Decision Support Tool: Description and selection of available documentation

Table of contents

1. Introduction

1.1 System overview

Decision Support Tool: Innovative Capacity Management at Schiphol Airport
Air Traffic Control the Netherlands (LVNL) continuously seeks to improve operations at Amsterdam Schiphol Airport, one of the busiest and most challenging European hubs. LVNL adds value for our airline customers by delivering an optimal performance in terms of safety, capacity, and environment. To further improve performance, LVNL developed the Decision Support Tool (DST), a groundbreaking innovation for Amsterdam ACC Operational Supervisors and Flow Management Position Controllers (FMPCs).

Less delay for airlines and travellers
The ultimate goal of DST is fewer delays, which is beneficial for both airlines and travellers. To keep the volume of flights to Schiphol Airport in balance with the capacity of the Dutch airspace, LVNL must frequently regulate flights. Regulating involves delaying flights at their departure airport by holding aircraft at the gate, resulting in a later arrival at Schiphol airport. These so called ATFM delays are costly for airlines and frustrating for travellers. DST uses new simulation techniques and machine learning to achieve an optimal balance between capacity and sustainability and does this well before flights enter the Dutch airspace. Finding this optimal balance for Schiphol Airport and the Dutch airspace is a highly challenging and complex process that involves integrating a large number of variables and factors. These include runway-changes, aircraft performance, runway capacities, weather conditions, traffic-demand predictions and expected air traffic controller workload.

Machine learning and simulation functionality
DST supports this complicated decision making process by predicting the available runways, sector and runway capacity, air traffic controller workload, weather data, and expected traffic load. To enhance the predictions LVNL uses advanced Machine Learning and Data Science techniques. With improved simulation capabilities, an optimal solution that causes as little delay as possible can be chosen. Supervisors and FMPCs can, for example, simulate different ATFM regulation rates, changing weather conditions, or runway configurations. Furthermore, it is possible to simulate the effectiveness of flow exclusion measures as well as the effect of planning business jets on a separate runway. DST continuously updates such simulations based on the latest information. As a result, supervisors and FMPCs are able to choose the optimal solution under changing conditions.

Benefits
In 2019, more than one million ATFM delay minutes were required in Dutch airspace in order to balance discrepancies between air traffic volume and capacity. Every prevented delay minute saves airlines an estimated €100 in costs. Much of the delay has external causes, such as adverse weather conditions (e.g., fog, or wind), and runway maintenance. This delay cannot be avoided. However, first results show that DST can potentially save

several percentage points, resulting in millions of annual cost savings for airlines. Furthermore, the use of DST can have a positive effect on environmental KPI's such as CO_2 emissions. With DST overloads of available capacity can be better predicted and avoided, resulting in less vectoring and use of holding patterns.

Operational acceptance

LVNL takes pride in both the advantages that DST brings to our customers as well as in the agile and innovative development process. DST has been developed by combining the knowledge of operational supervisors and FMPCs with the know-how of partners using the Scrum approach. This resulted in a groundbreaking system with enriched integrated data, a high tech back-end and an easy to use front-end, which was embraced from the start by the Operational Supervisors and Flow Management Position Controllers.

DST has been deployed alongside existing systems since April 2023 and is regularly updated with new releases. One of the latest developments is the Taskplanning functionality for the Operational Supervisors. The Taskplanning (in Dutch “Takenplanning”) module assists in the assignment of tasks to air traffic controllers. Purpose of the Taskplanning functionality is the optimal deployment of operational personnel.

1.1.1 System purpose

The Decision Support Tool is intended to support LVNL Supervisors in balancing capacity and demand of air traffic in the LVNL airspace and to and from Schiphol airport. LVNL seeks a decision support tool (DST) to support air traffic control supervisors in their decisions about flow and capacity management. Currently, the supervisors make the decisions based on their experience and expert judgment. A decision support system adds more depth, independence, consistency and accountability to the decisions. In addition, with increasing complexity and volume of air traffic, efficiency is gained and errors are reduced.

The Decision Support Tool shall provide the following main functions:

- Provide more accurate and complete insight into air traffic load.
- Provide more accurate and complete insight into capacity/workload of run ways, air space and air traffic controllers.
- Provide tools to balance capacity and demand, e.g.:
    - support for employing ATFM regulations;
    - balance ATFM regulations and FIR-delay;
    - composing and comparing different scenarios (weather, runway configuration changes).
- Signal possible overload situations.
- Assist in the assignment of tasks to air traffic controllers.(Taskplanning)

1.1.2 System Context

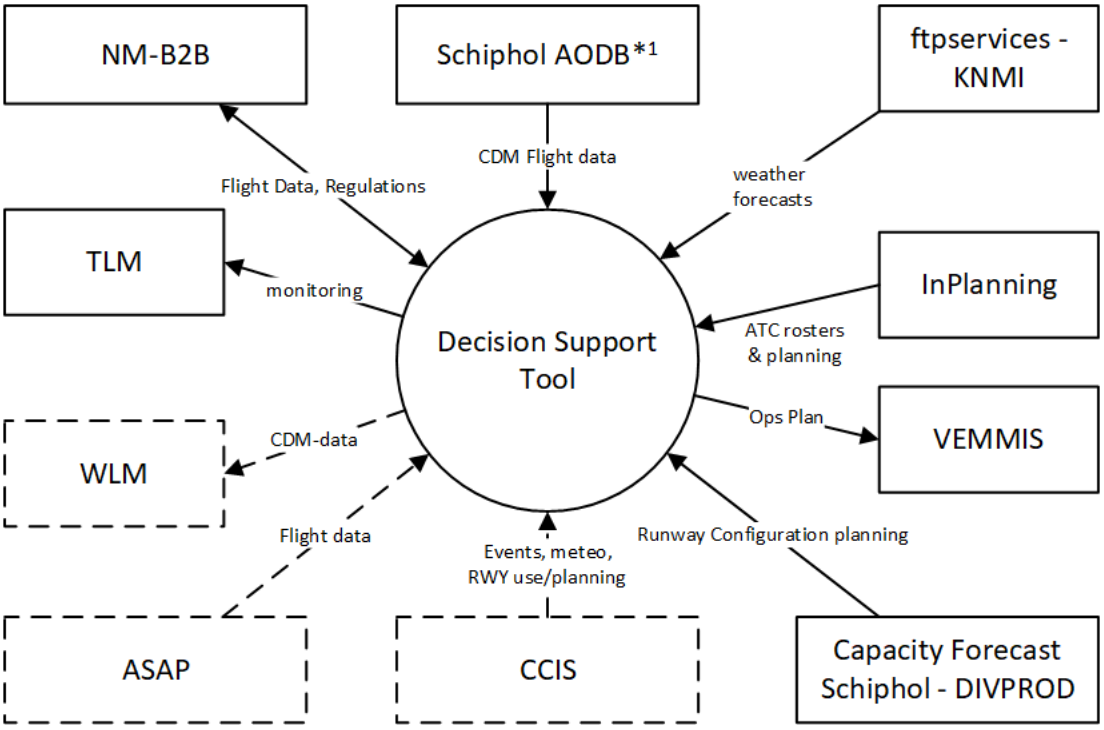


Figure 1-1 DST Context overview

The Decision Support Tool's context consists of:

1. Eurocontrol **Network Manager B2B** for retrieving among others: all current relevant Flight Data and Regulations. Also allows for communicating new and modified Regulations to Network Manager.
2. **Schiphol AODB:** Schiphol AODB for receiving CDM departure Flight Data.
   (*2 Until Schiphol AODB is actually available this data will be received through the Schiphol developer - Operational Flight API which provides the same data via another technical interface.)
3. **FTPPro – KNMI:** KNMI FTPro(although CCIS is prefered) provides high resolution (high altitude) wind forecasts and wind and visibility forecasts (Schiphol KansVerwachting, SKV).
4. **Capacity Forcast Schiphol - DIVPROD** – (although CCIS is prefered) provides DST with runway configuration and capacity planning data
5. **InPlanning** – InPlanning provides the DST Planningtool with roster and planning information of Air Traffic Controllers and Supervisors.
6. **TLM**: DST provides monitoring data to the Top-level Monitoring System.
7. **CCIS**: [Future Operating Capability] CCIS is the prefered supplier for: Capacity Forecast Schiphol, SKV (weather forecast), high resolution wind forecasts and runway use data
8. **ASAP:** [Future Operating Capability] The Arrival MANager system of LVNL will provide more accurate approach planning and landing times to DST.
9. **WLM**: [Future Operating Capability] DST will provide CDM-data received from Schiphol AODB to WLM.
10. **VEMMIS**: VEMMIS is an external database used by LVNL to store data in for further analyses. DST will insert Ops Plan data in the VEMMIS database.

2. Referenced documents

2.1 Normative Documents

Normative documents contain requirements and/or design constraints on the system.

Reference	Title	Version, date, other
[MRA]	Machine Reference Architecture	V4.92, 20-07-2021, author: LVNL
[DST_SSS]	System Subsystem Specification - Decision Support Tool	V0.9.11, 21-04-2021, author: DST Team
[ICD_CDM_AODB]	CISS WLM Flight	V1.0, 28-08-2021, author: Schiphol Group
[ICD_EIF_IMC]	IMC - Interface Control Document (EM based on Zabbix or Zabbix Sender Protocol)	v1.0, 2020-06-17, author: LVNL
[J_DST_016]	Standard for Information Technology Software Life Cycle Processes Software Development Acquirer-Supplier Agreement	1995, 1995-09-039, author: IEEE / EIA
[SDD_DST_CAP]	SDD – DST – Capacity Predictor	v2.0, 2020-03-04, author: DST Team
[SDD_DST_HIRES]	SDD - Decision Support Tool - High-resolution wind processor	V1,0, 2021-07-01, author: DST Team
[B2B_REF]	Eurocontrol B2B Reference Manual	V27.2.0.2.154, author: Eurocontrol

Table 1 Referenced normative documents

3. System-wide design decisions

3.1 Design principles

Design principles:

- Secure by design: Process isolation, encrypted communication, network segmentation and certificate based authentication and when needed authorisation.
- Highly available; resilient to failure of components and interfaces
- Fast detection and reporting of failures and errors
- Fast recovery from failures
- Automatic recovery from failures
- Easy deployment with minimum (operating) system configuration, preferably facilitate deployment on both physical and virtual infrastructure and various Operating Systems.
- Minimize impact of changing requirements, because the system will be developed iteratively without big-design-up-front.
- Use best programming language for the task at hand.

3.2 Design Decisions

SSDD-001 Client-Server Architecture

The system has a client-server architecture consisting of Frontend Services (client) and Backend Services (Server):

1. Backend Services provide services to the Frontend Services and to external systems through external interfaces. Backend services are only accessible via the external interfaces and the frontend services. The Backend Services can be deployed on multiple servers.
2. Frontend services make use of the Backend Services to provide the Human Machine Interface (user interface) to the users. The graphical user interfaces are web interfaces. The Frontend Services can used from multiple client systems (e.g. desktop PC's, remote desktop environments)

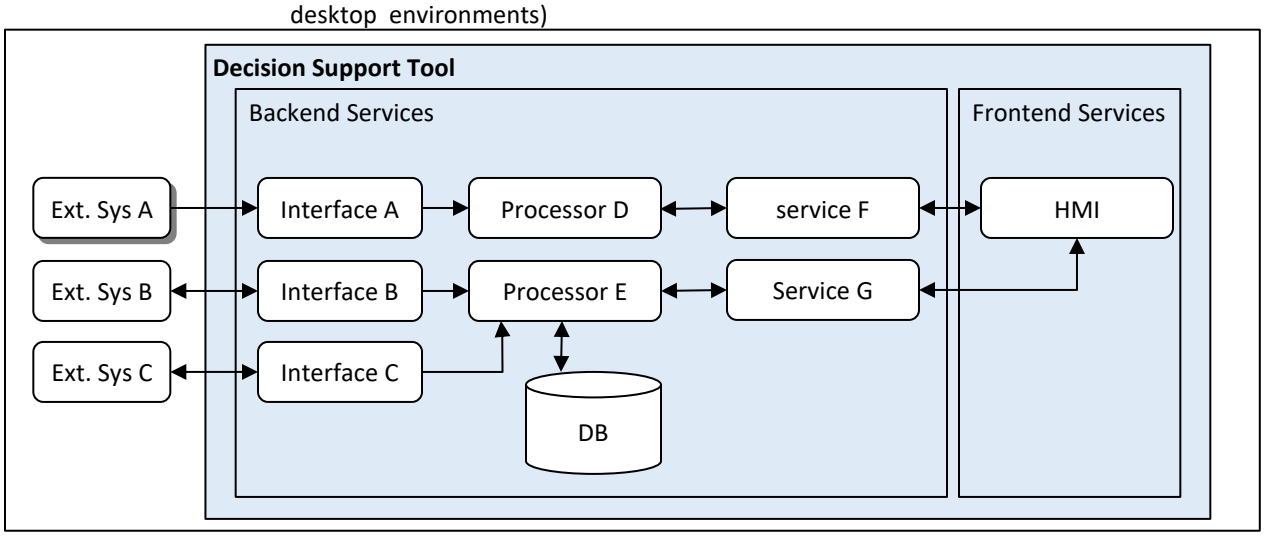


Figure 3-1 Client-Server Architecture

Rationale:

The separation of backend and frontend services makes it possible to apply strict network seggregation for securing backend services and the external systems from unwanted access other than through the HMI or possibly by other systems in the operational environment of LVNL which might be connected to the Decission Support Tool.

SSDD-002 Programming languages

The following programming languages will be used for the main software development:

- Java for backend services, or
- Python for machine learning components
- Typescript for in-browser HMI code

Note:

For support tools or other (simple) scripting purposes other languages such as Bash may be used.

SSDD-003 Frameworks

The following frameworks will be used for system development:

- Spring Framework
  For Backend Services developed in Java
- Angular
  For HMI development in the web frontend.
- SciKitPy
  A Python-based ecosystem of open-source software for mathematics, science, and engineering.
- SciKit
  Machine Learning tools for Python.

SSDD-004 Deployment using Docker Containers

The system shall be deployed using Docker containers and Docker Compose for basic container orchestration.

Rationale:

Docker containers make it easy to deploy the system and minimizes system environment differences between development, testing, acceptation and production environments. Docker Compose has the basic functionality needed for controlling the containers for the system and is easy to understand and use.

Docker isolates processes running in the containers from the host operating system and the other containers. This also makes is

SSDD-005 Operating System

The DST-backend component shall run on a Redhat Enterprise Linux Operating system.

Remark:

As the DST-backend components will run in Docker containers, switching to other another version of Linux will not require extensive changes to the software nor documentation.

Rationale:

Redhat Enterprise Linux is LVNL has standardized.

SSDD-006 Single internal time source

The system shall use a single time keeping component as source for the DST system time in both Live mode and Playback mode.

Rationale:

Both in Live and Playback mode all components should use the same time. In Live mode, that will be the same as the system time of the operating system. In Playback mode the time of the data being played back will be used.

SSDD-007 External Time source

The system shall use the system time provided by the operating system as source for the current time when running in Live mode.

Rationale:

LVNL has a Central Clock System (CCS) that is used for providing the current time to systems of which the time should be synchronized. This facility will be used by the NTP-daemon of the operating system for time synchronization.

SSDD-008 Contents of errormessages

Logged and presented error messages shall contain at least the following information:

1. Title
2. Date and time of occurrence
3. Component that caused the error
4. Description of the failure, if the title is not sufficient
5. When presenting to a user through an HMI: A description of what the user can do to remedy the error.

SSDD-009 Publish-subscribe preferred

Publish-subscribe communication is preferred above request-response communication between components. The Preferred protocol is AMQP, versions 0-9-1 and 1.0

Rationale:

Publish and subscribe based interface make it easier to distribute data:

- Components can be quickly adapted to consume additional data streams

- Components are loosely coupled at runtime, making them more robust and decouples their lifecycle from that of other components. For instance it is easier for components to be restarted, without having to restart the whole system. Also chances of propagation of errors by for instance time-outs caused by failing components are also reduced.

In addition the use of the AMQP protocol is defined in principle STD33 in the LVNL Machine Reference Architecture .

4. System architectural design

DST consists of two different applications: Decision Support Tool and Taskplanning (in Dutch “Takenplanning”). The Decision Support tool is the tool used to aid supervisors and FPMC personnel in capacity planning and management whereas the Taskplanning helps the supervisors to assign tasks and plan personnel during the day. Both applications are part of the DST project and are related but they can be seen as different applications. For readability the components of each application will be described separately.

4.1 System components – Decision Support Tool

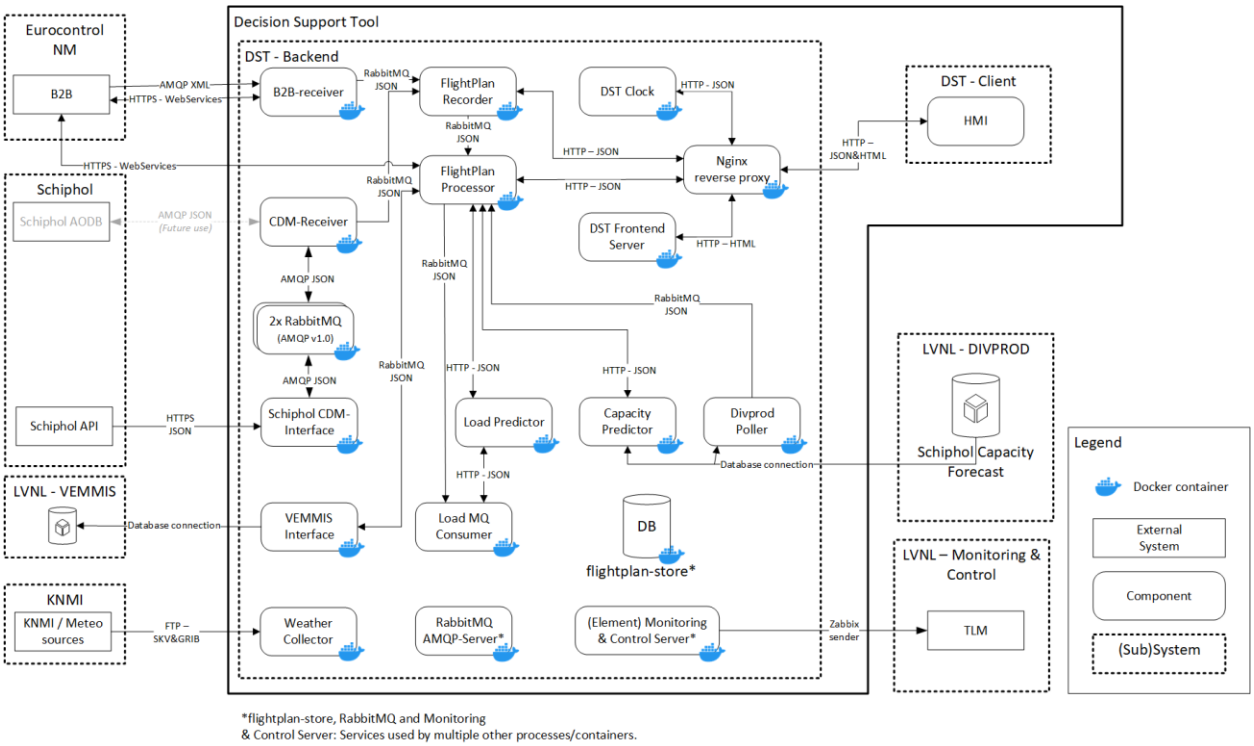


Figure 1 DST Component Overview

4.1.1 B2B-Receiver

The B2B-receiver is the main interface component to the Eurocontrol NM B2B system. It will receive flight plan data and regulations for processing by the Flightplan Recorder.

4.1.2 Capacity Predictor

The capacity predictor is used to predict the capacity of Schiphol for inbound and outbound traffic taking into account, among others: weather conditions, runway configurations and wake turbulence categories.

The Capacity predictor can also use many of the same parameters to calculate proposals for an optimal runway configuration for a period using many of the same parameters as used for capacity prediction.

4.1.2.1 Design decisions

SSDD-009 Compliance to runway planning regulations

The system shall take into account all relevant rules and regulations concerning (the planning of) runway use when proposing runway configuration.

Rationale:

This is needed to ensure compliance to rules and regulations. These include rules, for instance those for safety and efficiency, as well as government regulations and rules for noise reduction and other environmental factors. The resulting constraints of these rules and regulations are defined in Quick Reference Cards (QRC's), among others: QRC009 Baancombinaties en capaciteiten and QRC025 Baangebruik.

SSDD-010 Implementation of Arrival Manager algorithms

The system shall implement the following algorithms of an Arrival Manager system to enable scheduling of inbound flights and optimizing the sequence and timing of these inbound flights:

- Landing Interval (LIV) Calculation
  Taking into account:
  - Aircraft WTC (mix)
  - High resolution wind data
  - RECAT aircraft categories

Rationale:

The DST-System modifies (enhances) flight plans, especially estimates, by using machine learnings techniques and external data sources such as CDM-data from Schiphol AODB. The DST-System is capable of running simulations in which estimates can be further influenced by variations in, among others, ATFCM-regulations and runway configuration planning.
These algorithms allow more accurate prediction of the capacity of Schiphol for inbound traffic.

4.1.3 CDM Receiver

The CDM Receiver receives CDM flight plan data from the Schiphol AODB system. This data is used to enhance the outbound planning data for flights.

4.1.3.1 Schiphol CDM Interface

As The Schiphol AODB system is not yet available a temporary component, the Schiphol CDM Receiver, has been introduced that implements the same interfaces as the Schiphol AODB system. This temporary component uses the Schiphol developer – Operational Flight API to receive CDM data and supplies it to the CDM Receiver.

4.1.4 DST Clock

The DST-Clock provides other components, such as the HMI, with the current time. See design decision: SSDD-006

4.1.5 Element Monitor

This is the monitoring and control system for DST used for:

- Monitoring the status of components and interfaces
- Controlling components or the system as a whole
- Reporting the status of the system to the LVNL Top-level Monitoring System.

4.1.6 Flightplan Processor

The Flightplan Processor is the central data processor for the system. It receives data from multiple sources, processes it and provides data to other components such as the HMI and the Load Predictor.

The core functionality of the Flightplan Processor consists of:

- The Flightplan Processor serves as backend for the HMI.
- The Flightplan Processor stores the Flightplans and Regulations and provide them to other sub-systems when needed.
- Flightplans and regulations are received from external systems and merged into existing data when necessary.
- Calculation of the effects of Regulations using an approximation of the NM CASA algorithm.
- Calculation of the effects of excess demand in the FIR (in-FIR delay).
- Providing workload (WLM) data to the HMI.
- Performing simulations of the impact of changes like: weather conditions, ATFM regulations, runway configuration.

4.1.7 Flightplan Recorder

The Flightplan Recorder is responsible for storing all data which is required for:

- Running DST in Playback Mode
- Retraining machine learning models
- Auditing user actions
- Logging of technical system state

In addition, the Flightplan Recorder is capable of merging data from different sources and/or enhancing data using custom algorithms such as calculation of the Initial Approach Fix for inbound flights.

4.1.8 Flightplan Store

Database for persistent storage of among others:

- flight data,
- regulation data
- weather forecasts
- configuration/adaptation

4.1.9 Frontend Server

The backend webserver for the web-based HMI component.

4.1.10 High-Res Weather Interface

This components retrieves high resolution wind forecast data from KNMI and stores it in the Flightplan Store.

4.1.11 HMI

The HMI is the web-based user interface of the DST system that runs on the PC's of the users.

4.1.12 Load Predictor

The Load Predictor uses machine learning to supply the system with enhanced (better) arrival times of flights at the LVNL FIR airspace.

4.1.12.1 LoadMQ Consumer

The Load Predictor includes a component that is run as a separate process: the LoadMQ Consumer. This component receives updated flight data whenever that data changes and provides it to the Load Predictor.
See **Fout! Verwijzingsbron niet gevonden.** for more details.

4.1.13 RabbitMQ AMQP-server

This is the central message broker of the DST system which provides all internal publish and subscribe interfaces to the components of DST.

4.1.14 Reverse Proxy

An off the shelf reverse proxy server used to secure HTTP traffic using TLS (HTTPS) and also provide a secure single point of entry for all remote HTTP-traffic.

4.1.15 Divprod Poller

This module was developed as a temporary solution. It checks the DIVPROD-database for new SKV (weather forecast for Schiphol) data and new CFS (Schiphol capacity forecast) runway and notifies the Flightplan Processor when an update has been detected.

4.1.16 VEMMIS Interface

Ops-plan data that is received from the Flightplan Processor is stored with this module in the VEMMIS database of LVNL.

4.2 System Components – Taskplanning

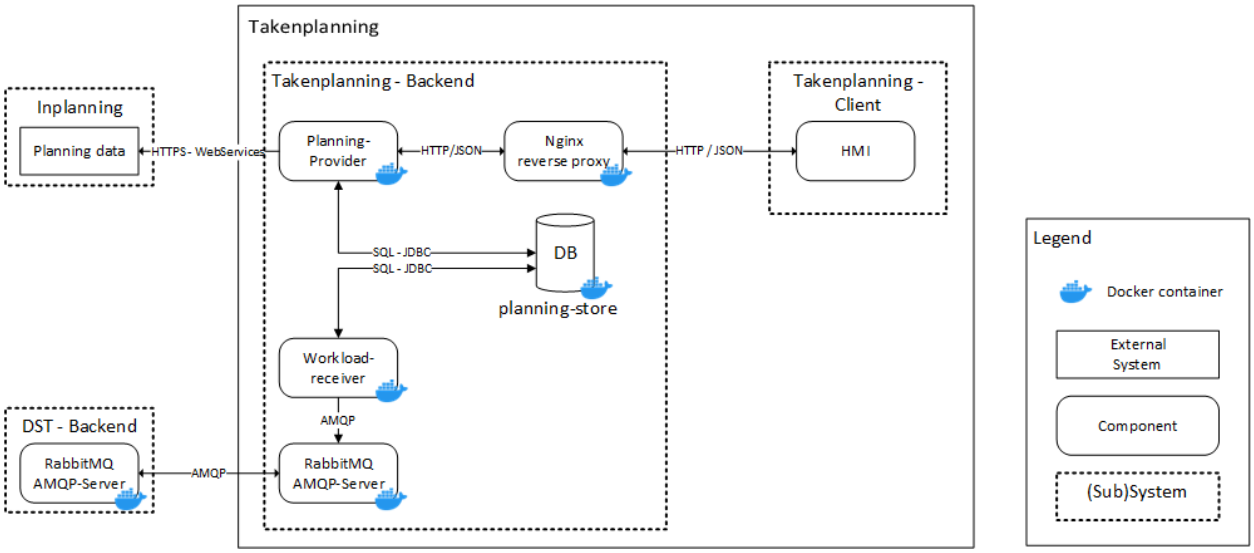


Figure 2: Taskplanning Component overview

4.2.1 Planning-Provider

The planning provider compent holds the main business logic of the Taskplanning application. It retrieves data from InPlanning on the scheduled air traffic controllers and solves the planning using the Timefold library. Data is retrieved and stored from the planning-store database and several HTTP endpoint are exposed for the frontend.

4.2.2 Planning store

PostgreSQL Database for persistent storage of among others:

- Solved rosters,
- Task schedules,
- Workload data

4.2.3 Workload Receiver

This component subscribes to the workload exchange of the RabbitMQ server and stores the data received in the planning-store.

4.2.4 RabbitMQ AMQP-server

This is the central message broker of the DST system which provides a link to the DST system for retrieval of workload data.

4.2.5 Reverse Proxy

An off the shelf reverse proxy server used to secure HTTP traffic using TLS (HTTPS) and also provide a secure single point of entry for all remote HTTP-traffic.

4.2.6 HMI

The HMI is the web-based user interface of the Taskplanning system that runs on the PC's of the users and the tablet in the operations room

4.3 Concept of execution – Decision Support Tool

This section describes the main data streams and interactions between the main components.

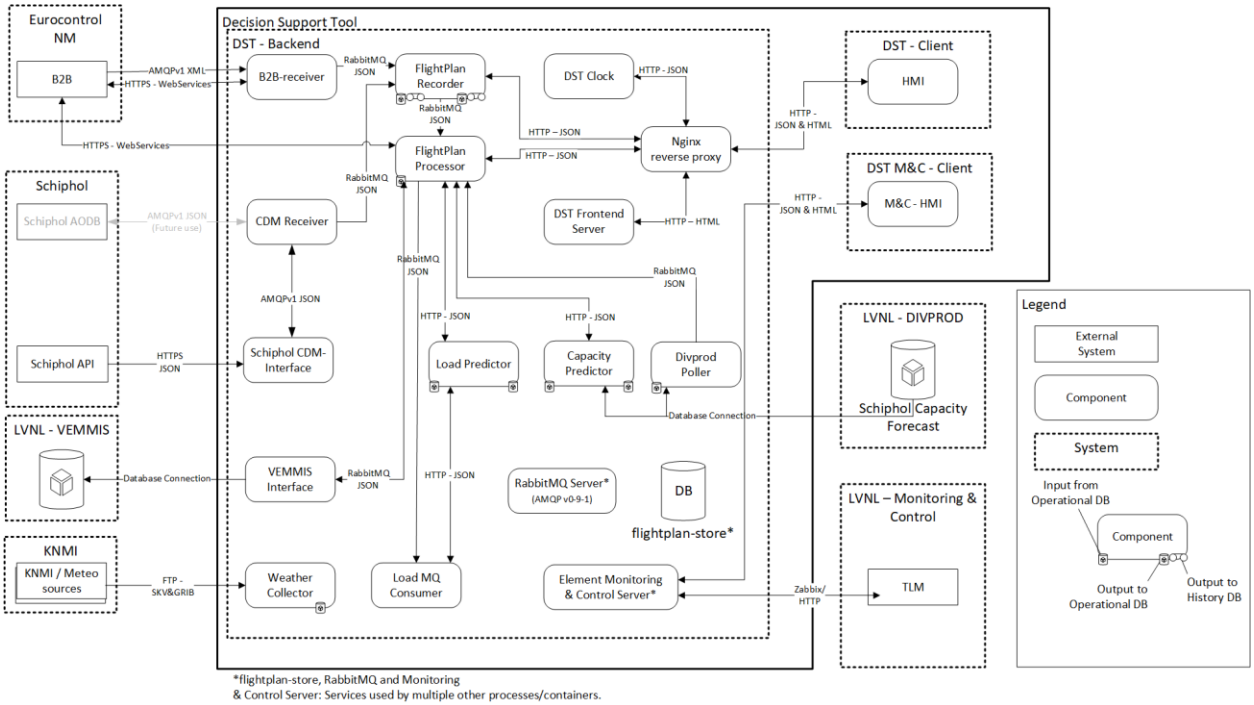


Figure 4-3 DST interfaces and protocols

4.3.1 Flight data processing

Sources of flight data: Eurocontrol NM B2B (EIF_B2B) and Schiphol AODB (EIF_CDM).
Flight data is received by the B2B Receiver and CDM Receiver components, transformed to the DST internal representation of flight data and send to the RabbitMQ server for distribution.

The Flightplan Recorder receives flight data, optionally stores it in the Flightplan Store and merges the data from the different sources and is capable of enhancing the data with its own algorithms and using other components. Enhanced flight data is sent to the RabbitMQ server for distribution and can also be stored in the Flightplan Store for later use in playback or training of machine learning models.

The Flightplan Processor reads flight data from the Flightplan Store, enhances the data using the Load Predictor and supplies the flight data to other components as the HMI.

4.3.2 Capacity analysis and prediction

The Flightplan Processor provides flight data, weather data and runway configuration data to the Capacity Predictor. The Capacity Predictor returns the expected traffic capacity for those conditions. This information can be used in simulations in the Flightplan Processor and can be used for display in the HMI for analysis by the users. The Capacity Predictor retrieves weather and runway planning data from the LVNL DIVPROD database and uses it for traffic capacity predictions. Along with traffic demand flight data this data is also used to for proposing runway configurations that would be appropriate for the traffic demand or proposing runway configurations that would be appropriate for given weather conditions.

Data received from the Capacity Predictor can be cached by the Flightplan Processor before distribution to other components.

4.4 ACC Taskplanning

The ACC Taskplanning, hereafter abbreviated as **ACTAP**, is a functionality with two main objectives:

1. **The dynamic determination of the "line schedule"**, referred to in these requirements as the *Task Schedule*. This will be based on the expected traffic load and workload. As an initial step, it may be chosen to read the line schedule from a file, before this schedule is dynamically determined by ACTAP itself.
2. **The actual task planning**, in which duties are assigned to the positions within the line schedule. The ACTAP system will take into account a large number of parameters, rules, and conditions, which will be described in detail in the user requirements section. In addition to ACTAP-specific parameters, rules, and conditions, information will be imported from the *inPlanning* system regarding the roster that indicates the available duties for a given day. Furthermore, all relevant employee information required for planning purposes will be retrieved from the inPlanning database. Examples include qualifications (for which sector and from which date), whether an employee is in training for a sector, and similar data.

In addition, ACTAP will provide functionality to:

a) Offer a user interface for the line schedule that allows it to be easily modified.
b) Offer a user interface to adjust the task planning as determined by ACTAP.
c) Provide capabilities to manage updates to the task planning. The update process may be automatic, manual, or a hybrid form in which the planning is dynamically updated while taking into account a *freeze horizon* and manual adjustments. This enables simulation capabilities while a frozen planning can be shared with personnel who are assigned to duties.
d) Manage ACTAP configurations in which all parameters and rules are bundled, allowing changes such as seasonal adjustments or revised agreements to be managed efficiently.

The primary task of the Taskplanning application is generating a task schedule for operational air traffic controllers using all constraints configured. The source of the roster is InPlanning which provides the air traffic controllers that are scheduled. (EIF_INPLAN)

Schedules are generated using the Timefold library. This is a commercial library where the task schedule, air traffic controllers with their shifts and constraints are provided to and which returns a generated planning including the penalties for unfulfilled constraints. The constraints are categorized in four categories: hard 0, hard 1, soft 0, soft 1

whereby hard 0 are this that are not possible (i.e. having two different positions at the same time), hard 1 constraint are legally not allowed (i.e. maximum duty time), soft 1 are constraints that are required but can be broken if the task schedule would not be possible otherwise (i.e. air traffic controllers starting first are required to finish work first) and soft 0 constraints are requested but will be violated first (i.e. the time between starting shifts and the first assignment less than x minutes)

The generated task schedules are shown on the HMI where they can be seen and mutated by the supervisors (i.e. pinning assignment, closing or opening shifts or tasks) and submitted for solving. For operational air traffic controllers a tablet is provided with a view of the task schedule.

4.5 Interface design

4.5.1 Interface identification and diagrams

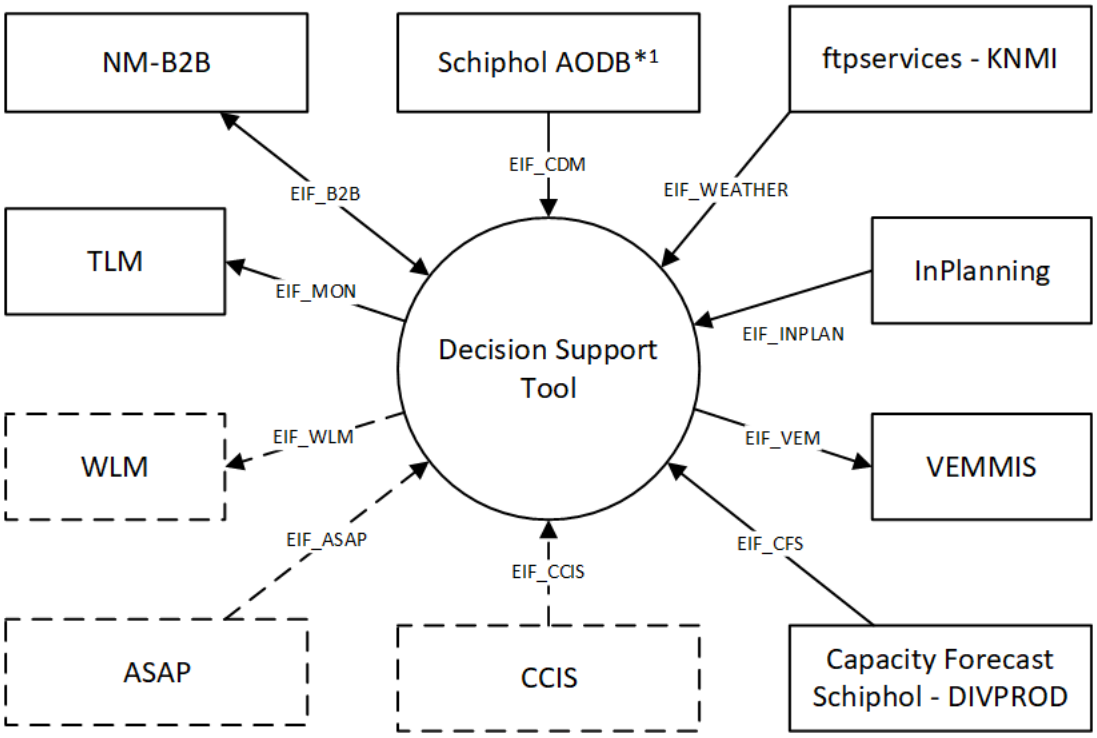


Figure 4 - Interface Identification

- **EIF_ASAP – ASAP Arrival Manager**
  [Future Operating Capability]
  Interface to the ASAP Arrival Manager to be used for receiving enhanced Schiphol inbound flight data.

- **EIF_B2B – Eurocontrol NM Business-2-Business**
  Interface to Eurocontrol Network Manager using the B2B web services for retrieving flight plan data and receiving and sending (future functionality) of Regulations data.

- **EIF_CCIS – LVNL Closed Circuit Information System**
  [Future Operating Capability]
  CCIS shall be used for retrieving: the actual and future runway configurations, the Schiphol weather

forecast: Schiphol KansVerwachting (SKV) and high resolution wind ('hoogtewinden') data.

- **EIF_CDM – Schiphol AODB CDM System**
  The interface to the Schiphol Operational Airport Database (AODB) System will provide the DST with CDM-data with detailed Schiphol flight departure data.

- **EIF_CFS – LVNL DIVPROD**
  This interface consists of a connection to the LVNL DIVPROD database for retrieval of the Capacity Forecast Schiphol data that consists of runway planning and flight capacity data.

- **EIF_MON – LVNL Top-Level Monitoring**
  [Future Operating Capability]
  This interface reports the status of the DST System to the LVNL Top-level Monitoring System.

- **EIF_WEATHER – KNMI ftpservices**
  The interface for retrieval of High Resolution Wind forecast and Schiphol KansVerwachting (SKV) data from KNMI.

- **EIF_WLM – LVNL WorkLoad Model**
  [Future Operating Capability]
  An interface to provide CDM-data to the WLM System.
  Remark: This interface will be implemented in a component that may be deployed stand-alone from the DST System.

- **EIF_VEM – LVNL VEMMIS**
  This interface consists of a connection to the external LVNL VEMMIS database and is used to store ops-pan data generated in DST for further analysis by LVNL.

- **EIF_PLANNING – InPlanning API**
  This interface consists of a connection to the external InPlanning application of LVNL. This application is used to schedule air traffic controllers in shifts and to keep track of their qualifications.

4.5.2 Internal Interface design

SSDD-010 Use AMQP 0-9-1-protocol for publish/subscribe communication

AMQP version 0-9-1 shall be used for publish/subscribe (also known al message-push) communication between system components by means of the RabbitMQ message broker.

Rationale:

RabbitMQ is an open-source, robuust, performant, well documented and mature message broker with AMQP 0-9-1 support. It is well supported and has interfacing libraries for all commonly used programming languages and all of the languages that will be used by DST (Java and Python).

SSDD-011 Use HTTP-protocol for request/response communication

HTTP shall be used for request/response communication between system components.

Rationale:

HTTP is an open standard and well supported by the languages and frameworks employed in building the DST-system.

Note:

In exceptional cases AMQP 0-9-1 may also be used for request/response communication. For instance when HTTP-communication is not possible of feasible.

SSDD-012 Use JSON-format for data transfer between components

JSON shall be used for the transfer of data between system components.

Rationale:

JSON is an open standard and well supported by the languages and frameworks employed in building the DST-system.

4.5.3 EIF_B2B – Network Manager B2B Web Services

Interface to Eurocontrol Network Manager using the B2B web services for retrieving flight plan data and receiving and sending (future functionality) of Regulations data.

The data received from B2B over a mostly publish-subscribe interface and is transformed to the DST internal representations of the entities before sending to RabbitMQ for distribution.

The interface is specified in the Eurocontrol document: NOP/B2B Reference Manuals[B2B_REF].

4.5.4 EIF_CDM – Schiphol AODB CMD Interface

Reception of CDM data from the Schiphol AODB system.

The data received from Schiphol AODB over a publish-subscribe interface and is transformed to the DST internal representations of the entities before sending to RabbitMQ for distribution.

The interface is specified in: [ICD_CDM_AODB] and

Note: Until the interface with the Schiphol AODB becomes available, a temporary component ‘Schiphol CDM Interface’ retrieves the data using a request-response interface and publishes for consumption by the CDM Receiver. This temporary component has implemented the Schiphol AODB side of the EIF_CDM interface to ensure no (major) changes are needed to the DST CDM Receiver component when the Schiphol AODB system is connected to DST.

4.5.5 EIF_CFS– LVNL DIVPROD

The EIF_CFS interface is used for retrieving the Capacity Forecast Schiphol with the runway configuration planning. In the Testing and Education environments these interfaces are used for retrieval of historic data.

The interface is specified in: [SDD_DST_CAP][ICD_CDM_AODB]

This interface consists of database connections to the LVNL DIVPROD Oracle database.

These interfaces are to be replaced by the EIF-CCIS interface in future development.

4.5.6 EIF_MON – LVNL Top-Level Monitoring

The interface for sending event messages to the Top-level Monitoring System (TLM). The TLM is to be kept up to date of the state of the DST-System as a whole.

The Interface is specified in: [ICD_EIF_IMC]

4.5.7 EIF_WEATHER – KNMI ftpservices

The interface for the input of high resolution 3D wind (‘hoogtewinden’) forecast and Schiphol Kansverwachting (SKV) data for the LVNL airspace.

The data is retrieved from a KNMI FTP-server, processed and stored in the Flightplan Store.

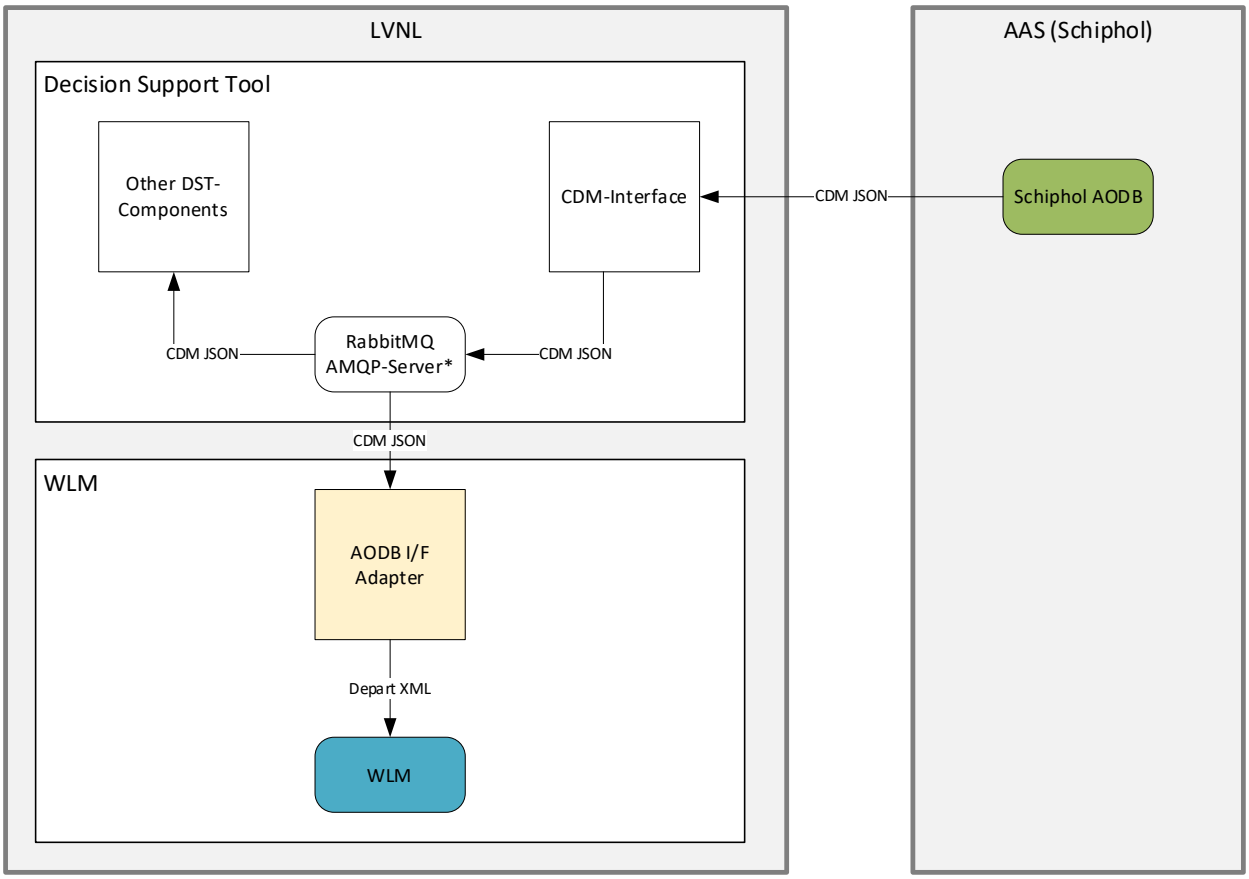
The Interface is specified in: [SDD_DST_HIRES]

Note: Currently the data is retrieved from the KNMI FTP-Pro system. The preferred interface for future development is EIF_CCIS.

4.5.8 EIF_WLM – WLM AODB Interface Adaptor [FOC]

This interface provides CDM-data in JSON format as received from Schiphol AODB to WLM.

The data received by the CDM-Interface is forwarded through the AMQP-Server to the AODB Interface Adapter of WLM. This adapter transforms the data to the XML-format that WLM currently receives from AMADEUS/Tower System.



5. DST Technology Overview

This chapter describes the technological architecture of DST and the associated Taskplanning Tool. It provides an overview of the technologies and components used, divided into the front-end, back-end, database, infrastructure, and DevOps pipeline. For each section, the programming languages, frameworks, and libraries in use are described, along with the role they play within the overall system. This overview forms the basis for understanding the architectural choices and the interdependencies between the different components of the applications.

5.1 Front-end

5.1.1 Languages

• **TypeScript**: A superset of JavaScript with support for objects, static typing, and additional language features. This is the standard language used by Angular and is used to build the front-end of DST and the Planning Tool.
• **HTML**: The standard markup language for structuring web pages.
• **SCSS**: A CSS preprocessor with variables and reusable styles, enabling the use of variables, mixins, and reusable themes, resulting in improved maintainability.

5.1.2 Frameworks

• **Angular**: A front-end framework for scalable single-page web applications that uses components to support reusable parts of the application.

5.1.3 Paid front-end library

• **amCharts 5**: An external library for interactive data visualizations, used to create the charts in DST.

5.2 Back-end

5.2.1 Languages

• **Java**: The primary language in which the back-end components are built. Java 21 (DST) and Java 25 (Planning Tool) are currently in use.
• **Python (DST only)**: Used for capacity calculations, machine learning, and scripting within DST.

5.2.2 Frameworks

• **Spring**: An open-source framework built on top of Java that provides support for dependency injection and modularization. Spring 7 is used.

• **Spring Boot**: An extension of Spring that enables faster startup of Spring applications and embeds server components within the application, simplifying development. Spring Boot 4 is used.

5.2.3 Paid library

• **TimeFold**: An external commercial library that provides the planning logic within the Planning Tool and is responsible for calculating and evaluating schedules.

5.3 Database

• **PostgreSQL**: An open-source relational database used for storing data in DST and the Planning Tool. Database access from the Java applications is handled via Spring JDBC and jOOQ.
• **Liquibase**: A migration tool for database version management within DST and the Planning Tool. Multiple schemas are used within DST to keep data separated.

5.4 Infrastructure

• **Linux (RHEL)**: Enterprise Linux operating system installed on the servers. Installation and configuration are performed by LVNL.
• **Podman (Planning Tool)**: A daemonless container engine that can be used as an alternative to Docker.
• **Docker (DST)**: A container engine that enables applications to run in a sandboxed environment without dependencies on the host system. DST uses 12 Docker containers that share a Docker network for internal communication.
• **Nginx**: A web and reverse proxy server that provides a layer between external users and internal applications and allows SSL to be configured centrally.
• **Zabbix**: A monitoring tool for systems and applications, used to monitor DST and the Planning Tool. It uses actuators and trapper items that are sent to the Zabbix server via a Zabbix Proxy/Agent.
• **RabbitMQ (DST)**: A message broker that enables asynchronous communication between DST applications via AMQP messages. The message payloads are in JSON format.

5.5 DevOps

• **Maven**: A build and dependency management tool used for building and testing applications.
• **GitLab**: A platform for source code version control.
• **Jenkins**: An automation tool used for build and deployment processes.
• **SonarQube**: A static code analysis tool that checks whether the code meets defined quality standards.

6. Technical Size

Below is an indication of the technical size of the current DST application:

- Server/back-end: 56K lines of code
- Frontend: 38K lines of code
- Capacity-api: 11K lines of code
- Machine Learning module/Load predictor: 4.5K lines of code
- Planningtool: 28K lines of code

Terms and Abbreviations

This chapter contains terms and abbreviations specific to the DST system. Terms and abbreviations which are considered common knowledge for stakeholders are not included.
Many of these terms can be found at these sites:

- https://ext.eurocontrol.int/lexicon/index.php/Main_Page
- https://www.eurocontrol.int/airial/
- https://www.skybrary.aero/index.php

The following terms appear frequently in this document:

Term / Abbreviation	Description
ASAP	The Arrival Manager system used by Schiphol approach air traffic controllers
ATA	Actual Time of Arrival, the time an aircraft landed on the runway
ATO	Actual Time Over, the time a Route Point was overflown or Airspace was entered.
ATFCM	Air Traffic Flow and Capacity Management is the regulation of air traffic in order to avoid exceeding airport or air traffic control capacity in handling traffic, and to ensure that available capacity is used efficiently.
B2B	Business 2 Business web services from Eurocontrol Network Manager providing among others flight and regulation data.
CCIS	Information system of LVNL for air traffic controllers, provides data on : weather, runway configuration planning and more.
CDM	Collaborative Decision Making. In the context of DST: a system which provides more accurate flight departure planning information.
CFS	Capacity Forecast Schiphol, the forecast of the runway configuration and capacity of Schiphol airport
DIVPROD	LVNL database containing among others Capacity Forecast Schiphol (CFS) data.
ETO	Estimated Time Over, the time an aircraft is estimated/predicted to overfly a Route Point or to enter an Airspace.
NM	Eurocontrol Network Manager, the air traffic management organisation responsible for flight plan distribution and demand/capacity balancing for Europe.
Reduced Coordination	Circumstances where Sectors usually controlled by military air traffic controllers are controlled by civilian air traffic controllers of LVNL. Under Reduced Coordination the Workload of LVNL civil air traffic controllers is lower because coordination with military Sectors is no longer required.
SKV	Schiphol KansVerwachting: weather forecast data for Schiphol airport containing runway visibility and wind (direction and speed) data.
VEMMIS	The VEMMIS database is the central LVNL database (Oracle) that stores operational air traffic data for monitoring, analysis, and decision-making within LVNL. The most important sources are AAA (Main ATC System), TWR-system and CCIS.