decisyon

Decisyon App Composer (DAC)

Data Architecture Overview

March 2019

V 2.1

Table of Contents

1. Introduction

Decisyon App Composer (DAC) is a code-free visual app development platform offering end-to-end built-in functionality for developing and deploying any Industrial IoT application. DAC's intuitive and simple drag & drop development environment allows everyone, from hardcore programmers to business analysts, to rapidly build IoT solutions that consume and analyze real-time data, visualize and analyze that data, support fact-based decision making via a unique collaborative space, and execute on those decisions.

We have also developed and deployed many vertical solutions for the manufacturing, renewable energy, aviation, pharmaceutical, transportation and automotive industries. With built-in microservices such as data management, BI, mashboarding, rules engine, collaboration and execution, Decisyon offers a dramatic increase in speed to outcome for building and modifying vertical solutions

2. DAC Terminology

Before describing DAC's Data strategy, let's review some basic concepts on which the platform is based on. Our data journey starts from the top and slices down towards the foundation of the DAC platform.

The following are the basic concepts and DAC's terminology:

- **Application**: is what the end user sees. It is a collection of mashboards and web pages interacting with each other to support a variety of processes (decisional, analytical …). At this stage, the interaction between the different mashboards is primarily data driven.

- **Mashboard**: is a collection of components/widgets displayed on a browser page. Each object interacts with the other in the same mashboard contributing to its purpose and mission. Each component is responsible for the interaction with a dataset coming from a local and/or remote source.

- **Components or Widgets**: These are the building blocks to create your mashboards. Some examples of basic components are:
    - Crosstabs
    - Charts
    - Time series
    - KPI's

  With DAC, you also get much more complex components such as:
    - Maps
    - Data Streaming display
    - Calendars
    - Social streams
- Custom Widgets

Each Object is data driven and can change its state according to its data and react to the changes. Components and Widgets are not simple data visualization tools but rather they interact with the data they display and even executing actions according to inputs. Each object is strictly connected with a local/external data source.

- **Metadata**: A collection of definitions and configurations that exist for every object in DAC. The governance and the behavior of the entire DAC platform is based on metadata, from the simplest form of the data mapping to the most complex aggregation of objects in mashboards and applications. Metadata is persisted in the form of relational tables in the RDBMS of your choice.

3. Data Mapping

DAC is a platform that orchestrates the data and processes amongst other functionalities. In this context, it is obvious that data mapping is one of the key elements of DAC's data architecture.

The mapping in DAC is done using a wizard that guides you through the definition process, starting from the field discovery on local/remote data source to the definition of the logical relationships between different data structures and even aggregation if needed. The configurations are stored in the metadata.

4. Data Source Definition and Usage

The Data Source layer is used by DAC to connect with the local/remote data structures processing the real time data. DAC can natively connect to numerous different databases and data systems via a variety of protocols including but not limited to RESTful APIs, Web Services using SOAP, Data Source native APIs and even JDBC.

Here are a few examples of supported data sources:

- RDBMS (Oracle, MSSQL Server, PostgreSQL, MySQL, …)
- Big Data (Hadoop Hive, Teradata, …)
- Columnar and In Memory DB (IQ, SAP Hana, Kognitio, …)
- Data Analytics Database (GreenPlum)
- Historian Database
- Company Systems (SAP, Manufacturing systems, Maintenance Systems, …)

The data sources connect our objects to the data and each single object can be connected to a different data source. As a result, DAC platform can readily orchestrate the data coming from different processes and sources into a single organic application.

5. Data Persistence

DAC's Data Architecture is primarily based on the Data Federation concept:

- DAC only persists the metadata about the different data sources and their mapping information in its underlying database and reads the data in real-time and on-demand from the mapped data sources to create all the dashboards, reports, pages and workflow in the application.
- DAC provides the ability to add to or modify the source data and writes the data updates back to the original data source. However, the data generally stays at the source and as a result DAC does not introduce any data synchronization issues in real-time applications.
- DAC also provides a number of configurable options to manage and persist parts or the entire data in a local DAC database. They are as follow:
    - Offers a variety of ETL options in conjunction with a data mapping wizard, to allow the data from different systems, and devices to be moved to the DAC database for persistence, data warehousing, historical analytics, etc.
    - Define a data model to manage aggregated and raw data from multiple sources locally
    - Cache frequently used unchanging, static data locally to avoid roundtrips to the data source and improve performance.
    - Ability to persist data coming from different sources and if needed, push them towards a storage service or similar.

Data sources are used both in **pull** (read) and in **push** (write) that is dependent on the objects they are connected to in the application. A large variety of use cases can benefit from this design:

- Collecting data coming from different systems, and devices and pushing them to an Analytic Service using REST APIs
- Connecting different sources of data in the enterprise and making them work together by orchestrating data coming from each system
- Persisting data coming from different sources, if needed, pushing them towards a storage service or similar.

Pushing data often means writing into a database, but in DAC, this is not necessarily the only case. Using DAC data sources, pushing data means more a form of interaction with the remote systems.

Persistency of a particular set of data is also achievable using some purpose-built objects in the platform. They are:

- DLR: Data Load via Report
- RDT: Report Data Transfer

In this case, DAC is able to persist the data starting from **data mapping (A)** to **data mapping (B)** were **A** and **B** are defined on different data sources. DAC can also interact with external ETL tools such as Talend for managing the job executions:

- On schedules
- On event
- On demand

6. Data Connectors

Data connector is a connection definition that is configured inside the DAC platform. The major advantage is the ability to completely decouple the data layer from the presentation and analysis layers in the application.

A data connector is essentially an abstraction of the physical data source we use underneath to access the data needed to perform calculations or to be displayed on a KPIs. This ensures the reusability of these components throughout DAC and the Application.

An example, is the ability to display the same data in 2 different formats (a table and a chart). In this case, there is only one data connector but the widgets using it are different and segregated, each one with his own filters and rules.

We support the following set of different native data sources families in the platform:

- Oracle
- SQL Server
- MySQL
- PostgreSQL
- DB2
- Apache Hive /HDFS
- Teradata Database
- Sybase
- Sybase IQ
- GreenPlum
- Sap Hana
- Kognitio
- OSIsoft PI Server
- Proficy Historian
- REST Connector

With the high level of abstraction within the DAC platform, adding a new data source family is easy and it usually involves the addition of a new driver to the platform. Once the appropriate driver is imported, the data connector can wrap it and different query languages can be used.

7. About Decisyon

Decisyon enables business users to rapidly build solutions using a code-free visual software development environment. Our products accelerate your data journey from aggregation to visualization, insight, analysis and decision thru action. Decisyon offers the leading code-free visual software development environment for mission critical enterprise operations requiring real-time awareness and adaptability to their business operations and processes. We have developed and deployed many vertical solutions for the manufacturing, renewable energy, aviation, pharmaceutical, financial services, transportation and automotive industries. With built-in microservices such as data management, BI, mashboarding, rules engine, collaboration and execution, Decisyon offers a dramatic increase in speed to outcome for building and modifying vertical solutions. Decisyon's products and software solutions, ideally suited for IIoT applications running on any PaaS, are used in over 200 companies globally. Decisyon is headquartered in San Francisco, California.

Corporate HQ

795 Folsom Street, 1st Floor
San Francisco, CA 94107

www.decisyon.com
sales@decisyon.com