Versione

3.2

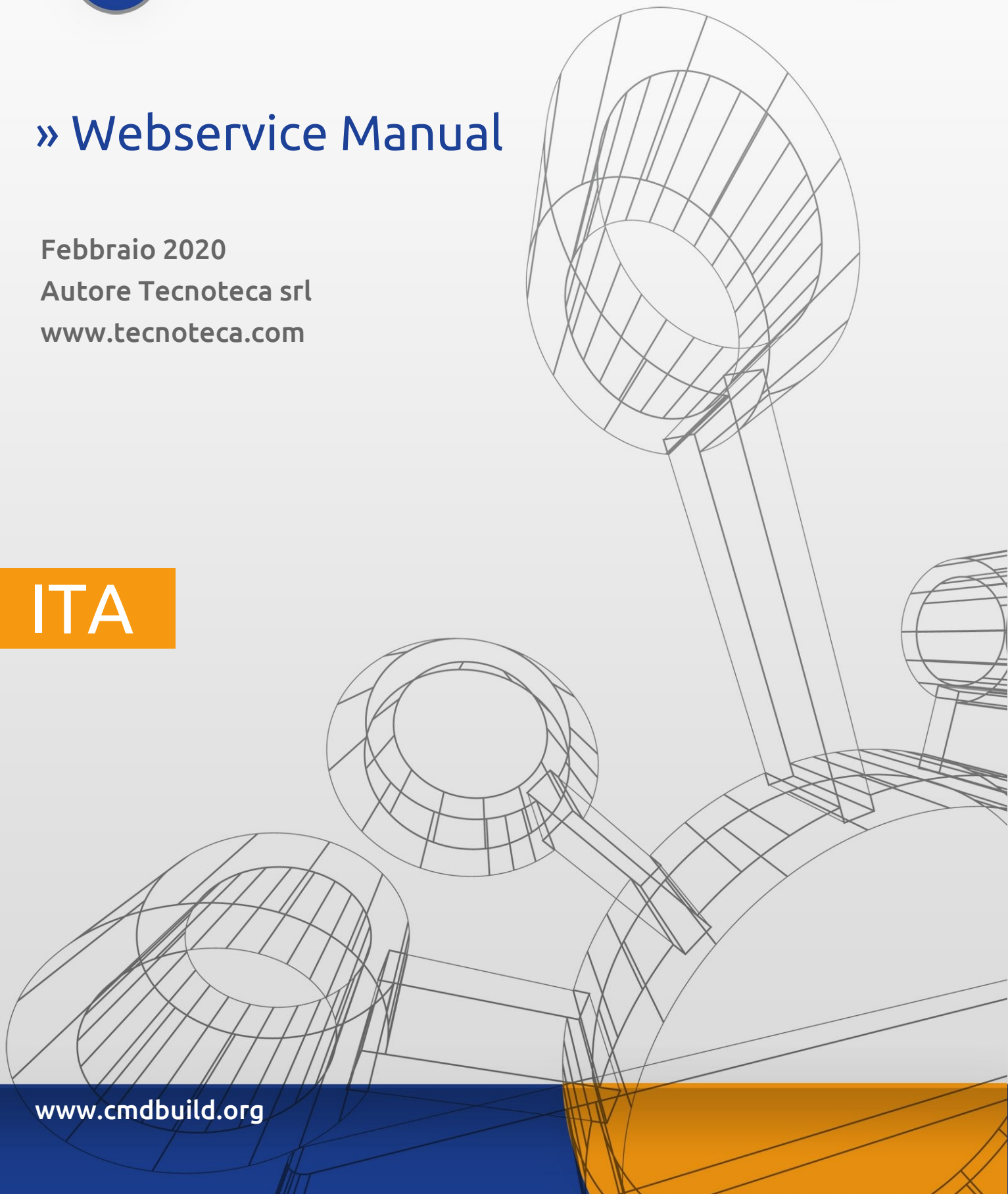
CMDBuild®

» Webservice Manual

Febbraio 2020

Autore Tecnoteca srl
www.tecnoteca.com

ITA



CMDBuild ® uses many great technologies from the open source community:
PostgreSQL, Apache, Tomcat, Eclipse, Ext JS, JasperSoft, JasperStudio, Enhydra Shark, TWE, OCS Inventory, Liferay, Alfresco, GeoServer, OpenLayers, Quartz, BiMserver.
We are thankful for the great contributions that led to the creation of these products.

CMDBuild ® è un prodotto di Tecnoteca S.r.l. che ne ha curato la progettazione e realizzazione, è maintainer dell'applicazione e ne ha registrato il logo.

Il sito ufficiale di CMDBuild è http://www.cmdbuild.org

Contenuti

1. Introduzione

1.1. Descrizione dell’applicazione

CMDBuild è un ambiente web open source tramite cui è possibile configurare applicazioni personalizzate per l'Asset Management.

Da un lato dispone di meccanismi nativi per l’amministratore, implementati in un codice “core” mantenuto separato dalla logica di business, per configurare il sistema in tutte le sue funzionalità.

Dall’altro genera dinamicamente una interfaccia web per gli operatori, consentendo loro di mantenere sotto controllo la situazione degli asset, di conoscerne in ogni momento la composizione, la dislocazione, le relazioni funzionali e le modalità di aggiornamento nel tempo, per gestirne il ciclo di vita in modo completo.

L’amministratore del sistema può costruire ed estendere autonomamente il proprio CMDB (da cui il nome del progetto), modellandolo su misura della propria organizzazione tramite una apposita interfaccia che consente di aggiungere progressivamente nuove classi di oggetti, nuovi attributi e nuove tipologie di relazioni. Può definire filtri, “viste” e permessi di accesso ristretti a righe e colonne di ciascuna classe.

L’amministratore può disegnare in modo visuale, con un editor esterno, workflow operanti sulle classi modellate nel database, importarli in CMDBuild e metterli a disposizione degli operatori che li eseguiranno secondo i flussi previsti e con gli automatismi configurati.

In modo analogo può disegnare in modo visuale, con un editor esterno, report di diverso genere (tabulati, stampe con grafici, etichette barcode, ecc) sui dati del CMDB, importarli nel sistema e metterli a disposizione degli operatori.

Può poi configurare delle dashboard, costituite da grafici che mostrino in modo immediato la situazione di alcuni indicatori dello stato corrente del sistema (KPI).

Un task manager incluso nell'interfaccia utente del Modulo di Amministrazione consente di schedulare in background diverse tipologie di operazioni (avvio di processi, ricezione e invio di mail, esecuzione di connettori) e di controlli sui dati del CMDB (eventi sincroni e asincroni) a fronte dei quali inviare notifiche, avviare workflow ed eseguire script.

Grazie all'integrazione con sistemi documentali che supportano lo standard CMIS (Content Management Interoperability Services), fra cui la diffusissima soluzione open source Alfresco, gli operatori potranno allegare alle schede archiviate in CMDBuild documenti, immagini, video ed altre tipologie di file.

E' anche possibile utilizzare funzionalità GIS per il georiferimento degli asset e la loro visualizzazione su una mappa geografica (servizi mappe esterni) e/o su planimetrie vettoriali (server locale GeoServer e database spaziale PostGIS) e funzionalità BIM per la visualizzazione di modelli 3D basati su file in formato IFC.

E’ poi incluso nel sistema un webservice REST, tramite cui gli utilizzatori di CMDBuild possono implementare soluzioni personalizzate di interoperabilità con sistemi esterni.

CMDBuild comprende inoltre due framework esterni:

- il CMDBuild Advanced Connector, scritto in linguaggio Java e configurabile in Groovy, che tramite tramite logiche native per la sincronizzazione di dati agevola la implementazione di connettori con fonti dati esterne, ad esempio con sistemi di automatic inventory o di virtualizzazione o di monitoraggio (fornito con licenza non open source a chi sottoscrive la Subscription annuale con Tecnoteca)

- il CMDBuild GUI Framework, che agevola la implementazione di interfacce grafiche aggiuntive, ad esempio pagine web semplificate per utenti non tecnici, da pubblicare su portali esterni (suggerita la soluzione open source Liferay) ed in grado di interagire con il CMDB tramite il webservice REST

CMDBuild dispone infine di una interfaccia “mobile” (per smartphone e tablet), implementata come “APP” multipiattaforma (iOS, Android) ed anch'essa in grado di interagire con il CMDB tramite il webservice REST (fornita con licenza non open source a chi sottoscrive la Subscription annuale con Tecnoteca).

CMDBuild è un sistema web enterprise: Java lato server, GUI web Ajax, architettura SOA (Service Oriented Architecture) basata su webservice, implementato riutilizzando le migliori tecnologie open source e seguendo gli standard di settore.

CMDBuild è un sistema in continua evoluzione, rilasciato per la prima volta nel 2006 ed aggiornato con più rilasci annuali per offrire sempre nuove funzionalità ed il supporto delle nuove tecnologie.

1.2. Sito web del progetto

CMDBuild dispone di un sito web dedicato al progetto: http://www.cmdbuild.org

Il sito raccoglie una estesa documentazione per chi desidera approfondire la caratteristiche tecniche e funzionali del progetto: brochure, slide, manuali (vedi paragrafo successivo), testimonianze, case history, newsletter, forum.

1.3. I moduli di CMDBuild

Il sistema CMDBuild comprende due moduli principali:

- il Modulo di Amministrazione, dedicato alla definizione iniziale ed alle successive modifiche del modello dati e delle configurazioni di base (classi e tipologie di relazioni, utenti e permessi, dashboard, upload report e workflow, opzioni e parametri)
- il Modulo di Gestione dati, dedicato alla consultazione ed aggiornamento delle schede e delle relazioni nel sistema, alla gestione di documenti allegati, all'avanzamento dei processi, alla visualizzazione di dashboard e produzione di report

Il Modulo di Amministrazione è riservato agli utenti abilitati al ruolo di amministratore, il Modulo di Gestione è utilizzato dagli operatori addetti alla consultazione ed aggiornamento dei dati.

1.4. Manualistica disponibile

Il presente manuale è dedicato alla descrizione del Modulo di Amministrazione, tramite cui il gestore del sistema può eseguire la configurazione del modello dati, definire utenti e permessi ed eseguire altre attività di servizio.

Sono disponibili sul sito di CMDBuild (http://www.cmdbuild.org) ulteriori manuali tecnici dedicati a:

- overview concettuale del sistema (“Overview Manual”)
- amministrazione del sistema (“Administrator Manual”)
- installazione e gestione tecnica del sistema (“Technical Manual”)
- configurazione dei workflow (“Workflow Manual”)
- utilizzo del webservice per l'interoperabilità con sistemi esterni (“Webservice Manual”)
- utilizzo di connettori per la sincronizzazione di dati con sistemi esterni (“Connectors Manual”)

1.5. Applicazioni basate su CMDBuild

Tecnoteca ha utilizzato il proprio ambiente CMDBuild per implementare due diverse soluzioni preconfigurate:

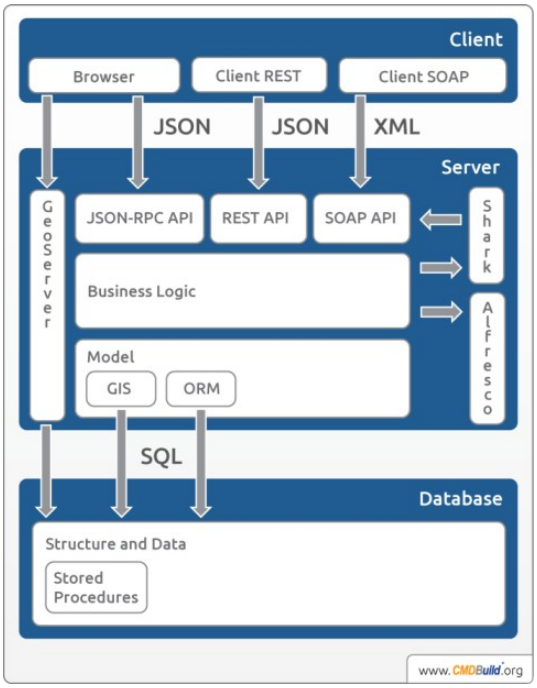
- CMDBuild READY2USE, per la gestione degli asset e dei servizi IT, orientato ad infrastrutture IT interne o a servizi erogati a clienti esterni (http://www.cmdbuild.org/it/prodotti/ready2use) secondo le best practice ITIL (Information Technology Infrastructure Library)
- openMAINT, per la gestione dell'inventario di asset ed impianti di patrimoni immobiliari e delle relative attività di manutenzione preventiva e a guasto (http://www.openmaint.org)

Entrambe le applicazioni sono rilasciate con licenza open source, con esclusione di alcune componenti esterne (connettori di sincronizzazione dati, portale Self-Service, APP mobile, ecc), riservate a chi sottoscrive la Subscription annuale con Tecnoteca.

2. Criteri di interoperabilità

2.1. Architettura SOA

Per rendere interoperabili applicazioni diverse è necessario che queste siano realizzate sotto forma di componenti cooperanti alla implementazione di servizi e che tali servizi vengano esposti attraverso interfacce di alto livello definite secondo protocolli standard.



CMDBuild è stato progettato in architettura SOA (Service Oriented Architecture):

- disaccoppiando i diversi livelli logici (vedi schema)
- implementando ed esponendo in ciascuno interfacce esterne specifiche come unica modalità di accesso ai relativi dati e metodi
- utilizzando le stesse interfacce sia per accesso interattivo da parte del client web che per accesso programmatico da parte di applicazioni esterne

Dal punto di vista tecnico è stato scelto di utilizzare la seguenti tecnologie dei web service:

1. protocollo SOAP
2. protocollo REST

Attraverso i web service, e compatibilmente con le politiche di sicurezza definite, CMDBuild rende quindi disponibili i i dati archiviati nel CMDB ed i relativi metodi di gestione per consentirne l'utilizzo nell'ambito di altre applicazioni interessate alle stesse informazioni, sia di gestione tecnica che dedicate ad esigenze amministrative.

3. Web services

3.1. Introduzione ai Web service

Un web service è un'interfaccia che descrive una collezione di operazioni, accessibili attraverso una rete mediante messaggistica XML.

Tramite un web service una applicazione può rendere accessibili le proprie funzionalità ad altre applicazioni operanti attraverso il web.

Le soluzioni più utilizzate ai giorni nostri sono:

- Web services SOAP
- Web Services REST

In questo capitolo entrambi gli standard verranno introdotti, con una lista delle funzioni avviabili e qualche esempio.

3.2. Introduzione Web Services SOAP

SOAP (Simlpe Object Access Protocol) è un protocollo basato sul linguaggio XML. Grazie all'utilizzo dell'XML, SOAP ci permette, a differenza di molti altri framework, una comunicazione indipendente dalla piattaforma e dal linguaggio utilizzato per scrivere i nostri applicativi.

La struttura di un messaggio SOAP è divisa in quattro parti:

- Envelope: un elemento obbligatorio che specifica l'inizio e la fine del messaggio;
- Header: un elemento falcoltativo che può contenere attributi opzionali;
- Body: un elemento obbligatorio che contiene l'effettivo messaggio che viene inviato;
- Fault: un elemento che fornisce eventuali errori generati durante il processing del messaggio;

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:soap="http://soap.services.cmdbuild.org">
   <soapenv:Header>
        ...
   <soapenv:Header/>
   <soapenv:Body>
        <soapenv:Fault>
        ...
        </soapenv:Fault>
   ...
   </soapenv:Body>
</soapenv:Envelope>
```

Nell'utilizzo di SOAP all'interno di CMDBuild, l'header verrà usato principalmente per autenticare l'utente, aggiungendo un campo security dove verranno forniti lo username e la password per poter utilizzare il servizio.

Nel campo body l'utente potrà specificare la funzione che dovrà essere chiamata, con la seguente sintassi:

```
<soap:functionName/>
```

Come esempio verrà ora riportata una richiesta SOAP per generare una sessione per l'utente specificato nell'header:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:soap="http://soap.services.cmdbuild.org">
   <soapenv:Header>
       <wsse:Security soapenv:mustUnderstand="1"
xmlns:wsse="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-
secext-1.0.xsd" xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-
wss-wssecurity-utility-1.0.xsd">
       <wsse:UsernameToken>
       <wsse:Username>username</wsse:Username>
       <wsse:Password Type="http://docs.oasis-open.org/wss/2004/01/oasis-
200401-wss-username-token-profile-1.0#PasswordText">password</wsse:Password>
 </wsse:UsernameToken></wsse:Security>
   <soapenv:Header/>
   <soapenv:Body>
      <soap:createSession/>
   </soapenv:Body>
</soapenv:Envelope>
```

Quando una richiesta SOAP viene eseguita, l'endpoint a cui deve'essere fatta questa richiesta, nel caso di CMDBuild, è il file Private.wsdl. Questo file fornisce una lista di tutte le funzioni avviabili dall'utente.

Per una descrizione più dettagliata di che chiamate poter fare con SOAP su CMDBuild, fare riferimento al capitolo 4.

3.3. Introduzione Web Services REST

REST (REpresentational State Transfer), diversamente da SOAP, è uno stile architetturale che fornisce un sistema di comunicazione senza stato, semplice e leggero

Il formato utilizzato per mandare e ricevere informazioni con un sistema REST è il JSON. Tale formato è equiparabile ad un semplice testo con una serie di coppie chiave-valore, come la seguente:

```
{
 "Key1":"value1",
 "Key2":"value2",
 …
}
```

Quando utilizziamo il sistema REST, le richieste possono essere effettuate ad una serie di endpoint, tramite semplici richieste HTML GET, PUT, POST, DELETE.

Oggetti come token di autenticazione, possono essere aggiunti nell'header della richiesta, eventuali dati che devono essere inviati in richieste PUT o POST possono essere aggiunti come parametri della richiesta o nel body della stessa.

Come esempio, se dovessimo generare una sessione come fatto precedentemente con i web services SOAP, dovremmo trovare l'endpoint relativo alle sessioni:

http://hostname:port/cmdbuild/services/rest/v3/sessions

Per poi effettuare una richiesta POST fornendo lo username e la password dell'utente da autenticare.

La risposta conterrà nel campo data le varie informazioni utili, fra cui un campo "sessionId" che conterrà l'id della sessione appena generata. (dalla versione 3.2, a causa di problemi di sicurezza, un parametro aggiuntivo va allegato alla richiesta per poter ottenere il sessionId in chiaro nella risposta, maggiori informazioni al capitolo 5.3.1)

```
{
 "success":"true",
 "data":["sessionId":"generatedSessionId",
       "username":"user",
       "password":"userPassword"]
}
```

Per una descrizione più dettagliata delle varie funzioni rest, fare riferimento al capitolo 5, per alcuni esempi di chiamate e risposte REST, fare riferimento al capitolo 6.

4. Web services SOAP

4.1. CMDBuild WSDL

Per ottenere una lista completa di tutte le funzioni disponibili chiamabili tramite webservice SOAP, CMDBuild fornisce un file wsdl denominato Private.wsdl. Può essere aperto da un normale editor di testo per visualizzare l'XML sorgente oppure può essere aperto da software quali SoapUI per avere una visualizzazione più semplificata del contenuto.

Nei prossimi paragrafi una lista di tutte le funzioni avviabili e una breve descrizione delle stesse verrà fornita.

4.2. Funzioni SOAP

Nelle tabelle che verranno presentate vi saranno descritte tutte le funzioni SOAP divise per categoria, bisogna tenere in considerazione che alcune funzioni possano cambiare in seguito ad aggiornamenti futuri, quindi sarà sempre buona norma verificare il formato di tali funzioni nel file WSDL.

4.2.1. Cards

Struttura dati cards:

- className
- id
- attributeList:
- beginDate
- user

Funzione	Parametri	Descrizione
getCard	-className -cardId -attributeList	Funzione usata per ottenere informazioni relative ad una specifica card appartenente ad una specifica classe
getCardHistory	-className -cardId -limit -offset	Funzione usata per ottenere lo storico di una specifica card appartenente ad una specifica classe. Con i parametri limit e offset i risultati possono essere filtrati
getCardList	-className -attributeList -queryType -orderType -limit -offset -fullTextQuery -cqlQuery	Funzione usata per ottenere una lista completa di cards appartenenti ad una classe specifica, con la possibilità di filtrare i risultati con i vari parametri forniti
getCardListExt	-className -attributeList	Funzione usata per ottenere una lista completa estesa di cards appartenenti ad una

	-queryType -orderType -limit -offset -fullTextQuery -cqlQuery	classe specifica
getCardListWithLongDateFormat	-className -attributeList -queryType -orderType -limit -offset -fullTextQuery -cqlQuery	Funzione usata per ottenere una lista completa di cards appartenenti ad una specifica classe con un formato long date format per la data
getCardMenuSchema		
createCard	-className -attributeList -beginDate -endDate -metadata	Funzione usata per creare una nuova card con i dati forniti nella richiesta
updateCard	-className -attributeList -beginDate -endDate -id -metadata	Funzione usata per aggiornare i dati di una card con i dati specificati nella richiesta
deleteCard	-className -cardId	Funzione usata per eliminare una card specifica

4.2.2. Sessioni

Funzione	Parametri	Descrizione
createSession		Crea una sessione per l'utente specificato nell'header della richiesta

4.2.3. Lookups

Struttura dati lookup:

- id
- type
- description
- code
- parent
- parentId
- position

- notes

Funzione	Parametri	Descrizione
getLookupById	-id	Funzione usata per ottenere una lookup con l'id specificato
getLookList		Funzione usata per ottenere l'intera lista di lookup usabili
getLookListByCode	-type -code	Funzione usata per ottenere una lista di lookup con un tipo specifico e un code specifico
createLookup	-code -description -notes -parent -type	Funzione usata per creare una nuova lookup con i valori specificati nei parametri
updateLookup	-code -description -id -notes -parent -type	Funzione usata per aggiornare una lookup con uno specifico id con i valori forniti nei parametri
deleteLookup	-id	Funzione usata per eliminare una lookup con uno specifico Id

4.2.4. Attributi

Struttura dati attributi:

- name
- value
- code

Funzione	Parametri	Descrizione
getAttributeList	-className	Funzione usata per ottenere una lista di tutti gli attributi relativi ad una classe specifica

4.2.5. Relazioni

Struttura dati relazioni:

- domainName
- class1Name
- card1Id
- class2Name
- card2Id

Funzione	Parametri	Descrizione
getRelationAttributes	-class1Name -class2Name -card1Id -card2Id -domainName	Funzione usata per ottenere gli attributi di una specifica relazione
getRelationHistory	-class1Name -class2Name -card1Id -card2Id -domainName	Funzione usata per ottenere lo storico di una specifica relazione
getRelationList	-className -cardId	Funzione usata per ottenere la lista completa di tutte le relazioni relative alla classe e card specificata
getRelationListExt	-domain -className -cardId	Funzione usata per ottenere la lista completa ed estesa di tutte le relazioni relative alla classe e card specificata
createRelation	-class1Name -class2Name -card1Id -card2Id -domainName	Funzione usata per creare una relazione fra due card specifiche
createRelationWithAttributes	-class1Name -class2Name -card1Id -card2Id -domainName -attributes	Funzione usata per creare una relazione fra due card specifiche con l'aggiunta degli attributi specificati
updateRelationAttributes	-class1Name -class2Name -card1Id -card2Id -domainName -attributes	Funzione usata per aggiornare gli attributi di una specifica relazione
deleteRelation	-class1Name -class2Name -card1Id -card2Id -domainName	Funzione usata per eliminare una relazione specifica

4.2.6. Classi

Funzione	Parametri	Descrizione
getClassSchema	-className	Funzione usata per ottenere lo schema di una classe specifica
getClassSchemaById	-classId -includeAttributes	Funzione usata per ottenere lo schema di una classe specifica con l'aggiunta eventuale di

		attributi
getClassSchemaByName	-className	

4.2.7. Funzioni

Funzione	Parametri	Descrizione
callFunction	-functionName -code -name -value	Esegue la funzione specificata nei parametri
getFunctionList		Fornisce una lista di tutte le funzioni disponibili

4.2.8. Attachments

Struttura dati attachments:

- category
- description
- filename
- version
- author
- created
- modified

Funzione	Parametri	Descrizione
copyAttachment	-sourceClassName -sourceId -destinationClassName -destinationId	Copia un attachment da una classe ad un’altra
updateAttachmentDescription	-className -filename -description	Modifica la descrizione di un attachment specifico
downloadAttachment	-className -objectId -filename	Scarica un attachment specifico

4.2.9. Reports

Funzione	Parametri	Descrizione
getBuiltInReport	-id -extension -params	Fornisce le informazioni di un report specifico
getReport	-id	Fornisce le informazioni di un report specifico

	-extension -params	
getReportList	-type -limit -offset	Fornisce una lista di tutti i report disponibili del tipo specificato
getReportParameters	-id -extension	Fornisce una lista di parametri per uno specifico report

4.2.10. Altre funzioni

Funzione	Parametri	Descrizione
abortWorkflow	-card	Termina un workflow specifico
generateDigest	-plainText -digestAlgorithm	Genera un digest con l'algoritmo di digest specificato
getActivityMenuSchema		Fornisce lo schema dell'activity menu
getActivityObjects	-className -cardId	Fornisce una lista di activity objects per una specifica card
getMenuSchema		
getProcessHelp	-className -cardId	
getReference	-className -query -orderType -limit -offset -fullTextQuery -cqlQuery	
getUserInfo		Fornisce varie informazioni riguardo all'utente autenticato
notify		
resumeWorkflow	-card	Riprende l'esecuzione di un workflow
suspendWorkflow	-card	Sospende l'esecuzione di un workflow
updateWorkflow	-card	Aggiorna un workflow specifico
sync	-xml	

5. Web service REST

5.1. Web service REST in CMDBuild

A differenza di SOAP, per i webservice REST non troviamo un unico endpoint al quale effettuare tutte le richieste, ma ve ne troviamo uno per ogni categoria.

Per esempio, se vogliamo operare su una card, l'endpoint relativo sarà:

http://hostname:port/cmdbuild/services/rest/v3/cards

5.2. Endpoint REST

In questo paragrafo verrà presentata una lista di tutti gli endpoint e la struttura dati degli elementi utilizzati.

Da notare che in alcuni casi degli endpoint possono contenere delle minime variazioni del percorso, ad esempio alcuni endpoint funzionano ugualmente sia per le cards che per le istanze di processo, quando entrambe le voci possono essere utilizzate verranno indicate con un simbolo "|" nel mezzo.

Quando un endpoint richieda l'aggiunta di informazioni nel percorso, ciò verrà indicato nella colonna path.

Quando un endpoint richiede dei query parameters aggiuntivi, questi verranno indicati nella colonna Parametri.

Una struttura dati viene presentata ogni qualvolta un metodo POST accetti un oggetto custom in formato json.

La maggiorparte degli endpoints forniranno una funzione per leggere l'intera lista degli oggetti che ci forniscono, una funzione per ottenere le informazioni di un oggetto specifico, una funzione per creare un nuovo oggetto, una funzione per aggiornare un oggetto esistente ed una funzione per eliminare un oggetto esistente. Ovviamente con l'aggiunta di funzioni più specifiche in alcuni casi.

Nel path di diversi endpoint, quando vengono utilizzati campi noti come "id", "classId", "cardId" o simili, questi dovranno essere sostituito con l'id (o il code nel caso delle classi), dell'oggetto di interesse.

5.2.1. Operazioni asincrone

http://hostname:port/cmdbuild/services/rest/v3/async

Funzione	Percorso	Parametri	Tipo	Descrizione
getAsyncJobStatus	/jobs/jobId		GET	Fornisce lo stato di un job asincrono
getAsyncJobResult	/jobs/jobId/response		GET	Fornisce i risultati di un job asincrono

5.2.2. Allegati

http://hostname:port/cmdbuild/services/rest/v3/processes|classes/classId/instances|cards/attachments

Struttura dati attachment:

- name String
- category String
- description String
- majorVersion boolean

Funzione	Percorso	Parametri	Tipo	Descrizione
read			GET	Fornisce la lista di attachments associati ad una card specifica
read	/attachmentId		GET	Fornisce le informaizoni relative ad un attachment specifico
download	/attachmentId/file:		GET	Scarica un attachment specifico
preview	/attachmentId/preview		GET	Fornisce una preview di un attachment specifico
getAttachment History	/attachmentId/history		GET	Fornisce lo storico di un attachment
downloadPreviousVersion	/attachmentId/history/version/file:		GET	Scarica una vecchia versione di un attachment
create	-attachment multipart json -file multipart dataHandler -copyFrom_class String -copyFrom_card Long -copyFrom_id String	-attachment multipart json -file multipart dataHandler -copyFrom_class String -copyFrom_card Long -copyFrom_id String	POST	Crea un nuovo attachment
update	/attachmentId	-attachment multipart json	PUT	Aggiorna un attachment esistente
delete	/attachmentId		DELETE	Elimina un attachment esistente

5.2.3. Audits

http://hostname:port/cmdbuild/services/rest/v3/system/audit

Funzione	Percorso	Parametri	Tipo	Descrizione
mark	/mark		GET	Fornisce la data corrente
getRequests	/requests	-since String -limit Long	GET	Fornisce una lista di tutte le request dopo una specifica data
getErrors	/errors	-since String -limit Long	GET	Fornisce una lista di tutte le richieste in errore dopo una specifica data
getRequests	/requests/id		GET	Fornisce le informazioni dettagliate di una richiesta specifica

5.2.4. Progetti bim

http://hostname:port/cmdbuild/services/rest/v3/bim/projects

Struttura dati Bim project:

- name String
- description String
- importMapping String
- projectId String
- parentId Long
- ownerClass String
- ownerCard String
- active boolean

Funzione	Percorso	Parametri	Tipo	Descrizione
getAll			GET	Fornisce una lista di tutti i Bim project disponibili
getOne	/id		GET	Fornisce le informazioni di uno specifico Bim project
createProjectWithFile		-data json -file multipart dataHandler -ifcFormat String	POST	Crea un nuovo Bim project
update	/id/file	-data json	PUT	Aggiorna un Bim project già esistente

downloadIfcFile	/id/file	-ifcFormat String	GET	Scarica il file ifc di un Bim project specifico
uploadIfcFile	/id/file	-file dataHandler -ifcFormat String	POST	Carica un nuovo file ifc per un Bim project specifico
delete	/id		DELETE	Elimina un bim project specifico

5.2.5. Valori bim

http://hostname:port/cmdbuild/services/rest/v3/bim/values

Funzione	Percorso	Parametri	Tipo	Descrizione
getOne	/globalId	-if_exists Boolean	GET	Fornisce i Bim values per un Bim project specifico

5.2.6. Avvio

http://hostname:port/cmdbuild/services/rest/v3/boot

Funzione	Percorso	Parametri	Tipo	Descrizione
status	/status		GET	Fornisce lo stato corrente del sistema
checkDatabaseConfig	/database/check	-dbConfig Map<String,String>	POST	
reconfigureDatabase	/database/configure	-file multipart dataHandler -dbConfig Map<String,String>	POST	
getPendingPatches	/patches		GET	Fornisce una lista di tutte le patch attualmente in attasa
applyPendingPatches	/patches/apply		POST	Comunica al sistema di applicare tutte le patch in attesa

5.2.7. Email eventi del calendario

http://hostname:port/cmdbuild/services/rest/v3/calendar/events/eventId/emails

Function	Path	Parameters	Type	Description
readAll		-limit Integer -start	GET	Fornisce una lista di email associate ad un evento specifico

		Integer -detailed Boolean		
read	/emailId		GET	Fornisce le informazioni di una email specifica per un evento specifico

5.2.8. Eventi del calendario

http://hostname:port/cmdbuild/services/rest/v3/calendar/events

Struttura dati evento:

- category String
- priority String
- job Long
- card Long
- sequence Long
- content String
- description String
- timeZone String
- eventEditMode String
- notifications List<String>
- partecipants List<String>
- onCardDeleteAction String
- type String
- begin String
- end String
- owner String
- status String
- source String
- notes String

Function	Path	Parameters	Type	Description
readMany		-filter String -sort String -limit Long -start Long	GET	Fornisce una lista di tutti gli eventi

		-detailed Boolean -positionOf Long -goToPage Boolean		
readOne	/eventId		GET	Fornisce le informazioni di un evento specifico
createUserEvent		-data json	POST	Crea un nuovo evento con i dati forniti
update	/eventId	-data json	PUT	Aggiorna un evento con i dati forniti
delete	/eventId		DELETE	Rimuove un evento
readHistory	/eventId/history	-limit Long -start Long -detailed Boolean	GET	Fornisce la history di un evento specifico
getHistoryRecord	/eventId/history/recordId		GET	Fornisce le informazioni di una versione datata di un evento specifico

5.2.9. Sequenze del calendario

http://hostname:port/cmdbuild/services/rest/v3/calendar/sequences

Struttura dati sequenza:

- category String
- priority String
- job Long
- card Long
- content String
- description String
- timeZone String
- title String
- eventCount Integer
- frequencyMultiplier Integer
- maxActiveEvents Integer
- eventEditMode String
- notifications List<String>
- partecipants List<String>

- onCardDeleteAction String
- sequenceParamsEditMode String
- showGeneratedEventsPreview Boolean
- eventType String
- firstEvent String
- lastEvent String
- trigger Long
- endType String
- events List<WsEventData>

Function	Path	Parameters	Type	Description
readOne	/sequenceId	-includeEvents Boolean	GET	Fornisce le informazioni di una sequenza specifica con la possibile aggiunta degli eventi correlati
readManyByCard	/by-card/cardId	-detailed Boolean -includeEvents Boolean	GET	Fornisce la lista di sequenze correlate ad una card
create		-data json	POST	Crea una nuova sequence con i dati forniti
update	/sequenceId	-data json	PUT	Aggiorna una sequenza esistente con i dati forniti
delete	/sequenceId		DELETE	Rimuove una sequenza
getEventsPreview	/_ANY/generate-events	-data json	POST	Genera degli eventi sulla base dei dati forniti

5.2.10. Trigger del calendario

http://hostname:port/cmdbuild/services/rest/v3/calendar/triggers

Struttura dati trigger:

- category String
- priority String
- job Long
- conditionScript String
- content String
- description String
- timeZone String
- eventCount Integer
- frequencyMultiplier Integer

- maxActiveEvents Integer
- delay String
- eventEditMode String
- eventTime String
- frequency String
- notifications List<String>
- partecipants List<String>
- onCardDeleteAction String
- sequenceParamsEditMode String
- showGeneratedEventsPreview Boolean
- active Boolean
- eventType String
- ownerClass String
- ownerAttr String
- endType String

Function	Path	Parameters	Type	Description
readOne	/triggerId		GET	Fornisce le informazioni di un trigger specifico
getSequencePreview	/triggerId/generate-sqeuence	-dateValue String	GET	Genera delle sequenze basate sul trigger
readMany			GET	Fornisce la lista completa dei trigger esistenti
create		-data json	POST	Crea un nuovo trigger con i dati forniti
update	/triggerId	-data json	PUT	Aggiorna un trigger esistente con i dati forniti
delete	/triggerId		DELETE	Elimina un trigger

5.2.11. Viste su eventi calendario

http://hostname:port/cmdbuild/services/rest/v3/calendar/views/viewId/events

Function	Path	Parameters	Type	Description
readOne	/eventId		GET	Fornisce le informazioni di un evento di una vista
readMany		-limit Long -start Long -detailed Boolean	GET	Fornisce gli eventi di una specifica vista

5.2.12. Valori Bim per card

http://hostname:port/cmdbuild/services/rest/v3/processes|classes/classId/cards|instances/cardId/bimvalue

Funzione	Percorso	Parametri	Tipo	Descrizione
getAllForCard		-if_exists Boolean -include_related Boolean	GET	Fornisce una lista di tutti i valori Bim per una card specifica

5.2.13. Allegati email

http://hostname:port/cmdbuild/services/rest/v3/processes|classes/classId/cards|instances/cardId/emails/emailId/attachments

Struttura dati attachment:

- file String
- category String
- description String
- majorVersion boolean

Funzione	Percorso	Parametri	Tipo	Descrizione
create		-attachment multipart json -file multipart dataHandler -copyFrom_class String -copyFrom_card Long -copyFrom_id String	POST	Crea un nuovo attachment per un'email specifica
read			GET	Fornisce una lista di allegati di una email di una specifica card
read	/attachmentId		GET	Fornisce le informazioni specifice di un allegato di una email
download	/attachmentId/file		GET	Scarica uno specifico attachment
preview	/attachmentId/preview		GET	Fornisce una preview di uno specifico attachment

update	/attachmentId	-attachment multipart json -file multipart dataHandler	PUT	Aggiorna un attachment esistente con le informaizoni fornite nei parametri
delete	/attachmentId		DELETE	Elimina un attachment specifico
getAttachmentHistory	/attachmentId/history		GET	Fornisce lo storico di un attachment specifico
downloadPreviousVersion	/attachmentId/history/version/file:		GET	Scarica una versione precedente di un attachment specifico

5.2.14. Email

http://hostname:port/cmdbuild/services/rest/v3/processes|classes/classId/cards|instances/cardId/emails

Struttura dati email:

- delay Long
- from String
- replyTo String
- to String
- cc String
- bcc String
- subject String
- body String
- contentType String
- account Long
- template Long
- autoReplyTemplate Long
- keepSynchronization boolean
- noSubjectPrefix boolean
- promptSynchronization boolean
- status String
- _expr String

Funzione	Percorso	Parametri	Tipo	Descrizione
readAll		-limit Integer -start Integer -detailed Boolean	GET	Fornisce una lista di tutte le email attualmente disponibili
read	/emailId		GET	Fornisce le informaizoni di un'email specifica
create		-data List<json> -apply_template Boolean -template_only Boolean	POST	Crea una nuova email con le informazioni fornite in emailData o con il template fornito
update	/emailId	-data json -apply_template Boolean -template_Only Boolean	PUT	Aggiorna una email esistente con le informazioni fornite in emailData
delete	/emailId		DELETE	Elimina una email specifica

5.2.15. Geo values

http://hostname:port/cmdbuild/services/rest/v3/processes|classes/classId/cards|instances/cardId/geovalues

Struttura dati geo values:

- _type String
- x Double
- y Double
- points List<WsPoint>

Point data structure:

- x Double
- y Double

Funzione	Percorso	Parametri	Tipo	Descrizione
getAllForCard			GET	Fornisce una lista di tutti i geovalue
get	/attributeId		GET	Fornisce le informazioni relative ad un geovalue
set	/attributeId	-data json	PUT	Aggiorna un geovalue esistente con data

delete	/attributeId		DELETE	Elimina un geovalue

5.2.16. Storico card

http://hostname:port/cmdbuild/services/rest/v3/processes|classes/classId/cards|instances/cardId/history

Funzione	Percorso	Parametri	Tipo	Descrizione
getHistory		-limit Long -start Long	GET	Fornisce lo storico di una specifica card
getHistoryRecord	/recordId		GET	Fornisce le informazioni di una vecchia versione di una card

5.2.17. Locks

http://hostname:port/cmdbuild/services/rest/v3/processes|classes/classId/cards|instances/cardId/lock

Funzione	Percorso	Parametri	Tipo	Descrizione
getLock			GET	Fornisce la lista di lock associati ad una card
createLock			POST	Crea un nuovo lock per una card specifica
releaseLock			DELETE	Elimina un lock esistente per una card specifica

5.2.18. Card print

http://hostname:port/cmdbuild/services/rest/v3/classes/classId/cards/cardId/print/file:

Funzione	Percorso	Parametri	Tipo	Descrizione
readOne		-extension String	GET	Scarica un file di estensione a scelta con un report di una specifica card con id cardId appartenente ad una classe con id classId

5.2.19. Relazioni

http://hostname:port/cmdbuild/services/rest/v3/processes|classes/classId/cards|instances/cardId/relations

Struttura dati relations:

- _id Long
- _type String
- _user String

- _sourceType String
- _sourceId Long
- _destinationType String
- _destinationId Long
- _is_direct boolean

Funzione	Percorso	Parametri	Tipo	Descrizione
read		-limit Long -start Long -detailed Boolean	GET	Fornisce una lista di relazioni per una card specifica
create		-data json	POST	Crea una nuova relazione con le informazioni fornite nel parametro relationData
update	/relationId	-data json	PUT	Aggiorna una relazione esistente
delete	/relationId		DELETE	Elimina una relazione specifica

5.2.20. Cards

http://hostname:port/cmdbuild/services/rest/v3/classes/classId/cards

Struttura dati cards:

La struttura dati di una card varia in base a come è stata configurata

Funzione	Percorso	Parametri	Tipo	Descrizione
readOne	/cardId	-includeModel Boolean -includeWidgets Boolean	GET	Fornisce le informazioni relative ad una card specifica
readMany		-filter String -sort String -limit Long -start Long -positionOfCard -goToPage Boolean	GET	Fornisce una lista di tutte le card disponibili per una specifica classe
create		-data json	POST	Crea una nuova card con le informazioni fornite nel parametro data

update	/cardId	-data json	PUT	Aggiorna una card esistente
delete	/cardId		DELETE	Elimina una card specifica

5.2.21. Attributi di classe

http://hostname:port/cmdbuild/services/rest/v3/processes|classes/classId/attributes

Struttura dati attributi:

- formatPattern String
- unitOfMeasure String
- unitOfMeasureLocation String
- visibleDecimals Integer
- preselectIfUnique boolean
- showThousandsSeparator boolean
- showSeconds boolean
- showSeparators boolean
- type String
- name String
- description String
- showInGrid boolean
- showInReducedGrid boolean
- domainKey String
- domain String
- direction String
- unique boolean
- mandatory boolean
- active boolean
- index Integer
- defaultValue String
- group String
- precision String
- scale String
- targetClass String
- maxLenght Integer
- editorType String
- lookupType String
- filter String
- help String

- showIf String
- validationRules String
- mode boolean
- autoValue String
- metadata map<String, String>
- classOrder Integer
- isMasterDetail Boolean
- masterDetailDescription String
- ipType String
- textContentSecurity String

Funzione	Percorso	Parametri	Tipo	Descrizione
read	/attrId		GET	Fornisce le informazioni di un attributo specifico
readAll		-limit Long -start Long	GET	Fornisce una lista di tutti gli attributi relativi ad una specifica classe
create		-data json	POST	Crea un nuovo attributo per una classe specifica con le informazioni fornite nel parametro data
update	/attrId	-data json	PUT	Aggiorna un attributo con le informazioni specificate nel parametro data
delete	/attrId		DELETE	Elimina un attributo esistente
reorder	/order	-attrOrder List<String>	POST	Riordina la lista di attributi di una classe secondo l'ordine specificato nel parametro attrOrder

5.2.22. Filtri di classe

http://hostname:port/cmdbuild/services/rest/v3/processes|classes/classId/filters

Struttura dati dei filtri:

- name String
- description String
- target String
- configuration String
- active Boolean
- shared Boolean

Funzione	Percorso	Parametri	Tipo	Descrizione
readAll		-limit Long -start Long -shared Boolean	GET	Fornisce una lista di filtri disponibili per una specifica classe
read	/filterId		GET	Fornisce le informazioni di uno specifico filtro
create		-data json	POST	Crea un nuovo filtro con le informazioni fornite nel parametro element
update	/filterId	-data json	PUT	Aggiorna un filtro esistente con le informazioni fornite nel parametro element
delete	/filterId		DELETE	Elimina un filtro specifico
getDefaultForRoles	/filterId/defaultFor		GET	Obtain a list of roles for a specific filter
updateWithPost	/filterId/defaultFor	-roles List<json>	POST	Aggiorna la lista dei ruoli per un filtro specifico con le informazioni fornite nel parametro roles

5.2.23. Report di classe

http://hostname:port/cmdbuild/services/rest/v3/classes

Funzione	Percorso	Parametri	Tipo	Descrizione
printClassReport		-detailed Boolean	GET	Fornisce i risultati di un report specifico per una classe o per un processo

5.2.24. Class print

http://hostname:port/cmdbuild/services/rest/v3/classes

Function	Path	Parameters	Type	Description
printClassReport	/classId/print/file	-filter String -sort String -limit Long -start Long -extension	GET	Fornisce il risultato di un report con l'estensione richiesta

		String -attributes String		
printClassSchemaReport	/classId/ print_schema/file	-file String -extenstion String	GET	Fornisce un file contenente lo schema della classe specificata
printSchemaReport	/print_schema/file	-file String -extension String	GET	Fornisce un file con lo schema dell'intero database

5.2.25. Classi

http://hostname:port/cmdbuild/services/rest/v3/classes

Struttura dati classe:

- name String
- description String
- defaultFilter Long
- defaultImportTemplate Long
- defaultExportTemplate Long
- _icon Long
- validationRule String
- type String
- messageAttr String
- flowStatusAttr String
- engine String
- parent String
- active Boolean
- prototype Boolean
- noteInline Boolean
- noteInlineClosed Boolean
- attachmentsInLine Boolean
- enableSaveButton Boolean
- attachmentTypeLookup String
- attachmentDescriptionMode String
- multitenantMode String
- stoppableByUser Boolean
- defaultOrder List<>

- formTriggers List<>
- contextMenuItems List<>
- widgets List<>
- attributeGroups List<>
- domainOrder List<String>
- formStructure JsonNode

Funzione	Percorso	Parametri	Tipo	Descrizione
readAll		-detailed Boolean -limit Long -start Long -filter String	GET	Fornisce una lista di tutte le classi disponibili
read	/classId		GET	Fornisce le informazioni di una classe specifica
create		-data json	POST	Crea una nuova classe con le informazioni fornite nel parametro data
update	/classId	-data json	PUT	Aggiorna una classe esistente con le informazioni fornite nel parametro data
delete	/classId		DELETE	Elimina una classe esistente

5.2.26. Configurazioni

http://hostname:port/cmdbuild/services/rest/v3/configuration

Funzione	Percorso	Parametri	Tipo	Descrizione
getPublicConfig	/public		GET	Fornisce la configurazione pubblica
getSystemConfig	/system		GET	Fornisce la configurazione di sistema

5.2.27. Componenti menu custom

http://hostname:port/cmdbuild/services/rest/v3/components/contextmenu

Struttura dati componenti custom:

- active boolean
- description String

Function	Path	Parameters	Type	Description
list			GET	Fornisce una lista di tutti i componenti custom
get	/id		GET	Fornisce le informazioni relative ad un componente custom specifico
delete	/id		DELETE	Elimina un componente custom specifico
create	/id/file	-extension String -parameters String	POST	Crea un nuovo componente custom con le informazioni fornite nei parametri
update		-file dataHandler -data json -merge Boolean	PUT	Aggorna un componente custom specifico con le informazioni fornite nei parametri
update	/id	-file dataHandler	PUT	Aggorna un componente custom specifico con le informazioni fornite nei parametri

5.2.28. Pagine custom

http://hostname:port/cmdbuild/services/rest/v3/custompages

Struttura dati pagine custom:

- description String
- active boolean

Funzione	Percorso	Parametri	Tipo	Descrizione
list			GET	Fornisce una lista di tutte le pagina custom
get	/id		GET	Fornisce le informazioni relative ad una pagina custom specifica
delete	/id		DELETE	Elimina una pagina custom esistente
create		-file dataHandler -data json -merge Boolean	POST	Crea una nuova pagina custom con le informazioni fornite nei parametri

update	/id	-file dataHandler	PUT	Aggorna una pagina custom specifica con le informazioni fornite nei parametri
update	/id	-data json	PUT	Aggorna una pagina custom specifica con le informazioni fornite nei parametri
download	/id/file	-extension String -parameters String		

5.2.29. Dashboard

http://hostname:port/cmdbuild/services/rest/v3/dashboards

Struttura dati dashboard:

- name String
- description String
- active boolean
- charts Object
- layout Object

Funzione	Percorso	Parametri	Tipo	Descrizione
getAll		-detailed Boolean -limit Integer -start Integer	GET	Fornisce la lista di tutte le dashboard
readOne	/id		GET	Fornisce una dashboard
create		-data json	POST	Crea una nuova dashboard con i dati forniti
update	/id	-data json	PUT	Aggiorna una dashboard esistente
delete	/id		DELETE	Cancella una dashboard

5.2.30. Attributi di dominio

http://hostname:port/cmdbuild/services/rest/v3/domains/domainId/attributes

Struttura dati attributi di dominio:

- formatPattern String
- unitOfMeasure String
- unitOfMeasureLocation String
- visibleDecimals Integer
- preselectIfUnique boolean

- showThousandsSeparator boolean
- showSeconds boolean
- showSeparators boolean
- type String
- name String
- description String
- showInGrid boolean
- showInReducedGrid boolean
- domainKey String
- domain String
- direction String
- unique boolean
- mandatory boolean
- active boolean
- index Integer
- defaultValue String
- group String
- precision String
- scale String
- targetClass String
- maxLenght Integer
- editorType String
- lookupType String
- filter String
- help String
- showIf String
- validationRules String
- mode boolean
- autoValue String
- metadata map<String, String>
- classOrder Integer
- isMasterDetail Boolean
- masterDetailDescription String
- ipType String
- textContentSecurity String

Funzione	Percorso	Parametri	Tipo	Descrizione
readAll		-limit Integer -start Integer	GET	Fornisce la lista di attributi di un dominio specifico
read	/attrId		GET	Fornisce le informazioni di un attributo specifico
create		-data json	POST	Crea un nuovo attributo con le informazioni fornite nel parametro data
update	/attrId	-data json	PUT	Aggiorna un attributo esistente con le informazioni fornite nel parametro data
delete	/attrId		DELETE	Elimina un dominio specifico
reorder	/order	-attrOrder List <String>	POST	Riordina la lista degli attributi con l'ordine fornito nel parametro attrOrder

5.2.31. Domini

http://hostname:port/cmdbuild/services/rest/v3/domains

Struttura dati dominio:

- source String
- name String
- description String
- destination String
- cardinality String
- descriptionDirect String
- descriptionInverse String
- indexDirect int
- indexInverse int
- descripttionMasterDetail String
- filterMasterDetail String
- isMasterDetail boolean
- active boolean
- disabledSourceDescendants List<String>
- disabledDestinationDescendants List<String>
- inline Boolean
- defaultClosed Boolean

Funzione	Percorso	Parametri	Tipo	Descrizione
readAll		-filter String -limit Integer -start Integer	GET	Fornisce una lista di tutti i domini disponibili
read	/domainId		GET	Fornisce le informazioni di un dominio specifico
create		-data json	POST	Crea un nuovo dominio con le informazioni fornite nel parametro data
update	/domainId	-data json	PUT	Aggiorna un dominio esistente con le informazioni fornite nel parametro data
delete	/domainId		DELETE	Elimina un dominio esistente

5.2.32. Account email

http://hostname:port/cmdbuild/services/rest/v3/email/accounts

Struttura dati account email:

- name String
- username String
- password String
- address String
- smtp_server String
- smtp_port Integer
- smtp_ssl boolean
- smtp_starttls boolean
- imap_output_folder String
- imap_server String
- imap_port Integer
- imap_ssl boolean
- imap_starttls boolean

Funzione	Percorso	Parametri	Tipo	Descrizione
readAll		-limit Long -offset Long -detailed Boolean	GET	Fornisce una lista di tutti gli account email disponibili

read	/accountId		GET	Fornisce le informazioni relative ad uno specifico account email
create		-data json	POST	Crea un nuovo account email con le informazioni fornite nel parametro data
update	/accountId	-data json	PUT	Aggiorna un account email con le informazioni fornite nel parametro data
delete	/accountId			Elimina un account email esistente

5.2.33. Coda email

http://hostname:port/cmdbuild/services/rest/v3/email/queue

Funzione	Percorso	Parametri	Tipo	Descrizione
triggerQueue	/trigger		POST	Innesca la coda email, inviando le email con stato outgoing
getOutgoingEmail	/outgoing		GET	Fornisce una lista di tutte le email con stato outgoing
sendSingleEmail	/outgoing/emailId/trigger		POST	Innesca l'invio di una email specifica

5.2.34. Templates email

http://hostname:port/cmdbuild/services/rest/v3/email/templates

Struttura dati template email:

- name String
- description String
- from String
- to String
- cc String
- bcc String
- subject String
- body String
- contentType String
- account String
- keepSynchronization boolean
- promptSynchronization boolean
- delay Long
- data Map<String, String>

Funzione	Percorso	Parametri	Tipo	Descrizione
readAll		-filter String -sort String -limit Long -start Long -detailed Boolean -includeBindings Boolean	GET	Fornisce una lista di tutti i template email disponibili
read	/templateId		GET	Fornisce le informazioni di un template email specifico
create		-data json	POST	Crea un nuovo template email con le informazioni fornite nel parametro data
update	/templateId	-data json	PUT	Aggiorna un template email con le informazioni fornite nel parametro data
delete	/templateId		DELETE	Elimina un template email esistente

5.2.35. Etl Gate

http://hostname:port/cmdbuild/services/rest/v3/etl/gates

Struttura dati Etl Gate:

- code String
- processingMode String
- allowPublicAccess Boolean
- enabled Boolean
- config Map<String, String>
- template String
- script String

Function	Path	Parameters	Type	Description
readAll		-limit Long -offset Long -detailed boolean	GET	Fornisce una lista dei gate Etl
read	/gateId		GET	Fornisce le informazioni di un gate etl specifico

create		-date json	POST	Crea un nuovo gate etl con le informazioni fornite
update	/gateId	-data json	PUT	Aggiorna un gate etl con le informazioni fornite
delete	/gateId		DELETE	Rimuove un gate Etl

5.2.36. Fk Domini

http://hostname:port/cmdbuild/services/rest/v3/fkdomains

Funzione	Percorso	Parametri	Tipo	Descrizione
readAll		-filter String -limit Long -start Long	GET	Fornisce una lista di tutte le FK per ogni dominio, con la possibilità di filtrare i risultati

5.2.37. Funzioni

http://hostname:port/cmdbuild/services/rest/v3/functions

Funzione	Percorso	Parametri	Tipo	Descrizione
readAll		-limit Integer -start integer -filter String -detailed Boolean	GET	Fornisce una lista di tutte le funzioni disponibili
read	/functionId		GET	Fornisce le informazioni di una funzione specifica
readInputParameters	/functionId/parameters	-limit Integer -start Integer	GET	Fornisce una lista di parametri di input per una funzione specifica
readOutputParameters	/functionId/attributes	-limit Integer -start Integer	GET	Fornisce una lista di parametri di output per una funzione specifica
call	/functionId/outputs	-parameters String -model String	GET	Esegue una funzione specifica

5.2.38. Geo attributi

http://hostname:port/cmdbuild/services/rest/v3/processes|classes/classId/geoattributes

Struttura dati geo attributi:

- name String
- subtype String
- description String
- index int
- visibility List
- zoomMin int
- zoomMax int
- zoomDef int
- style Map<String, Object>
- _icon Long

Funzione	Percorso	Parametri	Tipo	Descrizione
readAllAttributes		-limit Integer -start Integer -detailed Boolean -visible Boolean	GET	Fornisce una lista di tutti i geo attributi disponibili
reorder	/order	-attrOrder List<Long>	POST	Riordina la lista di geo attributi secondo l'ordine fornito nel parametro attrOrder
readAttribute	/attributeId		GET	Fornisce le informazioni relative ad un attributo specifico
create		-data json	POST	Crea un nuovo geo attributo con le informazioni fornite
updateVisibility	/visibility	-geoAttributes List<Long>	POST	Aggiorna la visibilità di un geo attributo
update	/attributeId	-data json	PUT	Aggiorna un geo attributo esistente con le informazioni fornite nei parametri
delete	/attributeId		DELETE	Elimina un geo attributo esistente

5.2.39. Geo style rules

http://hostname:port/cmdbuild/services/rest/v3/processes|classes/classId/geostylerules

Struttura dati geo style rules:

- name String
- description String
- owner String
- attribute String
- classattribute String
- function String
- alanalysistype String
- segments Integer
- rules JsonNode

Funzione	Percorso	Parametri	Tipo	Descrizione
readAll		-limit Integer -start Integer	GET	Fornisce una lista di geo style rules associati ad una classe
read	/rulesetId		GET	Fornisce le informazioni di un geo style rule specifico
create		-data json	POST	Crea un nuovo geo style rule con le informazioni fornite in data
update	/rulesetId	-data json	PUT	Aggiorna un geo style rule specifico con le informazioni fornite in data
delete	/rulesetId		DELETE	Elimina un geo style rule specifico
applyRules	/rulesetId/result	-cards String	GET	Fornisce i risultati di un geo style rule specifico
applyRules	/tryRules	-data json -cards String	POST	Fornisce i risultati di un geo style rule definito in data

5.2.40. Geo values

http://hostname:port/cmdbuild/services/rest/v3/processes|classes/_ANY/cards|instances/_ANY/geovalues

Funzione	Percorso	Parametri	Tipo	Descrizione
query		-attrs Set<Long> -area String -filter String -attach_nav_tree Boolean	GET	Fornisce una lista di tutti i geo values disponibili
queryArea	/area	-attribute Set<Long> -filter String	GET	
queryCenter	/center	-attribute Set<Long> -filter String	GET	

5.2.41. Geo server layers

http://hostname:port/cmdbuild/services/rest/v3/processes|classes/classId/cards|instances/cardId/geolayers

Struttura dati geo server layers:

- name String
- type String
- active Boolean
- index Integer
- geoserver_name String
- description String
- zoomMin int
- zoomMax int
- zoomDef int
- visibility List<String>

Funzione	Percorso	Parametri	Tipo	Descrizione
getOneForCard	/layerId		GET	Fornisce le informazioni di un layer specifico di una card

getAllForCard		-visible Boolean	GET	Fornisce una lista di tutti i geo server layers disponibili
create		-data json -file DataHandler	POST	Crea un nuovo geo server layer con le informazioni fornite nel parametro attributeData
update	/layerId	-data json -file DataHandler	PUT	Aggiorna un geo server layer con le informazioni fornite nel parametro attributeData
delete	/layerId		DELETE	Elimina un geo server later esistente
order	/order	-layerOrder List<Long>	POST	Riordina la lista di geo server layers con l'ordine fornito nel parametro layerOrder

5.2.42. Permessi

http://hostname:port/cmdbuild/services/rest/v3/roles/roleId/grants

Struttura dati permessi:

- mode String
- objectType String
- objectTypeName String
- filter String
- attributePrivileges Map<String, String>
- _card_create_disabled Boolean
- _card_update_disabled Boolean
- _card_delete_disabled Boolean
- _card_clone_disabled Boolean
- _card_relation_disabled Boolean
- _card_print_disabled Boolean
- _can_fc_attachment Boolean
- _on_filter_mismatch Boolean

Funzione	Percorso	Parametri	Tipo	Descrizione
readMany		-filter String -limit Long -start Long -include	GET	Fornisce una lista di permessi disponibili

		ObjectDescription Boolean -include RecordsWithoutGrant Boolean		
readOneByObject	/by-target/ objectType/ objectTypeName		GET	
update	/_ANY	-data List<json>	PUT	Aggiorna un permesso specifico con le informazioni fonite nel parametro data

5.2.43. Impersonificazione

http://hostname:port/cmdbuild/services/rest/v3/sessions/current/impersonate

Funzione	Percorso	Parametri	Tipo	Descrizione
impersonate	/username		POST	Inpersonifica uno user specifico con l'username fornito
deimpersonare			DELETE	Termina l'impersonificazione corrente

5.2.44. Jobs

http://hostname:port/cmdbuild/services/rest/v3/jobs

Struttura dati job:

- code String
- description String
- type String
- config Map<String, Object>
- enabled boolean

Funzione	Percorso	Parametri	Tipo	Descrizione
readMany		-limit Long -start Long	GET	Fornisce una lista di tutti i Job disponibili
readOne	/jobId		GET	Fornisce le informazioni di un job specifico
create		-data json	POST	Crea un nuovo Job con le informazioni fornite in data

update	/jobId	-data json	PUT	Aggiorna un Job esistente con le informazioni fornite in data
delete	/jobId		DELETE	Elimina un Job esistente
runJobNow	/jobId/run		POST	Esegue un Job esistente
getJobRuns	/jobId/runs	-limit Long -start Long	GET	Fornisce una lista di tutte le esecuzioni di un Job specifico
getJonRunErrors	/jobId/errors	-limit Long -start Long	GET	Fornisce una lista di tutti gli errori causati da un Job specifico
getJobRuns	/_ANY/runs	-limit Long -start Long	GET	Fornisce una list di tutte le esecuzioni di tutti i Job
getJonRunErrors	/_ANY/errors	-limit Long -start Long	GET	Fornisce una list di tutti gli errori causati da tutti i Job
getJobRuns	/jobId/runs/runId		GET	Fornisce le informazioni relative ad una specifica esecuzione di uno specifico Job

5.2.45. Configurazioni di lingua

http://hostname:port/cmdbuild/services/rest/v3/configuration/languages

Funzione	Percorso	Parametri	Tipo	Descrizione
getLoginLanguages			GET	Fornisce una lista di tutte le lingue disponibili

5.2.46. Lingue

http://hostname:port/cmdbuild/services/rest/v3/languages

Funzione	Percorso	Parametri	Tipo	Descrizione
readLanguages		-active Boolean	GET	Fornisce una lista di tutte le lingue disponibili

5.2.47. Locks

http://hostname:port/cmdbuild/services/rest/v3/locks

Funzione	Percorso	Parametri	Tipo	Descrizione
getLocks			GET	Fornisce una lista di tutti i Lock disponibili
getLock	/lockId		GET	Fornisce le informazioni relative ad un lock specifico
deleteLock	/lockId		DELETE	Elimina un lock esistente
deleteAllLocks	/_ANY		DELETE	Elimina tutti i lock esistenti

5.2.48. Tipi lookup

http://hostname:port/cmdbuild/services/rest/v3/lookup_types

Struttura dati tipi lookup:

- name String
- parent String

Funzione	Percorso	Parametri	Tipo	Descrizione
read	/lookupTypeId		GET	Fornisce le informazioni di un tipo di Lookup
readAll		-limit Long -start Long -filter String	GET	Fornisce una lista di tutti i tipi di Lookup disponibili
createLookupType		-data json	POST	Crea un nuovo tipo di lookup con le informazioni fornite nei parametri
updateLookupType	/lookupTypeId	-data json	PUT	Aggiorna un tipo di lookup con le informazioni fornite nei parametri
deleteLookupType	/lookupTypeId		DELETE	Elimina un tipo di lookup esistente

5.2.49. Lookup values

http://hostname:port/cmdbuild/services/rest/v3/lookup_types/lookupTypeId/values

Struttura dati lookup:

- code String
- description String
- index Integer
- active boolean

- parent_id Long
- default boolean
- note String
- text_color String
- icon_type String
- icon_image String
- icon_font String
- icon_color String

Funzione	Percorso	Parametri	Tipo	Descrizione
read	/lookupValueId		GET	Fornisce una lista di lookup values per un tipo di lookup specifico
readAll		-limit Long -start Long -filter String -active Boolean	GET	Fornisce una lista di tutti i loopup values
create		-data json	POST	Crea un nuovo lookup value con le informazioni fornite nei parametri
update	/lookupValueId	-data json	PUT	Aggiorna un lookup value esistente con le informazioni fornite nei parametri
delete	/lookupValueId		DELETE	Elimina un lookup value esistente
reorder	/order	-lookupValueIds List<Long>	POST	Riordina la lista di lookup value con l'ordine specificato nei parametri

5.2.50. Menu

http://hostname:port/cmdbuild/services/rest/v3/menu

Struttura dati menu:

- _id String
- menuType MenuItemType
- objectTypeName String
- objectDescription String
- children List<MenuNodes>

Struttura dati nodo radice del menu:

- group String
- children List<MenuNodes>

Funzione	Percorso	Parametri	Tipo	Descrizione
readAll		-detailed Boolean	GET	Fornisce una lista di tutti i menu disponibili
read	/menuId		GET	Fornisce le informazioni di un menu specifico
create		-data json	POST	Crea un nuovo menu con le informazioni specificate nel parametro data
update	/menuId	-data json	PUT	Aggiorna un menu esistente con le informazioni specificate nel parametro data
delete	/menuId		DELETE	Elimina un menu esistente

5.2.51. Minions

http://hostname:port/cmdbuild/services/rest/v3/system_services

Funzione	Percorso	Parametri	Tipo	Descrizione
getServiceStatus			GET	Fornisce una lista contenente lo stato di tutti i servizi
getServiceStatus	/serviceId		GET	Fornisce lo stato di un servizio specifico
startStatus	/serviceId/start		POST	Avvia un servizio specifico
stopService	/serviceId/stop		POST	Ferma un servizio specifico

5.2.52. Alberi di navigazione

http://hostname:port/cmdbuild/services/rest/v3/domainTrees

Struttura dati albero:

- name String
- description String
- nodes List<TreeNodes>
- active boolean

Struttura dati nodo albero:

- filter String
- targetClass String
- domain String

- direction String
- recursionEnabled Boolean
- showOnlyOne Boolean
- nodes List<TreeNodes>

Funzione	Percorso	Parametri	Tipo	Descrizione
readAll		-filter String -limit Long -start Long	GET	Fornisce una lista di tutti gli alberi di navigazione disponibili
read	/treeId	-treeMode String	GET	Fornisce le informazioni di un albero di navigazione specifico
create		-data json	POST	Crea un nuovo albero di navigazione con le informazioni specificate nel parametro data
update	/treeId	-data json	PUT	Aggiorna un albero di navigazione esistente con le informazioni fornite nel parametro data
delete	/treeId		DELETE	Elimina un albero di navigazione specifico

5.2.53. Configurazione processi

http://hostname:port/cmdbuild/services/rest/v3/configuration/processes

Function	Path	Parameters	Type	Description
readStatuses	/statuses		GET	Fornisce una lista contenente gli stati dei processi

5.2.54. Email istanze di processo

http://hostname:port/cmdbuild/services/rest/v3/processes/processId/instances/instanceId/activities/activityId/emails

Function	Path	Parameters	Type	Description
updateEmailWithCardData	/sync	-flowData	POST	Aggiorna una specifica email

5.2.55. Istanze di attività di processo

http://hostname:port/cmdbuild/services/rest/v3/processes/processId/instances/processInstanceId/activities

Funzione	Percorso	Parametri	Tipo	Descrizione
read			GET	Fornisce una lista di tutti i

				task disponibili
read	/processActivityId		GET	Fornisce le informazioni di un task specifico

5.2.56. Storico istanze di processo

http://hostname:port/cmdbuild/services/rest/v3/processes/processId/instances/instanceId/history

Funzione	Percorso	Parametri	Tipo	Descrizione
getHistory		-limit Long -start Long	GET	Fornisce lo storico di un'istanza di processo specifica
getHistoryRecord	/recordId		GET	Fornisce le informazioni di una specifica versione di un'istanza di processo

5.2.57. Istanze di processo

http://hostname:port/cmdbuild/services/rest/v3/processes/processId/instances

Funzione	Percorso	Parametri	Tipo	Descrizione
create		-data json	POST	Crea una nuova istanza di processo con le informazioni fornite nei parametri
update	/processInstanceId	-data json	PUT	Aggiorna un'istanza di processo con le informazioni fornite nei parametri
read	/processInstanceId	-include_tasklist Boolean	GET	Fornisce le informazioni di un'istanza di processo specifica
plotGraph	/processInstanceId /graph	-simplified Boolean	GET	
readMany		-filter String -sort String -limit Long -start Long -positionOf Long -positionOf_goToPage Boolean -include_tasklist Boolean	GET	Fornisce una lista di tutte le istanze di processo disponibili
delete	/processInstanceId		DELETE	

5.2.58. Attività di inizio processo

http://hostname:port/cmdbuild/services/rest/v3/processes/processId/start_activities

Funzione	Percorso	Parametri	Tipo	Descrizione
read			GET	Fornisce le attività di inizio processo disponibili

5.2.59. Definizione task processi

http://hostname:port/cmdbuild/services/rest/v3/processes/processId/activities

Struttura dati task definition:

- formStructure JsonNode

Function	Path	Parameters	Type	Description
getAllActivities		-limit Long -start Long	GET	Fornisce tutte le task definition per un processo specifico
getOne	/taskId		GET	Fornisce le informazioni di uno specifico task
update	/taskId	-data json	PUT	Aggiorna un task esistente

5.2.60. Task processi

http://hostname:port/cmdbuild/services/rest/v3/processes/processId/instance_activities

Function	Path	Parameters	Type	Description
getAllActivities		-limit Long -start Long	GET	Fornisce tutti i task per un processo specifico

5.2.61. Processi

http://hostname:port/cmdbuild/services/rest/v3/processes

Funzione	Percorso	Parametri	Tipo	Descrizione
readAll		-activeOnly Boolean -limit Long -start Long -detailed Boolean	GET	Fornisce una lista di tutti i processi disponibili
read	/processId		GET	Fornisce le informazioni riguardo un processo specifico
create		-data json	POST	Crea un nuovo processo con le informazioni fornite

update	/processId	-data json	PUT	Aggiorna un processo esistente
delete	/processId		DELETE	Elimina un processo
uploadNewXpdlVersion	/processId/versions	-file DataHandler	POST	Carica un nuovo xpdl
uploadXpdlVersionAndMigrateProcessToNewProvider	/processId/migration	-provider String -file DataHandler	POST	Carica un nuovo xpdl e forza tale processo ad utilizzare la nuova versione
getAllXpdlVersions	/processId/versions		GET	Fornisce una lista di tutti gli xpdl
getXpdlVersionFile	/processId/versions/planId/file		GET	Fornisce un xpdl specifico
getXpdlTemplateFile	/processId/template		GET	Fornisce un template xpdl specifico

5.2.62. Relazioni

http://hostname:port/cmdbuild/services/rest/v3/domains/domainId/relations

Struttura dati relazioni:

- _id Long
- _type String
- _sourceType String
- _sourceId Long
- _destinationType String
- _destinationId Long
- _is_direct Boolean

Funzione	Percorso	Parametri	Tipo	Descrizione
readAll		-limit Long -start Long -detailed Boolean	GET	Fornisce una lista di tutte le relazioni di un dominio specifico
read	/relationId		GET	Fornisce le informazioni di una relazione specifica
create		-data json	POST	Crea una nuova relazione
update	/relationId	-data json	PUT	Aggiorna una relazione
delete	/relationId		DELETE	Elimina una relazione
moveManyRelations	/_ANY/move	-data json	POST	

copyManyRelations	/_ANY/copy	-data json	POST	Copy a specific relation

5.2.63. Reports

http://hostname:port/cmdbuild/services/rest/v3/reports

Struttura dati report:

- _id Long
- code String
- description String
- _description_translation String
- active boolean
- title String
- query String

Funzione	Percorso	Parametri	Tipo	Descrizione
readAll		-filter String -limit Long -start Long -detailed Boolean	GET	Fornisce una lista di tutti i report disponibili
read	/reportId		GET	Fornisce le informazioni di un report specifico
readAllAttributes	/reportId/attributes	-limit Long -start Long	GET	Fornisce una lista di tutti gli attributi di un report specifico
download	/reportId/file:	-extension String -parametersStr String	GET	Scarica un report in un'estensione specificata
createReport		-data json -attachments List<Attachment>	POST	Crea un nuovo report
updateReport	/reportId	-attachments List<Attachment>	PUT	Aggiorna un report esistente
updateReportTemplate	/reportId/template-data json -attachments List<Attachment>	-data json -attachments List<Attachment>	PUT	Aggiorna il template di un report esistente

downloadTemplateFiles	/reportId/template:		GET	Scarica il template di un report specifico
deleteReport	/reportId		DELETE	Elimina un report esistente

5.2.64. Resources

http://hostname:port/cmdbuild/services/rest/v3/resources

Function	Path	Parameters	Type	Description
downloadCompanyLogo	/company_logo/file	-roleId	GET	Fornisce il logo dela company

5.2.65. Filtri sui ruoli

http://hostname:port/cmdbuild/services/rest/v3/roles/roleId/filters

Function	Path	Parameters	Type	Description
read			GET	Fornisce la lista di filtri definiti su un ruolo
updateWithPost		-data json	POST	Aggiorna un filtro definito su un ruolo

5.2.66. Ruoli

http://hostname:port/cmdbuild/services/rest/v3/roles

Struttura dati ruoli:

- type String
- name String
- description String
- email String
- active boolean
- processWidgetAlwaysEnabled boolean
- startingClass String

Funzione	Percorso	Parametri	Tipo	Descrizione
readOne	/roleId		GET	Fornisce le informazioni di uno specifico ruolo
readMany		-limit Long -offset Long	GET	Fornisce una lista di tutti i ruoli disponibili

readRoleUser	/roleId/users	-filter String -sort String -limit Long -start Long -assigned Boolean	GET	Fornisce una lista di ruoli disponibili ad uno specifico user
updateUsers	/roleId/users	-users json	PUT	Aggiorna i ruoli di uno specifico user
create		-jsonData String	POST	Crea un nuovo ruolo con le informazioni fornite
update	/roleId	-jsonData String	PUT	Aggiorna le informazioni di un ruolo esistente

5.2.67. Menu di sessione

http://hostname:port/cmdbuild/services/rest/v3/sessions/sessionId/menu

Funzione	Percorso	Parametri	Tipo	Descrizione
read		-flat Boolean	GET	Fornisce una lista di tutti i menu di sessione disponibili

5.2.68. Preferenze di sessione

http://hostname:port/cmdbuild/services/rest/v3/sessions/sessionId/preferences

Funzione	Percorso	Parametri	Tipo	Descrizione
read			GET	Fornisce una lista di tutte le preferenze di una specifica sessione
getUserConfig	/key	-key String	GET	Fornisce le informazioni di una specifica configurazione utente
updateUserConfigValue	/key	-key String -value String	PUT	Aggiorna le informazioni di una specifica configurazione utente
updateUserConfigValues		-data Map<String, String>	POST	Aggiorna le informazioni di varie configurazioni utente
deleteSystemConfigValue	/key		DELETE	Elimina una specifica configurazione

5.2.69. Sessioni

http://hostname:port/cmdbuild/services/rest/v3/sessions

Struttura dati sessione:

- username String
- password String
- role String
- scope String
- tenant Long
- ignoreTenants Boolean
- activeTenants List<Long>

Funzione	Percorso	Parametri	Tipo	Descrizione
create		-data json - includeExtendedData Boolean -scopeStr String -returnId Boolean	POST	Crea una nuova sessione con le informazioni fornite nei parametri
read	/sessionId	-include ExtendedData Boolean	GET	Fornisce le informazioni di una specifica sessione
read	/sessionId/ privileges		GET	Fornisce una lista di privilegi per una specifica sessione
readAll			GET	Fornisce una lista di tutte le sessioni disponibili
update	/sessionId	-data json - includeExtendedData Boolean	PUT	Aggiorna una sessione con le informazioni specificate nei parametri
delete	/sessionId		DELETE	Elimina una sessione
deleteAll	/all		DELETE	Elimina tutte le sessioni
keepAlive	/id/keepalive		POST	Tiene attiva una sessione

5.2.70. Configurazione di sistema

http://hostname:port/cmdbuild/services/rest/v3/system/config

Funzione	Percorso	Parametri	Tipo	Descrizione
getSystemConfig		-detailed	GET	Fornisce la configurazione

		Boolean		di sistema
getSystemConfig	/key	-includeDefault Boolean	GET	Fornisce una specifica configurazione
updateSystemConfigValue	/key	-value String -encrypt Boolean	PUT	Aggiorna una specifica configurazione
updateSystemConfigValues	/_MANY	-data Map<String, String>	PUT	Aggiorna molteplici configurazioni
deleteSystemConfigValue	/key		DELETE	Elimina una specifica configurazione
reloadConfig	/reload		POST	Ricarica la configurazione di sistema

5.2.71. Sistema

http://hostname:port/cmdbuild/services/rest/v3/system

Funzione	Percorso	Parametri	Tipo	Descrizione
status	/status		GET	Fornisce una lista contenente gli stati dei servizi del sistema
getClusterStatus	/cluster/status		GET	Fornisce lo stato del sistema di cluster
dropCache	/cache/drop		POST	Invalida tutta la cache di sistema
dropCache	/cache/cacheId/drop		POST	Invalida una cache specifica
getCacheStats	/cache/stats		GET	Fornisce una lista contenente tutte le cache
stopSystem	/stop		POST	Ferma l'esecuzione del sistema
reloadSystem	/reload		POST	Ricarica l'intero sistema
restartSystem	/restart		POST	Riavvia l'intero sistema
upgradeSystem	/upgrade	-file DataHandler	POST	Aggiorna il sistema con il file fornito nei parametri
dropAudit	/audit/drop		POST	Elimina gli audit
cleanupAudit	/audit/cleanup		POST	Pulisce gli audit di sistema
getAllPatches	/patches		GET	Fornisce una lista di tutte le patch
getAllTenants	/tenants		GET	Fornisce una lista di tutti i tenant disponibili
getSchedulerJobs	/scheduler/jobs		GET	Fornisce una lista di tutti i Job disponibili nello

				scheduler
triggerJobNow	/scheduler/job/ jobId/trigger	-jobId	POST	Attiva l'esecuzione di un job
getAllLoggers	/loggers		GET	Fornisce una lista di tutti i logger disponibili
updateLoggerLevel	/loggers/key	-loggerCategory String -loggerLevel String	POST	Aggiorna il livello di un logger specifico
addLoggerLevel	/loggers/key	-loggerCategory String -loggerLevel String	PUT	Aggiunge un livello ad un logger esistente
deleteLoggerLevel	/loggers/key	-loggerCategory String	DELETE	Elimina un logger specifico
receiveLogMessages	/logger/stream		POST	Abilita lo stream dei logger attuali
stopReceivingLogMessages	/logger/stream		DELETE	Ferma lo stream dei logger attuali
dumpDatabase	/database/dump		GET	Crea un dump del database usato dal sistema
reconfigureDatabase	/database/ reconfigure	-dbConfig Map<String, String>	POST	Riconfigura il database con la configurazione fornita nei parametri
importDatabaseFromDump	/database/ import	-file DataHandler	POST	Riconfigura il database con il file di configurazione fornito nei parametri
generateDebugInfo	/debuginfo/ download		GET	Genera un bug report
sendBugReport	/debuginfo/send	-message String	POST	Invia un bug report con il messaggio specificato
sendBroadcastMessage	/messages/ broadcast	-message String	POST	Invia un alert di sistema con il messaggio specificato

5.2.72. Tenants

http://hostname:port/cmdbuild/services/rest/v3/tenants

Funzione	Percorso	Parametri	Tipo	Descrizione
getAll		-limit Long -start Long	GET	Fornisce una lista di tutti i tenant disponibili
configureMultitenant	/configure	-configData json	POST	Imposta la configurazione multitenant con le informazioni fornite nei parametri

5.2.73. Traduzioni

http://hostname:port/cmdbuild/services/rest/v3/translations

Funzione	Percorso	Parametri	Tipo	Descrizione
getAll		-limit Long -start Long -filter String	GET	Fornisce una lista di tutte le traduzioni disponibili
getTranslationForKeyAndLang	/code	-lang String	GET	Fornisce la traduzione di un elemento specifico nella lingua specifica
setTranslation	/code	-data jdon	PUT	Imposta una traduzione di un elemento specifico
deleteTranslation	/code	-lang String	DELETE	Rimuove una traduzione specifica
export	/export	-language String -format String -filter String -separator String	GET	Fornisce un file del formato specificato contenente una specifica traduzione

5.2.74. Uploads

http://hostname:port/cmdbuild/services/rest/v3/uploads

Struttura dati upload:

- path String
- description String

Funzione	Percorso	Parametri	Tipo	Descrizione
readMany		-dir String	GET	Fornisce una lista di tutti i file in una directory specifica
readFile	/fileId		GET	Fornisce le informazioni di un file specifico
downloadFile	/fileId/file:		GET	Scarica un file specificp
downloadManyFiles	/_MANY/file:.zip	-dir	GET	Scarica molteplici file
downloadAllFiles	/_ANY/file:.zip		GET	Scarica tutti i file disponibili
loadFile		-file DataHandler -directoryPath String	POST	Carica un nuovo file nel sistema

loadZipFile	/_MANY	-dataHandler DataHandler	POST	Carica un file zip contenente uno o più file
updateFile	/fileId	-file DataHandler	PUT	Aggiorna un file esistente con le informazioni fornite nei parametri
deleteFile	/fileId		DELETE	Elimina un file specifico

5.2.75. Users

http://hostname:port/cmdbuild/services/rest/v3/users

Struttura dati user:

- username String
- description String
- email String
- password String
- initialPage String
- active boolean
- service boolean
- language String
- multiGroup boolean
- multiTenant boolean
- multiTenantActivationPrivileges String
- defaultUserGroup Long
- userTenants List<TenantInfo>
- userGroups List<UserRole>

Funzione	Percorso	Parametri	Tipo	Descrizione
readMany		-filter String -sort String -limit Long -start Long -detailed Boolean	GET	Fornisce una lista di tutti gli utenti disponibili
readOne	/userId		GET	Fornisce le informazioni di un utente specifico
create		-data json	POST	Crea un nuovo utente con le informazioni fornite nei

				parametri
update	/userId	-data json	PUT	Aggiorna un utente esistente con le informazioni fornite nei parametri
changePassword	/current/ password	-data json	PUT	Aggiorna la password dell'utente attualmente autenticato

5.2.76. Viste su cards

http://hostname:port/cmdbuild/services/rest/v3/views/viewId/cards

Funzione	Percorso	Parametri	Tipo	Descrizione
create		-data Map<String,Object>	POST	Crea una nuova vista con le informazioni fornite nei parametri
readOne	/cardId		GET	Fornisce le informazioni di una card con la vista specificata
readMany		-filter String -sort String -limit Long -start Long -positionOf Long -forDomainName String -forDomainDirection String -forDomainOriginId Long	GET	Fornisce una lista di informazioni con la vista specificata
update	/cardId	-data Map<String,Object>	PUT	Aggiorna le informazioni di una card con la vista specifica
delete	/cardId		DELETE	Elimina una card con la vista specificata

5.2.77. Viste

http://hostname:port/cmdbuild/services/rest/v3/views

Struttura dati viste:

- name String
- description String
- sourceClassName String

- sourceFunction String
- filter String
- active boolean
- type String

Funzione	Percorso	Parametri	Tipo	Descrizione
list			GET	Fornisce una lista di tutte le view disponibili
getOne	/viewId		GET	Fornisce le informazioni di una vista specifica
create		-data json	POST	Crea una nuova vista con le informazioni fornite nel parametro data
update	/viewId	-data json	PUT	Aggiorna una vista specifica con le informazioni fornite nel campo data
delete	/viewId		DELETE	Elimina una vista esistente

5.3. Esempi REST

In questo paragrafo verranno forniti vari esempi di chiamate REST. Da notare che CMDBuild nello scenario degli esempi seguenti è configurato con il database demo.dump.xz, se si vogliono riprodurre gli esempi allo stesso modo bisognerà caricare tale dump.

Per effettuare le chiamate mostrate si possono utilizzare vari strumenti, ad esempio possono essere effettuate via terminale con il tool curl su sistemi linux, oppure con il supporto di interfacce grafiche fornite da software quali “Postman” o qualsiasi altro software che possa effettuare richieste HTTP.

Ogni richiesta richiede che l’utente specifichi il campo “Cmdbuild-authorization” nell’header. Tale campo non è altro che l’id della sessione di un utente autenticato nel sistema. La prima chiamata esempio mostrerà, infatti, come ottenere tale id di sessione.

5.3.1. Generare un token di sessione

L’endpoint da utilizzare per generare un token di sessione è il seguente:

http://hostname:port/cmdbuild/services/rest/v3/sessions

La richiesta dovrà essere di tipo POST, l’utente e la password verranno inoltre fornite per poter creare la nuova sessione. Se nella risposta vogliamo ottenere il sessionId in chiaro, dalla versione 3.2 bisogna aggiungere un query parameter valorizzato a true alla richiesta, il parametro è ‘returnId’. Se il parametro non viene settato a true, il sessionId verrà rimpiazzato dal valore stringa ‘current’.

La richiesta apparirà come la seguente:

```
POST http://hostname:port/cmdbuild/services/rest/v3/sessions?scope=service
HTTP/1.1
Content-Type:application/json
{
 username : admin,
 password : admin
}
```

La risposta ottenuta apparirà come la seguente, dove la voce _id sarà l’id della sessione, seguita da altre informazioni quale la lista di ruoli disponibili e altre.

```
HTTP/1.1 200 OK
Content-Type:application/json
{
 “success”: true,
 “data” : {
       “_id”:“sessionId”,
```

```
       “username”:”admin”,
       “userDescription”:”Administrator”,
       “role”:”SuperUser”,
       “availableRoles”:[
       “SuperUser”
       ],
 ...
 }
}
```

Una sessione attiva verrà eliminata dopo un certo lasso di tempo, forzando l’utente a ricreare la sessione di tanto in tanto

Tale autenticazione non va effettuata ad ogni richiesta dato che l’utente può fornire l’id di sessione nell’header di ogni richiesta.

5.3.2. Ottenere la lista delle classi disponibili

Se l’utente volesse ottenere una lista di tutte le classi disponibili, l’endpoint da utilizzare è:

http://hostname:port/cmdbuild/services/rest/v3/classes

La richiesta sarà di tipo GET e apparirà come la seguente:

```
GET http://hostname:port/cmdbuild/services/rest/v3/classes?scope=service
HTTP/1.1
Cmdbuild-authorization:sessionId
```

La risposta satà come la seguente, i risultati all’interno della risposta dovrebbero essere 24 nel caso di utilizzo del database demo, ma per limitare la lunghezza della risposta solo il primo risultato è stato presentato:

```
HTTP/1.1 200 OK
Content-Type:application/json
{
 “success”: true,
 “data” : {
   “_id”:“Invoice”,
   "name": "Invoice",
   "description": "Invoice",
   "_description_translation": "Invoice",
   "prototype": false,
   "parent": "Class",
```

```
   "active": true,
   "type": "standard",
   "_can_read": true,
   "_can_create": true,
   "_can_update": true,
   "_can_clone": true,
   "_can_delete": true,
   "_can_modify": true,
   "defaultFilter": null,
   "description_attribute_name": "Description",
   "metadata": {},
   "_icon": null
  },
. . .
,
 "meta": {
  "total": 24
 }
}
```


Alla fine della risposta nel campo “meta” vari oggetti aggiuntivi vengono presentati, in questo caso esempio vediamo la voce “total” che ci presenta il numero totale di risultati forniti.

Si noti che i vari parametri presentati precedentemente nella docmentazione, possono essere aggiunti alla richiesta (parametri tipo activeOnly, detailed, limit, …). Se, per esempio, si volesse limitare il numero di risultati a 2, il parametro limit può essere aggiunto alla richiesta come di seguito:

```
http://hostname:port/cmdbuild/services/rest/v3/classes?
scope=service&limit=2
```

Stessa cosa per l’aggiunta di altri eventuali parametri.

5.3.3. Ottenere le informazioni di una specifica classe

Se, invece della lista delle classe, lo user volesse ottenere le informazioni di solo una classe specifica, l’endpoint sarà lo stesso con l’aggiunta dell’id della classe della quale si vogliono ottenere le informazioni:

http://hostname:port/cmdbuild/services/rest/v3/classes/classId

Dove classId assumerà il valore della classe che si vuole ottenere, per esempio se si volesse ottenere le informazioni della classe con _id=invoice, l’endpoint sarà:

http://hostname:port/cmdbuild/services/rest/v3/classes/Invoice

La richiesta sarà di tipo GET e apparirà come la seguente

```
GET http://hostname:port/cmdbuild/services/rest/v3/classes/Invoice?
scope=service HTTP/1.1
Cmdbuild-authorization:sessionId
```

La risposta conterrà le informazioni solo di quella specifica classe:

```
HTTP/1.1 200 OK
Content-Type:application/json
{
 “success”: true,
 “data” : {
   “_id”:“Invoice”,
   "name": "Invoice",
   "description": "Invoice",
   "_description_translation": "Invoice",
   "prototype": false,
   "parent": "Class",
   "active": true,
   "type": "standard",
   . . .
```

5.3.4. Creazione di una nuova classe

Se l’obiettivo della richiesta è quello di creare una nuova classe l’endpoint sarà

http://hostname:port/cmdbuild/services/rest/v3/classes

Le informazioni della nuova classe saranno inserite nella richiesta POST che verrà effettuata:

```
POST http://hostname:port/cmdbuild/services/rest/v3/classes HTTP/1.1
Cmdbuild-authorization:sessionId
Content-Type:application/json
{
 “name”:”testClass”,
 “type”:”standard”
}
```

In questo caso la classe è stata creata con solo le due informazioni principali (il nome ed il tipo della classe), nella richiesta possono essere aggiunti tutti i campi che si vogliono valorizzare.

Una volta che la richiesta è stata effettuata, la riposta conterrà le informazioni della classe appena creata:

```
HTTP/1.1 200 OK
Content-Type:application/json
 {
 "success": true,
 "data": {
  "_id": "testClass",
  "name": "testClass",
  "description": "",
  "_description_translation": "",
  "prototype": false,
  "parent": "Class",
  "active": true,
  "type": "standard",
   . . .
```

Se adesso venisse fatta una richiesta per ottenere informazioni relative alla classe appena creata:

```
GET http://hostname:port/cmdbuild/services/rest/v3/classes/testClass?
scope=service HTTP/1.1
Cmdbuild-authorization:sessionId
```

Otterremmo le informazioni appena aggiunte.

5.3.5. Aggiornare una classe esistente

Se una classe è già stata creata, viene fornita la possibilità di aggiornare le informazioni della classe tramite una richiesta PUT, l'endpoint sarà:

http://hostname:port/cmdbuild/services/rest/v3/classes/classId

E nella richiesta tutti i campi che richiedono un aggiornamento possono essere inseriti, per esempio se si volesse aggiornare il campo description della classe precedentemente creata a "test description" la richiesta apparirebbe come la seguente:

```
PUT http://hostname:port/cmdbuild/services/rest/v3/classes/testClass
HTTP/1.1
Cmdbuild-authorization:sessionId
Content-Type:application/json
{
```

```
 “name”:”testClass”,
 “type”:”standard”,
 “description”:”test description”
}
```

La risposta conterrà le informazioni della classe aggiornata:

```
HTTP/1.1 200 OK
Content-Type:application/json
 {
 "success": true,
 "data": {
  "_id": "testClass",
  "name": "testClass",
  "description": "test description",
  "_description_translation": "",
  "prototype": false,
  "parent": "Class",
  "active": true,
  "type": "standard",
   . . .
```

Se adesso venisse fatta una richiesta per ottenere informazioni relative alla classe appena aggiornata:

```
GET http://hostname:port/cmdbuild/services/rest/v3/classes/testClass?
scope=service HTTP/1.1
Cmdbuild-authorization:sessionId
```

Otterremmo le informazioni appena aggiunte.

6. Appendice: Glossario

6.1.1. ALLEGATO

Per “allegato” si intende un qualunque file associabile ad una scheda dati inserita nel sistema.

Per la gestione degli allegati CMDBuild utilizza in modalità embedded un qualunque sistema documentale compatibile con il protocollo standard CMIS (oppure il DMS Alfresco fino alla versione 3 tramite il proprio webservice nativo).

La gestione degli allegati supporta il versioning di file caricati più volte, con numerazione automatica.

6.1.2. ATTIVITA'

Per “attività” si intende uno dei passaggi che costituiscono il processo.

Una attività è caratterizzata da un nome, un esecutore, un tipo, eventuali attributi, eventuali metodi associati ad API di CMDBuild per poter essere eseguiti.

Per “istanza di attività” si intende una specifica attivazione di una attività, effettuata automaticamente dal sistema o manualmente da un operatore.

Vedi anche: Processo

6.1.3. ATTRIBUTO

Il termine indica nel sistema CMDBuild la generica tipologia di informazione descrittiva di una determinata classe.

CMDBuild consente tramite il Modulo Schema di creare nuovi attributi in una classe o in un dominio e di modificarne alcune caratteristiche.

Nella classe “Fornitore” gli attributi sono ad esempio il nome, l'indirizzo, il numero di telefono, ecc.

Ogni attributo corrisponde nel Modulo di Gestione a campi di inserimento dati sulla apposita scheda di gestione della classe e a colonne della corrispondente tabella nel database.

Vedi anche: Classe, Dominio, Relazione, Superclasse, Tipo di attributo

6.1.4. BIM

Metodologia che si pone l'obiettivo di supportare l’intero ciclo di vita di un edificio, dall'idea iniziale alla fase di costruzione, di utilizzo e manutenzione, fino alla eventuale demolizione finale.

La metodologia BIM (Building Information Modeling) è supportata da numerosi programmi informatici che possono interagire tramite un formato aperto di scambio dati denominato IFC (Industry Foundation Classes).

Vedi anche: GIS

6.1.5. CI

Si definisce Configuration Item (Elemento della Configurazione) ogni elemento che concorre a fornire il servizio IT all’Utente, considerato ad un livello di dettaglio sufficiente per la sua gestione tecnica e patrimoniale.

Esempi di CI sono: server, workstation, programma applicativo, sistema operativo, stampante, ecc

Vedi anche: Configurazione

6.1.6. CLASSE

Il termine rappresenta un tipo di dati complesso caratterizzato da un insieme di attributi che nel loro insieme descrivono quel tipo di dato.

Una classe modella una tipologia di oggetto da gestire nel CMDB, quale ad esempio un computer, una applicazione software, un servizio, un fornitore, ecc

CMDBuild consente all'Amministratore del Sistema, attraverso il Modulo Schema, di definire nuove classi e di cancellare o modificare la struttura di classi già definite.

Una classe è rappresentata a video da una apposita scheda di gestione dati e nel database da una tavola generata automaticamente al momento della definizione della classe.

Vedi anche: Scheda, Attributo

6.1.7. CONFIGURAZIONE

Il processo di Gestione della Configurazione ha lo scopo di mantenere aggiornata e disponibile per gli altri processi la base di informazioni relativa agli oggetti informatici gestiti (CI), alle loro relazioni ed alla loro storia.

E' uno dei principali processi gestiti dal sistema ITIL.

Vedi anche: CI, ITIL

6.1.8. DASHBOARD

Una dashboard corrisponde in CMDBuild ad una raccolta di grafici di diversa tipologia, tramite cui avere immediata evidenza di alcuni parametri chiave (KPI) relativi ad un particolare aspetto di gestione del servizio IT.

Vedi anche: Report

6.1.9. DATABASE

Il termine indica un insieme di informazioni strutturato ed organizzato in archivi residenti sull'elaboratore server, nonché l'insieme dei programmi di utilità dedicati alla gestione dei tali informazioni per attività quali inizializzazione, allocazione degli spazi, ottimizzazione, backup, ecc.

CMDBuild si appoggia sul database PostgreSQL, il più potente, affidabile e completo database Open Source, di cui utilizza in particolare le sofisticate funzionalità e caratteristiche object oriented.

6.1.10. DOMINIO

Un dominio rappresenta una tipologia di relazione fra una coppia di classi.

E' caratterizzato da un nome, dalle descrizioni della funzione diretta ed inversa, dai codici delle due classi e dalla cardinalità (numerosità degli elementi relazionabili) ammessa, nonché dagli eventuali attributi configurati.

CMDBuild consente all'Amministratore del Sistema, attraverso il Modulo Schema, di definire nuovi domini e di cancellare o modificare la struttura di domini già definiti.

E' possibile caratterizzare ciascun dominio tramite definizione di attributi custom.

Vedi anche: Classe, Relazione

6.1.11. FILTRO DATI

Un filtro dati è una restrizione della lista degli elementi contenuti in una classe, ottenuta specificando condizioni booleane (uguale, diverso, contiene, inizia, ecc) sui possibili valori assumibili da ciascun attributo della classe.

I filtri dati possono essere definiti ed utilizzati “una tantum”, oppure possono essere memorizzati dall'operatore e richiamati successivamente (dallo stesso operatore o da operatori di altri gruppi di utenti ai quali l'Amministratore del sistema abbia concesso l'utilizzo).

Vedi anche: Classe, Vista

6.1.12. GIS

Un sistema GIS è un sistema informatico in grado di produrre, gestire e analizzare dati spaziali associando a ciascun elemento geografico una o più descrizioni alfanumeriche.

Le funzionalità GIS implementate in CMDBuild consentono di creare attributi geometrici, in aggiunta a quelli testuali, tramite cui rappresentare su scala locale (planimetrie) o su scala più estesa (mappe esterne) elementi puntuali (ad esempio gli asset IT), poligonali (ad esempio linee dati) o aree (piani, stanze, ecc).

Vedi anche: BIM

6.1.13. GUI FRAMEWORK

E' una interfaccia utente completamente personalizzabile e orientata a fornire un accesso semplificato all'applicazione, pubblicabile su portali web di qualsiasi tecnologia ed interoperabile con CMDBuild tramite il webservice REST standard.

Vedi anche: Mobile, Webservice

6.1.14. ITIL

Sistema di "best practice" ormai affermatosi come "standard de facto", non proprietario, per la gestione dei servizi informatici secondo criteri orientati ai processi (Information Technology Infrastructure Library).

Fra i processi fondamentali coperti da ITIL ci sono quelli del Service Support, comprendenti l'Incident Management, il Problem Management, il Change Management, il Configuration Management ed il Release Management.

Per ogni processo considera la descrizione, i componenti di base, i criteri e gli strumenti consigliati per la misura della qualità del servizio, i ruoli e le responsabilità delle risorse coinvolte, i punti di integrazione con gli altri processi (per eliminare duplicazioni e inefficienze).

Vedi anche: Configurazione

6.1.15. LOOKUP

Con il termine “LookUp” si indica una coppia di valori del tipo (Codice, Descrizione) impostabili dall'Amministratore del Sistema tramite il Modulo Schema.

Tali valori vengono utilizzati dall'applicazione per vincolare la scelta dell'utente, al momento della compilazione del relativo campo sulla scheda dati, ad uno dei valori preimpostati.

Il Modulo Schema consente la definizione di nuove tabelle di “LookUp” secondo le necessità dell'organizzazione.

6.1.16. MOBILE

E' una interfaccia utente ottimizzata per strumenti “mobile” (smartphone e tablet), implementata come “app” multipiattaforma (iOS, Android) ed interoperabile con CMDBuild tramite il webservice REST standard.

Vedi anche: GUI Framework, Webservice

6.1.17. PROCESSO

Per “processo” (o workflow) si intende una sequenza di passaggi (“attività”) descritti nel sistema per svolgere in forma guidata e secondo regole prestabilite una determinata azione.

Per ogni processo saranno avviate in CMDBuild una serie di “istanze di processo”, una per ogni necessità di effettiva esecuzione dell'azione corrispondente, che avrà luogo su “asset” specifici e sarà svolta da utenti specifici.

Una ”istanza di processo” viene attivata tramite avvio e conferma del primo passaggio previsto e termina alla esecuzione dell'attività finale prevista nella definizione.

Vedi anche: Attività

6.1.18. RELAZIONE

Per “Relazione” si intende in CMDBuild un collegamento effettivo di due schede appartenenti a due classi, o in altri termini una istanza di un dato dominio.

Una relazione è quindi definita da una coppia di identificativi univoci delle due schede collegate e dall'identificativo del dominio utilizzato per il collegamento, nonché dalla valorizzazione degli eventuali attributi previsti nel dominio.

CMDBuild consente agli operatori del Sistema, attraverso il Modulo Gestione Dati, di definire nuove relazioni fra le schede archiviate nel database.

Vedi anche: Classe, Dominio

6.1.19. REPORT

Il termine indica in CMDBuild una stampa (in formato PDF o CSV) riportante in forma analitica le informazioni estratte da una o più classi fra le quali sia definita una catena di domini.

I report possono essere generati e modificati dagli operatori di CMDBuild tramite una apposita funzione del Modulo di Gestione Dati e la relativa definizione viene memorizzata nel database per poter essere riutilizzata successivamente.

Vedi anche: Classe, Dominio, Database

6.1.20. SCHEDA

Con il termine “Scheda” in CMDBuild si riferisce un elemento archiviato in una determinata classe.

Una scheda è caratterizzata da un insieme di valori assunti da ciascuno degli attributi definiti per la sua classe di appartenenza.

CMDBuild consente agli operatori del Sistema, attraverso il Modulo Gestione Dati, di archiviare nuove schede nel database e di aggiornare schede già archiviate.

Le informazioni di ogni scheda saranno memorizzate nel database alle opportune colonne di una riga della tavola generata per la classe di appartenenza della scheda.

Vedi anche: Classe, Attributo

6.1.21. SUPERCLASSE

Una superclasse è una classe astratta utilizzabile per definire una sola volta attributi condivisi fra più classi. Da tale classe astratta è poi possibile derivare classi reali che conterranno i dati effettivi e che comprenderanno sia gli attributi condivisi (specificati nella superclasse) che quelli specifici della sottoclasse.

Ad esempio è possibile definire la superclasse “Computer” con alcuni attributi base (RAM, HD, ecc) e le sottoclassi derivate “Desktop”, “Notebook”, “Server”, ciascuna delle quali con i soli attributi specifici.

Vedi anche: Classe, Attributo

6.1.22. TIPO DI ATTRIBUTO

Ogni attributo definito per una determinata classe è caratterizzato da un “Tipo” che determina le caratteristiche delle informazioni contenute e la loro modalità di gestione.

Il tipo di attributo viene definito con il Modulo Schema e può essere poi modificato entro alcuni limiti dipendenti dalla tipologia dei dati già archiviati.

CMDBuild gestisce i seguenti tipi di attributo: “Boolean” (booleano, Si / No), “Date” (data), “Decimal” (decimale), “Double” (virgola mobile in doppia precisione), “Inet” (indirizzo IP), “Integer” (numero intero), “LookUp” (tabellato da lista configurabile in “Impostazioni” / “LookUp”), “Reference” (riferimento o foreign key), “String” (stringa), “Text” (testo lungo), “TimeStamp” (data e ora).

Vedi anche: Attributo

6.1.23. VISTA

Una vista è un insieme di schede definito in modo “logico” anziché dal fatto di costituire l'intero contenuto di una classe nel CMDB.

In particolare una vista può essere definita in CMDBuild applicando un filtro ad una classe (quindi conterrà un insieme ridotto delle stesse righe) oppure specificando una funzione SQL che estragga attributi da una o più classi correlate.

La prima tipologia di vista mantiene tutte le funzionalità disponibili per una classe, la seconda consente la sola visualizzazione e ricerca con filtro veloce.

Vedi anche: Classe, Filtro

6.1.24. WEBSERVICE

Un webservice è un'interfaccia che descrive una collezione di operazioni, accessibili attraverso una rete mediante messaggistica XML.

Tramite un webservice una applicazione può rendere accessibili le proprie funzionalità ad altre applicazioni operanti attraverso il web.

CMDBuild dispone di un webservice SOAP e di un webservice REST.

6.1.25. WIDGET

Un widget è un componente grafico di una interfaccia utente di una applicazione software, che ha lo scopo di facilitare all'utente l'interazione con l'applicazione stessa.

CMDBuild prevede l'utilizzo di widget sotto forma di “pulsanti” posizionabili su schede dati o su schede di avanzamento di processi. I pulsanti aprono finestre di tipo “popup” tramite cui inserire se richiesto informazioni aggiuntive e visualizzare poi l'output della funzione richiamata.