Manuale Web Services 4.2

Titulus Web Services 4.2

Scopo

In questo documento si descrive la piattaforma dei Web Service esposti da Titulus XML. Vengono descritte le modalità per la connessione all’applicativo, nonché la ricerca, modifica e registrazione di documenti.

Riferimenti
Java DOC dei Web Service.

A questa documentazione si accede dalla pagina di presentazione dei servizi.

Pagina di presentazione dei servizi
Titulus espone una pagina di presentazione dei web service, il cui URL è

```
http://<host>:<port>/titulus_ws/services
```

dove <*host*> e <*port*> sono quelli della macchina che ospita i web service.

In questa pagina viene presentato un elenco dei servizi esposti e, per ognuno di essi, oltre all'elenco dei metodi che possono essere invocati, vengono forniti dei link per accedere al **WSDL**, al **javadoc** e alla **corrispondente pagina del manuale**.

A titolo di esempio, di seguito viene riportata una porzione di questa pagina.

Servizio di base
Il servizio di base è *Titulus4*, usato per l'accesso all'archivio dei documenti.

L'URL del servizio ha la seguente struttura (<*host*> e <*port*> sono quelli della macchina che ospita i web service):

```
http://<host>:<port>/titulus_ws/services/Titulus4
```

Il **WSDL** del servizio, invece, può essere recuperato al seguente URL (esso viene riportato anche nella pagina di presentazione dei servizi):

```
http://<host>:<port>/titulus_ws/services/Titulus4?wsdl
```

Servizi specifici
Oltre al servizio di base, Titulus espone i seguenti servizi che espongono delle funzionalità specifiche:

- TitulusOrgani4 (manuale)
- Acl4 (manuale)

Autenticazione

I web service di Titulus sono indipendenti dal tipo di autenticazione usata. E' quindi possibile impiegare la Basic Authentication di Tomcat, Shibboleth, l'autenticazione di dominio o altro.

Premesso ciò, tutte le chiamate http effettuate verso i servizi di Titulus 4 *devono* essere autenticate e, di conseguenza, devo avere il remote user (header http) valorizzato con la login specificata nell'autenticazione.

Questo tipicamente si traduce a livello del client nell'indicare uno username ed una password prima di effettuare la batteria di chiamate ai metodi esposti dal servizio.

Superata l'autenticazione, il servizio provvede al caricamento del profilo associato all'utente dichiarato nel login ed effettua l'operazione richiesta usando quel profilo.

Di seguito si riporta un esempio in linguaggio Java, realizzato con la libreria Axis. Lo stub client usato (nel codice "*titulus4*"), è stato generato a partire dal wsdl del servizio, mediante il tool WSDL2Java.

Nell'esempio viene effettuata l'autenticazione con l'utente "test-ws", che *deve essere registrato in ACL* (anagrafica di Titulus) con login "test-ws".

```
Titulus4ServiceLocator tsl = new Titulus4ServiceLocator();

// impostazione dello scope di chiamata a "session"
tsl.setMaintainSession(true);

Titulus4_PortType titulus4 = tsl.getTitulus4();

// impostazione della login per l'accesso autenticato al servizio
((Titulus4SoapBindingStub)titulus4).setUsername("test-ws");
((Titulus4SoapBindingStub)titulus4).setPassword("XXX");

...

// chiamate al servizio
titulus4.search(...);
...
titulus4.loadFirstDocument();

...
```

Con riferimento all'esempio sopra riportato, tutte le chiamate al servizio vengono effettuate usando il profilo associato all'utente "test-ws".

Il cambiamento dell'utente dichiarato nell'autenticazione scatena il cambiamento del profilo con cui vengono eseguite le successive chiamate al servizio.

In caso di mancata autenticazione o di utente inesistente (nell'anagrafica di Titulus) viene segnalato un errore.

Cambiamento del profilo usato per le operazioni

Come indicato nel paragrafo precedente, i metodi dei servizi di Titulus 4 operano impiegando il profilo (definito nell'anagrafica di Titulus) associato all'utente dichiarato nell'autenticazione della chiamata http e solitamente in questi casi si tratta di un *profilo applicativo*.

Qualora il client voglia operare come un utente "reale" di Titulus (es. "Mario Rossi") può cambiare il profilo usato invocando il metodo **setWSUser()**.

Il metodo funziona solo se il profilo applicativo del client (quello associato all'autenticazione) possiede il diritto di "*trusted application*"; in caso contrario restituisce un messaggio di errore.

Per i dettagli sul funzionamento del metodo si rimanda al javadoc - consultabile dalla Pagina di presentazione dei servizi.

Di seguito si riporta un esempio in linguaggio Java, realizzato con la libreria Axis. Lo stub client usato (nel codice "*titulus4*"), è stato generato a partire dal wsdl del servizio, mediante il tool WSDL2Java.

Nell'esempio viene illustrato un uso tipico del metodo setWSUser(); l'autenticazione viene effettuata con il solito utente "test-ws" - che *deve essere registrato in ACL* (anagrafica di Titulus) con login "test-ws".

```
Titulus4ServiceLocator tsl = new Titulus4ServiceLocator();

// impostazione dello scope di chiamata a "session"
tsl.setMaintainSession(true);

Titulus4_PortType titulus4 = tsl.getTitulus4();

// impostazione della login per l'accesso autenticato al servizio
((Titulus4SoapBindingStub)titulus4).setUsername("test-ws");
((Titulus4SoapBindingStub)titulus4).setPassword("XXX");

...

// il client, dopo aver chiesto all'utente la sua login ("mrossi"), effettua il cambiamento
// del profilo

String resp = titulus4.setWSUser("mrossi", null);

// lettura della risposta: sono stati trovati più profili associati a "mrossi" --> si propone
// l'elenco all'utente affinché ne scelga uno.
// La scelta comporta una nuova chiamata a setWSUser indicando anche la matricola del profilo
// scelto.

resp = titulus4.setWSUser("mrossi", pnumber);

// da questo momento in poi si opera usando il profilo individuato dalla coppia
// ("mrossi", pnumber)

titulus4.search(...);

...
```

Come illustrato nell'esempio, da quando setWSUser() ha cambiato il profilo - e fino ad una successiva chiamata a setWSUser() o fino allo scadere della sessione http - tutte le chiamate al servizio vengono effettuate usando il nuovo profilo e tutte le operazioni di modifica e inserimento vengono tracciate nella storia dei documenti di Titulus in un modo differente da quello consueto: l'operatore che ha compiuto l'operazione, infatti, viene tracciato come “Rossi Mario [tramite Applicazione Test WS]”, dove “Rossi Mario” è l'utente impostato con la chiamata a setWSUser(), mentre “Applicazione Test WS” rappresenta il profilo associato all'autenticazione.

Mantenimento della sessione http delle chiamate

I servizi di Titulus presuppongono il mantenimento di alcune informazioni di stato nella sessione http associata alle richieste del client. Tali informazioni - oltre a garantire migliori performance nella gestione dei profili acl - consentono, per esempio, la paginazione dei risultati delle ricerche ed il cambiamento di profilo illustrato nel paragrafo Cambiamento del profilo usato per le operazioni.

Il mantenimento delle sessioni http deve essere richiesto dal client; la modalità della richiesta dipende dalla libreria usata per implementare il client.

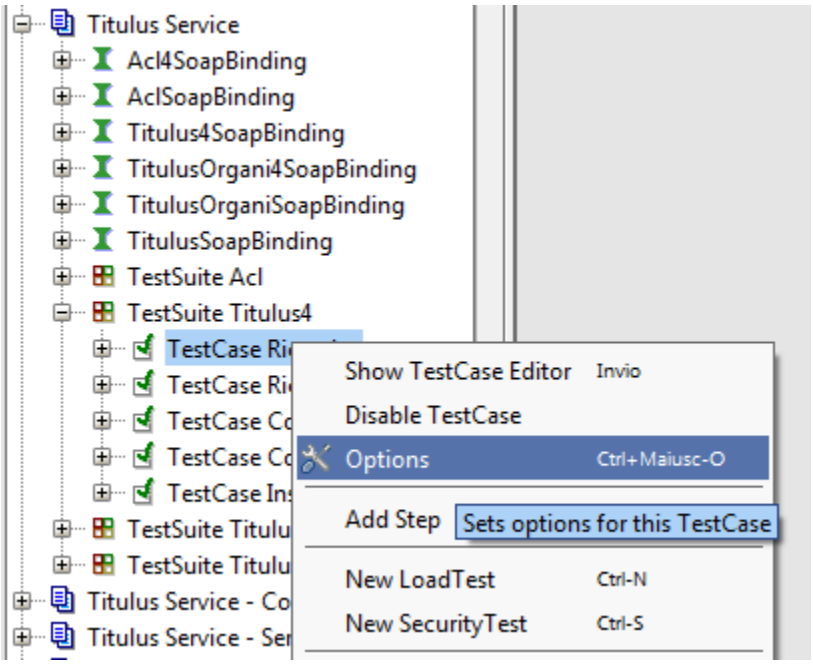
Di seguito si riporta un esempio in linguaggio Java, realizzato con la libreria Axis. Lo stub client usato (nel codice “*tsl*”), è stato generato a partire dal wsdl del servizio, mediante il tool WSDL2Java.

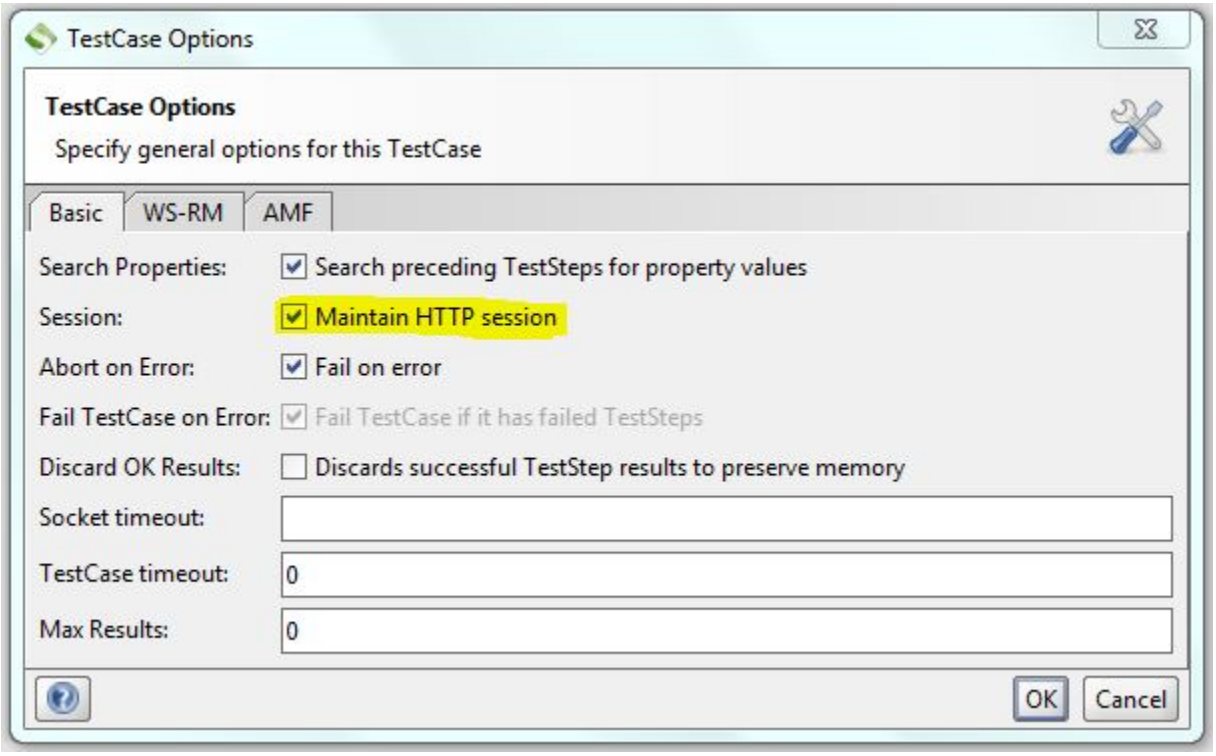
```
Titulus4ServiceLocator tsl = new Titulus4ServiceLocator();

// impostazione dello scope di chiamata a "session"
tsl.setMaintainSession(true);

...
```

Usando SoapUI per provare i servizi, è necessario creare un test case contenente le chiamate che interessano; andando, poi, nelle opzioni del test case è possibile impostare il mantenimento della sessione http, come illustrato nelle seguenti figure.





Il tempo di timeout - cioè di *inattività* - delle sessioni http dei servizi web di Titulus è di 5 minuti.

Le porte di accesso al servizio

Il Web Service espone una serie di porte d'accesso per la registrazione di nuovi documenti e la consultazione della base dati.

Di seguito riportiamo l'elenco dei metodi principali con una breve descrizione:

Ricerca

- **search()**: Effettua la ricerca dei termini espressi nella query secondo uno dei criteri impostati ed eventualmente raffinando rispetto all'esito di una ricerca precedente.
- **searchFullText()**: Effettua la ricerca *full text* del testo espresso secondo uno dei criteri impostati ed eventualmente raffinando rispetto all'esito di una ricerca precedente.
- **getDashboard()**: Recupera un set pre-configurato di vaschette, ovvero ricerche che raggruppano i documenti. Il metodo restituisce solo la label della vaschetta e la ricerca associata senza lanciarla.

Titoli e risultati di sintesi

- **firstTitlePage()**: Effettua il posizionamento sulla prima pagina dei titoli dei documenti del result set corrente.
- **nextTitlePage()**: Effettua il posizionamento sulla pagina successiva dei titoli dei documenti del result set corrente.
- **prevTitlePage()**: Effettua il posizionamento sulla pagina precedente dei titoli dei documenti del result set corrente.
- **lastTitlePage()**: Effettua il posizionamento sull'ultima pagina dei titoli dei documenti del result set corrente.
- **currentTitlePage()**: Effettua il posizionamento sulla pagina corrente dei titoli dei documenti del result set corrente.
- **titlePage()**: Effettua il posizionamento diretto sulla pagina dei titoli specificata (dei documenti del result set corrente).

Caricamento documenti (information unit)

- **loadDocument()**: Effettua il caricamento, tramite numero fisico, di una Information Unit con possibilità di locking.
- **loadFirstDocument()**: Effettua il caricamento della prima Information Unit del result set corrente.
- **loadNextDocument()**: Effettua il caricamento della successiva Information Unit del result set corrente.
- **loadLastDocument()**: Effettua il caricamento dell'ultima Information Unit del result set corrente.
- **loadPrevDocument()**: Effettua il caricamento della precedente Information Unit del result set corrente.
- **getDocumentURL()**: Fornisce l'url di accesso ad un documento.

Registrazione, modifica e cancellazione documenti

- **saveDocument()**: Effettua la registrazione di un nuovo documento.
- **saveModifiedDocument()**: Effettua il salvataggio di un documento modificato.
- **deleteDocument()**: Effettua, ove possibile, la rimozione di un documento.
- **unlockDocument()**: Effettua lo sblocco di un documento.
- **grantRight()**: Consente di assegnare un documento in Copia Conoscenza (CC), in Conferenza Di Servizi (CDS) o di modificare il Responsabile di Procedimento Amministrativo (RPA).
- **denyRight()**: Consente la rimozione di un utente dalla lista dei CC (Copia Conoscenza) o CDS (Conferenza Di Servizi) di un documento.
- **postIt()**: Consente l'inserimento di un postit in un documento.
- **cancelDocument()**: Consente, ove possibile, di annullare un documento.
- **applyRegistrationToDraft()**: Consente di protocollare un documento bozza.
- **applyRegistrationMark()**: Consente di applicare ad un documento la segnatura di protocollo.
- **getRegistrationMark()**: Restituisce la segnatura di protocollo di un documento.
- **replyDocument()**: Crea un documento non protocollato come copia di un documento originale e lo inserisce in un fascicolo.

Allegati multimediali

- **getAttachment()**: Effettua il download di un allegato di un Information Unit.
- **checkInContentFiles()**: Effettua l'upload (previo checkOutContentFile e quindi con versioning) di uno o più allegati (anche immagini) da associare ad un Information Unit.
- **checkOutContentFile()**: Effettua il download (eventualemente per apportare modifiche di versioning) di un allegato di un Information Unit.
- **unlockContentFile()**: Effettua lo sblocco (unlocking) di un allegato di un Information Unit precedentemente prenotato (checkOutContentFile).
- **validateContentFile()**: Verificare la validità dell'impronta (SHA256) dei file ed immagini associate ad un Information Unit.

Fascicoli

- **newFolder()**: Effettua la creazione di un nuovo fascicolo.
- **newSubFolder()**: Effettua la creazione di un nuovo sottofascicolo.
- **assignFolderRPA()**: Effettua il trasferimento di un fascicolo da un RPA ad un altro.
- **addInFolder()**: Effettua l'inserimento di un documento in un fascicolo.
- **removeFromFolder()**: Effettua la rimozione di un documento da un fascicolo.
- **addLinkToFolder()**: Effettua la creazione di un collegamento tra un documento ed un fascicolo.
- **deleteLinkToFolder()**: Effettua la rimozione del collegamento tra un documento ed un fascicolo.
- **getFolderContent()**: Effettua il recupero dei documenti contenuti e collegati ad un fascicolo.
- **getFolderHierarchy()**: Effettua il recupero della gerarchia (in forma sintetica) di sottofascicoli e documenti contenuti in un fascicolo.
- **getFolder()**: Effettua il recupero dei metadati XML rappresentativi di un fascicolo.
- **getDocumentFolders()**: Effettua il recupero dei metadati XML rappresentativi dei fascicoli (principale e collegati) di un documento.
- **closeFolder()**: Effettuano la chiusura di un fascicolo.
- **openFolder()**: Effettua la riapertura di un fascicolo.
- **deleteFolder()**: Effettua la rimozione di un fascicolo ma non del suo contenuto.

Workflow

- **getWorkflowId()**: Effettua il recupero delle eventuali istanze di workflow, attive o meno, associate ad un documento.
- **getWorkflowAction()**: Effettua il recupero delle eventuali azioni disponibili per una determinata istanza di workflow associata ad un documento.
- **startWorkflow()**: Effettua l'attivazione di un workflow associato ad un documento.
- **continueWorkflow()**: Effettua l'esecuzione di una specifica azione associata ad un istanza di workflow relativa ad un documento.

Informazioni sul servizio

- **getWsRelease()**: Restituisce le informazioni sulla release del servizio.

Per una descrizione dettagliata dei metodi e dei relativi parametri, si rimanda al Java Doc di riferimento. Segue una descrizione sommaria delle principali funzionalità messe a disposizione dal servizio.

Struttura delle tipiche risposte xml

I metodi di consultazione e di salvataggio dei documenti solitamente restituiscono due tipologie di risposte xml: un elenco di documenti (quando viene effettuata una ricerca); un singolo documento (quando si carica o si salva un documento).

Notare che in questo paragrafo per “documento” si intente un generico documento xml, quindi un documento - in arrivo, in partenza, tra uffici o non protocollato - con allegati, un fascicolo o un raccoglitore, ecc.

Di seguito si riporta la tipica struttura di un *elenco di documenti*:

```
<?xml version="1.0" encoding="UTF-8"?>
<Response xmlns:xw="http://www.kion.it/ns/xw"
          pageCount="358"
          pageIndex="1"
          seleId="530ca92062d5a38032434bc8"
          seleSize="3575"
          canNext="true"
          canLast="true"
          query="[/doc/@tipo]=arrivo">
  <doc physdoc="16108"
       nrecord="000016108-ADMNADM-d8834713-63ff-4198-b8c2-44813ef325cd"
       tipo="arrivo"
       anno="2013"
       cod_amm_aoo="ADMNADM"
       num_prot="2013-ADMNADM-0000009"
       data_prot="20130205"
       annullato="no">
    <oggetto xml:space="preserve">prova numero 2</oggetto>
    <classif xml:space="preserve" cod="01/01">01/01 - Normativa e relativa attuazione</classif>
    <rif_interni>
      <rif diritto="RPA" nome_persona="Amministratore Amministratore" nome_uff="Area Test"/>
    </rif_interni>
    <rif_esterni>
      <rif>
        <nome xml:space="preserve">Giuseppe Verdi</nome>
      </rif>
    </rif_esterni>
    <files>
      <xw:file name="U2yFRnXmsgzjqq3XDraJMw==_000001087-FS_FILES-231aad9a-589a-4412-a1bd-276219411f73.pdf"
title="regolaCodiceJava.pdf"/>
      <xw:file name="khR7k+Z91LgXrmbomg/m4Q==_000001088-FS_FILES-6d623bb7-6efd-4ef4-be2f-6d3bf046651a.docx"
title="Strumento per il fixing massivo.docx"/>
      <xw:file name="7WYbwb01KGC4V1Q1QWYoIA==_000001089-FS_FILES-8af60d4f-6445-4eb2-9a03-5df240b64cf5.pdf"
title="curriculum formativo(1).pdf"/>
      <xw:file name="6h8sPIxw7cPbdFhy00xN2w==_000001090-FS_FILES-40339b28-e2ec-4a5a-aab1-67393993fe6a.pdf"
title="curriculum formativo33.pdf"/>
    </files>
    <immagini>
      <xw:file name="bPv4AzCQf8s2bwkk/sZd/A==_000001091-FS_FILES-38542b5d-aa1c-49ad-a627-2bea9e30b518.jpg"
title="cool.jpg"/>
    </immagini>
  </doc>
  <doc ...>
    ...
  </doc>
  ...
</Response>
```

Il root element è sempre “Response” e il numero dei suoi attributi può variare a seconda del tipo di risposta. Nell'esempio riportato (risultato di una ricerca) figurano il numero di pagine (“pageCount”), l'indice della pagina corrente (“pageIndex”), l'id del result set (“seleId”), il numero di documenti trovati (“seleSize”) e la ricerca effettuata (“query”).

All'interno del root element c'è l'elenco dei documenti trovati, di cui vengono riportati i dati più significativi (dei file vengono mostrate solo le ultime versioni; per il resto si tratta di una versione semplificata dei documenti, senza modifiche alla struttura xml rispetto a quella che si ottiene caricandoli tramite i metodi *load*()*).

Il numero di documenti presenti nella Response varia a seconda del numero di documenti restituiti e del numero massimo di documenti che devono essere presenti in una pagina (con Titulus la paginazione dei risultati viene gestita dal servizio).

> Il codice xml dei documenti restituito nelle pagine di un result set consiste in una versione ridotta del documento vero e proprio. Quindi questo codice NON deve essere usato per modificare i documenti, pena la perdita di dati.

Quando, invece, viene restituito un singolo documento, la risposta xml ha la seguente struttura:

```
<?xml version="1.0" encoding="UTF-8"?>
<Response xmlns:xw="http://www.3di.it/ns/xw-200303121136"
          pageCount="1"
          pageIndex="1"
          seleId="3sed11f14cf760d801"
          seleSize="1"
          canSee="true"
          canEdit="true"
          canAddCC="true"
          canDelCC="true"
          canAddCDS="true"
          canDelCDS="true"
          canAddRPA="true"
          canAddPostIt="true"
          canAddFile="true"
          canAddImage="true"
          canInsInFolder="true"
          canLinkFolder="true"
          canCancel="true"
          canDelete="true">

  <Document physdoc="3892">
      ...
  </Document>
</Response>
```

Anche in questo caso il numero degli attributi del root element “Response” può variare a seconda del tipo di risposta. Nell'esempio riportato (caricamento di un documento trovato da una ricerca) figurano il numero di pagine (“pageCount”), l'indice della pagina corrente (“pageIndex”), l'id del result set (“seleId”) e il numero di documenti trovati (“seleSize”); segue, poi, un elenco di attributi “can*” che indica cosa può fare su quel documento l'utente usato dal client.

All'interno di “Response” c'è l'elemento “Document” con l'attributo “physdoc” (id univoco del documento).

Infine, come figlio di “Document” viene riportato il codice xml del documento vero e proprio. Per esempio

```
<?xml version="1.0" encoding="UTF-8"?>
<Response ...>
  <Document physdoc="...">
      <doc ...>
          ...
      </doc>
  </Document>
</Response>
```

per un documenti in arrivo, in partenza, tra uffici o non protocollato; oppure

```
<?xml version="1.0" encoding="UTF-8"?>
<Response ...>
  <Document physdoc="...">
      <fascicolo ...>
          ...
      </fascicolo>
  </Document>
</Response>
```

per un fascicolo.

Campi xml principali di un documento

> In questo paragrafo per “documento” si intende un documento in arrivo, in partenza, tra uffici o non protocollato.

Un documento Titulus possiede 4 tipi differenti di id univoci:

- **/doc/@physdoc**: id assegnato in automatico da Titulus e sempre presente;
- **/doc/@nrecord**: id assegnato in automatico da Titulus e sempre presente;
- **/doc/@num_prot**: numero di protocollo, presente solo nei documenti protocollati (cioè in arrivo, in partenza e tra uffici).
- **/doc/repertorio/@numero**: numero di repertorio, presente solo nei repertori.

Mediante questi id è possibile individuare univocamente il documento.

Di seguito si riportano i percorsi xml (xpath) dei principali campi di un documento:

XPath	Significato
/doc/@tipo	tipo (“arrivo” o “partenza” o “interno” o “varie”)
/doc/@cod_amm_aoo	codice AMMAOO
/doc/@anno	anno di inserimento (4 cifre)
/doc/@data_prot	data di protocollazione (nella forma “aaaammgg”)
/doc/@annullato	se vale “si”, significa che il documento è stato annullato
/doc/@bozza	se vale “si”, significa che il documento è una bozza
/doc/oggetto	oggetto
/doc/classif	classificazione
/doc/voce_indice	voce di indice
/doc/allegato	campo “Allegato” (*ripetibile*)
/doc/repertorio	contiene la descrizione del repertorio (*solo nei repertori*)
/doc/repertorio/@cod	contiene il codice del repertorio (*solo nei repertori*)
/doc/rif_esterni/rif	mittente o destinatario (*ripetibile*)
/doc/rif_interni/rif	riferimenti interni (RPA, CC…)
/doc/files	contiene i file allegati (non le immagini)
/doc/immagini	contiene le immagini allegate
/doc/minuta	contiene le informazioni sulla minuta (solo documenti tra uffici)
/doc/storia	contiene la storia degli interventi sul documento

Campi xml principali di un fascicolo

Un fascicolo Titulus possiede 3 tipi differenti di id univoci:

- **/fascicolo/@physdoc**: id assegnato in automatico da Titulus e sempre presente;
- **/fascicolo/@nrecord**: id assegnato in automatico da Titulus e sempre presente;
- **/fascicolo/@numero**: numero del fascicolo (sempre presente).

Mediante questi id è possibile individuare univocamente il fascicolo.

Di seguito si riportano i percorsi xml (xpath) dei principali campi di un fascicolo:

XPath	Significato
/fascicolo/@stato	stato (“aperto” o “chiuso”)
/fascicolo/@anno	anno di inserimento (4 cifre)
/fascicolo/@cod_amm_aoo	codice AMMAOO
/fascicolo/oggetto	oggetto
/fascicolo/classif	classificazione
/fascicolo/voce_indice	voce di indice
/fascicolo/rif_interni/rif	riferimenti interni (RPA e ITF)
/fascicolo/storia	contiene la storia degli interventi sul fascicolo

Campi xml principali di una seduta di Organi

Mentre le proposte e le comunicazioni di Titulus Organi sono normali documenti tra uffici e documenti non protocollati, le sedute sono documenti xml con un formato particolare.

Una seduta possiede 2 tipi differenti di id univoci:

- **/seduta/@physdoc**: id assegnato in automatico da Titulus e sempre presente;
- **/seduta/@nrecord**: id assegnato in automatico da Titulus e sempre presente;

Mediante questi id è possibile individuare univocamente la seduta.

Di seguito si riportano i percorsi xml (xpath) dei principali campi di una seduta:

XPath	Significato
/seduta/@stato	stato (“aperta”, “chiusa”, “alla firma” o “sospesa”)
/seduta/@anno	anno di inserimento (4 cifre)
/seduta/@cod_amm_aoo	codice AMMAOO
/seduta/@data_convocazione	data di convocazione (nella forma “aaaammgg”)
/seduta/@straordinaria	se “si”, indica una seduta straordinaria
/seduta/organo	descrizione dell'organo
/seduta/organo/@cod	codice dell'organo
/seduta/odg	contiene le comunicazioni e le proposte nell'ordine del giorno
/seduta/odg/proposta/@nrecord_prop	nrecord della comunicazione/proposta in odg
/seduta/odg/proposta/@cod_categoria	codice della categoria della comunicazione/proposta in odg
/seduta/odg/proposta/@tipo	tipo (“comunicazione” o “delibera”)
/seduta/odg/proposta/@numero_delibera	numero della delibera (per le proposte deliberate)
/seduta/odg/proposta/@risultato_seduta	risultato della seduta
/seduta/categorie	contiene le categorie delle comunicazioni/proposte
/seduta/componenti	contiene l'elenco dei componenti

/seduta/storia	contiene la storia degli interventi sulla seduta

Consultare la banca dati

Ricerca dei documenti

Una volta attivata la connessione, per consultare il database è sufficiente invocare il metodo search(). Tramite questo metodo è possibile ricercare un insieme di dati che rispondono a determinati criteri di ricerca espressi secondo un'opportuna sintassi . E' anche possibile effettuare una ricerca *full text* utilizzando il metodo searchFullText(), che a differenza di search() non accetta query ma una semplice stringa di testo che viene ricercata sia nei metadati che negli allegati.

Nella versione 4 di Titulus ogni information unit (documenti, fascicoli, ecc.) possiede un id univoco detto numero fisico, rappresentato dall'attributo *physdoc* del root element.
Le ricerche per numero fisico richiedono di usare la seguente sintassi:

```
[//@physdoc]=XXXX
```

dove XXXX rappresenta il numero fisico dell'information unit cercata.
Altre sintassi comportano una segnalazione di errore di campo inesistente.

Dashboard

La dashboard descrive un set di query di ricerca sui documenti in Titulus; ogni query, identificata da una label, raggruppa i documenti secondo un criterio logico prestabilito.

Grazie a questa funzionalità è possibile mettere a disposizione dei client interessati un insieme preconfigurato di ricerche.

La sua attivazione consiste nella stesura - concordata con chi sviluppa il client - di un file xml da inserire nella configurazione di Titulus.

Il recupero delle ricerche mediante le loro etichette può essere effettuato tramite il metodo **getDashboard()**.

Invocandolo senza indicare alcuna label, si ottiene l'intero insieme di ricerche:

```
<?xml version="1.0" encoding="UTF-8"?>
<tws_def_queries>
    <query label="L1">L1_query</query>
    <query label="L2">L2_query</query>
    ...
    <query label="Ln">Ln_query</query>
</tws_def_queries>
```

Invocandolo, invece, passando una label (es. “L1”), si ottiene la ricerca corrispondente:

```
<?xml version="1.0" encoding="UTF-8"?>
<tws_def_queries>
    <query label="L1">L1_query</query>
</tws_def_queries>
```

Se la label indicata non esiste, viene restituito un xml vuoto:

```
<?xml version="1.0" encoding="UTF-8"?>
<tws_def_queries>
</tws_def_queries>
```


Notare che al client vengono restituite delle ricerche. E' compito del client lanciarle mediante il metodo **search()**.

Di seguito si riportano alcuni esempi di query che è possibile inserire nella dashboard.

Documenti in cui una persona è in copia conoscenza:

```
([doc_rifinternirifdirittocodpersonascartato]=CC|COD_PERS|) AND NOT ([doc_rifinternirifnomeoper]="Gestione
Bozze") AND ([doc_annullato]="no") AND ([xml,/doc/storia/creazione/@data]={CREAZIONE_DA|CREAZIONE_A})
```

Documenti di una una persona è responsabile:

```
([doc_rifinternirifdirittocodpersonacodfasc]=RPA|COD_PERS|) AND NOT([doc_bozza]="si") AND ([doc_annullato]="
no") AND ([xml,/doc/storia/creazione/@data]={CREAZIONE_DA|CREAZIONE_A})
```

Bozze create da una persona:

```
([/doc/storia/creazione/@cod_oper/]="COD_PERS") AND ([doc_bozza]="si") AND NOT ([doc_personalviewcod]="CMZ" OR
"PRP")
```

Ecco cosa si ottiene tramite la chiamata *getDashboard()* (senza parametri):

```
<?xml version="1.0" encoding="UTF-8"?>
<tws_def_queries>
    <query label="docs_cc">([doc_rifinternirifdirittocodpersonascartato]=CC|COD_PERS|) AND NOT
([doc_rifinternirifnomeoper]="Gestione Bozze") AND ([doc_annullato]="no") AND ([xml,/doc/storia/creazione/@data]
={CREAZIONE_DA|CREAZIONE_A})</query>
    <query label="docs_rpa">([doc_rifinternirifdirittocodpersonacodfasc]=RPA|COD_PERS|) AND NOT([doc_bozza]="
si") AND ([doc_annullato]="no") AND ([xml,/doc/storia/creazione/@data]={CREAZIONE_DA|CREAZIONE_A})</query>
    <query label="docs_bozze">([/doc/storia/creazione/@cod_oper/]="COD_PERS") AND ([doc_bozza]="si") AND NOT
([doc_personalviewcod]="CMZ" OR "PRP")</query>
</tws_def_queries>
```

⚠ Nelle ricerche sopra riportate figurano le diciture “COD_PERS”, “CREAZIONE_DA” e “CREAZIONE_A” che il client deve sostituire con dei valori significativi prima di eseguirle:

```
COD_PERS = matricola della persona
```

```
CREAZIONE_DA = data di creazione da cui partire
```

```
CREAZIONE_A  = data di creazione a cui arrivare
```

Recupero documenti

L’esito positivo di una ricerca produce una “busta XML” (envelope) contenente un estratto dei primi N documenti rispondenti ai requisiti di ricerca.

Per sfogliare le pagine del result set si fa uso dei metodi firstTitlePage(), nextTitlePage(), prevTitlePage(), lastTitlePage(): ognuno di essi restituisce un envelope XML contenente un estratto di un insieme di N documenti, ove la dimensione N di una pagina è un parametro configurabile del metodo search().

Per navigare sui singoli documenti trovati si fa uso dei metodi loadFirstDocument(), loadNextDocument(), loadPrevDocument(), loadLastDocumento(): ognuno di essi restituisce un envelope XML contenente i metadati di registrazione di un documento.

È prevista anche una modalità di caricamento diretto di un documento, sulla base di un suo identificativo univoco (physdoc o nrecord): loadDocument(id, lock).

URL di un documento

Il metodo getDocumentURL() consente di ottenere un URL di accesso ad un documento. E' possibile passare al metodo un qualunque identificatore tra:

- nrecord
- numero di protocollo
- numero di repertorio

Se viene passato un identificatore diverso da quelli elencati, viene sollevata una eccezione. Se viene passato un identificatore formalmente corretto, ma che non corrisponde ad alcun documento, viene reso comunque un url valido, ma che aprirà una pagina di errore *Documento non trovato* in Titulus.

L'URL del documento viene allegato, grazie a questo metodo, alla Response dalle seguenti funzioni:

- saveDocument()
- saveModifiedDocument()

in questo modo:

```
<?xml version="1.0" encoding="UTF-8"?>
<Response ...>
<url>http:...</url>
  <Document physdoc="...">
      ...
  </Document>
</Response>
```

URL di un fascicolo

Il metodo getFolderURL() consente di ottenere un URL di accesso ad un fascicolo. E' possibile passare al metodo un qualunque identificatore tra:

- nrecord
- numero di fascicolo
- numero di physodc

Se viene passato un identificatore diverso da quelli elencati, viene sollevata una eccezione. Se viene passato un identificatore formalmente corretto, ma che non corrisponde ad alcun documento, viene reso comunque un url valido, ma che aprirà una pagina di errore *Documento non trovato* in Titulus.

L'URL del documento viene allegato, grazie a questo metodo, alla Response dalle seguenti funzioni:

- newFolder()
- newSpecialFolder()
- modifyFolder()
- newSubFolder()

in questo modo:

```
<?xml version="1.0" encoding="UTF-8"?>
<Response ...>
<url>http:...</url>
  <Document physdoc="...">
      <fascicolo ...>
          ...
      </fascicolo>
  </Document>
</Response>
```

Download file associati ed immagini

Ad ogni file (allegato multimediale) associato al metadato XML di un documento viene attribuito un id univoco di archivio. Il percorso XPath per recuperare tale identificativo nell'ambito del metadato XML è:

```
/doc/files/xw:file/@name    (per i file)
/doc/immagini/xw:file/@name (per le immagini)
```

All'interno di /doc/files e di /doc/immagini possono essere incapsulati 0 o più riferimenti ad allegati.

getAttachment()

Per richiedere il download di uno di questi file si invoca il metodo **getAttachment()** (per i dettagli si rimanda al javadoc).

Di seguito si riporta un esempio in linguaggio Java, realizzato con la libreria Axis. Lo stub client usato (nel codice "*this.titulus4*"), è stato generato a partire dal wsdl del servizio, mediante il tool WSDL2Java.

```
import org.dom4j.Document;
import org.dom4j.DocumentHelper;
import it.kion.titulus.ws.client_stub.Titulus4.AttachmentBean;

...

...

// loading first incoming document
this.titulus4.search("[/doc/@tipo]=arrivo", null, null);

String   resp    = this.titulus4.loadFirstDocument();
Document respDoc = DocumentHelper.parseText(resp);

// checking attachment
List files = respDoc.selectNodes("//doc/files/*[name()='xw:file'][not(@der_from)]");

for (int j = 0; j < files.size(); j++) {
    Element file   = (Element)files.get(j);
    String  fileId = file.attributeValue("name", "");

    // downloading file
    AttachmentBean attachmentBean = this.titulus4.getAttachment(fileId);

    byte[] attachment = attachmentBean.getContent();
    ...
}
```

Di seguito si riporta la struttura della risposta SOAP inviata al client (l'array di byte viene codificato come *xsi:type="soapenc:base64Binary"*):

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:xsd="http://www.w3.org/2001
/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <soapenv:Body>
      <ns1:getAttachmentResponse soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" xmlns:ns1="
http://ws.titulus.kion.it">
         <getAttachmentReturn href="#id0"/>
      </ns1:getAttachmentResponse>
      <multiRef id="id0" soapenc:root="0" soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" xsi:
type="ns2:AttachmentBean" xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/" xmlns:ns2="http://www.kion.
it/titulus">
         <content xsi:type="soapenc:base64Binary">JVBERi0...</content>
         <description xsi:type="soapenc:string" xsi:nil="true"/>
         <id xsi:type="soapenc:string">U2yFRnXmsgzjqq3XDraJMw==_000001087-FS_FILES-231aad9a-589a-4412-a1bd-
276219411f73.pdf</id>
         <mimeType xsi:type="soapenc:string">application/pdf</mimeType>
      </multiRef>
   </soapenv:Body>
</soapenv:Envelope>
```

Notare che del file viene restituito anche il mime type.

Consultazione delle voci di indice

Le voci di indice di Titulus possono essere consultate tramite i metodi search() e loadDocument*().

Di seguito si riportano alcuni esempi di ricerche di voci di indice.

Ricerca che trova tutte le voci di indice:

```
([UD,/xw/@UdType/]="indice_titolario")
```

Ricerca che trova tutte le voci di indice la cui descrizione contiene la parola *esse3* (ricerca full text sulla descrizione della voce di indice):

```
[/indice_titolario/@voce]=esse3

oppure

[tit_voce]=esse3
```

Ricerca che trova la voce di indice “Informativa giuridica - Diffusione materiale informativo e interpretativo su normativa, leggi e circolari” (ricerca esatta sulla descrizione della voce di indice):

```
[tit_voce_exact]="Informativa giuridica - Diffusione materiale informativo e interpretativo su normativa, leggi e circolari"
```

Di seguito si riportano i campi più significativi da usare nelle ricerche delle voci di indice.

campo	alias	significato
/indice_titolario/@voce	tit_voce	descrizione
/indice_titolario/@cod_amm_aoo	-	codice AMMAOO
/indice_titolario/compilazione_automatica/classif	-	descrizione della classificazione
/indice_titolario/compilazione_automatica/classif/@cod	-	codice della classificazione - es. 01/03
/indice_titolario/validita/@tipodoc	tit_validitatipodoc	tipologie dei documenti per cui è valida la voce di indice

Registrare un documento

La registrazione di un nuovo documento è gestita attraverso la chiamata al metodo saveDocument(), che ha come primo parametro il documento (i suoi metadati) in formato XML.

Il nuovo documento prima di essere salvato subisce un processo di validazione per mezzo di un XSD (XML Schema Definition) e successivamente un controllo semantico per verificarne la consistenza in funzione della tipologia indicata.

Per lo schema completo del documento xml si faccia riferimento al javadoc del metodo saveDocument().

Di seguito si riporta un esempio di inserimento in linguaggio Java, realizzato con la libreria Axis. Lo stub client usato (nel codice “*this.titulus4*”), è stato generato a partire dal wsdl del servizio, mediante il tool WSDL2Java.

```
import it.kion.titulus.ws.client_stub.Titulus4.SaveParams;

...

// *** composizione del nuovo documento (stringa XML) ***

...

// inserimento del nuovo documento con invio delle notifiche email ai suoi assegnatari
SaveParams params = new SaveParams();

params.setSendMail(true);

String savedDocumentStr = this.titulus4.saveDocument(newDoc, null, params);

log.debug("saved document is:\r\n\n" + savedDocumentStr + "\n");

...
```

Esempio di nuovo documento non protocollato

Di seguito si riporta un esempio di codice XML riferito alla richiesta di registrazione di un documento non protocollato (tipo “varie”):

```
<?xml version="1.0" encoding="UTF-8"?>

<doc tipo="varie">
  <repertorio cod="NOT">Atti di notifica</repertorio>
  <autore>Dante Alighieri</autore>
  <oggetto>Invio bozza opera La Divina Commedia</oggetto>
  <tipologia cod="Dichiarazione"/>
  <mezzo_trasmissione costo="10" valuta="euro" cod="Raccomandata"></mezzo_trasmissione>
  <note>Note al documento</note>
  <riferimenti xml:space="preserve">Volumi Inferno, Purgatorio, Paradiso</riferimenti>
  <xlink href="http://www.link.it" xml:space="preserve">descrizione link</xlink>
  <allegato>0 - nessun allegato</allegato>
  <voce_indice xml:space="preserve">Letteratura</voce_indice>
</doc>
```

L’elemento “repertorio” non è obbligatorio; tuttavia nel caso di registrazione di un documento appartenente ad un repertorio, deve essere indicato.

Ecco la risposta restituita dal metodo saveDocument():

```
<?xml version="1.0" encoding="UTF-8"?>

<Response xmlns:xw="http://www.kion.it/ns/xw"
          canSee="true"
          canEdit="true"
          canAddCC="true"
          canDelCC="true"
          canAddCDS="true"
          canDelCDS="true"
          canAddRPA="true"
          canAddPostIt="true"
          canAddFile="true"
          canAddImage="true"
          canInsInFolder="true"
          canLinkFolder="true"
          canCancel="true"
          canDelete="true">

  <Document physdoc="19643">
    <doc anno="2011" annullato="no" cod_amm_aoo="KIONCLE" data_prot="20110721" physdoc="19643" nrecord="
000019643-KIONCLE-e20f27d6-bf98-4b1a-a28d-654a21e05ada" scarto="10" tipo="varie">
      <repertorio cod="NOT" numero="NOT^KIONCLE-20110000006">Atti di notifica</repertorio>
      <autore xml:space="preserve">Dante Alighieri</autore>
      <oggetto xml:space="preserve">Invio bozza opera La Divina Commedia</oggetto>
      <tipologia cod="Dichiarazione"/>
      <mezzo_trasmissione cod="Raccomandata" costo="10" valuta="euro"/>
      <classif cod="01/01" xml:space="preserve">01/01 - Letteratura</classif>
      <note xml:space="preserve">Note al documento</note>
      <riferimenti xml:space="preserve">Volumi Inferno, Purgatorio, Paradiso</riferimenti>
      <xlink href="http://www.link.it" xml:space="preserve">descrizione link</xlink>
      <allegato xml:space="preserve">0 - nessun allegato</allegato>
      <voce_indice xml:space="preserve">Letteratura</voce_indice>
      <rif_interni>
        <rif diritto="RPA" cod_persona="11111" cod_uff="SI000006" nome_persona="Rossi Mario" nome_uff="
Controllo posta"/>
        <rif diritto="CC" cod_persona="3" cod_uff="00003" nome_persona="Verdi Luca" nome_uff="Smistamenti"/>
      </rif_interni>
      <storia>
        <creazione cod_oper="PI000121" cod_uff_oper="SI000084" data="20110721" oper="Applicazione Client WS"
ora="11:41:46" uff_oper="Ufficio Protocollo KION"/>
        <responsabilita cod_operatore="PI000121" cod_persona="11111" cod_uff="SI000006"  nome_persona="Rossi
Mario" nome_uff="Controllo posta" data="20110721" operatore="Applicazione Client WS (Ufficio Protocollo KION)"
ora="11:41:46"/>
        <assegnazione_cc cod_operatore="PI000121" cod_persona="3" cod_uff="00003" data="20110721" nome_persona="
Verdi Luca" nome_uff="Smistamenti" operatore="Applicazione Client WS (Ufficio Protocollo KION)" ora="11:41:46"/>
      </storia>
    </doc>
  </Document>
</Response>
```

Esempio di nuovo documento in arrivo

Di seguito si riporta un esempio di codice XML riferito alla richiesta di registrazione di un documento in arrivo (tipo “arrivo”):

```
<?xml version="1.0" encoding="UTF-8"?>

<doc tipo="arrivo">
  <prot_differito data_arrivo="20110709" xml:space="preserve">
      Data conclusione presentazione domande. Il carico di lavoro ha superato le capacità
      operative dell'ufficio.
  </prot_differito>
  <repertorio cod="V_D_CDA">Verbali Consiglio di Amministrazione</repertorio>
  <oggetto>Richiesta documentazione protocollo informatico</oggetto>
  <tipologia cod="Dichiarazione"/>
  <mezzo_trasmissione costo="23" valuta="euro" cod="Posta Ordinaria"></mezzo_trasmissione>
  <note>Note al documento</note>
  <riferimenti xml:space="preserve">CNIPA - Protocollo Informatico</riferimenti>
  <xlink href="http://www.link.it" xml:space="preserve">descrizione link</xlink>
  <allegato>0 - nessun allegato</allegato>
  <voce_indice xml:space="preserve">Documentazione</voce_indice>
  <rif_esterni>
    <rif_esterno data_prot="20040101" n_prot="1243">
      <nome cod="SE000095" xml:space="preserve">Ditta srl</nome>
      <referente cod="PE000928" nominativo="Mario Rossi"/>
      <indirizzo email="mrossi@ditta.it"
                 fax="051 451242"
                 tel="051 452844"
                 xml:space="preserve">
         Via Manzoni 2 - 40033 Casalecchio di Reno - BO - Italia
      </indirizzo>
    </rif_esterno>
  </rif_esterni>
</doc>
```

L’elemento “repertorio” non è obbligatorio, tuttavia nel caso di registrazione di un documento appartenente ad un repertorio, deve essere indicato.

Ecco la risposta restituita dal metodo saveDocument():

```
<?xml version="1.0" encoding="UTF-8"?>

<Response xmlns:xw="http://www.kion.it/ns/xw"
          canSee="true"
          canEdit="true"
          canAddCC="true"
          canDelCC="true"
          canAddRPA="true"
          canAddPostIt="true"
          canAddFile="true"
          canAddImage="true"
          canInsInFolder="true"
          canLinkFolder="true"
          canCancel="true">

  <Document physdoc="19644">
    <doc anno="2011" annullato="no" cod_amm_aoo="KIONCLE" data_prot="20110721" physdoc="19644" nrecord="
000019644-KIONCLE-ff424d72-5afe-41ce-9f5a-a10adc7f3bae" num_prot="2011-KIONCLE-0000128" scarto="01" tipo="
arrivo">
      <prot_differito data_arrivo="20110709" xml:space="preserve">
        Data conclusione presentazione domande. Il carico di lavoro ha superato le capacità
        operative dell'ufficio.
      </prot_differito>
      <repertorio cod="V_D_CDA" numero="V_D_CDA^KIONCLE-20110000047">Verbali Consiglio di Amministrazione<
/repertorio>
      <oggetto xml:space="preserve">Richiesta documentazione protocollo informatico</oggetto>
      <tipologia cod="Dichiarazione"/>
      <mezzo_trasmissione cod="Posta Ordinaria" costo="23" valuta="euro"/>
      <classif cod="06/04" xml:space="preserve">06/04 - Documentazione</classif>
      <note xml:space="preserve">Note al documento</note>
      <riferimenti xml:space="preserve">CNIPA - Protocollo Informatico</riferimenti>
      <xlink href="http://www.link.it" xml:space="preserve">descrizione link</xlink>
      <allegato xml:space="preserve">0 - nessun allegato</allegato>
      <voce_indice xml:space="preserve">Documentazione</voce_indice>
      <rif_esterni>
        <rif data_prot="20040101" n_prot="1243">
          <nome cod="SE000095" xml:space="preserve">Ditta srl</nome>
          <referente cod="PE000928" nominativo="Mario Rossi"/>
          <indirizzo email="mrossi@ditta.it"
                     fax="051 451242"
                     tel="051 452844"
                     xml:space="preserve">
            Via Manzoni 2 - 40033 Casalecchio di Reno - BO - Italia
          </indirizzo>
        </rif>
      </rif_esterni>
      <rif_interni>
        <rif diritto="RPA" cod_persona="11111" cod_uff="SI000006" nome_persona="Rossi Mario" nome_uff="
Controllo posta"/>
        <rif diritto="CC" cod_persona="3" cod_uff="00003" nome_persona="Verdi Luca" nome_uff="Smistamenti"/>
      </rif_interni>
      <storia>
        <creazione cod_oper="PI000121" cod_uff_oper="SI000084" data="20110721" oper="Applicazione Client WS"
ora="11:41:46" uff_oper="Ufficio Protocollo KION"/>
        <responsabilita cod_operatore="PI000121" cod_persona="11111" cod_uff="SI000006"  nome_persona="Rossi
Mario" nome_uff="Controllo posta" data="20110721" operatore="Applicazione Client WS (Ufficio Protocollo KION)"
ora="11:41:46"/>
        <assegnazione_cc cod_operatore="PI000121" cod_persona="3" cod_uff="00003" data="20110721" nome_persona="
Verdi Luca" nome_uff="Smistamenti" operatore="Applicazione Client WS (Ufficio Protocollo KION)" ora="11:41:46"/>
      </storia>
    </doc>
  </Document>
</Response>
```

Esempio di nuovo documento in partenza

Di seguito si riporta un esempio di codice XML riferito alla richiesta di registrazione di un documento in partenza (tipo “partenza”):

```
<?xml version="1.0" encoding="iso-8859-1"?>

<doc tipo="partenza">
  <repertorio cod="V_D_CDA">Verbali Consiglio di Amministrazione</repertorio>
  <oggetto>Richiesta documentazione protocollo informatico</oggetto>
  <tipologia cod="Dichiarazione"/>
  <mezzo_trasmissione costo="58" valuta="euro" cod="Posta Ordinaria"/>
  <note>Note al documento</note>
  <riferimenti xml:space="preserve">CNIPA - Protocollo Informatico</riferimenti>
  <xlink href="http://www.link.it" xml:space="preserve">descrizione link</xlink>
  <allegato>0 - nessun allegato</allegato>
  <voce_indice xml:space="preserve">Documentazione</voce_indice>
  <rif_esterni>
    <rif_esterno>
      <nome cod="SE000095" xml:space="preserve">Ditta srl</nome>
      <indirizzo xml:space="preserve">
        Via Manzoni, 2 - 40033 Casalecchio di Reno (BO) - Italia
      </indirizzo>
      <referente cod="PE000928" nominativo="Mario Rossi"/>
    </rif_esterno>
    <rif_esterno copia_conoscenza="si">
      <nome cod="SE000095" xml:space="preserve">Ditta srl</nome>
      <indirizzo xml:space="preserve">
        Via Manzoni, 2 - 40033 Casalecchio di Reno (BO) - Italia
      </indirizzo>
      <referente cod="PE000929" nominativo="Ermete Verdi"/>
    </rif_esterno>
  </rif_esterni>
</doc>
```

L’elemento “repertorio” non è obbligatorio, tuttavia nel caso di registrazione di un documento appartenente ad un repertorio, deve essere indicato.

Ecco la risposta restituita dal metodo saveDocument():

```
<?xml version="1.0" encoding="UTF-8"?>

<Response xmlns:xw="http://www.kion.it/ns/xw"
          canSee="true"
          canEdit="true"
          canAddCC="true"
          canDelCC="true"
          canAddRPA="true"
          canAddPostIt="true"
          canAddFile="true"
          canAddImage="true"
          canInsInFolder="true"
          canLinkFolder="true"
          canCancel="true">

  <Document physdoc="19674">
    <doc anno="2011" annullato="no" cod_amm_aoo="KIONCLE" data_prot="20110721" physdoc="19674" nrecord="
000019674-KIONCLE-7fc03c5f-ce2a-4fe5-8c57-6c219751f379" num_prot="2011-KIONCLE-0000130" scarto="01" tipo="
partenza">
      <repertorio cod="V_D_CDA" numero="V_D_CDA^KIONCLE-20110000047">Verbali Consiglio di Amministrazione<
/repertorio>
      <oggetto xml:space="preserve">Richiesta documentazione protocollo informatico</oggetto>
      <tipologia cod="Dichiarazione"/>
      <mezzo_trasmissione cod="Posta Ordinaria" costo="58" valuta="euro"/>
      <classif cod="06/04" xml:space="preserve">06/04 - Documentazione</classif>
      <note xml:space="preserve">Note al documento</note>
      <riferimenti xml:space="preserve">CNIPA - Protocollo Informatico</riferimenti>
      <xlink href="http://www.link.it" xml:space="preserve">descrizione link</xlink>
      <allegato xml:space="preserve">0 - nessun allegato</allegato>
      <voce_indice xml:space="preserve">Documentazione</voce_indice>
      <rif_esterni>
        <rif>
          <nome cod="SE000095" xml:space="preserve">Ditta srl</nome>
          <referente cod="PE000928" nominativo="Mario Rossi"/>
          <indirizzo email="mrossi@ditta.it"
                     fax="051 451242"
                     tel="051 452844"
                     xml:space="preserve">
            Via Manzoni 2 - 40033 Casalecchio di Reno - BO - Italia
          </indirizzo>
        </rif>
        <rif copia_conoscenza="si">
          <nome cod="SE000095" xml:space="preserve">Ditta srl</nome>
          <referente cod="PE000929" nominativo="Ermete Verdi"/>
          <indirizzo xml:space="preserve">
            Via Manzoni, 2 - 40033 Casalecchio di Reno (BO) - Italia
          </indirizzo>
        </rif>
      </rif_esterni>
      <rif_interni>
        <rif diritto="RPA" cod_persona="11111" cod_uff="SI000006" nome_persona="Rossi Mario" nome_uff="
Controllo posta"/>
        <rif diritto="CC" cod_persona="3" cod_uff="00003" nome_persona="Verdi Luca" nome_uff="Smistamenti"/>
      </rif_interni>
      <storia>
        <creazione cod_oper="PI000121" cod_uff_oper="SI000084" data="20110721" oper="Applicazione Client WS"
ora="11:41:46" uff_oper="Ufficio Protocollo KION"/>
        <responsabilita cod_operatore="PI000121" cod_persona="11111" cod_uff="SI000006"  nome_persona="Rossi
Mario" nome_uff="Controllo posta" data="20110721" operatore="Applicazione Client WS (Ufficio Protocollo KION)"
ora="11:41:46"/>
        <assegnazione_cc cod_operatore="PI000121" cod_persona="3" cod_uff="00003" data="20110721" nome_persona="
Verdi Luca" nome_uff="Smistamenti" operatore="Applicazione Client WS (Ufficio Protocollo KION)" ora="11:41:46"/>
      </storia>
    </doc>
  </Document>
</Response>
```

Esempio di nuovo documento tra uffici

Di seguito si riporta un esempio di codice XML riferito alla richiesta di registrazione di un documento tra uffici (tipo “interno”):

```
<?xml version="1.0" encoding="UTF-8"?>
<doc tipo="interno">
  <repertorio cod="REPI">Repertorio interno</repertorio>
  <minuta scarto="99">
      <classif cod="03/05" xml:space="preserve">03/05 - Assistenza telefonica</classif>
      <mittente nome_persona="Marrone Antonio"
                nome_uff="Archivio"
                cod_persona="PI000122"
                cod_uff="SI000085"/>
  </minuta>
  <oggetto>Richiesta documentazione protocollo informatico</oggetto>
  <tipologia cod="Dichiarazione"/>
  <mezzo_trasmissione costo="5" valuta="euro" cod="Fax"></mezzo_trasmissione>
  <note>Note al documento</note>
  <riferimenti xml:space="preserve">CNIPA - Protocollo Informatico</riferimenti>
  <xlink href="http://www.link.it" xml:space="preserve">descrizione link</xlink>
  <allegato>0 - nessun allegato</allegato>
  <voce_indice xml:space="preserve">Documentazione</voce_indice>
</doc>
```

L’elemento “repertorio” non è obbligatorio, tuttavia nel caso di registrazione di un documento appartenente ad un repertorio, deve essere indicato.

L’elemento “minuta” è obbligatorio.

Ecco la risposta restituita dal metodo saveDocument():

```
<?xml version="1.0" encoding="UTF-8"?>

<Response xmlns:xw="http://www.kion.it/ns/xw"
          canSee="true"
          canEdit="true"
          canEditCopy="true"
          canAddCC="true"
          canDelCC="true"
          canAddCDS="true"
          canDelCDS="true"
          canAddCDSM="true"
          canDelCDSM="true"
          canAddRPA="true"
          canAddPostIt="true"
          canAddFile="true"
          canAddImage="true"
          canInsInFolder="true"
          canLinkFolder="true"
          canInsCopyInFolder="true"
          canCancel="true">

  <Document physdoc="19660">
    <doc anno="2011" annullato="no" cod_amm_aoo="KIONCLE" data_prot="20110721" physdoc="19660" nrecord="
000019660-KIONCLE-886e681a-78c0-4ef1-abe8-6a05e3ee9630" num_prot="2011-KIONCLE-0000144" scarto="01" tipo="
interno">
      <minuta scarto="99">
        <classif cod="03/05" xml:space="preserve">03/05 - Assistenza telefonica</classif>
        <mittente nome_persona="Marrone Antonio" nome_uff="Archivio"/>
      </minuta>
      <oggetto xml:space="preserve">Richiesta documentazione protocollo informatico</oggetto>
      <tipologia cod="Dichiarazione"/>
      <mezzo_trasmissione cod="Fax" costo="5" valuta="euro"/>
      <classif cod="06/04" xml:space="preserve">06/04 - Documentazione</classif>
      <note xml:space="preserve">Note al documento</note>
      <riferimenti xml:space="preserve">CNIPA - Protocollo Informatico</riferimenti>
      <xlink href="http://www.link.it" xml:space="preserve">descrizione link</xlink>
      <allegato xml:space="preserve">0 - nessun allegato</allegato>
      <voce_indice xml:space="preserve">Documentazione</voce_indice>
      <rif_interni>
        <rif diritto="RPA" cod_persona="11111" cod_uff="SI000006" nome_persona="Rossi Mario" nome_uff="
Controllo posta"/>
        <rif diritto="CC" cod_persona="3" cod_uff="00003" nome_persona="Verdi Luca" nome_uff="Smistamenti"/>
        <rif cod_persona="PI000122" cod_uff="SI000085" diritto="RPAM" nome_persona="Marrone Antonio" nome_uff="
Archivio"/>
      </rif_interni>
      <storia>
        <creazione cod_oper="PI000121" cod_uff_oper="SI000084" data="20110721" oper="Applicazione Client WS"
ora="11:41:46" uff_oper="Ufficio Protocollo KION"/>
        <responsabilita cod_operatore="PI000121" cod_persona="11111" cod_uff="SI000006"  nome_persona="Rossi
Mario" nome_uff="Controllo posta" data="20110721" operatore="Applicazione Client WS (Ufficio Protocollo KION)"
ora="11:41:46"/>
        <responsabilita_minuta cod_operatore="PI000121" cod_persona="PI000122" cod_uff="SI000085" data="
20110721" data_visto="20110721" nome_persona="Marrone Antonio" nome_uff="Archivio" operatore="Applicazione
Client WS (Ufficio Protocollo KION)" ora="11:41:46" ora_visto="11:41:46" visto_da="Marrone Antonio (Archivio)"/>
        <assegnazione_cc cod_operatore="PI000121" cod_persona="3" cod_uff="00003" data="20110721" nome_persona="
Verdi Luca" nome_uff="Smistamenti" operatore="Applicazione Client WS (Ufficio Protocollo KION)" ora="11:41:46"/>
      </storia>
    </doc>
```

Osservazioni

- L'ordine degli elementi xml deve essere quello indicato dallo schema.
- E' consigliato l'uso delle *voci di indice* per l'impostazione della classificazione e degli smistamenti dei documenti (come mostrato negli esempi precedenti).
  Infatti questo approccio consente di evitare che i client siano consapevoli della classificazione e dei riferimenti interni che devono essere assegnati ai documenti da inserire, il che rende più agevole lo sviluppo e la configurazione del sistema client/server.

A causa di alcune limitazioni, attualmente non è possibile usare le voci di indice per indicare anche il mittente (RPA della minuta) di un documento tra uffici.

- Il campo “allegato” è ripetibile ed è svincolato dal numero di file/immagini del documento. Se non specificato, viene automaticamente valorizzato con “0 - nessun allegato”.
- *Dalla versione **3.11.0.7**:* Inserendo un repertorio, è necessario indicare il suo codice. La descrizione eventualmente passata (per compatibilità con le versioni precedenti) verrà ignorata nel salvataggio e verrà sostituita con la descrizione configurata in Titulus per il repertorio. Se il repertorio non esiste, è disattivato o non si hanno i i diritti di inserimento, vengono riportati degli errori specifici.
- *Versioni precedenti alla **3.11.0.7**:* Inserendo un repertorio, è necessario indicare il suo codice e la sua descrizione.
- I riferimenti esterni (mittente e destinatari) possono essere privi di codice. Qualora lo si indichi, esso deve essere un codice registrato nell’ anagrafica di Titulus.
- Nella risposta restituita dal metodo saveDocument(), gli attributi /Response/can* si riferiscono a cosa può fare l'utente usato dal client.
- A partire dalla *versione 3.12.2.0* di Titulus, per l'inserimento dei documenti la dtd è stata sostituita da un xml schema. Questo schema è disponibile per il download nel javadoc del metodo *Titulus.saveDocument()*.
- In ogni tipologia di documento è prevista la possibilità di inserire dei campi personalizzati. Questi campi devono essere inseriti come figli dell'elemento “/doc/extra”.
  Notare che, trattandosi di campi decisi dal cliente, tutto ciò che viene inserito nell'elemento “extra” non viene automaticamente visualizzato in Titulus. Per gestire questo contenuto in Titulus, infatti, è necessario realizzare delle viste personalizzate.
  Di seguito due note sulla validazione dell'elemento “extra”:
  - *release precedenti alla 3.12.2.0*: l'elemento “extra” viene definito nella dtd di inserimento come un elemento di tipo “ANY”. Questo significa che esso può contenere qualsiasi cosa, ma in ogni caso il suo contenuto deve essere dichiarato in una dtd proprietaria, cioè nel file “proprietario.dtd”. Dato che questo file risiede nell'installazione dei servizi, per poterlo aggiornare è necessario comunicare la dtd dei campi custom a chi si occupa dell'aggiornamento del software, cioè al supporto Titulus (supporto_titulus@kion.it). Se non si compila il file “proprietario.dtd”, l'inserimento dei documenti con campo “extra” valorizzato fallisce.
  - *release dalla 3.12.2.0 in avanti*: il contenuto dell'elemento “extra” non viene più validato dai servizi, quindi ogni client è libero di riempirlo senza preoccuparsi di definire una sua dtd. La sua validazione, quindi, è a carico del client.

Inserimento di un documento con file allegati

Con il metodo **saveDocument()** è possibile inserire un documento facendo l'upload dei file allegati in un'unica operazione (per i dettagli si rimanda al javadoc).

Il contenuto binario dei file da allegare e il nome del file devono essere inseriti valorizzando il bean AttachmentBean.

Di seguito si riporta un esempio in linguaggio Java, realizzato con la libreria Axis. Lo stub client usato (nel codice “*this.titulus4*”), è stato generato a partire dal wsdl del servizio, mediante il tool WSDL2Java.

Nell'esempio viene inserito un documento con 2 allegati e di questi viene richiesta la conversione in pdf.

```
import it.kion.titulus.ws.client_stub.Titulus4.AttachmentBean;
import it.kion.titulus.ws.client_stub.Titulus4.SaveParams;
...

...

// *** composizione del nuovo documento (stringa XML) ***

...

// *** aggiunta degli allegati al documento ***

AttachmentBean attachment1 = new AttachmentBean();
attachment1.setFileName("man_acl.pdf");
attachment1.setDescription("Manuale");
attachment1.setMimeType("application/pdf");
attachment1.setContent(getBytesFromFile(new File("man_acl.pdf")));

AttachmentBean attachment2 = new AttachmentBean();
attachment2.setFileName("documento.txt");
attachment2.setDescription("Questo è un testo");
attachment2.setMimeType("text/plain");
attachment2.setContent(getBytesFromFile(new File("documento.txt")));

AttachmentBean[] array = new AttachmentBean[]{attachment1, attachment2};

SaveParams params = new SaveParams();

params.setPdfConversion(true);

String resp = this.titulus4.saveDocument(newDoc, array, params);

log.debug("document now is:\r\n\n" + resp + "\n");

...
```

Gestione dei file allegati ai documenti

Aggiunta di allegati ad un documento

Per aggiungere degli allegati ad un documento che ne è privo, occorre invocare il metodo **checkInContentFiles()** (per i dettagli si rimanda al javadoc).

Di seguito si riporta un esempio in linguaggio Java, realizzato con la libreria Axis. Lo stub client usato (nel codice "*this.titulus4*"), è stato generato a partire dal wsdl del servizio, mediante il tool WSDL2Java.

Nell'esempio vengono inseriti 2 allegati in un documento e di questi viene richiesta la conversione in pdf.

```
import it.kion.titulus.ws.client_stub.Titulus4.AttachmentBean;
import it.kion.titulus.ws.client_stub.Titulus4.SaveParams;
...

...

// *** aggiunta degli allegati al documento con id "31" ***

AttachmentBean attachment1 = new AttachmentBean();
attachment1.setFileName("man_acl.pdf");
attachment1.setDescription("Manuale");
attachment1.setMimeType("application/pdf");
attachment1.setContent(getBytesFromFile(new File("man_acl.pdf")));

AttachmentBean attachment2 = new AttachmentBean();
attachment2.setFileName("documento.txt");
attachment2.setDescription("Questo è un testo");
attachment2.setMimeType("text/plain");
attachment2.setContent(getBytesFromFile(new File("documento.txt")));

AttachmentBean[] array = new AttachmentBean[]{attachment1, attachment2};

SaveParams params = new SaveParams();

params.setPdfConversion(true);

AttachmentBean[] respAttach = this.titulus4.checkInContentFiles("31", array, params);
String resp = this.titulus4.loadDocument("31", false);

log.debug("document now is:\r\n\n" + resp + "\n");
```

Versioning degli allegati

Per i file associati ad un documento è attivo il versioning: gli operatori autorizzati possono richiedere il check-out del file da aggiornare, per poi introdurre la nuova versione del file con un'operazione di check-in. Questa opzione è disponibile solo per documenti non protocollati o per bozze di documenti protocollati. Inoltre, il versioning non è possibile per le immagini.

Ogni versione di file viene incapsulata come elemento figlio dell'elemento (xw:file) corrispondente alla versione precedente:

```
<files>
    <xw:file convert="remove" name="XXX.txt" title="test_versions.txt">
      <chkin cod_operatore="PI000056" data="20101028" operatore="TEST WS" ora="16:26:20"/>
      <chkout cod_operatore="PI000056" data="20101028" operatore="TEST WS" ora="16:26:21"/>
      <xw:file convert="remove" name="YYY.txt" title="test_versions1.txt">
        <chkin cod_operatore="PI000056" data="20101028" operatore="TEST WS" ora="16:26:21"/>
        <chkout cod_operatore="PI000056" data="20101028" operatore="TEST WS" ora="16:26:21"/>

        ...

        <xw:file convert="remove" name="ZZZ.txt" title="test_versionsN.txt">
          <chkin cod_operatore="PI000056" data="20101028" operatore="TEST WS" ora="16:26:22"/>
        </xw:file>
      </xw:file>
    </xw:file>
</files>
```

I metodi messi a disposizione per il versioning sono:

- **checkOutContentFile()**: per prenotare un file da modificare
- **unlockContentFile()**: per sbloccare un file precedentemente prenotato
- **checkInContentFiles()**: per aggiornare un file precedentemente prenotato.

Il numero massimo di versioni di un file accettate è **90** (nota: è possibile configurare Titulus in modo tale che accetti un numero massimo di versioni inferiore). Il superamento di questa soglia comporta un'eccezione in fase di check-in da intercettare lato client.

Di seguito si riporta un esempio in linguaggio Java, realizzato con la libreria Axis. Lo stub client usato (nel codice "*this.titulus4*"), è stato generato a partire dal wsdl del servizio, mediante il tool WSDL2Java.

Nell'esempio viene aggiunta una nuova versione dell'allegato con id “*attachID*” nel documento “31”, e di questa viene richiesta la conversione in pdf.

```
import it.kion.titulus.ws.client_stub.Titulus4.AttachmentBean;
import it.kion.titulus.ws.client_stub.Titulus4.SaveParams;
...

...

// *** aggiunta di una nuova versione dell'allegato con id "attachID" nel documento "31" ***

// checkout
AttachmentBean attachBean = this.titulus4.checkOutContentFile("31", attachID, true);

// ********************************************************************************
// Aggiornamento del file
// ********************************************************************************
...

// checkin

AttachmentBean newAttachment = new AttachmentBean();
newAttachment.setId(attachID);
newAttachment.setFileName("nuova_versione.doc");
newAttachment.setDescription("Manuale versione 2");
newAttachment.setMimeType("application/msword");
newAttachment.setContent(getBytesFromFile(new File("nuova_versione.doc")));

AttachmentBean[] array = new AttachmentBean[]{newAttachment};

SaveParams params = new SaveParams();

params.setPdfConversion(true);

titulus4.checkInContentFiles("31", array, params);
docXML = titulus4.loadDocument("31", false);

System.out.println("document after checkin is:\r\n\n" + docXML + "\n");
```

Modificare un documento esistente

Per modificare un documento è necessario caricare il documento bloccandolo attraverso il metodo loadDocument().

In seguito al caricamento con flag di lock attivo, qualora si intenda abbandonare il documento, è necessario sbloccarlo per mezzo del metodo unlockDocument().

Il salvataggio in modifica di un documento precedentemente caricato e bloccato, deve avvenire per mezzo del metodo saveModifiedDocument(). Notare che il codice xml passato come primo parametro deve avere come radice l'effettiva radice del documento (per es. “doc”) e non la radice della “busta” XML restituita dal caricamento del documento.

Alcuni attributi ed elementi non sono modificabili e le loro modifiche vengono ignorate all'atto del salvataggio del documento. Di seguito una sintesi degli XPath non modificabili:

```
/doc/@nrecord
/doc/@tipo
/doc/@num_prot
/doc/@cod_amm_aoo
/doc/storia
/doc/rif_interni
/doc/rif_esterni
/doc/files
/doc/immagini
/doc/impronta
/doc/minuta
/doc/wflow
/doc/repertorio
```

Per i documenti di protocollo, si aggiungono i seguenti percorsi fra quelli non modificabili:

```
/doc/@anno
/doc/@data_prot
/doc/oggetto
/doc/allegato
```

Attribuire la responsabilità di un documento

La responsabilità di un documento viene assegnata in fase di registrazione dello stesso (si veda “Creare un nuovo documento”). In tale sede è necessario riportare gli estremi del responsabile all’interno dell’elemento /doc/rif_interni/rif_interno, impostando l’attributo diritto=”RPA”. Lo stesso dicasi per il responsabile della minuta (diritto=”RPAM”) e per i nominativi in copia conoscenza (diritto=”CC”).

Qualora in seguito alla registrazione si renda necessario il trasferimento del documento ad altro RPA, o l’aggiunta di nominativi fra i CC, si può far uso del metodo grantRight().

Ad esempio:

```
...
grantRight(id, "RPA", "Ciampi Carlo Azeglio", "Presidenza della Repubblica",
           "KIONCLE", true);
...
```

trasferisce la responsabilità del documento individuato dal suo id (physdoc o nrecord) a “Carlo Azeglio Ciampi”, inviando al nuovo RPA un'email di notifica dell'assegnazione di responsabilità.

Qualora si intenda rimuovere un nominativo dalle copie conoscenza di un documento, si può fare uso del metodo denyRight().

Entrambi i metodi (grantRight() e denyRight()) non richiedono il previo caricamento del documento (loadDocument()) ed il successivo salvataggio (saveModifiedDocument()), procedendo in autonomia a modificare i soli dati relativi a tali diritti nel documento.

Aggiungere un'annotazione ad un documento

È prevista la possibilità di aggiungere un'annotazione ad un documento già registrato. A tale scopo è stato previsto il metodo postIt() al quale è sufficiente trasmettere l'id del documento ed il testo dell'annotazione. Il metodo registra anche data, ora e nome dell’operatore che ha effettuato l’operazione.

A partire dalla release **3.13.1.0** dei servizi è possibile inserire delle annotazioni direttamente nel codice xml dei documenti in fase di salvataggio tramite il metodo saveDocument().
Si rimanda al xml schema di inserimento per maggiori dettagli (lo schema si può scaricare dalla pagina del javadoc del metodo menzionato).
Nelle release precedenti, invece, l'unico modo di aggiungere delle annotazioni è quello di usare il metodo postIt() dopo aver effettuato l'inserimento del documento.

Annullare un documento

È prevista la possibilità di annullare un documento già registrato. A tale scopo è stato previsto il metodo cancelDocument() al quale è sufficiente trasmettere l'id del documento ed il motivo dell’annullamento. All’atto dell’annullamento vengono registrati anche data, ora e nome dell’operatore che ha effettuato l’operazione.

Rimuovere un documento non protocollato

È possibile rimuovere un documento non protocollato e tutti i file ad esso associati per mezzo del metodo deleteDocument(). Non è possibile cancellare documenti protocollati.

Registrare un documento bozza

È possibile inserire nel sistema un documento protocollato in versione bozza (/doc/@bozza=”si”) sul quale è possibile effettuare opportune operazioni di modifica fino al momento della registrazione ufficiale attraverso il metodo applyRegistrationToDraft().

Gestione fascicolo

In Titulus un fascicolo è un contenitore di documenti ed altri fascicoli, che in questo caso vengono chiamati sottofascicoli. Tutti i documenti e i sottofascicoli devono essere omogenei tra loro per classificazione e responsabilità, cioè devono avere lo stesso responsabile e la stessa classificazione del fascicolo radice.

Di seguito si riportano i metodi esposti dai web service per la gestione dei fascicoli.

Consultazione dei fascicoli

Per consultare un fascicolo è possibile procedere in due modi: usando dei metodi specifici per i fascicoli; effettuando una ricerca e caricando i fascicoli trovati con i metodi più generici.

Consultazione con metodi specifici

Il servizio Titulus espone 4 metodi specifici per la consultazione dei fascicoli:

- getFolder()
- getFolderContent()
- getFolderHierarchy()
- getDocumentFolders()

I primi 3 metodi richiedono l'id del fascicolo, cioè uno di questi 3 campi: physdoc (/fascicolo/@physdoc), nrecord (/fascicolo/@nrecord) o il numero del fascicolo (/fascicolo/@numero).

Il quarto metodo, invece, prende in input l'id (nrecord o physdoc) di un documento.

getFolder() restituisce le informazioni sul fascicolo richiesto. Ecco un esempio di xml che viene restituito:

```
<?xml version="1.0" encoding="UTF-8"?>
<Response xmlns:xw="http://www.kion.it/ns/xw"
          canSee="true"
          canEdit="true"
          canAddRPA="true"
          canDelete="true">
  <Document physdoc="19835">
    <fascicolo physdoc="19835" nrecord="000019835-KIONCLE-773b17cd-8774-4b4b-8b14-e2263c9b54e7" stato="aperto"
numero="2012-KIONCLE-01/01.00001" scarto="99" anno="2012" cod_amm_aoo="KIONCLE">
      <rif_interni>
        <rif nome_persona="Grillini Federico" nome_uff="Sviluppo" cod_persona="PI000122" cod_uff="SI000085"
diritto="RPA"/>
      </rif_interni>
      <oggetto xml:space="preserve">Fascicolo documentazione logistica</oggetto>
      <soggetto xml:space="preserve">Kion</soggetto>
      <voce_indice xml:space="preserve">I/1 - Legislazione e circolari esplicative</voce_indice>
      <classif xml:space="preserve" cod="01/01">01/01 - Leggi e rispettive circolari applicative</classif>
      <note xml:space="preserve">Alcune note</note>
      <storia>
        <creazione oper="Grillini Federico" cod_oper="PI000122" uff_oper="Sviluppo" cod_uff_oper="SI000085"
data="20120308" ora="17:25:39"/>
        <responsabilita cod_persona="PI000122" cod_uff="SI000085" nome_persona="Grillini Federico" nome_uff="
Sviluppo" operatore="Grillini Federico (Sviluppo)" cod_operatore="PI000122" data_visto="20120308" ora_visto="17:
25:39" visto_da="Grillini Federico (Sviluppo)" data="20120308" ora="17:25:39"/>
      </storia>
    </fascicolo>
  </Document>
</Response>
```

Per avere delle informazioni sui campi del fascicolo, si veda il paragrafo Campi xml principali di un fascicolo.

getFolderContent() restituisce i documenti contenuti e collegati al fascicolo richiesto. Ecco un esempio di xml che viene restituito:

```
<?xml version="1.0" encoding="UTF-8"?>
<Response xmlns:xw="http://www.kion.it/ns/xw"
          pageCount="5"
          pageIndex="1"
          seleId="3sod11f14f5dd34e01"
          seleSize="50">

  <doc physdoc="..." nrecord="..." ...>
     ...
  </doc>
  ...
</Response>
```

In questo caso la struttura del codice xml è quella di un elenco di documenti.

getFolderHierarchy() consente di recuperare l'intera gerarchia (in forma sintetica) dei sottofascicoli e dei documenti contenuti (e anche collegati) in un fascicolo.

Ecco un esempio di xml che viene restituito:

```
<?xml version="1.0" encoding="UTF-8"?>
<Response xmlns:xw="http://www.kion.it/ns/xw">
  <fascicolo physdoc="..." numero="...">
   <oggetto>...</oggetto>
   <doc physdoc="..." data_prot="..." num_prot="...">
      <oggetto>...</oggetto>
   </doc>
   ...
   <fascicolo physdoc="..." numero="...">
      <oggetto>...</oggetto>
      <doc physdoc="..." data_prot="..." num_prot="...">
         <oggetto>...</oggetto>
      </doc>
   ...
   </fascicolo>
   ...
   ...
  </fascicolo>
</Response>
```

Dei fascicoli viene riportato il physdoc, il numero e l'oggetto; dei documenti il physdoc, la data di protocollazione, il numero di protocollo e l'oggetto.

getDocumentFolders() restituisce tutti i fascicoli in cui il documento è contenuto. Il documento può essere inserito in un fascicolo principale e un numero qualunque di eventuali fascicoli collegati. Bella busta xml restituita, i fascicoli collegati presentano un attibuto linked = true.

Ecco un esempio di xml che viene restituito:

```
<?xml version="1.0" encoding="UTF-8"?>
<Response xmlns:xw="http://www.kion.it/ns/xw">
  <fascicolo physdoc="..." numero="...">
    <oggetto ...</oggetto>
    ...
  </fascicolo>
  <fascicolo physdoc="..." numero="..." linked="true">
    <oggetto ...</oggetto>
    ...
  </fascicolo>
  ...
  <fascicolo physdoc="..." numero="..." linked="true">
    <oggetto ...</oggetto>
    ...
  </fascicolo>
</Response>
```

Consultazione con metodi generici

Qualora non si conosca l'id del fascicolo che si vuole consultare o qualora si vogliano leggere più fascicoli, è possibile effettuare delle ricerche con il metodo *search*.

I fascicoli così trovati possono essere poi caricati con i metodi *load*Document()*, proprio come se fossero dei normali documenti.

Inserimento e modifica dei fascicoli

Inserimento di fascicoli normali

L'inserimento di un nuovo fascicolo può essere effettuato tramite il metodo **newFolder()**.

Ecco un esempio di xml che occorre passare al metodo per la creazione di un fascicolo normale (si rimanda al javadoc del servizio per la dtd completa):

```
<?xml version="1.0" encoding="UTF-8"?>
<fascicolo anno="2011">
   <oggetto>fascicolo di prova creato mediante ws</oggetto>
   <classif cod="XX"/>
   <rif_interni>
       <rif diritto="RPA" nome_persona="VV" nome_uff="ZZ"/>
   </rif_interni>
   <voce_indice xml:space="preserve">Accordi bilaterali interuniversitari</voce_indice>
</fascicolo>
```

In risposta il metodo restituisce il codice xml del fascicolo creato:

```
<?xml version="1.0" encoding="UTF-8"?>
<Response xmlns:xw="http://www.kion.it/ns/xw"
          canSee="true"
          canEdit="true"
          canAddRPA="true">
  <Document physdoc="909">
    <fascicolo anno="2011" cod_amm_aoo="ADMNADM" physdoc="909" nrecord="000000909-ADMNADM-9e309372-d25c-43ac-
a200-e33ecefe9b96" numero="2011-ADMNADM-03/13.00006" scarto="99" stato="aperto">
      <oggetto xml:space="preserve">fascicolo di prova creato mediante ws</oggetto>
      <classif cod="03/13" xml:space="preserve">03/13 - Programmi di mobilità</classif>
      <rif_interni>
        <rif cod_persona="PI000001" cod_uff="SI000002" diritto="RPA" nome_persona="Grillini Federico" nome_uff="
Sviluppo"/>
      </rif_interni>
      <voce_indice xml:space="preserve">Accordi bilaterali interuniversitari</voce_indice>
      <storia>
        <creazione cod_oper="PI000001" cod_uff_oper="SI000002" data="20120312" oper="Grillini Federico" ora="16:
05:02" uff_oper="Sviluppo"/>
        <responsabilita cod_operatore="PI000001" cod_persona="PI000001" cod_uff="SI000002" data="20120312"
data_visto="20120312" nome_persona="Grillini Federico" nome_uff="Sviluppo" operatore="Grillini Federico
(Sviluppo)" ora="16:05:02" ora_visto="16:05:02" visto_da="Grillini Federico (Sviluppo)"/>
      </storia>
    </fascicolo>
  </Document>
</Response>
```

⚠ A causa di una limitazione, in inserimento di un nuovo fascicolo, il codice della classificazione e i nomi della persona e dell'ufficio RPA sono obbligatori anche quando si indica una voce di indice. E' possibile aggirare questa limitazione specificando dei dati inventati (es. "xx"), dato che la classificazione e il responsabile del fascicolo vengono recuperati dalla voce di indice (come mostrato nell'esempio precedente).

Conviene usare sempre le voci di indice per impostare la classificazione e gli smistamenti.

Inserimento di fascicoli speciali

Con newFolder() è possibile creare anche fascicoli speciali, cioè dello studente, del personale e di persona giuridica.

In questo caso il codice xml da dare in input ha una struttura differente rispetto a quella dei normali fascicoli.

Ecco un esempio di creazione di un fascicolo dello studente:

```
<?xml version="1.0" encoding="UTF-8"?>
<fascicolo_speciale categoria="studente"
                    data_nascita="11/02/1991"
                    cod_fis="02191651203"
                    matricola="kion002"
                    stuid="KN123X118"
                    data_immatricolazione="21/09/2011"
                    normativa="DM270 - D.M. 270/2004"
                    tipo_corso="L2 - CORSO DI LAUREA TRIENNALE"
                    corso_studio="101605 - SCIENZE DEI SERVIZI GIURIDICI (DM 270)">

  <oggetto>Rossi Gianluca</oggetto>
</fascicolo_speciale>
```

Alcune note sui campi:

- “categoria” vale “studente” per i fascicoli degli studenti;
- per ogni studente è obbligatorio inserire la matricola, il codice fiscale e lo stuid, cioè l'identificativo univoco della sua carriera (fornito dalla segreteria studenti);
- Nel campo “oggetto” devono essere dichiarati il cognome e il nome dello studente;
- è, inoltre, opportuno indicare i dati relativi al corso di studio seguito (“corso_studio”, “tipo_corso” e “normativa”) e la data di immatricolazione dello studente (“data_immatricolazione”).

Anche in questo caso il metodo risponde fornendo il codice xml del fascicolo creato:

```
<?xml version="1.0" encoding="UTF-8"?>
<Response xmlns:xw="http://www.kion.it/ns/xw"
          canSee="true"
          canEdit="true">
  <Document physdoc="19838">
    <fascicolo anno="2012"
               cod_amm_aoo="KIONCLE"
               physdoc="19838"
               nrecord="000019838-KIONCLE-9e309372-d25c-43ac-a200-e33ecefe9b96"
               numero="2012-KIONCLE-05/00.KN123X118"
               stato="aperto">
      <oggetto xml:space="preserve">Rossi Gianluca</oggetto>
      <fascicolo_speciale cod_fis="02191651203" corso_studio="101605 - SCIENZE DEI SERVIZI GIURIDICI (DM 270)"
data_immatricolazione="21/09/2011" data_nascita="11/02/1991" matricola="kion002" normativa="DM270 - D.M. 270
/2004" stuid="KN123X118" tipo_corso="L2 - CORSO DI LAUREA TRIENNALE"/>
      <classif cod="05/00" xml:space="preserve">05 - Studenti e laureati</classif>
      <storia>
        <creazione cod_oper="PI000122" cod_uff_oper="SI000085" data="20120313" oper="Grillini Federico" ora="09:
19:03" uff_oper="Sviluppo"/>
      </storia>
    </fascicolo>
  </Document>
</Response>
```

Per la creazione di un fascicolo del personale, invece, occorre indicare un xml del tipo:

```
<?xml version="1.0" encoding="UTF-8"?>
<fascicolo_speciale categoria="personale docente"
                    data_nascita="11/02/1991"
                    luogo_nascita="Bologna"
                    cod_fis="02191651203"
                    matricola="kion002"
                    data_assunzione="01/01/2000"
                    data_cessazione="01/09/2011">
  <oggetto>Verdi Matteo</oggetto>
</fascicolo_speciale>
```

dove:

- “categoria” per i fascicoli del personale vale “personale docente”, “personale non docente” e “personale non strutturato”;
- per ogni persona è obbligatorio inserire la matricola;
- Nel campo “oggetto” devono essere dichiarati il cognome e il nome della persona;

- è, inoltre, opportuno indicare il codice fiscale e le date di assunzione/cessazione.

Ecco un esempio di risposta del metodo contenente il codice xml del fascicolo creato:

```
<?xml version="1.0" encoding="UTF-8"?>

<Response xmlns:xw="http://www.kion.it/ns/xw"
          canSee="true"
          canEdit="true">
  <Document physdoc="19839">
    <fascicolo anno="2012"
               cod_amm_aoo="KIONCLE"
               physdoc="19839"
               nrecord="000019839-KIONCLE-9e309372-d25c-43ac-a200-e33ecefe9b96"
               numero="2012-KIONCLE-07/00.kion002"
               stato="aperto">
      <oggetto xml:space="preserve">Verdi Matteo</oggetto>
      <fascicolo_speciale categoria="personale docente" cod_fis="02191651203" data_assunzione="01/01/2000"
data_cessazione="01/09/2011" data_nascita="11/02/1991" luogo_nascita="Bologna" matricola="kion002"/>
      <classif cod="07/00" xml:space="preserve">07 - Personale</classif>
      <storia>
        <creazione cod_oper="PI000122" cod_uff_oper="SI000085" data="20120313" oper="Grillini Federico" ora="09:
44:19" uff_oper="Sviluppo"/>
      </storia>
    </fascicolo>
  </Document>
</Response>
```

Per la creazione di un fascicolo di persona giuridica, invece, occorre indicare un xml del tipo:

```
<?xml version="1.0" encoding="UTF-8"?>
<fascicolo_speciale categoria="personaGiuridica"
                    matricola="001"
                    id_acl="SE000001"
                    cod_fis="02191651203"
                    partita_iva="01234567890"
                    nazione="ITALIA"
                    posizione="Iscritto">
  <oggetto>KION spa</oggetto>
</fascicolo_speciale>
```

dove:

- “categoria” per i fascicoli del personale vale “personaGiuridica”;
- per ogni persona giuridica è obbligatorio inserire la matricola;
- Nel campo “oggetto” deve essere dichiarata la denominazione della persona giuridica;
- è, inoltre, opportuno indicare il codice fiscale, partita iva, nazione e lo stato posizione nell'elenco degli operatori economici.

Ecco un esempio di risposta del metodo contenente il codice xml del fascicolo creato:

```
<?xml version="1.0" encoding="UTF-8"?>

<Response xmlns:xw="http://www.kion.it/ns/xw"
          canSee="true"
          canEdit="true">
  <Document physdoc="19840">
    <fascicolo anno="2014"
               cod_amm_aoo="KIONCLE"
               physdoc="19840"
               nrecord="000019840-KIONCLE-9e309372-d25c-43ac-a200-e33ecefe9b97"
               numero="2014-KIONCLE-10/00.00001"
               stato="aperto">
      <oggetto xml:space="preserve">KION spa</oggetto>
      <fascicolo_speciale categoria="personaGiuridica"
                    matricola="001"
                    cod_fis="02191651203"
                    partita_iva="01234567890"
                    nazione="ITALIA"
                    posizione="Iscritto"/>
      <classif cod="10/00" xml:space="preserve">10 - Patrimonio, economato e provveditorato</classif>
      <storia>
        <creazione cod_oper="PI000122" cod_uff_oper="SI000085" data="20140227" oper="Grillini Federico" ora="09:
00:00" uff_oper="Sviluppo"/>
      </storia>
    </fascicolo>
  </Document>
</Response>
```


Nel caso dei fascicoli speciali le voci di indice non vengono usate e non è nemmeno necessario indicarne la classificazione, perché questa viene automaticamente recuperata dalla configurazione di Titulus.

Inserimento di sottofascicoli

Un fascicolo può contenere, oltre ai documenti, altri fascicoli, detti sottofascicoli; in altri termini, è possibile avere una vera e propria gerarchia di fascicoli.

La creazione dei sottofacicoli richiede l'invocazione del metodo **newSubFolder()**.

Si rimanda al javadoc per i dettagli sui suoi parametri.

Modifica dei fascicoli

Mediante il metodo **modifyFolder()** è possibile modificare i fascicoli (normali o speciali).

Questo metodo prende in input due parametri:

- L'id del fascicolo da modificare (physdoc, nrecord o numero del fascicolo);
- Il codice xml del fascicolo corretto con le modifiche.

Prima della modifica, quindi, il fascicolo deve essere caricato per ottenere il suo xml. Il caricamento lo si può effettuare tramite i metodi *getFolder()* o *loadD ocument*()* (si faccia riferimento alla parte di consultazione dei fascicoli). Notare che, se si usano i metodi generici *loadDocument*()* per il caricamento dei fascicoli, non è necessario caricare con lock il fascicolo da modificare, poiché questa operazione viene gestita automaticamente dal metodo modifyFolder().

Il codice xml del fascicolo da passare in input a modifyFolder() deve avere come radice “fascicolo”; questo xml deve essere estratto, cioè, dalla risposta restituita dai metodi di consultazione del fascicolo, come evidenziato nell'esempio seguente:

<?xml version=“1.0” encoding=“UTF-8”?>

<Response …>

<Document physdoc=“…”>

```
<fascicolo ...>
    ...
  </fascicolo>
```

</Document>

</Response>

Come i documenti, anche i fascicoli hanno dei campi non modificabili; qualora li si modifichi, essi vengono semplicemente riportati al valore originario.

Ecco l'elenco di questi campi:

- /fascicolo/@anno
- /fascicolo/@cod_amm_aoo
- /fascicolo/@numero
- /fascicolo/@nrecord
- /fascicolo/@stato
- /fascicolo/@data_chiusura
- /fascicolo/classif
- /fascicolo/rif_interni
- /fascicolo/storia
- /fascicolo/voce_indice
- /fascicolo_speciale/@stuid
- /fascicolo_speciale/@matricola

Il metodo modifyFolder() restituisce, infine, il codice xml del fascicolo modificato.

Apertura e chiusura dei fascicoli

Per chiudere un fascicolo aperto si può usare il metodo **closeFolder()**, che chiude il fascicolo indicato imponendo il valore “chiuso” nell'attributo /fascicolo /@stato.

E' anche possibile indicare dei parametri aggiuntivi relativi alla chiusura mediante il bean *CFParams*.

Attualmente per i fascicoli degli studenti (solo quelli radice, non i sottofascicoli) i parametri di chiusura sono obbligatori; in particolare è obbligatorio motivare la chiusura del fascicolo (per es. “Rinuncia”), valorizzando il corrispondente parametro del bean (*CFParams.setReason (String)*).

La riapertura di un fascicolo chiuso, invece, può essere effettuata tramite il metodo **openFolder()**.

Notare che la riapertura di un fascicolo chiuso è sempre necessaria quando si vogliono inserire in esso dei documenti.

Tutti i metodi menzionati in questo paragrafo individuano il fascicolo da aprire/chiudere tramite il suo id (physdoc, nrecord o numero).

Cancellazione di fascicoli e trasferimenti

La cancellazione di un fascicolo/sottofascicolo è possibile mediante il metodo **deleteFolder()**.

Qualora si voglia, invece, cambiare il responsabile RPA di un fascicolo (possibile solo per i fascicoli radice di una gerarchia di fascicoli), occorre impiegare il metodo **assignFolderRPA()**.

Notare, però, che questa operazione è onerosa in termini di tempo, poiché comporta la creazione di un nuovo fascicolo ed il trasferimento in esso di tutti i documenti del vecchio fascicolo, che, a operazione conclusa, viene chiuso. Quindi, essendo la chiamata sincrona, il tempo di risposta può essere lungo.

Fascicolazione dei documenti

Inserimento in un fascicolo

Per inserire un documento in un fascicolo bisogna invocare il metodo **addInFolder()**.

Il metodo prende in input un xml con la seguente struttura:

```
<?xml version="1.0" encoding="UTF-8" ?>
<fascicolo physdoc="..." nrecord="..." numero="...">
   <doc physdoc="..." nrecord="..." num_prot="..." minuta="si|no"/>
</fascicolo>
```

In questo xml occorre indicare il fascicolo in cui deve essere inserito il documento e il documento da inserire.

Per individuare il fascicolo è possibile usare uno di questi 3 id:

- physdoc
- nrecord
- numero del fascicolo

Per il documento, invece, può essere usato un id tra:

- physdoc
- nrecord
- numero di protocollo (se protocollato)

Infine, l'attributo “minuta” dell'elemento “doc” specifica - per un documento tra uffici - se inserire la minuta o l'originale nel fascicolo.

Il documento che si vuole fascicolare deve avere la stessa classificazione e lo stesso responsabile (RPA) del fascicolo, altrimenti si ottiene un errore.

Rimozione da un fascicolo

Per rimuovere un documento da un fascicolo bisogna invocare il metodo **removeFromFolder()**.

Il metodo prende in input un xml con la seguente struttura:

```
<?xml version="1.0" encoding="UTF-8" ?>
<fascicolo>
   <doc physdoc="..." nrecord="..." num_prot="..." minuta="si|no"/>
</fascicolo>
```

In questo caso, a differenza dell'inserimento in un fascicolo, è sufficiente indicare solo il documento che interessa; il metodo provvederà a rimuoverlo dal fascicolo a cui appartiene.

Gli id che si possono usare per individuare il documento sono gli stessi menzionati nella fascicolazione.

Inserimento di un collegamento ad un fascicolo

In Titulus un documento può essere assegnato ad un solo fascicolo nel rispetto delle regole di classificazione. È tuttavia possibile collegare un documento a più fascicoli; in questo caso il documento non appartiene fisicamente al fascicolo, ma è logicamente collegato ad esso. Esempio: la fattura per l'acquisto di un PC è fisicamente all'interno di un fascicolo dell'ufficio acquisti insieme alla richiesta di acquisto, alle varie offerte e all'ordine. È possibile che la stessa fattura sia logicamente inserita all'interno di un fascicolo dell'ufficio ragioneria insieme al suo mandato e ad altre fatture e mandati.

Per creare un collegamento a un fascicolo in un documento occorre invocare il metodo **addLinkToFolder()**.

Il codice xml da passare al metodo è molto simile a quello usato per la fascicolazione vera e propria, a cui si rimanda per i dettagli:

```
<?xml version="1.0" encoding="UTF-8" ?>
<fascicolo physdoc="..." nrecord="..." numero="...">
   <doc physdoc="..." nrecord="..." num_prot="..." />
</fascicolo>
```

Un documento può contenere più collegamenti a fascicoli differenti.

Rimozione di un collegamento ad un fascicolo

Per eliminare un collegamento a un fascicolo in un documento occorre invocare il metodo **deleteLinkToFolder()**.

Il codice xml da passare al metodo è molto simile a quello usato per la fascicolazione vera e propria, a cui si rimanda per i dettagli:

```
<?xml version="1.0" encoding="UTF-8" ?>
<fascicolo physdoc="..." nrecord="..." numero="...">
   <doc physdoc="..." nrecord="..." num_prot="..." />
</fascicolo>
```

Notare che, a differenza della rimozione di un documento da un fascicolo, in questo caso è necessario indicare il fascicolo a cui è collegato il documento, perché nel documento possono essere presenti dei collegamenti ad altri fascicoli.

Workflow

Invocando il metodo getWorkflowId() si può accedere al/ai workflow associato/i ad un documento e con il metodo getWorkflowAction() si ottengono le eventuali azioni disponibili.

Il metodo startWorkflow() avvia un flusso su un documento, mentre continueWorkflow() fa avanzare un workflow attivo.

Version History

Version history dei servizi web di Titulus 4.2

FAQ

Frequently asked questions

Altri manuali dei WS

Servizio Titulus Organi 4

Servizio restful Titulus Organi

Servizio ACL 4

Servizio Titulus4Publication