EASTIN Web services specs

In EASTIN Web services there are two groups of Web methods: the **batch methods** and the **live methods**. **Batch methods** are invoked by automatic processes which run in EASTIN central server and are used to update some almost-static information inside the EASTIN Portal (for example the ISO tree, the keyword lists, etc.). These methods are called with different frequencies (from once a day to once a month), depending on how often the retrieved information content is supposed to change inside each EASTIN partner's local system. For example the method which returns the description of ISO classes, used to update the ISO tree in EASTIN Portal, is called once a month, because the ISO classification is supposed to be almost constant. Instead the method which returns the number of products for a given ISO class is invoked once a day because new products could be often added inside the EASTIN partners' local systems or perhaps their description could have been modified.
On the other side **live methods** are invoked directly by the end users through the EASTIN Portal Web pages and they return the results of searches inside EASTIN partners' databases about products, actors (also called "organisations" in the EASTIN Web pages), and associated information.

In the following description the name of basic data types derives from the SOAP – XML Schema Definition standard (XSD). Each partner must cast these types to the specific types of the language/platform adopted to implement the Web services.

Batch methods

integer **GetIsoClassProductCount**(string isoCode)
Input parameters**:**
- string isoCode: a string representing a single ISO class (for example "12.22").

Returns:
- integer representing the number of products contained in the ISO class passed in the input parameter. Returns zero if no product belongs to the ISO class.

Frequency:
- Once a day; the first call to the Web method is executed at any moment between 04.00 AM (GMT +1:00) and 04.59 AM (GMT +1:00). The time at which the last call is executed is not defined.

This is a batch method which returns the number of products belonging to the ISO class whose ISO code is passed as a string parameter to the method. If no product belonging to the ISO class is found the method returns zero. The method is designed to work in batch mode. Once a day the ISO classification tree which is stored in the EASTIN central repository is visited by the batch process and for each node which is a leaf of the ISO tree the method is called, passing the ISO code of that node as parameter (so the number of calls to the Web method is equal to the number of leaf nodes of the EASTIN ISO classification tree). The method retrieves the number of products belonging to that ISO class and this information is updated in the EASTIN ISO tree.

IsoClassLocalizationDto **GetIsoClassLocalization**(string isoCode)[1]
Input parameters**:**

[1] This method has to be implemented only by a restricted set of authorized partners. For further information please contact the EASTIN portal administrators.

– string isoCode: a string representing a single ISO class (for example “12.22”).

Returns:

– IsoClassLocalizationDto object containing the description of the ISO class passed in the input parameter. If no description is found returns the null object.

Frequency:

– Once a month; the first call to the Web method is executed on the second day of every month at any moment between 03.00 AM (GMT +1:00) and 03.59 AM (GMT +1:00). The time at which the last call is executed is not defined.

This is a batch method which returns a single object belonging to the class IsoClassLocalizationDto, which represents an element of the ISO classification. The method searches into the local database for information about the ISO class whose ISO code is passed as string parameter to the method. For example if the value "12.22" is passed, the method will search for information about 12.22 ISO class. The information items retrieved by the method and stored in the IsoClassLocalizationDto object are:

- the ISO code;
- the title of the ISO class;
- the scope note of the ISO class (if it exists).

If no information for the ISO class is found the method returns the null object. The method is designed to work in batch mode. Once a month the ISO classification tree which is stored in EASTIN central repository is visited by the batch process and for each node, which represents an ISO class, the method is called, passing the ISO code of that node as parameter (so the number of calls to the Web method is equal to the number of nodes of the EASTIN ISO classification tree). The method retrieves the information about that ISO class and this information is updated in the EASTIN Portal ISO tree. For a complete description of the IsoClassLocalizationDto object see below.

KeywordDto[] **GetKeywords**()[2]

Input parameters**:**

– none.

Returns:

– KeywordDto[]: an array of KeywordDto objects containing information about keywords. If no keyword is found returns a not null KeywordDto[] array with zero elements.

Frequency:

– Once a month; the unique call to the Web method is executed on the third day of every month at any moment between 03.00 AM (GMT +1:00) and 03.59 AM (GMT +1:00).

This is a batch method which returns an array of objects belonging to the class KeywordDto. The method searches into the EASTIN partners’ local databases for the dictionary of (keywords –> ISO classes) which will be used in the keyword research of the EASTIN portal. This method requires no parameter. Each KeywordDto object contains the following information:

- the keyword id in the partner’s local database;
- the keyword text;
- an array of ISO codes which are related to this keyword.

[2] This method has to be implemented only by a restricted set of authorized partners. For further information please contact the EASTIN portal administrators.

If no keyword is found the method returns a not null array with zero elements. The method is designed to work in batch mode. Once a month the method is called and the returned information are updated in the EASTIN portal keyword lists. For a complete description of the KeywordDto object see below.

Live search methods

1. Product searches

SmallProductDto[] **FindSmallProducts**(string[] isoCodes, FeatureDto[] features, string commercialName, string manufacturer, dateTime insertDateMin, dateTime insertDateMax)
Input parameters**:**

- string[] **isoCodes**: an array of strings representing ISO classes (for example [“12.22”, “09.03.03”]);
- FeatureDto[] features: an array of FeatureDto objects (for a complete description of the FeatureDto object see “EASTIN custom data type” below);
- string **commercialName**: the whole or a part of the commercial name of the products to be searched;
- string **manufacturer**: the whole or a part of the manufacturer name of the products to be searched;
- dateTime **insertDateMin**: the lower bound for the insert date of the products to be searched;
- dateTime **insertDateMax**: the upper bound for the insert date of the products to be searched.

Returns:

- SmallProductDto[]: array of SmallProductDto objects containing each a light set of information about a product (for a complete description of the SmallProductDto object see below). If no product is found returns a not null SmallProductDto[] array with zero elements.

This method returns an array of objects belonging to the class SmallProductDto. The method implements five different kinds of searches:

1. If the **isoCodes** array is not void the method searches for all products belonging to the ISO classes passed, using an *OR* statement. For example if [“12.22”, “09.03.03”] is the isoCodes array, all products belonging to the 12.22 ISO class *OR* to the 09.03.03 class are returned.
2. If the **features** array is not void the method searches for all products that possess the indicated FeatureDto objects and whose measures for the respective features are compatible with the measure boundaries specified in the FeatureDto objects. For example if features contains the FeatureDtos [{“Width (cm)”, 30, 50}, {“Height (cm)”, 80, 100}] the method will search for all products having some widths in the range [30, 50] *AND* having some heights in the range [80, 100]. Note that if a product declares for example to have a fixable width between 20 and 40 it should be included in the search results since for some of its configurations it satisfies the boundaries. The product is included in the search results only if the compatibility between its measures and the given boundaries present in in the FeatureDto objects are satisfied for all FeatureDto objects. For a complete list of Features see the paragraph “*EASTIN feature vocabulary*” below.
3. If **commercialName** is not void the method searches a matching between the words contained in the commercialName parameter and the respective data in the EASTIN partner’s local database. Since into the commercialName parameter there could be one or more words, the method must split the words and search inside its database for products whose commercial name contains *all* these words (even if present as substrings inside of biggest strings). For example if commercialName = “quickie xenon” the method must search for all products whose commercial name contains both words “quickie” *AND* “xenon”.
4. If **manufacturer** is not void the method executes the search using the same criteria specified in 2 but applied to products’ manufacturer name.

5. If **insertDateMin** and **insertDateMax** are both not null all products whose insert date is included within the interval [insertDateMin, insertDateMax], endpoints included, are returned. These two parameters must be both not null or both null.

If more than one parameter is not void at the same time, the results coming from the matches for each parameter are merged together with an *AND* logic: only results satisfying the conditions specified for each parameter are returned. If no product is found the method returns a not null SmallProductDto[] array with zero elements.

ProductDto **GetProduct**(string productCode)
Input parameters**:**

- string **productCode**: the id of the product in the EASTIN partner's system.

Returns:

- ProductDto: an object containing detailed information about a single product. If no product is found than returns the null object.

This method returns an object belonging to the class ProductDto (for a complete description of the ProductDto object see below). The method searches into EASTIN partner's local databases for the product which has the id matching with the method parameter productCode. If no product is found the method returns the null object.

2. Actor searches

SmallActorDto[] **FindSmallActors**(string actorType, string[] isoCodes, string[] icfCodes, string actorName, dateTime insertDateMin, dateTime insertDateMax)
Input parameters**:**

- string **actorType**: the type of the actor;
- string[] **isoCodes**: an array of strings representing ISO classes (for example ["12.22", "09.03.03"]);
- string[] **icfCodes**: an array of strings representing the EASTIN ICF classes (for example ["b1", "d2"]) which are a subset of the official ICF classification;
- string **actorName**: the whole or a part of the name of the searched actor;
- dateTime **insertDateMin**: the lower bound for the insert date of the actors to be searched;
- dateTime **insertDateMax**: the upper bound for the insert date of the actors to be searched.

Returns:

- SmallActorDto[]: an array of SmallActorDto objects containing each a light set of information about an actor (for a complete description of the SmallActorDto object see below). If no actor is found returns a not null SmallActorDto[] array with zero elements.

This method returns an array of objects belonging to the class SmallActorDto. The method implements five different kinds of searches:

1. If the **type** parameter is not void the method searches for all actors belonging to the specified type; the possible values for type are: "*companies*", "*projects*" and "*serviceproviders*".
2. If the **isoCodes** array is not void the method searches for all actors belonging to the ISO classes passed, using an *OR* statement. For example if ["12.22", "09.03.03"] is the isoCodes array, all actors belonging to the 12.22 ISO class *OR* to the 09.03.03 class are returned.
3. If the **icfCodes** array is not void the method searches for all actors belonging to the ICF classes passed, using an *OR* statement. For example if ["b1", "d2"] is the icfCodes array, all actors belonging to the b1 ICF class *OR* to the d2 class are returned.

4. If **actorName** is not void the method searches a matching between the words contained in the actorName parameter and the respective data in the EASTIN partner's local database. Since into the actorName parameter there could be one or more words, the method must split the words and search inside its database for actors whose name contains *all* these words (even if present as substrings inside of biggest strings). For example if actorName = "metlex ltd" the method must search for all actors whose name contains both words "metlex" *AND* "ltd".
5. If **insertDateMin** and **insertDateMax** are both not null all actors whose insert date is included within the interval [insertDateMin, insertDateMax], endpoints included, are returned. These two parameters must be both not null or both null.

If more than one parameter is not void at the same time, the results coming from the matches for each parameter are merged together with an *AND* logic: only results satisfying the conditions specified for each parameter are returned. If no actor is found returns a not null SmallActorDto[] array with zero elements.

ActorDto **GetActor**(string actorType, string actorCode)
Input parameters**:**

- string **actorType**: the type of the actor;
- string **actorCode**: the id identifying a single actor inside the EASTIN partner's local system.

Returns:

- ActorDto: an object containing detailed information about a single actor (for a complete description of the ActorDto object see below). If no actor is found than returns the null object.

This method returns an object belonging to the class ActorDto. The method searches into EASTIN partner's local database for the actor of the type specified in the actorType parameter which has the id matching with the method parameter actorCode. If no actor is found the method returns the null object.

3. Associated information searches

SmallAssociatedInfoDto[] **FindSmallAssociatedInfos**(string infoType, string[] isoCodes, string[] icfCodes, string title, string author, dateTime insertDateMin, dateTime insertDateMax)
Input parameters**:**

- string **infoType**: the type of the associated information document;
- string[]**isoCodes**: an array of strings representing ISO classes (for example ["12.22", "09.03.03"]);
- string[]**icfCodes**: an array of strings representing EASTIN ICF classes (for example ["b1", "d2"]);
- string **title**: the whole or a part of the title (in the original language or in English) of the searched associated information document;
- string **author**: the whole or a part of the author names of the searched associated information document;
- dateTime **insertDateMin**: the lower bound for the insert date of the associated information documents to be searched;
- dateTime **insertDateMax**: the upper bound for the insert date of the associated information documents to be searched.

Returns:

- SmallAssociatedInfoDto[]: an array of SmallAssociatedInfoDto objects containing each a light set of information about an associated information document (for a complete description of the SmallAssociatedInfoDto object see below). If no associated information document is found returns a not null SmallAssociatedInfoDto[] array with zero elements.

This method returns an array of objects belonging to the class SmallAssociatedInfoDto. The method implements six different kinds of searches:

1. If the **type** parameter is not void the method searches for all associated information documents belonging to the specified type; the possible values are: "*articles*", "*casedescriptions*", "*ideas*", "*faqs*", "*forums*", "*news*" and "*regulations*".
2. If the **isoCodes** array is not void the method searches for all associated information documents belonging to the ISO classes passed, using an *OR* statement. For example if ["12.22", "09.03.03"] is the isoCodes array, all associated information documents belonging to the 12.22 ISO class *OR* to the 09.03.03 class are returned.
3. If the **icfCodes** array is not void the method searches for all associated information documents belonging to the ICF classes passed, using an *OR* statement. For example if ["b1", "d2"] is the icfCodes array, all associated information documents belonging to the b1 ICF class *OR* to the d2 class are returned.
4. If **title** is not void the method searches a matching between the words contained in the title parameter and the respective data in the EASTIN partner's local database. Since into the title parameter there could be one or more words, the method must split the words and search inside its database for associated information documents whose title (in original language *OR* in English if present) contains *all* these words (even if present as substrings inside of biggest strings). For example if title = "a guide to wheeled walking frames" the method must search for all associated information documents whose original title or whose English title contain all words "a", "guide", "to", "wheeled", "walking" and "frames".
5. If the **author** parameter is not void the method executes the search using the same criteria specified in 4 but applied to the name of the authors of the associated information document (in this case no distinction is needed between original language and English).
6. If **insertDateMin** and **insertDateMax** are both not null all associated information documents whose insert date is included within the interval [insertDateMin, insertDateMax], endpoints included, are returned. These two parameters must be both not null or both null.

If more than one parameter is not void at the same time, the results coming from the matches for each parameter are merged together with an *AND* logic: only results satisfying the conditions specified for each parameter are returned. If no associated information document is found the method returns a not null SmallAssociatedInfoDoc[] array with zero elements.

AssociatedInfoDto **GetAssociatedInfo**(string infoType, string associatedInfoCode)

Input parameters**:**

- string **infoType**: the type of the associated information document;
- string **associatedInfoCode**: the id identifying a single associated information document inside the EASTIN partner's local systems.

Returns:

- AssociatedInfoDto: an object containing detailed information about a single associated information document (for a complete description of the AssociatedInfoDto object see below). If no associated information document is found than returns the null object.

The method searches into the EASTIN partner's local database for the associated information document of the type specified in the infoType parameter which has the id matching with the method parameter associatedInfoCode. If no associated information document is found the method returns the null object.

EASTIN custom data types

As we have seen EASTIN Web services return basic SOAP types, such as String, Int and DateTime, but also custom defined types. A complete description of EASTIN custom defined types follows below. All mandatory fields are marked with a "*" (all the other fields can be considered as nullable). For the array fields in case they are empty do not assign a null value to them but a not null array of zero elements.

IsoClassLocalizationDto

- string IsoCode*: the code of the ISO class;
- string Title*: the name of the ISO class ;
- string ScopeNote: the ISO class description.

KeywordDto

- string KeywordId*: the id of the keyword in the partner's local database;
- string Text*: the keyword text;
- string[] IsoCodes*: the array of all ISO classification codes related to the keyword (for example ["12.22", "09.03.03"]).

FeatureDto

- integer FeatureId*: the ID of the EASTIN feature. For the complete list of EASTIN features and corresponding IDs see the paragraph "*EASTIN feature vocabulary*" below
- decimal ValueMin: the lower bound value of the measure specified for this feature;
- decimal ValueMax: the upper bound value of the measure specified for this feature.

SmallProductDto

- string ProductCode*: the id of the product in the partner's local database;
- string IsoCodePrimary*: the primary ISO Code of the product (for example "09.03.03");
- string[] IsoCodesOptional: the array of all secondary ISO classification codes of the product (for example ["12.22", "09.03.03"]);
- string CommercialName*: the commercial name of the product;
- string ManufacturerCode*: the id of the product's manufacturer in the partner's local database;
- string ManufacturerOriginalFullName*: the full name in the original language of the product's manufacturer;
- dateTime InsertDate*: the insert date of the product;
- dateTime LastUpdateDate*: the last update date of the product;
- string ThumbnailImageUrl: the URL of the small format picture of the product (used when displaying list of products in EASTIN Portal). The URL must be accessible on the Web by the end user's browser. Picture dimensions should be: width 90 px, height 90 px.

ProductDto

- string ProductCode*: the id of the product in the partner's local database;
- string IsoCodePrimary*: the primary ISO Code of the product (for example "09.03.03");
- string[] IsoCodesOptional: the array of all secondary ISO classification codes of the product (for example ["12.22", "09.03.03"]);
- string CommercialName*: the commercial name of the product;
- string ManufacturerCode*: the id of the product's manufacturer in the partner's local database;
- string ManufacturerOriginalFullName*: the full name in the original language of the product's manufacturer;
- dateTime InsertDate*: the insert date of the product;

- dateTime LastUpdateDate*: the last update date of the product;
- string ThumbnailImageUrl: the URL of the small format image of the product (used when displaying list of products in the EASTIN portal). The URL must be accessible on the Web by the end user's browser. Picture dimensions should be: width 90 px, height 90 px.
- bool IsReviewAllowed*: if true the end user is authorized to review this product;
- string ManufacturerAddress: the address of the product's manufacturer;
- string ManufacturerPostalCode: the postal code of the product's manufacturer;
- string ManufacturerTown: the town of the product's manufacturer;
- string ManufacturerCountry*: the country code of the product's manufacturer in ISO 3166-1-alpha-2 code (for example "IT", "US", etc.);
- string ManufacturerPhone: the phone of the product's manufacturer;
- string ManufacturerFax: the fax of the product's manufacturer;
- string ManufacturerEmail: the email of the product's manufacturer;
- string ManufacturerSkype: the Skype account name of the product's manufacturer;
- string ManufacturerWebSiteUrl: the Web site URL of the product's manufacturer;
- string[] ManufacturerSocialNetworkUrls: an array of URLs linking to the product's manufacturer page inside the main social networks (for example Facebook, Twitter, LinkedIn, etc.);
- string ImageUrl: the URL of the big format image of the product (used when displaying the detail view of the product in the EASTIN portal). The URL must be accessible on the Web by the end user's browser. Picture dimensions should be: width 450 px, height 450 px.
- string OriginalDescription: the description of the product in the original language;
- string EnglishDescription: the description of the product in English;
- string OriginalUrl: the URL of the Web page in the original language on the original EASTIN partner's Web site in which the product is presented. The URL must be accessible on the Web by the end user's browser;
- string EnglishUrl: the URL of the Web page in English on the original EASTIN partner's Web site in which the product is presented. The URL must be accessible on the Web by the end user's browser;
- string OriginalDownloadUrl: the URL of the download Web page in the original language on the original EASTIN partner's Web site in which the product is presented. The URL must be accessible on the Web by the end user's browser;
- string EnglishDownloadUrl: the URL of the download Web page in English on the original EASTIN partner's Web site in which the product is presented. The URL must be accessible on the Web by the end user's browser;
- string[] UserManualUrls: an array containing the URLs of product's user manuals;
- string[] VideoUrls: an array containing the URLs of product's demo videos;
- string[] BrochureUrls: an array containing the URLs of product's brochures;
- string[] FurtherInfoUrls: an array containing the URLs of other information available on the Web related to the product;
- FeatureDto[] Features: an array of FeatureDto objects containing all the EASTIN Taxonomy features (with measure values if needed) for this product.

SmallActorDto

- string ActorCode*: the id of the actor in the EASTIN partner's local database;
- string OriginalFullName*: the full name of the actor in the original language;
- string Country*: the country code of the actor in ISO 3166-1-alpha-2 code (for example "IT", "US", etc.);
- dateTime InsertDate*: the insert date of the actor in the EASTIN partner's local database;
- dateTime LastUpdateDate*: the insert date of the actor in the EASTIN partner's local database.

ActorDto

- string ActorCode*: the id of the actor in the EASTIN partner's local database;
- string OriginalFullName*: the full name of the actor in the original language;
- string Country*: the country code of the actor in ISO 3166-1-alpha-2 code (for example "IT", "US", etc.);
- dateTime InsertDate*: the insert date of the actor in the EASTIN partner's local database;
- dateTime LastUpdateDate*: the insert date of the actor in the EASTIN partner's local database;
- string ShortName*: the short name of the actor;
- string EnglishFullName*: the full name of the actor in English;
- string OriginalDescription: the description of the Actor in the original language;
- string EnglishDescription: the description of the Actor in English;
- dateTime StartDate*: the start date of the actor
- dateTime EndDate: the end date of the actor
- string ContactBody: the reference organization of the actor;
- string Address: the address of the actor;
- string PostalCode: the postal code of the actor;
- string Town: the town of the actor;
- string Phone: the phone of the actor;
- string Fax: the fax of the actor;
- string Email: the email of the actor;
- string Skype: the Skype account name of the actor;
- string WebSiteUrl: the Web site URL of the actor. The URL should be accessible on the Web by the end user's browser;
- string ContactPersonFullName: the complete name of the contact person for the actor;
- string OriginalUrl: the URL of the Web page in the original language on the original EASTIN partner's Web site in which the actor is presented. The URL must be accessible on the Web by the end user's browser;
- string EnglishUrl: the URL of the Web page in English on the original EASTIN partner's Web site in which the actor is presented. The URL must be accessible on the Web by the end user's browser
- string[] SocialNetworkUrls: an array of URLs linking to the actor page inside the main social networks (for example Facebook, Twitter, LinkedIn, etc.);
- string[] IcfCodes*: the array of all EASTIN ICF classification codes of the actor (for example ["b1", "d2"]);
- string[] IsoCodes*: the array of all ISO classification codes of the actor (for example ["12.22", "09.03.03"]);

SmallAssociatedInfoDto

- string AssociatedInfoCode*: the ID of the associated information document in the EASTIN partner's local database;
- string Authors*: a string containing the names (or the initials) of the authors of the associated information document (this is not an array but a single string);
- string OriginalTitle*: the original title in the native language of the associated information document
- string EnglishTitle*: the English translation of the original title of the associated information document
- string OriginalLanguage*: the ISO 639-1 code of the native language of the associated information document (for example: "en", "it", "de");
- dateTime InsertDate*: the insert date of the associated information document in the EASTIN partner's local database;
- dateTime LastUpdateDate*: the last update date of the associated information document in EASTIN partner's local database.

AssociatedInfoDto

- string AssociatedInfoCode*: the ID of the associated information document in the EASTIN partner's local database;
- string Authors*: a string containing the names (or the initials) of the authors of the associated information document (this is not an array but a single string);
- string OriginalTitle*: the original title in the native language of the associated information document
- string EnglishTitle*: the English translation of the original title of the associated information document
- string OriginalLanguage*: the ISO 639-1 code of the native language of the associated information document (for example: "en", "it", "de");
- dateTime InsertDate*: the insert date of the associated information document in the EASTIN partner's local database;
- dateTime LastUpdateDate*: the last update date of the associated information document in EASTIN partner's local database
- integer PublicationYear*: the publication year of the associated information document;
- string PublishingDetails: the publishing details (for example the publishing house) of the associated information document;
- string OriginalAbstract: the abstract of the associated information document in the original language;
- string EnglishAbstract: the abstract of the associated information document in the original language;
- string OriginalUrl: the URL of the Web page in the original language on the original EASTIN partner's Web site in which the associated information document is presented. The URL must be accessible on the Web by the end user's browser;
- string EnglishUrl: the URL of the Web page in English on the original EASTIN partner's web site in which the associated information document is presented. The URL must be accessible on the Web by the end user's browser;
- string OriginalDownloadUrl: the URL for the download of the associated information document in the original language;
- string EnglishDownloadUrl: the URL for the download of the associated information document in English;
- string ImageUrl: the URL of the picture related to the associated information document (used when displaying the detail view of the associated information document in EASTIN Portal). The URL must be accessible on the Web by the end user's browser;
- string[] FurtherInfoUrls: an array containing the URLs of other information present on the Web related to the associated information document;
- string[] IcfCodes*: the array of all EASTIN ICF classification codes of the associated information document (for example ["b1", "d2"]);
- string[] IsoCodes*: the array of all ISO classification codes of the associated information document (for example ["12.22", "09.03.03"]);

ANNEX 1 - EASTIN feature vocabulary

A *vocabulary* of features has been introduced in EASTIN to standardize the description of products' technical details. The *Vocabulary* is based on a two level hierarchy made up of *Clusters* and *Features*. Homogeneous Features are grouped together in the same Cluster. For example the Features "Windows", "Mac OS", "Linux", "Chrome OS" , etc... are all grouped in the Cluster "Operating System", while "Printer", "Visual display", "Tactile display", etc... are grouped in the Cluster "Output devices". Features can be of two types: *Measures*, that can have a numeric value or an interval specified (e.g. weight, length, ...), and *Attributes*, that do not have a specified value (i.e. are Boolean features). Overall 18 Clusters and 237 Features have been identified so far. The table below lists all the features (and their ID) identified.

ID	Name	Type	Description
1	**Overall dimensions**	cluster	
2	Width (cm)	measure	
3	Length (cm)	measure	
4	Height (cm)	measure	
5	Weight (kg)	measure	
6	**Capacity/Range**	cluster	
7	Magnification (x)	measure	
8	Number of keys	measure	
9	Number of input channels/devices/messages	measure	
10	Number of output channels/devices/messages	measure	
11	Signal range (m)	measure	
325	Max sound/speech volume (dB)	measure	
326	Max ringer/alarm volume (dB)	measure	
327	M rating (Hearing Aid Compatibility)	measure	*M rating corresponds to interference of Mobile phones with hearing aids set to "microphone mode". The higher the number following the 'M' the clearer the sound should be.*
328	T rating (Hearing Aid Compatibility)	measure	*T rating corresponds to interference of mobile phones with hearing aids set in "t-coil mode". The higher the number following the 'T' the clearer the sound should be.*
12	**Power sources**	cluster	
13	Battery - disposable	attribute	
14	Battery - rechargeable	attribute	
15	Mains electric	attribute	
16	Power via USB	attribute	
17	**Activation methods**	cluster	*How the device (or software) is activated*
18	Electro Myo Graphic Signal (EMG)	attribute	
19	Eye blink	attribute	
20	Acoustic	attribute	
21	Eye gaze	attribute	
22	Speech Recognition	attribute	
23	Mechanical (push, pull, grasp,…)	attribute	
24	Sip/Puff	attribute	
25	Tilt	attribute	
57	**Browsers**	cluster	*Type of browser supported by the device or software*
58	Chrome	attribute	
59	Firefox	attribute	
60	Internet Explorer	attribute	
61	Safari	attribute	
62	Opera	attribute	
63	**Languages**	cluster	
64	Danish	attribute	
65	Dutch	attribute	
66	English	attribute	

67	French	attribute	
68	German	attribute	
294	Greek	attribute	
295	Italian	attribute	
296	Portuguese	attribute	
297	Spanish	attribute	
298	Bulgarian	attribute	
299	Czech	attribute	
300	Estonian	attribute	
301	Finnish	attribute	
302	Hungarian	attribute	
303	Latvian	attribute	
304	Lithuanian	attribute	
305	Maltese	attribute	
306	Other European Languages	attribute	
307	Polish	attribute	
308	Romanian	attribute	
309	Slovak	attribute	
310	Slovenian	attribute	
311	Swedish	attribute	
330	Irish	attribute	
332	Non European Languages	attribute	
69	**Display characteristics**	cluster	
70	Black/white display	attribute	
71	Colour display	attribute	
72	3D	attribute	
73	**Linguistic representations**	cluster	
74	Sign language	attribute	
75	Braille	attribute	
76	Alphabetic	attribute	
77	Symbolic	attribute	
78	Simplified	attribute	
79	**Output devices**	cluster	*Output devices (or software components) the product includes or is designed to be used with*
80	Speakers/headphones	attribute	
81	Printer	attribute	
82	Visual screen/display	attribute	
83	Tactile display	attribute	
84	Vibrator	attribute	
85	Voice synthesis	attribute	
86	Recorded sound	attribute	
87	Environmental control devices	attribute	
89	**Functionalities**	cluster	
91	Word prediction/completion	attribute	
92	Spell correction	attribute	
93	Abbreviation expansion	attribute	
94	Highlights each word/sentence as it is read aloud	attribute	
95	Allows creation of macro function	attribute	
96	Programmable/configurable	attribute	
97	Calendar function	attribute	
98	Reminder	attribute	
100	Portable	attribute	
101	Built-in microphone	attribute	
102	Speech or acoustic signals on menus	attribute	
103	Switch controlled scanning	attribute	
104	**Input devices**	cluster	*input devices (or software components) the product includes*

			or is designed to be used with
105	Joystick	attribute	
107	Keys/keyboard	attribute	
108	Chording keyboard (e.g. Braille keyboard)	attribute	
109	Mouse	attribute	
110	Speech recognition	attribute	
111	Switch	attribute	
112	Touch screen	attribute	
113	Track pad (touch pad)	attribute	
114	Trackball	attribute	
115	Movement tracking system	attribute	
116	Eyegaze control system	attribute	
117	Video camera/webcam	attribute	
118	Microphone	attribute	
119	Accelerometer	attribute	
120	Biosignals sensor (EMG, EOG, EEG)	attribute	
121	**Input adjustments**	cluster	*Available adjustments or filtering options for the input devices (or software components)*
122	Speed	attribute	
123	Controls/keys activation delay	attribute	
124	Type of scanning	attribute	
125	Sensitivity	attribute	
126	Scanning speed	attribute	
127	Size of controls/keys	attribute	
128	Number of controls/keys configuration	attribute	
129	Font size on controls/keys	attribute	
130	Colour of controls/keys	attribute	
131	Filter on repeated activations	attribute	
132	Key repeat rate	attribute	
133	Microphone sensitivity	attribute	
134	**Output adjustments**	cluster	*Available adjustments or options for the output devices (or software components)*
135	Enlargement/zoom	attribute	
136	Font size	attribute	
137	Contrast	attribute	
138	Colours	attribute	
139	Image reversal	attribute	
140	Equalization control	attribute	
141	Volume	attribute	
142	Sound feedback	attribute	
329	Image freeze	attribute	
143	**Connectivity**	cluster	*How the device (or software component) connects to other devices or services*
144	PS2	attribute	
145	Serial	attribute	
146	USB	attribute	
147	Bluetooth	attribute	
148	Infrared	attribute	
149	Jack	attribute	
150	Other wireless	attribute	
151	WiFi	attribute	
152	Cloud or internet based application	attribute	
153	Induction loop	attribute	
154	Inductive coupling	attribute	
157	**Software license policies**	cluster	
158	Free and open source software	attribute	
159	Proprietary	attribute	

160	**Software price policies**	cluster	
161	Free of charge	attribute	
162	Bundled with operating system	attribute	
331	Priced	attribute	
163	**Subdivisions**	cluster	*EASTIN Subdivisions of the ISO 9999 classification*
164	Stationary image-enlarging reading apparatus	attribute	*Stationary system that displays an enlarged image of the subject that has been captured by a video camera*
165	Stationary image-enlarging reading apparatus with connection units for computers	attribute	*Stationary system, with connection unit for computer, that displays an enlarged image of the subject that has been captured by a video camera*
166	Portable image-enlarging reading apparatus	attribute	*portable system that displays an enlarged image of the subject that has been captured by a video camera*
167	Accessories for image-enlarging reading apparatus	attribute	*Accessories for image-enlarging reading apparatus, e.g. XY-tables.*
169	Concha/in-the-ear hearing aids	attribute	*Hearing aids placed in the outer part of the ear canal (concha). Allows room for controls in the hearing aid.*
170	Completely in-the-canal hearing aids	attribute	*Hearing aids placed in the ear canal. No controls on the apparatus.*
172	Behind-the-ear hearing-aids	attribute	*Behind-the-ear hearing-aids with an output <= 132 dB SPL*
173	Power behind-the-ear hearing-aids	attribute	*Behind-the-ear hearing-aids with an output > 132 dB SPL*
175	Voice amplifiers for personal use	attribute	*Devices for increasing the volume of a person's voice.*
177	Electric typewriters without memory	attribute	*Electric typewriters without memory*
178	Electric typewriters with memory	attribute	*Electric typewriters with memory*
179	Braille typewriters	attribute	*Manual Braille typewriters.*
180	Stenotype machines	attribute	*Manual stenotype machines punching Braille on a paper strip.*
181	Electric Braille typewriters	attribute	*Stationary electric Braille typewriters.*
183	Word-processing software	attribute	*Standard and specially designed word processing software and accessories for word processing software. Included is also integrated software with word processing.*
184	Desktop publishing software	attribute	*Software for layout and desktop publishing.*
186	Equipment for recording and/or replaying digital books	attribute	*Hardware devices for recording and/or replaying digital books, e.g. in DAISY format.*
187	Digital note recorders	attribute	*Note recorders and dictaphones, recording in digital format. With internal and/or external memory.*
188	Cassette recorders	attribute	*Tape recorders for recording and/or replaying cassette tapes. Included are note recorders and dictaphones with mini cassettes.*
189	Accessories for recording and/or replaying sound	attribute	
191	Real time captioning systems	attribute	*Hardware or software systems for decoding spoken output to provide video captions*
192	Delayed captioning systems	attribute	*Hardware or software systems that allow to prepare captioning in advance (not in real time)*
193	Captioning services	attribute	
195	Infrared (IR) systems for audio information	attribute	*Devices for receiving or transmitting audio information using infrared light; Included are, e.g., systems, transmitters and receivers for local one-way communication, e.g. personal remote voice transmission and voice transmission systems for auditorium.*
197	Induction-loop amplifiers	attribute	*Loop amplifiers using electromagnetic waves to transmit audio information to hearing aids. Designed for use in one or more rooms.*
198	Small induction-loop amplifiers	attribute	*Amplifiers for small loops, designed for one person. Included are, e.g., pillow loops, neck loops and clip-on equipment transmitting audio magnetically by using the pick-up loop inside the users hearing aids.*
199	Induction-loops	attribute	*Passive induction-loops without built-in amplifiers.*
201	Symbolic voice output communication	attribute	*Communication devices consisting of a touch-sensitive screen*

	devices		*divided into a given number of fields. When activating a field an auditive output with digital or synthetic speech is produced.*
202	Alphabetic communication devices	attribute	*Writing based communication devices with a standard keyboard. Features screen output, synthetic speech output or printed output.*
204	Face-to-face communication software	attribute	*Software that allow a computer or a mobile device (smart phone, PDA, tablet, and other) to work as a communicator.*
205	Tools for developing grids for communication software	attribute	
207	Mobile telephones	attribute	*Mobile phones used for wireless calls on the public mobile network.*
209	Telecommunication and telematics software	attribute	*Software, specifically designed for person with motor, sensory or cognitive disability, for verbal and visual communication between computers via the computer network*
210	Voice over IP Services	attribute	
212	Indicators with visual signal	attribute	*Devices that indicate with light or other visual signal that something is happening in the place where the transmitter is; they can transform, e.g. audible signal to visual signal; Included are, e.g., electronic babysitters, door signals, door signal indicators and door warners.*
214	Indicators with acoustic signals	attribute	*Devices that indicate with sound that something is happening in the place where the transmitter is; they can transform, e.g., visual signal to audible signal or they can increase the volume of a normal device; Included are, e.g., rain indicators and computer-signal indicators.*
216	Indicators with mechanical signals	attribute	*Devices that indicate with tactile signal that something is happening in the place where the transmitter is; they can transform, e.g. audible or visual signal to vibration or other tactile signal; Included are, e.g., indicating devices with vibration.*
218	Calendar software	attribute	*Software designed to help users to manage daily life. Included are also software or equipment for mobile phones, paging receivers etc.*
219	Electronic calendars	attribute	*Devices designed to help users to manage daily life. Usually, featuring a watch or maybe a calendar and telling the user when an activity is about to begin. The device may be stationary, portable or of pocket size. Output is available as text, speech, sound or symbols.*
221	Memory support products	attribute	*Devices for notifying or reminding a person about people, important activities or events of daily life; Included are, e.g., medication reminders, portable memo pads, memory support notebooks, talking picture systems and timed reminder systems.*
223	Activity monitoring systems without personal identification	attribute	*Alarm systems not featuring identification. The alarm is activated when a person leaves a certain area.*
224	Remote video monitoring systems	attribute	
225	Satellite navigation systems	attribute	*Monitoring and positioning systems that operate via satellite navigation. Included are, e.g., global positioning systems (GPS).*
227	Digital documents readers	attribute	*Software based systems able to transform digital documents (e.g. text files) into voice output*
228	Digital document reading (text to speech) service	attribute	*Web based services that transform digital documents into audio files*
230	Paper documents reading devices	attribute	*Hardware systems that transform the text written in paper document into alternative forms (e.g. enlarged text, synthetic speech, or tactile).*
231	OCR software	attribute	*Software used for the scanning and recognition of documents. Included is e.g. OCR software with text-to-speech technology.*
232	Portable scanner with electronic dictionary	attribute	*Portable devices featuring dictionary lookup.*

234	Braille note taking devices	attribute	*Electric portable note taking devices with Braille reading line*
236	Software interfaces for computers and mobile devices	attribute	*Complete software interfaces to facilitate the use of personal computer or mobile devices (e.g. tablet pc, smart phones).*
237	Operating systems	attribute	
239	Web browsers	attribute	*Web browsers with special features (e.g. voice output) to navigate the web.*
241	Keyboards with a special design	attribute	*Keyboards including e.g. enlarged and miniaturized keyboards, headpointer keyboards, ergonomic keyboards and one-hand keyboards.*
242	Programmable (concept) Keyboards	attribute	*Touch-sensitive programmable boards which can be divided into different numbers and sizes of active areas (keys). Each active area can be programmed to perform different actions.*
243	Keyboard shields and keyboard gloves	attribute	
244	Programmable keyboard configuration tool	attribute	*Software tools that allow to configure programmable keyboard and or to print overlay*
246	Software for accessing the computer in scanning mode	attribute	*Software that can be used, in combination with a switch, to control the computer in scanning mode.*
247	Eyegaze systems	attribute	*Systems that allow to control a computer, or other devices, through gaze*
248	Speech recognition software	attribute	*Software for command and control or text input to computers by speech (speech-to-text programs).*
249	Optical scanner, stationary	attribute	*Stationary devices that can transform text or illustrations printed on paper into an electronic format.*
250	Optical scanner, hand held	attribute	*Handheld devices that transform text or illustrations printed on paper into an electronic format.*
251	Datagloves	attribute	*Glove fitted with sensors, which records the movements of different parts of the glove and translates the movement to an input.*
252	EEG, EOG or EMG controlled input devices	attribute	*Input devices controlled by electric signals activated by brainwave signals (EEG), by facial muscle movements (EMG) or by eye movements (EOG).*
254	Switch interface	attribute	*Interface to connect switches to a device, to allow, for example, the control in scanning mode.*
255	Accessories for input devices	attribute	*Accessories used together with different types of input devices. Included are, e.g., adaptors, cables, boards, multi ports and joysticks.*
257	On screen keyboards	attribute	*Software applications that reproduce the keyboard on the device screen*
258	Mouse control software	attribute	*Software that allow controlling the mouse movement and/or click functions.*
259	Word prediction and word completion software	attribute	*Software designed to facilitate typing by completing words and/or predicting the next word in a sentence*
260	Software for adjusting input devices response	attribute	*Software that allow to modify the functioning and behavior of input devices (e.g. mouse, keyboard, switches, ...) through adjustments and filtering (e.g. filtering out involuntary repeated keypress, or allowing "hot keys" and "short cuts")*
261	Software based electronic dictionaries	attribute	*Electronic dictionaries working as independent programs or in conjunction with other software e.g. word processing software programs. Included are e.g. spelling dictionaries, foreign language dictionaries etc. Included are also picture-, symbol-, and sign language dictionaries.*
262	Computer based sound collections	attribute	*Collections of recorded words and sound effects for computers.*
264	Touch screens	attribute	*Touch screens consist of a touch sensitive display, divided into fields. The size, number and function of the fields can be customized.*
265	Trackballs, mousetrappers and touchpads	attribute	*A trackball in an upside-down mouse that rotates in place within a socket. The user rolls the ball to direct the cursor to the desired place on the screen. When using mouse trappers*

			and touch pads movement of the finger produces a corresponding cursor movement.
266	Traditional mouse devices and pen mouse devices	attribute	*Controlled by one hand. The mouse pointer is controlled by moving the mouse device on a given surface.*
267	Joystick mouse device	attribute	*Mouse devices with a joystick. Used to control the mouse pointer. Included are also mouth controlled joysticks.*
268	Switch operated computer mice	attribute	*Type of computer mouse where you can control all the mouse functions through switches.*
269	Computer and console joysticks	attribute	*Input devices, e.g. controllers, for playing electronic games on pc, Mac, Playstation, Nintendo, Xbox, or other platforms.*
271	Computer monitors	attribute	*Monitors for desktop computers.*
272	Screen filters	attribute	*Filters for computer monitors reducing specular reflection.*
274	Braille displays	attribute	*Displays converting text to Braille.*
276	Speech synthesizers	attribute	*Hardware or software system able to generate artificial human speech, also known as Text to Speech system*
278	Magnifying software	attribute	*Software that enlarges the text and graphics displayed on the screen of a computer or other electronic devices. May feature screen reading, colour choice and focus enhancement etc.*
279	Screen reader software	attribute	*Software that interpret what is being displayed on the screen and present it to the user with text-to-speech, sound icons, or a Braille output device.*
280	Software for adjusting color combination and text size	attribute	*Software that allow adjusting the color of text, background, images and other elements displayed on the screen, and/or to adjust the font size, to improve visualization.*
281	Software to modify the pointer appearance	attribute	*Software to modify the size, color, and/or shape of the pointer on the screen*
283	Single switches (switches with only one function)	attribute	*On/off switches (0/1 switches) which can be activated in different ways e.g. push activated, touch activated or sound activated etc. Single switches are used to control different products/assistive products.*
284	Two-four function control switches	attribute	*Switches controlling two to four functions.*
285	Five-or-more-function-contacts	attribute	*Five-or-more-function-contacts or wafer or star switch joysticks, where the function is similar to that of a digital joystick.*
287	Remote controller	attribute	
288	Receiver unit for environmental control	attribute	
289	Switch latches and timers	attribute	*Units controlling high current and low current devices with single switches.*
291	Environmental control software	attribute	*Software, standard or specifically designed, for controlling devices and automation systems.*
293	Software for composing music	attribute	*Software that allows a person to read and or compose music*
312	Body movement controlled mice	attribute	*Hardware devices that, using special sensors (e.g. video cameras, accelerometers, …), allow to control the mouse functions by moving a body part (e.g. the head)*
324	Tools and components for development of software products	attribute	*Tools and components for the development of accessible applications and assistive technology software products and services. Included are, for example, authoring tools for the development of accessible user interfaces*
315	**Operating systems**	cluster	
316	Windows	attribute	
317	Mac OS	attribute	
318	Linux	attribute	
319	Chrome OS	attribute	
320	iOS	attribute	
321	Android	attribute	
322	Windows mobile/phone	attribute	
323	Symbian	attribute	

ANNEX 2 - EASTIN Web service REST API implementation - Version 1.0

Introduction

In the following specifications the most widely accepted JSON formatting best practices have been adopted:

- All empty objects/properties in the JSON objects may be valorised to null or omitted.

- Numbers representing decimal values use the "." separator for the fractional part (ex: "value":3.56)

- The date properties must always be expressed in ISO 8601 format and use UTC time. The specific ISO format used is "yyyy-MM-ddTHH:mm:ss.fffz" (ex.: "insertDate":"2014-01-03T14:05:59.423Z") where:
  - yyyy: four digit year (ex.: 2016)
  - MM: two digit month (ex: 01 is January; the eventual leading zero must be specified)
  - dd: two digit day (ex: 03 is the third day of the month; the eventual leading zero must be specified)
  - HH: two digit hours in 24 hours format (ex.: 14 is two PM, 01 is one AM; the eventual leading zero must be specified)
  - mm: two digit minutes (ex.: 05; the eventual leading zero must be specified)
  - ss: two digit seconds (ex.: 59; the eventual leading zero must be specified)
  - fff: three digit milliseconds (ex.: 423, 120 (eventual ending zeros must be specified), 003 (eventual leading zeros must be specified)
  - z: a character indicating the time zone. It must always be equal to Z (indicating UTC time)

- For every method the results are embedded inside a wrapper JSON object with this format (many different implementations of this schema will be found in the examples below):
  ```
  {
  "apiVersion":"<API version>",
  "data": { <JSON object containing results> },
  "error": { <JSON object containing eventual errors> }
  }
  ```
  where **apiVersion** is always valorized (to "1.0" for the current version of the specs) while the valorizations of **data** and **error** are mutually exclusive. If present the error object has this format:
  ```
  "error":
  {
  "message":"<a string representing the error occurred>"
  }
  ```
  (even if it contains just a single property "error" has been designed as a complex JSON object because in future API versions other properties may be added)

- Each response should contain the proper HTTP status code, for example 200 for successful execution, 404 for resource not found, 500 for internal server error and so on.

Here below the detailed description of the implementation of the Web Service specs via a REST based API can be found.

Batch methods

Matching method in specifications: **GetIsoClassProductCount()**

URL:
http://<partner_server>/<partner_defined_subpath>/v1.0/isoclasses/productcount?iso=<iso_code_value>
(ex.: http://portale.siva.it/eastinwebapi/v1.0/isoclasses/productcount?iso=090603)

HTTP Verb:
GET

URL parameters:

- iso (ex.: iso=093603)

Request content parameters:
None.

Returns:
A JSON object. Ex.:

```
{
“apiVersion”:”1.0”,
"data":
{
“productCount”:156
},
"error":null                                   (this property may be omitted)
}
```

Notes
The **data** property is a JSON object containing a single property, productCount, which represents the method result.

Matching method in specifications: **GetIsoClassLocalization()**

URL:
http://<partner_server>/<partner_defined_subpath>/v1.0/isoclasses/localization?iso=<iso_code_value> (ex.: http://portale.siva.it/eastinwebapi/v1.0/isoclasses/localization?iso=090603)

HTTP Verb:
GET

URL parameters:

- iso (ex.: iso=093603)

Request content parameters:
None.

Returns:
A JSON object. Ex.:

```
{
“apiVersion”:”1.0”,
"data":
{
"isoCode":"09.36.03",
“title”:“Nail brushes”,
“scopeNote”:“Devices for scrubbing, cleaning and polishing nails; Assistive products to …”
},
"error":null                                    (this property may be omitted)
}
```

Notes
The **data** property in the returned JSON object is a JSON-serialized IsoClassLocalizationDto object.

Matching method in specifications: **GetKeywords()**

URL:
http://<partner_server>/<partner_defined_subpath>/v1.0/keywords (ex.:
http://portale.siva.it/eastinwebapi/v1.0/keywords)

HTTP Verb:
GET

URL parameters:
None

Request content parameters:
None.

Returns:
A JSON object: Ex.:

```
{
“apiVersion”:“1.0”,
"data":
{
“items”:
[
{
"keywordId":"E314R",
“text”:“wheelchairs”,
“isoCodes”:[“12.22.03”, “12.22.06”]
},
{
"keywordId":"E31234",
“text”:“poles”,
“isoCodes”:[“12.22.03”, “12.22.06”]
},
…
{
"keywordId":"DF234",
“text”:“hoisters”,
“isoCodes”:[“12.22.03”, “12.22.06”]
}
]
},
“error”:null                              (this property may be omitted)
}
```

Notes
The **data.items** property in the returned JSON object is an array of JSON-serialized KeywordDto objects.

Live search methods

1. Product searches

Matching method in specifications: **FindSmallProducts()**

URL:
http://<partner_server>/<partner_defined_subpath>/v1.0/products (ex.:
http://portale.siva.it/eastinwebapi/v1.0/products)

HTTP Verb:
POST

URL parameters:
None

Request content parameters:
A JSON object. Ex.:

```
{
“apiVersion”:“1.0”,
“params”:
{
“isoCodes”:[“12.22.03”, “12.22.06”],
“feaures”:
[
{ “featureId”:122, “valueMin”:0.0 “valueMax”:0.0 },
{ “featureId”:2, “valueMin”:8.0, “valueMax”:100.5 }
],
“commercialName”:“ministar”,
“manufacturer”:“offcarr”,
“insertDateMin”:“2014-03-31T13:22:05.245Z”,
“insertDateMax”:“2016-12-01T17:22:56.941Z”
}
}
```

Returns:
A JSON object. Ex.:

```
{
“apiVersion”:“1.0”,
"data":
{
“items”:
[
{
"productCode":"47056",
"isoCodePrimary":"12.22.03",
"isoCodesOptional":[“12.22.06”, “12.22.09”],
"commercialName":"Cleo ultralet kørestol til børn",
"manufacturerCode":"1A2",
"manufacturerOriginalFullName":"Sunrise Medical B.V.",
"insertDate":"2015-10-30T09:37:50.130Z",
"lastUpdateDate":"2016-11-30T10:33:38.204Z",
"thumbnailImageUrl": "http://portale.siva.it/files/images/product/thumbs/18459_s.jpg"
},
{
"productCode":"E23091",
"isoCodePrimary":"12.22.03",
"isoCodesOptional":[“12.22.06”],
"commercialName":"Quickie Neon² Swing Away",
"manufacturerCode":"1A2",
"manufacturerOriginalFullName":"Sunrise Medical B.V.",
"insertDate":"2006-11-13T09:37:50.031Z",
"lastUpdateDate":"2016-11-28T09:37:50.123Z",
"thumbnailImageUrl": "http://portale.siva.it/files/images/product/thumbs/28659_s.jpg"
},

...

{
"productCode":"E6501",
"isoCodePrimary":"12.22.03",
"isoCodesOptional":[“12.22”, “12.22.06”, “12.22.09”],
"commercialName":"Küschall Compact / Compact Junior 2009",
"manufacturerCode":"R13",
"manufacturerOriginalFullName":"Offcarr",
"insertDate":"1995-10-31T00:00:00.000Z",
"lastUpdateDate":"2016-11-28T00:00:00.000Z",
"thumbnailImageUrl": "http://portale.siva.it/files/images/product/thumbs/1839_s.jpg"
}
]
},
“error”:null                              (this property may be omitted)
}
```

Notes

Any of the fields in the **params** JSON object (representing the method’s parameters) may be empty; if no field is valorised, such as in this example:

```
{
“apiVersion”:“1.0”,
“params”:null                             (this property may be omitted)
}
```

the method should return all Products available in the db. The **isoCodes** array contains the ISO codes (specified as string). The **features** array contains the JSON serialization of FeatureDto objects (see the general specs above for the complete description). If a feature is of type **Attribute** its values from and to will be both 0.0 and won’t be considered in the query building process; if a feature is of type **Measure** than see the specs for **FindSmallProducts()** above for the detailed query building criteria. Feature of type **Cluster** will never be used as parameter. For a complete reference about Features see the **ANNEX - EASTIN feature vocabulary** above.

The **data.items** property in the returned JSON object is an array of JSON serialized SmallProductDto objects.

Matching method in specifications: **GetProduct()**

URL:

http://<partner_server>/<partner_defined_subpath>/v1.0/products/<productCode> (ex.: http://portale.siva.it/eastinwebapi/v1.0/products/EF4234)

HTTP Verb:

GET

URL parameters:

- productCode (ex.: EF4234)

Request content parameters:

None

Returns:

A JSON object: Ex.:

```
{
“apiVersion”:“1.0”,
“data”:
{
“productCode”:”EF4234”,
"isoCodePrimary":"12.22.03",
"isoCodesOptional":[“12.22.06”, “12.22.09”],
"commercialName":"Cleo ultralet kørestol til børn",
```

"manufacturerCode":"12",
"manufacturerOriginalFullName":"Sunrise Medical B.V.",
"insertDate":"2015-10-30T09:37:50.130Z",
"lastUpdateDate":"2016-11-30T10:33:38.204Z",
"thumbnailImageUrl":"http://portale.siva.it/files/images/product/thumbs/18659_s.jpg",
“isReviewAllowed”:true,
“manufacturerAddress”:“Via Trasimeno 3”,
“manufacturerPostalCode”:“20100”,
“manufacturerTown”:“Milano”,
“manufacturerCountry”:“IT”,
“manufacturerPhone”:“+39 02 419 2249”,
“manufacturerFax”:“+39 02 419 2224”,
“manufacturerEmail”: “info@sunrise.com”,
“manufacturerSkype”: “sunriseSkype”,
“manufacturerWebSiteUrl”:“http://www.sunrise.com”,
“manufacturerSocialNetworkUrls”:[“http://www.facebook.com/meyra/”, “http://www.linkedin.com/meyra”],
“imageUrl”: "http://portale.siva.it/files/images/product/thumbs/18659_b.jpg",
“originalDescription”:“Questo prodotto è composto da…”,
“englishDescription”:“This product is made of…”,
“originalUrl”:“http://portale.siva.it/it-IT/databases/products/detail/id-18564”,
“englishUrl”:“http://portale.siva.it/en-GB/databases/products/detail/id-18564”,
“originalDownloadUrl”:“http://portale.siva.it/it-IT/databases/products/download/id-18564”,
“englishDownloadUrl”:“http://portale.siva.it/en-GB/databases/products/download/id-18564”,
“userManualUrls”:[“http://www.someurl1.com”, “http://www.someurl2.com”],
“videoUrls”: [“http://www.someurl3.com”, “http://www.someurl4.com”],
“brochureUrls”: [“http://www.someurl5.com”, “http://www.someurl6.com”],
“furtherInfoUrls”: [“http://www.someurl7.com”, “http://www.someurl8.com”],
“features”:
[
{ “featureId”:122, “valueMin”:0.0, “valueMax”:0.0 },
{ “featureId”:2, “valueMin”:50.3”, “valueMax”:200.15 },
{ “featureId”:7, “valueMin”:1.0”, “valueMax”:10.0 }
]
},
“error”:null
}

Notes
The **data** property in the returned JSON object is a JSON serialized ProductDto object.

2. Actor searches

Matching method in specifications: **FindSmallActors()**

URL:
http://<partner_server>/<partner_defined_subpath>/v1.0/actors (ex.:
http://portale.siva.it/eastinwebapi/v1.0/actors)

HTTP Verb:
POST

URL parameters:
None

Request content parameters:
A JSON object. Ex.:

```
{
“apiVersion”:“1.0”,
“params”:
{
“actorType”:”serviceproviders”,
“isoCodes”:[“12.22.03”, “12.22.06”],
“icfCodes”: [“b1”, “d2”],
“actorName”:“me”,
“insertDateMin”:“2014-03-31T13:22:05.245Z”,
“insertDateMax”:“2016-12-01T17:22:56.941Z”
}
}
```

Returns:
A JSON object: Ex.:

```
{
“apiVersion”:“1.0”,
“data”:
{
“items”:
[
{
"actorCode ":"F356",
"originalFullName ":"Meyra Inc.",
"country":“DE”,
"insertDate":"2014-10-30T09:37:50.130Z",
"lastUpdateDate":"2015-11-30T10:33:38.204Z"
},
{
"actorCode ":"OJ2343456",
"originalFullName ":"Merac ltd.",
"country":“UK”,
"insertDate":"2013-10-30T09:37:50.130Z",
"lastUpdateDate":"2014-11-30T10:33:38.204Z"
},

...

{
```

```
"actorCode ":"23456",
"originalFullName ":"Melt Spa.",
"country":“IT”,
"insertDate":"2011-10-30T09:37:50.130Z",
"lastUpdateDate":"2016-11-30T10:33:38.204Z"
}
]
},
"error":null                              (this property may be omitted)
}
```

Notes
Any of the fields in the **params** JSON object (representing the method’s parameters) except **actorType** may be empty; if no other field is valorised, such as in this example:

```
{
“apiVersion”:“1.0”,
“params”:
{
“actorType”:“companies”,
“isoCodes”:null,                          (this property may be omitted)
“icfCodes”:null,                          (this property may be omitted)
“actorName”:null,                         (this property may be omitted)
“insertDateMin”:null,                     (this property may be omitted)
“insertDateMax”:null                      (this property may be omitted)
}
}
```

the method should return all Actors of the specified type available in the db. For the parameters the same considerations stand as for **FindSmallActors()** specs (see above).

The **data.items** property in the returned JSON object is an array of JSON serialized SmallActorDto objects.

Matching method in specifications: **GetActor()**

URL:
http://<partner_server>/<partner_defined_subpath>/v1.0/actors/<actorType>/<actorCode> (ex.: http://portale.siva.it/eastinwebapi>/v1.0/actors/companies/1R233)

HTTP Verb:
GET

URL parameters:
- actorType (ex.: companies)
- actorCode (ex.: 1R233)

Request content parameters:
None

Returns:
A JSON object: Ex.:

```
{
“apiVersion”:“1.0”,
“data”:
{
“actorCode”:”1R233”,
"originalFullName":"Meyra Srl",
"country":"IT",
"insertDate":"2015-10-30T09:37:50.130Z",
"lastUpdateDate":"2016-11-30T10:33:38.204Z",
"shortName":"Meyra",
“englishFullName”:“Meyra Gmbh”,
“originalDescription”:“Questa azienda è…”,
“englishDescription”:“This company is…”,
"startDate":"2015-10-30T09:37:50.130Z",
"endDate":"2016-11-30T10:33:38.204Z",
“contactBody”:“Meyra International”,
“address”:“Via Trasimeno 3”,
“postalCode”:“20100”,
“town”:“Milano”,
“phone”:“+39 02 419 2249”,
“fax”:“+39 02 419 2224”,
“email”:“info@sunrise.com”,
“skype”:“meyraSkype”,
“webSiteUrl”:“http://www.meyra.com”,
“contactPersonFullName”: “Mr. John Smith”,
“originalUrl”:“http://www.meyra.com/it”,
“englishUrl”:“http:// www.meyra.com/en”,
“socialNetworkUrls”:[“http://www.facebook.com/meyra/”, “http://www.linkedin.com/meyra”],
“icfCodes”: [“b1”, “d2”],
“isoCodes”:[“12.22.03”, “12.22.06”]
},
“error”:null                              (this property may be omitted)
}
```

Notes
The **data** property in the returned JSON object is a JSON serialized ActorDto object.

3. Associated information searches

Matching method in specifications: **FindSmallAssociatedInfos()**

URL:
http://<partner_server>/<partner_defined_subpath>/v1.0/associatedinfo (ex.: http://portale.siva.it/eastinwebapi/v1.0/associatedinfo)

HTTP Verb:
POST

URL parameters:
None

Request content parameters:
A JSON object. Ex.:

```
{
“apiVersion”:“1.0”,
“params”:
{
“infoType”:”articles”,
“isoCodes”:[“12.22.03”, “12.22.06”],
“icfCodes”: [“b1”, “d2”],
“title”:“Disabilità e lavoro”,
“author”:“Andrich”,
“insertDateMin”:“2014-03-31T13:22:05.245Z”,
“insertDateMax”:“2016-12-01T17:22:56.941Z”
}
}
```

Returns:
A JSON object: Ex.:

```
{
“apiVersion”:“1.0”,
"data":
{
“items”:
[
{
"associatedInfoCode ":"A34R324",
"authors":"R. Andrich, V. Gower",
"originalTitle":“Disabilità e lavoro”,
"englishTitle":“Disability and work”,
"originalLanguage":“it”,
"insertDate":"2014-10-30T09:37:50.130Z",
"lastUpdateDate":"2015-11-30T10:33:38.204Z"
},
{
"associatedInfoCode ":"235ERWT5",
"authors":"R. Andrich, V. Gower",
"originalTitle":“Disabilità e lavoro”,
"englishTitle":“Disability and work”,
"originalLanguage":“it”,
"insertDate":"2014-10-30T09:37:50.130Z",
"lastUpdateDate":"2015-11-30T10:33:38.204Z"
},
...

{
"associatedInfoCode ":"3455DFGSG",
"authors":"R. Andrich, V. Gower",
"originalTitle":“Disabilità e lavoro”,
"englishTitle":“Disability and work”,
"originalLanguage":“it”,
"insertDate":"2014-10-30T09:37:50.130Z",
"lastUpdateDate":"2015-11-30T10:33:38.204Z"
}
]
},
"error":null                          (this property may be omitted)
}
```

Notes
Any of the fields in the **params** JSON object (representing the method’s parameters) except **infoType** may be empty; if no other field is valorised, such as in this example:

```
{
“apiVersion”:“1.0”,
“params”:
{
“infoType”:“articles”,
“isoCodes”:null,                      (this property may be omitted)
“icfCodes”:null,                      (this property may be omitted)
“title”:null,                         (this property may be omitted)
“author”:null,                        (this property may be omitted)
“insertDateMin”:null,                 (this property may be omitted)
“insertDateMax”:null                  (this property may be omitted)
}
}
```

the method should return all AssociatedInfo of the specified type available in the db. For the parameters the same considerations stand as for **FindSmallAssociatedInfos()** specs (see above).

The **data.items** property in the returned JSON object is an array of JSON serialized SmallAssociatedInfoDto objects.

Matching method in specifications: **GetAssociatedInfo()**

URL:
http://<partner_server>/<partner_defined_subpath>/v1.0/associatedinfo/<infoType>/<associatedInfoCode>
(ex.: http://portale.siva.it/eastinwebapi/v1.0/associatedinfo/articles/1ERT244S)

HTTP Verb:

GET

URL parameters:

- infoType (ex.: articles)
- associatedInfoCode (ex.: 1ERT244S)

Request content parameters:
None

Returns:
A JSON object: Ex.:

```
{
“apiVersion”:“1.0”,
“data”:
{
“associatedInfoCode”:”1ERT244S”,
"authors":"R. Andrich, V. Gower",
"originalTitle":“Disabilità e lavoro”,
"englishTitle":“Disability and work”,
"originalLanguage":“it”,
"insertDate":"2014-10-30T09:37:50.130Z",
"lastUpdateDate":"2015-11-30T10:33:38.204Z",
"publicationYear":2011,
“publishingDetails”:“Associated Press”,
“originalAbstract”:“Questo articolo parla di…”,
“englishAbstract”:“This article talks about…”,
“originalUrl”:“http://portale.siva.it/it/associatedinfo/articles/1ERT244S”,
“englishUrl”:“ http://portale.siva.it/en/associatedinfo/articles/1ERT244S”,
“originalDownloadUrl”:“http://portale.siva.it/it/associatedinfo/articles/1ERT244S/download”,
“englishDownloadUrl”:“ http://portale.siva.it/en/associatedinfo/articles/1ERT244S/download”,
“imageUrl”:“http://portale.siva.it/en/associatedinfo/articles/1ERT244S_b.jpg”,
“furtherInfoUrls”: [“http://www.someurl1.com”, “http://www.someurl2.com”],
“icfCodes”: [“b1”, “d2”],
“isoCodes”:[“12.22.03”, “12.22.06”]
},
“error”:null                                    (this property may be omitted)
}
```

Notes
The **data** property in the returned JSON object is a JSON serialized AssociatedInfoDto object.