Technical Note

Hopp – requirements

This document contains a brief overview of Hopp with the aim of providing an understanding of

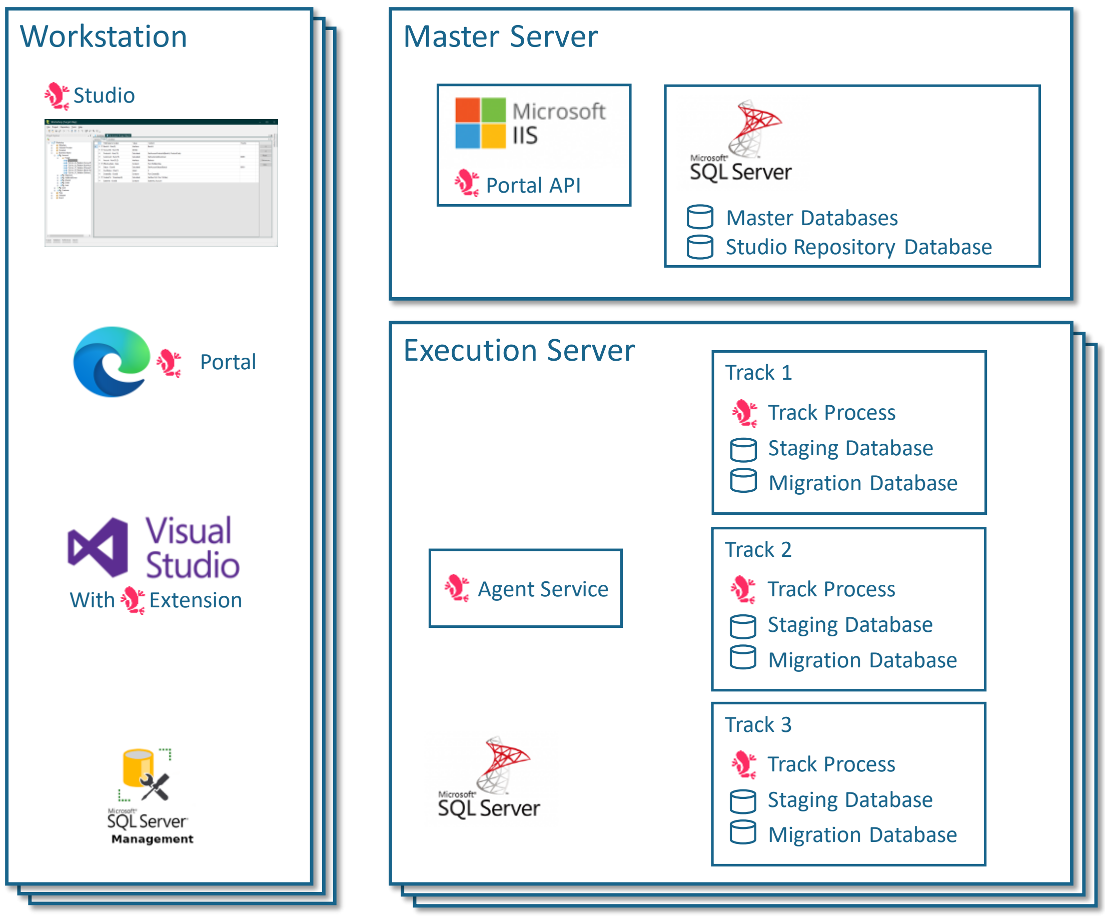
- *the Hopp components*
- *the Hopp architecture and technical requirements*

Hopp Components

The Hopp components, Studio, Engine, Runtime and Portal all collaborate to support the migration process all the way from the mapping to the execution and finally to exposing the migration results.

The diagram below shows the components as they are installed in an elaborate and full-scale migration setup in order fully illustrate which components are installed where.

Less elaborate setups are also possible and described further on.



Workstations

- Minimum Windows 10
- 64 bit
- Minimum 8GB of RAM

- Reasonable free disk space, minimum 50GB recommended

Studio

Studio is a Windows application installed on individual Workstations. Studio requires Crystal Reports Runtime and Sql Server LocalDB (both freeware). The Studio bootstrap installer includes and will automatically install these if needed.

Studio is used to produce the Hopp Source- and Target Maps.

Individual users collaborate by checking out and committing all changes to a central Sql Server repository database (on the Master Server).

Studio connects directly to the Repository Database:

- The database (normally port 1433) must be accessible from the Workstation
- The workstation user must be granted the db-owner role to the Repository database
  - Windows or Sql Server Authentication
  - In a corporate intranet setup, the access would normally be granted as Windows Authentication for an AD group

Backed by the repository, Studio offers significant advantages:

- Full history of all changes (commits) made to the mapping
- Rollback of commits
- Ability to retrieve the total mapping at any previous point in time
- Easy compare of different versions of single items
- Freeze of a version of a given map in order to protect a given project close to go-live from changes in other, less advanced projects
- Labeling for easy identification of a given point in time

The Maps are published from Studio to the Engine Framework.

Engine Framework

The Engine Framework contains a Code Generator (a console application) and a set of base libraries.

The Generator reads the maps published from Studio and generates c# code.

The generated code resides in Visual Studio c# class library projects that will compile to the Source- and Target Engine respectively.

Part of the mapping in Studio will be so-called Manual Rules. These exists as stubs in the generated code. The code must be manually supplemented with code to implement these rules.

Visual Studio is connected to any source control, for instance Azure DevOps Git. Normally, the generated files are exempted from source control (hopp can provide a suitable .gitignore file).

The engine class libraries in Visual Studio reference base libraries that are also installed on the Work Station as part of the Engine Framework.

Microsoft Visual Studio

Visual Studio is used to manage and build the generated engines. Any edition of Visual Studio will do – including Community Edition (free).

Hopp comes with a Visual Studio extension that facilitates:

- Generation of engine code
- Deployment of the engines to the Director Runtime
- Local debug to troubleshoot manual rule implementations

The Visual Studio extension supports Visual Studio 2022 and later versions.

Microsoft Sql Server Management Studio

Not strictly required by Hopp. However, when working with Source Maps in Hopp Studio, it is beneficial to be able to query and analyze the source data in the Hopp Staging Database.

Any Web Browser

A web browser is needed in order to access the Hopp Portal. All browsers – except Internet Explorer – are supported.

The Portal is a Blazor Single Page Application and as such runs especially well on Microsoft Edge.

Master Server

The Master Server is not heavily loaded and does not have to be very powerful.

Requirements:

- Windows Server 2016 and up
- 64 bit
- Sql Server 2016 and up – any licensed edition

Recommendations:

- 32GB RAM and up
- 8 logical cores and up
- 100GB free space and up

Software

IIS must be activated on the Master Server:

- The Master Server hosts the Portal API under Internet Information Services
- The Portal requires https. A trusted certificate must be installed on IIS
  - A self-signed certificate can optionally be used but will result in insecure connection warnings in most browsers
- A certificate to sign authorization tokens must be created with an RSA private key and installed on the server
- The DotNet SDK and DotNet Hosting Bundle must be installed according to the current DotNet version used by Hopp

Sql Server must be installed on the Master Server:

- Master Database for the Director to manage Migration Servers, Tracks and Engines
- Portal Database to host the Portal and also to manage Projects, Users and Authorizations

The Master Server also holds file directories to contain:

- the Director Runtime libraries
- the deployed Source- and Target Engines

Migration Servers

A Migration Server is where Hopp actually executes the generated engines in order to migrate the data. It is the Migration Server that carries the heavy load and powerful hardware means faster migration.

Requirements are:

- Windows Server 2016 and up
- 64 bit
- Sql Server 2016 and up – any licensed edition
- The DotNet SDK according to the current DotNet version used by Hopp

A recommended, initial setup on a Virtual Server could be:

- 64GB RAM
- 16 logical cores
- 1TB free space

Further on, performance and elapse time will benefit from a dedicated physical server:

- High number of cores
- More RAM
- Faster disks
    - Directly connected to physical box to avoid read/write over network
    - SSD or RAM disks – for instance Fusion IO

The Hopp Agent

The agent is a Windows Service running on the Migration Server. The sole job of the agent is to start Tracks on the Migration Server on requests received from the Portal.

The Agent must run under a Service Account that is recognized in Active Directory (this is not required in a single server setup, see below).

All Track Processes spawned by the Agent and all Job processes spawned by each Track Process will run under the same Service Account.

- The Agent must be able to reach the Master Database on the Master Server (Sql Server port)
- The Agent must have read access to a file share on the Master Server (NetBIOS ports)

Tracks

A Migration Server can (but is not required to) run multiple migrations in so-called Tracks. A Track is an isolated instance of the Hopp Runtime executing a given pair of Source- and Target Engines.

Depending on the setup in the Hopp Configuration (in the Master Database), different Tracks can run

- different aspects of a given migration project (for instance one track for historic transactions and another track for everything else for the same project)
- staggered tranches of the same migration project
- completely separate migration projects

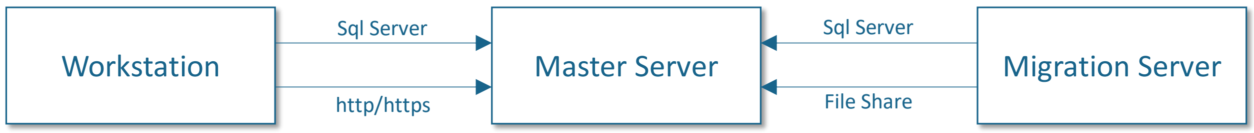
Any one track consists of

- A running Track Process that has loaded the generated Source- and Target Engines for the project
- A Staging Database to hold the source data
- A Migration Database to hold the migration results and events

The sole job of the Track Process is to start jobs requested from the Portal. The Track Process launches each job in a separate process on the server.

Connectivity

The network connectivity requirements that are listed in the sections above are consolidated in this diagram.



Configuration alternatives

The section above describes the full setup with individual workstations, a Master Server and one or more Migration Servers. This is the setup of a fully matured production environment for serious data migrations.

But Hopp can start out with a simpler configuration and evolve from there.

Single Server setup

It is common and indeed advisable to start out in a simpler, single server setup. In this setup a single server acts as both Master Server and Migration Server.

Team members and Business users have the Hopp client components installed on their own machines and use the Hopp client components (mainly Studio) locally.

- The local instance of Studio will communicate with the Repository database on the Master Server via an Sql connection

Single Machine Setup

An even simpler setup is to install all of Hopp on one, single machine. Users will then have to connect to this machine using a remote desktop connection and run the Hopp Workstation components on the machine itself.

A single machine setup is very suitable for a local POC. It is easy to setup and configure, requires no network configuration beyond ensuring RDP access. In the unlikely event the POC is unsuccessful, Hopp is easily removed leaving behind no alterations to network configuration, AD or similar behind.

Authentication and Authorization

Users collaborating on the mapping in Studio must be authorized in Sql Server to access the Repository database

To view data/events and to execute the migration in the Portal, users are authorized in the Portal.

The Portal uses best practice OIDC/OAuth authentication based on JSON Web Tokens. The Portal is configured to authenticate users in one of 2 ways:

- either - Windows Authentication (NTLM/Negotiate). The Portal will ask Windows for the authenticated user that accessing the Portal via a browser. This is the recommended alternative when inside a corporate intranet under AD control
- or - An application login form for user id and password. This is the recommended alternative when the Portal must be accessed from the internet by users unknown to Windows and AD

In the Portal, users are either authorized as internal team members that has access to all projects and tracks – or external users, that are assigned access to specific projects only.

Migration Environments

When using Hopp, a data migration project is different from an IT development project. No migration system is being developed – Hopp is the migration system, already developed and tested.

The migration project is concerned with developing and testing the mapping that defines the engines executing the migration.

POC environment

If the data provided for the POC case are either constructed or anonymized sufficiently, the POC can be established in an existing environment in the hopp infrastructure.

This is by far the easiest approach and there will be zero effort required to establish and potentially tear down a local Hopp environment for the POC.

In many ways, constructed data for a POC is recommended and preferred as this approach helps keep the scope and size of the POC under control.

Real environment

As outlined above, Hopp is executing data migrations in Tracks on Migration Servers.

A Track is migrating data for a specific Project and a Project may (but does not have to) separate the migration over multiple Tracks (for instance historic transactions in one tracks and everything else in another).

This model enables the Director to execute and manage multiple, independent and concurrent migration projects.

Test/Prod and Data

It is the firm recommendation of Hopp Tech that all migration iterations from the very beginning takes place on production data. It will never be possible to construct test data to replicate the diversity and quality problems the migration will encounter in production data.

The Runtime Tracks are agnostic – all is production data and the difference lies solely in where the target data ends up at the end of the day – in a test instance of the target system or in production.

Data protection is production

It follows, that the entire Hopp environment must configured within the security protocols and governance that applies to production data.

Execution is not production

This is data security only. It does not mean that the Hopp Runtime is part of - nor will impact - the daily, business-critical production.

Hopp only needs to receive source data to load in the Staging Database and will then execute the migration in isolation with no impact on production.

When Hopp is done, the target data is unloaded from Hopp and delivered from Hopp to the mechanism responsible for delivering/inserting the migrated data in the target system. This mechanism is outside the purview of Hopp and solely under the responsibility and control of the operations of the target system.

Performance

At mentioned earlier, it is the Migration Server that in fact is executing the migration jobs and elapse times can be reduced by investing in powerful hardware.

These elapse times naturally become critical when approaching Dress Rehearsal and go-live for a project.

But is important to note, that the capability of Hopp to safely execute only partial migration jobs come in very handy in the entire process before that. After all, if a job migrating only 100 Business Objects rejects all of them, it is not necessary to run and wait for the full volume before fixing the problem.