SIMCA®-online Technical Guide

September 5, 2024

Contents

1 Introduction

SIMCA®-online enables real-time multivariate process monitoring and control using SIMCA® models and data from a data source, such as a process historian.

This document provides guidance on planning, setting up, administering, understanding, and troubleshooting a SIMCA-online system.

This guide was updated for SIMCA-online 18, but most topics apply to previous versions as well. A changelog is available at the end of the document.

1.1 Additional documentation

This document is one of a set of related documents, each with different focus and target audience:

Source	What	Where
SIMCA-online web page	Introductory information and downloads	sartorius.com/umetrics-simca-online
SIMCA-online ReadMe and Installation.pdf	Installation and how to get started with SIMCA-online demo data	In the installation zip file
SIMCA-online Implementation Guide	Outlines SIMCA-online functionality, puts it in context with other Umetrics Suite software, describes requirements and best practices for successful deployment, and step-by-step installation instructions.	sartorius.com/umetrics-simca-online
SimApi Guide	Preparing for and performing SimApi installations, including troubleshooting. Also contains technical details on SimApis for developers.	sartorius.com/umetrics-simapi
SimApi User Guides	Documentation for each published SimApi with features, installation instructions, and configuration specifics.	sartorius.com/umetrics-simapi
SIMCA-online Technical Guide	Technical reference for SIMCA-online server installation planning, troubleshooting, and in-depth how SIMCA-online works.	sartorius.com/umetrics-simca-online
SIMCA-online help	Web-based help how to use SIMCA-online and how SIMCA-online works.	In the software itself, and on sartorius.com/umetrics-simca-online
SIMCA-online Web Client Installation Guide	Describes the installation of the SIMCA-online Web Client.	sartorius.com/umetrics-simca-online
Umetrics knowledge base	Searchable database with articles about each released software version, technical articles, and known issues in Umetrics Suite products.	sartorius.com/umetrics-kb
SIMCA help / user guide	How to use desktop SIMCA for creating projects and modelling data.	In SIMCA and on sartorius.com/umetrics-simca
Support web page	How to obtain support technical support.	sartorius.com/umetrics-support

2 Server Planning and IT

This chapter covers what you need to think about when planning a new server. Also see the SIMCA-online Implementation Guide with its requirements and best practices. This chapter also explains backup and restore procedures including disaster recovery, migration to newer SIMCA-online versions, and desktop client installations.

2.1 Computer System Recommendations

The knowledge base article for SIMCA-online 18 lists the system requirements for a SIMCA-online server.

We have no specific recommendations for which computer brand to use for the SIMCA-online server, nor specific recommendations for CPU speed, memory, and hard disk size. In this chapter, we outline the parameters that can be considered when selecting a server so that you can see some of the important factors. We do recommend that customers use server hardware and Windows version they are used to and manage the SIMCA-online server as any other server computer in their organization.

Running SIMCA-online server on virtual machines is supported.

We recommend you always install Windows security updates as soon as they are available. There have been no compatibility issues resulting from such updates. In general we recommend you keep all software used with SIMCA-online up to date.

Make sure the system clocks of all computers and data acquisition devices in your system are in sync (see 6.8 below).

2.1.1 One server computer per online server

We recommend that you run a single SIMCA-online server version per server computer. Multiple servers on the same computer compete for the same resources, affecting performance.

Another complication of running more than one online server on the same computer is interpreting log files if SimApis share the same configuration files and log files. To avoid this problem, you can use different SimApi instance names.

2.1.2 64-bit SIMCA-online server

SIMCA-online is only available in a 64-bit version starting with SIMCA-online 17. There is no longer a 32-bit version.

2.1.3 SIMCA-online server is a service – set it to start automatically

The SIMCA-online server is a Windows service. It can be started from the SIMCA-online Server Monitor icon, but also with the Services control panel like any other service.

For production use we recommend that you configure the SIMCA-online server service in Windows Services to start automatically when Windows starts by setting the Startup type **Automatic (Delayed start)**. Otherwise, the SIMCA-online server will require manual starting by an administrator.

2.1.4 Service account

The SIMCA-online server service runs as the service account **LocalSystem** by default. This effectively gives the service local administrator rights. This is not necessary for most installations. Instead **NetworkService** can be used ("has minimum privileges on the local computer and acts as the computer on the network.") or **LocalService** ("has minimum privileges on the local computer and presents anonymous credentials on the network."). Read more on service accounts at http://msdn.microsoft.com/en-us/library/windows/desktop/ms686005(v=vs.85).aspx

A general security best practice is to try running the services as **LocalService** in your environment. See if it works. If not, try another account with more privileges.

An Active Directory user account can also be used as the service account. This can be useful if the data source allows access to only specific user accounts. Note that some SimApis allow specifying credentials in the settings which means that the SIMCA-online service account does not affect data access. Learn more in the user guide of the specific SimApi.

If the SIMCA-online server runs on an Active Directory domain member server, user authentication and single sign-on will work without changing the service account to an Active Directory account.

These requirements apply to the service account:

- It must have the 'Log on as a Service' permission. This is automatically granted if you change the service account in the Services control panel in Windows.
- It must have file system read/write access to the %ProgramData%\Umetrics\SIMCA-online\[version] folder and the SIMCA-online Database Directory. This directory is specified in SIMCA-online Server Options.
- If the Web Server component is used, the user account also needs to be allowed to use HTTP or HTTPS on the port specified in SIMCA-online Server Options. See the topic 'Web Server, Web API and Web Client' in the SIMCA-online help to learn more.

2.1.5 Add a service dependency to start services in the correct order

If the SIMCA-online service must start **after** another service is started (for example the service of your data source), you can add a dependency in the service, so that Windows starts the services in the correct order. You can do this using the SC command line tool. Read more in the knowledge base article Server service doesn't start even if configured to start automatically (Q606) (sartorius.com).

2.2 Processor (CPU)

The SIMCA-online server is a multithreaded application. It splits work between different threads that run on the available CPU cores on the server.

The predictions (execution of project configurations) are multi-threaded and use as many cores as needed to ensure good throughput and responsiveness.

A fast CPU is beneficial for fast predictions, as are more cores. In a virtualized environment, the number of cores can be optimized to the actual need by monitoring CPU utilization in the virtual machine.

2.3 Disks and diskspace

SIMCA-online uses disks for storage of data in its Database directory. This folder is specified in the SIMCA-online Server Options application.

By using a Solid State (SSD) disk for the Database directory, you ensure optimal performance.

The disk space requirements for a project configuration depend on various things which are mainly project/model related; the two most important factors are execution interval and number of variables.

Here are some rough calculations.

- There are about 100,000,000 seconds in three years. Each variable for an observation takes about 10 bytes of storage.
- If a project configuration is running for three years, at the execution interval of 1 second and using a single variable (although highly unlikely), it will generate 1 GB of data.
- If, under the same circumstances, you store 100 variables at the execution interval of 10 seconds you'll get about 10 GB of data after three years.

This is the storage requirements for the raw process data and predicted data for a single unit and a single project. If projects and units are run in parallel, more data will be stored.

In addition, meta-data such as batch start time, batch stop time, prediction times, phase start, phase stop, alarms, notes, audit trails and similar are stored, and this also requires space.

2.4 System memory (RAM)

How much memory does the SIMCA-online server need for a project?

It is difficult to estimate the memory usage for a project because of the complexity and variation between projects, but as a rule of thumb, the memory usage is about two or three times the size of the SIMCA USP-file.

- For example, 20 projects of 50 MB each result in that at minimum 50*3*20 = 3000 MB or about 3 GB RAM is required.

2.5 Network

SIMCA-online server, data source and clients are separated by the network. This makes the system dependent on a high-quality network: if bandwidth is low, latency is high, or there are other communications problems, it will affect SIMCA-online.

We recommend a high bandwidth, low latency network.

The network applies both between server and its data source, and between the server and the SIMCA-online clients.

2.5.1 Network between server and data source

The server obtains its data from the data sources. If this network is slow the project execution on the server will be affected. It will not be able to execute as many project configurations in parallel as it would on a faster network because it will spend more time waiting for data source requests.

Server log files will show Debug-level entries like the following, that shows how long the data request took. In these examples both calls take more than 4 seconds which is very long:

[2019-10-25 14:29:31.134+02:00] [TID:23264] [Debug] DataSourceManager::ReadCurrentData()]

Read current data for 10 variables, return time 2019-10-25 14:29:31 (1572006571), elapsed time 4017 ms.

...

[2019-10-25 14:29:35.142+02:00] [TID:23264] [Debug] DataSourceManager::ReadHistoricalData()]

Read historical data for 10 variables, 6 observations, starting at 2019-10-25 14:29:09 (1572006549), interval 3 s, elapsed time 4006 ms.

2.5.2 Network between server and clients

The SIMCA-online desktop client works best if it is close to the SIMCA-online server in the network. A slow network connection between client and server will result in a non-responsive user interface.

To determine if the network is causing the problem, try running the client directly on the server computer. If the client works well there, while slow at the remote client computer, then the network is the cause of the problems.

For large distances between client and server, desktop virtualization solutions such as Remote Desktop or application virtualization can be used to run the client closer to the server to mitigate the problem.

2.5.3 Network ports and firewalls

A SIMCA-online server uses a single TCP listener port for its communications with SIMCA-online desktop clients. The Listener port can be changed in the SIMCA-online Server Options utility. The default port is 2371.

The SIMCA-online exposes an optional web API. It uses the TCP port configured in SIMCA-online Server Options. The web API is used by the SIMCA-online Web Client.

The communications between the server and its data source also uses the network. This varies between SimApi and data source. Refer to the individual user guide for the SimApi to learn more.

The relevant ports must be opened in a firewall between the communications endpoints.

2.6 Data source and SimApi

A commercial data source for process systems such as AVEVA PI System (formerly OSIsoft) is well suited for SIMCA-online because it provides the necessary logic and data structures to deal with process data and batches. It is also optimized for performance for the types of data requests that SIMCA-online makes.

A SimApi for the data source is required for SIMCA-online to be able to access data. The SimApi is a software layer that translates the requests from SIMCA-online to a protocol that the data source understands.

There are SimApis available for various data sources such as ODBC for databases such as SQL server, AVEVA PI System, and OPC UA. We provide the complete SimApi specification for developers wishing to implement a SimApi.

Learn more:

- The SIMCA-online Implementation Guide lists requirements and best practices relating to data sources and SimApis for a successful SIMCA-online deployment.
- SimApis are available for download at https://www.sartorius.com/umetrics-simapi. This page also shows available features in SimApis.
- The SimApi Guide shows how to install SimApis and many technical details.

2.7 SIMCA-online backup and restore

It is important to perform regular backups of a SIMCA-online server. A backup is performed by stopping the SIMCA-online server service, performing the backup, and then restarting the service.

This chapter describes in detail what to backup, how to do it, and how to restore data. This chapter is also useful for administrators who want to know where data is located.

Note that you cannot backup and restore individual project configurations in SIMCA-online.

2.7.1 Data locations - what to back up

- The Database directory of the SIMCA-online server. It stores data for project configurations, users, audit trails, and server log files. The database directory can be seen and changed in the SIMCA-online Server Options utility.
- The folder %programdata%\Umetrics\SIMCA-online Server\[version] (this is the parent folder of the default location for the Database directory). This folder contains the server configuration file SIMCAonlineserver.ini, the and the files containing the license.

In addition, the SimApi configuration settings should be backed up and documented so that you can redo all settings in the event of a disaster:

- SimApi settings folder: SimApis typically save their settings and log files in %programdata%\Umetrics\SimApi. Refer to the user guide or for the SimApi you are using, and the SimApi Guide, to learn exactly what you need to backup for a SimApi.

You also need copies of the SIMCA-online and SimApi installation files and any runtime software or drivers that the SimApis need to be able to do a disaster recovery or migration of SIMCA-online to a new server (see below).

2.7.2 Basic backup procedure

1. Stop the server service
2. Backup the database directory with all its contents
3. Backup the other contents of the parent folder of the database directory (see above for details)
4. Backup the SimApi settings
5. Restart the server service

2.7.3 Basic restore procedure

1. Stop the server service

2. On the server computer, empty the database directory
3. Restore the database directory from the backup
4. Restore the %programdata%\Umetrics\SIMCA-online Server\[version] folder (not needed if you just want to restore the Database directory)
5. Restore the SimApi settings (not needed if you just want to restore the Database directory)
6. Restart the server service

When starting the SIMCA-online service, the server resumes execution for configurations and catches up as needed following the settings in the configurations. See 6.3.

2.7.4 Disaster recovery - a complete reinstallation and restore of data

The above procedure is not enough if you want to do the restore as part of disaster recovery where you don't have SIMCA-online already installed.

For a complete disaster recovery, follow these steps:

1. Install the same SIMCA-online server version on the new computer, and all SimApis used on the server.
2. Restore the SimApi Settings or reconfigure the SimApi to connect to the data source. If reconfiguring, keep the same instance names of the SimApis so that SIMCA-online will be able to find nodes and tags using its old settings.
3. Restore database directory and the server configuration file as described above.
4. The backed up SIMCA-online license may not work for the new server computer if it was locked to the previous hardware. If so, request a new license in the SIMCA-online Server Options utility and install the new license before proceeding to try to start the service (otherwise your SimApis won't run and you will not get any data from the data source).

2.7.5 Backups with storage level volume snapshots

Sartorius has no direct experience making live backups of a running SIMCA-online server using technologies like Windows volume shadow copy snapshots, virtual machine snapshots, or similar technologies.

However, we have several customers that use such backups of SIMCA-online in their systems.

We recommend you try your backup technology on a test server, and make sure you can successfully restore to an earlier point in time.

2.8 Moving a SIMCA-online server to a new computer

If you want to move a SIMCA-online server installation from one computer to another you must install SIMCA-online including all SimApis on the target computer, and then migrate the database to it. This process is the same as for disaster recovery and is described in 2.7.4.

Once migrated, ask clients to log in to the new server's name instead.

2.9 Migrating to a new SIMCA-online server version

SIMCA-online server can be automatically migrated from an earlier server version. Learn more in the help topic Import data from previous SIMCA-online server versions.

Note: the new server will upgrade the database format to the new version, making it impossible to use the Database directory with the previous version.

2.10 Keeping an old server as an archive server

Sometimes you want to keep an old server to be able to look at historical data. Here are a few things to consider:

- Turn off execution for all project configurations on the archive server.

- Remove permissions for users of SIMCA-online so that they can only look at things, but do not have administrative rights (so that they cannot make any changes).
- Consider disabling SimApis in SIMCA-online Server Options so that you don't need to maintain a working connection to a data source to be able to start the SIMCA-online server.
- If the server license has an expiration date, talk to your sales representative about obtaining an updated license for the archive SIMCA-online server. Note that if you are not using any SimApis on the old server, you do not need a license. The server will work with all features enabled except connecting to data sources.

2.11 Installer customization and desktop client deployment

The SIMCA-online installation file (EXE-file) is used to install both the server and client of a SIMCA-online system. It supports customization from the command line. Read more about this, including how to perform a silent installation of the desktop client, in the knowledge base article SIMCA-online installer customization and client deployment (Q808) (sartorius.com).

Tips:

- you can create SIMCA-online client links (URIs) and distribute to the users, so that clients automatically connect to the right server (you can also open project configurations this way)
- learn where client options, plot settings, and other settings are saved in the client in the help topic, Client options - technical details. It also indicates how you can share settings between users.

3 Troubleshooting

In this chapter there are high level topics relating to troubleshooting a SIMCA-online system.

3.1 SIMCA-online technical support

Technical support for SIMCA-online is provided by the team of software developers responsible for SIMCA-online.

To solve your problem, we typically need detailed information such as version of the software, what you are trying to do, error messages, copies of server- and SimApi log files. If you need to send us large files, we can provide access to our file sharing service.

The quickest way to solve an issue is often to let us connect to your installation and look at it, using remote support software that we provide or tools that you prefer.

Learn how to contact us at sartorius.com/umetrics-support

3.2 Use SIMCA-online server logging and audit trails for finding problems

A SIMCA-online system is a client / server application with connections to external data sources.

Diagnosing problems and understanding the server can be done by analyzing server log files which contain time-stamped entries of what the server is doing. Learn more in the help topic 'Using SIMCA-online server logging'.

Throughout this document we'll refer to the server log and mention different log entries and when they are written to the log file.

The **Audit Trails** keep track of user-initiated changes made in SIMCA-online to the server or its project configurations. They are easy-to-use tools for looking up things like when batches or phases started and ended and when alarms were triggered. Learn more in help.

3.3 Server performance issues, troubleshooting and mitigation

What happens if you have too many projects, too large projects, too many variables or a too slow SimApi?

In such cases the server cannot keep up with reading data and executing models. This will show as unresponsive clients, lack of updates in plots, high CPU load, high disk activity on the server, trigger of 'Server execution falling behind' event (notification can be set up), and in general poor performance.

If the server computer has an overloaded CPU, the server will alert users running the SIMCA-online client with a message "The processor is working at a high load". This message is also logged to the server log.

The server log also contains timing information. Compare the elapsed times displayed with earlier logs to compare. Also see 2.5.1.

If memory were to run out on the server it can result in the server crashing, execution being stopped for a project configuration, or a range of various error messages. Specifically, "**Out of memory exception"** may be logged in the server log.

One way to track resource usage is to use the program Process Explorer from Windows Sysinternals to monitor the SIMCA-online server. It will show you details and graphs of memory usage and CPU utilization in greater detail than Windows Task Manager. Standard IT tools used by your company should also be used to monitor the server.

Here are a few things you can do to increase performance:

- Verify that other services on the server computer aren't competing with SIMCA-online for resources.

- Use a fast SSD for the Database directory for best performance when reading recent batches from the internal database and all other disk operations.
- Turn off execution for project configurations you do not need.
- Delete and purge old projects and configurations if possible. A deleted, but unpurged project is still read by the server when initializing and uses some memory.
- Reduce project usp-file size before uploading to SIMCA-online by using 'Save As' in desktop SIMCA. It is more efficient in reducing file size than 'Compact project file'. The size of the usp-file and workset batches affects project load time and performance when retrieving and plotting them. The length (number of observations) of recent batches has more effect on performance than the number of recent batches.
- Reduce the logging levels for the server- and SimApi logs because the act of logging can affect performance negatively.
- Close clients, configurations and plots when not needed, as many open project configurations and plots in clients result in a higher load on the server.
- Reduce recent time: The project configurations have settings that affect the overall performance: on the Time Range page you can reduce the time that is considered "recent" so that fewer batches and less data are displayed in plots by default.
- Optimize using Plot options: Each SIMCA-online client has Plot options that can be used to reduce the data displayed in the plots (this includes options for showing recent batches and workset batches in plots).
- Finally, you can consider splitting the load by having multiple server computers running SIMCA-online server.

3.4 Troubleshooting slow desktop clients

When SIMCA-online clients are slow to open workspaces, project configurations, plots it can have many causes. Here are some guidelines:

- Make sure client computers are connected to the server with a high bandwidth, low latency network (see 2.5.2).
- If a client is not used, consider logging out of the server as it reduces the load on the server.
- Close project configurations, and plot windows that are not needed any more.
- If a workspace is slow to open, make sure the workspace does not contain unnecessarily many project configurations and open plots. Close unneeded windows and overwrite the workspace again.
- If a client is slow to open after logging in, it can be caused by slow loading of the session-workspace that is automatically saved with all open windows when you last logged out from the server. To fix this either turn off the client option for Session workspaces, or make sure you close windows you don't need.
- Do not configure a too long Recent time range for batch project configuration that includes many batches. Data for all recent batches are loaded in the client which can be time consuming. Tip: In SIMCA-online 16 and later you can use a short Recent time range, and still open arbitrarily old reference batches in Production overview that can are displayed in all plots.

3.5 Using memory dump files for diagnosing crashes or non-responsive programs

If an Umetrics Suite product crashes or is unresponsive (appears to have hung), a memory dump of the program can be used to diagnose the problem. Read more, including how to create a dump file, in the knowledge base article Creating memory dump files for Umetrics support to debug crashed or non-responsive programs or services (Q809) (sartorius.com).

3.6 Troubleshooting project execution

When you experience symptoms for a specific project configuration such as lack of updates in plots and lists (no new data shows up), the wrong data shows up or that phases/data are missing, then you need to troubleshoot that project configuration.

Refer to the built-in help for SIMCA-online with its step-by-step instructions for project execution troubleshooting. Search for Troubleshoot Project Execution to find it.

4 Configuring Projects in SIMCA-online

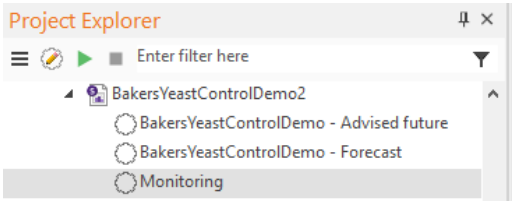
This chapter explains key concepts in the context of how you set up a SIMCA project to work in SIMCA-online with your data source. These terms are also important for the understanding of **project execution** which is described in the following chapters.

Also refer to SIMCA-online's help if you want to look up other terms not listed here. See the help topic **Project Explorer** for step-by-step instructions how to configure projects for use in SIMCA-online.

4.1 Concepts

4.1.1 Project Configuration

A **Project Configuration** in SIMCA-online refers to a SIMCA project (USP-file) that is configured to read data from tags in a data source, and to execute on the SIMCA-online server.



As the screenshot shows, you can have many configurations for one SIMCA project in SIMCA-online. The three configurations in this example have different settings and purpose but share the same project on the server.

Technically, SIMCA-online always executes (predicts) project configurations, and never projects (USPs) or just models.

4.1.2 Tags and nodes

A **tag** is an identifier of a column or "variable" in a data source.

A **node** is a container of tags. A node can also contain other nodes, similarly to how a file system has folders in folders.

Like in a file system, the node and tag names can be combined to a full path that uniquely identifies a particular tag. The tag paths are used in SIMCA-online or SIMCA when selecting tags to use. A tag path starts with a SimApi instance name followed by the node-structure, and ending with the tag name, each item separated with a colon (:). For example ":ODBCSQLServer:Node:SensorTag1".

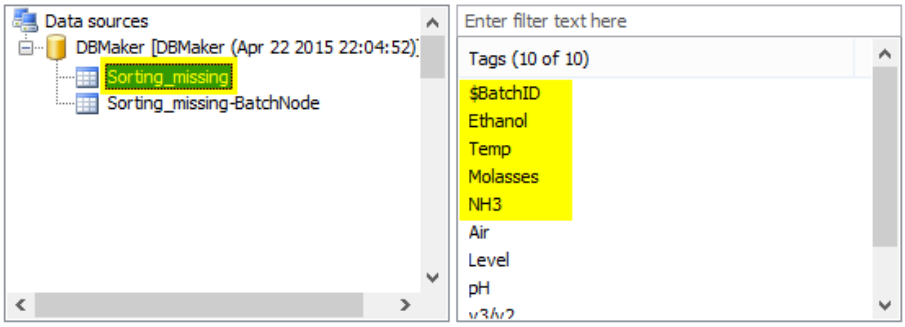


Figure 1. How a node looks in SIMCA-online. You see the node name Sorting_missing to the left, and the tags in that node listed to the right. The full tag path to the Ethanol tag in this screenshot is :DBMaker:Sorting_missing:Ethanol.

In SIMCA-online, tags are tied to variables in a model of a SIMCA project in the configuration wizard. For a model to be used in SIMCA-online, one tag in the data source is required for each model variable. Using tags, the SIMCA-online server can read values from the data source through the SimApi. Tags are also used in other aspects of the configuration, such as the batch id tag for a unit, in phase execution conditions and sleep conditions, and for write-back.

Sorting_missing [Continuous Node]

	1	2	3	4	5	6	7
1	Observation	$BatchID	Ethanol	Temp	Molasses	NH3	
5484	2015-05-07 09:22:50	gb_2015-05-07 09:09:20	0,05726	33,134	-1493	0,09287	6704,
5484	2015-05-07 09:23:00	gb_2015-05-07 09:09:20	0,055	33,1032	-1494	0,11787	6704,
5484	2015-05-07 09:23:10	hb_2015-05-07 09:23:10	0,0405	30,5541	-1495	0,09286	750,2
5484	2015-05-07 09:23:20	hb_2015-05-07 09:23:10	0,0253	30,8894	-1496	35,3357	2015,
5484	2015-05-07 09:23:30	hb_2015-05-07 09:23:10	0,09403	31,1833	-1497	78,7428	2257
5485	2015-05-07 09:23:40	hb_2015-05-07 09:23:10	0,2566	30,8559	-1498	84,7357	2443,
5485	2015-05-07 09:23:50	hb_2015-05-07 09:23:10	0,40119	30,6198	-1499	85,8143	2605,
5485	2015-05-07 09:24:00	hb_2015-05-07 09:23:10	0,5167	30,5541	-1500	88,3714	2790,

Figure 2. Here is how the same node as in the previous figure looks in DBMaker. The first column contains the time stamps of the observations. The remaining column headers are tags. Each column contains measurements of the tags. Each row constitutes one observation.

Tag names and node names are case sensitive. In SIMCA-online the instance name of a SimApi on that server, such as ":DBMaker" or ":ODBC", is also case sensitive. This rarely results in problems in SIMCA-online, but care must be taken when exporting project configurations to BCG-files (text files) and manually editing them with a text editor.

4.1.3 Batch node

A batch node is a node that keeps track of batches; their batch identifiers, start times, and end times. It is a requirement for batch project execution in SIMCA-online. Tip: if you don't have a batch node in your data source, you can use the Batch Context Generator in SIMCA-online. See the built-in help.

A batch node can optionally store batch data. Batch data can be values for batch conditions such as quality or yield, or data for local centering if it is used in a configuration.

Sorting_missing-BatchNode [Batch Node]

	1	2	3	4	5	6	7	8	9
1	Observation	B	Start Time	Stop Time	QP1	QP2	Innoc	Amount	Yield
667	Ta_2015-05-07 06:51:00	T	2015-05-07 06:51:00 (1430974260)	2015-05-07 07:04:40 (1430975080)	c	d		6960	0,505668
668	Va_2015-05-07 07:04:50	V	2015-05-07 07:04:50 (1430975090)	2015-05-07 07:18:30 (1430975910)	a	b			
669	Xa_2015-05-07 07:18:40	X	2015-05-07 07:18:40 (1430975920)	2015-05-07 07:32:20 (1430976740)	c	d	993,28	6597	0,504218
670	Za_2015-05-07 07:32:30	Z	2015-05-07 07:32:30 (1430976750)	2015-05-07 07:46:10 (1430977570)	a	b	952,32	5541	0,45427
671	ab_2015-05-07 07:46:20	a	2015-05-07 07:46:20 (1430977580)	2015-05-07 08:00:00 (1430978400)	c	d	967	5749	0,492269
672	bb_2015-05-07 08:00:10	b	2015-05-07 08:00:10 (1430978410)	2015-05-07 08:13:50 (1430979230)	a	b	914	5365	0,482205
673	cb_2015-05-07 08:14:00	c	2015-05-07 08:14:00 (1430979240)	2015-05-07 08:27:40 (1430980060)	c	d	952	5875	0,500618
674	db_2015-05-07 08:27:50	d	2015-05-07 08:27:50 (1430980070)	2015-05-07 08:41:30 (1430980890)	a	b	952	5835	0,505364
675	eb_2015-05-07 08:41:40	e	2015-05-07 08:41:40 (1430980900)	2015-05-07 08:55:20 (1430981720)	c	d	952	5973	0,4816
676	fb_2015-05-07 08:55:30	f	2015-05-07 08:55:30 (1430981730)	2015-05-07 09:09:10 (1430982550)	a	b	960	5402	0,46254
677	gb_2015-05-07 09:09:20	g	2015-05-07 09:09:20 (1430982560)	2015-05-07 09:23:00 (1430983380)	c	d	950	5982	0,47437
678	hb_2015-05-07 09:23:10	h	2015-05-07 09:23:10 (1430983390)	2015-05-07 15:28:23 (1431005303)	a	b	943	5977	0,470986

Figure 3. A sample batch node named **Sorting_missing-BatchNode**. The first column contains batch identifiers, followed by the start time and end time of the batch. The last five columns are tags with batch data (batch conditions). This example also shows the different data types which are defined in 4.1.4: QP1 and QP2 are qualitative (text) tags the remaining tags contain numerical data. Also notice the missing data highlighted in yellow in some cells.

4.1.4 Data types: numerical data, text data and missing data

For each tag, a SimApi can support three types of data: numerical, text and missing:

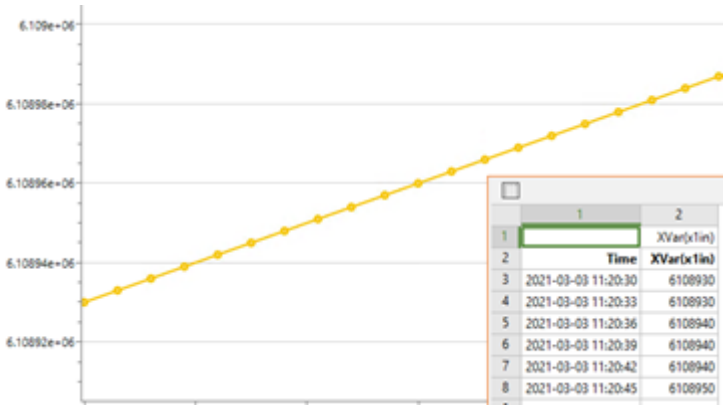
- **Numerical** data are real values, typically of process parameters, for example 6.5
- **Text/string** data are used for batch IDs, phase execution conditions or for qualitative variables. The values for text tag data are case sensitive. This means that the value "running" is not the same as "RUNNING".
- **Missing values** means there is no value to return, i.e., no data.

4.1.5 Numerical precision: single-precision floating point values

Numbers are read from data sources through SimApis as single-precision floating-point values. As such they can deal with both large and small values but with only about 6 or 7 significant digits. Learn more at Single-precision floating-point format – Wikipedia.

In SIMCA-online:

- If batch identifiers are numeric, and larger than 16777216, consecutive numbers will not be interpreted correctly. Learn more in 4.1.11.
- For model variables, a number cannot be large and have significant decimal places at the same time. For example, the number 1234567 is large, and cannot necessarily be distinguished from 1234567.890. If all measurements of a variable have a large baseline (maybe 1234000 in this example), you may want to subtract the baseline from all numbers both in the model and in your data source.
- There is a related issue in SIMCA-online lists that created from plots. The precision displayed in lists in more conservative than the actual floating-point values. In the below example notice that even though the line in the plot increments with 3 for each point, the spreadsheet shows only increments of 10. This only affects presentation in the spreadsheet and not computations in SIMCA-online.



	1	2
1		XVar(x1in)
2	Time	XVar(x1in)
3	2021-03-03 11:20:30	6108930
4	2021-03-03 11:20:33	6108930
5	2021-03-03 11:20:36	6108940
6	2021-03-03 11:20:39	6108940
7	2021-03-03 11:20:42	6108940
8	2021-03-03 11:20:45	6108950

4.1.6 Batch maturity and $Time

A batch maturity variable is used in SIMCA as the y-variable in a batch evolution model. In SIMCA-online you associate this variable in the model to a tag in a process node in the data source. SIMCA-online reads data for maturity from that tag at each execution interval for the model.

Values for a maturity variable must be strictly monotonic (increasing or decreasing during the phase evolution). Thus, it follows that maturity should normally **not** stand still during the batch evolution. Read more on this in 6.10.

If you don't use a maturity variable from your data source, your model instead uses **automatically generated time, or $Time,** as the y-variable. Values for this variable are computed automatically by SIMCA and SIMCA-online, and always start at 0 for each phase model.

If you use $Time, you should use the same execution interval in SIMCA-online as the sampling interval used when building the model, otherwise the speed of evolution in SIMCA-online will not match the SIMCA project.

We recommend using a maturity variable in the data source that has a natural meaning in the process, for example 'tank level'.

4.1.7 Batch conditions

A batch condition is a tag in a node that holds a single, time/maturity-independent, value that applies to the whole batch. For example, product quality, yield, or process set points. Unlike phase- and sleep conditions, a batch condition is not a logical expression, it simply is a tag or a variable in a SIMCA project.

Batch conditions are typically read as **batch data** using the SimApi:

4.1.8 Batch data

Batch data refers to a single observation with data for an entire batch (not associated with a specific maturity or time point). Batch attributes and local centering data (see 6.11) are read as batch data in SIMCA-online. Batch conditions are normally read as batch data too (unless they are configured to use discrete data retrieval).

The figure in 4.1.3 shows an example with five batch condition tags inside a batch node.

4.1.9 Batch attributes

Batch attributes are used to associate additional information to each batch (metadata about the batch). This information is visible in the desktop client as tooltips and can be used to color points in plots so that all batches with a certain value for a batch attribute get the same color. Learn more in the help.

Batch attributes are read as batch data using a SimApi.

4.1.10 Active batch

An active batch is a batch that is currently running in the data source. This is specified in the batch node as a batch that has started at some starting time, but not yet ended (has a null or empty value as its stop time).

4.1.11 Batch identifier (Batch ID)

A batch identifier, or batch ID, is a value that uniquely defines a batch within a project configuration. A batch ID cannot be reused for multiple batches, but workset batches in the project can have same names as batches later being predicted in SIMCA-online.

The batch ID is read from a tag in the data source during execution. This tag should be a string tag, or a numerical tag containing integers ranging from -16777216 to +16777216. For a larger numerical range SIMCA-online loses precision (see 4.1.5) and batch IDs won't be unique. If you use a text tag it is recommended that you trim whitespace around the batch ID strings.

The batch identifier tag is configured per unit, and thus is shared between all phases for that unit. Consequently, different batches can be active at the same time in a project configuration but in different units.

The batch ID read from the tag must keep its value for consecutive observations if the observation is to belong to that specific phase iteration for a specific batch.

Note: the **batch node** also contains a batch ID, and it must match the process data for SIMCA-online to execute the project configuration.

Chapter 5.2 covers how the server executes/predicts configurations and how the batch ID is used in detail.

4.1.12 Batch identifier filter

A batch identifier filter controls which batches are visible and thus can be predicted in a SIMCA-online configuration. The filter is set on the batch node page of the configuration. The default filter * means that all batches in the batch node will be visible. The batch identifier filter also supports regular expressions. See the help to learn more.

4.1.13 Phase execution condition

A phase execution condition is a logical expression that controls if a phase is executed. The expression is typically evaluated using data from one or more process tags.

You must match the expression with the data contained in the tag: text data should use text expressions and numerical data value expressions. For example, if the tag contains text the expression could be TextTag(":ODBC:node/tag1") == "P1" but if the tag contains numerical value the expression could be ValueTag(":ODBC:node/tag1") == 1.

Each phase model has its own phase execution condition. It is evaluated by the server on each execution of an observation to determine which phase it belongs to.

The phase execution condition must be true during the entire execution of a phase iteration. Once it is no longer true, the phase stops. Once stopped, the same phase iteration will not start again for that batch, but another phase iteration will start when the phase execution condition is true again. Also see 4.4 for information on phase iterations.

4.1.14 Sleep condition

A sleep condition is an optional logical expression using data from one or more tags. It is evaluated to determine if an observation should be ignored instead of predicted. Each unit has its own sleep condition. The same sleep condition is thus used for all phases of that unit.

If an observation is not predicted because of the sleep condition, missing values will be used instead in plots and lists.

Note that, as described above, the batch identifier tag still needs to hold the same batch id, and the phase execution condition must be fulfilled if the batch should be kept active in the phase.

4.1.15 Execution interval and fixed execution times

The execution interval, measured in seconds, specifies how often a project configuration will be executed, i.e., how often an observation will be read from the data source, conditions evaluated, and models predicted.

The exact time of predicting an observation is deterministic and computed from the execution interval. Thus, it does not matter when execution for a project configuration is started, for when the actual predictions will take place. Predictions are synchronized to UTC, universal coordinated time[1], and does not use local time as the third example illustrates

- An execution interval of 10 seconds would result in 6 executions per minute. They will always occur at HH:MM:00 UTC, HH:MM:10 UTC, HH:MM:20 UTC, HH:MM:30 UTC, HH:MM:40 UTC and HH:MM:50 UTC.
- An execution interval of 15 minutes would result in 4 executions per hour. They will always occur at HH:00:00 UTC, HH:15:00 UTC, HH:30:00 UTC and HH:45:00 UTC.
- An execution interval of 6 hours would result in 4 executions per day. They will always occur at 00:00:00 UTC, 06:00:00 UTC, 12:00:00 UTC and 18:00:00 UTC.

4.2 Batch configuration layout and plot layout

This chapter defines the higher-level concepts **unit groups, units, phase models** (also referred to simply as **phases**), and **sub-batches**, and how they affect the layout of plots and the overall execution. We also specify at which level the various settings apply in a configuration (some settings are global, other per unit, per phase, or per variable).

4.2.1 Definitions

A batch evolution model is built in SIMCA with a specific set of variables. The model is built to handle a specific phase. There can be one or more phases in a project. The models are saved in a SIMCA project (USP-file).

A **phase can only execute once for each batch, unless the project uses phase iterations.** This is explained further below.

SIMCA-online adds the following concepts:

Concept	Explanation
Unit	Where SIMCA-online execute phase models. • A unit in SIMCA-online can correspond to a physical unit in the process or be an abstraction used to control the plot layout and for settings that control the project

[1] https://en.wikipedia.org/wiki/Coordinated_Universal_Time

execution (you will learn more of this later in this and subsequent chapters).

- There can only be one active batch at the time in a unit[2]. If a real-world process has several active batches at the same time but in different process steps, these steps should correspond to different units in SIMCA-online to be able to handle the parallel batches.

Unit Group	An abstract container of one or more units of the same type. A unit group also holds optional sub-batches (see below). • A unit group specifies a set of phases that can be executed in its units.
Sub-batch	A place where models can be executed. Sub-batches are specified per unit group in parallel with units. Sub-batches are optional and not visible in the user interface by default. • Sub-batches can be useful if the batch is divided into parts that are individually run through the same physical unit sequentially or run through different physical units in parallel. In this case one model must be created for each part and the models then set to execute in individual sub-batches. The reason for this is that a phase can only execute once per batch. • Use the context menu (right-click) in the Unit groups page to display sub-batches and then to manage them (as for all settings on this page, modifying the layout is only possible when creating a new configuration). • Note that sub-batches are obsolete, and in most cases should be avoided in SIMCA-online since the only thing they bring is different evolution plot layout as described below. *Instead, phase iterations should be used.*

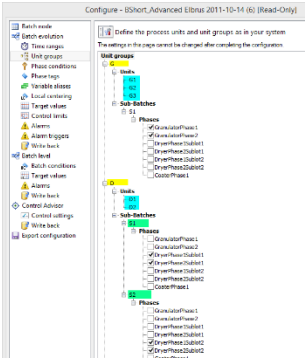
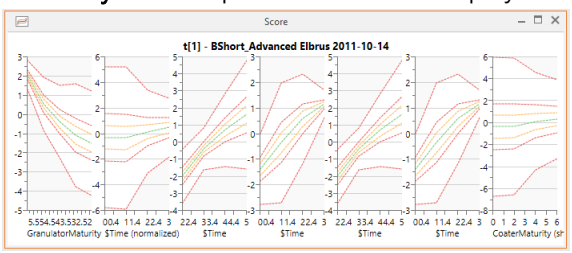


Figure 4. The Unit group page of an example configuration with a complex layout. Two unit groups are visible (colored yellow). These have three and two units respectively (blue). There are also two sub-batches present in the second unit group (green). Below you' ll see the batch evolution plot layout this results in.

4.2.2 Layout in evolution plots

Batch evolution plots can display the data in two different layouts, selected in the Properties pane in the desktop client.

- **Phase layout** - the phase models are displayed from left to right, ordered as in the SIMCA-project:



- **Unit** layout- unit groups, units, and sub-batches (when used) are used to organize the plots in the following way:
  - Unit groups are displayed from left to right. Each unit group gets the same size (width and height).

[2] As specified in ANSI/ISA-88.01-1995 (Batch Control, Part 1: Models and Terminology).

- Within a unit group, the individual units are listed from top to bottom.
- For each unit the individual sub-batches are listed from top to bottom.
- For each unit the individual phase models are listed from left to right, ordered as in the SIMCA-project.

Tip: the default layout can be changed in the per-user Plot Options, available on the File tab.

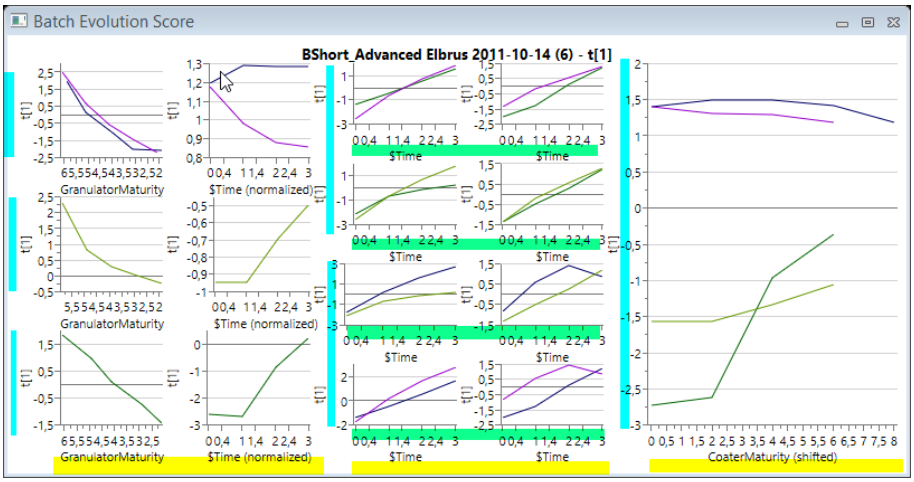


Figure 5. An example of a batch evolution plot with the Unit layout and a complex layout defined on the Unit group of the configuration. It has been annotated with colors and shows three-unit yellow groups from left to right. The unit groups contain three, two and one blue units respectively. The middle unit group also has two sub batches (green). Each individual sub plot shows data for a specific phase that has executed in that combination of unit and sub-batch.

4.2.3 Layout in batch level plots

Batch level plots don't use the unit layout, but rather display a simple grid of all batch level models that have been selected for execution in the configuration's batch level page.

Note that all models always are displayed in the grid, but that sub plots corresponding to some plots can be empty. This happens for example if one batch level model is a PCA-model, but the others are PLS. Since the PCA model doesn't have Y-variables that plot will be empty in a batch level Y-predicted plot.

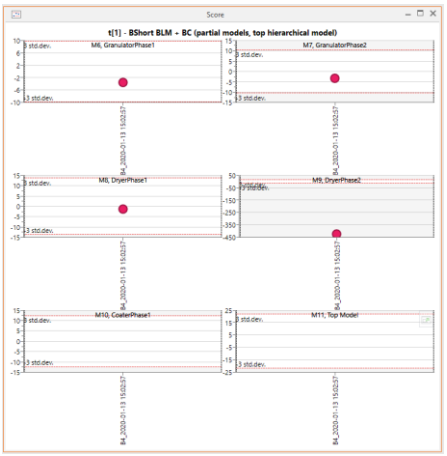


Figure 6. A batch level plot with the grid of sub plots for each batch level model.

4.2.4 Configuration settings are applied at different levels

When a project is configured in SIMCA-online, you make changes that apply at different levels.

The following table summarizes at which levels where the various settings apply. The first two columns show the configuration wizard page and explains the settings. The remaining columns are used to show at what level the settings apply, indicated with an 'o'.

Configuration Page	Setting	Unit	Sub-batch	Phase	Variable	Comment
Batch node	Batch node and batch identifier filter					For the entire configuration
Batch evolution	Execution interval, data retrieval mode			o		
Execution conditions	Batch identifier tag, Sleep condition	o				
	Phase execution condition	o	o	o		Per combination of unit, sub-batch, and phase
Phase tags	Associate variables to data source tags	o		o		Per combination of unit and phase.
Variable aliases	Alias for variable names				o	Per variable in any phase (aliases are not used for batch level variables)
Control limits					o	Per variable in any phase.
Local centering		o		o		Per combination of unit and phase.
Target values, Alarms, Alarm triggers, Write back				o		Per phase
Control Advisor				o		Per phase

This has the following implications:

- Settings made per phase (such as alarms) cannot be configured differently per unit.
- Settings made per phase and unit (such as phase tags) cannot be configured differently per sub-batch.
- Also note that Unit group is not present in the above table, since settings are not applied to them in a SIMCA-online configuration. However, they are used for plot layout as described previously.

4.3 Model execution limitations in projects without phase iterations

A SIMCA project can support phase iterations which means that the same phase can run more than once for a batch. This must be done in desktop SIMCA, explicitly adding phase iterations. Otherwise, the following limitations holds in SIMCA-online: **A phase model can only execute once per batch**.

This limitation comes from the fact that when the evolution level is summarized to the batch level, SIMCA and SIMCA-online can only handle the results of **one** execution of each phase.

This means that, while the graphical interface allows that the same phase model is selected in two units, that model will only execute once (for the unit that happens to execute first).

4.4 Phase iterations can run more than once in project with phase iterations

For projects created in SIMCA with **phase iterations**, it is possible to run the same batch in the same phase more than once.

See the user guide for SIMCA for information on how to build these models, and the SIMCA-online help for more information how phase iterations execute in SIMCA-online.

4.5 How to mix continuous and discrete phases in SIMCA-online

A phase model can only be configured for either discrete data retrieval or continuous data retrieval, not both. Since all phases are shared within a unit group it is recommended to create one logical unit group for discrete data and one logical unit group for continuous data.

Tips:

- Data from both models may be combined in the batch level.
- Steady-state models introduced in SIMCA 18 can be used to model continuous phases within a batch project.
- See 5.2.8 to learn how Baseline filter can be used to handle IPC data in a regular continuous phase.

5 Project Execution

This chapter describes how the server executes a project configuration and how you can troubleshoot problems. Execution is the core functionality of the server. It means:

1. loading data from the data source
2. deciding which model to predict
3. deciding when to predict
4. evaluate predictions to trigger alarms
5. make the data available to clients for visualization and interpretation

We'll mostly discuss a batch configuration in this chapter since batch projects are more complex than continuous projects. The differences are that batch configurations require a batch node and can have multiple models (evolution models for phases and batch level models) unlike continuous configurations which use a single model.

This chapter is the introduction to this topic; the next chapter contains advanced topics.

Occasionally in these chapters we will show the Debug-level entries that will be added to the server log in certain situations. This of course assumes that the log level has been set to Debug.

Note: SIMCA-online 17 optimized data reading to improve performance: many different data read calls through the SimApi are now grouped into a single call. Learn more in 5.6.

5.1 Continuous Project Execution

A continuous (non-batch) configuration has two settings that affect project execution:

- the execution interval - how often to sample data from the data source
- the prediction condition - a logical expression that must evaluate to true for the model to be executed. The prediction condition defaults to True (always execute), but that can be changed to use process data to decide if an observation should be predicted.

The following algorithm is run at each execution interval:

1. Data are read from the data source through the SimApi for all data needed by the model: both process data and control data. Control tags are the tags used in the execution condition. This is logged in the server log as `[Debug]` DataSourceManager::ReadCurrentData()] or DataSourceManager::ReadHistoricalData()].
2. The execution condition is evaluated for the control data to determine if the model should execute. If not, then exit this algorithm.
3. Evaluate any (univariate) target value alarms. Triggered alarms are logged as "Alarm triggered" in the server log event [Information] [ContinuousProjectCalculation::UpdateTargetValueAlarms()].
4. Compute the values for any generated variables.
5. Perform the actual predictions of the model. This is logged in the server log as [Debug] ContinuousProjectCalculation::PerformProjectExecution().
6. Evaluate any multivariate alarms. Triggered alarms are logged as "Alarm triggered" in the server log event [Information] [ContinuousProjectCalculation::UpdateAlarms()].

5.2 Batch evolution level execution

The following algorithm is applied consecutively to each unit in a configuration. The execution intervals of the phase models in the unit are used to determine at which times the algorithm is triggered (See 4.1.15). See later in this chapter for how this algorithm applies to more than one unit.

1. Data are read from the data source through the SimApi for all data needed by the model: process data and control data. Control tags refer to the batch ID tag, all tags used in the phase execution conditions, and all tags used in the sleep condition. This is logged in the server log as `[Debug]` DataSourceManager::ReadCurrentData()] or DataSourceManager::ReadHistoricalData()].
2. The control data, the batch node, and the configuration settings are used to evaluate if the model(s) in this unit should be executed for this observation. This is logged as [Debug] [SegmentCalculationEngine::EvaluateControlData()]:
    a. The value of the batch ID is checked. If it is empty/missing or does not match the batch identifier filter (see 4.1.12) then exit this algorithm. If a phase was already executing in this unit that phase is now finished.
    b. The batch node is consulted to check if the batch ID represents an active batch (without a stop time). If the batch is no longer active then exit this algorithm, any phase that was already executing in this unit is now finished. The batch node queries are logged as [Debug] [DataSourceManager::GetActiveBatches()] and ::GetBatchTimes() in the server log.
    c. The phase execution conditions is evaluated for the control data to determine if the phase is active in this unit. If it isn't active, then exit this algorithm. If another phase was already executing in this unit that phase is now finished.
    d. A check is made if the phase already has finished, if so then the same phase will not start again for this batch. However, a new phase iteration can start (see 4.4).
    e. The sleep condition is checked to determine if this observation is to be ignored.
3. Evaluate any (univariate) target value alarms. Triggered alarms are logged as “Alarm triggered” in the server log [Information].
4. Compute the values for any generated variables.
5. Perform the actual predictions of the model. This is logged in the server log as `[Debug]` BatchCalculation::PredictObservationLevel().
6. Evaluate model-based evolution level alarms. Triggered alarms are logged as “Alarm triggered” in the server log [Information].

After all units have executed as described above, batch level models are executed for the project configuration as described in 5.4.

Then the server waits (for this configuration) until the execution interval has passed and it is time for the next execution where it starts over from the beginning.

The above algorithm describes real-time normal execution, but it also applies for prediction of entire finished batches and catch-up (see 6.1).

5.2.1 What is required for a batch to execute in SIMCA-online?

As can be seen above, for a batch to be predicted in SIMCA-online for each observation the following must hold:

- A batch with the same ID must exist in the batch node.
- The observations has a timestamp between the batch node batch start time and end time.
- A batch with the same batch ID has not been previously executed; if the batch has already been predicted in the project configuration, a new batch with same name will not be predicted.
- The **same batch ID** must be used in the process data as in the batch node. Note that if the batch ID becomes missing for even a single observation that phase will stop and will not resume. Tip: this can be mitigated using the Control Tag Cache.
- The batch identity filter must be fulfilled.
- The phase execution conditions must be fulfilled.
- The sleep conditions must **not** be fulfilled.

Note that unlike for the batch ID, missing values are allowed in data for phase execution conditions and sleep conditions and can be handled by the server so that it for example ignores such observations. Use the `IsTagMissing()` function to ignore observations with missing value.

5.2.2 When does the server read data through the SimApi?

The server reads data for control tags to determine if the batch is active in a unit at the same time as the process data for all variables in the models are read. The server log file will show when these data calls are made if debug-level logging has been enabled.

5.2.3 Data source SimApi error handling in the SIMCA-online server

A SimApi can handle errors by returning missing values or by signaling error codes. It typically also logs error information to the SimApi log file. SIMCA-online behaves differently in these cases:

1. If missing values are returned, SIMCA-online will see these missing values but does not report it to the user unless target value alarms have been set up for missing values. Instead, the missing data manifests in SIMCA-online as lack of updates in plots, that batches don't start as expected, or that phases end prematurely because phase execution conditions no longer are true (see 5.1 above).
2. If an error code is signaled, SIMCA-online will see the error and log it to the server log. A notification message is displayed on each connected client. All data that was requested in that function call to the same SimApi are treated as missing values. The server then uses the execution logic to determine what should happen, as in case 1.

To determine the cause of the SimApi error or missing values, you can configure the server- and SimApi logs to contain debug information and analyze the log files.

5.2.4 Sleep conditions

Any tag (including a tag used in the phase execution condition) can be used in a sleep condition.

Tip: You can observe when observations are ignored because of sleep conditions in the server log: search for Information events with the text for "Sleeping observation at".

5.2.5 How are Generated Variables evaluated?

A value for a generated variable can only be computed when there is data for all variables used in the formula; one missing value in a dependent variable means that the generated variable will become missing too. This means that all variables used in a generated variable must be connected to tags in the data source.

A generated variable that uses previous values of variables (using functions such as Avg, CuSum, Lag, Delta, Diff1, Diff2, EWMA) use cached values for the previous values needed by the generated variable, ensuring good performance. When there are no cached values, such as in the beginning of a phase, missing values will be used.

A limitation for generated variables is that target value alarms (univariate alarms) cannot be used since they will never be triggered. However, in SIMCA-online 18 and later can now have trigger-based alarms for generated variables and Python preprocessing variables.

5.2.6 Problems with back-dated or out-of-order data in the data source

As can be seen in the execution algorithm above the server reads data observation by observation as the project configuration executes. The server will never go back to re-read data that arrived late in the data source during normal execution. Another way to put it is that SIMCA-online cannot use data that arrives in the future.

This means that a delay in the data source before the data is available will cause problems. If data for a variable isn't available, then the last value will be used. If this is old, it can cause problems. Note: this can partially be circumvented by using sleep conditions so that a phase sleeps until all data has been analyzed and entered in the data source.

The sequential nature of the server's execution also means that you cannot make changes in the data source in past data and expect the server to pick up those changes.

Another way back-dating can happen is if the data source receives out-of-order events from data collectors and then archives these in the correct order which effectively back-dates some data[3].

The only way to force a re-read of past data for a regular phase (with continuous retrieval mode) is to delete the batch and predict it again in SIMCA-online.

Discrete data retrieval mode for phases and Baseline filters are two ways to avoid these problems as you'll see next.

5.2.7 Phase execution for discrete data retrieval

Discrete data retrieval changes how data is requested so that **all data** is re-read each time the model is executed.

This allows SIMCA-online to handle delays in data entry and corrections in previous data which is useful in at-line and off-line data such as in-process control (IPC).

Once the server has read the data at a time of execution, it evaluates it by comparing it to any data previously read. Only if data has changed or is new, the whole model with be predicted again.

Note

- Discrete phase models must always use a maturity variable so that SIMCA-online can match the observations against the correct aligned observation in the phase model. Generated automatic time $Time does not work well in most situations.
- A limitation in how the server executes configuration is that a phase with **continuous** data retrieval mode always must exist in a configuration with a discrete phase. The continuous phase is required for the server to be able to pick up that the batch is active in the process.
- Since data can change during the lifetime of the phase, the multiple predictions of the whole model will likely re-trigger any alarms that has previously been reset by the user.
- If data has changed, the points visualized in plots might not be the values that triggered previous alarms that remain.

For more information about discrete data retrieval, see the help.

5.2.8 Mix IPC and continuous data in the same evolution model by using Baseline filters

As an alternative to discrete data retrieval described previously, SIMCA-online 17 added the feature Baseline filters.

Baseline filters are designed to solve the problem of getting values that applies to a previous batch in the same unit when using phase models that combine data from at-line IPC variables with continuous data. This setting makes SIMCA-online ignore values for variables in the beginning of the phase until they have changed, because only a changed value since the batch start time can belong to that batch. Learn more in the help topic 'Baseline filters to ignore variables with unchanged data at beginning of phase'.

5.3 Execution logic for configurations with more than one unit

SIMCA-online uses the execution interval for the phases (in all unit groups) to determine at which time points to obtain data from the data source.

At such a time point, the control data for the tags for batch identifier, phase execution conditions and sleep conditions are read for the units and phases that execute at that time.

The data is then evaluated and one or more models for that unit may execute. If the same model were to be specified for multiple units, the first unit to execute that model will prevent other units from executing it

[3] One data source this can happen in is in AVEVA PI where out-of-order (OOO) events from data interfaces can be inserted after the fact into the PI archive.

except if phase iterations are used, because then the model will result in one phase iteration in each unit executing simultaneously.

The detailed execution algorithm for a single unit in SIMCA-online is described in 5.1.

5.4 Batch Level model execution

In the basic case, a batch level model uses batch evolution data, i.e., scores or raw process data collected during the evolution of the batch. In addition, raw data statistics (computed from the evolution data, such as min, max, mean, std. dev., interquartile, slope), duration/endpoint variables for phases and batches, and batch conditions can be used in the batch level model. Learn more on these topics in SIMCA's help.

5.4.1 When are batch conditions read?

Batch conditions are normally read as **batch data** from a SimApi. Here are the rules that control how batch conditions are read:

- The server reads batch conditions when a batch is started, when a previously suspended batch is resumed, when a batch level model is about to be predicted, when control advisor prediction is run, and when a batch is finished.
- A project configured for Late entry data, with the setting 'Batch condition update interval', will refresh batch condition data at the configured interval regardless of if there are missing values or not. Learn more in help.

5.4.2 Batch conditions read as discrete data

Batch conditions can be configured to read data as discrete data. In this case many observations are read for a batch condition and SIMCA-online will use the *average* of those.

5.4.3 When are batch level models executed for active batches?

How does SIMCA-online know when to predict a batch level model for an active batch executed in real-time?

The short answer is: When the server has the values for **all maturity-dependent variables** used in the model. If at least one of the variables isn't available yet, the model will not be executed. Examples of maturity dependent variables are pH_M1_3, pH_M1_5 where the last number represents the maturity values.

Thus, unlike the offline software SIMCA which has all data available at once, SIMCA-online gets its data observation by observation sequentially as the phase evolves. SIMCA-online waits until it has data for all variables used in the model, before the batch level model is predicted.

This means:

- The phase must reach the maturity of the variables used in the batch level model for it to be predicted (see next heading).
- Partial models can be used in SIMCA to create batch level models that are executed one after the other as a batch evolves.
- If a model uses raw data statistics (computed min, max, std. dev, etc) or phase/model duration or endpoint, the entire phase (or batch) needs to finish before the batch level model can be executed.
- If batch conditions change, or if data read with the discrete data retrieval mode changes, a batch level model that use those data will be predicted again.
- Missing values for a batch condition variable do not prevent a batch level model from being predicted.
- If hierarchical models are used, the base models execute first, before the top-level model is executed.

When a batch level model is predicted it shows up in the server log as Debug-level BatchCalculation::PredictBatchLevel events.

Batch level alarms are evaluated after the batch level models have been predicted.

5.4.4 Why can batch level models be executed "too late" or not at all?

Sometimes, a batch level model does not execute when you expect. The following are three likely causes of this:

- The reasons listed under the previous heading.
- That maturity values usually aren't natural numbers like 1, 2, 3 but rather real numbers (for example 1.02, 1.89). Data is available at the batch level only when maturity is at or higher than the aligned maturities of the evolution model. Compare the maturities of your predicted batches with the aligned maturities of the evolution level model. The example below shows an example.
- The use of shifting, smoothing or normalization of the y-variables/maturity in the evolution models can change the values of the maturity. See 7.4.1.

For example

- Assume a batch level model uses Maturity at 0.0, 1.0, 2.0. We'll refer to these numbers as M0, M1, M2.
- Observations for a new batch come in at M0, M1.003, M1.191. Notice that we don't have a value for exactly M2 so the batch level model won't be executed now. Then if the phase ends without a new value at M2 or higher that batch level model won't be executed for M2.
- Another batch starts with values M0, M1, M2.1. For this batch the server will be able to interpolate a value for M2 so that the batch level model will be executed for M2.

5.4.4.1 Workarounds to deal with missing batch level predictions

These two workarounds can be used to deal with batch level models that aren't executed because of the problems with the maturity:

- Make sure a phase execute until the desired maturity has been reached, by configuring phase execution conditions so that the phase doesn't end too early.
- Make sure that the maturity data added to the data source always are at the exact desired values rather than keeping the real variation. For example, rather than exposing maturity M1.91 to SIMCA-online the data source rounds it to M2 which was used in the model.

5.4.5 The Batch Level Prediction setting in the configuration

A project configuration has a setting for controlling when batch models are executed:

Predict at Model Finish – the batch level model is executed as soon as it has all data it needs as described above.

This is logged in the server log as [Debug] [BatchCalculation::PredictBatchLevel()]

Predict at Batch Finish – the batch level model is executed when the batch is finished. This means at the time when the batch no-longer is active in the batch node of the project configuration. This uses the exact same rules for predictions as desktop SIMCA uses since at that point all data for the batch is available.

This prediction is done **regardless of how mature the batch is**. Interpolation and possibly extrapolation of data will be performed using the same rules as defined in SIMCA.

This is logged in the server log as [Debug] [BatchCalculation::PredictFinishedBatch()]

Predict at model finish and at batch finish - the batch level models are executed both at model finish and at batch finish. SIMCA-online preserves any batch level alarms between predictions.

Note: SIMCA-online only reads data once from the data source to do the batch level predictions, even if Predict at model finish and at batch finish is selected, because it always uses its cached data for the batch level model prediction at batch finish.

5.5 Real-time batch execution when execution is falling behind

When a batch project configuration executes in real-time, it requests data from the data source at the regular intervals configured for the phases. If the data source is slow so that data cannot be returned in time, the

SIMCA-online server will fall behind. This is noticeable in clients which will not see current data but can also be seen in the server log (see below).

The server uses an internal cache of the necessary data to reduce the number of data access calls to the data source. It also reads larger chunks of data when it sees that it is more than one observation behind. This increases performance when the data source is temporarily slow, and for servers with many executing project configurations.

Predicting past batches or catching up batches that are active in the data source, are also optimized similarly.

When the server notices that it is behind, it results in log entries like this explaining how far behind it is, and that cached data is used. Administrators can also subscribe to notifications to be alerted by an email when this happens.

[2019-10-25 12:40:30.834+02:00] [TID:22640] [Warning] **[BatchNormalExecution::Execute()]**

Performance warning: The execution is falling behind schedule. This indicates a slow data source, or a too busy server.

Configuration: BakersYeastMonitorAlarmDemo (2) (4ee88409-bb98-4d5b-a617-5d799b44ef6c)

Observation: 2019-10-25 12:40:06

Observations behind: 9

...

[2019-10-25 12:40:30.865+02:00] [TID:22640] [Debug] [**BatchCalculation::ExecuteBatch()]**

Predicted batch Ha_2019-10-25 12:38:18 (172), for observation at 2019-10-25 12:40:06 (1572000006), elapsed time 27 ms.

[2019-10-25 12:40:30.865+02:00] [TID:22640] [Debug] **[BatchNormalExecution::GetBatchInfo()]**

Using cached GetBatchTimes result.

Configuration: BakersYeastMonitorAlarmDemo (2) (4ee88409-bb98-4d5b-a617-5d799b44ef6c)

Batch: "Ha_2019-10-25 12:38:18", start time: 2019-10-25 12:38:18

Observation: 2019-10-25 12:40:06

Cached at: 2019-10-25 12:40:18

Furthermore, you can see that the server switches from using current data when it detects that is more than one observation behind:

[2019-10-25 14:29:31.134+02:00] [TID:23264] [Debug] **DataSourceManager::ReadCurrentData()]**

Read current data for 10 variables, return time 2019-10-25 14:29:31 (1572006571), elapsed time 4017 ms.

...

[2019-10-25 14:29:35.142+02:00] [TID:23264] [Debug] **DataSourceManager::ReadHistoricalData()]**

Read historical data for 10 variables, 6 observations, starting at 2019-10-25 14:29:09 (1572006549), interval 3 s, elapsed time 4006 ms.

...

5.6 Optimized reading from data sources improves performance

SIMCA-online 17 optimized data reading to improve performance: many different data read calls through the SimApi are now grouped into a single call. This means that the server significantly reduces the number of calls made through SimApis to data sources compared to previous versions.

This means that the server can spend less time waiting for slow data calls, and more time doing predictions or serving clients, resulting in better performance. Exactly how much better performance depends on the data source, the project configurations that are executing, and if Batch Context Generator is used.

See the help topic 'Optimized reading from data sources improves performance' to learn more about these optimizations. Starting in SIMCA-online 18 there is a setting that turn the optimized reading off.

5.7 Batch Control Tag Cache improves execution resiliency

The Control Tag Cache keeps phases alive during temporary data source problems. It re-uses last known good values for control tags (batch identifier tags and tags used in phase execution conditions) when the data source returns missing value or SimApi errors.

When the cache is active and all data are temporarily missing, plots typically show no data at all (missing points) for such observations. Later, when the data source problem is fixed, new observation will continue the phase execution.

The **control tag cache is opt-in** and is be turned on in the SIMCA-online Server Options utility. Learn more in help.

6 Project Execution - Advanced Topics

This chapter expands the topic of how the server executes a project configuration.

6.1 Several modes of execution

The server works in different modes when it executes project configurations. This can be useful to know to be able to interpret server logs and to optimize performance:

- **Normal execution**: one observation at a time in real time.
- **Manual prediction** of past batches initiated by the user. Data for these batches are first deleted (if already predicted) and then predicted as a whole.
- **Catch-up** of an **active** batch in the data source up to the current time ("fast forwarding"). The server reads data and performs predictions as fast as it can until it has caught up with current time, when the server switches to normal execution. This type of execution happens in two cases:
    1. When execution is started for a projection configuration, and a batch is active in the batch node. For technical reasons, the catch-up execution only starts when a phase is active in a unit. This means that for a multiphase project catch-up will not seem to do anything if execution is started in-between phases, but as soon as a phase starts catch-up will start too.
    2. When the server cannot keep up with real-time predictions because of high load, it switches to catch-up.
- **Initial catch-up prediction** of **finished, unpredicted** batches when execution is started for a projection configuration. The **catch-up time range** of the project configuration controls how far back the server will look for unpredicted batches when execution is started. These batches would otherwise have to be manually predicted by the user.
- **Recurring catch-up** for batch configurations is optional and runs at the configured catch-up interval. Recurring catch-up can be useful in the narrow case when entire finished batches are added to the data source with timestamps before the current time.

For a properly configured server, it does not matter with mode of execution the server uses; the results should be the same. However, out of sync clocks, or a poorly tuned data source can cause inconsistencies, which you'll learn later in this chapter.

6.2 What happens when execution is started for a project configuration

When execution is turned on for a project configuration the following happens

- If there were active batches when execution was stopped, those batches are resumed immediately, continuing from where the batch left off.
- The server starts executing phases. If the control data for a phase evaluates to true, a check is made if that batch is active in the batch node, and if so, the phase is started (and the batch is also started if it was previously not executing in SIMCA-online).

From this last point it follows that a batch may be active in the batch node, but still not be picked up until the execution condition is true for at least one phase. This is most noticeable if the execution interval is large. For example, 1 hour execution interval means that batches will only start at even hours regardless of when execution is started.

If initial catch-up is configured in the configuration, the server looks for finished batches within the time range and predicts those as well.

6.3 Server behavior when starting the service

When the SIMCA-online server service is started it will start execution for all project configurations which had execution turned on when the service was stopped. Tip: the option Resume execution in SIMCA-online Server Options can be unchecked if you don't want this behavior, for example if you want to start the service as quickly as possible without waiting for project execution to start.

6.4 When are batches deleted and what does it mean?

A batch is deleted from SIMCA-online in the following circumstances

- If you manually delete a batch (historical or a still active batch) in the Delete data dialog.
- If you predict a past, previously predicted batch, because this deletes all data for the batch, before predicting it again.

From this last point it follows that a batch also is deleted when you initiate a predict of it, but the server cannot obtain any data from the SimApi (perhaps simply because the data is no longer present in the data source).

When a batch is deleted, all predictions, and any notes and alarms are deleted for that batch. If a batch is deleted, and entry about this shows up in the audit trail for that configuration. Meta data, such as start and end times, and all alarms for the deleted batch always remain in the audit trail.

When you stop the server service, active batches are suspended in their current state, and are thus NOT deleted. Execution then resumes from that point when the service is restarted.

6.5 Data access through a SimApi can be a bottleneck

Data access through a SimApi is by default single-threaded in SIMCA-online. Executing project configurations will have to wait for each other to get data through the same SimApi.

SIMCA-online 18 supports a feature flag to turn on multi-threaded access through SimApi. Read more in the help topic Concurrent SimApi access.

If the data source itself is slow, or located far away over a slow network, this can make data access a bottleneck that can affect project execution negatively. This manifests itself as lack of updates in project configurations or a generally slow server. It can be seen in server logs as long elapsed times for data access calls (see 2.5.1).

The likelihood of data access becoming a bottleneck increases when:

- The execution intervals are short, resulting in too many data access calls.
- the network has low bandwidth or high latency between SIMCA-online server and data source.
- the data source itself is slow.

Before SIMCA-online 17 which optimized data reads as described in 5.5, the following also risked making data access a bottleneck:

- many executing project configurations use the same SimApi.
- the project configurations use the same execution interval, resulting in that they will ask for data at the same times.

6.6 When does the server use Current and Historical data respectively?

Current data is read by the server through the SimApi during normal execution of one observation at a time in real time.

Historical data is read when predicting past data, or, more subtly, whenever the server is catching up. See 6.1 above for when catch up can happen.

In 2.5.1 you can the server log Debug-level entries for when current and historical data are read.

6.7 Problems exposed by alternating between current and historical data

Reading current and historical data alternately by the server can lead to anomalies in data in the following situations:

- When the server clock and data source clock are not synchronized, as is described in 6.8 below.
- When data is backdated in the data source; that is when a row in a database is added with a time stamp of a past time point. In this case the data isn't there when SIMCA-online asked for it in real-time, but later when someone looks in the data source, he or she sees it and cannot understand why SIMCA-online missed it. See 5.2.6.
- When data is compressed and interpolated by the SimApi or the data source, data read as historical data can mismatch current data. See 6.9 below.

To detect issues like this you can create time stamps in the data source whenever rows are added. These time stamps can then be used together with server log entries to determine why the server didn't pick up the data as you expected.

6.8 On the importance of time synchronization between servers

It is important that the system time of the SIMCA-online server computer and the system time of the external data source are synchronized to the second. Otherwise, there can be inconsistencies in the data when timestamps are used in data queries.

Here is an example:

- When a SimApi is used to obtain the current data, the data source returns the data, and then SIMCA-online timestamps it with the time of that observation. Notice that SIMCA-online does **not** use the time of the data source.
- Now assume that 10 minutes later the server wants to re-read that same observation again. It asks the data source for data for that time point, and the data source returns it, but if the times are not synchronized the data returned won't match the data that was read as current data 10 minutes ago.

Here is real life example:

- One person observed that plots seemed 90 seconds off at sometimes during execution of a batch, but at other times the plot looked fine. It turned out that the server was overloaded and couldn't always keep up with real time execution. Instead, it switched to catch-up execution where it read larger chunks of historical data which then did not match the data that had been read as current data before.

6.9 Compression and interpolation account for differences between real-time and historical data

Some data sources (such as AVEVA PI) can use compression to reduce storage requirements for historical data.

When data is read through a SimApi, compression and interpolation in the data source will make the data look differently in the two cases: smooth interpolated lines for historical data, but repeated values in a stepwise fashion for real-time values as the following plot illustrates.

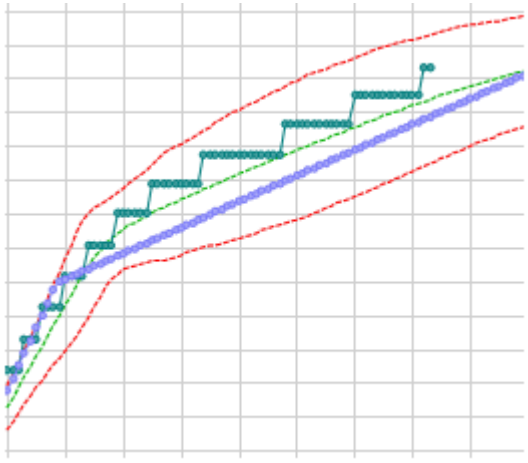


Figure 7. The effects of compression and interpolation. Dark green points are real-time values (compressed values), blue points show the same data but interpolated data read during a prediction of a finished batch.

These differences can constitute a large problem for the SIMCA model validity. This must be addressed in the data source (not in SIMCA-online).

For a thorough analysis of problems like this, including specifically for AVEVA PI, see the knowledge base article Historical data is different from data in real-time (Q612) (sartorius.com).

6.10 Maturity is always strictly monotonic – never stays still during evolution

A maturity variable in must always be strictly monotonic in Umetrics Suite products. It can never hold the same value for two consecutive observations.

If the variable isn't monotonic both SIMCA and SIMCA-online will apply a mandatory algorithm for ensuring a minimum increment of maturity. It works by adding a small amount to the maturity to make its value unique[4]. This happens regardless of any other maturity treatment such as smoothing.

This means that if you use a maturity variable that you expect to hold the same value for a number of observations, it actually will change its value slightly every observation. An example of such a maturity variable would be "Tank Level": If the flow to the tank is shut off for a period of time, then the Tank Level isn't increasing any more, but in SIMCA or SIMCA-online it will be increasing because of the above algorithm.

Tip: If this must be avoided, a sleep condition can be used in SIMCA-online to ignore observations. In the above example, the sleep condition could be true when the flow is 0.

6.11 Local centering

Local centering can be used to adjust values of variables in the phase models to compensate for systematic errors that varies over time between batches. Local centering can be applied in SIMCA when doing the offline modeling. Local centering can also be used in SIMCA-online when performing predictions, even if not used in the SIMCA project.

Here is how it works in SIMCA-online:

[4] The actual formula used in SIMCA-online is MinIncrement = (StopMaturity - StartMaturity) / AveragePhaseLength / 1000.

- Local centering is configured per unit and phase (see 4.2.4). For each variable you can associate a tag that will be used to read the value to use for local centering.
- For each batch and phase that is to be predicted SIMCA-online obtains the values for local centering by calling the SimApi function `ReadBatchData()` for special tags used for local centering. This means that a SimApi that supports batch data is required.
- If no value for local centering was obtained through the SimApi or if local centering is left unconfigured, SIMCA-online will not do any local centering.

7 Tips and Tricks for Data Scientists

This chapter is aimed at data scientists and other people who are implementing SIMCA-online solutions. It contains tips and explanations to questions that come up often.

7.1 Creating a batch context (batch node)

A batch node defines batch names and lifetimes and is required by SIMCA-online to execute batch configurations. If a batch node is unavailable in the data source there are ways to deal with it:

- Investigate how to enable batch node functionality in your data source. For example, if you use AVEVA PI, use Event Frames that are supported by SIMCA-online through the PI AF SimApi.
- Use the built-in Batch Context Generator with its Generator-type nodes in SIMCA-online. It is quick to get started with and easy to change but can cause performance issues. Therefore it is recommended to aim to replace Batch Context Generator with native data source batch node functionality, when possible.
- A batch context is a simple structure that can be created and updated manually in a relational database table created for this purpose, at least for testing and pilot projects. The user then simply adds a batch identifier, and start time when a batch starts, and stop time when it ends. You then connect to the table using the ODBC SimApi.
- A fourth option is creating the batch node from data inside the data source directly. In this case you read the data from the data source, and figure out when a new batch starts, and populate the node with batches (similarly to how Batch Context Generator works, but directly in your data source).

7.1.1 Creating a batch context from unit batch nodes in the data source

When the data source has many units but with no common batch node, the batch node can be created in the data source (not in SIMCA-online) by using the first start time of all units and the last stop time. The batch ID must also be the same in all units for this to work.

Although this sounds simple the algorithm can get rather complicated, especially if several exit criteria exist. The algorithm must also keep the batch “alive” when the batch is in-between units.

7.1.2 Use aggregation nodes in Batch Context Generator

SIMCA-online 17 introduced Aggregation nodes to create batch nodes that span multiple units or phases.

An aggregation node is populated with batches by querying its sub-nodes for batches and aggregating those to find the earliest start-time and stop-times. Batches are populated as the node is being used by project configurations. Learn more in the SIMCA-online help.

7.2 Prefer single, simple tags in phase execution conditions

A phase execution condition can use an expression using values of tags and logical and arithmetic operators and to determine if a phase is running or not. For example ValueTag(“odbc:node:flow”) > 0.55.

For simplicity and easier troubleshooting, it is recommended that you instead use tags that clearly show the state of the process in phase execution conditions. These tags can for example show if a certain valve is open or closed, or if a unit is active or not. You can also create tags with the sole purpose of being used in SIMCA-online so that you then can use conditions like TextTag(“odbc:node:phasetag”) ==”Phase1” to determine if the phase should be active.

7.3 Avoid using a SimApi that only supports current data, but not historical data

Some SimApis only support reading current data and not historical data. The OPC DA SimApi is such an example.

Having access to current data only means that the server cannot predict past data, or catch-up an already running batch when the execution is started. The server also cannot catch up if it falls behind after being overloaded. SIMCA-online automatically switches between real-time data and historical data as needed.

A data source that only provides current data, but not historical data can work for continuous projects in SIMCA-online. For batch project execution, historical data is required.

7.4 Modelling best practices

Data modelling in SIMCA is out of scope for this document. Consult with a Sartorius data scientist for guidance and best practices for multivariate model creation.

Here are some things to bear in mind and some tips:

- When developing models in a project in SIMCA it is useful to make sure to test these on batches you expect to encounter in real-time in SIMCA-online. This includes batches that are incomplete during evolution - for example 10% through a phase and batches with missing data.
- Consider how you use generated variables and Python preprocessing variables - generated variables are more complex to troubleshoot in SIMCA-online than in SIMCA. Especially when combined with shifting if used for a maturity.
- Do use SIMCA's Python scripts for creating Control Advisor forecast models, and testing and validating them in SIMCA before using in SIMCA-online.

Also see:

- Monitoring variables excluded from evolution models in SIMCA-online (Q929) (sartorius.com)
- Monitor variables without model influence in SIMCA-online (Q914) (sartorius.com).

7.4.1 Tip: create your own maturity generated variable instead of $Time

It is always best to use a real maturity variable in your process when available. If there isn't one, automatically generated time, $Time can be used. Learn more about 4.1.6.

You can also create your own generated variable and use that instead of $Time. In this example we create a new variable that will be the number of *days* since the phase started:

1. Add a generated variable based on the $time variable: divide the $time variable so it corresponds to parts of days.
2. Change the name of the variable to Age(days).
3. Edit the model, exclude $Time and change Age (days) to a Y-variable. Remove Smooth and Shift for the Age (days) variable.
4. Fit the model and save it.
5. Upload the project to SIMCA-online and configure it.

7.4.2 Best practices for Y-variable treatment smoothing, shifting and normalization

The y-variable used in an evolution model can be either time or maturity. In SIMCA, the y-variable can be configured in the model to be smoothed, shifted, and normalized. These options are described in detail in the SIMCA help.

For SIMCA-online, the setting Normalize should normally not be used. Using it would mean that batches look one way in real-time, but when the batch is finished the entire batch will be normalized and thus might change.

If you do use smoothing and shifting, be aware that the observations in the data source will not exactly match the ones that end up being used by SIMCA-online because the server will apply smoothing and/or shifting to your evolution level data. This also means that alarms triggered in real-time for a certain maturity value might be smoothed and shifted to a different maturity value when the batch has finished. This can result in alarms looking incorrect when looking at plots for finished batches. Learn more in the help topic ´About SIMCA-online alarms´.

Values for a maturity variable must always be strictly monotonic (increasing or decreasing) for a phase model. SIMCA-online and SIMCA ensures that this is always the case, regardless of the y-variable options mentioned above as described in 6.10.

7.4.3 Cropping in SIMCA-online

Cropping is set up in the SIMCA project and the purpose of it is to exclude observations so that they are not predicted, based on certain criteria. See the SIMCA documentation for more information.

Cropping the first N, last N, or observations satisfying a condition are supported in SIMCA-online. However, please note:

- Cropping first observation will cause a delay before data shows up in SIMCA-online when phases start because of the cropping.
- Cropping last N can be confusing in SIMCA-online, because cropping happens continuously for each observation as the model is executed. This means that N observations will always be cropped before anything shows up in plots. Observations will simply be delayed N observations before they show up in plots. This is since SIMCA-online does not know in advance when a phase or batch ends. For this reason, it is recommended that you avoid crop Last N.

In general, manual trimming of the dataset in SIMCA and the use of phase execution conditions in SIMCA-online to control phase execution, is a better alternative to cropping. This is because it is hard to create robust cropping rules in SIMCA that will work on future batches.

Cropped observations are logged to the server log as an [Information] event after the batch is finished (search for “Cropped observations for batch” to find it).

Also see the help topic ‘Trimming-Winsorizing’ in SIMCA-online for related functionality.

7.5 Changing execution interval for models

A model is built using a specific interval between observations in SIMCA. If you change this interval in SIMCA-online this can cause issues because some of the features in your model may not work because you effectively change the speed of you’re the model’s monitoring of the process.

Here are some examples where you shouldn’t change the execution interval:

- $time as the maturity variable in an evolution model.
- You significantly change the execution interval so that a phase becomes much longer or shorter than it was built. This results in the incorrect number of observations being fed to the models.
- You use lags based on a fixed number of observations.
- Duration variables are used in batch level models.
- Cropping that removes observations is used.

7.6 Qualitative model variables should be mapped to text tags

SIMCA-online supports qualitative variables in the SIMCA model. Each variable should be mapped to a **text tag** in the data source. SIMCA-online then encodes the qualitative data into additional expanded variables in the same way SIMCA does.

If qualitative variables are read as numerical values SIMCA-online interprets numerical values as already encoded values, but this is unsupported because it requires knowledge about how SIMCA encodes qualitative variables.

7.7 How can an external system know if a project configuration is executing correctly?

The value of the maturity variable or the time of the latest predicted observation in a project configuration can be used to determine if a project configuration is working correctly.

Use Write Back as described in 7.8 to automatically export the relevant values to a data source. In particular, the output variable Maturity in the batch evolution level could be written back, and then the value of that variable is increasing when a phase is executing.

7.8 Write Back – how to push values from SIMCA-online to an external data source

SIMCA-online has built in **Write Back** functionality that can be used to write back data to a data source during execution of the configurations. Write-back can be used to write back maturity, alarms, Control Advisor set points and other information. Many SimApis including PI, OPC, and ODBC supports write back. Write Back is configured in each project configuration.

Refer to the help for more information.

7.9 SIMCA-online Web API

The SIMCA-online server's Web API provides information about configurations, projects and models, and output data including predictions. The web API is used by the SIMCA-online Web Client.

As all APIs, it requires developer expertise to use, as well as SIMCA-online experience to know what the data exposed represents. Learn more in Using the SIMCA-online web API for developers (Q907) (sartorius.com).

Sartorius Stedim Data Analytics AB
Östra Strandgatan 24
903 33 Umeå
Sweden

Phone: +46 90-18 48 00
www.sartorius.com

The information and figures contained in these instructions correspond to the version date specified below.
Sartorius reserves the right to make changes to the technology, features, specifications and design of the equipment without notice.
Masculine or feminine forms are used to facilitate legibility in these instructions and always simultaneously denote all genders.