set as product data array

Appendix 2. UIWidget skeleton model


Event tick
Sequence
Random number
Map value to range
Add datapoint
On Widget Creation
Create chart
From Data product actor
Set parameters to data-product
Get data from data-product
Get data from data-product
Set to variable
Create chart
Update data fields