FlexIS DataInterface Documentation

Rev A, 18.11.2008
Rev B, 10.02.2009
Rev C, 12.06.2009
Rev D, 17.02.2011
Rev E, 03.03.2011
Rev F, 24.11.2011
Rev G, 12.04.2012
Rev H, 20.07.2012

Author: Melchior Rabe (Jetter AG)
Transfer: Andreas Helfenstein (Emhart Glass SA)

CONTENT

1 General overview

The FlexIS data interface is a software system allowing third party systems to retrieve data from FlexIS machines. The software runs on the FlexIS line server so access to all machines is possible through a single point of entry. The system is designed as server client architecture that offers the service through tcp ip (direct socket connection) and a http tunnel.

2 Basic structure

A third party system (called client from here on) sends a request to the FlexIS data interface (called server from here on). The server generates the appropriate response and sends it back to the client. Then the server disconnects. For each request a new connection is established.

All timestamps are combined date time values according to ISO 8601.

2.1 Request

The request is a xml document transferred either through a direct socket connection or in the body section of a http get request.

The root element contains a version number (currently `1.00.00.000`) and a head and a body section. The target element contains a string identifying the machine from which the data should be provided. The names correspond to that ones used in the line server software. The server can provide multiple types of data like production data, alarms, status information and event change logger data. The action element must currently have the value `get`. Other actions might be defined in future.

The body element contains signals and a filter element. A signal identifies a structural data element like "gobs cut" or "alarm text". The body can contain several different signals. The filter is an optional element (in general). If no filter is used the system might fall back to its internal defaults. The body must not contain more than one filter element. A filter can combine its elements with an *and* or an *or* relation. Possible child elements are filterItem defining an atomic criteria or filter defining a sub filter. A filterItem has an attribute defining the signal to be filtered and a value to be matched. It can filter on signals that don't occur in the signals list (or even can't occur in the list).

```
<?xml version="1.0" encoding="utf-8"?>
<FlexisDataInterfaceRequest
 xmlns="xsdFlexisDataInterfaceRequest"
 version="1.00.00.000">

  <head>
    <target>Line_29</target>
    <type>ProductionData</type>
    <action>get</action>
  </head>
  <body>
    <signal>SC_DEL</signal>
    <filter type="and">
      <filterItem signal="GRANULARITY">60</filterItem>
      <filterItem signal="TIME_FROM">2008-11-03T12:00:00</filterItem>
    </filter>
  </body>
</FlexisDataInterfaceRequest>
```

Example 1 Requesting production data (from all sections)

It is possible to fetch data from all sections at a time omitting the controller prefix.

```
<?xml version="1.0" encoding="utf-8"?>
<FlexisDataInterfaceRequest
 xmlns="xsdFlexisDataInterfaceRequest"
 version="1.00.00.000">

  <head>
    <target>Line_29</target>
    <type>ProductionData</type>
    <action>get</action>
  </head>
  <body>
    <signal>SC_1.SC_DEL</signal>
    <signal>MC.MC_CUT</signal>
    <filter type="and">
      <filterItem signal="GRANULARITY">60</filterItem>
      <filterItem signal="TIME_FROM">2008-11-03T12:00:00</filterItem>
    </filter>
  </body>
</FlexisDataInterfaceRequest>
```

Example 2 Requesting production data (from section 1 and machine controller)

2.2 Response

The response sent from the server to the client has a structure similar to the request. The root element contains a version attribute and contains a head and body child. Additional is the third child, a `messageList` (optional) that contains messages that give information about events happened while processing the request. The response contains either just a `messageList` (e.g. if the request was completely unreadable) or head and body (if no messages were created) or all three elements. The head contains the head elements of the request, a description of the target and the timestamp of the creation of the response. The body contains the data requested. It is grouped in sources (e.g. machine controller, section controller, …). Each source contains a list of samples containing all data for the given timestamp. The data elements have a signal attribute and an optional attribute type.

See chapter 3.1 and 3.3 for a specific response description depending on the data type.

```
<?xml version="1.0" encoding="UTF-8"?>
<FlexisDataInterfaceResponse version="1.00.00.000">
  <head>
    <target>Line_29</target>
      <type>ProductionData</type>
        <description></description>
        <timestamp>2008-11-17T14:47:36</timestamp>
  </head>
  <body>
    <source name="SC_1">
      <sample timestamp="2008-11-17T14:38:00">
        <data name="SC_DEL">NaN</data>
      </sample>
      <sample timestamp="2008-11-17T14:39:00">
         <data name="SC_DEL">1.23e2</data>
      </sample>
      <!-- more samples following -->
    </source>
    <!-- more sources following -->
  </body>
</FlexisDataInterfaceResponse>
```

Example 3 Sample response document

3 The request types

3.1 Round robin data

Round robin data is a generic request type. It is used for data stored in round robin databases. A signal identifier is composed by the database name and the signal name separated by a dot. E.g. `SC_1.SC_DEL` stands for signal `SC_DEL` from database `SC_1`. In principal there exist two types of signals, basic signals stored in the database and aggregated signals combining two or more basic signals to provide more convenience.

The signals and databases available are subject to the derived request types.

The data is returned (unless specified separately) as scientific notated float values (e.g. `5.9736e24`). If there is no data available the value `NaN` is returned. By default, total numbers for the requested granularity are returned. Examples are shown in Appendix F (Configuration Options).

3.2 Production data

The production data request is derived from round robin data and can be retrieved for a complete machine in one request.

The data is grouped in three parts, first the gob forming data collected by the machine controller (signal prefix `MC_`), then the data collected by each section (signal prefix `SC_`) and finally the data collected by the ware handling controller (signal prefix `WHC_`). As each section has its own database all signals starting with the section prefix are internally split into sections. So it is sufficient to add the signal once to retrieve data from all sections installed. If a section signal

should go only to a specific section the general round robin approach (`database.signal`) can be used.

3.3 Sql data

Sql data (stored in sql databases within the FlexIS system) offers more detailed opportunities for filtering than round robin data. Combinations of *and* and *or* filters can be used as well as comparison and bitwise operators. As bitwise operators work only in filters (which require actually a Boolean expression) the result of the bitwise expression is compared to zero and returns true if it is unequal zero. Take care that xml-entities like ">" or "&" have to be escaped in the request.

Operator	Meaning
& (&)	Bitwise *and* operation
|	Bitwise *or* operation
^	Bitwise *exclusive or* operation
< (<)	Less than
> (>)	Greater than
=	Equals (implicit operator if none is given)
<> (<>)	Unequal

```
<?xml version="1.0" encoding="utf-8"?>
<FlexisDataInterfaceRequest
  xmlns="xsdFlexisDataInterfaceRequest" version="1.00.00.000">
  <head>
    <target>Line_29</target>
    <type>SqlData</type>
    <action>get</action>
  </head>
  <body>
    <signal>SC_STATUS.TIMESTAMP</signal>
    <signal>SC_STATUS.SECTION</signal>
    <signal>SC_STATUS.TEXT</signal>
    <filter type="and">
      <filter type="or">
        <filterItem signal="SC_STATUS.VALUE">&amp;1</filterItem>
        <filterItem signal="SC_STATUS.VALUE">&amp;2</filterItem>
      </filter>
      <filterItem signal="SC_STATUS.SECTION">&gt;=5</filterItem>
      <filterItem signal="SC_STATUS.SECTION">&lt;=10</filterItem>
    </filter>
  </body>
</FlexisDataInterfaceRequest>
```

Example 4 Requesting status information from sections 5 to 10 having bits 0 and 1 set.

<= (<=)	Less than or equal
>= (>=)	Greater than or equal

Example 4 shows how to use the operators in a filter. These filter settings are internally resolved to `((0 <> (1 & VALUE) OR 0 <> (2 & VALUE)) AND SECTION >= 5 AND SECTION <= 10)`.

3.3.1 Alarms

Alarms are stored at two places. ALARMS contains the alarms that are still pending or not yet acknowledged. All other alarms are stored in the ALARM_HISTORY. An alarm consists of two records. First an alarm is set, then (if the cause is gone) it is reset. The set record has a positive error code and the corresponding reset has the same number multiplied by minus one. As long as the duration of an alarm is not needed it is strongly recommended to filter for positive error codes as this reduces the load on all participating systems. Furthermore most alarms are so called "dirac alarms" sending the reset just after the set.

As the datasource for this information is a sql database row filtering can be used and is recommended.

Take care that "greater / less than" characters in the request have to be replaced by their xml entities (see Example 5; filters for code and timestamp).

```
<?xml version="1.0" encoding="utf-8"?>
<FlexisDataInterfaceRequest
  xmlns="xsdFlexisDataInterfaceRequest"
  version="1.00.00.000">

  <head>
    <target>Line_29</target>
    <type>SqlData</type>
    <action>get</action>
  </head>
  <body>
    <signal>ALARM_HISTORY.TIMESTAMP</signal>
    <signal>ALARM_HISTORY.CONTROLLER</signal>
    <signal>ALARM_HISTORY.CODE</signal>
    <signal>ALARM_HISTORY.TEXT</signal>
    <filter type="and">
    <filterItem signal=" ALARM_HISTORY.CONTROLLER">1</filterItem>
    <filterItem signal=" ALARM_HISTORY.CODE">&gt; 0</filterItem>
    <filterItem signal=" ALARM_HISTORY.TIMESTAMP">&gt;=2010-06-
22T00:00:00</filterItem>
    </filter>
  </body>
</FlexisDataInterfaceRequest>
```

Example 5 Requesting alarms from section 1 newer than June, 22nd 2010

3.3.2 Status

Status information is available for section controllers (SC_STATUS), machine controller (MC_STATUS), and warehandling controller (WHC_STATUS in case of a dedicated whc, MC_STATUS in case of a mc-combi). In case of a mc-combi mc and whc status information is stored in MC_STATUS and has to be selected by filtering the respective addresses. A status word contains up to bit coded different status. Bit number 0 indicates the least significant bit, number 31 the most significant bit.

3.3.3 UI logger

The UI_LOGGER gives information about changes users made at the user interface. Currently only changes in event timing are logged.

APPENDIX

A Production data signals

A.1 Filter signals

Filter signals might be used to define the period of time and resolution of the data.

- **Start time** (`TIME_FROM`)
  Timestamp of the first sample to be returned given as timestamp according to ISO 8601 or any format rrdtool understands[1]. Defaults to `now-5min`.
- **End time** (`TIME_TO`)
  Timestamp of the last sample to be returned given as timestamp according to ISO 8601 or any format rrdtool understands. Defaults to `now`.
- **Resolution** (`GRANULARITY`)
  given in seconds. Defaults to 60 seconds.

A.2 General signals (defined on all controllers)

- **Job identifier** (`JOB_ID`)
  The id of the last job loaded at the specific sample.

A.3 Gob Forming (machine controller)

- **Number of gobs cut** (`MC_CUT`)
  Gobs cut by the shear. The counter increases by number of gobs[2] with every shear cut.
- **Number of gobs cut per section** (`MC_CUT_SC_X`)
  X is a number between 1 and 12.
- **Number of gobs not cut** (`MC_NOT_CUT`)
  Gobs not cut because the shear was stopped. This counter increases every time the shear would cut, if it were running, by number of gobs.
- **Number of gobs not cut per section** (`MC_NOT_CUT_SC_X`)
  X is a number between 1 and 12.
- **Number of gobs delivered** (`MC_DEL`)
  Gobs delivered by the gob distributor
- **Number of gobs delivered per section** (`MC_DEL_SC_X`)
  X is a number between 1 and 12.

A.4 Bottle Forming (per section)

- **Number of gobs requested** (`SC_REQ`)
  Gobs requested by the section.
- **Number of gobs delivered** (`SC_DEL`)
  Gobs delivered to the section.

[1] See http://oss.oetiker.ch/rrdtool/doc/rrdfetch.en.html (section AT-STYLE TIME SPECIFICATION)

[2] Number of gobs is the amount of glass streams processed in parallel (single gob, double gob, triple gob and quadruple gob).

- **Number of gobs lost by pressing MS[3] (**SC_LOST_MS**)**
  The number of gobs lost in the section setting it to MS while it is running (with glass).
- **Number of gobs not requested because of "stop" (**SC_NRQ_STOP**)**
  Gobs not requested because the section was set to "normal stop" or "maintenance stop".
- **Number of gobs not requested because of "delivery off" (**SC_NRQ_DEL_OFF**)**
  Gobs not requested because the delivery button was set to "delivery off".
- **Number of gobs not requested because of "manual swab" (**SC_NRQ_MAN_SWAB**)**
  Gobs not requested because the section entered a "manual swab" cycle.
- **Number of gobs not requested because of "alternate swab" (**SC_NRQ_ALT_SWAB**)**
  Gobs not requested because the section entered a 'alternate swab' cycle.
- **Number of gobs rejected because of "cold molds" (**SC_RJ_COMO**)**
  Gobs rejected because the machine entered a 'cold molds' cycle.
- **Number of gobs rejected because of "manual swab" (**SC_RJ_MAN_SWAB**)**
  Gobs rejected because the machine entered a 'manual swab' cycle.
- **Number of gobs rejected because of "alternate swab" (**SC_RJ_ALT_SWAB**)**
  Gobs rejected because the machine entered a "alternate swab" cycle.
- **Number of gobs rejected by RPC[4]**
  Gobs rejected on request of the RPC (by cavity)
  SC_RJ_SRV_1, SC_RJ_SRV_2, SC_RJ_SRV_3, SC_RJ_SRV_4
  For convenience the sum of these four counters is defined, too
  SC_RJ_SRV
- **Number of gobs rejected by any panel input**
  All combinations of the following four attributes are possible

Panel type	Switching type	Side	Cavity
Either static (STA) inputs (e.g. T600 or PPC[5]) or strobed (STR) inputs (e.g. T6000)	Either single (SGL) or continuous (CON) reject	Either blank (BK) or blow (BW) side	The cavity number (1..4)

This generates the following signals:

```
SC_RJ_STA_CON_BK_1, SC_RJ_STA_CON_BK_2,
SC_RJ_STA_CON_BK_3, SC_RJ_STA_CON_BK_4,
SC_RJ_STA_SGL_BK_1, SC_RJ_STA_SGL_BK_2,
SC_RJ_STA_SGL_BK_3, SC_RJ_STA_SGL_BK_4,
SC_RJ_STA_CON_BW_1, SC_RJ_STA_CON_BW_2,
SC_RJ_STA_CON_BW_3, SC_RJ_STA_CON_BW_4,
SC_RJ_STA_SGL_BW_1, SC_RJ_STA_SGL_BW_2,
```

[3] Maintenance Stop

[4] Remote Panel Controller

[5] Plunger Process Control

SC_RJ_STA_SGL_BW_3, SC_RJ_STA_SGL_BW_4,
SC_RJ_STR_CON_BK_1, SC_RJ_STR_CON_BK_2,
SC_RJ_STR_CON_BK_3, SC_RJ_STR_CON_BK_4,
SC_RJ_STR_SGL_BK_1, SC_RJ_STR_SGL_BK_2,
SC_RJ_STR_SGL_BK_3, SC_RJ_STR_SGL_BK_4,
SC_RJ_STR_CON_BW_1, SC_RJ_STR_CON_BW_2,
SC_RJ_STR_CON_BW_3, SC_RJ_STR_CON_BW_4,
SC_RJ_STR_SGL_BW_1, SC_RJ_STR_SGL_BW_2,
SC_RJ_STR_SGL_BW_3, SC_RJ_STR_SGL_BW_4

- **Aggregated signals defined for convenience**
  These signals create sums of the basic panel input signals. All counters for the attribute aggregated are added (e.g. SC_RJ_STA_CON_BK is the sum of SC_RJ_STA_CON_BK_1, SC_RJ_STA_CON_BK_2, SC_RJ_STA_CON_BK_3 and SC_RJ_STA_CON_BK_4). Here again all combinations are defined.

  - Adding all cavities
    SC_RJ_STA_CON_BK, SC_RJ_STA_SGL_BK
    SC_RJ_STA_CON_BW, SC_RJ_STA_SGL_BW
    SC_RJ_STR_CON_BK, SC_RJ_STR_SGL_BK
    SC_RJ_STR_SGL_BK, SC_RJ_STR_SGL_BW
  - Adding blank and blow side
    SC_RJ_STA_CON_1, SC_RJ_STA_CON_2,
    SC_RJ_STA_CON_3, SC_RJ_STA_CON_4,
    SC_RJ_STA_SGL_1, SC_RJ_STA_SGL_2,
    SC_RJ_STA_SGL_3, SC_RJ_STA_SGL_4,
    SC_RJ_STR_CON_1, SC_RJ_STR_CON_2,
    SC_RJ_STR_CON_3, SC_RJ_STR_CON_4,
    SC_RJ_STR_SGL_1, SC_RJ_STR_SGL_2,
    SC_RJ_STR_SGL_3, SC_RJ_STR_SGL_4
  - Adding single and continuous rejects
    SC_RJ_STA_BK_1, SC_RJ_STA_BK_2,
    SC_RJ_STA_BK_3, SC_RJ_STA_BK_4,
    SC_RJ_STA_BW_1, SC_RJ_STA_BW_2,
    SC_RJ_STA_BW_3, SC_RJ_STA_BW_4,
    SC_RJ_STR_BK_1, SC_RJ_STR_BK_2,
    SC_RJ_STR_BK_3, SC_RJ_STR_BK_4,
    SC_RJ_STR_BW_1, SC_RJ_STR_BW_2,
    SC_RJ_STR_BW_3, SC_RJ_STR_BW_4
  - Adding static and strobed inputs
    SC_RJ_CON_BK_1, SC_RJ_CON_BK_2
    SC_RJ_CON_BK_3, SC_RJ_CON_BK_4
    SC_RJ_SGL_BK_1, SC_RJ_SGL_BK_2
    SC_RJ_SGL_BK_3, SC_RJ_SGL_BK_4
    SC_RJ_CON_BW_1, SC_RJ_CON_BW_2
    SC_RJ_CON_BW_3, SC_RJ_CON_BW_4
    SC_RJ_SGL_BW_1, SC_RJ_SGL_BW_2
    SC_RJ_SGL_BW_3, SC_RJ_SGL_BW_4
  - Adding cavities and sides
    SC_RJ_STA_CON, SC_RJ_STA_SGL, SC_RJ_STR_CON, SC_RJ_STR_SGL

- Adding cavities and switching types
  SC_RJ_STA_BK, SC_RJ_STA_BW, SC_RJ_STR_BK, SC_RJ_STR_BW
- Adding sides and switching types
  SC_RJ_STA_1, SC_RJ_STA_2, SC_RJ_STA_3, SC_RJ_STA_4
  SC_RJ_STR_1, SC_RJ_STR_2, SC_RJ_STR_3, SC_RJ_STR_4
- Adding cavities and panel types
  SC_RJ_CON_BK, SC_RJ_SGL_BK, SC_RJ_CON_BW, SC_RJ_SGL_BW
- Adding panel types and sides
  SC_RJ_CON_1, SC_RJ_CON_2, SC_RJ_CON_3, SC_RJ_CON_4
  SC_RJ_SGL_1, SC_RJ_SGL_2, SC_RJ_SGL_3, SC_RJ_SGL_4
- Adding panel and switching types
  SC_RJ_BK_1, SC_RJ_BK_2, SC_RJ_BK_3, SC_RJ_BK_4
  SC_RJ_BW_1, SC_RJ_BW_2, SC_RJ_BW_3, SC_RJ_BW_4
- Adding cavities, sides and switching types
  SC_RJ_STA, SC_RJ_STR
- Adding cavities, sides and panel types
  SC_RJ_CON, SC_RJ_SGL
- Adding cavities, switching and panel types
  SC_RJ_BK, SC_RJ_BW
- Adding sides, switching and panel types
  SC_RJ_1, SC_RJ_2, SC_RJ_3, SC_RJ_4

- **Number of gobs rejected by WHC** (SC_RJ_WHC)
  If cross conveyor or ware transfer stand still, the gob distributor will lead the gobs into cullet.
- **All gobs rejected by a section** (SC_RJ)
  This is the sum of all basic signals containing RJ in their name. This includes Panel inputs, special cycles, RPC and WHC.

A.5 Hot end reject (ware handling controller)

- **Number of bottles rejected by DWR[6] being 'too wide'** (WHC_DWR_WIDE)
  Bottles detected as too wide (e.g. stuck together).
- **Number of bottles rejected being 'too thin'** (WHC_DWR_THIN)
  Bottles detected as too thin (e.g. squeezed).
- **Number of bottles rejected having 'wrong spacing'** (WHC_DWR_SPACE)
  Bottles detected with a space out of the limits.
- **Number of bottles out of DWR** (WHC_DWR_OUT)
  Bottles going out of the DWR equipment.
- **Number of counts on 'user input 1'** (WHC_USR_CNT_1)
  Free user counter input on Ware Handling Controller (input 102).
- **Number of counts on 'user input 2'** (WHC_USR_CNT_2)
  Free user counter input on Ware Handling Controller (input 103).

[6] Downware reject

- **Number of counts on 'user input 3' (**`WHC_USR_CNT_3`**)**
  Free user counter input on Ware Handling Controller (input ?).

A.6 Glass flow

Machine Controller		Section Controller			Warehandling controller
MC_NOT_CUT					
MC_CUT	not delivered	SC_NRQ_STOP			
		SC_NRQ_DEL_OFF			
		SC_NRQ_MAN_SWAB			
		SC_NRQ_ALT_SWAB			
	MC_DEL	SC_REQ	not delivered		
			SC_DEL	SC_RJ_STA_CON_BK_1	
				SC_RJ_STA_CON_BK_2	
				SC_RJ_STA_CON_BK_3	
				SC_RJ_STA_CON_BK_4	
				SC_RJ_STA_SGL_BK_1	
				SC_RJ_STA_SGL_BK_2	
				SC_RJ_STA_SGL_BK_3	
				SC_RJ_STA_SGL_BK_4	
				SC_RJ_STA_CON_BW_1	
				SC_RJ_STA_CON_BW_2	
				SC_RJ_STA_CON_BW_3	
				SC_RJ_STA_CON_BW_4	
				SC_RJ_STA_SGL_BW_1	
				SC_RJ_STA_SGL_BW_2	
				SC_RJ_STA_SGL_BW_3	
				SC_RJ_STA_SGL_BW_4	
				SC_RJ_STR_CON_BK_1	
				SC_RJ_STR_CON_BK_2	
				SC_RJ_STR_CON_BK_3	
				SC_RJ_STR_CON_BK_4	
				SC_RJ_STR_SGL_BK_1	
				SC_RJ_STR_SGL_BK_2	
				SC_RJ_STR_SGL_BK_3	
				SC_RJ_STR_SGL_BK_4	
				SC_RJ_STR_CON_BW_1	
				SC_RJ_STR_CON_BW_2	
				SC_RJ_STR_CON_BW_3	
				SC_RJ_STR_CON_BW_4	
				SC_RJ_STR_SGL_BW_1	
				SC_RJ_STR_SGL_BW_2	
				SC_RJ_STR_SGL_BW_3	
				SC_RJ_STR_SGL_BW_4	
				SC_RJ_COMO	
				SC_RJ_MAN_SWAB	
				SC_RJ_ALT_SWAB	
				SC_RJ_WHC	
				SC_LOST_MS	
				SC_RJ_SRV_1	
				SC_RJ_SRV_2	
				SC_RJ_SRV_3	
				SC_RJ_SRV_4	
				bottles made	WHC_DWR_WIDE
					WHC_DWR_THIN
					WHC_DWR_SPACE
					WHC_DWR_OUT
					WHC_USR_CNT_1
					WHC_USR_CNT_2
					WHC_USR_CNT_3

B Alarm and Alarm history signals

- **Alarm time (**`TIMESTAMP`**)**
  Date and time of the alarm in ISO 8601 format (not necessarily unique).

- **Error code (**CODE**)**
  The error code. A positive number indicates a set, negative numbers a reset.
- **Source of the alarm (**CONTROLLER**)**
  A numeric identifier for each controller. (0: machine controller, 1 to 12: section controller 1 to 12, 13: warehandling controller)
- **Severity (**CATEGORY**)**
  The category indicates the severity of the entry.

Value	Meaning
0	Error high
1	Warning high
2	Message high
3	Error medium
4	Warning medium
5	Message medium
6	Error low
7	Warning low
8	Message low

- **Alarm details (**INFO_1**,** INFO_2**,** INFO_3**,** INFO_4**)**
  Additional values that give more details about the error. The values and their meaning are specific for each error code.
- **Textual representation (**TEXT**)**
  The alarm coded as text as it appears on the UC.

C Section status

C.1 Signals

- **Log time (**TIMESTAMP**)**
  Date and time of the entry in ISO 8601 format (not necessarily unique).
- **Sequence number (ROW_ID)**
  A unique number for each entry in ascending sequence.
- **Section number (**SECTION**)**
  A number from 1 to 12 identifying the section controller.
- **Status word address (**ADDRESS**)**
  The address of the status word. Currently only address = 11 is used.
- **Status word value (**VALUE**)**
  The value of the status word.
- **Textual representation (**TEXT**)**
  The value converted to a comma separated text.

C.2 Status word bit numbers

Address	Bit number	Meaning
11	0	Initializing
11	1	Initialized
11	2	Stopping Maintenance

11	3	Stopped Maintenance
11	4	Stopping Normal
11	5	Stopped Normal
11	6	Stopping Alternate
11	7	Stopped Alternate
11	8	Starting
11	9	Running
11	10	MS Override
11	11	Calibrating
11	12	Calibrated
11	13	Delivery Enabled
11	14	Delivery Disabled Blow
11	15	Stop Alternate Activated
11	16	Emergency Stop
11	17	MS From Running
11	18	Calibration Armed
11	19	Synchronizing
11	20	Synchronized
11	21	Swab Glass On Enabled
11	22	Swab Glass On Active
11	23	Repositioning
11	24	Init Drum
11	25	Output Test Mode
11	26	Stopped After Spec. Cycle
11	27	Axis Setup Mode
11	28	Swab Glass Off Blank
11	29	Swab Glass Off Blow
11	30	Swab Glass Off Quick
11	31	WHHT connected

D Machine controller and warehandling controller status

D.1 Signals

- **Log time (`TIMESTAMP`)**
  Date and time of the entry in ISO 8601 format (not necessarily unique).

- **Sequence number (ROW_ID)**
  A unique number for each entry in ascending sequence.
- **Status word address (**ADDRESS**)**
  The address of the status word.

2700	Machine Contoller (MC)
2701	Shear mechanism
2702	Feeder mechanism (containing Feeder Plunger, Tube Rotation and Tube height axes)
3464	Warehandling Controller (WHC) in case of MC combi

- **Status word value (**VALUE**)**
  The value of the status word.
- **Textual representation (**TEXT**)**
  The value converted to a comma separated text.

D.2 Machine controller bit numbers

Address	Bit number	Meaning
2700	0	Initializing
2700	1	Initialized
2700	2	Emergency stop
2700	3	Stopped maintenance
2700	4	Starting
2700	5	Running
2700	6	Synchronizing
2700	7	Synchronized
2700	8	Changing speed
2700	9	Output test mode
2700	10	EFRA synchronized
2700	11..14	Not used
2700	15	Tandem communication failure
2700	16	Tandem synchronizing
2700	17	Tandem synchronized
2700	18	External feeder synchronized
2700	19..31	Not used
2701	0..15	Not used
2701	16	Shear initialized
2701	17	Shear emergency stop
2701	18	Shear stopped maintenance
2701	19	Shear not stopped maintenance
2701	20	Shear stopped normal

2701	21	Shear starting
2701	22	Shear running
2701	23	Shear jog mode
2701	24	Shear output test mode
2701	25	Gob distributor emergency stop
2701	26	Gob distributor stopped maintenance
2701	27	Gob distributor stopped normal
2701	28	Gob distributor starting
2701	29	Gob distributor running
2701	30	Gob distributor manual mode
2701	31	Gob distributor output test mode
2702	0..1	Not used
2702	2	Feeder plunger stopping maintenance
2702	3	Feeder plunger stopped maintenance
2702	4	Feeder plunger reference running
2702	5	Feeder plunger referenced
2702	6	Feeder plunger stopping normal
2702	7	Feeder plunger stopped normal
2702	8	Feeder plunger starting (automatic or manual)
2702	9	Feeder plunger running (automatic)
2702	10	Feeder plunger zero-Height adjust mode requested
2702	11	Feeder plunger zero-Height adjust mode active
2702	12	Feeder plunger output test mode requested
2702	13	Feeder plunger output test mode active
2702	14	Feeder plunger table loading
2702	15	Feeder plunger table loaded
2702	16..18	Not used
2702	19	Feeder plunger error
2702	20	Feeder plunger axis task busy
2702	21	Feeder plunger emergency stop
2702	22..24	Not used
2702	25	Tube rotation stopped
2702	26	Tube rotation running
2702	27	Tube height stopped
2702	28	Tube height running
2702	29..31	Not used

D.3 Warehandling controller bit numbers

Address (int / ext)	Bit number	Meaning
3460 / 200	0	Initializing
3461 / 200	1	Initialized
3462 / 200	2	Stopping maintenance
3463 / 200	3	Stopped maintenance
3464 / 200	4	Emergency stop
3465 / 200	5	Synchronizing
3466 / 200	6	Synchronized
3467 / 200	7	Changing speed
3468 / 200	8	Init drum
3469 / 200	9	Output test mode
3470 / 200	10	Conveyor auto mode
3471 / 200	11	Ware xfer auto mode
3472 / 200	12	Cross conveyor auto mode
3473 / 200	13	Conveyor stopped normal
3474 / 200	14	Conveyor running
3475 / 200	15	Ware xfer stopped normal
3476 / 200	16	Ware xfer running
3477 / 200	17	Cross conveyor stopped normal
3478 / 200	18	Cross conveyor running
3479 / 200	19	Stacker stopped normal
3480 / 200	20	Stacker running
3481 / 200	21	Stacker auto mode

E UI logger

E.1 Signals

- **Log time (**`TIMESTAMP`**)**
  Date and time of the entry in ISO 8601 format (not necessarily unique).
- **Entry type identifier (**`LOG_ID`**)**
  A number identifying the type of entry.
- **User (**`USER_ID`**)**
  The identifier of the operator that made the change.
- **Job (**`JOB_ID`**)**
  The identifier of the affected job.
- **Operating mode (**`MODE`**)**
  The job mode (online: 1, offline: 0, line server view mode: 2).

- **Logger information (**`INFO_1, INFO_2, INFO_3, INFO_4`**)**
  Additional information specifying the logged event (as text).
- **Textual representation (**`TEXT`**)**
  The logged event in text representation.

E.2 Entry type identifiers

LOG_ID	Text
1	Modification at Section %Info_1% of Event %Info_2% from %Info_3% to %Info_4% (ON-ANGLE)
2	Modification at Section %Info_1% of Event %Info_2% from %Info_3% to %Info_4% (OFF-ANGLE)
3	Copy from Section %Info_1% from Event %Info_2% to all Sections(ON-ANGLE)
4	Copy from Section %Info_1% from Event %Info_2% to all Sections(OFF-ANGLE)
5	Copy from Section %Info_1% from Event %Info_2% to all Sections(ON/OFF-ANGLE)
6	Copy from Section %Info_1% to all Sections(ALL EVENT-TIMING)
7	Modification at Section %Info_1% of Event %Info_2% from %Info_3% to %Info_4%(HHT-->ON-ANGLE)
8	Modification at Section %Info_1% of Event %Info_2% from %Info_3% to %Info_4%(HHT-->OFF-ANGLE)
9	Modification at Section %Info_1% of Event %Info_2% from %Info_3% to %Info_4%(HHT)
11	Modification at Section %Info_1% of %Info_2% from %Info_3% to %Info_4%
21	Modification at Section %Info_1% of Pusher-Differential from %Info_3% to %Info_4%

F Configuration Options

The following items can be configured in D:\FlexisDataInterface\cfg\DataInterfaceConfig.xml on the Lineserver computer (default values shown below):

```
<config>
<param name="serverSocketTimeout">15000</param>
<param name="serverPortNumber">8080</param>
<param name="defaultResponseFormat">compact</param>
<param name="defaultOutputStream">stdout</param>
<param name="fileLogging">disabled</param>
<param name="rrdSamplesAsRate">false</param>
<param name="logFile">d:\FlexisDataInterface\cfg\FDI.log</param>
<param name="sqlDefaultRowLimit”>100</param>
</config>
```

serverSocketTimeout

The maximum time span, in which a request should be transferred completely to the data interface. Allowed range [0..300000] [ms].

serverPortNumber

If a change of the port number is necessary, the access rules of VLAN124 of the Lineserver Cisco switch need to be adapted. Please consult the FlexIS team.

defaultResponseFormat

Currently not in use.

defaultOutputStream

If FlexisDataInterface server is started from the shell and defaultOutputStream is set to stdout, log messages will be printed to stdout of the shell (DOS command window).

fileLogging

If enabled, log messages are written into the log file specified under the 'logFile' parameter (see below). Enable only for troubleshooting and disable after trouble-shooting session to prevent filling of the harddisk.

rrdSamplesAsRate

false: total number for the requested granularity is returned (applies to round robin data only).
true: rate in [1/s] for the returned interval.
Refer to chapter 2.2 for an example.

logFile

Path for log file output (see above for enabling logging).

sqlDefaultRowLimit

Maximum number of lines returned for sql data if no filter criteria is defined in the request.

F.1 Examples for return values of production data

Please refer to the picture below.

The upper graph shows the original data in the database. Newer data is available in one minute samples, older data gets compressed to 5 minute samples or less resolution depending on the age. In the database only rates in [1/s] are stored. This corresponds to the answers with rrdSamplesAsRate = true.

The middle graph shows how an answer could look like if data is requested with a granularity of 60 [s] and rrdSamplesAsRate = false:
Newer data will be given back in one-minute samples, showing the counts collected during this minute.
Older – compressed – data will be given back as average number of counts per minute for the appropriate five-minutes interval.

The lower graph shows how an answer could look like if data is requested with a granularity of 300 [s] and rrdSamplesAsRate = false:
Newer data will be given back in five-minute samples, showing the counts col-

lected during this five minutes.
Older – compressed – data will be given back as well in five-minute samples, showing as well the counts collected during the appropriate five minutes.