

Decision Management Community Challenge

July 2022

Seth Meldon

Progress Corticon













Prompt:

Team	Player	Game Date	Efficiency
Mustungs	Brown	4/1/2022	good
	Brown	4/2/2022	better
	Brown	4/3/2022	best
	Robinson	4/1/2022	worst
	Robinson	4/2/2022	better
	Robinson	4/3/2022	best
	Smith	4/1/2022	bad
	Smith	4/2/2022	good
	Smith	4/3/2022	bad
Eagles	Black	4/1/2022	good
	Black	4/2/2022	better
	Black	4/3/2022	best
	White	4/1/2022	worst
	White	4/2/2022	better
	White	4/3/2022	best
	Green	4/1/2022	bad
	Green	4/2/2022	good
	Green	4/3/2022	worst

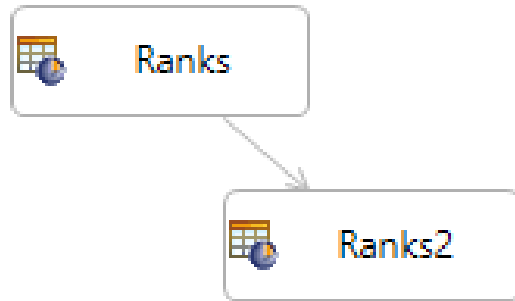
Your decision model should evaluate performance of different teams based on efficiency of their players. On the left you can see how different players performed during different games. Each player receives 5 points for “best” efficiency, 3 points for “better” efficiency, 2 for “good” efficiency. You need to subtract 2 points for “bad” efficiency and 5 points for “worst” efficiency. Which team got the most points?

Please [submit](#) your solutions using your favorite BR/DM tools.

First we'll define a data model to represent the various data fields, this will be defined in Corticon **Rule Vocabulary**:

- ▼  Game
 -  date
 -  efficiency
 -  efficiencyPoints
 - player (Player)
- ▼  Player
 -  efficiencyPoints
 -  name
 -  teamName
 - ⬅ game (Game)
 - team (Team)
- ▼  Team
 -  efficiencyPoints
 -  isWinner
 -  name
 - ⬅ player (Player)

The decision service is compiled from a **Ruleflow**, which contains two **Rulesheets**:



Rules are divided into Conditions and Actions in a rulesheet. Each column is a discrete rule contains any number of conditions (e.g. when Game.efficiency = 'Good') that may produce any number of actions (continuing the same example, we assign the value of 2 to the field Game.efficiencyPoints). Column 0 is an 'action only' rule, so it will always fire.

Blank rulesheet:

The screenshot displays a software interface for rule creation. On the left, there is a 'Rule Vocabulary' tree with categories like 'Team Vocab', 'Attribute Operators', 'Entity/Association Operators', and 'General'. A blue arrow labeled 'Rule Vocabulary' points to this tree. The main area is a 'Conditions' table with columns labeled 0, 1, 2, 3, and 4. A blue arrow labeled 'Rule #1' points to the first column (column 1). A black box highlights the first column of the 'Conditions' table. Below the 'Conditions' table is an 'Actions' table with a header 'Post Message(s)' and rows labeled A through I. At the bottom, there is a 'Rule Statements' table with columns 'Ref', 'ID', 'Post', 'Alias', and 'Text'. The interface also includes a 'Natural Language' button and various toolbars.

For both documentation and to simplify new user onboarding, we optionally can define Natural Language values to overlay over the completed syntax in a rulesheet:

The screenshot displays a rule editor interface with several panels:

- Team Vocab:** A tree view showing 'Team' as a child of 'Player', which is a child of 'Game'.
- Scope:** A list of entities: Game, Player, Team [Eagles], and Team [Mustungs].
- Filters:** Two filter rules:
 - Create an alias for all instances of Team where its child entity, Player, has a value for its teamName attribute of 'Mustungs'
 - Create an alias for all instances of Team where its child entity, Player, has a value for its teamName attribute of 'Eagles'
- Rule Operators:** A tree view of operators including Boolean, DateTime, Decimal, Integer, String, Entity/Association Operators, Collection, Entity, Sequence, General, Functions, and Literals.
- Conditions Table:**

	0	1	2	3	4
a		'worst'	'bad'	'good'	'better'
b					
c					
d					
e					
f					
~					
- Actions Table:**

	0	1	2	3	4
A	0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
B		-5	-2	2	3
C	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
D	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
E	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
F					
- Rule Statements Table:**

Ref	ID	Post	Alias	Text
A0		Info	Game	Initialize efficiency points for an individual game to be 0
1:5		Info	Game	Game assigned efficiency points of {{Game.efficiencyPoints}} based upon the efficiency category {{Game.efficiency}}
C0		Info	Player	Player efficiency points set to be the sum of their individual games' efficiency points: {{Player.efficiencyPoints}}
D0		Info	Mustungs	Mustungs net efficiency points calculated to be sum of Mustung players' net efficiency points {{Mustungs.efficiencyPoints}}
E0		Info	Eagles	Eagles net efficiency points calculated to be sum of Eagles players' net efficiency points {{Eagles.efficiencyPoints}}

These overlays are different Rule Statements. Rule Statements, while optional, are quite useful for rule observability. They will be sent back to the invoking application for all of the rules that fire—and only the rules that fire. This provides a clear description of each rule that is executed.

The second rulesheet is simpler, but also makes use of a concept called Filtered Aliases to represent how Corticon should handle different scenarios involving the same entity/attributes. We can thus define rule rules applicable only to the entity 'Team' only where its underlying 'teamName' attribute equals 'Eagles' and others for where it equals 'Mustungs'.

The screenshot shows the Corticon Rulesheet Editor interface for a rulesheet named 'Ranks2.ers'. The interface is divided into several sections:

- Scope:** A tree view on the left showing the project structure. It includes entities like 'Game', 'Player', and 'Team'. Under 'Team', there are two filtered aliases: 'Team [Eagles]' and 'Team [Mustungs]', each with its own set of filters and attributes like 'efficiencyPoints', 'isWinner', and 'name'.
- Conditions:** A table with columns for conditions (a-f) and rows for values 0, 1, and 2. Condition 'a' is defined as 'Eagles.efficiencyPoints > Mustungs.efficiencyPoints'.
- Actions:** A table with columns for actions (A-C) and rows for values 0, 1, and 2. Action 'A' is 'Mustungs.isWinner', 'B' is 'Eagles.isWinner', and 'C' is 'Post Message(s)'. The 'Post Message(s)' action is triggered in rows 1 and 2.
- Filters:** A list of filters for the filtered aliases: '1 Eagles.name = 'Eagles'' and '2 Mustungs.name = 'Mustungs''.
- Rule Statements:** A table at the bottom showing the generated rule statements. It has columns for Ref, ID, Post, Alias, and Text.

Ref	ID	Post	Alias	Text
1		Info	Eagles	Eagles are the winner with {{Eagles.efficiencyPoints}} points
2		Info	Mustungs	Mustungs are the winner with {{Mustungs.efficiencyPoints}} points

Prior to deploying the rules as a service we can define a rule test to validate the outputs produced by the rules for test inputs:

Prior to running test:

The screenshot displays a software interface for configuring a rule test. The main workspace is a table with three columns: 'Input', 'Output', and 'Expected'. The 'Input' column contains a hierarchical tree structure representing test data for two teams. The 'Output' and 'Expected' columns are currently empty. The interface includes a left sidebar with a 'Team Vocab' tree, a bottom toolbar with various tool icons, and a bottom status bar with a table for 'Severity' and 'Message'.

Team Vocab

- Game
- Player
- Team

Input

- Team [1]
 - efficiencyPoints
 - name [Mustungs]
 - player (Player) [1]
 - player (Player) [4]
 - name [Robinson]
 - teamName [Mustungs]
 - game (Game) [4]
 - date [4/1/2022]
 - efficiency [worst]
 - game (Game) [5]
 - date [4/2/2022]
 - efficiency [better]
 - game (Game) [6]
 - date [4/3/2022]
 - efficiency [best]
 - player (Player) [7]
- Team [2]
 - efficiencyPoints
 - name [Eagles]
 - player (Player) [10]
 - name [Black]
 - teamName [Eagles]
 - game (Game) [10]
 - game (Game) [11]
 - game (Game) [12]
 - date [4/3/2022]
 - efficiency [best]

Output

Expected

Rule Operators

- Attribute Operators
- Entity/Association Operators
- General

Rule Statements **Rule Messages** **Rule Trace** **Properties** **Problems** **Progress** **Comments** **Natural Language**

Severity	Message	Ent

After running test:

untitled_1

/Challenge July-2022/flow.erf Differences: 0

Input	Output	Expected
<ul style="list-style-type: none"> Team [1] <ul style="list-style-type: none"> efficiencyPoints name [Mustungs] player (Player) [1] player (Player) [4] <ul style="list-style-type: none"> name [Robinson] teamName [Mustungs] game (Game) [4] <ul style="list-style-type: none"> date [4/1/2022] efficiency [worst] game (Game) [5] <ul style="list-style-type: none"> date [4/2/2022] efficiency [better] game (Game) [6] <ul style="list-style-type: none"> date [4/3/2022] efficiency [best] player (Player) [7] Team [2] 	<ul style="list-style-type: none"> Team [1] <ul style="list-style-type: none"> efficiencyPoints [11] isWinner [true] name [Mustungs] player (Player) [1] player (Player) [4] <ul style="list-style-type: none"> efficiencyPoints [3] name [Robinson] teamName [Mustungs] game (Game) [4] <ul style="list-style-type: none"> date [2022-04-01T00:00:00-0400] efficiency [worst] efficiencyPoints [-5] game (Game) [5] <ul style="list-style-type: none"> date [2022-04-02T00:00:00-0400] efficiency [better] game (Game) [6] 	

Rule Statements Rule Messages Rule Trace Properties Problems Progress Comments Natural Language

Severity	Message	Entity
Info	Game assigned efficiency points of [5] based upon the efficiency category [best]	Game[6]
Info	Game assigned efficiency points of [5] based upon the efficiency category [best]	Game[12]
Info	Game assigned efficiency points of [5] based upon the efficiency category [best]	Game[15]
Info	Player efficiency points set to be the sum of their individual games' efficiency points: [10]	Player[1]
Info	Player efficiency points set to be the sum of their individual games' efficiency points: [3]	Player[4]
Info	Player efficiency points set to be the sum of their individual games' efficiency points: [-2]	Player[7]
Info	Player efficiency points set to be the sum of their individual games' efficiency points: [10]	Player[10]
Info	Player efficiency points set to be the sum of their individual games' efficiency points: [3]	Player[13]
Info	Player efficiency points set to be the sum of their individual games' efficiency points: [-5]	Player[18]
Info	Mustungs net efficiency points calculated to be sum of Mustung players' net efficiency points [11]	Team[1]
Info	Eagles net efficiency points calculated to be sum of Eagles players' net efficiency points [8]	Team[2]
Info	Mustungs are the winner with [11] points	Team[1]

If we toggle to the 'Rule Trace' panel, we can see precisely what changes the rules made to the input payload, the previous/new values, and the chronology of these changes:

/Challenge July-2022/flow.erf Differences: 0

Input

- Team [1]
 - efficiencyPoints
 - name [Mustungs]
 - player (Player) [1]
 - player (Player) [4]
 - name [Robinson]
 - teamName [Mustungs]
 - game (Game) [4]
 - date [4/1/2022]
 - efficiency [worst]

Output

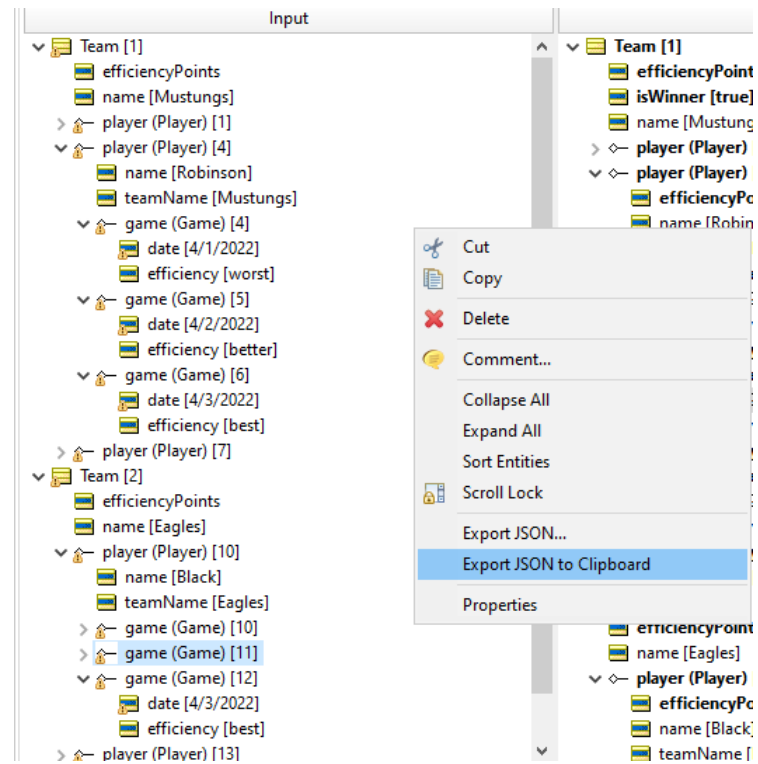
- Team [1]
 - efficiencyPoints [11]
 - isWinner [true]
 - name [Mustungs]
 - player (Player) [1]
 - player (Player) [4]
 - efficiencyPoints [3]
 - name [Robinson]
 - teamName [Mustungs]
 - game (Game) [4]
 - date [2022-04-01T00:00:00-0400]

Expected

Rule Statements
Rule Messages
Rule Trace
Properties
Problems
Progress
Comments
Natural Language

Sequence	Action	Element	Old Value	New Value	Association Entity	Location
26	Update Attribute	game (Game) [8]/efficiencyPoints	0	2		Ranks : 3
27	Update Attribute	game (Game) [10]/efficiencyPoints	0	2		Ranks : 3
28	Update Attribute	game (Game) [17]/efficiencyPoints	0	2		Ranks : 3
29	Update Attribute	game (Game) [2]/efficiencyPoints	0	3		Ranks : 4
30	Update Attribute	game (Game) [5]/efficiencyPoints	0	3		Ranks : 4
31	Update Attribute	game (Game) [11]/efficiencyPoints	0	3		Ranks : 4
32	Update Attribute	game (Game) [14]/efficiencyPoints	0	3		Ranks : 4
33	Update Attribute	game (Game) [3]/efficiencyPoints	0	5		Ranks : 5
34	Update Attribute	game (Game) [6]/efficiencyPoints	0	5		Ranks : 5
35	Update Attribute	game (Game) [12]/efficiencyPoints	0	5		Ranks : 5
36	Update Attribute	game (Game) [15]/efficiencyPoints	0	5		Ranks : 5
37	Update Attribute	player (Player) [1]/efficiencyPoints		10		Ranks : C0
38	Update Attribute	player (Player) [4]/efficiencyPoints		3		Ranks : C0
39	Update Attribute	player (Player) [7]/efficiencyPoints		-2		Ranks : C0
40	Update Attribute	player (Player) [10]/efficiencyPoints		10		Ranks : C0
41	Update Attribute	player (Player) [13]/efficiencyPoints		3		Ranks : C0
42	Update Attribute	player (Player) [18]/efficiencyPoints		-5		Ranks : C0
43	Update Attribute	Team [1]/efficiencyPoints		11		Ranks : D0
44	Update Attribute	Team [2]/efficiencyPoints		8		Ranks : E0
45	Update Attribute	Team [1]/isWinner		true		Ranks2 : 2

To test the JavaScript bundle we create, we can export the JSON from the input to the clipboard. This will be revisited momentarily.



Finally, we can generate JavaScript bundle which will run directly within a browser so that you can test the performance of the logic for yourself.

The screenshot shows a dialog box titled "Package Rules for Deployment" with the subtitle "Generate Ruleflow for JavaScript". Below the subtitle is the instruction "Select the ruleflow and target platform for JavaScript Deployment".

The dialog contains the following fields and options:

- Ruleflow:** A dropdown menu with the selected value "Challenge July-2022\flow.erf".
- Target platform:** A dropdown menu with "Browser" selected. The dropdown is open, showing a list of options: "Browser", "AWS Lambda", "Azure Functions", "Google Cloud Functions", and "Node".
- Bundle name:** An empty text input field.
- To directory:** An empty text input field.

At the bottom of the dialog, there are four buttons: "< Back", "Next >", "Finish", and a partially visible button on the right. The "Finish" button is highlighted with a blue border.

The output is a decisionServiceBundle.js file and a sample HTML page invoking the JavaScript based upon test data. These can be accessed online using Codepen [here](#). The page's 'input' side has been prepopulated with the JSON copied to the clipboard, which corresponds with the data from the player performance spreadsheet provided in the prompt.

Sample Calling Corticon JavaScript Decision Service

Enter the payload to pass to the decision service:

```
[
  {
    "efficiencyPoints": null,
    "name": "Mustungs",
    "player": [
      {
        "teamName": "Mustungs",
        "game": [
          {
            "date": "2022-04-01",
            "efficiency": "good"
          },
          {
            "date": "2022-04-02",
            "efficiency": "better"
          },
          {
            "date": "2022-04-03",
            "efficiency": "best"
          }
        ],
        "name": "Brown"
      },
      {
        "teamName": "Mustungs",
        "game": [
          {
            "date": "2022-04-01",
            "efficiency": "worst"
          },
          {
            "date": "2022-04-02",
            "efficiency": "better"
          }
        ]
      }
    ]
  }
]
```