Test Driven Development - CanCreateProjectTasksForNewQmfIdsWithBudgetValues

Description

This is a task to build functionality into the DirectCostProjectSync application service method.

Note

This is an additional test that adds extra functionality to our existing tests!

In the test details below, I have highlighted in blue any new data to be added or tested.

Implementation Guide

Test Driven Development

We want to approach this as Test Driven Development, so ideally you want to

- create an integration test, which should always **fail** to begin with
  - so run it to begin with, and make sure you get a red failure in your test list!
- then, implement the functionality in the DirectCostProjectSync method in WorksOrderServices
- re-run the test, which should now **pass**

Test Class

All your integration tests should be written in the same class: DirectCostProjectSyncTests.cs

- a single class to hold all DirectCostProjectSync tests, to be located in Vision.Api.DotNet.Tests.Integration.Builder.WorksOrders
- to inherit from WorksOrderTestBase.cs
- Common private methods can be used in the class for repeated common test assertions (e.g. checking for a new Qmf object being created with expected methods)

Code Implementation

All your tests should work by

- creating setup data as required (using the data provided by the base class Setup method wherever possible, and/or using additional ad-hoc data created within your test method if required by a specific test)
- making a single call using the API Builder to the DirectCostProjectSync controller (which in turn will call the DirectCostProjectSync method in WorksOrderServices)

The implementation of the code required to make your tests actually work, should all be written as follows

- The DirectCostProjectSync method in WorksOrderServices makes a CoreQueryOver call to get all works orders matching the supplied filter
- For every works order returned from the CoreQueryOver, the UpdateProjectsForWorksOrder method should be called from SAPManageProjectServices
    - SAPManageProjectServices should be injected into WorksOrderServices using Dependency Injection
- All implementation code should then be written inside the UpdateProjectsForWorksOrder method in SAPManageProjectServices
    - The method should always retrieve the project from SAP by passing in the WorksOrder.ERPPRojectId property as the SAP project ID
    - This Implementation code should use **only** the SAP building block methods you created for any SAP-specific operations

Failing Tests

Failing tests are expected to generate a custom exception specific to the error

- Custom exceptions are located in Vision.Api.DotNet.Common.Exceptions in the DirectCostProjectSync folder
- Exception names should ideally be comprised of the operation being tested (e.g. DirectCostProjectSync) followed by the error (e.g. ProjectDoesNotExistInSap) and finally ‘Exception’, e.g. DirectCostProjectSyncProjectDoesNotExistInSapException.cs
- Custom exceptions need entries to be added into Globalisation.Exceptions for both a DeveloperMessage and a standard user Message

Setup and Tear Down

The Integration Test will set up and tear down Vision-specific data to use in the tests

- Standard works order/ QMF data is provided by the base class WorksOrderTestBase Setup method; this should be used wherever possible
- Additional ad-hoc data can be created within your test methods as and when required; if this is done, the data should be added to the tear-down collections (e.g. _worksOrders) so it is removed by the base class TearDown method

The only pre-requisite data outside of the base class are definitions of project Ids that are known to exist in SAP; these should be declared as variables at the top of DirectCostProjectSyncTests.cs so the project Ids can be updated in the event of a SAP test tenant rebuild

Test name

- CanCreateProjectTasksForNewQmfsWithBudgetValues

Test aim

- To confirm that a direct cost project sync creates new project tasks in SAP, where
    - the Works Orders matching the filter have child Qmf lines
    - any of those child Qmf lines **do not** have matching project tasks under the Works Order's matching project in SAP
    - each Qmf line has a quantity greater than 1
    - the Qmf lines have associated Enquiry Qmf Lines

Prerequisite test data

- Project exists in SAP with known ID
    - *Project Id should be in the format e.g. WO12345 (matching a standard Vision works order number)*
- Works Order exists in Vision, and its associated ERPProjectId matches a known Project ID in SAP
- Works Order has Qmf lines that **do not** exist as project tasks under the corresponding SAP project
- Each identified Qmf line has an associated EnquiryQmfLine record, with a fully populated set of values
- Each identified Qmf line has a random Order Date within the last year
- Works Order has an associated Estimate record, with an associated Manufacturing Region

Filter to use in API Builder call

- WorksOrder/Id equal to Id of works order with Qmf lines that have no associated project tasks in SAP

- As part of this test, before making the call to WorksOrderServices, you will need to generate a QMFHomeCurrency record.
- To do this, for the QMF lines, in your tests use the ApiBuilder to create the Qmf lines using the ApiBuilder.PostRequest.ForQmfs instead of generating them in raw NHibernate.
- (Note you will still need to add them to your tear-down collection so they are removed afterwards; you will also need to add the associated

ProductionSchedules that will be created at the same time to their corresponding tear down collection.)

Expected test criteria to pass

- Success status code returned
- Result object returned in response shows the following counts
  - QMFLineCreatedCount: {number of tasks created in test}
  - QMFLineUpdatedCount: 0
  - QMFLineStoppedCount: 0
- New project task created for each QMF line that did not previously exist as a project task in SAP
  - Project Task ID: {{WO ERP Project ID}}-{{QMF ID}}
  - Project Task Name: {{WO ERP Project ID}}-{{QMF ID}} {{QMF Description}}
    - e.g. for a works order ERP Project ID WO12345, Qmf Id 123, Qmf Description “Hello World”
      - Id: “*WO12345-123*“
      - Name: “*WO12345-123 Hello World*“
  - Project Task status is released
  - Project tasks have **one of each of the following** associated … the calculations for each of these is included below.
    - Expenses - Subcontractor costs
    - Expenses - COS Labour Assembly Basic
    - Revenues - Turnover Trade
  - All expenses and revenues for the project tasks have the same currency as the Responsible Unit’s currency

Calculations for SAP budget values

The SAP budget values are the most complex part of this entire project. The following should guide you through the calculations involved.

There are two sets of calculations below

- one is the calculation you should use in your **implementation** (in the SAPManageProjectServices); this will be a little “neater” and use the helper properties on the domain object

- the other is what you should use in your **tests**; this will require some NHibernate coding and is a little more involved, however this will pay off as it will be a good verification test that confirms our logic is sound

In your Implementation

The SAP expenses and revenues for each QMF line should be calculated **in your application service** as follows.

The currency to use will be the Qmf.WorksOrder.ResponsibleUnitOrganisation.Currency.

SAP Expense: Subcontractor Costs

- Qmf.MaterialCost

SAP Expense: COS Labour Assembly Basic

- Qmf.AssemblyAndFabricationLabourCost + Qmf.ServiceLabourBudgetTotalCombined

SAP Revenue: Turnover Trade

- For the most recent record in the QMFHomeCurrency table for the QMF (by CreatedDateUtc) the value rate to use will be
  - If the Qmf.WorksOrder.ResponsibleUnitOrganisation.Currency = GBP, then **GBPToLocalValue**
  - If the Qmf.WorksOrder.ResponsibleUnitOrganisation.Currency = EUR, then **EuroToLocalValue**
  - If the Qmf.WorksOrder.ResponsibleUnitOrganisation.Currency = USD, then **USDToLocalValue**

In your Tests

The SAP expenses and revenues for each QMF line should be verified **in your tests** as follows.

SAP Expense: Subcontractor Costs

- The **Qmf Quantity**, multiplied by the sum of the following (taken from the associated **QMF Enquiry Line**), then all divided by the **Estimate Home Currency Exchange Rate** (see below)
  - Cost Price Per Unit In House Material
  - Cost Price Steel
  - Cost Price Copper
  - Landed Cost Per Unit

SAP Expense: COS Labour Assembly Basic

- The **Qmf Quantity**, multiplied by the sum of the following (taken from the associated **QMF Enquiry Line**, except for the Labour Rates which are on the Enquiry Qmf Line's associated **Estimate**), then all divided by the **Estimate Home Currency Exchange Rate** (see below)

    - Product Labour Hours Per Unit Mechanical **multiplied by** Labour Rate Mechanical
    - Product Labour Hours Per Unit Wiring **multiplied by** Labour Rate Wiring
    - Product Labour Hours Per Unit Testing **multiplied by** Labour Rate Testing
    - Product Labour Hours Per Unit Splitting **multiplied by** Labour Rate Splitting
    - Product Labour Hours Per Unit Fabrication **multiplied by** Labour Rate Fabrication
    - Service Labour Hours Per Unit Mechanical **multiplied by** Labour Rate Mechanical
    - Service Labour Hours Per Unit Testing **multiplied by** Labour Rate Testing
    - Service Labour Hours Per Unit Splitting **multiplied by** Labour Rate Splitting
    - Service Labour Hours Per Unit Service **multiplied by** Labour Rate Service

SAP Revenue: Turnover Trade

- The following property from the **most recent** QmfHomeCurrency record matching the Qmf line.
    - If the Qmf.WorksOrder.ResponsibleUnitOrganisation.Currency = GBP, then **GBPToLocalValue**
    - If the Qmf.WorksOrder.ResponsibleUnitOrganisation.Currency = EUR, then **EuroToLocalValue**
    - If the Qmf.WorksOrder.ResponsibleUnitOrganisation.Currency = USD, then **USDToLocalValue**

The **Estimate Home Currency Exchange Rate** (that you will need for the above is defined as:

For the most recent record in the CurrencyExchangeRate table

- with more than one associated CurrencyExchangeRateReviewer record
- where the currency is the same as the Qmf.WorksOrder.Estimate.ManufacturingRegion.Currency
- where the ActiveFromDateUtc is less than or equal to the Utc version of the Qmf.OrderDate

the exchange rate to use will be

- If the Qmf.WorksOrder.ResponsibleUnitOrganisation.Currency = GBP, then **GBPToLocalRate**
- If the Qmf.WorksOrder.ResponsibleUnitOrganisation.Currency = EUR, then **EuroToLocalRate**
- If the Qmf.WorksOrder.ResponsibleUnitOrganisation.Currency = USD, then **USDToLocalRate**