ARCOS LLC
8800 Lyra Drive, Suite 200
Columbus, OH 43240
(614) 396 – 5500

ARCOS LLC

Application Programming Interface (API)

Platform Integration Manual

March 2024
Release 24.16

ARCOS LLC
8800 Lyra Drive, Suite 200
Columbus, OH 43240
(614) 396 – 5500

Contents

ARCOS LLC
8800 Lyra Drive, Suite 200
Columbus, OH 43240
(614) 396 – 5500

Background

Technology

The ARCOS Application Programming Interface (“API”) is a Representational State Transfer (“REST”) style interface that allows clients to use Hyper Text Transfer Protocol (“HTTP”) methods to perform data procedures in ARCOS without the need for the traditional web application.

Common command line tools such as cURL(https://en.wikipedia.org/wiki/CURL) can be used to interact with the ARCOS API and perform any of the operations listed in this document.
The Web Application Description language (“WADL”) for the ARCOS API can be viewed by visiting https://qa.rostermonster.com/arcos/rest/application/wadl while authenticated in ARCOS or appropriately authenticated on a GET method call. The WADL contains a listing of the various methods that have been exposed to your instance.

For a complete XML schema reference (“XSD”), please consult the following resource:
https://qa.rostermonster.com/arcos/rest/application/wadl/xsd0.xsd

Purpose

The purpose of the ARCOS API is to allow users and organizations access to manage their data programmatically. For example, many customers choose to utilize the API for synchronizing employee data from an existing Human Resources system. A brief selection of some use cases for the ARCOS API would include:

- Seamlessly on-board employees by loading them to ARCOS as they appear in your HR system.
- Automatically create Callouts based on outage activity in an Outage Management System (“OMS”).
- Extract ARCOS Crew Manager data to perform data analytics and enable enhanced enterprise reporting with tools such as Microsoft PowerBI.
- Synchronize employee schedule data such as shifts and any variances in those shifts between your existing scheduling solutions and ARCOS.
- And many more…

ARCOS LLC
8800 Lyra Drive, Suite 200
Columbus, OH 43240
(614) 396 – 5500

Typical Usage

The API endpoints are all located at the base URL "https://rostermonster.com/arcos/rest" appropriate for a given environment, so the base URL of the QA environment API would be "https://qa.rostermonster.com/arcos/rest". All endpoints have a relative path that is appended to the base URL to define the actual URL of the endpoint (for example "https://qa.rostermonster.com/arcos/rest/crews").

This document will use the following format when describing any action performed against the API:

Method /Endpoint URL

Where **Method** is one of the standard HTTP Methods - GET, POST, PUT, DELETE and **Endpoint URL** is the path of the endpoint that should be acted upon - /cm/crews, /schedules, /loader, etc. The full path of the actual request would contain the **host** and *application path* as well (https://qa.rostermonster.com/arcos/rest/), but we will leave those out for the sake of brevity.

So, the command to obtain a collection of crews would be presented in this document as:

GET /cm/crews

while the actual URL into the API would look more like this:

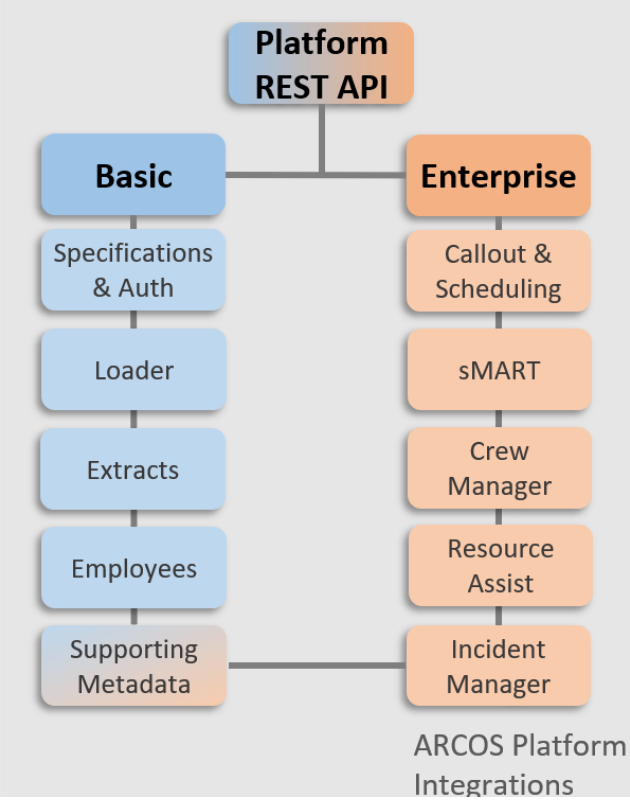
GET https://qa.rostermonster.com/arcos/rest/cm/crews.

Many endpoints will provide the ability to work with a specific object instance by specifying the id of the object explicitly. To obtain the Crew with an ID of 1007 for example, you would execute:

GET /crews/1007

ARCOS LLC
8800 Lyra Drive, Suite 200
Columbus, OH 43240
(614) 396 – 5500

License



ARCOS currently offers two Integrations Licenses, Basic and Enterprise, as described below.

Basic

The Basic Platform API License grants ARCOS Customers the following integration abilities:

- Access to the ARCOS Platform API specifications and authorization methods. Generally, access to the API.
- Access to the ARCOS Platform Loader and all associated functionality, includes loading:
  - Employee Data, including Extended Attributes
  - Schedule Event Data (SDI flat-file 15-day exclusive periods only)
  - Overtime Data
  - Crew Manager Work Order Data
  - Crew Manager Resource Data
  - Crew Manager Lodging Data
  - Access to the ARCOS Platform Extracts. This is typically for legacy reports that used to be dropped onto sFTP sites. Engineering may choose to support and develop new extracts available only via API as needed. By default, new customers will not have any Extracts available but will have access to the Extract methods.

- Access to the ARCOS Platform Employee Data. Basic customers have two options for managing employee data, loading data via the Loader and loading or extracting employee data directly from the employee's endpoint methods.
- Finally, the Basic API includes several Metadata endpoints for gathering information needed to support other API calls such as Contact IDs, Event IDs, Classification IDs, and Location IDs.

Enterprise

The Enterprise Platform API License grants ARCOS Customers, in addition to Basic API, the following integration abilities:

- Access to the ARCOS Platform Callout Data
  - Summarized and Detailed Callout data is available for any callout(s) in the Platform
  - Customers can generate new Callout's via the API
  - Callouts can be initiated via API, no need to login to the ARCOS Platform

- Access to sMART API Functionality
- Access to Crew Manager API Functionality
  - The ability to run audit reports and other reporting functionality
  - Manage Lodging
  - Manage Crew Makeup
  - Manage Resource Types and Resources
  - Manage Attributes at all Resource Levels (Employee, Resource, Lodging, Crew)

- Access to Resource Assist API Functionality
  - The ability to Extract, Import, and Delete RA Documents
- Access to Incident Manager API Functionality
  - Manage Events, Teams, Processes, Process Teams, Categories, Locations, Scenarios, Functions, Levels, Role Types, Role Assignments, Event Wrappers, Storm Codes, Event Categories, Nodes, States, Role Requests, Exceptions, Phases, Documents, Log Entries, Settings, and Associated Callouts.
- Access to Platform Schedule API Functionality
  - The ability to Extract, Load and Modify both Shifts and Schedules across the ARCOS platform.

ARCOS LLC
8800 Lyra Drive, Suite 200
Columbus, OH 43240
(614) 396 – 5500

HTTP Functionality

HTTP Request Methods

HTTP defines a set of request methods to indicate the desired action to be performed for a given resource. Although they can also be nouns, these request methods are sometimes referred to as *HTTP verbs*. Each of them implements a different semantic, but some common features are shared by a group of them: e.g. a request method can be safe, idempotent, or cacheable. Listed below are the methods supported by the ARCOS API. Source and backlinks: MDN Web Docs.

GET
The GET method requests a representation of the specified resource. Requests using GET should only retrieve data.

POST
The POST method is used to submit an entity to the specified resource, often causing a change in state or side effects on the server.

PUT
The PUT method replaces all current representations of the target resource with the request payload.

DELETE
The DELETE method deletes the specified resource.

PATCH
The PATCH method is used to apply partial modifications to a resource.

Specifications:

Specification	Title	Comment
RFC 7231, section 4: Request methods	Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content	Specifies GET, HEAD, POST, PUT, DELETE, CONNECT, OPTIONS, TRACE.
RFC 5789, section 2: Patch method	PATCH Method for HTTP	Specifies PATCH.

ARCOS LLC
8800 Lyra Drive, Suite 200
Columbus, OH 43240
(614) 396 – 5500

ARCOS Method Endpoint Examples

All examples shown are for the QA environment. To switch examples to Production, the base host changes from **qa.rostermonster.com** to **prod.rostermonster.com** and the API Access Key is updated as needed. The Access Key is obtained within ARCOS on the Sys Admin > Rest Config page. All calls for both QA and PROD are performed via the HTTPS protocol. Any call examples provided are in the form of cURL shell commands.

The Request Body for most documented calls may be issued in JSON or XML format. The Response Body typically may be returned in either JSON or XML format. To switch the formatting of the Response Body, pass the appropriate Accept Header. Valid Accept headers for most endpoints in this document are as follows:

- Accept: application/json
- Accept: application/xml
- Accept: */*

Any exceptions to the above statements will be noted where they apply.
Endpoints are unique URLs that represents a resource ARCOS has made available to the client. Below are some examples.

Employee: https://qa.rostermonster.com/arcos/rest/employees

Schedule: https://qa.rostermonster.com/arcos/rest/schedules/records

Crew: https://qa.rostermonster.com/arcos/rest/cm/crews

Endpoints may be further qualified by adding query string or path parameters as described further in this section. Below are some examples.

Query String:
https://qa.rostermonster.com/arcos/rest/employees?searchType=firstName&searchString=Joe

Path: https://qa.rostermonster.com/arcos/rest/cm/crews/2132

For simplicity, the remainder of this document will exclude the base URL, *qa/prod.rostermonster.com*, and the endpoint path part that consistently remains the same, */arcos/rest*, when displaying supported operations. All Provided supported operations will be documented like the following example:

Documented Operation Path: getRestConfigXml – **GET** /application/wadl

Complete Operation Path: getRestConfigXml
– **GET** https://qa.rostermonster.com/arcos/rest/application/wadl

Note the PATH difference; The full operation URL must be considered even where not further specified in this document.

API Response Status Codes

Every API call returns a single, standard HTTP status code that indicate the overall status of the transaction. Some common codes include the following:

200 OK
The request has succeeded. Data errors may still exist within the transaction, but the client and server are communicating and understanding each other.

201 Created
The POST succeeded. Data was added or changed as requested.

400 Bad Request
The request could not be understood by the server due to malformed syntax. Usually indicates something is wrong with the clients request formatting or method.

401 Unauthorized
The request requires user authentication. The client was not properly authenticated to the server. ARCOS supports Basic and Bearer Token authorization. See the Authentication section for additional detail.

404 Not Found
The server has not found anything matching the Request-URI. Usually, the clients request URL is invalid and the requested resource does not exist in the ARCOS API.

405 Method Not Allowed
The method specified in the Request-Line is not allowed for the resource identified by the Request-URI. This may indicate a valid ARCOS resource that the clients enterprise is not licensed to use. Speak to your ARCOS representative with any questions on this error.

500 Internal Server Error
The server encountered an unexpected condition which prevented it from fulfilling the request. Usually indicates something is wrong with the server or an unsupported operation is being performed on a valid resource.

ARCOS LLC
8800 Lyra Drive, Suite 200
Columbus, OH 43240
(614) 396 – 5500

Query Parameters

GET requests for endpoints may return hundreds or thousands of a given object type. Most URLs will be limited to returning some variable number of elements per time-period (1000 schedule records per hour for example). To help narrow the results, endpoints may have required or optional query parameters. These parameters can be provided by appending a query string, consisting of the "?" symbol and any number of "**key=value**" pairs separated by the "**&**" symbol, to the end of the endpoint URL.

For example, to obtain a collection of scheduled shifts between January 1, 2016, and March 31st, 2016, you would execute the command:

GET /scheduledShifts***?startTime=2016-01-01&endTime=2016-03-31***

The query string is in ***italics***.

Please refer to the specific endpoint's documentation for a complete listing of its required and optional query parameters.

Path Parameters

In addition to Query Parameters, many endpoints will support some limited Path Parameters as well. These parameters are inputted directly into the url in place of the parameter name presented, using the format **{parameterName}**. For example, the **{crewId}** Path Parameter in the following URL:

/cm/crews/{crewId}

would be replaced by the actual crewId you want to search for.

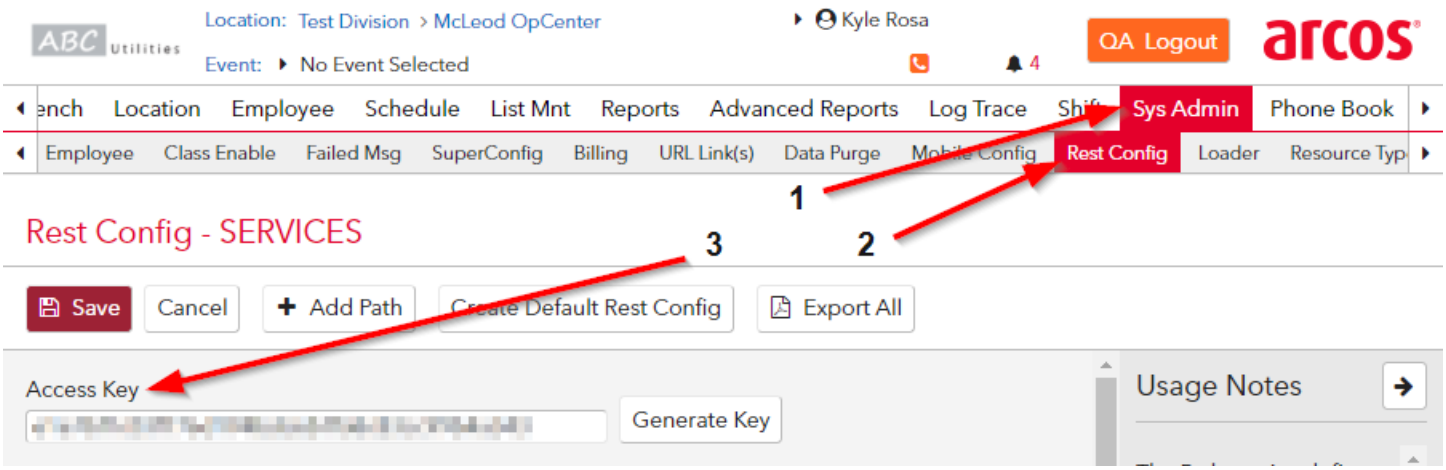
ARCOS LLC
8800 Lyra Drive, Suite 200
Columbus, OH 43240
(614) 396 – 5500

Authentication

All actions performed against the API must be authenticated. Token Authentication and standard HTTP Basic Authentication are supported.

HTTP Basic Authentication

Basic authorization is provided the Authorization header of the Request. The Header is “Authorization” while the value content is “Basic […].” […] represents your company schema, which is the short code you enter into the rostermonster.com URL when using the Web GUI, and your Access Key provided on the Sys Admin > REST Config page of the Web Application all encoded in Base64.



Above: Location of the API Access Key in the ARCOS Web Application.

Example:

Authorization: Basic Base64Encoded(COMPANY:AccessKey)

Example presented programmatically as:

“Authorization” : “Basic Q09NUEFOWTpBY2NIc3NLZXk=”

ARCOS LLC
8800 Lyra Drive, Suite 200
Columbus, OH 43240
(614) 396 – 5500

Token Authentication

Clients may use their API Access Key, available to System Administrators in Sys Admin > REST Config (See Figure above) within the Web Application, to obtain an Access Key that can be paired with their company schema and used to generate four (4) hour time limited authentication tokens to be used in future API calls.

To obtain an Access Token, issue a GET command and include your company name and accessToken as Query Parameters:

getToken - GET /auth?company=YourCompany&accessToken=accessKey

The company name, demonstrated as "YourCompany," is your Schema ID you use when navigating to the Web UI. Example: http://auss.rostermonster.com = AUSS.

The Access Token, demonstrated as "accessKey," is obtained from the ARCOS Platform Web UI, Sys Admin > REST Config tab.

Additionally, a POST method is also available allowing the user to supply the company and accessToken as part of a Multipart request payload:

postToken - POST /auth

Include a Multipart form with company and accessToken populated as described above in the getToken method.

The Response from ARCOS will contain an encoded Access Token to be used in future API Requests. The Access Token should be set in the Authorization Header of the Http Request as follows:

"Authorization" : "Bearer *[AccessTokenProvidedByArcos]*"

The Access Token may also be set in the Authorization Header of the Http Request as follows:

"Authorization" : "Token *[AccessTokenProvidedByArcos]*"
OR **"Authorization" : "*[AccessTokenProvidedByArcos]*"**

The Access Token will be valid for four (4) hours. Attempts to use an expired token will result in an HTTP return status of **401: Unauthorized**.

API clients can obtain an updated token at any time but should probably do so before the 4-hour window has elapsed to ensure operations are handled appropriately.

ARCOS LLC
8800 Lyra Drive, Suite 200
Columbus, OH 43240
(614) 396 – 5500

Method Categories and Types

The following sections provide information about how to use each of the methods available in the API. Each section will start with a brief description that explains how the method is used and is followed by a listing of the structures involved in any of the method supported operations.

Each of the supported operations will be presented with the HTTP Method to invoke and the URL to access for the operation. A table explaining the potential parameters for an operation will be given as well.

Some method sections present example calls to help illustrate potential use cases.

Application Data

Supported Operations

getRestConfigXml - GET /application/wadl

The Web Application Description Language (WADL) provides the ARCOS data format specification for use in building programmatic interfaces. The WADL Presented is unique based on the customer environment configuration. All licensed and configured methods will be provided in the WADL dynamically.

Response Payload

XML collection of the available API methods as documented throughout this manual.

getXsd - GET /application/wadl/xsd0.xsd

XSD (XML Schema Definition) is a World Wide Web Consortium (W3C) recommendation that specifies how to formally describe the elements in an Extensible Markup Language (XML) document. This operation provides a copy of all ARCOS Elements (“Data Structures”).

Response Payload

XML collection of ARCOS Data Structures. These Data Structures are also available in the Appendix – XSD section of this document.

ARCOS LLC
8800 Lyra Drive, Suite 200
Columbus, OH 43240
(614) 396 – 5500

Loader Data

The Loader Data Endpoint provides services to interact with the ARCOS file Loader. All existing loader file types are fully supported by the API. Typically, file loads are limited to specific times of day.
Please contact an ARCOS representative to determine the valid file loading times for your company.

Loader file specifications can be found in the Appendix – Flat File Specifications section of this document. Loader data structures can be found in the Appendix – XSD section of this document.

Supported Operations

getResults – GET /loader/results

Returns a collection of load results for loads that occurred between startDate and endDate. If no startDate/endDate is provided, then returns results for any loads that occurred in the past one (1) year.

Request Query Parameters

Parameter	Type	Required	Default	Description
startDate	Datetime	no	Now - 365 Days	
endDate	Datetime	no	Now	

getLoadSummary - GET /loader/results/{loadId}

Returns a collection of the load results for the matching {loadId}. If the loadId is not provided then returns results for any loads that occurred in the past one (1) year.

Request Query Parameters

Parameter	Type	Required	Default	Description
loadId	Int	yes	Now - 365 Days	

getLoadDetail - GET /loader/results/{loadId}/details

Returns a collection of the detailed load results for the matching {loadId}. {loadId} is required.

Request Query Parameters

Parameter	Type	Required	Default	Description
resultType	String	no	All Types	Used to filter result messages. Supply: Success, Warning, or Error.

uploadFile – POST /loader/requests

Load or test loading a supported flat file to ARCOS.

Request Payload

Multipart body that may contain the following form attributes:

Request Form Parameters

Parameter	Type	Required	Default	Description
validateOnly	String	No	0	If set to 1, a collection of possible Warnings will be returned, but no file load will be performed. If 0, ARCOS will attempt to load the file and modify system data.
file	UTF-8 encoded flat file target	Yes	None	The path of the file to be loaded. Flat File specifications are provided in the Appendix of this manual.
type	String	Yes	None	Valid values for the 'type' required parameter are: HRI, HRI1, IEX, OTI, OTI2, SDI, SEA, SHIPS, STARS, ULT, CMW1, CMR1, and CML1. Not all types will be available for all companies. If an unsupported type is provided, the load request will be denied.
runDate	Datetime	No	Now	
arg5	None	No	None	Not Used.

Response Payload

A loadRequest element representing the status of the operation. The 'loadId' field of each loadRequest will be set to the Load ID field of the newly created operation if successful or to zero if a failure occurred.

Example loadRequest

```
<loadRequest loadId="1633" filename="SERVICES-HRI-SERVICES-20210604.DAT" type="HRI1"
        runDate="Fri Jun 04 11:21:28 EDT 2021" dataRows="1"
        loadStatus="Data rows loaded and load initiated. Use loadId to check status">
        <warnings/>
</loadRequest>
```

Example loadResponse

```
<loadRequest loadId="1633" filename="SERVICES-HRI-SERVICES-20210604.DAT"
        type="HRI1" runDate="Mon Jun 28 13:14:14 EDT 2021" dataRows="6"
        loadStatus="Data rows loaded and load initiated. Use loadId to check status">
        <warnings/>
</loadRequest>
```

ARCOS LLC
8800 Lyra Drive, Suite 200
Columbus, OH 43240
(614) 396 – 5500

To load a file using the API

1. Following the guidance of the preceding sections of this manual, ensure you have a tool established to perform the API operation such as cURL.

2. Ensure you have a proper Authentication method prepared as described in the Authentication section of this manual.

3. Create the flat file that you desire to load by referring to the appropriate Flat File Specifications provided in the Appendix of this manual.

4. Execute a POST operation on the /loader/requests endpoint with appropriate parameters as defined in the uploadFile method section above.

5. The **loadId** provided in a successful response can be used with **getLoadSummary** and **getLoadDetail** as a way to retrieve the load results.

Example: Load HRI1 file to ARCOS QA using cURL

```
curl -i -X POST -H "Authorization: Basic [base64EncodedUserPass]"
-F "file=@CUSTOMER-HRI1-CUSTOMER-YYYYMMDDHHMM.DAT"
-F "type=HRI1" https://qa.rostermonster.com/arcos/rest/loader/requests
```

Example: The API will respond with the LoadRequest

```
<loadRequest loadId="1" filename="HRI_file.DAT" type="HRI" runDate="Wed Nov 09 13:29:46 EST
2016" dataRows="26" loadStatus="Pending"/>
```

ARCOS LLC
8800 Lyra Drive, Suite 200
Columbus, OH 43240
(614) 396 – 5500

Web UI Functionality

To optimize data importing, the loader tool can be used through the API as well as the web UI. ARCOS data importation is done through a flat file system.

To build a file

The Loader functionality allows customers the ability to load their own flat files (HRI, OTI, SDI, etc) into the ARCOS application. The most common file type available for loading these files is CSV. We will use a .csv file to show the processes of building a Human Resources Interface (HRI) file type. Please reference HRI (Human Resources / Employee Data) in this document appendix for formatting details.

Row / Column

For the HRI the required fields are –

HEADER -
- Record Type = 100
- Company Identifier = COMPANY NAME
- Export Date and Time = Time set for the export to take place
- Export Record Count = 1

BASIC EMPLOYEE DATA
- Record Type =200
- Emp_Id = 12345
- WEB Login ID = 98745
- VRU-ID = 11223344
- First Name = George
- Last name = Michael
- Job Classification Code = 1001
- Location Code = 9990

To load a file

Click on the Load Data button to begin the data loading process.

The ensuing page is composed of several ‘panes’. The pane at the top left of your screen contains the tools needed to load the data file.

Select File to Import > *Choose File* option allows searching your computer for a file to load
Type: Represents the type of data being loaded, or the Data Loader to use, i.e., HRI, OTI, SDI, etc. This selector is not used to specify the file type (extension) like .DAT, .txt or .xlsx.

The **Type** dropdown will contain the following *Load Types*:
- HRI
- HRI1
- IEX
- OTI
- OTI2
- SDI
- SEA
- SHIPS

- STARS
- ULT
- CMW1
- CMR1
- CML1

Import: provides the opportunity to preview your file; problems with fields/contents are indicated in the preview so that the errors can be corrected before the file is loaded. Hover-over help provides detail of the problem.

NOTE: Some corrections you could make on screen, through the Preview. For example, changing ONE date in ONE field makes sense to do in the Preview. Other corrections you would not make on screen. Major corrections you would make to the file and then re-Import the corrected file, for loading.

Load: loads the file AS MANIPULATED in the Data Preview.

In the event that the file you are attempting to Import/Load is not formatted as expected for the load Type you have indicated (in the Type dropdown), ARCOS will clearly identify the problems so that you can correct them. In the example below, you can see two types of validation warnings: the yellow triangle indicates a potential problem, whereas the red circle indicates an issue that would cause the load to fail.

Another pane on this page – the top right of your screen - provides file details

Data Preview includes *Search*, *Replace* and *Mask* options.

Search will find a string in any field/row in the data preview.

Replace allows “Find/Replace” (like MS Word) with dropdowns to identify the record type to search, the field/column to search.

Mask allows masking or hiding certain fields as part of the load process, so that existing data (in the ARCOS database) is overwritten with dummy data. Example: phone numbers present in the load file may be masked by 111-111-1111 when loaded to ARCOS. Similarly, masking e-mail addresses in the load file prevents an e-mail address from being loaded/saved into ARCOS.

NOTE: Once the Masking tool is used, the masked data cannot be restored by disabling the [Mask] flag. That is, for example, once phone numbers are masked by 1111111111, unchecking the Mask > Phone Numbers box does not restore the phone numbers that were previously masked. To restore these fields, the file will need to be imported again.

These tools will assist with making some corrections to the data in the file without leaving ARCOS.

Finally, a Validation pane at the right of the *Data Preview*, provides a concise list of any validation errors encountered in the imported file.

Validation pane lists errors or warnings found in the imported file, along with counts (how many times the error occurred)

Additional Notes on using the Load Data functionality

1. Ensure that your data file/type
2. Does not strip leading zeros
3. Does not convert dates entered without / to their exponential representations
4. Retaining the file name required by the automated Data Loader will allow the ARCOS System to recognize the appropriate load Type so that the ARCOS user doesn't need to select a Type.
5. If the filename is NOT in the format required by the ARCOS Data Loader, it will be the responsibility of the ARCOS user to select the correct Type for the file during the manual load.
6. File types supported (for loading from) are CSV or TXT, AND the same .DAT and/or encrypted file type that the automated Data Loader would be expecting. That is, if the file you're manually loading was created for the automated Data Loader, and is encrypted as ARCOS would expect, the manual Load option will recognize the encryption and decrypt appropriately.
7. There is no 'schedule' component to this new functionality. It is not possible to schedule a load to take place, using this functionality, at a future date/time.
8. Security: System Admin – New Security feature for Load Data will be made available, along with existing View Load Results feature

ARCOS LLC
8800 Lyra Drive, Suite 200
Columbus, OH 43240
(614) 396 – 5500

XWalks Data

The XWalks endpoint can be used to obtain cross walk IDs for Employees, Locations, and Job Classifications.

Supported Operations

getValues - GET /xwalks

This method returns a collection of Internal IDs and Customer IDs for Locations, Employees, and Classifications.
With the typical accepts header (*/*), the data returned by this method is JSON formatted, an unusual default compared to the majority of our APIs. This method may be forced to return XML formatted structures by passed an accept header of “application/xml” as described in the Background section of this manual.

Request Query Parameters

Parameter	Type	Required	Default	Valid Values	Description
type	String	Yes	None	LOC, EMP, CLASS	LOC – Locations EMP – Employees CLASS – Classifications You may provide this prarmeter multiple times to return multiple types.
val	String	No	None	Any Customer ID	Returns only results with a matching Customer ID.
searchType					

ARCOS LLC
8800 Lyra Drive, Suite 200
Columbus, OH 43240
(614) 396 – 5500

Subscriptions Data

The Subscriptions endpoint may be used to open a DataStream that will push data in real time whenever employee schedule changes are made. Note, Subscription services may require additional configuration by ARCOS.

Supported Operations

subscribeForPush - GET /subscriptions

This method is published in our Web Application Description language (WADL) but not yet described here. Subscription services may require additional configuration by ARCOS. If you require additional information on this method, please speak with an ARCOS Representative.

Request Query Parameters

Parameter	Type	Required	Default	Description
topic	String	no	Now - 365 Days	
mediaType	Datetime	no	Now	

cancelSubscription - DELETE /subscriptions

This method is published in our Web Application Description language (WADL) but not yet described here. Subscription services may require additional configuration by ARCOS. If you require additional information on this method, please speak with an ARCOS Representative.

ARCOS LLC
8800 Lyra Drive, Suite 200
Columbus, OH 43240
(614) 396 – 5500

Employee Data

The Employee endpoint can be used to obtain and modify data that is normally access through the Employee tab in the ARCOS Web Application. Employee standard properties will be returned using a collection of Attribute elements. Each Attribute will contain a 'key' field that can be used to uniquely identify the Employee property that is represented by the Attribute. A listing of the Attribute keys and the properties they represent is below. Some of the keys may not be available, depending on customer configuration. When creating and updating Employees using the API, the Attribute 'key' must be supplied.
Employee data structures can be found in the Appendix – XSD section.

Attribute Keys

Key	Description of the Attribute
vruId	The Voice Response Unit (VRU) ID
webId	The Web ID
empId	The Employee ID
location_id	Level 4 Location
class_id	Primary Class
status_id	Employee Status (1 = Active, 2 = Inactive, 5 = Deleted)
seniority	Seniority Date
service	Service Date
birth	Birthdate
altDate	Alternative Date
curr_assign	CurrAssign Date
new_emp	Is this a new Employee?
supervisor_id	The contactId of the Employee Supervisor
is_supervisor	Is this Employee a Supervisor?
sec_group_id	Security Group
la_group_id	Location Access Group
first_name	First name
middle_name	Middle name
last_name	Last name
nickname	Nickname
gender	M or F

comment	Availability Comment for the Employee
en_only	Employee Type
oms_id	Outage Management System ID
oms_user	Is this Employee an OMS User?
vehicle_type_id	The ID of the Vehicle Type for this Employee
no_cdl	No CDL
record_flag	Can this Employee Record Outbound Messages?
skips	Skips
adj_hours	Adjusted Hours for the Employee
vehicle_id	The ID of the Vehicle for this Employee
callback_emp_rel	Should this Employee be called after being released?
vehicle_str	The name of the Employee's Vehicle
radio	Radio for the Employee
exah_flag	EXAH
pay_type	Salaried or Hourly
shirt_size	S, M, L, XL, 2XL, 3XL, 4XL, 5XL
crew_leader	Is this Employee a Crew Leader?
rep_to_id	The ID of the Report to Location
sched_alert	Should this employee get a schedule alert?
work_loc	The name of the Work Location
hrs_worked_alert	Should this employee get an hours worked alert?

TransferPhone, CoPhone, and Pager devices should maintain a proper call order sequence with no overlap between devices. Only 3 email addresses are supported at this time. The number of WebDevices will vary with customer.

Supported Operations

searchEmployees - GET /employees

Returns a collection of Employee elements that match the provided query. This method supports pagination using the 'page' parameter, 'X-Arcos-Page' header, and 'X-Arcos-Total-Pages' header.

If the response payload could contain over 5000 employee records, the use of pagination is required or the request may fail with an error that the response contains over 5000 records and pagination is required.

Request Query Parameters

Parameter	Type	Required	Default	Valid Values	Description
searchType	String	Yes	None	contactid, phone, lastname, firstname, webid, vruid, email, locationaccessid, securitygroupid, deviceserviceid, classid, empid, extendedattribute, radio, locationid	The type of search to perform using the searchString parameter. **The searchType is case sensitive, so contactid does not equal contactID**.
searchString	String	Yes	None		The value for the searchType to use when finding Employees.
changedSince	Datetime	No	None	Reference Definitions	Only return records that have been modified since this date.
userChangesOnly	Bool	No	0	0, 1	Exclude from results any changes made by the Data Loader / API.
statusType	Int	No	None		
locationId	Int	No	None		
LocationName	String	No	None		
page	Int	No, unless response may contain over 5000 employee elements	None	Number of page	Activates and controls pagination for responses.
rosters	Int	No	None	1	Enable roster location / classification / List Status associated to the employee be displayed.

getEmployee - GET /employees/{contactId}

Returns the Employee element that matches the contactId. The “contactId” is the unique ARCOS identifier for the employee element.

Request Query Parameters

Parameter	Type	Required	Default	Valid Values	Description
contactId	int	No	None	contactId	Only show records that are for the contactId provided.

createEmployee - POST /employees

Create a new Employee.

Request Payload

The Employee element to create. The ‘contactId’ should be set to zero (“0”) when creating employees. Each Attribute object must contain the appropriate ‘key’ field for the save operation to be successful. Payload may also be supplied as Multipart and allows the following additional form parameters.

Request Form Parameters

Parameter	Type	Required	Default	Valid Values	Description
saveOptions		No			
noRosterRules		No			
noOverwrite		No			

Example

```
<employee contactId="0" statusColor="#000000">
  <attributes>
    <attribute name="vruId" value="7600002" key="vruId"/>
    <attribute name="webId" value="7600002" key="webId"/>
    <attribute name="status" value="Delete" id="5" key="status_id"/>
    <attribute name="firstName" value="Kyle" key="first_name"/>
    <attribute name="lastName" value="Rosa" key="last_name"/>
    <attribute name="location" value="McLeod OpCenter" id="9999" key="location_id"/>
    <attribute name="classification" value="Arcos User" id="500" key="class_id"/>
    <attribute name="securityGroup" value="Level 0" id="999" key="sec_group_id"/>
    <attribute name="locationAccessGroup" value="All Access" id="0" key="la_group_id"/>
    <attribute name="isSupervisor" value="0" key="is_supervisor"/>
    <attribute name="seniorityDate" value="10/13/2020" key="seniority"/>
    <attribute name="employeeType" value="Regular" id="0" key="en_only"/>
  </attributes>
  <address/>
  <emails>
    <email index="1" address="krosa@arcos-inc.com"
          description="krosa_ai" enabled="true" condensed="false" emailProtected="false"/>
  </emails>
  <phones>
    <phone sequence="1" duty="false" trusted="false" pinReq="false" smsEnabled="false"
          number="(111) 111-1112"
          phoneType="Regular Phone" firstPause="" firstTouchTone="" secondPause="" secondTouchTone=""
          comments=""/>
```

```
  </phones>
  <pagers/>
</employee>
```

Response Payload

An OpStatus element representing the status of the operation. The ‘id’ field of each OpStatus will be set to the contactId field of the newly created record if successful or to zero if a failure occurred.

Example

```
<opStatus id="1884" code="1" message="Contact Data saved successfully: Rosa, Kyle"/>
```

ARCOS LLC
8800 Lyra Drive, Suite 200
Columbus, OH 43240
(614) 396 – 5500

createEmployees - POST /employees/batch

Create multiple Employees.

Request Payload

The collection of Employee elements to create. The ‘contactId’ should be set to zero (“0”) when creating employees. Each Attribute object must contain the appropriate ‘key’ field for the save operation to be successful. Payload may also be supplied as Multipart and allows the following additional form parameters.

Request Form Parameters

Parameter	Type	Required	Default	Valid Values	Description
saveOptions		No			
noRosterRules		No			
noOverwrite		No			

Example

```
<employees>
 <employee contactId="0" statusColor="#000000">
  <attributes>
   <attribute name="vruId" value="7600002" key="vruId"/>
   <attribute name="webId" value="7600002" key="webId"/>
   <attribute name="status" value="Delete" id="5" key="status_id"/>
   <attribute name="firstName" value="Kyle" key="first_name"/>
   <attribute name="lastName" value="Rosa" key="last_name"/>
   <attribute name="location" value="McLeod OpCenter" id="9999" key="location_id"/>
   <attribute name="classification" value="Arcos User" id="500" key="class_id"/>
   <attribute name="securityGroup" value="Level 0" id="999" key="sec_group_id"/>
   <attribute name="locationAccessGroup" value="All Access" id="0" key="la_group_id"/>
   <attribute name="isSupervisor" value="0" key="is_supervisor"/>
   <attribute name="seniorityDate" value="10/13/2020" key="seniority"/>
   <attribute name="employeeType" value="Regular" id="0" key="en_only"/>
  </attributes>
  <address/>
  <emails/>
  <phones/>
  <pagers/>
 </employee>
</employees>
```

Response Payload
Common OpStatus.

Example

```
<opStatuses>
        <opStatus id="4290" code="1"
        message="VRU Pin has been reset.&#10; for: Polo, Marco&#10;Contact Data saved successfully: Polo,
Marco"/>
</opStatuses>
```

updateEmployee - PUT /employees

Update an existing Employee.

Request Payload
The Employee element to update. Each Attribute object must contain the appropriate 'key' field for the save operation to be successful. Payload may also be supplied as Multipart and allows the following additional form parameters.

Request Form Parameters

Parameter	Type	Required	Default	Valid Values	Description
saveOptions		No			
noRosterRules		No			
noOverwrite		No			

Example

```
<employee contactId="1884" statusColor="#000000">
  <attributes>
    <attribute name="vruId" value="5717781606" key="vruId"/>
    <attribute name="webId" value="krosa_ai" key="webId"/>
    <attribute name="status" value="Active" id="1" key="status_id"/>
    <attribute name="firstName" value="Kyle" key="first_name"/>
    <attribute name="lastName" value="Rosa" key="last_name"/>
    <attribute name="fullName" value="Kyle,Rosa" key="full_name"/>
    <attribute name="location" value="McLeod OpCenter" id="9999" key="location_id"/>
    <attribute name="classification" value="Technician" id="1145" key="class_id"/>
    <attribute name="securityGroup" value="Admin" id="500" key="sec_group_id"/>
    <attribute name="locationAccessGroup" value="All Access" id="0" key="la_group_id"/>
    <attribute name="isSupervisor" value="1" key="is_supervisor"/>
    <attribute name="supervisor" value="Sullivan, Jake P." id="1950" key="supervisor_id"/>
    <attribute name="seniorityDate" value="06/30/2020" key="seniority"/>
    <attribute name="employeeType" value="Regular" id="0" key="en_only"/>
  </attributes>
</employee>
```

Response Payload

An OpStatus element representing the status of the operation. The 'id' field of each OpStatus will be set to the contactId field of the newly created record if successful or to zero if a failure occurred.

Example

```
<opStatus id="1884" code="1" message="Contact Data saved successfully: Rosa, Kyle"/>
```

getExtendedAttributes – GET /employees/extendedAttributes

This method returns a collection of all available extendedAttribute details including their internal IDs, data type, description, name, and more.

Request Query Parameters

Parameter	Type	Required	Default	Valid Values	Description
noPseudo	Bool	No	0	0, 1	If 1, no Pseudo Attributes will be provided in the response element.

Response Payload

A collection of extendedAttributes, with or without Psuedo attributes included depending on query.

Example

```
<extendedAttributes>
  <extendedAttribute attributeId="1060" name="A TEST ROLE" description="A TEST ROLE" order="999"
        dataType="VALUES" attributeStatusId="0">
    <category name="Std" id="1000"/>
  </extendedAttribute>
  <extendedAttribute attributeId="1061" name="B-TEST-ROLE"
        description="B-TEST-ROLE" order="999" dataType="VALUES" attributeStatusId="0">
    <category name="Std" id="1000"/>
  </extendedAttribute>
  <extendedAttribute attributeId="1062" name="C-TEST-ROLE"
        description="C-TEST-ROLE" order="999" dataType="TEXT" attributeStatusId="0">
    <category name="Std" id="1000"/>
  </extendedAttribute>
</extendedAttributes>
```

getExtendedAttributeValues - GET /employees/attributeValues

Retrieve data of available attributes that could be associated to an employee's account. Employee Attributes are defined by ARCOS. Values can be updated in most cases on the /employees methods. Additional attributes can be created as needed by utilizing Extended Attribute functionality, when available.

The Attribute ID's provided by this endpoint can also be used to perform extracts via the /employees GET method with the searchType set to extendedAttribute and the search string containing values such as this example:

https://qa.rostermonster.com/arcos/rest/employees?searchType=extendedAttribute&searchString=536%3DInactive%3B&page=1

Which will return all Inactive employees in the target system with pagination enabled.

Request Query Parameters

Parameter	Type	Required	Default	Valid Values	Description
attrId	int	Yes	None	Saved Set ID's provided below	Returns saved set with the matching set ID

A table of System Attributes and their IDs is available below. These System Attribute ID's may be provided to this requests attrId parameter value.

System Attributes

Attribute ID	Description
502	Vru ID (text)
501	Web ID (text)
503	First Name (text)
504	Middle Name (text)
505	Last Name (text)
506	Nickname (text)
512	Address (text)
513	Street (text)
514	Street Two (text)
515	City (text)
516	State (AL,AK,NJ, OH,UT,VA etc.)
519	Area Code (text)
510	Location (values)
507	Class (values)
537	County Code (values)
522	Email (text)
527	Employee Type (values: Regular, Crew Mgr, Notify Only, Limited Use)
520	Exchange (text)
508	Gender (text)
521	Last Four (text)
511	Location Access (values)
518	Phone Number (text)
509	Security Group (values)
524	Seniority Date (date)
536	Status (values: Active, Inactive, Delete)

532	Vehicle String (text)
517	Zipcode (text)

getEmployeeStub – GET /employees/stubs/{contactId}

Employee Stubs represent a summary of employee data and contact details. This method returns an employeeStub element for the employee with the provided {contactId}.

Response Payload

A single employeeStub element for the record matching {contactId}.

Example

```
<employeeStub contactId="1884" firstName="Kyle" lastName="Rosa" firstLastName="Kyle Rosa"
fullName="Rosa, Kyle" vruId="5717781606" webId="krosa_ai" className="Technician"
locationName="McLeod OpCenter">
  <emails>
    <email>krosa@arcos-inc.com</email>
    <email>krosa@rostermonster.com</email>
  </emails>
  <phones>
    <phone>+11111111111</phone>
  </phones>
</employeeStub>
```

getEmployeeStubs – GET /employees/stubs

Employee Stubs represent a summary of employee data and contact details. This method returns an employeeStub element for the employees matching the request query.

Request Query Parameters

Parameter	Type	Required	Default	Valid Values	Description
searchType	String	Yes	None	contactid, phone, lastname, firstname, webid, vruid, email, locationaccessid, securitygroupid, deviceserviceid, classid, empid, extendedattribute, radio, locationid	The type of search to perform using the searchString parameter. **The searchType is case sensitive, so contactid does not equal contactID**.
searchString	String	Yes	None		The value for the searchType to use when finding Employees.
activeOnly	Bool	No	0	0, 1	Only return records that are for employees with a status of Active.
contactId	Int	No	None	contactId	

Response Payload

A collection of employeeStub elements for the records matching the request query.

Example

```
<employeeStub contactId="1884" firstName="Kyle" lastName="Rosa" firstLastName="Kyle Rosa"
fullName="Rosa, Kyle" vruId="5717781606" webId="krosa_ai" className="Technician"
locationName="McLeod OpCenter">
  <emails>
    <email>krosa@arcos-inc.com</email>
    <email>krosa@rostermonster.com</email>
  </emails>
  <phones>
    <phone>+11111111111</phone>
  </phones>
</employeeStub>
```

getEmployeeWorkingStatus - GET /employees/workingStatus/{contactId}

Returns a Boolean value of true or false. If the employee identified by {contactId} is presently on a "Working" type event in the ARCOS Schedule, the returned payload is "true". If the employee is not presently on a "Working" type event, the returned payload is "false".

runRosterRules - POST /employees/runRosterRules

This method exposes an ARCOS procedures to run Roster Rules in the Callout application.

Request Form Parameters

Parameter	Type	Required	Default	Description
runTest	Bool	No	False	If True, only test the execution of roster rules. If False, the roster rules will be updated.

Response Payload

Common OpStatus.

Example

```
<opStatus id="ba028767-5212-41c7-a288-683ba49e22fc" code="1"
message="runRosterRules process started successfully with UUID = ba028767-5212-41c7-a288-
683ba49e22fc."/>
```

getSafetyNoticeDetails - GET /employees/getResponse

Retrieve safety notice details for employees within the range of startDate and endDate.

Request Query Parameters

Parameter	Type	Required	Default	Valid Values	Description
startDate	Datetime	Yes	None	Valid Datetime that is prior to the provided endDate.	
endDate	Datetime	Yes	None	Valid Datetime that is after the provided startDate.	

saveEmployeeSafety - POST /employees/saveEmployeeSafety

This method is published in our Web Application Description language (WADL) but not yet described here. If you require additional information on this method, please speak with an ARCOS Representative.

Schedule Data

An Employee's schedule within Arcos ultimately consists of a collection of ScheduleRecords. These ScheduleRecords are created whenever an Employee is given a ScheduledShift, ScheduleException, or ShiftAssignment.

ShiftAssignments essentially represent collections of ScheduledShift objects for an Employee for a given week. Once the ShiftAssignment is created for an Employee, the corresponding ScheduledShift records are automatically created by Arcos. For example, if a Shift is defined as Monday - Friday from 8:00 - 5:00pm, then creating a ShiftAssignment for an Employee for that Shift would create 5 SheduledShift records, one for each day represented in the Shift. Deleting a ShiftAssignment will remove its corresponding ScheduledShift records.

Schedule data structures can be found in the Appendix – XSD section.

Web UI Functionality

Shift Module

Shifts in ARCOS define an employee's normal work schedule for a week. Shifts are created in the **Shift Module** and are assigned to employees using the **Week Views** on the *Schedule Module*. Once a shift is created, it is available to all locations within the company. Shifts can also have a ***Refer As*** value. The **Refer As** value allows the same shift to have multiple names. For example, a shift may be called M-F 07:00-16:00 in one department but be called Standard Shift in another department.

Once a shift is created, ARCOS creates a system-generated string that it stores in the Signature column on the Shift Schedule and subsequent shift pages called the ***Shift Signature***. The **Shift Signature** describes the days of the week and the time ranges for the shift.

***Shifts must be created prior to creating an employee's schedule.

There is a mouse-over on the column titled "Auto-generated shift signature." The following abbreviations are used for the days of the week:

- Sunday = Su
- Monday = M
- Tuesday = T
- Wednesday = W
- Thursday = R
- Friday = F
- Saturday = Sa

Example 1: If a shift is set for Monday through Friday with a start time of 12:00 and an end time of 20:30, the Signature will look like this: MTWRF12-2030.

Example 2: If a shift has multiple times for the days it is scheduled, such as Monday – Wednesday 12:00 – 20:30 and Thursday-Friday 10:00 – 18:15, the Signature will look like this: MTW12-2030RF10-1815.

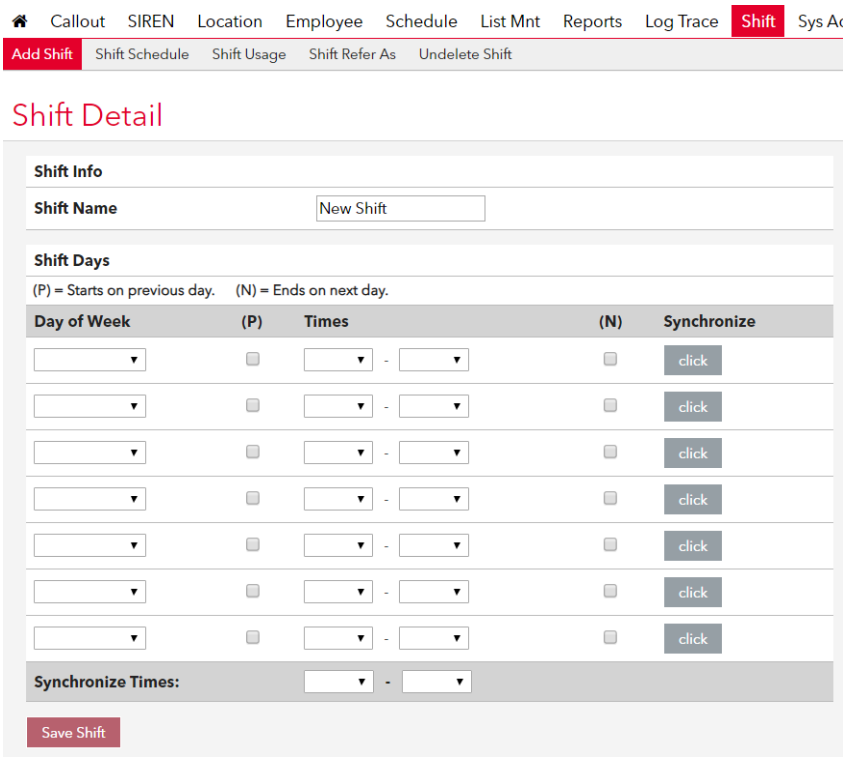
The following table provides a description of the functions associated with each button on the Shift tab.

Button	Function
Add Shift	Allows new shifts to be created in ARCOS.
Shift Schedule	Allows all shifts to be viewed. It is also the starting point for deletion and modification of shifts.
Shift Usage	Indicates how many employees have the shift assigned.
Shift Refer As	Allows Refer As values to be added and removed from shifts.
Undelete Shift	Allows deleted shifts to be restored to active status.

ARCOS LLC
8800 Lyra Drive, Suite 200
Columbus, OH 43240
(614) 396 – 5500

Adding Shifts

There is no limit to the number of shifts that can be added to ARCOS. Once a shift is added, it is available to all employees at all locations within a company. Once you create a shift, ARCOS creates a system-generated string that it stores in the Signature column on the Shift Schedule and subsequent shift pages. The information in this column is the days of the week and the time ranges for the shift as you enter it when you create a shift on the Shift Detail page. ARCOS uses this "Signature" that is created for each new, unique shift created on the Shift Detail page to prevent creation of duplicate shifts, so that you must use the Refer As functionality instead.



1. Click the **Shift** tab. The *Shift* screen displays.
2. Click **Add Shift**. The *Shift Detail* screen displays.
3. Click **OK** to the *Please enter a shift name* prompt.
4. Enter the name of the new shift in the **Shift Name** field.
5. Select the day the work week begins from the **Day of Week** dropdown menu in the first row.
6. Continue picking days of the week until you have your work week defined.
7. Select start and end times from the **Times** dropdown menus in each row and skip to *Step 9*, **-OR-**

   Select the start and end times of the two **Synchronize Times** dropdown menus at the bottom of the *Shift Days* table, if all or most of the times for each day's schedule are the same.
8. Click the Click button in the Synchronize column for each day to which you wish to copy the time that you set in the Synchronize Times field.
9. Click the checkbox in the (P) column or the (N) column, if the shift spans two days to tell ARCOS how you wish to "treat" the shift; it either starts on the previous day or ends on the next day.

10. Select P if the shift starts on the Previous day. (Selecting *Previous* indicates the shift STARTED yesterday and ENDS today.)
11. Select N if the shift ends on the Next day. (Selecting *Next* indicates the shift STARTS today and ENDS tomorrow.)
12. Repeat Step 9 for each day of the week that the schedule spans days.
13. Click Save Shift.

Note: Once the shift is assigned to employees, the shift cannot be modified.

Modifying Shifts

Shifts are modified on the **Shift Detail** screen. The *Shift Detail* screen is accessed through the *Shift Schedule* screen. The *Shift Schedule* screen is separated into two tables, one containing names of shifts that have *Refer As* names and the other containing names of shifts that do not have **Refer As** names. Modifications can be made to the Shift Name, Day of Week, Times, and the day on which the shift starts and ends.

Note: If the shift is assigned to employees, the days and times for the shift cannot be changed.

1. Click the **Shift Schedule** button on the *Shift* tab. The *Shift Schedule screen* displays.
2. Click the shift's name. The *Shift Detail page* will display.

Note: If the shift is assigned to employees, the day and time fields are grayed out.

3. Modify the necessary fields. (Modifications may be made to the Shift Name, Day of Week, Times, and the day on which the shift starts and ends.)
4. Click the Save Shift button.

Deleting a Shift

Shifts are deleted on the **Shift Detail** screen. Shifts can be deleted even if employees are currently assigned to them. When a shift is deleted with employees still assigned and working the shift, they will continue to have the shift assigned for those weeks and days and hours that are defined by the shift. However, the deleted shift will not be available to assign to employees via Week Views within the Schedule module.

Note: Once a shift has been deleted, it remains in the system and can be undeleted if needed at a future time.

1. Click the **Shift Schedule** button on the *Shift* tab. The *Shift Schedule screen* displays.
2. Click the shift's name. The *Shift Detail page* displays.
3. Check the **Delete Shift** checkbox.
4. Click the **Save Shift** button.
5. Click **OK** on the pop-up confirm deletion of the shift.

Undeleting a Shift

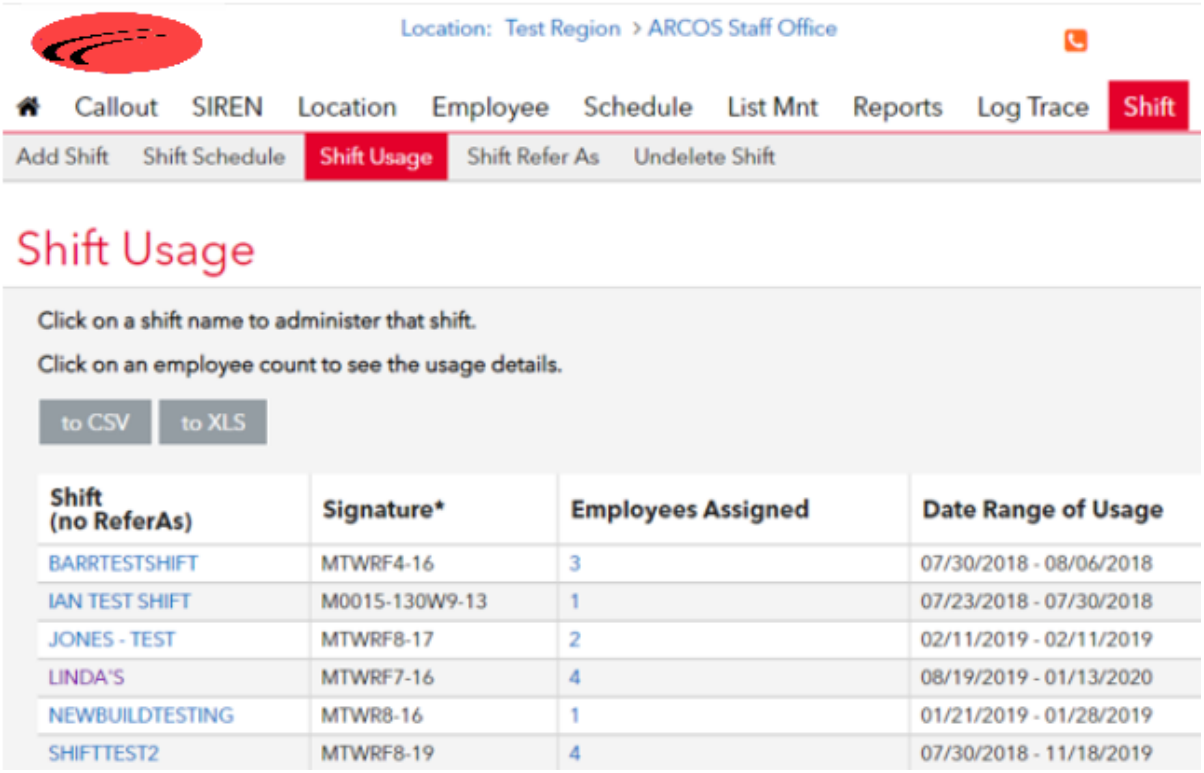
A shift that has been deleted can be restored to active status by undeleting it.

1. Click the **Undelete Shift** button on the *Shift* tab. The *Shift Schedule (Deleted Shifts) screen* displays.
2. Click the shift's name. The *Shift Detail page* displays.
3. Uncheck the **Delete Shift** checkbox.
4. Click the **Save Shift** button.

Viewing Shift Usage

The **Shift Usage** screen indicates how many employees have or have had the shift assigned to them at some point in time. It also displays the earliest and latest dates the shift has been assigned to employees through the entire company. The number of employees is a link to the **Shift Usage Detail** screen, which displays each employee who has been assigned that shift. The details include the employee's location, status, number of weeks assigned to the shift, and the date range of the shift assignment.

1. Click the **Shift Usage** button on the *Shift* tab. The *Shift Usage screen* displays.
2. Click the number in the *Employees Assigned* column next to the shift. The **Shift Usage Detail** page displays.



Schedule Module

ARCOS uses information found in the **Schedule** module to determine if an employee is available for a callout. The **Schedule** module displays employee shifts, working statuses, and schedule exceptions in a comprehensive graphical interface and provides easy administration of schedules and statuses on one screen. You can apply shifts for up to a year to one employee or several employees at a time. You can also apply rotating shifts.

There are six different views: **1 Day**, **2 Day**, **7 Day**, **14 Day**, **8 Week**, and **16 Week**. Each view provides the ability to apply selectors that limit the list based on *Location*, *Roster List*, *Primary Class*, *Supervisor,* or individually selected employees currently displayed.

The following table provides a description of the functions associated with each button on the **Schedule** tab.

Employees in **Deleted** or **Inactive** status will not display in this list.

Button	Function
16 Week View	Displays a 16-week view of shift assignments and allows modifications of the assignments. This view displays only the shift (or Refer As) name. Use this view or the 8 Week view to assign shifts to employees.
8 Week View	Displays an 8-week view of shift assignments and allows modifications of the assignments. This view displays only the shift (or Refer As) name. Use this view or the 16 Week view to assign shifts to employees.
14 Day View	Displays a 14-day view of employee work statuses, including shifts, callouts, holdovers, and exceptions. This view displays the shift (or Refer As) name **and** the actual hours the employee is scheduled to work each day. It also includes any exceptions.
7 Day View	Displays a 7-day view of employee work statuses, including shifts, callouts, holdovers, and exceptions. This view displays the shift (or Refer As) name **and** the actual hours the employee is scheduled to work each day. It also includes any exceptions.
2 Day View	Displays the current day's view plus the next day view of employee work statuses, including shifts, callouts, holdovers, and exceptions. This view displays the shift (or Refer As) name **and** the actual hours the employee hours the employee is scheduled to work. It also includes any exceptions.
1 Day View	Provides a list of current schedule exceptions, by employee, that can be modified to add or remove exceptions as required.
Schedule Exception	Provides a list of current schedule exceptions, by employee, that can be modified to add or remove exceptions as required.

Sorting Records in the Scheduler

You can sort all columns in the **Week** and **Day** views in the *Schedule* module in ARCOS. The same steps are required in any of the *Schedule* screens.

1. Place your cursor over the column by which you wish to sort.
2. Right-click the mouse and the *Sort Type selector displays*.
3. Select **Ascending**, **Descending**, or **Cancel**. The sort is executed and the screen refreshes.

ARCOS LLC
8800 Lyra Drive, Suite 200
Columbus, OH 43240
(614) 396 – 5500

Employee Schedule Selectors

Sometimes you need to limit the list of employees in a Schedule view so that you can apply shifts or perform other schedule-related tasks with a smaller group. To display a limited set of employees in the **Scheduler**, you can use one of the **Selector** options. These **Selectors** are the same in the **Week Views** and the **Day Views**. There are six **Selector** options in the **Schedule** module: Location, Roster List, Primary Class, Supervisor, Checked Only, and None. Selectors make it easier to assign shifts to multiple employees by hiding the employees who will not be assigned shifts at that time. This is especially useful when there are many employees in the same location.

Button	Function
Location or Loc	Allows you to select multiple locations for which to display schedules. Once you select locations, you can navigate away from the Schedules tab and when you return to the Schedule page, ARCOS will remember your selections. Also, before you logout, ARCOS prompts you to save any selections you made during the session.
Roster List	Allows you to display only selected the Roster List or Lists you wish to view. For example, a Roster List could include all employees on a callout list for emergencies, all employees who work maintenance, all employees who read meters, etc.
Primary Class or Pri Class	Allows you to display only Primary Class (Classification) lists you wish to view. For example, a Primary Class list could include all employees who are Journeymen, all who are Apprentices, all technicians, etc.
Supervisor	Allows you to display by Supervisor. A supervisor list only displays those employees who are direct reports of the selected supervisor. Not all companies use this feature.
Checked Only	Allows you to select employees from the list, using the checkbox to the employee names, and then view a list of only those employees and their schedules.
None	Allows you to reset your list once you have applied a filter or a combination of filters. None is the default display when you use either Week or Day View.

Displaying by Location

Displaying by location allows you to select multiple locations for which to display schedules.
The **Location** selector is available on all of the views in the **Schedule** tab. Once you select locations using the **Location** selector, you can navigate away form the **Schedule** tab and when you return, ARCOS remembers your selections and displays the Schedule page as such.

To Use the Location Filter

1. Click either a **Week View** or a **Day View** button.
2. Click **Location** in the *Filters* field.
3. Select the checkboxes to the left of the **Location** you wish to display.
4. Click **OK** on the *Location* window. The window closes and the page refreshes with employees from the

selected location.

Displaying by Roster List

A roster is a list of employees constructed either as a Workgroup or a Job Class . Both are composed in an order in which employees will be called out for work. The order is based on business rules of the company or associated union.

1. Click either a **Week View** or a **Day View** button.
2. Click **Roster List** in the *Selector* field.
3. Select the **Roster** containing the employees to which you wish to assign shifts.
4. Click **OK**. The screen refreshes and only those employees contained in the selected *Roster* display.
5. Click **None** to turn the **Roster List** Selector off and return to the full list of employees.

Displaying by Primary Class

Displaying a list sorted by **Primary Class** is not the same as by Job Class. The Primary Class selector lists all employees in a in the current location regardless of their roster list assignments.

1. Click either a **Week View** or a **Day View** button.
2. Click **Primary Class** (**Pri Class**) in the *Selector* field.
3. Select the **Primary Class** of employees to which you wish to assign shifts.
4. Click **OK**. The screen refreshes and only those employees contained in the selected *Primary Class* display.
5. Click **None** to turn the **Primary Class** Selector off and return to the full list of employees.

Displaying by Checked Only

Sometimes viewing a list by Workgroup or Primary Class will not result in displaying the employees necessary. For the situations when you need to "pick and choose" the employees in a location, use the **Checked Only** Selector.

1. Click either a **Week View** or a **Day View** button.
2. Select the checkboxes in the *Select* column on the left side of the page for each employee you wish to display.
3. Click **Checked Only** in the *Selector* field. The screen refreshes and only the selected employees display.
4. Click **None** to turn the **Checked Only** filter off.

Shift Masking

If a **Working Normal Shift** schedule record is covered (or partially covered) by a *Schedule Exception,* the portion of the shift that is covered is considered *masked*. For example, if an employee is scheduled to work

7:00 am to 4:00 pm but takes a half-day of vacation, half of the shift is "masked" by the vacation exception. This masking is indicated on the *Schedule Page* daily views by showing the shift records in a light gray font when they are masked by other exceptions. Basically, a masked shift record is ignored by ARCOS, and not displayed.

Other functions that *Shift Masking* affects:

1. **Overrides**: Consider an employee who has a scheduled shift, but is also on an exception, such as *Vacation*. A callout is performed, and Vacation is checked as an override. In the past, the employee would NOT be called, because ARCOS still considered them not available due to their shift. With the new shift masking, their *Vacation* record masks the shift record, so ARCOS considers them *Available* for the callout.
2. **Cumulative Hours Worked Calculation**: Many companies have an indicator on the *Working* page that turns an employee's row yellow or red based on how many hours they have worked. In the past, shift hours which were masked by an exception were actually counted into the *Cum Hours* calculation. Also, many companies have *Rest Rules* which perform a similar *Cumulative Work* calculation in order to determine if an employee should be placed on Rest. "Masked" shift records are no longer included in these calculations.
3. **Schedule Record Detail Popup**: If a shift is masked, even partially, there is an indicator in the Schedule Record detail popup that says, "Masked by exception?" YES.

ARCOS LLC
8800 Lyra Drive, Suite 200
Columbus, OH 43240
(614) 396 – 5500

Applying Shifts via ShiftAssign Roster Preference

ShiftAssign allows the user to assign a shift to an entire roster, in turn, scheduling all employees included in the selected roster to that shift.
This option can be found in the **Roster List Ordering** section of the **List Maintenance** tab and clicking the *admin* link for the roster that requires scheduling.

The **ShiftAssign, ShiftBaseDate** and **ShiftNumWeeks** roster preferences work together to ensure that employees on the roster always have a shift assigned.

- **ShiftAssign**: This can be a single shift name or "shift refer as" value, or a comma separated list of shift names or "refer as" values.

Note: *This field is case sensitive and must match what's listed on the* ***Shift*** *tab.*

  - Comma separated values will allow the ability to assign a rotating shift to a group of people
  - Allows up to 200 characters, if shift names are long, shift refer as values should be used
  - Can be changed once employees are assigned to the roster. You can add or remove shifts to the rotation, change the order of the rotation, change a non-rotating to a rotating and vise versus

- **ShiftBaseDate**: The date that the rotation starts. This is only used when creating a shift rotation
  - The value in this preference is mm/dd/yyyy and must be in the past.
  - The date entered would be the date the first shift within the rotation would be applied to.
  - Cannot be changed once employees are assigned to the roster.

- **ShiftNumWeeks**: The number of weeks into the future the shift configuration is applied to the employees on the roster
  - Value must be numeric and at minimum 4 and at most 99
  - Can be changed after employees are added to the roster. Shift weeks will be assigned or unassigned depending on which direction the preference is changed.

ARCOS LLC
8800 Lyra Drive, Suite 200
Columbus, OH 43240
(614) 396 – 5500

Shift Example

LINDA'S	MTWRF7-16	07:00-16:00	07:00-16:00	07:00-16:00	07:00-16:00	07:00-16:00

Shift Assign – Roster List Ordering Example

Shift Options

ShiftAssign	ShiftBaseDate	ShiftNumWeeks
LINDA'S		4

List Example

Select	Name	Shift	Mon 2/10	Tue 2/11	Wed 2/12	Thu 2/13	Fri 2/14
☐	Ballstaedt, Dan	LINDA'S	Sh 07:00-16:00	Sh 07:00-16:00	Sh 07:00-16:00	Sh 07:00-16:00	Sh 07:00-16:00
☐	Ingle, Ryan	LINDA'S	Sh 07:00-16:00	Sh 07:00-16:00	Sh 07:00-16:00	Sh 07:00-16:00	Sh 07:00-16:00
☐	Jones, Linda (Hot Mill)	LINDA'S	Sh 07:00-16:00	Sh 07:00-16:00	Sh 07:00-16:00	Sh 07:00-16:00	Sh 07:00-16:00
☐	Mansell, Doug	LINDA'S	Sh 07:00-16:00	Sh 07:00-16:00	Sh 07:00-16:00	Sh 07:00-16:00	Sh 07:00-16:00

ARCOS LLC
8800 Lyra Drive, Suite 200
Columbus, OH 43240
(614) 396 – 5500

Supported Operations

getRecords - GET /schedules/records

Returns a collection of ScheduleRecords that match the specified query.

Request Query Parameters

Parameter	Type	Required	Default	Description
startTime	Datetime	No	Now	Records returned will have started or ended at or after this date.
endTime	Datetime	No	Now + 1 Day	Records returned will have started or ended up to and including this time. The endTime - startTime range must be 15 days or less.
changedSince	Datetime	No	None	Only return records that have been modified since this date.
showInactive	Bool	No	None	Set to 1 to show deleted records in the results.
skipScheduledShifts	Bool	No	0	set to 1 to filter out ScheduledShift records.
skipExceptions	Bool	No	0	set to 1 to filter out ScheduleException records.
contactId	Array of int	No	None	Only show records that are for the contactIds provided.
locationId	Array of int	No	None	Only return records for Employees that are in the locationIds provided. locationIds will only be considered if contactId is not used.
eventId	Array of int	No	None	Only return records that have Event Types that match the eventIds provided.
userChangesOnly	Bool	No	0	
skipImported	Bool	No	0	
modifiedOnly	Bool	No	0	
custIdType	List	No	None	
custIdValue	List	No	None	

ARCOS LLC
8800 Lyra Drive, Suite 200
Columbus, OH 43240
(614) 396 – 5500

getShiftsAssignments - GET /schedules/shifts

Returns a collection of ShiftAssignments that match the specified query.

Request Query Parameters

Parameter	Type	Required	Default	Description
startTime	Datetime	no	now	Records returned will have assignmentStart dates at or after this date.
endTime	Datetime	no	now + 7 days	Records returned will have assignmentStart dates up to this date.
changedSince	Datetime	no	none	Only return records that have been modified since this date.
contactId	Array of int	no	none	Only show records that are for the contactIds provided. You may provide this parameter multiple times for multiple contactIds.
locationId	Array of int	no	none	Only return records for Employees that are in the locationIds provided. locationIds will only be considered if contactId is not used. You may pass multiple locationIds by passing this parameter multiple times.

getShifts - GET /shifts

Returns a collection of ShiftAssignments that match the specified query.

Request Query Parameters

Parameter	Type	Required	Default	Description
getShiftDays	int			
getReferAsList	int			

getEmployeeSchedules - GET /schedules/employeeSchedules

Returns a collection of EmployeeSchedules that match the specified query. The EmployeeSchedule will contain some basic information about the employee as well as their ShiftAssignments and ScheduleRecords for the given startTime and endTime.

Request Query Parameters

Parameter	Type	Required	Default	Description
startTime	Datetime	no	now	Records returned will have started or ended at or after this date.
endTime	Datetime	no	now + 1 day	Records returned will have started or ended up to and including this time. The endTime - startTime range must be 15 days or less.
changedSince	Datetime	no	none	Only return records that have been modified since this date.
skipShiftAssignments	int	no	0	set to 1 to filter out ShiftAssignment records.
skipScheduledShifts	int	no	0	set to 1 to filter out ScheduledShift records.
skipExceptions	int	no	0	set to 1 to filter out ScheduleException records.
showInactive	int	no	none	Set to 1 to show deleted records in the results.
contactId	Array of int	no	none	Only show records that are for the contactIds provided.
locationId	Array of int	no	none	Only return records for Employees that are in the locationIds provided. locationIds will only be considered if contactId is not used.

Sync - GET /schedules/sync

Returns information for schedules with assignment dates between startTime and endTime and/or schedules that have changed since the changedSince date. Please note that this method returns dates in Arcos time (EST), in this format: 2021-08-27T18:36:57-04:00. This method returns OutputStream, which contains XML or JSON string - depending on the 'Accept' header value set in the request.

Request Query Parameters

Parameter	Type	Required	Valid Values	Default	Description
startTime	Datetime	Y	ISO-8601 Datetime	null	An item must exist (not be deleted) AFTER this date. If changedSince is present, and startTime is not given, all records will be analyzed. In the absence of changedSince,

					both startTime and endTime must be given, and should be no more than 365 days apart.
endTime	Datetime	Y	ISO-8601 Datetime	null	An item must have been created BEFORE this date.If changedSince is present, and endTime is not given, all records will be analyzed. In the absence of changedSince, both startTime and endTime must be given, and should be no more than 365 days apart.
changedSince	Datetime	N	ISO-8601 Datetime	1970-01-01	Query will return only records which have either been added or changed since this date - but only if the record satisfies the startTime and endTime parameters. If the changedSince is provided, but the time range between startTime and endTime exceeds 365 days, OR one of both of those dates are missing, then the changedSince cannot go back more than 90 days.
userChangesOnly	Bool	N	0, 1	0	If set to 1, results will NOT include records changed by Loader processes.
action	String	N	See Description	employeeSchedules	Describes the format of response. The default value is employeeSchedules, which results in the response similar to schedules/employeeSchedules call, which schedule records wrapped into employeeSchedule element. Another valid option is schedulesOnly. That will return schedule elements only, without employee information.
skipShiftAssignments	Bool	N	0, 1	0	If set to 1, Shift Assignments will not be included.
skipScheduledShifts	Bool	N	0, 1	0	If set to 1, Scheduled Shifts records will NOT be included.
skipExceptions	Bool	N	0, 1	0	If set to 1, Schedule Exceptions records will NOT be included.

Response Payload

A collection of employeeSchedules that match the query parameters.

Example


```
<employeeSchedules>
   <employeeSchedule contactId="288" vruId="6143965108" webId="chapman_ai" empId="Not Found" name="C
hapman, Mitch" locationId="9999">
      <shiftAssignments></shiftAssignments>
      <scheduledShifts>
         <scheduledShift memexId="65792" contactId="288" treatAs="2021-10-19T00:00:00-
04:00" isHoliday="false" isWorkingHoliday="false" modified="false" eventId="1008" startTime="2021-10-
19T07:00:00-04:00" endTime="2021-10-19T14:00:00-
04:00" isActive="true" locationId="9999" vehicleId="0" restFlag="0" addDate="2020-09-27T00:48:03-
04:00" changeDate="2020-09-27T00:48:03-04:00"></scheduledShift>
      </scheduledShifts>
      <scheduleExceptions></scheduleExceptions>
   </employeeSchedule>
</employeeSchedules>
```

getScheduleRecord - GET /schedules/{memexId}

Returns the ScheduleRecord that matches the provided {memexId}.

Request Query Parameters

Parameter	Type	Required	Default	Description
tz	String	no	Location Time Zone	Datetime attributes in any element returned will be skewed to the supplied Time Zone. (Such as: “US/Eastern”)

Response Payload

The scheduledShift with matching {memexId}.

Example

```
<scheduledShift treatAs="2021-04-21T00:00:00-04:00" isHoliday="false"
isWorkingHoliday="false" modified="false" memexId="46646" contactId="1260" eventId="1008"
startTime="2021-04-21T10:00:00-04:00" endTime="2021-04-21T19:00:00-04:00" isActive="false"
locationId="0" vehicleId="0" restFlag="0" addDate="2020-05-13T11:54:57-04:00"
changeDate="2021-04-21T12:02:45-04:00"/>
```

createRecords - POST /schedules/records

Create new ScheduledShifts or ScheduleExceptions. The ‘memexId’ attribute should be set to zero when creating new records.

Request Payload

A collection of ScheduleRecord objects consisting of either ScheduledShifts or ScheduleExceptions.

Example

```
<scheduleRecords>
   <scheduledShift contactId="4353" endTime="2021-08-02T17:00:00-
04:00" eventId="1008" memexId="0" startTime="2021-08-02T09:00:00-04:00" />
</scheduleRecords>
```

Response Payload

A collection of OpStatus elements for each ScheduleRecord provided. The ‘id’ field of each OpStatus will be set to the memexId field of the newly created record if successful or to zero if a failure occurred.

Example

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<opStatuses>
   <opStatus id="77537" code="1" message="Schedule has been saved for contact 4353"/>
</opStatuses>
```

createShift - POST /shifts

Create new shiftAssignment patten templates.

Request Payload

A Shift objects consisting of shiftDays and optionally referAs.

Example

```
<shift name="tl_17">
   <shiftDays>
      <shiftDay dayOfWeek="Sunday" startTime="1400" endTime="0230" startsOnPreviousDay="true"
endsOnNextDay="false"/>
      <shiftDay dayOfWeek="Tuesday" startTime="0100" endTime="0530" startsOnPreviousDay="false"
endsOnNextDay="false"/>
      <shiftDay dayOfWeek="Wednesday" startTime="1900" endTime="0530" startsOnPreviousDay="true"
endsOnNextDay="false"/>
      <shiftDay dayOfWeek="Thursday" startTime="0100" endTime="0530" startsOnPreviousDay="false"
endsOnNextDay="false"/>
      <shiftDay dayOfWeek="Friday" startTime="2300" endTime="0530" startsOnPreviousDay="true"
endsOnNextDay="false"/>
   </shiftDays>
   <referAsList>
      <referAs referAs="TL-17_1" locationId="9999"/>
      <referAs referAs="TL-17_2" locationId="9999"/>
      <referAs referAs="TL-17_3" locationId="4177"/>
   </referAsList>
</shift>
```

Response Payload

A collection of OpStatus elements for each shift provided. The ‘id’ field of each OpStatus will be set to the shiftId field of the newly created record if successful or to zero if a failure occurred.

updateRecord - PUT /schedules/records/{memexId}

This method can be used to update the ScheduleRecord provided. Only ‘startTime’, ‘endTime’, ‘comment’, ‘vehicleId’, ‘isWorkingHoliday’, and ‘releaseFlag’ may be changed.

Request Payload

The ScheduledShift or ScheduleException that should be modified.

Example

```
<scheduledShift treatAs="2021-02-15T00:00:00-05:00" isHoliday="false" isWorkingHoliday="false"
modified="false" memexId="214247" contactId="1151" eventId="1008" startTime="2021-02-15T00:00:00-06:00"
endTime="2021-02-15T08:00:00-06:00" isActive="true" locationId="0" vehicleId="0" restFlag="0" addDate="2021-
02-25T09:35:25-05:00" changeDate="2021-02-25T09:35:25-05:00"/>
```

Response Payload

An OpStatus element representing the result of the operation.

Example

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<opStatus id="0" code="404" message="Schedule Record was not found for memexId 214247"/>
```

updateRecords - PUT /schedules/records

This method can be used to update the ScheduleRecords provided. Only ‘startTime’, ‘endTime’, ‘comment’, ‘vehicleId’, ‘isWorkingHoliday’, and ‘releaseFlag’ may be changed.

Request Payload

The ScheduledShifts or ScheduleExceptions that should be modified.

Response Payload

An OpStatus element representing the result of the operation.

deleteRecord - DELETE /schedules/records/{memexId}

Sets the delete flag on the ScheduleRecord specified by memexId. Future searches for ScheduleRecords will only return this record if the showInactive flag is set to one (1). Searches for ScheduleRecords by specific memexId, however, will return records regardless of the showInactive setting.

Response Payload

An OpStatus object representing the result of deleting the ScheduleRecord.

createShifts - POST /schedules/shifts

Create new ShiftAssignments. Employees will have at most 1 ShiftAssignment per week. Attempting to create multiple ShiftAssignments for the same week will result in the last ShiftAssignment being saved and the others being overwritten.
Note, the createShifts method is also used to Delete Shifts. ShiftAssignments are removed by setting the 'shiftId' to zero(0) for the desired assignment date.

Request Payload

A collection of ShiftAssignment objects in either XML or JSON data structures.

Response Payload

A collection of OpStatus elements for each ShiftAssignment provided. The 'id' field of each OpStatus will be set to the memexId field of the newly created ShiftAssignment if successful or to zero if a failure occurred.

getScheduleRequests - GET /schedules/scheduleRequests

This method returns a collection of Schedule Request records matching the provided search query.

Request Query Parameters

Parameter	Type	Required	Default	Description
supId	Int	No	None	
deleted_flag		No	None	
search_date	Datetime	No	None	
schedRequestIds	Int	No	None	
contactIds	Int	No	None	

getScheduleRequest - GET /schedules/scheduleRequest/{schedRequestId}

This method returns the Schedule Request records matching the provided {schedRequestId}.

ARCOS LLC
8800 Lyra Drive, Suite 200
Columbus, OH 43240
(614) 396 – 5500

Crew Manager Data

The Crew Manager Endpoint can be used to manage the data that is normally accessed using the Arcos Crew Manager Application. Users can create new instances of Crews, Resources, and Lodgings. The API also supports changing Attribute values and assigning Members to Crews.
All Crew Manager operations require a Point In Time (PIT) value to orient the operation within Crew Manager. If no PIT value is provided by the user, the current server time will be used.
Crew Manager data structures can be found in the Appendix – XSD section.

Crew Manager Background

- An electric industry enterprise software application designed for system-wide resource management
- Designed and field tested by leading-edge U.S. electric utilities
- Integrated with ARCOS Callout & Scheduling Suite and ARCOS Workbench
- SaaS solution 100% managed and delivered by ARCOS

Crew Manager Features

- Centralized anytime, anywhere access
- Distributed real-time updating
- Touch screen capable and PC interface
- Independent SaaS delivery infrastructure
- Enterprise scalable, configurable platform
- Real-time dashboard and ad-hoc reporting
- Event recording and playback

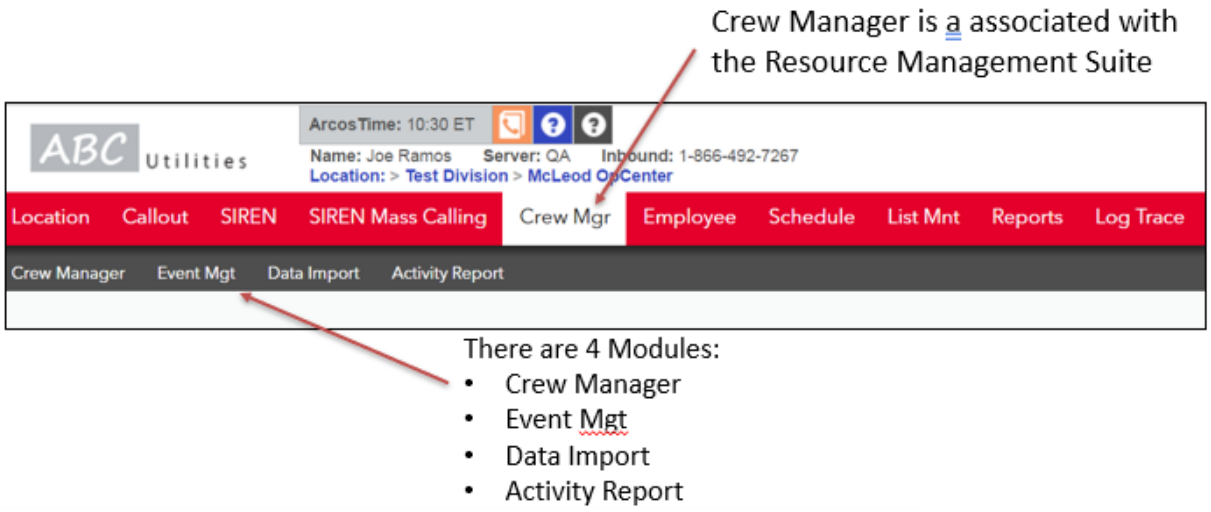
Crew Manager & Callout

- User access is dependent on their access level in Callout
- Employee Data shown in the “Crew Panel” is taken from Callout
- Integrations basic employee data between both products (Name, Phone number(s), Primary Classification, Schedules and Schedule Exceptions)
- If configured, completed callouts will show the accepted employees in Crew Manager
- Like Callout, Crew Manager is accessed via the Web UI
- Web Based, Android, and iOS clients available

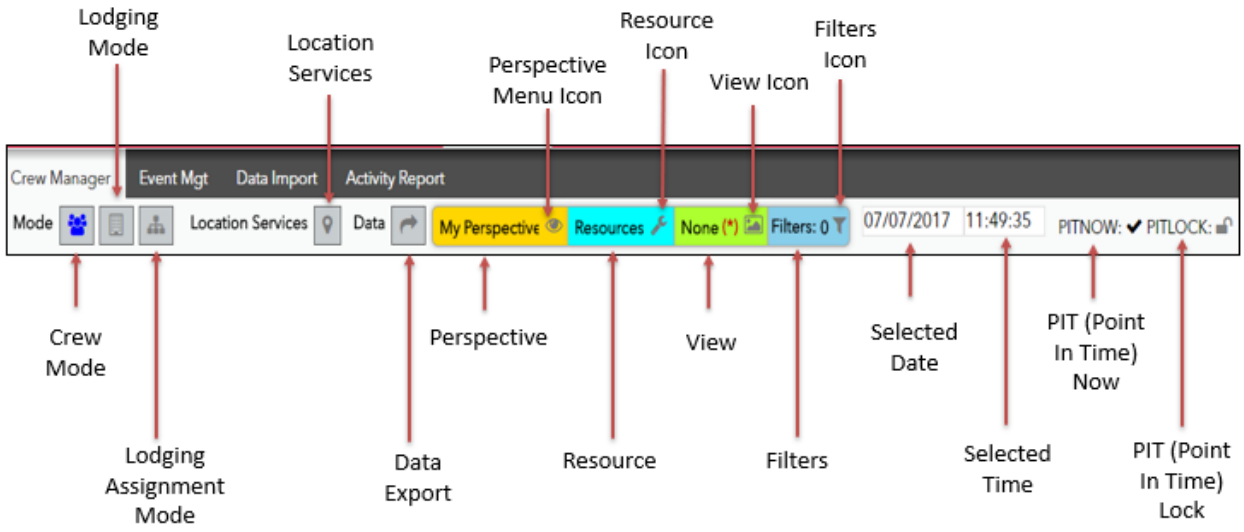
ARCOS LLC
8800 Lyra Drive, Suite 200
Columbus, OH 43240
(614) 396 – 5500

Crew Manager Overview

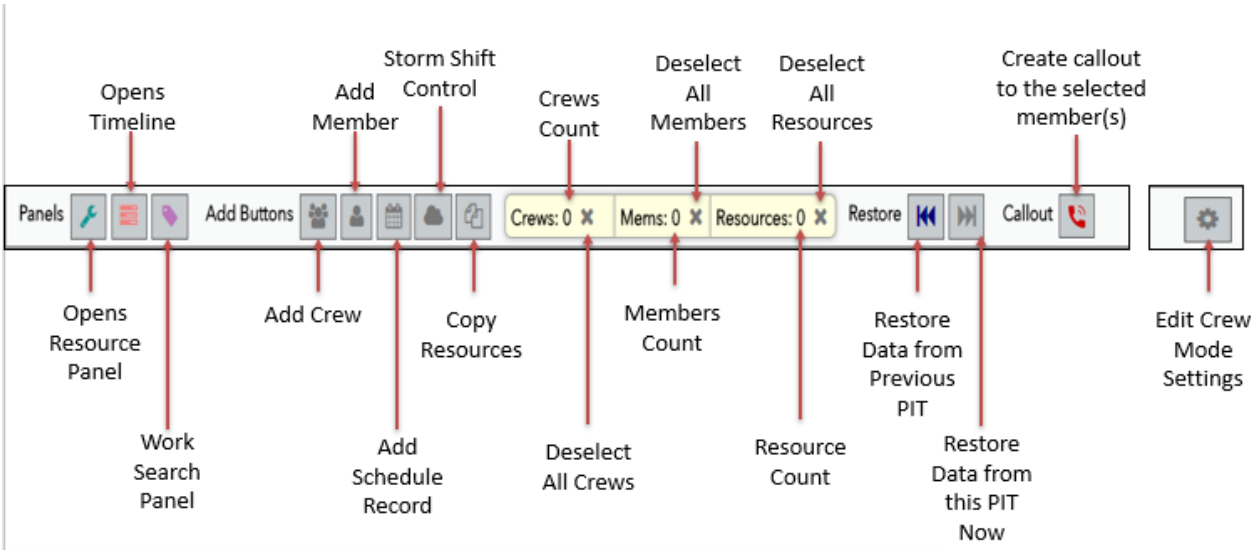
Crew Manager Panel



Crew Manager Panel – Mode Toolbar



Crew Manager Panel – Crew Mode Toolbar

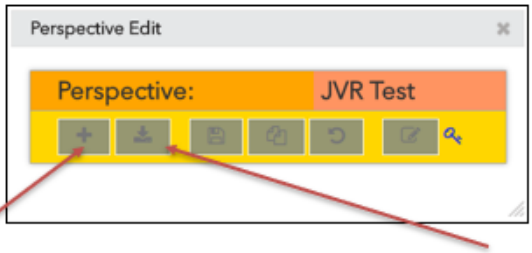


1.7 Perspectives

- Perspective is defined as - *A selection of data you want to view*

Click on the "Eye" to activate Perspective

Click "Perspective Menu" icon

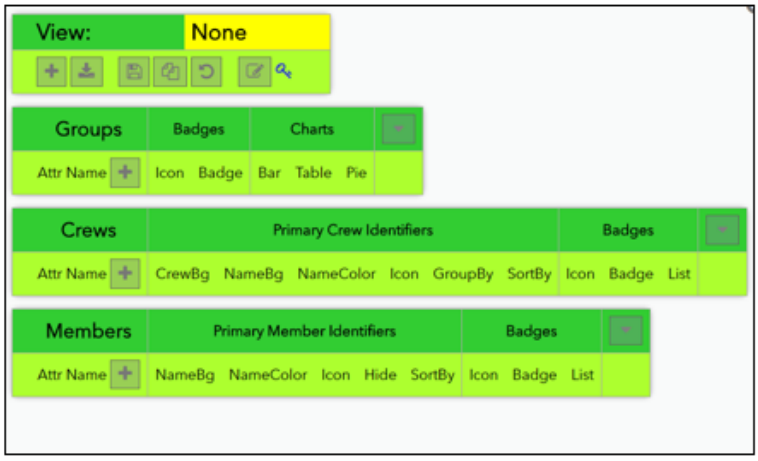


Click on the plus to create a new Perspective

Click on the down arrow to load an existing Perspective

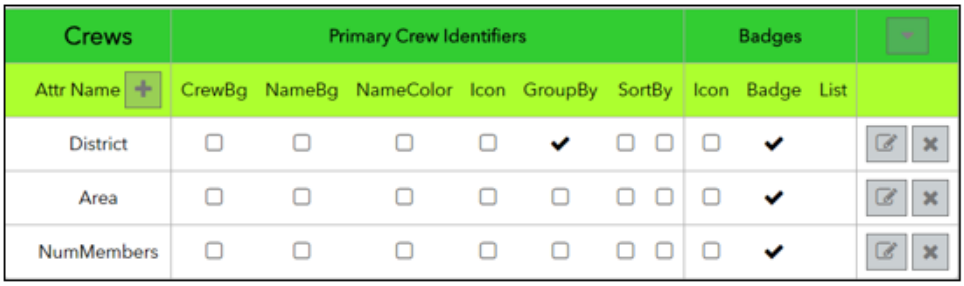
Views

- After determining your Perspective a "**View**" window will appear
- 3 panels will be available to work from
  - **Groups**
  - **Crews**
  - **Members**
- User can apply the appropriated "**Attributes**" for each panel

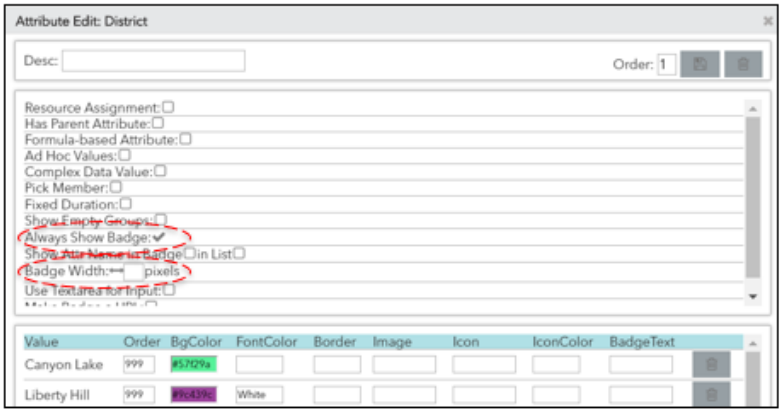


View – Crews

- Crews attributes allows the users to view data associated at the crew level
  - District
  - Working Area
  - Number of Members
  - ETC

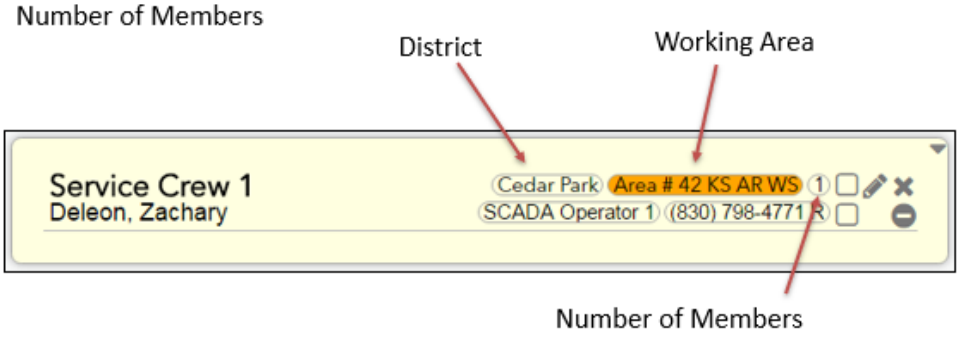


- Editing Attributes
  - Check "Always Show Badge"
  - Change Badge Width to Pixels by click on the "T"
  - Add Values
  - Note: when adding values to an attribute "None" and "Default" should always be an option



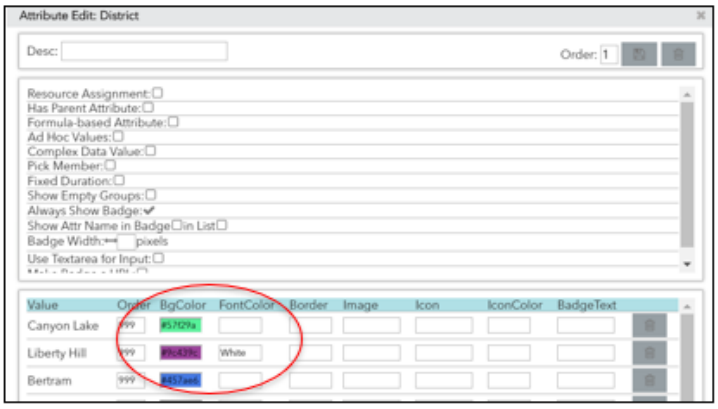
In this example the user has selected the following attributes "**Crew Attributes**" to view at the crew level:

- District
- Working Area
- Number of Members



Badge/Font colors can be added

- View – Edit Attribute
- In Value "BgColor" and "Font Color"
- Color name can by typed
- Double click within the box then pick the color



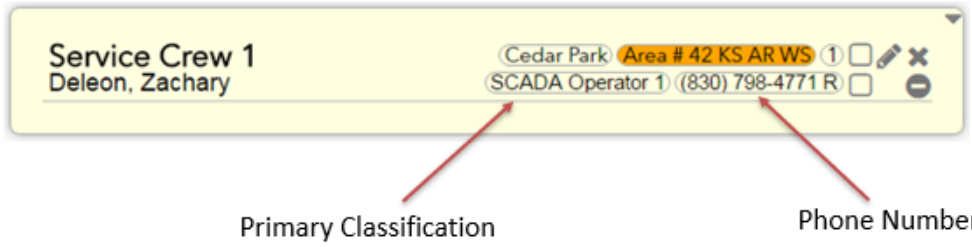
View – Members

- Member attributes allows the users to view data associated at the crew member level
  - Classification
  - Phone Number
  - Cumulative Hour Worked
  - ETC

Members	Primary Member Identifiers					Badges			
Attr Name +	NameBg	NameColor	Icon	Hide	SortBy	Icon	Badge	List	
PrimaryClass	☐	☐	☐	☐	☐	☐	✔		✎ ✖
Phone1	☐	☐	☐	☐	☐	☐	✔		✎ ✖
CumHrsWorked	☐	☐	☐	☐	☐	☐	✔		✎ ✖

In this example the user has selected the following "**Member Attributes**" to view at the crew member level:

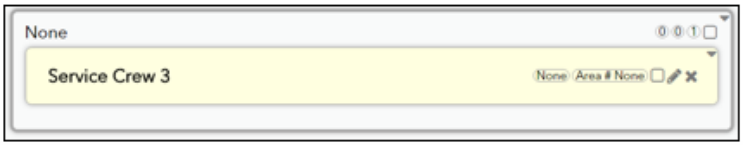
- Classification
- Phone Number



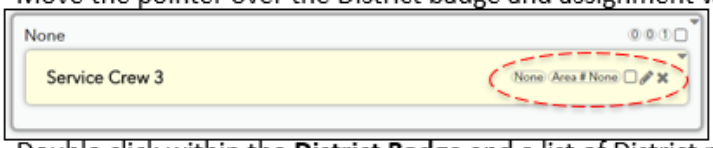
Crew Panel

- When you have completed the Views for Crews and Members, within the "**Crew Panel**" click on the "**Add Crew**" button 
- Enter the name of the crew and click on the Plus to same

- Your crew will now show in the Panel



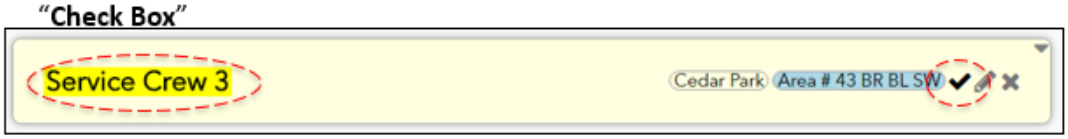
- Assigning a Crew to a District and an Area #:
  - Move the pointer over the District badge and assignment will show "**None**"



  - Double click within the **District Badge** and a list of District names will appear.
  - Do the same for "**Area #**"

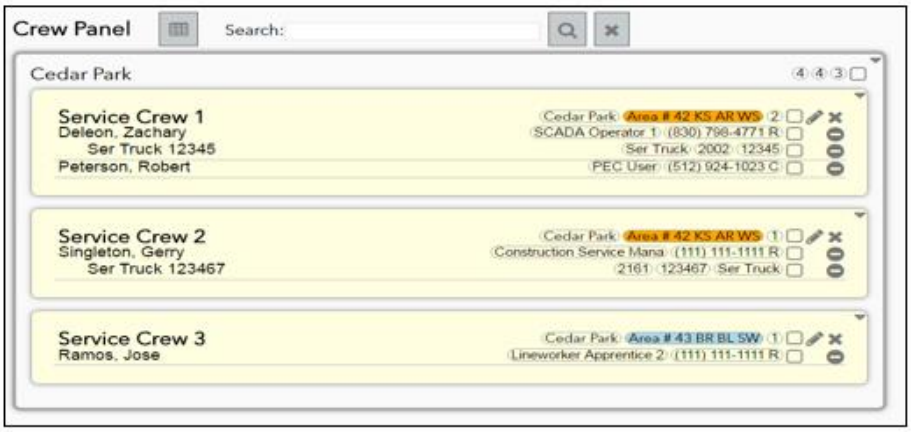
ARCOS LLC
8800 Lyra Drive, Suite 200
Columbus, OH 43240
(614) 396 – 5500

- Assigning a Crew Member(s)
  - Highlight the Crew which the member will be assigned to by clicking on the "**Check Box**"



  - To assign a Crew Member click on the "**Add Member**" button and type the members last name in "Search Name" and click on the "**Search Icon**"
  - Click "**Yes**" in the "**Add Member**?"

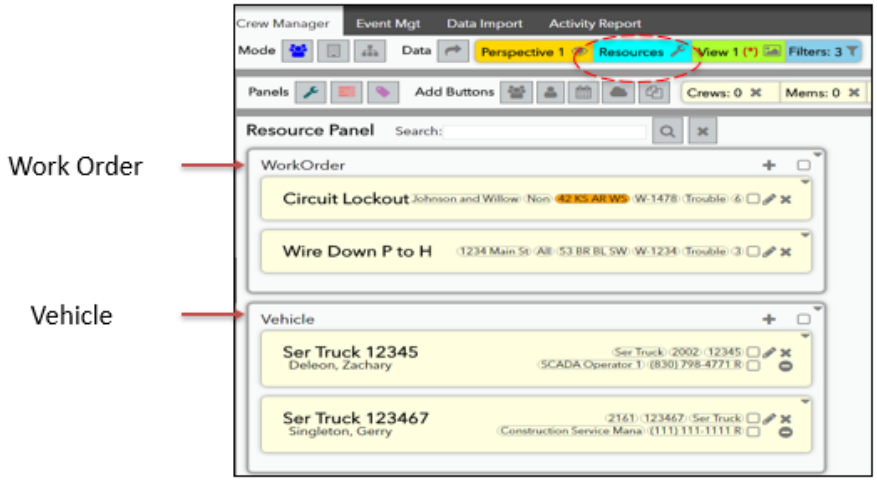
- Service Crew 3 has been add to the Cedar Park District, along with Service Crews 1 & 2



ARCOS LLC
8800 Lyra Drive, Suite 200
Columbus, OH 43240
(614) 396 – 5500

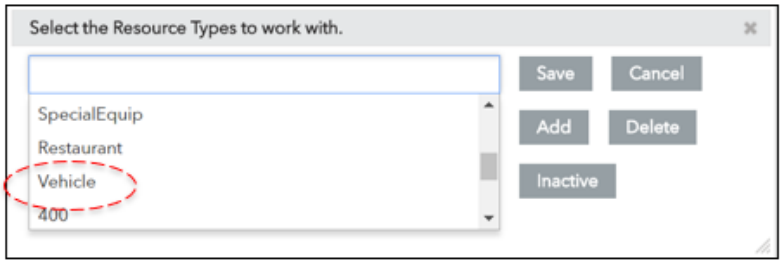
Resource Panel

- Resource Panel allows users to input, assign and track **Work Orders, Vehicles, ETC**



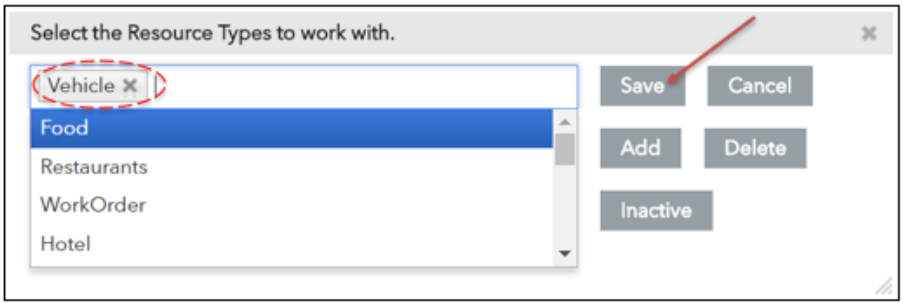
Resource Panel - Vehicles

- Tracking vehicles in Crew Manger
  - To open the **Resource Panel** click on the  icon
  - Click on the "**Resources**" word 
  - This will open "**Select the Resources Types to work with**"

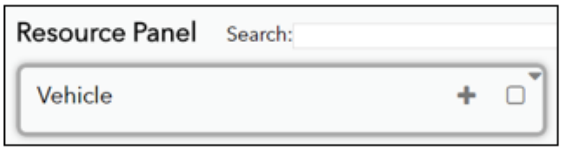


  - Scroll to find **Vehicle**

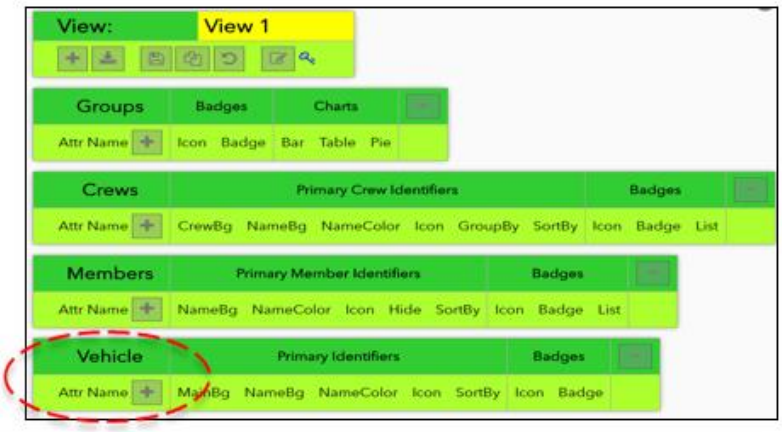
- Click on Vehicle to select. Then click on the Save button



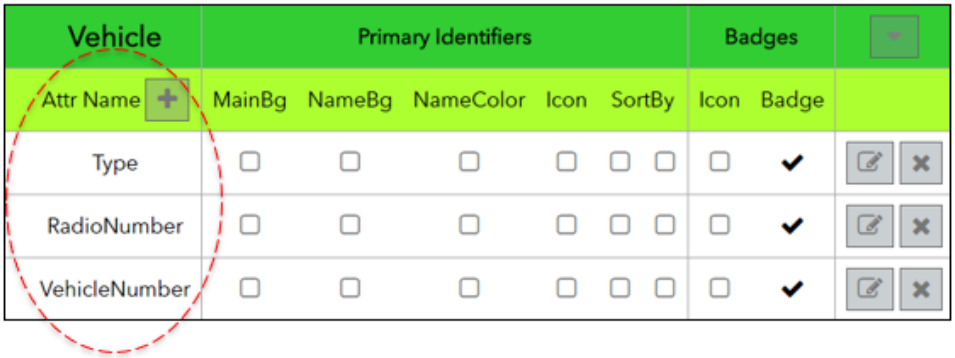
- Vehicle will now display in the Resource Panel



- After you have successfully perform this function, "**Vehicle**" will be add in the "**View**" area



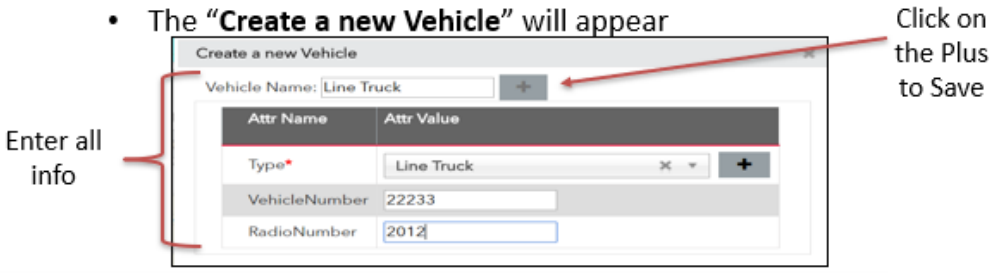
- Much like Groups, Crews, and Members, the next step is to apply the appropriate Attributes for Vehicle



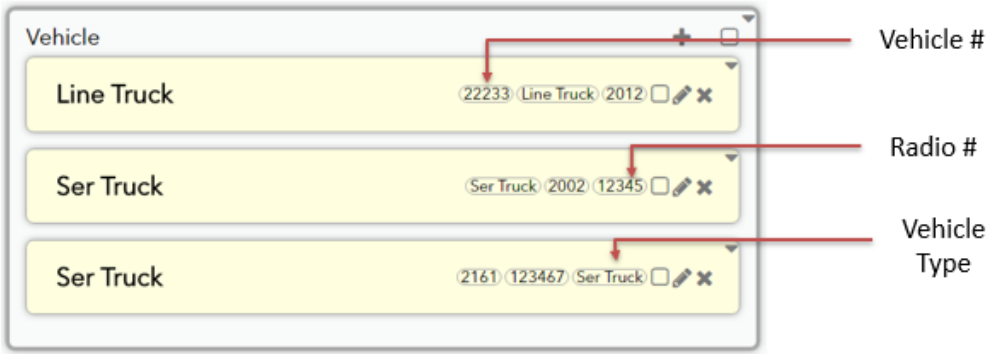
ARCOS LLC
8800 Lyra Drive, Suite 200
Columbus, OH 43240
(614) 396 – 5500

- To add a Vehicle to the **Resource Panel** click on the plus sign

- The "**Create a new Vehicle**" will appear



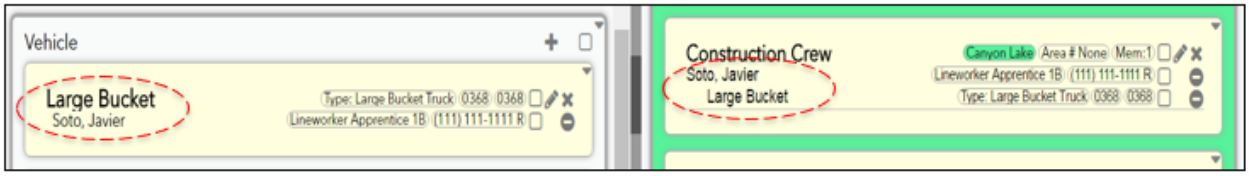
- The Vehicle panel will be populated with the necessary information
- Moving the mouse over the badges will highlight the appropriate info, i.e. type, radio #, vehicle #



ARCOS LLC
8800 Lyra Drive, Suite 200
Columbus, OH 43240
(614) 396 – 5500

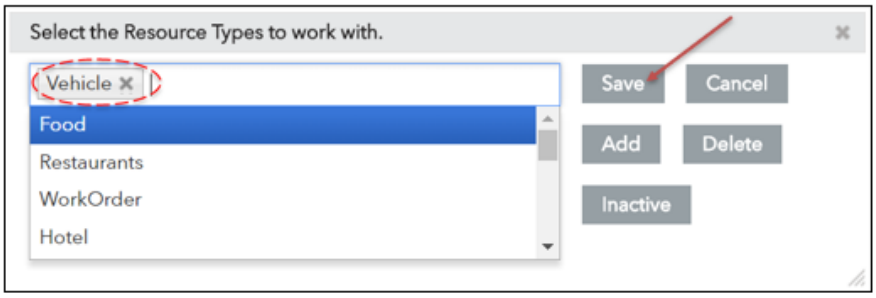
- Assigning a vehicle to a Crew
  - In the "**Resource Panel"**, click and drag the vehicle you want to assign on to the Crew
  - Vehicle assignments can be shown in both the "**Crew Panel**" &"**Resource Panel**"

Resource Panel **Crew Panel**



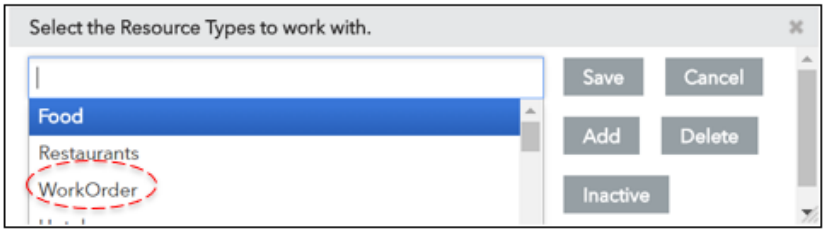
Resource Panel – Work Orders

- Click on work orders to select. Then click on the Save button



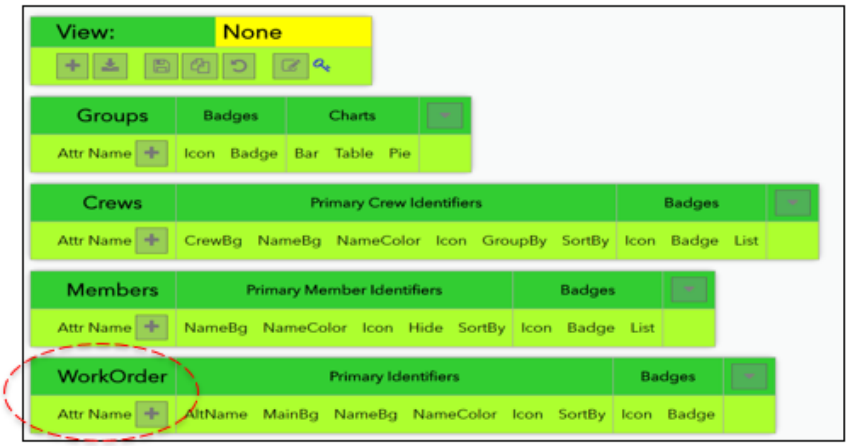
- Work Order will now display in the "**Resource Panel**"

- Tracking work orders in Crew Manger
  - To open the **Resource Panel** click on the  icon
  - Click on the "**Resources**" word Resources
  - This will open "**Select the Resources Types to work with**"

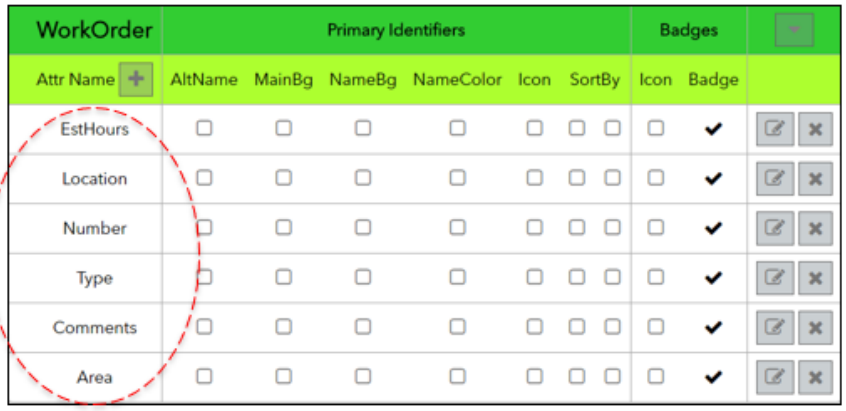


  - Scroll to find **Work Orders**

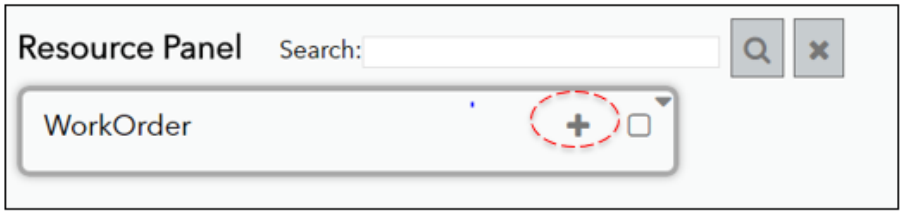
- After you have successfully perform this function, "**Work Order**" will be add in the "**View**" area



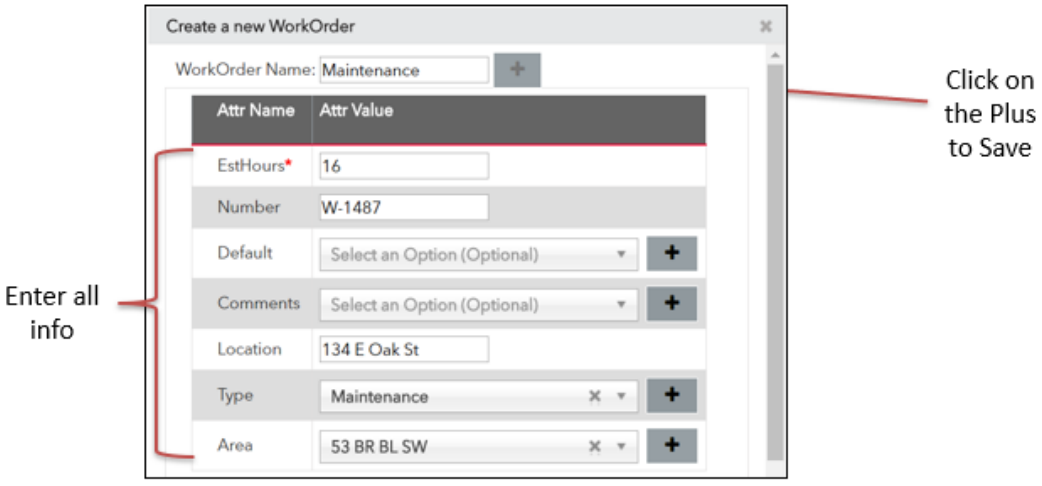
- Much like Groups, Crews, and Members, the next step is to apply the appropriate Attributes for Work Orders



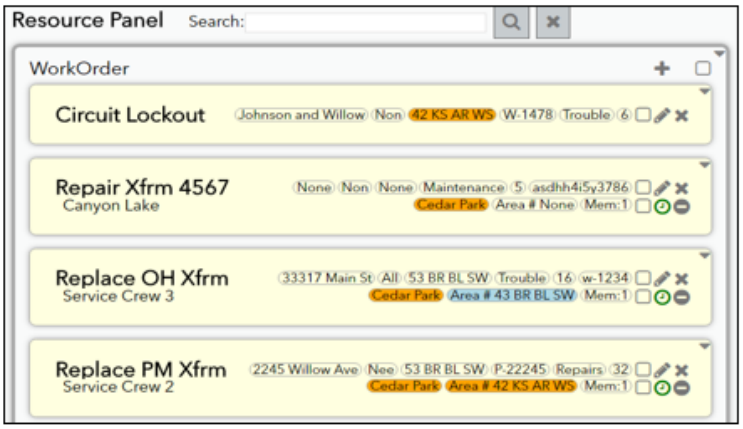
- To add a Work Order to the "**Resource Panel**" click on the plus sign



- To the "**Create a new Work Order**" will appear



- The Work Order panel will be populated with the necessary information
- Moving the mouse over the badges will highlight the appropriate info

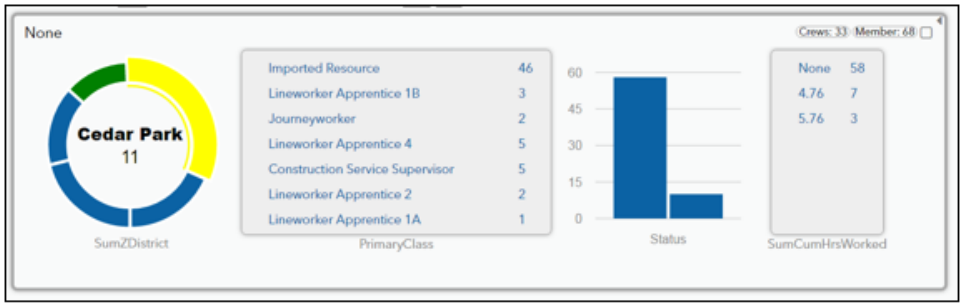


- Assigning a Work Order to a Crew
  - In the "**Resource Panel**", click and drag the work order you want to assign on to the Crew
  - Work Order assignments can be shown in both the "**Crew Panel**" &"**Resource Panel**"

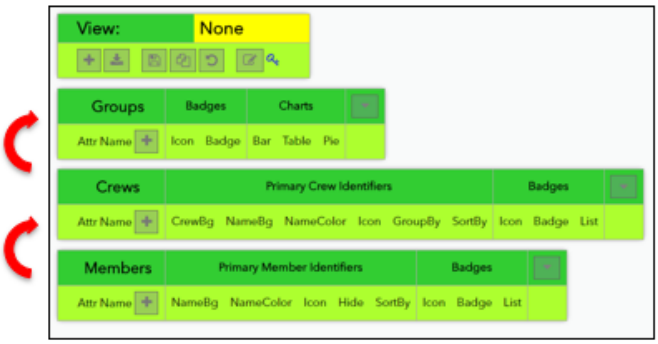
Resource Panel **Crew Panel**

Dashboard Reporting

- Dashboard Report provides the user matrix data from the Crews, Members levels



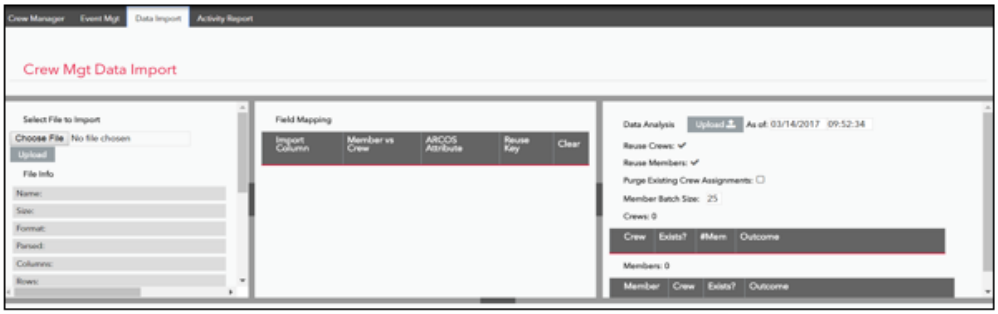
- Dashboard Report are displayed in the Groups level
- Data viewed is rolled up from either the Members or Crews levels
- Data is activated by applying formulas to the attributes you want to report on



- Dashboard data uses formulas and or complex data values
- Formulas can be found by clicking on the “Formula Help icon”

Data Import

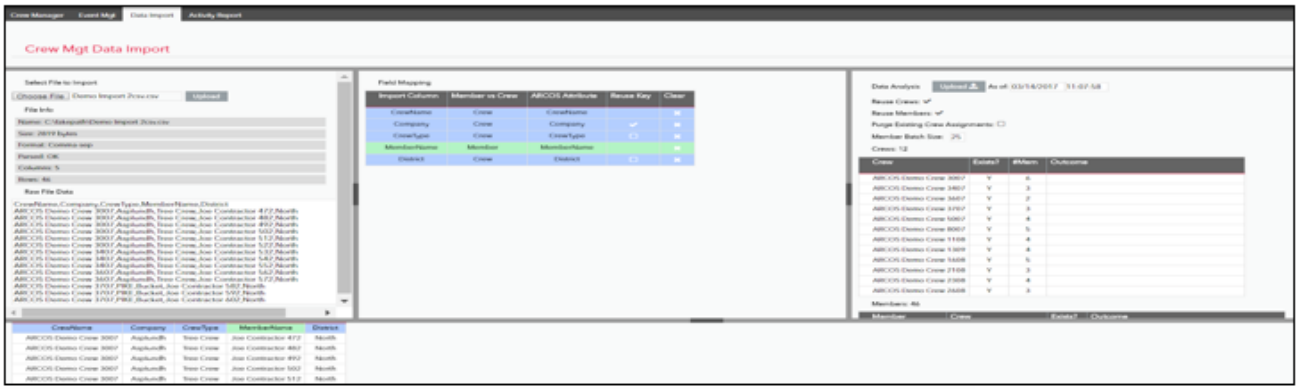
- User can load Contractor Crews and Mutual Assistance Crews by using Data Import
- Once this data is entered into Crew Manager, user can manage crews and resources just like their own crews



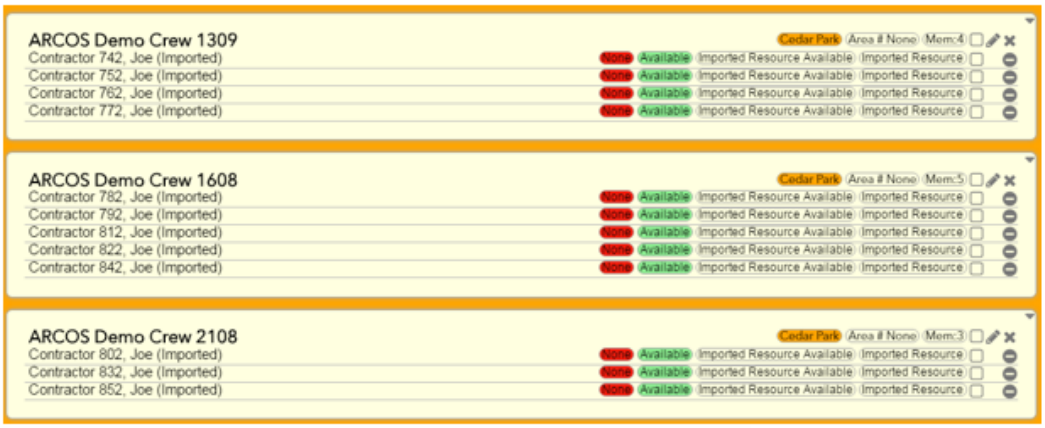
- Data is submitted by the Contract or Mutual Assistance Utility via CSV file
- Key information is required to properly upload the file

	A	B	C	D	E	F	G
1	CrewName	Company	CrewType	Team Lead	MemberName	District	PriClass
2	ARCOS Demo Crew 3007	Asplundh	Tree Crew	MacNeil	Joe Contractor 472	North	Trimmer
3	ARCOS Demo Crew 3007	Asplundh	Tree Crew	MacNeil	Joe Contractor 482	North	Trimmer
4	ARCOS Demo Crew 3007	Asplundh	Tree Crew	MacNeil	Joe Contractor 492	North	Trimmer
5	ARCOS Demo Crew 3007	Asplundh	Tree Crew	MacNeil	Joe Contractor 502	North	Trimmer
6	ARCOS Demo Crew 3007	Asplundh	Tree Crew	MacNeil	Joe Contractor 512	North	Trimmer
7	ARCOS Demo Crew 3007	Asplundh	Tree Crew	MacNeil	Joe Contractor 522	North	Trimmer
8	ARCOS Demo Crew 3407	Asplundh	Tree Crew	MacNeil	Joe Contractor 532	North	Trimmer

1. Choose File
2. Verify Field Mapping
3. Upload



- After a successful upload Contractor or Mutual Assistance Crews will display in the Crew Panel at the assigned work locations



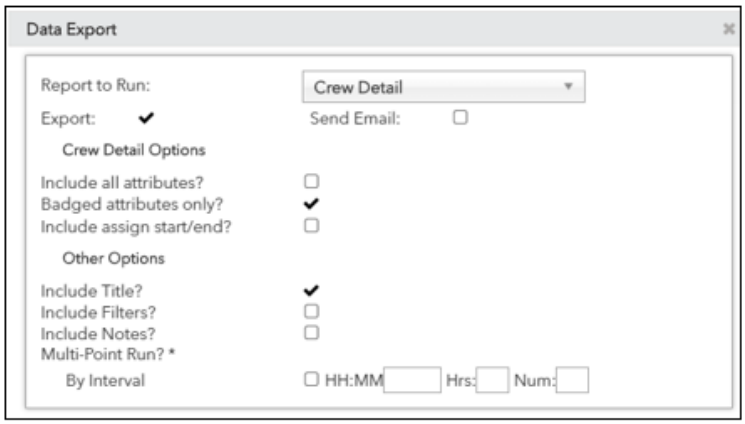
Data Export

ARCOS LLC
8800 Lyra Drive, Suite 200
Columbus, OH 43240
(614) 396 – 5500

- Data Export function allows user to build key reports with in Crew Manager
  - Crew Detail
  - Crew Summary
  - Daily Crew
  - Single Work Day
  - Matrix
  - Audit
  - Crew Detail w/ Resource
  - Resource Detail

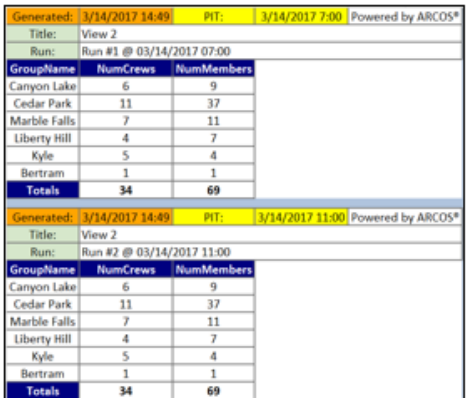
Note: report created in Data Export can not be batched

- The Data Export filter allows the user to select which report they would like to run



- After clicking on Run, report are exported into an Excel spreadsheet
- Multi-Point Run provides a user the option to allow all Crew Manager reports to be run for several periods

Generated:	3/14/2017 14:29	PIT:	3/14/2017 10:37	Powered by ARCOS®
Title:	View 2			
GroupName	**NumCrews**	**NumMembers**		
Canyon Lake	6	9		
Cedar Park	11	37		
Marble Falls	7	11		
Liberty Hill	4	7		
Kyle	5	4		
Bertram	1	1		
Totals	**34**	**69**		



Activity Report

- The Crew Manager Activity Report shows a list of all ARCOS users who are in a security group that has access to Crew Manager
- Included is:
  - The number of times the user accessed Crew Manager
  - The number of changes they made and saved within Crew Manager

- The search criteria is similar to Callout Reports
- Details allow the user to drill down for additional information

- Example: selection of a specific user

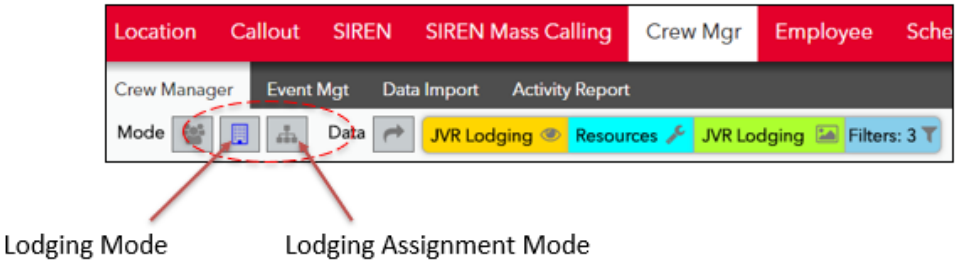
Session Start *sort*	Session End *sort*	Duration *sort*	# Actions *sort*	Total Actions *sort*	Detail *sort*
05/10/2017 10:30:09	05/10/2017 10:48:22	00:18:13	4		detail
Totals		00:18:13	4		

- Details of action taken

Date *sort*	Action *sort*	Type *sort*	Description *sort*	PIT *sort*
05/10/2017 10:32:59	Modify	Attr	members|PrimaryClass	None
05/10/2017 10:33:05	Modify	Attr	members|PrimaryClass	None
05/10/2017 10:33:18	Modify	Attr	members|PrimaryClass	None
05/10/2017 10:33:29	Modify	Attr	members|PrimaryClass	None

Lodging

- There are two key modes, Lodging and Lodging Assignments



Lodging Mode Lodging Assignment Mode

- Note – In larger utility, logistics i.e. lodging/meals are created and managed by the company's Storm Logistics Team

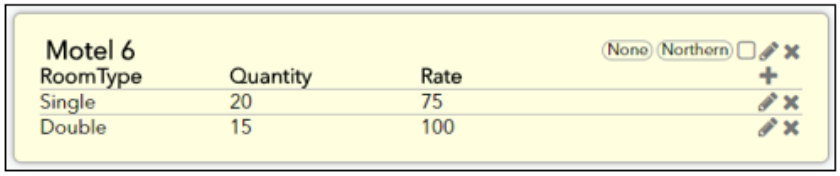
- In the Lodging Mode add the appropriate attributes for Lodgings and Room Types

- Establish Filters
- In the Add Button select Add Hotel

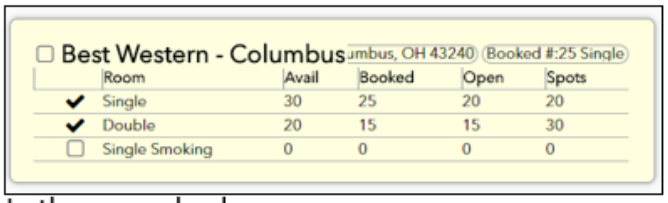
- Visible Days defaults, but can be changed based on the estimated duration of the storm

- In the Hotel Panel
  - Add the name of hotel

  - Next populate Room Type, Quantity, and Rate by click on the +

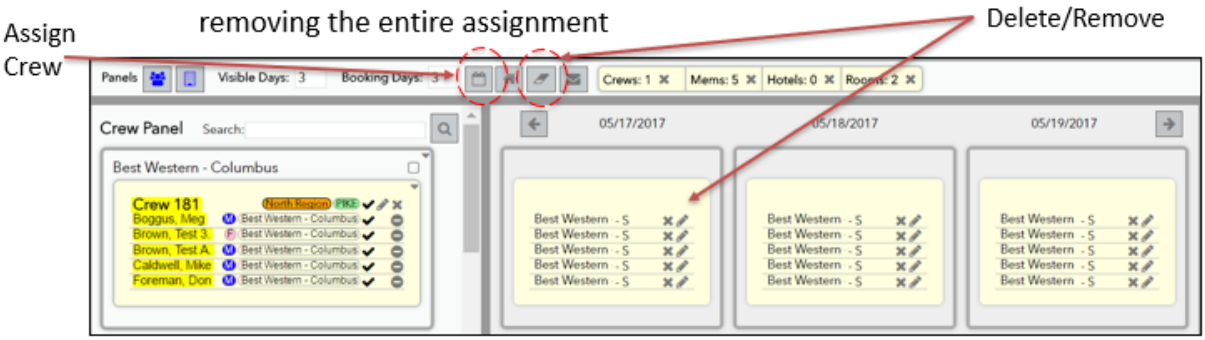


- Booking Rooms:
  - The hotel will the determine the number of specific room available



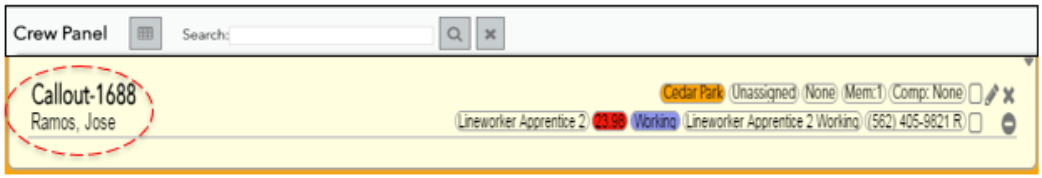
  - In the example above:
    - The utility booked rooms for 3 days
    - 30 available single rooms, 25 were booked
    - 20 double rooms, 15 were booked

- Booking Rooms:
  - Select the crew that will need rooms
  - Click on the Assign People to Selected Rooms
  - Users can either delete room assignments by employee or removing the entire assignment



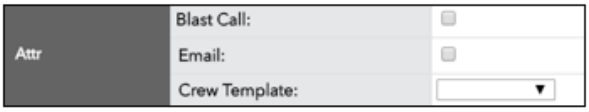
Crew Template

- The Crew Template feature allows Crew Manager users to:
  - View pending callouts
  - Employee(s) who accepted the callout

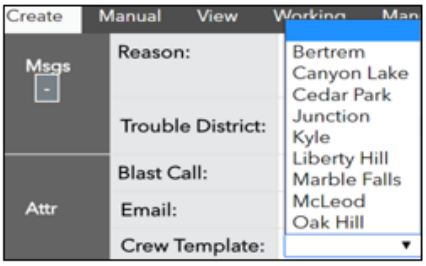


  - Detail instructions can be found via Online Docs> Appendix A> 2.16.3 (page 20)

- The Crew Template feature is found in Callout tab> Create> Attr



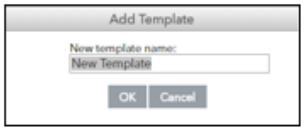
- Crew Template is populated with the districts names captured in Crew Manager and activated in the Sys Admin> Callout Crew Templates



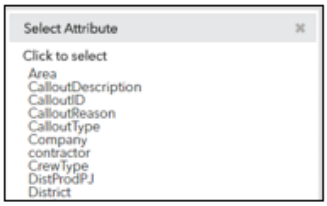
- To activate a district in Callout Crew Template

1. Click on the Add button 

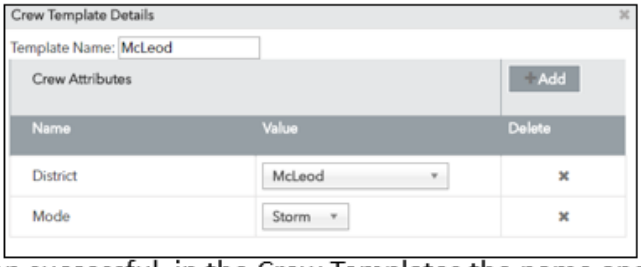
2. Enter in a name for the new template. Usually it's the district name



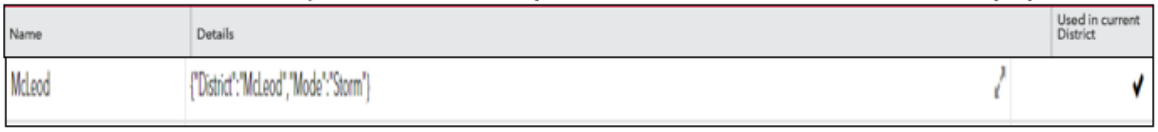
3. Within the Crew Template Details click on Add button 

4. Select the appropriate attribute 

5. For this example, attributes selected were District and Mode and the values were McLeod and Storm

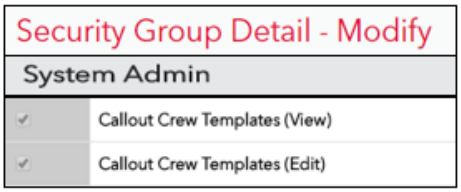


6. When successful, in the Crew Templates the name and details should populate



To enable the Callout Crew Template page

1. Click the SysAdmin tab.
2. Click the Security button.
3. Click the modify link for the security group you wish to modify.
4. Scroll down to the System Admin section.
5. Check the boxes for Callout Crew Template (View) and Callout Crew Template



Crew Manager Alerts

- ARCOS gives users the added ability to define custom email alerts for Crew Manager
  - Crew Manager Alerts can be used to alert administrators and supervisors of when an attribute value is changed
  - Employees can subscribe to alerts based on Perspective

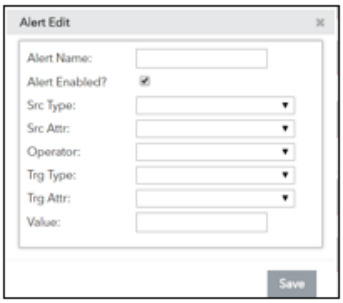
Src	Src Attr	Src Assign Attr	Operator	Trg	Trg Attr	Trg Assign Attr
members	StormRole		Any Change			

Employee	Perspective
Ramos, Joe	Any

To create a Crew Manager Alert

- Detail instructions can be found via Online Docs> Appendix A> 2.16.3 (page 28)

1. Click the Email /Alerts button on the SysAdmin tab.
2. Click the Crew Manager Alerts button.
3. Click the Add Alert button. The Alert Edit popup displays.
4. Type a name for the alert into the Alert Name textbox.



To create a Crew Manager Alert – Cont'd

5. Check the Alert Enabled checkbox to enable the alert. If this checkbox is unchecked, alerts will not be sent for the configured rule.
6. Select the type of attribute the alert is for from the Src (Source) Type dropdown. (The type I where the attribute resides, e.g. crew, member, lodging, resource type…)
7. Select the attribute to alert on from the Src Attr dropdown. The Src Attr dropdown will only display non-calculation and non-formula based attributes that are assignable to the selected type

ARCOS LLC
8800 Lyra Drive, Suite 200
Columbus, OH 43240
(614) 396 – 5500

To create a Crew Manager Alert – Cont'd

8. Select an operator from the Operator dropdown. If selecting "Any Change", skip to step 13.

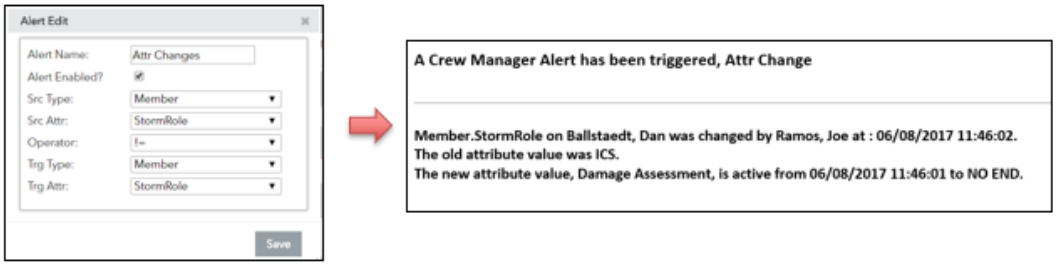
=	Equal is used to trigger an alert whenever the Src Attr value matches the value entered in the Value textbox or the attribute selected in the Trg fields.
!=	Does not equal is used to trigger an alert whenever the Src Attr value does not match the value entered in t he Value textbox or the attribute selected in the Trg fields.
<, <=, >, >=	Less than, less than or equal to, greater than and greater than or equal to are used to compare numeric values only. The Src Attr value can be compared to a value entered in the Value textbox or the attribute selected in the Trg fields.
Any Change	Any change is used to trigger an alert when any change is made to the Src Attr value. The Value textbox and the Trg fields should not be used with the "Any Change" operator

To create a Crew Manager Alert – Cont'd

9. Determine if the source attribute will be compared to a static value or a different attribute. Skip to step 11 if a different attribute.
10. Type the static value to compare the source attribute in the Value textbox. Skip to step 13.
11. Select the type of attribute the alert is for from the Trg Type dropdown. (The type is where the attribute resides, e.g. crew, member, lodging, resource type…) Note: The alert can only be saved if the Src Type and the Trg Type are the same type or if one of them is a lodging type

To create a Crew Manager Alert – Cont'd

12. Select the attribute to alert on from the Trg Attr dropdown. The Trg Attr dropdown will only display non-calculation and non-formula based attributes that are assignable to the selected type.
13. Click the Save button



ARCOS LLC
8800 Lyra Drive, Suite 200
Columbus, OH 43240
(614) 396 – 5500

ARCOS LLC
8800 Lyra Drive, Suite 200
Columbus, OH 43240
(614) 396 – 5500

Crew Manager Formulas

There are 28 formulas:

dataPoint()
or()
summarize()
count()
sum()
startOf()
endOf()
findMember()
memberAttr()
add()
diff()
multiply()
divide()
equals()
lessThan()
greaterThan()
lessThanOrEqual()
greaterThanOrEqual() concat()
contains()
assignCount()
assignSummarize()
assignSum()
resourceAttr()
resourceSummarize()
resourceSum()
lodgingAttr(), roomAttr(),
valueFor()

dataPoint(attrName.dataPoint)

The dataPoint() formula examines the same data class (eg: members, crews).
This allows you to break out an additional data point value into its own badge-able attribute.
The attrName needs to refer to an attribute definition with "addl data value points.

example: dataPoint (StormRole.Qualified)

or (attr1, attr2, attr3.dataPoint, ...)

The or() formula examines other attributes at the same data class.
The formula will check the value of each argument and return the first one with a value.
This is a way to compile multiple potential attributes into a single attribute.

example: or (PrimaryClass.WrenchTurner, StormRole.Qualified)

summarize(attrName) or summarize(attrName.dataPoint)

The summarize() formula looks at the data one level down and creates a summary of counts on the parent.
The resulting value is NOT usable as a badge on its own.
Rather, it is a pre-req for the count() function.

example: summarize (StormRole)
example: summarize (StormRole.Qualified)

count(summarizeAttr.value)

The count() formula looks at a "summarize" attribute at the same data level.
It returns the numeric count for the specified value.
In the examples, the "Summary" attribute would be a pre-req.

example: count (StormRoleSummary.Patroller)
example: count (StormRoleSummary.Lead Lineman)
example: count (QualifiedSummary.Yes)

sum(attrName) or sum(attrName.dataPoint)

The sum() formula works like summarize(), in that in looks at the data one level down and creates a value on the parent.
The attribute being summed needs to have a numeric value itself.

example: sum (StormRole.Qualified)
example: sum (NumQualified)

startOf(attrName) or startOf(attrName, date format)

The startOf() formula finds the start date/time (in relation to PIT) of the specified attribute at the member or crew level.
The date format argument is optional and a default format will be returned if none is supplied.

The "Status" keyword may be used in place of attrName and will return the start time of a member schedule record (PIT based).
"Status" keyword logic will only be triggered at the members level.
However, startOf() will still function at the crew level if a user defined attribute exists named "Status".

example: startOf (Supervisor)
example: startOf (Supervisor, m/d/yy HH:MM:ss)
example: startOf (Status, m/d HH:MM)

endOf(attrName) or endOf(attrName, date format)

The endOf() formula finds the end date/time (in relation to PIT) of the specified attribute at the member or crew level.
The date format argument is optional and a default format will be returned if none is supplied.

The "Status" keyword may be used in place of attrName and will return the end time of a member schedule record (PIT based).
"Status" keyword logic will only be triggered at the members level.
However, endOf() will still function at the crew level if a user defined attribute exists named "Status".

example: endOf (Supervisor)
example: endOf (Supervisor, m/d/yy HH:MM:ss)
example: endOf (Status, m/d HH:MM)

findMember(attr1.value, attr2.None, attr3, ...)

The findMember() formula finds one Member (based on sort order) within a crew matching the attribute criteria.
"None" is a supported value, and not supplying a value on an attribute will function as if "None" was supplied.

example: findMember (StormRole.Patroller, Supervisor.Yes)
example: findMember (Leader.Yes)

memberAttr(crewAttr.memberAttr)

example: memberAttr (Leader.Phone1)

The example formula returns the value of the Phone1 attribute for the member found by the findMember() formula in place on the Leader crew attribute.

add(attr1, attr2, attr3, ...)

The add() formula will sum the numeric values of two or more attributes at the same level (member, crew, group).
Also supports literal numeric values as 2nd+ argument.

example: add(NumWorking, NumException)

ARCOS LLC
8800 Lyra Drive, Suite 200
Columbus, OH 43240
(614) 396 – 5500

diff(attr1, attr2)

The diff() (aka subtract) formula will find the difference between two numeric attributes at the same level (member, crew, group).
The 2nd argument can be a literal numeric value.

example: diff(NumMembers, NumException)

multiply(attr1, attr2, attr3, ...)

The multiply() formula will multiply the numeric values of two or more attributes at the same level (member, crew, group).
Results are always rounded to two decimal places.
Also supports literal numeric values as 2nd+ argument.

example: multiply(NumMembers, 2)
example: multiply(NumTrucks, DailyFuel)

divide(attr1, attr2, round)

The divide() formula will divide two numeric attributes at the same level (member, crew, group).
The 2nd argument can be a literal numeric value.
The 3rd argument specifies how many decimal places to round to -- this is optional and defaults to 2. Valid values are from 0 to 10.

example: divide(NumMembers, NumVehicles)
example: divide(NumTickets, 8)

equals(attr1, attr2, ...)

The equals() formula will indicate if the values of two or more attributes are equal at the same level (member, crew, group).
Supports numeric and non-numeric attribute values.
Also supports literal numeric values as 2nd+ argument.

example: equals(HomeArea, WorkingArea)
example: equals(NumWorking, 0)

lessThan(attr1, attr2)
lessThanOrEqual(attr1, attr2)
greaterThan(attr1, attr2)
greaterThanOrEqual(attr1, attr2)

These four formulas all require two arguments, and will return Y or N when evaluated.
The 2nd argument can be a literal numeric value.

example: lessThan(RoomsAvailable, RoomsNeeded)
example: greaterThanOrEqual(NumCrews, 1)

concat(attr1, attr2, ... , separator)

The concat() formula can be used to combine two attributes into a single badge, to save on real-estate.
The separator is optional, and MUST be the last argument AND a single character in order to be recognized as a separator.

example: concat (Email, VehPhone, /)

contains (attr, pattern1, pattern2, ...)

The contains() formula will return "Y" or "N" based on finding a string/pattern in the specified attribute at the same level.

example: contains (PrimaryClass, Leader, Foreman, Chief)

assignCount(targetClass.assignAttr)

The assignCount() formula can only be used on Resource types (eg Vehicle, Restaurant, WorkOrder).
It is used to monitor how many crews or members are currently assigned to each resource instance.
The 2nd-part of the argument must refer to a Crew or Member level attribute which itself IS a resource assignment attribute.

example: assignCount (crews.BreakfastAssign)
example: assignCount (members.HotelAssign)

assignSummarize(targetClass.assignAttr.attrName)

The assignSummarize() formula can only be used on Resource types (eg Vehicle, Restaurant, WorkOrder).
It is used to summarize data using the crews or members who are currently assigned to each resource instance.
The 2nd-part of the argument must refer to a Crew or Member level attribute which itself IS a resource assignment attribute.
The 3rd-part of the argument must be a Crew or Member attribute (the thing being summarized).

example: assignSummarize (crews.BreakfastAssign.Company)
example: assignSummarize (members.HotelAssign.Gender)

assignSum(targetClass.assignAttr.attrName)

The assignSum() formula can only be used on Resource types (eg Vehicle, Restaurant, WorkOrder).
It is used to sum (add up) data using the crews or members who are currently assigned to each resource instance.
The 2nd-part of the argument must refer to a Crew or Member level attribute which itself IS a resource assignment attribute.
The 3rd-part of the argument must be a Crew or Member attribute (the thing being summed), and should have numeric values.

example: assignSum (crews.BreakfastAssign.Company)
example: assignSum (members.HotelAssign.Gender)

resourceAttr(assignmentAttr.resourceAttr)

This can be used at the crew or member level.
It returns the attribute value for the crew or member's assignment (if any).
The first half of the argument must be a crew|member resourceAssign attribute.

example: resourceAttr (BreakfastAssign.Address)
example: resourceAttr (VehicleAssign.Radio#)

Any resource attribute can be used as the 2nd part of the argument.

resourceSummarize(assignmentAttr.resourceAttr)

This can be used at the crew or member level.
It is used to summarize data using the assigned resources for the crew|member.
The first half of the argument must be a crew|member resourceAssign attribute.
Any resource attribute can be used as the 2nd part of the argument.

example: resourceSummarize (BreakfastAssign.name)
example: resourceSummarize (VehicleAssign.Type)

resourceSum(assignmentAttr.resourceAttr)

This can be used at the crew or member level.
It is used to sum (add up) data using the assigned resources for the crew|member.
The first half of the argument must be a crew|member resourceAssign attribute.
Any resource attribute can be used as the 2nd part of the argument, but it should have numeric values.

example: resourceSum (VehicleAssign.NumSeats)

lodgingAttr(attr) and roomAttr(attr)

This can only be used at the member level. It returns the attribute value for the member's current lodging (if any). example: lodgingAttr (name)

example: roomAttr (size)

Any lodging or roomType attribute can be used as the argument.

valueFor (attr.key)

This formula is used to extract values from attributes at the same level that contain key/value data pairs from other applications (e.g. sMART).

The formula contains a single argument with 2 parts.
The first part of the argument is the name of the attribute containing the key/value pairs.
The second part of the argument is the key name that you want to collect the value for.

example: SMART_locationData.latitude
example: SMART_locationData.longitude

ARCOS LLC
8800 Lyra Drive, Suite 200
Columbus, OH 43240
(614) 396 – 5500

Crew Manager & Platform API

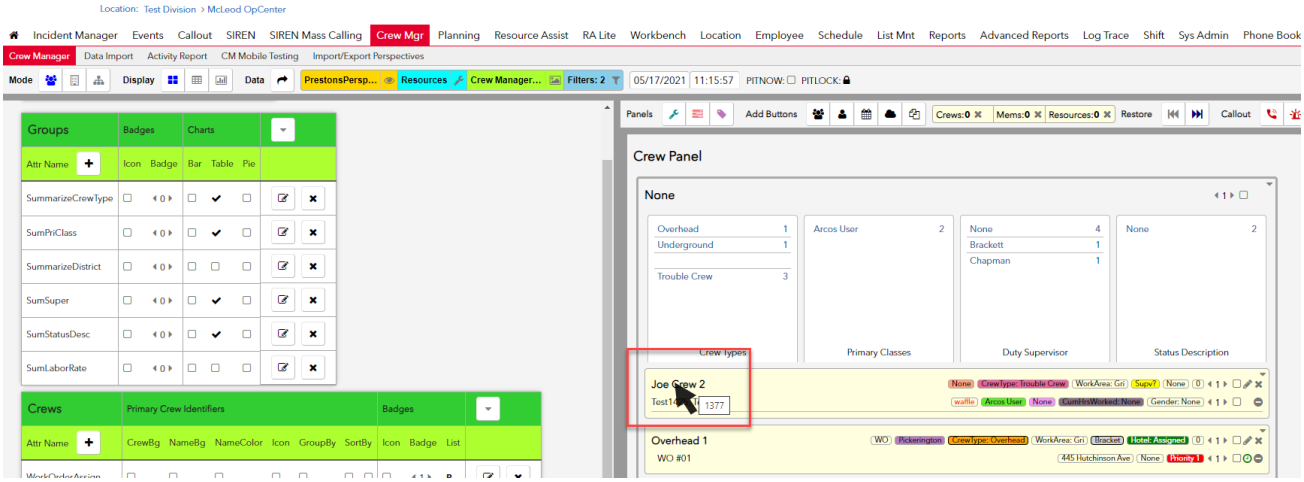
- It is essential to understand the basics of Crew Manager and use from a Business Perspective to effectively create API Calls to retrieve, create, update, or delete information.
- Perspective, Attributes, and Attribute Values must be configured within the application whether using the API or Data Loader. Attributes must exist and the values associated with those Attributes must exist or the API will return that the Attribute or Value was not found and any updates will fail.
- Whether Crews, Workorders, Equipment, or Hotels and Lodging, all resource types must have Attributes and associated values that can be filtered to bring the data in.
- Everything has an ID (Crews, Members, Equipment, WorkOrders, Assignments). When creating a new crew, resource, workorder, or hotel every new data element is assigned an ID that is used in subsequent updates or deletions. These IDs are set at time of creation and they are required for most updates.
- Must keep in mind PIT (Point in Time). If a resource is assigned an EndDate and you are looking at a PIT which is past that EndDate the API may not return the requested resource.
- All Attributes and Values are CASE Sensitive. If misspelled or case is incorrect, API Requests may not function as expected.

Point in Time (PIT)

The notion of PIT is used extensively in the Crew Manager Application, and as such, most ***/cm*** operations will accept an optional PIT parameter to determine the effective date for a given operation. If not provided, the current server time will be used.

Existing Perspectives in Crew Manager

Given access to ARCOS Crew Manager, you may take the existing information from ARCOS and apply it to the API endpoints you are needing to use. To obtain existing identifiers, like the Perspective ID or the Crew ID, you go to the Web Application and hover over the name of the resource you are looking for:



Supported Operations

getSavedSets - GET /cm/savedSets

Returns a collection of filters, perspective, and view set configurations. Also contains all ARCOS ID numbers required for data mapping between sets and other data extracts.

Request Query Parameters

Parameter	Type	Required	Valid Values	Default	Description
set_id	int	no	ARCOS Saved Set ID	None	Returns saved set with the matching ARCOS set ID
set_type	String	no	filter, perspective, viewSet	None	Returns a collection of saved sets matching the supplied type

groupsByPerspective - GET /cm/ groupsByPerspective/{perspectiveId}

Returns a collection of identifications and attributes based upon the Perspective ID provided.

Request Query Parameters

Parameter	Type	Required	Valid Values	Default	Description
set_id	int	no	Saved Set ID	None	Returns saved set with the matching set ID. Available through your web browser for Perspective details.

getCrews - GET /cm/crews

Returns a collection of Crew elements that are active as of PIT, optionally filtered by crewId or the 'attr' parameter. If no PIT is provided, then it defaults to now. This method supports pagination through the use of the 'page' parameter, 'X-Arcos-Page' header, and 'X-Arcos-Total-Pages' header.
If the response payload could contain over 5000 employee records, the use of pagination is required or the request may fail with an error that the response contains over 5000 records and pagination is required.

Request Query Parametersa

Parameter	Type	Required	Valid Values	Default	Description
pit	Datetime	no	Reference Definitions	Current Time	Establishes the POINT IN TIME for the call being made.
crew_id	List of int	no		none	Returns only the crews with matching ids
attr	String	yes			A search string to find the crews to return. The syntax is: **attr=crews;attributeName:value1,value2,value3** Replace **attributeName** with the name of the Crew Attribute to search and replace the **value1,value2,value3** clause with a comma separated list of values to match.
attributes	int	no	0, 1	1	Include AttributeRecords flag.
members	int	no	0, 1	1	Show Member Assignments.
scheds	int	no	0, 1	0	Show Member Schedules.
assigns	int	no	0, 1	0	Show Resource Assignments.

availDays	int	no	1 - 5	1	Number of days to consider for lodging based AttributeRecords.
zoom	int	no	1-72	36	Number of hours before and after PIT to search for Member Schedule Records.
changedSince	Datetime	no	*YYYYMMDD* T*HH:MM:SS* Z	Current Time	Provides attributes with a changed date that occurred after changedSince.
attrValDetails	bool	no	0,1		
perspective_id	int	no	ARCOS Saved Set ID	none	
resolvePickMember	bool	no	0,1	1	
userChangesOnly	bool	no	0,1	1	Provides elements only if they have been changed in the Web UI (Excludes API/Loader changes)
getDeleted	bool	no	0,1	0	
flexSearch					
page	Int	No, unless number of elements returned exceeds 5000	Number of page	none	Enables and controls Pagination. Value is the number of the page.

createCrew - POST /cm/crews

Creates or updates the target Crew and its corresponding attributes. For a new Crew to be generated, the ID should be set to “0” or left blank.

Request Payload

Example

```
<crew id="0" name="Sample Storm Crew"
createdDate="2019-05-15T13:24:55-04:00" changedDate="2021-04-05T10:20:07-04:00">
   <attributes>
      <attributeRecord name="WorkArea" value="Grid 25"/>
   </attributes>
   <tickets/>
   <memberCnt>0</memberCnt>
</crew>
```

Response Payload

A Status element will be returned which contains an array of OpStatus elements including the ARCOS generated unique id for the crew.

Example

```
<multistatus>
  <status id="1727" name="Sample Storm Crew" code="1" message="Element saved"/>
</multistatus>
```

getCrewsByIds - POST /cm/crewsByIds

This method accepts a Multipart payload to return data based on the following parameter values.

Request Query Parameters

Parameter	Type	Required	Valid Values	Default	Description
attributes	int	no	0,1	0	Enable AttributeRecords flag.
members	int	no	0, 1	0	Enable Member Assignments.
scheds	int	no	0, 1	0	Enable Member Schedules.
assigns	int	no	0, 1	1	Enable Resource Assignments.
availDays	int	no	1 - 5	1	List the number of days to consider for lodging based AttributeRecords.
attr	String	yes		None	A numberic string to enable the crews to added to the existing location. The syntax is: **attr=crews;attributeName:value1,value2,value3** Replace **attributeName** with the name of the Crew Attribute to search and replace the **value1,value2,value3** clause with a comma separated list of values to match.
zoom	int	no	1-72	36	Number of hours before and after PIT to enable for Member Schedule Records.
changedSince	Datetime	no	*YYYYMMDDTHH:MM:SSZ*	None	

getCrew - GET /cm/crews/{crewId}

Returns the Crew element corresponding to the provided Crew ID: {crewId}.

Request Query Parameters

Parameter	Type	Required	Valid Values	Default	Description
pit	Datetime	no	Reference Definitions	Current Time	Establishes the POINT IN TIME for the call being made.
crew_id	List of int	no		none	Returns only the crews with matching ids
attributes	int	no	0, 1	1	Include AttributeRecords flag.
members	int	no	0, 1	1	Show Member Assignments.
scheds	int	no	0, 1	0	Show Member Schedules.
assigns	int	no	0, 1	0	Show Resource Assignments.
availDays	int	no	1 - 5	1	Number of days to consider for lodging based AttributeRecords.
zoom	int	no	1-72	36	Number of hours before and after PIT to search for Member Schedule Records.
attrValDetails	bool	no	0,1		
perspective_id	int	no	ARCOS Saved Set ID	none	
resolvePickMember	bool	no	0,1	1	

getCrewForMember - GET /cm/members/{contactId}/crew

Returns the Crew element that the {contactId} is assigned to as of PIT. If the {contactId} is not assigned a Crew, a '404 Not Found' status is returned.

Request Query Parameters

Parameter	Type	Required	Valid Values	Default	Description
pit	Datetime	no	Reference Definitions	Current Time	Establishes the POINT IN TIME for the call being made.
crew_id	List of int	no		none	Returns only the crews with matching ids
attributes	int	no	0, 1	1	Include AttributeRecords flag.
members	int	no	0, 1	1	Show Member Assignments.
scheds	int	no	0, 1	0	Show Member Schedules.
assigns	int	no	0, 1	0	Show Resource Assignments.
availDays	int	no	1 - 5	1	Number of days to consider for lodging based AttributeRecords.
zoom	int	no	1-72	36	Number of hours before and after PIT to search for Member Schedule Records.
attrValDetails	bool	no	0,1		
perspective_id	int	no	ARCOS Saved Set ID	none	
resolvePickMember	bool	no	0,1	1	

getResources - GET /cm/resources

Returns a collection of Resource elements that are active as of PIT, optionally filtered by resource_id, the 'attr' parameter, or the resourceType parameter. This method supports pagination through the use of the 'page' parameter, 'X-Arcos-Page' header, and 'X-Arcos-Total-Pages' header.
If the response payload could contain over 5000 employee records, the use of pagination is required or the request may fail with an error that the response contains over 5000 records and pagination is required.

Request Query Parameters

Parameter	Type	Required	Valid Values	Default	Description
resource_id	List of int	no			Returns only the crews with matching ids
attributes	int	no	0, 1	1	Include AttributeRecords flag.
assigns	int	no	0, 1	0	Show Crew and Member Assignments.
resourceType	String	no	Existing Resource Name	none	Can be used to return only resources that match the type provided, for example **resourceType=WorkOrder** will only return WorkOrders.
attr	String	no		none	A search string to find the resources to return. The syntax is: **attr=resourceTypeName;attributeName:value1,value2,value3** Replace the resourceTypeName with an existing resourceType, **attributeName** with the name of the Resource Attribute to search and replace the **value1,value2,value3** clause with a comma separated list of values to match.
Page	Int	No, unless response contains more than 5000 records	Number of page	none	Enables and controls pagination. Value is the number of the page requested.

getResource - GET /cm/resources/{resourceId}

Returns the Resource element corresponding {resourceId}.

Request Query Parameters

Parameter	Type	Required	Valid Values	Default	Description
pit	Datetime	no	Datetime	Now	Returns only the crews with matching ids
attributes	int	no	0, 1	1	Include AttributeRecords flag.
assigns	int	no	0, 1	0	Show Crew and Member Assignments.
zoom	int	no	Number of whole hours	36	Number of hours before and after PIT to search for Member Schedule Records.

createCrews - POST /cm/batch/crews

This method is similar to the method that creates a single crew but can be used to create multiple crews in a single operation. Simply pass in a collection of Crew Elements in the request body.

Request Payload

A Crew Element (AttributeRecords and Member Assignments may be sent. Other Assignments or elements will result in an error or be ignored).

Example

```
<crews>
   <crew id="0" name="Sample Storm Crew - North">
      <attributes>
         <attributeRecord name="Storm North" value="North"/>
      </attributes>
   </crew>
   <crew id="0" name="Sample Storm Crew - South">
      <attributes>
         <attributeRecord name="Storm South" value="South"/>
      </attributes>
   </crew>
</crews>
```

Response Payload

A StatusList element will be returned which contains an array of OpStatus elements. There will be one OpStatus element for every operation that is involved for creating the Crew Element (Crew creation, AttributeRecord creations, and Member Assignments). The id field of the OpStatus will contain the id for the entity that was created during each operation step.

Example

```
<statusLists>
   <multistatus>
      <status id="2717" name="Sample Storm Crew - North" code="1" message="Element saved"/>
      <status id="15383" code="1" message="Attribute StormEvent set to North"/>
   </multistatus>
   <multistatus>
      <status id="2718" name="Sample Storm Crew - South" code="1" message="Element saved"/>
      <status id="15384" code="1" message="Attribute StormEvent set to South"/>
   </multistatus>
</statusLists>
```

createCrewMember - POST /cm/crews/{crewId}/members

Add a Member Assignment to the Crew.

Request Query Parameters

Parameter	Type	Required	Description
memberId	int	yes	The Arcos contactId of the Member to be added to the Crew
end	Datetime	no	The date the Member Assignment should end. If no value is provided, then the Member Assignment is considered permanent and all future Crew Assignments for this Member will be ended.

Example

```
<crew id="2711" name="Sample Crew ">
   <attributes>
      <attributeRecord name="Division" value="Central"/>
      <attributeRecord name="WorkLocation" value="Pleasant Grove"/>
      <attributeRecord name="Rotation_Group" value="Regular Schedule Days"/>
      <attributeRecord name="SectionCode" value="720"/>
      <attributeRecord name="CrewType" value="Overhead"/>
      <attributeRecord name="Supervisor" value="Rosa, Kyle"/>
      <attributeRecord name="NumMembers" value="3"/>
      <attributeRecord name="HasMembers" value="Yes"/>
   </attributes>
   <assignments>
      <assignment assignId="9319" assignStart="2021-05-19T03:48:39-04:00">
         <member crewId="2715" siteId="9999" id="3742" name="Rigdon, Preston">
            <attributes>
               <attributeRecord name="PrimaryLoc" value="Pleasant Grove"/>
               <attributeRecord name="PrimaryClass" value="Crew Chief"/>
               <attributeRecord name="Status" value="Available"/>
            </attributes>
         </member>
      </assignment>
   </assignments>
</crew>
```

Response Payload

An OpStatus object representing the status of the Member Assignment. The ‘id’ field of the OpStatus object will be set to the ‘assignId’ of the Member Assignment.

Example

```
<opStatus id="9322" code="1" message="Assignment saved"/>
```

updateCrewMember - DELETE /cm/crews/{crewId}/members

Remove a Member Assignment from the Crew.

Request Query Parameters

Parameter	Type	Required	Description
assignId	int	yes	The assignId of the Assignment to end.

Response Payload

An OpStatus object representing the status of ending the Member Assignment.

Example

```
<opStatus id="9084" code="1" message="Assignment saved"/>
```

deleteCrew - DELETE /cm/crews/{crewId}

This method will set the deleted date on the Crew element that matches the crewId provided, effectively removing this Crew from the Crew Manager application as of PIT.

Request URI Example

https://qa.rostermonster.com/arcos/rest/cm/crews/*2716*

Response Payload

An OpStatus element will be returned with the result of the operation.

Example

```
<opStatus id="2716" code="1" message="OK"/>
```

saveCrewAttribute - POST /cm/crews/{crewId}/attributes
This method can be used to change static Crew Attributes as of PIT.

AttributeRecords represent the setting of an Attribute to a specific 'value' for a given period of time ('start' to 'end'). When representing Resource Assignments, the 'value' of the AttributeRecord will be set to the 'resourceId' of the Resource that is assigned to the Crew or Member.

Most Attributes within Crew Manager are non-shared. This means that setting an Attribute value will actually end any existing AttributeRecord and start a new one as of PIT. Resource Assignments however, are non-shared AttributeRecords and can have multiple values at any one time.

AttributeRecords that are posted to this method will have their 'start' values set to PIT. If an 'end' field is not provided, then the AttributeRecord will be considered a permanent Attribute and all future AttributeRecords for that Attribute will be removed.

To Remove an Attribute value, set the 'value' field of the AttributeRecord to 'None'. When ending a Resource Assignment, the correct 'changeId' must be provided, for non-Resource related Attributes, the 'changeId' field is not required.

Request URI Example **-** *https://qa.rostermonster.com/arcos/rest/cm/crews/2711/attributes*

Payload Example

```
<attributes>
  <attributeRecord name="VehicleAssign" value="2583" end="2021-05-18T15:01:30-05:00"/>
</attributes>
```

Response Payload
A collection of OpStatus elements will be returned, one for each AttributeRecord that was provided.

Example

```
<opStatuses>
  <opStatus id="2711" code="1" message="Updating Attributes"/>
  <opStatus id="15376" code="1" message="Attribute VehicleAssign set to 2583"/>
</opStatuses>
```

saveMemberAttribute - POST /cm/members/{memberId}/attributes

This method can be used to change static Member Attributes as of PIT. See additional notes from the "Change Crew Attributes" section.

Request Payload - https://qa.rostermonster.com/arcos/rest/cm/crews/2711/members

A collection of AttributeRecord Elements.

Example

```
<attributes>
  <attributeRecord name="VehicleAssign" value="2583" />
</attributes>
```

Response Payload

A collection of OpStatus elements will be returned, one for each AttributeRecord that was provided.

Example

```
<opStatuses>
  <opStatus id="2711" code="1" message="Updating Attributes"/>
  <opStatus id="15431" code="1" message="Attribute VehicleAssign set to 2583"/>
</opStatuses>
```

Alternative endpoint: If the contactId is not the preferred ID value, the vruId or webId can be used as search parameters for the employee -
https://qa.rostermonster.com/arcos/rest/cm/members/0/attributes?searchType=webId&searchValue=prigdon

createResource - POST /cm/resources

Create a new instance of a Resource. This endpoint can be used to create the Resource and its starting AttributeRecords. **Note that 'resourceType' must be provided so that the appropriate Resource type can be created.**

Request Payload

A **Resource** Element and its AttributeRecords may be sent. Any Assignments given will be ignored (See the 'Change Crew Attributes' section for details about Resource Assignments.

Examples

```
<resource resourceType="WorkOrder" resourceTypeId="968" id="0" name="123456-1">
  <attributes>
    <attributeRecord name="Status" value="Open"/>
  </attributes>
</resource>
```

OR

```
<resource resourceType="Vehicle" resourceTypeId="1001" id="0" name="Truck">
```

```
    <attributes>
      <attributeRecord name="Make" value="Cybertruck" />
    </attributes>
  </resource>
```

Response Payload

An **StatusList** element will be returned which contains an array of OpStatus elements. There will be one OpStatus element for every operation that is involved for creating the Resource Element. The id field of the OpStatus will contain the id for the entity that was created during each operation step.

Example

```
  <multistatus>
    <status id="2583" name="Truck" code="1" message="Element saved"/>
    <status id="11810" code="1" message="Attribute Make set to Cybertruck"/>
  </multistatus>
```

createResources - POST /cm/batch/resources

This method is similar to the method that creates a single Resource and can be used to create multiple Resources in a single operation. Pass in a collection of Resource Elements in the request body.

Request Payload

A collection of Resource Elements.

Example

```
  <resource resourceType="Vehicle" resourceTypeId="1001" id="0" name="Truck">
    <attributes>
      <attributeRecord name="Make" value="Cybertruck" />
    </attributes>
    <attributes>
      <attributeRecord name="Make" value="Rivian" />
    </attributes>
    <attributes>
      <attributeRecord name="Make" value="Lightning" />
    </attributes>
  </resource>
```

Response Payload

A collection of StatusList elements will be returned, one for each of the Resource objects that were created.

Example

```
  <multistatus>
    <status id="2585" name="Truck" code="1" message="Element saved"/>
    <status id="11818" code="1" message="Attribute Make set to Cybertruck"/>
    <status id="11818" code="1" message="Attribute Make set to Rivian"/>
```

```
    <status id="11818" code="1" message="Attribute Make set to Lightning"/>
</multistatus>
```

saveResourceAttribute - POST /cm/resources/{resourceId}/attributes

This method can be used to change static Resource Attributes as of PIT. See additional notes from the "Change Crew Attributes" section.

Request Payload

A collection of AttributeRecord Elements.

Example

```
<attributes>
    <attributeRecord name="VehServiced" value="Y"/>
</attributes>
```

Response Payload

A collection of OpStatus elements will be returned, one for each AttributeRecord that was provided.

Example

```
<opStatuses>
    <opStatus id="1094" code="1" message="Updating Attributes"/>
    <opStatus id="0" code="1" message="Attribute VehServiced has been updated."/>
</opStatuses>
```

createWotes - POST /cm/resources/{resourceId}/timeEntries

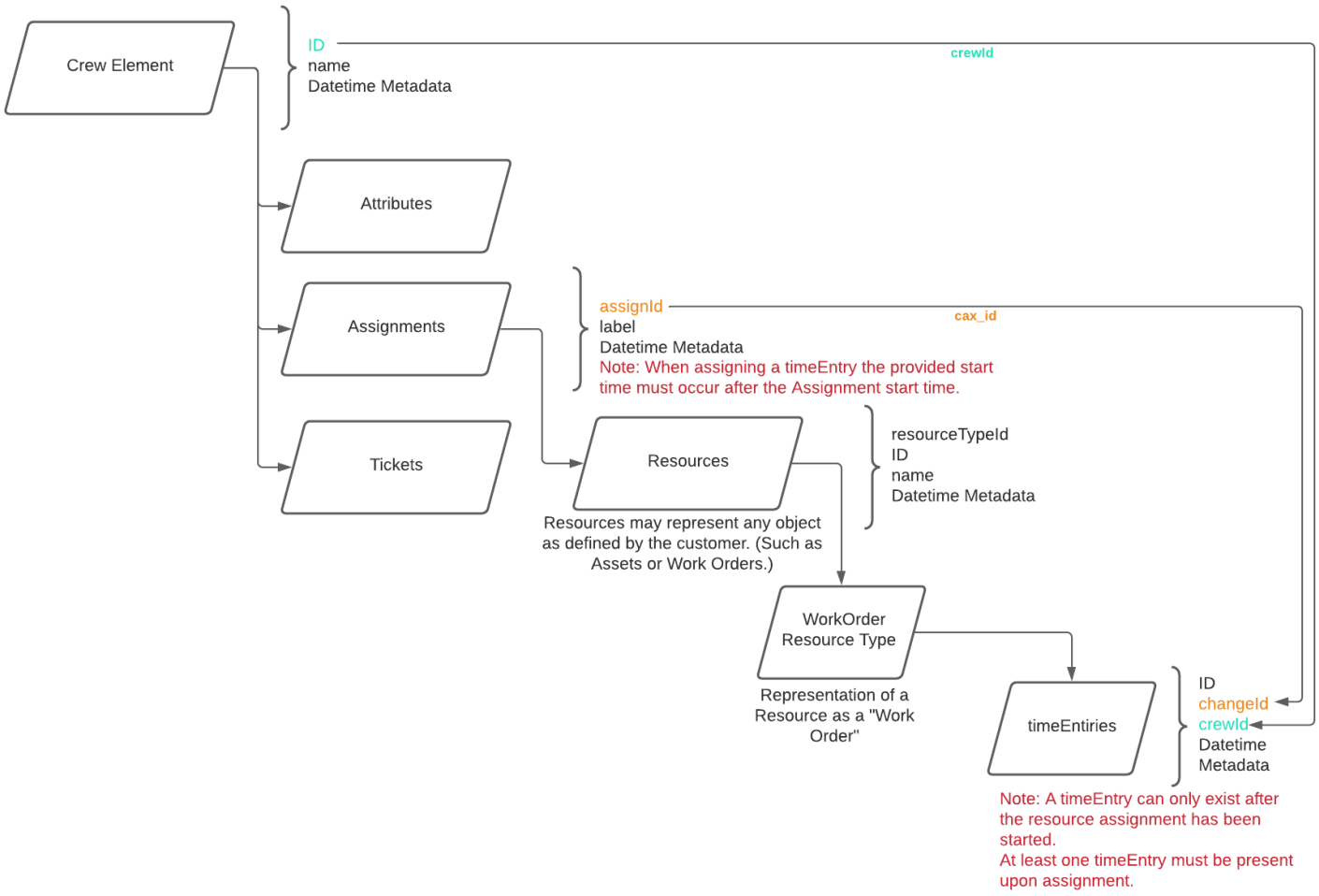
This method can be used to add new WorkOrderTimeEntry elements (Wotes) for the WorkOrder represented by resourceId.

This call does not create a standalone work order, refer to “Create a new Resource - POST /cm/resources” for creating a new work order.

WorkOrderTimeEntry elements represent a Crew working on a specific WorkOrder for a given period of time (‘start’ - ‘end’), so ‘start’ and ‘end’ or ‘hours’ must be provided.

The ‘changeId’ field is used to find the Crew Attribute that represents the WorkOrder Assignment and is required.

In order for a successful response for this call, you will need to know the start time of the assignment for the workOrder you are referencing. If the timeEntry is not within the parameters of the assigned work order the time entry will not be added or adjusted.



Request Payload

A collection of WorkOrderTimeEntry Elements.

Example

```
<workOrderTimeEntries>
  <workOrderTimeEntry changeId="6165" crewId="1624"
  start="2021-05-27T14:14:27-04:00" hours="6.5"/>
</workOrderTimeEntries>
```

Response Payload

A collection of OpStatus elements will be returned, one for each WorkOrderTimeEntry that was provided.

Example

```
<opStatuses>
  <opStatus id="1177" code="1" message="New TimeEntry created: 1177"/>
</opStatuses>
```

updateWotes - PUT /cm/resources/{resourceId}/timeEntries

This method can be used to update existing WorkOrderTimeEntry elements. Reference important notes in the "Create Work Order Time Entry" operation section.

Request Payload

A collection of WorkOrderTimeEntry Elements to update.

Example

```
<workOrderTimeEntries>
  <workOrderTimeEntry id="1170" changeId="15442" crewId="2714" start="2021-05-27T14:14:28-04:00"
end="2021-12-27T14:14:28-04:00" changeDate="2021-12-15T16:50:25-04:00"/>
</workOrderTimeEntries>
```

Response Payload

A collection of OpStatus elements will be returned, one for each WorkOrderTimeEntry that was provided.

Example

```
<opStatuses>
  <opStatus id="0" code="1" message="TimeEntry updated: 1170"/>
</opStatuses>
```

deleteWote - DELETE /cm/resources/{resourceId}/timeEntries/{woteId}

This method can be used to remove an existing WorkOrderTimeEntry element.

Response Payload

A collection of OpStatus elements will be returned, one for each WorkOrderTimeEntry that was provided.

deleteResource - DELETE /cm/resources/{resourceId}

This method will set the deleted date on the Resource element that matches the resourceId provided, effectively removing this Resource from the Crew Manager application as of PIT.

Request Payload

The method supports a Multipart Form payload containing a parameter of 'pit' and a value of ISO-8601 Datetime. The supplied 'pit' defines the Point In Time that the resourceId will be deleted.

Multipart Form Parameters

Parameter	Type	Required	Valid Values	Default	Description
pit	Datetime	no	Datetime	NOW	The Point in Time for the resourceId provided in the URI to be reflected as deleted in Crew Manager.

Response Payload

An OpStatus element will be returned with the result of the operation.

audit - GET /cm/audit

This method will return a set of any crews that exist(ed) and match(ed) the search attr or id provided within the startDate and endDate window will be returned. If crewAttr, memberAttr and resourceAttr parameters are set to 1, then all changeable attributes within those crews should be returned. Note that System, Calculated Value, and Pseudo attributes will not be returned in this dataset.

Request Query Parameters

Parameter	Type	Required	Valid Values	Default	Description
id	int	no		none	Returns only the crews with matching ids
attr	String	yes			A search string to find the crews to return. The syntax is: **attr=crews;attributeName:value1,value2,value3** Replace **attributeName** with the name of the Crew Attribute to search and replace the **value1,value2,value3** clause with a comma separated list of values to match.
crewAttr	bool	no	0, 1	1	Include AttributeRecords flag.
members	bool	no	0, 1	1	Include Members.
membersAttr	bool	no	0, 1	1	Include Member Attributes.
resources	bool	no	0, 1	1	Include Resources.
resourceAttr	bool	no	0, 1	1	Include Resource Attributes.
changedSince	Datetime	no	YYYYMMDDTHH:MM:SSZ	Current Time	Provides attributes with a changed date that occurred after changedSince.
userChangesOnly	bool	no	0,1	1	Provides elements only if they have been changed in the Web UI (Excludes API/Loader changes)

Response Payload

A collection of Crews that match the query parameters.

Example

```
<crews>
  <crew id="1155" name="Callout for Sch OT" createdDate="2018-01-09T21:19:57-
  05:00" deletedDate="" changeDate="2018-06-27T10:19:29-04:00">
  <attributes>
   <attributeRecord name="CrewType" value="Overhead" start="2018-04-16T12:49:10-04:00"
        end="2018-10-24T12:37:22-04:00" changeId="2108" changeDate="2018-10-24T12:37:23-04:00">
        </attributeRecord>
   <attributeRecord name="Crews" value="Troublemen" start="2018-04-16T12:49:10-04:00"
        end="" changeId="2107" changeDate="2018-04-16T12:49:18-04:00"></attributeRecord>
  </attributes>
  <assignments>
   <assignment assignId="1515" label="null" assignStart="2018-04-16T12:49:10-04:00"
        assignEnd="2018-10-09T11:16:37-04:00" changeDate="2018-10-09T11:16:38-04:00">
    <member crewId="1155" id="1014" name="Ballstaedt, Dan">
     <attributes>
      <attributeRecord name="LaborRate" value="75" start="2018-03-28T15:31:09-04:00"
              end="" changeId="1389" changeDate="2018-03-28T15:35:22-04:00"></attributeRecord>
     </attributes>
     <assignments>
      <assignment assignId="1018" label="Meals" assignStart="2017-03-20T10:05:43-04:00"
              assignEnd="" changeDate="2017-03-20T10:05:44-04:00">
               <resource resourceType="Meals" resourceTypeId="1002" id="1013" name="Breakfast Box" createdDate
               ="2017-03-20T09:29:25-04:00" deletedDate="" changeDate="2017-03-20T09:29:26-04:00">
          <attributes>
           <attributeRecord name="MealTime" value="Breakfast"
                    start="2017-03-20T09:29:25-04:00" end="" changeId="1033" changeDate="2017-03-
                    20T09:29:26-04:00">
                    </attributeRecord>
          </attributes>
         </resource>
        </assignment>
     </assignments>
    </member>
   </assignment>
   <assignment assignId="2105" label="WorkOrder" assignStart="2018-04-16T12:49:10-04:00"
        assignEnd="" changeDate="2018-04-16T12:49:18-04:00">
    <resource resourceType="WorkOrder" resourceTypeId="968" id="1080" name="Car Hit Pole"
        createdDate="2017-11-06T09:49:27-05:00" deletedDate=""
        changeDate="2017-11-06T09:49:29-05:00">
     <attributes>
      <attributeRecord name="Address" value="45336 Highway 10"
              start="2017-11-06T09:49:27-05:00" end="2018-10-24T12:38:16-04:00"
              changeId="1247" changeDate="2018-10-24T12:38:17-04:00">
              </attributeRecord>
     </attributes>
    </resource>
   </assignment>
  </assignments>
 </crew>
</crews>
```

getResourceTypes - GET /cm/resourceTypes

This method returns a collection of available resourceTypeDefinitions.

Response Payload

Example

```
<resourceTypeDefinitions>
  <resourceTypeDefinition id="1012" name="Vehicle"/>
  <resourceTypeDefinition id="1009" name="Equipment"/>
  <resourceTypeDefinition id="1010" name="Technology"/>
</resourceTypeDefinitions>
```

getLodgings - GET /cm/lodgings

This method returns a collection of lodgings and their attributes. This method supports pagination through the use of the 'page' parameter, 'X-Arcos-Page' header, and 'X-Arcos-Total-Pages' header.
If the response payload could contain over 5000 employee records, the use of pagination is required or the request may fail with an error that the response contains over 5000 records and pagination is required.

Request Query Parameters

Parameter	Type	Required	Valid Values	Default	Description
pit	datetime	No	Reference Definitions	none	Returns only the lodgings within provided point in time.
lodging_id	int	No		none	Returns only the lodging with matching ids.
attributes	int	No		none	Include AttributeRecords flag
assigns	int	No		none	Show Resource Assignments.
availDays	Int	No		none	Number of days to consider for lodging based AttributeRecords.
attr	list	No		none	A search string to find the lodgings to return.
zoom	Int	No		none	Number of hours before and after PIT to search for Member Lodging Records.
changedSince	string	No		none	Provides lodging with a changed date that occurred after changedSince.
userChangesOnly	int	No		none	Provides elements only if they have

					been changed in the Web UI (Excludes API/Loader changes)
getDeleted	int	No		none	Provides elements only if they have been removed
Page	Int	Only if the Response data is over 5000		none	Enables and controls pagination. Value is the number of the page requested.

Response Payload

Example

```
<lodgings>
  <lodging id="1025" name="South Boston Inn"
        createdDate="2018-03-13T11:29:12-04:00" changedDate="2021-05-12T13:56:08-04:00">
    <attributes>
      <attributeRecord name="City" value="Columbus"
        start="2018-03-13T11:29:12-04:00" changeId="1037" changeDate="2018-03-13T11:29:12-04:00"/>
      <attributeRecord name="Contact" value="Preston Rigdon"
        start="2021-05-12T10:24:36-04:00" changeId="1118" changeDate="2021-05-12T10:24:36-04:00"/>
      <attributeRecord name="Address" value="1234 Old Town Road"
        start="2021-05-12T10:48:32-04:00" changeId="1121" changeDate="2021-05-12T10:48:32-04:00"/>
      <attributeRecord name="Hotel" value="Fancy"
        start="2021-05-12T10:48:32-04:00" changeId="1122" changeDate="2021-05-12T10:48:32-04:00"/>
    </attributes>
  </lodging>
</lodgings>
```

getLodging - GET /cm/lodgings/{lodgingId}

This method returns a the lodging element for the provided {lodgingId}

Request Query Parameters

Parameter	Type	Required	Valid Values	Default	Description
pit	Datetime	No			Establishes the POINT IN TIME for the call being made.
attributes	int	No			Include AttributeRecords flag
assigns	int	No			Show lodging Assignments
availDays	int	No			Number of days to consider for lodging based AttributeRecords.
zoom	int	No			Number of hours before and after PIT to search for Lodging Records.

Response Payload

Example

```
<lodgings>
  <lodging id="1025" name="South Boston Inn"
           createdDate="2018-03-13T11:29:12-04:00" changedDate="2021-05-12T13:56:08-04:00">
    <attributes>
      <attributeRecord name="City" value="Columbus"
           start="2018-03-13T11:29:12-04:00" changeId="1037" changeDate="2018-03-13T11:29:12-04:00"/>
      <attributeRecord name="Contact" value="Preston Rigdon"
           start="2021-05-12T10:24:36-04:00" changeId="1118" changeDate="2021-05-12T10:24:36-04:00"/>
      <attributeRecord name="Address" value="1234 Old Town Road"
           start="2021-05-12T10:48:32-04:00" changeId="1121" changeDate="2021-05-12T10:48:32-04:00"/>
      <attributeRecord name="Hotel" value="Fancy"
           start="2021-05-12T10:48:32-04:00" changeId="1122" changeDate="2021-05-12T10:48:32-04:00"/>
    </attributes>
  </lodging>
</lodgings>
```

getCrewMembers - GET /cm/crews/{crewId}/members

Returns the members element for the provided {crewId} that is in effect at PIT. If PIT is not provided, the current time is used.

Request Query Parameters

Parameter	Type	Required	Valid Values	Default	Description
crewId	int	No	crewId	None	Returns only the crews with matching ids
pit	Datetime	No	Datetime	None	Establishes the POINT IN TIME for the call being made.
attributes	int	No	0, 1	1	Include AttributeRecords flag.
scheds	int	No	0, 1	1	Show Member Schedules.
attrValDetails	int	No	0, 1	0	
resolvePickMember	int	No	0, 1	1	

Response Payload

Example

```
<members>
  <member crewId="1624" siteId="9999" id="4123" name="Dailey, James">
    <attributes>
      <attributeRecord name="PrimaryLoc" value="McLeod OpCenter"/>
      <attributeRecord name="PrimaryClass" value="Arcos User"/>
      [...]
    </attributes>
    <scheduleRecords/>
  </member>
</members>
```

getResource - GET /cm/resources/{resourceId}

Returns the resource element for the provided {resourceId}.

Request Query Parameters

Parameter	Type	Required	Valid Values	Default	Description
resource_id	Int	no			Returns only the resources with matching ids.
attributes	Bool	no	0, 1	1	Include AttributeRecords flag.
assigns	Bool	no	0, 1	0	Show Crew and Member Assignments.
pit	Datetime	no		None	
zoom	Int	no		36	

getCrewsForRA - GET /cm/crewsForRa

This method is published in our Web Application Description language (WADL) but not yet described here. If you require additional information on this method, please speak with an ARCOS Representative.

Request Query Parameters

Parameter	Type	Required	Valid Values	Default	Description
startTime	Datetime	Yes		None	
endTime	Datetime	Yes		None	
perspective_id	Int	No		None	

getWorkIntegration - GET /cm/workIntegration

Returns the status of the Work Integration. If the Work Integration is enabled the response will be “true”. If the Work Integration is disabled the response will be “false”.

getRoomTypes - GET /cm/roomTypes

Provides the Room Types currently available in ARCOS as represented by the roomTypes element.

Request Query Parameters

Parameter	Type	Required	Valid Values	Default	Description
pit	Datetime	No		Now	
attributes	Bool	No	0, 1	1	

Response Payload

Example

```
<roomTypes>
  <roomType size="2" name="Jacuzzi" abbreviation="JZ" id="1008"
          createdDate="2019-04-15T08:54:29-04:00" changedDate="2019-04-15T09:42:46-04:00"/>
  <roomType size="50" name="Single" abbreviation="SINGLE" id="1009"
          createdDate="2019-04-15T08:54:29-04:00" changedDate="2019-04-15T09:44:27-04:00"/>
  <roomType size="60" name="Double" abbreviation="DBL" id="1011"
          createdDate="2019-04-18T11:46:42-04:00" changedDate="2019-04-18T11:49:13-04:00">
    <attributes>
      <attributeRecord name="Smoking" value="NO"
          start="2020-02-06T14:52:51-05:00" changeId="1006" changeDate="2020-02-06T14:55:18-05:00"/>
    </attributes>
  </roomType>
</roomTypes>
```

getAvlMemberBadges - GET /cm/avlMemberBadges

This method is published in our Web Application Description language (WADL) but not yet described here. If you require additional information on this method, please speak with an ARCOS Representative.

Request Query Parameters

Parameter	Type	Required	Valid Values	Default	Description
member_id	Int	Yes	memberId	None	Same as the contactId for the user
set_id	Int		setId	None	Possibly the perspectiveId
pit	Datetime	No	Datetime	Now	
resolvePickMember	Bool	No	0, 1	0	

getGroups - GET /cm/groups

This method returns a collection of crewGroups matching the attr query.

Request Query Parameters

Parameter	Type	Required	Valid Values	Default	Description
pit	Datetime	No		Now	
attributes	Bool	No		1	
members	Bool	No		1	
scheds	Bool	No		0	
assigns	Bool	No		0	
availDays	Int	No		3	
attr	String	Yes		None	
zoom	Int	No		36	
groupBy	String	No		None	
changedSince	Datetime	No		None	
perspectiveId	Int	No		None	

sync - GET /cm/sync

Returns information for the time range between startDate and endDate. The 'action' parameter determines

whether the method will return information for resources or lodgings.

Request Query Parameters

Parameter	Type	Required	Valid Values	Default	Description
startDate	Datetime	Y	ISO-8601 Datetime	null	An item must exist (not be deleted) AFTER this date. The max range between startDate and endDate is 365 days.
endDate	Datetime	Y	ISO-8601 Datetime	null	An item must have been created BEFORE this date. The max range between startDate and endDate is 365 days.
changedSince	Datetime	N	ISO-8601 Datetime	1970-01-01	Format example: 2020-07-10T06:55:23 Return only data changed since this date. NOTE: to be found, an entity should be active at time of PIT. This means that records created AFTER PIT, or deleted BEFORE PIT, will NOT be returned.
userChangesOnly	Bool	N	0, 1	0	If set to 1, results will NOT include records changed by Loader processes.
action	String	Y	See Description	null	The valid values are resources and lodgings.

Response Payload

This method streams out an XML or JSON strings, depending on the 'Accept' header value set in the request.

Example

```
<resources>
   <resource resourceType="WorkOrder" resourceTypeId="968" id="1006" name="KF-10/29" createdDate="2015-
10-27T23:17:23-04:00" deletedDate=""
changeDate="2015-10-28T07:19:36-04:00">
      <attributes>
         <attributeRecord name="EstHours" value="4" start="2015-10-27T23:17:23-
04:00" end="" changeId="1017" changeDate="2015-10-28T07:19:36-04:00"></attributeRecord>
         <attributeRecord name="LOC2" value="Test" start="2015-10-27T23:17:23-
04:00" end="" changeId="1018" changeDate="2015-10-28T07:19:36-04:00"></attributeRecord>
      </attributes>
   </resource>
</resources>
```

syncCrews - GET /cm/sync/crews

Returns information without specifying the precise PIT. Instead, takes the time range between startDate and endDate.

Request Query Parameters

Parameter	Type	Required	Valid Values	Default	Description
startDate	Datetime	Y	ISO-8601 Datetime	null	An item (crew, resource, attribute etc.) must exist (not be deleted) AFTER this date. The max range between startDate and endDate is 365 days.
endDate	Datetime	Y	ISO-8601 Datetime	null	An item (crew, resource, attribute etc.) must have been created BEFORE this date. The max range between startDate and endDate is 365 days.
attr	String	N	See Description	null	A search string to find the crews to return. Multiple attr parameters can be given. The syntax is: attr=crews;attributeName:value1,value2,value3. Replace attributeName with the name of the Crew Attribute to search, and replace the values with comma-separated list of values to match. Please note that multiple attr parameters imply the AND condition, so in this example we will look for crews with CrewType = Support AND Division = Underground: attr=crews;CrewType:Support&attr=crews;Division:Underground However, multiple values for the same attr imply the OR condition, so in this example we will look for crews with CrewType either 'Support' OR 'Maintenance': attr=crews;CrewType:Support,Maintenance. Special cases: - attributeName is 'name', as in attr=crews;name:BigBen - the search

					will look for crews with name = BigBen. - attributeName is 'any_attr', as in attr=crews;any_attr:BigBen - the search will look for crews which has any attribute with value BigBen.
Id	Int	N	See Description	null	Allows to specify crew_ids of interest. Multiple id parameters can be given. Each id parameter is added separately (e.g. id=10905&id=5292)
changedSince	Datetime	N	ISO-8601 Datetime	1970-01-01	Format example: 2020-07-10T06:55:23 Return only data changed since this date. NOTE: to be found, an entity should be active at time of PIT. This means that records created AFTER PIT, or deleted BEFORE PIT, will NOT be returned.
userChangesOnly	Bool	N	0, 1	0	If set to 1, results will NOT include records changed by Loader processes.
crewAttr	Bool	N	0, 1	0	Return attributes of crews.
members	Bool	N	0, 1	1	Return members assigned to crews.
memberAttr	Bool	N	0, 1	0	Return attributes of members assigned to found crews.
resources	Bool	N	0, 1	0	Return resources assigned to found crews.
resourceAttr	Bool	N	0, 1	0	Return attributes of resources assigned to found crews.
schedules	Bool	N	0, 1	0	Return schedule information for members assigned to found crews.

Response Payload

This method streams out an XML or JSON strings, depending on the 'Accept' header value set in the request.

Example

```
<crews>
   <crew id="1081" name="Underground" createdDate="2017-09-29T11:04:48-
04:00" deletedDate="" changeDate="2018-04-16T12:49:18-04:00">
      <attributes>
         <attributeRecord name="CrewType" value="Underground" start="2018-04-16T12:49:10-
04:00" end="" changeId="2115" changeDate="2018-04-16T12:49:18-04:00"></attributeRecord>
      </attributes>
      <assignments>
         <assignment assignId="1518" label="null" assignStart="2018-04-16T12:49:10-04:00" assignEnd="2019-
11-27T13:35:39-05:00" changeDate="2019-11-27T13:35:39-05:00">
```

```
            <member crewId="1081" id="1086" name="Livingston, Andrew">
               <attributes>
                  <attributeRecord name="Gender" value="Male" start="2018-05-03T08:58:20-
04:00" end="" changeId="1411" changeDate="2018-05-03T09:09:32-04:00"></attributeRecord>
               </attributes>
               <assignments></assignments>
               <scheduleRecords>
                  <scheduleRecord memexId="26787" eventid="1008" description="Working - Normal Shift" abbrevi
ation="sh" comment="" assignStart="2019-03-06T06:00:00-05:00" assignEnd="2019-03-06T18:00:00-
05:00"></scheduleRecord>
               </scheduleRecords>
            </member>
         </assignment>
      </assignments>
   </crew>
</crews>
```

createLodging - POST /cm/lodgings

Creates Lodging.

Request Payload

Example

```
<lodging id="0" name="South Boston Inn" available="16" createdDate="2021-05-11T00:00:00-
04:00" changedDate="2021-05-11T11:03:32-04:00"/>
```

Response Payload

A Status element will be returned which contains an array of MultiStatus elements including the ARCOS generated unique id for the lodging.

Example

```
<multistatus>
   <status id="1058" name="South Boston Inn" code="1" message="Element saved"/>
   <status id="1123" code="1" message="Attribute Capacity set to [ ]"/>
</multistatus>
```

createLodgings - POST /cm/batch/lodgings

Creates multiple Lodgings.

Request Payload

Example

```
<lodgings>
    <lodging id="0" name="Kyle's Airbnb"/>
```

```
    <lodging id="0" name="Kyle's Airbnb 2"/>
    <lodging id="0" name="Kyle's Airbnb 3"/>
</lodgings>
```

Response Payload

A Status element will be returned which contains an array of MultiStatus elements including the ARCOS generated unique id for the lodging.

Example

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<statusLists>
    <multistatus>
        <status id="1063" name="Kyle's Airbnb" code="1" message="Element saved"/>
        <status id="1128" code="1" message="Attribute Capacity set to []"/>
    </multistatus>
    <multistatus>
        <status id="1064" name="Kyle's Airbnb 2" code="1" message="Element saved"/>
        <status id="1129" code="1" message="Attribute Capacity set to []"/>
    </multistatus>
    <multistatus>
        <status id="1065" name="Kyle's Airbnb 3" code="1" message="Element saved"/>
        <status id="1130" code="1" message="Attribute Capacity set to []"/>
    </multistatus>
</statusLists>
```

deleteLodging - DELETE /cm/lodgings/{lodgingId}

Deletes a given lodging. Path parameter: lodgingId - id of the lodging to delete.

Response Payload

This method returns an OpStatus element.

ARCOS LLC
8800 Lyra Drive, Suite 200
Columbus, OH 43240
(614) 396 – 5500

getAttributes - GET /cm/attributes

Provides the crew manager attributes currently available in ARCOS as represented by the attributes element.

Example - https://qa.rostermonster.com/arcos/rest/cm/attributes?getValues=1

Request Query Parameters

Parameter	Type	Required	Valid Values	Default	Description
getValues	int	No	0,1	1	If set to 1 - the values defined for the attribute will also be returned. If set to 0 - only Attribute Definitions will be returned.
target	string	No	groups, crews, members, resources	1	If given, only attributes for this target will be returned.
inUseOnly	int	No	0,1	0	If set to 1, and target is set to one of following: 'crews', 'members', 'lodgings', 'roomTypes', then only attributes which have been actually used for that target will be returned.
perspectiveId	string	No			If given, only attributes present in this perspective will be returned.

Response Payload

Example

```
<attributeDefinitions>
   <attributeDefinition target="groups" name="NumCrews">
      <values/>
   </attributeDefinition>
   <attributeDefinition target="groups" name="NumCrewsWorking">
      <values/>
   </attributeDefinition>
   <attributeDefinition target="groups" name="NumMembers">
      <values/>
   </attributeDefinition>
```

saveLodgingAttribute - POST /cm/lodgings/{id}/attributes

Creates/updates Attributes for the given lodgings..

Request Payload

Example

```
<lodging id="0" name="South Boston Inn" available="16" createdDate="2021-05-11T00:00:00-
04:00" changedDate="2021-05-11T11:03:32-04:00"/>
```

Response Payload

This method returns a List of OpStatus elements, one OpStatus per attribute.

Example

```
<multistatus>
   <status id="1058" name="South Boston Inn" code="1" message="Element saved"/>
   <status id="1123" code="1" message="Attribute Capacity set to [ ]"/>
</multistatus>
```

createRoomTypes - POST /cm/batch/roomTypes

Creates/updates multiple room types and attributes for the given lodgings..

Request Payload

Example

```
<roomTypes>
   <roomType size="10" name="smoke" abbreviation="S" id="1010" createdDate="2021-11-14T14:34:02-
04:00" changedDate="2021-11-14T14:34:02-04:00"/>
   <roomType size="1" name="Double Smoke" abbreviation="2xS" id="1011" createdDate="2021-11-14T14:34:02-
04:00" changedDate="2021-11-14T15:34:48-04:00">
      <attributes></attributes>
   </roomType>
</roomTypes>
```

Response Payload

This method returns a List of OpStatus elements, one OpStatus per attribute.

Example

```
<statusLists>
   <multistatus>
```

```
    <status id="1010" code="1" message="Element saved"/>
  </multistatus>
  <multistatus>
    <status id="1011" code="1" message="Element saved"/>
  </multistatus>
</statusLists>
```

saveAttributes - POST /cm/batch/attributes

Used to add attributes to target entities. Targets can be crews, members, lodgings, roomTypes, or resource types defined in the system.

The “Point In Time” sets the effective date for a given operation. If not provided, the current server time will be used.

Request Payload

Example

```
<attributeCollections>
  <attributeCollection targetType="Members" id="0" searchType="webId" searchValue=" member_username">
    <attributes>
      <attributeRecord name="WorkedHrsThreshold" value="Active"/>
    </attributes>
  </attributeCollection>
</attributeCollections>
```

Response Payload

This method returns a List of OpStatus elements, one OpStatus per attribute.

Example

```
<statusLists>
  <multistatus>
    <status id="member_username" code="1" message="Updating Attributes"/>
    <status id="2914" code="1" message="Attribute WorkedHrsThreshold set to Active"/>
  </multistatus>
</statusLists>
```

createRoomType - POST /cm/roomType

Creates/updates room types and room attributes for the given lodging.

Request Payload

Example

```
<roomType size="10" name="smoke" abbreviation="S" id="0" createdDate="2021-11-14T14:34:02-04:00" changedDate="2021-11-14T14:34:02-04:00"/>
```

Response Payload

This method returns a List of OpStatus elements, one OpStatus per attribute.

Example

```
<multistatus>
   <status id="1010" code="1" message="Element saved"/>
</multistatus>
```

getAttributesByNames - GET /cm/attributesByName

Gets Attribute Definitions, and values pre-defined for those attributes.

Example - https://qa.rostermonster.com/arcos/rest/cm/attributes?getValues=1

Request Query Parameters

Parameter	Type	Required	Valid Values	Default	Description
attributesByName	string	No	0,1	1	If given, only attributes for this name will be returned
target	string	No			If given, only attributes for this target will be returned.

Response Payload

Example

```
<attributeDefinitions>
  <attributeDefinition target="groups" name="NumCrewsWorking" addDate="12/28/2017 12:21:56">
    <properties>
      <item name="ord" value="999"/>
      <item name="cd" value="03/05/2019 20:42:05"/>
      <item name="hasProp" value="1"/>
      <item name="sn" value="1"/>
      <item name="locked" value="1"/>
      <item name="pseudo" value="1"/>
      <item name="wt" value="px"/>
      <item name="desc" value=""/>
    </properties>
    <values>
      <value value="Default" ord="999" bgColor="" badgeText="" image="" fontColor="" borderColor=""/>
    </values>
```

```
    </attributeDefinition>
</attributeDefinitions>
```

saveRoomTypeAttribute - POST /cm/roomTypes/{id}/attributes

Request Payload

Example

```
<attributes>
    <attributeRecord name="Smoking" value="No" start="2021-12-15T09:26:46-
05:00" changeId="1010" changeDate="2021-12-15T09:26:46-05:00"/>
</attributes>
```

Response Payload

This method returns a List of OpStatus elements, one OpStatus per attribute.

Example

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<opStatuses>
    <opStatus id="1023" code="1" message="Updating Attributes"/>
</opStatuses>
```

deleteRoomType - DELETE /cm/roomTypes/{roomTypeId}

This method will set the deleted date on the roomType that matches the roomTypeId provided, effectively removing this room from the Crew Manager application at that PIT.

Example - https://qa.rostermonster.com/arcos/rest/cm/roomTypes/1023?pit=2021-12-13T14:34:02-04:00

Request Query Parameters

Parameter	Type	Required	Valid Values	Default	Description
roomTypeId	int	Yes			If given, only hotel room for this lodging will be returned
pit	string	No			If given, only attributes for this point in time will be returned.

Response Payload

Example

```
<opStatus id="1023" code="1" message="OK"/>
```

Extract Data

The Extract endpoint can be used to obtain a collection of available extract reports for a customer. In addition, when provided with a specific extract file name it will decrypt and send that file to the client.
Extract data structures can be found in the Appendix – XSD section.
For the specifications of any Extract files, please contact your ARCOS Representative or Support.

Supported Operations

getExtracts - GET /extracts

This method can be used to obtain a collection of available extract report files for the customer. If no extracts are configured, a 404 error response is expected. The error provided will advise that no extracts are configured

Response Payload

A collection of extract files for the customer, if extracts are enabled.

downloadExtract - GET /extracts/{extract}

This method can be used to download a specific extract. The extract in the path param should be one of the file names returned from the "Get Extracts" operation. If the file is not found, this will return a 404.

Response Payload

The client will download whichever extract they requested.

getConfig - GET /extracts/config

This method returns the current cron configuration for the production of extracts. Extracts are not available for all customers. Speak with an ARCOS Representative if you are interested in extracts.

Response Payload

A representation of the extract configuration. XML example provided below.

```
<extractConfiguration>
   <schedule minute="25" hour="4" dayOfMonth="4" month="*" dayOfWeek="*"
nextRun="06/04/2021 04:25"/>
   <properties>
      <item name="dateRange" value="month"/>
      <item name="includeChanged" value="yes"/>
      <item name="includeManualCC" value="yes"/>
   </properties>
</extractConfiguration>
```

updateConfig - PUT /extracts/config

This method allows adjustment of the cron configuration for the production of extracts. This method accepts the extractConfiguration data structure. Extracts are not available for all customers. Speak with an ARCOS Representative if you are interested in extracts.

Callout Data

The Callout endpoint can be used to obtain detailed data on callout history and to submit uninitiated or initiated callouts for a customer. The WADL (Web Application Description Language) provides a number of examples of

requests that can be run. Some requests have multiple parameters available to them to limit the search or directly find specific callout information. Below are some examples of those requests.
Callout data structures can be found in the Appendix – XSD section.

Attribute Key Descriptions

Key	Description of the Attribute
callout id	The CoMainId for the entire Callout event including all Sub Callouts, Attempts, Sequences, Call Results, and Contacts.
callout initiate	True/False. Sending True will initiate the callout without any user interaction.
calloutType name	The name for this Callout Type. Not Required for creating callouts.
calloutType id	The typeId for this Callout Type.
callout Description	Free-form field for entry of a human-readable description for the purpose of this Callout
location name	The name of the location this Callout was created within. Not Required for creating callouts.
location id	The locationId of the location this Callout was created within.
callout workStartDate	The Datetime the work assignment will begin for this Callout offer. Cannot be in the past.
callout workEndDate	The Datetime the work assignment will end for this Callout offer. Must occur after workStartDate.
creator id	The contactId of the ARCOS user that initially created this Callout. Alternatively, you may substitute this key with “vruId” and pass that ID.
creator name	The name of the ARCOS user that initially created this Callout. Not Required for creating callouts.
callout numberRequired	The number of positions that have been requested.
callout numberFilled	The number of positions that have been filled. Not Required for creating callouts.
callout numberOfCalls	The number of devices that have been called by this Callout. Not Required for creating callouts.
callout numberOfPeopleCalled	The number of employees who have been called by this Callout. Not Required for creating callouts.
callout percentFilled	The percentage of numberRequired / numberFilled. Not Required for creating callouts.
callout percentResonse	The percentage of employees who received calls and responded with either an

	Accept or a Decline. Not Required for creating callouts.
callout elapsedTime	The amount of time that has elapsed since this Callout was initiated. Not Required for creating callouts.
owner id	The contactId of the ARCOS user that currently has ownership of this Callout. Alternatively, you may substitute this key with “vruId” and pass that ID.
owner name	The name of the ARCOS user that currently has ownership of this Callout. Not Required for creating callouts.
attribute value	0 or 1. Defines if an attribute should be active or inactive for this Callout. The available attributes are dependent on system configuration. Not Required.
attribute name	Possible attributes are serialcall, blastcall, etc. The available attributes are dependent on system configuration. Not Required.
overrides value	The Event Name of an included Callout Override. The available overrides are dependent on system configuration. Not Required.
overrides key	The eventId of an included Callout Override. The available overrides are dependent on system configuration. Not Required.
subCallout id	The unique ARCOS key id for each Sub Callout. Not Required for creating callouts.
subCallout quantity	The number of Employees to be included in each Sub Callout.
subCallout status	The Status id of the Sub Callout. Possible values found in the Callout Status table. Not Required for creating callouts.
subCallout statusName	The Status name of the Sub Callout. Possible values found in the Callout Status table. Not Required for creating callouts.
callResult resultName	The Result Name of the call result. Possible values found in the Event Mappings Table. Not Required for creating callouts.
callResult contact id	contactId of the result contact. Not Required for creating callouts.
callResult contact vruId	vruId of the result contact. Not Required for creating callouts.
callResult contact webId	webId of the result contact. Not Required for creating callouts.
callResult contact name	Full name of the result contact. Not Required for creating callouts.

Supported Operations

getCallouts - GET /callout

This method can be used to obtain a collection of historical callouts.

Request Parameters

Parameter	Type	Required	Valid Values	Default	Description
coMainId	Int	yes	coMainId	None	You may supply multiple coMainId by passing this parameter multiple times.
basicData	Bool	No	True, False	False	When true, return only the primitive data about a callout. (Remove callResults)
acceptsOnly	Bool	No	True, False	False	When true, only return callResults that have a final outcome of Accepted.

Response Payload
A collection of historical callout data for the customer filtered by the above parameters.

Example

```
<callout id="1466" initiate="false">
   <calloutType name="System Outage Staffing" id="1235"/>
   <description>Sample Callout for Testing</description>
   <location name="Staff Site" id="9998"/>
   <workStartDate>06/02/2021 08:00:00</workStartDate>
   <workEndDate>06/03/2021 12:00:00</workEndDate>
   <calloutReason name="Customer Outage" id="1276"/>
   <creator id="3632">
      <name>Rigdon, Preston</name>
   </creator>
   <numberRequired>3</numberRequired>
   <numberFilled>1</numberFilled>
   <numberOfCalls>1</numberOfCalls>
   <numberOfPeopleCalled>3</numberOfPeopleCalled>
   <percentFilled>33</percentFilled>
   <percentResponse>66</percentResponse>
   <elapsedTime>00:00:00</elapsedTime>
   <owner id="3632">
      <name>Rigdon, Preston</name>
   </owner>
   <attributes>
      <attribute name="emaildev" value="1"/>
      <attribute name="resubmitdelay" value="15"/>
   </attributes>
   <overrides/>
   <qualifications/>
   <subCallouts/>
</callout>
```

getCallout - GET /callout/{coMainId}

This method can be used to obtain historical callout data for the target Callout Main ID ({coMainId}).

Request Parameters

Parameter	Type	Required	Valid	Default	Description

			Values		
coMainId	Int	yes	coMainId	None	
basicData	Bool	No	True, False	False	When true, return only the primitive data about a callout. (Remove callResults)
acceptsOnly	Bool	No	True, False	False	When true, only return callResults that have a final outcome of Accepted.

Response Payload
A collection of historical callout data for the Callout Main ID, {coMainId}.

Example

```
<callout id="1466" initiate="false">
   <calloutType name="System Outage Staffing" id="1235"/>
   <description>Sample Callout for Testing</description>
   <location name="Staff Site" id="9998"/>
   <workStartDate>06/02/2021 08:00:00</workStartDate>
   <workEndDate>06/03/2021 12:00:00</workEndDate>
   <calloutReason name="Customer Outage" id="1276"/>
   <creator id="3632">
      <name>Rigdon, Preston</name>
   </creator>
   <numberRequired>3</numberRequired>
   <numberFilled>1</numberFilled>
   <numberOfCalls>1</numberOfCalls>
   <numberOfPeopleCalled>3</numberOfPeopleCalled>
   <percentFilled>33</percentFilled>
   <percentResponse>66</percentResponse>
   <elapsedTime>00:00:00</elapsedTime>
   <owner id="3632">
      <name>Rigdon, Preston</name>
   </owner>
   <attributes>
      <attribute name="emaildev" value="1"/>
      <attribute name="resubmitdelay" value="15"/>
   </attributes>
   <overrides/>
   <qualifications/>
   <subCallouts/>
</callout>
```

createCallout - POST /callout

This method can be used to submit, modify, and initiate callouts in the customer system.

Request Payload

Example XML, ignore & remove comments preceded with "//" **(pink text listed above the xml line to clarify the data being processed)**

```
<callout id="0" initiate="true"> // Start the Callout immediately?
```

```
<calloutType name="Normal"/> // ARCOS Callout Type ID. Example: Emergency
<description>Sample Callout for Testing</description>
<location name="McLeod OpCenter"/> // ARCOS Location ID. Example: McLeod OpCenter
<troubleLoc/> // Optional, Trouble Location Vox ID
<workStartDate>2021-10-05T08:00:00-04:00</workStartDate>
<workEndDate>2021-10-05T12:00:00-04:00</workEndDate>
<calloutReason name="All Hands"/>
<creator id="3742"/> // User Contact ID (3742 is an Example)
<attributes>
   <attribute name="serialcall" value="1"/> // Attribute Name (serialcall)
   <attributes name="blastcall" value="1"/> // Value (0 = Off, 1 = On)
<overrides> // Providing any Override enables that Override.
   <overide value="Vacation" key="1017"/> // Override Name (Vacation) Override Key ID (1017)
</overrides>
<subCallouts> // Add specification to a callout such as location, quantity or working class.
   <subCallout quantity="1"> // Number of Responders requested for Callout
      <location name="McLeod OpCenter"/> // ARCOS Location ID. Example: McLeod OpCenter
      <classification name="Sample Test"/> // Roster Name/ID or Job Classification within the location that you wanting
to call
   </subCallout>
</subCallouts>
</callout>
```

Response Payload:
OpStatus response advising if the callout is saved successfully or if errors with the Request Payload are present. In the event the callout is saved successfully, the “id” returned is the CoMain ID. Upon successfully creating or modifying a callout the “code” returned will be “1”. Any code returned other than 1 indicates an error with the Request Payload.
Example

```
<opStatus id="1466" code="1" message="Callout saved successfully."/>
```

updateCallout – PUT /callout/{id}

This method has the same behavior as the createCallout method and is included to support REST Architecture standards.

getAllAttributes - GET /callout/allAttributes

This method can be used to retrieve a collection of all available Callout Attributes.
No parameters are available for this call.

Response Payload
A collection of all available callout attributes. Note, not all attributes displayed may be visible or configured in the customer instance.

```
<calloutAttribute id="1002" name="blastcall" attributeType="Callout" description="Blast Call"
fullDescription="During a callout, instead of calling an employee's devices one at a time, Blast Calling calls all of an emplo
yee's devices simultaneously. ARCOS still calls employees in the proper order according to business rules.Related Securi
ty Features: 121 - Callout - Blast Callout Option, 91 - SIREN - Blast Callout Option, 403 - Mobile - Callout Blast Callout Op
tion, 407 - Mobile - Notification Blast Callout Option."
activeDescription="An Active Input of 1 makes the Blast Calling checkbox visible. Setting the Active Input to 2 enables the
 'enhanced-
killer' functionality. The enhanced killer functionality will terminate all active calls to other devices for this employee when a
 final result is recorded from a device, with a roughly 10 second delay from when the final result is received."
valueDescription="Setting this value to 1 will default the Blast Call attribute to checked."/>
```

getCalloutTypes - GET /callout/types

This method can be used to retrieve a collection of all available Callout Types. The callout type ID(s) will be needed when posting a callout that is generated through the API.

Response Payload

A collection of all available callout types for the customer. Note, not all types displayed may be visible in the customer instance.

Example

```
<calloutType id="1235" name="System Outage Staffing" displayOrder="115">
  <category name="1" value="Regular"/>
  <attributes>
    <attribute id="1012" value="1479"/>
    <attribute id="1004" active="2" value="15"/>
    <attribute id="1018" active="1" value="1"/>
  </attributes>
  <overrides>
    <override value="Automatic">
      <event name="Working - Normal Shift" id="1008"/>
    </override>
  </overrides>
  <templates>1</templates>
  <recording>1</recording>
</calloutType>
```

getCalloutType - GET /callout/types/{id}

This method can be used to retrieve a collection of Callout Type data for the target Callout Type ID ({id}).

Response Payload

A collection of all attributes, templates, and recordings associated with the target Callout Type ID, {id}.

Sample Response -

```
calloutType id="1235" name="System Outage Staffing" displayOrder="115">
  <category name="1" value="Regular"/>
  <attributes>
    <attribute id="1012" value="1479"/>
    <attribute id="1004" active="2" value="15"/>
    <attribute id="1018" active="1" value="1"/>
  </attributes>
  <overrides>
    <override value="Automatic">
      <event name="Working - Normal Shift" id="1008"/>
    </override>
  </overrides>
  <templates>1</templates>
  <recording>1</recording>
</calloutType>
```

getCalloutResultsMultiples - GET /callout/results

This method can be used to retrieve a collection of callout results for a specified Callout Main ID. The Callout Main ID is provided as a URI attribute, CoMainId.

Example URI

https://qa.rostermonster.com/arcos/rest/callout/results?coMainId=1466

Response Payload

A collection of historical response data for the target CoMainId.

```
<calloutResults>
   <calloutResult coMainId="1466"/>
</calloutResults>
```

getCalloutResults - GET /callout/results/{coMainId}

This method can be used to retrieve a collection of callout results for a specified Callout Main ID, {coMainId}.

Response Payload

A collection of historical response data for the target {coMainId}.

```
<callout id="303796" initiate="true">
   <calloutType name="Normal" id="1001"/>
   <description>Normal callout created for testing purposes only</description>
   <location name="McLeod OpCenter" id="9999"/>
   <workStartDate>10/25/2021 14:31:00</workStartDate>
   <workEndDate>10/25/2021 18:31:00</workEndDate>
   <creator id="4076">
      <name>George, Andrew</name>
   </creator>
   <numberRequired>2</numberRequired>
   <numberFilled>2</numberFilled>
   <numberOfCalls>4</numberOfCalls>
   <numberOfPeopleCalled>4</numberOfPeopleCalled>
   <percentFilled>100</percentFilled>
   <percentResponse>100</percentResponse>
   <elapsedTime>00:02:40</elapsedTime>
   <owner id="4076">
      <name>Davis, Carson</name>
   </owner>
   <attributes>
      <attribute name="custommsgtxt" value="Please tell all residents to dispose of any canned goods. Please
work judiciously."/>
   </attributes>
   <overrides>
      <override value="Vacation" key="1017"/>
   </overrides>
   <qualifications/>
   <subCallouts/>
</callout>
```

getCallDetails - GET /callout/callDetails/{callId}

This method can be used to retrieve details on a specific device notification event, such as a phone call. The target Call is provided as {callId}.

Example URI

https://qa.rostermonster.com/arcos/rest/callout/callDetails/1466

Response Payload

A callDetail element containing the details of the target Call, {callId}.

Example

```
<callDetail callId="1466">
   <contactName></contactName>
   <type>VRU-Inbound</type>
   <callStart>05/21/2021 12:44:26</callStart>
   <callEnd>05/21/2021 12:45:38</callEnd>
   <duration>00:01:12</duration>
   <phoneNumber>+11111111111</phoneNumber>
   <host>qa-fsw-nvaaws-a02 : bandwidth1</host>
   <channel>8032aeb1-3b16-4930-ad7a-b9e95db5c231</channel>
   <result></result>
   <callDial></callDial>
   <timeRinging>00:00:00</timeRinging>
   <recordingFound>false</recordingFound>
   <failed>false</failed>
   <recordingURL>https://[...]/arcos/rest/callout/callDetails?co_call_id=1466/download
   </recordingURL>
   <digitsList>
      <digits runningTime="00:00:11" digits="778#"/>
      <digits runningTime="00:00:24" digits="1003008#"/>
      <digits runningTime="00:00:31" digits="0131#"/>
      <digits runningTime="00:00:52" digits="1003008#"/>
      <digits runningTime="00:00:59" digits="0131#"/>
   </digitsList>
   <cloudStorage>
      <cloudStorageInfo>s3|us-east-
1|qas3env|/services/co_call/1466.ulaw|c193cb2c1c700344e4fd54b2821daa380907314960b78f2cf0475d4ae3676
e1c</cloudStorageInfo>
   </cloudStorage>
</callDetail>
```

downloadRecording - GET /callout/callDetails/{callId}/download

This method can be used to retrieve a WAV file recording of the target call, {callId}.

Example URI

https://qa.rostermonster.com/arcos/rest/callout/callDetails/callId/download?wav=1

If the client platform does not support the process of downloading .wav files, this URI can be processed through your browser while logged into ARCOS.

Response Payload

A WAV format recording of the phone call that took place under the target Call ID, {callId}.

ARCOS LLC
8800 Lyra Drive, Suite 200
Columbus, OH 43240
(614) 396 – 5500

getCalloutReasons - GET /callout/reasons

This method can be used to retrieve a collection of all available Callout Reasons.

Response Payload

A collection of all available callout reasons for the customer. Note, not all reasons displayed may be active configured in the customer instance.

Example

```
<reasons>
  <reason id="0" description=" " displayOrder="5"/>
  <reason id="1677" description="12-Hour Schedules Invoked" displayOrder="10"/>
  <reason id="1255" description="911 Call" displayOrder="20"/>
  <reason id="1067" description="Accident" displayOrder="30"/>
  [...]
</reasons>
```

searchCallouts - GET /callout/search

This method provides several URI attributes to search for a specific callout and return data on any callouts found in the search.

Request Parameters

Parameter	Type	Required	Valid Values	Default	Description
start	Datetime	yes	Datetime	none	Records returned will have started or ended at or after this date.
end	Datetime	yes	Datetime	none	Records returned will have started or ended up to and including this time.
locations		no			
classes		no			
calloutTypes	List	no	numbers	none	Limit the data returned but the calloutType id.
searchMode	int	no			
includeOpen	int	no			

Response Payload

A collection of Callouts and all associated callout data for any callouts found matching the URI search parameters.

Example

```
<calloutSearchResults>
   <calloutSearchResult id="97319" createdBy="Rigdon, Preston" description="Fill Shift callout created by Presto
n Rigdon for McLeod OpCenter on Fri 11/06/2020 09:56:04 US/Eastern">
      <calloutType name="Fill Shift" id="1002"/>
      <calloutReason name="12-Hour Schedules Invoked" id="1677"/>
      <location name="McLeod OpCenter" id="9999"/>
      <subCallouts>
         <subCallout name="Sample Test" value="20" id="97450"/>
      </subCallouts>
   </calloutSearchResult>
</calloutSearchResults>
```

getDetailCalloutResults - GET /callout/search/detailResults

This method provides several URI attributes to search for a specific callout and return extended data on any callouts found in the search. This method supports pagination through the use of the ‘page’ parameter, ‘X-Arcos-Page’ header, and ‘X-Arcos-Total-Pages’ header.

If the response payload could contain over 5000 callout records, the use of pagination is required or the request may fail with an error that the response contains over 5000 records and pagination is required.

Request Parameters

Parameter	Type	Required	Valid Values	Default	Description
start	Datetime	yes	Datetime	none	Records returned will have started or ended at or after this date.
end	Datetime	yes	Datetime	none	Records returned will have started or ended up to and including this time. The endTime
locations	int	no	0, 1	0	Show Crew and Member Assignments.
classes	int	no	Number of whole hours	36	Number of hours before and after PIT to search for Member Schedule Records.
calloutTypes	int	no	numbers	none	Limit the data returned but the calloutType id.
searchMode	int	no			
includeOpen	int	no			
onlyOpen	int	no			
basicData	query	no			
acceptsOnly	query	no			
page	int	No, unless number of records exceeds 5000	Number of page	none	Enables and controls pagination. Value is the number of page requested

Response Payload

A collection of Callouts and all associated and extended callout data for any callouts found matching the URI search parameters. An example of the data returned by this method is too large to provide in this document. For the data specification, please see the Appendix – XML Schema Definitions (XSD) section of this manual.

getActivations - GET /callout/activations

This method can be used to retrieve a collection of all available pre-configured Callout Activations. Callout Activations can be initiated with the initiate method.

Request Parameters

Parameter	Type	Required	Valid Values	Default	Description
likeAct	Int	No	acivationId	None	If Activation ID is provided here, return any other Activations that occurred in the same location for the same callout type.
locations	List	No	List of locationId	None	Return only activations that exist in these locations.
level	List	No	1 - 5	None	Return only activations that match these levels.
type	String	No	calloutType name	None	Return only activations that contain a callout of this type.

Response Payload

A collection of Callout Activations that have been created in the customer instance.

Example

```
<calloutActivations>
   <calloutActivation id="1111" description="Standard callout" activationLevel="1" calloutId="1217">
      <calloutType name="Notification" id="1025"/>
      <status name="Closed" id="999"/>
      <location name="McLeod OpCenter" id="9998"/>
      <positions>
         <positions name="Environment/Safety" id="1159"/></positions>
   </calloutActivation>
</calloutActivations>
```

getConfiguredClasses - GET /callout/configuredClasses

This method can be used to retrieve a collection of “Classes” for callout purposes. This includes configured roster Primary Class Groups and Workgroup Rosters.

Request Parameters

Parameter	Type	Required	Valid Values	Default	Description
locationId	Int	Yes	locationId	None	If provided, then only supply Classifications present in this location.
getDisabled	Bool	No	0, 1	0	If 1 then also supply Disabled Classifications.
primaryOnly	Bool	No	0, 1	0	If 1 then only supply Primary Classifications.

Response Payload

A collection of available callout classes for the customer.

Example

```
<coLookups>
   <coLookup id="1151" order="1" enabled="true">
      <classification name="2019 Conference Shell" value="Workgroup" id="5127"/>
      <location name="McLeod OpCenter" id="9999"/>
   </coLookup>
   <coLookup id="1003" order="2" enabled="true">
      <classification name="Arcos User" value="Primary Class" id="500"/>
      <location name="McLeod OpCenter" id="9999"/>
   </coLookup>
   <coLookup id="1341" order="3" enabled="true">
      <classification name="Dana Test" value="Workgroup" id="5283"/>
      <location name="McLeod OpCenter" id="9999"/>
   </coLookup>
</coLookups>
```

getAttributes - GET /callout/attributes

This method can be used to retrieve a collection of all available Callout Attributes for the customer.

Request Parameters

Parameter	Type	Required	Valid Values	Default	Description
attributeType	int	No	0,1,2,3,4	0	Display only Attributes of the provided type.

Response Payload

A collection of all available callout attributes for the customer. Note, not all attributes displayed may be active configured in the customer instance.

Example

```
<calloutAttributes>
   <calloutAttribute id="1133" name="cust_pos_id" description="Customer identifier for position" fullDescription="
Customer identifier for position" valueDescription="remote key"/>
   <calloutAttribute id="1135" name="pos_title" description="Position Title" fullDescription="Position Title" activeD
escription="field label"/>
</calloutAttributes>
```

initiateCallout - POST /callout/{coMainId}/initiate

This method can be used to initiate an already existing callout matching {coMainId} that is ready for initiation.

Response Payload

```
<opStatus id="303779" code="1" message="Callouts initiated successfully"/>
```

applyAction – POST /callout/applyAction

This method can be used to apply various actions to an existing Callout. This includes Sub Callout list "Rollover" actions and stopping an "Active" or "Waiting" callout events.

Request Actions Parameters

Parameter	Type	Required	Valid Values	Default	Description
coMainId	int	Yes		null	The key identifier for the entire Callout event including all Sub Callouts, Attempts, Sequences, Call Results, and Contacts.
note	String				
coId	int				
status	String				Update the status of current callout listed

					by the coMainId
Quantity	int				
rollLoc	String				
rollLocId	int				
rollClass	String				
rollClassId	int				

Request Payload Example
https://qa.rostermonster.com/arcos/rest/callout/applyAction?status=Stop&coMainId=303775

Response Payload

```
<opStatus id="303775" code="1" message="Stop Callout 303775 Success."/>
```

getCalloutTypeLocations – GET /callout/types/{id}/locations

This method returns a collection of <attribute> elements representing the locations this callout type is available to be used within the customer schema.

Two attributes are provided:

Attribute	Datatype	Description
name	String	The long form name of the location as visible throughout the ARCOS platform.
value	Int	The internal ARCOS ID (locationId) for the provided location.

Response Payload
Collection of attributes.

Example

```
<attributes>
  <attribute name="4096" value="Gahanna Contractors"/>
  <attribute name="4097" value="Gahanna Engineering"/>
</attributes>
```

getCalloutStats – GET /callout/stats

This method provides Key Performance Indicator statistics for the provided coMainId(s) and/or ticketId(s).

Request Parameters

Parameter	Type	Required	Valid Values	Default	Description
coMainId	Int	Yes (If no ticketId)	coMainId	None	You may provide multiple coMainId's by passing this parameter multiple times.
ticketId	Int	Yes (If no coMainId)	ticketId	None	You may provide multiple ticketId's by passing this parameter multiple times.

Response Payload

Collection of calloutStats.

Example

```
<calloutStats>
  <calloutStat coMainId="2000">
    <coNumber>1</coNumber>
    <requestedNumber>2</requestedNumber>
    <filledNumber>2</filledNumber>
    <callsNumber>0</callsNumber>
    <peopleNumber>2</peopleNumber>
    <elapsed>0</elapsed>
    <elapsedRunning>0</elapsedRunning>
    <arriveSec>0</arriveSec>
    <arriveAcc>0</arriveAcc>
    <startTime>07/18/2019 10:35:30</startTime>
    <endTime>07/18/2019 10:35:30</endTime>
  </calloutStat>
</calloutStats>
```

updateCalloutStats – POST /callout/stats

This method is published in our Web Application Description language (WADL) but not yet described here. If you require additional information on this method, please speak with an ARCOS Representative.

ARCOS LLC
8800 Lyra Drive, Suite 200
Columbus, OH 43240
(614) 396 – 5500

sMART Data

The smart endpoint can be used to obtain real-time automatic vehicle localization (AVL) data from ARCOS sMART applications associated with the target instance.

getCurrentAVL – GET /smart/getCurrentAVL

This method provides automatic vehicle localization (AVL) data from ARCOS sMART applications associated with the target instance. This method does not support any additional request parameters.

Response Payload Example

Collection of currentAvl.


```
<currentAvl
  xmlns="http://samsix.com/cust/smart/Smart.ext">
    <entry>
      <vehicleid>53470</vehicleid>
      <signaltime>2021-04-16T11:49:38-04:00</signaltime>
      <speed>0</speed>
      <heading/>
      <signal_sequence>2</signal_sequence>
      <status>stopped</status>
      <userid>Android04-16@gmail.com</userid>
      <storm>Test 4/16</storm>
      <platform>Android</platform>
      <platformversion>10</platformversion>
      <lastlogin>2021-04-16T11:49:22-04:00</lastlogin>
      <company>AEP</company>
      <fullname>Android 4-16</fullname>
      <cell>111-222-3366</cell>
      <destination>The Ohio Statehouse</destination>
      <workers>5</workers>
      <numvehicles>5</numvehicles>
      <fuel>Gasoline &amp;amp; Diesel</fuel>
      <drivetime>8</drivetime>
      <eta/>
      <end_dist>2</end_dist>
      <application>rsales</application>
      <crewtype/>
      <longitude>-83.01354891</longitude>
      <latitude>39.97452005000001</latitude>
    </entry>
</currentAvl>
```

currentAvl Entry Definition

```
{
        vehicleid : int
        signaltime :
        xsd:dateTimespeed :
        int
        heading : int
        signal_sequence
        : intstatus : string
        userid : string
        storm : string
        platform : string
        platformversion :
        string lastlogin :
        xsd:dateTime
        company : string
        fullname : string
        cell : string
        destination :
        stringworkers :
        int
        numvehicles :
        int
        fuel : string
        drivetime : int
        eta : string
        end_dist : int
        application :
        stringcrewtype
        : string
        longitude : float
        latitude : float
}
```

currentAvl Entry Attribute Table

Note that Type, Min Length, and Max Length may change for any attribute during future releases of ARCOS; Such changes will be identified in our C.L.I.

Min Length and Max Length provided are the ARCOS software limitations. See Description for additional detail on typical anticipated value ranges.

Attribute	Type	Min Length	Max Length	Description
vehicleid	Int	1	2,147,483,647	The unique numeric ARCOS assigned ID for this Entry 'Vehicle'. The length of this value depends heavily on customer utilization of a particular instance. The value could reasonably be expected to remain between 1000 – 10000000.
signaltime	Datetime	N/A	N/A	The Datetime the last signal was received from this Entry. YYYY-MM-DDThh:mm:ss+hh:mm where '+' could also be '-'. See ISO-8601.

speed	Int	1	N/A	The speed of the entry as calculated by sMART. Defaults to '0'. ARCOS does not constrain the maximum value provided and it's typically expected to be between 0 – 220.
heading	Int	NULL	N/A	If the heading is not available, no value will be provided. The heading of the vehicle as provided by the sMART Mobile application. ARCOS does not constrain the maximum value provided and it's typically expected to be between 0 – 360.
signal_sequence	Int	N/A	N/A	The numeric sequence position used by ARCOS to define the bread crumb trail, mapping to the location values in the order of signal_sequence value received. ARCOS does not constrain the maximum value length provided and it's typically expected to be between 1 – 25000.
status	String	N/A	N/A	The movement status of this vehicle. Possible values are 'stopped' or 'moving'. ARCOS does not constrain the maximum value length provided and it's typically expected to be between 6 - 7 alpha characters.
userid	String	N/A	N/A	The email address of the user logged into the sMART Mobile Application. ARCOS does not constrain the maximum value length provided and it's typically expected to be between 12 - 50 alphanumeric characters including '@' and '.'.
storm	String	N/A	N/A	The name of the storm this vehicle is associated with. (Description of sMART Code) ARCOS does not constrain the maximum value length provided and it may be unexpectedly long depending on user input during storm creation. Recommended customers typically anticipate between 1 – 255 alphanumeric characters.
platform	String	N/A	N/A	The type of device being used by the vehicle. Possible values are 'iOS' or 'Android'. ARCOS does not constrain the maximum value length provided and it's typically expected to be between 3 - 7 alpha characters.
platformversion	String	N/A	N/A	The Operating System version number of the device being used by the vehicle. ARCOS does not constrain the maximum value length provided and it's typically expected to be between 1 - 8 alphanumeric characters. Such as: '13.3', '14.4', '9', '13.3.1', etc.
lastlogin	Datetime	N/A	N/A	The Datetime the last login from this Entry occurred. YYYY-MM-DDThh:mm:ss+hh:mm where '+' could also be '-'. See ISO-8601.

company	String	N/A	N/A	The name of the company this vehicle is associated with. ARCOS does not constrain the maximum value length provided and it may be unexpectedly long depending on user input during storm creation. Recommended customers typically anticipate between 1 – 255 alphanumeric characters.
fullname	String	N/A	N/A	The Full Name of the individual logged in as this vehicle. ARCOS does not constrain the maximum value length provided and it may be unexpectedly long depending on user input during mobile device event registration creation. Recommended customers typically anticipate between 1 – 255 alphanumeric characters.
cell	String	12	12	The Cell Phone Number of the individual logged in as this vehicle. Must be provided by a mobile user during event registration and will always be 12 characters in length, including two '-'. I.e. XXX-XXX-XXXX
destination	String	N/A	N/A	The name of the destination this vehicle is associated with. ('Name' of 'Location') ARCOS does not constrain the maximum value length provided and it may be unexpectedly long depending on user input during destination creation. Recommended customers typically anticipate between 1 – 255 alphanumeric characters.
workers	Int	N/A	N/A	The numeric number of workers reported as part of the entry during mobile user event registration. ARCOS does not constrain the maximum value length provided and it may be unexpectedly long depending on user input during event registration. Typical values are between 1 – 1000.
numvehicles	Int	1	N/A	The numeric number of vehicles reported as part of the entry during mobile user event registration. ARCOS does not constrain the maximum value length provided and it may be unexpectedly long depending on user input during event registration. Typical values are between 1 – 1000.
fuel	String	N/A	N/A	The type of fuel required by the vehicles in this entry. "Gasoline", "Gasoline & Diesel", and "Diesel" are the only currently supported values. Note the API will conduct HTML Code Formatting on values and as such '&' will be seen as '&'. See the HTML Charset.
drivetime	Int	N/A	N/A	The numeric number of hours reported as available for driving per each day as part of the entry during mobile user event registration. ARCOS does not constrain the maximum value length provided and it may be unexpectedly long depending on user input during event

				registration. Typical values are between 1 – 16.
eta	String	1	N/A	The number of hours estimated for vehicle arrival at destination. May be '_' when unknown or a numeric value typically between 0 – 168. ARCOS does not constrain the maximum value length provided.
end_dist	Int	N/A	N/A	The number of miles between the last known location ping of this vehicle entry and the target destination address. ARCOS does not constrain the maximum value length provided. Typically values will be between 0 – 1000.
application	String	N/A	N/A	The schema name, or company name, of the target system the application is authenticated against. ARCOS does not constrain the maximum value length provided, but these values are configured by ARCOS during initial system creation. Typically, values are alphanumeric and between 3 – 7 in length.
crewtype	String	NULL	N/A	This value is optional and may not be included in every Entry. ARCOS does not constrain the maximum value length provided and it's typically expected to be between 0 – 360 characters.
longitude	Float	17	19	Current longitude location of the vehicle as calculated by the sMART Mobile Application. The Longitude displayed contains 14 decimal places of precision. May also be cast as DECIMAL(16,14), for example. Consider casting the longitude and latitude into a Point, where supported.
latitude	Float	17	19	Current latitude location of the vehicle as calculated by the sMART Mobile Application. The latitude displayed contains 15 decimal places of precision. May also be cast as DECIMAL(17,15), for example. Consider casting the longitude and latitude into a Point, where supported.

ARCOS LLC
8800 Lyra Drive, Suite 200
Columbus, OH 43240
(614) 396 – 5500

Appendix

Flat File Specifications

This section contains Flat File specifications to be used with the ARCOS Platform Loader and Loader API methods.

Optional File Naming Standards

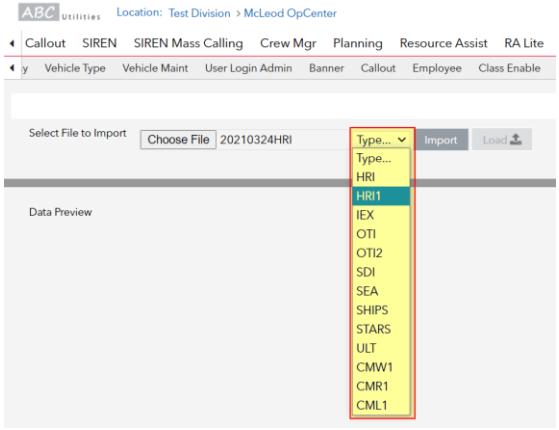
Optionally, all Flat Files may be loaded with the below naming convention and saved as a Pipe Delimited (.dat) file.

CUSTOMER-TYPE-CUSTOMER-YYYYMMDDHHMM.DAT

Replace the following in above file name:

- **CUSTOMER** - replaced with Company Acronym Schema. This is the same Company code used in the URL provided by ARCOS to access the Customer web interface. (e.g. *AUSS*.rostermonster.com, where *AUSS* is the Company Acronym Schema.)
- **TYPE** - replaced with the file type you are attempting to load. This includes HRI, HRI1, OTI, OTI2, and others identified in the Loader section of this document.

This file naming convention automatically populates the file type and confirms the company schema matches the customer ARCOS Platform. When loading the flat file without this naming convention the Type drop down menu will be used to select the file type on the Load Data page of the front-end Web Interface. Load files can be Comma Delimited (.csv), Pipe Delimited (.dat), or Excel 97-2003 Workbook (.xls).



Record Format Standards

All ARCOS Flat File specifications adhere to the following set of standards.

- Field values are variable length. Fixed length data should not be sent in any field.
    - I.e. if a field accepts a maximum length of seven (7) and your data value is four (4) long, the file does not include three (3) additional spaces. This applies to all record types, including all header & data records, when transmitted via API uploadFile method and Web UI Loader tool.
    - The Web UI Loader tool may accept fixed-length data values, but this is not a supported feature
    - Certain legacy and deprecated sFTP services may use fixed length data values, this is no longer supported
    - The API uploadFile method will not accept fixed-length data values
- Delimited by the Pipe Symbol "|" or the Comma Symbol ","
- New Line between records without a Carriage Return
- File must be encoded with UTF-8

If generating test files using Excel spreadsheets, certain Excel formats will drop leading zeros and may also change the encoding of the file from the required UTF-8.

Field Type Key for all support file standards:

Type N(X,X) = Numeric (Max Length, Max Decimal Length, if applicable)

Type X(X) = Alphanumeric and Special Characters (Max Length)

ARCOS LLC
8800 Lyra Drive, Suite 200
Columbus, OH 43240
(614) 396 – 5500

HRI (Human Resources / Employee Data)

The HRI file type allows the loading and modification of Employee data such as Name, Location, Class, and Contact Devices. HRI stands for Human Resources Information. HRI and HRI1 file types contain employee information and are used to add/update employee data.
HRI files only contain 100 and 200 row records while HRI1 contains additional record types as described in the HRI1 File Specification. Additionally, please note the format of 200 records is different between HRI and HRI1 files.

Record Numbers

The only two record types/numbers are 100 (Header) and 200 (Basic Employee Data) records.

Record Number	Data represented in the record(s)
100	Header
200	Basic Employee Data

Record Descriptions

100 – Header/Trailer Record

The Header Record will be used by ARCOS for data integrity checking. ARCOS will make a first pass through the file counting the records and will then verify that the Header / Trailer record count is correct prior to processing the file. If the actual number of records in the file does not match the count stored in the header record, ***the file will not be processed at all*** and an error report will be created.

Field Description and Type	Format and Notes
Record Type – Type N(3)	100 (Header Record) – REQUIRED See description above.
Company Identifier – Type X	[Company Code] ARCOS Customer Company Acronym (Company SCHEMA)
Export Date and Time – Type N(14)	Format MMDDYYYYHHMMSS, Hours (HH) are 24-hour clock. Export date cannot be more than 6.75 days from load date. Seconds are optional.
Overtime Effective Date– Type N(14)	Format MMDDYYYYHHMMSS, The date which OT Hours come into effect. Hours (HH) are 24-hour clock. Seconds are optional.
Export Record Count – Type N(5)	Total records not including header rows

ARCOS LLC
8800 Lyra Drive, Suite 200
Columbus, OH 43240
(614) 396 – 5500

200 - Data Record

Field Description and Type	Format and Notes
Record – Type N(3)	200 Data Record - **REQUIRED**
Employee Number – Type N	This is used to provide the employee's Web ID and VRU ID. It adds an xwalk value for the employee (which functions as the Employee ID) - **REQUIRED**
First Name – Type X(30)	**REQUIRED**
Middle Initial – Type X(1)	
Last Name – Type X(30)	**REQUIRED**
Department Number– Type N	This is used to identify the department the employee is assigned to (this is not directly represented in ARCOS) - **REQUIRED**
Job Class Number – Type X	This is the xwalk value for the employee's primary class - **REQUIRED**
Primary Phone Number – Type N(10)	The phone number with Sequence 1.
Union Seniority Date – Type N(8)	Format MMDDYYYY
Service Date – Type N(8)	Format MMDDYYYY
Reporting Location– Type X	This is used to identify the reporting location the employee is assigned to (this is not directly represented in ARCOS) - **REQUIRED**
Overtime Hours – Type N(10,2)	IF USED – would contain the number of overtime hours which employee has. Hours can be entered with a whole number (e.g. 350) or with two decimal places (e.g. 154.74)
Storm Code – Type X(9)	IF USED – would identify the unique ARCOS Event 'Storm Code' to associate this employee record to
Storm Company – Type X(30)	IF USED – would identify the name of the organization the employee resource reports to

The Department Number and Report Location do not actually correspond to any locations in ARCOS directly. Employees are assigned to locations using a combination of these values and Job Class Number. When ARCOS attempts to place the employee in a location, it will attempt to match a location's xwalk value to the string [Reporting Location]-[Department Number]-[Job Class Number].
There is no firm limit on length of department numbers and reporting locations. Xwalk values are limited to 50 characters and the format of the xwalk which ARCOS attempts to match the multi-part xwalk to contains two hyphens, meaning that the department number, job class number, and reporting location cannot exceed a combined length of 48 characters.
During an HRI update to an employee that already exists - If an optional field is left blank or null, ARCOS will not overwrite any data in that field in the ARCOS database with the blank/null.

Ref	Field Description and Format	Notes
1.	Record Type – Type N (3)	100 (Header Record) – **REQUIRED** **See Note above.**
2.	Company Identifier - Type X Process	***CUSTOMER*** Company Acronym
3.	Export Date and Time - Type N(12)	Format MMDDYYYYHHMM *Note: Hours (HH) are 24 hour clock. Export date cannot be more than 6.75 days from load date.*
4.	Export Record Count - Type N	Total records not including headers

	A	B	C	D	E
1					
2	100	AUSS	062120211410	6	
3	200	228626	228626	0001	Bruce
4	200	76246	76246	0002	Dick
5	200	78737626	78737626	1000	Clark
6	200	56847	56847	1001	Louis
7	200	774337	774337	0003	Peter
8	200	928766	928766	0004	Mary-Jane
9					

Wayne				1001
Grayson				1002
Kent				1001
Lane				1002
Parker				1002
Watson				1013

Above: Example diagram of mapping HRI fields into the CSV file format.

ARCOS LLC
8800 Lyra Drive, Suite 200
Columbus, OH 43240
(614) 396 – 5500

HRI1 (Human Resources / Employee Data, Extended)

The HRI file type allows the loading and modification of Employee data such as Name, Location, Class, and Contact Devices. HRI stands for Human Resources Information. HRI and HRI1 file types contain employee information and are used to add/update employee data.
HRI files only contain 100 and 200 row records while HRI1 contains additional record types as described below. Additionally, please note the format of 200 records is different between HRI and HRI1 files.

Record Numbers

The only two required record types/numbers are 100 (Header) and 200 (Basic Employee Data) records. Other record numbers may be used when additional data is sent to ARCOS.

Record Number	Data represented in the record(s)
100	Header
200	Basic Employee Data
201	Employee Email Addresses Data
202	Telephone Device Data
203	Pager Device Data
204	Transfer-To-Device Data
205	Extended Employee Attributes

Record Descriptions

In the Notes section, each *field is identified as REQUIRED or N/A. N/A fields must be included in the file structure as a blank column. Optional fields are assumed when not indicated as REQUIRED or N/A.*

100 – Header/Trailer Record

The Header Record will be used by ARCOS for data integrity checking. ARCOS will make a first pass through the file counting the records and will then verify that the Header / Trailer record count is correct prior to processing the file. If the actual number of records in the file does not match the count stored in the header record, ***the file will not be processed at all*** and an error report will be created.

Ref	Field Description and Type	Format and Notes
1.	Record Type – Type N(3)	100 (Header Record) – **REQUIRED** **See description above**
2.	Company Identifier – Type X	ARCOS Company Acronym (Company SCHEMA) – **REQUIRED**
3.	Export Date and Time – Type N(14)	Format MMDDYYYYHHMMSS *(Seconds are optional)Hours (HH) are 24-hour clock. Export date cannot be more than 6.75 days from load date.* – **REQUIRED**
4.	Export Record Count – Type N	Total records not including 100 header/trailer record – **REQUIRED**

200 - Data Record

Ref	Field Description and Type	Format and Notes
1.	Record – Type N(3)	200 Data Record - **REQUIRED**
2.	Emp_Id – Type X(11)	**REQUIRED**
3.	WEB Login ID – Type X(40)	**REQUIRED**
4.	VRU-ID – Type N(15)	**REQUIRED**
5.	First Name – Type X(30)	**REQUIRED**
6.	Middle Initial – Type X(1)	
7.	Last Name – Type X(30)	**REQUIRED**
8.	Nick Name – Type X(30)	
9.	Union Seniority Date – Type N(8)	Format MMDDYYYY
10.	Service Date – Type N(8)	Format MMDDYYYY
11.	Job Classification Code – Type X(50)	Xwalk job class ID – **REQUIRED**
12.	Street 1 Address – Type X(55)	*****Warning: the address information included here will be automatically overwritten by the load. This update will also cause a "look up" of Lat/Long data for the address, if your company has Closest-To-Trouble ('CTT') Add-On.**
13.	Street 2 Address – Type X(55)	*****See HRI/CTT warning above**
14.	City – Type X(30)	*****See HRI/CTT warning above**
15.	State – Type X(6)	*****See HRI/CTT warning above**
16.	Postal Zip Code – Type X(12)	*****See HRI/CTT warning above**
17.	Location Code – Type X(10)	Location Name or XWalk ID - **REQUIRED**
18.	Phone #1 number – Type N(10)	Extra characters and spaces in the phone numbers do NOT cause a problem. ARCOS will strip all of them out during the load. CUSTOMER should send phone numbers ***sequentially***. Example: If an employee has two phone numbers DO NOT send phone 1 in field 18 and send phone two in field 22 (Do not skip fields 20 and 21). You cannot populate phone 3 and not phone 1 and 2. During the load, if there is not a phone 1 it will skip the phone section all together
19.	Phone #1 type – Type X(3)	R=Regular Phone; C=Cell Phone; P=Pager; When Duty Phone functionality is turned on: R-D=Regular/Duty Phone; C-D=Cell/Duty Phone; P-D=Pager/Duty. Hyphen separates device abbreviation from Duty Flag. Duty phone is option and "-"(Field) is only used when indicating a duty phone flag. Additionally, Phone Type can be customer defined in Device Types Admin Page

Ref	Field Description and Type	Format and Notes
20.	Phone #2 number – Type N(10)	Extra characters and spaces in the phone numbers do NOT cause a problem. ARCOS will strip all of them out during the load.
21.	Phone #2 type – Type X(3)	R=Regular Phone; C=Cell Phone; P=Pager; When Duty Phone functionality is turned on: R-D=Regular/Duty Phone; C-D=Cell/Duty Phone; P-D=Pager/Duty. Hyphen separates device abbreviation from Duty Flag. Duty phone is option and "-"(Field) is only used when indicating a duty phone flag. Additionally, Phone Type can be customer defined in Device Types Admin Page
22.	Phone #3 number – Type N(10)	
23.	Phone #3 type – Type X(3)	R=Regular Phone; C=Cell Phone; P=Pager; When Duty Phone functionality is turned on: R-D=Regular/Duty Phone; C-D=Cell/Duty Phone; P-D=Pager/Duty. Hyphen separates device abbreviation from Duty Flag. Duty phone is option and "-"(Field) is only used when indicating a duty phone flag. Additionally, Phone Type can be customer defined in Device Types Admin Page
24.	YTD OT Hours – Type N(10,2)	Hours can be entered with a whole number (e.g. 350) or with two decimal places (e.g. 154.74)
25.	Effective Date of Hours – Type N(14)	Format MMDDYYYYHH24MMSS (Seconds are optional) Even though each '200' record will have an effective date of hours, ARCOS will only read the first date encountered. ARCOS will change all effective date of hours fields for all subsequent records to the first date encountered in the HRI file. A single effective date is recommended for all '200' records.
26.	Birthday – Type X(8)	Format MMDDYYYY
27.	Employee Status – Type X(1)	A = Active, I = Inactive, D = Deleted **First time employee record load will default to Active status. Null value on existing employee records will not change current status setting.**
28.	Schedule "code" (n/a)	
29.	Primary Email – Type X(100)	**If more than one email address is to be provided via HRI1+ Load, use 201 type Records.**

Ref	Field Description and Type	Format and Notes
30.	Notify Only – Type X(1)	Acceptable input: null, 0, 1 NULL input (blank) or 0 = NOT Notify Only 1 = Notify Only Any input deviating from the above will cause the loading of the contact record to fail.
31.	Radio – Type X(100)	
32.	Vacation Days – Hrs Entitled – Type N(10,2)	IF USED – Requires the number of vacation hours to which employee is entitled for the period – for use with Vacation Management Add-On. **All Hrs Entitled and Hrs Used Vac Mgt fields must be populated with a numeric value. Zero (0.00) is REQUIRED as a valid numeric value and will overwrite existing entry in the ARCOS Employee Vacation Management Page.**
33.	Personal Days – Hrs Entitled – Type N(10,2)	IF USED – Requires the number of personal-day hours to which employee is entitled for the period – for use with Vacation Management Add-On. **All Hrs Entitled and Hrs Used Vac Mgt fields must be populated with a numeric value. Zero (0.00) is REQUIRED as a valid numeric value and will overwrite existing entry in the ARCOS Employee Vacation Management Page.**
34.	Vacation Days – Hrs Used – Type N(10,2)	IF USED – Requires the number of vacation hours which employee has used for the period – for use with Vacation Management Add-On. **All Hrs Entitled and Hrs Used Vac Mgt fields must be populated with a numeric value. Zero (0.00) is REQUIRED as a valid numeric value and will overwrite existing entry in the ARCOS Employee Vacation Management Page.**
35.	Personal Days – Hrs Used – Type N(10,2)	IF USED – Requires the number of personal-day hours which employee has used for the period – for use with Vacation Management Add-On. **All Hrs Entitled and Hrs Used Vac Mgt fields must be populated with a numeric value. Zero (0.00) is REQUIRED as a valid numeric value and will overwrite existing entry in the ARCOS Employee Vacation Management Page.**
36.	Vacation Mgt – Effective Date – Type N(8)	IF USED – would contain the effective date of any hours loaded to ARCOS for use with Vacation Management Add-On. Format: MMDDYYYY
37.	Phone #4 – Type X(10)	IF USED – would identify a 4th employee phone number
38.	Phone #4 Type – Type X(1)	IF USED – would identify the device type of the employee's 4th device
39.	Phone #5 – Type X(10)	IF USED – would identify a 5th employee phone number

Ref	Field Description and Type	Format and Notes
40.	Phone #5 Type – Type X(1)	IF USED – would identify the device type of the employee's 5th device
41.	Storm Code – Type X(9)	IF USED – would identify the unique ARCOS Event 'Storm Code' to associate this employee record
42.	Storm Company – Type X(30)	IF USED – would identify the name of the organization the employee resource reports

ADDITIONAL NOTES

During an HRI1 update to an employee that already exists - If an optional field is left blank or null, ARCOS will not overwrite any data in that field in the ARCOS database with the blank/null. That is, ARCOS will not blank out data based on the HRI1 load.

Please consider if some fields need to be "Protected" from further HRI1 updates. This requirement should be discussed with your ARCOS Implementation Manager or Professional Services Consultant, who will further review the request with ARCOS Engineering team. *Not all fields are eligible for protection.*

201 - Employee Email Addresses Data

If employee record has only 1 email address with no qualifiers (additional fields being sent), there is no need to send 201-level records; field 29 in 200-record above can be used.

Ref	Field Description and Type	Format and Notes
1.	Record Type – Type N(3)	201 Email Address Record - **REQUIRED** **See description above.**
2.	VRU-ID – Type N(11)	**REQUIRED – Numeric with** NO Alphabetic characters
3.	Email – Type X(100)	The email address provided in this field must be in the format name@something.com or .net, otherwise ARCOS will not load it.
4.	Display Order – Type N(4)	Identifies the order to display the email address. The only options for Display Order is, 1, 2, or 3.
5.	Email – Description – Type X(100)	
6.	Email – Enabled? – Type X(1)	1 = yes, 0/null = no
7.	Email – Condensed? – Type X(1)	1 = yes, 0/null = no

*** Repeat record for additional email addresses.

202 - Additional Telephone Device Data Record

This record should be used only if additional device information – beyond phone number and device type - is needed.

Ref	Field Description and Type	Format and Notes
1.	Record Type – Type N(3)	202 Phone Device Record - **REQUIRED** **See description above.**
2.	VRU-ID – Type N(11)	**REQUIRED –** Non-numeric characters will not validate.
3.	Phone Number – Type N(10)	extra characters and spaces in the phone numbers do NOT cause a problem. ARCOS will strip all of them out during the load.
4.	Display Order – Type N(4)	This is the device display order in the employee record to (e.g. Phone 1, Phone 3, Phone 4)
5.	PIN Req – Type X(1)	1 = yes, 0/null = no
6.	Call Order Seq – Type X(1)	Valid values include 1,2, etc. up to maximum number of devices allowed, and S for Standby. Order of devices in the file determines the display order, but the information in this field will set the sequence (in which ARCOS would call them). Sequence CAN be null indicating that a device is NOT active. If duplicate sequence #s occur in the file, only the later-added device will get the sequence number. The end result will be that only one device may have sequence 1, for example.
7.	Phone On Duty? – Type N (1)	1 = yes, 0/null = no
8.	Phone Trusted Device? – Type N(1)	1 = yes, 0/null = no NOTE: ARCOS will validate that the number here has NOT already been identified as “Trusted” by another employee. If the number has been “Trusted” by another employee, ARCOS will NOT load it here.
9.	Phone Type – Type X (1)	R=Regular Phone, C=Cell Phone, P=Pager, or customer defined in Device Types Admin Page
10.	Phone – Pause first – Type N(1)	(Delay 1)
11.	Phone – Touch Tone first – Type N(10)	(Touchtone 1)
12.	Phone – Pause second – Type N(1)	(Delay 2)
13.	Phone – Touch Tone second – Type N(10)	(Touchtone 2)
14.	Phone – Comments – Type X(1000)	(Device Comments)
15.	SMS Enabled – Type N(1)	1 = enabled, 0 = disabled

***Repeat this record for additional devices.

203 - Pager Device Data Record

This record should be used only if additional pager device information – beyond number and device type - is needed.

Ref	Field Description and Type	Format and Notes
1.	Record Type – Type N(3)	203 Pager Device Data Record – **REQUIRED - See description above.**
2.	VRU-ID – Type N(11)	**REQUIRED –** Non-numeric characters will not validate.
3.	Pager Number – Type N(10)	
4.	Display Order – Type (4)	This is NOT Call Order Seq. This is the device display order in the employee record (e.g. Phone 1, Phone 3, Phone 4)
5.	Sequence – Type N(1)	Call Order Seq number set by customer to identify order phone numbers are called in a Callout.
6.	Pager On Duty? – Type N(1)	1 = yes, 0/null = no
7.	Pager Type – Type X(10)	Must exactly match a pager type defined for the customer system
8.	Pager PIN – Type N(4)	Expected numeric PIN, if required by pager

***One record per pager device.

204 - Transfer Number Data Record

This record should be used only if transfer device is used.

Ref	Field Description and Type	Format and Notes
1.	Record Type – Type N(3)	204 Transfer-To Number - **REQUIRED See description above.**
2.	VRU-ID – Type N(11)	**REQUIRED –** Non-numeric characters will not validate.
3.	Transfer to Phone Number – Type N(10)	
4.	Transfer to Phone Type – Type X(1)	
5.	Transfer to Phone Pause first – Type N(3)	
6.	Transfer to Phone Touch Tone first – Type N(10)	
7.	Transfer to Phone Pause second – Type N(3)	
8.	Transfer to Phone Touch Tone second – Type N(10)	
9.	Transfer to Phone Comments – Type X(1000)	

***One record per Transfer-To device.

ARCOS LLC
8800 Lyra Drive, Suite 200
Columbus, OH 43240
(614) 396 – 5500

205 - Extended Employee Attributes Data

The Name field, below, must be an exact match to the Name of the attribute in the ARCOS System, otherwise, ARCOS may create a new Attribute.

Ref	Field Description and Type	Format and Notes
1.	Record Type – Type N(3)	205 Extended Employee Attribute Record - **REQUIRED** **See description above.**
2.	VRU-ID – Type N(11)	**REQUIRED –** NO Alphabetic characters
3.	Name of Qualification or Attribute – Type X(50)	**Required; may come from the HR system**
4.	Attr_Type – Type X(20)	**Valid values: TEXT, VALUES, MULTIVALUES, NUMBER, DATE, TIME, CHECKBOX (these titles must be capitalized as in the examples above)**
5.	Attr_Category – Type X(10)	**Valid values: Std, Quals, Storm Role, etc.**
6.	VALUE of the attribute – Type X(100)	**The data value of the attribute named in (3) above. If the value is Yes or No only use Y or N.**

***Use multiple records to provide ARCOS with multiple attributes for an employee.

ARCOS LLC
8800 Lyra Drive, Suite 200
Columbus, OH 43240
(614) 396 – 5500

SEA (System Emergency Attributes Data)

SEA stands for System Emergency Attributes. This file type is used to update employee extended attributes.

Record Descriptions

100 – Header/Trailer Record

The Header Record will be used by ARCOS for data integrity checking. ARCOS will make a first pass through the file counting the records and will then verify that the Header / Trailer record count is correct prior to processing the file. If the actual number of records in the file does not match the count stored in the header record, ***the file will not be processed at all*** and an error report will be created.

Ref	Field Description and Type	Format and Notes
1.	Record Type – Type N(3)	100 (Header Record) – **REQUIRED** **See description above.**
2.	Company Identifier – Type X(10)	[Customer Code] ARCOS Customer Company Acronym (Company SCHEMA) – **REQUIRED**
3.	Export Date and Time – Type N(14)	Format MMDDYYYYHHMMSS (Seconds are optional) Hours (HH) are 24 hour clock. Export date cannot be more than 6.75 days from load date. Seconds are optional." in the 100 – Header/Trailer Record table field description and format sections
4.	Export Record Count – Type N	Total records not including headers

200 - Data Record

Ref	Field Description and Type	Format and Notes
1.	Record Type – Type N	**REQUIRED: 200**
2.	Employee ID – Type X	**REQUIRED**
3.	Employee First Name – Type X	**REQUIRED**
4.	Employee Last Name – Type X	**REQUIRED**
5.	SOS Target Device – Phone Number Home, Type N	
6.	SOS Target Device – Phone Number Work, Type N	
7.	SOS Target Device – Phone # Work Cell, Type N	
8.	SOS Target Device – Phone # Personal Cell, Type N	
9.	Email 1 Type X	
10.	Email 2 Type X	
11.	Email 3 Type X	
12.	Extended Attribute – Type X	**REQUIRED**: Column 12 must be the first extended attribute. If not target devices are used, those

Ref	Field Description and Type	Format and Notes
		columns will be blank and the first extended attribute will still be listed here. The data listed here MUST correspond with the first attribute listed in the Extended Attribute list in system admin.
13.	Extended Attribute – Type X	**Optional**: The data here MUST correspond to the second attribute listed in the Extended Attribute list in system admin.
14.	Extended Attribute – Type X	**Optional**: The data here MUST correspond to the third attribute listed in the Extended Attribute list in system admin.
15.	Extended Attribute – Type X	**Optional**: The data here MUST correspond to the fourth attribute listed in the Extended Attribute list in system admin.
16.	Extended Attribute – Type X	To load additional extended attributes, follow the model as outlined above.

ADDITIONAL NOTES:

There is no limit to the number of extended attributes that can be loaded.

Each new attribute column added file, must correspond to the order in which it is listed in the Extended Attribute list in system admin

The 100 record can be listed at the top or bottom. ARCOS will look for the 100 record type as the first or last record and process it correctly. It cannot be mixed within the 200 record rows.

If employees listed in the file do not exist in the ARCOS database, they will be added.

If the email is set to condensed mode, the extended attribute load removed the check box for condensed mode. The email will not receive the standard template. (The email is not disabled)

If an employee record has an email address that is not active, it will be set as active by the loader.

ARCOS LLC
8800 Lyra Drive, Suite 200
Columbus, OH 43240
(614) 396 – 5500

STARS (Employee Data, OT)

This file type is used by limited customers and is not recommended for implementation in any new instances. This file contains Employee Overtime data and is used to update overtime accruals.

Record Descriptions

100 – Header/Trailer Record

The Header Record will be used by ARCOS for data integrity checking. ARCOS will make a first pass through the file counting the records and will then verify that the Header / Trailer record count is correct prior to processing the file. If the actual number of records in the file does not match the count stored in the header record, ***the file will not be processed at all*** and an error report will be created.

Ref	Field Description and Type	Format and Notes
1.	Record Type – Type N	100 (Header Record) – **REQUIRED** **See description above.**
2.	Company Identifier – Type X	[Customer Code] ARCOS Customer Company Acronym (Company SCHEMA) – **REQUIRED**
3.	Export Date and Time – Type N(14)	Format MMDDYYYYHHMMSS (Seconds are optional) Hours (HH) are 24 hour clock.
4.	ARCOS Effective Date – Type N(14)	Format: MMDDYYYYHHMMSS (Seconds are optional) (24 Hr Clock in Eastern Time) *. Date / Time for hours to take effect in ARCOS
5.	Export Record Count – Type N(5)	Total records not including headers
6.	Pay Period Number – Type N(2)	This will vary from 1 to 26 during the year. When pay period 1 is received, all adjusted hours will be reset to 0.

200 - Data Record

Ref	Field Description and Type	Format and Notes
1.	Record Type – Type N(3)	**REQUIRED: 200**
2.	Employee ID – Type X(11)	**REQUIRED**
3.	Pay Code (Earnings Type) – Type X(3)	**REQUIRED** – Varies per company and OpCo
4.	Year to Date Hours – Type N(6)	**REQUIRED** Format 9999.99

ADDITIONAL NOTES: The 100 record can be listed at the top or bottom. ARCOS will look for the 100 record type as the first or last record and process it correctly. It cannot be mixed within the 200 record rows.

ARCOS LLC
8800 Lyra Drive, Suite 200
Columbus, OH 43240
(614) 396 – 5500

SHIPS (Employee Data, Demographics / Users)

This file type is used by limited customers and is not recommended for implementation in any new instances. This file contains Employee Overtime data and is used to update overtime accruals.

Record Descriptions

100 – Header/Trailer Record

The Header Record will be used by ARCOS for data integrity checking. ARCOS will make a first pass through the file counting the records and will then verify that the Header / Trailer record count is correct prior to processing the file. If the actual number of records in the file does not match the count stored in the header record, ***the file will not be processed at all*** and an error report will be created.

Ref	Field Description and Type	Format and Notes
1.	Record Type – Type N	100 (Header Record) – **REQUIRED** **See description above.**
2.	Company Identifier – Type X	[Customer Code] ARCOS Customer Company Acronym (Company SCHEMA) – **REQUIRED**
3.	Export Date and Time – Type N(14)	Format MMDDYYYYHHMMSS (Seconds are optional) Hours (HH) are 24 hour clock. – **REQUIRED**
4.	ARCOS Effective Date – Date / Time for hours to take effect in ARCOS – Type N(14)	**NOT USED** – Send a blank field to conform to the specification
5.	Export Record Count – Type N(5)	Total records not including headers – **REQUIRED**
6.	Pay Period Number – Type N(2)	This will vary from 1 to 26 during the year. When pay period 1 is received, all adjusted hours will be reset to 0. – **REQUIRED**

200 - Data Record

Ref	Field Description and Type	Format and Notes
1.	Record Type – Type N(3)	200 – **REQUIRED**
2.	Employee ID – Type X(11)	**REQUIRED**
3.	Emp_Net_Id – Type X(15)	**REQUIRED** – Referred to as NT ID
4.	Social Security Number – Type N(9)	Numeric No Dashes
5.	First Name – Type X(30)	**REQUIRED** – Names are stored in SHIPS in the format Last name Comma Space First name space middle name E.G. Smith, Tom Will need to parse SHIPS name into separate fields.
6.	Middle Initial – Type X(1)	
7.	Last Name – Type X(30)	**REQUIRED**
8.	Nick Name – Type X(10)	Nick Name X(10)
9.	Union Seniority Date – Type N(8)	Format MMDDYYYY

Ref	Field Description and Type	Format and Notes
10.	Service Date – Type N(8)	Format MMDDYYYY
11.	Job Classification Code – Type X(6)	**REQUIRED**
12.	Street 1 Address – Type X(55)	
13.	Street 2 Address – Type X(55)	
14.	City – Type X(30)	
15.	State – Type X(6)	
16.	Postal Zip Code – Type X(12)	
17.	Location Code – Type X(10)	Only the first 7 characters will be used until we upgrade SHIPS. Then it is a possibility that all 10 characters can be used. – **REQUIRED**
18.	Birth Date – Type N(8)	Format MMDDYYYY

ADDITIONAL NOTES: The 100 record can be listed at the top or bottom. ARCOS will look for the 100 record type as the first or last record and process it correctly. It cannot be mixed within the 200 record rows.

ARCOS LLC
8800 Lyra Drive, Suite 200
Columbus, OH 43240
(614) 396 – 5500

OTI (Overtime, YTD Hours)

OTI stands for Overtime Interface. OTI and OTI2 file types add and update overtime hours for employees. These file types are used to indicate that ARCOS would treat the hours in the file as a Year to Date (YTD) vs Daily hours, respectively.

ARCOS will calculate the number of OT hours which are effective by the effective date of the hours in the file.

Record Descriptions

100 – Header/Trailer Record

The Header Record will be used by ARCOS for data integrity checking. ARCOS will make a first pass through the file counting the records and will then verify that the Header / Trailer record count is correct prior to processing the file. If the actual number of records in the file does not match the count stored in the header record, ***the file will not be processed at all*** and an error report will be created.

Ref	Field Description and Type	Format and Notes
1.	Record – Type N(3)	100 (Header Record) See description above. – **REQUIRED**
2.	Company Identifier – Type X(3)	[Customer Code] ARCOS Customer Company Acronym (Company SCHEMA) – **REQUIRED**
3.	Export Date and Time – Type N(14)	Format MMDDYYYYHHMMSS (Seconds are optional) *Hours (HH) are 24 hour clock. Export date cannot be more than 6.75 days from load date.* – **REQUIRED**
4.	ARCOS Effective Date – Date / Time for hours to take effect in ARCOS – Type N(14)	Format: MMDDYYYYHHMMSS (Seconds are optional) (24 Hr Clock) – **REQUIRED**
5.	Export Record Count – Type N(5)	**REQUIRED** **DOES NOT INCLUDE HEADER ROW IN COUNT TOTAL**

200 - Data Record

Ref	Field Description and Type	Format and Notes
1.	Record – Type X(3)	200 Data Record – **REQUIRED**
2.	Emp_Id – Type X(11)	Usually 5 in length. Alpha Numeric. – **REQUIRED**
3.	Pay Code – Type X(can be of any length)	Typically ‘CUM HRS’ or ‘HRS Worked’ and ‘HRS Refused’ (with space in between words). If you will be using multiple paycodes please speak with your ARCOS Professional Services contact to verify. – **REQUIRED**
4.	Year to Date Hours – Type (10,2)	Summation of hours by earnings type. Format 9999.99 – **REQUIRED**

ARCOS LLC
8800 Lyra Drive, Suite 200
Columbus, OH 43240
(614) 396 – 5500

OTI2 (Overtime, Daily Hours)

OTI stands for Overtime Interface. OTI and OTI2 file types add and update overtime hours for employees. These file types are used to indicate that ARCOS would treat the hours in the file as a Year to Date (YTD) vs Daily hours, respectively.

ARCOS will calculate the number of OT hours which are effective by the effective date of the hours in the file.

Record Descriptions

100 – Header/Trailer Record

The Header Record will be used by ARCOS for data integrity checking. ARCOS will make a first pass through the file counting the records and will then verify that the Header / Trailer record count is correct prior to processing the file. If the actual number of records in the file does not match the count stored in the header record, ***the file will not be processed at all*** and an error report will be created.

Ref	Field Description and Type	Format and Notes
1.	Record Type – Type N(3)	100 (Header Record) See description above. – **REQUIRED**
2.	Company Identifier – Type X(3)	[Customer Code] ARCOS Customer Company Acronym (Company SCHEMA) – **REQUIRED**
3.	Export Date and Time – Type N(14)	Format MMDDYYYYHHMMSS (Seconds are optional) *Hours (HH) are 24 hour clock. Export date cannot be more than 6.75 days from load date.* – **REQUIRED**
4.	ARCOS Effective Date – Date / Time for hours to take effect in ARCOS – Type N(14)	Format: MMDDYYYYHHMMSS (Seconds are optional) (24 Hr Clock) – **REQUIRED**
5.	Export Record Count – Type N(5)	**REQUIRED** **DOES NOT INCLUDE HEADER ROW IN COUNT TOTAL**

201 - Data Record (YTD OT Values)

Ref	Field Description and Type	Format and Notes
1.	Record Type – Type X(3)	201 Data Record – **REQUIRED**
2.	Emp_Id – Type X(11)	Usually 5 in length. Alpha Numeric. – **REQUIRED**
3.	Pay Code – Type X(can be of any length)	Typically 'CUM HRS' or 'HRS Worked' and 'HRS Refused' (with space in between words). If you will be using multiple paycodes please speak with your ARCOS Professional Services contact to verify. – **REQUIRED**
4.	Daily Hours – Type N(10,2)	Hours for day for paycode. Format 9999.99 – **REQUIRED**
5.	Date of Daily Hours – Type N(8)	Format: MMDDYYYY – **REQUIRED**

202 - Data Record (YTD OT Values)

Ref	Field Description and Type	Format and Notes
1.	Record Type – Type X(3)	202 Data Record – **REQUIRED**
2.	Emp_Id – Type X(11)	Usually 5 in length. Alpha Numeric. – **REQUIRED**
3.	Pay Code – Type X(can be of any length)	Typically 'CUM HRS' or 'HRS Worked' and 'HRS Refused' (with space in between words). If you will be using multiple paycodes please speak with your ARCOS Professional Services contact to verify. – **REQUIRED**
4.	Year to Date Hours – Type (10,2)	Summation of hours by earnings type. Format 9999.99 – **REQUIRED**

ARCOS LLC
8800 Lyra Drive, Suite 200
Columbus, OH 43240
(614) 396 – 5500

SDI (Schedule Data)

Record Descriptions

100 – Header/Trailer Record

The Header Record will be used by ARCOS for data integrity checking. ARCOS will make a first pass through the file counting the records and will then verify that the Header / Trailer record count is correct prior to processing the file. If the actual number of records in the file does not match the count stored in the header record, ***the file will not be processed at all*** and an error report will be created.

Ref	Field Description and Type	Format and Notes
1.	Record – Type N(3)	100 (Header Record) See description above. – **REQUIRED**
2.	Company Identifier – Type X(3)	[Customer Code] ARCOS Customer Company Acronym (Company SCHEMA) – **REQUIRED**
3.	Export Date and Time – Type X(14)	Format MMDDYYYYHHMMSS (Seconds are optional) Hours (HH) are 24 hour clock. – **REQUIRED**
4.	Effective Date – Type X(8)	Format MMDDYYYY – **REQUIRED**
5.	Export Record Count – Type N(5)	Count of type 200 records – **REQUIRED**

ADDITIONAL NOTES:

Any Field not indicated as REQUIRED is be Optional.

Fields marked as “N/A” or ‘IF USED’ are either features currently not used by the customer or fields that have no bearing on this project. These fields should be left null in the load file. If data exists in these fields in the load file, it will be ignored during the load process.

At the start of each data load, ARCOS schedule data for the 14-day period starting with “Effective Date” will be purged from the ARCOS system, EXCEPT for any “protected” events. Speak with your ARCOS Representative if you need further information on your event configuration.

ARCOS will find records in the window based strictly on the START date/time of the record. I.e.: If an existing record starts yesterday and crosses midnight into today (or beyond), it will NOT be purged during the load.

ARCOS will ignore (and report as an error) any record with a start date prior to the effective date, or beyond the 14-day window end date.

For normal shift records – the TreatAs date in ARCOS will be determined using the Start Date/Time of the schedule record (only matters for shift records crossing midnight).

Field 6 of type ‘200’ rows will be stored in ARCOS as the Shift ReferAs value for the shift definition on the ARCOS Scheduler which will allow for visual representation of normal weekly shift assignments for employees. Only one value per week (Monday – Sunday) can be stored. If different shift names are sent in the ULT file for an employee that occur within the same effective week, errors will be logged and a ReferAs value will not be populated.

If a shift recorded is loaded in ARCOS by the ULT file and it lands on a day defined as a Holiday in ARCOS, by default that shift will be masked and the employee will be considered available for callout. Web Application users will have the responsibility of flagging the shift record in ARCOS as “Working Holiday” directly in ARCOS, if necessary.

200 – Data Record

Ref	Field Description and Type	Format and Notes
1.	Record – Type N(3)	200 Data Record - **REQUIRED**
2.	Emp_ID – Type X(11)	Alpha numeric - **REQUIRED**
3.	Event Start Date – Type X(12)	Format MMDDYYYYHHMM **- REQUIRED**
4.	Event End Date – Type X(12)	Format MMDDYYYYHHMM **- REQUIRED**
5.	Event Type – Type X(11)	Event type identifier **– REQUIRED**
6.	Shift Name – Type X(15)	ShiftID from database **-** To be used as ARCOS Shift ReferAs value if provided
7.	Shift Treat As – Type X(8)	Format MMDDYYYY

The field named 'Shift TreatAs' which exists at the end of the file will be used to indicate what the 'Treat As' date should be in ARCOS for the purpose of displaying the shift on the schedule. Currently, the 'Treat As' value is determined using the start date/time of each schedule record, as determined by the 'Event Start Date' field.

Examples to illustrate how the data will be depicted in the ARCOS schedule:

If this data is sent:

Ref	Field Description and Type	Value
1.	Record – Type N(3)	200
2.	Emp_Id – Type X(11)	123456
3.	Event Start Date – Type X(12)	090120142300
4.	Event End Date – Type X(14)	09022014070000
5.	Event Type – Type X(11)	1008
6.	Shift Name – Type X(15)	
7.	Shift Treat As – Type X(8)	No data OR Treat As = Event Start Date

The shift will appear on the schedule as: (This is how the interface currently behaves)

Select	Name	Shift	Mon 9/1
☐	Alexander, Matthew	OWL SHIFT	Sh 23:00-07:00(N)

If this data is sent:

Ref	Field Description and Type	Value
1.	Record – Type N(3)	200
2.	Emp_Id – Type X(11)	123456
3.	Event Start Date – Type X(14)	09012014230000
4.	Event End Date – Type X(14)	09022014070000
5.	Event Type – Type X(11)	108
6.	Shift Name – Type X(15)	
7.	Shift TreatAs – Type X(8)	09022014

The shift will appear on the schedule as:

Select	Name	Shift	Mon 9/1	Tue 9/2
☐	Alexander, Matthew	JK TEST SHIFT		Sh (P)23:00-07:00

The ‘P’ indicates that the start time is actually on the previous date, which is 9/1 in this example.

Caveats

- If the ‘Treat As’ field is not populated, the ‘Treat As’ date will be determined based on the start date/time of the schedule record as determined by the ‘Event Start Date’ field.
- If the ‘Treat As’ field is populated, the ‘Treat As’ date cannot be more than 1 day in the future or 1 day in the past from the Event Start Date field in the schedule record.
- If the date is invalid due to improper format or being more than 1 day removed from the Event Start Date, then the record will fail and be written into the failure file.

ULT (Schedule Data, Ultipro)

The ULT file type was developed to update ARCOS schedule records with information from the Ultipro scheduling system.

100 – Header/Trailer Record

The Header Record will be used by ARCOS for data integrity checking. ARCOS will make a first pass through the file counting the records and will then verify that the Header / Trailer record count is correct prior to processing the file. If the actual number of records in the file does not match the count stored in the header record, ***the file will not be processed at all*** and an error report will be created.

Ref	Field Description and Type	Format and Notes
1.	Record Type – Type N(3)	100 (Header Record) See description above. – **REQUIRED**
2.	Company Identifier – Type X(3)	[Customer Code] ARCOS Customer Company Acronym (Company SCHEMA) – **REQUIRED**
3.	Export Date and Time – Type X(14)	Format MMDDYYYYHHMMSS (Seconds are optional) Hours (HH) are 24-hour clock. – **REQUIRED**
4.	Effective Date – Type X(8)	Format MMDDYYYY – **REQUIRED**
5.	Export Record Count – Type N(5)	Count of type 200 records – **REQUIRED**

200 – Data Record

Ref	Field Description and Type	Format and Notes
1.	Record – Type N(3)	200 Data Record – **REQUIRED**
2.	Emp_Id – Type X(11)	**REQUIRED**
3.	Event Start Date – Type X(12)	Format MMDDYYYYHHMM – **REQUIRED**
4.	Event End Date – Type X(12)	Format MMDDYYYYHHMM – **REQUIRED**
5.	Event Type – Type X(11)	Event type identifier – **REQUIRED**
6.	Shift Name – Type X(15)	ShiftID from UltiPro To be used as ARCOS Shift ReferAs value

Notes: At the start of each data load, ARCOS schedule data for the 14-day period starting with “Effective Date” will be purged from the ARCOS system, EXCEPT for any “protected” events. Speak with your ARCOS Representative if you need further information on your event configuration.

ARCOS will find records in the window based strictly on the START date/time of the record. I.e.: If an existing record starts yesterday and crosses midnight into today (or beyond), it will NOT be purged during the load.

ARCOS will ignore (and report as an error) any record with a start date prior to the effective date, or beyond the 14-day window end date.

For normal shift records – the TreatAs date in ARCOS will be determined using the Start Date/Time of the schedule record (only matters for shift records crossing midnight).

Field 6 of type ‘200’ rows will be stored in ARCOS as the Shift ReferAs value for the shift definition on the ARCOS Scheduler which will allow for visual representation of normal weekly shift assignments for employees. Only one value per week (Monday – Sunday) can be stored. If different shift names are sent in the ULT file for an employee that occur within the same effective week, errors will be logged and a ReferAs value will not be populated.

If a shift recorded is loaded in ARCOS by the ULT file and it lands on a day defined as a Holiday in ARCOS, by default that shift will be masked and the employee will be considered available for callout. Web Application users will have the responsibility of flagging the shift record in ARCOS as “Working Holiday” directly in ARCOS, if necessary.

ARCOS LLC
8800 Lyra Drive, Suite 200
Columbus, OH 43240
(614) 396 – 5500

CML1 (Crew Manager Lodging)

The CM file types are for Crew Manager file loads through the Data Loader. These files are *not* used in the Data Import functionality within Crew Manager. These files need to be loaded using the Sys Admin > Loader functionality or Loader API methods.
The CML1 file is used to load Lodging data including the name, identified lodging attribute values, and capacity/room availability of each Hotel.

Record Descriptions

100 – Header/Trailer Record

The Header Record will be used by ARCOS for data integrity checking. ARCOS will make a first pass through the file counting the records and will then verify that the Header / Trailer record count is correct prior to processing the file. If the actual number of records in the file does not match the count stored in the header record, ***the file will not be processed at all*** and an error report will be created.

Ref	Field Description and Type	Format and Notes
1.	Record Type – Type N(3)	100 (Header Record) – **REQUIRED** **See description above.**
2.	Company Identifier – Type X(10)	[Customer Code] ARCOS Customer Company Acronym (Company SCHEMA) – **REQUIRED**
3.	Export Date and Time – Type N(14)	Format MMDDYYYYHHMMSS (Seconds are optional) Hours (HH) are 24 hour clock and seconds are optional. – **REQUIRED**
4.	Export Record Count – Type N	Total records not including headers – **REQUIRED**
5.	Logic Indicator – Type N	100 – This field is a place holder and must be populated with 100 at this time. – **REQUIRED**

ADDITIONAL NOTES:

Any Field not indicated as REQUIRED should be Optional.

Fields marked as “N/A” or ‘IF USED’ are either features currently not used by the customer or fields that have no bearing on this project. These fields should be left null in the load file. If data exists in these fields in the load file, it will be ignored during the load process.

ARCOS LLC
8800 Lyra Drive, Suite 200
Columbus, OH 43240
(614) 396 – 5500

101 – Column Names Record

The Column Names should be written as the second to the last record in the file; it will be used by ARCOS to match values provided the file with pre-configured attributes in Crew Manager. ARCOS will make a pass through the 101 record, and will validate that matching Crew Manager Work attributes have been configured. IF the 101 record and the number of values in the 200 record do not match, ARCOS will reject the file.

If the number of columns in the 101 and the 200 records match, but ARCOS does not find a matching Crew Manager Work attribute it will load the file, but will ignore any attributes that do not match configured Crew Manager Work attributes.

Ref	Field Description and Type	Format and Notes
1.	Record Type – Type N(3)	101 (Header Record) – **REQUIRED** **See description above.**
2.	Column Name – Type X	**name** (Must be entered in all lowercase to identify the Work Order name) – **REQUIRED**
3.	Column Name – Type X	[Crew Manager Attribute Name] – **REQUIRED**
4.	Column Name – Type X	[Crew Manager Attribute Name] – **REQUIRED**
5.	Column Name – Type X	Column names will continue until each Crew Manager attribute name to be included has been identified Column names must match the Crew Manager attribute name exactly. **Case sensitive** – **REQUIRED**

ARCOS LLC
8800 Lyra Drive, Suite 200
Columbus, OH 43240
(614) 396 – 5500

CMR1 (Crew Manager Resource Types)

The CM file types are for Crew Manager file loads through the Data Loader. These files are *not* used in the Data Import functionality within Crew Manager. These files need to be loaded using the Sys Admin > Loader functionality or Loader API methods.
The CMR1 file is used to upload Resource Types and the identified resource types' attribute values.

Record Descriptions

100 – Header/Trailer Record

The Header Record will be used by ARCOS for data integrity checking. ARCOS will make a first pass through the file counting the records and will then verify that the Header / Trailer record count is correct prior to processing the file. If the actual number of records in the file does not match the count stored in the header record, the file will not be processed and an error report will be created.

Ref	Field Description and Type	Format and Notes
1.	Record Type – Type N(3)	100 (Header Record) – **REQUIRED** **See description above.**
2.	Company Identifier – Type X	[Company Code] ARCOS Customer Company Acronym (Company SCHEMA) – **REQUIRED**
3.	Export Date and Time – Type N(12)	Format MMDDYYYYHHMMSS Seconds are Optional Hours (HH) are 24 hour clock. – **REQUIRED**
4.	Export Record Count – Type N	Total records not including headers – **REQUIRED**
5.	Logic Indicator – Type N	100 – This field is a place holder and must be populated with 100 at this time. – **REQUIRED**
6.	Resource Name – Type X	This field must match the resource name exactly. Case sensitive. Example: If resource section called Vehicle, field must also contain Vehicle, If field input is vehicle or VEHICLE file load would fail. – **REQUIRED**

ADDITIONAL NOTES:

Any Field not indicated as REQUIRED should be Optional.

Fields marked as “N/A” or ‘IF USED’ are either features currently not used by the customer or fields that have no bearing on this project. These fields should be left null in the load file. If data exists in these fields in the load file, it will be ignored during the load process.

ARCOS LLC
8800 Lyra Drive, Suite 200
Columbus, OH 43240
(614) 396 – 5500

101 – Column Names Record

The Column Names should be written as the second to the last record in the file; it will be used by ARCOS to match values provided the file with pre-configured attributes in Crew Manager. ARCOS will make a pass through the 101 record and will validate that matching attributes in the Crew Manager Resource Type (identified in 100 Record) have been created. IF the 101 record row attributes identified and the number of values separated by a delimiter in the 200 record do not match, ARCOS will reject the file.
If the number of columns in the 101 and the 200 records match, but ARCOS does not find a matching Crew Manager Resource Type attribute it will load the file, but will ignore any attributes that do not match configured Crew Manager Resource Type attributes.
The 101 record is limited to only identify thirty (30) values including the 101 record identifier and the Resource Type name

Ref	Field Description and Type	Format and Notes
1.	Record Type – Type N	101 (Header Record) – **REQUIRED** **See description above.**
2.	Resource Type Name – Type X	**name** (Must be entered in all lowercase to identify the Work Order name) – **REQUIRED**
3.	Resource Type Attribute Name – Type X	[Crew Manager Attribute Name] – **REQUIRED**
4.	Resource Type Attribute Name – Type X	[Crew Manager Attribute Name] – **REQUIRED**
5.	Resource Type Attribute Name – Type X	Column names will continue until each Crew Manager attribute name to be included has been identified Column names must match the Crew Manager attribute name exactly. **Case sensitive** – **REQUIRED**

CMW1 (Crew Manager Workorders)

The CM file types are for Crew Manager file loads through the Data Loader. These files are *not* used in the Data Import functionality within Crew Manager. These files need to be loaded using the Sys Admin > Loader functionality or Loader API methods.
The CMW1 file is used to add, update, or delete WorkOrder Resources and the identified WorkOrder attribute values in Crew Manager.

Record Descriptions

100 – Header/Trailer Record

The Header Record will be used by ARCOS for data integrity checking. ARCOS will make a first pass through the file counting the records and will then verify that the Header / Trailer record count is correct prior to processing the file. If the actual number of records in the file does not match the count stored in the header record, ***the file will not be processed at all*** and an error report will be created.

Ref	Field Description and Type	Format and Notes
1.	Record Type – Type N	100 (Header Record) – **REQUIRED** **See description above.**
2.	Company Identifier – Type X	[Company Code] ARCOS Customer Company Acronym (Company SCHEMA) – **REQUIRED**
3.	Export Date and Time – Type N(14)	Format MMDDYYYYHHMMSS Seconds are Optional Hours (HH) are 24 hour clock. – **REQUIRED**
4.	Export Record Count – Type N	Total records not including headers – **REQUIRED**
5.	Logic Indicator – Type N	100 – **REQUIRED** Notes: This will be a numeric indicator of both the field mapping logic, as well as the logic for ADD, UPDATE and DELETE for the customer. - ARCOS will overwrite ALL Workorder information if a WO ID already exists in ARCOS, and it is in the OPEN Status. - ARCOS will not update anything if WO ID STATUS is = ASSIGNED, IN-PROGRESS, CLOSED

ADDITIONAL NOTES:

Any Field not indicated as REQUIRED should be Optional.

Fields marked as “N/A” or ‘IF USED’ are either features currently not used by [CUSTOMER] or fields that have no bearing on this project. These fields should be left null in the load file. If data exists in these fields in the load file, it will be ignored during the load process.

Logic Indicator – your ARCOS Implementation Manager or Professional Services consultant will work with you on the logic indicator required for your file load. The Logic indicator provides ARCOS with information as to how your file is loaded, which fields are “Protected” from further CMW1 updates,

which fields are updated, when records are added, when records are deleted, etc… Please note that the details of this requirement should be discussed with your ARCOS Implementation Manager or Professional Services Consultant, who may further review the request with ARCOS Engineering team.

101 – Column Names Record

The Column Names should be written as the second to the last record in the file; it will be used by ARCOS to match values provided the file with pre-configured attributes in Crew Manager. ARCOS will make a pass through the 101 record, and will validate that matching Crew Manager Work attributes have been configured. IF the 101 record and the number of values in the 200 record do not match, ARCOS will reject the file.

If the number of columns in the 101 and the 200 records match, but ARCOS does not find a matching Crew Manager Work attribute it will load the file, but will ignore any attributes that do not match configured Crew Manager Work attributes.

Ref	Field Description and Type	Format and Notes
1.	Record Type – Type N	101 (Header Record) – **REQUIRED** **See description above.**
2.	Work Order Name – Type X	**name** (Must be entered in all lowercase to identify the Work Order name) – **REQUIRED**
3.	Column Name – Type X	[Crew Manager Attribute Name] – **REQUIRED**
4.	Column Name – Type X	[Crew Manager Attribute Name] – **REQUIRED**
5.	Column Name – Type X	Column names will continue until each Crew Manager Work Order attribute name to be included has been identified Column names must match the Crew Manager Work Order attribute name exactly. **Case sensitive** – **REQUIRED**

IEX (IEX Scheduling System)

The IEX file type was developed to update ARCOS schedule records with information from the IEX scheduling system. IEX files update schedule data and are primarily used by a single customer.
Please speak with an ARCOS Representative for this specification.

XML Schema Definitions (XSD)

The XSD provided in this section represents the data structures available within ARCOS. The entire XSD for your environment can be obtained by following the instructions in the *Application – Supported Operations* section.

Common Data Structures

statusList

```
<xs:complexType name="statusList">
  <xs:sequence>
```

```
    <xs:element name="status" type="opStatus" minOccurs="0" maxOccurs="unbounded" />
  </xs:sequence>
 </xs:complexType>
```

opStatus

```
 <xs:complexType name="opStatus">
  <xs:sequence />
  <xs:attribute name="id" type="xs:string" />
  <xs:attribute name="name" type="xs:string" />
  <xs:attribute name="code" type="xs:int" use="required" />
  <xs:attribute name="message" type="xs:string" />
 </xs:complexType>
```

Description
The statusList of opStatus provides status information about the outcome of any data modification request.

Notes
The id attribute value will be set to the ID of the entity that was created or updated.
The code attribute value will typically be a ‘1’ on success and any other value to indicate an error or other information.
The message attribute value will provide feedback on the status of the operation. If an error occurred, this attribute value will usually contain a description of the problem. Example XML response indicating a malformed request was provided by the client:

```
<opStatus id="-1" code="400" message="HTTP 400 Bad Request"/>
```

ARCOS LLC
8800 Lyra Drive, Suite 200
Columbus, OH 43240
(614) 396 – 5500

Loader / Extract Data Structures

Extract

```
<xs:complexType name="extract">
  <xs:sequence />
  <xs:attribute name="name" type="xs:string" />
</xs:complexType>
<xs:complexType name="extractConfiguration">
  <xs:sequence>
    <xs:element name="schedule" type="extractSchedule" minOccurs="0" />
    <xs:element name="properties" type="mapElementArray" minOccurs="0" />
  </xs:sequence>
</xs:complexType>
```

extractSchedule

```
<xs:complexType name="extractSchedule">
  <xs:sequence />
  <xs:attribute name="minute" type="xs:int" use="required" />
  <xs:attribute name="hour" type="xs:int" use="required" />
  <xs:attribute name="dayOfMonth" type="xs:string" />
  <xs:attribute name="month" type="xs:string" />
  <xs:attribute name="dayOfWeek" type="xs:string" />
  <xs:attribute name="nextRun" type="xs:string" />
</xs:complexType>
```

Row

```
<xs:complexType name="row">
  <xs:sequence>
    <xs:element ref="warning" minOccurs="0" maxOccurs="unbounded" />
  </xs:sequence>
  <xs:attribute name="id" type="xs:int" use="required" />
  <xs:attribute name="type" type="xs:int" use="required" />
</xs:complexType>
```

Warning

```
<xs:complexType name="warning">
  <xs:sequence />
  <xs:attribute name="message" type="xs:string" />
  <xs:attribute name="value" type="xs:string" />
  <xs:attribute name="severity" type="xs:string" />
  <xs:attribute name="columnId" type="xs:int" />
  <xs:attribute name="title" type="xs:string" />
</xs:complexType>
```

loadDetail

```
<xs:complexType name="loadDetail">
  <xs:sequence>
    <xs:element name="data" type="xs:string" minOccurs="0" />
    <xs:element name="warnings" minOccurs="0">
      <xs:complexType>
        <xs:sequence>
          <xs:element ref="warning" minOccurs="0" maxOccurs="unbounded" />
        </xs:sequence>
      </xs:complexType>
    </xs:element>
  </xs:sequence>
  <xs:attribute name="type" type="xs:string" />
  <xs:attribute name="result" type="xs:string" />
</xs:complexType>
```

loadRequest

```
<xs:complexType name="loadRequest">
  <xs:sequence>
    <xs:element name="warnings" minOccurs="0">
      <xs:complexType>
        <xs:sequence>
          <xs:element ref="row" minOccurs="0" maxOccurs="unbounded" />
        </xs:sequence>
      </xs:complexType>
    </xs:element>
  </xs:sequence>
  <xs:attribute name="loadId" type="xs:int" use="required" />
  <xs:attribute name="filename" type="xs:string" />
  <xs:attribute name="type" type="xs:string" />
  <xs:attribute name="runDate" type="xs:string" />
  <xs:attribute name="dataRows" type="xs:int" use="required" />
  <xs:attribute name="loadStatus" type="xs:string" />
</xs:complexType>
```

loadResult

```
<xs:complexType name="loadResult">
  <xs:sequence />
  <xs:attribute name="loadId" type="xs:int" use="required" />
  <xs:attribute name="filename" type="xs:string" />
  <xs:attribute name="type" type="xs:string" />
  <xs:attribute name="date" type="xs:string" />
  <xs:attribute name="loadedBy" type="xs:string" />
  <xs:attribute name="dataRows" type="xs:int" use="required" />
  <xs:attribute name="success" type="xs:int" use="required" />
  <xs:attribute name="warning" type="xs:int" use="required" />
  <xs:attribute name="failure" type="xs:int" use="required" />
  <xs:attribute name="result" type="xs:string" />
</xs:complexType>
```

ARCOS LLC
8800 Lyra Drive, Suite 200
Columbus, OH 43240
(614) 396 – 5500

Callout Data Structures

Attempt

```
<xs:complexType name="attempt">
  <xs:sequence>
    <xs:element name="sequences" minOccurs="0">
      <xs:complexType>
        <xs:sequence>
          <xs:element ref="sequence" minOccurs="0" maxOccurs="unbounded" />
        </xs:sequence>
      </xs:complexType>
    </xs:element>
  </xs:sequence>
  <xs:attribute name="id" type="xs:int" />
  <xs:attribute name="attempt" type="xs:int" />
</xs:complexType>
```

Roster

```
<xs:complexType name="roster">
  <xs:sequence>
    <xs:element name="location" type="attribute" minOccurs="0" />
    <xs:element name="classification" type="attribute" minOccurs="0" />
    <xs:element name="listStatus" type="attribute" minOccurs="0" />
    <xs:element name="employees" minOccurs="0">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="employee" type="calloutEmployee" minOccurs="0" maxOccurs="unbounded" />
        </xs:sequence>
      </xs:complexType>
    </xs:element>
  </xs:sequence>
</xs:complexType>
```

Attribute

```
<xs:complexType name="attribute">
  <xs:sequence />
  <xs:attribute name="name" type="xs:string" />
  <xs:attribute name="value" type="xs:string" />
  <xs:attribute name="id" type="xs:int" />
  <xs:attribute name="key" type="xs:string" />
  <xs:attribute name="xwalk" type="xs:string" />
</xs:complexType>
```

Sequence

```
<xs:complexType name="sequence">
  <xs:sequence>
    <xs:element name="chainDelay" type="xs:int" minOccurs="0" />
    <xs:element name="manualPopup" type="xs:boolean" />
    <xs:element name="multiPosChunk" type="xs:int" minOccurs="0" />
    <xs:element ref="roster" minOccurs="0" />
    <xs:element name="displayName" type="xs:string" minOccurs="0" />
    <xs:element name="callResults" minOccurs="0">
      <xs:complexType>
        <xs:sequence>
          <xs:element ref="callResult" minOccurs="0" maxOccurs="unbounded" />
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="seqDelays" minOccurs="0">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="seqDelay" type="callResult" minOccurs="0" maxOccurs="unbounded" />
        </xs:sequence>
      </xs:complexType>
    </xs:element>
  </xs:sequence>
  <xs:attribute name="id" type="xs:int" />
  <xs:attribute name="sequence" type="xs:int" />
</xs:complexType>
```

calloutEmployee

```
<xs:complexType name="calloutEmployee">
  <xs:sequence>
    <xs:element name="callOrder" type="xs:int" />
    <xs:element name="id" type="employeeId" minOccurs="0" />
    <xs:element name="hriData" type="employee" minOccurs="0" />
    <xs:element name="positionData" minOccurs="0">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="positionData" type="jsonObject" minOccurs="0" maxOccurs="unbounded" />
        </xs:sequence>
      </xs:complexType>
    </xs:element>
  </xs:sequence>
</xs:complexType>
```

employeeId

```
<xs:complexType name="employeeId">
  <xs:sequence>
    <xs:element name="webId" type="xs:string" minOccurs="0" />
    <xs:element name="vruId" type="xs:string" minOccurs="0" />
    <xs:element name="emplId" type="xs:string" minOccurs="0" />
    <xs:element name="name" type="xs:string" minOccurs="0" />
  </xs:sequence>
  <xs:attribute name="id" type="xs:int" />
 </xs:complexType>
```

callResult

```
 <xs:complexType name="callResult">
  <xs:sequence>
    <xs:element name="resultName" type="xs:string" minOccurs="0" />
    <xs:element name="userName" type="xs:string" minOccurs="0" />
    <xs:element name="reason" type="xs:string" minOccurs="0" />
    <xs:element name="primLocName" type="xs:string" minOccurs="0" />
    <xs:element name="primClassName" type="xs:string" minOccurs="0" />
    <xs:element name="contact" type="employeeId" minOccurs="0" />
    <xs:element name="exceptionId" type="xs:int" minOccurs="0" />
    <xs:element name="attemptSourceName" type="xs:string" minOccurs="0" />
    <xs:element name="attemptDate" type="xs:dateTime" minOccurs="0" />
    <xs:element name="resultDate" type="xs:dateTime" minOccurs="0" />
    <xs:element name="coCallId" type="xs:int" minOccurs="0" />
    <xs:element name="phoneNumber" type="xs:string" minOccurs="0" />
    <xs:element name="callStart" type="xs:dateTime" minOccurs="0" />
    <xs:element name="callEnd" type="xs:dateTime" minOccurs="0" />
    <xs:element name="manualBypass" type="xs:string" minOccurs="0" />
    <xs:element name="failed" type="xs:boolean" />
    <xs:element name="contactOverride" type="xs:boolean" />
    <xs:element name="cttInfo" type="xs:string" minOccurs="0" />
    <xs:element name="inUseLabel" type="xs:string" minOccurs="0" />
    <xs:element name="expDate" type="xs:dateTime" minOccurs="0" />
    <xs:element name="nonResponse" type="xs:boolean" />
    <xs:element name="finalResult" type="xs:boolean" />
    <xs:element name="addDate" type="xs:dateTime" minOccurs="0" />
    <xs:element name="coTypeName" type="xs:string" minOccurs="0" />
    <xs:element name="amountName" type="xs:string" minOccurs="0" />
    <xs:element name="ccResultName" type="xs:string" minOccurs="0" />
    <xs:element name="chargeStatus" type="xs:string" minOccurs="0" />
    <xs:element name="chargeDate" type="xs:dateTime" minOccurs="0" />
  </xs:sequence>
  <xs:attribute name="resultId" type="xs:int" />
  <xs:attribute name="subCalloutId" type="xs:int" />
  <xs:attribute name="attemptNumber" type="xs:int" />
  <xs:attribute name="sequenceNumber" type="xs:int" />
  <xs:attribute name="callOrder" type="xs:int" />
  <xs:attribute name="subAttemptNumber" type="xs:string" />
 </xs:complexType>
```

callDetail

```
<xs:complexType name="callDetail">
  <xs:sequence>
    <xs:element name="contactName" type="xs:string" minOccurs="0" />
    <xs:element name="type" type="xs:string" minOccurs="0" />
    <xs:element name="callStart" type="xs:string" minOccurs="0" />
    <xs:element name="callEnd" type="xs:string" minOccurs="0" />
    <xs:element name="duration" type="xs:string" minOccurs="0" />
    <xs:element name="phoneNumber" type="xs:string" minOccurs="0" />
    <xs:element name="host" type="xs:string" minOccurs="0" />
    <xs:element name="channel" type="xs:string" minOccurs="0" />
    <xs:element name="result" type="xs:string" minOccurs="0" />
    <xs:element name="additionalDigits" type="xs:string" minOccurs="0" />
    <xs:element name="callDial" type="xs:string" minOccurs="0" />
    <xs:element name="timeRinging" type="xs:string" minOccurs="0" />
    <xs:element name="recordingFound" type="xs:boolean" />
    <xs:element name="failed" type="xs:boolean" />
    <xs:element name="recordingURL" type="xs:string" minOccurs="0" />
    <xs:element name="recordingURLWav" type="xs:string" minOccurs="0" />
    <xs:element name="transferCallIds" minOccurs="0">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="transferCalls" type="xs:string" minOccurs="0" maxOccurs="unbounded" />
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="digitsList" minOccurs="0">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="digits" type="digitsReceived" minOccurs="0" maxOccurs="unbounded" />
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="cloudStorage" minOccurs="0">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="cloudStorageInfo" type="xs:string" minOccurs="0" maxOccurs="unbounded" />
        </xs:sequence>
      </xs:complexType>
    </xs:element>
  </xs:sequence>
  <xs:attribute name="callId" type="xs:int" />
</xs:complexType>
```

Callout

```
<xs:complexType name="callout">
  <xs:sequence>
    <xs:element name="calloutType" type="attribute" minOccurs="0" />
    <xs:element name="description" type="xs:string" minOccurs="0" />
    <xs:element name="location" type="attribute" minOccurs="0" />
    <xs:element name="workStartDate" type="xs:string" minOccurs="0" />
    <xs:element name="workEndDate" type="xs:string" minOccurs="0" />
    <xs:element name="calloutReason" type="attribute" minOccurs="0" />
    <xs:element name="troubleLoc" type="attribute" minOccurs="0" />
    <xs:element name="creator" type="employeeId" minOccurs="0" />
    <xs:element name="numberRequired" type="xs:int" minOccurs="0" />
    <xs:element name="numberFilled" type="xs:int" minOccurs="0" />
    <xs:element name="numberOfCalls" type="xs:int" minOccurs="0" />
    <xs:element name="numberOfPeopleCalled" type="xs:int" minOccurs="0" />
    <xs:element name="percentFilled" type="xs:int" minOccurs="0" />
    <xs:element name="percentResponse" type="xs:int" minOccurs="0" />
    <xs:element name="elapsedTime" type="xs:string" minOccurs="0" />
    <xs:element name="owner" type="employeeId" minOccurs="0" />
    <xs:element name="attributes" minOccurs="0">
      <xs:complexType>
        <xs:sequence>
          <xs:element ref="attribute" minOccurs="0" maxOccurs="unbounded" />
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="overrides" minOccurs="0">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="override" type="attribute" minOccurs="0" maxOccurs="unbounded" />
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="qualifications" minOccurs="0">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="qualification" type="attribute" minOccurs="0" maxOccurs="unbounded" />
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="subCallouts" minOccurs="0">
      <xs:complexType>
        <xs:sequence>
          <xs:element ref="subCallout" minOccurs="0" maxOccurs="unbounded" />
        </xs:sequence>
      </xs:complexType>
    </xs:element>
  </xs:sequence>
  <xs:attribute name="id" type="xs:int" />
  <xs:attribute name="initiate" type="xs:boolean" />
</xs:complexType>
```

subCallout

```
<xs:complexType name="subCallout">
  <xs:sequence>
    <xs:element name="location" type="attribute" minOccurs="0" />
    <xs:element name="classification" type="attribute" minOccurs="0" />
    <xs:element name="currAtt" type="xs:int" minOccurs="0" />
    <xs:element name="beingCalled" type="xs:int" minOccurs="0" />
    <xs:element name="status" type="xs:int" minOccurs="0" />
    <xs:element name="statusName" type="xs:string" minOccurs="0" />
    <xs:element name="numberRequired" type="xs:int" minOccurs="0" />
    <xs:element name="numberFilled" type="xs:int" minOccurs="0" />
    <xs:element name="numberOfCalls" type="xs:int" minOccurs="0" />
    <xs:element name="numberOfPeopleCalled" type="xs:int" minOccurs="0" />
    <xs:element name="percentFilled" type="xs:int" minOccurs="0" />
    <xs:element name="percentResponse" type="xs:int" minOccurs="0" />
    <xs:element name="elapsedTime" type="xs:string" minOccurs="0" />
    <xs:element name="defaultAction" type="xs:string" minOccurs="0" />
    <xs:element name="alternateActions" type="xs:string" minOccurs="0" />
    <xs:element name="attributes" minOccurs="0">
      <xs:complexType>
        <xs:sequence>
          <xs:element ref="attribute" minOccurs="0" maxOccurs="unbounded" />
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="positions" minOccurs="0">
      <xs:complexType>
        <xs:sequence>
          <xs:element ref="position" minOccurs="0" maxOccurs="unbounded" />
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="attempts" minOccurs="0">
      <xs:complexType>
        <xs:sequence>
          <xs:element ref="attempt" minOccurs="0" maxOccurs="unbounded" />
        </xs:sequence>
      </xs:complexType>
    </xs:element>
  </xs:sequence>
  <xs:attribute name="id" type="xs:int" />
  <xs:attribute name="quantity" type="xs:int" use="required" />
</xs:complexType>
```

digitsReceived

```
<xs:complexType name="digitsReceived">
  <xs:sequence />
  <xs:attribute name="runningTime" type="xs:string" />
  <xs:attribute name="digits" type="xs:string" />
</xs:complexType>
```

Position

```
<xs:complexType name="position">
  <xs:sequence>
    <xs:element name="additionalData">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="entry" minOccurs="0" maxOccurs="unbounded">
            <xs:complexType>
              <xs:sequence>
                <xs:element name="key" minOccurs="0" type="xs:anyType" />
                <xs:element name="value" minOccurs="0" type="xs:anyType" />
              </xs:sequence>
            </xs:complexType>
          </xs:element>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
  </xs:sequence>
  <xs:attribute name="customerPositionId" type="xs:string" />
  <xs:attribute name="positionTitle" type="xs:string" />
  <xs:attribute name="quantity" type="xs:int" />
  <xs:attribute name="workStartDate" type="xs:string" />
  <xs:attribute name="workEndDate" type="xs:string" />
</xs:complexType>
```

calloutActivation

```
<xs:complexType name="calloutActivation">
  <xs:sequence>
    <xs:element name="calloutType" type="attribute" minOccurs="0" />
    <xs:element name="status" type="attribute" minOccurs="0" />
    <xs:element name="location" type="attribute" minOccurs="0" />
    <xs:element name="positions" minOccurs="0">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="positions" type="attribute" minOccurs="0" maxOccurs="unbounded" />
        </xs:sequence>
      </xs:complexType>
    </xs:element>
  </xs:sequence>
  <xs:attribute name="id" type="xs:int" use="required" />
  <xs:attribute name="description" type="xs:string" />
  <xs:attribute name="activationLevel" type="xs:int" use="required" />
  <xs:attribute name="calloutId" type="xs:int" use="required" />
</xs:complexType>
```

calloutAttribute

```
<xs:complexType name="calloutAttribute">
  <xs:sequence />
  <xs:attribute name="id" type="xs:int" use="required" />
  <xs:attribute name="name" type="xs:string" />
  <xs:attribute name="attributeType" type="xs:string" />
  <xs:attribute name="modifierType" type="xs:string" />
  <xs:attribute name="description" type="xs:string" />
  <xs:attribute name="fullDescription" type="xs:string" />
  <xs:attribute name="activeDescription" type="xs:string" />
  <xs:attribute name="valueDescription" type="xs:string" />
</xs:complexType>
```

calloutAttributeValue

```
<xs:complexType name="calloutAttributeValue">
  <xs:sequence />
  <xs:attribute name="id" type="xs:int" use="required" />
  <xs:attribute name="active" type="xs:string" />
  <xs:attribute name="value" type="xs:string" />
</xs:complexType>
```

calloutResult

```
<xs:complexType name="calloutResult">
  <xs:sequence>
    <xs:element ref="callResult" minOccurs="0" maxOccurs="unbounded" />
  </xs:sequence>
  <xs:attribute name="coMainId" type="xs:int" />
</xs:complexType>
```

calloutSearchResult

```
<xs:complexType name="calloutSearchResult">
  <xs:sequence>
    <xs:element name="calloutType" type="attribute" minOccurs="0" />
    <xs:element name="calloutReason" type="attribute" minOccurs="0" />
    <xs:element name="workStartDate" type="xs:dateTime" minOccurs="0" />
    <xs:element name="strWorkStartDate" type="xs:string" minOccurs="0" />
    <xs:element name="location" type="attribute" minOccurs="0" />
    <xs:element name="subCallouts" minOccurs="0">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="subCallout" type="attribute" minOccurs="0" maxOccurs="unbounded" />
        </xs:sequence>
      </xs:complexType>
    </xs:element>
  </xs:sequence>
  <xs:attribute name="id" type="xs:int" use="required" />
  <xs:attribute name="createdBy" type="xs:string" />
  <xs:attribute name="description" type="xs:string" />
</xs:complexType>
```

calloutStat

```
<xs:complexType name="calloutStat">
  <xs:sequence>
    <xs:element name="coNumber" type="xs:int" minOccurs="0" />
    <xs:element name="requestedNumber" type="xs:int" minOccurs="0" />
    <xs:element name="filledNumber" type="xs:int" minOccurs="0" />
    <xs:element name="callsNumber" type="xs:int" minOccurs="0" />
    <xs:element name="peopleNumber" type="xs:int" minOccurs="0" />
    <xs:element name="elapsed" type="xs:int" minOccurs="0" />
    <xs:element name="elapsedRunning" type="xs:int" minOccurs="0" />
    <xs:element name="arriveSec" type="xs:int" minOccurs="0" />
    <xs:element name="arriveAcc" type="xs:int" minOccurs="0" />
    <xs:element name="startTime" type="xs:string" minOccurs="0" />
    <xs:element name="endTime" type="xs:string" minOccurs="0" />
    <xs:element name="dispatchTime" type="xs:string" minOccurs="0" />
    <xs:element name="enrouteTime" type="xs:string" minOccurs="0" />
    <xs:element name="arrivalTime" type="xs:string" minOccurs="0" />
    <xs:element name="updateResult" type="xs:string" minOccurs="0" />
  </xs:sequence>
  <xs:attribute name="coMainId" type="xs:int" />
  <xs:attribute name="ticketId" type="xs:string" />
</xs:complexType>
```

Override

```
<xs:complexType name="override">
  <xs:sequence>
    <xs:element name="event" type="attribute" minOccurs="0" />
  </xs:sequence>
  <xs:attribute name="value" type="xs:string" />
</xs:complexType>
```

classReq

```
<xs:complexType name="classReq">
  <xs:sequence />
  <xs:attribute name="classId" type="xs:int" use="required" />
  <xs:attribute name="className" type="xs:string" />
  <xs:attribute name="quantity" type="xs:int" use="required" />
</xs:complexType>
```

coCrew

```
<xs:complexType name="coCrew">
  <xs:sequence>
    <xs:element name="requiredClasses" minOccurs="0">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="crewComponent" type="classReq" minOccurs="0" maxOccurs="unbounded" />
        </xs:sequence>
      </xs:complexType>
    </xs:element>
  </xs:sequence>
  <xs:attribute name="id" type="xs:int" use="required" />
  <xs:attribute name="name" type="xs:string" />
</xs:complexType>
```

calloutType

```
<xs:complexType name="calloutType">
```

```
  <xs:sequence>
    <xs:element name="category" type="attribute" minOccurs="0" />
    <xs:element name="attributes" minOccurs="0">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="attribute" type="calloutAttributeValue" minOccurs="0" maxOccurs="unbounded" />
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="overrides" minOccurs="0">
      <xs:complexType>
        <xs:sequence>
          <xs:element ref="override" minOccurs="0" maxOccurs="unbounded" />
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="templates" type="xs:int" />
    <xs:element name="recording" type="xs:int" />
  </xs:sequence>
  <xs:attribute name="id" type="xs:int" use="required" />
  <xs:attribute name="name" type="xs:string" />
  <xs:attribute name="displayOrder" type="xs:int" use="required" />
</xs:complexType>
```

coLookup

```
<xs:complexType name="coLookup">
  <xs:sequence>
    <xs:element name="classification" type="attribute" minOccurs="0" />
    <xs:element name="location" type="attribute" minOccurs="0" />
  </xs:sequence>
  <xs:attribute name="id" type="xs:int" use="required" />
  <xs:attribute name="order" type="xs:int" use="required" />
  <xs:attribute name="enabled" type="xs:boolean" use="required" />
</xs:complexType>
```

Reason

```
<xs:complexType name="reason">
  <xs:sequence />
  <xs:attribute name="id" type="xs:int" use="required" />
  <xs:attribute name="description" type="xs:string" />
  <xs:attribute name="displayOrder" type="xs:int" use="required" />
</xs:complexType>
```

coRoster

```
<xs:complexType name="coRoster">
  <xs:sequence>
    <xs:element name="rosterSequences" minOccurs="0">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="sequence" type="rosterSequence" minOccurs="0" maxOccurs="unbounded" />
        </xs:sequence>
      </xs:complexType>
    </xs:element>
  </xs:sequence>
  <xs:attribute name="coLookupId" type="xs:int" use="required" />
  <xs:attribute name="rosterId" type="xs:int" use="required" />
  <xs:attribute name="rosterName" type="xs:string" />
  <xs:attribute name="locationId" type="xs:int" use="required" />
  <xs:attribute name="classId" type="xs:int" use="required" />
  <xs:attribute name="className" type="xs:string" />
  <xs:attribute name="listType" type="xs:string" />
  <xs:attribute name="userId" type="xs:int" use="required" />
  <xs:attribute name="displayOrder" type="xs:int" use="required" />
  <xs:attribute name="listId" type="xs:int" use="required" />
</xs:complexType>
```

rosterSequence

```
<xs:complexType name="rosterSequence">
  <xs:sequence>
    <xs:element name="member" type="rosterEmployee" minOccurs="0" maxOccurs="unbounded" />
  </xs:sequence>
  <xs:attribute name="sequenceNumber" type="xs:int" use="required" />
</xs:complexType>
```

rosterEmployee

```
<xs:complexType name="rosterEmployee">
  <xs:sequence />
  <xs:attribute name="callOrder" type="xs:int" use="required" />
  <xs:attribute name="contactName" type="xs:string" />
  <xs:attribute name="contactId" type="xs:int" use="required" />
  <xs:attribute name="locationName" type="xs:string" />
  <xs:attribute name="locationId" type="xs:int" use="required" />
</xs:complexType>
```

listStatus

```
<xs:complexType name="listStatus">
  <xs:sequence />
  <xs:attribute name="id" type="xs:int" use="required" />
  <xs:attribute name="name" type="xs:string" />
  <xs:attribute name="restrict" type="xs:string" />
  <xs:attribute name="exclusive" type="xs:string" />
  <xs:attribute name="rank" type="xs:int" use="required" />
  <xs:attribute name="primary" type="xs:boolean" use="required" />
  <xs:attribute name="displayOrder" type="xs:int" use="required" />
</xs:complexType>
```

rosterList

```
<xs:complexType name="rosterList">
```

```
  <xs:sequence>
   <xs:element name="archives" minOccurs="0">
    <xs:complexType>
     <xs:sequence>
      <xs:element name="archive" type="xs:string" minOccurs="0" maxOccurs="unbounded" />
     </xs:sequence>
    </xs:complexType>
   </xs:element>
  </xs:sequence>
 </xs:complexType>
```

rosterPreference

```
 <xs:complexType name="rosterPreference">
  <xs:sequence>
   <xs:element name="category" type="attribute" minOccurs="0" />
  </xs:sequence>
  <xs:attribute name="id" type="xs:int" use="required" />
  <xs:attribute name="name" type="xs:string" />
  <xs:attribute name="useStringValue" type="xs:boolean" use="required" />
  <xs:attribute name="value" type="xs:string" />
  <xs:attribute name="multiplier" type="xs:string" />
  <xs:attribute name="description" type="xs:string" />
  <xs:attribute name="displayOrder" type="xs:int" use="required" />
 </xs:complexType>
```

runRosterRulesRequest

```
<xs:complexType name="runRosterRulesRequest">
  <xs:sequence>
    <xs:element name="contactIds" minOccurs="0">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="contactId" type="xs:int" minOccurs="0" maxOccurs="unbounded" />
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="rosterIds" minOccurs="0">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="rosterId" type="xs:int" minOccurs="0" maxOccurs="unbounded" />
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="attrIds" minOccurs="0">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="attrId" type="xs:string" minOccurs="0" maxOccurs="unbounded" />
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="runTest" type="xs:int" />
  </xs:sequence>
</xs:complexType>
```

coContact

```
<xs:complexType name="coContact">
  <xs:sequence />
  <xs:attribute name="id" type="xs:int" use="required" />
  <xs:attribute name="order" type="xs:int" use="required" />
  <xs:attribute name="previewId" type="xs:int" use="required" />
  <xs:attribute name="processCoId" type="xs:int" use="required" />
</xs:complexType>
```

contactInfo

```
<xs:complexType name="contactInfo">
  <xs:sequence />
  <xs:attribute name="id" type="xs:int" use="required" />
  <xs:attribute name="name" type="xs:string" />
  <xs:attribute name="nodeId" type="xs:int" use="required" />
  <xs:attribute name="start" type="xs:string" />
  <xs:attribute name="end" type="xs:string" />
  <xs:attribute name="coId" type="xs:int" use="required" />
</xs:complexType>
```

ARCOS LLC
8800 Lyra Drive, Suite 200
Columbus, OH 43240
(614) 396 – 5500

Employee Data Structures

Address

```
<xs:complexType name="address">
  <xs:sequence />
  <xs:attribute name="street" type="xs:string" />
  <xs:attribute name="streetTwo" type="xs:string" />
  <xs:attribute name="city" type="xs:string" />
  <xs:attribute name="state" type="xs:string" />
  <xs:attribute name="zip" type="xs:string" />
</xs:complexType>
```

emailAddress

```
<xs:complexType name="emailAddress">
  <xs:sequence />
  <xs:attribute name="index" type="xs:int" use="required" />
  <xs:attribute name="address" type="xs:string" />
  <xs:attribute name="description" type="xs:string" />
  <xs:attribute name="enabled" type="xs:boolean" use="required" />
  <xs:attribute name="condensed" type="xs:boolean" use="required" />
  <xs:attribute name="emailProtected" type="xs:boolean" use="required" />
</xs:complexType>
```

coPhone

```
<xs:complexType name="coPhone">
  <xs:complexContent>
    <xs:extension base="transferPhone">
      <xs:sequence />
      <xs:attribute name="sequence" type="xs:int" />
      <xs:attribute name="duty" type="xs:boolean" use="required" />
      <xs:attribute name="trusted" type="xs:boolean" use="required" />
      <xs:attribute name="pinReq" type="xs:boolean" use="required" />
      <xs:attribute name="smsEnabled" type="xs:boolean" use="required" />
      <xs:attribute name="temporary" type="xs:boolean" />
      <xs:attribute name="expirationDate" type="xs:string" />
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

Employee

```
<xs:complexType name="employee">
  <xs:sequence>
    <xs:element name="attributes" minOccurs="0">
      <xs:complexType>
        <xs:sequence>
          <xs:element ref="attribute" minOccurs="0" maxOccurs="unbounded" />
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element ref="address" minOccurs="0" />
    <xs:element name="emails" minOccurs="0">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="email" type="emailAddress" minOccurs="0" maxOccurs="unbounded" />
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="phones" minOccurs="0">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="phone" type="coPhone" minOccurs="0" maxOccurs="unbounded" />
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="pagers" minOccurs="0">
      <xs:complexType>
        <xs:sequence>
          <xs:element ref="pager" minOccurs="0" maxOccurs="unbounded" />
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element ref="transferPhone" minOccurs="0" />
    <xs:element name="webDevices" minOccurs="0">
      <xs:complexType>
        <xs:sequence>
          <xs:element ref="webDevice" minOccurs="0" maxOccurs="unbounded" />
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="extendedAttribute" type="mapElementArray" minOccurs="0" />
  </xs:sequence>
  <xs:attribute name="contactId" type="xs:int" use="required" />
  <xs:attribute name="statusColor" type="xs:string" />
</xs:complexType>
```

transferPhone

```
<xs:complexType name="transferPhone">
  <xs:sequence />
  <xs:attribute name="number" type="xs:string" />
  <xs:attribute name="phoneType" type="xs:string" />
  <xs:attribute name="firstPause" type="xs:string" />
  <xs:attribute name="firstTouchTone" type="xs:string" />
  <xs:attribute name="secondPause" type="xs:string" />
  <xs:attribute name="secondTouchTone" type="xs:string" />
  <xs:attribute name="comments" type="xs:string" />
</xs:complexType>
```

Pager

```
<xs:complexType name="pager">
  <xs:sequence />
  <xs:attribute name="sequence" type="xs:int" />
  <xs:attribute name="number" type="xs:string" />
  <xs:attribute name="onDuty" type="xs:boolean" use="required" />
  <xs:attribute name="pagerType" type="xs:string" />
  <xs:attribute name="pin" type="xs:string" />
  <xs:attribute name="temporary" type="xs:boolean" />
  <xs:attribute name="expirationDate" type="xs:string" />
</xs:complexType>
```

webDevice

```
<xs:complexType name="webDevice">
  <xs:sequence />
  <xs:attribute name="serviceName" type="xs:string" />
  <xs:attribute name="target" type="xs:string" />
  <xs:attribute name="description" type="xs:string" />
  <xs:attribute name="aux" type="xs:string" />
  <xs:attribute name="enabled" type="xs:boolean" use="required" />
</xs:complexType>
```

Classification

```
<xs:complexType name="classification">
   <xs:sequence>
      <xs:element name="locations" minOccurs="0">
         <xs:complexType>
            <xs:sequence>
               <xs:element name="location" type="xs:int" minOccurs="0" maxOccurs="unbounded" />
            </xs:sequence>
         </xs:complexType>
      </xs:element>
      <xs:element name="xwalks" minOccurs="0">
         <xs:complexType>
            <xs:sequence>
               <xs:element name="xwalk" type="xs:string" minOccurs="0" maxOccurs="unbounded" />
            </xs:sequence>
         </xs:complexType>
      </xs:element>
   </xs:sequence>
   <xs:attribute name="classId" type="xs:int" use="required" />
   <xs:attribute name="name" type="xs:string" />
   <xs:attribute name="classType" type="xs:string" />
   <xs:attribute name="order" type="xs:int" use="required" />
</xs:complexType>
```

employeeStub

```
<xs:complexType name="employeeStub">
   <xs:sequence>
      <xs:element name="emails" minOccurs="0">
         <xs:complexType>
            <xs:sequence>
               <xs:element name="email" type="xs:string" minOccurs="0" maxOccurs="unbounded" />
            </xs:sequence>
         </xs:complexType>
      </xs:element>
      <xs:element name="phones" minOccurs="0">
         <xs:complexType>
            <xs:sequence>
              <xs:element name="phone" type="xs:string" minOccurs="0" maxOccurs="unbounded" />
            </xs:sequence>
         </xs:complexType>
      </xs:element>
   </xs:sequence>
   <xs:attribute name="contactId" type="xs:int" use="required" />
   <xs:attribute name="firstName" type="xs:string" />
   <xs:attribute name="lastName" type="xs:string" />
   <xs:attribute name="middleName" type="xs:string" />
   <xs:attribute name="firstLastName" type="xs:string" />
   <xs:attribute name="fullName" type="xs:string" />
   <xs:attribute name="vruId" type="xs:string" />
   <xs:attribute name="webId" type="xs:string" />
   <xs:attribute name="employeeId" type="xs:string" />
   <xs:attribute name="className" type="xs:string" />
   <xs:attribute name="locationName" type="xs:string" />
   <xs:attribute name="emplSource" type="xs:string" />
</xs:complexType>
```

extendedAttribute

```
<xs:complexType name="extendedAttribute">
   <xs:sequence>
      <xs:element name="category" type="attribute" minOccurs="0" />
   </xs:sequence>
   <xs:attribute name="attributeId" type="xs:string" />
   <xs:attribute name="name" type="xs:string" />
   <xs:attribute name="description" type="xs:string" />
   <xs:attribute name="order" type="xs:int" use="required" />
   <xs:attribute name="dataType" type="xs:string" />
   <xs:attribute name="attributeStatusId" type="xs:int" use="required" />
</xs:complexType>
```

extendedAttributeValue

```
<xs:complexType name="extendedAttributeValue">
   <xs:sequence />
   <xs:attribute name="value" type="xs:string" />
   <xs:attribute name="order" type="xs:int" use="required" />
</xs:complexType>
```

Schedule Data Structures

Shift

```
<xs:complexType name="shift">
  <xs:sequence>
    <xs:element name="shiftDays" minOccurs="0">
      <xs:complexType>
        <xs:sequence>
          <xs:element ref="shiftDay" minOccurs="0" maxOccurs="unbounded" />
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="referAsList" minOccurs="0">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="referAs" type="shiftReferAs" minOccurs="0" maxOccurs="unbounded" />
        </xs:sequence>
      </xs:complexType>
    </xs:element>
  </xs:sequence>
  <xs:attribute name="id" type="xs:int" use="required" />
  <xs:attribute name="name" type="xs:string" />
  <xs:attribute name="signature" type="xs:string" />
</xs:complexType>
```

shiftDay

```
<xs:complexType name="shiftDay">
  <xs:sequence />
  <xs:attribute name="dayOfWeek" type="xs:string" />
  <xs:attribute name="startTime" type="xs:string" />
  <xs:attribute name="endTime" type="xs:string" />
  <xs:attribute name="startsOnPreviousDay" type="xs:boolean" use="required" />
  <xs:attribute name="endsOnNextDay" type="xs:boolean" use="required" />
</xs:complexType>
```

shiftReferAs

```
<xs:complexType name="shiftReferAs">
  <xs:sequence />
  <xs:attribute name="referAs" type="xs:string" />
  <xs:attribute name="locationId" type="xs:int" use="required" />
</xs:complexType>
```

employeeSchedule

```
<xs:complexType name="employeeSchedule">
  <xs:sequence>
    <xs:element name="shiftAssignments" minOccurs="0">
      <xs:complexType>
        <xs:sequence>
          <xs:element ref="shiftAssignment" minOccurs="0" maxOccurs="unbounded" />
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="scheduledShifts" minOccurs="0">
      <xs:complexType>
        <xs:sequence>
          <xs:element ref="scheduledShift" minOccurs="0" maxOccurs="unbounded" />
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="scheduleExceptions" minOccurs="0">
      <xs:complexType>
        <xs:sequence>
          <xs:element ref="scheduleException" minOccurs="0" maxOccurs="unbounded" />
        </xs:sequence>
      </xs:complexType>
    </xs:element>
  </xs:sequence>
  <xs:attribute name="contactId" type="xs:int" use="required" />
  <xs:attribute name="vruId" type="xs:string" />
  <xs:attribute name="webId" type="xs:string" />
  <xs:attribute name="empId" type="xs:string" />
  <xs:attribute name="name" type="xs:string" />
  <xs:attribute name="locationId" type="xs:int" use="required" />
</xs:complexType>
```

shiftAssignment

```
<xs:complexType name="shiftAssignment">
  <xs:sequence />
  <xs:attribute name="contactId" type="xs:int" use="required" />
  <xs:attribute name="custIdType" type="xs:string" />
  <xs:attribute name="custIdValue" type="xs:string" />
  <xs:attribute name="shiftId" type="xs:int" use="required" />
  <xs:attribute name="shiftReferAs" type="xs:string" />
  <xs:attribute name="assignmentStart" type="xs:string" />
  <xs:attribute name="modified" type="xs:boolean" use="required" />
  <xs:attribute name="addDate" type="xs:string" />
  <xs:attribute name="changeDate" type="xs:string" />
</xs:complexType>
```

scheduledShift

```
<xs:complexType name="scheduledShift">
  <xs:complexContent>
    <xs:extension base="scheduleRecord">
      <xs:sequence />
      <xs:attribute name="treatAs" type="xs:string" />
      <xs:attribute name="isHoliday" type="xs:boolean" use="required" />
      <xs:attribute name="isWorkingHoliday" type="xs:boolean" use="required" />
      <xs:attribute name="modified" type="xs:boolean" use="required" />
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

scheduleRecord

```
<xs:complexType name="scheduleRecord">
  <xs:sequence>
    <xs:element name="attributes" minOccurs="0">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="attributes" type="attribute" nillable="true" minOccurs="0" maxOccurs="unbounded" />
        </xs:sequence>
      </xs:complexType>
    </xs:element>
  </xs:sequence>
  <xs:attribute name="memexId" type="xs:int" use="required" />
  <xs:attribute name="contactId" type="xs:int" use="required" />
  <xs:attribute name="custIdType" type="xs:string" />
  <xs:attribute name="custIdValue" type="xs:string" />
  <xs:attribute name="name" type="xs:string" />
  <xs:attribute name="eventId" type="xs:int" use="required" />
  <xs:attribute name="startTime" type="xs:string" />
  <xs:attribute name="endTime" type="xs:string" />
  <xs:attribute name="comment" type="xs:string" />
  <xs:attribute name="isActive" type="xs:boolean" use="required" />
  <xs:attribute name="locationId" type="xs:int" use="required" />
  <xs:attribute name="vehicleId" type="xs:int" use="required" />
  <xs:attribute name="restFlag" type="xs:int" use="required" />
  <xs:attribute name="addDate" type="xs:string" />
  <xs:attribute name="changeDate" type="xs:string" />
</xs:complexType>
```

scheduleException

```
<xs:complexType name="scheduleException">
  <xs:complexContent>
    <xs:extension base="scheduleRecord">
      <xs:sequence />
      <xs:attribute name="coId" type="xs:long" use="required" />
      <xs:attribute name="estimatedEndTime" type="xs:string" />
      <xs:attribute name="releaseFlag" type="xs:int" use="required" />
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

scheduleRequestRecord

```
<xs:complexType name="scheduleRequestRecord">
```

```
  <xs:sequence />
  <xs:attribute name="schedRequestId" type="xs:int" use="required" />
  <xs:attribute name="memexId" type="xs:int" use="required" />
  <xs:attribute name="contactId" type="xs:int" use="required" />
  <xs:attribute name="name" type="xs:string" />
  <xs:attribute name="eventId" type="xs:int" use="required" />
  <xs:attribute name="startDate" type="xs:dateTime" />
  <xs:attribute name="endDate" type="xs:dateTime" />
  <xs:attribute name="requestStatus" type="xs:string" />
  <xs:attribute name="isDeleted" type="xs:boolean" use="required" />
  <xs:attribute name="declineReason" type="xs:string" />
  <xs:attribute name="comments" type="xs:string" />
  <xs:attribute name="notifiedDateString" type="xs:string" />
  <xs:attribute name="eventAbbrev" type="xs:string" />
  <xs:attribute name="tzName" type="xs:string" />
  <xs:attribute name="empName" type="xs:string" />
  <xs:attribute name="empLocId" type="xs:int" use="required" />
  <xs:attribute name="empLocName" type="xs:string" />
  <xs:attribute name="addDate" type="xs:dateTime" />
  <xs:attribute name="startDateString" type="xs:string" />
  <xs:attribute name="endDateString" type="xs:string" />
 </xs:complexType>
```

scheduleEvent

```
 <xs:complexType name="scheduleEvent">
  <xs:sequence>
   <xs:element name="eventTraits" minOccurs="0">
    <xs:complexType>
     <xs:sequence>
      <xs:element name="eventTrait" type="eventTraitValue" minOccurs="0" maxOccurs="unbounded" />
     </xs:sequence>
    </xs:complexType>
   </xs:element>
   <xs:element name="isEnabled" type="xs:int" />
  </xs:sequence>
  <xs:attribute name="eventId" type="xs:int" use="required" />
  <xs:attribute name="description" type="xs:string" />
  <xs:attribute name="abbreviation" type="xs:string" />
  <xs:attribute name="recording" type="xs:int" use="required" />
 </xs:complexType>
```

ARCOS LLC
8800 Lyra Drive, Suite 200
Columbus, OH 43240
(614) 396 – 5500

Crew Manager Data Structures

Group

```
<xs:complexType name="group">
  <xs:complexContent>
    <xs:extension base="attributeElement">
      <xs:sequence />
      <xs:attribute name="viewId" type="xs:int" />
      <xs:attribute name="crewIds">
        <xs:simpleType>
          <xs:list itemType="xs:int" />
        </xs:simpleType>
      </xs:attribute>
      <xs:attribute name="memberCnt" type="xs:int" use="required" />
      <xs:attribute name="bgColor" type="xs:string" />
      <xs:attribute name="fontColor" type="xs:string" />
      <xs:attribute name="borderColor" type="xs:string" />
      <xs:attribute name="chartAttributes">
        <xs:simpleType>
          <xs:list itemType="xs:string" />
        </xs:simpleType>
      </xs:attribute>
      <xs:attribute name="ord" type="xs:string" />
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

roomCapacity

```
<xs:complexType name="roomCapacity">
  <xs:sequence />
  <xs:attribute name="roomType" type="xs:int" use="required" />
  <xs:attribute name="quantity" type="xs:int" use="required" />
  <xs:attribute name="rate" type="xs:double" use="required" />
</xs:complexType>
```

workTicket

```
<xs:complexType name="workTicket">
  <xs:sequence />
  <xs:attribute name="id" type="xs:int" />
  <xs:attribute name="crewId" type="xs:int" />
  <xs:attribute name="description" type="xs:string" />
  <xs:attribute name="start" type="xs:string" />
  <xs:attribute name="end" type="xs:string" />
</xs:complexType>
<xs:complexType name="member">
  <xs:complexContent>
    <xs:extension base="attributeElement">
      <xs:sequence>
        <xs:element name="scheduleRecords" minOccurs="0">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="scheduleRecord" type="memberScheduleRecord" minOccurs="0" maxOccurs="unbounded" />
            </xs:sequence>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
      <xs:attribute name="crewId" type="xs:int" />
      <xs:attribute name="siteId" type="xs:int" />
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

memberScheduleRecord

```
<xs:complexType name="memberScheduleRecord">
  <xs:sequence />
  <xs:attribute name="memexId" type="xs:int" />
  <xs:attribute name="eventId" type="xs:int" />
  <xs:attribute name="description" type="xs:string" />
  <xs:attribute name="comment" type="xs:string" />
  <xs:attribute name="startTime" type="xs:string" />
  <xs:attribute name="endTime" type="xs:string" />
</xs:complexType>
```

Crew

```
<xs:complexType name="crew">
  <xs:complexContent>
    <xs:extension base="attributeElement">
      <xs:sequence>
        <xs:element name="tickets" minOccurs="0">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="ticket" type="workTicket" minOccurs="0" maxOccurs="unbounded" />
            </xs:sequence>
          </xs:complexType>
        </xs:element>
        <xs:element name="memberCnt" type="xs:int" />
      </xs:sequence>
      <xs:attribute name="bgColor" type="xs:string" />
      <xs:attribute name="fontColor" type="xs:string" />
      <xs:attribute name="nameBgColor" type="xs:string" />
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

Lodging

```
<xs:complexType name="lodging">
  <xs:complexContent>
    <xs:extension base="attributeElement">
      <xs:sequence>
        <xs:element name="rooms" minOccurs="0">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="capacity" type="roomCapacity" minOccurs="0" maxOccurs="unbounded" />
            </xs:sequence>
          </xs:complexType>
        </xs:element>
        <xs:element name="availabilityList" minOccurs="0">
          <xs:complexType>
            <xs:sequence>
              <xs:element ref="availability" minOccurs="0" maxOccurs="unbounded" />
            </xs:sequence>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

Availability

```
<xs:complexType name="availability">
  <xs:sequence />
  <xs:attribute name="id" type="xs:int" />
  <xs:attribute name="lodgingId" type="xs:int" />
  <xs:attribute name="roomTypeId" type="xs:int" />
  <xs:attribute name="date" type="xs:string" />
  <xs:attribute name="available" type="xs:int" use="required" />
  <xs:attribute name="booked" type="xs:int" use="required" />
  <xs:attribute name="rate" type="xs:double" use="required" />
  <xs:attribute name="note" type="xs:string" />
  <xs:attribute name="changeDate" type="xs:string" />
  <xs:attribute name="user" type="xs:string" />
</xs:complexType>
```

roomType

```
<xs:complexType name="roomType">
  <xs:complexContent>
    <xs:extension base="attributeElement">
      <xs:all />
      <xs:attribute name="size" type="xs:int" use="required" />
      <xs:attribute name="name" type="xs:string" />
      <xs:attribute name="abbreviation" type="xs:string" />
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

Resource

```
<xs:complexType name="resource">
  <xs:complexContent>
    <xs:extension base="attributeElement">
      <xs:sequence />
      <xs:attribute name="resourceType" type="xs:string" />
      <xs:attribute name="resourceTypeId" type="xs:int" />
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

classResourceTypes

```
<xs:complexType name="classResourceTypes">
  <xs:sequence />
  <xs:attribute name="resourceId" type="xs:int" use="required" />
  <xs:attribute name="name" type="xs:string" />
  <xs:attribute name="description" type="xs:string" />
  <xs:attribute name="customFlag" type="xs:int" use="required" />
</xs:complexType>
```

workOrder

```
<xs:complexType name="workOrder">
  <xs:complexContent>
    <xs:extension base="resource">
      <xs:sequence>
        <xs:element name="workOrderTimeEntries" minOccurs="0">
          <xs:complexType>
            <xs:sequence>
              <xs:element ref="workOrderTimeEntry" minOccurs="0" maxOccurs="unbounded" />
            </xs:sequence>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

workOrderTimeEntry

```
<xs:complexType name="workOrderTimeEntry">
  <xs:sequence />
  <xs:attribute name="id" type="xs:int" />
  <xs:attribute name="changeId" type="xs:int" />
  <xs:attribute name="crewId" type="xs:int" />
  <xs:attribute name="start" type="xs:string" />
  <xs:attribute name="end" type="xs:string" />
  <xs:attribute name="hours" type="xs:double" />
  <xs:attribute name="comments" type="xs:string" />
  <xs:attribute name="changeDate" type="xs:string" />
</xs:complexType>
```

Assignment

```
<xs:complexType name="assignment">
  <xs:choice>
    <xs:element ref="attributeElement" />
    <xs:element ref="crewGroup" />
    <xs:element ref="crew" />
    <xs:element ref="member" />
    <xs:element ref="lodging" />
    <xs:element ref="roomType" />
    <xs:element ref="resource" />
    <xs:element ref="workOrder" />
  </xs:choice>
  <xs:attribute name="assignId" type="xs:int" />
  <xs:attribute name="label" type="xs:string" />
  <xs:attribute name="assignStart" type="xs:string" />
  <xs:attribute name="assignEnd" type="xs:string" />
  <xs:attribute name="changeDate" type="xs:string" />
</xs:complexType>
```

attributeRecord

```
<xs:complexType name="attributeRecord">
  <xs:sequence />
  <xs:attribute name="name" type="xs:string" />
  <xs:attribute name="value" type="xs:string" />
  <xs:attribute name="start" type="xs:string" />
  <xs:attribute name="end" type="xs:string" />
  <xs:attribute name="changeId" type="xs:int" />
  <xs:attribute name="changeDate" type="xs:string" />
  <xs:attribute name="bgColor" type="xs:string" />
  <xs:attribute name="badgeText" type="xs:string" />
  <xs:attribute name="image" type="xs:string" />
  <xs:attribute name="ord" type="xs:string" />
  <xs:attribute name="fontColor" type="xs:string" />
  <xs:attribute name="borderColor" type="xs:string" />
  <xs:attribute name="chartBgColors" type="xs:string" />
</xs:complexType>
```

attributeCollection

```
<xs:complexType name="attributeCollection">
  <xs:sequence>
    <xs:element name="attributes" minOccurs="0">
      <xs:complexType>
        <xs:sequence>
          <xs:element ref="attributeRecord" minOccurs="0" maxOccurs="unbounded" />
        </xs:sequence>
      </xs:complexType>
    </xs:element>
  </xs:sequence>
  <xs:attribute name="targetType" type="xs:string" />
  <xs:attribute name="id" type="xs:int" />
  <xs:attribute name="name" type="xs:string" />
  <xs:attribute name="searchType" type="xs:string" />
  <xs:attribute name="searchValue" type="xs:string" />
</xs:complexType>
```

attributeElement

```
<xs:complexType name="attributeElement" abstract="true">
  <xs:sequence>
    <xs:element name="attributes" minOccurs="0">
      <xs:complexType>
        <xs:sequence>
          <xs:element ref="attributeRecord" minOccurs="0" maxOccurs="unbounded" />
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="assignments" minOccurs="0">
      <xs:complexType>
        <xs:sequence>
          <xs:element ref="assignment" minOccurs="0" maxOccurs="unbounded" />
        </xs:sequence>
      </xs:complexType>
    </xs:element>
  </xs:sequence>
  <xs:attribute name="id" type="xs:int" />
  <xs:attribute name="name" type="xs:string" />
  <xs:attribute name="createdDate" type="xs:string" />
  <xs:attribute name="deletedDate" type="xs:string" />
  <xs:attribute name="changedDate" type="xs:string" />
  <xs:attribute name="searchType" type="xs:string" />
  <xs:attribute name="searchValue" type="xs:string" />
  <xs:attribute name="delete" type="xs:boolean" />
</xs:complexType>
```

attributeDefinition

```
<xs:complexType name="attributeDefinition">
  <xs:sequence>
    <xs:element name="properties" type="mapElementArray" minOccurs="0" />
    <xs:element name="values" minOccurs="0">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="value" type="attributeValue" minOccurs="0" maxOccurs="unbounded" />
        </xs:sequence>
      </xs:complexType>
    </xs:element>
  </xs:sequence>
  <xs:attribute name="target" type="xs:string" />
  <xs:attribute name="name" type="xs:string" />
  <xs:attribute name="addDate" type="xs:string" />
</xs:complexType>
```

attributeValue

```
<xs:complexType name="attributeValue">
  <xs:sequence>
    <xs:element name="dataPoints" type="mapElementArray" minOccurs="0" />
  </xs:sequence>
  <xs:attribute name="value" type="xs:string" />
  <xs:attribute name="ord" type="xs:string" />
  <xs:attribute name="bgColor" type="xs:string" />
  <xs:attribute name="badgeText" type="xs:string" />
  <xs:attribute name="image" type="xs:string" />
  <xs:attribute name="fontColor" type="xs:string" />
  <xs:attribute name="borderColor" type="xs:string" />
</xs:complexType>
```

savedSet

```
<xs:complexType name="savedSet">
  <xs:sequence>
    <xs:element name="propertyMap" type="mapElementArray" minOccurs="0" />
  </xs:sequence>
  <xs:attribute name="id" type="xs:int" use="required" />
  <xs:attribute name="userId" type="xs:int" use="required" />
  <xs:attribute name="isPublic" type="xs:boolean" use="required" />
  <xs:attribute name="userName" type="xs:string" />
  <xs:attribute name="ownerId" type="xs:int" use="required" />
  <xs:attribute name="ownerName" type="xs:string" />
  <xs:attribute name="name" type="xs:string" />
  <xs:attribute name="type" type="xs:string" />
  <xs:attribute name="specialFlag" type="xs:string" />
  <xs:attribute name="changeDate" type="xs:string" />
</xs:complexType>
```

crewManagerData

```
<xs:complexType name="crewManagerData">
  <xs:sequence>
    <xs:element name="groups" minOccurs="0">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="group" type="group" minOccurs="0" maxOccurs="unbounded" />
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="lodgings" minOccurs="0">
      <xs:complexType>
        <xs:sequence>
          <xs:element ref="lodging" minOccurs="0" maxOccurs="unbounded" />
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="resources" minOccurs="0">
      <xs:complexType>
        <xs:sequence>
          <xs:element ref="resource" minOccurs="0" maxOccurs="unbounded" />
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="attributes" minOccurs="0">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="attribute" type="attributeDefinition" minOccurs="0" maxOccurs="unbounded" />
        </xs:sequence>
      </xs:complexType>
    </xs:element>
  </xs:sequence>
</xs:complexType>
```

Filter

```
<xs:complexType name="filter">
  <xs:complexContent>
    <xs:extension base="savedSet">
      <xs:sequence>
        <xs:element name="entries" minOccurs="0">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="entry" type="filterEntry" minOccurs="0" maxOccurs="unbounded" />
            </xs:sequence>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

filterEntry

```
<xs:complexType name="filterEntry">
  <xs:sequence>
    <xs:element name="values" minOccurs="0">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="values" type="xs:string" minOccurs="0" maxOccurs="unbounded" />
        </xs:sequence>
      </xs:complexType>
    </xs:element>
  </xs:sequence>
  <xs:attribute name="id" type="xs:string" />
</xs:complexType>
```

viewSet

```
<xs:complexType name="viewSet">
  <xs:complexContent>
    <xs:extension base="savedSet">
      <xs:sequence>
        <xs:element name="targets" minOccurs="0">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="target" type="xs:string" minOccurs="0" maxOccurs="unbounded" />
            </xs:sequence>
          </xs:complexType>
        </xs:element>
        <xs:element name="selectedAttributes" minOccurs="0">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="attribute" type="viewAttribute" minOccurs="0" maxOccurs="unbounded" />
            </xs:sequence>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

viewAttribute

```
<xs:complexType name="viewAttribute">
  <xs:sequence />
  <xs:attribute name="targetAdo" type="xs:string" />
  <xs:attribute name="badge" type="xs:int" use="required" />
  <xs:attribute name="icon" type="xs:int" use="required" />
  <xs:attribute name="tab" type="xs:int" use="required" />
  <xs:attribute name="bar" type="xs:int" use="required" />
  <xs:attribute name="pie" type="xs:int" use="required" />
  <xs:attribute name="list" type="xs:string" />
</xs:complexType>
```

resourceTypeDefinition

```
<xs:complexType name="resourceTypeDefinition">
  <xs:sequence />
  <xs:attribute name="id" type="xs:int" />
  <xs:attribute name="name" type="xs:string" />
  <xs:attribute name="description" type="xs:string" />
</xs:complexType>
```

Perspective

```
<xs:complexType name="perspective">
  <xs:complexContent>
    <xs:extension base="savedSet">
      <xs:sequence>
        <xs:element name="filters" minOccurs="0">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="filter" type="xs:string" minOccurs="0" maxOccurs="unbounded" />
            </xs:sequence>
          </xs:complexType>
        </xs:element>
        <xs:element name="views" minOccurs="0">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="view" type="xs:string" minOccurs="0" maxOccurs="unbounded" />
            </xs:sequence>
          </xs:complexType>
        </xs:element>
        <xs:element name="resourceTypes" minOccurs="0">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="resourceTypes" type="xs:string" minOccurs="0" maxOccurs="unbounded" />
            </xs:sequence>
          </xs:complexType>
        </xs:element>
        <xs:element name="notes" minOccurs="0">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="notes" type="xs:string" minOccurs="0" maxOccurs="unbounded" />
            </xs:sequence>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

Checkpoint Data Structures

bulkCheckin

```
<xs:complexType name="bulkCheckin">
  <xs:sequence />
  <xs:attribute name="contactIds">
    <xs:simpleType>
      <xs:list itemType="xs:int" />
    </xs:simpleType>
  </xs:attribute>
  <xs:attribute name="checkpointIds">
    <xs:simpleType>
      <xs:list itemType="xs:int" />
    </xs:simpleType>
  </xs:attribute>
  <xs:attribute name="timestamps">
    <xs:simpleType>
      <xs:list itemType="xs:string" />
    </xs:simpleType>
  </xs:attribute>
  <xs:attribute name="userId" type="xs:int" use="required" />
</xs:complexType>
```

Checkpoint

```
<xs:complexType name="checkpoint">
  <xs:sequence />
  <xs:attribute name="checkpointId" type="xs:int" use="required" />
  <xs:attribute name="eventId" type="xs:int" use="required" />
  <xs:attribute name="name" type="xs:string" />
  <xs:attribute name="description" type="xs:string" />
  <xs:attribute name="isActive" type="xs:boolean" use="required" />
  <xs:attribute name="eventName" type="xs:string" />
  <xs:attribute name="activatedOn" type="xs:string" />
  <xs:attribute name="validScans" type="xs:int" use="required" />
  <xs:attribute name="invalidScans" type="xs:int" use="required" />
  <xs:attribute name="templateId" type="xs:int" use="required" />
  <xs:attribute name="status" type="checkpointStatus" />
</xs:complexType>
```

checkpointActivity

```
<xs:complexType name="checkpointActivity">
  <xs:sequence />
  <xs:attribute name="activityId" type="xs:int" use="required" />
  <xs:attribute name="checkpointId" type="xs:int" use="required" />
  <xs:attribute name="resourceType" type="resourceType" />
  <xs:attribute name="resourceId" type="xs:int" use="required" />
  <xs:attribute name="result" type="xs:int" use="required" />
  <xs:attribute name="entryType" type="entryType" />
  <xs:attribute name="scannedBy" type="xs:int" use="required" />
  <xs:attribute name="createdOn" type="xs:string" />
  <xs:attribute name="salt" type="xs:string" />
</xs:complexType>
```

checkpointAssignment

```
<xs:complexType name="checkpointAssignment">
```

```
  <xs:sequence />
  <xs:attribute name="contactId" type="xs:int" use="required" />
  <xs:attribute name="checkpointId" type="xs:int" use="required" />
  <xs:attribute name="isActive" type="xs:boolean" use="required" />
  <xs:attribute name="activatedOn" type="xs:string" />
  <xs:attribute name="deactivatedOn" type="xs:string" />
  <xs:attribute name="createdBy" type="xs:int" use="required" />
  <xs:attribute name="createdOn" type="xs:string" />
  <xs:attribute name="updatedBy" type="xs:int" use="required" />
  <xs:attribute name="updatedOn" type="xs:string" />
</xs:complexType>
```

checkpointTemplate

```
<xs:complexType name="checkpointTemplate">
  <xs:sequence />
  <xs:attribute name="templateId" type="xs:int" use="required" />
  <xs:attribute name="name" type="xs:string" />
  <xs:attribute name="description" type="xs:string" />
  <xs:attribute name="isActive" type="xs:boolean" use="required" />
  <xs:attribute name="createdOn" type="xs:string" />
</xs:complexType>
```

checkpointStatus

```
<xs:simpleType name="checkpointStatus">
  <xs:restriction base="xs:string">
    <xs:enumeration value="Deleted" />
    <xs:enumeration value="Active" />
    <xs:enumeration value="Archived" />
  </xs:restriction>
</xs:simpleType>
```

resourceType

```
<xs:simpleType name="resourceType">
  <xs:restriction base="xs:string">
    <xs:enumeration value="Employee" />
  </xs:restriction>
</xs:simpleType>
```

entryType

```
<xs:simpleType name="entryType">
  <xs:restriction base="xs:string">
    <xs:enumeration value="Scanned" />
    <xs:enumeration value="Manual" />
  </xs:restriction>
</xs:simpleType>
```

ARCOS LLC
8800 Lyra Drive, Suite 200
Columbus, OH 43240
(614) 396 – 5500

Resource Assist Data Structures

raDocument

```
<xs:complexType name="raDocument">
   <xs:sequence>
      <xs:element name="user" type="attribute" minOccurs="0" />
      <xs:element name="storage" type="cloudStorage" minOccurs="0" />
   </xs:sequence>
   <xs:attribute name="id" type="xs:int" use="required" />
   <xs:attribute name="requestId" type="xs:int" use="required" />
   <xs:attribute name="attachmentType" type="xs:int" use="required" />
   <xs:attribute name="deleted" type="xs:int" use="required" />
   <xs:attribute name="fileSize" type="xs:int" use="required" />
   <xs:attribute name="displayName" type="xs:string" />
   <xs:attribute name="fileName" type="xs:string" />
   <xs:attribute name="description" type="xs:string" />
   <xs:attribute name="addDate" type="xs:dateTime" />
</xs:complexType>
```

ARCOS LLC
8800 Lyra Drive, Suite 200
Columbus, OH 43240
(614) 396 – 5500

Incident Manager Data Structures

cloudStorage

```
<xs:complexType name="cloudStorage">
   <xs:sequence />
   <xs:attribute name="type" type="xs:string" />
   <xs:attribute name="id" type="xs:int" use="required" />
   <xs:attribute name="stringId" type="xs:string" />
   <xs:attribute name="name" type="xs:string" />
   <xs:attribute name="service" type="xs:string" />
   <xs:attribute name="region" type="xs:string" />
   <xs:attribute name="bucket" type="xs:string" />
   <xs:attribute name="hash" type="xs:string" />
   <xs:attribute name="added" type="xs:dateTime" />
</xs:complexType>
```

Document

```
<xs:complexType name="document">
   <xs:sequence>
      <xs:element name="storage" type="cloudStorage" minOccurs="0" />
      <xs:element name="event" type="attribute" minOccurs="0" />
      <xs:element name="user" type="attribute" minOccurs="0" />
      <xs:element name="roles" minOccurs="0">
         <xs:complexType>
            <xs:sequence>
               <xs:element name="role" type="attribute" minOccurs="0" maxOccurs="unbounded" />
            </xs:sequence>
         </xs:complexType>
      </xs:element>
      <xs:element name="link" type="xs:string" minOccurs="0" />
   </xs:sequence>
   <xs:attribute name="id" type="xs:int" use="required" />
   <xs:attribute name="name" type="xs:string" />
   <xs:attribute name="description" type="xs:string" />
   <xs:attribute name="template" type="xs:boolean" use="required" />
   <xs:attribute name="version" type="xs:int" use="required" />
   <xs:attribute name="parentId" type="xs:int" />
   <xs:attribute name="respondent" type="xs:string" />
   <xs:attribute name="added" type="xs:dateTime" />
   <xs:attribute name="active" type="xs:boolean" use="required" />
</xs:complexType>
```

documentRoute

```
<xs:complexType name="documentRoute">
   <xs:sequence>
      <xs:element name="document" type="attribute" minOccurs="0" />
      <xs:element name="role" type="attribute" minOccurs="0" />
      <xs:element name="email" type="routeEmail" minOccurs="0" />
   </xs:sequence>
   <xs:attribute name="id" type="xs:int" use="required" />
   <xs:attribute name="act" type="xs:int" use="required" />
   <xs:attribute name="targetType" type="xs:int" use="required" />
   <xs:attribute name="targetVal" type="xs:string" />
   <xs:attribute name="startDate" type="xs:dateTime" />
   <xs:attribute name="endDate" type="xs:dateTime" />
   <xs:attribute name="lastSent" type="xs:dateTime" />
   <xs:attribute name="recurMins" type="xs:int" use="required" />
   <xs:attribute name="active" type="xs:boolean" use="required" />
   <xs:attribute name="emailFlag" type="xs:int" use="required" />
   <xs:attribute name="notificationFlag" type="xs:int" use="required" />
</xs:complexType>
```

routeEmail

```
<xs:complexType name="routeEmail">
   <xs:sequence>
      <xs:element name="emails" type="xs:string" minOccurs="0" />
   </xs:sequence>
   <xs:attribute name="subject" type="xs:string" />
   <xs:attribute name="replyTo" type="xs:string" />
   <xs:attribute name="message" type="xs:string" />
</xs:complexType>
```

emCrew

```
<xs:complexType name="emCrew">
   <xs:sequence />
   <xs:attribute name="id" type="xs:int" use="required" />
   <xs:attribute name="name" type="xs:string" />
   <xs:attribute name="addDate" type="xs:string" />
   <xs:attribute name="createDate" type="xs:string" />
</xs:complexType>
```

emLocation

```
<xs:complexType name="emLocation">
  <xs:sequence>
    <xs:element name="category" type="locationCategory" minOccurs="0" />
  </xs:sequence>
  <xs:attribute name="id" type="xs:int" use="required" />
  <xs:attribute name="name" type="xs:string" />
  <xs:attribute name="address" type="xs:string" />
  <xs:attribute name="formattedAddress" type="xs:string" />
  <xs:attribute name="lat" type="xs:string" />
  <xs:attribute name="lon" type="xs:string" />
  <xs:attribute name="phone" type="xs:string" />
  <xs:attribute name="messageId" type="xs:int" use="required" />
  <xs:attribute name="displayOrder" type="xs:int" use="required" />
  <xs:attribute name="changeDate" type="xs:string" />
</xs:complexType>
```

locationCategory

```
<xs:complexType name="locationCategory">
  <xs:sequence />
  <xs:attribute name="id" type="xs:int" use="required" />
  <xs:attribute name="name" type="xs:string" />
</xs:complexType>
```

emMassCall

```
<xs:complexType name="emMassCall">
  <xs:sequence />
  <xs:attribute name="id" type="xs:int" use="required" />
  <xs:attribute name="name" type="xs:string" />
  <xs:attribute name="status" type="xs:string" />
</xs:complexType>
```

emProcess

```
<xs:complexType name="emProcess">
  <xs:sequence>
    <xs:element name="role" type="attribute" minOccurs="0" />
    <xs:element name="responseGroup" type="attribute" minOccurs="0" />
    <xs:element name="steps" minOccurs="0">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="steps" type="processStep" minOccurs="0" maxOccurs="unbounded" />
        </xs:sequence>
      </xs:complexType>
    </xs:element>
  </xs:sequence>
  <xs:attribute name="id" type="xs:int" use="required" />
  <xs:attribute name="name" type="xs:string" />
  <xs:attribute name="description" type="xs:string" />
  <xs:attribute name="previewStartDate" type="xs:string" />
</xs:complexType>
```

processStep

```
<xs:complexType name="processStep">
   <xs:sequence>
      <xs:element name="attributes" minOccurs="0">
         <xs:complexType>
            <xs:sequence>
               <xs:element name="attribute" type="stepAttribute" minOccurs="0" maxOccurs="unbounded" />
            </xs:sequence>
         </xs:complexType>
      </xs:element>
   </xs:sequence>
   <xs:attribute name="id" type="xs:int" use="required" />
   <xs:attribute name="processId" type="xs:int" use="required" />
   <xs:attribute name="type" type="xs:int" use="required" />
   <xs:attribute name="description" type="xs:string" />
   <xs:attribute name="order" type="xs:int" use="required" />
   <xs:attribute name="details" type="xs:string" />
</xs:complexType>
```

stepAttribute

```
<xs:complexType name="stepAttribute">
   <xs:sequence />
   <xs:attribute name="stepId" type="xs:int" use="required" />
   <xs:attribute name="type" type="xs:string" />
   <xs:attribute name="attrId" type="xs:int" use="required" />
   <xs:attribute name="value" type="xs:string" />
   <xs:attribute name="order" type="xs:int" use="required" />
</xs:complexType>
```

Event

```
<xs:complexType name="event">
  <xs:sequence>
    <xs:element name="eventTeam" type="attribute" minOccurs="0" />
    <xs:element name="scenario" type="attribute" minOccurs="0" />
    <xs:element name="root" type="eventNode" minOccurs="0" />
    <xs:element name="phases" minOccurs="0">
      <xs:complexType>
        <xs:sequence>
          <xs:element ref="phase" minOccurs="0" maxOccurs="unbounded" />
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="acknowledgedContacts" minOccurs="0">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="contact" type="attribute" minOccurs="0" maxOccurs="unbounded" />
        </xs:sequence>
      </xs:complexType>
    </xs:element>
  </xs:sequence>
  <xs:attribute name="id" type="xs:int" use="required" />
  <xs:attribute name="name" type="xs:string" />
  <xs:attribute name="severity" type="xs:int" use="required" />
  <xs:attribute name="template" type="xs:boolean" use="required" />
  <xs:attribute name="addDate" type="xs:string" />
  <xs:attribute name="startDate" type="xs:string" />
  <xs:attribute name="endDate" type="xs:string" />
  <xs:attribute name="active" type="xs:boolean" use="required" />
  <xs:attribute name="changeDate" type="xs:string" />
  <xs:attribute name="notificationCo" type="xs:int" />
  <xs:attribute name="earliestActionDate" type="xs:string" />
</xs:complexType>
```

eventNode

```
<xs:complexType name="eventNode">
  <xs:sequence>
    <xs:element name="location" type="attribute" minOccurs="0" />
    <xs:element name="roleRequests" minOccurs="0">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="request" type="roleRequest" minOccurs="0" maxOccurs="unbounded" />
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="contacts" minOccurs="0">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="contact" type="contactInfo" minOccurs="0" maxOccurs="unbounded" />
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="children" type="eventNode" minOccurs="0" maxOccurs="unbounded" />
  </xs:sequence>
  <xs:attribute name="id" type="xs:int" use="required" />
  <xs:attribute name="eventId" type="xs:int" use="required" />
  <xs:attribute name="reportsTo" type="xs:int" use="required" />
  <xs:attribute name="roleId" type="xs:int" use="required" />
  <xs:attribute name="deputy" type="xs:boolean" use="required" />
  <xs:attribute name="prefix" type="xs:string" />
  <xs:attribute name="suffix" type="xs:string" />
  <xs:attribute name="state" type="xs:int" use="required" />
</xs:complexType>
```

roleRequest

```
<xs:complexType name="roleRequest">
   <xs:sequence>
      <xs:element name="event" type="attribute" minOccurs="0" />
      <xs:element name="status" type="attribute" minOccurs="0" />
      <xs:element name="role" type="attribute" minOccurs="0" />
      <xs:element name="employees" minOccurs="0">
         <xs:complexType>
            <xs:sequence>
               <xs:element name="employee" type="requestEmployee" minOccurs="0" maxOccurs="unbounded" />
            </xs:sequence>
         </xs:complexType>
      </xs:element>
      <xs:element name="attributes" minOccurs="0">
         <xs:complexType>
            <xs:sequence>
               <xs:element name="attribute" type="requestAttribute" minOccurs="0" maxOccurs="unbounded" />
            </xs:sequence>
         </xs:complexType>
      </xs:element>
   </xs:sequence>
   <xs:attribute name="id" type="xs:int" use="required" />
   <xs:attribute name="startDate" type="xs:string" />
   <xs:attribute name="endDate" type="xs:string" />
   <xs:attribute name="eventNode" type="xs:int" use="required" />
   <xs:attribute name="changedDate" type="xs:string" />
   <xs:attribute name="coMainId" type="xs:int" use="required" />
   <xs:attribute name="coId" type="xs:int" use="required" />
   <xs:attribute name="callAvailOnly" type="xs:boolean" use="required" />
   <xs:attribute name="quantity" type="xs:int" use="required" />
</xs:complexType>
```

requestEmployee

```
<xs:complexType name="requestEmployee">
   <xs:sequence>
      <xs:element name="employee" type="attribute" minOccurs="0" />
      <xs:element name="result" type="attribute" minOccurs="0" />
      <xs:element name="resultSource" type="attribute" minOccurs="0" />
      <xs:element name="team" type="attribute" minOccurs="0" />
      <xs:element name="schedule" type="attribute" minOccurs="0" />
   </xs:sequence>
   <xs:attribute name="requestId" type="xs:int" use="required" />
   <xs:attribute name="order" type="xs:int" use="required" />
   <xs:attribute name="resultDate" type="xs:string" />
   <xs:attribute name="calls" type="xs:int" use="required" />
   <xs:attribute name="emails" type="xs:int" use="required" />
   <xs:attribute name="texts" type="xs:int" use="required" />
   <xs:attribute name="notifications" type="xs:int" use="required" />
   <xs:attribute name="comment" type="xs:string" />
   <xs:attribute name="teamOrder" type="xs:int" use="required" />
   <xs:attribute name="userValue1" type="xs:string" />
   <xs:attribute name="userValue2" type="xs:string" />
</xs:complexType>
```

requestAttribute

```
<xs:complexType name="requestAttribute">
   <xs:sequence />
   <xs:attribute name="requestId" type="xs:int" use="required" />
   <xs:attribute name="attrId" type="xs:int" use="required" />
   <xs:attribute name="value" type="xs:string" />
</xs:complexType>
```

Phase

```
<xs:complexType name="phase">
   <xs:sequence>
      <xs:element name="activationRoles" type="xs:int" minOccurs="0" maxOccurs="unbounded" />
      <xs:element name="deactivationRoles" type="xs:int" minOccurs="0" maxOccurs="unbounded" />
   </xs:sequence>
   <xs:attribute name="id" type="xs:int" use="required" />
   <xs:attribute name="eventId" type="xs:int" use="required" />
   <xs:attribute name="name" type="xs:string" />
   <xs:attribute name="order" type="xs:int" use="required" />
</xs:complexType>
```

eventCategory

```
<xs:complexType name="eventCategory">
   <xs:sequence />
   <xs:attribute name="id" type="xs:int" use="required" />
   <xs:attribute name="name" type="xs:string" />
   <xs:attribute name="group" type="xs:string" />
   <xs:attribute name="addDate" type="xs:string" />
</xs:complexType>
```

eventHistory

```
<xs:complexType name="eventHistory">
   <xs:sequence>
```

```
      <xs:element name="event" type="attribute" minOccurs="0" />
      <xs:element name="transactions" type="eventTransaction" minOccurs="0" maxOccurs="unbounded" />
   </xs:sequence>
</xs:complexType>
```

eventTransaction

```
<xs:complexType name="eventTransaction">
   <xs:sequence>
      <xs:element name="user" type="attribute" minOccurs="0" />
   </xs:sequence>
   <xs:attribute name="transaction" type="xs:string" />
   <xs:attribute name="date" type="xs:string" />
   <xs:attribute name="source" type="xs:string" />
   <xs:attribute name="eventId" type="xs:int" use="required" />
</xs:complexType>
```

eventTeam

```
<xs:complexType name="eventTeam">
   <xs:sequence>
      <xs:element name="employees" minOccurs="0">
         <xs:complexType>
            <xs:sequence>
               <xs:element name="employeePerm" type="teamPermission" minOccurs="0" maxOccurs="unbounded" />
            </xs:sequence>
         </xs:complexType>
      </xs:element>
      <xs:element ref="readyCheckConfig" minOccurs="0" />
      <xs:element name="tz" type="attribute" minOccurs="0" />
      <xs:element name="ignoredRoles" type="xs:int" minOccurs="0" maxOccurs="unbounded" />
      <xs:element name="ignoredFunctions" type="xs:int" minOccurs="0" maxOccurs="unbounded" />
   </xs:sequence>
   <xs:attribute name="id" type="xs:int" use="required" />
   <xs:attribute name="name" type="xs:string" />
   <xs:attribute name="teamDesc" type="xs:string" />
   <xs:attribute name="active" type="xs:boolean" use="required" />
   <xs:attribute name="activeEvents" type="xs:int" use="required" />
   <xs:attribute name="scenarios" type="xs:int" use="required" />
   <xs:attribute name="roles" type="xs:int" use="required" />
   <xs:attribute name="teams" type="xs:int" use="required" />
   <xs:attribute name="changeDate" type="xs:string" />
</xs:complexType>
```

teamPermission

```
<xs:complexType name="teamPermission">
   <xs:sequence>
      <xs:element name="permissions" type="xs:int" />
   </xs:sequence>
   <xs:attribute name="teamId" type="xs:int" use="required" />
   <xs:attribute name="contactId" type="xs:int" use="required" />
</xs:complexType>
```

readyCheckConfig

```
<xs:complexType name="readyCheckConfig">
   <xs:sequence>
      <xs:element name="incidentStartNotification" type="xs:int" />
      <xs:element name="autoInitiateRequest" type="xs:boolean" />
```

```
      <xs:element name="autoInitiateNextRequest" type="xs:boolean" />
      <xs:element name="onDeckMinutes" type="xs:int" />
      <xs:element name="sendDeactivation" type="xs:boolean" />
      <xs:element name="requireAcknowledge" type="xs:boolean" />
      <xs:element name="calloutNotification" type="xs:boolean" />
      <xs:element name="smsNotification" type="xs:boolean" />
   </xs:sequence>
</xs:complexType>
```

eventWrapper

```
<xs:complexType name="eventWrapper">
   <xs:sequence>
      <xs:element name="incidents" minOccurs="0">
         <xs:complexType>
            <xs:sequence>
               <xs:element name="incident" type="event" minOccurs="0" maxOccurs="unbounded" />
            </xs:sequence>
         </xs:complexType>
      </xs:element>
      <xs:element name="crews" minOccurs="0">
         <xs:complexType>
            <xs:sequence>
               <xs:element name="crew" type="emCrew" minOccurs="0" maxOccurs="unbounded" />
            </xs:sequence>
         </xs:complexType>
      </xs:element>
      <xs:element name="stormcodes" minOccurs="0">
         <xs:complexType>
            <xs:sequence>
               <xs:element name="stormcodes" type="stormCode" minOccurs="0" maxOccurs="unbounded" />
            </xs:sequence>
         </xs:complexType>
      </xs:element>
      <xs:element name="callouts" minOccurs="0">
         <xs:complexType>
            <xs:sequence>
              <xs:element name="callouts" type="callout" minOccurs="0" maxOccurs="unbounded" />
            </xs:sequence>
         </xs:complexType>
      </xs:element>
      <xs:element name="masscalls" minOccurs="0">
         <xs:complexType>
            <xs:sequence>
               <xs:element name="masscalls" type="emMassCall" minOccurs="0" maxOccurs="unbounded" />
            </xs:sequence>
         </xs:complexType>
      </xs:element>
   </xs:sequence>
   <xs:attribute name="id" type="xs:int" use="required" />
   <xs:attribute name="name" type="xs:string" />
   <xs:attribute name="description" type="xs:string" />
   <xs:attribute name="deleted" type="xs:boolean" use="required" />
   <xs:attribute name="addDate" type="xs:string" />
   <xs:attribute name="closeDate" type="xs:string" />
   <xs:attribute name="closed" type="xs:boolean" use="required" />
   <xs:attribute name="categoryId" type="xs:int" use="required" />
```

```
    <xs:attribute name="email" type="xs:string" />
    <xs:attribute name="application" type="xs:string" />
    <xs:attribute name="startDate" type="xs:string" />
    <xs:attribute name="endDate" type="xs:string" />
    <xs:attribute name="categoryName" type="xs:string" />
    <xs:attribute name="categoryGroup" type="xs:string" />
</xs:complexType>
```

stormCode

```
<xs:complexType name="stormCode">
    <xs:sequence />
    <xs:attribute name="application" type="xs:string" />
    <xs:attribute name="enabled" type="xs:int" use="required" />
    <xs:attribute name="storm" type="xs:string" />
    <xs:attribute name="stormCode" type="xs:string" />
    <xs:attribute name="stormId" type="xs:int" use="required" />
    <xs:attribute name="wrapperId" type="xs:int" use="required" />
</xs:complexType>
```

icsFunction

```
<xs:complexType name="icsFunction">
    <xs:sequence>
        <xs:element name="eventTeam" type="attribute" minOccurs="0" />
    </xs:sequence>
    <xs:attribute name="id" type="xs:int" use="required" />
    <xs:attribute name="name" type="xs:string" />
    <xs:attribute name="color" type="xs:string" />
    <xs:attribute name="active" type="xs:boolean" use="required" />
</xs:complexType>
```

Level

```
<xs:complexType name="level">
   <xs:sequence>
      <xs:element name="eventTeam" type="attribute" minOccurs="0" />
      <xs:element name="roles" minOccurs="0">
         <xs:complexType>
            <xs:sequence>
               <xs:element ref="role" minOccurs="0" maxOccurs="unbounded" />
            </xs:sequence>
         </xs:complexType>
      </xs:element>
   </xs:sequence>
   <xs:attribute name="id" type="xs:int" use="required" />
   <xs:attribute name="name" type="xs:string" />
   <xs:attribute name="logicalLevel" type="xs:int" use="required" />
   <xs:attribute name="leaderName" type="xs:string" />
   <xs:attribute name="assistantName" type="xs:string" />
   <xs:attribute name="deletedFlag" type="xs:int" use="required" />
</xs:complexType>
```

Role

```
<xs:complexType name="role">
   <xs:sequence>
      <xs:element name="eventTeam" type="attribute" minOccurs="0" />
   </xs:sequence>
   <xs:attribute name="id" type="xs:int" use="required" />
   <xs:attribute name="employeeCount" type="xs:int" use="required" />
   <xs:attribute name="templateCount" type="xs:int" use="required" />
   <xs:attribute name="prefix" type="xs:string" />
   <xs:attribute name="heading" type="xs:string" />
   <xs:attribute name="leaderName" type="xs:string" />
   <xs:attribute name="assistantName" type="xs:string" />
   <xs:attribute name="nameMethod" type="xs:int" use="required" />
   <xs:attribute name="functionId" type="xs:int" use="required" />
   <xs:attribute name="levelId" type="xs:int" use="required" />
   <xs:attribute name="active" type="xs:boolean" use="required" />
   <xs:attribute name="displayOrder" type="xs:int" use="required" />
   <xs:attribute name="callAvailOnly" type="xs:boolean" use="required" />
</xs:complexType>
```

logEntryCategory

```
<xs:complexType name="logEntryCategory">
   <xs:sequence>
      <xs:element name="name" type="xs:string" minOccurs="0" />
      <xs:element name="level" type="xs:int" />
      <xs:element name="deleted" type="xs:int" />
   </xs:sequence>
   <xs:attribute name="id" type="xs:int" use="required" />
</xs:complexType>
```

nodeDocument

```
<xs:complexType name="nodeDocument">
  <xs:sequence>
    <xs:element name="documents" type="document" minOccurs="0" maxOccurs="unbounded" />
  </xs:sequence>
  <xs:attribute name="id" type="xs:int" use="required" />
</xs:complexType>
```

processTeam

```
<xs:complexType name="processTeam">
  <xs:sequence>
    <xs:element name="responseGroup" type="attribute" minOccurs="0" />
    <xs:element name="processes" minOccurs="0">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="process" type="emProcess" minOccurs="0" maxOccurs="unbounded" />
        </xs:sequence>
      </xs:complexType>
    </xs:element>
  </xs:sequence>
  <xs:attribute name="id" type="xs:int" use="required" />
  <xs:attribute name="name" type="xs:string" />
</xs:complexType>
```

requestContact

```
<xs:complexType name="requestContact">
  <xs:sequence>
    <xs:element name="event" type="attribute" minOccurs="0" />
    <xs:element name="requestStatus" type="attribute" minOccurs="0" />
    <xs:element name="contactResult" type="attribute" minOccurs="0" />
    <xs:element name="role" type="attribute" minOccurs="0" />
  </xs:sequence>
  <xs:attribute name="requestId" type="xs:int" use="required" />
  <xs:attribute name="contactId" type="xs:int" use="required" />
  <xs:attribute name="requestStartDate" type="xs:string" />
  <xs:attribute name="requestEndDate" type="xs:string" />
</xs:complexType>
```

Scenario

```
<xs:complexType name="scenario">
  <xs:sequence>
    <xs:element name="eventTeam" type="attribute" minOccurs="0" />
    <xs:element name="eventTemplates" type="xs:int" minOccurs="0" maxOccurs="unbounded" />
  </xs:sequence>
  <xs:attribute name="id" type="xs:int" use="required" />
  <xs:attribute name="name" type="xs:string" />
  <xs:attribute name="active" type="xs:boolean" use="required" />
  <xs:attribute name="changeDate" type="xs:string" />
</xs:complexType>
```

Settings

```
<xs:complexType name="settings">
   <xs:sequence />
   <xs:attribute name="severityLevels" type="xs:int" use="required" />
   <xs:attribute name="severityDescending" type="xs:boolean" use="required" />
   <xs:attribute name="severityMax" type="xs:int" use="required" />
</xs:complexType>
```

stateChange

```
<xs:complexType name="stateChange">
   <xs:sequence>
      <xs:element name="roles" minOccurs="0">
         <xs:complexType>
            <xs:sequence>
               <xs:element name="role" type="xs:int" minOccurs="0" maxOccurs="unbounded" />
            </xs:sequence>
         </xs:complexType>
      </xs:element>
   </xs:sequence>
   <xs:attribute name="id" type="xs:int" use="required" />
   <xs:attribute name="state" type="xs:int" use="required" />
   <xs:attribute name="status" type="xs:int" use="required" />
   <xs:attribute name="eventId" type="xs:int" use="required" />
</xs:complexType>
```

templateRoleCollection

```
<xs:complexType name="templateRoleCollection">
   <xs:sequence>
      <xs:element name="roles" type="xs:int" minOccurs="0" maxOccurs="unbounded" />
   </xs:sequence>
   <xs:attribute name="eventId" type="xs:int" use="required" />
   <xs:attribute name="scenarioId" type="xs:int" use="required" />
   <xs:attribute name="severity" type="xs:int" use="required" />
</xs:complexType>
```

events.Event

```
<xs:complexType name="events.Event">
  <xs:sequence>
    <xs:element name="incidents" minOccurs="0">
      <xs:complexType>
        <xs:sequence>
          <xs:element ref="incident" minOccurs="0" maxOccurs="unbounded" />
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="crews" minOccurs="0">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="crew" type="emCrew" minOccurs="0" maxOccurs="unbounded" />
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="stormcodes" minOccurs="0">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="stormcodes" type="stormCode" minOccurs="0" maxOccurs="unbounded" />
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="callouts" minOccurs="0">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="callouts" type="callout" minOccurs="0" maxOccurs="unbounded" />
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="masscalls" minOccurs="0">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="masscalls" type="emMassCall" minOccurs="0" maxOccurs="unbounded" />
        </xs:sequence>
      </xs:complexType>
    </xs:element>
  </xs:sequence>
  <xs:attribute name="id" type="xs:int" use="required" />
  <xs:attribute name="name" type="xs:string" />
  <xs:attribute name="description" type="xs:string" />
  <xs:attribute name="deleted" type="xs:boolean" use="required" />
  <xs:attribute name="addDate" type="xs:string" />
  <xs:attribute name="closeDate" type="xs:string" />
  <xs:attribute name="closed" type="xs:boolean" use="required" />
  <xs:attribute name="startDate" type="xs:string" />
  <xs:attribute name="endDate" type="xs:string" />
  <xs:attribute name="categoryId" type="xs:int" use="required" />
  <xs:attribute name="email" type="xs:string" />
  <xs:attribute name="application" type="xs:string" />
  <xs:attribute name="categoryName" type="xs:string" />
  <xs:attribute name="categoryGroup" type="xs:string" />
</xs:complexType>
```

Incident

```
<xs:complexType name="incident">
  <xs:sequence>
    <xs:element name="eventTeam" type="attribute" minOccurs="0" />
    <xs:element name="scenario" type="attribute" minOccurs="0" />
    <xs:element name="root" type="eventNode" minOccurs="0" />
    <xs:element name="phases" minOccurs="0">
      <xs:complexType>
        <xs:sequence>
          <xs:element ref="phase" minOccurs="0" maxOccurs="unbounded" />
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="acknowledgedContacts" minOccurs="0">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="contact" type="attribute" minOccurs="0" maxOccurs="unbounded" />
        </xs:sequence>
      </xs:complexType>
    </xs:element>
  </xs:sequence>
  <xs:attribute name="id" type="xs:int" use="required" />
  <xs:attribute name="name" type="xs:string" />
  <xs:attribute name="severity" type="xs:int" use="required" />
  <xs:attribute name="template" type="xs:boolean" use="required" />
  <xs:attribute name="addDate" type="xs:string" />
  <xs:attribute name="startDate" type="xs:string" />
  <xs:attribute name="endDate" type="xs:string" />
  <xs:attribute name="deleted" type="xs:boolean" use="required" />
  <xs:attribute name="changeDate" type="xs:string" />
  <xs:attribute name="notificationCo" type="xs:int" />
  <xs:attribute name="earliestActionDate" type="xs:string" />
</xs:complexType>
```

ARCOS LLC
8800 Lyra Drive, Suite 200
Columbus, OH 43240
(614) 396 – 5500

Metadata Structures

XWalk

```
<xs:complexType name="xwalk">
  <xs:sequence />
  <xs:attribute name="type" type="xs:string" />
  <xs:attribute name="arcosId" type="xs:string" />
  <xs:attribute name="custId" type="xs:string" />
</xs:complexType>
```

ARCOS LLC
8800 Lyra Drive, Suite 200
Columbus, OH 43240
(614) 396 – 5500

Miscellaneous Data Structures

mapElement

```
<xs:complexType name="mapElement">
  <xs:sequence />
  <xs:attribute name="name" type="xs:string" />
  <xs:attribute name="value" type="xs:string" />
</xs:complexType>
```

jsonObject

```
<xs:complexType name="jsonObject">
  <xs:complexContent>
    <xs:extension base="hashMap">
      <xs:sequence />
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

hashMap

```
<xs:complexType name="hashMap">
  <xs:complexContent>
    <xs:extension base="abstractMap">
      <xs:sequence />
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

abstractMap

```
<xs:complexType name="abstractMap" abstract="true">
  <xs:sequence />
</xs:complexType>
```

mapElementAttay

```
<xs:complexType name="mapElementArray" final="#all">
  <xs:sequence>
    <xs:element name="item" type="mapElement" minOccurs="0" maxOccurs="unbounded" nillable="true" />
  </xs:sequence>
</xs:complexType>
```

Session

```
<xs:complexType name="session">
  <xs:sequence />
  <xs:attribute name="hash" type="xs:string" />
  <xs:attribute name="company" type="xs:string" />
  <xs:attribute name="userId" type="xs:int" use="required" />
  <xs:attribute name="name" type="xs:string" />
  <xs:attribute name="logon" type="xs:dateTime" />
  <xs:attribute name="agent" type="xs:string" />
  <xs:attribute name="application" type="xs:string" />
  <xs:attribute name="superUser" type="xs:boolean" use="required" />
  <xs:attribute name="locationId" type="xs:int" use="required" />
  <xs:attribute name="locationName" type="xs:string" />
  <xs:attribute name="locationAccess" type="xs:boolean" use="required" />
  <xs:attribute name="className" type="xs:string" />
  <xs:attribute name="logonStr" type="xs:string" />
  <xs:attribute name="server" type="xs:string" />
</xs:complexType>
```

Product

```
<xs:complexType name="product">
  <xs:sequence />
  <xs:attribute name="name" type="xs:string" />
  <xs:attribute name="status" type="xs:string" />
  <xs:attribute name="description" type="xs:string" />
</xs:complexType>
```

eventTrait

```
<xs:complexType name="eventTrait">
  <xs:sequence />
  <xs:attribute name="eventTraitId" type="xs:int" use="required" />
  <xs:attribute name="name" type="xs:string" />
  <xs:attribute name="description" type="xs:string" />
</xs:complexType>
```

eventTraitValue

```
<xs:complexType name="eventTraitValue">
  <xs:sequence />
  <xs:attribute name="eventTraitId" type="xs:int" use="required" />
  <xs:attribute name="value" type="xs:int" use="required" />
</xs:complexType>
```

safetyDetails

```
<xs:complexType name="safetyDetails">
   <xs:sequence />
   <xs:attribute name="safetyNoticeId" type="xs:string" />
   <xs:attribute name="fullName" type="xs:string" />
   <xs:attribute name="contactId" type="xs:string" />
   <xs:attribute name="noticeRecievedDate" type="xs:string" />
   <xs:attribute name="noticeLocation" type="xs:string" />
   <xs:attribute name="homeLocation" type="xs:string" />
   <xs:attribute name="responseId" type="xs:string" />
   <xs:attribute name="phoneNumbers">
      <xs:simpleType>
         <xs:list itemType="xs:string" />
      </xs:simpleType>
   </xs:attribute>
   <xs:attribute name="description" type="xs:string" />
   <xs:attribute name="noticeSentDate" type="xs:string" />
   <xs:attribute name="viewEmpDetails" type="xs:boolean" use="required" />
   <xs:attribute name="reponseDesc" type="xs:string" />
</xs:complexType>
```

scheduleType

```
<xs:complexType name="scheduleType">
   <xs:sequence />
   <xs:attribute name="id" type="xs:int" use="required" />
   <xs:attribute name="name" type="xs:string" />
   <xs:attribute name="on" type="xs:int" use="required" />
</xs:complexType>
```

Location

```
<xs:complexType name="location">
   <xs:sequence>
      <xs:element name="childLocations" minOccurs="0">
         <xs:complexType>
            <xs:sequence>
               <xs:element ref="location" minOccurs="0" maxOccurs="unbounded" />
            </xs:sequence>
         </xs:complexType>
      </xs:element>
   </xs:sequence>
   <xs:attribute name="locationId" type="xs:int" use="required" />
   <xs:attribute name="name" type="xs:string" />
   <xs:attribute name="parentLocationId" type="xs:int" use="required" />
   <xs:attribute name="rank" type="xs:int" use="required" />
   <xs:attribute name="order" type="xs:int" use="required" />
   <xs:attribute name="timezone" type="xs:string" />
   <xs:attribute name="pagerDelay" type="xs:int" use="required" />
</xs:complexType>
```

locationExt

```
<xs:complexType name="locationExt">
  <xs:sequence>
    <xs:element name="xwalks" minOccurs="0">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="xwalk" type="xs:string" minOccurs="0" maxOccurs="unbounded" />
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="childLocations" minOccurs="0">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="location" type="locationExt" minOccurs="0" maxOccurs="unbounded" />
        </xs:sequence>
      </xs:complexType>
    </xs:element>
  </xs:sequence>
  <xs:attribute name="locationId" type="xs:int" use="required" />
  <xs:attribute name="name" type="xs:string" />
  <xs:attribute name="parentLocationId" type="xs:int" use="required" />
  <xs:attribute name="rank" type="xs:int" use="required" />
  <xs:attribute name="order" type="xs:int" use="required" />
  <xs:attribute name="timezone" type="xs:string" />
  <xs:attribute name="pagerDelay" type="xs:int" use="required" />
</xs:complexType>
```

Configs

```
<xs:complexType name="configs">
  <xs:sequence>
    <xs:element name="enableSafetyManagementNotification" type="xs:boolean" minOccurs="0" />
    <xs:element name="mapTimeout" type="xs:int" minOccurs="0" />
    <xs:element name="locationPingFreq" type="xs:int" minOccurs="0" />
    <xs:element name="captureMobileData" type="xs:boolean" minOccurs="0" />
    <xs:element name="captureWebData" type="xs:boolean" minOccurs="0" />
    <xs:element name="onlyTrackWorking" type="xs:boolean" minOccurs="0" />
    <xs:element name="available" type="xs:string" minOccurs="0" />
    <xs:element name="rest" type="xs:string" minOccurs="0" />
    <xs:element name="working" type="xs:string" minOccurs="0" />
    <xs:element name="exception" type="xs:string" minOccurs="0" />
    <xs:element name="defaultSafetyNoticeExp" type="xs:double" minOccurs="0" />
  </xs:sequence>
</xs:complexType>
```

userDetailConfiguration

```
<xs:complexType name="userDetailConfiguration">
   <xs:sequence>
      <xs:element name="name" type="xs:string" minOccurs="0" />
      <xs:element name="attributeId" type="xs:string" minOccurs="0" />
      <xs:element name="id" type="xs:int" minOccurs="0" />
      <xs:element name="order" type="xs:int" minOccurs="0" />
      <xs:element name="employeeView" type="xs:boolean" minOccurs="0" />
      <xs:element name="rosterView" type="xs:boolean" minOccurs="0" />
      <xs:element name="calloutsWorking" type="xs:boolean" minOccurs="0" />
      <xs:element name="calloutsAccepts" type="xs:boolean" minOccurs="0" />
      <xs:element name="adminView" type="xs:boolean" minOccurs="0" />
      <xs:element name="type" type="xs:string" minOccurs="0" />
   </xs:sequence>
</xs:complexType>
```

Message

```
<xs:complexType name="message">
   <xs:sequence>
      <xs:element name="category" type="messageCategory" minOccurs="0" />
      <xs:element name="audioText" type="xs:string" minOccurs="0" />
      <xs:element name="fullText" type="xs:string" minOccurs="0" />
      <xs:element name="condensedText" type="xs:string" minOccurs="0" />
      <xs:element name="lastModified" type="xs:string" minOccurs="0" />
      <xs:element name="voiceGender" type="xs:string" minOccurs="0" />
      <xs:element name="voiceRate" type="xs:string" minOccurs="0" />
      <xs:element name="numLocsUsing" type="xs:int" />
   </xs:sequence>
   <xs:attribute name="id" type="xs:int" use="required" />
   <xs:attribute name="description" type="xs:string" />
   <xs:attribute name="displayOrder" type="xs:int" use="required" />
   <xs:attribute name="purge" type="xs:boolean" use="required" />
   <xs:attribute name="filename" type="xs:string" />
   <xs:attribute name="fileLength" type="xs:long" use="required" />
   <xs:attribute name="duration" type="xs:string" />
   <xs:attribute name="lastUsed" type="xs:string" />
   <xs:attribute name="name" type="xs:string" />
</xs:complexType>
```

messageCategory

```
<xs:complexType name="messageCategory">
   <xs:sequence />
   <xs:attribute name="id" type="xs:int" use="required" />
   <xs:attribute name="name" type="xs:string" />
</xs:complexType>
```

jobStatus

```
<xs:complexType name="jobStatus">
   <xs:sequence>
      <xs:element name="responseBody" type="xs:string" minOccurs="0" />
   </xs:sequence>
   <xs:attribute name="uuid" type="xs:string" />
   <xs:attribute name="httpStatus" type="xs:string" />
</xs:complexType>
```

Subscription

```
<xs:complexType name="subscription">
   <xs:sequence />
   <xs:attribute name="topic" type="xs:string" />
   <xs:attribute name="mediaTypeString" type="xs:string" />
</xs:complexType>
```

systemEvent

```
<xs:complexType name="systemEvent">
   <xs:sequence>
      <xs:element name="url" type="xs:string" minOccurs="0" />
      <xs:element name="payload" type="xs:anyType" minOccurs="0" />
   </xs:sequence>
   <xs:attribute name="systemId" type="xs:string" />
   <xs:attribute name="topic" type="xs:string" />
   <xs:attribute name="id" type="xs:string" />
   <xs:attribute name="date" type="xs:string" />
   <xs:attribute name="actionString" type="xs:string" />
</xs:complexType>
```

apiCallLogData

```
<xs:complexType name="apiCallLogData">
   <xs:sequence />
   <xs:attribute name="uuid" type="xs:string" />
   <xs:attribute name="headers" type="xs:string" />
   <xs:attribute name="body" type="xs:string" />
   <xs:attribute name="responseHeaders" type="xs:string" />
   <xs:attribute name="responseBody" type="xs:string" />
</xs:complexType>
```

ARCOS LLC
8800 Lyra Drive, Suite 200
Columbus, OH 43240
(614) 396 – 5500

Definitions

Term	Definition
API	Application Programming Interface.
HTTP	Hyper Text Transfer Protocol - The protocol of the World Wide Web.
Method	An HTTP request verb used to denote the type of request that is being performed. Typical HTTP Methods include GET, POST, PUT, DELETE, etc.
URL	Uniform Resource Locator - The address of a resource on the World Wide Web. URLs are used to locate a particular resource or group of resources in REST based web services.
HTTP Status	The numeric value for the HTTP Response that indicates whether or not a Request was successful. 200 = OK, 401 = Unauthorized, etc.
REST	Representational State Transfer - A web service approach that relies on standard HTTP methods and URLs to locate and act on resources.
Header	A Key/Value pair in an HTTP Request or Response.
Access Key	A unique key that can be used to obtain an Access Token.
Access Token	A token that must be sent in the Authorization Header of all HTTP Requests when accessing the API.
Attribute	A named property of some resource.
Assignment	A timed grouping between 2 or more entities.
Crew	A Crew in the ARCOS Crew Manager Application. Crews have any number of Attributes, can have Member Assignments, and can have Resource Assignments.
Employee	An employee in the ARCOS system.
Shift	A template that defines a collection of start and end times for a weekly work schedule for an Employee.
Shift Signature	A generated short-hand notation used to represent the working schedule that a Shift covers. For example, if a Shift is defined to include all weekdays from 8:00 am to 5:00 pm, the signature could be MTWRF8-17.
Shift Assignment	The mapping of an Employee to a Shift for a specific week.
Event Type	Used to indicate additional information about the type of Schedule Record that exists for an Employee. Event Types may have any number of Event Traits associated with them.
Event Trait	A property of an Event Type that defines how the Type should be considered for things like auto-extending records, callout, holdover, etc.
Schedule Record	A record indicating the start and end time for an Employee for a specific Event Type.
Scheduled Shift	A Schedule Record that with an Event Type of 1008 - Normal Working Shift. This is the record type typically used to indicate day-to-day scheduled work for an Employee. This type of record is automatically created whenever an Employee is given a Shift Assignment.
Schedule Exception	A Schedule Record that represents some Event Type other than Normal Working Shift. Examples include Planned Overtime, Rest, Sick, Vacation, Callout, etc.
Holiday	A Holiday is a named date that can be used to mask Scheduled Shifts for Employees working in a Location.
XSD	XSD (XML Schema Definition) is a World Wide Web Consortium (W3C) recommendation that specifies how to formally describe the elements in an Extensible Markup Language (XML) document.

| Term | Definition |

xsd:dateString	A string in the format YYYY-DD-MM used to represent dates, where YYYY indicates the year, MM indicates the month (01-12), and DD (01-31) indicates the day.
xsd:dateTime	A string in the format YYYY-MM-DDThh:mm:ss[Z|(+|-)hh:mm] used to represent a date and time [with optional timezone portion]. The date is handled just like xsd:dateString. The time portion includes hh to indicate hours (00-23), mm to indicate minutes (00-59), and ss to indicate seconds (00-59). Specifying Z for the timezone will use the UTC timezone, otherwise the offset in hours (+ or -) should be used.

ARCOS LLC
8800 Lyra Drive, Suite 200
Columbus, OH 43240
(614) 396 – 5500

Document Version History

Version	Date	Description	Author(s)
1.0	08/31/2016	Initial Creation	T. Schneider
1.1 – 1.16	12/16/2016 – 04/24/2019	Updated Loader Data Section Crew Manager Data Crew/Resource Post, Setting Attributes, Setting Wotes Added Schedule Data section Added Employee Data section Added Extracts Data section Added optional parameters for GET schedules/records Add method for returning Crew for a Member Added Basic Authentication Schedule Clarifications Changed attr to required in CM GET Added page parameters Added definitions for xsd:dateString and xsd:dateTime	R. Hutchinson
21.12	03/05/2021	- Version format changed to follow active platform deployment schedule (QA Environment) - Document formatting standard updated - Changed Token Authentication expiration from 8 hours to 4 hours to match current state - Made several grammatical corrections - Updated /application endpoints to match current state - Removed outdated FAQs - Added Callout Endpoint details	K. Rosa
21.14	03/19/2021	- Minor formatting changes - Added Platform API License Section	K. Rosa
21.22	Unpublished	- Corrected grammatical errors. -Updated Request Query Parameters for Find Crews -Listed parameters for v Get Crew - GET /cm/crews/{crewId} - Updated Authorization instructions - Minor correction to Basic API License summary - Added OpStatus Error Example - Added additional details on Token Authentication -Added information to the Supported Operations (Employees) table -Added example of a POST call and their responses creating employees through the API -Added example of a PUT call for updating employee data through the API	P. Rigdon K. Rosa

Version	Date	Description	Author(s)
21.24	06/04/2021	- Added HTTP Functionality section - Added additional Crew Manager functionality information - Added instructions for reviewing Data Loads via the Loader - Added additional detail to callout section. - Added an Appendix containing Flat File specifications for use with the Loader. - Added request/response example for /cm/crews/. - Added request/response examples for POST /cm/batch/resources. - Moved all data structures to new Appendix section and added any missing data structures	P. Rigdon K. Rosa
21.26	6/18/2021	- Added additional detail around shift/schedule functionality - Added the following application method: *getXsd* - Added the following callout methods: *getCalloutTypeLocations, getCalloutStats, updateCalloutStats, updateCallout* - Added the following subscription methods: *cancelSubscription, subscribeForPush* - Added the following employee methods: *getExtendedAttributes, getExtendedAttributeValues, getEmployeeStub, getEmployeeStubs, saveEmployeeSafety, runRosterRules, getEmployeeWorkingStatus, getSafetyNoticeDetails* - Added the following schedule methods: *getScheduleRequests, getScheduleRequest* - Added the following Crew Manager methods: *createTimeEntires, getResourceTypes, getLodgings, getCrewMembers, getResource, getCrewsForRA, getLodging, getWorkIntegration, getRoomTypes, getAvlMemberBadges, getGroups* - Added the following xwalks methods: *savXwalks* - Added additional details across many existing methods. - Added param table for *getExtendedAttributeValues* - Added detail about possible HRI1 205 Record field 7 (HLE-150) for future release. -Updated Loader Data images with images that show do not show customer data. - Added breakdown image for the loader data HRI guide. - Updated step 5 on how to upload a file through the API - Updated getAttributes call to list param table with attributeType	P. Rigdon K. Rosa
21.34	8/19/2021	- Corrected the HRI1 Specification Phone Types Datatype definition	K. Rosa

Version	Date	Description	Author(s)
21.36	8/26/2021	- Corrected the getShifts to show as getShiftsAssignments - Updated response payload for Xwalks method and getRecords parameters - Added Workbench API details and Features methods	P. Rigdon K. Rosa
21.38	9/9/2021	- Added Storm Code and Storm Company fields to the HRI/HRI1 specifications	K. Rosa
21.40	9/20/2021	- Removed Workbench API details and Features methods. This information will be added back at a later date - Added the STARS flat-file specification - Added the SHIPS flat-file specification - Clarified flat-file standard to include additional commentary advising that provided via the Web UI or API uploadFile method are not to contain any fixed-length data values	K. Rosa
21.40.1	9/23/2021	- Revised the SHIPS format 100 record field 4 note	K. Rosa
21.40.2	9/27/2021	- Minor formatting fixes - Added additional details regarding pagination support to the searchEmployees, getDetailCalloutResults, getCrews, getResources, and getLodgings methods - Added clarifying detail to the getExtracts method based on functionality added in 21.40	K. Rosa
21.40.3	9/28/2021	- Updated the deleteResource method with additional details of the Multipart request payload - Added details for the createShift method	K. Rosa
21.44	10/25/2021	- Updated createCallout example to have a subcallout listed. Pg. 97 - Added additional supported Authorization headers - Added sMART Data section and getCurrentAVL method. - Added Crew Manager Sync methods. - Added Schedules Sync method. - Updated POST-applyAction method with initial definition and examples - Updated XML Examples formatting to Arial/Font:10/Line spacing: 0 - Added example for callout/{coMainId} - Updated the parameter tables for all GET cm/ methods - Updated examples for Crew Manager WOTES - Updated POST/cm/resources/{resourceId}/attributes	P. Rigdon K. Rosa

Version	Date	Description	Author(s)
21.52	12/22/2021	- Added additional Crew Manager (cm) methods for lodging: updated createRoomType, added saveRoomTypeAttribute, added getAttributes, added getAttributesByNames, added deleteRoomType, added createLodgings, added deleteLodging.	P. Rigdon
22.02	01/01/2022	- The getXsd method has been renamed getRestConfigXml - Updated savXwalks to getValues displayName	K. Rosa P.Rigdon
22.08	02/24/2022	- Updated getConfiguredClasses to correctly indicate locationId is a required parameter	K. Rosa
22.16	4/1/2022	- Updated Crew Manager section, added overview and formulas	K. Rosa
	10/28/2022	- Updated Employee Extended Attributes endpoint to match WADL configuration	P. Rigdon
23.18	04/24/2023	- Updated members/{memberId}/attributes to include alternate endpoint to to pass webId or vruId for the employee identification	P.Rigdon
23.26	05/29/2023	- Added “roster” parameter to the GET /employees endpoing	P.Rigdon

24.16	03/25/2024	- Updated, Parameter “Page” to “page” in *GET/cm/crews* - Added request payload WorkOrder example in *POST/cm/resources* - Added, “Each id parameter is added separately (e.g. id=10905&id=5292)” to *GET/cm/sync/crews* - Updated “0,1” to “groups, crews, members, resources” in *GET/cm/attributes* request query parameter target - Updated **Optional File Naming Standards** section - Updated, “N(X) = Numeric (Max Length)” to “N(X,X) = Numeric (Max Length, Max Decimal Length, if applicable)” in **Record Format Standards** - Updated “See Note above.” to “See description above.” - Updated, “Company Acronym” to “[Company Code] ARCOS Customer Company Acronym (Company SCHEMA)” in the 100 – Header/Trailer Record table notes section, Company Identifier - Updated, “Export Date and Time - Type N(12)” to “Export Date and Time - Type N(14)” in the 100 – Header/Trailer Record table field description and format section - Updated, “Format MMDDYYYYHHMM Hours (HH) are 24 hour clock. Export date cannot be more than 6.75 days from load date.” to “Format MMDDYYYYHHMMSS, Hours (HH) are 24-hour clock. Export date cannot be more than 6.75 days from load date. Seconds are optional.” in the 100 – Header/Trailer Record table field description and format sections - Updated “Total records not including header records” to “Total records not including 100 header/trailer record” for the Export Record Count int eh 100 – Header/Trailer Record. - Add “In the Notes section, each field is identified as REQUIRED or N/A. N/A fields must be included in the file structure as a blank column. Optional fields are assumed when not indicated as REQUIRED or N/A.” to the Record Descriptions section - Removed “For each field, below, ARCOS LLC will indicate whether the field is REQUIRED, Optional or N/A based on the specific ARCOS configuration.” - Updated “, when Duty Phone functionality is available.” to “When Duty Phone functionality is turned on:” - Updated “e.g. 154.74 hours (No Limit)” to note “Hours can be entered with a whole number (e.g.	G. Brown

		350) or with two decimal places (e.g. 154.74)" - Update, "IF USED – would contain number of personal-day hours to which employee is entitled for the period – for use with Vacation Management Add-On." to "IF USED – Requires the number of vacation hours to which employee is entitled for the period – for use with Vacation Management Add-On.**All Hrs Entitled and Hrs Used Vac Mgt fields must be populated with a numeric value. Zero (0.00) is REQUIRED as a valid numeric value and will overwrite existing entry in the ARCOS Employee Vacation Management Page."** - Deleted, "Any Field not indicated as REQUIRES should be Optional. Fields marked as "N/A" or 'IF USED' are either features currently not used or fields that have no bearing on this project. These fields should be left null in the load file. If data exists in these fields in the load file, it will be ignored during the load process." - Updated, "Field Description and Format" to "Field Description and Type" - Updated, "Notes" to "Format and Notes" - Deleted, "For each field, below, indicated whether the field is REQUIRED, Optional or N/A based on the specific ARCOS configuration." - Updated "X" to "Type X" - Updated "N" to "Type N" - Updated, "Year to Date Hours 9999.99" to "Year to Date Hours" - Updated "Summation of hours by earnings type." to "Summation of hours by earnings type. Format 9999.99" - Updated Field Description and Type "MMDDYYYY" to "– Type N(8)" - Added "Format MMDDYYYY" to Format and Notes section - Updated, "NO Alphabetic characters" to "Non-numeric characters will not validate" - Updated, "This is NOT Call Order Seq." to "This is the device display order in the employee record to (e.g. Phone 1, Phone 3, Phone 4)" - Updated, "(1,2, or3)" to "set by customer to identify order phone numbers are called in a Callout" - Add, "Expected numeric PIN, if required by pager" - Updated "(1K)" to "(1000)" - Updated, "R=Regular; C=Cell Phone; P=Pager;" to "R=Regular Phone, C=Cell Phone, P=Pager, or customer defined in Device Types Admin Page"	

Version	Date	Description	Author(s)
		- Updated, “Column Name” to “Work Order Name – Type X” - Updated “name - Type X” to “name must be entered in all lowercase to identify the Work Order name – **REQUIRED”**	