OpenI40™ platform architecture guide 1.0 (Beta preview)

Summary

Introduction

The OpenI40™ advanced production scheduler platform supports various deployment options and has multiple modules varying from an embedded use to various standalone microservices architectures deployment scenarios.

As software architect or sysadmin responsible for its installation or embedded use or integration, there are various scenarios you can choose or integration paths you can use.

All its architecture is thought to replace its components with custom made ones having full control of what the software does.

List of the executable OpenI40™ modules

Module name	Module description
OpenI40™ platform standalone monolithic application	Monolithic application that connects to the platform database and let user interact with scheduler Web GUI with full user functions, let user run scheduling algorithms with full settings and save scheduling results in the database.
OpenI40™ standalone advanced production scheduler	Let user interact with scheduler Web GUI with full user functions, let user run scheduling algorithms with full settngs. Can be configured to load and save data using remote REST services data source(s) or structured JSON data source.
OpenI40™ REST persistence server.	Publishes REST services to implement remote load/save functionalities for the OpenI40™ standalone advanced production scheduler.
OpenI40™ REST integration server.	Publishes REST services to READ/WRITE OpenI40 ™ database tables remotely. Is thought to be used by 3rd party integration software.
OpenI40™ direct database to database integration server	Is a fully configurable/customizable module that implements direct 3rd party database integration with an advanced timestamped synchronization algorithm.

How to launch the executable artifacts of the architecture

1st install java 9+ runtime

All the OpenI40™ components are delivered in java maven artifacts (.jar files) so you can launch them installing a java runtime platform version 9 or more on your system downloading it from: https://www.oracle.com/java/technologies/downloads/ or https://openjdk.java.net/projects/jdk/ .

Basic OpenI40™ executable artifact launch command

Once Java 9+ is installed you can run each OpenI40™ executable artifact with:

Java -jar <component artifact file name>

Other useful options:

Run the application with specific maximum available memory

Java -Xmx2048m -jar <component artifact file name>

Run the application with custom configuration .yml or .properties file

Useful to configure differently ports, addresses and all internal components configuration(s):

Java -jar <component artifact file name> --spring.config.location=file:<path of your config file>

Or

Java -Dspring.config.location=file:<path of your config file> -jar <component artifact file name>

(read https://www.baeldung.com/spring-properties-file-outside-jar)

Scheduler datasources concept

The scheduler can hold in memory and schedule multiple datasets simultaneously each “dataset” is identified with a triple key(s) identifier :

dataSourceName: the name of the datasource

dataSetName: the name of the dataset

dataSetVariant: the name of the variant of the dataset

description: Description of the datasource displayed on the software GUI

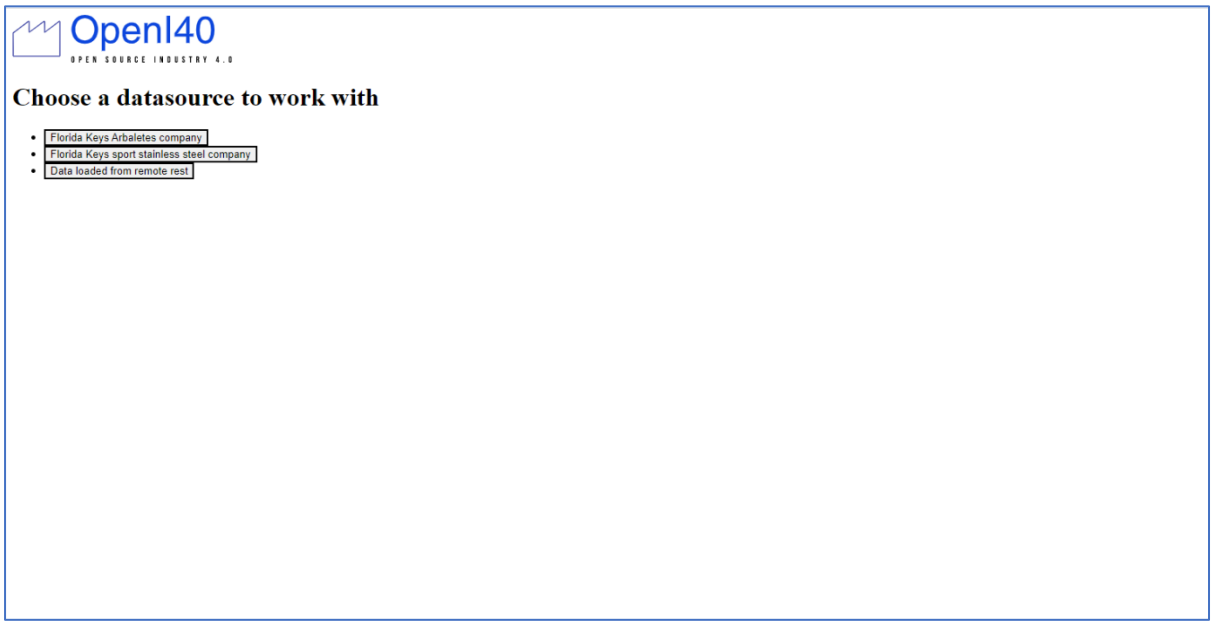


Figure 1 APS Gui Home page, choosing wich data source to work with

You’ll find this keys in multiple configuration options in OpenI40™ modules.

OpenI40™ platform standalone monolithic application

Maven artifact: groupId**:** com.openi40.platform artifactId**:** openi40.app

Executable jar name: com.openi40.platform.app.bootable.jar

Default url for client connections: http://<your server address>:8080/openi40/index.html

Swagger page: http://<your server address>:8080/openi40/swagger-ui.html

This application packages all the OpenI40™ advanced production scheduler functions with also full web visual interface features with the ability to load and save data in a configured remote database.

This database can be chosen from almost every vendor(s) product and has to be create following next chapter(s) on the **OpenI40™ database** section instructions.

In principle with this application, you can have a fully functional APS scheduler with ability to load/save on its own database with a single point of installation in your network.

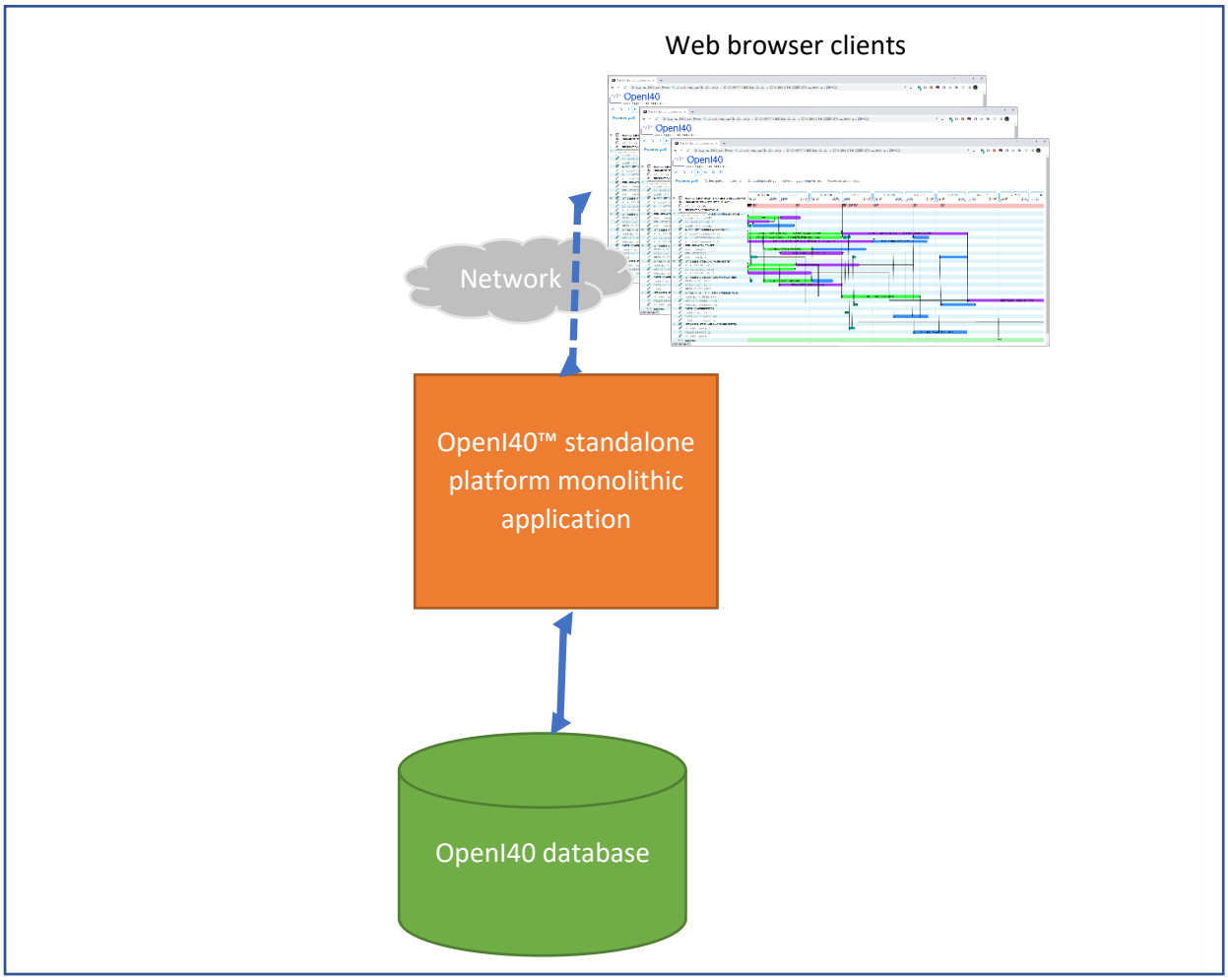


Figure 2Simple monolithic application configuration

You have to modify or create the application.yml module configuration including both the **OpenI40™ database** and **OpenI40™ standalone advanced production scheduler** components (read them in the following chapters).

```
hibernate.cache.use_second_level_cache: false

spring.datasource.driver-class-name: <eventually indicate the jdbc driver
class>
spring.datasource.hikari.connectionTimeout: <connection timeout time>
spring.datasource.hikari.maximumPoolSize: <pool size>
spring.datasource.username: <database connection username>
spring.datasource.password: <database password>
spring.datasource.url:  <jdbc connection string>

# put only 1 node of the following:
com.openi40.platform.dbchannels:
        useJpaStreaming: <Boolean, if true database is read with
        streaming architecture/jpa api>
        batchingSize: <batching of loaded records>
        configs:
          -
           dataSourceName: <data source name code>
           dataSetName: <data set name code>
           dataSetVariant: <data set variant code>
           dataSourceDescription: <data set description to display>
```

OpenI40™ platform microarchitecture components

It follows the list of separate services you can combine to build your specific platform configuration.

OpenI40™ database

The technical design of the OpenI40™ platform and the way it's developed is fully cross database platform compatible with almost all vendors product.

How to setup the database

You can find the scripts for the database tables creation on the **create.sql** DDL script, **the insert.sql** to insert some example data, and **alter.sql** for the foreign keys creation.

These scripts are thought to run with latest PostgreSQL versions but can adapted to run on almost every database platform.

How to configure database connectivity

On each OpenI40™ components that have a direct connection with the database these are the properties to configure:

```
hibernate.cache.use_second_level_cache: false

spring.datasource.driver-class-name: <eventually indicate the jdbc driver
class>
spring.datasource.hikari.connectionTimeout: 20000
spring.datasource.hikari.maximumPoolSize: 5
spring.datasource.username: <database connection username>
spring.datasource.password: <database password>
spring.datasource.url:  <jdbc connection string>
```

OpenI40™ standalone advanced production scheduler

Maven artifact: groupId**:** com.openi40.scheduler artifactId**:** Engine.Standalone

Executable jar name: com.openi40.scheduler.engine.bootable.jar

Default url for client connections: http://<your server address>:8080/openi40/index.html

Swagger page: http://<your server address>:8080/openi40/swagger-ui.html

This standalone application has the following functionality:

- publishes the full Web user interface
- full advanced production scheduler functions and MRP functions
- can work with multiple configurable datasets and let the user schedule/interact with them
- can load/save datasets from remote instances of the OpenI40™ REST persistence server(s) or from 3rd party REST web services.
- Can be configured to load/save data set from JSON files.

If this application is launched without overriding its application.yml settings it will show two example datasets to test the application.

How to configure

The topic of doing configuration(s) on a scenario with the use of this component is quite advanced and technical, you can find all the information on how to do it on **Appendix A**

OpenI40™ REST persistence server

Maven artifact: groupId**:** com.openi40.platform artifactId**:** openi40.persistence.server

Executable jar name: com.openi40.persistence.server.app.bootable.jar

Swagger page: http://<your server address>:8081/openi40-restserver/swagger-ui.html

This standalone application implements REST services in the format that can be used from the OpenI40™ standalone advanced production scheduler to load/save remotely scheduling data.

This application can be run in multiple instances with load balancer in high availability configurations.

Multiple database+OpenI40 REST persistence server couples configured as multiple data source can be used to publish and schedule multiple data source in the same standalone APS engine.

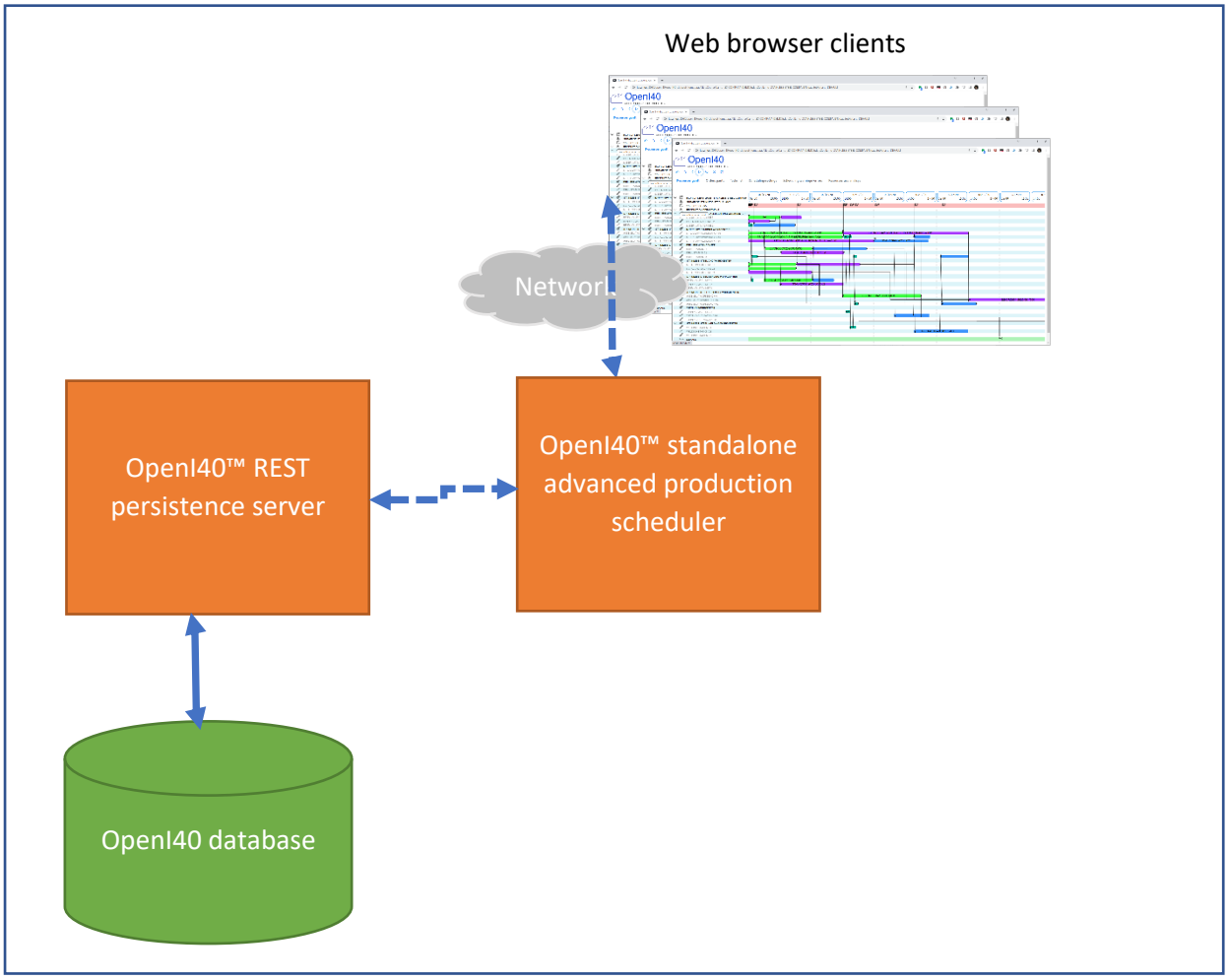


Figure 3Typical REST persistence server + standalone scheduling engine configuration

How to configure

This microservice is configurable in the application.yml (or application.properties) to connect to the OpenI40™ database, let's see the "**How to setup database**" section in the **OpenI40™ database** chapter.

If you search information on how to configure **OpenI40™ standalone advanced production scheduler** connectivity versus the **OpenI40™ rest persistence server** (Figure 2 configuration) let's have a look on **Appendix A**

Example:

```
hibernate.cache.use_second_level_cache: false

spring.datasource.driver-class-name: <eventually indicate the jdbc driver
class>
spring.datasource.hikari.connectionTimeout: <connection timeout time>
spring.datasource.hikari.maximumPoolSize: <pool size>
spring.datasource.username: <database connection username>
spring.datasource.password: <database password>
spring.datasource.url:  <jdbc connection string>

# put only 1 node of the following:
com.openi40.platform.dbchannels:
        useJpaStreaming: <Boolean, if true database is read with
        streaming architecture/jpa api>
        batchingSize: <batching of loaded records>
        configs:
          -
           dataSourceName: <data source name code>
           dataSetName: <data set name code>
           dataSetVariant: <data set variant code>
           dataSourceDescription: <data set description to display>
```

Using this configuration all the published rest services will use the dataSourceName,dataSetName,dataSetVariant as keys to access data.

OpenI40™ REST integration server

Maven artifact: groupId**:** com.openi40.platform artifactId**:** openi40.dbmodel.integration.app

Executable jar name: com.openi40.dbmodel.integration.app.bootable.jar

Swagger page: http://<your server address>:8082/openi40-integration/swagger-ui.html

This microservice implements REST API services allowing remote 3rd party software to inject/extract data in the OpenI40™ database.

OpenI40™ REST integration server is useful for developers/architects, system integrators, software houses to integrate the OpenI40™ platform with their MES/ERP solution.

More information(s) about this module and developing integrations around it can be found in the document: "**OpenI40™ platform integration guide**".

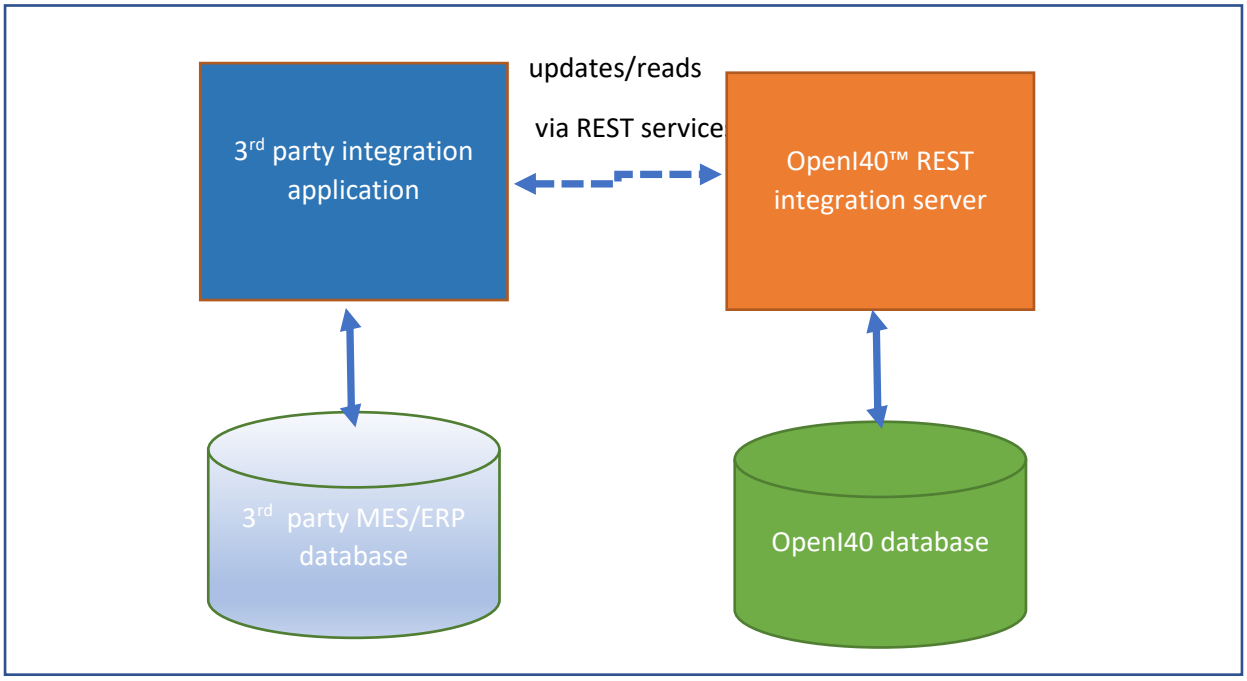


Figure 4 Typical REST integration configuration with REST integration server

How to configure

This microservice is configurable in the application.yml (or application.properties) to connect to the OpenI40™ database, let's see the "**How to setup database**" section in the **OpenI40™ database** chapter.

OpenI40™ direct database to database integration server

Maven artifact: groupId**:** com.openi40.platform artifactId**:** openi40.generical.dbintegration

Executable jar name: com.openi40.generical.dbintegration.app.bootable.jar

Swagger page: http://<your server address>:8084/openi40-dbintegration/swagger-ui.html

This microservice implements a fully configurable and customizable direct 3rd party database to OpenI40™ database that works on both full data synchronization and incremental data synchronization using timestamps

The OpenI40™ direct database to database integration server can be used to obtain a very efficient frequent data synchronization between 3rd party software to the OpenI40™ platform obtaining a "nearly online" integration where the APS work on data sampled few minutes before.

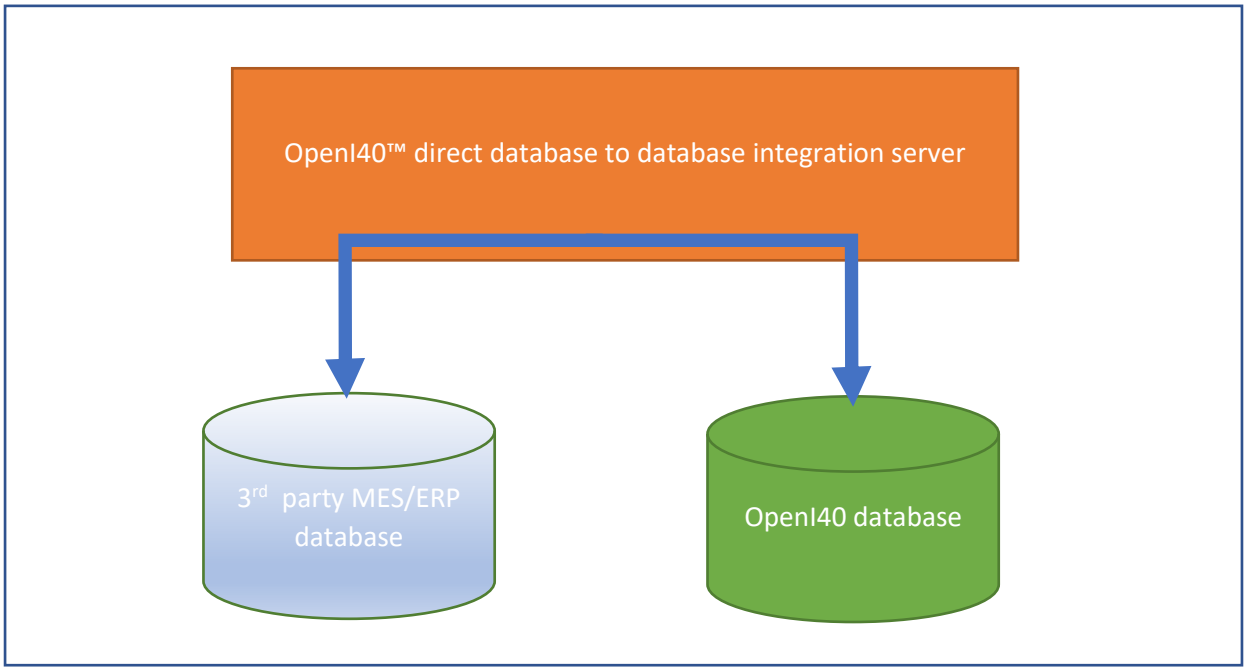


Figure 5 Typical database to database integration using the OpenI40(tm) database to database integration server

How to configure

More information(s) about this module, how to configure it and developing integrations around it can be found in the document: "**OpenI40™ platform integration guide**".

Classical 3rd party MES/ERP integration + OpenI40™ platform monolithic application deployment schemes

This configuration with only 2 components enables to have integration with 3rd party mes/erp system and also the complete APS running with the ability to load/save data in the OpenI40™ database.

Figure 5 shows a configuration to be used if you want to develop the integration of the OpenI40™ platform with your MES/ERP product using REST services.

In this configuration you'll update and extract data from the OpenI40™ database using the rest services provided by the OpenI40™ rest integration server, the OpenI40™ standalone platform monolithic application will read data directly from the database and let users interact with it.

More information(s) about developing/configuring integration with OpenI40™ tools can be found in the document: "**OpenI40™ platform integration guide**".

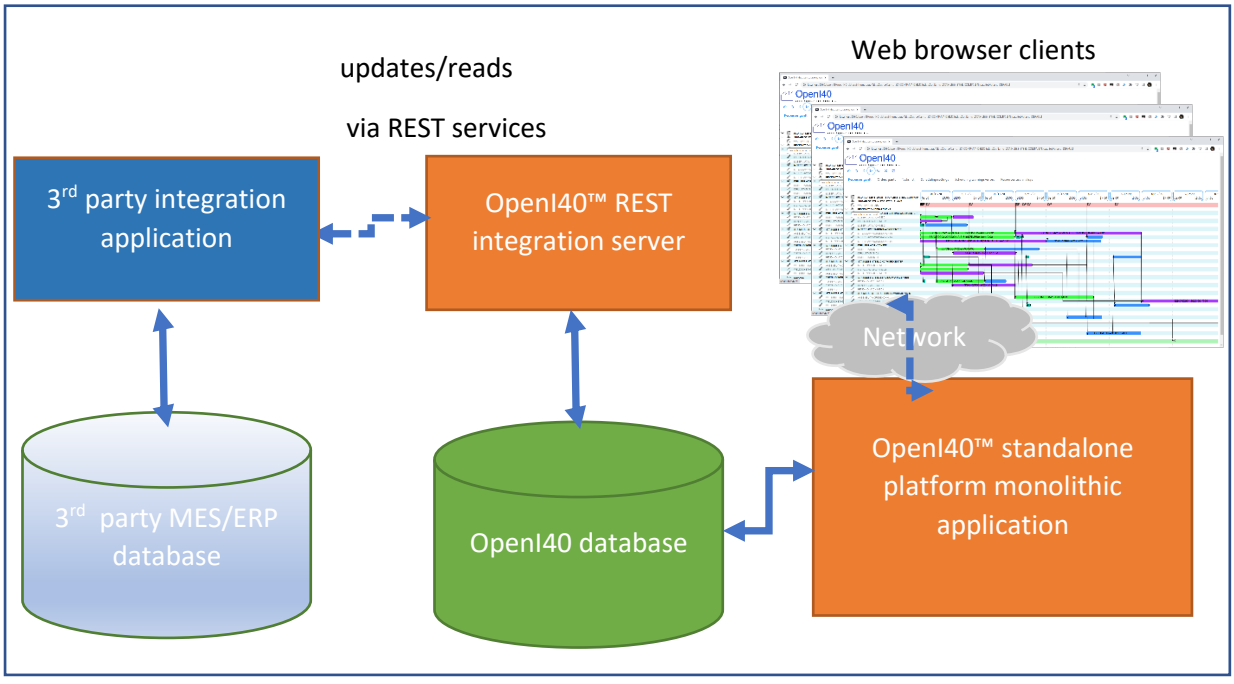


Figure 6 Fully functional and integrated via REST with your ERP/MES solution

Figure 5 shows a configuration to be used if you want to develop the integration of the OpenI40™ platform with your MES/ERP product using REST services.

In this configuration you'll update and extract data from the OpenI40™ database configuring or customizing the OpenI40™ direct database to database integration server, the OpenI40™ standalone platform monolithic application will read data directly from the database and let users interact with it.

More information(s) about developing/configuring integration with OpenI40™ tools can be found in the document: "**OpenI40™ platform integration guide**".

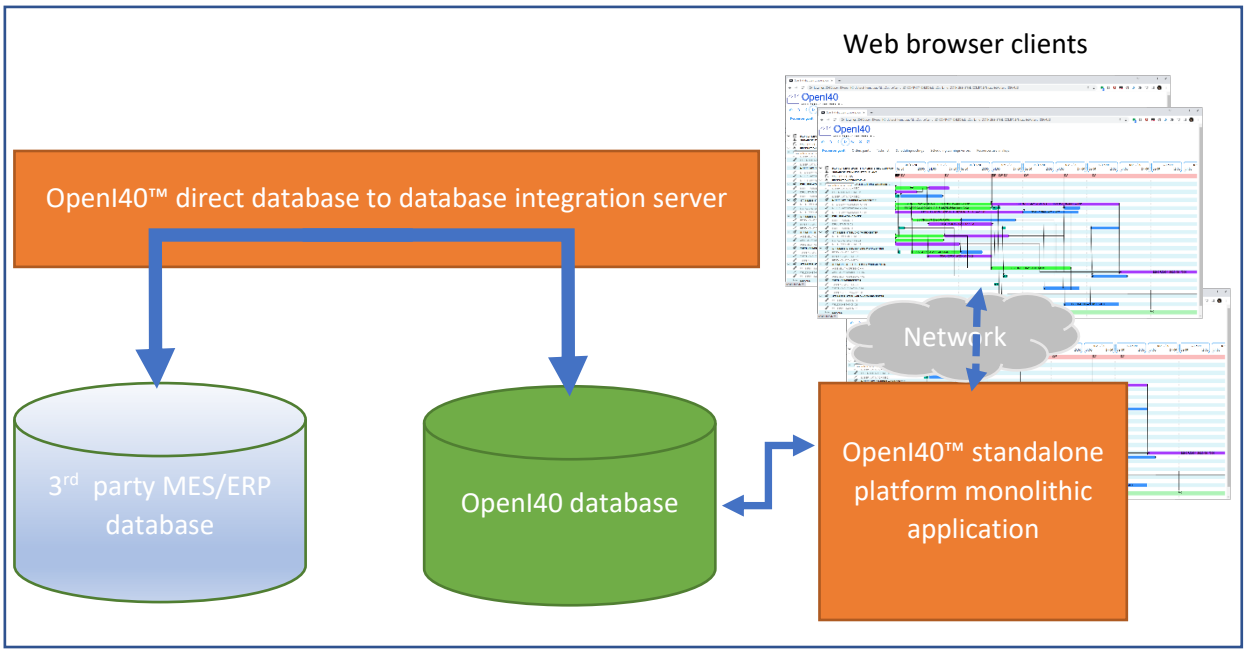
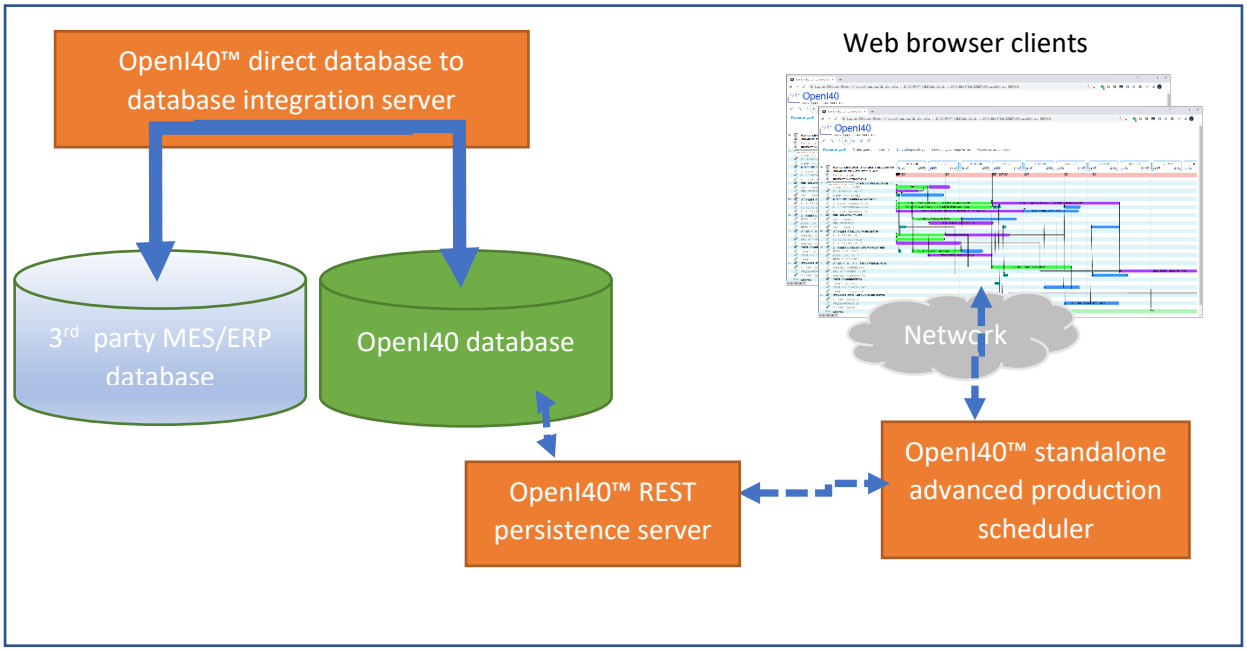


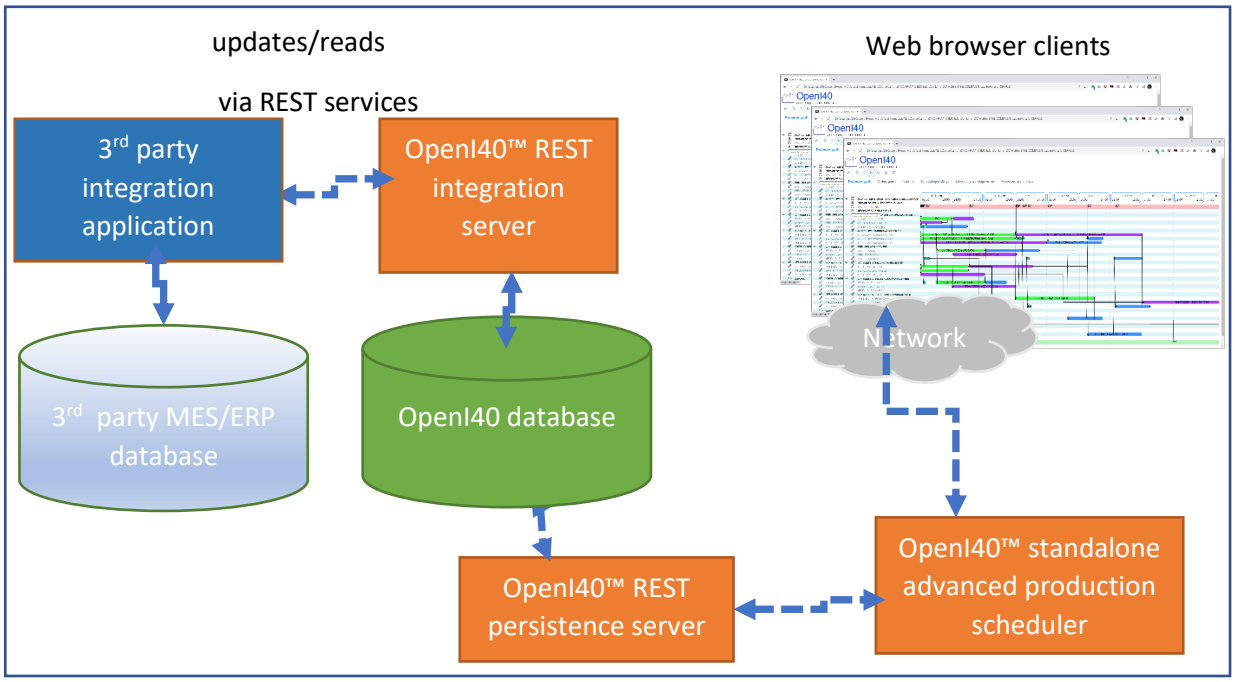
Figure 7 Fully functional and integrated via db to db with your ERP/MES solution

3rd party MES/ERP integration + OpenI40™ platform with separated OpenI40™ persistence server + OpenI40™ standalone advanced production scheduler

This configuration split the load/save stage of the APS and the pure scheduling Engine that acts as a client of the OpenI40™ persistence server. You create clusters for both the persistence server and the REST integration server.

More information(s) about developing/configuring integration with OpenI40™ tools can be found in the document: "**OpenI40™ platform integration guide**".





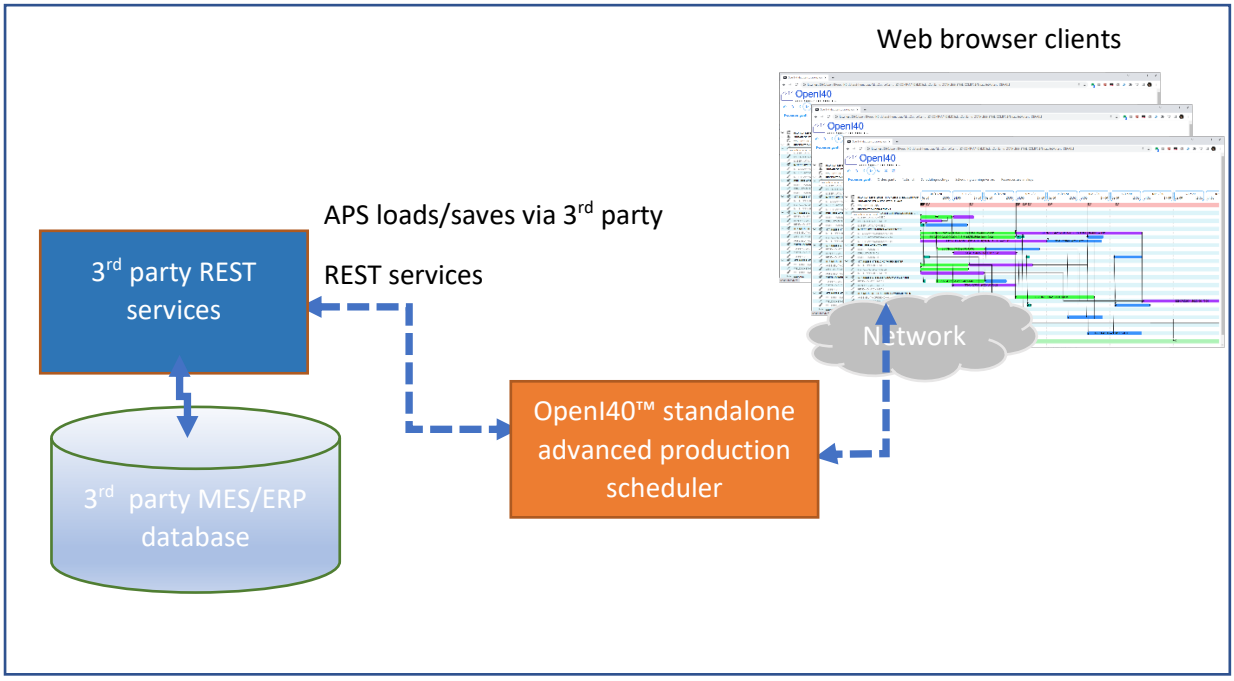
3rd party MES/ERP to OpenI40™ standalone advanced production scheduler direct REST integration.

More information(s) about developing/configuring integration with OpenI40™ tools can be found in the document: "**OpenI40™ platform integration guide**".

Using this integration schema you can provide REST services for data load and save to the OpenI40 APS standalone engine.

Read on the Appendix A how to configure the OpenI40™ standalone APS to use REST services.

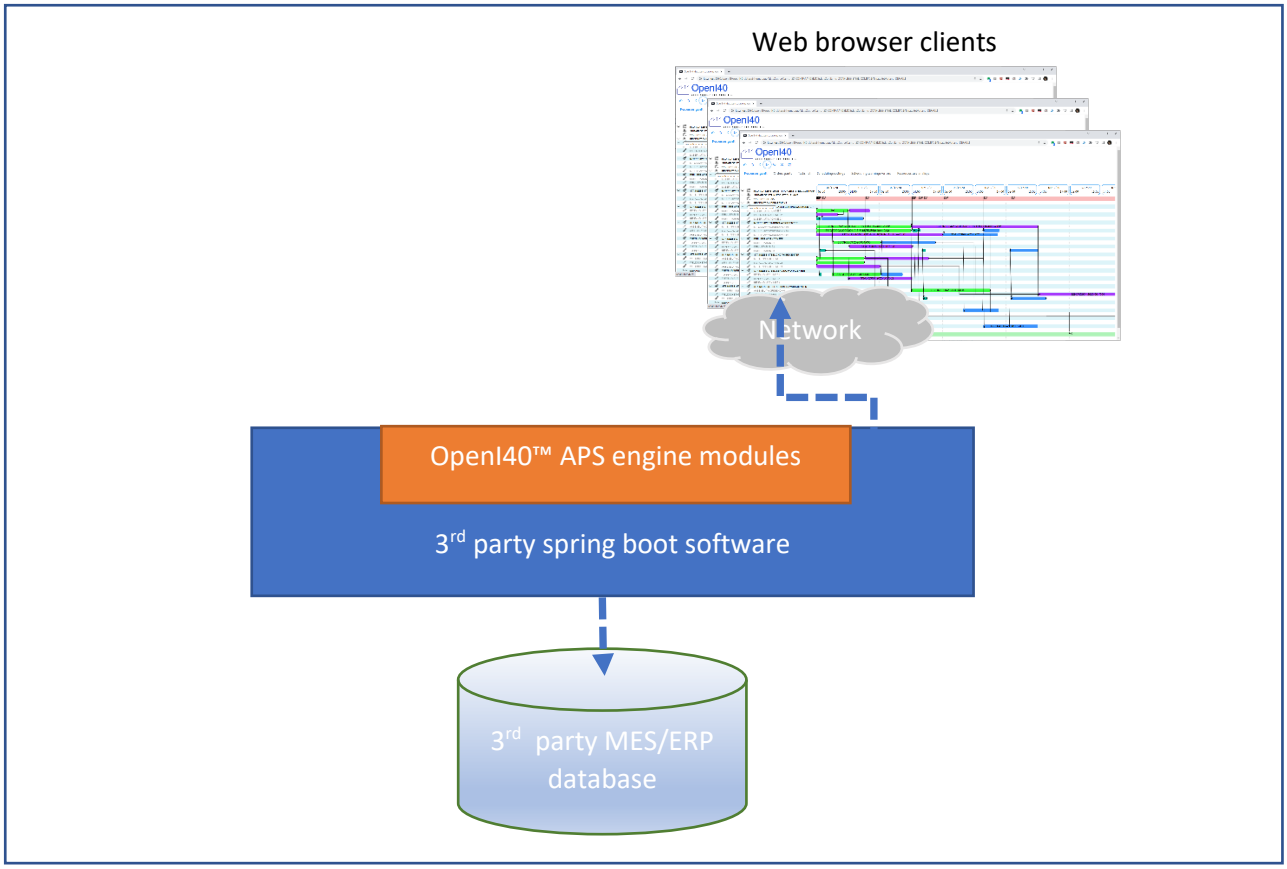
We suggest to directly use the swagger stub of the **OpenI40™ REST persistence server** to generate (using swagger-codegen) skeleton of services and data models in your preferred software architecture.



3rd party MES/ERP software embedding of OpenI40™ advanced production scheduler engine modules

To implement this embedding pattern is mandatory to use spring boot technologies and develop specific services/components coherent with OpenI40™ architecture.

More information(s) about developing/configuring integration with OpenI40™ tools can be found in the document: "**OpenI40™ platform integration guide**".



OpenI40™ detailed maven artifacts list

It follows the list of OpenI40™ maven artifacts and their architectural role description

Maven group	Artifact id	
com.openi40	parent	General Root project
com.openi40.commons	parent	Common packages parent project
	Mapper	Data layers mapping framework
	WebConfigs	Common web/rest/swagger/websockets configurations
com.openi40.scheduler	parent	APS engine root project
	Common	Common engine data architecture
	SchedulerKernelArchitecture	APS engine core business logic architecture patterns package
	Input.Model	Data input model for APS engine
	Output.Model	Output data model for APS engine
	Client.Model	Data model for REST client gui rappresentation
	CommonDataModel	Data model annotations for input data.
	Model	Internal APS engine data rappresentation
	Model.Daos	DAO pattern(s) implementation to interact with internal APS engine data model.
	Engine	APS Engine algorithms
	Engine.Standalone	APS Engine standalone bootable module
	InputChannels	Input channels abstraction layer
	Output.Channels	Output channels abstraction layer
	InputOutputConfig	Input & output configuration data rappresentation
	MapperModules	Root pom project for internal APS engine mappers
	InputModel2ApsModelMappers	Input data model to internal engine data model mappers
	ApsModel2OutputModel	Internal engine data model to output data model mappers
	ApsModel2ClientModelMappers	Internal engine data model to client data model mappers
	Client.RestApi	Rest API for Angular GUI client or other clients.
	ApsDataCache	Engine data cache services, core functionalities for access in memory scheduler data.
	angular-app	Angular GUI application
com.openi40.platform	parent	Root POM project for OpenI40 platform projects

	Persistence.Input.Channel	JPA OpenI40™ database input channel implementation
	Persistence.Output.Channel	Database output channel implementation
	Persistence.Config	Database input/output spring boot configuration data model
	openi40.app	OpenI40™ platform standalone application (APS engine + database channels)
	openi40.persistence.server	OpenI40™ persistence REST server
	openi40.dbmodel	JPA OpenI40™ single table(s) mapping data model
	openi40.dbmodel.repositories	Spring boot data/jpa repositories for single table(s) data model
	openi40.dbmodel.integration.app	OpenI40™ database integration REST services
	openi40.generical.dbintegration	OpenI40™ direct database to database integration configurable/customizable application.

Appendix A: advanced configuration details

OpenI40™ standalone advanced production scheduler advanced configuration(s)

Configure scheduler datasource on REST web services input & output

Let's configure:

REST input services configuration:

```
com.openi40.scheduler.apsinputchannels:
        httpClientInputs:
          -
             dataSourceName: <data source name code>
             dataSetName: <data set name code>
             dataSetVariant: <data set variant code>
             description: <displayed description>
             dataImporterId: <data importer unique identifier>
             useBasicAuthentication: false
             baseUrl: <remote base url>
             entitiesSetting:

               -
                 entityName: <entity name>
                 path: <absolute or relative URL to GET resource>
                 relativePath: <boolean, false if url is absolute>
                 multipleObjectsAsArray: <Boolean, if true data is
returned as json array otherwise is a stream of json objects with
only new line as separator>
```

List of entity names, you can see more information on these entities meaning/content on the integration guide:

- ApsWindowInputDto
- TimesheetMetaInfoInputDto
- ProductiveCompanyInputDto
- PlantInputDto
- DepartmentInputDto
- WorkCenterInputDto
- MachineInputDto
- ChangeOverMatrixItemInputDto
- ResourceGroupInputDto
- SecondaryResourceInputDto
- WarehouseInputDto
- ProductInputDto
- StockSupplyInputDto
- CycleModelInputDto
- SalesOrderInputDto
- PurchaseOrderInputDto
- ProductiveCompanyProductSettingInputDto

- PlantProductSettingInputDto
- WarehouseProductSettingInputDto
- WorkOrderInputDto
- TaskInputDto
- TaskRelationInputDto
- TaskProductionMaterialReservationInputDto
- TaskStockMaterialReservationInputDto
- TaskPurchaseMaterialReservationInputDto
- PeggingInputDto
- ApsSchedulingSetInputDto

REST output services configuration

```
com.openi40.scheduler.apsoutputchannels:
        restConfigs:
          -
            dataSourceName: <data source name code>
            dataSetName: <data set name code>
            dataSetVariant: <data set variant code>
            description: <displayed description>
            dataImporterId: <data exporter unique identifier>
            baseURL: <remote base url>
            entitiesSetting:
              -
                entityName: TaskOutputDto
                path: <remote path>
                nEntriesGrouping: 1000
                relativePath: false
              -
                entityName: TaskRelationOutputDto
                path: <remote path>
                nEntriesGrouping: 1000
                relativePath: false
              -
                entityName: WorkOrderOutputDto
                path:  <remote path>
                nEntriesGrouping: 1000
                relativePath: false
              -
                entityName: PeggingOutputDto
                path:  <remote path>
                nEntriesGrouping: 1000
                relativePath: false
              -
                entityName: ApsSchedulingSetOutputDto
                path:  <remote path>
                nEntriesGrouping: 1000
                relativePath: false
```

To successfully complete input & output configurations for REST services endpoints, dataSourceName,dataSetName,dataSetVariant of both input/output must have same configuration values.

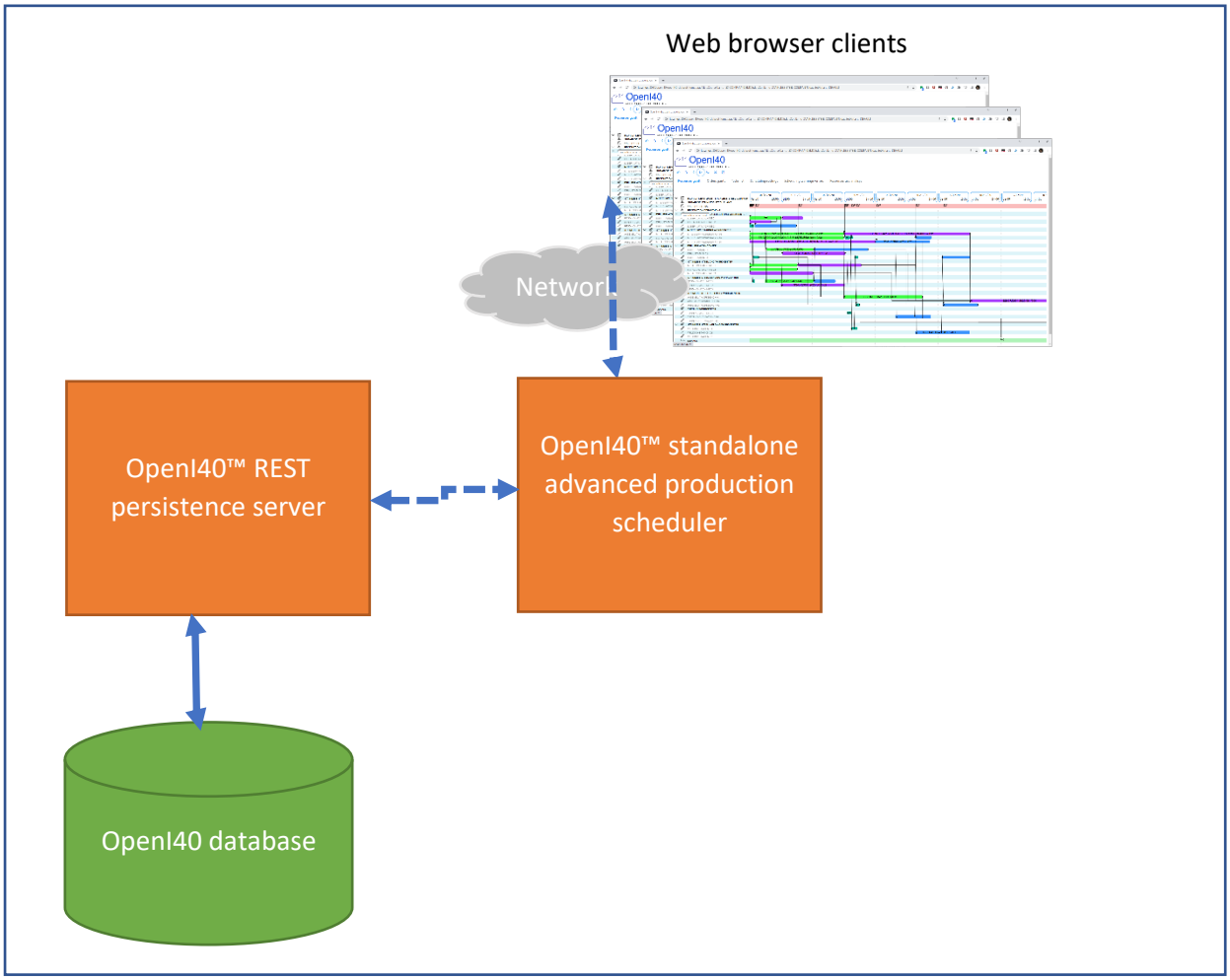
Configure OpenI40™ standalone advanced production scheduler to load data from external single JSON file

Example:

```
com.openi40.scheduler.apsinputchannels:
      jsonFileInputs:
          -
            dataSourceName: ARBALETE-DEMO
            dataImporterId: demosImporter
            dataSetName: ARBALETE-COMPANY
            dataSetVariant: DEFAULT
            description: 'Florida Keys Arbaletes company'
            baseFolder: demosource
            createURIByEntityName: false
            hasIncrementalUpdates: false
            singleApsInputUri: ArbaleteCompany.json
          -
            dataSourceName: SS-COMPANY-DEMO
            dataImporterId: demosImporter1
            dataSetName: STAINLESS-STEEL-COMPANY
            dataSetVariant: DEFAULT
            description: 'Florida Keys sport stainless steel
company'
            baseFolder: demosource
            createURIByEntityName: false
            hasIncrementalUpdates: false
            singleApsInputUri: StainlessSteelCompany.json
```

Configuring OpenI40™ REST persistence server ←→ OpenI40™ standalone advanced production scheduler connectivity

Configuration schema image



OpenI40™ REST persistence server configuration on application.yml:

```
hibernate.cache.use_second_level_cache: false

#spring.datasource.driver-class-name: org.postgresql.Driver
spring.datasource.hikari.connectionTimeout: 20000
spring.datasource.hikari.maximumPoolSize: 5
spring.datasource.username: postgres
spring.datasource.password: <password>
spring.datasource.url:
jdbc:postgresql://192.168.23.135:5432/OPENI40

com.openi40.platform.dbchannels:
        useJpaStreaming: false
        batchingSize: 1000
        configs:
          -
           dataSourceName: REST-LINK
```

```
            dataSetName: REST-DATASET
            dataSetVariant: DEFAULT
            dataSourceDescription: Data loaded from rest via db
```

OpenI40™ standalone production scheduler configuration on application.yml:

```
com.openi40.scheduler.apsinputchannels:
        httpClientInputs:
         -
            dataSourceName: REST-LINK
            dataSetName: REST-DATASET
            dataSetVariant: DEFAULT
            description: Data loaded from remote rest
            dataImporterId: demosRestImporter
            useBasicAuthentication: false
            baseUrl: http://localhost:8081/openi40-restserver/
            entitiesSetting:
              -
                entityName: ApsWindowInputDto
                path: http://localhost:8081/openi40-
restserver/input/ApsWindow/stream/REST-LINK/REST-DATASET/DEFAULT/
                relativePath: false
                multipleObjectsAsArray: false
              -
                entityName: TimesheetMetaInfoInputDto
                path: http://localhost:8081/openi40-
restserver/input/TimesheetMetaInfo/stream/REST-LINK/REST-
DATASET/DEFAULT/
                relativePath: false
                multipleObjectsAsArray: false
              -
                entityName: ProductiveCompanyInputDto
                path: http://localhost:8081/openi40-
restserver/input/ProductiveCompany/stream/REST-LINK/REST-
DATASET/DEFAULT/
                relativePath: false
                multipleObjectsAsArray: false
              -
                entityName: PlantInputDto
                path: http://localhost:8081/openi40-
restserver/input/Plant/stream/REST-LINK/REST-DATASET/DEFAULT/
                relativePath: false
                multipleObjectsAsArray: false
              -
                entityName: DepartmentInputDto
                path: http://localhost:8081/openi40-
restserver/input/Department/stream/REST-LINK/REST-DATASET/DEFAULT/
                relativePath: false
```

```
      multipleObjectsAsArray: false
    -
      entityName: WorkCenterInputDto
      path: http://localhost:8081/openi40-
restserver/input/WorkCenter/stream/REST-LINK/REST-DATASET/DEFAULT/
      relativePath: false
      multipleObjectsAsArray: false
    -
      entityName: MachineInputDto
      path: http://localhost:8081/openi40-
restserver/input/Machine/stream/REST-LINK/REST-DATASET/DEFAULT/
      relativePath: false
      multipleObjectsAsArray: false
    -
      entityName: ChangeOverMatrixItemInputDto
      path: http://localhost:8081/openi40-
restserver/input/ChangeOverMatrixItem/stream/REST-LINK/REST-
DATASET/DEFAULT/
      relativePath: false
      multipleObjectsAsArray: false
    -
      entityName: ResourceGroupInputDto
      path: http://localhost:8081/openi40-
restserver/input/ResourceGroup/stream/REST-LINK/REST-
DATASET/DEFAULT/
      relativePath: false
      multipleObjectsAsArray: false
    -
      entityName: SecondaryResourceInputDto
      path: http://localhost:8081/openi40-
restserver/input/SecondaryResource/stream/REST-LINK/REST-
DATASET/DEFAULT/
      relativePath: false
      multipleObjectsAsArray: false
    -
      entityName: WarehouseInputDto
      path: http://localhost:8081/openi40-
restserver/input/Warehouse/stream/REST-LINK/REST-DATASET/DEFAULT/
      relativePath: false
      multipleObjectsAsArray: false
    -
      entityName: ProductInputDto
      path: http://localhost:8081/openi40-
restserver/input/Product/stream/REST-LINK/REST-DATASET/DEFAULT/
      relativePath: false
      multipleObjectsAsArray: false
    -
      entityName: StockSupplyInputDto
```

```
                path: http://localhost:8081/openi40-
restserver/input/StockSupply/stream/REST-LINK/REST-DATASET/DEFAULT/
                relativePath: false
                multipleObjectsAsArray: false
              -
                entityName: CycleModelInputDto
                path: http://localhost:8081/openi40-
restserver/input/CycleModel/stream/REST-LINK/REST-DATASET/DEFAULT/
                relativePath: false
                multipleObjectsAsArray: false
              -
                entityName: SalesOrderInputDto
                path: http://localhost:8081/openi40-
restserver/input/SalesOrder/stream/REST-LINK/REST-DATASET/DEFAULT/
                relativePath: false
                multipleObjectsAsArray: false
              -
                entityName: PurchaseInputDto
                path: http://localhost:8081/openi40-
restserver/input/Purchase/stream/REST-LINK/REST-DATASET/DEFAULT/
                relativePath: false
                multipleObjectsAsArray: false
              -
                entityName: PurchaseOrderInputDto
                path: http://localhost:8081/openi40-
restserver/input/PurchaseOrder/stream/REST-LINK/REST-
DATASET/DEFAULT/
                relativePath: false
                multipleObjectsAsArray: false
              -
                entityName: ProductiveCompanyProductSettingInputDto
                path: http://localhost:8081/openi40-
restserver/input/ProductiveCompanyProductSetting/stream/REST-
LINK/REST-DATASET/DEFAULT/
                relativePath: false
                multipleObjectsAsArray: false
              -
                entityName: PlantProductSettingInputDto
                path: http://localhost:8081/openi40-
restserver/input/PlantProductSetting/stream/REST-LINK/REST-
DATASET/DEFAULT/
                relativePath: false
                multipleObjectsAsArray: false
              -
                entityName: WarehouseProductSettingInputDto
                path: http://localhost:8081/openi40-
restserver/input/WarehouseProductSetting/stream/REST-LINK/REST-
DATASET/DEFAULT/
```

```
                relativePath: false
                multipleObjectsAsArray: false
              -
                entityName: WorkOrderInputDto
                path: http://localhost:8081/openi40-
restserver/input/WorkOrder/stream/REST-LINK/REST-DATASET/DEFAULT/
                relativePath: false
                multipleObjectsAsArray: false
              -
                entityName: TaskInputDto
                path: http://localhost:8081/openi40-
restserver/input/Task/stream/REST-LINK/REST-DATASET/DEFAULT/
                relativePath: false
                multipleObjectsAsArray: false
              -
                entityName: TaskRelationInputDto
                path: http://localhost:8081/openi40-
restserver/input/TaskRelation/stream/REST-LINK/REST-DATASET/DEFAULT/
                relativePath: false
                multipleObjectsAsArray: false
              -
                entityName:
TaskProductionMaterialReservationInputDto
                path: http://localhost:8081/openi40-
restserver/input/TaskProductionMaterialReservation/stream/REST-
LINK/REST-DATASET/DEFAULT/
                relativePath: false
                multipleObjectsAsArray: false
              -
                entityName: TaskStockMaterialReservationInputDto
                path: http://localhost:8081/openi40-
restserver/input/TaskStockMaterialReservation/stream/REST-LINK/REST-
DATASET/DEFAULT/
                relativePath: false
                multipleObjectsAsArray: false
              -
                entityName: TaskPurchaseMaterialReservationInputDto
                path: http://localhost:8081/openi40-
restserver/input/TaskPurchaseMaterialReservation/stream/REST-
LINK/REST-DATASET/DEFAULT/
                relativePath: false
                multipleObjectsAsArray: false
              -
                entityName: PeggingInputDto
                path: http://localhost:8081/openi40-
restserver/input/Pegging/stream/REST-LINK/REST-DATASET/DEFAULT/
                relativePath: false
                multipleObjectsAsArray: false
```

```
            -
              entityName: ApsSchedulingSetInputDto
              path: http://localhost:8081/openi40-
restserver/input/ApsSchedulingSet/stream/REST-LINK/REST-
DATASET/DEFAULT/
              relativePath: false
              multipleObjectsAsArray: false

com.openi40.scheduler.apsoutputchannels:
        restConfigs:
          -
            dataSourceName: REST-LINK
            dataSetName: REST-DATASET
            dataSetVariant: DEFAULT
            description: Data loaded from remote rest
            dataImporterId: demosRestExporter
            baseURL: http://localhost:8081/openi40-restserver/
            entitiesSetting:
              -
                entityName: TaskOutputDto
                path: http://localhost:8081/openi40-
restserver/output/Task/vectorialSave/REST-LINK/REST-DATASET/DEFAULT/
                nEntriesGrouping: 1000
                relativePath: false
              -
                entityName: TaskRelationOutputDto
                path: http://localhost:8081/openi40-
restserver/output/TaskRelation/vectorialSave/REST-LINK/REST-
DATASET/DEFAULT/
                nEntriesGrouping: 1000
                relativePath: false
              -
                entityName: WorkOrderOutputDto
                path: http://localhost:8081/openi40-
restserver/output/WorkOrder/vectorialSave/REST-LINK/REST-
DATASET/DEFAULT/
                nEntriesGrouping: 1000
                relativePath: false
              -
                entityName: PeggingOutputDto
                path: http://localhost:8081/openi40-
restserver/output/Pegging/vectorialSave/REST-LINK/REST-
DATASET/DEFAULT/
                nEntriesGrouping: 1000
                relativePath: false
              -
                entityName: ApsSchedulingSetOutputDto
```

```
                path: http://localhost:8081/openi40-
restserver/output/ApsSchedulingSet/vectorialSave/REST-LINK/REST-
DATASET/DEFAULT/
                nEntriesGrouping: 1000
                relativePath: false
```