De Pauta I2 2025-1

1. Pregunta UML 1 (30 ptos)

Una de las grandes zonas de investigación de los últimos años ha sido el espacio y la posible búsqueda de planetas que contengan vida. Con esto en mente, y gracias a sus investigaciones sobre esto y el ramo de Arquitectura de Sistemas de Software, los contactaron del Servicio Secreto de Investigaciones Planetarias (SSIP abreviado) para entrar a su equipo y ayudarles a diseñar un nuevo sistema para realizar sondeos en otros planetas.

Para esto, con el equipo diseñaron un sistema en donde tendrán una unidad central en el espacio que capte puntos de interés en otros planetas, y puedan recopilar imágenes de estos lugares mediante millones de sondas, baratas y desechables, que compartirán su ubicación en todo momento. Es así, como les piden diseñar un diagrama inicial del sistema teniendo en consideración que debe estar funcionando correctamente, independiente de la cantidad de sondas a enviar y de las dificultades de transmisión que se puedan obtener.

El sistema tendría como componente central un satélite con el servidor central en donde se recibiría la información de todas las sondas. Sus funcionalidades principales deben ser: poder recibir la ubicación de las sondas, recibir información desde distintos satélites de la tierra con puntos de interés de otras zonas del espacio, aceptar conexiones de las distintas sondas, recibir las imágenes para almacenarlas hasta que las vayan a recuperar en persona y dirigir a las sondas en el espacio para que lleguen a su ubicación.

Por otro lado, las sondas espaciales saldrán de la Tierra y flotarán en el espacio hasta recibir una zona de interés a la que dirigirse, tomar las imágenes y luego ser desechadas en el espacio. En general sus funcionalidades deberían ser: poder buscar y conectarse a la unidad central sabiendo su posición en el espacio y dónde debiese estar el satélite principal, enviar su ubicación constantemente, poder tomar fotografías al detectar que encontraron el punto de interés, recibir las indicaciones de la unidad central para poder dirigirse, y enviar las imágenes hasta recibir una respuesta que fue enviada correctamente para luego apagarse para siempre en el frio espacio.

Puede suponer que tiene un equipo lo suficientemente grande para desarrollar todo lo que sea necesario, el SSIP se hará cargo de eso, y que las señales por el espacio pueden viajar como si fueran por la red, pero asumiendo la latencia por distancia.

1.a) (2 ptos) Dada la gran carga de datos que se deben estar transmitiendo y recibiendo por parte de la unidad central, les piden agregar en su diagrama al menos 2 medidas de mejora de escalabilidad que sean pertinentes en su sistema. Indique y justifique cada uno.

Respuesta: *Se espera una respuesta con medidas que apliquen directamente a la Unidad Central, como replicaciones internas con Load Balancers, mecanismos para disminuir la cantidad de datos que se deban transmitir e incluso mecanismos de datos para poder mantener toda la información disponible dentro del sistema, como sharding de acuerdo a las ubicaciones de las sondas.*

No son válidas respuestas que apliquen directamente a mejorar las máquinas, ya que por sí sola los sistemas no serían capaces de soportar la demanda y se requieren mecanismos dentro de la arquitectura.

Si se responde solo esta pregunta de UML se corregirá con un total de 4 puntos donde se espera además que se indiquen los problemas actuales que tendría la unidad central dentro de las justificaciones para ver que se atacan correctamente en las soluciones.

1.b) (2 ptos) Uno de los miembros de su equipo les presenta varias inquietudes sobre los problemas de latencia que podría conllevar este sistema cuando las sondas se van alejando en el espacio. Para tranquilizarlo le dice que va a pensar en al menos 1 medida para poder solventar esos problemas. Justifíquelo(s) y debe agregarlo(s) a su diagrama.

Respuesta: *Se espera una respuesta con medidas de mejora de performance vistas en clases o que apliquen al sistema. En particular se debe ver que esta es una abstracción del problema de la red pero manejada sin cables, por lo que soluciones simples que son válidas es aplicar sistemas intermediarios de comunicación que sirvan de caché o redes P2P para hacer un envío de datos sin pérdidas de latencia e información.*

Si se responde solo esta pregunta de UML se corregirá con un total de 4 puntos donde los 2 puntos restantes se espera que se indique el problema en específico que afecta la medida y el paralelismo con el problema de la red.

1.c) (1 pto) Como van a estar recibiendo datos de distintos satélites de la Tierra, cada uno de estos podría ocupar formatos distintos o valores que no existan en otros satélites. Por eso, en su equipo les proponen usar una base de datos de tipo Llave Valor para guardar la información fácilmente y no estar modificando todos los datos apenas lleguen para que entren en el schema. Indique si es buena opción, el por qué, y si no lo es indique como una mejor opción.

Respuesta: *Se espera una respuesta en donde se indique que no es buena opción por las limitaciones de ese tipo de base de datos, ya que no permitiría búsquedas en base a los valores, y por lo tanto sería mejor una base de datos de tipo documentos o una columnar muy bien justificada en base a sus capacidades de búsqueda de grandes datos de información.*

Para esta pregunta no es válido indicar que se puede usar una base de datos SQL estándar porque queremos un sistema sin schema; o de grafos ya que no nos interesa revisar las relaciones de los datos.

Si se responde solo esta pregunta de UML se corregirá con un total de 2 puntos donde el punto restante se evaluará la correcta descripción de las ventajas de el tipo de bases de datos a utilizar.

1.d) (4 ptos) Indique (mínimo 3, pero pueden ser más) atributos de calidad vistos en clases que Ud. considere son fundamentales en este problema. Enumérelos en orden de prioridad y justifique brevemente.

Respuesta:

Son válidos:

- *Performance*
- *Escalabilidad*
- *Confiabilidad*
- *Integrabilidad: Solo para conexiones con otros satélites de la Tierra*
- *Otro bien justificado*

No son válidos:

- *Mantenibilidad: No se pide manejo de errores*
- *Flexibilidad: No se pide agregar nuevas funcionalidades*
- *Soporte: No hay manejo de tickets de usuario*
- *Usabilidad: No se pide interfaz de usuario*

Si se responde solo esta pregunta de UML se corregirá con un total de 8 puntos donde los 4 puntos restantes se evaluarán en base a las correctas justificaciones del sistema. Es válido corregir un 4to NFR si el estudiante agregó un extra para distribuir el puntaje.

1.e) (5 ptos) Indique (mínimo 2, pero pueden ser más) los estilos arquitectónicos vistos en clases que va a emplear para solucionar el problema. Indique qué aspecto del problema ataca cada estilo (responsabilidad).

Respuesta:

Son válidos:

- *En Capas*
- *Microkernel*
- *Main y Subrutinas: Válido solo para las sondas si está bien justificado en base a la capacidad bloqueante, pero aprovechando el uso de todos los recursos en envío de información*
- *En lotes secuenciales*
- *Otro bien justificado*

No son válidos:

- *EDA: Los envíos de información se exigen síncronos para obtener la respuesta de que fueron correctamente recibidos.*
- *Pub/Sub: La topología de las sondas indica que sería de muchos publicadores a un subscriptor, hay mejores estilos a usar que no generan problemas de asincronía.*
- *Pizarra*
- *Basado en reglas*

Si se responde solo esta pregunta de UML se corregirá con un total de 10 puntos donde los 5 puntos restantes se evaluarán en base a las correctas justificaciones del sistema. Es válido corregir un 3er estilo si el estudiante agregó un extra para distribuir el puntaje, pero debe verse reflejado en el diagrama.

1.f) (12 ptos) Confeccione un diagrama de componentes UML indicando los componentes arquitectónicos y sus relaciones, además de sus estilos especificados previamente, tal que reflejen una solución al problema planteado. Añada tanto detalle como haga falta para que quede claro cómo soluciona el problema planteado. Incluya anotaciones explicando funcionamiento y supuestos, así como nombres de interfaces.

***Respuesta:** Si no es un diagrama de componentes UML no se corrige. Como puede haber muchas soluciones al problema, se analizará en base a descuentos:*

- *(-3 c/u) No usan los estilos o el componente que envía los datos que justificaron previamente*
- *(-2 c/u) Falta un requisito funcional*
- *(-1 c/u) Interfaces mal implementadas - Máximo de descuento de 8 puntos.*
- *(-1 c/u) Componentes desagregados (sin conectores)*
- *(-1 c/u) No hay nombres en interfaces - Máximo de descuento de 4 puntos.*

Como la respuesta variará en base a lo indicado en preguntas previas no hay diagrama de referencia por esta vez.

Si se responde solo esta pregunta de UML se corregirá con un total de 24 puntos donde cada descuento del diagrama descontará 1 punto extra y no habrá límite de descuento por interfaces mal implementadas.

1.g) (4 ptos) Identifique 2 trade-off originados de la solución de su diagrama. Comente como las solucionaría y las implicancias de estas soluciones. No se corregirán aquellas que apunten al formato de cómo se diagramó o si falto explotar algún componente

***Respuesta:** Se esperan trade-off asociados a su solución, puede ser por una decisión de implementar un estilo u herramienta o por el mismo diseño de su sistema. Sobre la distribución de puntaje:*

- *(1pto) Identificar un trade-off correctamente*
- *(1pto) Identificar correctamente una posible solución aplicable a su sistema*

Si se responde solo esta pregunta de UML se corregirá con un total de 8 puntos donde los 4 ptos restantes se distribuirán en base al tipo de argumento que diga el alumno por cada trade-off.

2. Pregunta UML 2 (30ptos)

El cálculo correcto de impuestos a las personas es esencial para evitar casos indeseables sobre los contribuyentes, como dobles cobros, cobros por ingresos no existentes o posibles declaraciones erróneas. Para ayudar con este proceso, el servicio de recaudación al contribuyente (SRC) le pide que modele una versión inicial de un reporte automatizado el cual recoge información de diversos sistemas con datos del contribuyente y se le propone como sugerencia al contribuyente, el cual puede aceptarla o no. Cada uno de estos sistemas tiene las siguientes características

- Sistemas previsionales: Estos sistemas tienen el registro de cotizaciones de cada empleado dependiente. Son sistemas que suelen tener un performance y disponibilidad decente, pero sufren de degradación de servicio en peaks de demanda (e.g. fin de mes, de año)
- Sistemas ERP de retenedores de impuestos: Entidades que retienen los impuestos de emisores de boletas, que luego se cuentan en los cálculos del SRC cuando se les piden datos de ciertos contribuyentes. Estos sistemas son rápidos y eficientes, pero cada uno posee su propia interfaz operativa, con algún protocolo propio de transmisión y presentación
- Sistemas de previsión de salud: Estas entidades registran los aportes que se les dieron para seguros de salud y registran pagos de licencias y otros. Aunque tienen una interfaz estandarizada, sufren de baja performance al operar por su falta de incentivos de modernización, y algunos requieren de uso de scrapping
- Sistemas internos del SRC: La solución tendrá acceso a todos los recursos internos de la organización, que contienen información de contribuyentes, declaraciones, propiedades muebles e inmuebles. Estos recursos (servicios y bases de datos) son fácilmente accesibles, performantes y tienen interfaz estable. Sin embargo, varios de los datos a los que se requiere acceder requieren secuencias de transacciones muy precisas entre servicios que pueden fallar repentinamente.

Diseñe una solución capaz de acceder a todos estos sistemas y generar las recomendaciones requeridas en base a llamadas a una API gRPC. Asuma que posee una librería de código simple llamada QuickCalcTaxes la cual puede calcular el formulario de declaración del contribuyente a partir de la data extraída de estos sistemas.

2.a) (4 ptos) Para cada sistema listado, proponga un plan para abordar los problemas que pueden presentarse según la descripción de estos

El alumno debe listar los 4 sistemas disponibles en el enunciado. Es importante que como cada sistema tiene desafíos distintos, estos sean abordados de forma individual por sistema. Si el alumno enumerara soluciones transversales, no suma y resta si son implementaciones erróneas.

Sistema A

Sistema B

Sistema C

Sistema D

2.b) (4 ptos) Indique (mínimo 3, pero pueden ser más) atributos de calidad vistos en clases que Ud. considere son fundamentales en este problema. Enumérelos en orden de prioridad y justifique brevemente

Principalmente deben aparecer (para tener puntaje completo deben estar):

- *Integrabilidad: Debe tener en cuenta las necesidades de los otros sistemas disponibles para consultar, lo que implica integrabilidad.*
- *Flexiblidad: Este sistema ha añadido funcionalidad a causa de varios sistemas adicionales y es esperable que esta tendencia continúe. Un sistema modular compatible con plugins o microservicios es una buena opción*
- *Confiabilidad: Impuestos requieren calculos precisos con data confiable. Aparece en el enunciado*

Secundarios: Suman puntaje parcial pero no pueden reemplazar a los anteriores (0.25 c/u)

- *Performance*
- *Escalabilidad*
- *Seguridad*

Otros no suman y restan si están mal justificados

Usabilidad –1 pto por mencionarlo

2.c) (5 ptos) Indique (mínimo 2, pero pueden ser más) los estilos arquitectónicos vistos en clases que va a emplear para solucionar el problema. Indique qué aspecto del problema ataca cada estilo (responsabilidad).

2.d) (12 ptos) Confeccione un diagrama de componentes UML indicando los componentes arquitectónicos y sus relaciones, además de sus estilos especificados previamente, tal que reflejen una solución al problema planteado. Añada tanto detalle como haga falta para que

quede claro cómo soluciona el problema planteado. Incluya anotaciones explicando funcionamiento y supuestos, así como nombres de interfaces

> ***Respuesta:*** *Debe visualizarse el componente QuickCalcTaxes. Es clave que aparezca integrado dentro de otro componente.*
>
> *Adicionalmente, debería salir una interfaz lollipop con gRPC, con un sistema de resolución de llamada antes de ese componente.* ***NO*** *debe aparecer un frontend, puesto que está fuera del scope de modelación del sistema.*
>
> *Si no es un diagrama de componentes UML digital no se corrige. Como puede haber muchas soluciones al problema, se analizará en base a descuentos:*
>
> - *(-3 c/u) No usan los estilos o el componente que envía los datos que justificaron previamente*
> - *(-2 c/u) Falta un requisito funcional*
> - *(-1 c/u) Interfaces mal implementadas*
> - *(-1 c/u) Componentes desagregados (sin conectores)*
> - *(-2) No hay nombres en interfaces*

2.e) (4 ptos) Identifique 2 trade-off originados de la solución de su diagrama. Comente como las solucionaría y las implicancias de estas soluciones. No se corregirán aquellas que apunten al formato de cómo se diagramó o si falto explotar algún componente

> ***Respuesta:*** *Se esperan trade-off asociados a su solución, puede ser por una decisión de implementar un estilo u herramienta o por el mismo diseño de su sistema. Sobre la distribución de puntaje:*
>
> - *(1pto) Identificar un trade-off correctamente*
> - *(1pto) Identificar correctamente una posible solución aplicable a su sistema*