Componente Java para la integración con BDU

Dirección General de Tecnologías de la Información y Comunicaciones

Áreas de Gobierno Tecnológico de SSII y del Dato

Servicio Andaluz de Salud
Consejería de Sanidad, Presidencia y Emergencias

Contenido

- Dirección General de Tecnologías de la Información y Comunicaciones
  - Áreas de Gobierno Tecnológico de SSII y del Dato
- 1. Introducción
- 2. Dependencias
- 3. Historial de Cambios
- 4. Integración de la librería
  - Integración con el api client
  - Configuración
    - JAX-WS
    - JPA
- Operación
  - Criterias
  - Métodos de servicio

Resumen

- **Versión:** v01r21
- **Fecha publicación:** 26 feb 2018
- **Entrada en vigor desde:** 26 feb 2018

blocked URL

Cumplimiento normativo

Las normas expuestas son de obligado cumplimiento. La STIC podrá estudiar los casos excepcionales los cuales serán gestionados a través de los responsables del proyecto correspondiente y autorizados por el Área de Gobernanza de la STIC. Asimismo cualquier aspecto no recogido en estas normas deberá regirse en primera instancia por las guías técnicas correspondientes al esquema nacional de seguridad y esquema nacional de interoperabilidad según correspondencia y en su defecto a los marcos normativos y de desarrollo software establecidos por la Junta de Andalucía, debiendo ser puesto de manifiesto ante la STIC.

La STIC se reserva el derecho a la modificación de la norma sin previo aviso, tras lo cual, notificará del cambio a los actores implicados para su adopción inmediata según la planificación de cada proyecto.

En el caso de que algún actor considere conveniente y/o necesario el incumplimiento de alguna de las normas y /o recomendaciones, deberá aportar previamente la correspondiente justificación fehaciente documentada de la solución alternativa propuesta, así como toda aquella documentación que le sea requerida por la STIC para proceder a su validación técnica.

Contacto Arquitectura: l-arquitectura.stic@juntadeandalucia.es

Histórico de cambios

Los cambios en la normativa vendrán acompañados de un registro de las modificaciones. De este modo se podrá realizar un seguimiento y consultar su evolución. Ordenándose de mas recientes a menos recientes, prestando especial cuidado a las cabeceras de la tablas dónde se indican las fechas de entrada en vigor y versión.

Versión	v01r21	**Fecha publicación**	26 feb 2018	**Fecha entrada en vigor**	26 feb 2018
Alcance					
• Integración con BDU para desarrollos Java desde la versión 2.2.0 del componente.					

1. Introducción

Esta librería permitirá de hacer uso de la Api proporcionada por Bdu y tener acceso a utilidades relacionadas con la operativa de **Bdu**.

Actualmente la librería tiene las siguiente funcionalidades:

- Versión <= 1.2
    - Consulta detallada.
    - Consulta corta.
    - Consulta recién nacido
- Versión >= 2.0
    - Se mantienen las interfaces de las versiones anteriores 1.2.
    - Se añaden métodos para consulta paginadas, funcionales solamente en la implementación del conector JPA. El conector actual basado en WS lanzará NotSupportedException.
- Versión >= 2.3 | 3.0.1.1
    - Se añaden servicios de alta de usuarios y recién nacidos para la implementación WS.
- Versión >= 2.4 | 3.1.0
    - Se añaden servicios de modificación de datos de usuarios para la implementación WS.

De cara a priorizar futuras funcionalidades, si algún proveedor tiene la necesidad de alguna funcionalidad adicional se deberá de comunicar a l-arquitectura.stic.sspa@juntadeandalucia.es

> ⚠ **Nota sobre el versionado dual (Ramas 2.x vs 3.x)**
>
> Para dar soporte a las diferentes plataformas tecnológicas del SAS, la librería mantiene dos líneas de desarrollo concurrentes. La rama **2.x** está diseñada para mantener la retrocompatibilidad con entornos **JDK 1.7 / Java EE**. Por su parte, la rama **3.x** está adaptada a las nuevas arquitecturas basadas en **JDK 21 / Jakarta EE**. Ambas ramas son funcionalmente equivalentes y evolucionan en paralelo en cuanto a contratos y servicios públicos se refiere, siendo la 3.0.1 equivalente a la base funcional de la 2.3.3, y nivelándose progresivamente en versiones posteriores.

2. Dependencias

BduApiClient (JDK 1.7)	MacoApiClient	sasutils-xml	sasutils-parse	sasutils-file	sasconfiguration-annotation	estructuraApiClient-api	claveequipo-api-client
2.5.0	2.0.1	2.0.2	2.0.2	2.0.2	1.1.0	2.1.4	1.1.0
2.4.0	2.0.1	2.0.2	2.0.2	2.0.2	1.1.0	2.1.4	1.1.0
2.3.3	2.0.1	2.0.2	2.0.2	2.0.2	1.1.0	2.1.4	1.1.0
2.3.2	2.0.1	2.0.2	2.0.2	2.0.2	1.1.0	2.1.4	1.1.0
2.3.1	2.0.1	2.0.2	2.0.2	2.0.2	1.1.0	2.1.4	1.1.0
2.3.0	2.0.0	2.0.2	2.0.2	2.0.2	1.1.0	2.1.4	1.1.0
2.2.0	2.0.0	2.0.2	2.0.2	2.0.2	1.1.0	2.1.4	1.1.0
2.0.0.1	1.8.0.2	1.1.0.1	1.1.0.1	1.1.0.1	1.1.0	2.1.2.2	1.0.0.1
1.2.0.2	1.8.0.2	1.0.0.1	1.0.0.1	1.0.0.1	1.0.0.1	-	-
1.2.0.1	1.8.0.1	1.0.0.1	1.0.0.1	1.0.0.1	1.0.0.1	-	-
1.1.0.2	1.7.2.2	1.0.0.1	1.0.0.1	1.0.0.1	-	-	-
1.1.0.1	1.7.2.2	1.0.0.1	1.0.0.1	1.0.0.1	-	-	-
1.0.1.1	1.7.0.1	1.0.0.1	1.0.0.1	1.0.0.1	-	-	-
1.0.0.1	1.3.1.1	1.0.0.1	1.0.0.1	1.0.0.1	-	-	-

BduApiClient (JDK 21)	MacoApiClient	sasutils-xml	sasutils-parse	sasutils-file	sasconfiguration-annotation	estructuraApiClient-api	claveequipo-api-client
3.2.0	3.0.1	2.0.2	2.0.2	2.0.2	3.0.2	3.0.0	2.0.1
3.1.0	3.0.1	2.0.2	2.0.2	2.0.2	3.0.2	3.0.0	2.0.1
3.0.1.1	3.0.1	2.0.2	2.0.2	2.0.2	3.0.2	3.0.0	2.0.1
3.0.0.1	3.0.1	2.0.2	2.0.2	2.0.2	3.0.2	3.0.0	2.0.0.1

3. Historial de Cambios

- Corrección en la obtención de información de desplazamiento de usuarios para determinar si el usuario está desplazado (OTARQ-2644). Esta versión resuelve un bug por el cual no se indicaba adecuadamente si el desplazamiento se encontraba activo o no. Igualmente permite recuperar la fecha de inicio del desplazamiento, y los meses de su duración.

- Se incluyen funcionalidades de modificación de datos de usuarios en BDU a través del conector SOA (OTARQ-2586), por medio del componente BduRepository (BduWriteableRepository):
    - updatePatientBasicData => MODUSU01: Modificación de datos personales
    - updatePatientAddress => MODUSU02: Modificación de domicilio y datos de contacto
    - updatePatientDisplacement => MODUSU03: Modificación de datos de desplazamiento
    - updatePatientControlData => MODUSU04: Modificación de datos de control

- Forzado de endpoints HTTPS.
- Actualización del componente a JDK 21 y Jakarta EE 10 (Versión v3.x):
    - Se refactorizan los pom.xml para centralizar la gestión del versionado de los componentes y plugins en el proyecto padre.
    - Actualización a Jakarta EE 10.0.0 (La versión con soporte oficial para JDK 21 es la 11.0.0 - https://jakarta.ee/specifications/platform/11/)
        - Refactorizado de paquetes al actualizar a Jakarta 10.0.0:
            - javax.annotation.* => jakarta.annotation.*
            - javax.ejb.* => jakarta.ejb.*
            - javax.el.* => jakarta.el.*
            - javax.enterprise.* => jakarta.enterprise.*
            - javax.faces.* => jakarta.faces.*
            - javax.inject.* => jakarta.inject.*
            - javax.persistence.* => jakarta.persistence.*
            - javax.servlet.* => jakarta.servlet.*
    - Actualización de schemas xml: beans.xml, persistence.xml y diversos entity-mappings.xml
    - Actualización de Mockito a 5.17.0
        - Refactorizado de paquetes al actualizar a Mockito 5.17.0:
            - org.mockito.Matchers.any => org.mockito.ArgumentMatchers.any
            - org.mockito.Matchers.anyObject => org.mockito.ArgumentMatchers.any
            - org.mockito.Matchers.eq => org.mockito.ArgumentMatchers.eq
    - Actualización de JUnit 4 a JUnit 5
    - Actualización de dependencias::

        - javax.jws:jsr181-api:1.0-MR1 y javax.xml.ws:jaxws-api:2.3.1 => jakarta.jws:jakarta.jws-api:3.0.0 y jakarta.xml.ws:jakarta.xml.ws-api:3.0.0
    - Eliminación de dependencia obsoleta powermock e incompatible con JUnit5. Se suple con una implementación nativa:
        - Se adapta el test es.ja.csalud.sas.componentescomunes.bduapiclient.jpa.control.jpa.PartialListQueryTest
    - Actualización de Shrinkwrap de 2.2.4 a 3.3.4
    - Actualización de Arquillian de 1.1.11.Final a 1.9.4.Final
        - Reemplazo de Arquillian-Openliberty por Arquillian-Payara ya que se presentaban problemas con Jakarta 10. Arquillian-Openliberty según la documentación requiere arquillian-support-jakarta:3.0 (Actualmente solo disponible en SNAPSHOT en repositorio git - https://github.com/OpenLiberty/liberty-arquillian/blob/main/pom.xml), pero esa versión aún se encuentra en desarrollo, siendo la última publicada en mvnrepository la 2.1 - https://mvnrepository.com/artifact/io.openliberty.arquillian/arquillian-liberty-support-jakarta
        - Inclusión de configuración de arquillian-payara (Payara) en src/testpayara/config/glassfish-resources.xml
        - Inclusión de almacén de certificados personalizado para conexiones seguras entre Payara y la infraestructura del SAS (/cacerts-sas-ssl.p12)
        - Configuración de profile "integration-managed" para pruebas de integración
        - Correcciones y ajustes en pruebas de integración de los componentes de la librería (BaseBduSearchTestIT, BduFrameworkBaseTestIT)
    - Actualización de Dropwizard Metrics a 4.2.32
    - Actualizados componentes comunes del SAS:
        - sasutils.version: 2.0.2 (Sin cambios)
        - macoApiClient: 2.0.1 => 3.0.1
        - claveequipoApiClient: 1.1.0 => 2.0.1
        - estructuraApiClient: 2.1.4 => 3.0.0
        - sasconfiguration-annotation: 1.1.0 => 3.0.2
        - sasconfiguration-api: 1.2.0 => 3.0.2
        - sasconfiguration-cdi: 2.2.1 => 3.0.2
        - sasconfiguration-core: 2.2.1 => 3.0.2
    - Se crea una base de datos en memoria para pruebas con H2, eliminando la dependencia de tener que tener acceso a los esquemas de Oracle.
    - Migración del proyecto de ejemplo "bduApiClient-testproject" para hacer uso de las nuevas versiones de componentes.

- Añadido configuración de endpoints para poder soportar HTTPS

- Modificación de la política de versionado con reducción a tres dígitos
- Alta de recién nacidos a través del web-service de BDU expuesto para tal fin y bajo el contrato de la versión 2.12.
- Alta de usuarios en BDU a través del web-service de BDUexpuesto para tal fin y bajo el contrato actual de la versión 2.12.
- Fixes para el servicio de consulta de recién nacidos en base al nuhsa de los progenitores
- Maestras conteniendo enumerados para los elementos más habituales relacionados con BDU según la spec del último contrato disponible en la documentación de la OTI que está versionado como 2.3
- Proyecto de ejemplo con interacción mixta de las dos implementaciones en base a identificación de servicios con @Named
- Modificaciones en las entities a nivel de API:
    - En las políticas de los getters nulos, devolviéndose ahora objetos vacíos listos para su uso en lugar de nulos.
    - Nuevos elementos de negocio para direcciones,adscripción, etc..
- Se añade una nueva interfaz para la interacción con los datos de BDU para el alta de usuarios BduRepository, esta sustituirá a futuro el componente de negocio actual BduSearch, pero aún no se marca como deprecado.
- Se actualizan las dependencias de todos los componentes estructurales pasando a ser:
    - macoApiClient:2.0.1
    - Jpa-impl:
        - estructuraApiClient:2.1.4
        - claveEquipoApiClient:1.1.0
    - sasutils:2.0.2

- Se incluyen los campos citizenshipcode y citizenship para consultar la nacionalidad del paciente.
- Se incluye la fecha de fallecimiento, para pasivos por fallecimiento y nacidos exitus.
- Se actualiza la versión del contrato del ws, pasándo a ser la 2.16 de WSBDU
- Se incorporan las consultas:
    - Consulta de nacionalidad.
    - Consulta de fecha de fallecimiento.

- Alta de nuevos usuarios en bdu a través del modelo INSUSU01 [bduApiClient-ws-impl]
    - No soporta desplazamientos estos deben hacerse a través del modelo de modificación de datos en BDU
- Alta de recien nacidos en bdu a través del modelo INSUSU03 [bduApiClient-ws-impl]
    - No soporta desplazamientos estos deben hacerse a través del modelo de modificación de datos EN BDU
    - La bandera relativa a "nacido muerto" se calcula en base al tipo de pasivo indicado para el paciente.
    - LA bandera relativa a "no identificable" se calcula en base los id del paciente sino dispone de ninguno de:
        - DOCUMENTO IDENTIFICATIVO
        - E111

        - CODSNS
        - NUHSA
        - CIP
- Actualización de dependencias

- Actualización de las dependencias a sas-utils en su versión mas reciente 1.1.0.1
- Se modifican los servicios de bduSearch para dar soporte a la paginación de registros.
- Se incluye el conector JPA para consulta de BDU.
- Se incluyen valores comunes de maestras relacionadas con bdu en el empaquetado bduApiClient-maestras.
- Se incluyen dependencias a estructuraApiClient para poder traer la información relativa a los profesionales y estructura que se requiera para las consultas detalladas.
- Creación de proyecto *test-project* de ejemplo para futuras intregaciones con bdu

- Hotfix para cambiar a la versión 1.8.0.2 de la librería de maco

- Adaptación a la librería de sasconfiguration

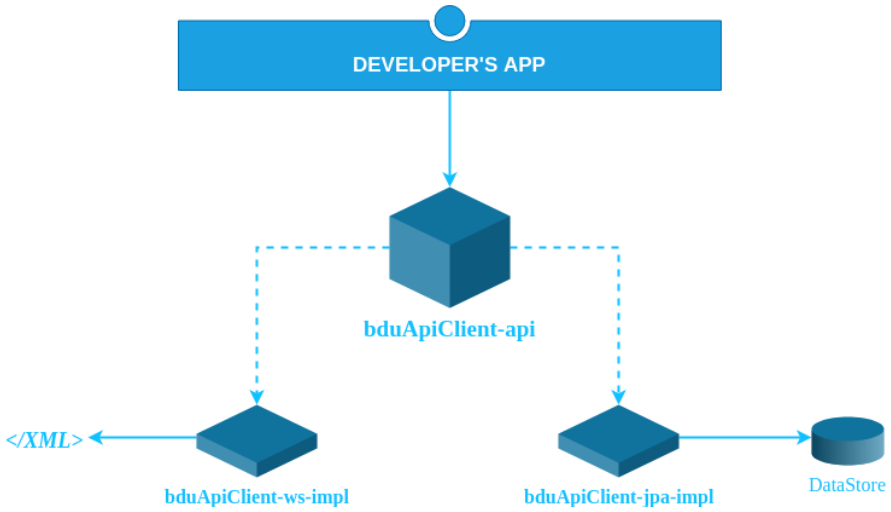
- Hotfix para cambiar el orden de generación de las propiedades del header de los mensajes

- Uso de la nueva librería de maco 1.7.2.2

- Modificado el constructor de la clase de configuración.

- Añadidos servicios de consulta:
    - Consulta detallada.
    - Consulta corta.
    - Consulta recien nacido

4. Integración de la librería



Integración con el api client

La API de integración con bdu, está construida en base a un contrato de servicio especificado en "**bduApiClient-api**" que es implementado por conectores específicos a la solución tecnológica que se decida en el ámbito del proyecto. Actualmente las soluciones tecnológicas aportadas resuelven los datasources de BDU mediante JAX-WS ó JPA.

Para incluir la librería en un proyecto Maven es necesario realizar los siguientes pasos:

1. Agregar el repositorio de la Junta de Andalucía. Para ello, seguir las indicaciones de la página Repositorio de artefactos.

2. Especificar la dependencia principal contra la API en el POM del proyecto que requiere la funcionalidad

```
        ...
    <dependencies>
        <!--BDU-->
        <dependency>
            <groupId>es.ja.csalud.sas.componentescomunes.bduapiclient</groupId>
            <artifactId>bduApiClient-api</artifactId>
            <version>x.x.x</version>
        </dependency>
...
```

- Conector JPA

```
        ...
    <dependencies>
        <!--BDU-->
        <dependency>
            <groupId>es.ja.csalud.sas.componentescomunes.bduapiclient</groupId>
            <artifactId>bduApiClient-jpa-impl</artifactId>
            <version>x.x.x</version>
        </dependency>
...
```

- Conector JAX-WS

```
        ...
    <dependencies>
        <!--BDU-->
        <dependency>
            <groupId>es.ja.csalud.sas.componentescomunes.bduapiclient</groupId>
            <artifactId>bduApiClient-ws-impl</artifactId>
            <version>x.x.x</version>
    </dependency>
...
```

Configuración

Según sea la solución tecnológica utilizada la configuración de la infraestructura necesaria para interaccionar con BDU es diferente, aunque ambas soluciones basadas en SasConfiguration.

JAX-WS

Para que la librería se autoconfigure mediante la librería sasconfiguration, se tendrán que realizar las siguientes acciones:

- Añadir la siguiente dependencia:

```
        ...
    <dependencies>
        <!--BDU-->
        <dependency>
            <groupId>es.ja.csalud.sas.componentescomunes.bduapiclient</groupId>
            <artifactId>bduApiClient-config</artifactId>
            <version>x.x.x</version>
        </dependency>
...
```

- Agregar las dependencias necesarias de la librería sasconfiguration
- definir los siguientes parámetros:
    - es.ja.csalud.sas.componentescomunes.bduapiclient.config.url | Se configura el valor de la url dónde se encuentra el WS de BDU.

- es.ja.csalud.sas.componentescomunes.bduapiclient.config.modulecode | Se configura el código maco del módulo que representa la aplicación.

De forma alternativa se podrán definir manualmente estas configuraciones. Estos parámetros se definirán usando la librería sasconfiguration.(SAS configuration)

Actualmente existe un objeto de configuración:

- **`BduApiConfig`** gestiona la configuración general de la librería.

  Ejemplo:

```
        ...
@Produces public BduApiConfig getBduApiConfig() {
        EnumMap<BduEndPointType, String> endPoints = new EnumMap<BduEndPointType, String>
(BduEndPointType.class);
        endPoints.put(BduEndPointType.SOLICITUD, "/wsBDU.ashx");
        final BduApiConfig bduApiConfig = new BaseBduApiConfig("http://servicios.pre.sas.junta-andalucia.
es/usuario", endPoints);
        bduApiConfig.setModuleCode("CÓDIGO_EN_MACO_MODULO_SISTEMA");
        return bduApiConfig;
 }
...
```

Para su uso mediante la Factoría, se deberá de crear la factoría pasándole al constructor como parámetro la configuración de la librería.

```
        ...
public BduFactory getBduFactory() {
 return new BduFactory(getBduApiConfig());
}
...
```

JPA

La configuración del conector JPA espera una clase del tipo "BduApiConfig" propia de JPA, debido a las características propias de las impl de ws y jpa se mantendrán separadas las dos necesidades, atendiendo al principio de segregación de responsabilidades, y permitiendo la coexistencia de varias implementaciones de los servicios expuestos en caso de ser necesario.

Básicamente define una serie de timeouts para los servicios integrados, ya que el resto de configuración necesaria para el datasource se realizará como es habitual en un proyecto JPA mediante su "persistence. xml" correspondiente en el proyecto de su aplicación, así como el mapeo de tablas a entidades.

Operación

Criterias

El objeto ***QuerysBduUserSearchDto*** , es la representación de los criterios de búsqueda que son soportados por el api client. Dicho objeto permite los siguientes valores.

- **Flags,** estos son los modificadores soportados:
    - *detailedInfo*: Booleano indicando si se desea la información detallada (true) o resumida (false). Default: false.
    - *famarciaInfo*: Booleano indicando si se desea la información detallada sobre farmacia. Default: False
    - *fullAddress*: Booleano indicando si se desea la información detallada del domicilio del paciente. Default: False.
- **Identificadores** de usuario único
    - Nhusa: Alfanumérico.
    - codSns: Alfanumérico.
    - socialSecurityNumber: Alfanumérico.
    - documentIdType + documentType + documentId: La tripla permite identificar un paciente por el identificador definido, en caso de no informar de forma explicita el tipo de identificador, se supone por defecto DNI.
- **Filtros** de búsqueda
    - birthDateYear: Alfanumérico.
    - name: Alfanumérico. Soporta modificadores "startWith","endsWith","contains" y "exact match".
    - lastname1: Alfanumérico. Soporta modificadores "startWith","endsWith","contains" y "exact match".
    - lastName2: Alfanumérico. Soporta modificadores "startWith","endsWith","contains" y "exact match".

Los filtros de búsqueda basados en **cadenas que soportan modificadores** se habilitan pasando al criteria el wildcard "*" y en base a su posición se comportan de un modo u otro. Las restricciones impuestas para la longitud de las cadenas entrantes como parámetros son dependientes

- **startWith**: "*<search_string>"
- **endsWith**: "<search_string>*"
- **contains**: "*<search_string>*"
- **exact match**: "<match_string>"

Es obligatorio informar al menos uno de los siguientes campos para poder hacer uso del criteria, en caso de no informarse se dará una "UnsupportedOperationException" al operar el servicio.

- Un identificador.
- Una lista de nhusas con al menos un elemento.
- Una tupla de nombre,apellido1 y apellido2 en cualquiera de sus posibles combinaciones.

Métodos de servicio

Los métodos de servicio que la API proporciona son los siguientes:

Búsqueda de usuarios (consulta detallada/corta)

La búsqueda de usuarios se realizará mediante los métodos "findUsers" que soportan el criteria descrito anteriormente. Adicionalmente para el conector JPA dichos métodos soportan la indicación de valores para la paginación ó la definición de un valor máximo para el número de resultados devueltos.

Los métodos sin paginación se han marcado para su retirada por obsolescencia y serán retirados a partir de la versión 3.0 del api client. La información devuelta para los métodos paginados devolverán en lugar de una lista de pacientes un objeto de tipo "Page" de pacientes.

Una cuestión importante a tener en cuenta es que el conector basado en JAX-WS no dispone de soporte de paginación ni de máximo número de resultados devueltos actualmente debido a las limitaciones de los contratos de interoperación, y en caso de hacer uso de dichos métodos de servicio se elevarán excepciones fuera de la capa de servicio.

Búsqueda de recién nacidos

La búsqueda de recién nacidos nos devolverá una lista de usuarios que están asociados al nhusa indicado y también tienen soporte para la paginación.

Modificación de usuarios:

La librería proporciona cuatro métodos especializados para la modificación de datos de pacientes existentes en el BDU, cada uno correspondiente a un servicio específico del Sistema de Información Diraya. Todos los métodos requieren un usuario MACO (`petitionerUser`) con permisos adecuados y devuelven el paciente actualizado confirmado por el BDU.

Modificación de datos básicos (MODUSU01)

El método `updatePatientBasicData` recibe un objeto `UpdateBduUserBasicDataDto`. Y permite modificar información administrativa fundamental del paciente.

Modificación de domicilio y datos de contacto (MODUSU02)

El método `updatePatientAddress` recibe un objeto `UpdateBduUserAddressDto`. Y permite actualizar la dirección de residencia y la información de contacto del paciente.

Modificación de datos de control (MODUSU03)

El método `updatePatientControlData` recibe un objeto `UpdateBduUserControlDataDto`. Y permite modificar la situación administrativa y fechas de control del paciente.

Modificación de datos de desplazamiento (MODUSU04)

El método `updatePatientDisplacement` recibe un objeto `UpdateBduUserDisplacementDto`. Y permite registrar o modificar la información de desplazamiento temporal del paciente.

Todos los métodos de servicio están preparados para elevar excepciones internas encapsuladas en una excepción de tipo *BduApiClientException,* sin embargo las implementaciones finales de los conectores a los datasource (JAX-WS ó JPA) son los que disponen de los valores de categorización y tipificación de la excepción.

Paginación

Conector JPA

El conector JPA dispone de características internas para paginar sobre el número de resultados devueltos, además es posible especificar conjuntos de valores especiales que permiten conocer el total del registros del dataset que puede ser devuelto por la búsqueda al definir ambos valores de "page" y "count" a cero.

La implementación actual lo que espera por parte de los desarrolladores es que se haga un uso razonable de los servicios de búsqueda, haciendo uso de las características de paginación cuando sea apropiado, y consultado la información detallada sólo en aquellos casos en los que una navegación en profundidad sea necesaria.

Por otro lado **el conector JPA no soporta la devolución de información detallada paginada** po r cuestiones de rendimiento, devolviendo una excepción en caso de un uso inadecuado.

En el caso de hacer uso de las búsquedas sin soporte para paginación los límites establecidos para los contratos del servicio SOA CCUSU01 serán los aplicables, actualmente 50 registros.

Conector JAX-WS

El conector JAX-WS no tiene soporte a paginación y su limitación sobre el número de resultados devueltos será la aplicable definida en los contratos de operación del servicio CCUSU01.

En caso de operar los servicios con paginación haciendo uso de este conector, se elevarán excepciones hasta la capa de servicio.