
A DYNAMICSCON PRESENTATION
POWERED BY DUG
DYNAMICSCON VIRTUAL
MARCH 2023
CUSTOMER ENGAGEMENT
DYNAMICSCON.COM

How the Right Fusion Dev Balance Saved the Day

Rami Mounla

Agenda

About Myself

Challenge

Options

Fusion Development Process

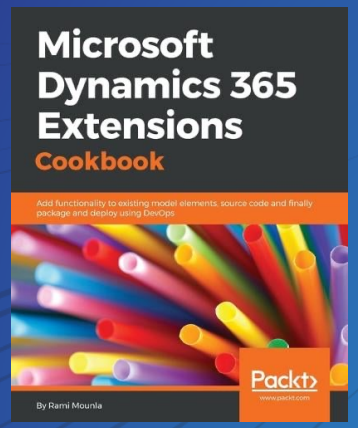
Results

Who am I?

Wellington

Power Platform UG

MVP Microsoft® Most Valuable Professional



DXC TECHNOLOGY

Microsoft

mscrm-addons.com
Your company for MS-CRM ADD-ONS!

purely CRM

POWERED BY DUG
DYNAMICSCON VIRTUAL

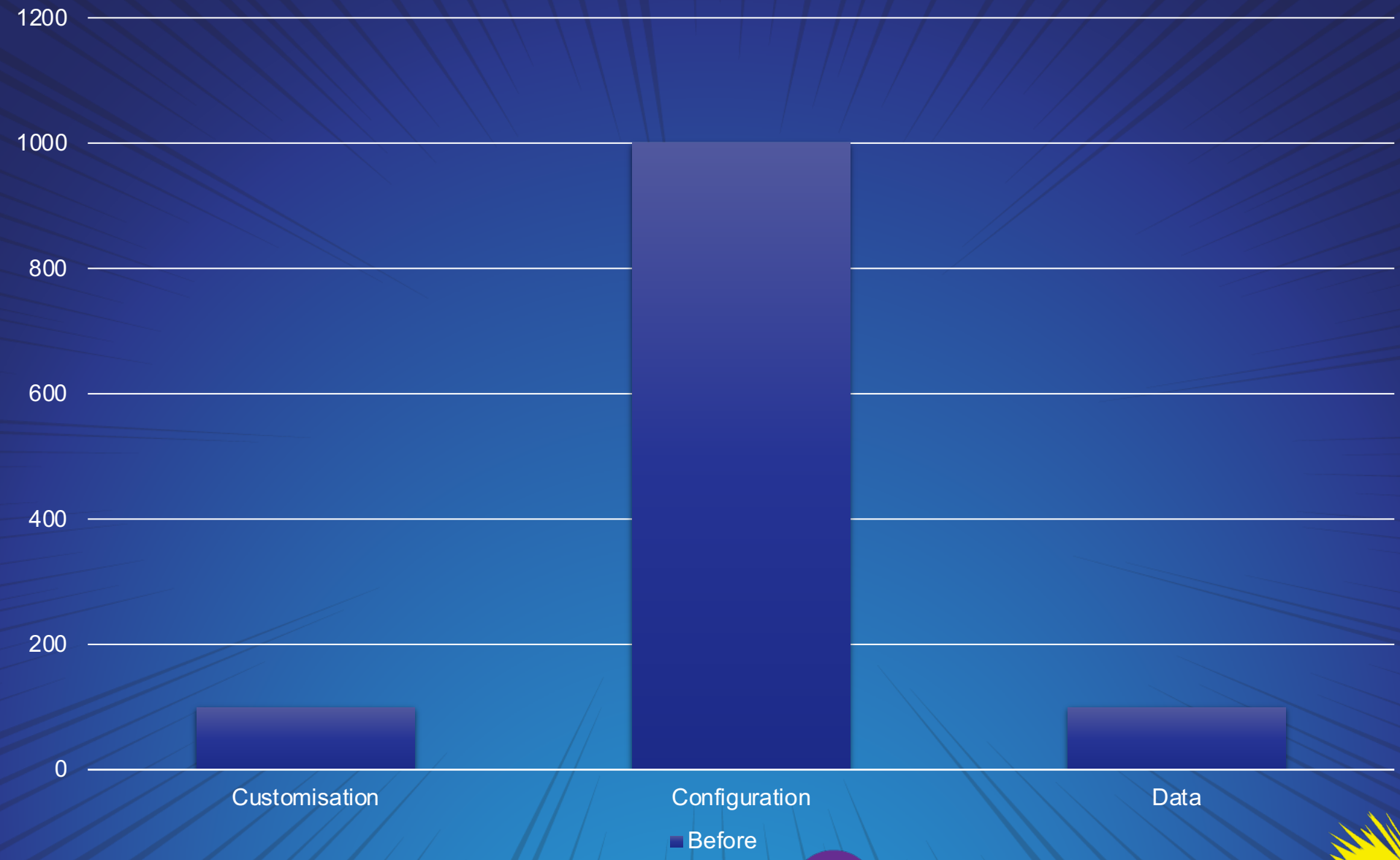
If you want to *succeed*

embrace **fusion** development

What is Fusion Development?

#ProCodeNoCodeUnite

Work Distribution Per Sprint



Challenges

Customer's Challenges

- Rules are **scattered** across the platform (Power Automate Flows, Plugins, and Custom Actions).
- Rules are **hard coded** and require coding changes and release.
- Finding the rules to change requires investigation and **understanding of the code base**.
- High-complexity rules in Power Automate are difficult to manage.
- Limited capability for a power user to **maintain** the rules.
- **Numerous rules tend to get complex.**
- Rules are **difficult to test**.

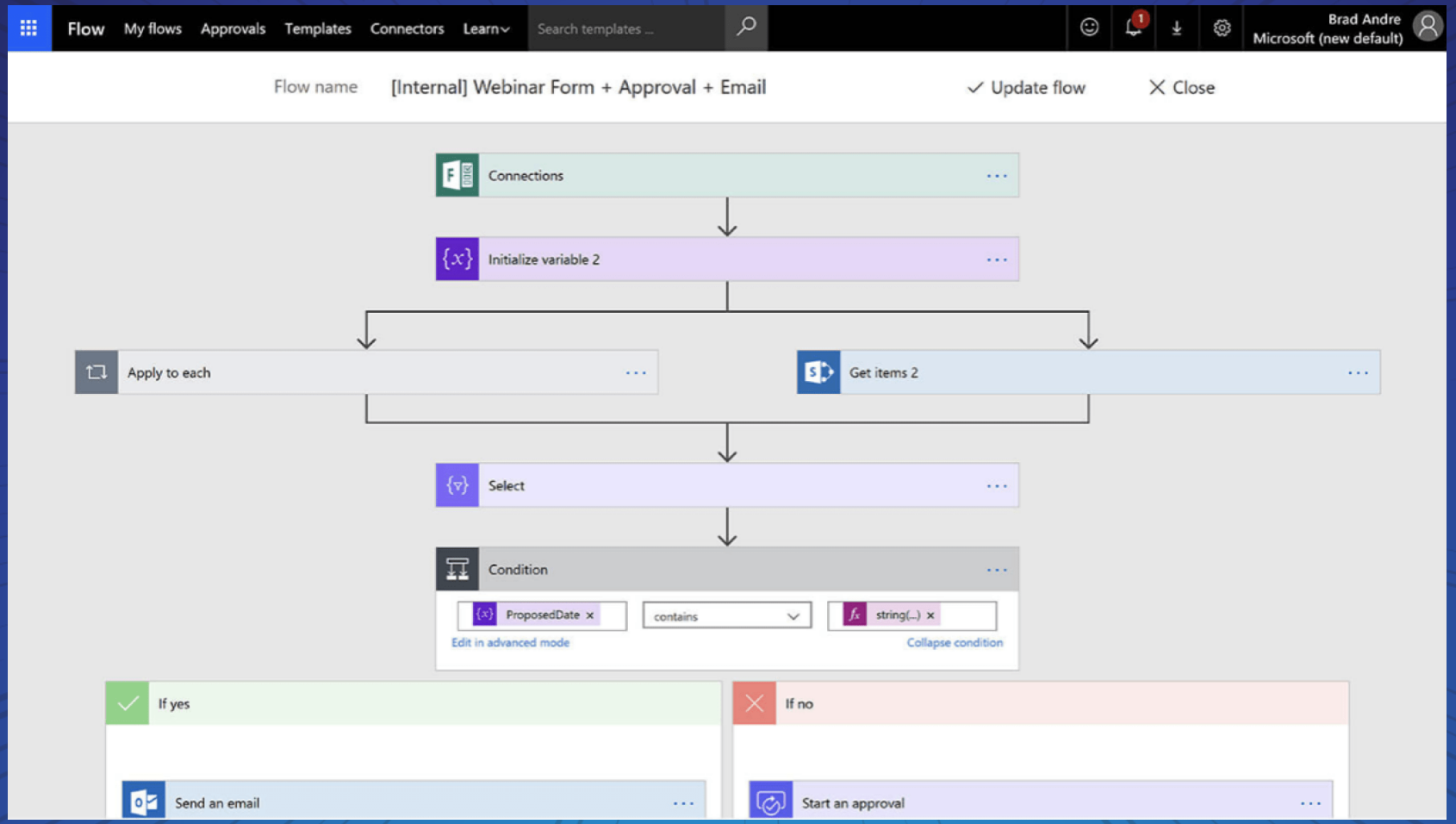
High Level Requirements

- **A centralized location to manage the rules.**
- **Rules are easy to maintain by a power user – even with 100s of rules and 100s of criteria.**
- **Rules can easily be deployments across SDLC environments.**

High Level Non-functional Requirements

- **Preference to leverage a SaaS rules' engine.**
- **SaaS solution needs to be located near-shore.**
- **Engine should not contain sensitive information at rest (stateless is preferred).**
- **Engine needs to be performant (to quantify).**
- **SaaS solution needs to meet a level of availability i.e. DR (to quantify).**
- **Rules are stored securely.**

Challenges – Complex Power Automate



Microsoft

mscrm-addons.com
Your company for MS-CRM ADD-ONS!

purely CRM

POWERED BY DUG
DYNAMICSCON VIRTUAL

Power Automate Executions

Edit flow | Get .csv file

Jun 29, 09:14 PM (2 wk ago)	00:00:32	Succeeded
Jun 29, 04:57 PM (2 wk ago)	00:00:01	Succeeded
Jun 29, 04:30 PM (2 wk ago)	00:00:01	Succeeded
Jun 29, 04:23 PM (2 wk ago)	00:00:01	Succeeded
Jun 29, 03:10 PM (2 wk ago)	00:00:01	Succeeded
Jun 29, 02:43 PM (2 wk ago)	00:00:01	Succeeded
Jun 29, 02:23 PM (2 wk ago)	00:00:01	Succeeded
Jun 29, 02:05 PM (2 wk ago)	00:00:01	Succeeded
Jun 29, 02:02 PM (2 wk ago)	00:00:01	Succeeded
Jun 29, 02:00 PM (2 wk ago)	00:00:01	Succeeded

Power Automate Executions In Memory

Flows > Update Macron Test

28-day run history (i) Edit columns All runs

Start	Duration	Status
Nov 2, 10:57 PM ()	23 ms	Succeeded
Nov 2, 10:57 PM ()	34 ms	Succeeded
Nov 2, 10:57 PM ()	23 ms	Succeeded
Nov 2, 10:57 PM ()	11 ms	Succeeded
Nov 2, 10:56 PM ()	39 ms	Succeeded

Options

Competition

- ✅ Low Code
- ❌ Difficult to maintain
- ❌ Not a rules engine

- ✅ Mature
- ❌ Costly
- ❌ Generates Code

- ✅ OOTB D365 Connector
- ❌ ”Connects” to Dataverse
- ❌ Slow

- ✅ Leader in Financial
- ❌ Hosted Outside
- ❌ Complex Pattern

How to Build a Framework?

1. Evaluate feasibility – What are we gaining? Saving?
2. Acknowledge limitations and roadmap (nice to have)
3. Ensure the customer is on board
4. Ensure the team is on board (FUSION)
5. Support it with testing – including stress test
6. Come up with guiding principles

Evaluation Engine Principles (easy to implement)

- Leverage out-of-the-box platform strength and capabilities
- Leverage out-of-the-box schema configuration and security
- The engine is not responsible to calculate/rollup values
- The engine is not responsible for any integrations
- The engine executes within the platform, does not require any external infrastructure

Evaluation Engine Principles (Reusable Generic)

- The engine is an API extension to the base platform (can be called from outside the platform using API extensions)
- No dependencies on any of the existing components
- The engine is loosely coupled with rules and existing schema
- The engine is unaware about the rules and the existing schema during the design/built time

Evaluation Engine Principles (SOLID)

SOLID

- S – Single Responsibility
- O – Closed for Modification, Open for Extension
- L - Liskov substitution principle - design by contract
- I - Interface segregation principle: "Clients should not be forced to depend upon interfaces that they do not use.“
- D - Depend upon abstractions, [not] concretion

Features

Authoring

- Define Rules
- Version
- Effective Dates

Release

- Data Driven / Secured at Rest by Dataverse
- Deployment Data Secured in ADO
- Easy to Transport using DevOps

Capabilities

Payload Generation

- Key/Value Pairs
- 1:N Lists

Execution

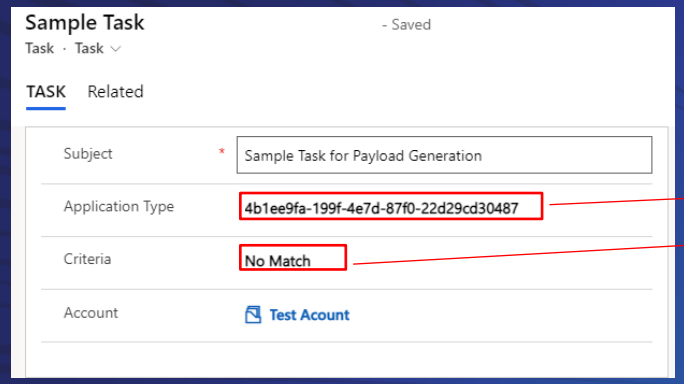
- Feed the rules and the payload to the engine
- Identify the result

Execution Result

- Update records
- Execute Action
- Call a Power Automate
- Create records (e.g. tasks)

Implementation Vision

Execution Sequence



1

Automation Trigger
&
Changes Record State

Trigger Evaluation

2

Payload Generator

{
ApplicationType: 4b1ee9fa,
Criteria: No Match
}

3

Get Rules

{
Rule1,
Rule2
}

4

Evaluate Rules

Result

5

Update Record Attribute

Call Power Automate

6

Execute Custom Action

Microsoft

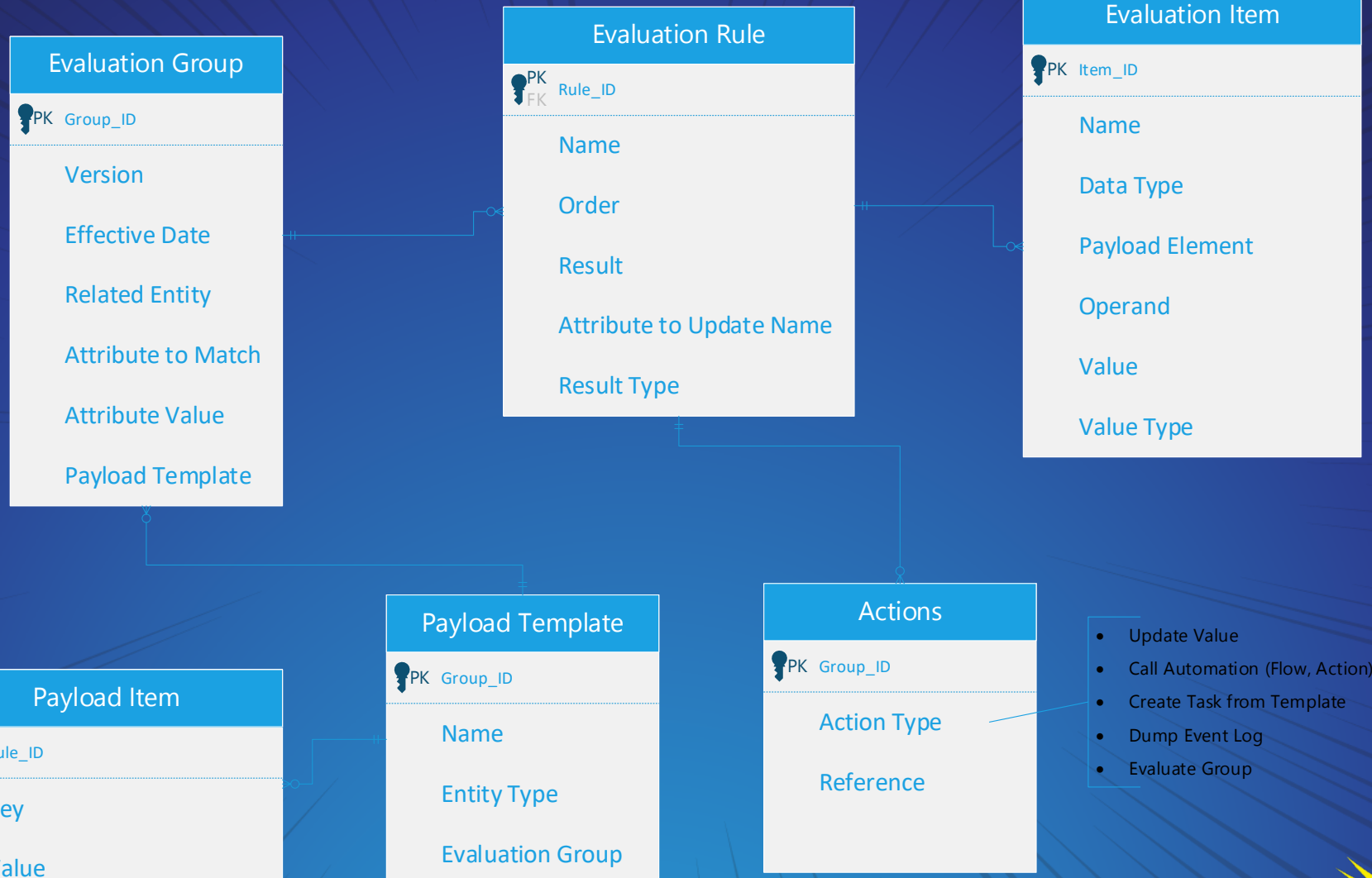
mscrm-addons.com
Your company for MS-CRM ADD-ONS!

purely CRM

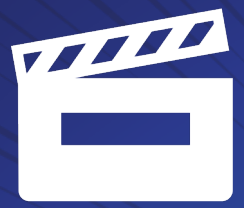
POWERED BY DUG
DYNAMICSCON VIRTUAL

Code Details

ERD

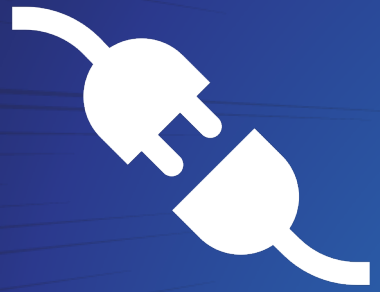


Design



Custom Action

Unit Testing



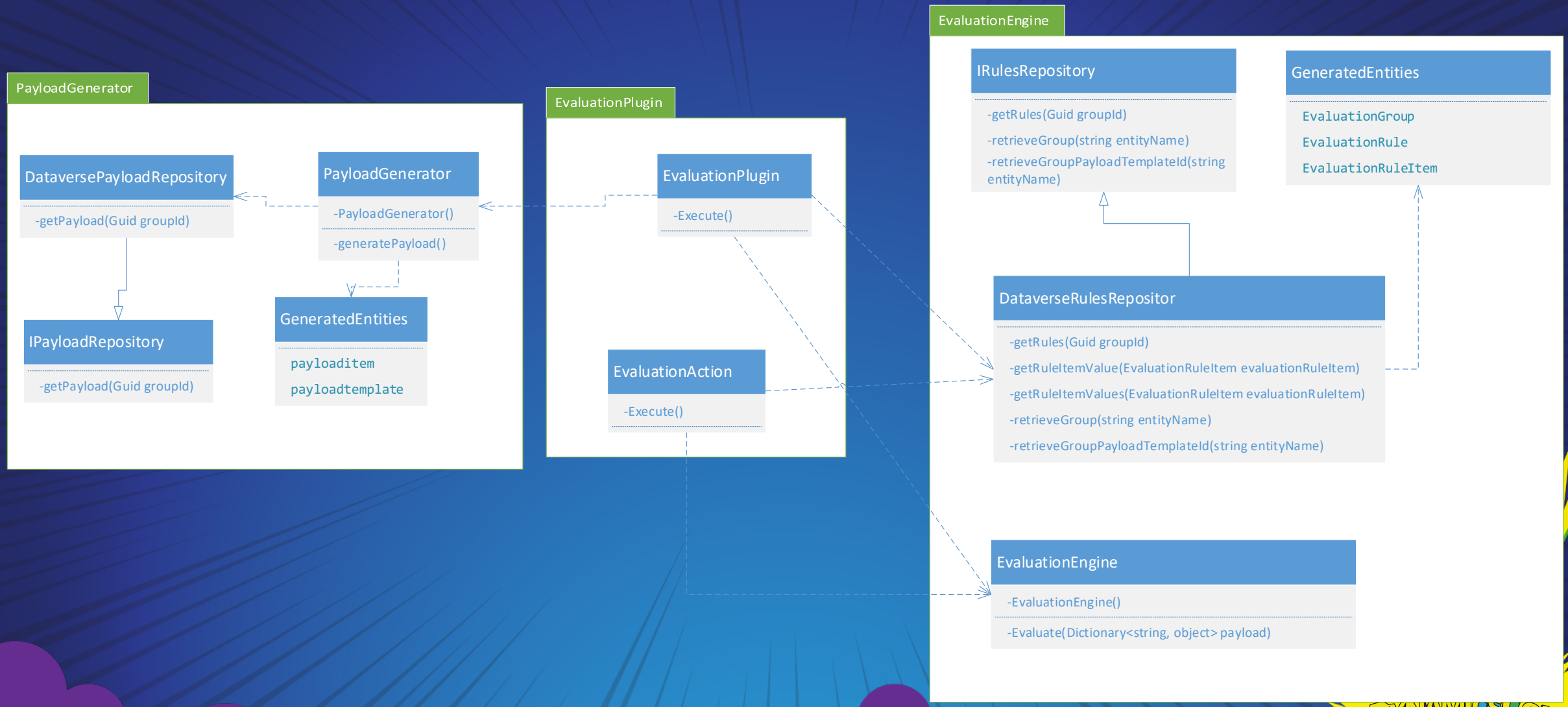
Plug-in

Evaluation Engine
Business Logic

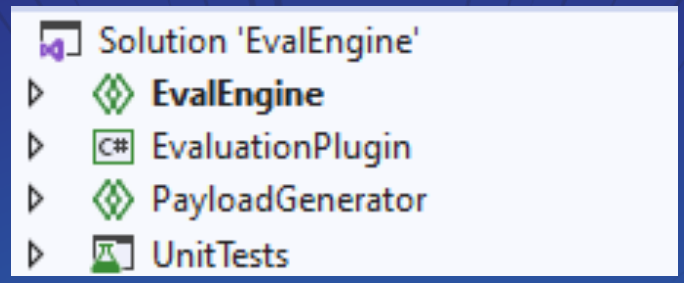
Data Layer

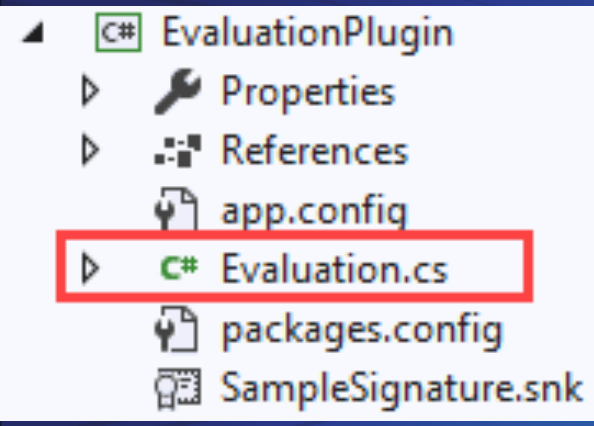
Custom API

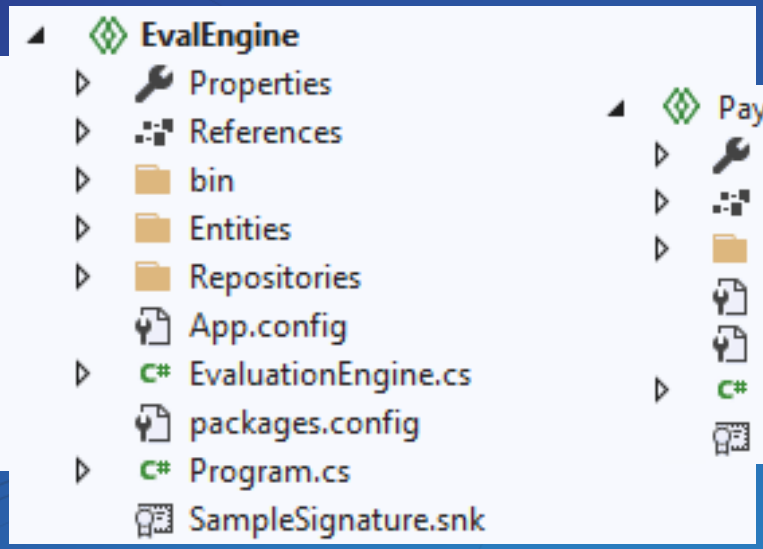
Class Diagram

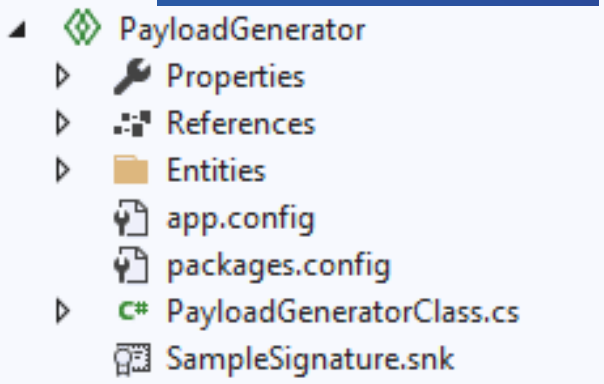


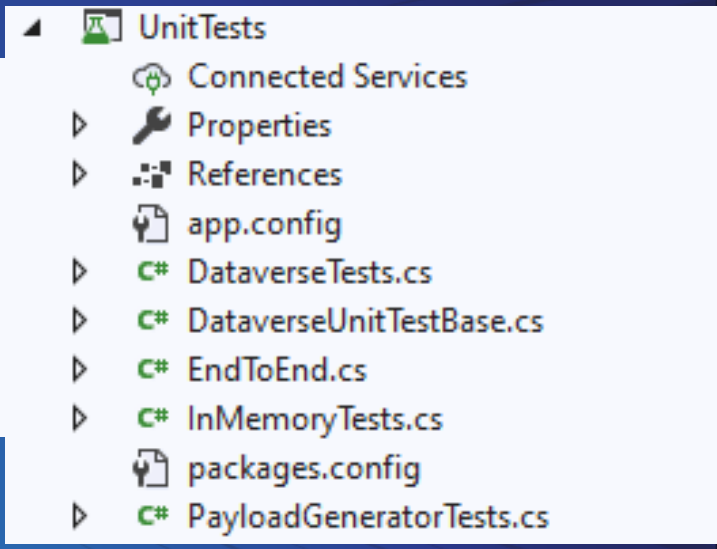
Visual Studio

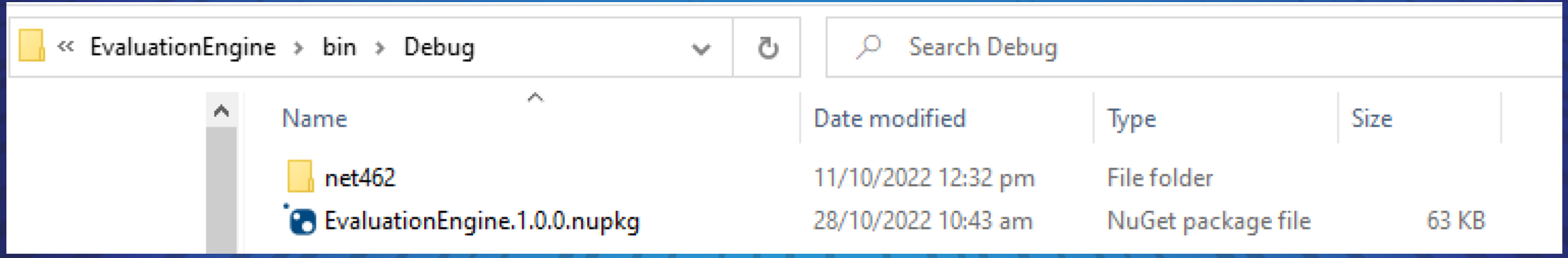






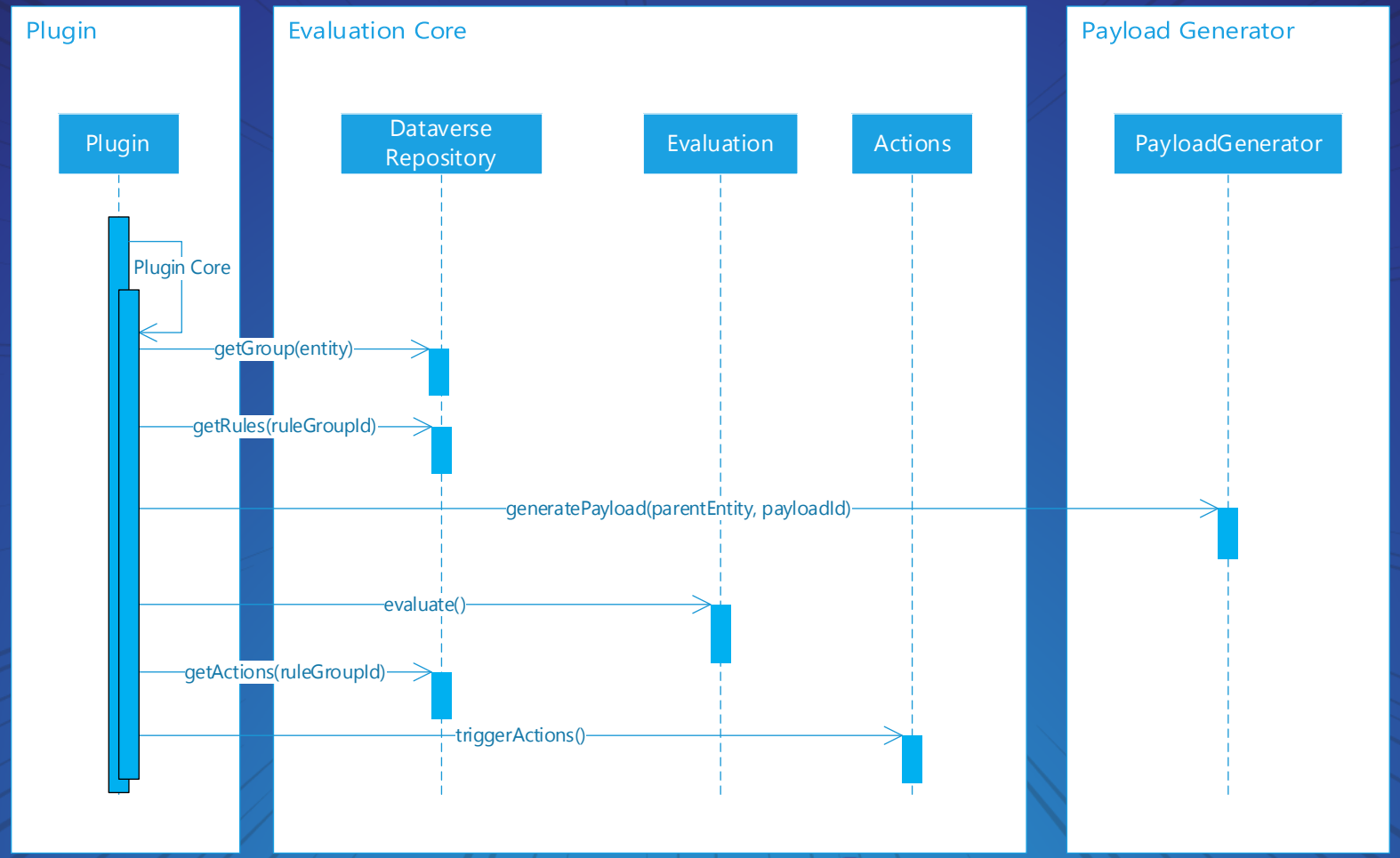






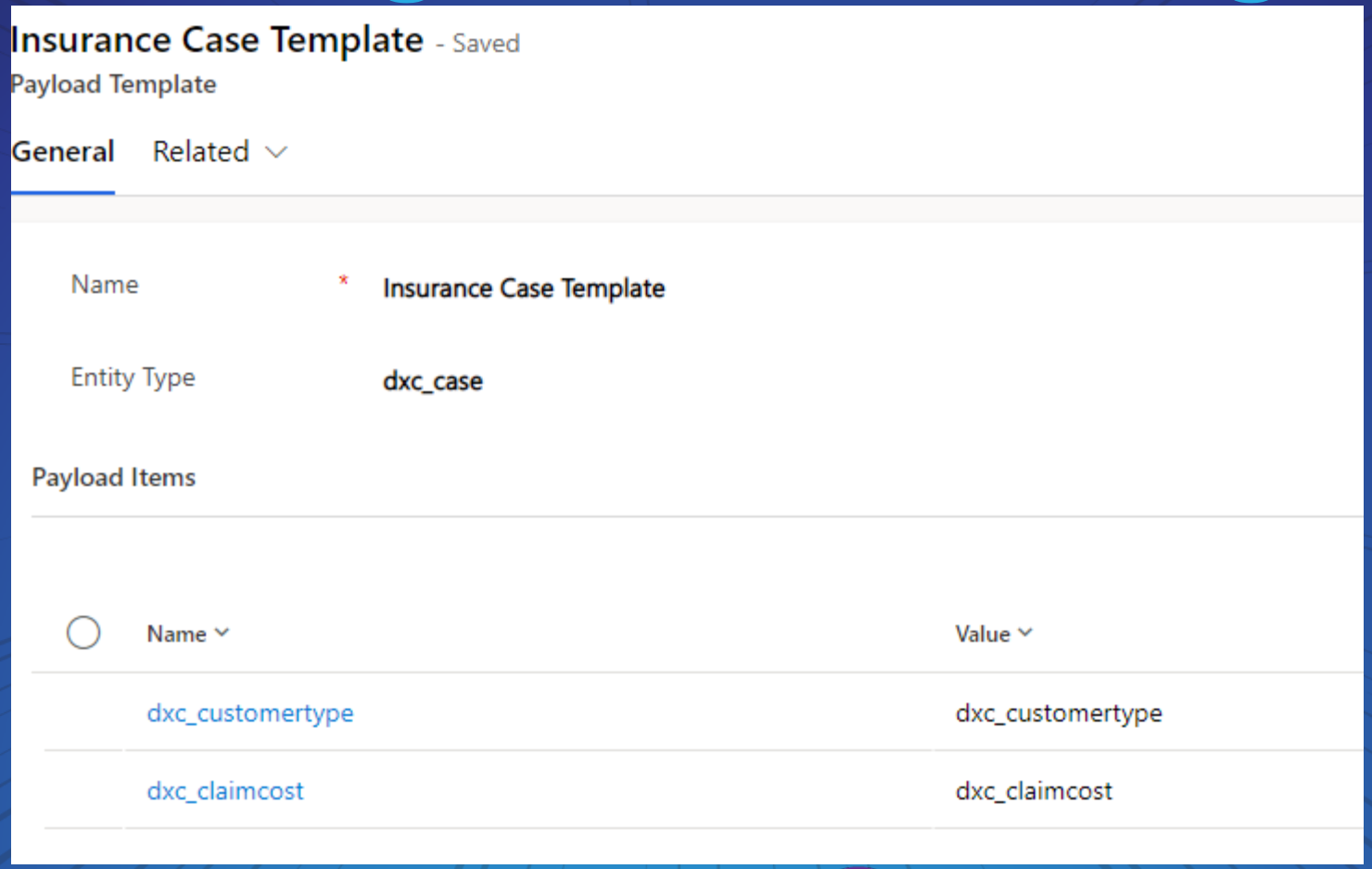
Evaluation Engine Customisation Sequence Diagram

Each call is responsible for a portion of the execution. Most calls are focused on gathering data. The last call updates the appropriate records.

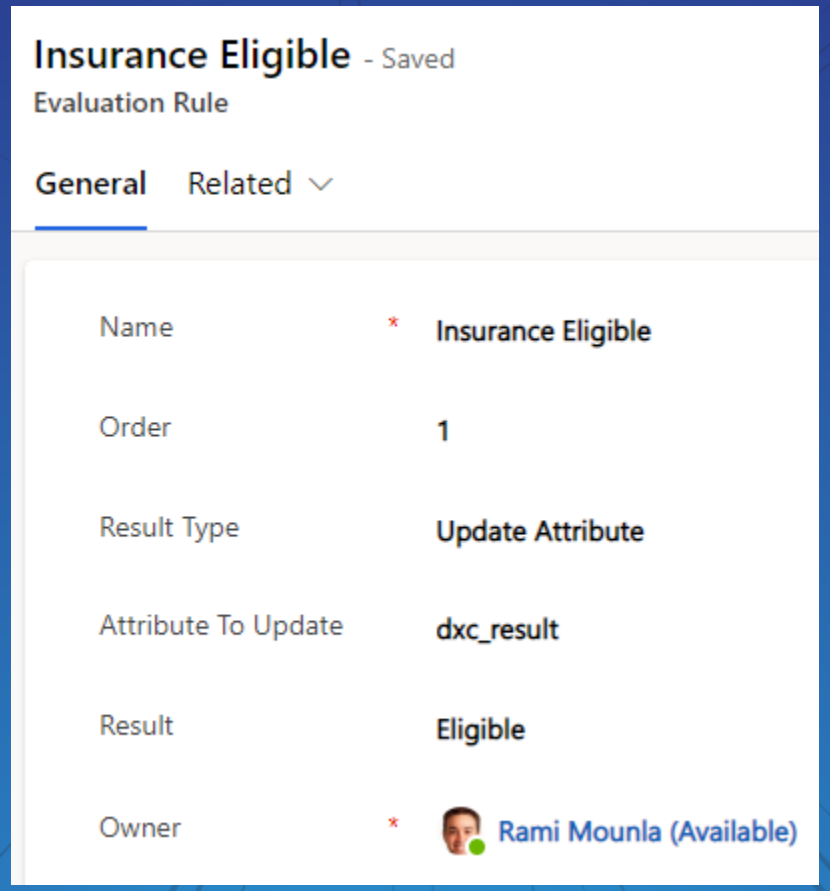


Authoring

Payload Generation Configuration/Authoring



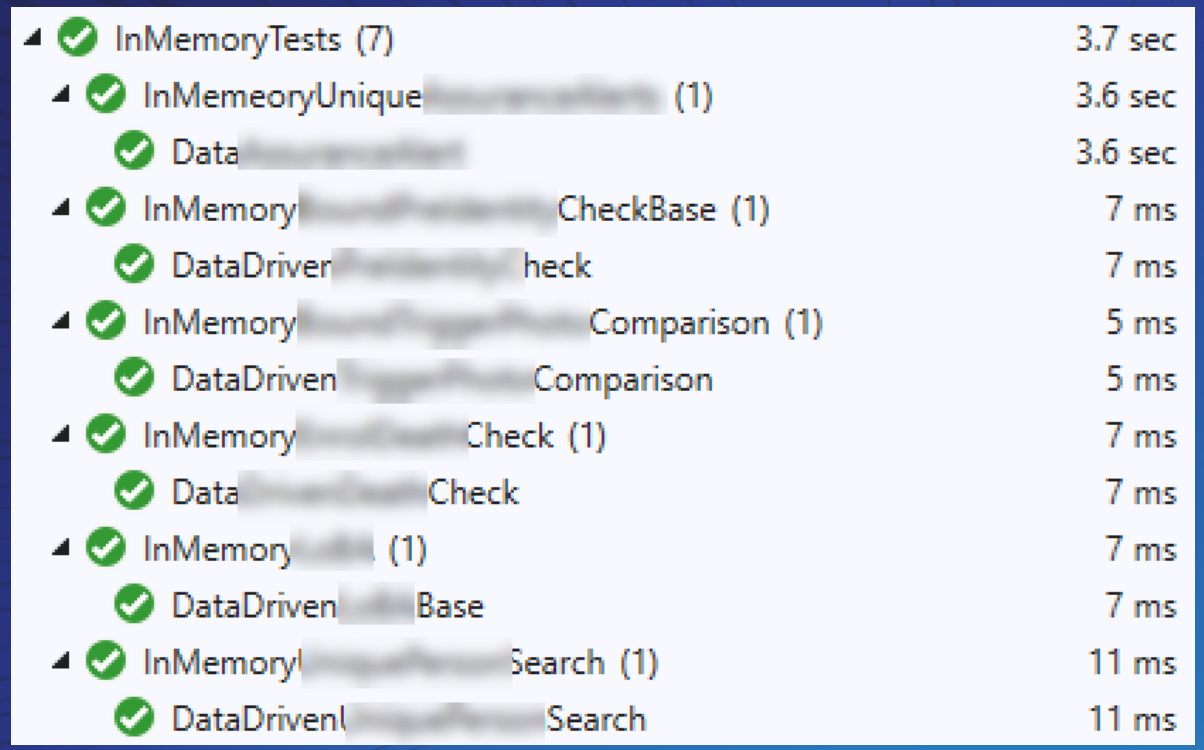
Trigger Configuration



Demo

Rule Authoring & Execution

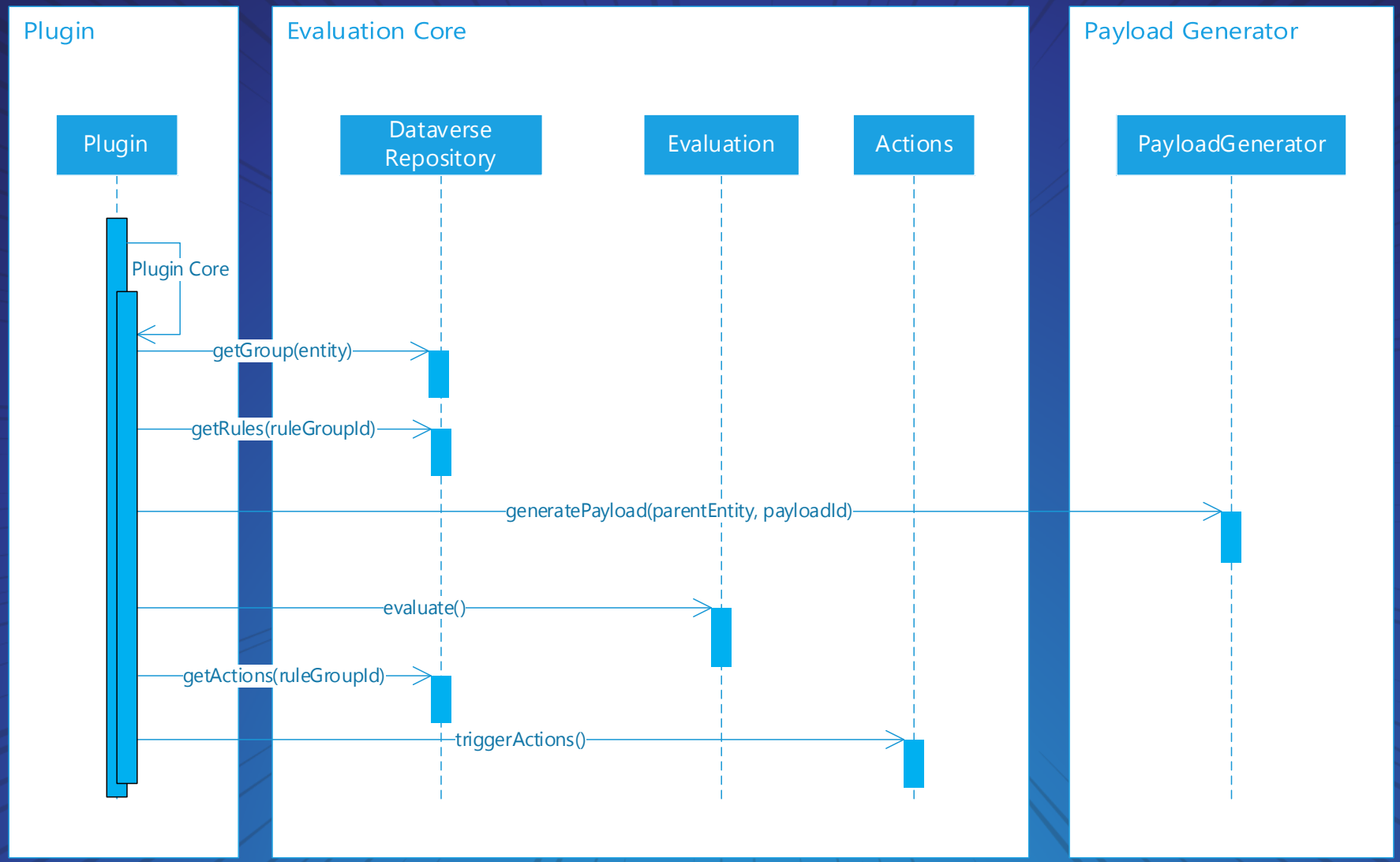
Unit Testing



Testing Speed

Performance Testing

Evaluation Engine Customisation Sequence Diagram

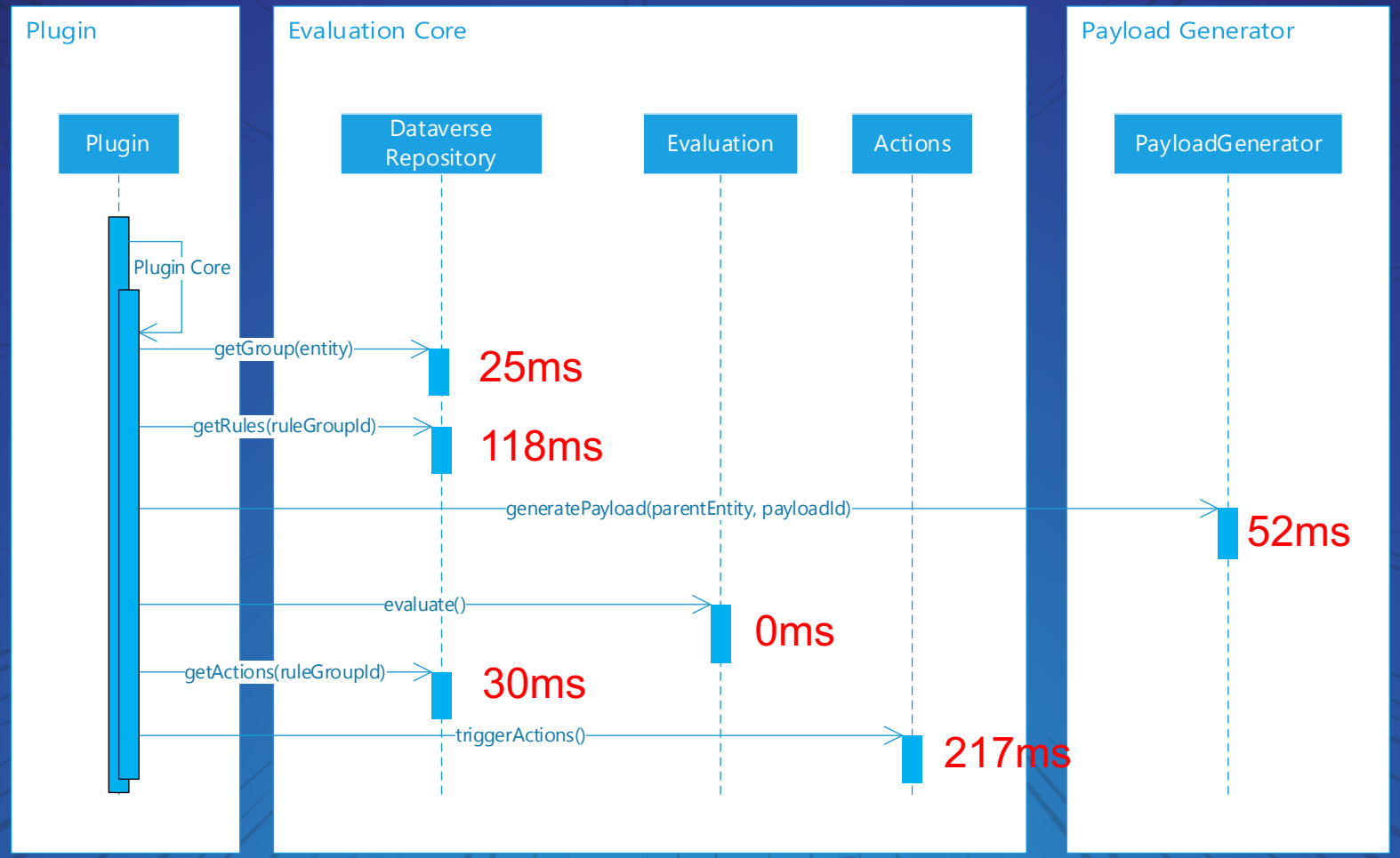


Results

Summary of all results based on 1209 execution.

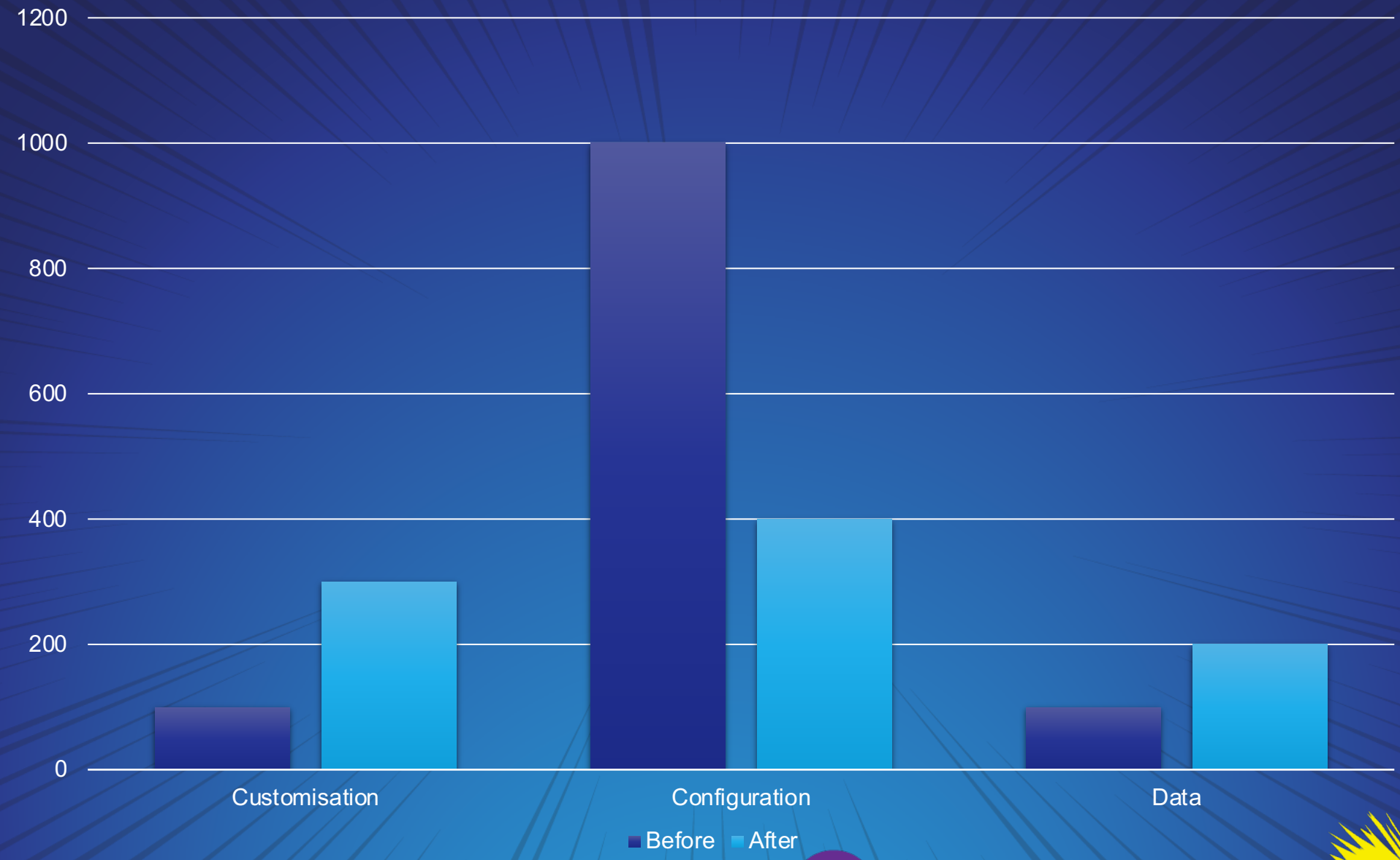
Method	Minimum (ms)	Maximum (ms)	Average (ms)
getGroup	12	513	25
getRules	91	341	118
generatePayload	40	242	52
evaluate	0	1	0
getActions	12	131	30
Execute actions 'Update Task as Completed'	0	0	0
Execute actions 'Update Task as Failed'	0	0	0
Execute actions 'Update Parent Entity'	47	259	65
Execute actions 'Create Exception Task'	136	1086	217
Totals	338	2573	507

Evaluation Engine Customisation Sequence Diagram



Final Results

Work Distribution Per Sprint



Where We Are Today

Complexity implementation completed / reduced

Data driven by functional analysis – no need for further customisation

Faster time to delivery

Easy to build, easy to maintain

Increased test coverage – better regression

Better work distribution

Lower defect rate

Team significantly more engaged

Final Words

Leverage Fusion Development

Utilise the platform's capabilities to their fullest

Get the most out of your team's diversity

Nurture people's passion

Challenge your team to look at better alternatives

Question the status quo

Explore other options that work better for your circumstances


THANK YOU
Questions?