Nombre _____

Pontificia Universidad Católica de Chile
Escuela de Ingeniería
Departamento de Ciencia de la Computación

14 de septiembre de 2018

IIC2113 – Diseño Detallado de Software

Interrogación 1

Instrucciones:

- Sea preciso: no es necesario escribir extensamente pero sí ser preciso.
- En caso de ambigüedad, utilice su criterio y explicite los supuestos que considere convenientes.
- Responda y entregue cada pregunta en hojas separadas. Si no responde una pregunta debe entregar de todas formas la hoja correspondiente a la pregunta.
- Indique su nombre en cada hoja de respuesta.
- Esta interrogación fue diseñada para durar 150 minutos.

1. (1.0 pts)

a. Explique a qué se refieren los conceptos "cohesión" y "acoplamiento" en el diseño de *software*. ¿Existe una alguna correlación entre ellos?

b. ¿Cómo afecta un *software* altamente acoplado en la extensión de este?

c. Explique qué problemas conlleva el mal uso de herencia en la programación orientada a objetos.

d. *Python* ofrece el módulo ***urllib*** que sirve para trabajar con URLs (abrir, leer, *parsear*). También, existe la librería externa ***requests*** que ofrece las mismas funcionalidades y basa su implementación en ***urllib***. Sin embargo, ***requests*** es una de las librerías con más descargas en *Python* e incluso tiene mayor popularidad que ***urllib***. Basado en un principio fundamental del diseño, comente a qué se puede deber esta situación.

Nombre _____

2. (0.8 pts)

Considere el siguiente contexto:

Mercado Público es un sitio web del gobierno de Chile donde las instituciones que dependen del Estado publican licitaciones para satisfacer sus necesidades. En ese contexto, *Chile Crece Contigo* (parte del Ministerio de Desarrollo social) ha generado una licitación para el desarrollo de un *software* de gestión interna.

Considerando el modelo 4+1, señale la vista y diagrama más atingente para presentar a los siguientes actores. Justifique su elección.

i. Analista de negocio de *Chile Crece Contigo*
ii. Jefe tecnológico de *Chile Crece Contigo*
iii. Potenciales usuarios del sistema
iv. Arquitecto de software de *Chile Crece Contigo*

Nombre _____

3. (1.2 pts)

a. Considere el siguiente extracto de código:

```ruby
class FileParser
  def initialize(client, file)
    @client = client
    @file = file
  end

  def parse
    # which format uses the client?
    case @client.file_format
      when :xml
        parse_xml
      when :csv
        parse_csv
    end

    @client.last_parse = Time.now
    @client.save!
  end

  private

  def parse_xml
    # parse xml
  end

  def parse_csv
    # parse csv
  end
end
```

Identifique el principio *SOLID* predominante que no se respeta y modifique el código para que sí lo haga.

b. Explique a qué se refiere el principio *Liskov Substitution*. Acompañe su explicación junto con un ejemplo concreto en *ruby* (o *pseudo-ruby*) que no cumpla con el principio. Luego, realice una versión modificada de su ejemplo que sí cumpla con el principio.

Nombre _____

4. (3.0 pts)

Considere el siguiente contexto:

"Alan busca cumplir el sueño de toda su vida: dedicarse a la composición de música electrónica, para ser usado como base en canciones de trap latino. A raíz de esto decidió dejar su trabajo *full-time* como *Senior SQL Injector* en Radius. Sin embargo, como este empleo no estaba bien remunerado, Alan no contaba con el dinero suficiente para adquirir un software para componer música. Esto lo llevó a la siguiente determinación: desarrollar su propio *audio editing software*, ajustado a sus necesidades musicales. Este editor —bautizado como *Droptable*— será una aplicación de escritorio, de código abierto y desarrollada en Electron. Este es un *framework* que permite crear aplicaciones multiplataforma, con el uso de tecnologías de la web: HTML, CSS y JavaScript.

Droptable debe soportar *multitracking edition*. En otras palabras, una canción estará compuesta por uno o más *tracks*, donde cada uno de ellos representa una fuente de sonido, que generalmente es un instrumento musical. El principal beneficio de este sistema es la posibilidad de grabar cada instrumento de forma aislada, para luego conseguir un sonido cohesivo con la ayuda del software. Además, cada uno de estos *tracks* está compuesto por notas musicales. Para una primera versión de Droptable, estas notas sólo tendrán un nombre (e.g. *do mayor* en piano), una duración en milisegundos, una altura (representado con una frecuencia en Hertz), una posición temporal en el *track* y un archivo de formato FLAC con el sonido.

Una funcionalidad esencial que Droptable debe ofrecer es la opción de aplicar distintos filtros y efectos como, por ejemplo, reducción de ruido, normalización, ecualización y *reverb*. Para no reinventar la rueda, la mayoría de los efectos musicales se obtendrá con el uso de la popular librería *brytiago.js*. Sin embargo, Alan deberá ajustar estas funciones de la librería para poder aplicarlas tanto en una nota en particular, como en una sección específica del *track*, o en una canción de forma completa.

Entre otros *features* que mejoran la usabilidad, Droptable debe ofrecer la posibilidad de copiar, cortar y pegar secciones de un *track*. Además, para corregir potenciales errores, debe contar con la funcionalidad de deshacer y rehacer las acciones realizadas. Junto con esto, el software debe ser capaz de importar y exportar archivos en distintos formatos de audio (*e.g.* OGG, WAV, MP3, OZUNA, FLAC), o con distintos niveles de calidad del sonido, para trabajar en un formato común.

Al crear un primer prototipo de la aplicación, Alan notó un problema: cada uno de los archivos en formato FLAC, asociados a una nota en particular, tiene un tamaño cercano a la centena de kilobytes. De esta forma, al crear un *track* de más de diez minutos, Droptable se volvía inutilizable; esto ocurría aunque se estuviese trabajando únicamente con la misma nota musical.

Nombre _____

Motivado por este inconveniente a solucionar, Alan también evaluó que, en una fase posterior al primer MVP, sería una buena idea extender el *back-end* de la aplicación para registrar métricas sobre el rendimiento de Droptable. Esto permitiría saber, por ejemplo, cuánto tiempo y memoria está tomando aplicar cada uno de los efectos y filtros. La idea es desarrollar estas funcionalidades una vez que la primera parte esté completa."

a. Identifique exactamente 5 patrones de diseño que podrían ser utilizados en la resolución del problema enunciado. Además, por cada uno de ellos justifique de forma concisa su elección, basada en los requisitos de la aplicación.
b. A partir de los patrones identificados, escoja 2 de ellos y realice un diagrama de clases *UML 2* que refleje la combinación en **el contexto de la aplicación descrita**. Se debe realizar un solo diagrama que refleje la interacción cohesiva entre ambos patrones. Su diagrama debe detallar todo lo necesario para implementar la solución, además de señalar claramente dónde se implementa cada patrón en el diagrama y explicar el rol de cada participante.