ObSys Manual – Part 1

ObSys is a suite of applications for Windows PCs, consisting of a controller application called ObServer and a range of viewing applications including ObView, AlarmManager, and DataManager.

ObServer contains North's interface technology, block-based programming language, and easy-to-use information services. ObServer can work as a stand-alone controller, or alongside other North controllers and display systems to create a larger control or monitoring solution.

ObSys Manual – Part 1 covers the controller application - ObServer.

This document relates to ObServer Version 2.0, part of ObSys

Please read the *ObSys Tutorial – Part 1* alongside this document, available from *www.northbt.com*

Contents

What is ObSys? 5

PC Hardware for ObSys 6
- System requirements 6

What is ObServer? 7
- Interface Technology 7
- Programmable Control 7
- Information Services 7
- Typical Use 8

Quick Start 9
- Install ObSys 9
- Set-up the local ObServer 9
- Set-up a remote ObServer 9

ObSys Software Licence 10
- Interface Licences 10

ObServer 11
- Interface Drivers 11

Functional Organisation 12
- Functions 12
- Settings 12
- Organisation of this Manual 12

Platform Information 13
- Operating System Information 13
- Data Storage 13
- Reset/Restart Information 13
- Software Versions 13
- Debug Recording 13

Interfacing ObServer to other Systems 14
- Started Interfaces 14
- Installed Drivers 14
- Interface Licences 14

Transferring Values 15
- Reading the Value 15
- Writing the Value 15
- Transfer Tasks 15

Essential Data 16
- Structure 16
- Pages 16
- Objects 16
- Value Reading and Writing 17
- Adjustability 17
- Alarm Monitoring 17
- Value Logging 17

Controlling using Time and Date 18
- Calendar 18
- Timers 18
- Profilers 18

ObVerse Cause-and-Effect Strategy 19
Properties 19
Modules 19
Editing ObVerse 19

Alarm Basics 20
Format 20
Alarm Flow through ObServer 20
Alarm Translation 20
Alarm Prioritisation 20
Alarm Delivery 21
History of Alarms 21
Alarm Stores 21
Emailing Alarms 22
Alternative Alarm Destinations 22

Communicating with other North IP Devices 23
Key Security 23
Checking Communications 23

Security Server 24
User Information 24
Groups 24
Use by other Tasks 24

WebView Server 26
Network Settings 27
Security Settings 27
Customising WebView 27

Telnet 28
Establishing a Session 28
IP Information Service 28
Query Response Service 28

Updating ObSys 30

Object Specifications 31
Example Object Reference 31
Device Top-Level Objects 31
Configuration 33
Platform Information 35
Operating System 35
Data Storage 36
Last Restart 36
Software Versions 37
Software Version Detail 37
Recording Configuration 38
ObServer Start-up 39
Interfaces Configuration 40
Installed Drivers 40
Interface Licences 40
Object Aliases 41
Object Alias Entry 41
WebView Server Configuration 42
WebView Server Network Interface 43
WebView Server Alarms 43
WebView Server Alarm Lists 43
WebView Server Security 44
WebView Server Security LDAP Setup 46
WebView Server Security LDAP Group 47

North IP Devices Configuration 48
North IP Device Configuration 49
Essential Data Configuration 50
Essential Data Page Configuration 51
Essential Data Object Configuration 53
Essential Data Task Control 55
Time Control Configuration 56
ObVerse Processor Configuration 57
ObVerse Object Modules 58
Security Server Configuration 59
Data Transfer Configuration 60
Data Transfer Task 60
Alarm Translator Configuration 61
Alarm Prioritizer Configuration 63
Alarm Delivery Configuration 64
Alarm History Configuration 65
Alarm Store Configuration 66
Alarm Store Archive Configuration 66
Alarm Email Configuration 67
Alarm Email Destination Configuration 70
Email Address Setup 71
Telnet Setup 72
Essential Values 73
Essential Values Page 73
Time Control 74
Calendar 75
Calendar Exception Date 76
Timer 77
Profiler 78
ObVerse 79
Security Server 80
Security Server Group 80
Security Server User 81
Data Transfer 82
Transfer 82
Alarm Translator 83
Alarm Translator Entry 84
Alarm Prioritizer 85
Alarm Prioritizer Entry 85
Alarm Delivery 86
Alarm Delivery Destination 86
Alarm History 88
Alarm Store 88
Alarm Emailer 89
Alarm Emailer Destination 89
North IP Devices 90

Appendix A – ObServer.ini File 91

Appendix B - Ethernet/IP Protocols 92

Warranty 93

ObServer Versions 94

What is ObSys?

ObSys is a suite of applications that run in Microsoft Windows, which in turn runs on PC hardware. It consists of the following main applications:

ObServer

A building controller containing North's interface technology, block-based programming language, and easy-to-use information services, including WebView.

ObView

A viewing application that communicates to, and through, ObServer. ObView is used for our Engineering Software. The engineer can make new views of their site and equipment.

AlarmManager

An alarm viewing application that communicates to, and through ObServer. AlarmManager allows multi-user viewing of alarm lists, including acknowledgments.

DataManager

A logging application that communicates to, and through, ObServer. DataManager stores logs in CSV files and can analyse values to determine averages and standard deviations.

This document *ObSys Manual - Part 1* covers the use of ObServer.

The document *ObSys Manual - Part 2* covers the use of ObView, AlarmManager, and DataManager applications.

PC Hardware for ObSys

ObSys can be installed on various types of PC hardware. The engineer chooses the hardware to match their requirements:

Hardware Type	Typical Use
Desktop PC	Permanent device for displaying system data Applications used: ObServer, ObView, AlarmManager, DataManager
Laptop PC	Temporary device for displaying system data Temporary device for engineering system setup Applications used : ObServer, ObView
Embedded PC	Permanent control device for monitoring and controlling Applications used : ObServer
Server PC	Permanent control device for monitoring and controlling Applications used : ObServer

System requirements

ObSys requires a current supported version of Microsoft Windows, including: Windows 7 SP1 or later for client PCs; and Windows 2008 R2 or later for servers. Virtual environments (VMware, Hyper-V) are only supported when direct access to a USB port can be provisioned.

The minimum systems requirements for the version of Windows installed are typically sufficient for ObSys.

ObSys additionally requires:

- 300MB of available hard disk space
- USB 1.1 compatible Type A port for ObSys Licence Device.

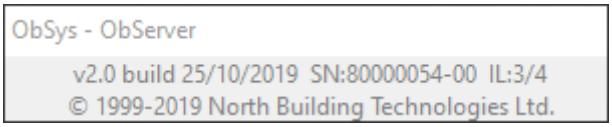
Depending on your use of ObSys, you may also require:

- Additional hard disk space for mimic files
- Serial port (COM)
- Ethernet network adapter
- Sound card and speakers.

What is ObServer?

ObServer is the more powerful of North's building controllers, which also includes Commander. The controllers contain North's interface technology, block-based programming language, and easy-to-use information services. ObServer can work as a stand-alone controller, or alongside other North controllers and display systems to create a larger control or monitoring solution.

The ObServer window, if visible, shows the version of ObServer; its serial number and interface licences (used/total); and a copyright notice. If communication port debugging is enabled, the build text and copyright text is overwritten temporarily with the transmitted and received characters for the port as they occur.



Interface Technology

ObServer includes North's interface technology. ObServer can access values from thousands of different third-party systems in a common way, using North drivers. This ability allows ObServer to pass data between different systems and enables different sub-systems within a building to be fused together to form a single, coherent system.

Programmable Control

ObVerse is North's block-based programming language. It is available in all North controllers. Although it is easy to use, it provides real flexibility during engineering, allowing the engineer to incorporate design changes with minimal effort. Date and timer functions are standard, along with feedback control and logic.

Information Services

ObServer supports North's standard protocol, allowing communications with other North products such as Commander.

ObServer has a built-in Web server called WebView. It supports both automatic displays of Essential Data. It also allows the engineer to create pages using graphics and dynamic items. Logs can also be viewed using the web server.

The engineer can extend information services by using North's driver technology: values from within ObServer can be made available to, say, BACnet and Modbus devices.

Typical Use

ObServer's powerful feature list means it can perform a wide range of tasks. The following are typical applications…

Single-device Integration of Systems

Creating an interface between two or more systems is easy with ObServer. Interlocking an air-conditioning system with an underfloor heating system is a common problem, although the possibilities are endless.

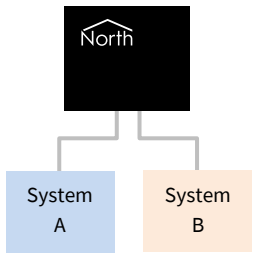


Fig. 1 Integration

Building-wide Management System

Connecting different systems, in different locations, to a common LAN (usually an intranet or VPN) allows users to manage these systems using a web browser anywhere on the LAN and adjust settings as necessary.

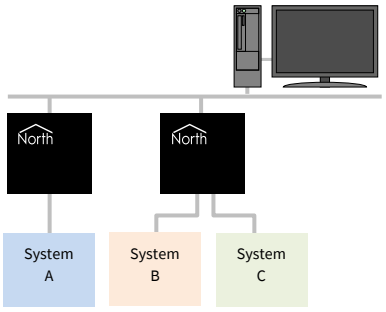


Fig. 2 Building Management System

Remote Alarm Management System

ObServer can be part of a remote management system that connects different systems in different buildings and provides information back to a central display. The central display can be web-browser based or can be North's ObSys management software.

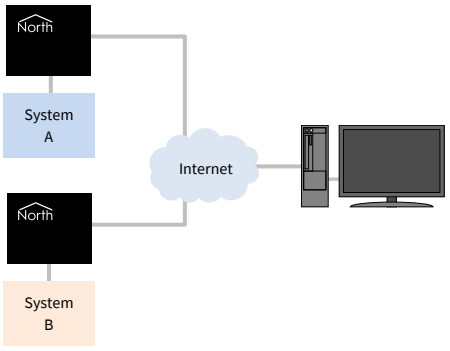


Fig. 3 Remote Management

Quick Start

Install ObSys

Install the North ObSys package onto your PC platform. It is available to download from *www.northbt.com*

Set-up the local ObServer

When ObSys is installed, the Engineering software is also installed, which can be used to engineer the local ObServer.

Run the **Start Engineering** application.

Select **ObServer** then press **Scan** to rescan the local ObServer objects. The Site Label should appear in the list.

Refer to the *ObServer Tutorial* document for more details, including full walk-throughs of all the main features.

Set-up a remote ObServer

If ObSys is installed on a different PC, for example an embedded PC, then it is possible to use a second PC as engineering software.

Run the **Start Engineering** application.

Select **North IP Devices**, then press **Scan**. Once scanning is complete, a list of accessible North devices should appear, including PCs running ObSys

Select the required **ObSys device**, and press **Scan** to discover the remote ObServer objects. The remote Site Label should appear in the list.

ObSys Software Licence

North can supply an ObSys Licence Device, sometimes called a dongle, to control how ObSys runs.

If no Licence Device is fitted, ObSys will run in '12-hour' mode. Start ObSys, and it will continue to run until you stop it, or until 12-hours has passed. You can restart ObSys if necessary, although automatic restarting is not allowed. This mode is suited to using ObSys as an engineering tool for other North devices. Some free-to-use drivers require no licences to start, including Modbus, BACnet, and Zip, and this mode allows these drivers to be used for testing.

If a Licence Device is fitted, ObSys will run permanently. The Licence Device can contain Interface Licences to allow interfaces to be run within the PC. This is suited to using ObSys as an embedded controller or the top-end of a system.

Interface Licences

North supply ObSys with a certain number of interface licences, which control the number of interfaces that can be used at a time. Each interface usually requires one interface licence before it will start, although certain drivers require zero licences.

It is possible to upgrade the number of interface licences available within the attached ObSys Licence Device. Simply call North support while you are on site to activate a new licence.

Refer to the *Interfacing ObSys to other Systems* section for more details.

For more details on the Software Terms and Conditions, please visit *www.northbt.com/go/eula*.

ObServer

The main operating features are provided in ObServer.exe. This is loaded onto the PC during ObSys installation.

These features evolve over time as newer ideas are incorporated, but contain the following areas:

- General platform information, such as firmware versions
- LAN port setup, including IP addressing and time synchronisation
- Real-time clock and time zone, including daylight saving
- Interfacing to other systems
- Transferring values between systems
- Database of essential values
- Controlling using time and date
- ObVerse cause-and-effect strategy
- Alarm handling, including translating, prioritising, filtering, delivery, and audit trails
- Communicating with other North IP devices
- Security server
- Telnet server
- WebView server

Interface Drivers

Interface drivers are provided in individual software files, to simplify updating. Interface drivers can be updated as newer versions are produced.

Popular interface drivers include:

- Modbus – including Modbus over TCP/IP and serial line protocols
- BACnet/IP – as both a client and a server
- ZipMaster – to communicate and control a North Zip network
- JSON – an API for third-party software and web services
- Echelon LonWorks – for accessing LonWorks compatible and LonMark certified devices
- Kentec – one of many available fire panel interfaces
- Galaxy – one of many security interfaces
- Crestron – one of many user-interface connections
- APC – for UPS status

For the latest list of drivers available for ObSys, visit *northbt.com/go/drivers*.

Functional Organisation

The different areas of function within ObServer (or any other North product) are organised, and therefore engineered, in the same way - using objects.

This object system, which has been common in all North products, have used in products for many years, and is extremely simple to understand, and yet gives a flexible, extendable system.

The object system has two simple rules:

- Each system or device appear as an **object**
- Each object can either **contain** other sub-objects, or have a **value** that can read (and usually written)

ObServer appears on the LAN as an object; it contains sub-objects that represent the sub-functions within ObServer; each of these contains sub-objects representing sub-functions; and so on until we get to sub-objects that have values, which can be viewed and possibly adjusted.

Functions

Functions within ObServer are represented by 'container' objects. A container object can either have a pre-determined list of sub-objects (called a fixed container object) or can have a variable list of sub-objects (called a variable container object).

North's object engineering software shows the contents of a container object, and allows the engineer to navigate up, down and across these container objects. It also allows the engineer to scan and find the sub-objects within any variable container object.

For example, ObServer is a variable container object with its contents changing as interfaces are started and stopped.

Settings

Settings within ObServer are represented using value objects. The value of an object can be viewed by engineering software or by other devices; some value objects can also have their value adjusted.

North's object engineering software shows the current values of these objects and allows the engineer to make changes to objects that are adjustable.

For example, ObServer's Site Label is a value object that can be viewed and adjusted.

Organisation of this Manual

The following chapters in this manual describe the functional areas within ObServer. These areas appear as sub-objects of ObServer within the object engineering software, ObView.

Refer to the *ObSys Tutorial – Part 1* for more help on using the object engineering software to engineer ObServer, including full examples.

The final chapter, *Object Specifications*, details the contents to each of ObServer's container objects. It lists each of the fixed and variable container objects and describes the type and attributes of each value object.

Platform Information

The *Platform Information* object provides general information about ObServer and the PC.

Operating System Information

ObServer allows access to the operating system version, along with access to memory availability and disk space availability

Data Storage

ObServer typically stores any changes made to disk if necessary. Rather than continuously writing to disk, this occurs periodically. This can extend the life of disk drives, especially if flash drives are used for storage.

Reset/Restart Information

ObServer holds information about the last restart to occur, including the assumed cause. This information is sometimes useful when solving problems.

Software Versions

ObServer's software is made up from a collection of different modules. The software version and build date for each module is available, again to help when solving problems.

Debug Recording

ObServer can write information to a text file to assist with debugging. The North support team may request that this information is captured. Different types of information can be recorded:

Information – general start-up information
Objects – object request/reply messages between modules
COMs – bytes being sent and received on COM ports
Internal Comms – fast synchronous messages between internal modules

ObServer can also display traffic from a chosen port on its window, instead of the build and copyright text.

Interfacing ObServer to other Systems

One of ObServer's strengths is its ability to interface to several systems, including those from other manufacturers. Drivers are the software convertors that allow interfaces to work.

The driver required for a particular interface is selected from those installed in ObServer. ObServer starts these interfaces when they are specified, and whenever ObServer is restarted. Drivers can be stopped if they are no longer needed.

Started Interfaces

ObServer allows up to ten interfaces to be started and used at the same time, although interface licences may limit this.

Installed Drivers

The list of drivers currently installed within the ObServer can be viewed – and the names copied to the current interface list to start an interface using a particular driver. The same driver can be used on several interfaces concurrently. For example, ObServer can start interfaces to two separate Zip systems, each with 16 Zip modules.

Other drivers can be installed at any time. For details on how to do this, see *Updating ObSys*.

Interface Licences

Each ObSys Licence Device is supplied with a certain number of interface licences. It is possible to add more licences on site, although a phone call is required to check/verify settings.

Order Codes for ObServer are as follows:

North Order Code	Interface Licences
OBS/0	0
OBS/1	1
OBS/2	2
OBS/3	3
OBS/4	4

Order Code for an additional interface licence

North Order Code	Interface Licences
UPGD/1	1

Call North support on +44 (0) 1273 694422 for help with adding more licences.

Transferring Values

Transferring values from one place to another is the simplest form of integration. ObServer supports this with Data Transfers – where each transfer reads from one place and then writes to another.

ObServer supports up to 1000 Data Transfers.

Each transfer contains a source object reference, a source read rate, the last value read, a destination object reference, and a destination write rate.

ObServer's object system means that any value available from ObServer, including configuration objects and other attached systems, can be used with transfers.

Reading the Value

The source read rate should be configured to suit the type of value being transferred. For example, room temperatures are typically read every minute or so, and operating set points read every 15 minutes.

Setting the source read rate to 'as soon as possible' (ASAP) instructs ObServer to read the value every time the transfer is processed – if all transfers have a read rate set to ASAP, the reading occurs sequentially.

Remember to consider the time needed to read values because they do not occur instantaneously. If it takes 1 second to read a value from an external system, then it will take 100 seconds to read 100 values. Even if you specify the 100 values read every minute, the best that could occur would be every 100 seconds.

Writing the Value

Rather than read-then-write continuously, each transfer will usually only write the value to the destination object when the value changes – called change-of-value (COV) writing. This eases the workload within the destination system and saves time and effort within ObServer.

It is also possible, to have the transfer perform a background write periodically. Occasionally it is useful to have the last value re-written in this way – for instance, if a system forgets values after a power failure, or if a local adjustment needs to be reset after a certain time.

Remember COV writes always happen, so consider if a background write is needed and how frequently it needs to occur. Writing the value takes precedence over reading, so set the write rate to a longer time than the read rate. A background write only usually needs to occur every few hours.

Transfer Tasks

By default, ObServer works through the list of transfers sequentially – after finishing transfer 1, it starts transfer 2, and so on, until it reaches the last transfer, after which it starts work on transfer 1 again.

This is simple to understand but can be quite slow. If values are from several different systems, it is possible to allow the Data Transfer system to try to perform two or more transfers simultaneously. However, this feature should be used with care, and should be fully tested before handover, because if several tasks are reading from or writing to the same system, overloading of the system's communications can occur.

Essential Data

Several different modules within ObServer need access to a database of values – called Essential Data.

Essential Data allows the engineer to configure a list of values that other ObServer modules can then distribute – as web pages, as BACnet points, as Modbus points, on Zip displays…

Structure

Essential Data consists of a list of configurable pages, each of which has a list of configurable objects. In total, 1280 database objects are available. The engineer may choose from a mix of page and object counts: 80 pages of 16 objects (default), 128 pages of 10 objects, 40 pages of 32 objects, or 20 pages of 64 objects. Older versions of ObServer only support 30 pages of 16 objects (total 480 objects).

An Extra Data driver is also available to extend the number of database values available within ObServer.

Pages

The engineer configures a label for each page. Other ObServer modules use the label when displaying the value from within the database – if a page has no label, then the page does not appear in label-based hierarchical views, such as those available as web pages.

Each page has a remote object prefix. This allows objects within the page to reference remote data relative to this prefix. When un use, if any object in the page fails to read its remote object, Essential Data assumes all the objects will fail, and therefore saves time and effort. It also allows easier copying of pages when several devices of one type are used.

Objects

For each object within a page, the engineer configures a label. The label determines whether the object appears – if an object has no label, the object does not appear in label-based hierarchical views.

The engineer configures the value type for an object. The type determines whether a value is made available via a particular protocol – for example, profile objects are not made available over the Modbus protocol. The type also controls the appearance of the value on hierarchical views, including when the user adjusts the value.

Type	Use	Example
Text	Text string of ASCII characters	Kingston Lane
NoYes	Binary state: 0 or 1 meaning No or Yes	1 *(means Yes)*
OffOn	Binary state: 0 or 1 meaning Off or On	0 *(means Off)*
Number	Integer, positive or negative	-12300
Float	Floating-point value, with decimal places	21.54
Times	List of on-off times	08:30-12:00,13:00-17:30
DateTime	A moment in time	25/12/15|12:30
Date	A particular day	01/01/19
ENum	A number representing something	3 *(means Automatic)*
Profile	List of time-value changes	09:00=21,12:00=20,18:00=19

Value Reading and Writing

The engineer can configure the database object to read periodically from, or write periodically to, any object accessible to ObServer. This allows slow-access data to be collected and ready for speedy delivery when needed.

As a new value is read, the object also recalculates the alarms state, sends alarm messages if necessary, and logs the value.

ObServer considers each object in a linear fashion, beginning with the first object on the first page. If it is time to read (or write) the value, this is started; if not, the object is passed over. When the final object has been checked, ObServer starts from the first object again.

If all objects are set to read ASAP (as soon as possible), ObServer performs each in turn. If it takes one second to read an object, and there are five hundred objects to be read, the whole process will take five hundred seconds. Please consider the read rate of objects, allowing time for the object to perform its remote action.

If an object is set to read periodically, and is set to adjustable, then an adjustment will cause three attempts at writing the new value to the object, before reading is resumed.

Adjustability

By enabling adjustability, and configuring value high and low limits, the engineer can control whether the user can modify the value of an item – and if this occurs, the value will be written to the associated object.

Alarm Monitoring

When a value is updated, it is compared against the value high and low limits, and an alarm message generated as necessary if its alarm state changes.

Value Logging

If necessary, Essential Data can log the value periodically, to create a history of the value over time. Other devices can access the data log.

Essential Data uses the configured log rate to sample the value – if the log rate is set to 15 minutes, one day will use up to 96 readings.

A total of 200 objects can have logging enabled, each of which can record 2000 samples.

If the value has not changed from the last sample, then rather than waste sample space, the log records start-and-end sample times to save space. Using this method, 15 minute sampling will use a maximum of 96 readings per day, but could be considerably less if, for example, the value is an alarm state.

Controlling using Time and Date

Timers allow users to control when things happen in the day, and when they do not. Using timers can save energy, while keeping the occupants happy. ObServer supports time control using the calendar, timers, and profilers within it's Time Control area.

ObServer's single calendar determines today's day-type. Today's day-type is used by 20 timers to determine which of their on-off time-lists to use, and by 20 profilers to determine which of their time-value profiles to use.

Calendar

ObServer supports 10 different day-types: one of them is an off day-type, leaving nine to configure. They are numbered 0 (off), and 1-9.

If you have a centralised calendar elsewhere on the system, ObServer can request the day-type from this – in which case you need to specify the object reference of the current day-type in the central calendar.

If you are calculating the day-type in ObServer, it works in the following way. The calendar determines whether today's date is an exception date – if it is, then that exception day-type is used; otherwise the day-type of the standard day-of-week is used.

The calendar re-calculates the day-type every minute, based on the day-types and the exception dates.

Any North device that can access ObServer's objects can also access the calendar objects. The Calendar is also available via the Web Server.

Timers

ObServer uses timers to control off/on processes. Each timer produces an off or an on state, which can be accessed by other tasks (**Error! Reference source not found.**).

Each timer has a time-list for each of the possible day-types and uses that time-list on days that have that day-type. A time-list is a list of on-off time-periods. The timer re-calculates the state of the timer every minute.

Any North IP device can access the Timers. They are also available to view and edit via the Web Server.

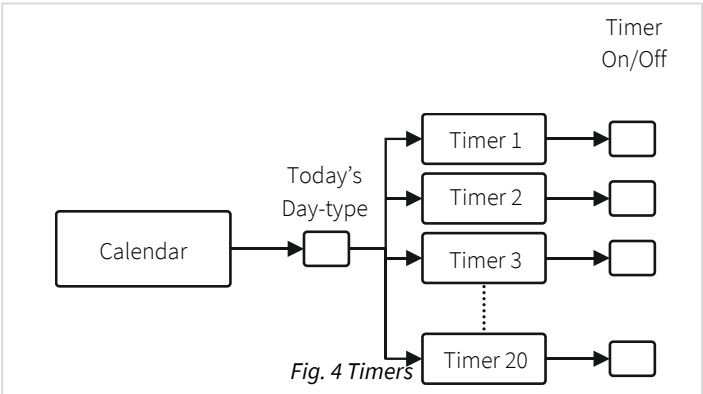


Fig. 4 Timers

Profilers

ObServer uses profilers to control a variable value throughout the day.

Each profiler has a list of profiles, one for each possible day-type, and uses that profile on days that have that day-type. Each profile is a list of time-value pairs, each pair representing a change-point. When the current time matches a change-point time, the value is set to that of the change-point. The value can also be set by other tasks: the value then acts as a temporary value, until the next change-point occurs.

WebView can be used to view and adjust the calendar, timers, and profilers.

ObVerse Cause-and-Effect Strategy

Sometimes you need to do more than simply transfer a value from one object to another – you need to calculate something, delay something, or perform a more complex function on a value. North provides this flexibility with ObVerse, a cause-and-effect programming language.

ObVerse consists of a range of modules. The engineer selects modules and links them together to perform a desired strategy.

ObVerse strategy runs in an ObVerse processor within a device.

ObVerse processors come in two types:

- Standard Processor – with logic, maths, and control modules
- Advanced Processor – with the same features found in a standard processor plus extended maths and logic, display, application execution, directory and file services, and user-defined modules.

ObServer has four ObVerse standard processors.

ObVerse strategy is made up from properties, modules, and comments.

Properties

ObVerse properties are containers for storing data values. They can carry a value from one module to another, or between the processor and other functions within ObServer.

Properties have a data type, to define the type of value they hold – like a number or a text string.

Properties sometimes hold values passing only between modules in the same processor. In ObVerse, we call these private properties, as their value is private to the processor.

Properties sometimes hold values passing between the processor and another module within ObServer. In ObVerse, we call these public properties, as their value is publicly available as an object.

Modules

Modules calculate values. They take one or more inputs and calculate one or more outputs.

Different modules are available to perform different operations. The range of modules supported depends on the processor.

In ObServer's ObVerse standard processor, modules perform the following types of operation: Maths, Logic, Control, Timers, System, and Object.

Editing ObVerse

You can create and edit ObVerse strategy using North's ObvEditor application, installed as part of the ObSys software. ObvEditor provides drag-and-drop graphical editing of ObVerse, uploading and downloading of ObVerse strategy, and run-time monitoring of the strategy within the processor.

For further information on ObVerse in ObServer, including properties and modules, refer to the *ObVerse Manual – Standard Processor*.

Alarm Basics

As well as responding to object requests, ObServer can process alarm messages – messages that contain text information about events that have occurred.

Different users demand a wide variety of alarm processing – ranging from simple histories of alarms, to lists of critical events that require user acknowledgement, to sending messages to mobiles phones or by email.

Format

All North-format alarms are text-based, and have the following fields:

System and Point – identity of the system, device, and point that has changed

Condition – condition that the point has changed to

Priority – importance of the notification, between 1 (most) and 9 (least)

Date & Time – date and time that the condition changed

Refer to *Object Specifications* section for details of alarms sent by each module.

Alarm Flow through ObServer

When an alarm is generated, either by some function within ObServer or from one of the interfaces, they flow between modules within ObServer in a particular way (**Error! Reference source not found.**).

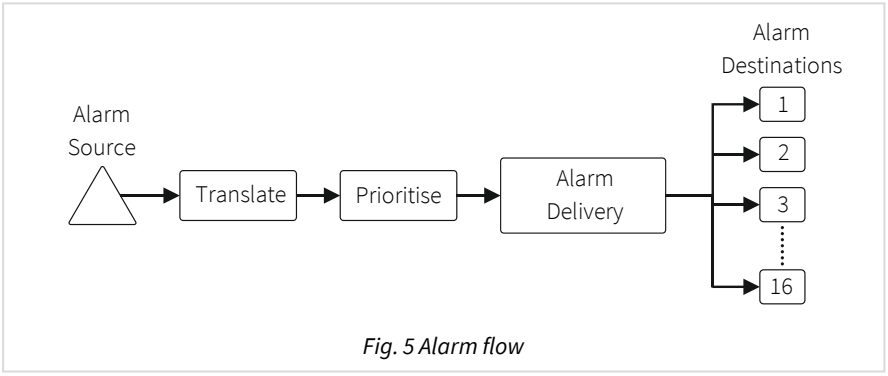


Fig. 5 Alarm flow

Alarm Translation

All alarms arriving or being produced by ObServer are passed to the Alarm Translator.

The Translator checks for text within the alarm that matches any of its translation entries. If any are found, they are replaced with the translated text.

The Translator first searches its own database of 1000 text phrases to translate individual fields of an alarm message, then searches a *comma-separated value (csv) file* to translate the entire alarm message. If no matches are found, then the alarm message remains unchanged.

As examples: simple 'Device 1' text can be translated to 'Ground Floor Corridor'; capitalised words can be translated to the correct upper/lower case.

Alarm Prioritisation

After Translation, alarms are passed to the Prioritiser.

If the alarm text matches a re-prioritiser entry, the alarm priority is replaced with a new priority.

Whether re-prioritised or not, if the alarm has a non-zero priority it is passed to Alarm Delivery – otherwise it is deleted.

As examples: any alarm with the word ‘Fire’ in the condition can be re-prioritised to priority 1; any condition matching ‘Ok’ can be reprioritised to level 4 – information only

Alarm Delivery

After Translation and Prioritisation, all alarms are passed into the Alarm Delivery area. Its role is to distribute copies of the alarm to other alarm processing destinations, depending on the value of the alarm fields.

ObServer’s alarm delivery supports up to 16 destinations.

ObServer has six built-in alarm processing destinations: Alarm History, four Alarm Stores, and Alarm Emailer. Other destinations are available within other North devices and can appear within drivers and external systems as interfaces are started.

Filtering Delivery

Besides having a destination enable, alarms can be filtered, so that only alarms with a priority in a certain range are sent to a destination. This allows only the relevant alarms to be sent to certain destinations.

Filtering can also cover text matching and not-matching, to force alarms from systems to be sent only to one destination, for example, while all other alarms are sent to a different destination.

Delivery to any one of a Group of Destinations

Sometimes users may need alarms sending to any of several destinations. For example, alarms must be sent to any one of a range of on-call users (depending upon which has their PC running).

Alarm Delivery supports each destination being a member of the any group, an indication that when one destination accepts the alarm, it does not send to any of the other members of the any group.

History of Alarms

Sometimes, all that is required is a list of the latest alarms. This can act as an audit trail or can be viewed more regularly to check on new alarm conditions. However, no user action is necessary, and as new alarms occur, old alarms are lost.

ObServer’s alarm history holds the last 500 alarms.

Alarm Stores

ObServer has four Alarm Stores. Each store holds alarms that require action by a user.

When an alarm arrives, the store holds the alarm as a new alarm, requiring action.

The user can, from the web page or an AlarmManager application, acknowledge that they have seen the alarm. It remains in the store but is marked with the date and time of acknowledgement.

The user can then delete the alarm from the store. It can then be added to an alarm archive file, which is a CSV file store on a disk.

Alarm Stores have privilege levels required for viewing, silencing (acknowledging) and deleting alarms in the store.

Each Alarm Store holds up to 4000 alarms. If the store becomes full, it will reject any new alarms until space becomes available by a user deleting older alarms.

Emailing Alarms

Some users want important alarms and event messages delivered directly to their inbox. ObServer has alarm emailing facilities to do this.

ObServer's alarm emailer supports 127 destination groups. Each group has its own filtering options, so only alarms from a predetermined system are emailed to the group's recipients, and an option to send in a text-only or friendlier HTML format. A group supports up to five email address recipients.

The Alarm Emailer can connect to an SMTP relay server that supports authentication, but not to a server that requires encryption. You can typically connect to an SMTP relay server on your local network (e.g. Windows IIS SMTP Server), from your Internet service provider (e.g. BT Internet, Virgin Media), or from a paid SMTP service provider (e.g. smtp2go.com, authsmtp.com, serversmtp.com, etc.).

SMTP relay servers that require TLS encryption (e.g. Gmail, Outlook, Office 365, AWS SES, etc.) are not directly supported. To access a service requiring TLS, use an SMTP proxy such as Windows IIS SMTP Server or Stunnel.

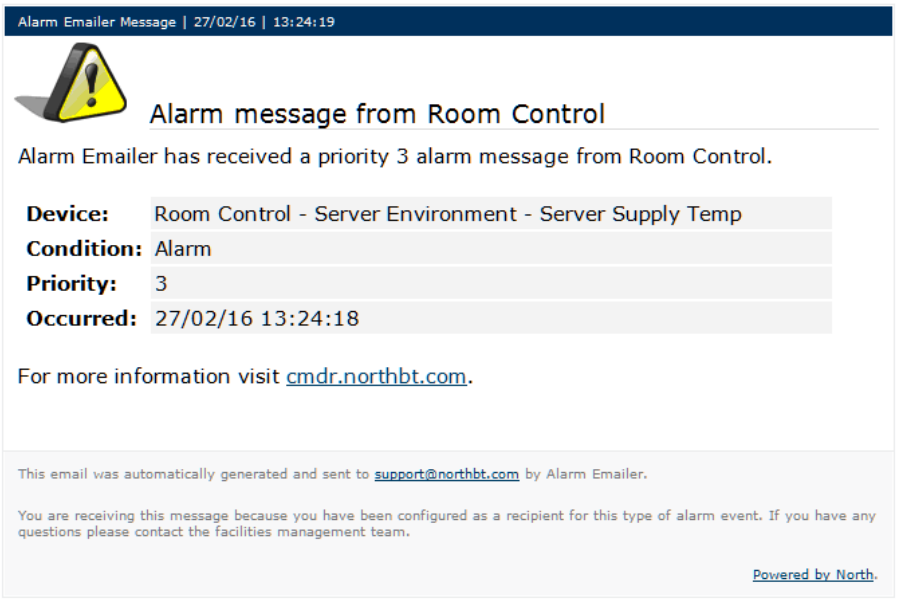


Fig. 6 Email from ObServer's alarm emailer

Alternative Alarm Destinations

By adding more interfaces, it is possible to add more ways of dealing with alarms, these include the following:

Driver	Use
AlmSet	Convert alarm text to an object write
AlmPrint	Print alarms to the default Windows printer
Printer	Print alarms to a serial printer
ESPA444	Pager and messaging systems support ESPA 4.4.4
GSMSMS	SMS messages to mobile phones, via a GSM modem

Communicating with other North IP Devices

All North devices that support Ethernet also support the North IP protocol – ObSys and Commander. The North IP protocol allows the devices to communicate efficiently using objects.

Key Security

It is easiest to allow all North devices on a network to communicate openly. However, it is possible to add an authentication key to a device. This will stop other devices being able to communicate with it unless they also know the device's authentication key. Each device could have a unique key – so Device A may be able to request values from Device B, but not vice versa.

Checking Communications

It is possible to configure ObServer to check that other North IP devices are still available and communicating and generate alarm messages when they are not.

Security Server

When ObServer is connected to a network, the engineer must consider security – to control who can and who cannot view and modify values, and who can engineer ObServer itself.

North products, including ObServer, that require user authentication have access to a central user database that holds user information. Remote doors could use this, for example, to ask the central user database for authentication when a user requires access.

ObServer has Security Server, its own user database – it holds information, including privilege levels, for different users.

ObServer's Security Server can support up to 2000 users.

It is possible for a task within ObServer to use a security server elsewhere in the system – for example in a Commander or within a different PC running ObSys.

User Information

Besides a name, each user has a user ID (or card) and a password – together these form a coded token. It is the coded token that is passed around a system – the password is never seen.

Each user is given eight privilege levels, one for each of eight different areas. The privilege level is in the range 0 (no security clearance) to 7 (maximum clearance).

Individual users can also be enabled or disabled. Users can be limited to access between certain dates – it is possible to see the last date that a security clearance was requested for the user.

Groups

Each user can be a member of up to three groups. Each group has a list of privilege levels, which act as a base level for users that are a member of the group. This allows a whole group of users to be controlled quickly and easily.

When groups are used, a user is enabled when any of the groups they are a member of is enabled (and the user is enabled).

The user's privilege level in a particular area is calculated as the highest level specified in that area from the individual user information, and any group they are a member of.

Use by other Tasks

A task that requires security clearance will have one or more access security objects. Each access security object has a two-digit value, which controls access to a particular feature – such as adjusting a value or viewing a page.

The two-digit value is made up of the area digit (1-8), followed by the minimum privilege level (1-7). The user must have a privilege value equal to (or greater than) that specified, in the area specified, before the task will allow access to the feature.

For example, if the minimum privilege level is '6' in area '2', then the two-digit value is '26'. If the value is set to '00', then no security checks are needed.

When the user wants to access the protected feature, the following occurs:

- The task requests identification from the user: this may be done by asking for a name and password; or by scanning a security ID card
- The task encodes the identification, and requests the privilege levels for the user with that identification from the Security Server
- The task receives the eight privilege levels from the server
- The task checks whether the user's privilege level in an area is sufficient to allow access, and if so will grant access
- The task may send an alarm message, indicating whether the user was allowed, or whether the ID was not recognised.

WebView Server

ObServer's web server, called WebView, builds pages with simple HTML, which the user can view in any browser, including those on mobile phones and tablets (Fig. 7). WebView is more powerful than the web server within Commander.

WebView creates a website automatically, with information from the following functional areas:

- Essential values including ExtraData values
- Calendar and timers
- Data logs from Essential value and DataManager
- Alarms from history and Alarm Stores
- Object editing

WebView also supports engineer-designed items. These are created, edited, and downloaded to WebView using an application called WebViewEditor.

- Mimics
- Templates
- User Objects

If necessary, it is possible to disable the web server completely, or disable the displaying of information from certain areas.

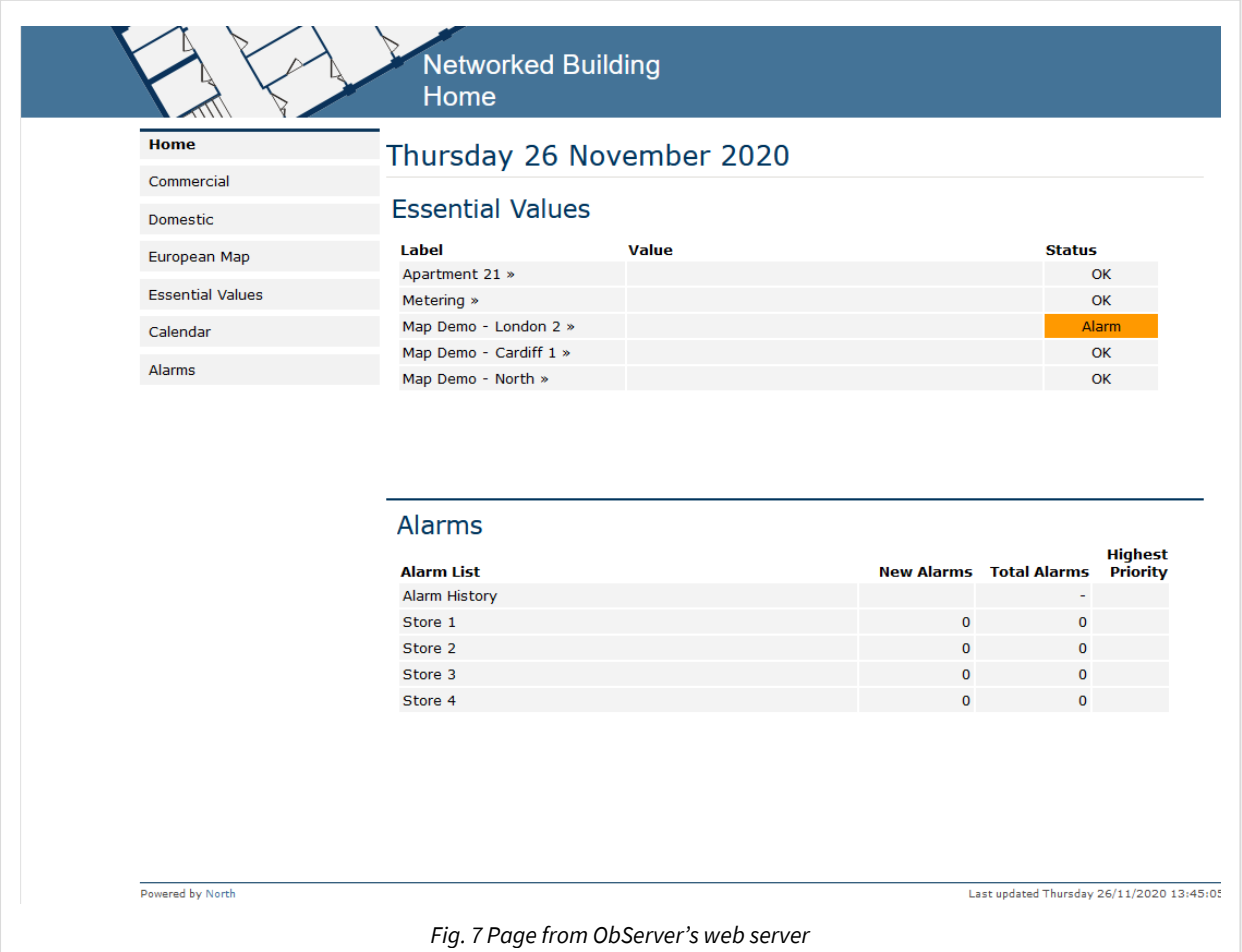


Fig. 7 Page from ObServer's web server

A demonstration of ObServer's web server is available visit *webview.northbt.com*.

Network Settings

By default, the web server is enabled on all available network interfaces using HTTP (port 80). You can restrict this to a single IP address if required, and change the port number.

If HTTPS is desired, then a reverse proxy is required. Use Windows IIS or Stunnel to provide this along with your own TLS certificate.

Security Settings

It is possible to specify whether users must sign-in before they can view or adjust values on the website.

User authentication can be provided by North's *Security Server*, or your organisation's Active Directory/LDAP server.

Customising WebView

All web pages are built using a template. This can be customised to your requirements using WebView Editor.

WebView also supports engineer-designed pages and objects. Again, these are created, edited, and downloaded to WebView using WebView Editor.

For further information on configuring WebView in ObServer, refer to the *WebView Manual*.

Telnet

Sometimes it is necessary to talk to ObServer without using web pages or engineering software. ObServer has a Telnet server that can be enabled to provide simple text-based access to any object values within ObServer.

ObServer supports two services within the Telnet session – query-response and IP-information.

Establishing a Session

Establish a Telnet session to ObServer by opening a TCP/IP connection on port 23.

Messages are formatted as a line of ASCII text. Each line of text, or message, must end with a carriage-return (control code 0x0D) and line-feed (control code 0x0A) – represented in this manual using the symbol: ↵.

By default, Telnet is disabled, but you can enable Telnet, and configure a user-name, to act as a simple password.

Once connected to ObServer, enter the user-name and then the service:

```
Telnet – North Telnet
User:*****↵
Service:
```

For help, enter '?' at the service prompt.

To close a session, send a line-feed at the service prompt. Alternatively, the session will automatically close after 5 minutes of inactivity.

IP Information Service

The Telnet IP-information service is used to discover the current network settings of ObServer.

At the service prompt, enter 'ipc' and ObServer will respond with the current IP configuration:

```
Service:ipc↵
IP Configuration..
Network 1:
   Address. . 192.168.192.167
   Mask . . . 255.255.255.0
   Gateway. . 0.0.0.0
```

Query Response Service

Using the Telnet query-response service, you can request and adjust the value of any object within, or connected to, ObServer.

At the service prompt, enter 'qr' and the Telnet session will enter query-response mode:

```
Service:qr↵
Q:
```

Enter the query command at the Q: prompt and ObServer will respond at the R: prompt.

Reading a Value

At the query prompt, enter the object reference to read. ObServer responds with the object value.

```
Q:object↵
R:value
```

For example, to read the local date & time (object O.T):

```
Q:O.T↵
R:01/12/15|15:40:08
Q:
```

Writing a Value

At the query prompt, enter the object reference and value to set. ObServer will respond with 'Ok' to indicate the object adjusted successfully.

```
Q:object=value↵
R:Ok
```

For example, to set the ObServer's Site Label (object O.L) to 'New ObServer':

```
Q:O.PL=New ObServer↵
R:Ok
Q:
```

Error Response

If the service encounters an error when processing a query command, it will respond with the error prompt (E:) followed by one of the following three-character error code:

Error Code	Reason
OBJ	Object reference invalid
ACT	Action invalid – cannot read/write this object
VAL	Value invalid
FLT	General fault
DDV	Device delivery fault
PDV	Point delivery fault
NDV	Network delivery fault
???	Unknown error

For example, attempt to read the invalid object reference ABCDE:

```
Q:ABCDE↵
E:OBJ
Q:
```

Other APIs

In addition to Telnet, other APIs are available via drivers –including JSONData and DataSync.
Search 'API' in North Product Documentation app for more details.

Updating ObSys

Occasionally you may wish to update your ObSys package – to access improvements and new features.

North try to ensure an upwardly compatible path for our software and drivers, although sometimes this is not possible. Therefore, before updating your ObSys software, ensure you have a complete backup of the ObSys system.

The latest version of ObSys is available from our web site, *www.northbt.com/support* . Ask North for access details. Simply download then run the ObSys setup application.

Object Specifications

Once ObServer has started, several objects become available within the top-level object of the device. These objects may contain sub-objects (and each of these may contain sub-objects, and so on) – the whole object structure being a multi-layer hierarchy. It is possible to navigate around the objects using the ObSys Engineering Software.

Each object is specified below, along with its sub-objects.

Example Object Reference

An example of a reference to an object within ObServer: the Configuration object (O) contains Platform Information (PI), which contains ObServer Version (PV) – therefore the complete object reference is 'O.PI.PV'.

An example of a reference to an object in a different device: the IP network object (IP) contains an alias to another device (A1), which contains the object above (O.PI.PV) – therefore the complete object reference is 'IP.A1.O.PI.PV'.

Device Top-Level Objects

Object Type: *[ObSys]*

When ObServer is started, the objects below become available within the top-level object of the device.

Description	Reference	Type
Site Label Label used to identify this device, including scanning and within alarms. Adjust within the Configuration object.	PL	Obj\Text; Max. 30 chars
Configuration Set up ObServer's main features	O	Variable Container: *[ObServer v20]*
Essential Values Contains the values configured within Essential Data	UD	Variable Container: *[UserData\PageList]*
Time Control Time and date based control using calendar, timers and profilers	CT	Variable Container: *[CalTimer v20]*
ObVerse Contains public properties configured in ObVerse Processor *x*. This object is only available once ObVerse has been downloaded to the processor. The processor number, *x* is in the range 1...4. *ObVerse* is set in public property L within the ObVerse.	P*x*	Variable Container: Type is based on the filename of the ObVerse. For example, if saved as the file 'TypeInfo\ObVerse\Process.obv', then the type will be: *[ObVerse\Process]*
Security Server Edit users and groups in the database for authentication	TK	Fixed Container: *[TokenMax v20\2000]* *[TokenMax v20\1000]* *[TokenMax v20\400]* *[TokenMax v20\100]*
Data Transfer Transfer values from one place to another	TX	Fixed Container: *[TransMax v14\1000]* *[TransMax v14\500]* *[TransMax v14\100]*
Alarm Translation Provides translation of text within alarms	AX	Fixed Container: *[AlmXlate v11]*

Description	Reference	Type
Alarm Prioritisation Provides re-prioritisation of alarms	AP	Fixed Container: *[AlmPrior v10]*
Alarm Delivery Configure destinations to route alarm events	AR	Fixed Container: *[AlmRoute v11]*
Alarm History Contains a list of the latest alarms received	AH	Fixed Container: *[AlarmHistory v11\100]* *[AlarmHistory v11\500]*
Alarm Store *x* Alarm Store *x* information, where *x* is the range 1…4	AS*x*	Fixed container: *[AlarmStore v13]*
Alarm Emailer Destination for alarms to be emailed to a recipient	AE	Variable Container: *[AlmEmail v22]*
North IP Devices Available North IP-compatible devices on the network	IP	Variable Container: *[IpBus Net]*
Interface Setup Set up the driver, started on interface c. This object is only available once an interface has been started. The interface number, *c*, is in the range 1…10	M*c*	Fixed Container: Refer to specific driver manual for type
Interface System Access the system connected to interface *c*. This object is only available once an interface has been started. The interface number, *c*, is in the range 1…10	S*c*	Fixed or Variable Container: Refer to specific driver manual for type
Third-party Application x If third-party applications have been installed, they may appear here as objects, Where *x* is in the range 1...10	USx	Fixed or Variable Container: Refer to specific third-party applications for type information

Configuration

Object Type: *[ObServer v20]*

The ObServer Configuration object contains the following objects:

Description	Reference	Type
Site ID This identifies the current instance of ObServer. It specifies the folder used for instance data storage. Usually 'DefaultSite'.	PN	Obj\Text; Max. 15 chars
Site Label Label used to identify this device, including scanning and within alarms.	PL	Obj\Text; Max. 30 chars; Adjustable
Local Date & Time Local date and time from the PC's real-time-clock	T	Obj\DateTime; Adjustable
Platform Information General information about ObServer, including hardware switches, version information, and debug recording	PI	Fixed Container: *[ObServer v20\Platform v12]*
ObServer Startup Contains a list of other applications to start when ObServer is started	N	Fixed Container: *[ObServer v20\Startup]*
Interfaces Start an interface, list installed drivers, and review interface licence information	O	Fixed Container: *[ObServer v20\Interfaces v10]*
Object Aliases A list of object reference shortcuts that are replaced when modules	A	Fixed Container: *[ObServer v20\Aliases]*
WebView Server Enable the WebView server, edit security, and select the features available from the web site	WS	Fixed Container: *[OSM v20\WebView v14]*
North IP Devices Manage the list of connected IP devices	IP	Fixed Container: *[OSM v20\IpBus v21]*
Essential Data Configure the database of values used by multiple ObServer modules	UD	Fixed Container: *[OSM v20\UserData v31\Format0]* *[OSM v20\UserData v31\Format1]* *[OSM v20\UserData v31\Format2]* *[OSM v20\UserData v31\Format3]*
Time Control Set the system label and access security	CT	Fixed Container: *[OSM v20\CalTimer v20]*
ObVerse Processor *x* Edit the ObVerse for this processor. Where *x* is in the range 1…4	P*x*	Fixed Container: *[OSM v20\OBVProcess v11]*
Security Server Set the label and maximum users available	TK	Fixed Container: *[OSM v20\TokenMax v20]*
Data Transfer Set the system label, maximum transfers available, and tasks	TX	Fixed Container: *[OSM v20\TransMax v14]*
Alarm Translator Used to translate text within alarm messages, before passing via Alarm Prioritizer to Alarm Delivery	AX	Fixed Container: *[OSM v20\AlmXlate v11]*
Alarm Prioritizer Used to re-prioritise alarm messages before passing to Alarm Delivery	AP	Fixed Container: *[OSM v20\AlmPrior v10]*

Description	Reference	Type
Alarm Delivery Set the system label	AR	Fixed Container: *[OSM v20\AlmRoute v11]*
Alarm History Set the system label and access security	AH	Fixed Container: *[OSM v20\AlarmHistory v11]*
Alarm Store *x* Specify the operation of alarm store *x*, where x is in the range 1…4	AS*x*	Fixed Container: *[OSM v20\AlarmStore v13]*
Alarm Email Configure the SMTP server parameters and destination recipient address	AE	Fixed Container: *[OSM v20\AlarmEmail v22]*
Telnet Setup Enable the Telnet service, label, and user-password	TN	Fixed Container: *[OSM v20\Telnet v10]*
Data Manager Used to configure DataManager application, if running	DM	Variable Container: *[DataManager v11]*
Alarm Manager Used to configure AlarmManager application, if running	AM	Fixed Container: *[AlarmManager v20]*
Password Authentication Used to configure Password Authentication for other ObSys applications	OP	Fixed Container: *[ObPsw v20]*
Third-party Application x Configuration If installed, the configuration objects of third-party applications may appear here, where *x* is in the range 1...10	UC*x*	Fixed or Variable container: Refer to Third-party applications documentation for more details

Platform Information

Object Type: *[ObServer v20\Platform v12]*

The Platform Information object provides *general information about ObServer*.

Description	Reference	Type
ObServer Version The ObServer's main software version and release date	PV	Obj\Text
ObSys Release Version Version number of the main ObSys release	HV	Obj\Text
Reset Platform Set to perform a reset of the Windows operating system. The value indicates the seconds before restart	!rst	Obj\Num; Range 0...1000
Operating System Information on operating system, memory, and disk space available	OS	Fixed Container: *[ObServer v20\Platform v12\OS]*
Data Storage Specifies when to save changes to disk	DS	Fixed Container: *[ObServer v20\Platform v12\DataStore]*
Last Restart Information about the last restart	LR	Fixed Container: *[ObServer v20\Platform v12\Restart]*
Software Versions Software versions for each of ObServer's main modules and started interfaces	V	Fixed Container: *[ObServer v20\Platform v12\Versions]*
Recording Enable debug recording, and select which information is recorded	R	Fixed Container: *[ObServer v20\Platform v12\Record]*

Operating System

Object Type: *[ObServer v20\Platform v12\OS]*

An Operating System object contains the following objects:

Description	Reference	Type
Windows Version Version of operation system	WV	Obj\Enum Values: 0=Unknown, 9=Windows 10, 10=later
Memory Available %	MA	Obj\Num; Range 0…100
Disk Space Available (MB)	DA	Obj\Num;
Windows Description	WL	Obj\Text

Data Storage

Object Type: *[ObServer v20\Platform v12\DataStore]*

A Data Storage object specifies how often ObServer should store data to disk.

ObServer only stores data to disk when necessary. However, the data for some modules, such as ObvProcess, can change constantly. Writing data at higher rates can wear out some types of disk.

Description	Reference	Type
Force Write Now When set to 'Yes', ObServer writes data out to disk immediately ObServer automatically writes data to disk when it is closed.	F	Obj\NoYes; Adjustable
Write Period (mins) Specifies how often changed data should be written to disk, in minutes	T	Obj\Num; Adjustable

Last Restart

Object Type: *[ObServer v20\Platform v12\Restart]*

A Last Restart object contains information about the last time ObServer was powered-up or reset.

Description	Reference	Type
Last Start Date & Time	ST	Obj\DateTime
Reset Count Number of times ObServer has been restarted	RC	Obj\Num; Adjustable

Software Versions

Object Type: *[ObServer v20\Platform v12\Versions]*

A Software Versions object holds the version and release date of the sub-components within ObServer, and contains the following objects:

Description	Reference	Type
Web Server	WS	Fixed Container: *[ObServer v20\Platform v12\Detail]*
Essential Data	UD	Fixed Container: *[ObServer v20\Platform v12\Detail]*
Time Control	CT	Fixed Container: *[ObServer v20\Platform v12\Detail]*
ObVerse Processor	P1	Fixed Container: *[ObServer v20\Platform v12\Detail]*
Security Server	TK	Fixed Container: *[ObServer v20\Platform v12\Detail]*
Data Transfer	TX	Fixed Container: *[ObServer v20\Platform v12\Detail]*
Alarm Translator	AX	Fixed Container: *[ObServer v20\Platform v12\Detail]*
Alarm Prioritizer	AP	Fixed Container: *[ObServer v20\Platform v12\Detail]*
Alarm Delivery	AR	Fixed Container: *[ObServer v20\Platform v12\Detail]*
Alarm History	AH	Fixed Container: *[ObServer v20\Platform v12\Detail]*
Alarm Store	AS1	Fixed Container: *[ObServer v20\Platform v12\Detail]*
Alarm Email	AE	Fixed Container: *[ObServer v20\Platform v12\Detail]*
North IP Devices	IP	Fixed Container: *[ObServer v20\Platform v12\Detail]*
Telnet	TN	Fixed Container: *[ObServer v20\Platform v12\Detail]*
Interface *x* Where *x* is in the range 1…10	M*x*	Fixed Container: *[ObServer v20\Platform v12\Detail]*

Software Version Detail

Object Type: *[ObServer v20\Platform v12\Detail]*

A Software Version Detail object contains the following objects:

Description	Reference	Type
Module Version Software component name, version and release date	DV	Obj\Text; Max chars 32
OSM Surround Version Common library version and release date	PV	Obj\Text; Max chars 32

Recording Configuration

Object Type: *[ObServer v20\Platform v12\Record]*

The Recording Configuration object enables *debug recording* for ObServer. The record file created may then be requested by North support to assist you.

ObServer can record to a file (F) activity of object requests (O), COM port activity (C), general information (I), and internal communications (T). Beware of enabling all record options, particularly Record COMs (C), as this can put a significant loading on ObServer.

A record of activity is appended to the record file until it reaches a particular size (approximately 10MB), and based on the Record Mode (M):

- Off – recording disabled
- Record – record activity until the file becomes full
- Until restart – record activity, recreating the file when full, and stopping when ObServer is closed
- Always – record activity, recreating the file when full, or when ObServer is started.

Description	Reference	Type
Record Mode Starts recording details, on the categories selected, and for how long. Set 'Record' to record activity until the file becomes full. Set 'Until restart' to record activity, recreating the file when full, and stopping when ObServer is stopped. Set 'Always' to record activity, recreating the file when full or when ObServer is started.	M	Obj\ENum; Range 0…3; Adjustable Values: 0=Off, 1=Record, 2=Until restart, 3=Always
Record Filename The full path of the file to record debug information. Adjusting this value will delete any existing file (with the same filename) and re-start recording.	F	Obj\File; Adjustable; Max chars: 127
Record Information Record general information, such as ObServer starting, in the record file	I	Obj\NoYes; Adjustable
Record Objects Record object request and reply messages between modules, plus any other additional debug messages from a driver	O	Obj\NoYes; Adjustable
Record COMs Record bytes sent and received on COM ports	C	Obj\NoYes; Adjustable
Record Internal Comms Record internal module fast request/reply messages	T	Obj\NoYes; Adjustable
Window Watch Set to 'OSM*x*' to watch requests from OSMx (where *x* is the OSM number) Set to 'COM*x*' to watch comms from port COMx (where *x* is the COM port number)	W	Obj\Text; Adjustable; Max.chars: 20
Window Hidden Whether the ObServer window is hidden. When ObServer is running with no Security Device, the window is always visible	H	Obj\NoYes; Adjustable

ObServer Start-up

Object Type: *[ObServer v20\Startup]*

This object contains a list of commands that are triggered when ObServer starts. The command line is passed to the operating system once ObServer has loaded all OSMs.

Typically this is used to start other ObSys applications, such as AlarmManager, DataManager, or an ObView mimic page.

Example commands

To start AlarmManager:
`AlmManager.exe`

To start DataManager:
`DataManager.exe`

To start an ObView mimic, titled 'floorplan.obv':
`ObView.exe floorplan`

Description	Reference	Type
Command *x* Command *x* is triggered when ObServer starts, where *x* is in the range 1…20	C*x*	Obj\Text; Max.chars:127; Adjustable

Interfaces Configuration

Object Type: *[ObServer v20\Interfaces v10]*

The Interfaces Configuration object allows to ObServer to *start interfaces to other systems*, list the installed drivers, and review licencing information.

Description	Reference	Type
Interface *x* Name of driver to start at interface *x*, where *x* is in the range 1...10	O*x*	Obj\Text; Max. 16 char; Adjustable
Installed Drivers List of installed drivers	A	Fixed Container: *[ObServer v20\Interfaces v10\Avail]*
Interface Licences Licencing information and upgrade	IL	Fixed Container: *[ObServer v20\Interfaces v10\IntLic]*

Installed Drivers

Object Type: *[ObServer v20\Interfaces v10\Avail]*

Drivers files have the extension '.OSM' and are installed within the folder:

C:\Program Files (x86)\North Building Technologies\ObSys\OSMs

An Installed Drivers object contains the following objects:

Description	Reference	Type
Installed Driver List of drivers available. The list index, *x,* is in the range 1…250	A*x*	Obj\Text; Max chars 16; Adjustable

Interface Licences

Object Type: *[ObServer v20\Interfaces v10\IntLic]*

Each ObServer is supplied with a certain number of *interface licences*. This information is held within the ObSys Licence Device. It is possible to add more licences on site, although a phone call is required to check/verify settings.

Call North support on +44 (0) 1273 694422 for help with adding a licence.

Description	Reference	Type
ObSys Serial Number ObServer's unique serial number	S	Obj\Text; Max chars 16
Total Licences Total number of interface licences installed	T	Obj\Num; in the range 0…9
Available Licences Number of interface licences available. Each started interface will typically use one licence	A	Obj\Num; in the range 0…9
Upgrade Licences Key used to add more interface licences	U	Obj\Text; Max chars 32; Adjustable

Object Aliases

Object Type: *[ObServer v20\Aliases]*

This object contains a list of aliases. When a module specifies an object reference within ObServer, ObServer checks the alias list, and if a matching alias is found, the alias part of the object reference is replaced with text specified in the alias

Description	Reference	Type
Alias *x* Information about Alias *x*, where *x* is in the range 1…128	A*x*	Fixed container: *[ObServer v20\Aliases\Entry]*

Object Alias Entry

Object Type: *[ObServer v20\Aliases\Entry]*

This object contains details of an alias, including the alias itself, and the replacement text.

For example, if an object request to object reference CTRL.S1.V is requested from ObServer, and an alias entry has Alias set to 'CTRL' and Aliased Object set to 'S1.U13', the ObServer passes the object request to object with reference 'S1.U13.S1.V'

Description	Reference	Type
Alias The alias to search for at the start of the object reference	F	Obj\Text; Adjustable; Max.chars: 16
Aliased Object The text to replace the alias with if found	T	Obj\Text; Adjustable; Max.chars: 16

WebView Server Configuration

Object Type: *[OSM v20\WebView v14]*

The WebView Server Configuration object enables *ObServer's web server* and selects what information is accessible from it.

Description	Reference	Type
Enable Web Server	E	Obj\NoYes; Adjustable
Show Calendar Enables access to the calendar, timers and profilers from the website	C.E	Obj\NoYes; Adjustable
Show Essential Values Enables access to values from Essential Data	UD.EV	Obj\NoYes; Adjustable
Show Values in Sub-menu Places pages from Essential Values and Extra Values in their own sub-menus on the website. Without this option enabled, pages are displayed in the main menu	UD.PL	Obj\NoYes; Adjustable
Show Data Manager logs Enables viewing of DataManager logs from the web server	L.ED	Obj\NoYes; Adjustable
Show Essential Data logs Enables viewing of Essential Data logs from the web server	L.EU	Obj\NoYes; Adjustable
Show Object List Enables full object access from the website. See also Security object to set the access security level required	S.OL.E	Obj\NoYes; Adjustable
Network Interface Allows control of the IP network interfaces to serve pages on	N	Fixed container: *[OSM v20\WebView v14\Network]*
Alarms Configure options for showing alarms on the web server	AM	Fixed Container: *[OSM v20\WebView v14\Alarms]*
Security Configure sign-in and access security levels	S	Fixed Container: *[OSM v20\WebView v14\Security]*

WebView Server Network Interface

Object Type: *[OSM v20\WebView v14\Network]*

Select which network interfaces the WebView server will be available to serve web pages.

A Web Server Network Interface object contains the following objects:

Description	Reference	Type
Current IP address Shows a list of IP addresses on which the web server is operating – the IP addresses are separated by a '|' character. For example '192.168.6.7|127.0.0.1'	CIA	Obj\Text
Available IP address Shows a list of IP addresses on which the web server could operate. For example '192.168.6.7|127.0.0.1'	AIA	Obj\Text
Force IP address If written to, forces WebView to serve webpages only on the specified IP address.	IA	Obj\IP; Adjustable Default: '0.0.0.0' (use all available)
TCP Port Specifies the port number to serve web pages from. Typically, 80, but if a web-server is already using that port on the PC, WebView can operate from another port	PN	Obj\Num; High limit: 65535 Default: 80

WebView Server Alarms

Object Type: *[OSM v20\WebView v14\Alarms]*

A Web Server Alarms object contains the following objects:

Description	Reference	Type
New Alarms Available Whether a new alarm is available on any of the following alarm lists	NEW	Obj\NoYes
List *x* Specification for an alarm list *x* to show on the web pages served, where *x* is in the range 1…10	L*x*	Fixed Container: *[OSM v20\WebView v14\Alarms\List]*

WebView Server Alarm Lists

Object Type: *[OSM v20\WebView v14\Alarms\List]*

Select which Alarm Stores and history are available from WebView. For an Alarm Store, typically set the Object Reference (O) to 'AS1' for Alarm Store 1, 'AS2', for Alarm Store 2, 'AH' for Alarm History, etc.

A Web Server Alarms object contains the following objects:

Description	Reference	Type
Object Reference Object reference of the alarm list – this can be an Alarm Store or an Alarm History	O	Obj\Obj; Adjustable
Label (optional) An optional label for the alarm list. If not specified then the label from the alarm list is used.	L	Obj\Text; Max 20 chars; Adjustable

WebView Server Security

Object Type: *[OSM v20\WebView v14\Security]*

The WebView Server Security object contains *security settings* to enable user sign-in and control access to the server.

Note that WebView will cache a user's credentials for approximately 5 minutes, any changes to a user's security level during that time will not be updated until the cache expires, or the user signs-out using the web page '/auto/signout'.

Description	Reference	Type
Method Specifies method of webpage security to use	M	Obj\ENum; Adjustable; where: 0=None, 1=North User Token; 2=LDAP Server
Security Server Object If Method is set to 'North User Token', this specifies the Security Server to use for token validation	TO	Obj\Obj; Adjustable Default: 'TK' (ObServer's local Security Server)
Sign-in expires after (s) The duration to automatically sign-out a user from when they sign-in. To disable sign-in expiry, set to '0' (default).	PWL	Obj\Num; Adjustable 0, 360…6500s
Editor Access Enable Specifies whether the WebView Editor application can access the server for editing	E.E	Obj\NoYes; Adjustable
Editor Access Security Specifies privilege area/level required to access web pages for editing. See *Security Server*	E.AS	Obj\Num; Range: 0, 10…87; Adjustable
Object List Access Security Specifies privilege area/level required to access web pages for Object List. See *Security Server*	OL.AS	Obj\Num; Range: 0, 10…87; Adjustable
DataManager Access Security Specifies privilege area/level required to access web pages for DataManager. See *Security Server*	DL.AS	Obj\Num; Range: 0, 10…87; Adjustable
Send Value Change Alarms If set to 'Yes', WebView sends alarm messages when a web user changes a setting	VC	Obj\NoYes; Adjustable
LDAP Setup Specifies setting for LDAP/Active Directory server	L	Fixed Container: *[OSM v20\WebView v14\Security\LDAP]*

Alarm Fields

Alarms are sent by WebView when a user signs-in and, if enabled, when a value is changed. WebView places the following information into North-format fields:

System – 'WebView'

Priority – '4'

Date & Time – from ObServer

When a user signs-in, point and condition fields contain:

Point – 'User signed-in'

Condition – *user-name*

When a user changes a value, point and condition fields contain:

Type of object adjusted	Point	Condition
Alarm	Alarm *list name*	Delete by *user-name* Note by *user-name* Silence by *user-name* Clear by *user-name*
Calendar	Calendar *object-reference*	Adjusted by *user-name*
General object	Object *object-reference*	*value* by *user-name*

Alarm Examples

System	Point	Condition	Priority	Date	Time
WebView	User signed-in	Mary Coombs	4	01/04/23	10:16:02
WebView	Object IP.A1.S1.M8.SP	21 by Mary Coombs	4	01/04/23	10:52:16

WebView Server Security LDAP Setup

Object Type: *[OSM v20\WebView v14\Security\LDAP]*

A WebView Server Security LDAP Setup object contains LDAP or Active Directory server information and privilege levels to use when the LDAP sign-in option is enabled.

Description	Reference	Type
LDAP Server The hostname or IP address of the server. E.g. 'dc1.domain.com'	S	Obj\Text; Adjustable; Max.chars: 120
LDAPS Connect to server using a secure connection (SSL/TLS)	TLS	Obj\NoYes; Adjustable
LDAP Users Base Dn The distinguished name to search within the LDAP/Active Directory server for users. E.g. 'CN=Users, DC=domain,DC=com'	B	Obj\Text; Adjustable; Max.chars: 120
LDAP Group Attribute Name of the attribute holding a user's group membership within an LDAP record, typically 'memberOf'	GA	Obj\Text; Adjustable; Max.chars: 60 Default: 'memberOf'
LDAP User Attribute The name of the attribute holding the sign-in id within the user's LDAP record. Typically this is 'userPrincipalName' for <user@domain.com> sign-in, or 'sAMAccountName' for <user> sign-in.	GA	Obj\Text; Adjustable; Max.chars: 60
LDAP Default Domain Domain name to use when connecting to LDAP server, E.g. 'domain.com' This is used when a user's sign-in id does not contain a domain.	DD	Obj\Text; Adjustable; Max.chars: 60
LDAP Group *x* Information about LDAP Group *x*, where *x* is in the range 1…32	G*x*	Fixed Container: *[OSM v20\WebView v14\Security\LDAP\Group]*
Area *x* Label Label for Security Area *x*, where *x* is in the range 1…8	A*x*	Obj\Text; Adjustable; Max.chars: 20
Level x Label Label for Security Level *x*, where *x* is in the range 0…7	L*x*	Obj\Text; Adjustable; Max.chars: 20
LDAP Debug Enable Enables additional information in the ObServer Record File. Use to assist in resolving LDAP connection issues (passwords will not be recorded). See *Debug recording*	DE	Obj\NoYes; Adjustable

Operation of LDAP Authentication

When a user signs-in, WebView performs the following actions:

- Connect to **LDAP Server**
- Authenticate with the LDAP server using <userId> and <password> provided at sign-in, along with the domain. If <userId> does not contain an @domain, then **LDAP Default Domain** is used
- Searches **LDAP Users Base Dn** for a user containing **LDAP User Attribute** = <userId>. E.g. userPrincipalName = <userId>
- Retrieves **LDAP Group Attribute** (member Of) for user. Any groups matching an **LDAP Group *x*** entry, then user is authenticated with levels of each Privilege Area

- LDAP attribute displayName is used as the authenticated user's name in WebView.

WebView Server Security LDAP Group

Object Type: *[OSM v20\WebView v14\Security\LDAP\Group]*

A WebView Server Security LDAP Group object contains.

Description	Reference	Type
Distinguished Name Full distinguishedName of group, e.g. 'CN=BMS Users,CN=Users,DC=domain,DC=com'	DN	Obj\Text: 120 chars; Adjustable
Label	L	Obj\Text: 20 chars; Adjustable
Privilege Level in Area *x* User's privilege level in Area *x*, where *x* is in the range 1…8. See *user information*	P*x*	Obj\Num: 0…7; Adjustable

North IP Devices Configuration

Object Type: *[OSM v20\IPBus v21]*

A North IP Devices Configuration object manages a list of Commanders, ObSys PCs, etc. that are available on the LAN or WAN. Refer to *Communicating with other North IP Devices* for an introduction on this module.

North IP devices on the LAN can be discovered automatically. By default, new devices on the local network are found when the North IP Devices system object is scanned from engineering software. In addition to this, the list can be scanned immediately using the Scan device list object (AF).

Devices that are outside the local network, or use an encryption key, must be added to the device list manually.

Description	Reference	Type
Scan device list Discover North IP devices on the LAN. Select 'scan now' to perform an immediate scan for new devices. Use 'wipe then scan', to erase the list and then scan for devices. After performing an action, refresh the object view. Note: will not find devices that have an encryption key set.	AF	Obj\ENum; Range 0...2; Adjustable Values: 0=Auto, 1=Scan now, 2=Wipe then scan
Local encryption key If specified, all other devices that communicate with this ObServer must be set to use this encryption key. Once set, reads as '****'	K	Obj\Text; Max chars 8; Adjustable
Device *x* North IP device *x* configuration, where *x* is in the range 1...200	A*x*	Fixed Container: *[OSM v20\IPBus v21\Alias]*

North IP Device Configuration

Object Type: *[OSM v20\IPBus v21\Alias]*

A North IP Device Configuration object contains an object reference, IP address, and encryption key for a Commander, ObSys PC, etc.

The module can also periodically check communications with the other device and generate an alarm message when communications fail.

Description	Reference	Type
Reference A short object reference for the device	N	Obj\Text; Max chars 5; Adjustable
IP address The IP address of the device	A	Obj\IP; Adjustable
Encryption key If the device has IP encryption enabled (with the Local Encryption Key object), this key must match that in the device. Always reads as '****'	K	Obj\Text; Max chars 8
Comms Check Rate If required, ObServer will periodically check it can request objects from the IP device, and will send an alarm message if comms cannot be established	R	Obj\ENum; Adjustable; where: 0=Off, 1=1m, 2=5m, 3=15m, 4=1h, 5=4h
Comms Fails Count of consecutive fails, or 0 if last check was successful	F	Obj\Num; Range 0...9

Alarm Fields

Alarms can be sent by the module to indicate a communications failure when value of Comms Fail reaches 3.

The North IP Devices module places the following information into the North-format fields:

System – 'North IP Devices'

Point – from Reference (N), then ' at ', then the IP Address (A)

Condition – either 'Comms Fail' or 'Comms Ok'

Priority – '2'

Date & Time – from ObServer

Alarm Examples

System	Point	Condition	Priority	Date	Time
North IP Devices	A1 at 192.168.192.167	Comms Fail	2	24/03/15	10:16:02
North IP Devices	A1 at 192.168.192.167	Comms Ok	2	24/03/15	10:52:16

Essential Data Configuration

Object Type: *[OSM v20\UserData v31\Format0]*
Object Type: *[OSM v20\UserData v31\Format1]*
Object Type: *[OSM v20\UserData v31\Format2]*
Object Type: *[OSM v20\UserData v31\Format3]*

The Essential Data object contains the configuration for *ObServer's database of values*.

Essential Data consists of a list of configurable pages, each of which has a list of configurable objects. In total 1280 database objects are available.

Description	Reference	Type
Essential Data Label	DL	Obj\Text; Max chars 20; Adjustable
Page x Object Layout Arrangement of pages and objects within the database. Choose from 80 pages of 16 objects (default), 128 pages of 10 objects, 40 pages of 32 objects, or 20 pages of 64 objects. Rescan the ObServer Configuration and this object after changing the value	PO	Obj\ENum; Adjustable Values: 0=80x16, 1=128x10, 2=40x32, 3=20x64
Alarm State Overall alarm state of all pages within the database – out-of-range alarm, communications fault	S	Obj\ENum; where: 0=Ok, 1=Alarm, 2=Comms, 3=Alarm & Comms
Log Channels Free Count of available object logging channels	FL	Obj\Num; Range 0...200
Page *x* Configure database page *x*. Where *x* is in the range 1…128, depending on Page x Object Layout object (PO) above	P*x*	Fixed Container depending on Page x Object Layout. 80 page x16 object layout: *[OSM v20\UserData v31\PageDef0]* 128 page x10 object layout: *[OSM v20\UserData v31\PageDef1]* 40 page x32 object layout: *[OSM v20\UserData v31\PageDef2]* 20 page x64 object layout: *[OSM v20\UserData v31\PageDef3]*
Task Control Allows control and monitoring of the module's operation	TI	Fixed Container: *[OSM v20\UserData v31\TaskInfo]*

Essential Data Page Configuration

Object Type: *[OSM v20\UserData v31\PageDef0]*
Object Type: *[OSM v20\UserData v31\PageDef1]*
Object Type: *[OSM v20\UserData v31\PageDef2]*
Object Type: *[OSM v20\UserData v31\PageDef3]*

The Essential Data Page object contains the configuration for a page within *ObServer's database of values*. Each page has a label, access security, remote object prefix, and up to 64 objects.

The Remote Object Prefix (RP), if set, applies a root object to all remote objects within the page. It allows the page to deal with a particular container object, for example an energy meter or fan coil, without the need to enter the full object reference. A benefit of this is that the same page can be copied and used like a template for similar devices. Then by only changing the Remote Object Prefix the same information from an energy meter or fan coil at a different address can be accessed.

When the Remote Object Prefix (RP) is set, then Essential Data uses this to optimize communications in a fault condition. When Object 1 (O1) is in a communications fault alarm state, then only this object on the page performs the remote action – the other objects on the page will automatically fail. Once this first object is communicating again, then normal operation will resume.

Description	Reference	Type
Label Required to enable the page to other modules	L	Obj\Text; Max chars 20; Adjustable
Remote Object prefix Optional prefix to add in front of each database object's Remote Object	RP	Obj\Obj; Adjustable
Access Security Area and minimum privilege level required to read all objects within the page from other modules. See *Security Server*	AS	Obj\Num; Range 0, 10...87; Adjustable
Page Alarm State Overall alarm state of all objects within the page – out-of-range alarm, communications fault	S	Obj\ENum; Range 0...3 Values: 0=Ok, 1=Alarm, 2=Comms, 3=Alarm & Comms
Comms Alarm Enable/Priority Enables communications fault alarms for the page, see below	P	Obj\Num; Range 0…9; Adjustable; where: 0=no alarms are sent, 1…9=alarm priority (1 is the highest alarm priority, and 9 is the lowest)
Object *x* Configure database object *x*. Where *x* is in the range 1...64, depending on Page x Object Layout object (PO)	O*x*	Fixed Container: *[OSM v20\UserData v31\ObjDef]*

Alarm Fields

Alarms can be sent by the page to indicate the communication state of objects within the page.

The Essential Data module places the following information into the North-format fields:

System – from the Essential Data Label object (DL)

Point – from the page's Label object (L)

Condition – either 'Communications Fault' or 'Communications Ok'

Priority – set using Comms Alarm Enable/Priority object (P)

Date & Time – from ObServer

Alarm Examples

System	Point	Condition	Priority	Date	Time
Essential Values	Air conditioning	Communications Fault	3	23/03/15	13:33:59
Essential Values	Air conditioning	Communications Ok	3	24/03/15	07:30:16

Essential Data Object Configuration

Object Type: *[OSM v20\UserData v31\ObjDef]*

Essential Data Object contains the configuration for an object within *ObServer's database of values*.

Essential Data allows the engineer to configure a list of values that other ObServer modules can then distribute – as web pages, as BACnet points, as Modbus points, on Zip displays…

Description	Reference	Type
Label Required to enable the object to other modules	L	Obj\Text; Max. 20 chars; Adjustable
Type Specifies the type of value that the database object should hold	T	Obj\ENum; Range 0...9; Adjustable Values: 0=Text, 1=NoYes, 2=OffOn, 3=Number, 4=Float, 5=Times, 6=DateTime, 7=Date, 8=ENum, 9=Profile
Adjustable Specifies whether the user can adjust the value from other modules	A	Obj\NoYes; Adjustable
Units Optional unit of measurement	U	Obj\Text; Max. 8 chars; Adjustable
Access Security Area and minimum privilege level required to adjust the object from other modules. See *Security Server*	AS	Obj\Num; Range 0, 10...87; Adjustable
Type ENum Alternatives When Type is 'ENum', set this to a comma-separated list of value labels. i.e. 'value-0,value-1,value-2', etc.	EA	Obj\Text; Max. 30 chars; Adjustable
Type Float Dps/Time Periods When Type is 'Float', set this to the number of decimal places to display the current value. When Type is 'Times' or 'Profile', set this to the number of on-off or time-value periods.	D	Obj\Num; Adjustable Type Float: Range 0…4 Type Times: Range 0…4 Type Profile: Range 0…8
Value High Limit Used as alarm limits, if alarms are enabled, and used as value limits when adjustments are made	VH	Obj\Float; Decimal places=4; Adjustable
Value Low Limit Used as alarm limits, if alarms are enabled, and used as value limits when adjustments are made	VL	Obj\Float; Decimal places=4; Adjustable
Current Value Database object's value	V	Obj\Text; Max chars 32; Adjustable
Value Alarm State Delay Duration the Current Value must remain outside the alarm limits, before Value Alarm State is set to an out-of-range alarm NOTE: Only available in v3.1 and later	AD	Obj\ENum: 0…9; Adjustable Values: 0=None, 1=1 sec, 2=5 secs, 3=15 secs, 4=1 min, 5=5 mins, 6=15 mins, 7=1 hr, 8=4 hrs, 9=12 hrs
Value Alarm State Object's alarm state – out-of-range alarm, communications fault	S	Obj\ENum; Range 0...3 Values: 0=Ok, 1=Alarm, 2=Comms, 3=Alarm & Comms
Value Last Updated Time and date the current value was last set, either by a user or remote action	VT	Obj\DateTime
Value Alarm Enable/Priority Enables out-of-range value alarms, see below	P	Obj\Num; Range 0...9; Adjustable Values: 0=no alarms are sent, 1…9=alarm priority (1 is the highest alarm priority, and 9 is the lowest)

Description	Reference	Type
Remote Action Selects that the current value is periodically read from or written to the Remote Object. When set to 'read', and Adjustable to 'yes', then the current value will also be written to the Remote Object when adjusted by a user.	RA	Obj\ENum; Range 0..2; Adjustable Values: 0=None, 1=Read, 2=Write
Remote Object Object reference to read or write. If the Page has a remote object prefix set, then this will be inserted before the Remote Object	RO	Obj\Obj; Adjustable
Remote Rate Frequency to read or write the Current Value. If Remote Action is 'write', then the value is always written on a change-of-value. Set a rate to perform an additional background write periodically.	RR	Obj\ENum; Range 0..9; Adjustable Values: 0=ASAP/COV, 1=1 sec, 2=5 secs, 3=15 secs, 4=1 min, 5=5 mins, 6=15 mins, 7=1 hr, 8=4 hrs, 9=12 hrs
Remote Fails Count of times the remote object has continuously failed to read or write	RF	Obj\Num; Range 0..9
Data Log Enable/Rate Enables logging the current value periodically. There are a limited number of logging channels available, refer to Log Channels Free (FL)	LR	Obj\ENum; Adjustable Values: 0=Disable, 1=1 min, 2=5 mins, 3=15 mins, 4=1 hr, 5=4 hrs, 6=12 hrs, 7=24 hrs
Data Log Access historical log of the value, if enabled	LOG	Obj\Log

Alarm Fields

Alarms can be sent by an object to indicate an out-of-range value alarm state.

The Essential Data module places the following information into the North-format fields:

System – from the Essential Data Label object (DL)

Point – from page Label (L), then ' - ', and then the object Label (L)

Condition – either 'Alarm' or 'Ok'

Priority – set using Value Alarm Enable/Priority object (P)

Date & Time – from ObServer

Alarm Examples

System	Point	Condition	Priority	Date	Time
Essential Values	Air conditioning - Room temp	Alarm	3	24/03/15	10:16:02
Essential Values	Air conditioning - Room temp	Ok	3	24/03/15	10:52:16

Essential Data Task Control

Object Type: *[OSM v20\UserData v31\TaskInfo]*

The Essential Data Task Control object is used to control and monitor the operation of *ObServer's database of values*.

Description	Reference	Type
Enable Option to enable or disable the operation of Essential Data, or to only allow read remote actions. 'Enable reading' will stop all writing to the remote objects and only permit remote actions set to read.	E	Obj\ENum: Range 0...2 ; Adjustable Values: 0=Disable, 1=Enable all, 2=Enable reading
Value being read Page and object index that Essential Data is currently processing with a remote action of read	RT	Obj\Text
Value being written Page and object index that Essential Data is currently processing with a remote action of write	WT	Obj\Text

Time Control Configuration

Object Type: *[OSM v20\CalTimer v20]*

A Time Control Configuration object contains a label and access security objects for *ObServer's calendar, timers and profilers*.

Description	Reference	Type
Time Control Label	DL	Obj\Text; Max chars 20; Adjustable
Security: Read Access Area and minimum privilege level required to read the calendar and timers. See *Security Server*	AS	Obj\Num; Range 0, 10...87; Adjustable
Security: Edit Timer Area and minimum privilege level required to adjust a timer or profiler	AS.ET	Obj\Num; Range 0, 10…87; Adjustable
Security: Edit Calendar Area and minimum privilege level required to adjust the calendar	AS.EC	Obj\Num; Range 0, 10…87; Adjustable

ObVerse Processor Configuration

Object Type: *[OSM v20\OBVProcess v11]*

An ObVerse Processor Configuration contains an object to engineer the strategy within the *ObVerse Processor*. Use the North ObvEditor application to create and edit a cause-and-effect strategy.

Description	Reference	Type
ObVerse Edits ObVerse strategy within the processor. Connect using ObvEditor software	P	Obj\EProcess; Adjustable
Idles per second Number of times the complete ObVerse strategy is processed per second. Not available in driver version 1.0.	IR	Obj\Num
Object Modules Monitor the operation of object module types within the processor. These include Object-Read and Object-Write modules. Not available in driver version 1.0.	T	Fixed container: *[OSM v20\OBVProcess v11\ObjMods]*

Alarm Fields

Alarms can be sent by any alarm module within the ObVerse strategy. Refer to *ObVerse Manual: Standard Processor* for more information on the alarm module.

The ObVerse Processor places the following information into the North-format fields:

System – from label of the process

Point – from alarm module's point object

Condition – from alarm module's condition field

Priority – from alarm module's priority field

Date & Time – from ObServer

Alarm Examples

System	Point	Condition	Priority	Date	Time
Main Plant	Temperature	Too High	3	23/03/16	13:33:59
Main Plant	Temperature	OK	3	23/03/16	15:21:16

ObVerse Object Modules

Object Type: *[OSM v20\OBVProcess v11\ObjMods]*

ObVerse Object Modules contains objects to enable and monitor the operation of object modules types within the processor. These modules perform a remote object operation and include the Object-Read, Object-Write, and Alarm modules.

Description	Reference	Type
Operation Temporarily disable the remote object operation of all object module types in the processor	D	Obj\ENum; Adjustable Values: 0=Enabled, 1=Disabled
Active Module Indicates which object module the processor is performing the remote object operation	M	Obj\Text Format: *item=module-type object-reference* Examples: V6=!Obv\ObjRead S1.M3.UI1.V V21=!Obv\ObjWrite UD.P2.O6 V36=!Obv\Alarm ALARM
Last Failed Module Indicates the last object module the processor failed to perform the remote object operation	FM	Obj\Text Format: *item=module-type object-reference*
Last Failed Time Indicates the date and time the processor failed to perform the remote object operation	FDT	Obj\DateTime

Security Server Configuration

Object Type: *[OSM v20\TokenMax v20]*

A Security Server Configuration contains objects to set a label, and the number of users stored by *ObServer's Security Server*.

Description	Reference	Type
Security Server Label	DL	Obj\Text; Max chars 20; Adjustable
Maximum Users Sets the maximum number of users available within the Security Server. Set this to 100, 400, 1000 or 2000 only.	MU	Obj\Num; Range 100...2000; Adjustable

Data Transfer Configuration

Object Type: *[OSM v20\TransMax v14]*

A Data Transfer Configuration object enables the *transferring of values by ObServer*, and contains objects to set the maximum transfers available and to monitor its operation.

Description	Reference	Type
Transfer Enable Option to enable or disable the operation of transfers, or to only allow reading. 'Enable reading' will stop all writing to destination objects and only read source objects.	E	Obj\ENum; Range 0...2; Adjustable Values: 0=Disable, 1=Enable All, 2=Enable Reading
Maximum Transfers Sets the maximum number of transfers available. Set this to 100, 500 or 1000 only.	MT	Obj\Num; Range 100...1000; Adjustable
Maximum Tasks Sets the number of simultaneous transfer tasks performed. Typically set this to '1'.	XT.M	Obj\Num; Range 1…4; Adjustable
Task *x* Allows monitoring of the module's four transfer tasks. The task number, *x*, is in the range 1…4	XT*x*	Fixed Container: *[OSM v20\TransMax v14\Task]*

Data Transfer Task

Object Type: *[OSM v20\TransMax v14\Task]*

A Data Transfer Task object contains information about a transfer task.

Description	Reference	Type
Process Current action the transfer task is performing	P	Obj\ENum; Range 0...2 Values: 0= Idle, 1=Reading, 2=Writing
Transfer Number Transfer the task is currently performing an action on	X	Obj\Num; Range 0..1000

Alarm Translator Configuration

Object Type: *[OSM v20\AlmXlate v11]*

An Alarm Translator Configuration object contains a label for the *alarm translation system*, along with other setup objects.

Description	Reference	Type
Label	DL	Obj\Text; Max chars 20; Adjustable
Alarm Transmission OK This informs whether the comms from the Alarm Translator to the Alarm Prioritizer module is Ok	DS	Obj\NoYes
Ignore Case When comparing alarm text, whether to ignore the case of the characters	IC	Obj\NoYes; Adjustable
Re-route untranslated alarms After Translation, alarms are typically passed to the Prioritizer. This object enables re-routing to the Untranslated Alarm Object (AO2) if an alarm is not matched with any words in the Translator.	RU	Obj\NoYes; Adjustable
Untranslated Alarm Object If Re-route untranslated alarms (RU) is enabled, this specifies the object to route untranslated alarms. E.g. 'AH.ALARM' for Alarm History, or set to '' (blank) to delete untranslated alarms	AO2	Obj\Obj; Adjustable
CSV Filename The file name of a translation csv file, if required. Default file is '\AlmInfo\AlmXlate.csv', located in the ObSys data folder	F.FN	Obj\File; Adjustable
File Status If a csv file is specified, whether the file has been read successfully, or the type of error if unable to read	F.S	Obj\ENum; where: 0=Ok, 1=Not found, 2=Format, 3=In Use
Monitor File Frequency (secs) If a csv file is specified, the read rate for the file, to allow for changes	F.ST	Obj\Num; Adjustable; Range 1…3600
File Translations Count If a file is specified, the number of translations loaded from the csv file	F.C	Obj\Num; Range: 0…10000

Alarm Translator CSV File Format

Alarm Translator can use a comma-separated value (csv) file to translate an entire alarm message. Each line within the csv file specifies the system, point, and condition fields to find; then the replacement system, point, condition, and priority fields:

```
find-system,find-point,find-condition,replace-system,replace-point,replace-condition,replace-priority
```

Individual fields may contain a comma, however the field should then be enclosed in double-quotes (see example below). Fields may not contain the separator character '|'. Lines starting with a semi-colon are ignored as a comment line.

Example CSV file

```
;find-system,find-point,find-condition,replace-system,replace-point,replace-
condition,replace-priority
BMS,OLD BOILERHOUSE HDR FLW TEMP,occurred,Engineering,Boilerhouse,Header Flow Temp,3
BMS,OLD BOILERHOUSE HWS TEMP,occurred,Engineering,Boilerhouse,HWS Temp High,3
BMS,"PM Suite,Atrium R:03 Supply Fan",OCCURRED,Engineering,C.B.L AHU No3,Air Failed,3
BMS,HSDU VENT RM 1030 FILTER ALARM,OCCURRED,Engineering,Room 1030,Filter Alarm,3
BMS,ROOFTOP TANKS CENTRE LIFT 1 CAR,OCCURRED,Security,Lift 1,Panic Alarm,1
BMS,ARCHIVE STORE INTRUDER ALARM,OCCURRED,Security,Archive Store,Intruder Alarm,2
```

Alarm Prioritizer Configuration

Object Type: *[OSM v20\AlmPrior v10]*

An Alarm Prioritizer Configuration object contains a label for the *alarm prioritisation system*.

Description	Reference	Type
Label	DL	Obj\Text; Max chars 20; Adjustable
Alarm Transmission OK This informs whether the comms from the Prioritizer to the Alarm Delivery module is Ok	DS	Obj\NoYes

Alarm Delivery Configuration

Object Type: *[OSM v20\AlmRoute v11]*

An Alarm Delivery Configuration object contains a label for the *alarm delivery system*.

Description	Reference	Type
Alarm Delivery Label	DL	Obj\Text; Max chars 20; Adjustable

Alarm History Configuration

Object Type: *[OSM v20\AlarmHistory v11]*

An Alarm History Configuration object contains a system label and access security objects for *ObServer's alarm history list*.

Description	Reference	Type
Alarm History Label	DL	Obj\Text; Max.chars 20; Adjustable
History Size Specifies the size of the alarm history. Set to either 100 or 500	MS	Obj\Num; Adjustable; Range 100…500
Read Access Security Area and minimum privilege level required to read the alarms. See *Security Server*	AS	Obj\Num; Range 0, 10...87; Adjustable
Delete Access Security Area and minimum privilege level required to delete the alarms	AS.D	Obj\Num; Range 0, 10...87; Adjustable

Alarm Store Configuration

Object Type: *[OSM v20\AlarmStore v13]*

An Alarm Store Configuration object contains a system label and access security objects for *ObServer's alarm store list*.

Description	Reference	Type
Alarm Store Label	DL	Obj\Text; Max chars 20; Adjustable
Read Access Security Area and minimum privilege level required to read the alarms. See *Security Server*	AS	Obj\Num; Range 0, 10...87; Adjustable
Silence Access Security Area and minimum privilege level required to silence the alarms	AS.S	Obj\Num; Range 0, 10...87; Adjustable
Delete Access Security Area and minimum privilege level required to delete the alarms	AS.D	Obj\Num; Range 0, 10...87; Adjustable
Delete All Alarms If set to 'Yes' all alarms are immediately deleted from the store	DAA	Obj\NoYes; Adjustable
Archive Setup of the alarm archive for this store	A	Fixed container: *[OSM v20\AlarmStore v13\Archive]*

Alarm Store Archive Configuration

Object Type: *[OSM v20\AlarmStore v13\Archive]*

An Alarm Store Archive Configuration object contains a system label and access security objects for *ObServer's alarm store list*.

Description	Reference	Type
Enable Whether archiving is enabled	E	Obj\NoYes; Adjustable
CSV File The full filename of the archive file into which Alarm Store puts alarm archive entries	F	Obj\File; Adjustable
Maximum Store (MB) Specifies the size the archive file must reach before it is 'shrunk' by 10%	MS	Obj\Float; Range 0…10; Decimals: 2; Adjustable
Status (store available) Whether the Alarm Store can currently write to the archive file. When the archive file is opened/locked by other applications, the Status may become 'No'	S	Obj\NoYes
Temporary File in Use When the Store becomes unavailable, alarms are archived temporarily in a different file, which is appended to the archive file when possible	TS	Obj\NoYes

Alarm Email Configuration

Object Type: *[OSM v20\AlmEmail v22]*

An Alarm Email Configuration object contains objects that set up how *ObServer's alarm emailer* connects to an SMTP relay server, and also contains the set up of 127 destinations to provide email addresses for recipients.

Description	Reference	Type
Alarm Email label	DL	Obj\Text; Max chars 20; Adjustable
Email server IP/FQDN IP address or host name of SMTP relay server, e.g. '102.13.14.15' or 'mail.domain.com'. If a host name is used, ensure a DNS server address is also configured for the PC	IA	Obj\Text; Max chars 127; Adjustable
Email server port TCP/IP port number of SMTP service, typically 25. If a firewall is in use, ensure it permits outbound traffic on this port number. SMTP servers supporting encryption are not supported, so check compatibility of servers using ports 465 or 587.	PN	Obj\Num: 1…65535; Adjustable Default: 25
Email server reachable Indicates that the email server IP and port have been configured correctly	DS	Obj\NoYes
Server timeout (s) Maximum time to wait for a response from the SMTP server	TO	Obj\Num; Range 5..120; Adjustable
Authentication name Username to use when sign-in is required. A sign-in may be required on some public SMTP relay servers, such as from BT Internet. The authentication supported by the Alarm Email module is AUTH LOGIN.	A.ID	Obj\Text; Max chars 40; Adjustable
Authentication password Password to use when sign-in is required	A.PW	Obj\Text; Max chars 40; Adjustable If a password is set, it will read as '****'
From name Label describing the email author, ObServer. If not specified, Alarm Email Label (DL) will be used	FT	Obj\Text; Max chars 40; Adjustable
From email address Email address to use for the email author. Usually required by the SMTP server	DFA	Obj\Text; Max chars 40; Adjustable
Template HTML Optional template file used to send emails in html format. See *Email templates* Filename may include the full path or, if starting '\', be relative to the ObSys data folder.	TF.HTM	Obj\File; Adjustable
Template Text Optional template file used to send emails in text format. See *Email templates* Filename may include the full path or, if starting '\', be relative to the ObSys data folder.	TF.TXT	Obj\File; Adjustable

Description	Reference	Type
Client Domain Name Optional hostname ObServer uses to identify itself when connecting to the SMTP server. E.g. 'obsys.domain.com' If not specified, then ObServer's IP address will be used	FQDN	Obj\Text; Max chars 127; Adjustable
Web page address Optional url to include in the email message. Configure the link with the url of ObServer's website, e.g. 'webview.domain.com' Value may contain variables supported by Alarm Emailer, e.g. $(destination)	WV	Obj\Text; Max chars 120; Adjustable
Sent email count Total number of emails sent since ObServer last started	SC	Obj\Num
Failed email count Count of times email messages have continuously failed to send	FC	Obj\Num
Last Fail Message Message from server indicating reason last message failed to send	FM	Obj\Text
Sent test to destination Triggers a test email message to the destination number specified	TST	Obj\Num; Range 0...127; Adjustable
Debug enable Enables the recording of communication with the SMTP server. Use this when instructed by North support to assist with fault finding sending emails to an SMTP server. See *Debug recording*	DE	Obj\NoYes; Adjustable
Destination *x* Configure a destination with filtering options and email address. The destination, *x*, is in the range 1...127	D*x*	Fixed Container: *[OSM v20\AlmEmail v22\Dest]*

Email Template

Alarm Emailer is pre-configured with templates for sending emails in HTML and text format (see Fig. 6).

Use Template HTML object (TF.HTM) to specify your own valid html template file. Many email clients only support limited html and style sheet content. If you wish to include images within the content, these may be served from WebView.

Use Template Text object (TF.TXT) to specify your own text template file. This could be used to send alarm events to an external system, for machine-to-machine (M2M) communication.

Alarm Email Variables

When creating a template, variables can be inserted which Alarm Emailer replaces with fields from the alarm message.

Variable	Field	Description
$(device)	Alarm system name	First field of the alarm message. This is the originating system label
$(point)	Alarm point name	Second field of the alarm message. This is the label of the point in alarm
$(condition)	Alarm condition	Third field of the alarm message. This is the alarm condition
$(priority)	Alarm priority	Alarm priority number, in the range 1…10; where 1 is the most important
$(date)	Alarm date	Date within the alarm message in the format dd/mm/yy
$(time)	Alarm time	Time within the alarm message in the format hh:mm:ss

Variable	Field	Description
$(icon)	Alarm icon	Html image tag used to indicate the alarm priority. Only use this field within the html template
$(destination)	Destination label	Label of the destination
$(webview)	Webview uri	Web page address (WV) object
$(to)	To email address	Destination email address
$(label)	Label	Alarm Email Label (DL) object
$(nowd)	Current date	Current date in the format dd/mm/yy
$(nowt)	Current time	Current time in the format hh:mm:ss

File Attachments

Alarms with matching records in the ObSys alarm database will be attached to the html email message on sending. Add required files to the alarm database and ensure the filename begins 'email'. For more information refer to the Alarm Database within *ObSys Manual - Part 2*.

Alarm Email Destination Configuration

Object Type: *[OSM v20\AlmEmail v22\Dest]*

An Alarm Email Destination Configuration contains up to five addresses of where an alarm message is sent by *ObServer's alarm emailer*. An optional comparison method can be specified to filter alarms received from the system's ALARM object.

Description	Reference	Type
Label Description of destination group	L	Obj\Text; Max chars 20; Adjustable
Send as text only Send emails as text-only format rather than HTML	EF	Obj\NoYes; Adjustable
Comparison Method Optional filter, so that only alarms containing certain text are emailed	C	Obj\ENum; Adjustable; where: 0=Begins with, 1=Contains, 2=Does not begin, 3=Does not contain, 4=Always send
Comparison String Optional text used with the comparison method to filter incoming alarms. String comparisons are case insensitive	S	Obj\Text; Max chars 20; Adjustable
Address *x* Configure an email recipient. The address, *x*, is in the range 1...5	TO*x*	Fixed Container: *[OSM v20\AlmEmail v22\Addr]*

Email Address Setup

Object Type: *[OSM v20\AlmEmail v22\Addr]*

An Email Address Setup object contains the following objects.

When the Fails (F) count reaches 3 for the address, then Enable (E) is set to 'No' and a destination failed alarm is sent.

Description	Reference	Type
Email address Single email address to email alarms, e.g. 'name@domain.com'	A	Obj\Text; Max chars 63; Adjustable
Enable Enables emailing alarms to the address	E	Obj\NoYes; Adjustable
Fails Count of times email messages have continuously failed to send to this address	F	Obj\Num; Range 0...9; Adjustable

Alarm Fields

Alarms are sent by an address to indicate a fail state.

The Alarm Email module places the following information into the North-format fields:

System – from the Alarm Email Label object (DL)

Point – from the email destination's Label object (D*x*.L), or 'Destination *x*' if no label is set, then ' - ', then the Email address object (D*x*.TO*x*.A)

Condition – set to 'Destination Failed'

Priority – set to '3'

Date & Time – from ObServer

Alarm Examples

System	Point	Condition	Priority	Date	Time
Alarm Emailer	Destination 1 - name@domain.com	Destination Failed	3	23/03/15	13:33:59

Telnet Setup

Object Type: *[OSM v20\Telnet v10]*

A Telnet Setup object controls access to ObServer from Telnet clients elsewhere on the LAN.

When enabled, the session label is returned when a Telnet client opens the session. The client is then prompted for a user. If successfully authenticated, the session then asks which service the telnet client wants.

For information on establishing a Telnet session and the commands supported, refer to the *Telnet* section.

Description	Reference	Type
Session Label Provided when a Telnet session is opened	DL	Obj\Text; Max chars 20; Adjustable
Telnet Enabled Enables the Telnet service	TE	Obj\NoYes; Adjustable
User/Password Username required to access the Telnet service	PSW	Obj\Text; Max chars 7; Adjustable
Connections Number of Telnet sessions currently established	CC	Obj\Num: 0…4

Essential Values

Object Type: *[UserData\PageList]*

The Essential Values object contains pages of values from *ObServer's database of values*.

Configure the database using the *Essential Data Configuration* object.

Description	Reference	Type
Page Label Page of values with label *Page Label.* The page number, *x*, is in the range 1...128	P*x*	Variable Container: *[UserData\Page]*
Alarm State Overall alarm state of all pages within the database – out-of-range alarm, communications fault	S	Obj\ENum; Range 0...3 Values: 0=Ok, 1=Alarm, 2=Comms, 3=Alarm & Comms

Essential Values Page

Object Type: *[UserData\Page]*

An Essential Values Page object contains a value for each of the objects configured in the database.

Description	Reference	Type
Label Object's value with label *Label*. The object number, *x*, is in the range 1...64	O*x*	Type as configured for the object in the database, could be: Obj\Text Obj\NoYes Obj\OffOn Obj\Num Obj\Float Obj\Times Obj\DateTime Obj\Date Obj\ENum Obj\Profile If the object has been set to Adjustable, then the value will be adjustable here.
Alarm State Alarm state for all objects within the page – out-of-range alarm, communications fault	S	Obj\ENum; Range 0...3 Values: 0=Ok, 1=Alarm, 2=Comms, 3=Alarm & Comms

Time Control

Object Type: *[CalTimer v20]*

A Time Control object is used to provide time-based control. The control can be simple timer on-off control, for things such as lighting. The control can also be profile-based control, for things such as temperature set points.

For an overview, see the section on *Controlling using Time and Date*.

Description	Reference	Type
Calendar Set day-types for standard days of the week and exception days	C	Fixed Container: *[CalTimer v20\Calendar]*
Timer *x* Set a time-list for each day-type. Timer number, *x*, is in the range 1...20	T*x*	Fixed Container: *[CalTimer v20\Timer]*
Profiler *x* Set a profile for each day-type. Profiler number, *x*, is in the range 1...20	P*x*	Fixed Container: *[CalTimer v20\Profiler]*

Calendar

Object Type: *[CalTimer v20\Calendar]*

A Calendar object is used to determine today's day-type. This is calculated from a set of day-types based on the day-of-week, which can be overridden by a list of exception dates. Each day-type can be assigned a label. If necessary, rather than calculate today's day-type, a Calendar can read a day-type calculated by a Calendar in a different device.

Description	Reference	Type
Source Object Optional current day-type object of the calendar in a different device, e.g. 'IP.A1.CT.C.T'	SO	Obj\Obj; Adjustable
Source Read Rate The rate to read the calendar source object in a different device	SR	Obj\ENum; Range 0...9; Adjustable Values: 0=ASAP, 1=1s, 2=5s, 3=15s, 4=1m, 5=5m, 6=15m, 7=1h, 8=4h, 9=12h
Source Fail Count Count of consecutive times the source object has failed to read successfully	SF	Obj\Num; Range 0...9
Day-type *x* Label Label for day-type *x*, where *x* is in the range 0..9. The label for day-type 0 is fixed to 'Off'.	T*x*.L	Obj\Text; Max chars 20; Adjustable
Current Day-type Calculated day-type for today	T	Obj\Num; Range 0...9
Current Day-type label Calculated day-type for today	T.L	Obj\Text; Max chars 20
Monday Day-type Set to the day-type required for each standard Monday	D1.T	Obj\Num; Range 0...9; Adjustable
Tuesday Day-type Set to the day-type required for each standard Tuesday	D2.T	Obj\Num; Range 0...9; Adjustable
Wednesday Day-type Set to the day-type required for each standard Wednesday	D3.T	Obj\Num; Range 0…9; Adjustable
Thursday Day-type Set to the day-type required for each standard Thursday	D4.T	Obj\Num; Range 0...9; Adjustable
Friday Day-type Set to the day-type required for each standard Friday	D5.T	Obj\Num; Range 0...9; Adjustable
Saturday Day-type Set to the day-type required for each standard Saturday	D6.T	Obj\Num; Range 0...9; Adjustable
Sunday Day-type Set to the day-type required for each standard Sunday	D7.T	Obj\Num; Range 0...9; Adjustable
Exceptions Used Count of exceptions currently in use	EC	Obj\Num; Range 0...40
Exception Date *x* List of non-standard exception days. Where *x* is in the range 1...40	E*x*	Fixed Container: *[CalTimer v20\Calendar\Except]*
***dd/mm/yy* day-type** Alternative object to set the exception day-type for a specific date, *dd/mm/yy*	ED*dd.mm.yy*	Obj\Num; Range 0…9; Adjustable

Calendar Exception Date

Object Type: *[CalTimer v20\Calendar\Except]*

A Calendar Exception Date contains the date and day-type to use on a specific exception day.

Description	Reference	Type
Date The date to apply the exception	D	Obj\Date; Adjustable
Day-type Set to the day-type to use on the exception date	T	Obj\Num; Range 0...9

Timer

Object Type: *[CalTimer v20\Timer]*

A Timer object uses today's day-type, from the *Calendar*, to select one of several on-off times to be used today. It uses the current time to determine from those on-off times to determine whether it should be on or off at this time. It can calculate a profile, based on simple on and off values.

A Timer object contains the following objects:

Description	Reference	Type
Label	L	Obj\Text; Max chars 20; Adjustable
Day-type *x* Times Times for day-type *x*, where *x* is in the range 1..9	T*x*	Obj\Times; Max periods 5; Adjustable
State Current state of timer, based on today's times, and the current time	S	Obj\OffOn
Destination Object Optional object to write State to when it changes	DO	Obj\Obj; Adjustable
Destination Fails Count of consecutive fails when writing to destination object, or 0 if successfully written	DF	Obj\Num; Range 0...9
Today's Times A copy of the times for today's day-type. Adjusting this object will adjust the times for today's day-type	TT	Obj\Times; Adjustable; Max periods 5
Profile Off Value Used for converting Today's Times to Today's Profile, the value in the profile when the time is off	V0	Obj\Float; Adjustable; Range 0.0…100.0
Profile On Value Used for converting Today's Times to Today's Profile, the value in the profile when the time is on	V1	Obj\Float; Adjustable; Range 0.0...100.0
Today's Profile A calculated profile based on Today's Times	TP	Obj\Profile; Max points 8

Profiler

Object Type: [CalTimer v20\Profiler]

A Profiler uses today's day-type, from a *Calendar*, to select one of several profiles to be used today. It then uses the current time, along with the change-points specified in today's profile, to determine whether to change the profiler's value. It can also calculate a set of on-off times, based on a switch level.

A Profiler object contains the following objects:

Description	Reference	Type
Label	L	Obj\Text; Max chars 20; Adjustable
Day-type *x* Profile Profile for day-type *x*, where *x* is in the range 1..9	P*x*	Obj\Profile; Max points 8; Adjustable
Value Current value of the profiler, based on today's profile and current time	V	Obj\Float; Range 0.00...100.0; Adjustable
Destination Object Optional object to write State to when it changes	DO	Obj\Obj; Adjustable
Destination Fails Count of consecutive fails when writing to destination object, or 0 if successfully written	DF	Obj\Num; Range 0…9
Today's Profile A copy of the profile for today's day-type. Adjusting this object will adjust the profile for today's day-type	TP	Obj\Profile; Adjustable; Max points 8
Time Switch Value Used for converting Today's Profile to Today's Times. The threshold value in today's profile below which the time is off, and above which the time is on	TV	Obj\Float; Adjustable; Range 0.00…100.0
Today's Times Calculated on-off times based on Today's Profile	TT	Obj\Times; Max period 5

ObVerse

Object Type: *[ObVerse\Process]*

The ObVerse object contains a list of the public properties available within the *ObVerse processor*.

The public properties are defined by the engineer of the ObVerse, so cannot be listed here. Instead, we list an object for each property type available.

Objects may be adjustable depending on the ObVerse. If the property is linked to the output of a module, then the module controls the value of the property, and so the property is not adjustable here. If the property has no module output linked to it, then the property is adjustable here.

For more information, refer to the *ObVerse Manual – Standard Processor* document.

Description	Reference	Type
ENum Label ENum property, holding an enumerated value, with reference *r*	*r*	Obj\ENum Text alternatives for each number are defined in the property
Float Label Float property, holding a floating-point value, with reference *r*	*r*	Obj\Float Value high and low limits, and decimal places are defined in the property
NoYes Label NoYes property, holding a binary state, with reference *r*	*r*	Obj\NoYes
Num Label Num property, holding an integer value, with reference *r*	*r*	Obj\Num: -2147483648… 2147483647 Value high, and low limits are defined in the property
Obj Label Obj property, holding an object reference, with reference *r*	*r*	Obj\Obj
OffOn Label OffOn property, holding a binary state, with reference *r*	*r*	Obj\OffOn
Text Label Text property, holding a text string value, with reference *r*	*r*	Obj\Text: max 31 chars. Maximum length is defined in the property

Security Server

Object Type: *[TokenMax v20\100]*
Object Type: *[TokenMax v20\400]*
Object Type: *[TokenMax v20\1000]*
Object Type: *[TokenMax v20\2000]*

The Security Server object provides user authentication to ObServer and connected devices. For an overview, refer to the *Security Server* section.

Use the Editor Sign-in object to sign-in before configuring users and groups in the server.

Set the maximum users available in the server using the *Security Server Configuration* object.

Description	Reference	Type
Editor Access Allowed Indicates if the editor has signed-in	MP.S	Obj\NoYes
Editor Sign-in Write the password to sign-in, or any other value to sign-out. A password recovery key is returned when reading this object	MP.L	Obj\Text; Adjustable
Change Editor password Once signed-in, set a new editor password	ML.P	Obj\Text; Max 20 chars; Adjustable
Next available user Indicates the next unused user record available	FT	Obj\Num; Range 0..400
Area x Label Label of Security Area x, where x is in the range 1...8	Ax	Obj\Text; Max chars 20; Adjustable
Privilege Level x Label Label of Security Privilege Level x, where x is in the range 0…7	Lx	Obj\Text; Max Chars 20; Adjustable
Group x Set privilege levels for a group of users. The group number, *x*, is in the range 1…31	G*x*	Fixed Container: *[TokenMax v20\Group]*
User *x* Configure a user's authentication credentials. The user number, *y*, is in the range 1…2000, depending on the maximum users available set in the configuration object	U*y*	Fixed Container: *[TokenMax v20\User]*

Security Server Group

Object Type: *[TokenMax v20\Group]*

A Security Server Group object contains privilege levels for a *group of users*.

Description	Reference	Type
Name Group name or description	N	Obj\Text; Max chars 20; Adjustable
Enabled Enable or disable all users that are a member of this group	E	Obj\NoYes; Adjustable
Privilege Level in Area *x* Area *x* privilege level, where *x* is in the range 1...8	P*x*	Obj\Num; Range 0...7; Adjustable

Security Server User

Object Type: *[TokenMax v20\User]*

A Security Server User object contains *user information* and credentials to provide authentication.

Description	Reference	Type
Name Username or description	N	Obj\Text; Max chars 20; Adjustable
Enabled If set to yes, the user is enabled if all groups that they belong to are also enabled	E	Obj\NoYes; Adjustable
User ID/Card Sign-in username or access card number	ID	Obj\Text; Max Chars 20; Adjustable
Password This object always has a value of '****' when read	PW	Obj\Text; Max chars 20; Adjustable
Token This is a combination of the User ID and coded password	T	Obj\Text; Max chars 30
Group x Group x membership information, where x is in the range 1…3 When a user is a member of a group, then the group's privilege levels are combined with the user's own.	Gx	Obj\Num; Range 0…31; Adjustable
Privilege Level in Area *x* User's privilege level in Area *x*, where *x* is in the range 1...8	P*x*	Obj\Num; Range 0...7; Adjustable
Valid Start Date Optionally enables a temporary user between the start and end dates specified. For permanent users, set the start and end dates to the same value	SD	Obj\Date; Adjustable
Valid End Date Optionally enables a temporary user between the start and end dates specified. For permanent users, set the start and end dates to the same value	ED	Obj\Date; Adjustable
Last Validation Date Date the user was last validated successfully	LD	Obj\Date

Data Transfer

Object Type: *[TransMax v14\100]*
Object Type: *[TransMax v14\500]*
Object Type: *[TransMax v14\1000]*

A Data Transfer object contains up to 1000 transfers, where each transfer reads from one place and then writes to another. For an overview, see the section on *Transferring Values*.

Set the maximum transfers available using the *Data Transfer Configuration* object.

Description	Reference	Type
Transfer *x* Configure a transfer's information. The transfer number, *x*, is in the range 1..1000, depending on the maximum transfers available set in the configuration object	T*x*	Fixed Container: *[TransMax v14\Transfer]*

Transfer

Object Type: *[TransMax v14\Transfer]*

A Transfer object contains the following objects:

Description	Reference	Type
Source Object Object reference to read	SO	Obj\Obj; Adjustable
Source Read Rate Frequency to read the value. See *Reading the Value*	SR	Obj\ENum; Range 0...9; Adjustable Values: 0=ASAP, 1=Never, (2 and 3 not allowed,) 4=1m, 5=5m, 6=15m, 7=1h, 8=4h, 9=12h
Source Read Fails Count of times the remote object has continuously failed to read	SF	Obj\Num; Range 0...9
Value Transfer's current value	V	Obj\Text; Max chars 32; Adjustable
Destination Write Object Object reference to write	DO	Obj\Obj; Adjustable
Destination Write Rate Frequency to write the value. The value is always written on a change-of-value. Set a rate to perform an additional background write periodically. See *Writing the Value*	DR	Obj\ENum; Range 0...10; Adjustable Values: 0=COV, 1=Never, (2 and 3 not allowed,) 4=1m, 5=5m, 6=15m, 7=1h, 8=4h, 9=12h, 10=Every 00:00
Destination Write Fails Count of times the remote object has continuously failed to write	DF	Obj\Num; Range 0...9

Alarm Translator

Object Type: *[AlmXlate v11]*

The Alarm Translator object contains 1000 translator entries that are first used to translate individual text fields within an incoming alarm. Following this, Translator then searches its *comma-separated value (csv) file* to translate the entire alarm message.

All alarms arriving or being produced by ObServer are passed to the ALARM object for translation. For an overview, see the section on *Alarm Basics*.

Description	Reference	Type
Route New Alarm Alarm generated by ObServer are automatically sent to this object	ALARM	Obj\Alarm; Adjustable
Last Alarm Received This shows the last alarm received, to aid with debugging	LA	Obj\Alarm
Translator *x* Translator *x* information, where *x* is in the range 1...1000	X*x*	Fixed Container: *[AlmXlate v11\Xlator]*

Alarm Translator Examples

Consider the following translator entries:

	Translate From text	Translate To text
X1	Controller 22	Boiler Controller
X2	Input Sensor	Return Air Temp
X3	Sensor Fault	Sensor Fail
X4	Sensor High	High Alarm
X5	Sensor Low	Low Alarm
X6	Panel 1 Loop 2 Device 3 Zone 4	Entrance Lobby

..and the following csv file data:

From			To Translation			
System	**Point**	**Condition**	**System**	**Point**	**Condition**	**Priority**
BMS	HTHW PUMPS SEC	OCCURRED	Engineering	Boiler House Pump	Trip	2
BMS	VCB ARCH INT	OCCURRED	Security	Archive Store	Intruder Alarm	1
BMS	HSDU FILTER ALM	OCCURRED	Engineering	Room 1030	Filter Dirty	3

Using this configuration data, alarms would be translated as follows…

Example 1: All three alarm fields translated (using translator entries only)

	System	Point	Condition	Priority
From	Controller 22	Input Sensor	Sensor Fault	*any*
To	Boiler Controller	Return Air Temp	Sensor Fail	*any*

Example 2: Only alarm's system field is translated

	System	Point	Condition	Priority
From	Controller 22	Input State	Off	*any*
To	Boiler Controller	Input State	Off	*any*

Example 3: Only alarm's condition text is translated

	System	Point	Condition	Priority
From	Controller 24	Input Sensor	Sensor High	*any*
To	Controller 24	Return Air Temp	High Alarm	*any*

Example 4: Entire alarm translated, including priority (using CSV entry)

	System	Point	Condition	Priority
From	BMS	VCB ARCH INT	OCCURRED	*any*
To	Security	Archive Store	Intruder Alarm	1

Alarm Translator Entry

Object Type: *[AlmXlate v11\Xlator]*

The Alarm Translator Entry object contains text for a particular translation. All three text fields in an alarm message are searched – system, point, and condition. The From Text (F) must match the entire field exactly to be replaced with the To Field (T).

Description	Reference	Type
From Text Text to match within an alarm message text field	F	Obj\Text: 62 chars; Adjustable
To Text If the From Text (F) matches, text to replace alarm message field with	T	Obj\Text: 62 chars; Adjustable

Alarm Prioritizer

Object Type: *[AlmPrior v10]*

The Alarm Prioritizer object contains 100 reprioritizer entries that specify how alarms should be checked for re-prioritisation.

All alarms arriving or being passed by Alarm Translator are passed to the ALARM object for reprioritisation. For an overview, see the section on *Alarm Basics*.

Description	Reference	Type
Route New Alarm	ALARM	Obj\Alarm; Adjustable
Reprioritizer *x* Reprioritizer *x* information, where x is in the range 1...100	P*x*	Fixed Container: *[AlmPrior v10\Prior]*

Alarm Prioritizer Entry

Object Type: *[AlmPrior v10\Prior]*

The Alarm Prioritizer Entry object contains information about reprioritising alarms.

If the incoming alarm fields matches all of System contains, Point contains, and Condition contains (any blank entries will match), then the alarm is reprioritised to the New Priority.

Description	Reference	Type
Alarm System field contains Matching can occur anywhere in the field, and is case sensitive	SC	Obj\Text; Adjustable; Max.chars:40
Alarm Point field contains Matching can occur anywhere in the field, and is case sensitive	PC	Obj\Text; Adjustable; Max.chars:40
Alarm Condition field contains Matching can occur anywhere in the field, and is case sensitive	CC	Obj\Text; Adjustable; Max.chars:40
New Priority If an alarm is reprioritized, and this entry is 0, the alarm is discarded	NP	Obj\Num; Adjustable; Range: 0…9

Alarm Delivery

Object Type: *[AlmRoute v11]*

The Alarm Delivery object contains 16 destinations to route North-format alarms received.

All alarms arriving or being produced by ObServer are passed (via Alarm Translator and Alarm Prioritizer) to the ALARM object for distribution. For an overview, see the section on *Alarm Basics*.

Description	Reference	Type
Route New Alarm	ALARM	Obj\Alarm; Adjustable
Destination *x* Destination *x* information, where x is in the range 1..16	D*x*	Fixed Container: *[AlmRoute v11\Dest]*

Alarm Delivery Destination

Object Type: *[AlmRoute v11\Dest]*

An Alarm Delivery Destination object contains the following objects:

Description	Reference	Type
Label Description of this destination	L	Obj\Text; Max 20 chars; Adjustable
Enable If set to yes, alarms can be delivered to this destination	E	Obj\NoYes; Adjustable
Destination Object Object reference to route alarms. Typically this object reference ends '.ALARM'	R	Obj\Obj; Adjustable
Add Device Label If set to yes, prefixes the alarm system field with ObServer's label	CL	Obj\NoYes; Adjustable
Fail State Indicates the Destination Object is failing to accept alarms	F	Obj\NoYes
High Priority Only alarms with a priority number equal or greater than this value are delivered. To disable priority filtering, set the high and low priority to the same value. See *Filtering Delivery*	HP	Obj\Num; Range 0..9; Adjustable
Low Priority Only alarms with a priority number equal or less than this value are delivered. To disable priority filtering, set the high and low priority to the same value.	LP	Obj\Num; Range 0..9; Adjustable
Comparison Method Optional text filter. Specify the method to use along with up to three comparison strings. See *Filtering Delivery*	C	Obj\ENum; Range 0..; Adjustable Values: 0=Begins with, 1=Contains, 2=Not begin with, 3=Not contain
Comparison String *x* Text string to use with the comparison method. The string number, *x*, is in the range 1...3	S*x*	Obj\Text; Max 20 chars; Adjustable
Any Group Join this destination to the any group. When one destination accepts the alarm, it will not be sent to any other destinations in the group. See *Delivery to a Group*	A	Obj\NoYes; Adjustable

Alarm Fields

Alarms are sent by the delivery destination to indicate a fail state.

The Alarm Delivery module places the following information into the North-format fields:

System – from the Alarm Delivery Label object (O.AR.DL)

Point – from the delivery destination's Label object (L), or 'Destination *x*' if no label is set

Condition – either 'Comms Fail' or 'Comms Ok'

Priority – set to '1'

Date & Time – from ObServer

Alarm Examples

System	Point	Condition	Priority	Date	Time
Alarm Delivery	Destination 1	Comms Fail	1	23/03/15	13:33:59
Alarm Delivery	Destination 1	Comms Ok	1	24/03/15	07:30:16

Alarm History

Object Type: *[AlarmHistory v11\100]*
Object Type: *[AlarmHistory v11\500]*

An Alarm History object contains a *list of the last 100 or 500 alarms* received.

Deliver new alarms, using *Alarm Delivery*, to the ALARM object.

Description	Reference	Type
Alarm count To clear all alarms, set value to '0'	C	Obj\Num; Range 0...500; Adjustable
Store new alarm	ALARM	Obj\Alarm; Adjustable
Alarm *x* List of alarms, where Alarm 1 is the last alarm received. The alarm number, *x*, is in the range 1...500	H*x*	Obj\Alarm

Alarm Store

Object Type: *[AlarmStore v13]*

An Alarm Store object contains a *list of up to 4000 alarms* waiting for user acknowledgement. Use WebView or AlarmManager to view and manage the alarms in the store.

Deliver new alarms, using *Alarm Delivery*, to the ALARM object. If you want to archive alarms immediately, deliver new alarms to the ARCHIVE object.

Description	Reference	Type
Store Name	L	Obj\Text
Alarm Count Total number of alarms in the store	AC	Obj\Num: 0…4000
New Alarms Total number of new alarms in the store	NC	Obj\Num: 0…4000
Free Count Space available in the store	FC	Obj\Num: 0…4000
Alarms Present Indicates alarms are in the store	AP	Obj\NoYes
New Alarms Present Indicates new alarms are in the store	NP	Obj\NoYes
Last Alarm Received	LA	Obj\Alarm
Store new alarm	ALARM	Obj\Alarm; Adjustable
Archive new alarm	ARCHIVE	Obj\Alarm; Adjustable
Archive Error Indicates the archive file is unavailable	AE	Obj\NoYes

Alarm Emailer

Object Type: *[AlmEmail v22]*

An Alarm Emailer contains 127 destinations to send alarms, plus a system ALARM object that sends to multiple destinations based on their comparison settings. The email destinations are set in the *Alarm Email Configuration* object.

Description	Reference	Type
Destination Label The label of the destination, where *x* is in the range 1...127, depending whether each has been set up	D*x*	Fixed Container: *[AlmEmail v22\Dest]*
Deliver New Alarm Delivers an alarm to multiple destinations based on the comparison settings configured in each	ALARM	Obj\Alarm; Adjustable

Alarm Emailer Destination

Object Type: *[AlmEmail v22\Dest]*

An Alarm Emailer Destination object contains a queue of alarms to send via email to a destination.

Deliver new alarms, using *Alarm Delivery*, to the ALARM object.

Description	Reference	Type
Deliver New Alarm	ALARM	Obj\Alarm; Adjustable
Alarms in Queue	C	Obj\Num; Range 0...31

North IP Devices

Object Type: *[IPBus Net]*

The North IP Devices object provides access to other *North devices on the IP network*.

Scan the object to automatically discover North IP devices on the local sub-network, or add a device via the *North IP Devices Configuration* object.

Description	Reference	Type
North Device's Label Access device with label *North Device's Label*	*ref*	Variable container. Type varies depending on the North Device. Typically this is *[ObSys]* for other ObServer based devices, or *[Commander v20\Device]* for Commander.

Appendix A – ObServer.ini File

ObServer saves its basic parameters in a file called ObServer.ini file within the ObSys data folder.

The ObServer Site ID, usually 'DefaultSite', is used as a section title within the file. Within the section the following values may be stored:

Item	Value
Label	The current *Site Label*
OSM*x*	ObServer Module to load into *interface x*
StartTime	When ObServer was last started, in the format: <date>|<time>
Reset Counter	ObServer Reset Count
Hidden	Whether the ObServer *window is hidden*, where: 0=No, 1=Yes
X	ObServer Window screen x-coordinate, if visible
Y	ObServer Window screen y-coordinate, if visible
Watch	Set to 'COM'<port> or 'OSM'<num> to view communication on COM port <port> or view requests from ObServer Module <num>
RecordMode	Mode to record to file, where: 0=Off, 1=on until full, 2=on until close, 3=always
RecordFile	Full file name of file to record into. Typically: 'C:\ProgramData\North Building Technologies\ObSys\ObServer.txt'
RecordSize	Maximum size of record file, before it is recreated
RecordComms	Whether to record COM port communications, where: 0=No, 1=Yes
RecordObserver	Whether to record ObServer running information, where: 0=No, 1=Yes
RecordOsmMsgs	Whether to record OSM object communications, where: 0=No, 1=Yes
RecordFastComms	Whether to record synchronised object communications, where: 0=No, 1=Yes
FastObServer	ObServer by default acts as a processor-sharing application. Set this option to allow ObServer to run as fast as possible. Set the value to 'Y' or 'Yes' to enable
ObserverSpeed	When not in fast comms, count of idles to perform before relinquishing control back to the operating system
LicType	Type of last dongle found
LTO	Time offset to apply to UTC to determine local time, in seconds
TZ*x*	Time Zone Daylight Saving information x (where x is in the range 1...20) which defines when a shift to the local time is applied, with value in the format: <date>|<time>|<offsetsecords>
Start*x*	*Command line x* to trigger when ObServer starts, where *x* is in the range 1...20
A*x*	Alias x definition (where x is in the range 1...256) describing an object translation, in the format: <from-obj>,<to-obj>
OnlyShowAliases	Whether a scan returns only items defined in Aliases, Set the value to 'Y' or 'Yes'
StorePeriod	Period between storing Module data to disk, in minutes
Enhancements	Disables other ObSys applications, such as ObView, AlarmManager and DataManager, when set to anything other than 'Y' or 'Yes'

Appendix B - Ethernet/IP Protocols

Besides the protocols used by the operating system, ObServer itself may implement the following protocols across one or more of the PCs Ethernet ports.

Drivers may also use other ports. For information on a driver's IP port usage, refer to the driver manual.

Telnet

If enabled, ObServer supports an incoming Telnet session.

Ports used: TCP 23 (local)

Simple Mail Transfer Protocol (SMTP)

Once Alarm Emailer is configured, this SMTP client sends emails to the specified server.

Ports used: TCP 25 (configurable, remote), TCP ephemeral (local)

Hypertext Transfer Protocol (HTTP)

The Web Server, if enabled, serves simple HTML pages when requested.

Ports used: TCP 80 (configurable, local)

North/IP

The North/IP Protocol allows North products that support IP networking to communicate.

Ports used: UDP 37926

Warranty

North warrants goods of its manufacture as being free of defective materials and faulty workmanship for one year from the date of purchase. If this product should become defective, please contact our support team. This warranty becomes invalid if the product has been tampered with or used in an environment that is unsuitable.

If you require further help, please contact our support team on +44 (0) 1273 694422, or visit *www.northbt.com/support*

ObServer Versions

Build Date	Details
01/03/2014	ObServer: now starts automatically when launching apps. ObServer has a new-style start window, and the icon has been removed from task bar. Alarm Emailer: AlmEmail v2.2 features improved routing of alarms. Alarm History: General performance improvements. Alarm Store: Added object to delete all alarms when store is full. Essential Data: This new version of UserData v2.3 includes a new profile object type. North IP Devices: IPBus v2.1 includes enhancements to support ObView 'Extras' menu features, and improved network scanning. ObVerse Processor: New modules types added include Linearize, BitToNum, NumToBit, BitsToByte, and ByteToBits. Security Server: TokenMax v2.0 resolves an issue of writing an incorrect privilege level. Telnet: Improved performance of incoming TCP/IP connections. Time Control: This new version of CalTimer v2.0 includes a new profile object type and 20 new profile objects. WebView Server: WebView v1.3 has been updated to support the new profile object type, and the latest Time Control module. Some minor issues have also been resolved.
20/08/2014	Essential Data: for object write actions, changed ASAP to write on change-of-value. WebView: updated RemoteNum and RemoteRange to include a state for failed object read, and audio support. Alarm Store: resolved issue with archive file not reducing to defines maximum size. Telnet: improved response with large messages. Resolved issue in closing active session when Ethernet disconnected. Time Control: resolved issue with value written to destination object.
31/03/2016	ObVerse Processor: new Raise-Lower and Usage modules Essential Data: number of objects per page now configurable. Total database values expanded to 1280. Load the ExtraData driver to increase this to 2304 values. ObServer: Support for FastComms. Used by drivers to talk to each other faster in the same North device. WebView: minor enhancements Data Transfer: resolved an issue when the destination write rate was set to 1 min. This prevented the transfer from reading the source object.
01/11/2016	WebView: version 1.4 with new objects to disable WebView Editor access (preventing HTTP PUT and DELETE methods for enhanced security). Changed operation of 'show logs' object to now only disable the page list navigation. Direct in-page links are now possible Essential Data: value of object type 'Times' optimized to allow more on-off time periods ObVerse Processor: resolved issue with Raise-Lower module.
17/03/2017	ObVerse Processor: you can now link a module's inputs and outputs directly to values in Essential Data WebView: resolved issue when displaying Data Manager logs with long names.
01/11/2017	Telnet: supports up to 4 sessions. Optimisations to Alarm Store, Alarm Email, Essential Data, and ObVerse Processor.
07/12/2017	Essential Data: resolved issue with Task Control object. On reading this object it reports an incorrect value, and adjusting may cause operation to be disabled.
01/03/2018	Essential Data: fixed issue when an object is configured with a remote action of 'read', and the current value does not change for 24 days, then Essential Data will stop reading the object for the next 24 days, after which it will start reading again. If ObServer is restarted during this time, then reading will resume and the timer restarted.
01/11/2018	ObServer: speed and performance optimization North IP Devices: Speed improvements ObSave: the time taken to save a complete Commander object to a backup file has been reduced by 80%. In addition, ObSave can now include ObVerse Processors as part of the backup. ObView, ObLoad, ObSearch applications also updated. ObVerse Processor: option to cancel a Get ObVerse or Put ObVerse action. ObvEditor application has a smoother page refresh in Watch mode, enforces property references to four characters, and resolves a few issues when repositioning an item. ObVerse Processor resolves

Build Date	Details
	issues with the following modules: Rescale (divide by zero), On-Off-Delay (restart), and Counter (reset when input 1). Data Transfer: added support for transferring values from times and profile object types.
01/06/2019	Essential Data: added Alarm Delay object (AD). Resolved issue when object value adjusted multiple times before remote device acknowledges first write. In this situation, last adjustment was not written to remote device. WebView: faster page loading from logs containing a large amount of data, and pages with Essential Data values. ObLoad and ObSave: applications updated with more command line options.
01/09/2019	Essential Data: when Remote Fails are 9, retry period reduced from 15min to 5min.
28/07/2020	Essential Data: resolved issue when linking to invalid page/object within ObvProcessor Data Transfer: resolved issue in calculating value change, resulting in write action when not needed Telnet Server: Resolved an issue with Telnet not re-enabling after being disabled.
25/08/2021	Essential Data: resolved issue with communications alarm state clearing, when value had not changed. North IP Devices: resolved issue with comms check when reference contained an alias.
02/06/2022	ObVerse Processor: Added Times-State, Profile-Value, Date-Pulse, and Smooth module types; and Profile, Times, and DateTime property types. Essential Data: Float-type object, added 0.5 dp option
06/09/2022	Alarm Emailer: improved sending on poor connections. Added fail message object (FM)
03/02/2023	WebView: add support for LDAPS. Updated alarms sent. Alarm Emailer: resolved issue causing email to resend
01/06/2024	North IP Devices: operates with network port translation (replies sent to UDP source port) ObVerse Processor: general performance improvements.

Next Steps…

If you require help, contact support on 01273 694422 or visit *www.northbt.com/support*

North Building Technologies Ltd
+44 (0) 1273 694422
support@northbt.com
www.northbt.com

Author: TM
Checked by: JF

Document issued 29/05/2024.