Plan of Actions – Important enhancements to the

[3:33 PM] Sanket Taur

<https://sdlcwiki.electrolux.com/display/CCTPRG/Spare+parts+prediction>

[3:43 PM] Sanket Taur

<https://sdlc.electrolux.com/secure/RapidBoard.jspa?rapidView=11843&projectKey=CCTPRG&view=planning&selectedIssue=CCTPRG-14166&issueLimit=100>

[3:46 PM] Sanket Taur

<https://sdlcwiki.electrolux.com/display/CCTPRG/Architecture+and+Design+Patterns>

1- Reduce unnecessary executions, by reducing the number of functions per request or per usecase

Per our design only 2 are needed, and in a stateless solution like the one we currently have, only 1 function is needed

Diagram

Description automatically generated

2- Naming convention and standards for the use cases, function names, resource names and APIs

c4c\_isp\_patch\_fn,c4c\_isp\_get\_fn,execute\_rule\_fn,c4c\_isp\_update\_to\_isp2

c4c\_isp2\_isp4\_patch,c4c\_isp2\_isp4\_get\_fn,execute\_rule\_function \_isp2\_isp4,c4c\_isp2\_update\_to\_isp4

Names should be meaningful and representative. --🡪 IN PROGRESS

Diagram

Description automatically generated

3- Transform the functions to stateful using mongo collection to store events and trigger action or reaction

DB Collection(s) for C4C SBS Extension Events and Error handling

4- Refactor functions and have a shared interface and different triggers for the usecases/functions

C4CEvent extension store/filter function --> Common Interface

C4CEvent extension Facts function --> For each use case or shared

Diagram

Description automatically generated

[Yesterday 4:34 PM] Sanket Taur

1- Reduce unnecessary executions, by reducing the number of functions per request or per usecase

Per our design only 2 are needed, and in a stateless solution like the one we currently have, only 1 function is needed

Diagram

Description automatically generated

2- Naming convention and standards for the usecases, function names, resource names and APIs

c4c\_isp\_patch\_fn,c4c\_isp\_get\_fn,execute\_rule\_fn,c4c\_isp\_update\_to\_isp2

c4c\_isp2\_isp4\_patch,c4c\_isp2\_isp4\_get\_fn,execute\_rule\_function \_isp2\_isp4,c4c\_isp2\_update\_to\_isp4

Names should be meaningful and representative.

Diagram

Description automatically generated

3- Transform the functions to statefull using mongo collection to store events and trigger action or reaction

DB Collection(s) for C4C SBS Extension Events and Error handling

4- Refactor functions and have a shared interface and different triggers for the usecases/functions

C4CEvent extension store/filter function --> Common Interface

C4CEvent extension Facts function --> For each usecase or shared

Diagram

Description automatically generated

5- Use values to store all configurations/data for the different usecases

Here below an example from spare parts prediction; wjere all c4c request configs are stored:url,collection,query,filter --- > **IN PROGRESS**

Graphical user interface, text, application, email

Description automatically generated

6- Use Azure repo to store functions and other resources for the different usecases

7- Add unit testing and integration testing to the different usecases

please explore

<https://jestjs.io/> for unit testing

[newman-contract - npm (npmjs.com)](https://www.npmjs.com/package/newman-contract) for contract testing

8- Create cicd pipelines (containing tests) to atlas cloud

Jest