UNIVERSIDAD ORT
Uruguay

Bernard Wand-Polak
Cuareim 1451
11.100 Montevideo, Uruguay
Tel. 2902 15 05 Fax 2908 13 70
www.ort.edu.uy

EVALUACIÓN	Obligatorio	**GRUPO**	M7A,M7B, N7A	**FECHA**	23-abr-2019
MATERIA	Arquitectura de Software				
CARRERA	ID-AN				
CONDICIONES	- Puntos: Máximo: 35 Mínimo: 15 - Fecha máxima de entrega: 26-Junio-2019 LA ENTREGA SE REALIZA EN FORMA ONLINE EN ARCHIVO NO MAYOR A 40MB EN FORMATO ZIP, RAR O PDF. **IMPORTANTE:** - Inscribirse - Formar grupos de hasta dos personas. - Subir el trabajo a Gestión antes de la hora indicada, ver hoja al final del documento: “RECORDATORIO”				

Bernard Wand-Polak
Cuareim 1451
11.100 Montevideo, Uruguay
Tel. 2902 15 05 Fax 2908 13 70
www.ort.edu.uy

Descripción del problema

La empresa de software *FlightData* dedicada a vender servicios de análisis de datos desea lanzar un producto que brinde a las aerolíneas de USA diversidad de datos sobre los vuelos domésticos entre ciudades del país.

Estos datos son recolectados por la Autoridad de Aviación (*AvAuth*) y actualmente está desarrollando un conjunto de servicios que publican información sobre los vuelos, aerolíneas y aeropuertos para quien desee consumirla.

El objetivo de *FlightData* es brindar un servicio para que las empresas, en base a los datos recabados por la Autoridad de Aviación, puedan obtener información sobre la puntualidad de los vuelos, las causas de demora, si hubieron desvíos a otros aeropuertos, etc. y de esta forma de mejorar el servicio que prestan sus clientes.

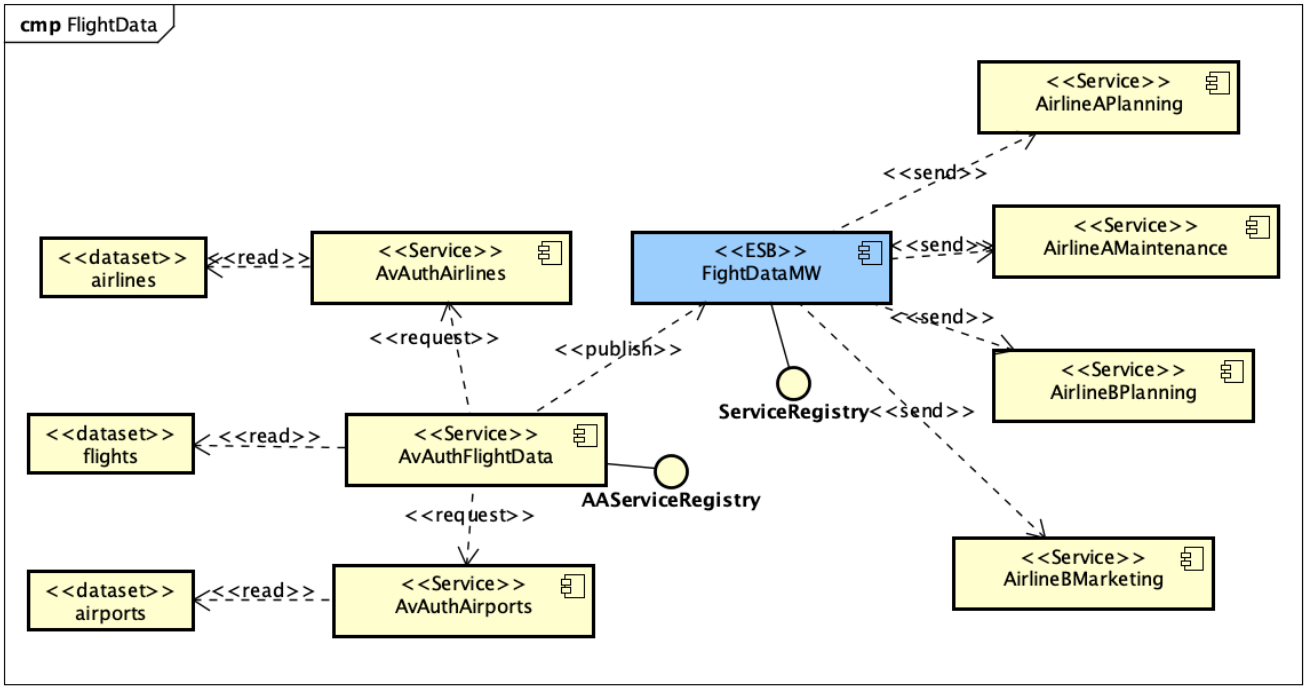
El sistema que se quiere desarrollar es un middleware el cual permite que los distintos sistemas de las empresas de aviación se conecten al mismo para recibir la información que publica la autoridad de aviación. Para ello el middleware debe proveer un conjunto de APIs que permita que los sistemas de las aerolíneas se registren para recibir los datos que necesitan. A modo de ejemplo, los sistemas de mantenimiento de las aerolíneas podrán recibir los datos y motivos de cancelación de sus vuelos, los sistemas de planificación de vuelos los datos sobre la puntualidad de partida y arribo, etc.

Debido a que los distintos sistemas de las aerolíneas requieren determinados datos y en determinados formatos, parte de las responsabilidades del middleware será la de transformar la información que producen los servicios de la autoridad de aviación para que se adecue a las necesidades de los sistemas de las aerolíneas.

Debido a que los servicios de la autoridad de aviación aún no están disponibles y para poder desarrollar la prueba de concepto del producto se decidió emular sus servicios utilizando una base de datos histórica y pública de los vuelos internos registrados en Estados Unidos durante el año 2015 https://www.kaggle.com/usdot/flight-delays

El siguiente diagrama muestra el concepto de la solución a alto nivel.

Bernard Wand-Polak
Cuareim 1451
11.100 Montevideo, Uruguay
Tel. 2902 15 05 Fax 2908 13 70
www.ort.edu.uy



A continuación se describen los requerimientos para los distintos elementos de la solución.

Servicios provistos por al Autoridad de Aviación

Debido a que estos servicios aún no se encuentran disponibles, y al ser necesario contar con ellos para probar el middleware, se espera que se implementen versiones simples de los mismos en base a los siguientes requerimientos.

NOTA: Es importante tener en cuenta que los datos que retornan o envían estos servicios debe ser exactamente los que se encuentran en las fuentes de datos asociadas, con algunas adiciones que se describen más adelante.

Servicio: AvAuthAirports

REQ1 - Obtener datos del aeropuerto

El servicio debe retornar el nombre del aeropuerto en base a la Id del mismo.

Los datos que retornan son los especificados en la siguiente fuente
https://1drv.ms/u/s!Aoos6SsR2UdQk8151lnCAjXWCBSIHg

Servicio: AvAuthAirlines

REQ1 - Obtener datos de la aerolínea

El servicio debe retornar el nombre de la Aerolínea en base a la Id del misma.

Los datos que retornan son los especificados en la siguiente fuente https://1drv.ms/x/s!Aoos6SsR2UdQk8137sna0YCPmp5kdg

Bernard Wand-Polak
Cuareim 1451
11.100 Montevideo, Uruguay
Tel. 2902 15 05 Fax 2908 13 70
www.ort.edu.uy

Servicio: AvAuthFlightData

Este servicio emulará el servicio de la Autoridad de Aviación que publicará la información sobre los vuelos. Debido a que aún no se sabe la frecuencia de publicación que utilizará la Autoridad de Aviación, la implementación que se realice debe permitir especificar la cantidad y frecuencia de registros a publicar. De esta forma se podrá emular distintas cargas de envío de datos.

REQ 1 - Registro de servicios cliente

El servicio permitirá que los servicios cliente registren la API REST (utilizando AAServiceRegistry) mediante la cual se enviaran los datos. En el caso de *FlightData* será la interfaz del middleware dedicada a recibir la información de vuelos.

REQ 2 - Envio de rafagas de datos

El servicio deberá recibir parámetros sobre los lotes de datos a enviar. Para brindar flexibilidad en las pruebas los parámetros de envío se leerán de un archivo de texto con registros indicando:

- Cantidad de lotes de datos a enviar - cuando este parámetro tiene el valor 0 se deberá enviar todos los registros de la fuente de datos en lotes con la cantidad de registros indicados por el siguiente parámetro (Tamaño del lote).

  Si el parámetro es mayor que cero indica la cantidad de lotes que se deben enviar, donde cada lote contiene la cantidad de registros indicados por el siguiente parámetro (Tamaño del lote).
- Tamaño del lote - número de registros a enviar juntos.
- Offset de comienzo de lectura del primer lote en la fuente de datos. En caso que la cantidad de lotes sea 0 este parámetro se ignora debido a que se envían todos los registros de la fuente de datos. En caso que la cantidad de lotes sea mayor que 0 este parámetro indica el número de registro de la fuente de datos desde donde se debe comenzar a leer.

Una vez leídos los parámetros el servicio envía mediante las APIs registradas los datos solicitados y repetirá el proceso hasta haber consumido todos los registros en el archivo de parámetros.

Los datos que se envían son los especificados en la siguiente fuente https://1drv.ms/u/s!Aoos6SsR2UdQk84HvALR26KLKVj7_g
a los que se debe agregar previo a enviarlos los nombres de las aerolíneas y los datos de los aeropuertos que proveen los servicios *AvAuthAirlines* **y** *AvAuthAiports***.** A su vez se debe anexar un timestamp indicando el momento del envío del registro.

Bernard Wand-Polak
Cuareim 1451
11.100 Montevideo, Uruguay
Tel. 2902 15 05 Fax 2908 13 70
www.ort.edu.uy

FlightDataMW

Este servicio provee mecanismos para integrar los servicios de la Autoridad de Aviación y los servicios de las aerolíneas, registrados en tiempo de ejecución. Una de las particularidades de este servicio es que permitirá procesar los datos recibidos (por ejemplo, validaciones y transformaciones) previo a enviarlos a los servicios de las aerolíneas mediante las APIs registradas. El servicio también permitirá asociar "disparadores" que permitirán seleccionar qué datos se envían a los servicios registrados por las aerolíneas.

REQ 1 - Registro de Interfaz de servicio

Para cada servicio provisto por las aerolíneas se debe poder:

- Registrar una “interfaz de servicio” que permita acceder a su funcionalidad.
- Especificar propiedades sobre la forma de usar cada interfaz (por ejemplo, el formato de las estructuras de datos que se intercambian por la interfaz, o el tipo de comunicación).
- Especificar ante qué valor o expresión en base a valores de los datos del servicio *AvAuthFlightData* se debe invocar la interfaz ("valores disparadores"). Por ejemplo, un servicio de aerolíneas dedicado a la planificación de vuelos podrá registrarse para recibir los datos de los vuelos cancelados o cancelados por determinado motivo.
- Especificar los datos que el servicio desea recibir de los provistos por la fuente de datos del servicio *AvAuthFlightData*.
- Especificar las validaciones y transformaciones que se deben realizar previo a enviar los datos.

Para cualquier consumidor de servicios el middleware debe poder:

- Localizar una interfaz de un servicio registrado
- Facilitar la invocación de las funcionalidades del servicio a consumir

REQ 2 - Envío de datos a servicios registrados

Ante la recepción de los datos enviados por *AvAuthFlightData* y para cada servicio de aerolíneas registrado el servicio deberá:

- seleccionar los datos a enviar en base a los datos que el servicio espera recibir y a los valores "disparadores" del mensaje.
- Realizar las validaciones y transformaciones especificadas para la interfaz a invocar. En caso de que se registren valores inválidos se debe registrar el evento de que el registro contiene datos erróneos y el registro no se debe enviar.

Bernard Wand-Polak
Cuareim 1451
11.100 Montevideo, Uruguay
Tel. 2902 15 05 Fax 2908 13 70
www.ort.edu.uy

- Enviar los datos mediante la interfaz de servicio registrada. Se debe asegurar que cada aerolínea reciba solamente datos de sus vuelos.

Para la prueba de concepto se requiere el envió de datos para al menos dos servicios distintos de una aerolínea y que para cada servicio se realice al menos una transformación y una validación en base a un "valor disparador" distinto para cada servicio. Ver apartado Servicios Provistos por las Aerolíneas.

REQ 3 - Gestión de errores y fallas

El sistema debe proveer suficiente información que permita conocer el detalle de las tareas que se realizan. En particular, en el caso de ocurrir una falla o cualquier tipo de error, es imprescindible que el sistema provea toda la información necesaria que permita a los administradores hacer un diagnóstico rápido y preciso sobre las causas. Se espera que la solución contemple la posibilidad de poder cambiar las herramientas o librerías concretas que se utilicen para producir esta información, así como reutilizar esta solución en otras aplicaciones, con el menor impacto posible en el código.

REQ 4 - Autenticación de operaciones

Se requiere que toda interacción desde *FlightDataMW* hacia los servicios receptores de datos esté autenticada.

REQ 5 - Manejo de carga

Debido a que el servicio *AvAuthFlightData* envía "Rafagas de datos" es importante que *FlightDataMW* asegure que no se pierdan datos debido a una pobre capacidad de procesamiento. Se deberá lograr la mayor capacidad de procesamiento posible sin pérdida de datos y logrando la mejor latencia posible. Es parte de la prueba de concepto determinar la capacidad máxima de procesamiento que puede manejar *FlightDataMW* manteniendo una variación de latencia razonable.

REQ 6 - Incorporación de nuevos procesamientos de datos

El diseño deberá permitir incorporar nuevas validaciones y transformaciones a los datos que envía a los servicios de las aerolíneas con el menor costo posible y de forma de reusar las mismas.

REQ 7 - Cambio en las interfaces de los servicios de las aerolíneas

Los cambios en los tipos de interfaces, los datos que se reciben, y las validaciones y transformaciones requeridas por los servicios de las aerolíneas deben de tener el menor impacto posible en el middleware, en lo posible se quiere poder incorporar los cambios sin necesidad de modificar el código del middleware.

Bernard Wand-Polak
Cuareim 1451
11.100 Montevideo, Uruguay
Tel. 2902 15 05 Fax 2908 13 70
www.ort.edu.uy

Servicios provistos por las aerolíneas

Para poder probar el funcionamiento de *FlightDataMW* se espera que se implementen dos servicios que reciban los datos del middleware. Estos servicios solo deben registrar la información recibida para poder visualizarla y verificar el correcto funcionamiento de *FlightDataMW.* A los datos recibidos se les debe anexar el timestamp de recepción y mostrar la diferencia entre el timestamp de envío y el de recepción, para poder de esta forma visualizar la latencia del mensaje.

A continuación se describen dos implementaciones para probar.

ServicioDeManenimientoAerolinea

Este servicio expone una API REST para recibir la información - en formato JSON - sobre la cancelación de vuelos. El middleware deberá enviar los datos de los vuelos de esta aerolínea realizando en base a los siguientes criterios, validaciones y transformaciones:

- Se deben enviar los datos cuando el campo "CANCELLED" tenga el valor 1 = cancelled.
- Los campos que debe enviar son:
    - TIMESTAMP, DATE, AIRLINE, FLIGHT_NUMBER, ORIGIN_AIRPORT (Airport name), DESTINATION_AIRPORT (Airport name), SCHEDULED_DEPARTURE, CANCELLED, CANCELLATION_REASON
- Las transformaciones que debe realizar son:
    - el campo DATE debe convertirse a un campo con el formato (dd/mm/aaaa) y consolidarse a partir de los datos YEAR, MONTH, DAY de la fuente de datos.
    - el campo CANCELLED deberá convertirse a true o false
    - el campo CANCELLATION_REASON debe convertirse al motivo en formato texto: A - Airline/Carrier; B - Weather; C - National Air System; D - Security
- Se debe validar los siguientes casos
    - el campo DATE debe contener una fecha válida.
    - el campo CANCELLED debe contener valores 0 o 1.
    - el caso que CANCELLED sea 1 el campo CANCELLATION_REASON no debe ser vacío.

ServicioDePlanificacionAerolinea

Este servicio expone una API REST - en formato XML -para recibir la información sobre la cancelación de vuelos. El middleware deberá enviar los datos de los vuelos de esta aerolínea realizando en base a los siguientes criterios, validaciones y transformaciones:

Bernard Wand-Polak
Cuareim 1451
11.100 Montevideo, Uruguay
Tel. 2902 15 05 Fax 2908 13 70
www.ort.edu.uy

- Se deben enviar los datos cuando el campo "DEPARTURE_DELAY" sea mayor o igual a -10 y 10
- Los campos que debe enviar son:
    - TIMESTAMP, DATE, AIRLINE, FLIGHT_NUMBER, ORIGIN_AIRPORT (Airport name), DESTINATION_AIRPORT (Airport name), SCHEDULED_DEPARTURE, DEPARTURE_DELAY, DEPARTURE_TIME, SCHEDULED_ARRIVAL, ARRIVAL_TIME, DELAY_REASON
- Las transformaciones que debe realizar son:
    - el campo DATE debe convertirse a un campo con el formato (dd/mm/aaaa) y consolidarse a partir de los datos YEAR, MONTH, DAY de la fuente de datos.
    - El campo DELAY_REASON debe ser un string que especifica las razones de la demora en base a que los campos AIR_SYSTEM_DELAY,SECURITY_DELAY, AIRLINE_DELAY, LATE_AIRCRAFT_DELAY, WEATHER_DELAY tengan un valor.
- Se debe validar los siguientes casos
    - el campo DATE debe contener una fecha válida.
    - los campos DEPARTURE_DELAY, DEPARTURE_TIME no deben ser vacíos.

Condiciones generales del trabajo obligatorio

1. Restricciones

- La implementación del backend debe desarrollarse en NodeJS utilizando las tecnologías vistas en el curso. Es opcional y abierta la elección de packages que puedan ayudar al desarrollo, teniendo que justificar la elección en la documentación.
- Todo el código fuente, documentación, archivos de configuración o cualquier otro artefacto referido al desarrollo de los prototipos debe gestionarse en el repositorio Git asignado en la Organización de Github del curso.

2. Objetivo del trabajo

El objetivo de este obligatorio es:

- Diseñar y construir una arquitectura de software que pueda ser demostrada en la defensa.
- Producir un documento de arquitectura que sirva para explicar las estructuras diseñadas en función de los atributos de calidad relevantes que se hayan identificado.

Se pueden proveer datos de prueba precargados en la base de datos con el fin de facilitar la demostración.

3. Entregables

- Documento con la descripción de la arquitectura diseñada siguiendo el modelo **Views&Beyond** tratado en el curso. **Prestar especial atención** a los criterios de corrección detallados más adelante. La entrega de la documentación deberá realizarse mediante el sistema de entregas gestion.ort.edu.uy
- Código fuente en repositorio Git con la implementación de los requerimientos solicitados.

Bernard Wand-Polak
Cuareim 1451
11.100 Montevideo, Uruguay
Tel. 2902 15 05 Fax 2908 13 70
www.ort.edu.uy

- **[opcional]** Guías de instalación, configuración o uso, archivos de configuración o datos, para poder ejecutar la aplicación (pueden facilitarse también en el mismo repositorio Git del código).

4. Consultas

- Todas las consultas deben realizarse mediante el foro creado para este propósito. No se deben enviar consultas a las cuentas de correo personales de los docentes.
- Se intentará responder a las consultas en el menor tiempo posible, pero en ocasiones se puede esperar una demora de hasta 48 horas en responder.
- No se responderán preguntas durante las últimas 48 horas previas a la entrega.

5. Criterio de corrección

La evaluación del obligatorio se divide en aspectos teóricos y prácticos. Para cada uno de ellos se aplican determinados criterios de corrección. A continuación se muestra la distribución de puntajes para cada aspecto, seguida del detallado de cada aspecto.

Funcionalidad	30%
Requerimientos no funcionales	25%
Documentación	20%
Calidad del diseño	10%
Calidad del código y control de versiones	10%
Demostración al cliente	5%
TOTAL	**100%**

5.1. Funcionalidad

En este apartado, se identifican los siguientes criterios de corrección:

- Se implementaron sin errores todos los requerimientos funcionales propuestos.
- Se implementaron todos los requerimientos funcionales propuestos pero se detectan errores menores que no afectan el uso normal del sistema.
- Se implementaron los principales requerimientos funcionales, aunque no todos, pero se detectan errores menores que no afectan el uso normal del sistema.
- Se implementaron los principales requerimientos funcionales, aunque no todos, pero se detectan errores que afectan el uso normal del sistema.
- Los requerimientos funcionales implementados son básicos y/o se detectan errores que afectan el uso normal del sistema.

Bernard Wand-Polak
Cuareim 1451
11.100 Montevideo, Uruguay
Tel. 2902 15 05 Fax 2908 13 70
www.ort.edu.uy

5.2. Requerimientos no funcionales

Para cada RNF propuesto o identificado se aplican las siguientes prioridades en la corrección:

- Se implementaron satisfactoriamente tácticas adecuadas para cada RNF y se verifica su funcionamiento.
- Se implementaron las tácticas adecuadas al RNF pero no se verifica su funcionamiento.
- Se implementaron parcialmente las tácticas adecuadas al RNF.
- Las tácticas implementadas no son las adecuadas para el RNF.
- No se implementó ninguna táctica para satisfacer el RNF.

5.3. Documentación

Una documentación aceptable cumple con lo siguiente:

- Incluye al menos una vista de tipo Módulos, una de tipo Componentes y Conectores y una de tipo Asignación.
- Cada vista está documentada en base a la guía vista en el curso (representación primaria, catálogo de elementos, justificaciones de diseño, guías de variabilidad)
- Los diagramas se realizan en UML 2 y su uso es correcto.
- Las justificaciones de las decisiones de diseño tomadas son relevantes desde el punto de vista arquitectónico y permiten entender el “por qué” del diseño (y no el “qué hace” o el “cómo está hecho”).

5.4. Calidad de diseño

Un diseño aceptable cumple lo siguiente:

- La paquetización del código representa la descomposición lógica (módulos) de la aplicación.
- Las estructuras y comportamientos documentados sirven como guía para la comprensión del código implementado.
- Las tácticas y patrones arquitectónicos se implementan correctamente a partir de su objetivo y teniendo en cuenta sus ventajas y desventajas para favorecer o inhibir atributos de calidad.
- Se gestionan los posibles errores en todos los casos.

5.5. Calidad del código y control de versiones

Un código de calidad aceptable cumple lo siguiente:

- Cumple las convenciones de codificación de NodeJS (https://docs.npmjs.com/misc/coding-style)
- La gestión del código del obligatorio debe realizarse utilizando el repositorio Git de Github, apoyándose en el flujo de trabajo recomendado GitFlow (nvie.com/posts/a-successful-git-branching-model).
- Luego de la entrega se le debe dar acceso a los docentes al repositorio. (solo es necesario darle acceso a los dos docentes de su dictado)

5.6. Demostración al cliente

Cada equipo deberá realizar una demostración al cliente de su trabajo. Dentro de los aspectos que un cliente espera se encuentran:

- Ver la demo cuando él esté listo y no tener que esperar a que el equipo se apronte.
- Probar la solución y que la misma funcione sin problemas o con problemas mínimos, y de buena calidad de interfaz de usuario.

Bernard Wand-Polak
Cuareim 1451
11.100 Montevideo, Uruguay
Tel. 2902 15 05 Fax 2908 13 70
www.ort.edu.uy

- Conocer las capacidades de cada uno de los integrantes del equipo, pudiendo preguntar a cualquier integrantes sobre la solución, su diseño, el código y sobre cómo fue construida, y así apreciar que fue un trabajo en equipo. Todos los integrantes deben conocer toda la solución.
- Verificar el aporte individual al trabajo por parte de cada uno de los integrantes del equipo y en función de los resultados, se podrán otorgar distintas notas a los integrantes del grupo. Se espera que cada uno de los integrantes haya participado en la codificación de parte significativa del obligatorio.

Esta demostración al cliente hará las veces de defensa del trabajo.

NOTA: El incorrecto funcionamiento de la instalación puede significar la no corrección de la funcionalidad. En el caso de la demostración al cliente sea en el laboratorio, cada grupo contará con 15 minutos para la instalación de la aplicación. Luego de transcurridos los mismos el cliente comenzará a molestarse impactando en su percepción sobre el trabajo. A su vez el cliente podrá realizar preguntas a cualquiera de los integrantes las cuales podrán afectar la valoración que hace de cada integrante del equipo.

Bernard Wand-Polak
Cuareim 1451
11.100 Montevideo, Uruguay
Tel. 2902 15 05 Fax 2908 13 70
www.ort.edu.uy

RECORDATORIO: IMPORTANTE PARA LA ENTREGA

➢ **Obligatorios** (Cap.IV.1, Doc. 220)

La entrega de los obligatorios será en formato digital online, a excepción de algunas materias que se entregarán en Bedelía y en ese caso recibirá información específica en el dictado de la misma.

Los principales aspectos a destacar sobre la **entrega online de obligatorios** son:

1. La entrega se realizará desde **gestion.ort.edu.uy**
2. Previo a la conformación de grupos cada estudiante deberá estar inscripto a la evaluación. S**ugerimos realizarlo con anticipación.**
3. **Uno de los integrantes del grupo de obligatorio será el administrador del mismo** y es quien formará el equipo y subirá la entrega
4. Cada equipo debe entregar **un único archivo en formato zip o rar** (los documentos de texto deben ser pdf, y deben ir dentro del zip o rar)
5. El archivo a subir debe tener **un tamaño máximo de 40mb**
6. Les sugerimos **realicen una 'prueba de subida' al menos un día antes**, donde conformarán el **'grupo de obligatorio'.**
7. La **hora tope para subir el archivo será las 21:00** del día fijado para la entrega.
8. La entrega se podrá realizar desde cualquier lugar (ej. hogar del estudiante, laboratorios de la Universidad, etc)
9. Aquellos de ustedes que presenten alguna dificultad con su inscripción o tengan inconvenientes técnicos, por favor pasar por la oficina del Coordinador o por Coordinación adjunta **antes de las 20:00hs.** del día de la entrega

Si tuvieras una situación particular de fuerza mayor, debes dirigirte con suficiente antelación al plazo de entrega, al Coordinador de Cursos o Secretario Docente.