ARPACAL **REGIONECALABRIA** **REPUBBLICA ITALIANA**

ALLEGATO 012

SPECIFICHE DEL SISTEMA DI INTERSCAMBIO DATI TRA CER E CEN PREDISPOSTO DA ISPRA

ISPRA (Istituto Superiore per la Protezione e la Ricerca Ambientale) ha predisposto uno specifico sistema informativo nell'ambito del SINANET (Sistema Informativo Nazionale Ambientale) denominato CER**2**CEN ed al contempo ne ha predisposto una "Guida d'uso al sistema di trasferimento dati".

Tra il 2018 e il 2019 è stata effettuata da ISPRA una riprogettazione del sistema di trasferimento dati dai Catasti Regionali (CER) verso il Catasto Nazionale (CEN) delle Sorgenti di Campo Elettromagnetico istituito dal D.M. 13 febbraio 2014.

L'Area Agenti Fisici ed il Servizio DG-SINA dell'ISPRA hanno svolto un lavoro di analisi dell'esistente, di sviluppo di una nuova piattaforma di amministrazione per la gestione puntuale dei dati e di implementazione di una nuova architettura basata su *RESTful* Web *services* (RWS) per consentire la sincronizzazione dei dati aggiornati dei diversi Catasti Regionali con il Catasto Nazionale.

Nella guida che segue viene illustrato il funzionamento di questo sistema di trasferimento che, assieme al CEN e ai CER, costituisce il Sistema Informativo SINANET CER**2**CEN.

Il Sistema Informativo SINANET CER2CEN – Guida d'uso al sistema di trasferimento dati

Tra il 2018 e il 2019 è stata effettuata da ISPRA una riprogettazione del sistema di trasferimento dati dai Catasti Regionali (CER) verso il Catasto Nazionale (CEN) delle Sorgenti di Campo Elettromagnetico istituito dal D.M. 13 febbraio 2014. L'Area Agenti Fisici e il Servizio DG-SINA dell'ISPRA hanno svolto un lavoro di analisi dell'esistente, di sviluppo di una nuova piattaforma di amministrazione per la gestione puntuale dei dati e di implementazione di una nuova architettura basata su *RESTful* Web *services* (RWS) per consentire la sincronizzazione dei dati aggiornati dei diversi Catasti Regionali con il Catasto Nazionale.

In questa guida viene illustrato il funzionamento di questo sistema di trasferimnto che, assieme al CEN e ai CER, costituisce il Sistema Informativo SINANET CER2CEN.

RESTful Web Services

L'indirizzo per il servizio fullrest api è attualmente: https://193.206.192.27/cen_uni/api

La richiesta aspettata è **HTTP GET** per list/view/, per le altre azioni (add, edit, delete e login) la richiesta è **HTTP POST.**

Autenticazione

Tutte le richiesta alle REST API necessitano di essere autenticate attraverso il meccanismo della HTTP Bearer Authentication.

Codici di errore

Di seguito alcuni codici di errore e di status che le REST API possono restituire:

HTTP Status Code:

- 200 - Richiesta completata correttamente
- 204 - Risorsa eliminata
- 400 - Parametro mancante
- 401 - Autenticazione fallita
- 404 - Risorsa non trovata
- 405 - Metodo HTTP non consentito
- 500 - Errore imprevisto del server

Ad ogni chiamata HTTP GET e HTTP POST completata con successo (HTTP status uguale a 200) le REST API restituiscono come contenuto un JSON/XML.

In caso di chiamata HTTP DELETE completata con successo (HTTP status uguale a 200 o 204) il contenuto della risposta sarà il record cancellato.

Se una chiamata dovesse terminare con un errore (HTTP status diverso da 200 o 204), viene restituito un JSON/XML esplicativo dell'errore avvenuto: errorCode - Codice HTTP status errorMsg - Codice di errore esplicativo del problema occorso.
Ad esempio nel caso l'autenticazione fallisca le REST API restituiscono un HTTP Status 401 e come contenuto: { "errorMsg":"BAD_CREDENTIALS","errorCode":401 }

Esempi di risposta

Per dati JSON la richiesta HTTP è:

```
https://193.206.192.27/cen_uni/api
{ "action": "add", "object": "ELF_TBL_SOGGETTI", "ID_SOGGETTO": 1, ... }
Content-Type: application/json
Accept: application/json
```

- HTTP Response (success)

```
200 OK
{
    "success": true,
    "version": "XX.X.X",
    " ELF_TBL_SOGGETTI ": {
        "ID": 1,
        "ID_SOGGETTO": 1,
        ...
    }
}
```

- HTTP Response (failure)

```
200 OK
{
    "success": false, "version": "XX.X.X
    ", "failureMessage": "<failed
    reason>"
}
```

Tabella riassuntiva RESTful Services

Action	Parameters	Request	Response
list (Ottieni lista di record)	**action**=list **object**=<Table> **start**=<StartRecordNumber> (facoltativo, numero di record iniziale, default = 1) **recperpage**=<RecordsPerPage> (Opzionale, record per pagina, impostazione predefinita = [Record per pagina], richiede l'attivazione dell'impostazione [Dimensioni pagina selezionabili]) **order**=<Field> (facoltativo, ordina per il campo specificato, richiede l'attivazione dell'impostazione [Sort] nella pagina dell'elenco) **orderby**=ASC|DESC (Facoltativo, ordina in ordine crescente (ASC) o decrescente (DESC), richiede l'attivazione dell'impostazione [Ordina] nella pagina di elenco) **<Field>**=<FieldValue> (facoltativo, campo di ricerca per valore, richiede l'attivazione dell'impostazione [Ricerca avanzata / estesa] nella pagina di elenco)	**Without URL Rewrite** GET /api/?action=list&object=TBL_ELF_SOGGETTI **With URL Rewrite** GET /api/list/TBL_ELF_SOGGETTI	**Successful response** { "success":true,"version":"15.0.7", "TBL_ELF_SOGGETTI":[{ "ID": 1, "TBL_ELF_SOGGETTI": 1, ... }, { "ID": 2, "TBL_ELF_SOGGETTI": 1, ... }] } **Failed response** { "success": false,"version": "15.0.7", "failureMessage": "<failed reason>" }
view (Ottieni un singolo record per chiave)	**action**=view **object**=<Table> **<KeyField>**=<KeyFieldValue> (valore campo chiave)	**Without URL Rewrite** GET /api/?action=view&object=TBL_ELF_SOGGETTI&ID=1 **With URL Rewrite** GET /api/view/ TBL_ELF_SOGGETTI /1	**Successful response** { "success": true, "version": "15.0.7", " TBL_ELF_SOGGETTI ": { "ID": 1, "TBL_ELF_SOGGETTI": 1, ... } } **Failed response** Vedere la risposta non riuscita per la lista
add (Inserisci un nuovo record)	**action**=add **object**=<Table> **<Field>**=<FieldValue> (campo da inserire) id_utente	**Without URL Rewrite** POST /api/ action=add&object=TBL_ELF_SOGGETTI=1&... **With URL Rewrite** POST /api/add/TBL_ELF_SOGGETTI=1&...	**Successful response** { "success": true, "version": "15.0.7", " TBL_ELF_SOGGETTI ": { "ID": 16, "TBL_ELF_SOGGETTI": 1, ... } } **Failed response** Vedere la risposta non riuscita per la lista

edit (Aggiorna un campo esistente per chiave)	**action**=edit **object**=<Table> **<KeyField>**=<KeyFieldValue> (valore campo chiave) **<Field>**=<FieldValue> (campo da aggiornare)	**Without URL Rewrite** POST /api/ action=edit&object=TBL_ELF_SOGGETTI=2&ID=1... **With URL Rewrite** POST /api/edit/ TBL_ELF_SOGGETTI /2	**Successful response** { "success": true, "version": "15.0.7", " TBL_ELF_SOGGETTI ": { "TBL_ELF_SOGGETTI": 2, ... } } **Failed response** Vedere la risposta non riuscita per la lista
delete (Cancella un record esistente per chiave)	**action**=delete **object**=<Table> **<KeyField>**=<KeyFieldValue> (valore campo chiave)	**Without URL Rewrite** POST /api/action=delete&object= TBL_ELF_SOGGETTI &ID=1 **With URL Rewrite** POST /api/delete/ TBL_ELF_SOGGETTI /1	**Successful response** { "success": true, "version": "15.0.7", "cars": { "ID": 1, "TBL_ELF_SOGGETTI": 1, ... } } **Failed response** Vedere la risposta non riuscita per la lista
login (Autenticare un utente)	action=login **username**=<UserName> (user name value) **password**=<Password> (password value)	**Without URL Rewrite** POST/api/action=login&username=admin&password=master **With URL Rewrite** POST/api/loginusername=admin&password=master	**Successful response** { "JWT": "<JsonWebToken>" } **Failed response** 401 Unauthorized

Esempi Curl: (se viene restituito un errore di certificato inserire -k prima di -X)

login:

curl -X POST "https://193.206.192.27/cen_uni/api/index.php" -d "action=login&username=Sostituire_con_nome_regione&password=a"

Restituisce il token autenticato per una sessione di 20 minuti

---

list:

curl -X GET "https://193.206.192.27/cen_uni/api/?action=list&object=elf_tbl_soggetti" -H "Authorization: Bearer sostituire_me_con_il_token "

Restituisce tutti i propri record

---

view:

curl -X GET "https://193.206.192.27/cen_uni/api/?action=view&object=elf_tbl_soggetti&ID_SOGGETTO=1ISP&id_utente=Y" -H "Authorization: Bearer sostituire_me_con_il_token "

id_utente=Y è fisso, identifica la regione con codice da verificare in https://193.206.192.27/cen_uni/docu.php Gestione Decodifiche comuni Regioni

---

add:

curl -X POST "https://193.206.192.27/cen_uni/api/index.php" -d "action=add&object=elf_tbl_soggetti&ID_SOGGETTO=XXX&id_utente=Y" -H "Authorization: Bearer

sostituire_me_con_il_token "

id_utente=Y è fisso, identifica la regione con codice da verificare in https://193.206.192.27/cen_uni/docu.php Gestione Decodifiche comuni Regioni

---

edit:

curl -X POST "https://193.206.192.27/cen_uni/api/index.php" -d "action=edit&object=elf_tbl_soggetti&ID_SOGGETTO=XXX&id_utente=Y&RAG_SOCIALE=test edit ZZZ" -H

"Authorization: Bearer sostituire_me_con_il_token "

id_utente=Y è fisso, identifica la regione con codice da verificare in https://193.206.192.27/cen_uni/docu.php Gestione Decodifiche comuni Regioni

---

---

delete:

curl -X POST "https://193.206.192.27/cen_uni/api/index.php" -d "action=delete&object=elf_tbl_soggetti&ID_SOGGETTO=XXX&id_utente=Y" -H "Authorization: Bearer sostituire_me_con_il_token "

id_utente=Y è fisso, identifica la regione con codice da verificare in https://193.206.192.27/cen_uni/docu.php Gestione Decodifiche comuni Regioni

---