



Integration Administration Guides

TFS — Last update: 2020/02/06

Basis Technologies

Table of Contents

- 1. Introduction 1**
 - 1.1. Document Audience 2
 - 1.2. Integration with ActiveControl 3
 - 1.3. Integration Scenarios 4
- 2. Integration Architecture 5**
 - 2.1. The Domain Controller 6
 - 2.2. The Integration Framework Architecture 7
 - 2.3. Integration Process Flow 8
 - 2.4. Inbound Integration Scenarios..... 12
 - 2.5. Inbound Process Flow 13
 - 2.6. Inbound Integration Process..... 15
 - 2.7. Outbound Integration Process 16
 - 2.8. Connector Functionality..... 17
- 3. Technical Prerequisites 18**
- 4. ActiveControl Domain Controller Setup and Configuration..... 19**
 - 4.1. Check SICF for active Services 20
 - 4.2. Class Builder 22
 - 4.3. Inbound Integration 24
 - 4.3.1. Team Foundation Server Application 25
 - 4.3.2. TFS Alert Variable..... 27
 - 4.3.3. TFS URL 28
 - 4.3.4. TE_TFS_COLLECTION 29
 - 4.3.5. Field Map..... 30
 - 4.4. Outbound Integration..... 34

1. Introduction

This Integration Guide is intended to give the reader an overview of the capabilities of the TE Integration Engine and the out of the box (OOTB) integration scenarios supported. It also contains detailed technical specifications of the currently available communication techniques and a detailed configuration guide.

1.1. Document Audience

The intended audience for this document are the technical teams looking to implement integration between ActiveControl and third party tools, such as ticketing systems. It does not detail how TE can be configured to manage the change process and it assumes a reasonable knowledge of standard change processes with SAP.

1.2. Integration with ActiveControl

ActiveControl offers a variety of ways to integrate inbound and outbound scenarios using documented API's. ActiveControl provides an Integration Framework that can manage outbound interactions with external systems (including queuing, re-sends, error processing and reporting) and inbound integration scenarios – those initiated by a system external to TE – by exposing several fully documented API's and web-services that allow manipulation of TE objects by these systems.

In addition, as TE is a Netweaver certified product, all standard SAP integration techniques are available, including tRFC and IDoc communication. But for the purposes of this document, it is assumed that web services will be the preferred integration method, and these are therefore described in detail in this document.

Important: Terminology of Inbound and Outbound in this document. Inbound refers to triggers and data from a 3rd party tool into SAP/AC. Outbound refers to triggers and data from SAP/AC into a 3rd party tool.

1.3. Integration Scenarios

The standard integration scenario is to combine TE and a third party IT Service Management product, and possibly a Test Automation product to create a fully integrated end-to-end process for managing change. This typically requires both inbound and outbound integration:

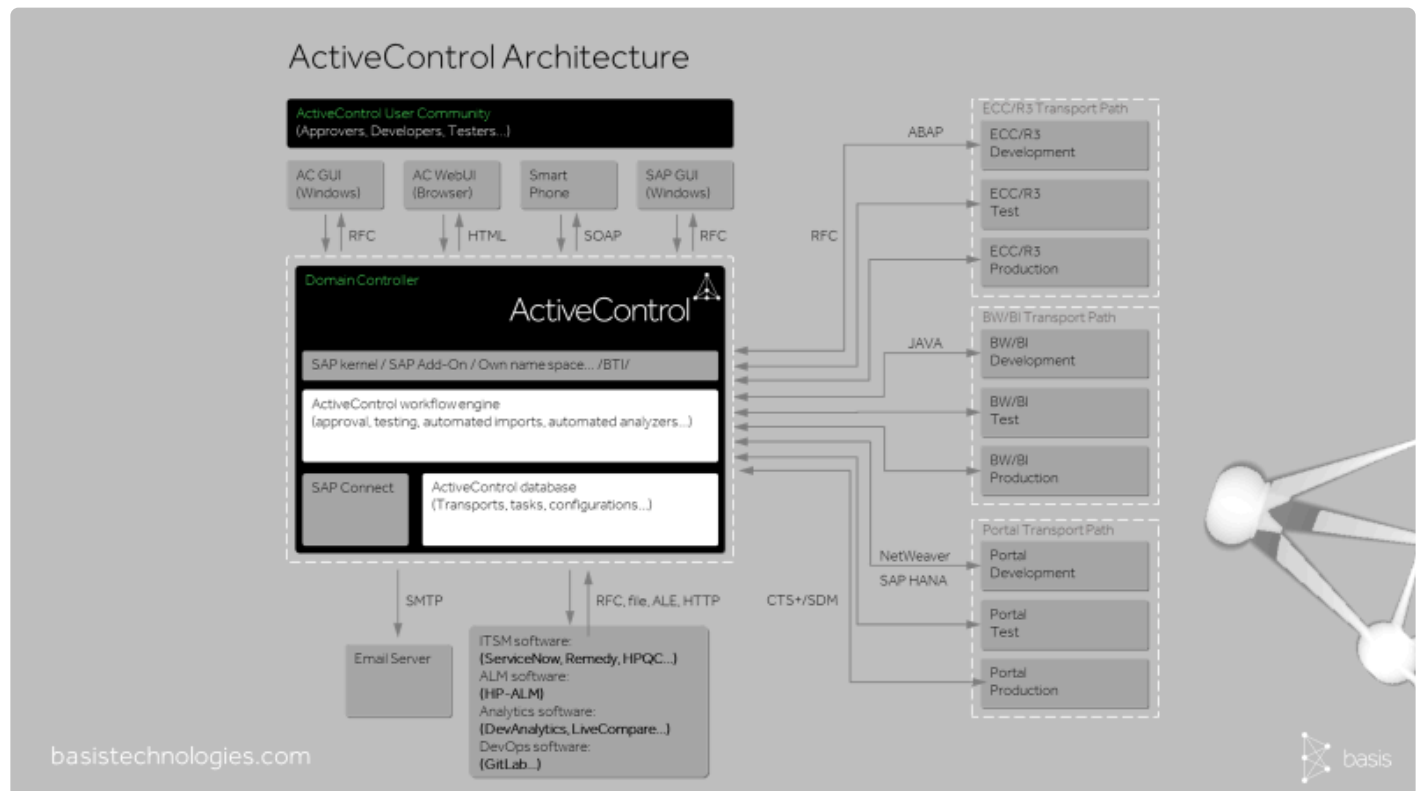


1. Change created in third party ITSM system
2. Change approved for development in ITSM system
3. Change interfaced to TE (inbound integration)
4. Change managed through TE for deployment to Test and Pre-Prod with updates sent to ITSM system to reflect progress (outbound integration)
5. Change deployed to production through TE and ITSM system updated (outbound integration)
6. Change verified and closed in ITSM system

This document will describe, in detail, exactly how such integration can be easily accomplished using the ActiveControl Integration framework, but it should be noted that the framework can be extended for use in many other integration scenarios.

2. Integration Architecture

The architecture of ActiveControl can be broken down into: client software, a controlling SAP system, other participating SAP systems and integration systems. The diagram below illustrates the central role of the controlling SAP system – referred to as the ActiveControl “domain controller”.



2.1. The Domain Controller

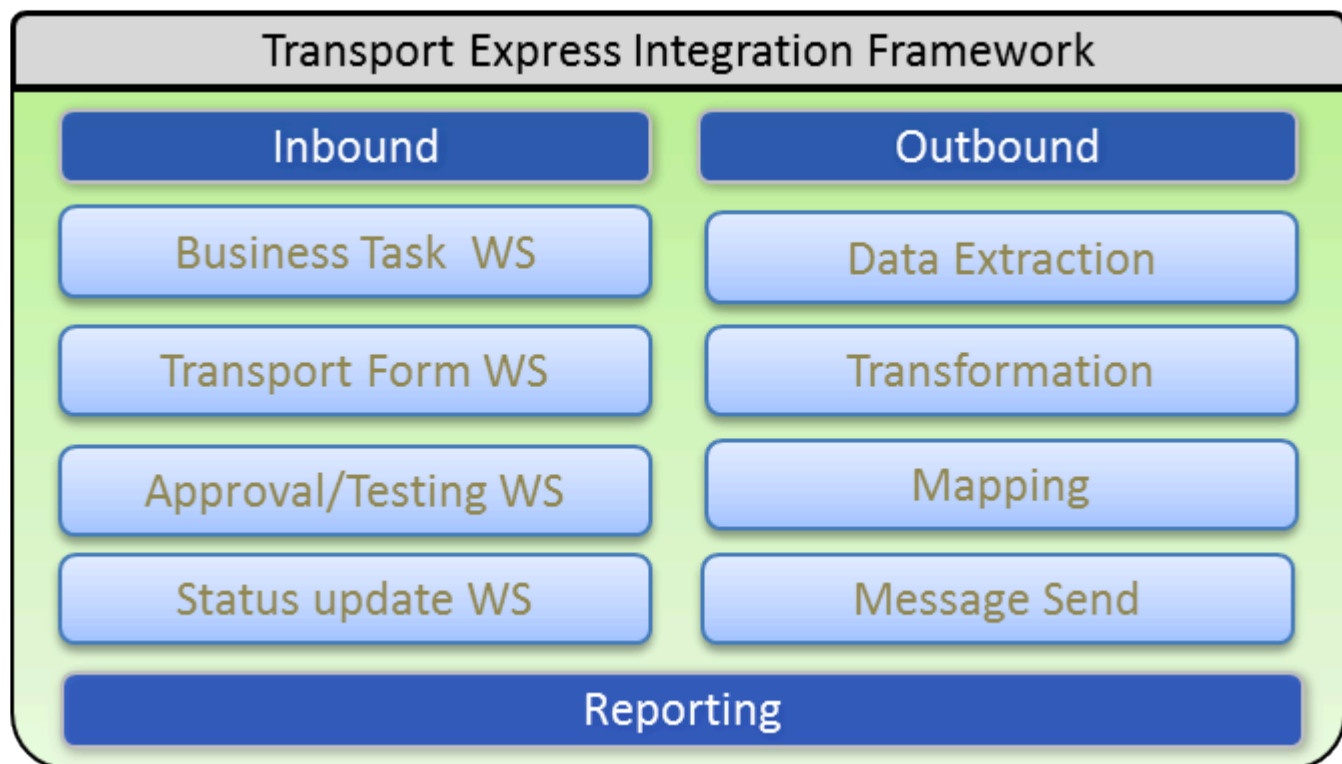
Like the Transport Management System, ActiveControl has the concept of a “domain controller”. The domain controller does not need to be configured in any special way, it is simply the SAP system that the ActiveControl client software connects to, and is where ActiveControl configuration and application data is stored.

The server software runs mostly within the ActiveControl domain controller. When necessary, the domain controller connects to the other SAP systems to gather change request information and to perform transports. These connections are made using SAP's remote function call (RFC) protocol.

The Integration Engine is part of the Domain Controller and manages communication with external products and systems.

2.2. The Integration Framework Architecture

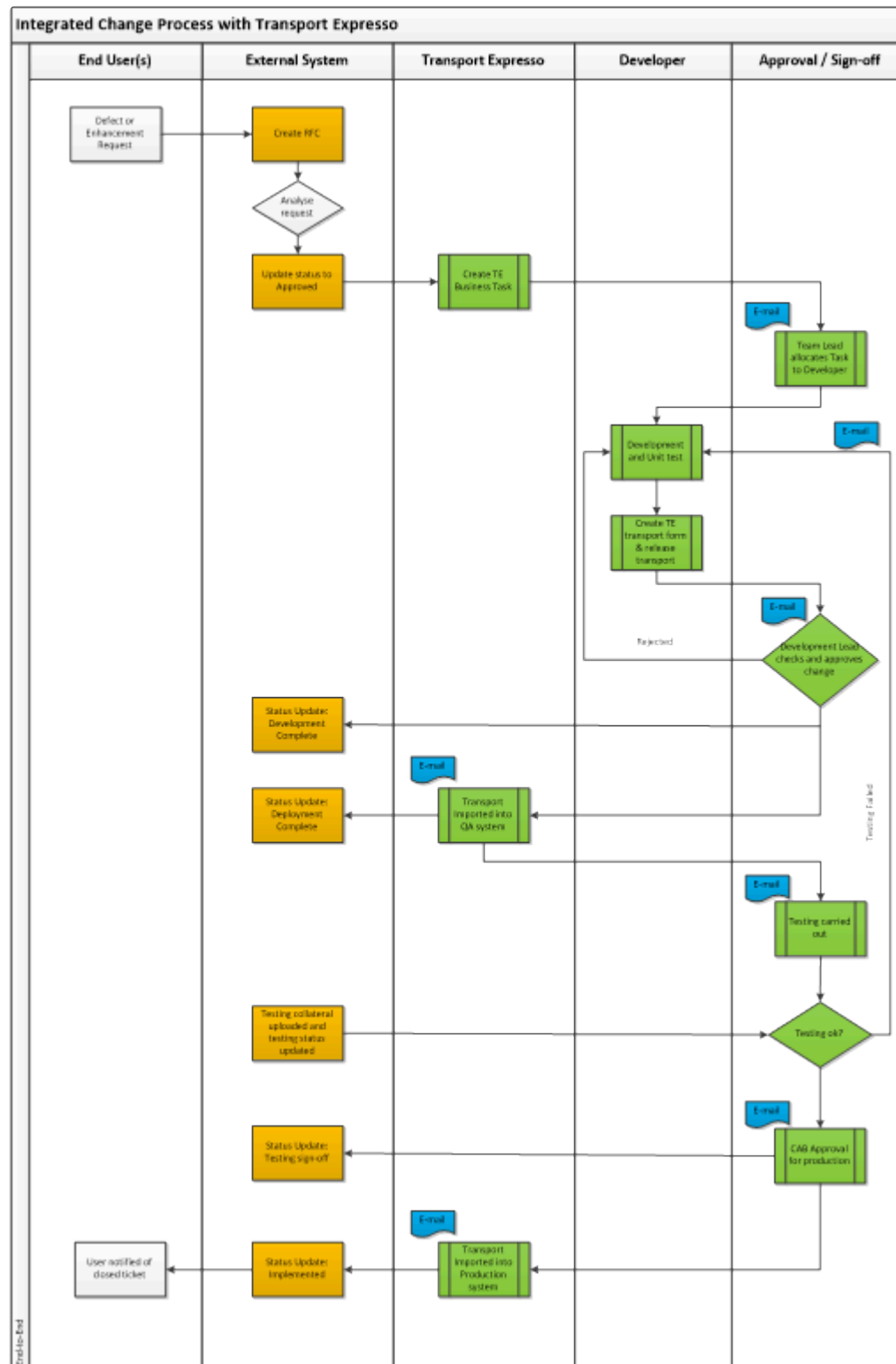
The Integration Framework is divided between inbound and outbound processes. For inbound calls, those made by a third party system into TE, a number of web services are exposed allowing the external system to manipulate ActiveControl objects. Calls to TE web services will return appropriate error messages, but expect the calling system to deal with queuing, service levels and retries for failed integration transactions. For outbound calls there is a configurable framework that includes data extraction, transformation, mapping and sending routines, alongside error detection, correction and reporting, as can be seen in the table below.



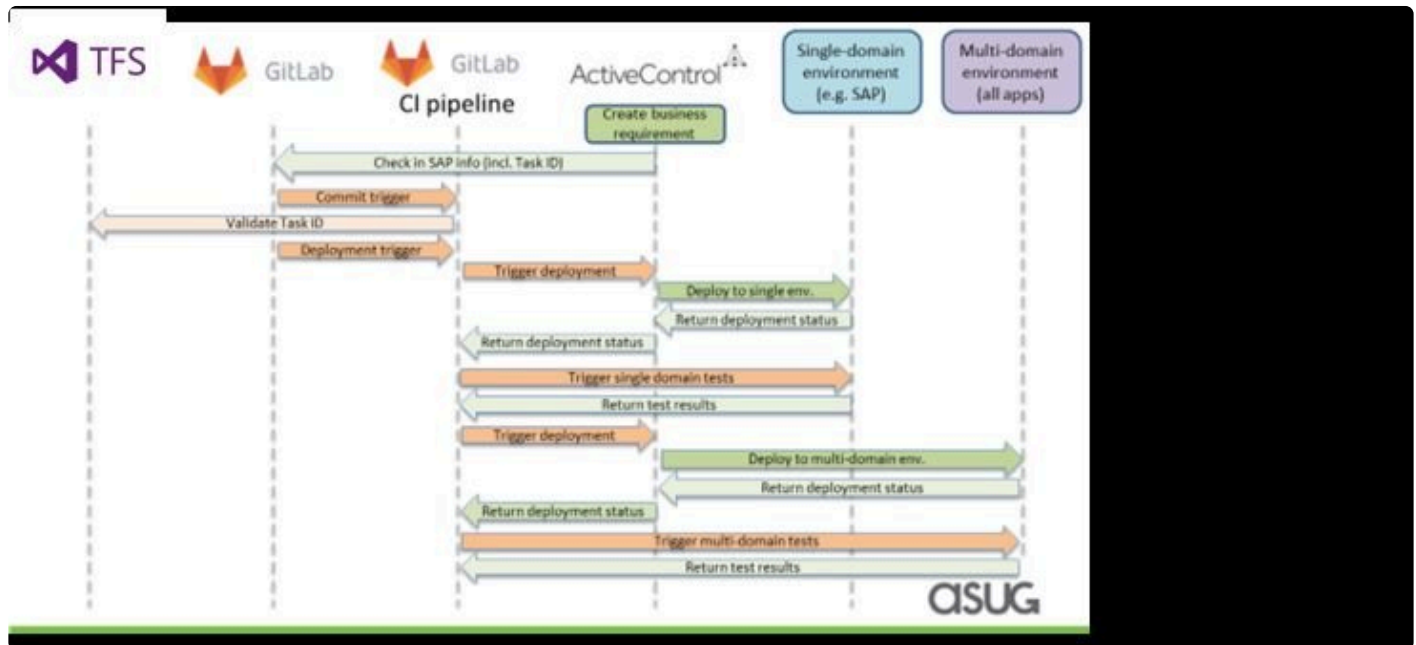
ActiveControl provides Data Extraction and Message Send components for some standard scenarios and third party tools, but these can be enhanced by the addition of custom extraction and send routines plugged into the standard framework. So if, for example, you use an in-house ITSM solution, a new send component can be developed and plugged into the integration framework to facilitate communication between it and ActiveControl. All other standard framework functions, such as data extraction, mapping and error correction remain unchanged and can be used as-is.

2.3. Integration Process Flow

The ActiveControl Integration Framework provides an open architecture for passing messages into and out of the system in a multitude of ways. Although integration can be set up in many ways, one of the more common scenarios is explained in detail below:



DevOps CI/CD inclusion:



In this scenario we have bi-directional integration between an external ticketing system and ActiveControl. This gives a direct link between the ticketing system and the underlying technical changes that make up the business change. So, whether looked at from the perspective of the ticketing system, or through TE, there is only one version of the 'truth' for all changes across the landscape.

From a more detailed perspective, we can look at the integration scenarios:

1. Once a proposed enhancement or defect resolution is approved and a system change is deemed necessary, the external system creates a Business Task in TE representing the change. The ticket in the ITSM system and the TE Business Task are then tied together for the remainder of the process
2. The creation of the Business task in TE marks the start of the development process. The Task can be allocated to a developer who then performs the development and/configuration, and completes unit testing.
3. Once the developer has finished their work, they release the technical change (the transport) and the development team lead is notified by ActiveControl and approves the change. TE will automatically run a number of configured analysis checks at this point to ensure the change is ok to move on in the process.
4. The change is imported into the Quality Assurance system (maybe after another approval from the Testing manager) and is now ready for testing.
5. TE updates the status of the ticket in the ITSM system to show that it is now in testing or ready to be tested.
6. Test collateral and results can be added to either the ticketing system or TE and the ITSM system automatically updated.
7. CAB approval is sought and TE analysis is completed in real time to report dependencies between changes and the impact of different approval scenarios.
8. Once approved by CAB the status of the change in the ticketing system is updated and the change is imported into the Production system at the appropriate time

9. The ticketing system is updated to show the change has been implemented.

2.4. Inbound Integration Scenarios

For inbound integration scenarios TE provides several SOAP web services. Currently, these are:

- Create a Business Task
- Change a Business Task
- Read a Business Task
- Analyse a Business Task
- Read the results of an analysis for a Business Task
- Approve a Business Task
- Enter Test Results for a Business Task

Each web service is detailed in the following sections.

2.5. Inbound Process Flow

The standard inbound integration process flows would be:

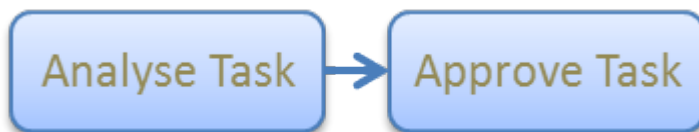
- Create/Change a Business Task in ActiveControl

Creating or changing a Business Task requires simple calls to the appropriate web service. When changing a Task, the current field values should be read first to ensure changed data is not overwritten. The process flow should therefore be:



- Approve a Business Task

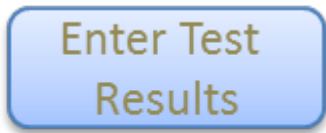
When approving a Business Task it is important that the Task Analysis is completed to first to ensure that the approval can take place safely. The approval web service will not stop the approval if analysis results are ignored. The process flow for approving a Business Task in TE should therefore be:



The analysis results for a Task can be retrieved for any specific Target/Location by calling the Analysis Read web service.



- Enter test results for a Business Task. When entering test results for a Business Task, it must be decided if this result is simply being saved or saved and approved. Only by using the save and approve will the change move to the following control point in the Path.



2.6. Inbound Integration Process

There are two inbound calls in the above scenario:

1. Creation of the Business Task in TE
2. Approval of Testing/Entry of test results once testing complete

Both of these calls would be web service calls to standard TE APIs (although alternative techniques are available and are described later in this document). The calling system (i.e. the ticketing system) would be responsible for queuing of messages and ensuring errors were dealt with appropriately. Some mapping may be required depending on the data passed from the ticketing system to TE for classification of the change.

2.7. Outbound Integration Process

The outbound calls from TE to the external ticketing system can all be based on the Deployment Status of a change within TE. Integration scenarios based on TE status changes are delivered as standard with the TE Integration Engine and therefore require no development.

The steps to set up this type of status based integration are:

1. Complete base TE Integration engine configuration. This includes identifying the end points of the integration and any mapping requirements. The mapping engine can be configured for most standard scenarios, but if complex mapping is required, ActiveControl user exits can be implemented to enhance the standard mapping routines. For more details on TE user exits and how they are implemented, please refer to the ActiveControl Administration Guide.
2. A trigger program should be scheduled to pick up the Task status changes that need to be interfaced to the external system(s). This trigger program selects the appropriate TE records, dependent on the configuration set up above, and passes it through the mapping engine. It then stores the mapped integration transactions into a set of standard tables. See Outbound Configuration section below.

Program Name: /BTI/TE_INTEG_TRIGGER

3. A send program is then scheduled to pick up the mapped transactions and send them out to the configured external systems. It retrieves the required records and then uses the configured send methods for each particular integration scenario to actually push the data out to the receiving systems. If a standard send method is not available for a particular external system (maybe the ticketing system is a 'home-grown' application), then custom send methods can be created and utilised in the Integration Framework. See Outbound Configuration section below.

Program Name: /BTI/TE_INTEG_SEND

4. The outcome of the send process is recorded for audit purposes. If successful, any updates configured are made to the TE data objects, alternatively if errors have occurred, the send program will try to re-send (if configured to do so) a certain number of times before marking the transaction in error and sending a notification to the relevant person(s) within the organisation.

5. At any time, the Integration Reporting Console can be used to see the status of all integrations, the status and history of each transaction and can also be used to update the underlying transactional data, if required, to fix errors. See Outbound Configuration section below.

Program Name: /BTI/TE_RINTEG_AUDIT

2.8. Connector Functionality

From the standard process flow above, it is envisaged that the developed connector will provide the services required on the ITSM side to initiate a web service call to Transport Express to create a TE Business Task (which will be the representation of the ITSM ticket within TE).

The following functions need to be available within the connector:

1. Initiating integration: The exact conditions required for the integration to be initiated will vary from client to client. This means that a flexible, configurable way needs to be developed to initiate integration. A set of conditions need to be able to be created, including the value(s) of any field on the incident/change ticket and its status, which when met, initiates the integration to ActiveControl.
2. Default values: When initiating integration to ActiveControl, we need to be able to specify default values for the mandatory fields on the ActiveControl Task. These are:
 - a. Project
 - b. Group
 - c. Type
3. Mapping: The fields on the ITSM ticket need to be able to be mapped to fields in ActiveControl. Any fields on the ITSM ticket, including any customer defined fields, need to be able to be mapped to any field in the ActiveControl Create Task WS, including custom fields.
4. Processing: Once the ITSM ticket meets one of the conditions to fire off integration and the fields have been mapped to the ActiveControl Web Service, the connector should be able to call the TE web service. A system username and password will need to be passed to enable authentication for some clients, other clients may use SSO. Both authentication methods should be available. ActiveControl will either return an error or the internal number of the Task that has been created. The connector needs to be able to update a field on the ITSM ticket with the TE Task number or to store the error message.
5. Error processing: The connector should be able to be configured to try the integration more than once and to store any error messages that are returned. After a configured number of retries, an email (or other notification) needs to be sent to a configurable list of users, informing them of the error and the ITSM ticket involved. An administrator must be able to manually re-send the integration record if the maximum number of retries has been exceeded.

It should be noted that the connector will not be able to cater for all possible scenarios that may be required by a customer. It is really just a starter which may be extended by the client themselves.

3. Technical Prerequisites

- Confirmation of ITSM, ALM, DevOps or Machine Learning tool,
- Confirmation of established connectivity technology method; webservice or API with SAP and the 3rd party tool. Able to provide a WSDL/url.
- Review SSL certificates are the same between SAP and 3rd party tools. Locally signed certificates are not accepted. Verisign is one certificate provider.
- SCOT (this is required for sending out AC email notifications)
- Active and functioning Web Services on Domain Controller SOAMANAGER
- Age of systems (oldest systems to be managed by AC NW 7.0x)

4. ActiveControl Domain Controller Setup and Configuration

This section guides you through the steps that are needed to configure outbound integration within Transport Espresso.

The Integration configuration is maintained through the SAP standard SM30/31 transactions where table entries can be created and updated.

4.1. Check SICF for active Services

The TFS notification will make a call to the AC service called tfsnotification that must be active and the service handler class /BTI/TE_CL_TFS_INBOUND_WS will create/update the business task.

Default_host > BTI > te_web_services

Define Services

Create Host/Service External Aliases System Monitor Active

Filter Details

Virtual Host: Service Path:

ServiceName:

Description:

Lang.: Ref.Service:

Apply Reset Fine-Tune

Virtual Hosts / Services	Documentation	Reference Service
▼ default_host	VIRTUAL DEFAULT HOST	
▼ BTI	Basis technologies	
• tessocntl	Set TE logon cookie for SSO	
• TE_mobile	TE Mobile API	
• te_web_services	Transport Express webservice	
• tfsnotification	TFS notification	
▶ sap	SAP NAMESPACE; SAP IS OBLIGED NOT T...	
▶ sap_java	VM Container Engine for Java Applications	
• SAPconnect	SAPCONNECT (E)SMTP	

Right-click on the service and chose 'Test Service' to read the url.

url: <http://bti1033.bti.local:8000/bti/tfsnotification?sap-client=100>

TFS Setting: To be set by TFS expert

Choose the option SOAP and give the url copied from 'Test Service' and add the condition as required.

DefaultCollection Projects Favorites Work items Pull requests

Security Notifications

Edit subscription

Description
TE<>TFS

Deliver to Address

SOAP http://bti1033.bti.local:8000/bti/tification?sap-client=100

Filter

☒ Any team project ☐ A specific team project TE_TFS_INT

Field	Operator	Value
-------	----------	-------

+ Add new clause

When the service tfsnotification is triggered by the TFS alert, the handler class /BTI/TE_CL_TFS_INBOUND_WS will read the workitems's information that are in the XML formation.

4.2. Class Builder

Confirm Activate Integration Classes

Class Builder: Display Class /BTI/TE_CL_INTEGRATION_TFS

←→

Local TypesImplementation

Class Interface

/BTI/TE_CL_INTEGRATION_TFS

Implemented / Active

Properties

Interfaces

Friends

Attributes

Methods

Events

Types

Aliases

Parameters

Exceptions

Filter

Method	Level	Vis...	M...	Description
GET_DATA	Insta...	Pub...		Get data of object
PROCESS_INTEGRATION_DATA	Insta...	Pub...		Process integration data
GET_INT_LIST_DATA	Insta...	Pub...		Get list of Integration Systems
GET_STATUS_DESCRIPTION	Insta...	Pub...		Get description of integration systems
SET_DATA	Insta...	Pub...		Set status data of object
CONSTRUCTOR	Insta...	Pub...		
HTTP_CALL	Insta...	Pub...		
READ_TFS_COLLECTION	Insta...	Pub...		

/BTI/TE_CL_INTEGR_CREATE

Class Builder: Display Class /BTI/TE_CL_INTEGR_CREATE

←→

Local Definitions/ImplementationsClass documentationText elements

Class/Interface

/BTI/TE_CL_INTEGR_CREATE

Implemented / Active

Proper...

Interfaces

Friends

Attribu...

Methods

Events

Types

Aliases

Properties

Filter

Attribute	Level	Visibility	R...	Typing	Associated Type	Description	Initial Value
/BTI/TE_IF_INTEGRATION_ACTION-OUTCO	Constant	Public	<input type="checkbox"/>	Type	CHAR1	Single-Character Flag	'S'
/BTI/TE_IF_INTEGRATION_ACTION-OUTCO	Constant	Public	<input type="checkbox"/>	Type	CHAR1	Single-Character Flag	'T'
/BTI/TE_IF_INTEGRATION_ACTION-OUTCO	Constant	Public	<input type="checkbox"/>	Type	CHAR1	Single-Character Flag	'E'
/BTI/TE_IF_INTEGRATION_ACTION-OUTCO	Constant	Public	<input type="checkbox"/>	Type	CHAR1	Single-Character Flag	'I'
/BTI/TE_IF_INTEGRATION_ACTION-OUTCO	Constant	Public	<input type="checkbox"/>	Type	CHAR1	Single-Character Flag	'U'
HEADER	Instance Attribute	Private	<input type="checkbox"/>	Type	/BTI/TE_IF_INTEGRATION_POLLER=		
UTILITY	Instance Attribute	Private	<input type="checkbox"/>	Type	Ref To /BTI/TE_CL_INTEG UTILITY	Utility methods for integration	
CONTAINER	Instance Attribute	Private	<input type="checkbox"/>	Type	Ref To /BTI/TE_IOC_CONTAINER		
LOGGER	Instance Attribute	Protected	<input type="checkbox"/>	Type	Ref To /BTI/TE_IF_LOGGER	Collects log messages	

/BTI/TE_CL_INTEGR_UPDATE

Class Builder: Display Class /BTI/TE_CL_INTEGR_UPDATE

Class/Interface: /BTI/TE_CL_INTEGR_UPDATE Implemented / Active

Properties | Interfaces | Friends | Attributes | Methods | Events | Types | Aliases

Attribute	Level	Visibility	R...	Typing	Associated Type	Description	Initial Value
/BTI/TE_IF_INTEGRATION_ACTION-OUTCO	Constant	Public	<input type="checkbox"/>	Type	CHAR1	Single-Character Flag	'S'
/BTI/TE_IF_INTEGRATION_ACTION-OUTCO	Constant	Public	<input type="checkbox"/>	Type	CHAR1	Single-Character Flag	'T'
/BTI/TE_IF_INTEGRATION_ACTION-OUTCO	Constant	Public	<input type="checkbox"/>	Type	CHAR1	Single-Character Flag	'E'
/BTI/TE_IF_INTEGRATION_ACTION-OUTCO	Constant	Public	<input type="checkbox"/>	Type	CHAR1	Single-Character Flag	'I'
/BTI/TE_IF_INTEGRATION_ACTION-OUTCO	Constant	Public	<input type="checkbox"/>	Type	CHAR1	Single-Character Flag	'D'
HEADER	Instance Attribute	Private	<input type="checkbox"/>	Type	/BTI/TE_IF_INTEGRATION_POLLER-		
UTILITY	Instance Attribute	Private	<input type="checkbox"/>	Type	Ref To /BTI/TE_CL_INTEG_UTILITY	Utility methods for integration	
CONTAINER	Instance Attribute	Private	<input type="checkbox"/>	Type	Ref To /BTI/TE_IOC_CONTAINER		
LOGGER	Instance Attribute	Protected	<input type="checkbox"/>	Type	Ref To /BTI/TE_IF_LOGGER	Collects log messages	

/BTI/TE_CL_INTEGR_TESTRES

Class Builder: Display Class /BTI/TE_CL_INTEGR_TESTRES

Class/Interface: /BTI/TE_CL_INTEGR_TESTRES Implemented / Active

Properties | Interfaces | Friends | Attributes | Methods | Events | Types | Aliases

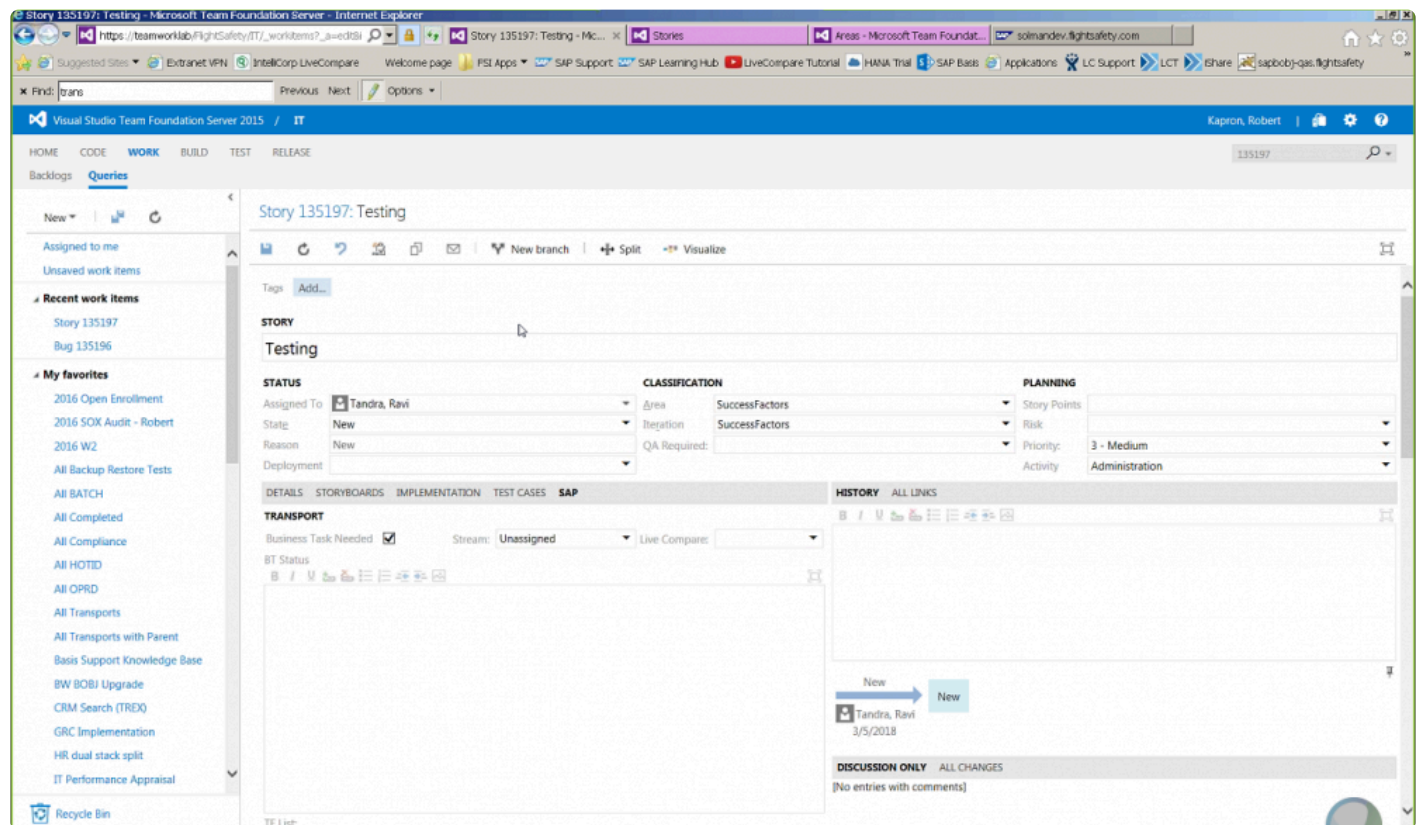
Attribute	Level	Visibility	R...	Typing	Associated Type	Description	Initial Value
/BTI/TE_IF_INTEGRATION_ACTION-OUTCO	Constant	Public	<input type="checkbox"/>	Type	CHAR1	Single-Character Flag	'S'
/BTI/TE_IF_INTEGRATION_ACTION-OUTCO	Constant	Public	<input type="checkbox"/>	Type	CHAR1	Single-Character Flag	'T'
/BTI/TE_IF_INTEGRATION_ACTION-OUTCO	Constant	Public	<input type="checkbox"/>	Type	CHAR1	Single-Character Flag	'E'
/BTI/TE_IF_INTEGRATION_ACTION-OUTCO	Constant	Public	<input type="checkbox"/>	Type	CHAR1	Single-Character Flag	'I'
/BTI/TE_IF_INTEGRATION_ACTION-OUTCO	Constant	Public	<input type="checkbox"/>	Type	CHAR1	Single-Character Flag	'D'
CONFIG	Instance Attribute	Private	<input type="checkbox"/>	Type	Ref To /BTI/TE_CL_CONFIG_HOLDER		
HEADER	Instance Attribute	Private	<input type="checkbox"/>	Type	/BTI/TE_IF_INTEGRATION_POLLER-		
UTILITY	Instance Attribute	Private	<input type="checkbox"/>	Type	Ref To /BTI/TE_CL_INTEG_UTILITY	Utility methods for integration	
CONTAINER	Instance Attribute	Private	<input type="checkbox"/>	Type	Ref To /BTI/TE_IOC_CONTAINER		
LOGGER	Instance Attribute	Protected	<input type="checkbox"/>	Type	Ref To /BTI/TE_IF_LOGGER	Collects log messages	

4.3. Inbound Integration

4.3.1. Team Foundation Server Application

TFS admin needs to create alert filters mentioning BTI soap service.

These filters will temporarily hold lists of newly created tickets for the integration to pick up and create Business Tasks in TE.



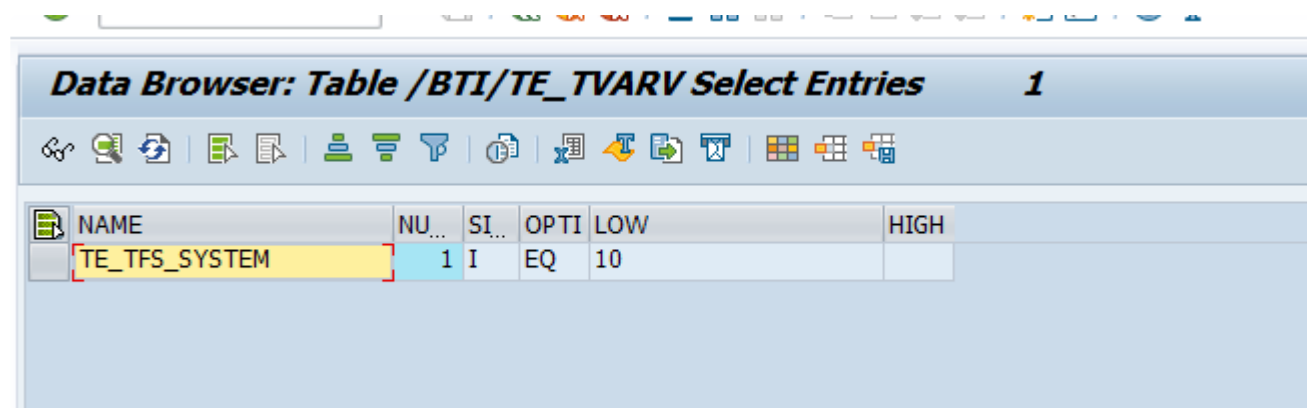
> Speaking: robert

4.3.2. TFS Alert Variable

/BTI/TE_INT_SYST is where you define the integration system. Those configured systems will be populated in SEND/TRIGGER Program. But for TFS, it is the TFS alert that is triggering SAP soap, hence sap system, at runtime, doesn't know which integration system it should pick up, so you need to maintain the integration system in TVARV table with variant variable TE_TFS_SYSTEM

/BTI/TE_TVARE – TE Integration Variables	
Field	Description
NAME	Name of the integration system created in /BTI/TE_INT_SYST
NUMBER	Configure the system number(the number set in table /BTI/TE_INT_SYST) in table /BTI/TE_TVARE against the variant variable 'TE_TFS_SYSTEM'
SIGN / INCL/EXCL	I = Include, E = Exclude data
OPTION	
SELECTION VALUE	

Configure the system number (the number set in table /BTI/TE_INT_SYST) in table /BTI/TE_TVARE against the variant variable 'TE_TFS_SYSTEM'



The screenshot shows the SAP Data Browser interface for the table /BTI/TE_TVARE. The title bar indicates 'Data Browser: Table /BTI/TE_TVARE Select Entries' with 1 entry selected. The table has columns: NAME, NU..., SI..., OPTI, LOW, and HIGH. The selected entry is for the integration system 'TE_TFS_SYSTEM' with a system number of 1, sign 'I' (Include), option 'EQ', and a low value of 10.

NAME	NU...	SI...	OPTI	LOW	HIGH
TE_TFS_SYSTEM	1	I	EQ	10	

4.3.3. TFS URL

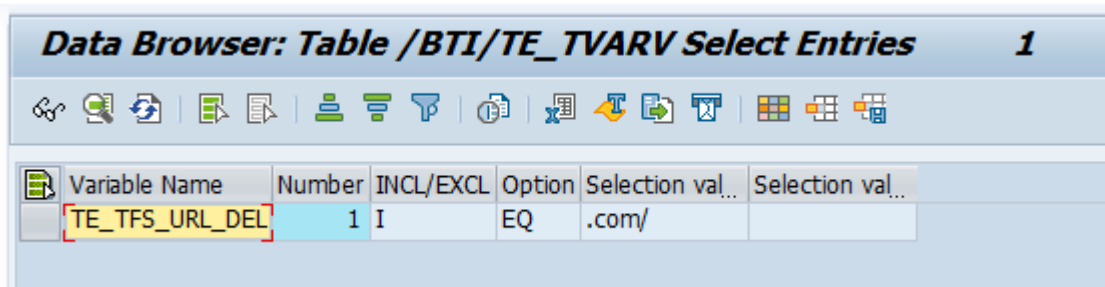
Have to maintain an entry in TVARV table. It is a pattern to read the collection from url. So maintain the word (with end '/') just before the collection in the URL.

Eg: <https://teamworklab.flightsafety.com/FlightSafety/Services/v3.0/LocationService.aspx> the collection name is FlightSafety. Hence the word with '/' just before the collection name has to be maintained.

Another example for better understanding

<http://vsalm:8080/tfs/FabrikamFiberCollection/Services/v3.0/LocationService.aspx>

Here need to maintain /tfs/ as 'FabrikamFiberCollection' is the collection name.

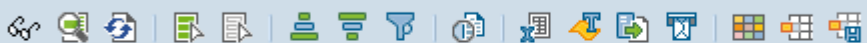



The screenshot shows a 'Data Browser' window titled 'Table /BTI/TE_TVARE Select Entries 1'. It contains a table with the following data:

Variable Name	Number	INCL/EXCL	Option	Selection val...	Selection val...
TE_TFS_URL_DEL	1	I	EQ	.com/	

4.3.4. TE_TFS_COLLECTION

If only one collection is used, then maintain the collection name in TVARV with variable TE_TFS_COLLECTION. If a work item created from a collection other than the one maintained, then the business task reference will have that collection name as prefix to avoid overwriting.

<i>Data Browser: Table /BTI/TE_TVARE Select Entries</i>						<i>1</i>
						
 Variable Name	Number	INCL/EXCL	Option	Selection value	Selection val...	
TE_TFS_COLLECTION	1	I	EQ	FlightSafety		

4.3.5. Field Map

When passing data from TFS XML format to sap structure, the inbound integration class needs to know the equivalent TE field of TFS fields. Whereas with /BTI/TE_INT_MAPP, the webservice CREATE_TASK & CHANGE_TASK wouldn't be working if we use /BTI/TE_INT_MAPP for mapping.

To read the fields and values that are in xml format sent from TFS, the class /BTI/TE_CL_TFS_INBOUND_WS need to know the equivalent TE field of TFS fields. So fill /BTI/TE_INT_FMAP for initial mapping if parsing 3rd party data to sap structure unless it is mapped in the 3rd party side itself.

/BTI/TE_INT_FMAP – Integration Mapping Is to map the TE field to the equivalent fields of TFS system	
Field	Description
EXTSYS_NO	Main external system identifier, this is the identifier of the system that you wish to integrate with we can have as many systems as we want. An example of this could be: 1 – Remedy, 2 – Solution Manager, 3 – ServiceNow
SEQUENCE_NO	Doesn't have to be in sequence but just to have part of key
CLASS	TASK to create business task or REQUEST to create transport form
EXT_FIELDNAME	TFS field name to be mapped to TE fields
BTI_TE_FIELDNAME	TE business task field

Data Browser: Table /BTI/TE_INT_FMAP Select Entries 8					
EXTSYS_NO	SEQUENCE_NO	CLASS	EXT_FIELDNAME	BTI_TE_FIELDNAME	
10	1	TASK	System.Id	REFERENCE	
10	2	TASK	System.AreaId	GROUPID	
10	3	TASK	Microsoft.VSTS.Common.Activity	TYPEID	
10	4	TASK	System.Title	CAPTION	
10	5	TASK	System.TeamProject	PROJECTID	
10	6	TASK	System.AssignedTo	TESTERID	
10	7	TASK	System.Description	TEXT	
10	8	TASK	Microsoft.VSTS.Common.Priority	PRIORITY	

LIMITATION:

1. A mandatory entry to have the TFS collection in a custom field in TE is required with the value 'TFS_COLLECTION' in the field EXT_FIELDNAME and the TE custom field no in the field BTI_TE_FIELDNAME in table /BTI/TE_INT_FMAP.

The possible issue without this setting is explained further.

There is a possibility for 2 work items to have a same ID from different collection. Given that work item ID is saved as a TE reference, task for the work item that has the same ID from different collection can be overwritten. So to avoid this issue, this entry is mandatory (eg: ref screenshot below) and the default collection name must be defined in the table /BTI/TE_TVARV with variant TE_TFS_COLLECTION. If the collection name of story is different from the default maintained in the TVARV table, then the reference of BT will be CollectionName_WorkItemID. If it is same as the default, then it is only the workitem ID will be reference of BT..

New Entries: Overview of Added Entries				
TE Integration: Field mapping between 3rd party and TE				
Ext.Sys.No	Sequence	Class	TE Field Reference	
1	13	TASK	505	

Data Browser: Table /BTI/TE_TVARRV Select Entries 1						
NAME	NU...	SI...	OPTI	LOW	HIGH	
TE_TFS_COLLECTION	1	I	EQ	DefaultCollection		

1. Fields called BT Status and TF list are required to be updated in TFS apart from deployment status which is a direct mapped field. But for the former 2 fields, this requires some reading extra tables to make the value to pass to TFS. Hence exit /BTI/TE_EXIT_SAMPLE_0990 will read the list of transports, description and their location in HTML to have the list in table format in TFS field "TF List". Likewise, reads the status of business task and send to TFS field "BT Status". This requires configuration in table /BTI/TE_INT_FMAP to identify to which TFS fields the values were to be passed. The equivalent TE fields for BT status and TF list in TFS are TE_TFS_BT_STATUS and TE_TFS_TF_LIST

New Entries: Overview of Added Entries				
TE Integration: Field mapping between 3rd party and TE				
Ext.Sys.No	Sequence	Class	External Reference	
1	11	TASK	FlightSafety.BTStatus	
1	12	TASK	FlightSafety.TFList	

For updating the TFS list and status in TFS workitem and querying TFS back after the alert to read all fields required for creating a BT in TE.

Requires patch transport "D00K902343 Updating the TF List and Status in TFS workitem and querying" as of AC v7.03.

2. Create an entry in table /BTI/TE_INT_FMAP. In column "External Reference" fill in the TFS reference field for BT needed

Data Browser: Table /BTI/TE_INT_FMAP Select Entries 1					
External System Numb...	Sequence	Class	External Reference	TE Field Reference	
10	10	TASK	System.BTNeeded	TE_TFS_BT_NEEDED	

Example update in TFS:

Task 10: 09897

Tags [Add...](#)

09897

STATUS		CLASSIFICATION	
Assigned To	Kalidass	Area	TE_TFS_INT
State	Active	Iteration	TE_TFS_INT\Iteration 1
Reason	Work started		

DESCRIPTION IMPLEMENTATION			HISTORY	ALL LINK
ACDK900484	1_TE_TFS	Kali Dev(Test Queue)		
ACDK900486	2_TE_TFS	Kali Dev(Test Queue)		

4.4. Outbound Integration

This section guides you through the steps that are needed to configure outbound integration within ActiveControl.

The Integration configuration is maintained through the SAP standard SM30/31 functions where table entries can be created and updated.

Configuration Overview

The table below shows a list of database tables with descriptions that need to be maintained followed by a more in depth description of how to configure the tables.

Name	Description
/BTI/TE_INT_SYST	Integration System List Table
/BTI/TE_INT_CLAS	Integration Object Class List
/BTI/TE_INT_PC	Process Codes
/BTI/TE_INT_PROC	Process Identifier
/BTI/TE_INT_MAPP	Mapping Table
/BTI/TE_INT_CONV	Conversions
/BTI/TE_INT_USR	Notification Users
/BTI/TE_INT_FLDE	Complex Mapping (user exits)
/BTI/TE_INT_FILT	Filter Values