IRWIN

Integration Specification

Contract API v11

Prepared By: IRWIN Core Team

Last Updated: 6/9/2026

Contents

1 Introduction

Integrated Reporting of Wildfire Information (IRWIN) is a Wildland Fire Information and technology (WFIT) affiliated investment intended to enable an "end-to-end" reporting capability. IRWIN provides data exchange capabilities between existing applications used to manage data related to wildland fire incidents and resources. IRWIN services are focused on the goals of reducing redundant data entry, identifying authoritative data sources, and improving the consistency, accuracy, and availability of operational data. By interconnecting systems, new and updated information is automatically available to the different interagency systems and to a dashboard to provide queries and reports. This capability supports a number of needs and provides benefits throughout the wildland fire community, including:

1. Allow consistent reporting of data
2. Reduce the duplicate entry of data
3. Identify authoritative sources of data
4. Speed access to data located in diverse source systems
5. Increase data accuracy, and
6. Increase the availability of data

To facilitate this, IRWIN provides a common data exchange capability across all participating functional areas for capturing, reporting, and sharing Resource Contract information. It is an objective for IRWIN to facilitate data integration services among systems to support near real-time availability of new and updated information to the relevant interagency systems. This is primarily accomplished by integrating these systems through the IRWIN Application Programming Interface (API): a RESTful web API providing a common method to exchange wildland fire data.

1.1 Purpose and Audience

The Contract Integration Specification introduces and expands upon those topics necessary to begin data exchange through the IRWIN Contract API. A formal discovery process is required to obtain an authentication credential, which allows access to the Contract IRWIN API. This document is not a replacement for that process. In addition, IRWIN provides a separate API for data exchange of Incident, Resource, Frequency, and Learning information.

The IRWIN Community is comprised of the IRWIN Core Team and IRWIN Extended Teams. The IRWIN Core Team is responsible for developing and supporting the technical integration based on requirements provided by the Wildland Fire Community. The IRWIN Core team is comprised of technical developers, data architects, business leads and implementation leads. IRWIN Extended Teams represent the technical and businesspersons who support a system that exchanges data with other systems through the IRWIN Integration Service.

This document is intended for extended teams and particularly their system developers responsible for modifying their application for data exchange within the IRWIN Resource Contract integration services.

1.1.1 Associated Documents

IRWIN Data Mapping Workbook
A workbook containing sheets for the IRWIN data element details and Authoritative Data Source (ADS) matrix. https://www.wildfire.gov/application/irwin-integrated-reporting-wildfire-information

1.2 Communication Network

1.2.1 IRWIN Observer

Observer is a tool for discovering IRWIN incidents and resources and understanding the data exchange transactions that have occurred. Observer is available for all three IRWIN environments and can be used to interact with both **root** and **next** versions of the IRWIN APIs.

- TEST: https://irwint.doi.gov/observer
- TEST/next: https://irwint.doi.gov/observer?v=next
- OAT: https://irwinoat.doi.gov/observer
- OAT/next: https://irwinoat.doi.gov/observer?v=next
- Production: https://irwin.doi.gov/observer

1.2.2 IRWIN Website

Public facing site providing information regarding IRWIN.

https://www.wildfire.gov/application/irwin-integrated-reporting-wildfire-information

1.3 Points of Contact

Kara Stringer – IRWIN Business Lead

kara_stringer@ios.doi.gov

435.400.4301

Brandon Green - IRWIN Project Manager

Brandon_Green@ios.doi.gov

410.303.3307

2 Conceptual Architecture

The IRWIN Contract API (Application Programmer Interface) is designed to broker common wildland fire contracting data across various applications. This **RESTful API** exposes standard **Add, Query, and Update** utility operations, allowing integrated systems to share operational data. Although the API is customized, it follows standard extension guidelines of the underlying ArcGIS Server software. These custom operations:

- Validate data standards
- Enforce updates on an element-by-element basis only by authenticated systems
- Provide operations specific to the business needs of the wildland fire community

The API's role is to provide the ability for many disparate systems to create and edit contract information or retrieve updated contract data on demand. With the understanding that these external systems leverage different core technologies, languages, platforms, are in varying lifecycle stages, or have different business rules, the API provides a common, flexible approach to integration yet enforces NWCG accepted standards and business workflows.

The workflow for contract management can be accessed by clicking here.

This diagram provides a reference for external system business and technical persons for designing and testing the IRWIN integration interface.

3 Environments

IRWIN has three API environments: TEST, Operational Acceptance Testing (OAT), and Production (PROD). During any release, the release package is promoted from TEST to OAT to PROD. Each promotion only occurs after appropriate testing and acceptance. Each Extended System is given credentials to authenticate to each environment.

Within the TEST and OAT environments, there is a **"next" folder**. The software exposed in TEST/next and OAT/next is considered under-development. The software at the root is considered stable and is identical to the API on PROD. The following table describes each environment's intended purpose:

For more information about the release workflow across these environments, please reference the Release Management Plan.
https://www.wildfire.gov/application/irwin-integrated-reporting-wildfire-information.

3.1 Accessing Root v Next APIs

The root and next services are accessed through IRWIN's ArcGIS REST Services Directory, which varies by IRWIN environment:

Environment	ArcGIS REST Services Directory URL
TEST	https://irwint.doi.gov/arcgis/rest/services
OAT	https://irwinoat.doi.gov/arcgis/rest/services
Production	https://irwin.doi.gov/arcgis/rest/services

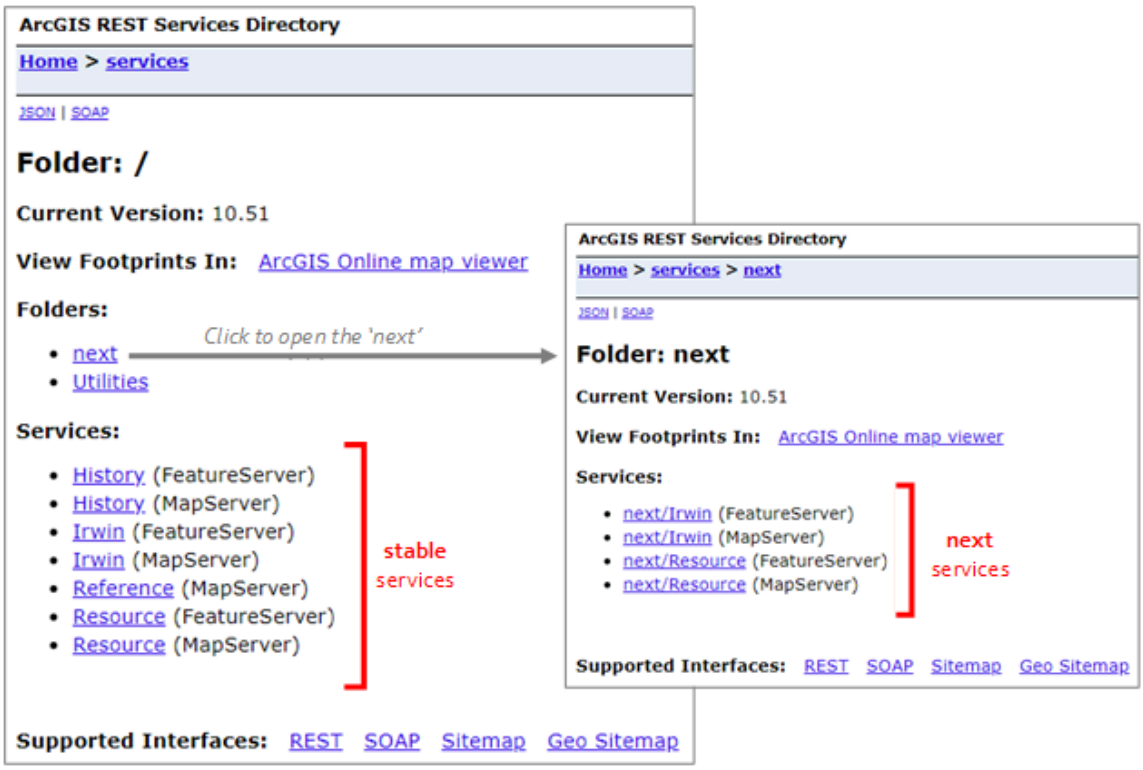
The root services can be accessed through the Services portion of the ArcGIS REST Services Directory. The Services portion of the ArcGIS REST Services Directory is accessible on all IRWIN environments.

The TEST/next and OAT/next services can be accessed by selecting 'next' under the Folders portion of the ArcGIS REST Services Directory. The 'next' folder is only accessible on IRWIN's TEST and OAT environments.

The figure below illustrates how to access both root and next services. These screens are for example purposes only, so the version number and layer/table list may be different when accessing real time.

	TEST	OAT	Production
root	Extended Systems testing against released software. https://irwint.doi.gov/arcgis/rest/services	Extended Systems testing & QA against released software. https://irwinoat.doi.gov/arcgis/rest/services	Extended Systems using IRWIN API as an integratio service. https://irwin.doi.gov/arcgis/ services

next	IRWIN Core Team testing against under-development software. https://irwint.doi.gov/arcgis/rest/services/next	Extended Systems testing against under-development software. https://irwinoat.doi.gov/arcgis/rest/services/next	



ArcGIS API REST Endpoints Example Screen

3.2 Checking the API Version

To validate or check the API version to which you are connecting, use the value "IRWIN_API_Version" which is a property that can be found by reading the JSON response of either the root or any layer of the feature service.

4 Approach to Integration

Integration with IRWIN's Contract API involves analyzing the extended team system's user workflows to understand Contracts and Contract line items (ContractRelationships) using the (ADD), read (QUERY), and edit or invalidate existing Contracts and Contract line items (UPDATE). Each of these actions are "Integration Points" where the partner system may be adjusted to include calls to IRWIN web services.

> NOTE: IRWIN does not push data to connected systems but rather allows connected systems to publish and consume data via services.

This analysis is known as the "Discovery Process". Over the course of this process, the primary focus is to understand alignment with common workflows to discover Integration Points. The workflows are diagrammed and then expanded to analyze how the integration with IRWIN might occur, as well as if additional requirements need to be analyzed with the Core team.

The outcome of the discovery process is a mutual understanding between the IRWIN Core and Extended teams regarding how to integrate.

Following the Discovery Process, the system is provided credentials to TEST and OAT environments to begin their development against the IRWIN API. External systems using the API will require a level of integration testing, facilitated by the IRWIN business leads, before being issued credentials to the Production environment. This integration testing occurs between January and March each year.

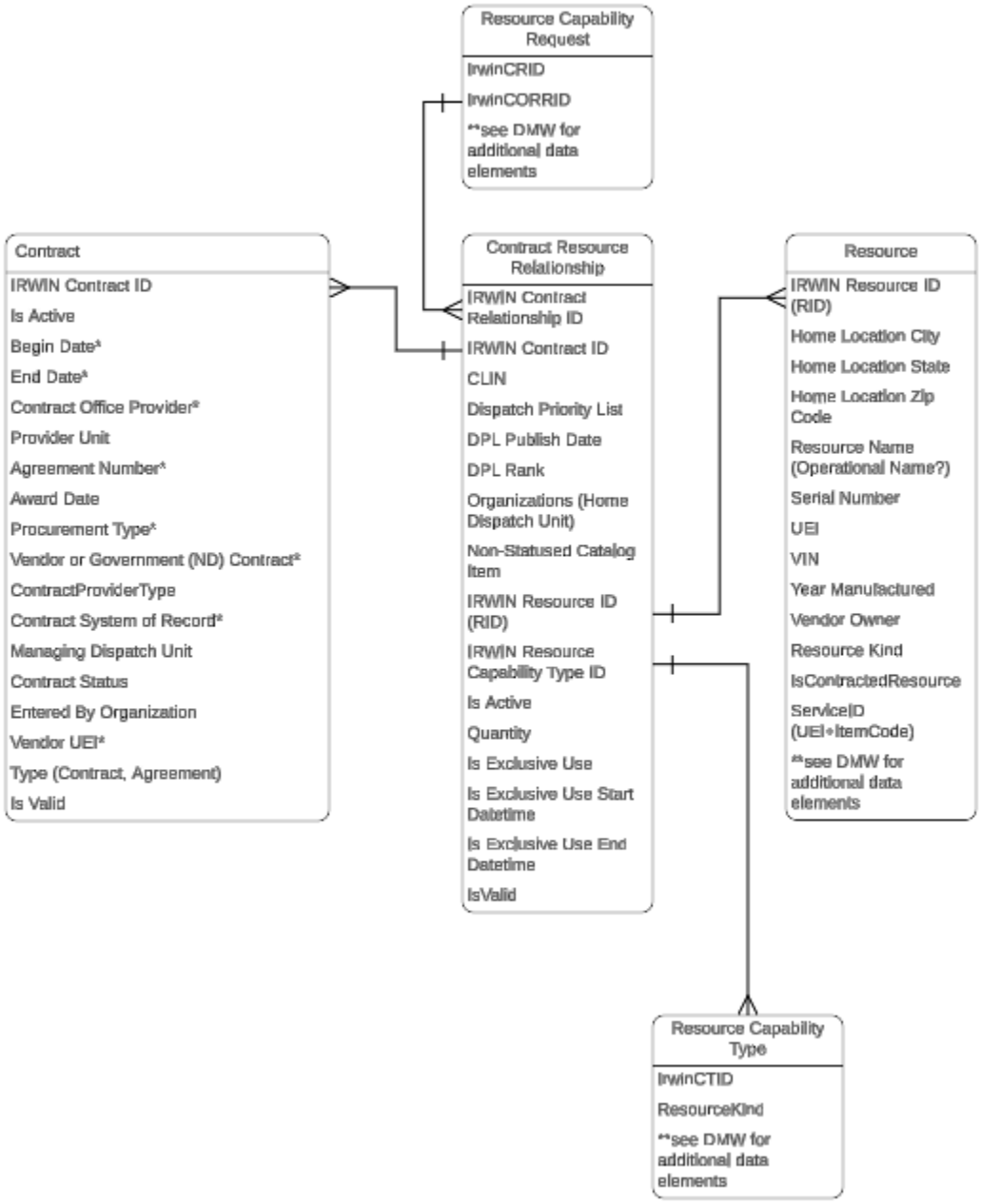
Besides the specific integration points mentioned above, applications will need to maintain synchronization with IRWIN in order to acquire data updates from other participating systems. This may be accomplished by creating a process to continuously poll IRWIN at predefined intervals (seconds or minutes), or as users access the contract data. Synchronization also includes validation on specific "patterns" that may affect the data, and potentially trigger specific actions to occur, such as the invalidating contract line items when the contract is no longer valid. In this manner, all systems should maintain a high degree of data integrity, allowing for more efficient data integration across applications.

5 Development Considerations

This section provides guidance to developers in understanding the main areas of coding for interacting with the Contract API to exchange data. The Contract feature service contains two feature data layers, allowing RESTful interaction via exposed web operations. The figure below depicts the Contract Feature Server layer and related tables.

Contract ER Diagrams and Workflows are documented in the Lucid charts, IRWIN V11: Lucidchart.

Contract API tables +



- *Reference the IRWIN Data Mapping Worksheet, CONTRACT tab for details on every data element.*

- Contract Table
  - Describes the shared data elements of a Contract. Key values are Agreement Number and IRWIN Contract ID.

- Contract Resource Relationship Table
    - Defines relationships between the Contract and the associated Contract Line Items.

- Resource Table

    - Describes Resources available in IRWIN. A contracted resource, as identified in the Contract Resource Relationship table (contract line items) and is linked to the Resources table using the IrwinRID.

- Capability Type

    - Describes the Capability Type of the contracted Resource as identified in the Contract Resource Relationship table (contract line items) and is linked to the Resources table using the IrwinRID. Example: Backhoe (BHOE)

- Resource Capability Request

    - Describes requested Resources for an Incident. A filled Capability Request will contain the Resource ID (IrwinRID) of the dispatched resource. If the Resource is Contracted, the Agreement Number is available within this table.

5.1 Authentication and Authorization

All integrated systems are provided with a system level account to authenticate with the IRWIN Incident API. Systems will authenticate by acquiring a short-lived token string via the GenerateToken operation and adding its value to a token parameter when making any succeeding web calls to IRWIN. As part of the GenerateToken response, the token's expiration time is provided. Once the token's expiration time is met, the integrated system needs to request a new token.

> NOTE: The maximum token lifetime that may be requested in IRWIN is 60 minutes. A token should not be requested more than twice per hour. Best practice is requesting once every hour.

When generating a token, a parameter named 'client' is supplied. There are several options for this parameter as described in the online documentation. It is recommended that systems use the 'referer' client as the supplied parameter. This option avoids issues such as IPs that may change between requests of getting the token and subsequent calls that use the token, and is a more stable option in environments where IPs may not always remain the same. To implement this, when the token is requested, the client value is supplied as 'referer', and a value is supplied for the optional 'referer' parameter. This value for 'referer' can be any value, but will need to be supplied on subsequent calls that use the token and the two values must match.

Sample input:
username='yoursystem',

password='yourpassword',
client='referer',
referer='YOUR REFERER VALUE',
expiration=60,
f=json

On subsequent calls that use this token, the token value must be supplied, and the REFERER header value in any HTTP requests must match the value for 'referer' provided in the token request.

Documentation for the GenerateToken operation can be found at:

Documentation | Esri Developer

Once a credential system connects to IRWIN, access to individual API operations and data elements is based on authorization roles. Each integrated system is placed into a role defined during the discovery process. For example, all read only systems have an "IRWINREAD" role. Integrated systems that will write are described as "CAD" and "non-CAD" and have IRWINCAD and IRWINREADWRITE roles, respectively. For the Contracts API, these roles are "Contracting" and "ContractsReadOnly".

For a full list of available roles relative to contracts, reference the table below.

Contract API Role	Detail
CONTRACTING	• Table permissions ○ CONTRACT – Create, Read, Update ○ ContractResourceRelationship – Create, Read, Update ○ Resources (including CapabilityRequest) - Read
CONTRACTSREADONLY	• Table permissions ○ CONTRACT – Read only ○ ContractResourceRelationship – Read only ○ Resources (including CapabilityRequest) – Read only

5.2 Key Data Concepts

The Contract systems will have the capability to add and update data within two key tables:

1. Contracts: This entity tracks the high-level agreement between an agency and a provider.
2. ContractResourceRelationships: This table is equivalent to contract line items and establishes the relationship between contract agreements and specific resources.

Each contract is assigned a Contract System of Record, which corresponds to the contracting system that originally created the contract. Only the assigned Contract System of Record is authorized to perform subsequent updates to that contract or its associated data.

Understanding the fundamental data elements of the Contracts and ContractResourceRelationships tables is critical for effectively integrating with the IRWIN system. For a detailed reference on these data elements, please consult the IRWIN Data Mapping Workbook.

5.2.1 Workflow & Logic

Based on the provided architectural diagrams, here are the integration specifications for the New Contract Workflow, focusing on the data exchange between the Contracting System, IRWIN, and downstream systems like IROC.

A. Initiation & Modification

Trigger: A new contract is awarded or a "novation modification" occurs in the Contracting System.

Initial Sync: If a DPL (Dispatch Priority List) exists, the system sends contract info to IRWIN.

Query Phase: The Contracting System queries IRWIN for existing resources before creation to prevent duplicates.

B. Resource Creation Logic

Non-Unique Resource Path: If a resource is identified as non-unique (e.g., generic equipment or personnel pools):

Set IsContractedResource = true.

Set NonStandardResource = true.

Uniqueness Constraint: IRWIN enforces uniqueness using a composite key: ServiceID + UEI.

Unique/Standard Path: Follows the standard identity field requirements.

C. Resource-Contract Relationship

If the resource/capability already exists in IRWIN, a relationship is created.

If not, the Contracting System executes an Add Resource and/or Add Capability request.

5.2.2 Data Schema & Mapping

Resource Submittal from Contracting Systems:

Field	Type	Description / Notes	Required?
ResourceID	PK	Unique identifier derived by IRWIN.	n/a
ResourceKind	string	Valid Values are Equipment, Crews, Aircraft, Overhead, Team, Module, Overhead Group, Equipment Group, Aircraft Group. This value cannot be changed on update.	Yes
SerialNumber	int	A set of numbers that is put on equipment by the manufacturer so that each has a different number and can be recognized.	Yes for Equipment
VIN	string	The VIN/PIN/HIN that identifies the eqiupment. VIN: vehicle's identification number (VIN).	Yes for Equipment
YearManufactured	int	The calendar year when the resource was manufactured.	Yes for Equipment

IsContractedResource	int	Default is 1.	Yes, must be '1' for contract system resources
ResourceSOR	string	Currently ‘IIPA' or 'HEMS’.	Yes
GeneralStatus	string	Default is 'Available'.	Yes
HomeDispatchUnit	FK (string)	Must be a valid Unit ID.	Yes
UEI	string	Unique Entity Identifier for the vendor.	Yes
OwnerVendor	string	Name of the vendor tied to the resource.	Yes
HomeCity	string	City of the resource	Yes
HomeState	string	State (must be 2-letter abbreviation) of the resource	Yes
HomeZipCode	string	Zipcode of the resource	Yes

If ‘non-standard’ / 'non-unique resource, the following data elements are required:

Field	Type	Description / Notes
ResourceID	PK	Unique identifier derived by IRWIN.
ResourceKind	string	Valid Values are Equipment, Crews, Aircraft, Overhead, Team, Module, Overhead Group, Equipment Group, Aircraft Group. This value cannot be changed on update.
IsContractedResource	int	Default is 1.
ResourceSOR	string	Currently ‘IIPA' or 'HEMS’.
GeneralStatus	string	Default is 'Available'.
HomeDispatchUnit	FK (string)	Must be a valid Unit ID.
UEI	string	Unique Entity Identifier for the vendor.
OwnerVendor	string	Name of the vendor tied to the resource.
HomeCity	string	City of the resource
HomeState	string	State (must be 2-letter abbreviation) of the resource

HomeZipCode	string	Zipcode of the resource
ServiceID	string	Unique Identifier provided by the contracting system to uniquely identify a resource that does not meet data standards.

5.3 Reading Contracts

A Contract record can be sent to IRWIN once it is awarded or when a "novation modification" occurs. The Contract System of Record (SOR) is assigned by IRWIN based on the system responsible for creating the record. Only the assigned Contract SOR system is authorized to perform subsequent updates to that record.

New Contract records can be created using the ArcGIS REST AddFeatures operation. This operation supports both individual and batch creation of features within the relevant data layer. To ensure successful creation, refer to the IRWIN Data Mapping Workbook for detailed information on the required fields, data types, valid values, and validation rules for Contract and ContractResourceRelationship data elements.

5.3.1 Maintaining Synchronization

Synchronization can be accomplished by periodically "polling" IRWIN using the ArcGIS REST Query operation referencing the Contract feature layer index number. The criteria included in the query's "where clause" can be written based on how the system wants to maintain synchronization.

Continuous Polling

Systems which store and process contract data locally may periodically poll IRWIN to acquire updated contract information. Using the Query API operation, systems will request updates between a start and end date/time on which the contract or contract resource relationship was modified or created – essentially requesting blocks of time. Updates returned are the current state of each contract that has been created or updated since the start date/time. This standard long-polling technique will require the application to maintain a fairly high level of synchronization in order to preserve cross-system integrity and maintain a smaller request payload.

Using Query, there are two common ways to implement this pattern:

1. Allow IRWIN to provide inputs for syncing: The most common way to accomplish sync is to provide an original value for the modifiedOnDateTime parameter to be used for the Query where clause and query for all contract and/or contract resource relationships after this date. This is the best way to keep up to date if the integrated system intends to maintain and process IRWIN updates locally. Note that the modifedOnDateTime is set when the contract is first created or modified, so examining dates modified will also capture newly created contract records.

2. Request distinct blocks of time: Request distinct blocks of time by setting an upper and lower limit for the modifiedOnDateTime parameters in the Query where clause. This allows the client to fully control the block of time, and as long as blocks are contiguous, no updates will be missed.

5.5 Contract API Record Creation

5.5.1 Contract Record

A Contract record can be sent to IRWIN once it is awarded or a "novation modification" occurs. The Contract SOR (System of Record) will be assigned by IRWIN based on the system which created that record. Subsequent updates can only be performed by the Contract SOR system.

New Contract records can be created by using the ArcGIS REST AddFeatures operation. This generic create operation allows for individual or batch features to be created against the underlying data layer. Reference the *IRWIN Data Mapping Workbook* for details regarding required fields, data types, valid values and validation rules for Contract and Contract Resource Relationship data elements.

Documentation for AddFeatures can be found at:
https://developers.arcgis.com/documentation/.

Systems pushing data to IRWIN Contracts must ensure records are unique:

- Required data elements are identified in the Data Mapping Worksheet
- Composite Key: AgreementNumber + ContractProviderUnit + CreatedBySystem.
- Exclusion: Records marked IsValid = False are ignored during the uniqueness check.
- Action: IRWIN will reject any records that violate this uniqueness constraint.
- The following fields must be restricted to these specific domain values
    - ContractStatus: Active, Awarded, Suspended, Terminated, Expired.
    - ContractType: IBPA, EU, OC, CWN, EERA, IDIQ, LUA.
- The Contracting System should query IRWIN for existing resources before creation to prevent duplicates.
- IRWIN will create and assign the IrwinCONID which will be used to access the related Contract Resource Relationship data.

5.5.2 Create Contract Resource Relationship

IRWIN ContractResourceRelationship:

- Required Key Values
    - IrwinCONID – IRWIN Contracts *(must match existing Contract)*
    - IrwinRID – IRWIN Resources (*must match existing IrwinRID in Resources)*
    - IrwinCTID – IRWIN Resource Capability Type *(must match existing IrwinCTID in CapabilityTypes)*
    - Status Dependency:

        - If the parent Contract status is NOT Active, then the child ContractResourceRelationship must have IsActive set to False, regardless of the ContractLineItemStatus.
    - Owner Derivation:
        - If IsContractedResource is True → OwnerOrganization is derived from OwnerVendor.
        - If IsContractedResource is False → OwnerOrganization is derived from ProviderUnit.AgencyName.
        - Note: This field is Read-Only for external systems; IRWIN handles the derivation.
    - Refer to the Data Mapping Worksheet for other required fields

5.5.3 Creating a Resource record as Contracted Resource

Resources should be created following the guidelines outlined in the Irwin Integration Specification Resources API v11 with these additional actions:

- Set is IsContractedResource to True (1)
- UEI of a valid vendor is required when IsContractedResource is True
- isBulkResource and isNonStandard should be reviewed and updated when applicable
- AgreementNumber of the Contract is required
- ResourceSOR = 'NONE'
    - Once the Resource is created, IROC will set the ResourceSOR='IROC' and assume ownership of the Resource.
- IRWIN will generate the IrwinRID

When creating a resource, if it is a non-unique resource then IsContractedResource is true, NonStandardResource is true, and IRWIN will use ServiceID and UEI for uniqueness. If non-unique, the following fields are required:

- VendorOwner
- UEI
- ServiceID
- ResourceKind
- HomeCity
- HomeState
- HomeZipCode
- ProviderUnit
- ResourceSOR
- GeneralStatus

5.6 Contract Modifications and Updates

Contract and ContractResourceRelationships are updated in IRWIN using the ArcGIS REST UpdateFeatures operation. This generic update operation allows for individual or batch features to be updated against the underlying data layer. UpdateFeatures will accept one or more standard feature objects, which express the data elements the user wishes to update.

Updates can only be performed by the recorded Contract System of Record (*ContractSOR data element*)

The JSON response will contain a key of 'success' with a value of 'true' or 'false'. When false, additional details of failure will be included in the response. *(see Section 6: Error Handling)*

Updates must contain the associated IRWIN assigned ID and the data elements to be updated.

- Contracts – IrwinCONID
- ContractResourceRelationship - IrwinCORRID

Documentation for UpdateFeatures can be found at:
https://developers.arcgis.com/documentation/.

5.6.1 Contract Novation

When there is a modification through a contract novation or an update to the contract status, the recorded Contract System of Record (ContractSOR) will perform the following updates to IRWIN:

- Contract
  - Set isActive to False (0)
  - Set ContractStatus to 'Expired'
  - Must include the IrwinCONID in the request
- ContractResourceRelationship
  - Set isActive to False (0)
  - Set ContractLineItemStatus = 'Expired'
  - Must include the IrwinCORRID in the request

5.6.2 Contract Expiration

When a contract Expires or is Cancelled, the recorded Contract System of Record (ContractSOR) will perform the following updates to IRWIN:

- Contract
  - Set isActive to False (0)
  - Set ContractStatus to 'Expired' or 'Suspended'
  - Must include the IrwinCONID in the request

- ContractResourceRelationship
    - Set isActive to False (0)
    - Set ContractLineItemStatus = 'Expired' or Suspended'
    - Must include the IrwinCORRID in the request

5.7 Filling Resource Capability Request with Contracted Resources

On an unfilled request, IROC or HEMS will assign a Contracted Resource.

1. Contracting System will create a Contract Resource Relationship to the existing Resource.
    a. *(See Section 5.5.2 Create Contract Resource Relationship)*
2. Fill Capability Request
    a. Resource ordering system will update CapabilityRequest
    b. Set IRWINCORRID of the corresponding Contract record
    c. Set AgreementNumber to the Agreement# of the corresponding Contract
    d. IROC creates 'Fill with Agreement' request and sets:
        i. AgreementNumber - Agreement # or Contract ID
        ii. FulFillmentStatus to 'Filled'
    e. Additional requirements are documented in the Irwin Integration Specification Resources API v11

5.8 Auto-Generated Values in IRWIN

The IRWIN API has a series of auto-generated data elements, many are accompanied by additional logic. These auto-generated elements are spelled out below:

1. IRWIN automatically sets the following fields when a new record is created:
    a. CreatedOnDateTime = (now in UTC)
    b. CreatedBySystem = userid of system submitting the record
    c. IrwinCONID = GUID randomly generated – Contracts table
    d. IrwinCORRID = GUID randomly generated – ContractResourceRelationship table

2. IRWIN automatically sets the following fields when any data element is updated:
    a. ModifiedOnDateTime = (now in UTC)
    b. ModifiedBySystem = (now in UTC)

- If IsValid is set to 0 (false), IRWIN expires all relationships related to the Contract (i.e. the contracts's IrwinCONID is contained within the IrwinCONID of the ContractResourceRelationship table).

5.9 Record Validation

IRWIN will perform these validations at the time the record is created:

Contracts

- All required data elements are populated in the request
- AgreementNumber need to be unique across Contract records
- BeginDate and EndDate are valid epoch date in milliseconds
- ProcurementType must be one of the following values:
    - IBPA - Incident Blanket Purchase Agreement
    - EU - Exclusive Use
    - OC - On-Call
    - CWN - Call When Needed
    - EERA - Emergency Equipment Rental Agreement
    - IDIQ - Indefinite Delivery Indefinite Quantity
    - LUA - Land Use Agreement
- ContractProviderType must be one of the following values:
    - Vendor
    - Government
    - Cooperator
- ContractSOR must be a valid IRWIN username (ex. iipa)

6 Error Handling

The IRWIN API returns a variety of indicators and status codes detailing the success or failure of actions. Upon addFeatures or updateFeatures, a boolean "success" property is returned, indicating if the action was successful or not. If false, an error property is also returned which lists the error code (indicating the kind of error) and description (providing the actual error messages).

Note: If an element is prevented from being updated, that element(s) will be listed in the skippedElements portion of the response.

Additional Documentation for error responses can be found at:
https://developers.arcgis.com/documentation/.

IRWIN Exception Codes include:

Description	Code
Initialization Exception	8001
Configuration Exception	8002
Unauthorized Exception	8003
Validation Exception	8004
Json Geometry Exception	8005
Operation Failed Exception	8006

6.1 Validation Errors

If the request results in one or more validation errors, the response will include an "error" object with the "code" property specified as 8004. The "description" property of the error object will be an array of validation error objects. Each validation error that is relevant will be included as a separate object with one of the following codes and messages.

JSON Syntax:

```
{
  "objectId":  <objectId>, //int, objectId value of the updated/inserted feature
  "globalId":  <globalId>, //string, string globalId value of updated/inserted feature
  "success":  <true | false>, //boolean, false if edit was not applied
  "error": { //only returned if success is false
    "code":  <code>, //integer, error code
    "description": [ //array of validation error objects
```

```
      {
        "error": {
          "code":  <code>, //integer, error code
          "message":  <message>, //string, validation error message/description
          "conflictObjectId": <conflictObjectId> //integer, only returned if validation error is a "unique" conflict
        }
      }
    ]
  }
}
```

In the following tables, text highlighted in gray represents example values only; the actual text may vary based on the input and/or context.

Code	Example Message	What does it mean?	How to fix it?
101	value (This is wrong!) must be composed of alphanumeric, hyphen, or period characters.	The value may only contain letters, numbers, hyphens (-), and/or periods (.)	Remove any characters from the value that are not letters, numerical digits, hyphens, or periods.
103	value (X) must be an accepted value ([A|B|C]).	The value must be one of a defined list (or "domain").	Ensure the value you are passing matches one of the specified values in the domain values for this field. **Note that case may be important.**
105	Invalid type. Expected number.	The value must be a number; spaces, letters, or other non-numeric characters are not allowed. Periods and hyphens may be allowed if they are	Ensure the value is a number with no spaces.

		contextually relevant, such as for floating point or negative values.	
	value (10) must not exceed 5.	The numerical value must be less than or equal to the stated maximum.	Lower the value to less than or equal to the maximum.
106	value (abcdefg) must not exceed 6 characters.	The value is too long.	Shorten the length of the value.
107	Invalid type. Expected number.	The value must be a number; spaces, letters, or other non-numeric characters are not allowed. Periods and hyphens may be allowed if they are contextually relevant, such as for floating point or negative values.	Ensure the value is a number with no spaces.
	value (1) must exceed 5.	The numerical value must be more than or equal to the stated minimum.	Raise the value to more than or equal to the minimum.
108	value (tooshort) must be at least 15 characters.	The value is too short.	Lengthen the value to be at least the minimum length; spaces are not valid padding. Typically this means left-padding the value with zeroes.
109	value is not nullable.	The value cannot be omitted on addFeatures or nullified on updateFeatures.	For addFeatures requests, you must include a non-null value. For updateFeatures, ensure the value is not being set to "null".
110	value contains disallowed characters.	The value contains characters that are not allowed for this field, such as spaces or special characters.	Check the value to ensure it contains only characters relevant to this field data.

111	a value is required for IrwinCAD systems.	(Systems with IrwinCAD role only) This value is required on addFeatures.	Ensure the value is not missing or null.
112	a value is required.	This value is required.	Ensure the value is not missing or null.
113	value (XXWRNG) must begin with a valid state code.	The first two characters of this string value must be a valid state abbreviation (or, in certain cases, "CA" or "MX").	Ensure the first two characters are a valid NWCG-standard state code (or "CA" or "MX", if the relevant data falls within Canada or Mexico accordingly)
114	value (123) must be type string.	The value must be a string, and cannot be passed as a JSON-specified type of boolean, number, array, or object.	Ensure the value is enclosed by quotes.
	value (3.14) must be type integer.	The value must be a whole number.	Ensure the numerical value is a whole number with no decimal.
	value (wrong) must be type float.	The value must be a number.	Ensure the value is a number.
	value (Sunday, 23 Feb 2019) must be type epoch datetime (long integer).	The value must be a valid datetime value in Unix-time (or "epoch"-time) format.	Ensure that the value is a datetime value, expressed as a whole number of milliseconds after 12:00 am, January 1, 1970. This will be a 13-digit number.
	value ([34.01,-117.34]) must be type geometry.	The value was not recognized as a correctly formatted JSON geometry.	Ensure the value is a correctly formed JSON object, with (at a minimum) a value for "x" and "y".

	value ({"lat":34.01,"lon":-117.34}) must be a correctly formed geometry object	The value was not recognized as a correctly formatted JSON geometry.	Ensure the value is a correctly formed JSON object, with (at a minimum) a value for "x" and "y".
201	Error querying for IrwinID 069BA152-C519-4502-A560-3F72754FB862: Database error	Depending on the error, this may indicate a server failure of some kind.	Check your input data and try again. If this happens repeatedly, please report it to the IRWIN implementation team.
	Error parsing IrwinID	The IRWIN API couldn't parse the specified ID value.	Ensure the IrwinID/IrwinFID is a valid GUID, specified as a string value with no leading/trailing braces.
205	value (069BA152-C519-4502-A560-3F72754FB862) must be an existing IrwinID.	The specified IrwinID was not found in the incident layer.	Check to make sure you are using the correct IrwinID.
	Error querying for IrwinID (069BA152-C519-4502-A560-3F72754FB862): Database error	Depending on the error, this may indicate a server failure of some kind.	Check your input data and try again. If this happens repeatedly, please report it to the IRWIN implementation team.
206	value (069BA152-C519-4502-A560-3F72754FB862) must be an existing IrwinID.	The specified IrwinID was not found in the incident layer.	Check to make sure you are using the correct IrwinID.
	Error querying for IrwinID (069BA152-C519-4502-A560-3F72754FB862): Database error	Depending on the error, this may indicate a server failure of some kind.	Check your input data and try again. If this happens repeatedly, please report it to the IRWIN implementation team.

	value (069BA152-C519-4502-A560-3F72754FB862) is not valid.	The specified IrwinID refers to an incident that contains IsValid = 0 (false).	Check to make sure you are using the correct IrwinID. Otherwise, you may need to update the relevant incident to set IsValid = 1.
210	Cannot determine parent IrwinID.	The IRWIN API couldn't find or parse the specified ParentIrwinID value. That is, a valid ParentIrwinID was not found in the request.	Ensure the ParentIrwinID is a valid GUID, specified as a string value with no leading/trailing braces.
	Cannot determine child IrwinID.	The IRWIN API couldn't find or parse the specified ChildIrwinID value. That is, a valid ChildIrwinID was not found in the request.	Ensure the ChildIrwinID is a valid GUID, specified as a string value with no leading/trailing braces.
	Error querying for IrwinIds (069BA152-C519-4502-A560-3F72754FB862, 069BA152-C519-4502-A560-3F72754FB863): Database error	Depending on the error, this may indicate a server failure of some kind.	Check your input data and try again. If this happens repeatedly, please report it to the IRWIN implementation team.
	Parent incident type category must be 'CX' to create a relationship of type 'Complex'.	It is only possible to create an incident relationship with RelationshipType = 'Complex' if the parent incident has IncidentTypeCategory = 'CX'.	Ensure the ParentIrwinID refers to an incident of type category 'CX'.
	Child incident type category must be 'WF' to create a relationship of type 'Complex'.	It is only possible to create an incident relationship with RelationshipType = 'Complex' if the child incident has IncidentTypeCategory = 'WF'.	Ensure the ChildIrwinID refers to an incident of type category 'WF'.

	Parent incident type category must be 'WF' to create a relationship of type 'Merge'.	It is only possible to create an incident relationship with RelationshipType = ‘Merge’ if the parent incident has IncidentTypeCategory = ‘WF’.	Ensure the ParentIrwinID refers to an incident of type category ‘WF’.
	Child incident type category must be 'WF' to create a relationship of type 'Merge'.	It is only possible to create an incident relationship with RelationshipType = ‘Merge’ if the child incident has IncidentTypeCategory = ‘WF’.	Ensure the ChildIrwinID refers to an incident of type category ‘WF’.
211	value cannot be changed.	Once this value is set, it cannot be changed.	Remove this value from future updateFeature requests for this record.
	value cannot be changed to CX.	The value cannot be changed to the value identified in the error message (in this case, ‘CX’).	Change the value or remove it from the updateFeature request.
212	If value is CX, it cannot be changed.	Once this value is set to the value identified in the error message (in this case, ‘CX’), it cannot be changed.	Remove this value from future updateFeature requests for this record.
213	Unable to validate whether value is required because the value of 'IncidentTypeKind' could not be parsed.	IncidentTypeKind was either not included in the request or could not be parsed correctly.	Ensure there is a valid value for IncidentTypeKind in the request.
	a value is required for incident type kind FI category WF.	If the incident has IncidentTypeKind = ‘FI’ and IncidentTypeCategory = ‘WF’, the specified value is required to be non-null.	Set a non-null value for the specified field.

800	The following IrwinIDs are already assigned to this UniqueFireIdentifier **2019-AZXYZ-LOCALID1**: 069BA152-C519-4502-A560-3F72754FB862	UniqueFireIdentifier consists of *{FireDiscoveryDateTime**Year**}-{POOProtectingUnit}-{LocalIncidentIdentifier}*. If the combination of these three fields is identical to that of another incident, this validation – and the request – will fail.	Ensure the three values are unique in combination. Typically, the LocalIncidentIdentifier may be the easiest to change to ensure a unique combination.
900	IrwinID is invalid.	The IRWIN API couldn't find or parse the specified ID value. That is, the ID was not found in the request.	Ensure the IrwinID/ IrwinRSID is a valid GUID, specified as a string value with no leading/trailing braces.
901 -904	IrwinID (069BA152-C519-4502-A560-3F72754FB862) not found.	The IRWIN API was unable to find a record in the relevant table/layer with the specified ID.	Ensure the IrwinID/ IrwinRSID is a valid GUID and refers to an existing record in the relevant layer.
905	Error querying for IrwinID 069BA152-C519-4502-A560-3F72754FB862: Database error	Depending on the error, this may indicate a server failure of some kind.	Check your input data and try again. If this happens repeatedly, please report it to the IRWIN implementation team.

7 Contingency Plan

Contingency plan documents are stored at https://www.wildfire.gov/application/irwin-integrated-reporting-wildfire-information.

8 Document Versions

Date	Author	Changes
03/10/2026	Stephen Bankston	Initial Document creation
03/13/2026	Stephen Bankston	Updated for V11 changes