
Systecon

Systecon

Automatiserad Validering av LORA-resultat med hjälp av Opus Suite kommandoradsgränssnitt

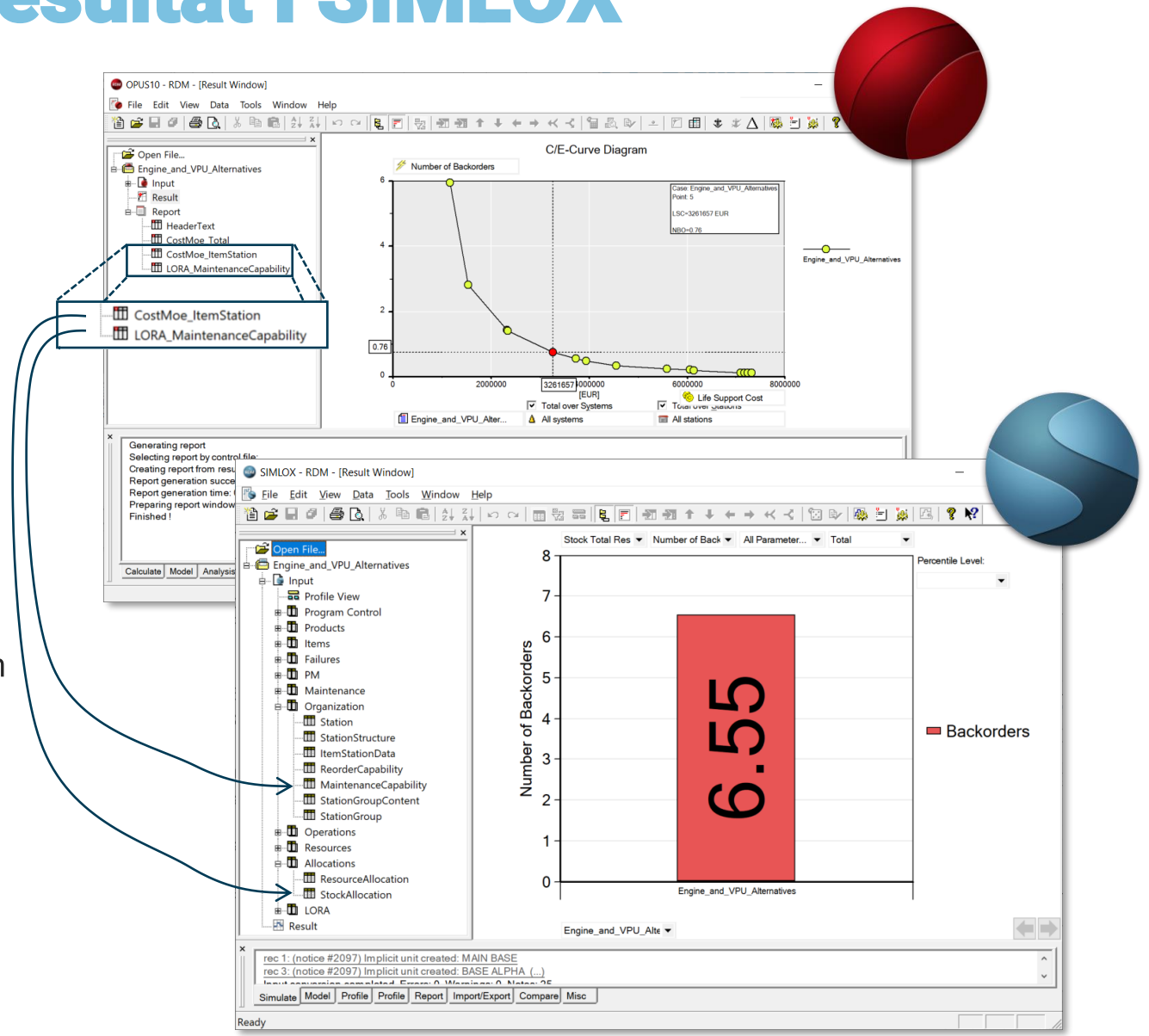
2021-05-20

Adam Duracz

Mjukvaruutvecklare

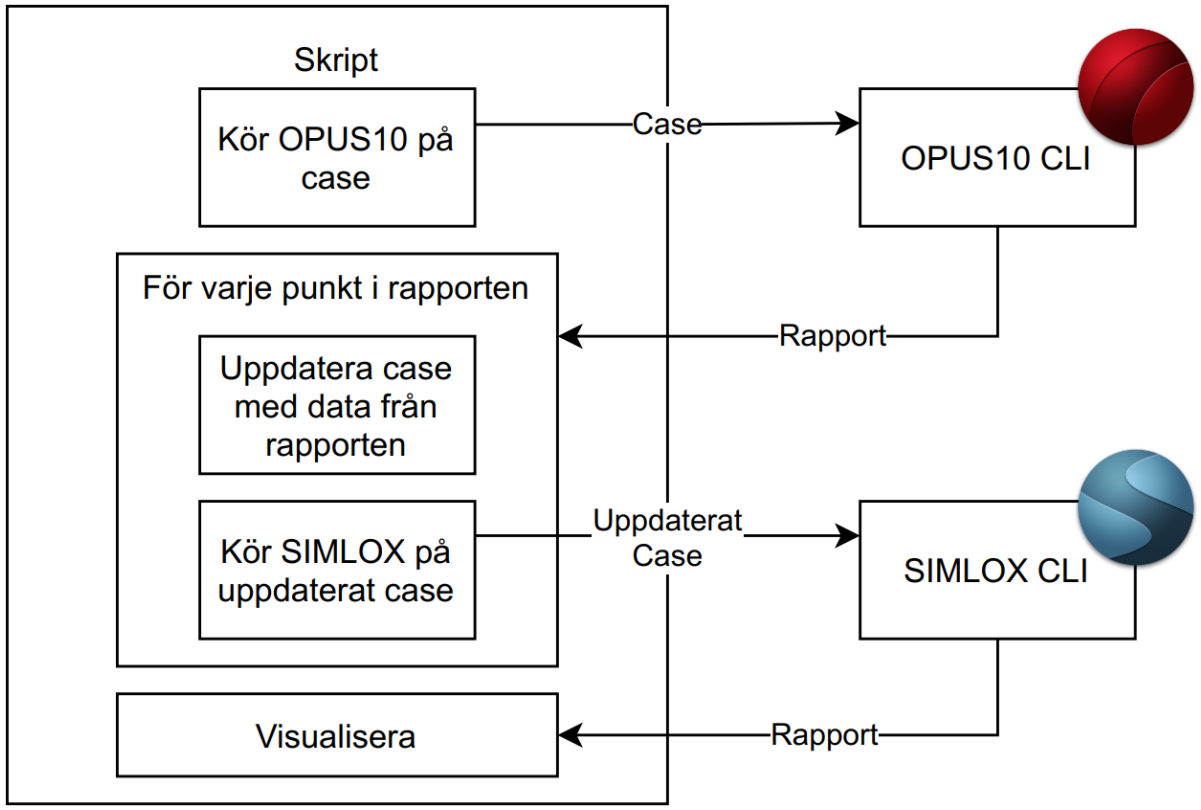
Validering av OPUS10-resultat i SIMLOX

- Den nya modellen i RDM gör att Opus Suite-applikationerna är lättare att integrera
- Ett exempel på sådan integration är att använda SIMLOX för att validera resultatmått från OPUS10
- För att t.ex. validera **NBO** för en punkt i LORA-caset vi såg nyss räcker det att:
  - Uppdatera caset genom att kopiera relevanta kolumner i alla rader för punkten
    - Underhållförmågor från rapporttabellen **LORA_MaintenanceCapability** till indatatabellen **MaintenanceCapability**
    - Lagernivåer från rapporttabellen **CostMoe_ItemStation** till indatatabellen **StockAllocation**
  - Köra caset i SIMLOX



Automatiserad validering m.h.a. Opus Suite CLI

- Vill man validera fler än en handfull punkter så tar det lång tid att göra detta för hand
- Opus Suite har ett kommandoradsgränssnitt (CLI) som går att anropa från ett skript för att automatisera sådana repetitiva uppgifter, alltså:
  - Kör caset i OPUS10 och generera en rapport
  - För varje punkt i rapporten
    - Uppdatera caset genom att kopiera relevanta kolumner i alla rader för punkten
      - Underhållförmågor från rapporttabellen LORA_MaintenanceCapability till indatatabellen MaintenanceCapability
      - Lagernivåer från rapporttabellen CostMoe_ItemStation till indatatabellen StockAllocation
    - Köra caset i SIMLOX



Vill du veta mer om hur vi tillsammans kan skapa samhällsnytta?

Rehnsgatan 20, Stockholm

08-459 07 50

info@systecon.se

systecongroup.com/se

Validering m.h.a Opus Suite CLI från Python

Location of Repair Analysis

[1]:
```
import os, pandas, cli

case = 'Engine_and_VPU_Alternatives.opi'

opus_suite = cli.OpusSuite()
report = opus_suite.opus10(case)
capabilities = report['report']['tables']['LORA_MaintenanceCapability']

pandas.DataFrame(capabilities)
```

[1]:

	POINT	COGID	SGID	LEVLID
0	1	ENGINE	REGIONAL GROUP	MAINT_LEVEL_3
1	1	VPU	REGIONAL GROUP	MAINT_LEVEL_3
2	2	ENGINE	REGIONAL GROUP	MAINT_LEVEL_3
3	2	VPU	REGIONAL GROUP	MAINT_LEVEL_3
4	3	ENGINE	CENTRAL	MAINT_LEVEL_3
...	...	...	...	...
75	38	VPU	REGIONAL GROUP	MAINT_LEVEL_2
76	39	ENGINE	CENTRAL	MAINT_LEVEL_3
77	39	VPU	REGIONAL GROUP	MAINT_LEVEL_3
78	40	ENGINE	CENTRAL	MAINT_LEVEL_3
79	40	VPU	REGIONAL GROUP	MAINT_LEVEL_2

80 rows × 4 columns

Validering m.h.a Opus Suite CLI från Python

[2]:
```python
# Generate report for each maintenance capability alternative produced by the LORA run in SIMLOX
point_reports = []
points = list(dict.fromkeys([ p['POINT'] for p in capabilities ]))
for point in points:

    print(f'\r[{"■"*int(point)}{" "*(len(points)-int(point))}] Executing SIMLOX for LORA point {point}', end='')

    point_input = opus_suite.load_forms(case)

    # Set active point in the Control table
    point_input['model']['tables']['Control'][0].update({ 'APID' : point })

    # Insert capabilities into MaintenanceCapability table
    for capability in capabilities:
        if capability['POINT'] == point:
            point_input['model']['tables']['MaintenanceCapability'].append({
                'COGID'  : capability['COGID']  ,
                'SGID'   : capability['SGID']   ,
                'LEVLID' : capability['LEVLID'] ,
                'ENDBL'  : 'N'                  ,
                'NOTE'   : ''
            })

    # Insert StockAllocation table entries for this point
    point_input['model']['tables']['StockAllocation'] = [
        { c : row.get(c) for c in ['POINT','IID','STID','STSIZ','ROSIZ', 'AINST', 'ISTOH', 'NOTE'] }
        for row in report['report']['tables']['CostMoe_ItemStation']
        if row['POINT'] == point
    ]

    # Save the updated input case as a file and run it in SIMLOX
    point_input_path = os.path.join('.','data',f'input_{point}.opi')
    opus_suite.store_forms(point_input, point_input_path)
    point_report = opus_suite.simlox(point_input_path)

    point_reports.append(point_report)

print(f'\rDone computing results for all {len(points)} LORA points.' + ' '*100, end='')
```

```
Done computing results for all 40 LORA points.
```

Validering m.h.a Opus Suite CLI från Python

[3]:
```python
import plotly.graph_objects as go

moe = 'NBO'

x      = [ row['LSC'] for row in report['report']['tables']['CostMoe_Total'] ]
y_lora = [ row[moe]   for row in report['report']['tables']['CostMoe_Total'] ]
y_simlox = [
    [ row[moe] for row in point_report['report']['tables']['StockTotalResultsTotal'] ][0]
    for point_report in point_reports
]

fig = go.Figure()
fig.add_trace(go.Scatter(name='OPUS10 LORA'      , x=x, y=y_lora  ))
fig.add_trace(go.Scatter(name='SIMLOX Validation', x=x, y=y_simlox))
fig.show()
```