Online system overview

BackgroundSystem

Frontend

CommS

Center Online

Data Layer

UnitManager

AlarmManager

ActivityManager

Services

ConrollersPersistence

Controllers

ActivityInstanceLifeCycle

ActivityInstanceLifeCycle

ActivityInstanceLifeCycle

ActivityController

ActivityController

ActivityController

# Itcs components

## Overview

**Online area**

**Iqubes area**

**Communication area**

**ITCS data providers’ area**

Unit Manager

Unit Controller

Unit Controller

ITCS Manager

ITCS server

Comm.S

ITCS Client

**Gorba**

**3rd tier**

## Flow between Itcs server and Itcs client

ITCS Server

ITCS Client

The ITCS server provides scheduled and real-time timetable data to the ITCS client according to the subscription that the client did.

The subscription contains at least the provider name but it could also contain the area display name, the line reference identifier and the direction identifier.

If the server handles all these filters, then data returned back will be filtered in order to reduce the flow of data from ITCS server to ITCS client.

The ItcsFilters are specified by the customer and define the explicit mapping between data sent by the Itcs Server and the StopPoints. A “whitelist” logic is used meaning that the BackgroundSystem is interested only in data specified in those filters.  
The ItcsClient uses these filters to produce the necessary subscriptions for the server to minimize the traffic without losing any required information. Management and optimization of the actual subscriptions to the server are left to the ItcsClient implementation (this is an optional behavior for the ItcsClient[[1]](#footnote-1)).  
Once subscribed to the server, the ItcsClient completely forget these filters.

At the moment only the protocol “VDV 453” is implemented.

Below you find an example xml string for a subscription from ItcsClient for display area “12345” with optional filters line “10” and direction “Zoo”:

<AboAnfrage Sender="**ItcsClient**" Zst="2001-08-08T05:00:00">

<AboAZBRef AboID="25" VerfallZst="2001-08-09T00:00:00">

<AZBID>**12345**</AZBID>

<LinienID>**10**</LinienID>

<RichtungsID>**Zoo**</RichtungsID>

<FruehesteAbfahrtszeit>

**2001-08-08T05:00:00**

</FruehesteAbfahrtszeit>

<SpaetesteAbfahrtszeit>

**2001-08-08T23:00:00**

</SpaetesteAbfahrtszeit>

</AboAZBRef>

**</AboAnfrage>**

When the ITCS client receives data from ITCS server, it filters this data according to the hysteresis value, that means it detects if the data has to be abandoned. In case of the data must be forwarded to the ITCS Manager, it converts them into timetable entries understandable by the ITCS Manager. This step makes the exchanged data format totally independent from the point of view of the online system.

## Flow between Itcs client and Itcs Manager

ITCS Manager

Itcs Manager Service

ITCS Client

Filter: Display area, Line, direction

The ITCS Manager handles several things:

It’s the only one component that is able to access to the database (read and write). Like this we have only one access point to the database.

It does the filtering according to the filters stored in database: when the ITCS Manager receives timetable entries from ITCS Client, it has to filter the timetable entries according to the filters stored into database. This operation could be the same done by the ITCS server before sending data but this is the final gate to ensure that each unit receives only the necessary data. This filtering is done by display area, line if exists and direction if exists.

Once this filter is applied, the ITCS Manager has to know to which units the timetable entries must be sent. It can do it with the stop points that are linked to the display area. Each stop point has one or more units.



NOTE: this diagram should be discussed and verified.

## Flow between ITCS Manager and Unit Manager

\*

ITCS Manager

Itcs Manager Service

Filter: Display area, Line, direction

Unit manager

Unit Controller

Unit Controller

From ITCS Manager to Unit manager, only DTO objects are sent containing all information we need to create the comm.s message to send to units.

The data contained in the DTO objects comes from the timetable entries supplied by the ITCS client(s).

## ITCS text mapping and splitting

ITCS Manager

Itcs Manager Service

ITCS Client

Text mapping

**Text splitting**

When the Itcs manager receives texts from ITCS server via ITCS client, there is a mapping text operation before sending the text to the unit manager. The text could be longer or smaller than the original text regarding the product type. Depending on the product type properties, it could be that a unit doesn’t support a specific text type (e.g. no platform texts). In that case, these texts are neither mapped nor sent to the unit.

Finally, destination texts are split using the comma “,” as separator character. Splitting is executed only if the separator character (“,”) is not surrounded by any space. Examples:

|  |  |
| --- | --- |
| Original text | Resulting (split) texts |
| “Destination1,Destination2” | “Destination1”, “Destination2” |
| “Destination1, Destination2” | “Destination1, Destination2” |
| “Destination1 ,Destination2” | “Destination1 ,Destination2” |

Internally, the text mapping is seen as “atomic” operations (mapping + splitting) according to the type of the text (Platform, Destination, Line) exposed by the ItcsTextMappingService.

Splitting can be disabled in the configuration xml file for the BackgroundSystem.  
The text mapping (+ splitting) should be the last operation before sending text to the unit manager. To make this operation faster, a cache will be used containing all mappings stored into database.

Examples of the entire mapping + splitting operation (bold in “Texts after mapping” or “Texts after splitting” marks texts received by units):

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Source text | Text type | Unit supports type | Texts after mapping | Texts after splitting |
| “Destination1” | Destination | Yes | “Destination1” | “**Destination1**” |
| “Destination2” | Destination | Yes | “DestinationA” | “**DestinationA**” |
| “Destination3” | Destination | Yes | “DestinationA,DestinationB” | “**DestinationA**”, “**DestinationB**” |
| “Destination4” | Destination | Yes | “DestinationA, Destination B” | “**DestinationA, DestinationB**” |
| “Destination1,Destination2” | Destination | Yes | “Destination1,Destination2” | “**Destination1**”, “**Destination2**” |
| “Platform1” | Platform | Yes | “**Platform1**” | - |
| “Platform2” | Platform | Yes | “**PlatformA**” | - |
| “Platform1,Platform2” | Platform | Yes | “**Platform1,Platform2**” | - |
| “Line1” | Line | Yes | “**Line1**” | - |
| “Line2” | Line | Yes | “**LineA**” | - |
| “Line1,Line2” | Line | Yes | “**Line1,Line2**” | - |
| “Platform3” | Platform | No | - | - |

For administration purpose, we will add a timestamp field called LastUse to enable to remove some unused text since a long time. This information is refreshed in the cache each time the text is used, and once a day, the database is refreshed with all the updated values.

Fields we need for the mapping:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ID | Type | SourceText | ITCSProvider ID | Product type ID | Mapped Text | TTSText | LastUse (timestamp) |
| Mapping id | Type of the text:  Destination  Line  Platform | Text sent by the ITCS provider | Identifier of the provider that sends the text | Nullable  Identifier of the type. If this filed is null, that means it’s the default mappin for the text. | Nullable  The text to be displayed by the unit | Nullable | Indicate when this text was used for the last time. Usefull for administration to remove unused text since a long time. |
| ***1*** | ***Dest*** | ***Banhof*** | ***1*** | ***Null*** | ***Banhof St Gallen*** | ***null*** | ***10/01/2013*** |
| ***2*** | ***Dest*** | ***Banhof*** | ***1*** | ***2*** | ***Bnf SG*** | ***Banhof Saint Gallen*** | ***10/01/2013*** |
| ***3*** | ***Line*** | ***Ref01*** | ***1*** | ***Null*** | ***1*** | ***Line 1*** | ***09/01/2013*** |
| ***4*** | ***Line*** | ***RefOldLine*** | ***1*** | ***Null*** | ***9875*** | ***Line 9*** | ***01/01/2010*** |

In case of the Mapped text is null; the Source text is sent.

If the TTS text is null, an empty string is sent to the unit, the unit use the text for text to speech.

**Remarks**: splitting is valid only for destination texts and supports more than 2 different texts (for instance: “Destination1,Destination2,Destination3” -> “Destination1”, “Destination2”, “Destination3”.  
Units only receive final texts that should be displayed. Intermediate texts are never sent to units if they shouldn’t be displayed.  
If a TTS text exist in database, but the unit doesn’t support TTS, this text is not sent to the unit.

### Adding new text

If a received text doesn’t exist into the cache, the new text is added into database and into the cache. The product type ID, the mapped text and theTTS text are unassigned.

### Export text mapping

The data containing the mapping could be exported as CSV file to enable the administrator to fill the missing fields or to add mapping.

### Import text mapping

In case of data import by the administrator, the cache should be refreshed on the fly, that means the Itcs Manager must be informed that new data are available for text mapping.

## PerturbationClient with suspension/resumption of operations

The perturbation client is a component introduced in the release 1.4. It is responsible to connect to a (configurable) GTFS server providing GTFS feeds[[2]](#footnote-2). The perturbation client converts the entries into operations and forwards them to the BackgroundSystem; finally, the BS forwards operations to units.  
The client monitors the entries: as soon as an entry is not anymore in the feed, the clients revokes first and deletes later the corresponding operations.

For an overview of the specifications for the (VBSG) GTFS handling, please look at the documents “Einschränkungen der GTFS-Schnittstelle zwischen icenter.online und Störungsmanager” (H:\35\_DEVELOPMENT\D-Projects\DPRO2-116-GTFS-Schnittstelle\Project Management\Functional Requirements\Einschräkungen GTFS v02.pdf) and “Spezifikation GTFS Schnittstelle Störungsmanager GLUE – Hintergrundsystem DFI Anzeiger Gorba” (H:\35\_DEVELOPMENT\D-Projects\DPRO2-116-GTFS-Schnittstelle\Project Management\Functional Requirements\Spezifikation\_GTFS\_Schnittselle\_Glue\_Gorba109.pdf) that are attached to this document in Appendix A – GTFS documents.

Operations created with the perturbation client can be identified by a prefix (that can be defined in the BackgroundSystem configuration file) followed by the identifier of the entry generating it. Typical example: **SMA: 1453255**.

It is possible that operations created with the perturbation client (but, more in general, with different applications) are in conflict created with Online (for instance, two InfoTexts on the same unit, same row). In such case, the operation(s) created by the perturbation client has higher priority over the one(s) created by Online and it is triggers the suspension/resumption mechanism.

Conflicts are evaluated for each unit, inspecting the operations that should be currently active. If the system finds two conflicting activities defined at the same time on a unit, then the one of the two operations containing the activities is suspended[[3]](#footnote-3).  
If multiple activities are defined in operations, it is enough to find one conflict to suspend/resume entire operations. For instance, let’s consider the following operations defined for a unit at the same time:

* Operation1 (Online)
  + InfoText
  + DeleteTrips
* Operation2 (PerturbationClient)
  + InfoText
  + DisplayOnOff

If the two InfoText activities are in conflict, then the Operation1 is suspended completely, and it will be resumed once the Operation2 has been revoked.  
Priorities are fixed for applications (PerturbationClient has always higher priority over Online), but they are anyway handled as a numeric value greater than or equals to zero in operations (0 -> highest priority, MaxValue -> lowest priority) making it flexible for future usages.

Remark: suspension/resumption are evaluated on a unit base, meaning that the same operation can be active on one unit but suspended on a different one; furthermore, suspension and resumption are “atomic” operations on a unit base, meaning that activities coming from an operation (with higher priority) can’t be started until ALL activities coming from a conflicting operation (with lower priority) are suspended.

The perturbation client is a console application that connects to the BackgroundSystem services using the binding defined in the configuration.

## Alerts

Starting from version 1.4 the BackgroundSystem exposes a new service called « AlertService » that can be used (either internally by the BS itself or externally by other applications) to send alerts.  
Alerts are email notifications sent to a list of recipients including information about a specific event that needs attention by the recipients.

Currently the BS supports three kinds of alerts:

* UnitDisconnected: sent when a unit disconnects from the system. A threshold timeout value can be defined to procrastinate the alert in case the unit reconnects soon. A unit disconnection starts the timer for the defined value (BackgroundSystem configuration file) and sends the alert when it elapses; if the unit reconnects, the timer is canceled until the next disconnection
* UnitAlarm: sent when a major alarm (severity AboveNormal or Critical) is generated
* ItcsServerUnavailable: sent whenever it is not possible for the ItcsClient to send a request to the relative Itcs server or receive a reply from it

For each kind of alert it is possible to use an HTML template for the emails to be generated. Templates are defined using the Razor syntax.

1. Important: the current version of the ItcsClient (VDV453) doesn’t optimize subscriptions in any way. There will be one subscription for each display area. [↑](#footnote-ref-1)
2. Feeds have rules/conventions specific to the customer VBSG. [↑](#footnote-ref-2)
3. The concept is valid for any number of operations. There’s always only one active operation while the others are suspended. [↑](#footnote-ref-3)