

**Interface Design Specification for Service Optimization Implementation at**



**Date: 20 September 2013**



Copyright Notice

Copyright© 2009 ClickSoftware, Inc. All rights reserved. This document must not be reproduced (in any form or in any medium) in whole or in substantial part, except under an agreement with, or with the prior written consent of, ClickSoftware, Inc.

This document, and the information contained therein, is also issued in confidence, for use and disclosure solely in relation to the purpose or purposes for which it was supplied. No information as to the content or subject matter of this document or any part thereof arising directly or indirectly there from shall be used or disclosed, orally or in writing, or communicated in any manner whatsoever, including but not limited to tendering or manufacturing purposes, to any third party or any employee thereof except under an agreement with, or with the prior written consent of, ClickSoftware, Inc.

Trademark

ClickSchedule and ClickAnalyze are registered trademarks of ClickSoftware, Inc.

World Headquarters: ClickSoftware, Incorporated

35 Corporate Drive, Suite 140

Burlington, MA 01803

1-781-272-5903

Local Office: The Priory, Stomp Road, Burnham, Bucks, SL1 7LW, UK

Tel: +44 (0) 1628 607000

**ClickSoftware Internet Web Site:** <http://www.clicksoftware.com>

**Author**

Duncan Hardie, Senior Technical Consultant

**ClickSoftware Project Team**

Project Manager: John Bardell

Solution Architect: Shai Nahari

PS QA Lead: Anna Broitman

Senior Technical Consultant: Anthony Brian

Senior Technical Consultant: Dima Gorbonos

Senior Technical Consultant: Phil Weighill-Smith

Senior Technical Consultant: Duncan Hardie

**Table of Contents**

[1 Introduction 6](#_Toc370915627)

[1.1 About this Document 6](#_Toc370915628)

[1.2 Disclaimer 6](#_Toc370915629)

[1.3 Change History 7](#_Toc370915630)

[1.4 Terms and Abbreviations 7](#_Toc370915631)

[1.5 In Scope 9](#_Toc370915632)

[1.6 Out of Scope 9](#_Toc370915633)

[2 Integration Overview 11](#_Toc370915634)

[2.1 Landscape Overview 11](#_Toc370915635)

[2.2 Service Optimization Web Services 12](#_Toc370915636)

[2.2.1 Available Services 12](#_Toc370915637)

[2.2.2 Namespaces 12](#_Toc370915638)

[2.2.3 Target URL 13](#_Toc370915639)

[2.2.4 Utilizing SOAP 1.1 13](#_Toc370915640)

[2.2.5 Data Formats 13](#_Toc370915641)

[2.2.6 Cloud Offline Development 14](#_Toc370915642)

[2.3 Task Unique identifiers 15](#_Toc370915643)

[2.4 On-Line Integration Activities 18](#_Toc370915644)

[2.4.1 GetAddressGeocode - Address Validation 18](#_Toc370915645)

[2.4.2 ExtendedGetAppointmentsEx2 – Visit Appointment Bookings 18](#_Toc370915646)

[2.4.3 ProcessTaskEx – Tasks Create or Update 18](#_Toc370915647)

[2.4.4 GetTasks – Retrieving Latest Visit Information 19](#_Toc370915648)

[2.4.5 GetResources – Retrieving Engineers 19](#_Toc370915649)

[2.5 Additional Objects via Periodic Import 19](#_Toc370915650)

[2.6 Intergration Scenarios 20](#_Toc370915651)

[2.6.1 Appointment Booking 20](#_Toc370915652)

[2.6.2 Task Creation/Modification 22](#_Toc370915653)

[2.6.3 BSS Business Scenario elaboration 36](#_Toc370915654)

[2.7 Excel imports with Click Data Import tool 38](#_Toc370915655)

[2.7.1 Engineer Create or Update 40](#_Toc370915656)

[2.7.2 GEHCContract Create or Update 40](#_Toc370915657)

[2.7.3 GEHCSite Create or Update 40](#_Toc370915658)

[2.7.4 GEHCSystem Create or Update 40](#_Toc370915659)

[2.7.5 Skill Create 41](#_Toc370915660)

[2.7.6 Task Create or Update 41](#_Toc370915661)

[3 Inbound Interfaces 42](#_Toc370915662)

[3.1 Overview 42](#_Toc370915663)

[3.2 Incoming Messages Interfaces 42](#_Toc370915664)

[3.2.1 Incoming Messages Summary 44](#_Toc370915665)

[3.2.2 Message Header 45](#_Toc370915666)

[3.3 GetAddressGeocode 46](#_Toc370915667)

[3.4 ExtendedGetAppointmentsEx2 48](#_Toc370915668)

[3.5 ProcessTaskEx 61](#_Toc370915669)

[3.6 ExecuteMultiOperations 107](#_Toc370915670)

[3.7 GetTasks 113](#_Toc370915671)

[3.8 GetResources 115](#_Toc370915672)

[4 Error Handling 120](#_Toc370915673)

[4.1 Incoming Messages 120](#_Toc370915674)

[5 Outbound Interfaces 125](#_Toc370915675)

[5.1 FTP File System 125](#_Toc370915676)

[5.2 Messages Destinations 125](#_Toc370915677)

[5.3 File Naming 126](#_Toc370915678)

[5.4 Email 128](#_Toc370915679)

[5.4.1 Email Request Trigger 128](#_Toc370915680)

[5.4.2 Message Fields 128](#_Toc370915681)

[5.4.3 Message Content 129](#_Toc370915682)

[5.4.4 UNID generation 132](#_Toc370915683)

[5.4.5 Task ID Revision & Status Triggers 132](#_Toc370915684)

[5.4.6 Message Fields 133](#_Toc370915685)

[5.4.7 Message Content 133](#_Toc370915686)

[5.4.8 Outbound Messages XSD 136](#_Toc370915687)

[6 Outgoing Integration Manager 137](#_Toc370915688)

[6.1 Integration Logs 138](#_Toc370915689)

[Appendix A: Approvals 139](#_Toc370915690)

[Appendix B: C# Snippets 140](#_Toc370915691)

# Introduction

## About this Document

This document is the Interface Design Specification (IDS) for the interface of the GEHC systems MUST (Multiple Unit Service Tool, GEHC’s bespoke Job management system) to ClickSchedule. It contains detailed information about the various messages that will be used as part of the interfacing exercise.

This document is intended to complement the:

* High Level Solution Specification (BSS) document for the ClickSchedule and ClickMobile implementation at GEHC;
* Standard product documentation set provided by ClickSoftware:
  + Service Optimization Web Service API Developer’s Guide;
  + Service Optimization Integration Overview;
  + Service Optimization Integration Guide;
  + Service Optimization Scalability White Paper;

The information contained within this document does not, in any way, replace the information in the above documents. In the case of any conflicts across documentation, please consult with the implementation team.

## Disclaimer

It is important to note that this document is being written before the completion of the Detailed Designed Specification (DDS) document. Additional properties, message parameters changes or additional messages may be required and will be updated in this document during the design and/or development phase.

## Change History

| Date | Version | Who | What |
| --- | --- | --- | --- |
| 19-Aug-13 | 0.1 | Duncan Hardie | Initial draft |
| 08-Sep-13 | 0.2 | Shai Nahari | Initial Review |
| 11-Sep-13 | 0.3 | Duncan Hardie | Amendments during on-site IDS workshop |
| 13-Sep-13 | 0.4 | Duncan Hardie | Amendments after on-site IDS workshop |
| 20-Sep-13 | 0.5 | Duncan Hardie | Amendments after reviews |
| 27-Sep-13 | 0.6 | Duncan Hardie | Amendments after reviews |
| 01-Oct-13 | 0.7 | Duncan Hardie | Engineer & multi task updates |
| 24-Oct-13 | 0.8 | Duncan Hardie | Aligned with main dev database and QA findings addressed |
| 29-Oct-13 | 0.9 | Duncan Hardie | IsMST inclusion & other corrections |
| 07-Nov-13 | 1.0 | Duncan Hardie | Minor corrections detected by QA |
| 19-Nov-13 | 1.1 | Duncan Hardie | Minor corrections detected by QA |
| 22-Nov-13 | 1.2 | Duncan Hardie | WCF code sample |
| 03-Dec-13 | 1.3 | Duncan Hardie | Updated service proxy class |
| 14-Mar-14 | 1.4 | Duncan Hardie | HTTPS & Credentials |
| 14-April-14 | 1.5 | Duncan Hardie | * UNID alignment * CancellationReason * Group.Body XML * Outbound Message fields/Examles * Part Pickup have dependancies |

## Terms and Abbreviations

| Term | Description |
| --- | --- |
| ClickSoftware | ClickSoftware is the supplier of the Service Optimization applications. |
| SO | Service Optimization. A Software Product suite providing services for workforce management and optimization. ClickSchedule is one of the products in the suite. |
| Task | This refers to the call, task, operation or appointment to which the Resource is scheduled (equivalent of an operation in MUST)  ClickSchedule uses the term “Task” to represent the work to be scheduled. This term will appear throughout the ClickSchedule client application, and solution documentation. |
| Resource | Represents the resources assigned to carry out the service work for GEHC. The field resource can be an Engineer, a Crew or a Contractor. They have no direct access to the ClickSchedule client, but will have access to the ClickMobile client. |
| CS | ClickSchedule |
| XML | Extensible Markup Language. A text-based (user friendly) language. XML enables generic information to be served, received, and processed across distributed, platform independent (Unix/WinNT) systems. XML interfaces overcome DCOM/CORBA limitations across a LAN/WAN by using the standard HTTP (internet) protocol. |
| HTTP | The Hypertext Transfer Protocol (HTTP) is the set of rules for exchanging files (text, graphic images, sound, video, and other multimedia files) on the World Wide Web. Relative to the TCP/IP suite of protocols (which are the basis for information exchange on the Internet), HTTP is an application protocol. |
| Web Service | Web services are application programming interfaces (API) that can be accessed over a network, such as the Internet, and executed on a remote system hosting the requested services. ClickSchedule exposes a Web Service interface via the SOAP protocol. |
| Integration Manager | A component of the Service Optimization Server. Developed by ClickSoftware, to facilitate the integration between the ClickSoftware product suite and external applications. |
| Incoming Messages | Messages that an external system sends to the ClickSoftware Service Optimization server. |
| Outgoing Messages | Messages that the ClickSoftware Service Optimization server sends to an external system. These messages are based on events or actions that occur on the Service Optimization server. |
| Import Tool | Excel based tool which allows data migration into the Service Optimization server. Usually used for initial load of objects such as tasks, resources and dictionary objects into the system. |
| MUST | Multiple Unit Service Tool. GEHC’s bespoke Job management system. |
| SLA | Service Level Agreement. This is the time within which the service provider has agreed to address some activity related to a task. |
| Task | This refers to the call, task, operation or appointment to which the Resource is scheduled (equivalent of an operation / visit in MUST)  ClickSchedule uses the term “Task” to represent the work to be scheduled. This term will appear throughout the ClickSchedule client application, and solution documentation. |
| Job | A sequence of MUST Activities relating to a specific piece of GEHC equipment. |
| OOB/OOTB | Out of the box |
| MST | Multi Stage Task |

## In Scope

This IDS describes only the integration relevant parts of the GEHC-CKSW solution. i.e the electronic information flow between the existing GEHC systems and the CKSW systems. Other parts that are relevant for the solution but are irrelevant for the integration will not be described in this document but rather in the BSS.

The main entities which are integrated and described in this IDS:

* Task
* Resource (Engineer)
* Appointment Booking
* GCHCSystem
* GEHCContract
* GEHCSite

## Out of Scope

The following objects are not integrated and hence are out of scope for the IDS. These objects are described in the BSS and will be handled manually in the CKSW system.

* Transactional Data
* Jobs of type preventive maintenance
* Master Data
* Regions
* Districts
* Skills – Can be managed by Excel import;
* Calendars
* Mapping Table Post codes to District
* Mapping Table Site codes to District
* Mapping Job Type, Equipment Type to Task duration
* Mapping table Job Priority to SLA Duration

Any other mapping tables

* Code Lists
* GEHCCancellationReason
* GEHCCRMSystem
* GEHCIncompleteReason
* GEHCLanguage
* GEHCRejectionReason
* GEHCSystemStatus
* GEHCTaskSubType
* Task Status Codes
* Job Types
* Equipment Types
* Completion Codes
* Non Availability Types
* Engineer Types

# Integration Overview

## Landscape Overview

The following diagram illustrates the systems in the landscape and the relations between them:



Trace (Skills)



MUST (Job Management)



Service Optimization

Server



Skills Lookup

Get Appointments

Check address

Create/Update Tasks

Get Tasks



ClickMobile (Resource)



CS Web Client  
(Dispatcher)



Resource Skills



Excel

Site, System and Contract Definitions

Task Updates



FTP



Mail (SMTP)

CS Web client (Planner)

Desk client

ClickMobile client (Engineer)

Interfaces described in this document

Figure 1 – Integration Landscape

ClickSchedule has an object oriented service architecture, exposing Web Services that allow for platform-independent communication utilizing the SOAP data format.

On the GEHC side there are two systems which are relevant for the integration:

* MUST – is the GEHC Enterprise Ticketing System, dealing with incidents, troubles and changes. It is the single hub of all demand towards ClickSoftware with regards to Tasks;
* Email exchange – To create customer emails based on data provided by SO;

The MUST SDT will invoke Web Service operations in the Service Optimization integration servers. The operations perform specific functions on various objects in the system such as creating/updating/cancelling tasks, appointment bookings and associated data. The operations contain all the necessary data elements in an XML format wrapped in a standard SOAP envelope, sent via an HTTP connection to the Service Optimization Web Services.

## Service Optimization Web Services

## Available Services

The following Web Services exposed for integration are applicable to the integration relevant for GEHC covered by this document:

* **ScheduleService.svc -** Contains ClickSchedule operations;
* **ServiceOptimizationService.svc** - Contains generic optimization operations;

## Namespaces

Unless otherwise noted, the namespace of all Service Optimization Web Services is:

http://www.clicksoftware.com

The namespace of the xsi:type attribute is:

http://www.w3org/2001/XMLSchema-instance

The namespace of the OptionalParameters SOAP header is:

http://www.clicksoftware.com/OptionalParameters

Note: Where possible namespaces will be included within SOAP examples.

## Target URL

The target URL for each web operation will be identified for each message.

## Utilizing SOAP 1.1

Messages received via the integration layer will be in a SOAP 1.1 compliant format. It is possible to send the outgoing messages from ClickSchedule with or without the SOAP envelope. According to requirements gathered in the integration session of the Solution Review workshop, the requirement is to send Outgoing messages with a SOAP envelope. An example of the ProcessTaskEx Operation correctly wrapped in a SOAP envelope can be found here, 3.5.2.

Note: For the remainder of this document the proper SOAP envelope is assumed in all examples and for simplification only the contents of the body tag will be discussed and shown.

## Data Formats

#### XML Format

Illegal XML characters have to be replaced by entity references. Placing characters such as "<" in an XML element, will generate an error since the parser interprets it as the start of a new element. To overcome this, the characters musts be replaced with the corresponding XML symbol. There are 5 predefined entity references in XML (valid for all incoming and outgoing messages):

|  |  |  |
| --- | --- | --- |
| &lt; | < | less than |
| &gt; | > | greater than |
| &amp; | & | ampersand |
| &apos; | ' | apostrophe |
| &quot; | " | quotation mark |

**Note:** Only the characters "<" and "&" are strictly illegal in XML. Apostrophes, quotation marks and greater than signs are legal, but it is a good habit to replace them.

For example:

<District>North & East</message>

Should be sent as:

<District>North &amp; East</message>

**Note:** XML tags are case sensitive. Tag “<Task>” and tag <task> will not be parsed as the same way and will be considered as different tags.

#### Boolean format

FALSE could be sent as **“0”** or **“false”**

TRUE could be sent as **“1”** or **“true”**

#### Date & Time Formats

All Date & Time properties\attributes must be passed in the 24-hour format with accordance to ISO 8601 format. Acceptable formats include:

* yyyy-mm-ddThh:mm:ss e.g. 2001-06-30T11:00:00
* yyyy-mm-dd hh:mm:ss e.g. 2001-06-30 11:00:00

In addition, the seconds in all the time attributed must be “00” (hh:mm:00), for example 11:00:00 is valid but 11:00:02 is not valid.

## Cloud Offline Development

It is possible to construct Click objects that will work with the Web Services while an environment is unavailable, proxy .net classes can be used to code against.

Once an environment is available, then the proxy classes can be excluded from a VS project and actual Service References defined the same name as the proxy classes can be added and consumed.

Note: The proxy classes must have differing namespaces so common objects between services are distinct within the object model the \*.config file contains the binding information.



#### Credentials

The SOSAP packed must be transmitted via HTTPS, challenge response basic authentication is enable, for the UAT environment he credentials are as follows:

* User Name: "csodprod/gehc.integration”
* Password: "w6int.service"

Note: Credentials are subject to change, the production credentials will be communicated in an appropriate manor.

#### WCF code sample

Below is a simple example of how to connect to the GE Integration services using code controlled bindings and Client credentials:

|  |
| --- |
| using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Xml.Serialization;  using System.ServiceModel;  using SOServiceInt;  namespace ConsoleApplication1  {  class Program  {  public static BasicHttpBinding basicHttpBinding = new BasicHttpBinding(new BasicHttpSecurityMode());  public static EndpointAddress serviceOptimizationServiceEndpointAddress = new EndpointAddress("https://gehc-so.clicksoftware.com/SO/IntegrationServices/ServiceOptimizationService.svc/BasicHttpInt");  static void Main(string[] args)  {  basicHttpBinding.ReaderQuotas = System.Xml.XmlDictionaryReaderQuotas.Max;  basicHttpBinding.Security.Mode = BasicHttpSecurityMode.Transport;  basicHttpBinding.Security.Transport.ClientCredentialType = HttpClientCredentialType.Basic;  basicHttpBinding.Security.Transport.ProxyCredentialType = HttpProxyCredentialType.None;  GetAddressGeocodeRequest getAddressGeocodeRequest = new GetAddressGeocodeRequest();  getAddressGeocodeRequest.OptionalParameters = new OptionalParameters() { CallerIdentity = "MUST", ErrorOnNonExistingDictionaries = false };  getAddressGeocodeRequest.NumOfMatches = 10;  getAddressGeocodeRequest.Location = new GeocodeLocation();  getAddressGeocodeRequest.Location.PostCode = "SL1 7LW";  getAddressGeocodeRequest.Location.Country = "UNITED KINGDOM";  List<GetAddressGeocodeResponse> getAddressGeocodeResponses = new List<GetAddressGeocodeResponse>();  using (ServiceOptimizationServiceClient serviceOptimizationServiceClient = new ServiceOptimizationServiceClient(basicHttpBinding, serviceOptimizationServiceEndpointAddress))  {  serviceOptimizationServiceClient.ClientCredentials.UserName.UserName = "gehc.integration";  serviceOptimizationServiceClient.ClientCredentials.UserName.Password = "w6int.service";    serviceOptimizationServiceClient.Open();  var result = serviceOptimizationServiceClient.GetAddressGeocode(getAddressGeocodeRequest.OptionalParameters, getAddressGeocodeRequest.Location, getAddressGeocodeRequest.NumOfMatches);  GetAddressGeocodeResponse getAddressGeocodeResponse = new GetAddressGeocodeResponse();  getAddressGeocodeResponse.Result = result;  getAddressGeocodeResponses.Add(getAddressGeocodeResponse);  }  }  }  } |

## Task Unique identifiers

#### Task (Visit) UNID

OOTB, CallID is used in conjunction with the Number property to uniquely identify a Task in Click. Dependent tasks (in MSTchain) that related Tasks must have the same CallID and a different Number than the Primary Task.

When referencing a Task via integration, CallID and Number must always be specified, other mandatory properties are only needed upon creation unless an update alteration to that property value is required.

MUST Job visits do not have a UNID. A derived Task.TaskID will be created for MUST to correlate visit records based on their existing data querying. Click will return Task updates via integration and the TaskID passed by a MUST interface to correlate the correct MUST records for update.

The TaskID will be composed by a concatenation of values common between MUST & Click. Given the nature of the data to be correlated the TaskID is dynamic so both the old & new TaskID values will be passed by the outbound message to MUST.

The Old value will be used to correlate the existing MUST record & the new value will replace the MUST records current TaskID (the old TaskID).

The TaskID is first generated by MUST on creation of the task, for any of the concatenation elements where data is not available unique values should be defined per the format below. As the Task details are changed in Click (assigned or reassigned to engineers) the TaskID will change, the old TaskID (1 pervious to the changed value) along with the newly created TaskID will be included in outbound integration to MUST.

|  |  |  |
| --- | --- | --- |
|  |  |  |
| CallID | Country Code – MUSTJobNumber – Numeric Suffix  e.g. “GB-B3607131-1” | * On single Task creation the CallID; * On Pick Up, double manning and training tasks the CallID will be set to be equal to the main Task’s CallID, which is the same as the main task’s MUSTJobNumber; |
| Number | Visit number | Any unique number per CallID, gaps are permitted; |
| MUSTJobNumber | MUST job number | Set by MUST Needs to remain unchanged; |
| TaskID (Derived UNID) | Initial value form MUST:  CallID-DummyEngineerBadgeNumber\_MUSTVisitDate  Click Schedule Assigned job:  MUSTJobNumber\_Engineer.MUSTID\_ClickAssignmentStart | * DummmyEngineerBadgeNumber = For Dummy records overlapping start times are negated, TaskID will remain unique; * MUSTVisitDate = Now(), for dummy jobs; * Click Assignment start = assignment start time on dispatch change (should not be constantly revised); * Click Schedule will update the TaskID and communicate to MUST upon the following task update outgoing message as detailed in [Task](#_Task) section; * In case the TaskID is rejected by MUST it will update the task with an acceptable alternate ID; |

Note: The CallID and Number combination must be unique and cannot be of a Task that exists in the system.

Note: Training tasks TaskID may have differing MUSTJobNumber that that in the main Task.

Example UNIDs:

* CallID = “FR-A1234567”;
  + For a single/main Task the CallID is the same as the MUSTJobNumber;
  + Pickup, Training and Double Manning Tasks will have the same CallID as the main one;
* Number = “1”
  + Pickup, Training and Double Manning Tasks will have a differing number than the main one;
* MUSTJobNumber = “FR-A1234567”;
  + Actual MUST Job MUSTJobNumber;
* TaskID = “FR-A1234567\_D4\_20130909T0958“
  + MUSTJobNumber = “FR-A1234567”;
  + EngineerUNID (BadgeID) = “D4”;
  + MUSTStartDate / assignment start on dispatch date / unique date selected by MUST SDT macro = “20130909T0958” ;

Note: If a Training jobs main job is cancelled then the related training job needs to be cancelled;

Note: Visits can be cancelled in Click and send back to MUST for deletion via Web services;

Note: If jobs are cancelled or closed from MUST, related visits are cancelled in Click;

## On-Line Integration Activities

## GetAddressGeocode - Address Validation

Initially MUST tasks may have semi or inaccurate addresses which require user validation and clean up within MUST. Ensure that the MUST data entry is sufficient for SO to determine accurate coordinates the GetAddressGeocode message should be sent to Click. The results will provide the MUST user the coordinates that Click will derive from the entered address details, ensuring data validation.

## ExtendedGetAppointmentsEx2 – Visit Appointment Bookings

Appointment Booking is the means by which Appointments can be requested from ClickSchedule via the integration layer.

The available appointment slots for a specific task are requested from the Service Optimization server. The Service Optimization server responds with timeslots within which the task could be scheduled. A timeslot is selected matching the Customer's preference time period (assumed up to 14 consecutive days for full day slots, 14 days for AM/Pm slots and 3 days for 2h slots), an Appointment Booking Profile name, and inserts all Task Details prior to asking for available appointment slots.

The Appointment Search can occur for tasks that already exist in ClickSchedule and for new tasks that have not yet been created in ClickSchedule. Sending a request for a Task that already exists in ClickSchedule is called "Rebooking".

This scenario does not deal with task creation – it only covers the query for the available appointment slots.

## ProcessTaskEx – Tasks Create or Update

ClickSchedule will be the main interface for the control and scheduling of tasks, while all tasks (with certain exceptions described in section **Error! Reference source not found.**) will be created by the MUST and sent to ClickSchedule. Once a task resides in the ClickSchedule repository it may be scheduled, re-scheduled or unscheduled by ClickSchedule automatically or by the planner. Some of these actions may cause a message to be generated and transmitted back to MUST containing information about the task execution progress. Upon the configuration of the system, it will be possible to select in which cases messages should be sent from ClickSchedule to the MUST in a manner that no unnecessary messages are sent.

* MUST is owner of Tasks;
* Most tasks in the system will be created in MUST and sent to SO via integration. Updates to these tasks applied in SO will be communicated back to MUST;

## GetTasks – Retrieving Latest Visit Information

To retrieve details about tasks within Click the GetTasks action is used, see 3.7 for the SOAP payload required to return tasks by CallID.

To retrieve tasks by MUSTJobNumber the Indexes would change to the syntax.

## GetResources – Retrieving Engineers

To retrieve details about engineers within Click the GetResources action is used, see 3.7 for the SOAP payload required to return tasks by Index Keys, namely ID & MUSTID.

## Additional Objects via Periodic Import

The following business objects are maintained through integration or manually synchronized between systems via the import tool using Excel templates or the ExecuteMultipleOperations action:

* Engineer, see 2.7.1
* Skills, see 2.7.2
* GCHCSystem, see **Error! Reference source not found.**
* GEHCContract, see **Error! Reference source not found.**
* GEHCSite, see **Error! Reference source not found.**

## Intergration Scenarios

This chapter describes the scenarios for integration, below are the use cases which are relevant for the integration.

## Appointment Booking

GEHC would like to use the CKSW’s Appointment Booking capabilities in order to find available appointment slots and schedule appointments. Users would be able to quickly see the available slots. As part of the AB scenario the ServiceOptimizationService.GetAddressGeocode may be used to validate the location of jobs. The GetAddressGeocode would be itteratvley called from the MUST SDT macro until an address is deemed accurate.

The planned process is composed in the following steps:

1. GetAddressGeocode iterations to return accurate location, see 3.3;
2. Get available time slots for appointments within a relatively short time horizon (e.g one week). Slots shall retrieved graded or non-graded, see 3.4.
   1. Repeat the process described in step 2 for additional weeks upon user’s choice;
   2. Graded appointment may degrade performance;
   3. Related tasks
3. User selects the required time slot and books the appointment,
4. Tasks are created in CS per the integration scenarios listed 2.6.

#### AB Conformation (Booking the appointment)

After getting the possible slots MUST can schedule an appointment. Once the necessary information is gathered a Task can be created by invoking the service ServiceOptimizationService.ProcessTaskEx, see 3.5. A response will be sent back to with the details of the task that was created in SO.

The following sequence diagram describes the process:

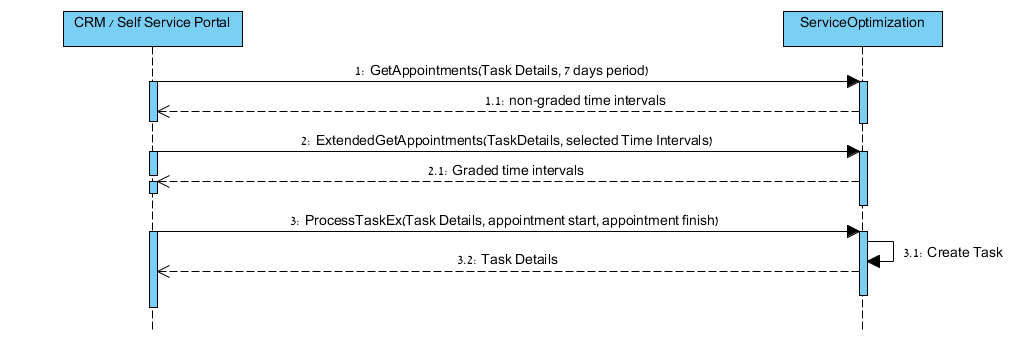


Figure 2 - appointment booking flow

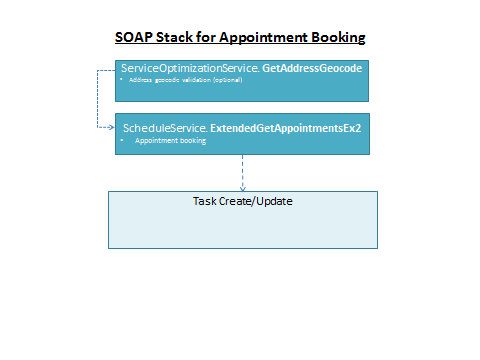


Figure 3 - Sequence Diagram Create Task

## Task Creation/Modification

Different integration scenarios are required for modifying Tasks in Click, depending on the scenario for the required Task the SOAP messages will differ. These scenarios are as follows:

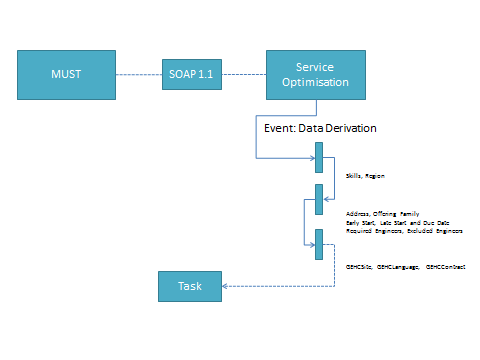
1. Single Task
2. Multi Manning task (MST)
3. Parts Pickup Task (MST)
4. Training task (MST)

Task are not expected to be created in SO, tasks will always originate from MUST.

When a Task is created in MUST, an incoming message will be sent to SO invoking the service ScheduleService.ProcessTaskEx passing the Task fields as described in chapter 3.5.2. SO will then process the message as described in the sequence diagram below, performing various data derivations.

A TaskID value is also essential for correlating updates between Click & MUST, given the MUSTID is not unique between jobs within MUST a derived TaskID is generated by both MUST & Click platforms, and maintained via integration. Upstream TaskID values from Click will contain both new & old (altered value) TaskID, see 5 for Click outbound message details.

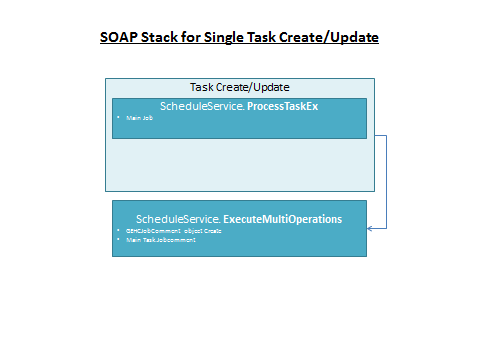
The Task Unique identifiers are described in greater detail here: 2.3



#### Single Task

The AB scenario may optionally precede this scenario. The SOAP action for modifying a task (Create or Update) is detailed in 3.5. The UNID values should ahead to the GEHC standards, 2.3.

A single Task creation scenario will use the ProcessTaskEx action to create or modify the Task, given the debrief comments may have a considerable payload these are created separately and linked to the Task JobComments field. Using a SOAP message for the ExecuteMultipleOperations action the GEHCJobComments Object containing the comments is created and then linked to the Task.JobComments, if there are no Comments the Object should still be created with no Text value. The SOAP stack will resemble the following:



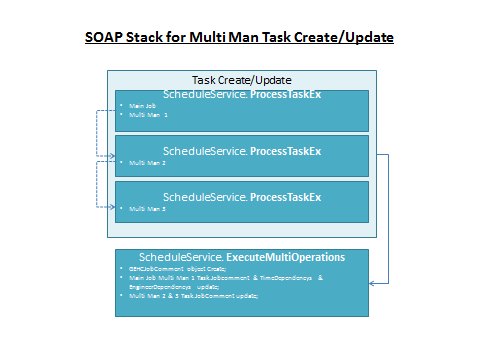
#### Multi Manning Task

The AB scenario may optionally precede this scenario; Task Dependencies and MSTs detailed in this scenario are applicable to the AB scenario and must be included in the Related Tasks SOAP body element.

Multi manning tasks is a visit of a job that require two or more resources to perform a unit of work. The creation of Multi Manning Tasks will be done by creating multiple tasks for the same time time, site and for different engineers, one Task will hold the dependancies and required skills, the rest of the tasks may have skills listed optionaly. Creation of these remains the same as a Single Task in that the ProcessTaskEx will be executed for each Task requiring an engineer. The UNID values should ahead to the GEHC standards, 2.3.

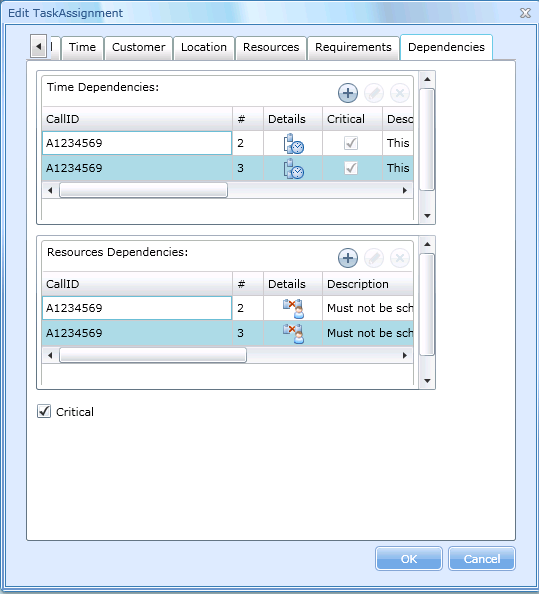
1. The preceding ExecuteMultipleOperations following the X number of ProcessTaskEx for the required number of engineers will do the following:
2. Create the GEHCJobComments Object;
   1. Multi Manning jobs will all have the same MUSTJubNumber;
3. Link all Multi Manning MSTs to the GEHCJobComments Object;
4. Create dependencies on the Main/Primary Task:
   1. Task.Critical for all tasks must be True on all Tasks holding dependencies;
   2. Dependencys.Critical must be set to True
   3. Time Dependency to start MST’s at the same time as each other;
   4. Resource dependency for MST’s so that resources must not be the same;

Note: All main tasks for this scenario should be flagged as Critical e.g. Task.Critical = True.

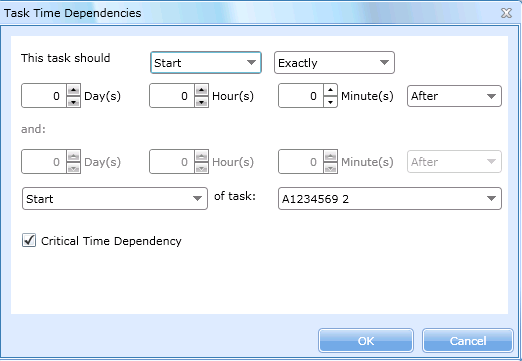


Screenshots from the Web frontend shows the resulting dependencies in the GUI.

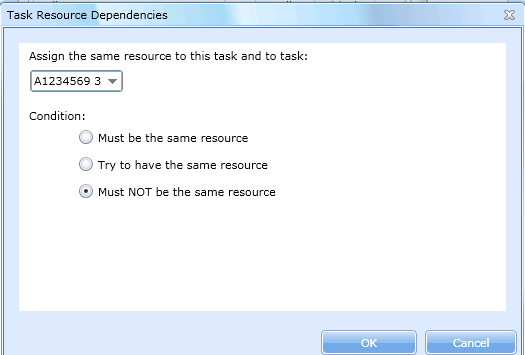
Main Task Dependencies:



Task Time Dependancy:



Task Resource/Engineer Dependency:



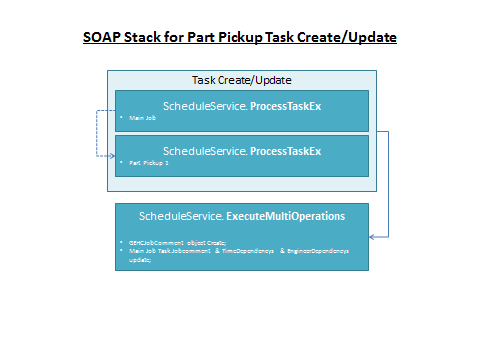
#### Parts Pickup Task

The AB scenario may optionally precede this scenario; Task Dependencies and MSTs detailed in this scenario are applicable to the AB scenario and must be included in the Related Tasks SOAP body element.

Parts Pickup tasks are jobs that require a preceding Task to that of the main Task to collect parts. The creation of Tasks remains the same as a Single Task in that the ProcessTaskEx will be executed for main and Parts Pickup Tasks. The UNID values should adhere to the GEHC standards, 2.3.

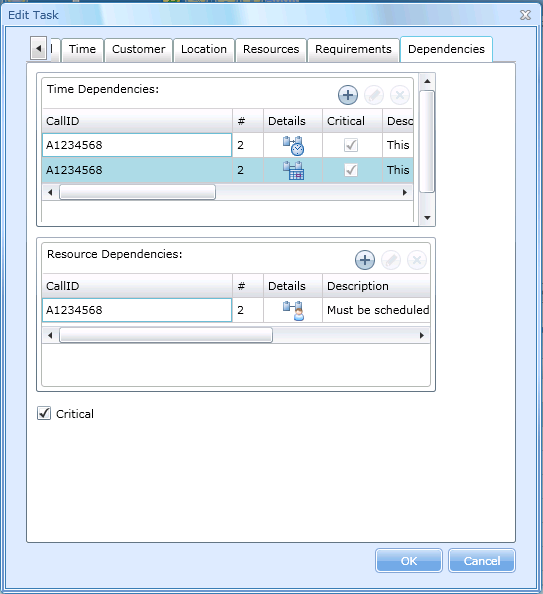
1. The preceding ExecuteMultipleOperations following the ProcessTaskEx for the Parts Pickup and main job will do the following:
2. Create the GEHCJobComments Object;
3. Link only the Main Job/task to the GEHCJobComments Object;
   1. GEHCJobComments can be large and not required for the pickup;
4. Create dependencies on the Main/Primary Task:
   1. Task.Critical for all tasks must be True on all Tasks holding dependencies;
   2. Dependencys.Critical must be set to True
   3. Time Dependency to start MST’s sequentially, one after another;
   4. Time Dependency to start MST’s on the same day as each other (optional);
   5. Resource dependency for MST’s so that resources must the same;

Multi Parts Pickup jobs are not covered in detail within this document however the dependency structure would remain the same, all MSTs link to the main task. All main tasks for this scenario should be flagged as Critical e.g. Task.Critical = True.

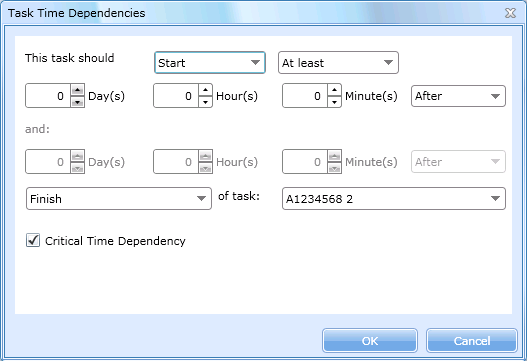


Screenshots from the Web frontend shows the resulting dependencies in the GUI.

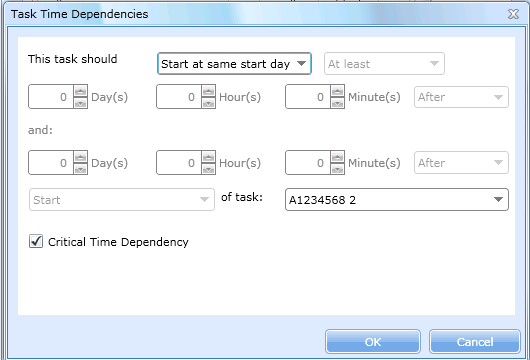
Main Task Dependencies:



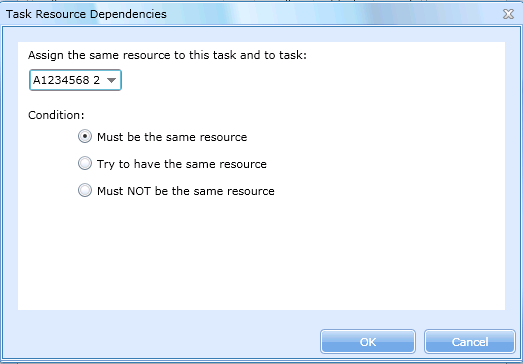
Task Time Dependancy 1:



Task Time Dependancy 2:



Task Resource/Engineer Dependency:



#### Training Task

The AB scenario may optionally precede this scenario; Task Dependencies and MSTs detailed in this scenario are applicable to the AB scenario and must be included in the Related Tasks SOAP body element.

Training Tasks are jobs that require a trainee to accompany the main task. The creation of Tasks differs from the Single Task in that an engineer that requires training can be set to the Task at creation, the ProcessTaskEx will be executed for each Task requiring an engineer. The UNID values should ahead to the GEHC standards, 2.3, the TaskID and MUSTJobNumber may differ between training and those on the main Job.

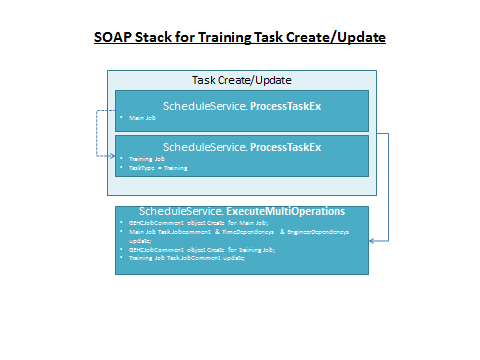
**When both Tasks released from MUST:**

1. The preceding ExecuteMultipleOperations following the X number of ProcessTaskEx for the required number of engineers will do the following;
2. Create the GEHCJobComments Objects;
   1. Training jobs will have a differing MUSTJobNumber by default;
   2. Each MUSTJobNumber will have a GEHCJobComments record, therefore the main task should point to a different GEHCJobComments record than the training record;
3. Link each Task to relevant GEHCJobComments Object;
4. Create dependencies on the Main/Primary Task:
   1. Dependencys.Critical must be set to True
   2. Time Dependency to start MST’s at the same time as each other;
   3. Resource dependency for MST’s so that resources must not be the same;

Note: GEHC may decide that a Training Job is linked to a JobComment with an ID other than the trining jobs MUSTJobNumber, the SOAP would reference accordingly;

**When Training Task is released from MUST after Main Tssk is in place:**

1. The preceding ExecuteMultipleOperations following the X number of ProcessTaskEx for the required number of engineers will do the following;
2. Create the GEHCJobComments Object for the training;
   1. Training jobs will have a differing MUSTJobNumber by default;
3. Link the Training Task to the GEHCJobComments Object;
4. Create dependencies on the Main/Primary Task:
   1. Dependencys.Critical must be set to True
   2. Time Dependency to start MST’s at the same time as each other;
   3. Resource dependency for MST’s so that resources must not be the same;



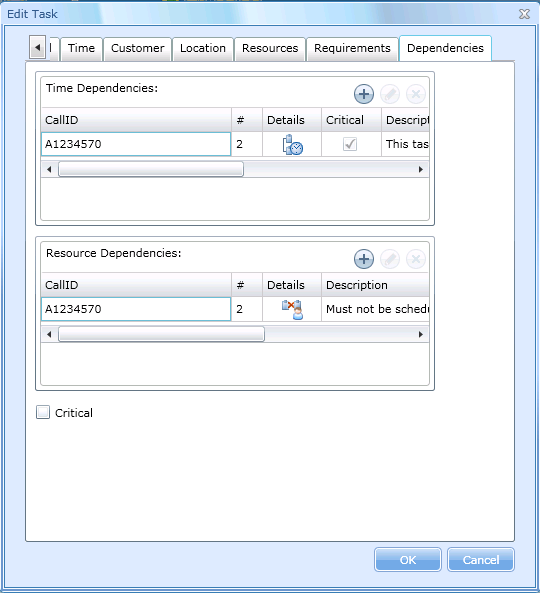
Below is the EngineerRequirements fragment for the ProcessTaskEx.Task payload to create the training task.

|  |
| --- |
| <ProcessTaskEx xmlns = "http://www.clicksoftware.com">  <Task>  …  <TaskType>  <Name>Training</Name>  </TaskType>  **<RequiredEngineers>**  **<RequiredEngineer>**  **<ID>Trainee01</ID>**  **<District>**  **<Name>UK South</Name>**  **</District>**  **</RequiredEngineer>**  **</RequiredEngineers>**…  </Task>  …  </ProcessTaskEx> |

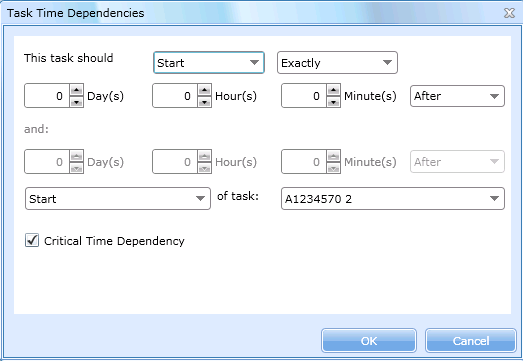
Screenshots from the Web frontend shows the resulting dependencies in the GUI.

Main Task Dependencies:

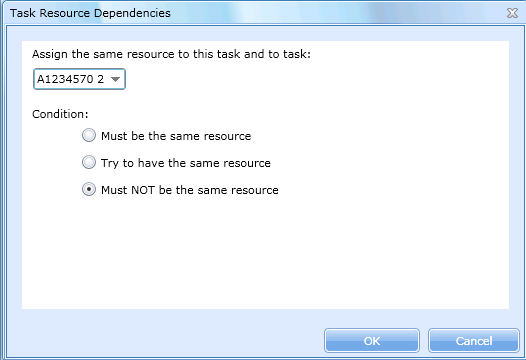
Training Jobs will be linked with a dependency on the Main Task, Training Jobs will be identified by the TaskType “Training”. Both the main Task and training Task need the same CallID & differing Number values with the training a higher number than the main task. Also the Tasks to be created prior to the dependency relationship link. Once both Tasks are in place then a TimeDependencys object will be placed on the Main Task linking the MSTs.



Task Time Dependancy:



Task Resource/Engineer Dependency:



#### Cancelling Tasks

Jobs can be cancelled and interrupted. Interrupted Jobs are jobs that that have Tasks in progress, for example onsite, to cancel a job the MUST status will be set to INT which will send a ProcessTaskEx action to Click to change the the Task Status to “Cancelled”. All relevant tasks in the MST chain must then have their Task Status set to “Cancelled” also.

Cancelled jobs are those Tasks that are no longer required, in the future & not in progress.

Multi Man and Training Jobs run in parallel. If a Training jobs main job is cancelled then the related training job needs to be Cancelled iteratively using the ProcessTaskEx action.

Related tasks can be retrieved from Click using the GetTasks action defined in section 3.7 the Task.CallID should be used to gather the MST group.

A generic event customisation created for GE will cascade/disseminate cancellations from main tasks (tasks that hold the dependancies. So GE will only need to do cancellation of tasks for which are not main tasks.

#### MST Considerations

In order for the scheduling engine to start the scheduling of MST tasks once all relevant tasks are inplace the Task.IsMST property should be set:

* IsMST = true
  + For tasks that have or are dependant tasks and or part of an MST chain;
  + The flag should be set to true as the earliest possible opportunity, before the ExecuteMultiOperations actualy sets dependancies;
* IsMST = false
  + For tasks that haven’t got or arnt dependancies and not part of an MST chain;

Shoud a task be created with an IsMST value of false and and then later becomes part of an MST chain (for example training is required significantly later than the main task creation), then the IsMST value should be set to true, this can be made at the same time as the dependancies are created with the ExecuteMultiOperations.

## BSS Business Scenario elaboration

This section elaborates at a higher level the types of actions that were defined in the BSS.

#### Business Scenario 1B: Dispatch Work

When a Task is updated in MUST, an incoming message will be sent to SO invoking the service ScheduleService.ProcessTaskEx. Data will be derived to populate values for the task within Click, no data derivation is performed upon Task update. Below are characteristics for GEHC integration ProcessTaskEx messages:

* Task object reference fields (the business key) CallID & Number, cannot be updated since this is the unique identifier of the Task;
* A MUSTJobNumber shouldn’t be altered within MUST or Click;
* Scheduling relevant changes of the Task such as updates of the Job type, due date, priority etc. will trigger rescheduling based on the criteria set on the BSS.
* Changes of the task geographical data  (Postcode and house number ) will not influence district determination;

Most derivations in SO are defined only for Task creation, GEHC are due to identify how updates on derivation fields will be re-derived and how this will affect the schedule;

Note: Should the Engineer or Appointment times change and outbound message will be sent to MUST with the regenerated TaskID and old TaskID, as defined in **Error! Reference source not found.**:

##### Activity: Allocate Resource to Activity

When a resource/engineer is allocated or reallocated to a Task within Click or MUST the TaskID is automatically regenerated to include the Engineer ID, see: 2.3 & **Error! Reference source not found.**

Its essential the TaskID is maintained in both MUST and Click so that Tasks & MUST actions can be correlated so that both systems share the same version data.

When any property that composes the TaskID is altered in either system then both the old and new TaskID values need to be exchanged between Click & MUST. Therefore when a Click Task is allocated an engineer an outbound message will to be sent to MUST via FTPS.

MUST will be sent an UpdateTaskEmail message to notify the customer of the job allocation. See, 5.4.1

#### Business Scenario 5.0 – Schedule a Training job

Training Jobs are created with ScheduleService.ProcessTaskEx in the same way as normal jobs.

An MST dependency start to start of type critical with the main Task will ensure that the training job will be created and performed alongside the trainer.

Note: Outbound messages for email will be suspended for training tasks, see 5.4.1.

#### Business Scenario 6.0 – Job Interrupted

##### Activity: Change Job Status to INT (interrupted)

To ascertain whether the MUST job status can change to “INT” the Task Status a GetTasks call referencing the TaskID will need to be made to ensure that neither a primary or training job has been started for the MUST TaskID of the call being cancelled.

##### Activity: Cancel all Appointments in the Future related to this Job

The UpdateObjects action can be used to change the status of tasks to a completed status. A Condition can beset within the task Body element to apply the change to all tasks in the group.

#### Business Scenario 10 – Update Job Details

When a Job is updated in MUST the updates are passed to Click by the ScheduleService.ProcessTaskEx in the same way as MUST would create exchange the message on creation of a new job.

Should any of the following properties change outbound messages will trigger as defined in: 5

#### Task.Status change to On Route

When an engineer is On Route an UpdateTaskEmail outbound message needs to be sent from Click. See: 5.4.3.3

#### Task cancelled in MUST (BSS Use case MUST3)

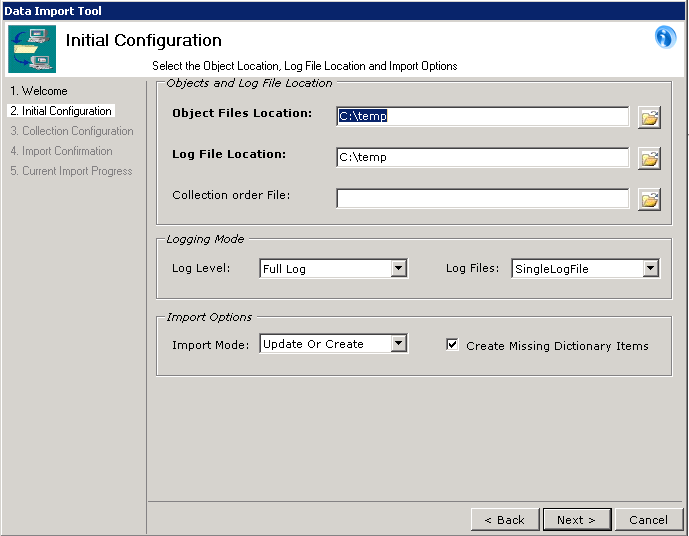
MUST may decide to cancel the task in CKSW if the work shall not be performed after the task was already created. Tasks could be cancelled as long as the status is not yet “Assigned”. Cancellation is triggered via calling ScheduleService.ProcessTaskEx with status = cancelled. Task status will be set accordingly in SO.

## Excel imports with Click Data Import tool

The Data Import tool should be used to maintain specific GEHC objects. It provides GEHC simple interoperability between Excel & the SO platform.

The Data Import tool is activated by the “Import/Export Data” option in Service Optimization Administration tool.

The tool accepts input data stored in Microsoft Excel files, and it can add new data to the database or update existing data. The UI is as follows:



Note: Multi value fields (for example Skills and Languages) are derived by splitting fields, the “relational” worksheet will be included in the GEHC Excel templates. One or other approach should be used however on import. The collection order file is as follows:



## Engineer Create or Update

It is expected for the Click Import export tool to help maintain the Skills collection. Below is an Excel template that for the object:



Resources will be created and updated in some manner by GEHC staff external to ClickSchedule and these updates communicated to ClickSchedule via flat files of an agreed format.

Resource deletion is not supported through this interface since the GEHC staff will not know whether a given resource can safely be removed from the ClickSchedule repository – there may well still be references to the resource from other objects in the repository. As such, deletion should be modelled by simply updating the target resource to set it as inactive.

Resource home bases are geo-coded during import into ClickSchedule.

Below are the property details:

| Property | Visible Name | Description | Type | Access | Owner | Mandatory? |
| --- | --- | --- | --- | --- | --- | --- |
| **Personal** | | | | | | |
| Name[[1]](#footnote-1) | Name | The resource’s name, for display purposes only. | String (256) | Non-Editable | GEHC | Yes |
| ID | ID | Unique identifier of the resource. The GEHC SSO identity or (prefixed) MUST ID depending on whether the engineer is in MUST or Siebel. | String (64) | Non-Editable | GEHC | Yes |
| MUSTID | MUST ID | The resource’s unique identifier within MUST (with any necessary prefix). | String (64) | Non-Editable | GEHC | Yes[[2]](#footnote-2) |
| SSO | SSO | The resource’s unique SSO identifier. This is used when logging into the ClickMobile client. | String (64) | Non-Editable | GEHC | Yes |
| EngineerType | Type | The resource’s type. Defaults to “Field Service Engineer”. | Dictionary (ResourceType) | Non-Editable | GEHC | No |
| Characteristic | Characteristic | A formatted string, typically used to hold key MUST characteristics such as modalities, SAV#, flattened skills list etc. allowing searching and filtering of the resources against these characteristics. | String (500) | Non-Editable | GEHC | No |
| ServiceArea | Service Area | The name of the service area in which the resource operates. | String (100) | Non-Editable | GEHC | No |
| MobilePhone | Mobile Phone | The resource’s mobile phone number. Enables the Dispatcher to call the resource if needed. | String (64) | Non- Editable | GEHC | No |
| Calendar | Calendar | The (name/ID of the) calendar that defines the resource’s working time. Defaults to the District’s calendar. | Reference to Calendar | Editable | GEHC /CS | No |
| Active | Active | An active Resource is one to which work can be assigned. A Resource that is unavailable for a long period of time (e.g. maternity leave) will be marked as inactive. | Yes/No (Boolean) | Editable | CS | Yes |
| Internal[[3]](#footnote-3) | Internal | Internal resources are employees of GEHC. External resources are employees of sub-contractors of GEHC. | Yes/No (Boolean) | Non- Editable | GEHC | Yes |
| TravelPolicy | Travel Policy | Engineers may either travel to and from their home base in company time (i.e. working hours) or their own time, as controlled by this setting. | Dictionary (TravelProfile) | Non-Editable | GEHC | Yes |
| LunchBreakDuration | Lunch Break Duration | Specifies how long the resource’s lunch break should be. | Duration | Non-Editable | GEHC | Yes |
| LunchBreakType | Lunch Break Type | Specifies how the Lunch break should be included. Based on “After Time” or “After Duration” values. | Reference to LunchBreakType | Non- Editable | GEHC | Yes |
| LunchStartsFrom | Lunch Starts No Earlier Than | When applicable in the resource’s region, this specifies the earliest time of day when the resource may start their lunch. | Date | Editable when Lunch BreakType is “Break After Time” | GEHC | When LunchBreakType is “Break After Time” |
| LunchStartsAfter | Lunch Starts After Working At Least | When applicable in the resource’s region, this specifies the duration after a resource starts work for the day that must pass before they may start their lunch. | Duration | Editable when Lunch BreakType is “Break After Duration” | GEHC | When LunchBreakType is “Break After Duration” |
| FSELanguages | Languages | A sequence of semi-colon separated language names. | String (1024) | Hidden | GEHC | No |
| Languages | Languages | The set of languages spoken by the resource. This is used to limit the tasks that the resource is automatically scheduled to. If unspecified the resource is assumed to speak all languages. Derived from the FSELanguages. | Dictionary (Language, Multi-value) | Non-Editable | CS | No |
| MobileClient | Has Mobile Client | Indicates if the resource is using ClickMobile. Defaults to No. | Yes/No (Boolean) | Editable | CS | No |
| LoginName | Mobile Login | Specifies the username associated to the CM Touch login | String (128) | Editable when MobileClient is Yes | CS | When MobileClient is Yes |
| MobileWebClientSettings | Mobile Template | Mobile settings template name. This needs to be set to a valid selection when the MobileClient property is set true otherwise an error will occur during dispatch to this resource. | Reference to UserSettings | Editable when MobileClient is Yes | CS | When MobileClient is Yes |
| MaxTimeFromHB | Maximum Travel Time From HB | The maximum time that the resource can travel between their home base and a task. Use of this parameter ensures that regional level automatic scheduling does not result in unnecessarily large amounts of travel. | Duration | Editable | CS | No |
| **Location** | | | | | | |
| Region | Country (Region) | The highest level in the organizational hierarchy. | Dictionary (Region) | Non-Editable | GEHC | Yes |
| District | Zone (District) | The second level in the organizational hierarchy. | Dictionary (District) | Non-Editable | GEHC | Yes |
| Street1 | Street | The resource's home base address, i.e. the location of where the resource normally starts and finishes their working day. | String (64) | Non- Editable | GEHC | Yes, enough to enable geo-coding. |
| City | City | String (64) | Non-Editable | GEHC |
| State | County | String (64) | Non-Editable | GEHC |
| Postcode | Postcode | String (64) | Non- Editable | GEHC |
| Country ID | Country | The country in which the address exists. | Dictionary (Country) | Non- Editable | GEHC |
| Longitude | Longitude | The geo-coded location for the resource’s home base. These are derived on creation or update of the resource’s address. | Number | Hidden | CS | No |
| Latitude | Latitude | Number | Hidden | CS | No |
| **Professional** | | | | | | |
| AllowedTaskTypes | Allowed Job Type | The type of task that the resource is permitted to undertake. This is used to limit the tasks that are automatically scheduled to the resource. When unspecified a resource can perform all types of task. | Dictionary (TaskType) | Non-Editable | GEHC | No |
| Skills | Skills | The set of skills, by level, that the resource has. This is used to limit the tasks that are automatically scheduled to the resource. | Graded Skills  (Multi-value) | Non-Editable | GEHC | No |
| ManagerID | Manager ID | The resource’s manager’s GEHC SSO identity. Not currently used. | String (64) | Hidden | GEHC | Yes |
| ManagerName[[4]](#footnote-4) | Manager Name | The resource’s manager’s name. For display purposes only. | String (256) | Non-Editable | GEHC | Yes |
| ManagerContactNumber[[5]](#footnote-5) | Manager Contact Number | The phone number via which the resource may contact their manager. For display purposes only. | String (64) | Non-Editable | GEHC | No |
| ManagerEmail | Manager E-mail Address | The e-mail address for the manager. For use when automated e-mails are to be sent. | String (256) | Non-Editable | GEHC | Yes |

## GEHCContract Create or Update

It is expected for the Click Import export tool to help maintain the GEHCContract collection. Below is an Excel template that for the object:



## GEHCSite Create or Update

It is expected for the Click Import export tool to help maintain the GEHCSite collection. Below is an Excel template that for the object:



## GEHCSystem Create or Update

It is expected for the Click Import export tool to help maintain the GCHCSystem collection. Below is an Excel template that for the object:



## Skill Create

It is expected for the Click Import export tool to help maintain the Skill collection. Below is an Excel template that for the object:



Resource skill details are maintained in Trace and are exported from there into specially formatted files for upload into ClickSchedule. As long as the correct resource identifier is supplied, i.e. the resources and their skills both use the same resource identifiers, these values can easily be combined with the other resource information within ClickSchedule in order to provide a complete picture of the resource.

Note: This is used to create new skills which are important for matching Engineers and Tasks to requirements.

## Task Create or Update

It is expected for the Click Import export tool to help maintain the Task collection. Below is an Excel template that for the object:



## GEHCContract Create or Update

It is expected for the Click Import export tool to help maintain the Task collection. Below is an Excel template that for the object:



# Inbound Interfaces

## Overview

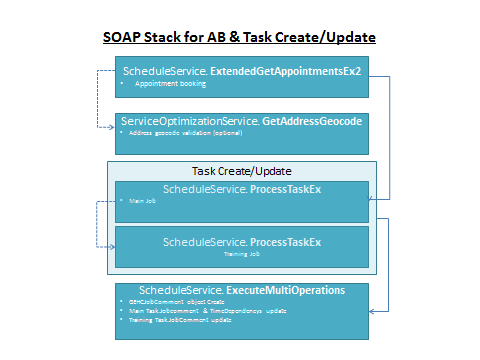
Messages will be passed back and forth between ClickSchedule to MUST via an HTTP POST messages. The message transmission is bi-directional and occurs in response to actions that are performed in either system.

The Click Platform IIS settings can support a variety of authentication mechanisms, the most common being Windows Authentication or Basic Authentication. On establishing an authentication standard the SOAP POST message will need to adhere to this agreed standard.

The SOAP actions listed this section below are more in-depth explanations of the messages required for the scenarios defined in, 2.6.

## Incoming Messages Interfaces

The following section defines the various Inbound Message operations. Each message definition is provided with the required message properties/tags, containing sample data values.

****

## Incoming Messages Summary

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Object** | **Function in ClickSchedule** | **Source** | **Method** | **Operation** | **Scenario** |
| Task | Create / Update | MUST | ScheduleService.svc | ProcessTaskEx | **Error! Reference source not found.** |
| Get Appointmentslots | MUST | ScheduleService.svc | ExtendedGetAppointmentsEx2 | 3.2.5 |
|  |
| GEHCSystem | Create / Update | Excel |  | CS Import tool | **Error! Reference source not found.** |
| GEHCContract | Create / Update | Excel |  |  | **Error! Reference source not found.** |
| GEHCSite | Create / Update | Excel |  |  | **Error! Reference source not found.** |
| Skills | Create | Excel |  |  | 2.7.2 |
| Engineer | Create / Update | Excel |  |  | 2.7.1 |

## Message Header

Every Web Service request can contain an OptionalParameters header. The parameters that can be included in this header are:

**Table 1 Optional Parameters and Headers**

|  |  |  |  |
| --- | --- | --- | --- |
| Parameter | Description | Mandatory | Required Value |
| CallerIdentity | To identify the source of an incoming message, an external system must specify the CallerIdentity attribute (i.e. the name of the system sending the message). The ClickSchedule Integration Manager uses the CallerIdentity attribute to prevent infinite loops of incoming and outgoing messages. For example, MUST and ClickSchedule can update a task’s status and both systems notify each other of a new task status. In this case, an infinite loop of updating messages might occur. To prevent this, the ClickSchedule Integration Manager blocks the outgoing message to the original source if the CallerIdentity making the update is the same as the Outgoing Message Destination.  Note: CallerIdentity should be explicitly consistent across the PI farm. | **Yes** | **MUST** |
| ExtraData | Additional information that can be interpreted by a custom Web Operation | **No** |  |
| ErrorOnNonExistingDictionaries | If true and the operation encounter an input dictionary item that does not exist in the dictionary, the operation reports an error and the call fails. If false (default), the operation adds the missing item to the dictionary.  Note: You need to add this tag to all related tasks in an MST chain. | **Yes** | **true** |

## GetAddressGeocode

To validate an address use the GetAddressGeocode service to ensure that longitudes and latitudes are accurately returned. To validate the address of a visit the ServiceOptimizationService.GetAddressGeocode webservice can be invoked to return longitudes and latitudes, further validation can be performed on these values form the MUST macro.

#### Request Header

|  |  |
| --- | --- |
| URL | SOAP Action |
| http://<server>/SO/IntegrationServices/ ServiceOptimizationService.svc /BasicHttpInt | http://www.clicksoftware.com/ ServiceOptimizationService.svc / GetAddressGeocode |

|  |
| --- |
| POST http://localhost/SO/IntegrationServices/ServiceOptimizationService.svc/BasicHttpInt HTTP/1.1  Content-Type: text/xml; charset=utf-8  VsDebuggerCausalityData: uIDPo9pJb1zm9v1PhRr1n7gAH6EAAAAATHw2igLDSkyMhsho/tJ6CtFSEeKw8SRGr6RXCHYXYZoACQAA  **SOAPAction: "http://www.clicksoftware.com/ServiceOptimizationService/GetAddressGeocode"**  Accept-Encoding: gzip, deflate,gzip, deflate,gzip, deflate  Authorization: Negotiate TlRMTVNTUAADAAAAAAAAAFgAAAAAAAAAWAAAAAAAAABYAAAAAAAAAFgAAAAAAAAAWAAAAAAAAABYAAAAFcKY4gYAchcAAAAP+PI0dxBtloEtBGNI74gliA==  Host: localhost  Content-Length: 787  Expect: 100-continue |

#### Request Message

|  |
| --- |
| <s:Envelope xmlns:s = "http://schemas.xmlsoap.org/soap/envelope/">  <s:Header>  <h:OptionalParameters  xmlns:h = "http://www.clicksoftware.com/OptionalParameters"  xmlns = "http://www.clicksoftware.com/OptionalParameters"  xmlns:xsi = "http://www.w3.org/2001/XMLSchema-instance"  xmlns:xsd = "http://www.w3.org/2001/XMLSchema">  <CallerIdentity>MUST</CallerIdentity>  <ErrorOnNonExistingDictionaries>false</ErrorOnNonExistingDictionaries>  </h:OptionalParameters>  </s:Header>  <s:Body xmlns:xsi = "http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd = "http://www.w3.org/2001/XMLSchema">  <GetAddressGeocode xmlns = "http://www.clicksoftware.com">  <Location>  <Street>39</Street>  <PostCode>TN255AB</PostCode>  <Country>UNITED KINGDOM</Country>  </Location>  <NumOfMatches>10</NumOfMatches>  </GetAddressGeocode>  </s:Body>  </s:Envelope> |

#### Response Message

|  |
| --- |
| <s:Envelope xmlns:s = "http://schemas.xmlsoap.org/soap/envelope/">  <s:Body>  <GetAddressGeocodeResponse xmlns = "http://www.clicksoftware.com">  <Result>  <NumOfMatchesFound>1</NumOfMatchesFound>  <Locations>  <Location>  <Street></Street>  <City></City>  <CitySubdivision></CitySubdivision>  <PostCode>TN255AB</PostCode>  <State></State>  <StateSubdivision></StateSubdivision>  <Latitude>51194210</Latitude>  <Longitude>918789</Longitude>  <Country>UNITED KINGDOM</Country>  <QualityValues>MEDIUM. Postcode Data, 26,;</QualityValues>  <GISDataSource>12</GISDataSource>  </Location>  </Locations>  </Result>  </GetAddressGeocodeResponse>  </s:Body>  </s:Envelope> |

## ExtendedGetAppointmentsEx2

This operation proposes appointment options for a Task. An appointment is a time slot for the Task to be scheduled within. The operation returns the permitted time slots for an appointment. It does not book the appointment in ClickSchedule or store any other information in the database.

Once GEHCs Click environment is mature sample messages will be created for this document.

#### Appointment Message Fields

ExtendedGetAppointmentsEx2 Elements:

| Element | Value | Comments |
| --- | --- | --- |
| Task | Task details | Contains the task object |
| SchedulePolicy | Name of the Policy | “Standard” |
| TimeIntervals | NA | Not required |
| Profile | The name of an appointment booking profile | Out of the list of AB Profiles in BSS:  a. ‘AM-PM’  b. ‘ONE HOUR  c. ‘TWO HOURS’ |
| Period | <Start>[date/time]</Start>  <Finish>[date/time]</Finish> | Contains the customer preference <Start> and <Finish> date and time search period for the appointment booking. Assumed to be up to 30 days duration in total. |
| ExcludeCurrentAppointment | Rebooking = true;  New appt = false | If "true" and the task already has an appointment slot, the server omits the existing appointment from the response |
| TimeOut | 60 | The maximum time, in seconds, that the appointment-booking service should search for appointments. The default is 50 seconds; it is recommended not exceed this time. And recommended to specify it. |
| GradeAppointments | True | GEHC will try grading |
| RelatedTasks | Details of dependent Tasks if they exist | For Training tasks it is expected that MUST will send Start to finish time and same engineer relations between the primary and the training tasks to follow.  Please note that related Tasks must have the same CallID + Different Number as the primary Task.  In addition the time dependency has to have a Critical Tag, and the Engineer dependency doesn’t. See example below. |
| SameSlotTasks | False | Make the resources be at the same place at the same time. |
| UnScheduleRelatedTasks | True | Always needs to be set as true. |
| ErrorOnNonExistingDictionaries | True | Always need to be sent as true. |

**Appointment for a New Task (MUST to SO)**

If the appointment request is for a new Task, the CallID and Number must be unique and cannot be of a Task that exists in the system. If the CallID specified exists in the system, appointments will be returned based on the details of the existing task.

**Appointment for an Existing Task (MUST to SO)**

If the Task exists and a new appointment is being requested, then the CallID and Number should be set to that of the Task for which appointments are requested. When rebooking, the existing appointment slot would be excluded due to the flag ExcludeCurrentAppointment that is set to true.

Note:

Rebooking does not require all properties to be sent again. If any property in the Task is also updated, then the changed properties should be sent over upon AB request. The change will temporarily be done during the AB process and needs to be sent again upon Task Update (booking).

#### Required Task fields for Appointment booking

For GEHC to return the best quality time slots Task fields that affect the schedule should be included, below is a list of the fields which directly affect the scheduling policy. GEHC should collate and pass these fields where possible to ExtendedGetAppointmentsEx2:

| ClickSchedule Property | Comments |
| --- | --- |
| CallID | CallID = WO/Service Order Number |
| Number | Number = Operation Number |
| TaskID | GEHC TaskID, needs to be unique |
| MUSTJobNumber | The MUSTJobNumber |
| TaskType | The tasktype will derive the task duration and priority and skills. |
| EarlyStart | The earliest start time for the task. |
| DueDate | The date within which the task should get completed |
| Region | The lowest level of the navigation tree. |
| Postcode | The postcode for the task location. |
| CountryID | Should be "UNITED KINGDOM" |
| DueDate | When the Task is Due |
| Duration | How long the task should take |
| EngineerDependencies | Related tasks |
| EngineerRequirements | Related Engineers |
| EngineerType | What kind of Engineer is required |
| ExcludedEngineers | Which Engineers should be excluded |
| ExcludedFSEs | List of Engineers to be excluded |
| Languages |  |
| LateStart |  |
| Latitude | Location: If both Latitude & Longitude are set to zero “0”, the location will be derived |
| Longitude | Location: If both Latitude & Longitude are set to zero “0”, the location will be derived  Its assumed that an engineer will never be required off the east cost of Africa. |
| RequiredEngineers | Which Engineers are required |
| RequiredFSEs | List of required Engineers |
| RequiredSkills1 | Skill information |
| SkillLevel | Skill information |
| TaskLanguages | List of Languages |
| TaskType | Task Type |
| TimeDependencies | Time Dependencies for MSTs |

**Note**: The sample in section 3.4.4 shows only the mandatory fields for this message. All available task properties should be included, as long as the mandatory fields in the table above are included.

Rebooking ExtendedGetAppointmentsEx2 Mandatory Task properties for a new appointment:

| ClickSchedule Property | Comments |
| --- | --- |
| CallID | CallID = WO/Service Order Number |
| Number | Number = Operation Number |

#### Related Tasks

Tasks that require training provision will need AB related Task information to be sent along with the primary task in the ExtendedGetAppointmentsEx2 message. Tasks requiring Related tasks are MST’s a list of these are defined here: 3.5.3

The Time dependencies need to be included with the tasks as defined in this document per the type of message being created. For example, Training jobs would have the Training job as a related task to the main task & the Main Task would have the dependency in place.

#### Rebooking (MUST to SO)

GE can manage rebooking’s. The following processes should be used depending on requirements:

1. An actual rebooking using the AB & ProcessTaskEx;
2. Cancelation of an existing Task and creation of a new task.

#### Request Header

ExtendedGetAppointmentsEx2 URL and SOAP Action:

|  |  |
| --- | --- |
| URL | SOAP Action |
| http://<server>/SO/IntegrationServices/ScheduleService.svc/BasicHttpInt | <http://www.clicksoftware.com/ScheduleService/ExtendedGetAppointmentsEx2> |

Below is an example SOAP Request Header POST & SOAPAction are the vital in the packet construct:

|  |
| --- |
| **POST http://localhost/SO/IntegrationServices/ScheduleService.svc/BasicHttpInt HTTP/1.1**  **Content-Type: text/xml; charset=utf-8**  **VsDebuggerCausalityData: uIDPoyZwRBA4o+ZIjUJgVKwnisgAAAAAAti4wLNcdU2GFmZTltP1jDZd07W5LWhNkQM7RvKIBe8ACQAA**  **SOAPAction: "http://www.clicksoftware.com/ScheduleService/ExtendedGetAppointmentsEx2"**  **Host: localhost**  **Content-Length: 8246**  **Expect: 100-continue**  **Accept-Encoding: gzip, deflate** |

#### Request Message

The following is a sample SOAP message for an AB three man job. As per the scenario spesifications Dependencies are always added to the Main task (see bold text), MST’s are added to the RelatedTasks node.

|  |
| --- |
| <s:Envelope xmlns:s = "http://schemas.xmlsoap.org/soap/envelope/">  <s:Header>  <h:OptionalParameters  xmlns:h = "http://www.clicksoftware.com/OptionalParameters"  xmlns = "http://www.clicksoftware.com/OptionalParameters"  xmlns:xsi = "http://www.w3.org/2001/XMLSchema-instance"  xmlns:xsd = "http://www.w3.org/2001/XMLSchema">  <CallerIdentity>MUST</CallerIdentity>  <ErrorOnNonExistingDictionaries>false</ErrorOnNonExistingDictionaries>  </h:OptionalParameters>  </s:Header>  <s:Body xmlns:xsi = "http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd = "http://www.w3.org/2001/XMLSchema">  <ExtendedGetAppointmentsEx2 xmlns = "http://www.clicksoftware.com">  <Task>  <CallID>GB-A1234579-1</CallID>  <Number>1</Number>  <EarlyStart>2013-09-25T04:13:25.9936565-04:00</EarlyStart>  <DueDate>2013-10-09T04:13:25.9936565-04:00</DueDate>  <LateStart>2013-10-09T03:13:25.9936565-04:00</LateStart>  <Priority>10</Priority>  <Customer>Cristina Hardie</Customer>  <District>  <Name>UK South</Name>  </District>  <Postcode>TN25 5AB</Postcode>  <OpenDate>2013-09-25T04:13:25.9936565-04:00</OpenDate>  <TaskType>  <Name>Installation</Name>  </TaskType>  <Duration>900</Duration>  <NumberOfRequiredEngineers>1</NumberOfRequiredEngineers>  **<TimeDependencies>**  **<TaskTimeDependency>**  **<TaskKey>**  **<CallID>GB-A1234579-1</CallID>**  **<Number>2</Number>**  **</TaskKey>**  **<RelationType>1</RelationType>**  **<RelationOperator>2</RelationOperator>**  **<UpperBound>0</UpperBound>**  **<LowerBound>0</LowerBound>**  **<Critical>true</Critical>**  **</TaskTimeDependency>**  **<TaskTimeDependency>**  **<TaskKey>**  **<CallID>GB-A1234579-1</CallID>**  **<Number>3</Number>**  **</TaskKey>**  **<RelationType>1</RelationType>**  **<RelationOperator>2</RelationOperator>**  **<UpperBound>0</UpperBound>**  **<LowerBound>0</LowerBound>**  **<Critical>true</Critical>**  **</TaskTimeDependency>**  **</TimeDependencies>**  **<EngineerDependencies>**  **<TaskEngineerDependency>**  **<TaskKey>**  **<CallID>GB-A1234579-1</CallID>**  **<Number>2</Number>**  **</TaskKey>**  **<RelationType>2</RelationType>**  **</TaskEngineerDependency>**  **<TaskEngineerDependency>**  **<TaskKey>**  **<CallID>GB-A1234579-1</CallID>**  **<Number>3</Number>**  **</TaskKey>**  **<RelationType>2</RelationType>**  **</TaskEngineerDependency>**  **</EngineerDependencies>**  <ContactPhoneNumber>+447720449438</ContactPhoneNumber>  <Latitude>51194268</Latitude>  <Longitude>918754</Longitude>  <Street>Kent</Street>  <City>Ashford</City>  <CountryID>  <Name>UNITED KINGDOM</Name>  </CountryID>  <CustomerEmail>dhardie@gmail.com</CustomerEmail>  <TaskSubType>  <Name>MRI</Name>  </TaskSubType>  <TaskID>FR-A1234579-1\_D4\_20130925T041325</TaskID>  <MUSTJobNumber>A1234579</MUSTJobNumber>  <SystemID>  <ID>SystemID001</ID>  </SystemID>  <IsSafety>false</IsSafety>  <CustomerExpectation>The Customer Expectation</CustomerExpectation>  <UseDistrictCalendar>false</UseDistrictCalendar>  <ContractOfferingFamily>Gold</ContractOfferingFamily>  <PreferredFSEs>FHollande;NSarkozy</PreferredFSEs>  <ExcludedFSEs>JChirac;FMitterrand;Vd’Estaing</ExcludedFSEs>  <RequiredFSEs>FHollande</RequiredFSEs>  <SkillLevel>1</SkillLevel>  <OwnerName>Charles de Gaulle</OwnerName>  <OwnerSSO>Charles de Gaulle</OwnerSSO>  <CRMSystemName>  <Name>CRMSystem001</Name>  </CRMSystemName>  <PartDeliveryType>In the back of a van</PartDeliveryType>  <PartComment>Bend your knees when you pick it up, its heavy!</PartComment>  <PartEstimatedDeliveryDate>2013-09-25T04:13:25.9936565-04:00</PartEstimatedDeliveryDate>  <ReactivationDate>2013-09-25T04:13:25.9936565-04:00</ReactivationDate>  <MacroVersion>MacroVersion001</MacroVersion>  <TaskLanguages>English;French</TaskLanguages>  </Task>  <SchedulePolicy>Standard</SchedulePolicy>  <Profile>TWO HOURS</Profile>  <Period>  <Start>2013-09-26T04:13:26.1986375-04:00</Start>  <Finish>2013-10-09T04:13:26.1976614-04:00</Finish>  </Period>  <ExcludeCurrentAppointment>false</ExcludeCurrentAppointment>  <TimeOut>60</TimeOut>  <GradeAppointments>false</GradeAppointments>  <RelatedTasks>  <Task>  <CallID>GB-A1234579-1</CallID>  <Number>2</Number>  <EarlyStart>2013-09-25T04:13:25.9936565-04:00</EarlyStart>  <DueDate>2013-10-09T04:13:25.9936565-04:00</DueDate>  <LateStart>2013-10-09T03:13:25.9936565-04:00</LateStart>  <Priority>10</Priority>  <Customer>Cristina Hardie</Customer>  <District>  <Name>UK South</Name>  </District>  <Postcode>TN25 5AB</Postcode>  <OpenDate>2013-09-25T04:13:25.9936565-04:00</OpenDate>  <TaskType>  <Name>Installation</Name>  </TaskType>  <Duration>900</Duration>  <NumberOfRequiredEngineers>1</NumberOfRequiredEngineers>  <ContactPhoneNumber>+447720449438</ContactPhoneNumber>  <Latitude>51194268</Latitude>  <Longitude>918754</Longitude>  <Street>Kent</Street>  <City>Ashford</City>  <CountryID>  <Name>UNITED KINGDOM</Name>  </CountryID>  <CustomerEmail>dhardie@gmail.com</CustomerEmail>  <TaskSubType>  <Name>MRI</Name>  </TaskSubType>  <TaskID>FR-A1234579-1\_D4\_20130925T041425</TaskID>  <MUSTJobNumber>A1234579</MUSTJobNumber>  <SystemID>  <ID>SystemID001</ID>  </SystemID>  <IsSafety>false</IsSafety>  <CustomerExpectation>The Customer Expectation</CustomerExpectation>  <UseDistrictCalendar>false</UseDistrictCalendar>  <ContractOfferingFamily>Gold</ContractOfferingFamily>  <PreferredFSEs>FHollande;Nicolas Sarkozy</PreferredFSEs>  <ExcludedFSEs>JChirac;FMitterrand;Vd’Estaing</ExcludedFSEs>  <RequiredFSEs>FHollande</RequiredFSEs>  <SkillLevel>1</SkillLevel>  <OwnerName>Charles de Gaulle</OwnerName>  <OwnerSSO>Charles de Gaulle</OwnerSSO>  <CRMSystemName>  <Name>CRMSystem001</Name>  </CRMSystemName>  <PartDeliveryType>In the back of a van</PartDeliveryType>  <PartComment>Bend your knees when you pick it up, its heavy!</PartComment>  <PartEstimatedDeliveryDate>2013-09-25T04:13:25.9936565-04:00</PartEstimatedDeliveryDate>  <ReactivationDate>2013-09-25T04:13:25.9936565-04:00</ReactivationDate>  <MacroVersion>MacroVersion001</MacroVersion>  <TaskLanguages>English;French</TaskLanguages>  </Task>  <Task>  <CallID>GB-A1234579-1</CallID>  <Number>3</Number>  <EarlyStart>2013-09-25T04:13:25.9936565-04:00</EarlyStart>  <DueDate>2013-10-09T04:13:25.9936565-04:00</DueDate>  <LateStart>2013-10-09T03:13:25.9936565-04:00</LateStart>  <Priority>10</Priority>  <Customer>Cristina Hardie</Customer>  <District>  <Name>UK South</Name>  </District>  <Postcode>TN25 5AB</Postcode>  <OpenDate>2013-09-25T04:13:25.9936565-04:00</OpenDate>  <TaskType>  <Name>Installation</Name>  </TaskType>  <Duration>900</Duration>  <NumberOfRequiredEngineers>1</NumberOfRequiredEngineers>  <ContactPhoneNumber>+447720449438</ContactPhoneNumber>  <Latitude>51194268</Latitude>  <Longitude>918754</Longitude>  <Street>Kent</Street>  <City>Ashford</City>  <CountryID>  <Name>UNITED KINGDOM</Name>  </CountryID>  <CustomerEmail>dhardie@gmail.com</CustomerEmail>  <TaskSubType>  <Name>MRI</Name>  </TaskSubType>  <TaskID>FR-A1234579-1\_D4\_20130925T041525</TaskID>  <MUSTJobNumber>A1234579</MUSTJobNumber>  <SystemID>  <ID>SystemID001</ID>  </SystemID>  <IsSafety>false</IsSafety>  <CustomerExpectation>The Customer Expectation</CustomerExpectation>  <UseDistrictCalendar>false</UseDistrictCalendar>  <ContractOfferingFamily>Gold</ContractOfferingFamily>  <PreferredFSEs>FHollande;Nicolas Sarkozy</PreferredFSEs>  <ExcludedFSEs>JChirac;FMitterrand;Vd’Estaing</ExcludedFSEs>  <RequiredFSEs>FHollande</RequiredFSEs>  <SkillLevel>1</SkillLevel>  <OwnerName>Charles de Gaulle</OwnerName>  <OwnerSSO>Charles de Gaulle</OwnerSSO>  <CRMSystemName>  <Name>CRMSystem001</Name>  </CRMSystemName>  <PartDeliveryType>In the back of a van</PartDeliveryType>  <PartComment>Bend your knees when you pick it up, its heavy!</PartComment>  <PartEstimatedDeliveryDate>2013-09-25T04:13:25.9936565-04:00</PartEstimatedDeliveryDate>  <ReactivationDate>2013-09-25T04:13:25.9936565-04:00</ReactivationDate>  <MacroVersion>MacroVersion001</MacroVersion>  <TaskLanguages>English;French</TaskLanguages>  </Task>  </RelatedTasks>  <UnScheduleRelatedTasks>false</UnScheduleRelatedTasks>  <SuggestCandidateResources>false</SuggestCandidateResources>  <ParallelFactor>0</ParallelFactor>  <UseSLRCache>false</UseSLRCache>  <UsePartitionControl>false</UsePartitionControl>  </ExtendedGetAppointmentsEx2>  </s:Body>  </s:Envelope> |

Note: The related Tasks and the Primary should have the same CallID (same job number) and a different Number (different Task Number). This setting will be considered in ClickSchedule as dependent Tasks.

#### Response Message

The following responses indicate that the appointment request was processed. The server will return all valid appointment slots found. There are therefore two possible responses – either the server finds appointments, or finds no valid appointments.

Appointment with no slot:

|  |
| --- |
| <ExtendedGetAppointmentsEx2Response xmlns = "http://www.clicksoftware.com">  <OptionalAppointments/>  </ExtendedGetAppointmentsEx2Response> |

Appointments with slots:

|  |
| --- |
| <s:Envelope xmlns:s = "http://schemas.xmlsoap.org/soap/envelope/">  <s:Body>  <ExtendedGetAppointmentsEx2Response xmlns = "http://www.clicksoftware.com">  <OptionalAppointments>  <TimeInterval>  <Start>2013-09-26T08:00:00</Start>  <Finish>2013-09-26T18:00:00</Finish>  <Grade>0</Grade>  </TimeInterval>  <TimeInterval>  <Start>2013-09-27T08:00:00</Start>  <Finish>2013-09-27T18:00:00</Finish>  <Grade>0</Grade>  </TimeInterval>  <TimeInterval>  <Start>2013-09-30T08:00:00</Start>  <Finish>2013-09-30T18:00:00</Finish>  <Grade>0</Grade>  </TimeInterval>  <TimeInterval>  <Start>2013-10-01T08:00:00</Start>  <Finish>2013-10-01T18:00:00</Finish>  <Grade>0</Grade>  </TimeInterval>  <TimeInterval>  <Start>2013-10-02T08:00:00</Start>  <Finish>2013-10-02T18:00:00</Finish>  <Grade>0</Grade>  </TimeInterval>  <TimeInterval>  <Start>2013-10-03T08:00:00</Start>  <Finish>2013-10-03T18:00:00</Finish>  <Grade>0</Grade>  </TimeInterval>  <TimeInterval>  <Start>2013-10-04T08:00:00</Start>  <Finish>2013-10-04T18:00:00</Finish>  <Grade>0</Grade>  </TimeInterval>  <TimeInterval>  <Start>2013-10-07T08:00:00</Start>  <Finish>2013-10-07T18:00:00</Finish>  <Grade>0</Grade>  </TimeInterval>  <TimeInterval>  <Start>2013-10-08T08:00:00</Start>  <Finish>2013-10-08T18:00:00</Finish>  <Grade>0</Grade>  </TimeInterval>  </OptionalAppointments>  </ExtendedGetAppointmentsEx2Response>  </s:Body>  </s:Envelope> |

## ProcessTaskEx

This operation is used to process a task in the ClickSchedule repository. This operation can handle both the creation and update of tasks in SO. The sender doesn’t need to know if the task already exists in the CKSW repository or not. In case the task exists (identified by its object reference) SO will update it, otherwise it will create a new task.

This web operation processes a task object in the following ways:

* The operation adds a new task to the database, or it updates an existing task;
* It can run the scheduling-workflow logic service to assign the task;
* It can run the schedule-update logic service if particular task properties have been updated;
* It can run different scheduling-workflow or schedule-update configurations for different task groups;
* It can send multistage task directives in a single web operation call;
* It can block invalid or undesired task status updates;

When creating a new task the message should include all mandatory task properties as described in the table below.

#### New Task Message Fields

Following is the task information that shall be passed through the integration and the corresponding fields on the MUST side.

| Property | Visible Name | | Description | Type | Access | Owner | Mandatory? | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **General** | | | | | | | | |
| CallID | Call ID | | These values form the unique identifier for the task in ClickSchedule. The CallID is defaulted to the MUST Job Number, while the Number is defaulted to 1, but can be different (e.g. when creating dependencies between “primary” and associated “training” or pickup tasks). The TaskID is created initially by MUST and subsequently managed and synchronized between both systems in order to define a value that can be consistently derived over time by MUST in order to identify a unique visit. | String (64) | Editable | GEHC | Yes | |
| Number | Number | | Number | Editable | GEHC |
| TaskID | Task ID | | String (64) | Non-Editable | GEHC |
| MUSTJobNumber | MUST Job Number | | The ID that uniquely identifies the associated job in MUST. | String (64) | Non-Editable | GEHC | Yes | |
| SystemID | System ID | | The DisplayID of the related System. This is derived from the task’s associated System.  References the system against which the task is to be carried out. | Reference to System | Non- Editable | GEHC | Yes | |
| CRMSystemName | CRM System Name | | The name of the CRM system that created the task. Not currently used. | Dictionary (CRMSystem) | Non-Editable | GEHC | Yes | |
| IsSafety | Safety? | | Identifies whether the task is a safety activity. Defaults to No. | Yes/No (Boolean) | Non-Editable | GEHC | No | |
| Priority | Priority | | The priority (importance) of the task. The higher the number the higher the priority.  This is a computed priority value provided by GEHC. | Number | Non-Editable | GEHC | Yes | |
| TaskType | Job Type | | The type of job the task is a part of, or the type of activity the task represents. | Dictionary (TaskType) | Non-Editable | GEHC | Yes | |
| TaskSubType | Job Sub-Type | | The sub-type of the task. | Dictionary (TaskSubType) | Non-Editable | GEHC | Yes | |
| Contract | Contract | | The currently active contract for the target system. Derived from the Open Date and the Contract instances associated with the task’s System. | Reference to Contract | Non-Editable | CS | Yes | |
| CustomerExpectation | Customer Expectation | | The customer’s expectations. For display purposes only. | String (256) | Non-Editable | GEHC | No | |
| Status | Status | | Current status of the task. | Dictionary (TaskStatus) | Editable | CS | Yes | |
| InJeopardy | In Jeopardy | | This flag is automatically set by ClickSchedule when the task goes into jeopardy. The Dispatcher is responsible for clearing the jeopardy. | Yes/No (Boolean) | Editable | CS | No | |
| JeopardyState | Jeopardy State | | Indicates the type of jeopardy situation the task is in, e.g. not scheduled and approaching due date, late to travel etc. | Dictionary (JeopardyState) | Non-Editable | CS | No | |
| **Time** | | | | | | | | |
| OpenDate | Open Date | | The date on which the task was initially logged. If customer-initiated, this is the date and time at which the customer made the initial call. | Date and Time | Non- Editable | GEHC | Yes | |
| EarlyStartOffset | Early Start Offset | | This is an interval, in seconds, after the Open Date after which the task can be booked. May be derived from the associated Contract. | Duration | Non-Editable | GEHC/CS | No | |
| LateStartOffset | Late Start Offset | | This is an interval, in seconds, after the Open Date before which the task can be booked. May be derived from the associated Contract. | Duration | Non-Editable | GEHC/CS | No | |
| EarlyStart | Early Start | | The earliest time the work on the task should ideally start. This is used as an objective rather than applied as a rule and may be derived from the task’s early start offset details. | Date and Time | Non-Editable | GEHC | Yes | |
| Duration | Duration | | The estimated duration for performing the work (excluding any necessary travel). | Duration | Non-Editable | GEHC | Yes | |
| LateStart | Late Start | | The latest time that work on the task must start in order to meet the SLA. This may be derived from the task’s late start offset details. | Date and Time | Non-Editable | GEHC | Yes | |
| DueDate | Due Date | | The latest time on which the work should finish. This is derived from the LateStart and Duration. | Date and Time | Hidden | CS | Yes | |
| ReactivationDate | Reactivation Date | | The time at which the task is set to reactivate. | Date and Time | Hidden | GEHC | No | |
| AppointmentStart | Earliest Arrival On Site | | The start time of the appointment window provided to the customer. | Date and Time | Non-Editable | GEHC | No | |
| Appointmen Finish | Latest Arrival On Site | | The end time of the appointment window provided to the customer. | Date and Time | Non-Editable | GEHC | No | |
| UseDistrictCalendar | Early/Late Used District Calendar | | Indicates whether the Early and Late Start derivations are expected to use the district’s calendar. When omitted or set false the early/late start values are simple offsets from “now” and do not take non-working time into consideration. | Yes/No (Boolean) | Non-Editable | GEHC | No | |
| **Customer** | | | | | | | | |
| ContactName | Contact‌ Name | | The name of the customer’s contact. | String (128) | Non-Editable | GEHC | No | |
| ContactPhone‌Number | Contact‌ Phone‌ Number | | The phone number for the customer’s contact. | String (64) | Non- Editable | GEHC | No | |
| Customer‌Email | Contact E-mail Address | | The customer contact e-mail address. | String (128) | Non- Editable | GEHC | No | |
| ContractOfferingFamily | Contract Offering Family | | If not provided at the task creation, the offering family is derived from the task’s Contract details. | String (64) | Non-Editable | GEHC | No | |
| TaskLanguages | Languages | | A sequence of semi-colon separated language names. | String (1024) | Hidden | GEHC | No | |
| Languages | Languages | | The language(s) accepted at the customer site for this task. This is derived from the TaskLanguages or, when that is unspecified, the task’s Site details. | Dictionary (Language, Multi-value) | Non-Editable | CS | No | |
| **Location** | | | | | | | | |
| Region | Organization Country | | The first level in the hierarchy. Derived automatically from the District. | Dictionary (Region) | Non- Editable | CS | | Yes |
| District | Organization Zone | | The second level in the hierarchy. | Dictionary (District) | Non- Editable | GEHC | | Yes |
| SiteID | Site | | The DisplayID of the related Site. This is derived from the task’s associated Site. | Reference to Site | Non- Editable | CS | | Yes |
| IsMobileSite | Is Mobile System? | | Indicates whether the system is part of a mobile site. | Yes/No (Boolean) | Non-Editable | GEHC | | Yes |
| Street1 | Street | | The work site address.  When not specified this is derived from the task’s System’s Site address. | String (64) | Non- Editable | GEHC/ CS | | No |
| City | City | | String (64) | Non-Editable | GEHC/ CS | |
| State | County | | String (64) | Non-Editable | GEHC/ CS | |
| Postcode | Postcode | | String (64) | Non- Editable | GEHC/ CS | |
| CountryID | Country | | The country in which the address exists. | Dictionary (Country) | Non-Editable | GEHC/ CS | |
| Latitude | Latitude | | The geo-coded location for the task. These are derived on creation or update of the task’s derived (or specified) address. | Number | Hidden | CS | | No |
| Longitude | Longitude | | Number | Hidden | CS | | No |
| **Resource** | | | | | | | | |
| PreferredFSEs | Preferred FSEs | | A sequence of semi-colon separated IDs used to identify the preferred resources. | String (1024) | Hidden | GEHC | No | |
| PreferredEngineers | Preferred Resources | | The resources to one of which it is preferred to schedule the task if possible/sensible. Derived from the PreferredFSEs. | Reference to Engineer  (Multi-value) | Non-Editable | CS | No | |
| ExcludedFSEs | Excluded FSEs | | A sequence of semi-colon separated IDs used to identify the excluded resources. | String (1024) | Hidden | GEHC | No | |
| ExcludedEngineers | Excluded Resources | | The optional set of resources to whom the task cannot be automatically scheduled. This is typically used after a given resource rejects a task on some grounds in order to prevent the task being re-assigned to them. Initially derived from ExcludedFSEs and the associated System’s excluded engineers. | Reference to Engineer  (Multi-value) | Editable | CS | No | |
| RequiredFSEs | Required FSEs | | A sequence of semi-colon separated IDs used to identify the required resources. | String (1024) | Hidden | GEHC | No | |
| RequiredEngineers | Required Resources | | The set of resources from which the scheduler may select a single resource to schedule the task to. Initially derived from RequiredFSEs. | Reference to Engineer  (Multi-value) | Editable | GEHC/ CS | No | |
| **Requirements** | | | | | | | | |
| SkillLevel | Level | | The skill level associated with the task. | Number | Hidden | GEHC | Yes | |
| RequiredSkills1 | Required Skills | | The skills that a resource must have to be able to perform the work. This is automatically derived from the task’s associated system’s product ID and the task’s skill level. | List of Skills  (Multi-value) | Non - Editable | CS | No | |
| **Dependencies** | | | | | | | | |
| TimeDependancies | | Time Dependencies | Defines time relationships between tasks that are part of a multi-staged chain. | List of time dependencies  (Multi-value) | Non - Editable | GEHC | No | |
| EngineerDependancies | | Resource Dependencies | Defines Field Engineer relationships between tasks that are part of a multi-staged tasks chain. | List of Field Engineer dependencies  (Multi-value) | Non - Editable | GEHC | No | |
| Critical | | Critical | Defines if this task cannot be scheduled, others in the MST chain will also not be scheduled. | Boolean | Non - Editable | GEHC | No | |
| **Assignment** (this tab is only relevant for scheduled tasks) | | | | | | | | |
| Start | Assignment Start | | The time at which the resource is scheduled to start work on site. | Date and Time | Editable | CS | Yes | |
| Finish | Assignment Finish | | Time at which the resource is scheduled to finish work on site. | Date and Time | Editable | CS | Yes | |
| Engineers | Resource | | The (single) resource scheduled to perform the task. | Reference to Engineer  (Multi-value) | Non-Editable | CS | Yes | |
| **Details** | | | | | | | | |
| OwnerName | Owner Name | | The name of the Dispatcher who created the task via the MUST integration. | String (64) | Non-Editable | GEHC | No | |
| OwnerSSO | Owner SSO | | The SSO identity for the Dispatcher who created the task. | String (64) | Non-Editable | GEHC | Yes[[6]](#footnote-6) | |
| CTCDebrief | CTC Debrief | | The CTC debrief. | String (4000) | Non- Editable | GEHC | Yes | |
| Notes | Task Note | | A note combining the mobile and part comment. Where required, the part pickup location may be included here (when the part is to be obtained from a location related to the site address and when there is no separate part pickup task). | String (4000) | Non-Editable | GEHC | No | |
| JobComments | Job Comments | | The full job comments. | Linked GEHCJobComments | Non-Editable | GEHC | No | |
| FMIDueDate | FMI Due Date | | The date at which the FMI is due. Only relevant for FMI type jobs. For display purposes only. | Date and Time | Non-Editable | GEHC | No | |
| FMINumber | FMI Number | | The FMI number. Only relevant for FMI type jobs. | String (64) | Non-Editable | GEHC | No | |
| FMIRecall | FMI Recall? | | Indicates whether the FMI is a recall. Only relevant for FMI type jobs. | Yes/No (Boolean) | Non-Editable | GEHC | No | |
| PMSchedule | PM Schedule | | Identifies the PM schedule code. Only relevant for PM type jobs. | String (2) | Non-Editable | GEHC | No | |
| MacroVersion | Macro Version | | An identifier for the version of the macro through which the task instance was created. | String (64) | Hidden | GEHC | No | |
| SuperPowerTaskFlag | Able to Unschedule Other Task? | | Controls whether a task can unschedule other scheduled tasks on the Gantt. Note that safety and other super power tasks cannot be unscheduled by one of these tasks.  Default is “No”. | Yes/No (Boolean) | Non-Editable | GEHC | No | |
| IsMDT | Is MDT? | | Automatically derived based on the task’s duration, this is set true when the task is deemed an MDT candidate. | Yes/No (Boolean) | Non-Editable | CS | No | |
| TrainingJobNumber | Training Job Number | | The associated on-the-job training task MUST job number. | String (64) | Non-Editable | GEHC | No | |
| TraineeFSEs | Trainee FSEs | | A sequence of semi-colon separated IDs used to identify the associated on-the-job training resources. | String (1024) | Hidden | GEHC | No | |
| **Parts** | | | | | | | | |
| PartEstimatedDeliveryDate | Estimated Delivery Date | | The date at which the parts should have been delivered. When specified this limits the earliest time at which the task can be automatically scheduled. | Date and Time | Non-Editable | GEHC | No | |
| PartDeliveryType | Delivery Type | | The method of delivery. | String (64) | Non-Editable | GEHC | No | |
| **Closure** (shown when the task can be or is cancelled, rejected, completed or incompleted)  The tab will be shown in the following statuses: Tentative, Rejected, Rejected By FSE, New, Cancelled, Acknowledged, Completed, Incompleted) | | | | | | | | |
| IncompleteReason | Incomplete Reason | | The reason stated when a resource closes the task as incomplete. | Dictionary (IncompletionReason) | Editable when status is Incomplete | CS/CM | Yes when the Status is Incomplete | |
| IncompletionComment | Incomplete Comment | | A free text comment accompanying the incompletion reason. | String (256) | Editable when Status is incomplete | CS/CM |
| CancellationReason | Cancellation Reason | | The reason stated when the task is cancelled. | Dictionary (CancellationReason) | Editable when status is cancelled | CS | Yes when the Status is Cancelled | |

Note: Longitude & latitude values can be passed should the within the ProcessTaskEx message should the MUST macro deem the values accurate, if noth are set to zero then the location will be derived.

#### Updating existing Tasks

For tasks that have been commited to Click and need updating only the fields listed below & those fields that require alterdation should be listed in the Task element.

| ClickSchedule Property | Comments |
| --- | --- |
| CallID | CallID = WO/Service Order Number |
| Number | Number = Operation Number |

#### Single Task Request Message

My way of example for the Business scenarios 2.6.2 the SOAP stack for creating a single job follows the following pattern , this is also described here 2.6.2.1 :

* ProcessTaskEx – Main Job
* ExecuteMultipleOperations, see 0
  + Operation 1:
    - CreateOrUpdate ExecuteMultipleOperations;
  + Operation 2:
    - Update Main Tasks JobComments to new GEHCJobComments;

#### Request Header

The HTTP POST header directs the message to the correct location and activates the correct Web Service Action. Embedded in the header are other relevant values such as the authentication credentials relevant to the implementation of the integration. Below are the targets for ProcessTaskEx:

|  |  |
| --- | --- |
| URL | SOAP Action |
| http://<server>/SO/IntegrationServices/ ScheduleService.svc/BasicHttpInt | http://www.clicksoftware.com/ScheduleService.svc/ProcessTaskEx |

Below is an example SOAP Request Header POST & SOAPAction are the vital in the packet construct:

|  |
| --- |
| **POST http://localhost/SO/IntegrationServices/ScheduleService.svc/BasicHttpInt HTTP/1.1**  Content-Type: text/xml; charset=utf-8  VsDebuggerCausalityData: uIDPo9GM4VxxH2RNtGkSAw0s8DcAAAAASUK1UaLnjEeXDvVNqe55pZ9bprkShflAo6BQrVJhLuYACQAA  **SOAPAction: "http://www.clicksoftware.com/ScheduleService/ProcessTaskEx"**  Accept-Encoding: gzip, deflate,gzip, deflate,gzip, deflate  **Authorization: Negotiate TlRMTVNTUAADAAAAAAAAAFgAAAAAAAAAWAAAAAAAAABYAAAAAAAAAFgAAAAAAAAAWAAAAAAAAABYAAAAFcKY4gYAchcAAAAPEDE6SVReDscg3gnz8cC+Qg==**  Host: localhost  Content-Length: 3043  Expect: 100-continue |

#### Request Message

The following SOAP example will create a task in ClickSchedule, note the Envelope, Header and Body, name spaces are important.

SOAP Request Envelope:

|  |
| --- |
| <s:Envelope xmlns:s = "http://schemas.xmlsoap.org/soap/envelope/">  <s:Header>  <h:OptionalParameters  xmlns:h = "http://www.clicksoftware.com/OptionalParameters"  xmlns = "http://www.clicksoftware.com/OptionalParameters"  xmlns:xsi = "http://www.w3.org/2001/XMLSchema-instance"  xmlns:xsd = "http://www.w3.org/2001/XMLSchema">  <CallerIdentity>MUST</CallerIdentity>  <ErrorOnNonExistingDictionaries>false</ErrorOnNonExistingDictionaries>  </h:OptionalParameters>  </s:Header>  <s:Body xmlns:xsi = "http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd = "http://www.w3.org/2001/XMLSchema">  <ProcessTaskEx xmlns = "http://www.clicksoftware.com">  <Task>  <CallID>GB-A1234567-1</CallID>  <Number>1</Number>  <DueDate>2013-11-07T06:35:33.3954348-05:00</DueDate>  <Priority>10</Priority>  <Customer>Cristina Hardie</Customer>  <District>  <Name>UK South</Name>  </District>  <Postcode>TN25 5AB</Postcode>  <OpenDate>2013-10-24T06:35:33.3954348-04:00</OpenDate>  <TaskType>  <Name>Installation</Name>  </TaskType>  <Duration>900</Duration>  <NumberOfRequiredEngineers>1</NumberOfRequiredEngineers>  <ContactPhoneNumber>+447720449438</ContactPhoneNumber>  <Latitude>51194268</Latitude>  <Longitude>918754</Longitude>  <Street>Kent</Street>  <City>Ashford</City>  <CountryID>  <Name>UNITED KINGDOM</Name>  </CountryID>  <CustomerEmail>dhardie@gmail.com</CustomerEmail>  <TaskSubType>  <Name>MRI</Name>  </TaskSubType>  <TaskID>FR-A1234567-1\_D4\_20131024T063533</TaskID>  <MUSTJobNumber>A1234567</MUSTJobNumber>  <SystemID>  <ID>DARK1</ID>  </SystemID>  <IsSafety>false</IsSafety>  <CustomerExpectation>The Customer Expectation</CustomerExpectation>  <UseDistrictCalendar>false</UseDistrictCalendar>  <ContractOfferingFamily>Gold</ContractOfferingFamily>  <PreferredFSEs>FHollande;Nicolas Sarkozy</PreferredFSEs>  <ExcludedFSEs>JChirac;FMitterrand;Vd’Estaing</ExcludedFSEs>  <RequiredFSEs>FHollande</RequiredFSEs>  <SkillLevel>1</SkillLevel>  <OwnerName>Charles de Gaulle</OwnerName>  <OwnerSSO>Charles de Gaulle</OwnerSSO>  <CRMSystemName>  <Name>MUST</Name>  </CRMSystemName>  <PartDeliveryType>In the back of a van</PartDeliveryType>  <PartComment>Bend your knees when you pick it up, its heavy!</PartComment>  <PartEstimatedDeliveryDate>2013-10-24T06:35:33.3954348-04:00</PartEstimatedDeliveryDate>  <ReactivationDate>2013-10-24T06:35:33.3954348-04:00</ReactivationDate>  <MacroVersion>MacroVersion001</MacroVersion>  <TaskLanguages>English;French</TaskLanguages>  <IsMST>false</IsMST>  </Task>  <ReturnAssignment>true</ReturnAssignment>  <TaskRequestedProperties>  <Item>Key</Item>  <Item>CallID</Item>  <Item>Number</Item>  <Item>TaskID</Item>  <Item>MUSTJobNumber</Item>  </TaskRequestedProperties>  <AssignmentRequestedProperties>  <Item>Key</Item>  <Item>Start</Item>  <Item>Finish</Item>  </AssignmentRequestedProperties>  <ReturnSchedulingError>false</ReturnSchedulingError>  </ProcessTaskEx>  </s:Body>  </s:Envelope> |

#### Response Message

On success, a message will be returned with task data according to the settings of the TaskRequestedProperties in the request message. For example if the request message specified the requested properties.

TaskRequestedProperties Example:

|  |
| --- |
| <TaskRequestedProperties>  <Item>Key</Item>  <Item>CallID</Item>  <Item>Number</Item>  <Item>TaskID</Item>  <Item>MUSTJobNumber</Item>  </TaskRequestedProperties> |

The response should resemble the following:

|  |
| --- |
| <s:Envelope xmlns:s = "http://schemas.xmlsoap.org/soap/envelope/">  <s:Body>  <ProcessTaskExResponse xmlns = "http://www.clicksoftware.com">  <ReturnCode>Success</ReturnCode>  <Task>  <Key>39765236</Key>  <CallID>GB-A1234567-1</CallID>  <Number>1</Number>  <TaskID>FR-A1234567-1\_D4\_20131024T063533</TaskID>  <MUSTJobNumber>A1234567</MUSTJobNumber>  </Task>  </ProcessTaskExResponse>  </s:Body>  </s:Envelope> |

#### Multi Stage Task considerations

GEHC will have MST’s in a number of scenarios:

* Training Jobs;
* Multi Manning;
* Parts Pickup;

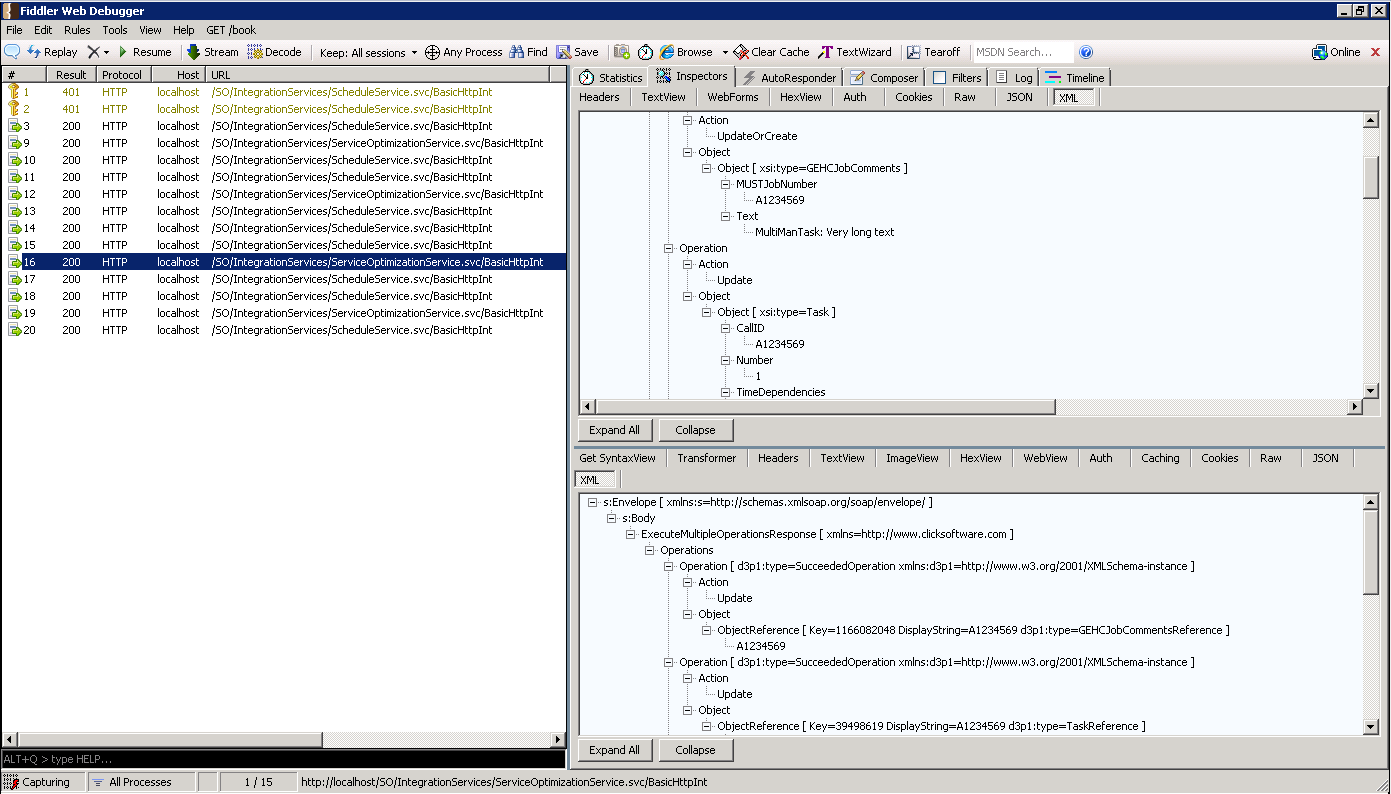
#### Training Task

My way of example for the Business scenarios 2.6.2 the SOAP stack for creating a training job follows the following pattern , this is also described here 2.6.2.4 :

* ProcessTaskEx – Main Job
  + IsMST: set to true so the BGO can release tasks once comments are added;
* ProcessTaskEx – Training Job;
  + IsMST: set to true so the BGO can release tasks once comments are added;
* ExecuteMultipleOperations, see 3.6
  + Operation 1:
    - CreateOrUpdate ExecuteMultipleOperations;
  + Operation 2:
    - Update Main Tasks JobComments to new GEHCJobComments;
    - Update Main Task with Time Dependancies;
    - Update IsMST for tasks that were orggionaly single tasks;
  + Operation 3:
    - Update Training Tasks JobComments to new GEHCJobComments;

As with all MST as detailed in 2.6.2.6, should a single task require a training job later, then the single task would become an MST task. The IsMST flag on both main and training tasks would be set to “True”. The ExecuteMultipleOperations setting the dependency can be used to make this change to the main task.

Below is a screen shot of the SOAP stack depicting a single task, Parts Pickup, multi man, training and a GetTasks a fragment of the Request and response from the multi man ExecuteMultipleOperations call:



#### ProcessTaskEx First

SOAP Request Envelope for the main job:

|  |
| --- |
| <s:Envelope xmlns:s = "http://schemas.xmlsoap.org/soap/envelope/">  <s:Header>  <h:OptionalParameters  xmlns:h = "http://www.clicksoftware.com/OptionalParameters"  xmlns = "http://www.clicksoftware.com/OptionalParameters"  xmlns:xsi = "http://www.w3.org/2001/XMLSchema-instance"  xmlns:xsd = "http://www.w3.org/2001/XMLSchema">  <CallerIdentity>MUST</CallerIdentity>  <ErrorOnNonExistingDictionaries>false</ErrorOnNonExistingDictionaries>  </h:OptionalParameters>  </s:Header>  <s:Body xmlns:xsi = "http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd = "http://www.w3.org/2001/XMLSchema">  <ProcessTaskEx xmlns = "http://www.clicksoftware.com">  <Task>  <CallID>GB-A1234570-1</CallID>  <Number>1</Number>  <DueDate>2013-11-07T06:36:58.9805354-05:00</DueDate>  <Priority>10</Priority>  <Customer>Cristina Hardie</Customer>  <District>  <Name>UK South</Name>  </District>  <Postcode>TN25 5AB</Postcode>  <OpenDate>2013-10-24T06:36:58.9805354-04:00</OpenDate>  <TaskType>  <Name>Installation</Name>  </TaskType>  <Duration>900</Duration>  <NumberOfRequiredEngineers>1</NumberOfRequiredEngineers>  <ContactPhoneNumber>+447720449438</ContactPhoneNumber>  <Latitude>51194268</Latitude>  <Longitude>918754</Longitude>  <Street>Kent</Street>  <City>Ashford</City>  <CountryID>  <Name>UNITED KINGDOM</Name>  </CountryID>  <CustomerEmail>dhardie@gmail.com</CustomerEmail>  <TaskSubType>  <Name>MRI</Name>  </TaskSubType>  <TaskID>FR-A1234570-1\_D4\_20131024T063658</TaskID>  <MUSTJobNumber>A1234570</MUSTJobNumber>  <SystemID>  <ID>DARK1</ID>  </SystemID>  <IsSafety>false</IsSafety>  <CustomerExpectation>The Customer Expectation</CustomerExpectation>  <UseDistrictCalendar>false</UseDistrictCalendar>  <ContractOfferingFamily>Gold</ContractOfferingFamily>  <PreferredFSEs>FHollande;Nicolas Sarkozy</PreferredFSEs>  <ExcludedFSEs>JChirac;FMitterrand;Vd’Estaing</ExcludedFSEs>  <RequiredFSEs>FHollande</RequiredFSEs>  <SkillLevel>1</SkillLevel>  <OwnerName>Charles de Gaulle</OwnerName>  <OwnerSSO>Charles de Gaulle</OwnerSSO>  <CRMSystemName>  <Name>MUST</Name>  </CRMSystemName>  <PartDeliveryType>In the back of a van</PartDeliveryType>  <PartComment>Bend your knees when you pick it up, its heavy!</PartComment>  <PartEstimatedDeliveryDate>2013-10-24T06:36:58.9805354-04:00</PartEstimatedDeliveryDate>  <ReactivationDate>2013-10-24T06:36:58.9805354-04:00</ReactivationDate>  <MacroVersion>MacroVersion001</MacroVersion>  <TaskLanguages>English;French</TaskLanguages>  <IsMST>true</IsMST>  </Task>  <ReturnAssignment>true</ReturnAssignment>  <TaskRequestedProperties>  <Item>Key</Item>  <Item>CallID</Item>  <Item>Number</Item>  <Item>TaskID</Item>  <Item>MUSTJobNumber</Item>  </TaskRequestedProperties>  <AssignmentRequestedProperties>  <Item>Key</Item>  <Item>Start</Item>  <Item>Finish</Item>  </AssignmentRequestedProperties>  <ReturnSchedulingError>false</ReturnSchedulingError>  </ProcessTaskEx>  </s:Body>  </s:Envelope> |

SOAP Response Envelope:

|  |
| --- |
| <s:Envelope xmlns:s = "http://schemas.xmlsoap.org/soap/envelope/">  <s:Body>  <ProcessTaskExResponse xmlns = "http://www.clicksoftware.com">  <ReturnCode>Success</ReturnCode>  <Task>  <Key>39459176</Key>  <CallID>GB-20130913\_1-1</CallID>  <Number>1</Number>  <TaskID>dhardie2013-09-13T14:31:33.9516734Z</TaskID>  </Task>  <Assignment>  <Key>2218052</Key>  <Revision>1</Revision>  <Stamp>  <CreatedBy>MUST</CreatedBy>  <TimeCreated>2013-09-13T10:31:43</TimeCreated>  <CreatingProcess>0</CreatingProcess>  <ModifiedBy>MUST</ModifiedBy>  <TimeModified>2013-09-13T10:31:43</TimeModified>  <ModifyingProcess>1784</ModifyingProcess>  </Stamp>  <Task Key = "39459176" DisplayString = "20130913\_1">  <CallID>GB-20130913\_1-1</CallID>  <Number>1</Number>  </Task>  <Start>2013-09-13T15:37:00</Start>  <Finish>2013-09-13T15:52:00</Finish>  <Engineers>  <Engineer Key = "789620737" DisplayString = "dhardie002">  <ID>dhardie002</ID>  <District Key = "124422144" DisplayString = "UK South">UK South</District>  </Engineer>  </Engineers>  <Comment></Comment>  <Location></Location>  <BinaryData>BAAAAAAAOGAAAAAABAAAAAAABAAAAAAAFGAAAAAAAAAABAIKABPC</BinaryData>  <Latitude>51194268</Latitude>  <Longitude>918754</Longitude>  <GISDataSource>0</GISDataSource>  <AssignedEngineers>dhardie002</AssignedEngineers>  <LogicPolicy>Schedule Workflow Stage</LogicPolicy>  <IsCrewAssignment>0</IsCrewAssignment>  <CountryID Key = "126116068" DisplayString = "UNITED KINGDOM">UNITED KINGDOM</CountryID>  <Street></Street>  <City></City>  <State></State>  <Postcode></Postcode>  <NonAvailabilityType Key = "-1"></NonAvailabilityType>  <IgnoreInRoster>0</IgnoreInRoster>  <ContractorIndex>0</ContractorIndex>  <StateSubdivision></StateSubdivision>  <CitySubdivision></CitySubdivision>  <ID></ID>  <WorkAgreementID></WorkAgreementID>  <ExternalRefID></ExternalRefID>  <IsBreakIncluded>0</IsBreakIncluded>  <IncludedBreakDuration>0</IncludedBreakDuration>  <AbsenceRequest Key = "-1"></AbsenceRequest>  <ExternalComment></ExternalComment>  <AssignmentSource>0</AssignmentSource>  <IsMST>true</IsMST>  </Assignment>  </ProcessTaskExResponse>  </s:Body>  </s:Envelope> |

#### ProcessTaskEx Second

SOAP Request Envelope for the training job:

|  |
| --- |
| <s:Envelope xmlns:s = "http://schemas.xmlsoap.org/soap/envelope/">  <s:Header>  <h:OptionalParameters  xmlns:h = "http://www.clicksoftware.com/OptionalParameters"  xmlns = "http://www.clicksoftware.com/OptionalParameters"  xmlns:xsi = "http://www.w3.org/2001/XMLSchema-instance"  xmlns:xsd = "http://www.w3.org/2001/XMLSchema">  <CallerIdentity>MUST</CallerIdentity>  <ErrorOnNonExistingDictionaries>false</ErrorOnNonExistingDictionaries>  </h:OptionalParameters>  </s:Header>  <s:Body xmlns:xsi = "http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd = "http://www.w3.org/2001/XMLSchema">  <ProcessTaskEx xmlns = "http://www.clicksoftware.com">  <Task>  <CallID>GB-A1234570-1</CallID>  <Number>2</Number>  <DueDate>2013-11-07T06:36:58.9805354-05:00</DueDate>  <Priority>10</Priority>  <Customer>Cristina Hardie</Customer>  <District>  <Name>UK South</Name>  </District>  <Postcode>TN25 5AB</Postcode>  <OpenDate>2013-10-24T06:36:58.9805354-04:00</OpenDate>  <TaskType>  <Name>Training</Name>  </TaskType>  <Duration>900</Duration>  <RequiredEngineers>  <RequiredEngineer>  <ID>Trainee01</ID>  </RequiredEngineer>  </RequiredEngineers>  <NumberOfRequiredEngineers>1</NumberOfRequiredEngineers>  <ContactPhoneNumber>+447720449438</ContactPhoneNumber>  <Latitude>51194268</Latitude>  <Longitude>918754</Longitude>  <Street>Kent</Street>  <City>Ashford</City>  <CountryID>  <Name>UNITED KINGDOM</Name>  </CountryID>  <CustomerEmail>dhardie@gmail.com</CustomerEmail>  <TaskSubType>  <Name>MRI</Name>  </TaskSubType>  <TaskID>FR-A1111111-1\_D4\_20131024T063758</TaskID>  <MUSTJobNumber>A1111111</MUSTJobNumber>  <SystemID>  <ID>DARK1</ID>  </SystemID>  <IsSafety>false</IsSafety>  <CustomerExpectation>The Customer Expectation</CustomerExpectation>  <UseDistrictCalendar>false</UseDistrictCalendar>  <ContractOfferingFamily>Gold</ContractOfferingFamily>  <PreferredFSEs>FHollande;Nicolas Sarkozy</PreferredFSEs>  <ExcludedFSEs>JChirac;FMitterrand;Vd’Estaing</ExcludedFSEs>  <RequiredFSEs>FHollande</RequiredFSEs>  <SkillLevel>1</SkillLevel>  <OwnerName>Charles de Gaulle</OwnerName>  <OwnerSSO>Charles de Gaulle</OwnerSSO>  <CRMSystemName>  <Name>MUST</Name>  </CRMSystemName>  <PartDeliveryType>In the back of a van</PartDeliveryType>  <PartComment>Bend your knees when you pick it up, its heavy!</PartComment>  <PartEstimatedDeliveryDate>2013-10-24T06:36:58.9805354-04:00</PartEstimatedDeliveryDate>  <ReactivationDate>2013-10-24T06:36:58.9805354-04:00</ReactivationDate>  <MacroVersion>MacroVersion001</MacroVersion>  <TaskLanguages>English;French</TaskLanguages>  <IsMST>true</IsMST>  </Task>  <ReturnAssignment>true</ReturnAssignment>  <TaskRequestedProperties>  <Item>Key</Item>  <Item>CallID</Item>  <Item>Number</Item>  <Item>TaskID</Item>  <Item>MUSTJobNumber</Item>  </TaskRequestedProperties>  <AssignmentRequestedProperties>  <Item>Key</Item>  <Item>Start</Item>  <Item>Finish</Item>  </AssignmentRequestedProperties>  <ReturnSchedulingError>false</ReturnSchedulingError>  </ProcessTaskEx>  </s:Body>  </s:Envelope> |

#### ExecuteMultipleOperations

SOAP Request Envelope:

|  |
| --- |
| <s:Envelope xmlns:s = "http://schemas.xmlsoap.org/soap/envelope/">  <s:Header>  <h:OptionalParameters  xmlns:h = "http://www.clicksoftware.com/OptionalParameters"  xmlns = "http://www.clicksoftware.com/OptionalParameters"  xmlns:xsi = "http://www.w3.org/2001/XMLSchema-instance"  xmlns:xsd = "http://www.w3.org/2001/XMLSchema">  <CallerIdentity>MUSTMB</CallerIdentity>  <ErrorOnNonExistingDictionaries>true</ErrorOnNonExistingDictionaries>  </h:OptionalParameters>  </s:Header>  <s:Body xmlns:xsi = "http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd = "http://www.w3.org/2001/XMLSchema">  <ExecuteMultipleOperations xmlns = "http://www.clicksoftware.com">  <Operations>  <Operation>  <Action>UpdateOrCreate</Action>  <Object>  <Object xsi:type = "GEHCJobComments">  <MUSTJobNumber>A1234570</MUSTJobNumber>  <Text>TrainingTask: Very long text</Text>  </Object>  </Object>  </Operation>  <Operation>  <Action>Update</Action>  <Object>  <Object xsi:type = "Task">  <CallID>GB-A1234570-1</CallID>  <Number>1</Number>  <TimeDependencies>  <TaskTimeDependency>  <TaskKey>  <CallID>GB-A1234570-1</CallID>  <Number>2</Number>  </TaskKey>  <RelationType>1</RelationType>  <RelationOperator>2</RelationOperator>  <UpperBound>0</UpperBound>  <LowerBound>0</LowerBound>  <Critical>true</Critical>  </TaskTimeDependency>  </TimeDependencies>  <EngineerDependencies>  <TaskEngineerDependency>  <TaskKey>  <CallID>GB-A1234570-1</CallID>  <Number>2</Number>  </TaskKey>  <RelationType>2</RelationType>  </TaskEngineerDependency>  </EngineerDependencies>  <JobComments>  <MUSTJobNumber>A1234570</MUSTJobNumber>  </JobComments>  <TrainingJobNumber>A1234570</TrainingJobNumber>  </Object>  </Object>  </Operation>  <Operation>  <Action>UpdateOrCreate</Action>  <Object>  <Object xsi:type = "GEHCJobComments">  <MUSTJobNumber>A1111111</MUSTJobNumber>  <Text>TrainingTask: Very long text</Text>  </Object>  </Object>  </Operation>  <Operation>  <Action>Update</Action>  <Object>  <Object xsi:type = "Task">  <CallID>GB-A1234570-1</CallID>  <Number>2</Number>  <JobComments>  <MUSTJobNumber>A1111111</MUSTJobNumber>  </JobComments>  <TrainingJobNumber>A1111111</TrainingJobNumber>  </Object>  </Object>  </Operation>  </Operations>  <OneTransaction>false</OneTransaction>  <ContinueOnError>false</ContinueOnError>  </ExecuteMultipleOperations>  </s:Body>  </s:Envelope> |

SOAP Response Envelope:

|  |
| --- |
| <s:Envelope xmlns:s = "http://schemas.xmlsoap.org/soap/envelope/">  <s:Body>  <ExecuteMultipleOperationsResponse xmlns = "http://www.clicksoftware.com">  <Operations>  <Operation d3p1:type = "SucceededOperation" xmlns:d3p1 = "http://www.w3.org/2001/XMLSchema-instance">  <Action>Update</Action>  <Object>  <ObjectReference  Key = "1166016513"  DisplayString = "A1234567"  d3p1:type = "GEHCJobCommentsReference">A1234567</ObjectReference>  </Object>  </Operation>  <Operation d3p1:type = "SucceededOperation" xmlns:d3p1 = "http://www.w3.org/2001/XMLSchema-instance">  <Action>Update</Action>  <Object>  <ObjectReference  Key = "39459176"  DisplayString = "20130913\_1"  d3p1:type = "TaskReference">  <CallID>GB-20130913\_1-1</CallID>  <Number>1</Number>  </ObjectReference>  </Object>  </Operation>  <Operation d3p1:type = "SucceededOperation" xmlns:d3p1 = "http://www.w3.org/2001/XMLSchema-instance">  <Action>Update</Action>  <Object>  <ObjectReference  Key = "39459177"  DisplayString = "20130913\_1"  d3p1:type = "TaskReference">  <CallID>GB-20130913\_1-1</CallID>  <Number>2</Number>  </ObjectReference>  </Object>  </Operation>  </Operations>  </ExecuteMultipleOperationsResponse>  </s:Body>  </s:Envelope> |

#### Additional Trainees

To add more trainees an additional ProcessTaskEx call would be made. The rules for the Trainee task would follow those specified in 3.5.4.

The Task.Number would differ from the Numbers of the other tasks in this job and the ExecuteMultipleOperations would contain the relevant new JobComments object, task link and dependancies on the main Task.

#### Multi Manning

My way of example for the Business scenarios 2.6.2 the SOAP stack for creating a training job follows the following pattern , this is also described here 2.6.2.2 :

* ProcessTaskEx – Main Job
  + IsMST: set to true so the BGO can release tasks once comments are added;
* ProcessTaskEx – Secondary Task 1;
  + IsMST: set to true so the BGO can release tasks once comments are added;
* ProcessTaskEx – Secondary Task 2;
  + IsMST: set to true so the BGO can release tasks once comments are added;
* ExecuteMultipleOperations, see 3.6
  + Operation 1:
    - CreateOrUpdate ExecuteMultipleOperations;
  + Operation 2:
    - Update Main Tasks JobComments to new GEHCJobComments;
    - Update Main Task with Time Dependancies;
    - Update Main Task with Engineer Dependancies;
  + Operation 3:
    - Update Secondary Tasks JobComments to new GEHCJobComments;

#### ProcessTaskEx First

|  |
| --- |
| <s:Envelope xmlns:s = "http://schemas.xmlsoap.org/soap/envelope/">  <s:Header>  <h:OptionalParameters  xmlns:h = "http://www.clicksoftware.com/OptionalParameters"  xmlns = "http://www.clicksoftware.com/OptionalParameters"  xmlns:xsi = "http://www.w3.org/2001/XMLSchema-instance"  xmlns:xsd = "http://www.w3.org/2001/XMLSchema">  <CallerIdentity>MUST</CallerIdentity>  <ErrorOnNonExistingDictionaries>false</ErrorOnNonExistingDictionaries>  </h:OptionalParameters>  </s:Header>  <s:Body xmlns:xsi = "http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd = "http://www.w3.org/2001/XMLSchema">  <ProcessTaskEx xmlns = "http://www.clicksoftware.com">  <Task>  <CallID>GB-A1234569-1</CallID>  <Number>1</Number>  <DueDate>2013-11-07T06:36:51.6853928-05:00</DueDate>  <Priority>10</Priority>  <Customer>Cristina Hardie</Customer>  <District>  <Name>UK South</Name>  </District>  <Postcode>TN25 5AB</Postcode>  <OpenDate>2013-10-24T06:36:51.6853928-04:00</OpenDate>  <TaskType>  <Name>Installation</Name>  </TaskType>  <Duration>900</Duration>  <NumberOfRequiredEngineers>1</NumberOfRequiredEngineers>  <ContactPhoneNumber>+447720449438</ContactPhoneNumber>  <Latitude>51194268</Latitude>  <Longitude>918754</Longitude>  <Street>Kent</Street>  <City>Ashford</City>  <CountryID>  <Name>UNITED KINGDOM</Name>  </CountryID>  <CustomerEmail>dhardie@gmail.com</CustomerEmail>  <TaskSubType>  <Name>MRI</Name>  </TaskSubType>  <TaskID>FR-A1234569-1\_D4\_20131024T063651</TaskID>  <MUSTJobNumber>A1234569</MUSTJobNumber>  <SystemID>  <ID>DARK1</ID>  </SystemID>  <IsSafety>false</IsSafety>  <CustomerExpectation>The Customer Expectation</CustomerExpectation>  <UseDistrictCalendar>false</UseDistrictCalendar>  <ContractOfferingFamily>Gold</ContractOfferingFamily>  <PreferredFSEs>FHollande;Nicolas Sarkozy</PreferredFSEs>  <ExcludedFSEs>JChirac;FMitterrand;Vd’Estaing</ExcludedFSEs>  <RequiredFSEs>FHollande</RequiredFSEs>  <SkillLevel>1</SkillLevel>  <OwnerName>Charles de Gaulle</OwnerName>  <OwnerSSO>Charles de Gaulle</OwnerSSO>  <CRMSystemName>  <Name>MUST</Name>  </CRMSystemName>  <PartDeliveryType>In the back of a van</PartDeliveryType>  <PartComment>Bend your knees when you pick it up, its heavy!</PartComment>  <PartEstimatedDeliveryDate>2013-10-24T06:36:51.6853928-04:00</PartEstimatedDeliveryDate>  <ReactivationDate>2013-10-24T06:36:51.6853928-04:00</ReactivationDate>  <MacroVersion>MacroVersion001</MacroVersion>  <TaskLanguages>English;French</TaskLanguages>  <IsMST>true</IsMST>  </Task>  <ReturnAssignment>true</ReturnAssignment>  <TaskRequestedProperties>  <Item>Key</Item>  <Item>CallID</Item>  <Item>Number</Item>  <Item>TaskID</Item>  <Item>MUSTJobNumber</Item>  </TaskRequestedProperties>  <AssignmentRequestedProperties>  <Item>Key</Item>  <Item>Start</Item>  <Item>Finish</Item>  </AssignmentRequestedProperties>  <ReturnSchedulingError>false</ReturnSchedulingError>  </ProcessTaskEx>  </s:Body>  </s:Envelope> |

#### ProcessTaskEx Second

|  |
| --- |
| <s:Envelope xmlns:s = "http://schemas.xmlsoap.org/soap/envelope/">  <s:Header>  <h:OptionalParameters  xmlns:h = "http://www.clicksoftware.com/OptionalParameters"  xmlns = "http://www.clicksoftware.com/OptionalParameters"  xmlns:xsi = "http://www.w3.org/2001/XMLSchema-instance"  xmlns:xsd = "http://www.w3.org/2001/XMLSchema">  <CallerIdentity>MUST</CallerIdentity>  <ErrorOnNonExistingDictionaries>false</ErrorOnNonExistingDictionaries>  </h:OptionalParameters>  </s:Header>  <s:Body xmlns:xsi = "http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd = "http://www.w3.org/2001/XMLSchema">  <ProcessTaskEx xmlns = "http://www.clicksoftware.com">  <Task>  <CallID>GB-A1234569-1</CallID>  <Number>2</Number>  <DueDate>2013-11-07T06:36:51.6853928-05:00</DueDate>  <Priority>10</Priority>  <Customer>Cristina Hardie</Customer>  <District>  <Name>UK South</Name>  </District>  <Postcode>TN25 5AB</Postcode>  <OpenDate>2013-10-24T06:36:51.6853928-04:00</OpenDate>  <TaskType>  <Name>Installation</Name>  </TaskType>  <Duration>900</Duration>  <NumberOfRequiredEngineers>1</NumberOfRequiredEngineers>  <ContactPhoneNumber>+447720449438</ContactPhoneNumber>  <Latitude>51194268</Latitude>  <Longitude>918754</Longitude>  <Street>Kent</Street>  <City>Ashford</City>  <CountryID>  <Name>UNITED KINGDOM</Name>  </CountryID>  <CustomerEmail>dhardie@gmail.com</CustomerEmail>  <TaskSubType>  <Name>MRI</Name>  </TaskSubType>  <TaskID>FR-A1234569-1\_D4\_20131024T063751</TaskID>  <MUSTJobNumber>A1234569</MUSTJobNumber>  <SystemID>  <ID>DARK1</ID>  </SystemID>  <IsSafety>false</IsSafety>  <CustomerExpectation>The Customer Expectation</CustomerExpectation>  <UseDistrictCalendar>false</UseDistrictCalendar>  <ContractOfferingFamily>Gold</ContractOfferingFamily>  <PreferredFSEs>FHollande;Nicolas Sarkozy</PreferredFSEs>  <ExcludedFSEs>JChirac;FMitterrand;Vd’Estaing</ExcludedFSEs>  <RequiredFSEs>FHollande</RequiredFSEs>  <SkillLevel>1</SkillLevel>  <OwnerName>Charles de Gaulle</OwnerName>  <OwnerSSO>Charles de Gaulle</OwnerSSO>  <CRMSystemName>  <Name>MUST</Name>  </CRMSystemName>  <PartDeliveryType>In the back of a van</PartDeliveryType>  <PartComment>Bend your knees when you pick it up, its heavy!</PartComment>  <PartEstimatedDeliveryDate>2013-10-24T06:36:51.6853928-04:00</PartEstimatedDeliveryDate>  <ReactivationDate>2013-10-24T06:36:51.6853928-04:00</ReactivationDate>  <MacroVersion>MacroVersion001</MacroVersion>  <TaskLanguages>English;French</TaskLanguages>  <IsMST>true</IsMST>  </Task>  <ReturnAssignment>true</ReturnAssignment>  <TaskRequestedProperties>  <Item>Key</Item>  <Item>CallID</Item>  <Item>Number</Item>  <Item>TaskID</Item>  <Item>MUSTJobNumber</Item>  </TaskRequestedProperties>  <AssignmentRequestedProperties>  <Item>Key</Item>  <Item>Start</Item>  <Item>Finish</Item>  </AssignmentRequestedProperties>  <ReturnSchedulingError>false</ReturnSchedulingError>  </ProcessTaskEx>  </s:Body>  </s:Envelope> |

#### ProcessTaskEx Third

|  |
| --- |
| <s:Envelope xmlns:s = "http://schemas.xmlsoap.org/soap/envelope/">  <s:Header>  <h:OptionalParameters  xmlns:h = "http://www.clicksoftware.com/OptionalParameters"  xmlns = "http://www.clicksoftware.com/OptionalParameters"  xmlns:xsi = "http://www.w3.org/2001/XMLSchema-instance"  xmlns:xsd = "http://www.w3.org/2001/XMLSchema">  <CallerIdentity>MUST</CallerIdentity>  <ErrorOnNonExistingDictionaries>false</ErrorOnNonExistingDictionaries>  </h:OptionalParameters>  </s:Header>  <s:Body xmlns:xsi = "http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd = "http://www.w3.org/2001/XMLSchema">  <ProcessTaskEx xmlns = "http://www.clicksoftware.com">  <Task>  <CallID>GB-A1234569-1</CallID>  <Number>3</Number>  <DueDate>2013-11-07T06:36:51.6853928-05:00</DueDate>  <Priority>10</Priority>  <Customer>Cristina Hardie</Customer>  <District>  <Name>UK South</Name>  </District>  <Postcode>TN25 5AB</Postcode>  <OpenDate>2013-10-24T06:36:51.6853928-04:00</OpenDate>  <TaskType>  <Name>Installation</Name>  </TaskType>  <Duration>900</Duration>  <NumberOfRequiredEngineers>1</NumberOfRequiredEngineers>  <ContactPhoneNumber>+447720449438</ContactPhoneNumber>  <Latitude>51194268</Latitude>  <Longitude>918754</Longitude>  <Street>Kent</Street>  <City>Ashford</City>  <CountryID>  <Name>UNITED KINGDOM</Name>  </CountryID>  <CustomerEmail>dhardie@gmail.com</CustomerEmail>  <TaskSubType>  <Name>MRI</Name>  </TaskSubType>  <TaskID>FR-A1234569-1\_D4\_20131024T063851</TaskID>  <MUSTJobNumber>A1234569</MUSTJobNumber>  <SystemID>  <ID>DARK1</ID>  </SystemID>  <IsSafety>false</IsSafety>  <CustomerExpectation>The Customer Expectation</CustomerExpectation>  <UseDistrictCalendar>false</UseDistrictCalendar>  <ContractOfferingFamily>Gold</ContractOfferingFamily>  <PreferredFSEs>FHollande;Nicolas Sarkozy</PreferredFSEs>  <ExcludedFSEs>JChirac;FMitterrand;Vd’Estaing</ExcludedFSEs>  <RequiredFSEs>FHollande</RequiredFSEs>  <SkillLevel>1</SkillLevel>  <OwnerName>Charles de Gaulle</OwnerName>  <OwnerSSO>Charles de Gaulle</OwnerSSO>  <CRMSystemName>  <Name>MUST</Name>  </CRMSystemName>  <PartDeliveryType>In the back of a van</PartDeliveryType>  <PartComment>Bend your knees when you pick it up, its heavy!</PartComment>  <PartEstimatedDeliveryDate>2013-10-24T06:36:51.6853928-04:00</PartEstimatedDeliveryDate>  <ReactivationDate>2013-10-24T06:36:51.6853928-04:00</ReactivationDate>  <MacroVersion>MacroVersion001</MacroVersion>  <TaskLanguages>English;French</TaskLanguages>  <IsMST>true</IsMST>  </Task>  <ReturnAssignment>true</ReturnAssignment>  <TaskRequestedProperties>  <Item>Key</Item>  <Item>CallID</Item>  <Item>Number</Item>  <Item>TaskID</Item>  <Item>MUSTJobNumber</Item>  </TaskRequestedProperties>  <AssignmentRequestedProperties>  <Item>Key</Item>  <Item>Start</Item>  <Item>Finish</Item>  </AssignmentRequestedProperties>  <ReturnSchedulingError>false</ReturnSchedulingError>  </ProcessTaskEx>  </s:Body>  </s:Envelope> |

#### ExecuteMultipleOperations

|  |
| --- |
| <s:Envelope xmlns:s = "http://schemas.xmlsoap.org/soap/envelope/">  <s:Header>  <h:OptionalParameters  xmlns:h = "http://www.clicksoftware.com/OptionalParameters"  xmlns = "http://www.clicksoftware.com/OptionalParameters"  xmlns:xsi = "http://www.w3.org/2001/XMLSchema-instance"  xmlns:xsd = "http://www.w3.org/2001/XMLSchema">  <CallerIdentity>MUSTMB</CallerIdentity>  <ErrorOnNonExistingDictionaries>true</ErrorOnNonExistingDictionaries>  </h:OptionalParameters>  </s:Header>  <s:Body xmlns:xsi = "http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd = "http://www.w3.org/2001/XMLSchema">  <ExecuteMultipleOperations xmlns = "http://www.clicksoftware.com">  <Operations>  <Operation>  <Action>UpdateOrCreate</Action>  <Object>  <Object xsi:type = "GEHCJobComments">  <MUSTJobNumber>A1234569</MUSTJobNumber>  <Text>MultiManTask: Very long text</Text>  </Object>  </Object>  </Operation>  <Operation>  <Action>Update</Action>  <Object>  <Object xsi:type = "Task">  <CallID>GB-A1234569-1</CallID>  <Number>1</Number>  <Critical>true</Critical>  <TimeDependencies>  <TaskTimeDependency>  <TaskKey>  <CallID>GB-A1234569-1</CallID>  <Number>2</Number>  </TaskKey>  <RelationType>1</RelationType>  <RelationOperator>2</RelationOperator>  <UpperBound>0</UpperBound>  <LowerBound>0</LowerBound>  <Critical>true</Critical>  </TaskTimeDependency>  <TaskTimeDependency>  <TaskKey>  <CallID>GB-A1234569-1</CallID>  <Number>3</Number>  </TaskKey>  <RelationType>1</RelationType>  <RelationOperator>2</RelationOperator>  <UpperBound>0</UpperBound>  <LowerBound>0</LowerBound>  <Critical>true</Critical>  </TaskTimeDependency>  </TimeDependencies>  <EngineerDependencies>  <TaskEngineerDependency>  <TaskKey>  <CallID>GB-A1234569-1</CallID>  <Number>2</Number>  </TaskKey>  <RelationType>2</RelationType>  </TaskEngineerDependency>  <TaskEngineerDependency>  <TaskKey>  <CallID>GB-A1234569-1</CallID>  <Number>3</Number>  </TaskKey>  <RelationType>2</RelationType>  </TaskEngineerDependency>  </EngineerDependencies>  <JobComments>  <MUSTJobNumber>A1234569</MUSTJobNumber>  </JobComments>  </Object>  </Object>  </Operation>  <Operation>  <Action>Update</Action>  <Object>  <Object xsi:type = "Task">  <CallID>GB-A1234569-1</CallID>  <Number>2</Number>  <Critical>true</Critical>  <JobComments>  <MUSTJobNumber>A1234569</MUSTJobNumber>  </JobComments>  </Object>  </Object>  </Operation>  <Operation>  <Action>Update</Action>  <Object>  <Object xsi:type = "Task">  <CallID>GB-A1234569-1</CallID>  <Number>3</Number>  <Critical>true</Critical>  <JobComments>  <MUSTJobNumber>A1234569</MUSTJobNumber>  </JobComments>  </Object>  </Object>  </Operation>  </Operations>  <OneTransaction>false</OneTransaction>  <ContinueOnError>false</ContinueOnError>  </ExecuteMultipleOperations>  </s:Body>  </s:Envelope> |

#### Parts Pickup

My way of example for the Business scenarios 2.6.2 the SOAP stack for creating a training job follows the following pattern , this is also described here 2.6.2.3 :

* ProcessTaskEx – Main Job
  + IsMST: set to true so the BGO can release tasks once comments are added;
* ProcessTaskEx – Pickup Task 1;
  + IsMST: set to true so the BGO can release tasks once comments are added;
* ExecuteMultipleOperations, see 3.6
  + Operation 1:
    - CreateOrUpdate ExecuteMultipleOperations;
  + Operation 2:
    - Update Main Tasks JobComments to new GEHCJobComments;
    - Update Main Task with Time Dependancies;
      * Start after the finish of the pickup;
      * Start on same day as the pickup (optional);
    - Update Main Task with Engineer Dependancies;
  + Operation 3:
    - Update Pickup Tasks JobComments to new GEHCJobComments;

Note: The same day dependency can be added for pickup jobs that should span multiple days.

#### ProcessTaskEx First

|  |
| --- |
| <s:Envelope xmlns:s = "http://schemas.xmlsoap.org/soap/envelope/">  <s:Header>  <h:OptionalParameters  xmlns:h = "http://www.clicksoftware.com/OptionalParameters"  xmlns = "http://www.clicksoftware.com/OptionalParameters"  xmlns:xsi = "http://www.w3.org/2001/XMLSchema-instance"  xmlns:xsd = "http://www.w3.org/2001/XMLSchema">  <CallerIdentity>MUST</CallerIdentity>  <ErrorOnNonExistingDictionaries>false</ErrorOnNonExistingDictionaries>  </h:OptionalParameters>  </s:Header>  <s:Body xmlns:xsi = "http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd = "http://www.w3.org/2001/XMLSchema">  <ProcessTaskEx xmlns = "http://www.clicksoftware.com">  <Task>  <CallID>GB-A1234568-1</CallID>  <Number>1</Number>  <DueDate>2013-11-07T06:36:44.6138-05:00</DueDate>  <Priority>10</Priority>  <Customer>Cristina Hardie</Customer>  <District>  <Name>UK South</Name>  </District>  <Postcode>TN25 5AB</Postcode>  <OpenDate>2013-10-24T06:36:44.6138-04:00</OpenDate>  <TaskType>  <Name>Installation</Name>  </TaskType>  <Duration>900</Duration>  <NumberOfRequiredEngineers>1</NumberOfRequiredEngineers>  <ContactPhoneNumber>+447720449438</ContactPhoneNumber>  <Latitude>51194268</Latitude>  <Longitude>918754</Longitude>  <Street>Kent</Street>  <City>Ashford</City>  <CountryID>  <Name>UNITED KINGDOM</Name>  </CountryID>  <CustomerEmail>dhardie@gmail.com</CustomerEmail>  <TaskSubType>  <Name>MRI</Name>  </TaskSubType>  <TaskID>FR-A1234568-1\_D4\_20131024T063644</TaskID>  <MUSTJobNumber>A1234568</MUSTJobNumber>  <SystemID>  <ID>DARK1</ID>  </SystemID>  <IsSafety>false</IsSafety>  <CustomerExpectation>The Customer Expectation</CustomerExpectation>  <UseDistrictCalendar>false</UseDistrictCalendar>  <ContractOfferingFamily>Gold</ContractOfferingFamily>  <PreferredFSEs>FHollande;Nicolas Sarkozy</PreferredFSEs>  <ExcludedFSEs>JChirac;FMitterrand;Vd’Estaing</ExcludedFSEs>  <RequiredFSEs>FHollande</RequiredFSEs>  <SkillLevel>1</SkillLevel>  <OwnerName>Charles de Gaulle</OwnerName>  <OwnerSSO>Charles de Gaulle</OwnerSSO>  <CRMSystemName>  <Name>MUST</Name>  </CRMSystemName>  <PartDeliveryType>In the back of a van</PartDeliveryType>  <PartComment>Bend your knees when you pick it up, its heavy!</PartComment>  <PartEstimatedDeliveryDate>2013-10-24T06:36:44.6138-04:00</PartEstimatedDeliveryDate>  <ReactivationDate>2013-10-24T06:36:44.6138-04:00</ReactivationDate>  <MacroVersion>MacroVersion001</MacroVersion>  <TaskLanguages>English;French</TaskLanguages>  <IsMST>true</IsMST>  </Task>  <ReturnAssignment>true</ReturnAssignment>  <TaskRequestedProperties>  <Item>Key</Item>  <Item>CallID</Item>  <Item>Number</Item>  <Item>TaskID</Item>  <Item>MUSTJobNumber</Item>  </TaskRequestedProperties>  <AssignmentRequestedProperties>  <Item>Key</Item>  <Item>Start</Item>  <Item>Finish</Item>  </AssignmentRequestedProperties>  <ReturnSchedulingError>false</ReturnSchedulingError>  </ProcessTaskEx>  </s:Body>  </s:Envelope> |

#### ProcessTaskEx Second

|  |
| --- |
| <s:Envelope xmlns:s = "http://schemas.xmlsoap.org/soap/envelope/">  <s:Header>  <h:OptionalParameters  xmlns:h = "http://www.clicksoftware.com/OptionalParameters"  xmlns = "http://www.clicksoftware.com/OptionalParameters"  xmlns:xsi = "http://www.w3.org/2001/XMLSchema-instance"  xmlns:xsd = "http://www.w3.org/2001/XMLSchema">  <CallerIdentity>MUST</CallerIdentity>  <ErrorOnNonExistingDictionaries>false</ErrorOnNonExistingDictionaries>  </h:OptionalParameters>  </s:Header>  <s:Body xmlns:xsi = "http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd = "http://www.w3.org/2001/XMLSchema">  <ProcessTaskEx xmlns = "http://www.clicksoftware.com">  <Task>  <CallID>GB-A1234568-1</CallID>  <Number>2</Number>  <DueDate>2013-11-07T06:36:44.6138-05:00</DueDate>  <Priority>10</Priority>  <Customer>Cristina Hardie</Customer>  <District>  <Name>UK South</Name>  </District>  <Postcode>TN25 5AB</Postcode>  <OpenDate>2013-10-24T06:36:44.6138-04:00</OpenDate>  <TaskType>  <Name>Parts Pickup</Name>  </TaskType>  <Duration>900</Duration>  <NumberOfRequiredEngineers>1</NumberOfRequiredEngineers>  <ContactPhoneNumber>+447720449438</ContactPhoneNumber>  <Latitude>51194268</Latitude>  <Longitude>918754</Longitude>  <Street>Kent</Street>  <City>Ashford</City>  <CountryID>  <Name>UNITED KINGDOM</Name>  </CountryID>  <CustomerEmail>dhardie@gmail.com</CustomerEmail>  <TaskSubType>  <Name>MRI</Name>  </TaskSubType>  <TaskID>FR-A1234568-1\_D4\_20131024T063744</TaskID>  <MUSTJobNumber>A1234568</MUSTJobNumber>  <SystemID>  <ID>DARK1</ID>  </SystemID>  <IsSafety>false</IsSafety>  <CustomerExpectation>The Customer Expectation</CustomerExpectation>  <UseDistrictCalendar>false</UseDistrictCalendar>  <ContractOfferingFamily>Gold</ContractOfferingFamily>  <PreferredFSEs>FHollande;Nicolas Sarkozy</PreferredFSEs>  <ExcludedFSEs>JChirac;FMitterrand;Vd’Estaing</ExcludedFSEs>  <RequiredFSEs>FHollande</RequiredFSEs>  <SkillLevel>1</SkillLevel>  <OwnerName>Charles de Gaulle</OwnerName>  <OwnerSSO>Charles de Gaulle</OwnerSSO>  <CRMSystemName>  <Name>MUST</Name>  </CRMSystemName>  <PartDeliveryType>In the back of a van</PartDeliveryType>  <PartComment>Bend your knees when you pick it up, its heavy!</PartComment>  <PartEstimatedDeliveryDate>2013-10-24T06:36:44.6138-04:00</PartEstimatedDeliveryDate>  <ReactivationDate>2013-10-24T06:36:44.6138-04:00</ReactivationDate>  <MacroVersion>MacroVersion001</MacroVersion>  <TaskLanguages>English;French</TaskLanguages>  <IsMST>true</IsMST>  </Task>  <ReturnAssignment>true</ReturnAssignment>  <TaskRequestedProperties>  <Item>Key</Item>  <Item>CallID</Item>  <Item>Number</Item>  <Item>TaskID</Item>  <Item>MUSTJobNumber</Item>  </TaskRequestedProperties>  <AssignmentRequestedProperties>  <Item>Key</Item>  <Item>Start</Item>  <Item>Finish</Item>  </AssignmentRequestedProperties>  <ReturnSchedulingError>false</ReturnSchedulingError>  </ProcessTaskEx>  </s:Body>  </s:Envelope> |

#### ExecuteMultipleOperations

|  |
| --- |
| <s:Envelope xmlns:s = "http://schemas.xmlsoap.org/soap/envelope/">  <s:Header>  <h:OptionalParameters  xmlns:h = "http://www.clicksoftware.com/OptionalParameters"  xmlns = "http://www.clicksoftware.com/OptionalParameters"  xmlns:xsi = "http://www.w3.org/2001/XMLSchema-instance"  xmlns:xsd = "http://www.w3.org/2001/XMLSchema">  <CallerIdentity>MUSTMB</CallerIdentity>  <ErrorOnNonExistingDictionaries>true</ErrorOnNonExistingDictionaries>  </h:OptionalParameters>  </s:Header>  <s:Body xmlns:xsi = "http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd = "http://www.w3.org/2001/XMLSchema">  <ExecuteMultipleOperations xmlns = "http://www.clicksoftware.com">  <Operations>  <Operation>  <Action>UpdateOrCreate</Action>  <Object>  <Object xsi:type = "GEHCJobComments">  <MUSTJobNumber>A1234568</MUSTJobNumber>  <Text>PickupTask: Very long text</Text>  </Object>  </Object>  </Operation>  <Operation>  <Action>Update</Action>  <Object>  <Object xsi:type = "Task">  <CallID>GB-A1234568-1</CallID>  <Number>1</Number>  <Critical>true</Critical>  <TimeDependencies>  <TaskTimeDependency>  <TaskKey>  <CallID>GB-A1234568-1</CallID>  <Number>2</Number>  </TaskKey>  <UpperBound>0</UpperBound>  <LowerBound>0</LowerBound>  <RelationType>0</RelationType>  <RelationOperator>0</RelationOperator>  <Critical>true</Critical>  </TaskTimeDependency>  **<TaskTimeDependency>**  **<TaskKey>**  **<CallID>GB-A1234568-1</CallID>**  **<Number>2</Number>**  **</TaskKey>**  **<UpperBound>0</UpperBound>**  **<LowerBound>0</LowerBound>**  **<RelationType>4</RelationType>**  **<RelationOperator>0</RelationOperator>**  **<Critical>true</Critical>**  **</TaskTimeDependency>**  **</TimeDependencies>**  <EngineerDependencies>  <TaskEngineerDependency>  <TaskKey>  <CallID>GB-A1234568-1</CallID>  <Number>2</Number>  </TaskKey>  <RelationType>1</RelationType>  </TaskEngineerDependency>  </EngineerDependencies>  <JobComments>  <MUSTJobNumber>A1234568</MUSTJobNumber>  </JobComments>  </Object>  </Object>  </Operation>  </Operations>  <OneTransaction>false</OneTransaction>  <ContinueOnError>false</ContinueOnError>  </ExecuteMultipleOperations>  </s:Body>  </s:Envelope> |

Note: The TimeDependencies in bold is the optional same day dependency, exclude ethis should a pickup be required on a different day than the main task

#### Additional Pickups

To add more pickups additional ProcessTaskEx calls would be made. The rules for the Pickup tasks would follow those specified in 3.5.6

The Task.Number would differ from the Numbers of the other tasks in this job and the ExecuteMultipleOperations would contain the relevant dependancies on the main Task.

#### Task Cancellation

Only Tasks with statuses that can be Cancelled should be ProcessTaskEx, the relevant statuses are “New”, “Tentative”, “Rejected (FSE)”, “Rejected”, “Acknowledged”. To locate tasks for a job the GetTasks action is used, see 3.7

Once a Tasklist is returned by GetTasks the ProcessTaskEx can update the relevant task with a Status value of “Cancelled”.

To get the groups of tasks Click Grouping objects are used, by passing the filter parameter to the group in the GetTasks SOAP. This groups are based off the Click boilerplate “Statuses that can be Cancelled”.

#### Message Fields

In order to Cancel a task only a sub set of the Task properties need to be posted. As part of the workflow the Task Status needs to be queries to ensure that a task cancelation can take place, see 2.6.3.3 & 3.7.

Below are the task fields required for cancelling a task:

| Property | Visible Name | Description | Type | Access | Owner | |
| --- | --- | --- | --- | --- | --- | --- |
| **General** | | | | | |
| CallID | Call ID | These values form the unique identifier for the task in ClickSchedule. The CallID is defaulted to the MUST Job Number, while the Number is defaulted to 1, but can be different (e.g. when creating dependencies between “primary” and associated “training” or pickup tasks). | String (64) | Editable | GEHC | |
| Number | Number | Number | Editable | GEHC | |
| Task Status | Task Status | Cancellation | String (64) | Editable | GEHC | |
| CancellationReason | Cancellation Reason | The reason stated when the task is cancelled. | Dictionary (CancellationReason) | Editable when status is cancelled | CS | |

#### Single Task

By way of example for the Business scenarios 2.6.2.5 the SOAP stack for canceling a Training Task follows the following pattern:

* GetTasks
  + Using the “Statuses that can be Cancelled by CallID” group to return the relevant tasks;
* ProcessTaskEx – for returned Tasks;
  + Each Task returned by GetTasks needs cancelling;

GetTasks Request:

|  |
| --- |
| <s:Envelope xmlns:s = "http://schemas.xmlsoap.org/soap/envelope/">  <s:Header>  <h:OptionalParameters  xmlns:h = "http://www.clicksoftware.com/OptionalParameters"  xmlns = "http://www.clicksoftware.com/OptionalParameters"  xmlns:xsi = "http://www.w3.org/2001/XMLSchema-instance"  xmlns:xsd = "http://www.w3.org/2001/XMLSchema">  <CallerIdentity>MUST</CallerIdentity>  <ErrorOnNonExistingDictionaries>false</ErrorOnNonExistingDictionaries>  </h:OptionalParameters>  </s:Header>  <s:Body xmlns:xsi = "http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd = "http://www.w3.org/2001/XMLSchema">  <GetTasks xmlns = "http://www.clicksoftware.com">  <Group>  <Name>Statuses that can be Cancelled by CallID</Name>  <GroupParameters>  <GroupParameter>A1234567</GroupParameter>  </GroupParameters>  <GroupType>2</GroupType>  </Group>  <EnableGroupOnTheFly>true</EnableGroupOnTheFly>  <RequestedProperties>  <Item>Key</Item>  <Item>CallID</Item>  <Item>Number</Item>  <Item>TaskID</Item>  <Item>MUSTJobNumber</Item>  </RequestedProperties>  <GetAssignments>false</GetAssignments>  </GetTasks>  </s:Body>  </s:Envelope> |

GetTasks Response:

|  |
| --- |
| <s:Envelope xmlns:s = "http://schemas.xmlsoap.org/soap/envelope/">  <s:Body>  <GetTasksResponse xmlns = "http://www.clicksoftware.com">  <Tasks></Tasks>  </GetTasksResponse>  </s:Body>  </s:Envelope> |

Note: This example returned no tasks, this is because the task was On Site and was not cancellable. The following MST examples will show more complex results, the same processes should be applied to single Tasks.

#### Training Task

By way of example for the Business scenarios 2.6.2.5 the SOAP stack for canceling a Training Task follows the following pattern:

* GetTasks
  + Using the “Statuses that can be Cancelled by TrainingJobNumber” group to teturn te relevant tasks;
    - TrainingJobNumber will return the Main Task
  + Or Using the “Statuses that can be Cancelled by CallID” group to return te relevant tasks;
    - CallID should be used to return the Training task;
* ProcessTaskEx – for returned Tasks;
  + Each Task returned by GetTasks needs cancelling;

Note: It may be more convenient to gather training tasks from MUST using the TrainingJobNumber, so the TrainingJobNumber has been set as a group and included in these examples.

GetTasks Request:

|  |
| --- |
| <s:Envelope xmlns:s = "http://schemas.xmlsoap.org/soap/envelope/">  <s:Header>  <h:OptionalParameters  xmlns:h = "http://www.clicksoftware.com/OptionalParameters"  xmlns = "http://www.clicksoftware.com/OptionalParameters"  xmlns:xsi = "http://www.w3.org/2001/XMLSchema-instance"  xmlns:xsd = "http://www.w3.org/2001/XMLSchema">  <CallerIdentity>MUST</CallerIdentity>  <ErrorOnNonExistingDictionaries>false</ErrorOnNonExistingDictionaries>  </h:OptionalParameters>  </s:Header>  <s:Body xmlns:xsi = "http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd = "http://www.w3.org/2001/XMLSchema">  <GetTasks xmlns = "http://www.clicksoftware.com">  **<Group>**  **<Name>Statuses that can be Cancelled by TrainingJobNumber</Name>**  **<GroupParameters>**  **<GroupParameter>A1111111</GroupParameter>**  **</GroupParameters>**  **<GroupType>2</GroupType>**  **<Body><CONDITION TYPE=""100"" TYPE\_TEXT=""Logic Condition""><OPERATION TYPE=""500"" TYPE\_TEXT=""And"" /><CONDITION TYPE=""100"" TYPE\_TEXT=""Logic Condition""><OPERATION TYPE=""501"" TYPE\_TEXT=""Or"" /><CONDITION TYPE=""101"" TYPE\_TEXT=""Arithmetic Condition""><VALUE TYPE=""200"" TYPE\_TEXT=""Property Value""><ID>9</ID></VALUE><OPERATION TYPE=""600"" TYPE\_TEXT=""="" /><VALUE TYPE=""201"" TYPE\_TEXT=""Const Value"">124135424</VALUE></CONDITION><CONDITION TYPE=""101"" TYPE\_TEXT=""Arithmetic Condition""><VALUE TYPE=""200"" TYPE\_TEXT=""Property Value""><ID>9</ID></VALUE><OPERATION TYPE=""600"" TYPE\_TEXT=""="" /><VALUE TYPE=""201"" TYPE\_TEXT=""Const Value"">124135426</VALUE></CONDITION><CONDITION TYPE=""101"" TYPE\_TEXT=""Arithmetic Condition""><VALUE TYPE=""200"" TYPE\_TEXT=""Property Value""><ID>9</ID></VALUE><OPERATION TYPE=""600"" TYPE\_TEXT=""="" /><VALUE TYPE=""201"" TYPE\_TEXT=""Const Value"">124149760</VALUE></CONDITION><CONDITION TYPE=""101"" TYPE\_TEXT=""Arithmetic Condition""><VALUE TYPE=""200"" TYPE\_TEXT=""Property Value""><ID>9</ID></VALUE><OPERATION TYPE=""600"" TYPE\_TEXT=""="" /><VALUE TYPE=""201"" TYPE\_TEXT=""Const Value"">124149763</VALUE></CONDITION><CONDITION TYPE=""101"" TYPE\_TEXT=""Arithmetic Condition""><VALUE TYPE=""200"" TYPE\_TEXT=""Property Value""><ID>9</ID></VALUE><OPERATION TYPE=""600"" TYPE\_TEXT=""="" /><VALUE TYPE=""201"" TYPE\_TEXT=""Const Value"">124149762</VALUE></CONDITION></CONDITION><CONDITION TYPE=""101"" TYPE\_TEXT=""Arithmetic Condition""><VALUE TYPE=""200"" TYPE\_TEXT=""Property Value""><ID>128</ID></VALUE><OPERATION TYPE=""600"" TYPE\_TEXT=""="" /><VALUE TYPE=""201"" TYPE\_TEXT=""Const Value"">%1</VALUE></CONDITION></CONDITION></Body>**  **</Group>**  <EnableGroupOnTheFly>true</EnableGroupOnTheFly>  <RequestedProperties>  <Item>Key</Item>  <Item>CallID</Item>  <Item>Number</Item>  <Item>TaskID</Item>  <Item>MUSTJobNumber</Item>  </RequestedProperties>  <GetAssignments>false</GetAssignments>  </GetTasks>  </s:Body>  </s:Envelope> |

GetTasks Response:

|  |
| --- |
| <s:Envelope xmlns:s = "http://schemas.xmlsoap.org/soap/envelope/">  <s:Body>  <GetTasksResponse xmlns = "http://www.clicksoftware.com">  <Tasks>  <Task d3p1:type = "Task" xmlns:d3p1 = "http://www.w3.org/2001/XMLSchema-instance">  <Key>39516374</Key>  <CallID>GB-A1234570-1</CallID>  <Number>2</Number>  <TaskID>FR-A1111111-1\_D4\_20130926T073852</TaskID>  <MUSTJobNumber>A1111111</MUSTJobNumber>  </Task>  </Tasks>  </GetTasksResponse>  </s:Body>  </s:Envelope> |

The following series of ProcessTaskEx requests pass the task.Stats Cancelled to the Tasks retured in the response of the prior GetTasks request:

Note: Any Tasks with a non canciable status will not be returned by the GetTasks & therfore wont be cacnclled.

|  |
| --- |
| <s:Envelope xmlns:s = "http://schemas.xmlsoap.org/soap/envelope/">  <s:Header>  <h:OptionalParameters  xmlns:h = "http://www.clicksoftware.com/OptionalParameters"  xmlns = "http://www.clicksoftware.com/OptionalParameters"  xmlns:xsi = "http://www.w3.org/2001/XMLSchema-instance"  xmlns:xsd = "http://www.w3.org/2001/XMLSchema">  <CallerIdentity>MUST</CallerIdentity>  <ErrorOnNonExistingDictionaries>false</ErrorOnNonExistingDictionaries>  </h:OptionalParameters>  </s:Header>  <s:Body xmlns:xsi = "http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd = "http://www.w3.org/2001/XMLSchema">  <ProcessTaskEx xmlns = "http://www.clicksoftware.com">  <Task>  <Key>39516374</Key>  <CallID>GB-A1234570-1</CallID>  <Number>2</Number>  <Status>  <Name>Cancelled</Name>  </Status>  <CancellationReason>  <Name>Other</Name>  </CancellationReason>  </Task>  <ReturnAssignment>false</ReturnAssignment>  <TaskRequestedProperties>  <Item>Key</Item>  <Item>CallID</Item>  <Item>Number</Item>  <Item>TaskID</Item>  <Item>MUSTJobNumber</Item>  </TaskRequestedProperties>  <ReturnSchedulingError>false</ReturnSchedulingError>  </ProcessTaskEx>  </s:Body>  </s:Envelope> |

#### Multi Manning Task

By way of example for the Business scenarios 2.6.2.5 the SOAP stack for canceling a multi man follows the following pattern:

* GetTasks
  + Using the “Statuses that can be Cancelled by CallID” group to return te relevant tasks;
* ProcessTaskEx – for returned Tasks;
  + Each Task returned by GetTasks needs cancelling

GetTasks Request:

|  |
| --- |
| <s:Envelope xmlns:s = "http://schemas.xmlsoap.org/soap/envelope/">  <s:Header>  <h:OptionalParameters  xmlns:h = "http://www.clicksoftware.com/OptionalParameters"  xmlns = "http://www.clicksoftware.com/OptionalParameters"  xmlns:xsi = "http://www.w3.org/2001/XMLSchema-instance"  xmlns:xsd = "http://www.w3.org/2001/XMLSchema">  <CallerIdentity>MUST</CallerIdentity>  <ErrorOnNonExistingDictionaries>false</ErrorOnNonExistingDictionaries>  </h:OptionalParameters>  </s:Header>  <s:Body xmlns:xsi = "http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd = "http://www.w3.org/2001/XMLSchema">  <GetTasks xmlns = "http://www.clicksoftware.com">  **<Group>**  **<Name>Statuses that can be Cancelled by CallID</Name>**  **<GroupParameters>**  **<GroupParameter>A1234569</GroupParameter>**  **</GroupParameters>**  **<GroupType>2</GroupType>**  **<Body><CONDITION TYPE=""100"" TYPE\_TEXT=""Logic Condition""><OPERATION TYPE=""500"" TYPE\_TEXT=""And"" /><CONDITION TYPE=""100"" TYPE\_TEXT=""Logic Condition""><OPERATION TYPE=""501"" TYPE\_TEXT=""Or"" /><CONDITION TYPE=""101"" TYPE\_TEXT=""Arithmetic Condition""><VALUE TYPE=""200"" TYPE\_TEXT=""Property Value""><ID>9</ID></VALUE><OPERATION TYPE=""600"" TYPE\_TEXT=""="" /><VALUE TYPE=""201"" TYPE\_TEXT=""Const Value"">124135424</VALUE></CONDITION><CONDITION TYPE=""101"" TYPE\_TEXT=""Arithmetic Condition""><VALUE TYPE=""200"" TYPE\_TEXT=""Property Value""><ID>9</ID></VALUE><OPERATION TYPE=""600"" TYPE\_TEXT=""="" /><VALUE TYPE=""201"" TYPE\_TEXT=""Const Value"">124135426</VALUE></CONDITION><CONDITION TYPE=""101"" TYPE\_TEXT=""Arithmetic Condition""><VALUE TYPE=""200"" TYPE\_TEXT=""Property Value""><ID>9</ID></VALUE><OPERATION TYPE=""600"" TYPE\_TEXT=""="" /><VALUE TYPE=""201"" TYPE\_TEXT=""Const Value"">124149760</VALUE></CONDITION><CONDITION TYPE=""101"" TYPE\_TEXT=""Arithmetic Condition""><VALUE TYPE=""200"" TYPE\_TEXT=""Property Value""><ID>9</ID></VALUE><OPERATION TYPE=""600"" TYPE\_TEXT=""="" /><VALUE TYPE=""201"" TYPE\_TEXT=""Const Value"">124149763</VALUE></CONDITION><CONDITION TYPE=""101"" TYPE\_TEXT=""Arithmetic Condition""><VALUE TYPE=""200"" TYPE\_TEXT=""Property Value""><ID>9</ID></VALUE><OPERATION TYPE=""600"" TYPE\_TEXT=""="" /><VALUE TYPE=""201"" TYPE\_TEXT=""Const Value"">124149762</VALUE></CONDITION></CONDITION><CONDITION TYPE=""101"" TYPE\_TEXT=""Arithmetic Condition""><VALUE TYPE=""200"" TYPE\_TEXT=""Property Value""><ID>3</ID></VALUE><OPERATION TYPE=""600"" TYPE\_TEXT=""="" /><VALUE TYPE=""201"" TYPE\_TEXT=""Const Value"">%1</VALUE></CONDITION></CONDITION></Body>**  **</Group>**  <EnableGroupOnTheFly>true</EnableGroupOnTheFly>  <RequestedProperties>  <Item>Key</Item>  <Item>CallID</Item>  <Item>Number</Item>  <Item>TaskID</Item>  <Item>MUSTJobNumber</Item>  </RequestedProperties>  <GetAssignments>false</GetAssignments>  </GetTasks>  </s:Body>  </s:Envelope> |

GetTasks Response:

|  |
| --- |
| <s:Envelope xmlns:s = "http://schemas.xmlsoap.org/soap/envelope/">  <s:Body>  <GetTasksResponse xmlns = "http://www.clicksoftware.com">  <Tasks>  <Task d3p1:type = "Task" xmlns:d3p1 = "http://www.w3.org/2001/XMLSchema-instance">  <Key>39516370</Key>  <CallID>GB-A1234569-1</CallID>  <Number>1</Number>  <TaskID>FR-A1234569-1\_D4\_20130926T073730</TaskID>  <MUSTJobNumber>A1234569</MUSTJobNumber>  </Task>  <Task d3p1:type = "Task" xmlns:d3p1 = "http://www.w3.org/2001/XMLSchema-instance">  <Key>39516371</Key>  <CallID>GB-A1234569-1</CallID>  <Number>2</Number>  <TaskID>FR-A1234569-1\_D4\_20130926T073830</TaskID>  <MUSTJobNumber>A1234569</MUSTJobNumber>  </Task>  <Task d3p1:type = "Task" xmlns:d3p1 = "http://www.w3.org/2001/XMLSchema-instance">  <Key>39516372</Key>  <CallID>GB-A1234569-1</CallID>  <Number>3</Number>  <TaskID>FR-A1234569-1\_D4\_20130926T073930</TaskID>  <MUSTJobNumber>A1234569</MUSTJobNumber>  </Task>  </Tasks>  </GetTasksResponse>  </s:Body>  </s:Envelope> |

The following series of ProcessTaskEx requests pass the task.Stats Cancelled to the Tasks returned in the response of the prior GetTasks request:

Note: Any Tasks with a non canciable status will not be returned by the GetTasks & therefore wont be cancelled.

|  |
| --- |
| <s:Envelope xmlns:s = "http://schemas.xmlsoap.org/soap/envelope/">  <s:Header>  <h:OptionalParameters  xmlns:h = "http://www.clicksoftware.com/OptionalParameters"  xmlns = "http://www.clicksoftware.com/OptionalParameters"  xmlns:xsi = "http://www.w3.org/2001/XMLSchema-instance"  xmlns:xsd = "http://www.w3.org/2001/XMLSchema">  <CallerIdentity>MUST</CallerIdentity>  <ErrorOnNonExistingDictionaries>false</ErrorOnNonExistingDictionaries>  </h:OptionalParameters>  </s:Header>  <s:Body xmlns:xsi = "http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd = "http://www.w3.org/2001/XMLSchema">  <ProcessTaskEx xmlns = "http://www.clicksoftware.com">  <Task>  <Key>39516370</Key>  <CallID>GB-A1234569-1</CallID>  <Number>1</Number>  <Status>  <Name>Cancelled</Name>  </Status>  <CancellationReason>  <Name>Other</Name>  </CancellationReason>  </Task>  <ReturnAssignment>false</ReturnAssignment>  <TaskRequestedProperties>  <Item>Key</Item>  <Item>CallID</Item>  <Item>Number</Item>  <Item>TaskID</Item>  <Item>MUSTJobNumber</Item>  </TaskRequestedProperties>  <ReturnSchedulingError>false</ReturnSchedulingError>  </ProcessTaskEx>  </s:Body>  </s:Envelope> |

|  |
| --- |
| <s:Envelope xmlns:s = "http://schemas.xmlsoap.org/soap/envelope/">  <s:Header>  <h:OptionalParameters  xmlns:h = "http://www.clicksoftware.com/OptionalParameters"  xmlns = "http://www.clicksoftware.com/OptionalParameters"  xmlns:xsi = "http://www.w3.org/2001/XMLSchema-instance"  xmlns:xsd = "http://www.w3.org/2001/XMLSchema">  <CallerIdentity>MUST</CallerIdentity>  <ErrorOnNonExistingDictionaries>false</ErrorOnNonExistingDictionaries>  </h:OptionalParameters>  </s:Header>  <s:Body xmlns:xsi = "http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd = "http://www.w3.org/2001/XMLSchema">  <ProcessTaskEx xmlns = "http://www.clicksoftware.com">  <Task>  <Key>39516371</Key>  <CallID>GB-A1234569-1</CallID>  <Number>2</Number>  <Status>  <Name>Cancelled</Name>  </Status>  <CancellationReason>  <Name>Other</Name>  </CancellationReason>  </Task>  <ReturnAssignment>false</ReturnAssignment>  <TaskRequestedProperties>  <Item>Key</Item>  <Item>CallID</Item>  <Item>Number</Item>  <Item>TaskID</Item>  <Item>MUSTJobNumber</Item>  </TaskRequestedProperties>  <ReturnSchedulingError>false</ReturnSchedulingError>  </ProcessTaskEx>  </s:Body>  </s:Envelope> |

|  |
| --- |
| <s:Envelope xmlns:s = "http://schemas.xmlsoap.org/soap/envelope/">  <s:Header>  <h:OptionalParameters  xmlns:h = "http://www.clicksoftware.com/OptionalParameters"  xmlns = "http://www.clicksoftware.com/OptionalParameters"  xmlns:xsi = "http://www.w3.org/2001/XMLSchema-instance"  xmlns:xsd = "http://www.w3.org/2001/XMLSchema">  <CallerIdentity>MUST</CallerIdentity>  <ErrorOnNonExistingDictionaries>false</ErrorOnNonExistingDictionaries>  </h:OptionalParameters>  </s:Header>  <s:Body xmlns:xsi = "http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd = "http://www.w3.org/2001/XMLSchema">  <ProcessTaskEx xmlns = "http://www.clicksoftware.com">  <Task>  <Key>39516372</Key>  <CallID>GB-A1234569-1</CallID>  <Number>3</Number>  <Status>  <Name>Cancelled</Name>  </Status>  <CancellationReason>  <Name>Other</Name>  </CancellationReason>  </Task>  <ReturnAssignment>false</ReturnAssignment>  <TaskRequestedProperties>  <Item>Key</Item>  <Item>CallID</Item>  <Item>Number</Item>  <Item>TaskID</Item>  <Item>MUSTJobNumber</Item>  </TaskRequestedProperties>  <ReturnSchedulingError>false</ReturnSchedulingError>  </ProcessTaskEx>  </s:Body>  </s:Envelope> |

ProcessTaskExResponse:

|  |
| --- |
| <s:Envelope xmlns:s = "http://schemas.xmlsoap.org/soap/envelope/">  <s:Body>  <ProcessTaskExResponse xmlns = "http://www.clicksoftware.com">  <ReturnCode>Success</ReturnCode>  <Task>  <Key>39516372</Key>  <CallID>GB-A1234569-1</CallID>  <Number>3</Number>  <TaskID>FR-A1234569-1\_D4\_20130926T073930</TaskID>  <MUSTJobNumber>A1234569</MUSTJobNumber>  </Task>  <LinkedTasks>  <AssignedTask>  <Task>  <Key>39516370</Key>  <CallID>GB-A1234569-1</CallID>  <Number>1</Number>  <TaskID>FR-A1234569-1\_D4\_20130926T073730</TaskID>  <MUSTJobNumber>A1234569</MUSTJobNumber>  </Task>  </AssignedTask>  <AssignedTask>  <Task>  <Key>39516371</Key>  <CallID>GB-A1234569-1</CallID>  <Number>2</Number>  <TaskID>FR-A1234569-1\_D4\_20130926T073830</TaskID>  <MUSTJobNumber>A1234569</MUSTJobNumber>  </Task>  </AssignedTask>  </LinkedTasks>  </ProcessTaskExResponse>  </s:Body>  </s:Envelope> |

#### Parts Pickup Task

Cancelling Parts Pickup Tasks is a variation on a theme of the Multi Manning process, see 3.5.7.4 for the process and SOAP syntax.

## ExecuteMultiOperations

For creation and linking of the GEHCJobComments to Tasks use the ExecuteMultiOperations action with two UpdateOrCreate Operations per Task.

For new Tasks pass the Operation UpdateOrCreate GEHCJobComments Object, a blank GEHCJobComments object with therefor be created and linked to the Task for future updates. GEHCJobComments objects that exist for the MUSTJobNumber will be linked and remain unchanged.

Rules to follow for individual SOAP payloads:

1. One GEHCJobComments UpdateOrCreate Object per distinct MUSTJobNumber;
   1. Multiple GEHCJobComments can be referenced only when following tasks have MUSTJobMumbers that differ from each other;
2. Task objects should follow UpdateOrCreate Objects;
3. Dependencies should always be on the main task;
4. Parts Pickup tasks shouldn’t have job comments assigned to the pickup task;
5. Training tasks should have separate JobComments as the main Task;

**Note: When updating a GEHCJobComments that’s already in the system and linked to future Tasks it is recommended that the existing Tasks are relinked to the GEHCJobComment. This will force Click Mobile to push the updated comment to the Engineers device. Use GetTasks 3.7 action to identify existing tasks and include these future Tasks along side the current task Update.**

#### Message Fields

| Property | Visible Name | Owner |
| --- | --- | --- |
| Object[‘GEHCJobComments’] | Action  “UpdateOrCreate” | This will Update or Create a new comments record.  **Keep this value unchanged as “UpdateOrCreate”.** |
| Object[‘GEHCJobComments’] | MUSTJobNumber  “A1234567” | This locates the comments record by MUSTJobNumber, multiple Tasks can share the same comments. |
| Object[‘GEHCJobComments’] | Text  “Very long text” | Unlimited text string of the the actual comments. |
| Object[‘Task’] | Action  “Update” | Will update the specified Task with the MUSTJobNumber key of the comments record of the first object.  Keep this value unchanged as “Update”. |
| Object[‘Task’] | CallID  “20130913\_0” | Required Task UNID. |
| Object[‘Task’] | Number  “1” | Required Task UNID. |
| Object[‘Task’]/JobComments | MUSTJobNumber  “A1234567” | The MUSTJobNumber to be linked to the task, the same value referenced in the first object. |

#### Request Header

|  |  |
| --- | --- |
| URL | SOAP Action |
| http://<server>/SO/IntegrationServices/ ServiceOptimizationService.svc /BasicHttpInt | http://www.clicksoftware.com/ ServiceOptimizationService.svc / ExecuteMultiOperations |

Below is an example SOAP Request Header POST & SOAPAction are the vital in the packet construct:

|  |
| --- |
| **POST http://localhost/SO/IntegrationServices/ServiceOptimizationService.svc/BasicHttpInt HTTP/1.1**  Content-Type: text/xml; charset=utf-8  VsDebuggerCausalityData: uIDPo4+iEsiarM5Bqxl4Uz61/84AAAAAy3iFgXz+90yD1GTUa6LjRAobIse7uFVNoS/p6G5/rBwACQAA  **SOAPAction: "http://www.clicksoftware.com/ServiceOptimizationService/ExecuteMultipleOperations"**  Host: localhost  Content-Length: 1163  Expect: 100-continue  Accept-Encoding: gzip, deflate |

#### Request Message

|  |
| --- |
| <s:Envelope xmlns:s = "http://schemas.xmlsoap.org/soap/envelope/">  <s:Header>  <h:OptionalParameters  xmlns:h = "http://www.clicksoftware.com/OptionalParameters"  xmlns = "http://www.clicksoftware.com/OptionalParameters"  xmlns:xsi = "http://www.w3.org/2001/XMLSchema-instance"  xmlns:xsd = "http://www.w3.org/2001/XMLSchema">  <CallerIdentity>MUSTMB</CallerIdentity>  <ErrorOnNonExistingDictionaries>true</ErrorOnNonExistingDictionaries>  </h:OptionalParameters>  </s:Header>  <s:Body xmlns:xsi = "http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd = "http://www.w3.org/2001/XMLSchema">  <ExecuteMultipleOperations xmlns = "http://www.clicksoftware.com">  <Operations>  <Operation>  <Action>UpdateOrCreate</Action>  <Object>  <Object xsi:type = "GEHCJobComments">  <MUSTJobNumber>A1234570</MUSTJobNumber>  <Text>TrainingTask: Very long text</Text>  </Object>  </Object>  </Operation>  <Operation>  <Action>Update</Action>  <Object>  <Object xsi:type = "Task">  <CallID>GB-A1234570-1</CallID>  <Number>1</Number>  <TimeDependencies>  <TaskTimeDependency>  <TaskKey>  <CallID>GB-A1234570-1</CallID>  <Number>2</Number>  </TaskKey>  <RelationType>1</RelationType>  <RelationOperator>2</RelationOperator>  <UpperBound>0</UpperBound>  <LowerBound>0</LowerBound>  <Critical>true</Critical>  </TaskTimeDependency>  </TimeDependencies>  <EngineerDependencies>  <TaskEngineerDependency>  <TaskKey>  <CallID>GB-A1234570-1</CallID>  <Number>2</Number>  </TaskKey>  <RelationType>2</RelationType>  </TaskEngineerDependency>  </EngineerDependencies>  <JobComments>  <MUSTJobNumber>A1234570</MUSTJobNumber>  </JobComments>  </Object>  </Object>  </Operation>  <Operation>  <Action>UpdateOrCreate</Action>  <Object>  <Object xsi:type = "GEHCJobComments">  <MUSTJobNumber>A1111111</MUSTJobNumber>  <Text>TrainingTask: Very long text</Text>  </Object>  </Object>  </Operation>  <Operation>  <Action>Update</Action>  <Object>  <Object xsi:type = "Task">  <CallID>GB-A1234570-1</CallID>  <Number>2</Number>  <JobComments>  <MUSTJobNumber>A1111111</MUSTJobNumber>  </JobComments>  </Object>  </Object>  </Operation>  </Operations>  <OneTransaction>false</OneTransaction>  <ContinueOnError>false</ContinueOnError>  </ExecuteMultipleOperations>  </s:Body>  </s:Envelope> |

#### Response Message

On success, a message will be returned with the GEHCJobComments & Task data according to the settings of the TaskRequestedProperties in the request message, with no SOAP errors.

|  |
| --- |
| <s:Envelope xmlns:s = "http://schemas.xmlsoap.org/soap/envelope/">  <s:Body>  <ExecuteMultipleOperationsResponse xmlns = "http://www.clicksoftware.com">  <Operations>  <Operation d3p1:type = "SucceededOperation" xmlns:d3p1 = "http://www.w3.org/2001/XMLSchema-instance">  <Action>Update</Action>  <Object>  <ObjectReference  Key = "1166082050"  DisplayString = "A1234570"  d3p1:type = "GEHCJobCommentsReference">A1234570</ObjectReference>  </Object>  </Operation>  <Operation d3p1:type = "SucceededOperation" xmlns:d3p1 = "http://www.w3.org/2001/XMLSchema-instance">  <Action>Update</Action>  <Object>  <ObjectReference  Key = "39498624"  DisplayString = "A1234570"  d3p1:type = "TaskReference">  <CallID>GB-A1234570-1</CallID>  <Number>1</Number>  </ObjectReference>  </Object>  </Operation>  <Operation d3p1:type = "SucceededOperation" xmlns:d3p1 = "http://www.w3.org/2001/XMLSchema-instance">  <Action>Update</Action>  <Object>  <ObjectReference  Key = "1166082051"  DisplayString = "A1111111"  d3p1:type = "GEHCJobCommentsReference">A1111111</ObjectReference>  </Object>  </Operation>  <Operation d3p1:type = "SucceededOperation" xmlns:d3p1 = "http://www.w3.org/2001/XMLSchema-instance">  <Action>Update</Action>  <Object>  <ObjectReference  Key = "39498625"  DisplayString = "A1234570"  d3p1:type = "TaskReference">  <CallID>GB-A1234570-1</CallID>  <Number>2</Number>  </ObjectReference>  </Object>  </Operation>  </Operations>  </ExecuteMultipleOperationsResponse>  </s:Body>  </s:Envelope> |

## GetTasks

Relevant for checking a task status for the cancelation of tasks in MUST see, 2.6.3.3

#### Request Header

|  |  |
| --- | --- |
| URL | SOAP Action |
| http://<server>/SO/IntegrationServices/ ScheduleService.svc /BasicHttpInt | http://www.clicksoftware.com/ ScheduleService.svc / GetTasks |

Below is an example SOAP Request Header POST & SOAPAction are the vital in the packet construct:

|  |
| --- |
| **POST http://localhost/SO/IntegrationServices/ScheduleService.svc/BasicHttpInt HTTP/1.1**  Content-Type: text/xml; charset=utf-8  VsDebuggerCausalityData: uIDPo6yjCfo2OcBNrx5NYnBiK28AAAAA3IZra5vu906MJE4Q0k96BkAVgUDZvoNJlsf45h1WYYEACQAA  **SOAPAction: "http://www.clicksoftware.com/ScheduleService/GetTasks"**  Host: localhost  Content-Length: 1069  Expect: 100-continue  Accept-Encoding: gzip, deflate |

#### Request Message

The Indexes objects filter the required value:

|  |
| --- |
| <s:Envelope xmlns:s = "http://schemas.xmlsoap.org/soap/envelope/">  <s:Header>  <h:OptionalParameters  xmlns:h = "http://www.clicksoftware.com/OptionalParameters"  xmlns = "http://www.clicksoftware.com/OptionalParameters"  xmlns:xsi = "http://www.w3.org/2001/XMLSchema-instance"  xmlns:xsd = "http://www.w3.org/2001/XMLSchema">  <CallerIdentity>MUST</CallerIdentity>  <ErrorOnNonExistingDictionaries>false</ErrorOnNonExistingDictionaries>  </h:OptionalParameters>  </s:Header>  <s:Body xmlns:xsi = "http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd = "http://www.w3.org/2001/XMLSchema">  <GetTasks xmlns = "http://www.clicksoftware.com">  <Indexes>  **<Index>**  **<LowBound>**  **<Property>**  **<Name>CallID</Name>**  **<Value>A1234569</Value>**  **</Property>**  **</LowBound>**  **<HighBound>**  **<Property>**  **<Name>CallID</Name>**  **<Value>A1234569</Value>**  **</Property>**  **</HighBound>**  **</Index>**  **</Indexes>**  <EnableGroupOnTheFly>false</EnableGroupOnTheFly>  <RequestedProperties>  <Item>Key</Item>  <Item>CallID</Item>  <Item>Number</Item>  <Item>TaskID</Item>  <Item>MUSTJobNumber</Item>  </RequestedProperties>  <GetAssignments>false</GetAssignments>  </GetTasks>  </s:Body>  </s:Envelope> |

#### Request Message non-cachd

Should te above message not work a non-cashed approach can be used, alter the body as required with the following body values:



|  |
| --- |
| <s:Envelope xmlns:s = "http://schemas.xmlsoap.org/soap/envelope/">  <s:Header>  <h:OptionalParameters  xmlns:h = "http://www.clicksoftware.com/OptionalParameters"  xmlns = "http://www.clicksoftware.com/OptionalParameters"  xmlns:xsi = "http://www.w3.org/2001/XMLSchema-instance"  xmlns:xsd = "http://www.w3.org/2001/XMLSchema">  <CallerIdentity>MUST</CallerIdentity>  <ErrorOnNonExistingDictionaries>false</ErrorOnNonExistingDictionaries>  </h:OptionalParameters>  </s:Header>  <s:Body xmlns:xsi = "http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd = "http://www.w3.org/2001/XMLSchema">  <GetTasks xmlns = "http://www.clicksoftware.com">  <Group>  <Body>&lt;CONDITION TYPE="100" TYPE\_TEXT="Logic Condition"&gt;&lt;OPERATION TYPE="500" TYPE\_TEXT="And" /&gt;&lt;CONDITION TYPE="100" TYPE\_TEXT="Logic Condition"&gt;&lt;OPERATION TYPE="501" TYPE\_TEXT="Or" /&gt;&lt;CONDITION TYPE="101" TYPE\_TEXT="Arithmetic Condition"&gt;&lt;VALUE TYPE="200" TYPE\_TEXT="Property Value"&gt;&lt;ID&gt;9&lt;/ID&gt;&lt;/VALUE&gt;&lt;OPERATION TYPE="600" TYPE\_TEXT="=" /&gt;&lt;VALUE TYPE="201" TYPE\_TEXT="Const Value"&gt;124135424&lt;/VALUE&gt;&lt;/CONDITION&gt;&lt;CONDITION TYPE="101" TYPE\_TEXT="Arithmetic Condition"&gt;&lt;VALUE TYPE="200" TYPE\_TEXT="Property Value"&gt;&lt;ID&gt;9&lt;/ID&gt;&lt;/VALUE&gt;&lt;OPERATION TYPE="600" TYPE\_TEXT="=" /&gt;&lt;VALUE TYPE="201" TYPE\_TEXT="Const Value"&gt;124135426&lt;/VALUE&gt;&lt;/CONDITION&gt;&lt;CONDITION TYPE="101" TYPE\_TEXT="Arithmetic Condition"&gt;&lt;VALUE TYPE="200" TYPE\_TEXT="Property Value"&gt;&lt;ID&gt;9&lt;/ID&gt;&lt;/VALUE&gt;&lt;OPERATION TYPE="600" TYPE\_TEXT="=" /&gt;&lt;VALUE TYPE="201" TYPE\_TEXT="Const Value"&gt;124149760&lt;/VALUE&gt;&lt;/CONDITION&gt;&lt;CONDITION TYPE="101" TYPE\_TEXT="Arithmetic Condition"&gt;&lt;VALUE TYPE="200" TYPE\_TEXT="Property Value"&gt;&lt;ID&gt;9&lt;/ID&gt;&lt;/VALUE&gt;&lt;OPERATION TYPE="600" TYPE\_TEXT="=" /&gt;&lt;VALUE TYPE="201" TYPE\_TEXT="Const Value"&gt;124149763&lt;/VALUE&gt;&lt;/CONDITION&gt;&lt;CONDITION TYPE="101" TYPE\_TEXT="Arithmetic Condition"&gt;&lt;VALUE TYPE="200" TYPE\_TEXT="Property Value"&gt;&lt;ID&gt;9&lt;/ID&gt;&lt;/VALUE&gt;&lt;OPERATION TYPE="600" TYPE\_TEXT="=" /&gt;&lt;VALUE TYPE="201" TYPE\_TEXT="Const Value"&gt;124149762&lt;/VALUE&gt;&lt;/CONDITION&gt;&lt;/CONDITION&gt;&lt;CONDITION TYPE="101" TYPE\_TEXT="Arithmetic Condition"&gt;&lt;VALUE TYPE="200" TYPE\_TEXT="Property Value"&gt;&lt;ID&gt;3&lt;/ID&gt;&lt;/VALUE&gt;&lt;OPERATION TYPE="600" TYPE\_TEXT="=" /&gt;&lt;VALUE TYPE="201" TYPE\_TEXT="Const Value"&gt;%1&lt;/VALUE&gt;&lt;/CONDITION&gt;&lt;/CONDITION&gt;</Body>  <GroupParameters>  <GroupParameter>A1234567</GroupParameter>  </GroupParameters>  <GroupType>2</GroupType>  </Group>  <EnableGroupOnTheFly>true</EnableGroupOnTheFly>  <RequestedProperties>  <Item>Key</Item>  <Item>CallID</Item>  <Item>Number</Item>  <Item>TaskID</Item>  <Item>MUSTJobNumber</Item>  </RequestedProperties>  <GetAssignments>false</GetAssignments>  </GetTasks>  </s:Body>  </s:Envelope> |

#### Response Message

The response showing all Tasks with a common CallID, the TaskID is unique per job:

|  |
| --- |
| <s:Envelope xmlns:s = "http://schemas.xmlsoap.org/soap/envelope/">  <s:Body>  <GetTasksResponse xmlns = "http://www.clicksoftware.com">  <Tasks>  <Task d3p1:type = "Task" xmlns:d3p1 = "http://www.w3.org/2001/XMLSchema-instance">  <Key>39498619</Key>  <CallID>GB-A1234569-1</CallID>  <Number>1</Number>  **<TaskID>FR-A1234569-1\_D4\_20130920T062212</TaskID>**  <MUSTJobNumber>A1234569</MUSTJobNumber>  </Task>  <Task d3p1:type = "Task" xmlns:d3p1 = "http://www.w3.org/2001/XMLSchema-instance">  <Key>39498622</Key>  <CallID>GB-A1234569-1</CallID>  <Number>2</Number>  **<TaskID>FR-A1234569-1\_D4\_20130920T062312</TaskID>**  <MUSTJobNumber>A1234569</MUSTJobNumber>  </Task>  <Task d3p1:type = "Task" xmlns:d3p1 = "http://www.w3.org/2001/XMLSchema-instance">  <Key>39498623</Key>  <CallID>GB-A1234569-1</CallID>  <Number>3</Number>  **<TaskID>FR-A1234569-1\_D4\_20130920T062412</TaskID>**  <MUSTJobNumber>A1234569</MUSTJobNumber>  </Task>  </Tasks>  </GetTasksResponse>  </s:Body>  </s:Envelope> |

## GetResources

Relevant for checking an Engineer for validation in MUST SDT see, 2.4.5

#### Request Header

|  |  |
| --- | --- |
| URL | SOAP Action |
| http://<server>/SO/IntegrationServices/ ServiceOptimizationService.svc /BasicHttpInt | http://www.clicksoftware.com/ ServiceOptimizationService.svc / GetResources |

Below is an example SOAP Request Header POST & SOAPAction are the vital in the packet construct:

|  |
| --- |
| **POST http://localhost/SO/IntegrationServices/ServiceOptimizationService.svc/BasicHttpInt HTTP/1.1**  Content-Type: text/xml; charset=utf-8  VsDebuggerCausalityData: uIDPoxMv7Ia20XVEowaaG98v7MgAAAAAFU4MNu/D50mCm4NIqeeTjMSXXUUhI5ZLhXvzBx+m0GkACQAA  **SOAPAction: "http://www.clicksoftware.com/ServiceOptimizationService/GetResources"**  Host: localhost  Content-Length: 894  Expect: 100-continue  Accept-Encoding: gzip, deflate  Connection: Keep-Alive |

#### Request Message

The Indexes objects filter the required value:

|  |
| --- |
| <s:Envelope xmlns:s = "http://schemas.xmlsoap.org/soap/envelope/">  <s:Header>  <h:OptionalParameters  xmlns:h = "http://www.clicksoftware.com/OptionalParameters"  xmlns = "http://www.clicksoftware.com/OptionalParameters"  xmlns:xsi = "http://www.w3.org/2001/XMLSchema-instance"  xmlns:xsd = "http://www.w3.org/2001/XMLSchema">  <CallerIdentity>MUST</CallerIdentity>  <ErrorOnNonExistingDictionaries>true</ErrorOnNonExistingDictionaries>  </h:OptionalParameters>  </s:Header>  <s:Body xmlns:xsi = "http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd = "http://www.w3.org/2001/XMLSchema">  <GetResources xmlns = "http://www.clicksoftware.com">  <Indexes>  <Index>  **<LowBound>**  **<Property>**  **<Name>MUSTID</Name>**  **<Value>MUSTIDTrainer01</Value>**  **</Property>**  **</LowBound>**  **<HighBound>**  **<Property>**  **<Name>MUSTID</Name>**  **<Value>MUSTIDTrainer01</Value>**  **</Property>**  </HighBound>  </Index>  </Indexes>  <EnableGroupOnTheFly>false</EnableGroupOnTheFly>  </GetResources>  </s:Body>  </s:Envelope> |

Or for ID:

|  |
| --- |
| <s:Envelope xmlns:s = "http://schemas.xmlsoap.org/soap/envelope/">  <s:Header>  <h:OptionalParameters  xmlns:h = "http://www.clicksoftware.com/OptionalParameters"  xmlns = "http://www.clicksoftware.com/OptionalParameters"  xmlns:xsi = "http://www.w3.org/2001/XMLSchema-instance"  xmlns:xsd = "http://www.w3.org/2001/XMLSchema">  <CallerIdentity>MUST</CallerIdentity>  <ErrorOnNonExistingDictionaries>true</ErrorOnNonExistingDictionaries>  </h:OptionalParameters>  </s:Header>  <s:Body xmlns:xsi = "http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd = "http://www.w3.org/2001/XMLSchema">  <GetResources xmlns = "http://www.clicksoftware.com">  <Indexes>  **<Index>**  **<LowBound>**  **<Property>**  **<Name>ID</Name>**  **<Value>Trainer01</Value>**  **</Property>**  **</LowBound>**  **<HighBound>**  **<Property>**  **<Name>ID</Name>**  **<Value>Trainer01</Value>**  **</Property>**  **</HighBound>**  **</Index>**  </Indexes>  <EnableGroupOnTheFly>false</EnableGroupOnTheFly>  </GetResources>  </s:Body>  </s:Envelope> |

#### Response Message

The response showing the required Engineer:

|  |
| --- |
| <s:Envelope xmlns:s = "http://schemas.xmlsoap.org/soap/envelope/">  <s:Body>  <GetResourcesResponse xmlns = "http://www.clicksoftware.com">  <Engineers>  <Engineer d3p1:type = "Engineer" xmlns:d3p1 = "http://www.w3.org/2001/XMLSchema-instance">  <Key>789620736</Key>  <Revision>2</Revision>  <Stamp>  <CreatedBy>CLICKSOFTWARE\Duncan.Hardie</CreatedBy>  <TimeCreated>2013-10-01T09:50:08</TimeCreated>  <CreatingProcess>0</CreatingProcess>  <ModifiedBy>CLICKSOFTWARE\Duncan.Hardie</ModifiedBy>  <TimeModified>2013-10-01T09:53:00</TimeModified>  <ModifyingProcess>0</ModifyingProcess>  </Stamp>  <Name>Trainer01</Name>  <ID>Trainer01</ID>  <Region Key = "124022784" DisplayString = "UK">UK</Region>  <District Key = "124422144" DisplayString = "UK South">UK South</District>  <Postcode></Postcode>  <Calendar Key = "-1"></Calendar>  <EngineerType Key = "-1"></EngineerType>  <Tools></Tools>  <Active>1</Active>  <Skills></Skills>  <TravelSpeed>0</TravelSpeed>  <Internal>0</Internal>  <Efficiency>0</Efficiency>  <MobileClient>0</MobileClient>  <AvailabilityFactor>0</AvailabilityFactor>  <Latitude>0</Latitude>  <Longitude>0</Longitude>  <GISDataSource>0</GISDataSource>  <Street></Street>  <City></City>  <State></State>  <CountryID Key = "126116068" DisplayString = "UNITED KINGDOM">UNITED KINGDOM</CountryID>  <Contractor>0</Contractor>  <FixedTravel>0</FixedTravel>  <Company Key = "-1"></Company>  <Trainers></Trainers>  <Contract Key = "-1"></Contract>  <IgnoreAllPreferences>0</IgnoreAllPreferences>  <IgnoreFairnessCalculation>0</IgnoreFairnessCalculation>  <PreferenceApproved>0</PreferenceApproved>  <PreferenceApprovalDate>1899-12-30T00:00:00</PreferenceApprovalDate>  <RosterApproved>0</RosterApproved>  <RosterApprovalDate>1899-12-30T00:00:00</RosterApprovalDate>  <PeriodicEfficiencies></PeriodicEfficiencies>  <LunchBreakDuration>0</LunchBreakDuration>  <LunchStartsFrom>1899-12-30T00:00:00</LunchStartsFrom>  <HasDynamicData>0</HasDynamicData>  <LocationID></LocationID>  <Crew>0</Crew>  <MobileClientSettings Key = "-1"></MobileClientSettings>  <TimePhasedSkills></TimePhasedSkills>  <MobileWapClientSettings Key = "-1"></MobileWapClientSettings>  <LastAllocationFinish>1899-12-30T00:00:00</LastAllocationFinish>  <RelocationSource Key = "-1"></RelocationSource>  <CrewForExternalUse>0</CrewForExternalUse>  <MaxDistanceFromHB>0</MaxDistanceFromHB>  <StateSubdivision></StateSubdivision>  <CitySubdivision></CitySubdivision>  <SOLicenseInactive>0</SOLicenseInactive>  <MobileWebClientSettings Key = "-1"></MobileWebClientSettings>  <RequiredCrewSize>0</RequiredCrewSize>  <PartsStock Key = "-1"></PartsStock>  <MobilePhone></MobilePhone>  <ExternalRefID></ExternalRefID>  <Email></Email>  <LoginName></LoginName>  <ApproverLoginID></ApproverLoginID>  <TravelPolicy Key = "-1"></TravelPolicy>  <DispatchPolicy Key = "-1"></DispatchPolicy>  <SOUserGroup Key = "-1"></SOUserGroup>  <RelatedPartsLocation Key = "-1"></RelatedPartsLocation>  <MobileKey></MobileKey>  <Team Key = "-1"></Team>  <MUSTID>MUSTIDTrainer01</MUSTID>  <Characteristic></Characteristic>  <ServiceArea></ServiceArea>  <LunchStartsAfter>0</LunchStartsAfter>  <MaxTimeFromHB>0</MaxTimeFromHB>  <AllowedTaskTypes Key = "-1"></AllowedTaskTypes>  <ManagerID></ManagerID>  <ManagerName></ManagerName>  <ManagerContactNumber></ManagerContactNumber>  <ManagerEmail></ManagerEmail>  <Languages></Languages>  <FSELanguages></FSELanguages>  </Engineer>  </Engineers>  </GetResourcesResponse>  </s:Body>  </s:Envelope> |

# Error Handling

## Incoming Messages

When ClickSchedule receives a message, it returns an immediate result (Synchronously) indicating whether the operation has succeeded or failed. It is extremely important that the GEHC / MUST will check the returned message in order to know whether the action did, or did not, take place in ClickSchedule. Possible errors that the GEHC systems might receive include: ClickSchedule server is down, Unknown Task/Engineer, Database is down, etc.

#### Successful Message

When no error occurs the message response is returned with no error. For example a successful createResource call, see for example.

#### Messages returned with Error – General

On error, the following message will be returned within the result message:

|  |
| --- |
| <s:Envelope >  …  <faultcode>Error code</faultcode>  <detail>  <string Error description  …  </string>  …  </s:Envelope> |

For example when trying to update a task with non-existing call ID:

|  |
| --- |
| <s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/">  <s:Body>  <s:Fault>  <faultcode>s:ServiceOptimization.ErrorNumbers.55560</faultcode>  <faultstring xml:lang="en-US">Null or empty configuration</faultstring>  <detail>  <string xmlns="http://schemas.microsoft.com/2003/10/Serialization/">W6ServiceHost.Bindings.W6Exception: Null or empty configuration  at W6ServiceHost.W6MessageProcessor.HandleSXPErrors(String sxpResponseBody, Boolean isSxpContext)  at W6ServiceHost.W6MessageProcessor.ProcessMessage(String requestBody, String processorTypeName, Boolean isSxpContext, W6MessageSettingProperty messageSettings)  at W6ServiceHost.W6MessageProcessor.ProcessMessage(Message requestMessage, String contractNamespace, String contractName)  at W6Services.W6ServiceBase`1.ProcessMessage(Message message)  at SyncInvokeProcessMessage(Object , Object[] , Object[] )  at System.ServiceModel.Dispatcher.SyncMethodInvoker.Invoke(Object instance, Object[] inputs, Object[]&amp; outputs)  at W6ServiceHost.Behaviors.W6OperationInvoker.Invoke(Object instance, Object[] inputs, Object[]&amp; outputs)  Service Optimization Error:  Error Number: 55560  Error Description: Null or empty configuration  Error Source:  at W6ThrowComException(Int32 rc, CStringT&lt;wchar\_t\,StrTraitMFC\_DLL&lt;wchar\_t\,ATL::ChTraitsCRT&lt;wchar\_t&gt; &gt; &gt;\* strDescription, CStringT&lt;wchar\_t\,StrTraitMFC\_DLL&lt;wchar\_t\,ATL::ChTraitsCRT&lt;wchar\_t&gt; &gt; &gt;\* strSource)  at W6ThrowComException(CW6Exception\* exp)  at W6XMLProcessor.W6SXPTaskOperations.AddOneTask(CW6InMsgProcessor\* theParser, CW6InMsgProcessor\* taskParser, Boolean bMainTask)  at W6XMLProcessor.W6SXPTaskOperations.ManipulateTasksAndAssignments(CW6InMsgProcessor\* theParser)  at W6XMLProcessor.W6SXPTaskOperations.ProcessMessage(String strMessageBody)  </string>  </detail>  </s:Fault>  </s:Body>  </s:Envelope> |

#### Message sent with inconsistent data

Technically a message with missing data or pieces of data that do not make sense with other can be sent and will be accepted in SO. There will be no special validations or consistency checks on the CKSW side (for example Task with Job Type and Equipment type that do not fit together can go through).

The assumption is that the sender knows what he is doing and that checks are already performed on the sending side.

#### Message sent with non-existing dictionary items

Sender must prevent creation of objects containing non-existing dictionary items. This shall be done by using the ErrorOnNonExistingDictionaries tag within the message soap header. This will instruct SO to avoid creation of missing dictionary items “on the fly”.

For example:

|  |
| --- |
| <s:Header >  …  <opt:OptionalParameters xmlns:opt="http://www.clicksoftware.com/OptionalParameters">  <opt:ErrorOnNonExistingDictionaries>true</opt:ErrorOnNonExistingDictionaries>  </opt:OptionalParameters>  ….  </s:Header> |

With this SO will fail the creation of the object. For example:

|  |
| --- |
| <ProcessTaskEx\_Request\_0>  …  <s:Header >  <opt:OptionalParameters xmlns:opt="http://www.clicksoftware.com/OptionalParameters">  <opt:ErrorOnNonExistingDictionaries> true</opt:ErrorOnNonExistingDictionaries>  </opt:OptionalParameters>  </s:Header>  …  <Task>  <CallID>GB-SR-2013-0610-test-1</CallID>  <TaskOwner>ED</TaskOwner>  <Active>1</Active>  <Priority>2</Priority>  <WorkOrderType>Incident</WorkOrderType>  <EquipmentType>Incident Transmission 1</EquipmentType>  <Postcode>9711 KN</Postcode>  </Task>  </ProcessTaskEx>  </s:Body>  </s:Envelope>  </RequestMessage>  </ProcessTaskEx\_Request\_0> |

Response:

|  |
| --- |
| <s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/">  <s:Body>  <s:Fault>  <faultcode>s:ServiceOptimization.ErrorNumbers.20648</faultcode>  <faultstring xml:lang="en-US">Incident Transmission 1 cannot be read as an object of collection 122, (see tag 'EquipmentType').</faultstring>  <detail>  …  </detail>  </s:Fault>  </s:Body>  </s:Envelope> |

#### Sender requesting non-existing attribute in requested properties

No error is returned. Only the correct requested propertied are returned. Note that the property names are case sensitive.

# Outbound Interfaces

The sections below show the Email & Task messages that ClickSchedule will be configured to send to MUST via FTP. Each message will result from a certain action on a Task/Assignment that exists within ClickSchedule. The associated business scenarios are referenced in the trigger sections 5.4.1 & 5.4.5, as are the message structures **Error! Reference source not found.** & 5.4.7.

## FTP File System

Outbond messages will go to a GEHC FTP endpoint. A custom message broker will route the Click generated XML from an HTTP POST message to a text file across FTP. The FTP endpoint will have the FS structure below:

Folders on the FTP endpoint:

<FTPRoot>\Click Schedule\OutgoingMessages\EmailTrigger\

<FTPRoot>\Click Schedule\OutgoingMessages\TaskUpdate\

The “to be processed” folder structure is split between PROD & DEV to delimitate between the environments in use:

..\EmailTrigger\PROD\_ToBeProcessed

..\EmailTrigger\DEV\_ToBeProcessed

Folders on the SDT macro machine for holding processed file logs:

<Localroot>\SendApptEmail\ToBeProcessed

<Localroot>\SendApptEmail\ProcessedPass

<Localroot>\SendApptEmail\ProcessedFail

<Localroot>\TaskRevisions\ToBeProcessed

<Localroot>\TaskRevisions\ProcessedPass

<Localroot>\TaskRevisions\ProcessedFail

## Messages Destinations

Each Click function will post messages containing the required data to outgoing destinations which will be defined in Click.

Once GEHC and Click have analyzed the external systems this document wil be updated with the exact outbound destination definitions.

Outgoing Messages Destinations Example:

|  |  |
| --- | --- |
| **Setting** | Value |
| **MUST** | |
| TimeZone | None |
| Send HTTP as QueryString | False |
| Send HTTP asynchronously | False |
| Send dates in ISO Format | True |
| Send Message ID | True |
| Send as SOAP | True |
| Address | TBD (based on communication endpoints) for example:  http://<HOST>:<PORTNUMBER>/Adapter/MessageServlet.svc?QueryString |

The message destination for each outgoing message will be identified for each message section.

## File Naming

To exchange data between Click and MUST an FTP queue will be queried by a MUST Agent, the message content parsed and loaded into MUST in the order that the messages arrived on the FTP server.

The File Name of the files will be concatenation of the Task details & outbound message:

|  |  |
| --- | --- |
| Value | Example |
| Outbound Message Time | 20130908T0900 |
| Header | * EmailTaskCreated * EmailTaskAppointmentTimeUpdate * EmailTaskEnRoute * EmailTaskCancel * TaskAppointment * TaskStatusAcknowledged * TaskStatusAssigned * TaskStatusCanceled |
| OldTaskID(for Task\*)  Or  MessageID (for Email\*) | Task\*:   * FR-A1234567\_D4\_20130909T0958   Email\*:   * 101445649 |
| .xml |  |

Note: Values will be separated by an underscore character \_ .

Example of file names:

|  |
| --- |
| 20131106T094611\_EmailTaskEnRoute\_101445649.xml  20131106T094449\_EmailTaskCancel\_101445643.xml  20131106T094611\_TaskAppointment\_A1234567\_GB-3013\_20131106T111200.xml  20131106T112628\_TaskStatusCancelled\_FR-A1111111\_D4\_20131106T101505.xml  20131106T094540\_TaskStatusAssigned\_FR-A1234567\_D4\_20131106T093535.xml |

## Email SendApptEmail

GEHC will provide an HTTP POST compatible email interface endpoint for the Click destination to send an HTTP message containing relevant customer appointment information to.

Training or Parts Pickup jobs are identified and excluded from the SendApptEmail outbound message, see 2.6.3.2

## Email Request Trigger

The following events will trigger the outbound message for only Tasks that are not pickup or training:

|  |  |
| --- | --- |
| **Outbound Action Name** | **When the Event Fires** |
| EmailTaskCreated | New Task Creation |
| EmailTaskAppointmentTimeUpdate | Task Update appointment times change |
| EmailTaskEnRoute | Task Status change to “En Route”; |
| EmailTaskCancel | Task Status change to “Cancel”; |

Note: Should any of the values included in the messages below be changed while the Task is the event fire status listed above, the outbound message will be resent with the new data.

## Message Fields

Below is a proposal for the task fields that shall be sent out in the outgoing message.

* EmailTaskCreated
  + CallID
  + Number
  + MUSTJobNumber
  + ContactName
  + ContactPhoneNumber
  + CustomerEmail
  + Customer
  + ContractOfferingFamily
  + Languages
* EmailTaskAppointmentTimeUpdate/ EmailTaskEnRoute/ EmailTaskCancel
  + CallID
  + Number
  + MUSTJobNumber
  + ContactName
  + ContactPhoneNumber
  + CustomerEmail
  + CustomerExpectation
  + ContractOfferingFamily
  + Languages
  + AppointmentStart
  + AppointmentFinish
  + Status
  + PrevTask
    - Status (not EmailTaskAppointmentTimeUpdate)
    - AppointmentStart
    - AppointmentFinish
* EmailTaskAppointmentTimeUpdate/ EmailTaskEnRoute
  + Assignment
    - Start
    - Finish

## Message Content

#### EmailTaskCreated

|  |
| --- |
| <EmailTaskCreated  Destination = "MainInt"  CreatedBy = "MUST"  MessageID = "101396480">  <Task>  <CallID>GB-A1234567-1</CallID>  <ContactName></ContactName>  <ContactPhoneNumber>+447720449438</ContactPhoneNumber>  <ContractOfferingFamily>Gold</ContractOfferingFamily>  <Customer>Cristina Hardie</Customer>  <MUSTJobNumber>A1234567</MUSTJobNumber>  <CustomerEmail>dhardie@gmail.com</CustomerEmail>  <Number>1</Number>  </Task>  </EmailTaskCreated> |

#### EmailTaskAppointmentTimeUpdate

|  |
| --- |
| <EmailTaskAppointmentTimeUpdate  Destination = "MainInt"  CreatedBy = "CLICKSOFTWARE\Duncan.Hardie"  MessageID = "101396482">  <Task>  <AppointmentFinish>2013-10-12T06:05:00</AppointmentFinish>  <AppointmentStart>2013-09-27T06:05:00</AppointmentStart>  <CallID>GB-A1234567-1</CallID>  <ContactName></ContactName>  <ContactPhoneNumber>+447720449438</ContactPhoneNumber>  <ContractOfferingFamily>Gold</ContractOfferingFamily>  <CustomerEmail>dhardie@gmail.com</CustomerEmail>  <CustomerExpectation>The Customer Expectation</CustomerExpectation>  <MUSTJobNumber>A1234567</MUSTJobNumber>  <Number>1</Number>  </Task>  <Assignment><Finish>2013-12-02T14:00:00</Finish><Start>2013-12-02T13:30:00</Start></Assignment>  <PrevTask>  <Task>  <AppointmentFinish>2013-10-12T06:05:12</AppointmentFinish>  <AppointmentStart>2013-09-27T06:05:12</AppointmentStart>  </Task>  </PrevTask>  </EmailTaskAppointmentTimeUpdate> |

#### EmailTaskEnRoute

|  |
| --- |
| <EmailTaskEnRoute  Destination = "MainInt"  CreatedBy = "CLICKSOFTWARE\Duncan.Hardie"  MessageID = "101396484">  <Task>  <AppointmentFinish>2013-10-12T06:05:00</AppointmentFinish>  <AppointmentStart>2013-09-27T06:05:00</AppointmentStart>  <CallID>GB-A1234567-1</CallID>  <ContactName></ContactName>  <ContactPhoneNumber>+447720449438</ContactPhoneNumber>  <ContractOfferingFamily>Gold</ContractOfferingFamily>  <CustomerEmail>dhardie@gmail.com</CustomerEmail>  <CustomerExpectation>The Customer Expectation</CustomerExpectation>  <MUSTJobNumber>A1234567</MUSTJobNumber>  <Number>1</Number>  <Status>  <Name>En Route</Name>  </Status>  </Task>  <Assignment><Finish>2013-12-02T14:00:00</Finish><Start>2013-12-02T13:30:00</Start></Assignment>  <PrevTask>  <Task>  <AppointmentFinish>2013-10-12T06:05:00</AppointmentFinish>  <AppointmentStart>2013-09-27T06:05:00</AppointmentStart>  <Status>  <Name>Assigned</Name>  </Status>  </Task>  </PrevTask>  </EmailTaskEnRoute> |

#### EmailTaskCancel

|  |
| --- |
| <EmailTaskCancel  Destination = "MainInt"  CreatedBy = "CLICKSOFTWARE\Duncan.Hardie"  MessageID = "101396485">  <Task>  <AppointmentFinish>2013-10-12T06:05:00</AppointmentFinish>  <AppointmentStart>2013-09-27T06:05:00</AppointmentStart>  <CallID>GB-A1234567-1</CallID>  <ContactName></ContactName>  <ContactPhoneNumber>+447720449438</ContactPhoneNumber>  <ContractOfferingFamily>Gold</ContractOfferingFamily>  <CustomerEmail>dhardie@gmail.com</CustomerEmail>  <CustomerExpectation>The Customer Expectation</CustomerExpectation>  <MUSTJobNumber>A1234567</MUSTJobNumber>  <Number>1</Number>  <Status>  <Name>Cancelled</Name>  </Status>  </Task>  <PrevTask>  <Task>  <AppointmentFinish>2013-10-12T06:05:00</AppointmentFinish>  <AppointmentStart>2013-09-27T06:05:00</AppointmentStart>  <Status>  <Name>En Route</Name>  </Status>  </Task>  </PrevTask>  </EmailTaskCancel> |

TaskTo maintain a UNID between MUST and Click a TaskID will be maintained within Click, see **Error! Reference source not found.**

TaskID is not static therefor two-way messaging between MUST and Click is required for the maintenance of correlation.

* MUST will create the initial TaskID UNID;
* As Click data changes affect the TaskID, outbound messages will trigger when specific events take place, these will be processed in sequence by the GEHC FTP Agent Service;
* The GEHC FTP Agent Service can propose an alternative TaskID should a Click proposal be non-unique and update Click via inbound integration of the required TaskID using ProcessTaskEx, also see **Error! Reference source not found.**;

## Task UNID generation

Should the MUST FTP agent service reject a new TaskID defined within the outbound text file the FTP agent will define an appropriate TaskID. This TaskID will be posted back to Click via a ProcessTaskEx action inbound message. The elements to be passed via ProcessTaskEx to update the relevant Task are:

* CallID;
* Number;
* TaskID;

Given outbound messaging will fire on TaskID update there is are no considerations regarding the CallerIdentity.

When MUST has no assigned engineers or appointment dates, the TaskID needs to be made unique using dummy data. The context of the fields which will be concatenated for the TaskID ensures a unique value: “MUSTCountryID-MUSTJobNumber\_EngineerBadgeNumber\_MUSTVisitDate”

## Task ID Revision & Status Triggers

The following events will trigger the outbound message for only primary Tasks:

**TaskID**

* Task Status change to “Assigned” (from not assigned to assigned);

**Status**

* Task Update appointment change;
* Task Status change to “Acknowledged”;
* Task Cancelation;

The following events will trigger the outbound message for Tasks except “Parts Pickup”:

|  |  |
| --- | --- |
| **Outbound Action Name** | **When the Event Fires** |
| TaskAppointment | Task Update once Status is “En Route” & beyond (e.g. “Completed”, “Incomplete”, “On Site”); |
| TaskStatusAcknowledged | Task Status change to “Acknowledged”; |
| TaskStatusAssigned | Task Status change to “Assigned”; |
| TaskStatusCancelled | Task Cancelation; |

Note: Should any of the values included in the messages below be changed while the Task is the event fire status listed above, the outbound message will be resent with the new data.

## Message Fields

Below is a proposal for the task fields that shall be sent out in the outgoing message.

* TaskStatusCanceled
  + AppointmentStart
  + AppointmentFinish
  + Status
  + TaskID
  + Languages
  + Previous Task Details
* TaskStatusAcknowledged/ TaskStatusAssigned/
  + Engineers
    - Engineer
      * MUSTID
      * Name
  + TaskAppointment
* TaskAppointment/ TaskStatusAcknowledged/ TaskStatusAssigned
  + Assignment
    - Start
    - Finish

## Message Content

Below is sample XML for each of the Task outbound triggers:

#### TaskAppointment

|  |
| --- |
| <TaskAppointment Destination="TaskRevisions" CreatedBy="david.gray" MessageID="101429784">  <Engineers>  <Engineer>  <MUSTID>GB-2251</MUSTID>  <Name>Gray, David Ivor (Dave Gray)</Name>  </Engineer>  </Engineers>  <Task>  <AppointmentFinish></AppointmentFinish>  <AppointmentStart></AppointmentStart>  <Status>  <Name>En Route</Name>  </Status>  <TaskID>B1060133\_GB-2251\_20140415T080000</TaskID>  </Task>  <Assignment>  <Finish>2014-04-15T18:00:00</Finish>  <Start>2014-04-15T08:00:00</Start>  </Assignment>  <PrevTask>  <Task>  <AppointmentFinish></AppointmentFinish>  <AppointmentStart></AppointmentStart>  <TaskID>B1060133\_GB-2251\_20140415T080000</TaskID>  </Task>  </PrevTask>  </TaskAppointment> |

#### TaskStatusAcknowledged

|  |
| --- |
| <TaskStatusAcknowledged Destination="TaskRevisions" CreatedBy="wajid.hussain" MessageID="101429789">  <Engineers>  <Engineer>  <MUSTID>GB-5386</MUSTID>  <Name>Hussain, Wajid</Name>  </Engineer>  </Engineers>  <Task>  <AppointmentFinish>2014-04-15T13:00:00</AppointmentFinish>  <AppointmentStart>2014-04-15T08:00:00</AppointmentStart>  <Status>  <Name>Acknowledged</Name>  </Status>  <TaskID>B5783122\_GB-5386\_20140415T090000</TaskID>  </Task>  <Assignment>  <Finish>2014-04-15T16:00:00</Finish>  <Start>2014-04-15T09:00:00</Start>  </Assignment>  <PrevTask>  <Task>  <AppointmentFinish>2014-04-15T13:00:00</AppointmentFinish>  <AppointmentStart>2014-04-15T08:00:00</AppointmentStart>  <TaskID>B5783122\_GB-5386\_20140415T090000</TaskID>  </Task>  </PrevTask>  </TaskStatusAcknowledged> |

#### TaskStatusAssigned

|  |
| --- |
| <TaskStatusAssigned Destination="TaskRevisions" CreatedBy="CSODPROD\annette.hughes">  <Engineers>  <Engineer>  <MUSTID>GB-3997</MUSTID>  <Name>Ward, Philip Thomas (Phil)</Name>  </Engineer>  </Engineers>  <Task>  <AppointmentFinish></AppointmentFinish>  <AppointmentStart></AppointmentStart>  <Status>  <Name>Assigned</Name>  </Status>  <TaskID>GB-B8935131\_GB-SDTGB\_20140416T080100</TaskID>  </Task>  <Assignment>  <Finish>2014-04-15T20:56:00</Finish>  <Start>2014-04-15T17:56:00</Start>  </Assignment>  <PrevTask>  <Task>  <AppointmentFinish></AppointmentFinish>  <AppointmentStart></AppointmentStart>  <TaskID>GB-B8935131\_GB-SDTGB\_20140416T080100</TaskID>  </Task>  </PrevTask>  </TaskStatusAssigned> |

#### TaskStatusCancelled

|  |
| --- |
| <TaskStatusCancelled Destination="TaskRevisions" CreatedBy="CSODPROD\kat.orr">  <Task>  <AppointmentFinish>2014-04-08T13:00:00</AppointmentFinish>  <AppointmentStart>2014-04-08T08:00:00</AppointmentStart>  <Status>  <Name>Cancelled</Name>  </Status>  <TaskID>D8537141\_GB-3982\_20140408T090000</TaskID>  </Task>  <PrevTask>  <Task>  <AppointmentFinish>2014-04-08T13:00:00</AppointmentFinish>  <AppointmentStart>2014-04-08T08:00:00</AppointmentStart>  <TaskID>D8537141\_GB-3982\_20140408T090000</TaskID>  </Task>  </PrevTask>  </TaskStatusCancelled> |

## Outbound Messages XSD

The following XSD files can be used for validating the message on retrieval:

|  |
| --- |
|  |

|  |
| --- |
|  |

# Outgoing Integration Manager

The Service Optimization Integration Manager may encounter two types of errors when it attempts to send an outgoing message:

* The destination address or a translator may be temporarily inaccessible.
* The destination may be accessible, but the processing application returns an error message (a response message containing an <Error> element or a Status="2" attribute).

In the former case, the Integration Manager attempts to resend the message.

In the latter case, the Integration Manager logs the error (if logging is enabled), and it does not attempt to resend the message.

The Integration Manager stores outgoing messages in the W6OUTGOING\_MESSAGES table of the Service Optimization database. The MessageStatus column of this table records the status of each message.

The possible MessageStatus values are:

|  |  |
| --- | --- |
| Message Status | Description |
| 0 | The message has not yet been sent to its destination. |
| 1 | The message was sent successfully; the destination application did not return an error. |
| 2 | The message was sent, but the destination application returned an error. The Integration Manager does not attempt to resend. |
| 3 | Temporarily, the message could not be sent, for example due to a communication error. The Integration Manager attempts to resend. |

ClickSchedule uses the MessageStatus to purge old messages from the database:

* Messages with status 1 will be purged after three days;
* Message with status 2 will be purged after three days;
* Message with status 0/3 will not be purged. ClickSchedule would resend them.

The Integration Manager sets the message status according to the content of the message response. For example, it sets a status of 2 if the response message contains an <Error> element.

Optionally, the destination component can override this behaviour by including a Status attribute in the response. The attribute specifies the status value that the Integration Manager should set. For example, the following response message to an UpdateTask message causes the Integration Manager to set a status of 2:

<CreateTaskResponse Status="2" />

For more information on the outgoing messages error handling please refer to chapter 5 in the Service Optimization Integration Guide.

In case of success, an Outgoing message response should be:

<CreateTaskResponse Status="1" />

Note that if the integration will not respond to Outgoing messages properly, there is a 5 second delay between every Outgoing message and it might lead the integration layer to hang (!)

## Integration Logs

The Service Optimization Integration Manager is able to log both incoming and outgoing messages to / from the system.

The integration logs may contain all messages and their results along with the relevant time stamps.

The log can be found on the SO server under the path C:\Temp\

For more information on the integration logs please refer to chapter 3 in the Service Optimization Integration Guide.

Note: Appointment values are converted from type date to type string & concatenated as a suffix to Engineer as string.

# Appendix A: Approvals

By signing this document, both individuals acknowledge their acceptance of the contents represented in the Interface Design Specification (version X, dated X).

|  |  |
| --- | --- |
| Company Name | GEHC |
| Name: |  |
| Title: |  |
| Signature: |  |
| Date: |  |
|  |  |
| ClickSoftware, Inc. |  |
| Name: |  |
| Title | Project Manager |
| Signature: |  |
| Date: |  |

Please fax a signed copy of this page of the document to your ClickSoftware Project Manager.

ClickSoftware, Inc.

Fax: 781-272-6409

# Appendix B: C# Snippets

These snippets are not intended for client deplowment, they are intended to show how the sample message in this document were produced.

|  |
| --- |
| using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  namespace PS\_GEHC\_Int\_Tests  {  class Program  {  static void Main(string[] args)  {  PS\_GEHC\_Int\_Tests GEHC\_Int\_Tests = new PS\_GEHC\_Int\_Tests();  GEHC\_Int\_Tests.doGetResources("MUSTID", "MUSTIDTrainer01");  GEHC\_Int\_Tests.doGetResources("ID", "Trainer01");  GEHC\_Int\_Tests.doGetAddressGeocodeRequest();  //Dppointment booking for three man job  GEHC\_Int\_Tests.doExtendedGetAppointmentsEx2Request(3);  //Create a Single Task  GEHC\_Int\_Tests.doMultiManning(1);  //Get Tasks with statuses that can be Cancelled by CallID  GEHC\_Int\_Tests.doCancelTasksByCallID("A1234567");  //Create a Parts Pickup Task  GEHC\_Int\_Tests.doPartPickup();  //Get Tasks with statuses that can be Cancelled by CallID  GEHC\_Int\_Tests.doCancelTasksByCallID("A1234569");  //Create a Multi Maning Task  GEHC\_Int\_Tests.doMultiManning(3);  //Get Tasks with statuses that can be Cancelled by CallID  GEHC\_Int\_Tests.doCancelTasksByCallID("A1234569");  //Create a Training Task;  GEHC\_Int\_Tests.doTraining();  //Get Tasks with statuses that can be Cancelled by TrainingJobNumber  GEHC\_Int\_Tests.doCancelTasksByTrainingJobNumber("A1111111");  }  }  } |

|  |
| --- |
| using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using MoreLinq;  namespace PS\_GEHC\_Int\_Tests  {  class PS\_GEHC\_Int\_Tests  {  private string[] taskRequestedProperties = new string[] { "Key", "CallID", "Number", "TaskID", "MUSTJobNumber" };  enum GroupNames {ByCallID, ByTrainingJobNumber};  enum TaskDependencys { SingleTask, PickupTask, MultiManTask, TrainingTask, CancelTask };  public void doGetResources(string Name, string Value)  {  string ServiceOptimizationServiceUri = "http://localhost:8888/SO/IntegrationServices/ServiceOptimizationService.svc";  ServiceOptimizationService serviceOptimizationService;  WebService\_BasicHttpInt.ServiceOptimizationService\_BasicHttp(ServiceOptimizationServiceUri);  SO\_WebServices.ServiceOptimizationServiceInt.GetResourcesRequest getResourcesRequest = new SO\_WebServices.ServiceOptimizationServiceInt.GetResourcesRequest();  SO\_WebServices.ServiceOptimizationServiceInt.Engineer[] engineers;  getResourcesRequest.OptionalParameters = new SO\_WebServices.ServiceOptimizationServiceInt.OptionalParameters() { CallerIdentity = "MUST", ErrorOnNonExistingDictionaries = true };  List<SO\_WebServices.ServiceOptimizationServiceInt.IndexesIndex> indexesIndexs = new List<SO\_WebServices.ServiceOptimizationServiceInt.IndexesIndex>();  SO\_WebServices.ServiceOptimizationServiceInt.IndexesIndex indexesIndex = new SO\_WebServices.ServiceOptimizationServiceInt.IndexesIndex();  indexesIndex.LowBound = new SO\_WebServices.ServiceOptimizationServiceInt.Property[1] {new SO\_WebServices.ServiceOptimizationServiceInt.Property(){Name = Name, Value = new SO\_WebServices.ServiceOptimizationServiceInt.KeyValue() { Value = Value } } };    indexesIndex.HighBound = indexesIndex.LowBound;  indexesIndexs.Add(indexesIndex);  getResourcesRequest.Indexes = new SO\_WebServices.ServiceOptimizationServiceInt.Indexes() { Index = indexesIndexs.ToArray() };  using (serviceOptimizationService = new ServiceOptimizationService(ServiceOptimizationServiceUri))  {  serviceOptimizationService.Open();  engineers = serviceOptimizationService.GetResources(getResourcesRequest.OptionalParameters, getResourcesRequest.Indexes, getResourcesRequest.Group, getResourcesRequest.EnableGroupOnTheFly, getResourcesRequest.KeySet, getResourcesRequest.RequestedProperties);  }  }  public void doGetAddressGeocodeRequest()  {  string ServiceOptimizationServiceUri = "http://localhost:8888/SO/IntegrationServices/ServiceOptimizationService.svc";  ServiceOptimizationService serviceOptimizationService;  WebService\_BasicHttpInt.ServiceOptimizationService\_BasicHttp(ServiceOptimizationServiceUri);  SO\_WebServices.ServiceOptimizationServiceInt.GetAddressGeocodeRequest getAddressGeocodeRequest = new SO\_WebServices.ServiceOptimizationServiceInt.GetAddressGeocodeRequest();  getAddressGeocodeRequest.OptionalParameters = new SO\_WebServices.ServiceOptimizationServiceInt.OptionalParameters() { CallerIdentity = "MUST" };  getAddressGeocodeRequest.Location = new SO\_WebServices.ServiceOptimizationServiceInt.GeocodeLocation();  getAddressGeocodeRequest.Location.Street = "39";  getAddressGeocodeRequest.Location.PostCode = "TN255AB";  getAddressGeocodeRequest.Location.Country = "UNITED KINGDOM";  getAddressGeocodeRequest.NumOfMatches = 10;  //SO\_WebServices.ServiceOptimizationServiceInt.GetAddressGeocodeResponse getAddressGeocodeResponse = new SO\_WebServices.ServiceOptimizationServiceInt.GetAddressGeocodeResponse();  SO\_WebServices.ServiceOptimizationServiceInt.Result result;  using (serviceOptimizationService = new ServiceOptimizationService(ServiceOptimizationServiceUri))  {  serviceOptimizationService.Open();  result = serviceOptimizationService.GetAddressGeocode(getAddressGeocodeRequest.OptionalParameters, getAddressGeocodeRequest.Location, getAddressGeocodeRequest.NumOfMatches);  }  }  public void doCancelTasksByTrainingJobNumber(string TrainingJobNumber)  {  string ScheduleServiceUri = "http://localhost:8888/SO/IntegrationServices/ScheduleService.svc";  ScheduleService scheduleService;  WebService\_BasicHttpInt.ScheduleService\_BasicHttp(ScheduleServiceUri);  SO\_WebServices.ScheduleServiceInt.Task[] tasks;  tasks = GetTasksRequestByGroup(GroupNames.ByTrainingJobNumber, TrainingJobNumber);  foreach(SO\_WebServices.ScheduleServiceInt.Task task in tasks)  {  SO\_WebServices.ScheduleServiceInt.ProcessTaskExRequest processTaskExRequest = new SO\_WebServices.ScheduleServiceInt.ProcessTaskExRequest();  SO\_WebServices.ScheduleServiceInt.ProcessTaskExResponse processTaskExResponse = new SO\_WebServices.ScheduleServiceInt.ProcessTaskExResponse();  processTaskExRequest.OptionalParameters = new SO\_WebServices.ScheduleServiceInt.OptionalParameters(){CallerIdentity="MUST"};  processTaskExRequest.TaskRequestedProperties = taskRequestedProperties;    processTaskExRequest.Task = new SO\_WebServices.ScheduleServiceInt.Task();  processTaskExRequest.Task.Key = task.Key;  processTaskExRequest.Task.KeySpecified = true;  processTaskExRequest.Task.CallID = task.CallID;  processTaskExRequest.Task.Number = task.Number;  processTaskExRequest.Task.NumberSpecified = true;  processTaskExRequest.Task.Status = new SO\_WebServices.ScheduleServiceInt.TaskStatusReference() { Name = "Cancelled" };  using (scheduleService = new ScheduleService(ScheduleServiceUri))  {  scheduleService.Open();  scheduleService.ProcessTaskEx(processTaskExRequest.OptionalParameters,  ref processTaskExRequest.Task, ref processTaskExRequest.Assignment, processTaskExRequest.ReturnAssignment,  processTaskExRequest.RelatedTasks, processTaskExRequest.SchedulingWorkflow, processTaskExRequest.LogicDomain,  processTaskExRequest.SchedulingHorizon, processTaskExRequest.TaskRequestedProperties, processTaskExRequest.AssignmentRequestedProperties,  processTaskExRequest.ReturnSchedulingError, out processTaskExResponse.LinkedTasks, out processTaskExResponse.SchedulingError);  }  }  }  public void doCancelTasksByCallID(string CallID)  {  string ScheduleServiceUri = "http://localhost:8888/SO/IntegrationServices/ScheduleService.svc";  ScheduleService scheduleService;  WebService\_BasicHttpInt.ScheduleService\_BasicHttp(ScheduleServiceUri);  SO\_WebServices.ScheduleServiceInt.Task[] tasks;  tasks = GetTasksRequestByGroup(GroupNames.ByCallID,CallID);  foreach (SO\_WebServices.ScheduleServiceInt.Task task in tasks)  {  SO\_WebServices.ScheduleServiceInt.ProcessTaskExRequest processTaskExRequest = new SO\_WebServices.ScheduleServiceInt.ProcessTaskExRequest();  SO\_WebServices.ScheduleServiceInt.ProcessTaskExResponse processTaskExResponse = new SO\_WebServices.ScheduleServiceInt.ProcessTaskExResponse();  processTaskExRequest.OptionalParameters = new SO\_WebServices.ScheduleServiceInt.OptionalParameters() { CallerIdentity = "MUST" };  processTaskExRequest.TaskRequestedProperties = taskRequestedProperties;  processTaskExRequest.Task = new SO\_WebServices.ScheduleServiceInt.Task();  processTaskExRequest.Task.Key = task.Key;  processTaskExRequest.Task.KeySpecified = true;  processTaskExRequest.Task.CallID = task.CallID;  processTaskExRequest.Task.Number = task.Number;  processTaskExRequest.Task.NumberSpecified = true;  processTaskExRequest.Task.Status = new SO\_WebServices.ScheduleServiceInt.TaskStatusReference() { Name = "Cancelled" };  using (scheduleService = new ScheduleService(ScheduleServiceUri))  {  scheduleService.Open();  scheduleService.ProcessTaskEx(processTaskExRequest.OptionalParameters,  ref processTaskExRequest.Task, ref processTaskExRequest.Assignment, processTaskExRequest.ReturnAssignment,  processTaskExRequest.RelatedTasks, processTaskExRequest.SchedulingWorkflow, processTaskExRequest.LogicDomain,  processTaskExRequest.SchedulingHorizon, processTaskExRequest.TaskRequestedProperties, processTaskExRequest.AssignmentRequestedProperties,  processTaskExRequest.ReturnSchedulingError, out processTaskExResponse.LinkedTasks, out processTaskExResponse.SchedulingError);  }  }  }  SO\_WebServices.ScheduleServiceInt.Task[] GetTasksRequestByGroup(GroupNames GroupName, string ParamValue)  {  string groupName = "";  if (GroupNames.ByCallID == GroupName)  {  groupName = "Statuses that can be Cancelled by CallID";  }  if (GroupNames.ByTrainingJobNumber == GroupName)  {  groupName = "Statuses that can be Cancelled by TrainingJobNumber";  }  string ScheduleServiceUri = "http://localhost:8888/SO/IntegrationServices/ScheduleService.svc";  ScheduleService scheduleService;  WebService\_BasicHttpInt.ScheduleService\_BasicHttp(ScheduleServiceUri);  SO\_WebServices.ScheduleServiceInt.GetTasksRequest getTasksRequest = new SO\_WebServices.ScheduleServiceInt.GetTasksRequest();  SO\_WebServices.ScheduleServiceInt.GetTasksResponse getTasksResponse = new SO\_WebServices.ScheduleServiceInt.GetTasksResponse();  SO\_WebServices.ScheduleServiceInt.Task[] Tasks;  getTasksRequest.OptionalParameters = new SO\_WebServices.ScheduleServiceInt.OptionalParameters();  getTasksRequest.OptionalParameters.CallerIdentity = "MUST";  getTasksRequest.RequestedProperties = taskRequestedProperties;  getTasksRequest.EnableGroupOnTheFly = true;    getTasksRequest.Group = new SO\_WebServices.ScheduleServiceInt.Group();  //getTasksRequest.Group.Name = groupName;  getTasksRequest.Group.GroupParameters = new string[] { ParamValue };    getTasksRequest.Group.GroupType = 2;  getTasksRequest.Group.GroupTypeSpecified = true;  getTasksRequest.Group.Body = @"<CONDITION TYPE=""100"" TYPE\_TEXT=""Logic Condition""><OPERATION TYPE=""500"" TYPE\_TEXT=""And"" /><CONDITION TYPE=""100"" TYPE\_TEXT=""Logic Condition""><OPERATION TYPE=""501"" TYPE\_TEXT=""Or"" /><CONDITION TYPE=""101"" TYPE\_TEXT=""Arithmetic Condition""><VALUE TYPE=""200"" TYPE\_TEXT=""Property Value""><ID>9</ID></VALUE><OPERATION TYPE=""600"" TYPE\_TEXT=""="" /><VALUE TYPE=""201"" TYPE\_TEXT=""Const Value"">124135424</VALUE></CONDITION><CONDITION TYPE=""101"" TYPE\_TEXT=""Arithmetic Condition""><VALUE TYPE=""200"" TYPE\_TEXT=""Property Value""><ID>9</ID></VALUE><OPERATION TYPE=""600"" TYPE\_TEXT=""="" /><VALUE TYPE=""201"" TYPE\_TEXT=""Const Value"">124135426</VALUE></CONDITION><CONDITION TYPE=""101"" TYPE\_TEXT=""Arithmetic Condition""><VALUE TYPE=""200"" TYPE\_TEXT=""Property Value""><ID>9</ID></VALUE><OPERATION TYPE=""600"" TYPE\_TEXT=""="" /><VALUE TYPE=""201"" TYPE\_TEXT=""Const Value"">124149760</VALUE></CONDITION><CONDITION TYPE=""101"" TYPE\_TEXT=""Arithmetic Condition""><VALUE TYPE=""200"" TYPE\_TEXT=""Property Value""><ID>9</ID></VALUE><OPERATION TYPE=""600"" TYPE\_TEXT=""="" /><VALUE TYPE=""201"" TYPE\_TEXT=""Const Value"">124149763</VALUE></CONDITION><CONDITION TYPE=""101"" TYPE\_TEXT=""Arithmetic Condition""><VALUE TYPE=""200"" TYPE\_TEXT=""Property Value""><ID>9</ID></VALUE><OPERATION TYPE=""600"" TYPE\_TEXT=""="" /><VALUE TYPE=""201"" TYPE\_TEXT=""Const Value"">124149762</VALUE></CONDITION></CONDITION><CONDITION TYPE=""101"" TYPE\_TEXT=""Arithmetic Condition""><VALUE TYPE=""200"" TYPE\_TEXT=""Property Value""><ID>3</ID></VALUE><OPERATION TYPE=""600"" TYPE\_TEXT=""="" /><VALUE TYPE=""201"" TYPE\_TEXT=""Const Value"">%1</VALUE></CONDITION></CONDITION>";  //getTasksRequest.Group.ScriptProgID = "W6PCustomVCLib.W6POQLGroupEngine, W6PCustomVCLib";  //getTasksRequest.Group.IsGroupUnique = false;  //getTasksRequest.Group.IsGroupUniqueSpecified = true;  /\*  getTasksRequest.Indexes = new SO\_WebServices.ScheduleServiceInt.Indexes(){Distinct= false,DistinctSpecified=true, Index = new SO\_WebServices.ScheduleServiceInt.IndexesIndex[1]};  getTasksRequest.Indexes.Index = new SO\_WebServices.ScheduleServiceInt.IndexesIndex[1];  getTasksRequest.Indexes.Index[1].LowBound = new SO\_WebServices.ScheduleServiceInt.Property[1];  getTasksRequest.Indexes.Index[1].LowBound[1].Value = new SO\_WebServices.ScheduleServiceInt.KeyValue { Value = "TrainingJobNumber" };  getTasksRequest.Indexes.Index[1].LowBound[1].Name = "";  \*/  using (scheduleService = new ScheduleService(ScheduleServiceUri))  {  scheduleService.Open();  Tasks = scheduleService.GetTasks(getTasksRequest.OptionalParameters,  getTasksRequest.Indexes, getTasksRequest.KeySet, getTasksRequest.Group,  getTasksRequest.EnableGroupOnTheFly, getTasksRequest.RequestedProperties,  getTasksRequest.GetAssignments, getTasksRequest.AssignmentRequestedProperties,  out getTasksResponse.Assignments);  }  return Tasks;  }  public void doPartPickup()  {  string ScheduleServiceUri = "http://localhost:8888/SO/IntegrationServices/ScheduleService.svc";  ScheduleService scheduleService;  WebService\_BasicHttpInt.ScheduleService\_BasicHttp(ScheduleServiceUri);  //Create MST Task  string jobNumber = "A1234568";  SO\_WebServices.ScheduleServiceInt.ProcessTaskExRequest[] processTaskExRequests;  SO\_WebServices.ScheduleServiceInt.TaskTypeReference[] taskTypeReferences = new SO\_WebServices.ScheduleServiceInt.TaskTypeReference[] { new SO\_WebServices.ScheduleServiceInt.TaskTypeReference { Name = "Installation" }, new SO\_WebServices.ScheduleServiceInt.TaskTypeReference { Name = "Parts Pickup" } };  processTaskExRequests = MakeProcessTaskExRequests(jobNumber, taskTypeReferences);  SO\_WebServices.ScheduleServiceInt.ProcessTaskExResponse processTaskExResponse = new SO\_WebServices.ScheduleServiceInt.ProcessTaskExResponse();  using (scheduleService = new ScheduleService(ScheduleServiceUri))  {  scheduleService.Open();  foreach (SO\_WebServices.ScheduleServiceInt.ProcessTaskExRequest processTaskExRequest in processTaskExRequests)  {  scheduleService.ProcessTaskEx(processTaskExRequest.OptionalParameters,  ref processTaskExRequest.Task, ref processTaskExRequest.Assignment, processTaskExRequest.ReturnAssignment,  processTaskExRequest.RelatedTasks, processTaskExRequest.SchedulingWorkflow, processTaskExRequest.LogicDomain,  processTaskExRequest.SchedulingHorizon, processTaskExRequest.TaskRequestedProperties, processTaskExRequest.AssignmentRequestedProperties,  processTaskExRequest.ReturnSchedulingError, out processTaskExResponse.LinkedTasks, out processTaskExResponse.SchedulingError);  }  }  string ServiceOptimizationServiceUri = "http://localhost:8888/SO/IntegrationServices/ServiceOptimizationService.svc";  ServiceOptimizationService serviceOptimizationService;  WebService\_BasicHttpInt.ServiceOptimizationService\_BasicHttp(ServiceOptimizationServiceUri);  SO\_WebServices.ServiceOptimizationServiceInt.ExecuteMultipleOperationsRequest executeMultipleOperationsRequest;  executeMultipleOperationsRequest = MakeExecuteMultipleOperationsRequest(processTaskExRequests, TaskDependencys.PickupTask);  SO\_WebServices.ServiceOptimizationServiceInt.ExecuteMultipleOperationsResponse executeMultipleOperationsResponse = new SO\_WebServices.ServiceOptimizationServiceInt.ExecuteMultipleOperationsResponse();  using (serviceOptimizationService = new ServiceOptimizationService(ServiceOptimizationServiceUri))  {  serviceOptimizationService.Open();  serviceOptimizationService.ExecuteMultipleOperations(executeMultipleOperationsRequest.OptionalParameters, executeMultipleOperationsRequest.Operations, executeMultipleOperationsRequest.OneTransaction, executeMultipleOperationsRequest.ContinueOnError);  }  }  public void doExtendedGetAppointmentsEx2Request(int NumberOfMen = 2)  {  string ScheduleServiceUri = "http://localhost:8888/SO/IntegrationServices/ScheduleService.svc";  ScheduleService scheduleService;  WebService\_BasicHttpInt.ScheduleService\_BasicHttp(ScheduleServiceUri);  //Create MST Task  string jobNumber = "A1234579";  SO\_WebServices.ScheduleServiceInt.ProcessTaskExRequest[] processTaskExRequests;  SO\_WebServices.ScheduleServiceInt.TaskTypeReference[] taskTypeReferences = new SO\_WebServices.ScheduleServiceInt.TaskTypeReference[NumberOfMen];  for (int i = 0; i < NumberOfMen; i++)  {  //Make all multi man jobs the same type for this example  taskTypeReferences[i] = new SO\_WebServices.ScheduleServiceInt.TaskTypeReference();  taskTypeReferences[i].Name = "Installation";  }  processTaskExRequests = MakeProcessTaskExRequests(jobNumber, taskTypeReferences);  SO\_WebServices.ServiceOptimizationServiceInt.ExecuteMultipleOperationsRequest executeMultipleOperationsRequest;  executeMultipleOperationsRequest = MakeExecuteMultipleOperationsRequest(processTaskExRequests, TaskDependencys.MultiManTask);  SO\_WebServices.ScheduleServiceInt.ExtendedGetAppointmentsEx2Request extendedGetAppointmentsEx2Request = new SO\_WebServices.ScheduleServiceInt.ExtendedGetAppointmentsEx2Request();  extendedGetAppointmentsEx2Request.OptionalParameters = new SO\_WebServices.ScheduleServiceInt.OptionalParameters();  extendedGetAppointmentsEx2Request.OptionalParameters.CallerIdentity = "MUST";  List<SO\_WebServices.ScheduleServiceInt.Task> tasks = new List<SO\_WebServices.ScheduleServiceInt.Task>();  int y = 0;  foreach (SO\_WebServices.ScheduleServiceInt.ProcessTaskExRequest processTaskExRequest in processTaskExRequests)  {  if (y == 0)  {  extendedGetAppointmentsEx2Request.Task = processTaskExRequest.Task;  }  else  {  tasks.Add(processTaskExRequest.Task);  }  y = y + 1;  }  extendedGetAppointmentsEx2Request.RelatedTasks = tasks.ToArray();  SO\_WebServices.ServiceOptimizationServiceInt.Task task = (SO\_WebServices.ServiceOptimizationServiceInt.Task)(SO\_WebServices.ServiceOptimizationServiceInt.BaseObject)executeMultipleOperationsRequest.Operations[1].Object.Object;    List<SO\_WebServices.ScheduleServiceInt.TaskTaskTimeDependency> taskTaskTimeDependencys = new List<SO\_WebServices.ScheduleServiceInt.TaskTaskTimeDependency>();  foreach (SO\_WebServices.ServiceOptimizationServiceInt.TaskTaskTimeDependency timeDependency in task.TimeDependencies)  {  SO\_WebServices.ScheduleServiceInt.TaskTaskTimeDependency taskTaskTimeDependency = new SO\_WebServices.ScheduleServiceInt.TaskTaskTimeDependency();  taskTaskTimeDependency.Critical = timeDependency.Critical;  taskTaskTimeDependency.CriticalSpecified = timeDependency.CriticalSpecified;  taskTaskTimeDependency.LowerBound = timeDependency.LowerBound;  taskTaskTimeDependency.LowerBoundSpecified = taskTaskTimeDependency.LowerBoundSpecified;  taskTaskTimeDependency.RelationOperator = timeDependency.RelationOperator;  taskTaskTimeDependency.RelationOperatorSpecified = timeDependency.RelationOperatorSpecified;  taskTaskTimeDependency.RelationType = timeDependency.RelationType;  taskTaskTimeDependency.RelationTypeSpecified = timeDependency.RelationTypeSpecified;  taskTaskTimeDependency.TaskKey = new SO\_WebServices.ScheduleServiceInt.TaskReference() { CallID = timeDependency.TaskKey.CallID, Number = timeDependency.TaskKey.Number, NumberSpecified = timeDependency.TaskKey.NumberSpecified };  taskTaskTimeDependency.UpperBound = timeDependency.UpperBound;  taskTaskTimeDependency.UpperBoundSpecified = taskTaskTimeDependency.UpperBoundSpecified;  taskTaskTimeDependencys.Add(taskTaskTimeDependency);  }  List<SO\_WebServices.ScheduleServiceInt.TaskTaskEngineerDependency> taskTaskEngineerDependencys = new List<SO\_WebServices.ScheduleServiceInt.TaskTaskEngineerDependency>();  foreach (SO\_WebServices.ServiceOptimizationServiceInt.TaskTaskEngineerDependency engineerDependency in task.EngineerDependencies)  {  SO\_WebServices.ScheduleServiceInt.TaskTaskEngineerDependency taskTaskEngineerDependency = new SO\_WebServices.ScheduleServiceInt.TaskTaskEngineerDependency();  taskTaskEngineerDependency.RelationType = engineerDependency.RelationType;  taskTaskEngineerDependency.RelationTypeSpecified = engineerDependency.RelationTypeSpecified;  taskTaskEngineerDependency.TaskKey = new SO\_WebServices.ScheduleServiceInt.TaskReference() { CallID = engineerDependency.TaskKey.CallID, Number = engineerDependency.TaskKey.Number, NumberSpecified = engineerDependency.TaskKey.NumberSpecified };  taskTaskEngineerDependencys.Add(taskTaskEngineerDependency);  }  extendedGetAppointmentsEx2Request.Task.TimeDependencies = taskTaskTimeDependencys.ToArray();  extendedGetAppointmentsEx2Request.Task.EngineerDependencies = taskTaskEngineerDependencys.ToArray();  extendedGetAppointmentsEx2Request.Profile = "TWO HOURS";  extendedGetAppointmentsEx2Request.Period = new SO\_WebServices.ScheduleServiceInt.TimeInterval() { Finish = System.DateTime.Now.AddDays(14), Start = System.DateTime.Now.AddDays(1) };  extendedGetAppointmentsEx2Request.ExcludeCurrentAppointment = false;  extendedGetAppointmentsEx2Request.TimeOut = 60;  extendedGetAppointmentsEx2Request.GradeAppointments = false;  extendedGetAppointmentsEx2Request.SchedulePolicy = "Standard";    using (scheduleService = new ScheduleService(ScheduleServiceUri))  {  scheduleService.Open();  scheduleService.ExtendedGetAppointmentsEx2(extendedGetAppointmentsEx2Request.OptionalParameters,extendedGetAppointmentsEx2Request.Task,extendedGetAppointmentsEx2Request.SchedulePolicy,extendedGetAppointmentsEx2Request.TimeIntervals,extendedGetAppointmentsEx2Request.Profile,extendedGetAppointmentsEx2Request.Period,extendedGetAppointmentsEx2Request.ExcludeCurrentAppointment,extendedGetAppointmentsEx2Request.TimeOut,extendedGetAppointmentsEx2Request.GradeAppointments,extendedGetAppointmentsEx2Request.SchedulingHorizon,extendedGetAppointmentsEx2Request.DoubleBookingTasks,extendedGetAppointmentsEx2Request.UnmovableTasks,extendedGetAppointmentsEx2Request.RelatedTasks,extendedGetAppointmentsEx2Request.SameSlotTasks,extendedGetAppointmentsEx2Request.LogicDomainForDoubleBooking,extendedGetAppointmentsEx2Request.UnScheduleRelatedTasks,extendedGetAppointmentsEx2Request.UnScheduleRelatedTasksGroup,extendedGetAppointmentsEx2Request.SuggestCandidateResources,extendedGetAppointmentsEx2Request.PerformanceParameters,extendedGetAppointmentsEx2Request.ParallelFactor,extendedGetAppointmentsEx2Request.UseSLRCache,extendedGetAppointmentsEx2Request.UsePartitionControl);  }  }  public void doMultiManning(int NumberOfMen = 2)  {  string ScheduleServiceUri = "http://localhost:8888/SO/IntegrationServices/ScheduleService.svc";  ScheduleService scheduleService;  WebService\_BasicHttpInt.ScheduleService\_BasicHttp(ScheduleServiceUri);  //Create MST Task  string jobNumber = "A1234569";  if(NumberOfMen == 1)  {  jobNumber = "A1234567";  }  SO\_WebServices.ScheduleServiceInt.ProcessTaskExRequest[] processTaskExRequests;  SO\_WebServices.ScheduleServiceInt.TaskTypeReference[] taskTypeReferences = new SO\_WebServices.ScheduleServiceInt.TaskTypeReference[NumberOfMen];  for (int i = 0; i < NumberOfMen; i++)  {  //Make all multi man jobs the same type for this example  taskTypeReferences[i] = new SO\_WebServices.ScheduleServiceInt.TaskTypeReference();  taskTypeReferences[i].Name= "Installation";  }    processTaskExRequests = MakeProcessTaskExRequests(jobNumber, taskTypeReferences);  SO\_WebServices.ScheduleServiceInt.ProcessTaskExResponse processTaskExResponse = new SO\_WebServices.ScheduleServiceInt.ProcessTaskExResponse();  using (scheduleService = new ScheduleService(ScheduleServiceUri))  {  scheduleService.Open();  foreach (SO\_WebServices.ScheduleServiceInt.ProcessTaskExRequest processTaskExRequest in processTaskExRequests)  {  scheduleService.ProcessTaskEx(processTaskExRequest.OptionalParameters,  ref processTaskExRequest.Task, ref processTaskExRequest.Assignment, processTaskExRequest.ReturnAssignment,  processTaskExRequest.RelatedTasks, processTaskExRequest.SchedulingWorkflow, processTaskExRequest.LogicDomain,  processTaskExRequest.SchedulingHorizon, processTaskExRequest.TaskRequestedProperties, processTaskExRequest.AssignmentRequestedProperties,  processTaskExRequest.ReturnSchedulingError, out processTaskExResponse.LinkedTasks, out processTaskExResponse.SchedulingError);  }  }  string ServiceOptimizationServiceUri = "http://localhost:8888/SO/IntegrationServices/ServiceOptimizationService.svc";  ServiceOptimizationService serviceOptimizationService;  WebService\_BasicHttpInt.ServiceOptimizationService\_BasicHttp(ServiceOptimizationServiceUri);  SO\_WebServices.ServiceOptimizationServiceInt.ExecuteMultipleOperationsRequest executeMultipleOperationsRequest;  executeMultipleOperationsRequest = MakeExecuteMultipleOperationsRequest(processTaskExRequests, TaskDependencys.MultiManTask);  SO\_WebServices.ServiceOptimizationServiceInt.ExecuteMultipleOperationsResponse executeMultipleOperationsResponse = new SO\_WebServices.ServiceOptimizationServiceInt.ExecuteMultipleOperationsResponse();  using (serviceOptimizationService = new ServiceOptimizationService(ServiceOptimizationServiceUri))  {  serviceOptimizationService.Open();  serviceOptimizationService.ExecuteMultipleOperations(executeMultipleOperationsRequest.OptionalParameters, executeMultipleOperationsRequest.Operations, executeMultipleOperationsRequest.OneTransaction, executeMultipleOperationsRequest.ContinueOnError);  }  }  public void doTraining()  {  string ScheduleServiceUri = "http://localhost:8888/SO/IntegrationServices/ScheduleService.svc";  ScheduleService scheduleService;  WebService\_BasicHttpInt.ScheduleService\_BasicHttp(ScheduleServiceUri);  //Create MST Task  string jobNumber = "A1234570";  SO\_WebServices.ScheduleServiceInt.ProcessTaskExRequest[] processTaskExRequests;  SO\_WebServices.ScheduleServiceInt.TaskTypeReference[] taskTypeReferences = new SO\_WebServices.ScheduleServiceInt.TaskTypeReference[] { new SO\_WebServices.ScheduleServiceInt.TaskTypeReference() { Name = "Installation" }, new SO\_WebServices.ScheduleServiceInt.TaskTypeReference() { Name = "Training" } };  processTaskExRequests = MakeProcessTaskExRequests(jobNumber, taskTypeReferences);  SO\_WebServices.ScheduleServiceInt.ProcessTaskExResponse processTaskExResponse = new SO\_WebServices.ScheduleServiceInt.ProcessTaskExResponse();  using (scheduleService = new ScheduleService(ScheduleServiceUri))  {  scheduleService.Open();  foreach (SO\_WebServices.ScheduleServiceInt.ProcessTaskExRequest processTaskExRequest in processTaskExRequests)  {  scheduleService.ProcessTaskEx(processTaskExRequest.OptionalParameters,  ref processTaskExRequest.Task, ref processTaskExRequest.Assignment, processTaskExRequest.ReturnAssignment,  processTaskExRequest.RelatedTasks, processTaskExRequest.SchedulingWorkflow, processTaskExRequest.LogicDomain,  processTaskExRequest.SchedulingHorizon, processTaskExRequest.TaskRequestedProperties, processTaskExRequest.AssignmentRequestedProperties,  processTaskExRequest.ReturnSchedulingError, out processTaskExResponse.LinkedTasks, out processTaskExResponse.SchedulingError);  }  }  string ServiceOptimizationServiceUri = "http://localhost:8888/SO/IntegrationServices/ServiceOptimizationService.svc";  ServiceOptimizationService serviceOptimizationService;  WebService\_BasicHttpInt.ServiceOptimizationService\_BasicHttp(ServiceOptimizationServiceUri);  SO\_WebServices.ServiceOptimizationServiceInt.ExecuteMultipleOperationsRequest executeMultipleOperationsRequest;  executeMultipleOperationsRequest = MakeExecuteMultipleOperationsRequest(processTaskExRequests, TaskDependencys.TrainingTask);  SO\_WebServices.ServiceOptimizationServiceInt.ExecuteMultipleOperationsResponse executeMultipleOperationsResponse = new SO\_WebServices.ServiceOptimizationServiceInt.ExecuteMultipleOperationsResponse();  using (serviceOptimizationService = new ServiceOptimizationService(ServiceOptimizationServiceUri))  {  serviceOptimizationService.Open();  serviceOptimizationService.ExecuteMultipleOperations(executeMultipleOperationsRequest.OptionalParameters, executeMultipleOperationsRequest.Operations, executeMultipleOperationsRequest.OneTransaction, executeMultipleOperationsRequest.ContinueOnError);  }  }  SO\_WebServices.ServiceOptimizationServiceInt.ExecuteMultipleOperationsRequest MakeExecuteMultipleOperationsRequest(SO\_WebServices.ScheduleServiceInt.ProcessTaskExRequest[] processTaskExRequests, TaskDependencys taskDependencys)  {  int RequestCount = processTaskExRequests.GetUpperBound(0) + 1;  List<string> jobNumbers = new List<string>(); //Keep track of the Job Numbers for this Payload  //Time calcs  DateTime now = DateTime.Now;  SO\_WebServices.ServiceOptimizationServiceInt.ExecuteMultipleOperationsRequest executeMultipleOperationsRequest = new SO\_WebServices.ServiceOptimizationServiceInt.ExecuteMultipleOperationsRequest();  executeMultipleOperationsRequest.ContinueOnError = false;  executeMultipleOperationsRequest.OneTransaction = false;  executeMultipleOperationsRequest.OptionalParameters = new SO\_WebServices.ServiceOptimizationServiceInt.OptionalParameters();  executeMultipleOperationsRequest.OptionalParameters.CallerIdentity = "MUSTMB";  executeMultipleOperationsRequest.OptionalParameters.ErrorOnNonExistingDictionaries = true;  List<SO\_WebServices.ServiceOptimizationServiceInt.StandardOperation> standardOperations = new List<SO\_WebServices.ServiceOptimizationServiceInt.StandardOperation>();  SO\_WebServices.ServiceOptimizationServiceInt.Task[] task = new SO\_WebServices.ServiceOptimizationServiceInt.Task[RequestCount];  //Only create dependancies for secondary/alturnate tasks except for multi maning  List<SO\_WebServices.ServiceOptimizationServiceInt.TaskTaskTimeDependency> taskTaskTimeDependencys = new List<SO\_WebServices.ServiceOptimizationServiceInt.TaskTaskTimeDependency>();  List<SO\_WebServices.ServiceOptimizationServiceInt.TaskTaskEngineerDependency> taskTaskEngineerDependencys = new List<SO\_WebServices.ServiceOptimizationServiceInt.TaskTaskEngineerDependency>();  for (int i = 0; i < RequestCount; i++)  {  SO\_WebServices.ServiceOptimizationServiceInt.StandardOperation standardOperation = new SO\_WebServices.ServiceOptimizationServiceInt.StandardOperation();  //Only UpdateOrCreate GEHCJobComments if its not been done yet in this payload  if (jobNumbers.Any(jNum => jobNumbers.Contains(processTaskExRequests[i].Task.MUSTJobNumber)) == false)  {  standardOperation = new SO\_WebServices.ServiceOptimizationServiceInt.StandardOperation();  standardOperation.Action = "UpdateOrCreate";  SO\_WebServices.ServiceOptimizationServiceInt.GEHCJobComments jobComments = new SO\_WebServices.ServiceOptimizationServiceInt.GEHCJobComments();  jobComments.MUSTJobNumber = processTaskExRequests[i].Task.MUSTJobNumber;  if (taskDependencys == TaskDependencys.PickupTask)  {  jobComments.Text = "PickupTask: Very long text";  }  if (taskDependencys == TaskDependencys.MultiManTask && RequestCount > 1)  {  jobComments.Text = "MultiManTask: Very long text";  }  if (taskDependencys == TaskDependencys.SingleTask || (taskDependencys == TaskDependencys.MultiManTask & RequestCount == 1))  {  jobComments.Text = "SingleTask: Very long text";  }  if (taskDependencys == TaskDependencys.TrainingTask)  {  jobComments.Text = "TrainingTask: Very long text";  }  standardOperation.Object = new SO\_WebServices.ServiceOptimizationServiceInt.BaseObjectWrapper();  standardOperation.Object.Object = (SO\_WebServices.ServiceOptimizationServiceInt.BaseObject)jobComments;  jobNumbers.Add(processTaskExRequests[i].Task.MUSTJobNumber);  //Exclude the JobComment from the Pickup Task  //if (taskDependencys != TaskDependencys.PickupTask || IsOdd(i) == false)  //{  standardOperations.Add(standardOperation);  //}  }  standardOperation = new SO\_WebServices.ServiceOptimizationServiceInt.StandardOperation();  if (taskDependencys != TaskDependencys.PickupTask || IsOdd(i) == false)  {  //Dont append the the JobComments to the PartPickup  standardOperation.Action = "Update";  //General Task Details  task[i] = new SO\_WebServices.ServiceOptimizationServiceInt.Task();  task[i].CallID = processTaskExRequests[i].Task.CallID;  task[i].Number = processTaskExRequests[i].Task.Number;  task[i].NumberSpecified = true;  //Make the task critical for MultiManTask and PickupTask  if (taskDependencys == TaskDependencys.MultiManTask | taskDependencys == TaskDependencys.PickupTask)  {  task[i].Critical = true;  task[i].CriticalSpecified = true;  }  if (taskDependencys == TaskDependencys.TrainingTask)  {  task[i].TrainingJobNumber = processTaskExRequests[i].Task.MUSTJobNumber;  }  //The JobComments  task[i].JobComments = new SO\_WebServices.ServiceOptimizationServiceInt.GEHCJobCommentsReference() { MUSTJobNumber = processTaskExRequests[i].Task.MUSTJobNumber };  }  //Dependancies  if ((IsOdd(i) & i > 0) || taskDependencys == TaskDependencys.MultiManTask && i > 0)  {  SO\_WebServices.ServiceOptimizationServiceInt.TaskTaskTimeDependency taskTaskTimeDependency = new SO\_WebServices.ServiceOptimizationServiceInt.TaskTaskTimeDependency();  SO\_WebServices.ServiceOptimizationServiceInt.TaskTaskEngineerDependency taskTaskEngineerDependency = new SO\_WebServices.ServiceOptimizationServiceInt.TaskTaskEngineerDependency();  SO\_WebServices.ServiceOptimizationServiceInt.TaskReference mainTaskReference = new SO\_WebServices.ServiceOptimizationServiceInt.TaskReference() { CallID = processTaskExRequests[i].Task.CallID, Number = processTaskExRequests[i].Task.Number, NumberSpecified = true };  if (taskDependencys == TaskDependencys.MultiManTask || taskDependencys == TaskDependencys.TrainingTask)  {  taskTaskTimeDependency.Critical = true;  taskTaskTimeDependency.CriticalSpecified = true;  taskTaskTimeDependency.TaskKey = mainTaskReference;  taskTaskTimeDependency.RelationType = 1;  taskTaskTimeDependency.RelationTypeSpecified = true;  taskTaskTimeDependency.RelationOperator = 2;  taskTaskTimeDependency.RelationOperatorSpecified = true;  taskTaskTimeDependencys.Add(taskTaskTimeDependency);  if (taskDependencys != TaskDependencys.MultiManTask)  {  task[i - 1].TimeDependencies = new SO\_WebServices.ServiceOptimizationServiceInt.TaskTaskTimeDependency[] { taskTaskTimeDependency };  }  if (taskDependencys == TaskDependencys.MultiManTask || taskDependencys == TaskDependencys.TrainingTask)  {  //EngineerDependencies  taskTaskEngineerDependency.TaskKey = mainTaskReference;  taskTaskEngineerDependency.RelationType = 2;  taskTaskEngineerDependency.RelationTypeSpecified = true;  taskTaskEngineerDependencys.Add(taskTaskEngineerDependency);  if (taskDependencys == TaskDependencys.TrainingTask)  {  //Always link MultiManTasks to main task  task[0].TimeDependencies = taskTaskTimeDependencys.ToArray();  task[0].EngineerDependencies = taskTaskEngineerDependencys.ToArray(); //Always assign to main task, dont create MST dependancy chains  }  }    }  if (taskDependencys == TaskDependencys.PickupTask)  {  taskTaskTimeDependencys = new List<SO\_WebServices.ServiceOptimizationServiceInt.TaskTaskTimeDependency>();  taskTaskEngineerDependencys = new List<SO\_WebServices.ServiceOptimizationServiceInt.TaskTaskEngineerDependency>();  //TimeDependencys start to start  taskTaskTimeDependency.Critical = true;  taskTaskTimeDependency.CriticalSpecified = true;  taskTaskTimeDependency.TaskKey = mainTaskReference;  taskTaskTimeDependency.RelationType = 0;  taskTaskTimeDependency.RelationTypeSpecified = true;  taskTaskTimeDependency.RelationOperator = 0;  taskTaskTimeDependency.RelationOperatorSpecified = true;  taskTaskTimeDependency.UpperBound = 0;  taskTaskTimeDependency.UpperBoundSpecified = true;  taskTaskTimeDependency.LowerBound = 0;  taskTaskTimeDependency.LowerBoundSpecified = true;  taskTaskTimeDependencys.Add(taskTaskTimeDependency);  taskTaskTimeDependency = new SO\_WebServices.ServiceOptimizationServiceInt.TaskTaskTimeDependency();  //TimeDependencys start same day optional  taskTaskTimeDependency.Critical = true;  taskTaskTimeDependency.CriticalSpecified = true;  taskTaskTimeDependency.TaskKey = mainTaskReference;  taskTaskTimeDependency.RelationType = 4;  taskTaskTimeDependency.RelationTypeSpecified = true;  taskTaskTimeDependency.RelationOperator = 0;  taskTaskTimeDependency.RelationOperatorSpecified = true;  taskTaskTimeDependency.UpperBound = 0;  taskTaskTimeDependency.UpperBoundSpecified = true;  taskTaskTimeDependency.LowerBound = 0;  taskTaskTimeDependency.LowerBoundSpecified = true;  taskTaskTimeDependencys.Add(taskTaskTimeDependency);  task[i - 1].TimeDependencies = taskTaskTimeDependencys.ToArray();  //EngineerDependencies  taskTaskEngineerDependency.TaskKey = mainTaskReference;  taskTaskEngineerDependency.RelationType = 1;  taskTaskEngineerDependency.RelationTypeSpecified = true;  taskTaskEngineerDependencys.Add(taskTaskEngineerDependency);  task[i - 1].EngineerDependencies = taskTaskEngineerDependencys.ToArray();  }  standardOperation.Object = new SO\_WebServices.ServiceOptimizationServiceInt.BaseObjectWrapper();  standardOperation.Object.Object = (SO\_WebServices.ServiceOptimizationServiceInt.BaseObject)task[i];  if (!Object.ReferenceEquals(standardOperation.Object.Object, null))  {  standardOperations.Add(standardOperation);  }  }  else  {  standardOperation.Object = new SO\_WebServices.ServiceOptimizationServiceInt.BaseObjectWrapper();  standardOperation.Object.Object = (SO\_WebServices.ServiceOptimizationServiceInt.BaseObject)task[i];  if (!Object.ReferenceEquals(standardOperation.Object.Object, null))  {  standardOperations.Add(standardOperation);  }  }  }  if (taskDependencys == TaskDependencys.MultiManTask)  {  //Always link MultiManTasks to main task  task[0].TimeDependencies = taskTaskTimeDependencys.ToArray();  task[0].EngineerDependencies = taskTaskEngineerDependencys.ToArray(); //Always assign to main task, dont create MST dependancy chains  }    //Distinct any duplicate JobComments  executeMultipleOperationsRequest.Operations = standardOperations.ToArray();  return executeMultipleOperationsRequest;  }  SO\_WebServices.ScheduleServiceInt.ProcessTaskExRequest[] MakeProcessTaskExRequests(string callID, SO\_WebServices.ScheduleServiceInt.TaskTypeReference[] taskTypeReferences)  {  int RequestCount = taskTypeReferences.GetUpperBound(0) +1;    //Time calcs  DateTime now = DateTime.Now;  DateTime dueDate = now.AddDays(14);  dueDate.AddHours(1); //Due a day and an three hours from now;  //DateTime earlyStart = dueDate.AddHours(-4); //Early start 4 hours before its due  DateTime earlyStart = now;  DateTime lateStart = dueDate.AddHours(-1); //Late start 1 hour before its due  DateTime appointmentStart = now;  DateTime appointmentFinish = dueDate.AddDays(1); //1 day in the future  //TaskID concats  string MUSTCountryID = "FR";  string MUSTBadgeNumber = "D4";  string nowStr = now.ToString("yyyyMMddTHHmmss");    SO\_WebServices.ScheduleServiceInt.ProcessTaskExRequest[] processTaskExRequests = new SO\_WebServices.ScheduleServiceInt.ProcessTaskExRequest[RequestCount];  for (int i=0;i<RequestCount;i++)  {  nowStr = now.AddMinutes(i).ToString("yyyyMMddTHHmmss"); //Increment the mins per task  string trainingJobNumber = "A1111111";  string jobNumber = callID;  string taskID;  if(taskTypeReferences[i].Name == "Training")  {  taskID = MUSTCountryID + "-" + trainingJobNumber + "\_" + MUSTBadgeNumber + "\_" + nowStr;  jobNumber = trainingJobNumber;  }  else  {  taskID = MUSTCountryID + "-" + callID + "\_" + MUSTBadgeNumber + "\_" + nowStr;  }  SO\_WebServices.ScheduleServiceInt.ProcessTaskExRequest processTaskExRequest = new SO\_WebServices.ScheduleServiceInt.ProcessTaskExRequest();  //OptionalParameters  processTaskExRequest.OptionalParameters = new SO\_WebServices.ScheduleServiceInt.OptionalParameters();  processTaskExRequest.OptionalParameters.CallerIdentity = "MUST";  processTaskExRequest.OptionalParameters.ErrorOnNonExistingDictionaries = false;  //RequestedProperties  processTaskExRequest.TaskRequestedProperties = taskRequestedProperties;  processTaskExRequest.AssignmentRequestedProperties = new string[] {"Key", "Start", "Finish" };  //ReturnAssignment  processTaskExRequest.ReturnAssignment = true;  processTaskExRequest.Task = new SO\_WebServices.ScheduleServiceInt.Task();  //General Task Details  processTaskExRequest.Task.CallID = callID;  processTaskExRequest.Task.Number = (i + 1);  processTaskExRequest.Task.NumberSpecified = true;  processTaskExRequest.Task.TaskID = taskID; //e.g. FR-A1234567\_D4\_20130909T0958  processTaskExRequest.Task.MUSTJobNumber = jobNumber;  processTaskExRequest.Task.SystemID = new SO\_WebServices.ScheduleServiceInt.GEHCSystemReference() { ID = "DARK1" };  processTaskExRequest.Task.CRMSystemName = new SO\_WebServices.ScheduleServiceInt.GEHCCRMSystemReference() {Name = "MUST" };  processTaskExRequest.Task.IsSafety = false;  processTaskExRequest.Task.IsSafetySpecified = true;  processTaskExRequest.Task.Priority = 10;  processTaskExRequest.Task.PrioritySpecified = true;  /\*  foreach (SO\_WebServices.ScheduleServiceInt.TaskTypeReference taskTypeReference in taskTypeReferences)  {  if (taskTypeReference.Name == "Parts Pickup")  {  processTaskExRequest.Task.Critical = true;  processTaskExRequest.Task.CriticalSpecified = true;  }  }  \* \*/  processTaskExRequest.Task.TaskType = taskTypeReferences[i];  processTaskExRequest.Task.TaskSubType = new SO\_WebServices.ScheduleServiceInt.GEHCTaskSubTypeReference() {Name = "MRI"};  processTaskExRequest.Task.CustomerExpectation = "The Customer Expectation";  processTaskExRequest.Task.SuperPowerTaskFlag = false;  processTaskExRequest.Task.SuperPowerTaskFlagSpecified = false;    //Time  processTaskExRequest.Task.OpenDate = now;  processTaskExRequest.Task.OpenDateSpecified = true;  /\*  processTaskExRequest.Task.EarlyStart = earlyStart;  processTaskExRequest.Task.EarlyStartSpecified = true;  processTaskExRequest.Task.LateStart = lateStart;  processTaskExRequest.Task.LateStartSpecified = true;  \*/  processTaskExRequest.Task.Duration = 900;  processTaskExRequest.Task.DurationSpecified = true;  processTaskExRequest.Task.ReactivationDate = now;  processTaskExRequest.Task.ReactivationDateSpecified = true;  //processTaskExRequest.Task.AppointmentStart = appointmentStart;  //processTaskExRequest.Task.AppointmentStartSpecified = true;  //processTaskExRequest.Task.AppointmentFinish = appointmentFinish;  //processTaskExRequest.Task.AppointmentFinishSpecified = true;  processTaskExRequest.Task.UseDistrictCalendar = false;  processTaskExRequest.Task.UseDistrictCalendarSpecified = true;  //Customer  processTaskExRequest.Task.Customer = "Cristina Hardie";  processTaskExRequest.Task.ContactPhoneNumber = "+447720449438";  processTaskExRequest.Task.CustomerEmail = "dhardie@gmail.com";  processTaskExRequest.Task.ContractOfferingFamily = "Gold";  processTaskExRequest.Task.TaskLanguages = "English;French";  //processTaskExRequest.Task.Contract = new SO\_WebServices.ScheduleServiceInt.GEHCContractReference() { ID = "MASTERDARK1" };  //Location  processTaskExRequest.Task.District = new SO\_WebServices.ScheduleServiceInt.DistrictReference() { Name = "UK South" };  //processTaskExRequest.Task.IsMobileSite = true;  //processTaskExRequest.Task.IsMobileSiteSpecified = true;  processTaskExRequest.Task.Street = "39 Bramble Lane";  processTaskExRequest.Task.City = "Ashford";  processTaskExRequest.Task.Street = "Kent";  processTaskExRequest.Task.Postcode = "TN25 5AB";  processTaskExRequest.Task.CountryID = new SO\_WebServices.ScheduleServiceInt.CountryReference() { Name = "UNITED KINGDOM" };  //Resource  processTaskExRequest.Task.PreferredFSEs = "FHollande;Nicolas Sarkozy";  processTaskExRequest.Task.ExcludedFSEs = "JChirac;FMitterrand;Vd’Estaing";  processTaskExRequest.Task.RequiredFSEs = "FHollande";  if (taskTypeReferences[i].Name == "Training")  {  string[] trainees = { "Trainee01" };  List<SO\_WebServices.ScheduleServiceInt.EngineerReference> engineerReferences = new List<SO\_WebServices.ScheduleServiceInt.EngineerReference>();  foreach (string trainee in trainees)  {  SO\_WebServices.ScheduleServiceInt.EngineerReference engineerReference = new SO\_WebServices.ScheduleServiceInt.EngineerReference() { ID = trainee };  engineerReferences.Add(engineerReference);  }  processTaskExRequest.Task.RequiredEngineers = engineerReferences.ToArray();  }  //Set the IsMST flag if mor ethan one request is made  if (RequestCount >1)  {  processTaskExRequest.Task.IsMST = true;  processTaskExRequest.Task.IsMSTSpecified = true;  }  else  {  processTaskExRequest.Task.IsMST = false;  processTaskExRequest.Task.IsMSTSpecified = true;  }  //Requirements  processTaskExRequest.Task.SkillLevel = 1;  processTaskExRequest.Task.SkillLevelSpecified = true;  //Details  processTaskExRequest.Task.OwnerName = "Charles de Gaulle";  processTaskExRequest.Task.OwnerSSO = "Charles de Gaulle";  processTaskExRequest.Task.MacroVersion = "MacroVersion001";  //Parts  processTaskExRequest.Task.PartEstimatedDeliveryDate = now;  processTaskExRequest.Task.PartEstimatedDeliveryDateSpecified = true;  processTaskExRequest.Task.PartDeliveryType = "In the back of a van";  processTaskExRequest.Task.PartComment = "Bend your knees when you pick it up, its heavy!";  //To be derrived  processTaskExRequest.Task.Region = new SO\_WebServices.ScheduleServiceInt.RegionReference() { Name = "UK" };  processTaskExRequest.Task.NumberOfRequiredEngineers = 1;  processTaskExRequest.Task.NumberOfRequiredEngineersSpecified = true;  processTaskExRequest.Task.DueDate = dueDate;  processTaskExRequest.Task.DueDateSpecified = true;  processTaskExRequest.Task.Longitude = 918754;  processTaskExRequest.Task.LongitudeSpecified = true;  processTaskExRequest.Task.Latitude = 51194268;  processTaskExRequest.Task.LatitudeSpecified = true;    processTaskExRequests[i] = processTaskExRequest;    }  return processTaskExRequests;  }    SO\_WebServices.ScheduleServiceInt.GetTasksRequest GetTasksRequestByCallID(string CallID)  {  SO\_WebServices.ScheduleServiceInt.GetTasksRequest getTasksRequest = new SO\_WebServices.ScheduleServiceInt.GetTasksRequest();  getTasksRequest.OptionalParameters = new SO\_WebServices.ScheduleServiceInt.OptionalParameters();  getTasksRequest.OptionalParameters.CallerIdentity = "MUST";  getTasksRequest.RequestedProperties = taskRequestedProperties;  //getTasksRequest.Indexes.  List<SO\_WebServices.ScheduleServiceInt.IndexesIndex> indexesIndexs = new List<SO\_WebServices.ScheduleServiceInt.IndexesIndex>();  SO\_WebServices.ScheduleServiceInt.IndexesIndex indexesIndex = new SO\_WebServices.ScheduleServiceInt.IndexesIndex();  indexesIndex.LowBound = new SO\_WebServices.ScheduleServiceInt.Property[1];  indexesIndex.LowBound[0] = new SO\_WebServices.ScheduleServiceInt.Property() { Name = "CallID", Value = new SO\_WebServices.ScheduleServiceInt.KeyValue() { Value = CallID } };    indexesIndex.HighBound = indexesIndex.LowBound;    indexesIndexs.Add(indexesIndex);  getTasksRequest.Indexes = new SO\_WebServices.ScheduleServiceInt.Indexes();  getTasksRequest.Indexes.Index = indexesIndexs.ToArray();  return getTasksRequest;  }  public static bool IsEven(int value)  {  return value % 2 == 0;  }  public static bool IsOdd(int value)  {  return value % 2 != 0;  }  }  } |

1. ClickSchedule uses a single, combined name property OOTB instead of separate first and last names. [↑](#footnote-ref-1)
2. For those engineers that are in MUST. [↑](#footnote-ref-2)
3. This is essentially the opposite of a “Contractor” indicator and provides OOTB access to the same data. [↑](#footnote-ref-3)
4. As with all name storage in ClickSchedule, this is a composite of first and last names. [↑](#footnote-ref-4)
5. As the FSE is expected to contact their manager in some scenarios, it might be helpful to have this value so that it can be displayed on the mobile device under the My View app. This will be removed if GEHC cannot supply this value. [↑](#footnote-ref-5)
6. Permits Dispatchers to filter their task list to show only tasks created by themselves, or managers to find tasks created by specific Dispatcher(s). [↑](#footnote-ref-6)