Systecon

Developer Interview Task – Front-End

Introduction

Systecon offers the Opus Suite tool suite to its customers. The tool suite consist of Windows base GUI applications with powerful capabilities for logistics support optimization and simulation.

A REST API has been developed around the tool suite in order to interact with the tools through the web. The API allows the user to upload and edit models, run the optimization/simulation, and present results. This enables customer specific web applications that offers a subset of the functionality of the GUI application.

Customers typically want dashboard style web applications where they can manage their models, trigger runs, and analyze the result.

Task

Your task is to develop a web dashboard with three views, where the end user can view some results in a chart, view some result in tabular form, and manipulate the model. Screenshots from the existing GUI application is provided for reference, but there is no requirement on the look and feel, as long as the same information is shown. In fact, you are encouraged to give the application a modern look and feel.

Deliverables:

- Basic wire frames for the application.
- Motivation to why the particular technology/framework was used.
- A working prototype of the application
- Demo of the application, including an overview of the code.

Setup

For the practical tasks, you are free to select the language/tool you feel familiar with and that is appropriate for the task at hand. This is a take home task that is performed on your own environment. Don’t spend more than a day, an effort of 4-8 hours is expected. It’s up to you to determine the scope, both with respect to number of views and level of detail.

Data

The data set is provided in several formats:

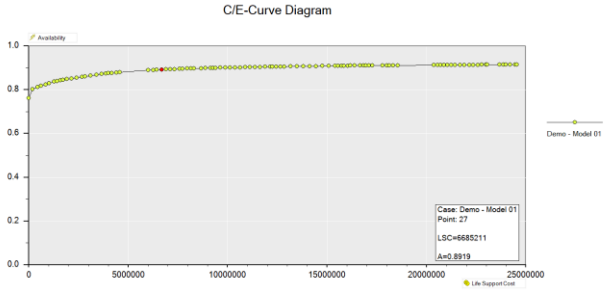
- CSV-file
- Json file
- Sqlite database

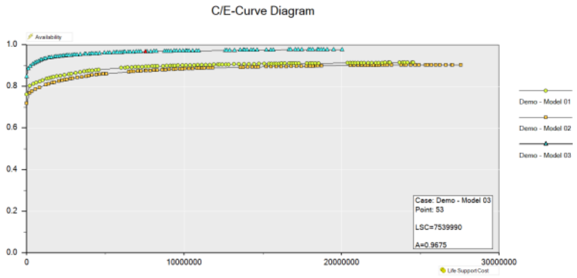
Feel free to use the data source you are most comfortable with.

Application Requirements

The web based application shall have three separate views, see the requirements for each view below. It can be a single page or multi page app.

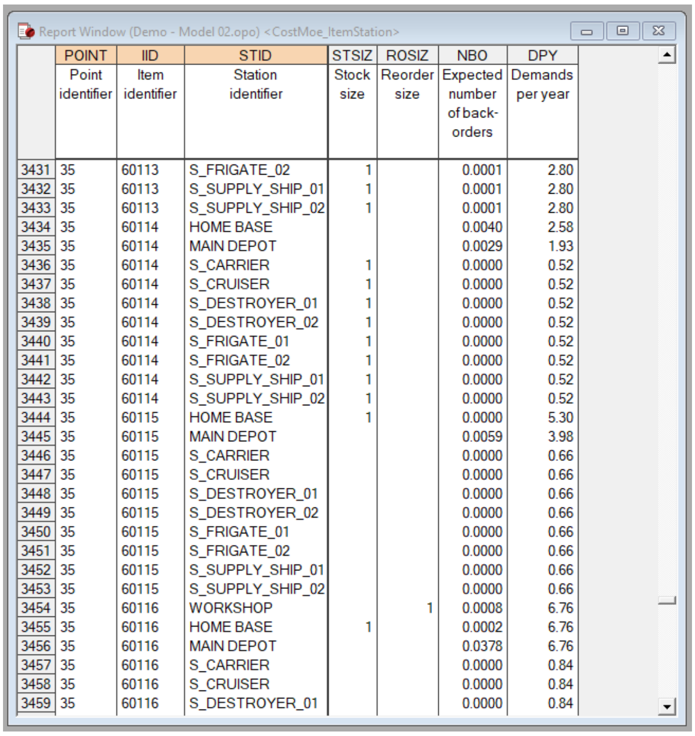
View 1: Result chart





The result chart shows cost (x-axis, denoted LSC) vs. availability (y-axis, denoted A). As this is a core element of the result analysis, it is important that curves for multiple models can be shown in the same graph (right above), as well as results for a single model (left above).

View 2: Result table



Report Window (Demo - Model 02.opo) <CostMoe_ItemStation>

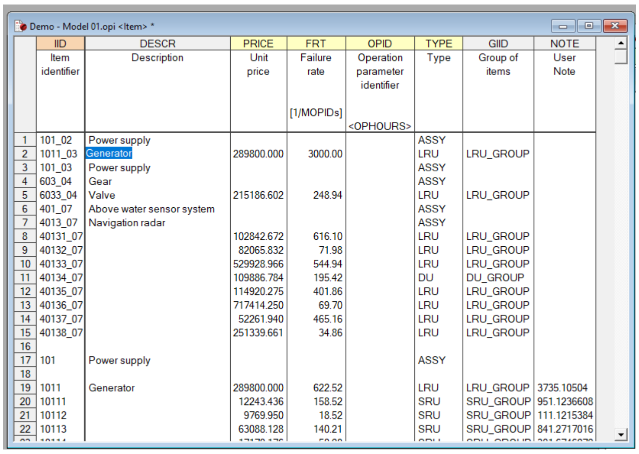
	POINT	IID	STID	STSIZ	ROSIZ	NBO	DPY
	Point identifier	Item identifier	Station identifier	Stock size	Reorder size	Expected number of back-orders	Demands per year
3431	35	60113	S_FRIGATE_02	1		0.0001	2.80
3432	35	60113	S_SUPPLY_SHIP_01	1		0.0001	2.80
3433	35	60113	S_SUPPLY_SHIP_02	1		0.0001	2.80
3434	35	60114	HOME BASE			0.0040	2.58
3435	35	60114	MAIN DEPOT			0.0029	1.93
3436	35	60114	S_CARRIER	1		0.0000	0.52
3437	35	60114	S_CRUISER	1		0.0000	0.52
3438	35	60114	S_DESTROYER_01	1		0.0000	0.52
3439	35	60114	S_DESTROYER_02	1		0.0000	0.52
3440	35	60114	S_FRIGATE_01	1		0.0000	0.52
3441	35	60114	S_FRIGATE_02	1		0.0000	0.52
3442	35	60114	S_SUPPLY_SHIP_01	1		0.0000	0.52
3443	35	60114	S_SUPPLY_SHIP_02	1		0.0000	0.52
3444	35	60115	HOME BASE	1		0.0000	5.30
3445	35	60115	MAIN DEPOT			0.0059	3.98
3446	35	60115	S_CARRIER			0.0000	0.66
3447	35	60115	S_CRUISER			0.0000	0.66
3448	35	60115	S_DESTROYER_01			0.0000	0.66
3449	35	60115	S_DESTROYER_02			0.0000	0.66
3450	35	60115	S_FRIGATE_01			0.0000	0.66
3451	35	60115	S_FRIGATE_02			0.0000	0.66
3452	35	60115	S_SUPPLY_SHIP_01			0.0000	0.66
3453	35	60115	S_SUPPLY_SHIP_02			0.0000	0.66
3454	35	60116	WORKSHOP		1	0.0008	6.76
3455	35	60116	HOME BASE	1		0.0002	6.76
3456	35	60116	MAIN DEPOT			0.0378	6.76
3457	35	60116	S_CARRIER			0.0000	0.84
3458	35	60116	S_CRUISER			0.0000	0.84
3459	35	60116	S_DESTROYER_01			0.0000	0.84

The result table view shows optimization results in a tabular format. The result table view shall at a minimum show the headers (POINT, IID, STID, …) and the corresponding data.

Good-to-have requirements are:

- Sorting
- Filtering
- Pagination (split large data set into smaller parts)

View 3: Model View Editor



Demo - Model 01.opi <Item> *

	IID	DESCR	PRICE	FRT	OPID	TYPE	GIID	NOTE
	Item identifier	Description	Unit price	Failure rate	Operation parameter identifier	Type	Group of items	User Note
				[1/MOPIDs]	<OPHOURS>			
1	101_02	Power supply				ASSY		
2	1011_03	Generator	289800.000	3000.00		LRU	LRU_GROUP	
3	101_03	Power supply				ASSY		
4	603_04	Gear				ASSY		
5	6033_04	Valve	215186.602	248.94		LRU	LRU_GROUP	
6	401_07	Above water sensor system				ASSY		
7	4013_07	Navigation radar				ASSY		
8	40131_07		102842.672	616.10		LRU	LRU_GROUP	
9	40132_07		82065.832	71.98		LRU	LRU_GROUP	
10	40133_07		529928.966	544.94		LRU	LRU_GROUP	
11	40134_07		109886.784	195.42		DU	DU_GROUP	
12	40135_07		114920.275	401.86		LRU	LRU_GROUP	
13	40136_07		717414.250	69.70		LRU	LRU_GROUP	
14	40137_07		52261.940	465.16		LRU	LRU_GROUP	
15	40138_07		251339.661	34.86		LRU	LRU_GROUP	
16								
17	101	Power supply				ASSY		
18								
19	1011	Generator	289800.000	622.52		LRU	LRU_GROUP	3735.10504
20	10111		12243.436	158.52		SRU	SRU_GROUP	951.1236608
21	10112		9769.950	18.52		SRU	SRU_GROUP	111.1215384
22	10113		63088.128	140.21		SRU	SRU_GROUP	841.2717016

The model view editor shall be able to load a table in the model (on demand or on page load), and the user shall be able to edit data in the table. For this task it is sufficient to indicate (message box or similar) that a data element has been edited. There is no need to write the edit back to the source.

For the model view editor it is sufficient to work with a single model. All tables in the model are available in the data set, but it is sufficient to work with a single table.