



# DRDM Platform Overview

Digital Rights for Data  
Management

Supported By





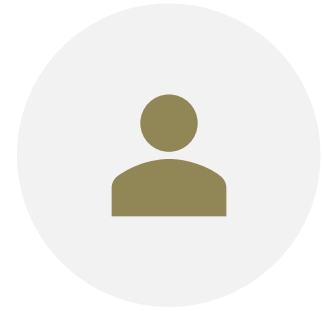
**DIGITAL  
CONTRACTS**



**AUTOMATED  
DECLARATIONS**



**INTELLIGENT  
DATA SALES**



**WORKFLOW  
ENHANCEMENT**

**What is the DRDM proposition?**

# Digital Contract Benefits?



## DIGITAL CONTRACTS

- **Automation & Efficiency** - Eliminates manual paperwork with automated renewals and product and price changes.
- **Compliance & Security** - Immutable records with digital signature and audit trails.
- **Real-Time Updates** - Instant amendments, notifications, and live tracking of contract status.
- **Cost Savings** - Reduces manual administration costs and hard storage costs.
- **Integration** - Seamlessly connects with client onboarding checks with our Workflow and APIs.

# What can automated Declarations achieve?



## AUTOMATED DECLARATIONS



**Efficiency** - Reduces manual effort with automated data collection.



**Accuracy** - Pulls usage data from entitlement systems, minimising errors.



**Compliance** - Ensures timely, verified declarations for audits. Automated reminders



**Cost Savings** - Lowers administrative overhead and compliance costs.



**Integration** - Connects with LSEG DACS, Bloomberg EMRS, and in-house entitlements.

# How can DRDM help me drive Sales?



## INTELLIGENT DATA SALES



**Targeted Sales** - Identifies over-utilised contracts for upselling.



**Revenue Growth** - Maximise earnings by tracking redistribution.



**Market Insights** - Analyses client usage trends with AI for product targeting.



**Automation** - Streamlines offers and pricing updates.



**Integration** - Works with CRM, entitlement, and reporting systems.



**Supermarket** - Publish your products and pricing to clients instantaneously.

# What is the DRDM proposition?



WORKFLOW  
EFFICIENCIES



**Automation** - Reduces manual tasks with streamlined processes.



**Accuracy** - Ensures consistent, error-free contract management.



**Time Savings** - Speeds up renewals, approvals, and audits.



**Resource Optimisation** - Frees staff for higher-value tasks.



**Integration** - Connects with APIs, KYC, and compliance systems.

**startTrigger**

Enabled triggers

At 12:00 AM (UTC), every day

Edit triggers

Test JSON Parameters

```
{"webhookParam1": "va1ue"}
```

**query1**

Retool AI Edit

Action

Classify text

Examples: [Customer support](#), [sentiment analysis](#)

Input \*

Find all CONTRACTS EXPIRING in next 3 months

Classification labels

Type labels and press 'Enter' to add

Model

gpt-4o-mini

Inputs Data JSON Settings

Action type

Get contents of a file

File input

Data\_Notification/mnemonic\_aid.csv

More files exist. Filter by name to see them.

Inputs Data JSON Settings

```
1 import base64
2 import pandas as pd
3 from io import StringIO
4
5 filestrings = {
6     "pdp_prices_file": pdp_prices_file_data_base64data,
7     "m_prices_file": m_prices_file_data_base64data,
8     "aid_prices_file": aid_prices_file_data_base64data,
9     "mnemonic_aid_file": mnemonic_aid_file_data_base64data
10 }
11
12 dataframes = {}
13
14 for key, encoded_str in filestrings.items():
15     decoded_str = base64.b64decode(encoded_str).decode("utf-8")
16     string_data = StringIO(decoded_str)
17     df = pd.read_csv(string_data, sep=",")
18     dataframes[key] = df
19
20 return dataframes
```

```
1 import pandas as pd
2 import logging
3 import json
4 import base64
5 import io
6 import openpyxl
7 from io import StringIO
8
9 # Convert a CSV or Excel formatted
10 # string to a DataFrame
11
12 # Detect if the file is an Excel file
13 is_excel = False
14
15 # Attempt to open the bytes as
16 # a pandas DataFrame
17 is_excel = True
18
19 # Attempt to open the bytes as
20 # a pandas DataFrame
21 pass
22
23 # Detect if the file is an Excel file
24 # Read the Excel file into a DataFrame
25 df = pd.read_excel(io.BytesIO(csv_bytes))
26
27 # Print the detected CSV file format
28
29 # Attempt to decode the bytes
30 csv_string = file_bytes.decode("utf-8")
31
32 # Attempt to decode using
33 # for enc in common_encodings:
34     try:
35         csv_string = file_bytes.decode(enc)
36         print("Failed to decode using", enc)
37     except UnicodeDecodeError:
38         pass
39
40 # List of common encodings
41 common_encodings = ["utf-8", "latin-1", "ascii", "utf-16", "utf-32"]
42
43 # Attempt to decode using
44 # for enc in common_encodings:
45     try:
46         csv_string = file_bytes.decode(enc)
47         print("Failed to decode using", enc)
48     except UnicodeDecodeError:
49         pass
50
51 # Initialize variables
52 header_line_index = None
53
54 # Find the line containing the
55 # header
56 for index, line in enumerate(csv_string.splitlines()):
57     # Check if the line is a header
58     if not(line.strip().startswith("#")):
59         header_line_index = index + 1
60         break
```

Simple or Complex workflows to automate your Exchange processes.

# **BROADHEAD** **TECHNOLOGIES**

Supported By

