PyChrono: Una alternativa "*open source*" para la simulación de sistemas mecánicos

Luis U. Medina Uzcátegui, Instituto de Diseño y Métodos Industriales, Facultad de Ciencias de la Ingeniería, Universidad Austral de Chile, Valdivia. luis.medina@uach.cl

RESUMEN

La pandemia ha acelerado los procesos de virtualización en la docencia universitaria. Ingeniería no ha estado exenta a este fenómeno. El requerimiento de acceso y uso de aplicaciones computacionales para la simulación de sistemas sigue creciendo, y el uso de aplicaciones distribuidas bajo esquemas de licencia libre surge como una alternativa frente a las aplicaciones comerciales, cuyo acceso puede estar limitado por aspectos como: tipo de licencia comercial adquirida, velocidad de conectividad vía remota, requerimientos de hardware, costo, entre otras. Para ingeniería mecánica, y otras disciplinas, existe una diversidad, reconocida y validada, de alternativas no comerciales para la simulación de sistemas mecánicos. No obstante, aún persiste una brecha entre su uso y el de sus contrapartes comerciales, generalmente atribuida a características de interfaz y soporte para el usuario. En este artículo se presenta una alternativa *open source* para la simulación de sistemas, que puede ser invocada a través de aplicaciones o rutinas creadas en Python. La alternativa, denominada PyChrono, consiste en un paquete de funciones (*wrapper*), desarrolladas originalmente en C++, que puede ser empleado desde un entorno de desarrollo integrado (IDE) para Python. PyChrono forma parte de un proyecto desarrollado en conjunto por investigadores de la Universidad de Wisconsin-Madison (USA) y la Universidad de Parma (Italia), cuyo objetivo es ofrecer una plataforma abierta para representar la dinámica de sistemas mediante la simulación multifísica. En este trabajo se argumentan ventajas y limitaciones de esta aplicación, que no solo podría emplearse como un recurso docente para el aprendizaje integral y aplicado de Python, sino también para fines de investigación y desarrollo en ingeniería.

PALABRAS CLAVES: Simulación *multifísica* de sistemas, Python, Project Chrono, PyChrono.

INTRODUCCIÓN

La pandemia ha irrumpido en los procesos de aprendizaje, en todos los niveles de formación. La interacción entre estudiantes y docentes sigue realizándose a través de un entorno virtual, al cual todos los actores se han debido aproximar de forma vertiginosa, como consecuencia de la vigencia e incertidumbre vinculadas a la actual crisis sanitaria. Para atenuar esta irrupción, no solo se requiere adaptar y acelerar la aplicación sistemática de metodologías para que los estudiantes y docentes puedan desenvolverse en este contexto, sino también la necesidad de una comunicación, activa y frecuente, entre ambos actores (Gelles et al., 2020).

En ingeniería, al igual que en otras carreras universitarias, se requiere examinar alternativas para poder atenuar las limitaciones que surgen al realizar experiencias de aprendizajes dominadas por actividades no presenciales. La simulación de la dinámica de sistemas, asistida por aplicaciones computacionales, es una de las experiencias que puede ser afectada en esta crisis. Por ejemplo, el acceso remoto a una licencia comercial, de una determinada aplicación computacional, puede convertirse para el estudiante de ingeniería en una restricción relevante. Adicionalmente, las licencias comerciales, distribuidas para fines docentes y académicos, suelen presentar limitaciones en cuanto a su capacidad para la simulación del comportamiento de sistemas mecánicos (*e.g.* número de grados de libertad limitados, solo admiten la simulación del sistema en condiciones de equilibrio estático, número limitado de usuarios en caso de conexión remota, entre otras).

Una alternativa para paliar estas dificultades la representa el uso de aplicaciones computacionales distribuidas a través de licencias no comerciales. En la actualidad, la evidencia bibliográfica disponible da cuenta de diversas alternativas “open source”, que no solo pueden ser destinadas a fines docentes sino también como herramientas para el desarrollo y la investigación aplicada en ingeniería, *e.g.*(Buffoni et al., 2021).

El presente trabajo dirige su atención hacia una opción no comercial para asistir al usuario en la simulación de la dinámica de sistemas mecánicos. La opción, denominada *Chrono*, es un proyecto desarrollado desde hace más de veinte años -y de forma conjunta- por investigadores de la Universidad de Wisconsin-Madison (USA) y la Universidad de Parma (Italia) (Serban et al., 2018). Bajo la hipótesis de que este proyecto, al ofrecer la facilidad del uso de sus capacidades mediante la programación en *Python*, podría también emplearse como un recurso para estrategias de aprendizaje de programación de Python. Su integración a tales estrategias resultaría de particular interés para el estudiante de ingeniería mecánica, y afines, al ofrecer un componente motivacional y aplicado, orientado al logro de competencias de aprendizaje asociadas con un lenguaje de programación. De hecho, el uso de Python sigue en constante crecimiento en ingeniería, frente a otros lenguajes de alto nivel similares, pero distribuidos mediante licencias comerciales (Fraanje et al., 2016).

Además de las secciones previas, la estructura de este artículo consta de cuatro secciones. En la siguiente sección se expone una sinopsis de las características generales del proyecto *Chrono*, que es el código abierto original en el cual se basa la aplicación computacional denominada *PyChrono* por sus autores. Seguidamente, se reconocen aspectos específicos sobre los requerimientos y limitaciones al emplear *PyChrono* a través de un entorno de desarrollo integrado (IDE) para poder programar en Python la simulación de la dinámica de un sistema mecánico. Luego, en una siguiente sección se ilustra un ejemplo, que presenta la simulación de la respuesta estacionaria de un modelo de rotor de Jeffcott sobre soportes flexibles. Como forma de validación, los resultados de la simulación se comparan con los resultados experimentales publicados por otros investigadores (Alsaleh et al., 2020). Finalmente, se concluye sobre la factibilidad del uso de esta herramienta computacional como un recurso académico para ingeniería mecánica y afines.

CHRONO: DESCRIPCIÓN Y ENTORNO GENERALES

Chrono es un proyecto de código abierto desarrollado originalmente para simular: (i) sistemas de cuerpos rígidos vinculados, cuya dinámica está representada por ecuaciones diferenciales, (ii) la dinámica de sistemas de cuerpos deformables, representada por ecuaciones diferenciales parciales, (iii) la interacción fluido-sólido, en la cual su dinámica está definida por sistemas de ecuaciones diferenciales ordinarias y parciales acopladas (Serban et al., 2018). El código está desarrollado fundamentalmente en C++ (Tasora et al., 2015), y puede ser compilado para poder ser usado de forma amplia por terceros, gracias a su distribución bajo licencia BSD-3 (McKusick et al., 2015).

En la actualidad, *Chrono* ha alcanzado una madurez estable. Diversas referencias están disponibles como soporte de su validación, realizada a través de la comparación con resultados experimentales, o bien con los obtenidos mediante aplicaciones comerciales, *e.g* (Sunday et al., 2020). El código sigue siendo desarrollado de forma modular para incorporar más funcionalidades. Sus funcionalidades básicas pueden ser resumidas como se indica a continuación (Serban et al., 2018):

Dinámica de sistemas de múltiples cuerpos. Mediante el módulo fundamental de Chrono (*i.e.* Chrono: Engine) puede simularse desde la dinámica de sistemas mecánicos formados por la

vinculación de un número reducido de cuerpos (*e.g.* sistemas robóticos), incluyendo contacto en presencia de fricción seca, hasta sistemas que en los cuales puede existir un flujo granular (*e.g.* el flujo asociado al transporte de material granular), en los cuales suceden múltiples *(i.e.* millones) de contactos simultáneos. En particular, para los problemas que involucran contacto entre cuerpos, es posible seleccionar entre dos modelos para simular el contacto: (a) modelo NSC (*non-smooth contact* model, o modelo de contacto "no blando") y modelo SMC (*smooth contact model*, o modelo de contacto "blando"). Ambos modelos son utilizados con frecuencia para el estudio de problemas de contacto (Flores & Ambrósio, 2010). El módulo Chrono:Engine contiene varias funciones que permite definir desde el tipo de vinculación (*e.g.* articulaciones) entre cuerpos, así como también distintos tipos de actuadores (*i.e.* motores). Es este módulo el que presenta las rutinas para la solución de los sistemas de ecuaciones algebraicas que proviene del modelo numérico del sistema mecánico simulado. El módulo ofrece al programador la posibilidad de seleccionar más de diez esquemas de integración numérica (*e.g.* Euler explicito, Euler implícito, RungeKutta, Newmark).

Formulación lineal y no lineal de elementos finitos. Esta funcionalidad está también incluida en el módulo fundamental del proyecto. Chrono cuenta con más de treinta tipos de elementos para abordar problemas en los cuales están presentes cuerpos que pueden estar sujetos a deformaciones significativas, así como a movimiento general (*i.e.* traslación y rotación en tres dimensiones) (Serban et al., 2018).

Dinámica de vehículos. A través del módulo específico, identificado como Chrono::Vehicle, puede efectuarse la simulación de la dinámica de vehículos terrestres. Este módulo incluye diferentes modelos de vehículos, ya parametrizados para facilidad del usuario, incluyendo además varios modelos para los subsistemas asociados a un vehículo terrestre, tales como suspensión, transmisión, neumáticos, sistema de dirección y otros (Serban et al., 2019). La capacidad de este módulo también comprende diversos modelos de terreno para evaluar la interacción entre los neumáticos y el terreno, así como el desempeño del vehículo en determinado tipo de superficie.

Dinámica de sistemas con interacción fluido-sólido. El módulo de Chrono, denominado Chrono::FSI, permite la simulación dinámica de cuerpos rígidos, o deformables, que interactúan con fluidos. El algoritmo desarrollado para este módulo se basa en una versión extendida de la *metodología para la hidrodinámica de la partícula "blanda"* (Liu & Liu, 2010), versión que los autores identifican como *XSPH*, conforme a su denominación en inglés (Pazouki et al., 2014).

Co-simulación, procesamiento de cómputo en paralelo y distribuido. Estas funcionalidades se concentran en tres módulos dedicados: Chrono::Cosimulation, Chrono::Distributed, and Chrono::Parallel (Serban et al., 2018). El uso de estos módulos está orientado a objetivos que pueden comprender desde la *cosimulación* de sistemas, asistida con otra aplicación computacional, hasta el uso eficiente de hardware mediante la ejecución en paralelo o distribución de tareas vinculadas a la simulación de sistemas complejos (Serban et al., 2018).

Animación o visualización. El módulo Chrono: Irrlicht se emplea por defecto, como medio para la animación y visualización en 3D de la simulación del sistema mecánico en estudio. Este módulo invoca a "Irrlicht" (*Irrlicht Engine - A free open source 3D engine*, 2021), un conocido *motor*, desarrollado en C++, multiplataforma y de código abierto, utilizado también como recurso de animación para videojuegos y otras aplicaciones.

Chrono cuenta con otros módulos que permiten extender las capacidades de este proyecto, tales como la simulación dinámica de la respuesta de sensores inerciales, GPS, entre otras capacidades. Una descripción detallada de las funcionalidades está disponible en (*Project Chrono - An Open-Source Physics Engine*, 2018).

Desde un proyecto desarrollado en C++, el usuario puede importar y utilizar los módulos de *Chrono*. Para ello, debe instalarse previamente la aplicación, para que luego puedan ser invocadas las funciones requeridas, según los módulos instalados por el usuario. Conforme a lo indicado por sus creadores, *Chrono* es multiplataforma: puede ser instalado en Linux, Mac OSX y Windows (Tasora, Serban, et al., 2020).
Para el usuario no familiarizado con C++, el proyecto ofrece la opción de utilizar *PyChrono*, un paquete de Python (*Python wrapper*) de funciones que, una vez instalado, puede ser empleado desde una rutina escrita en Python. No obstante, tal como se refiere en la siguiente sección esta opción presenta ciertas limitaciones respecto al uso de la aplicación original escrita en C++.

PYCHRONO: REQUERIMIENTOS Y LIMITACIONES

PyChrono puede definirse como una interfaz para que el usuario de Python pueda importar y utilizar las funciones del proyecto Chrono desde un algoritmo escrito en Python. En particular, para el usuario de ingeniería, que no cuente con experiencia en programación en C++, esto facilita el uso de las funciones ofrecidas por este proyecto. El uso de sus funciones desde Python permite integrar las funcionalidades de Chrono a otras aplicaciones, también disponibles en Python, y de utilidad en el procesamiento de datos.
El procedimiento recomendado para la instalación de *PyChrono* consiste en utilizar una versión precompilada que puede ser instalada de forma simple, en un entorno previamente definido por el usuario. La instalación de cualquiera de las dos versiones precompiladas de *PyChrono*, *i.e.* la versión “estable” o la versión en desarrollo, resulta más simple si se realiza a través de Anaconda (*Anaconda | The World’s Most Popular Data Science Platform*, 2021). Anaconda es una conocida plataforma que permite la integración y uso de múltiples herramientas computacionales, de código abierto, bajo un entorno integrado.
No todos los módulos de Chrono están disponibles en las versiones precompiladas de *PyChrono*. Adicional a esta limitación, y aunque la sintaxis de programación en Python puede resultar menos complicada que la de C++; un algortimo desarrollado en Python, que use funciones de Chrono mediante PyChrono, suele ejecutarse más lento que su versión análoga escrita en C++. Para la simulación de sistemas complejos (*e.g.* sistemas que involucran flujo granular, múltiples grados de libertad o colisiones múltiples) esta diferencia puede ser relevante en cuanto al tiempo de procesamiento computacional requerido.
Al instalar la versión precompilada de PyChrono, el usuario cuenta con una colección importante de ejemplos desarrollados en Python: más de cincuenta ejemplos se encuentran organizados en carpetas, desde las cuales el usuario puede conocer la sintaxis para el uso de las funcionalidades específicas de Chrono desde Python. La revisión sistemática de estos ejemplos es la mejor opción para iniciarse en el uso de PyChrono. Desde los ejemplos básicos sobre manipulación vectorial, cuaterniones, sistemas de coordenadas, hasta aquellos referidos a problemas discretos de colisión entre sólidos rígidos, incluyendo ejemplos básicos que ilustran la simulación de elementos deformables, sujetos a traslaciones o rotaciones. Esta estrategia, junto con la revisión o consulta en el foro de usuarios del proyecto, sugiere un procedimiento de autoaprendizaje de esta herramienta computacional.
El ejemplo que se ilustra en la siguiente sección fue realizado utilizado la versión pychrono-6.0.0-py38_223 para Windows 64 bits, y que en formato comprimido tiene un tamaño igual a 173 MB. Esta versión fue instalada en un notebook personal, que cuenta con un procesador Intel® Core ™ i7-8550U, con un CPU @1.80GHz, y 8 GB en RAM. *PyChrono* fue instalado empleando la plataforma Anaconda, creando previamente un entorno (*environment*) dedicado. Este entorno específico ya contiene, por defecto, Python, en su versión 3.8.5, junto con aplicaciones complementarias útiles para el procesamiento de vectores y matrices (*i.e.* matplotlib, numpy,

scipy). La versión instalada de *PyChrono* requiere 590 MB en disco duro. El código del ejemplo en cuestión, fue escrito usando *Spyder* como IDE, en su versión 4.2.1, y que suele estar instalado por defecto en cualquier entorno creado desde Anaconda. Una vez instaladas las señaladas aplicaciones, todas se ejecutan de forma local en el computador: no se requiere conexión a internet durante su ejecución.
El ejemplo presentado en la próxima sección es una modificación, y extensión, de un código que forma parte de los ejemplos disponibles de PyChrono, una vez que es instalada cualquiera de sus dos versiones precompiladas. El ejemplo original, que fue desarrollado por Alessandro Tasora, consiste en un modelo de rotor de Jeffcott sobre soportes rígidos, operando en condición de resonancia (Alessandro Tasora, 2019). Aunque el ejemplo tiene un propósito ilustrativo, su geometría y propiedades están basadas en una referencia (Alsaleh et al., 2020), con la finalidad de comparar los resultados de la simulación con resultados experimentales.

UN EJEMPLO: ROTOR DE JEFFCOTT SOBRE SOPORTES FLEXIBLES

Un rotor de Jeffcott es un modelo simple, pero efectivo para la descripción y compresión del comportamiento dinámico de máquinas rotativas (Childs, 1993). La dinámica del rotor de Jeffcott puede estudiarse: (i) de forma experimental, (ii) a través de un modelo matemático analítico (*e.g.* utilizando un modelo discreto de dos grados de libertad), o bien, (iii) mediante un modelo numérico, en el cual suele representarse la flexibilidad del eje a través de una formulación de elementos finitos.
El presente ejemplo corresponde a un modelo de rotor de Jeffcott sobre soportes flexibles. La Fig. 1 presenta una imagen de la animación generada junto con la simulación de la respuesta dinámica del rotor. En el código del ejemplo fueron “importados” los siguientes módulos: *pychrono* (módulo con las funciones básicas), *pychrono.fea* (módulo de elementos finitos), *pychrono.pardisomkl* (módulo con rutinas para la solución del sistema de ecuaciones asociado al modelo matemático) y *pychrono.irrlicht* (módulo que permite generar la animación del modelo mediante la aplicación *Irrlicht*) . Adicional a estos módulos, fueron importados desde el código escrito otras aplicaciones “open source” (*i.e.* numpy, matplotlib y scipy), previamente instaladas mediante Anaconda, y con la finalidad facilitar el procesamiento de los resultados de la simulación.
Las dimensiones y propiedades mecánicas del modelo corresponden a las indicadas por los autores (Alsaleh et al., 2020), quienes realizan un estudio experimental y teórico de las vibraciones laterales de un rotor de Jeffcott desbalanceado sobre soportes flexibles.
Los resultados experimentales reportados por los mencionados investigadores se asumen como valores de referencia para la validación de esta simulación. En la Tabla 1 se identifican las dimensiones y propiedades principales del modelo.
En el montaje experimental se instalaron dos pares de sensores de proximidad para las mediciones de vibración laterales del eje. En cada punto de medición se dispuso de dos sensores para la medición de desplazamientos vertical y transversal. Cada punto de medición se ubicó ubicados axialmente una distancia igual a 18 cm desde el soporte más cercano (Alsaleh et al., 2020).
Las propiedades de rigidez y amortiguación viscosa equivalentes de cada soporte fueron obtenidas a partir de los valores experimentales de rigidez y amortiguación equivalentes del sistema, estimadas experimentalmente por (Alsaleh et al., 2020).

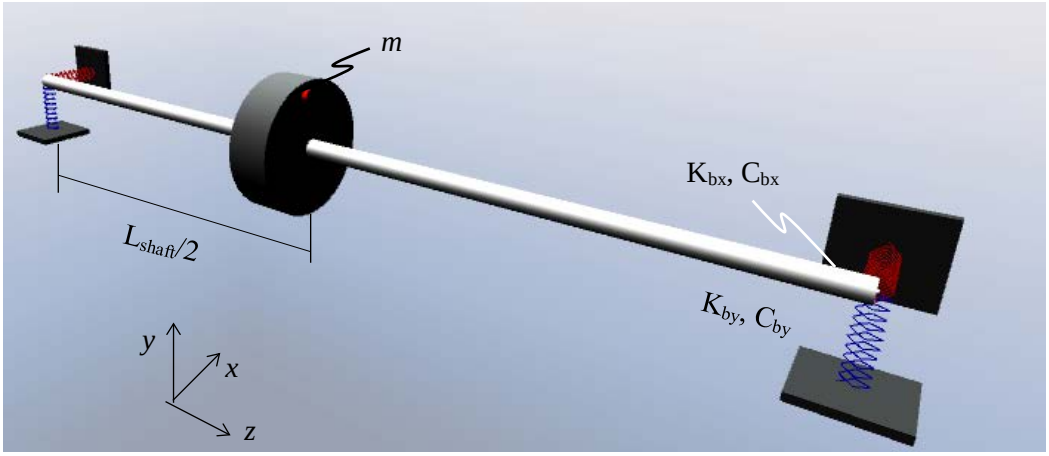

Figura 1. Modelo de Rotor de Jeffcott sobre soportes flexibles en PyChrono.

Tabla 1. Dimensiones y propiedades del Rotor de Jeffcott sobre soportes flexibles (Alsaleh et al., 2020)

Eje	
longitud entre soportes:	$L_{shaft} = 500 \cdot 10^{-3}$ m.
diámetro:	$\Phi_{shaft} = 10 \cdot 10^{-3}$ m.
módulo de elasticidad:	$E_{shaft} = 210 \cdot 10^{9}$ Pa.
Disco	
diámetro:	$\Phi_{disk} = 75 \cdot 10^{-3}$ m.
espesor:	$t_{disk} = 25 \cdot 10^{-3}$ m.
masa:	$m_{disk} = 0{,}8$ Kg.
posición en eje: equidistante respecto a soportes	$L_{shaft}/2 = 250 \cdot 10^{-3}$ m
Soporte	
rigidez equivalente en dirección transversal:	$K_{bx} = 183{,}281 \cdot 10^{3}$ N/m.
amortiguación equivalente en dirección transversal:	$C_{bx} = 2{,}383$ N·s/m.
rigidez equivalente en dirección vertical:	$K_{by} = 112{,}245 \cdot 10^{3}$ N/m.
amortiguación equivalente en dirección vertical:	$C_{by} = 1{,}497$ N·s/m.

Para el modelo desarrollado en este ejemplo, la flexibilidad del eje, así como su distribución de masa e inercia, se representan mediante una formulación en elementos finitos, contemplado elementos tipo viga, con seis grados de libertad por nodo, y dividiendo uniformemente la longitud del eje. Un valor de densidad típico de 7800 Kg/m^3 fue asumido para el acero del eje y el disco.
Para efectos de comparación, se utilizaron dos tipos de formulación para elementos tipos viga, disponibles en PyChrono: la formulación de Euler-Bernoulli y la formulación basada en análisis *isogeométrico* (IGA), que puede utilizarse también para casos en los que se presenten "grandes" deformaciones en una viga (Tasora, Benatti, et al., 2020). De esa forma, el eje se representó mediante un total de 25 elementos para la formulación Euler-Bernoulli, mientras que con la formulación IGA se dividió el eje en 33 elementos de segundo orden. Estas formulaciones pueden utilizarse de forma simple, al invocar funciones de PyChrono que reciben como argumentos las propiedades geométricas y materiales del eje, así como el número total de elementos en los cuales se desea discretizar el eje.
La contribución de cada soporte a la dinámica del rotor se introduce mediante un arreglo resorte-amortiguador lineal con los valores de rigidez y amortiguación viscosa equivalentes indicados en la Tabla 1. La Fig. 1 ilustra los resortes que simulan la flexibilidad y amortiguación aportada por cada soporte, los cuales se consideran idénticos (Alsaleh et al., 2020).

Según los resultados experimentales obtenidos por (Alsaleh et al., 2020), se identifica una primera frecuencia crítica cercana a los 215.1 rad/s (2054 RPM). Por consiguiente, el rotor puede ser considerado "rígido" para velocidades de operación inferiores a los 2000 RPM, aproximadamente. Para efectos de comparación, se han agregado en la Tabla 2 los resultados experimentales obtenidos por los referidos autores. Estas amplitudes de vibración corresponden a tres casos independientes de desbalance: el primer caso resulta al agregar en el plano del rotor una masa igual a 0.4g, mientras que los valores asociados a 0.8g y 1.2g, representan el segundo y tercer caso, respectivamente. En todos los casos, la masa se ubicó una distancia radial constante, igual a 30 mm, medida desde el centro del eje (Alsaleh et al., 2020).
En la Tabla 2 se muestran los valores experimentales de las amplitudes *peak-peak* ($A_{peak\text{-}peak}$) de vibración, medidas en dos condiciones de velocidad de operación distintas: 1000 RPM y 3600 RPM. Se ha identificado como "Soporte interno" a las mediciones registradas por el par de sensores cuya ubicación axial es cercana al soporte que se encuentra entre el motor y el disco del rotor del banco de ensayos. En contraste, las mediciones identificadas como "Soporte externo" se refieren a las mediciones vertical (y_{sen}) y transversal (x_{sen}) registradas por el otro par de sensores.

Tabla 2. Resultados experimentales reportados por (Alsaleh et al., 2020)

Ω[RPM]	GDL	Soporte interno $A_{peak\text{-}peak}$ [mils]			Soporte externo $A_{peak\text{-}peak}$ [mils]		
		m= 0,4g	*m= 0,8g*	*m= 1,2g*	*m= 0,4g*	*m= 0,8g*	*m= 1,2g*
1000	y_{sen}	0,614	0,970	1,137	0,685	1,040	1,437
	x_{sen}	0,569	0,880	1,295	0,417	0,647	0,858
3600	y_{sen}	1,074	1,891	2,719	1,232	2,140	3,260
	x_{sen}	1,339	2,276	2,978	1,152	1,986	2,834

Alsaleh et al. proponen un modelo discreto de dos grados de libertad para este rotor. Este modelo logra reproducir el orden de magnitud de las amplitudes de vibración del rotor. No obstante, si se comparan los resultados de este modelo analítico con los valores experimentales, los errores relativos varían desde 1% hasta un 92%.
En las Tablas 3 y 4 se muestran los valores de las amplitudes de vibración obtenidas a partir del modelo desarrollado en PyChrono y para las dos formulaciones de elementos tipo viga considerados. La Tabla 3 corresponde a los resultados en el punto de medición identificado como "Soporte interno", mientras que en la Tabla 4 se muestran las amplitudes simuladas para las mediciones cercanas al "Soporte externo".
Todos los resultados de la simulación han sido generados con un paso de integración igual a $\Delta t = 10^{-4}$ s. y utilizando el método implícito de Euler para el avance en el tiempo. La rutina de solución (*solver*) seleccionada fue *ChSolverPardisoMKL()*, que es la recomendada por los creadores de PyChrono en la simulación de la dinámica de modelos que involucran formulación en elementos finitos. Por limitaciones de espacio, en este artículo no se desarrolla un análisis de convergencia del modelo planteado.
Para efectos de un análisis formal de convergencia, la aplicación permite variar tipo de elemento de viga empleado, número total de elementos, así como seleccionar entre más de diez métodos de integración numérica.

Tabla 3. Resultados de simulación utilizando dos tipos de formulación (“Soporte interno”)

Ω[RPM]	GDL	Euler-Bernoulli $A_{peak\text{-}peak}$ [mils]			IGA $A_{peak\text{-}peak}$ [mils]		
		m= 0,4g	*m= 0,8g*	*m= 1,2g*	*m= 0,4g*	*m= 0,8g*	*m= 1,2g*
1000	y_{sen}	0,461	0,922	1,382	0,456	0,913	1,372
	x_{sen}	0,406	0,813	1,219	0,402	0,804	1,206
3600	y_{sen}	1,227	2,715	3,557	1,166	2,332	3,498
	x_{sen}	1,213	2,534	3,718	1,186	2,369	3,547

En las Tablas 5 y 6 se muestran los errores relativos de las amplitudes de vibración mostradas en las tablas anteriores. El máximo error relativo corresponde a 44%, para el caso en el cual se utiliza la formulación de Euler-Bernoulli y cuando se agrega una masa de 0,8g, a una velocidad de operación igual a 3600 RPM. Esta diferencia significativa entre el resultado del modelo y el experimental sucede para las mediciones cercanas al “Soporte interno”
Con la excepción de este error, puede observarse que ambas formulaciones presentan errores relativos de similar magnitud, tanto para las mediciones cercanas al “Soporte interno” como al “Soporte externo.
Los errores de los resultados de las simulaciones se atribuyen a diferentes causas. Entre ellas, se debe mencionar que el modelo no contempla el desbalance original que presenta el rotor, así como la suposición de que las propiedades de ambos soportes son idénticas.

Tabla 4. Resultados de simulación utilizando dos tipos de formulación (“Soporte externo”)

Ω[RPM]	GDL	Euler-Bernoulli $A_{peak\text{-}peak}$ [mils]			IGA $A_{peak\text{-}peak}$ [mils]		
		m= 0,4g	*m= 0,8g*	*m= 1,2g*	*m= 0,4g*	*m= 0,8g*	*m= 1,2g*
1000	y_{sen}	0,421	0,841	1,260	0,4193	0,8401	1,262
	x_{sen}	0,368	0,735	1,103	0,3675	0,7353	1,103
3600	y_{sen}	1,119	2,476	3,245	1,081	2,161	3,242
	x_{sen}	1,097	2,291	3,362	1,092	2,180	3,264

En adición a esto, no se dispone de información sobre las estimaciones de incertidumbres experimentales, vinculadas a las mediciones de las amplitudes de vibración, que son utilizadas como valores referenciales. Otra contribución relevante a la incertidumbre del modelo es atribuible a los valores de rigidez y amortiguación viscosa equivalentes de los soportes, que son obtenidos experimentalmente por (Alsaleh et al., 2020). Por otra parte, no se cuenta con medición de fase, como tampoco se conocen las posiciones angulares de las masas de desbalance usadas en las pruebas experimentales. Estas fuentes de incertidumbres limitan la posibilidad de mejorar (calibrar o sintonizar) el modelo numérico con el experimental. Sin embargo, el ejemplo presentado logra ilustrar las capacidades de PyChrono para la simulación de sistemas mecánicos.
El tiempo máximo requerido para cada simulación, sin la ejecución “simultánea” de la animación, fue menor 11 minutos, aproximadamente. Para las simulaciones realizadas con la formulación de Euler-Bernoulli, el tiempo de ejecución estimado fue de 5 minutos, mientras que en el caso de los resultados generados con la formulación IGA, el tiempo estimado fue de 10 minutos, aproximadamente. La animación “simultánea” añade un tiempo adicional estimado en 4 minutos para la formulación Euler-Bernoulli, y de 6 minutos para la formulación IGA.

Tabla 5. Errores relativos para los resultados de la simulación (“Soporte interno”)

Ω[RPM]	GDL	Euler-Bernoulli $E_{rel}(\%)$			IGA $E_{rel}(\%)$		
		m= 0,4g	*m= 0,8g*	*m= 1,2g*	*m= 0,4g*	*m= 0,8g*	*m= 1,2g*
1000	y_{sen}	-25%	-5%	22%	-26%	-6%	21%
	x_{sen}	-29%	-8%	-6%	-29%	-9%	-7%
3600	y_{sen}	14%	44%	31%	9%	23%	29%
	x_{sen}	-9%	11%	25%	-11%	4%	19%

Tabla 6. Errores relativos para los resultados de la simulación (“Soporte externo”)

Ω[RPM]	GDL	Euler-Bernoulli $E_{rel}(\%)$			IGA $E_{rel}(\%)$		
		m= 0,4g	*m= 0,8g*	*m= 1,2g*	*m= 0,4g*	*m= 0,8g*	*m= 1,2g*
1000	y_{sen}	-39%	-19%	-12%	-39%	-19%	-12%
	x_{sen}	-12%	14%	29%	-12%	14%	29%
3600	y_{sen}	-9%	16%	0%	-12%	1%	-1%
	x_{sen}	-5%	15%	19%	-5%	10%	15%

Todas estas estimaciones de tiempo de ejecución son referenciales, ya que pueden variar por la ejecución síncrona de otros procesos (programas) en el computador durante las simulaciones.

CONCLUSIONES

En este trabajo se ha hecho una exposición sucinta de las capacidades de PyChrono para simular la dinámica de sistemas mecánicos desde un entorno de programación de Python. PyChrono puede ser integrado con otros paquetes de utilidad en ingeniería desde un IDE para Python (*e.g.* Spyder). Su instalación es simple y los requerimientos de hardware no son mayores a los que puede demandar un software similar que sea distribuido mediante licencia comercial.

Pese a las incertidumbres vinculadas a los valores experimentales, usados como referencias, el ejemplo presentado logra ilustrar las capacidades de simulación de PyChrono.

Con la asistencia de PyChrono, la generación de modelos de sistemas mecánicos mediante Python ofrece la posibilidad de integrarla a estrategias de aprendizaje de un lenguaje de alto nivel como Python. Esto representa un potencial componente motivacional para el aprendizaje de este lenguaje de programación.

PyChrono no ofrece una interfaz gráfica al usuario similar a la que si presentan sus contrapartes comerciales. La dirección en la que apuntan las nuevas versiones de esta aplicación computacional “open source” sugiere que no es el objetivo de los creadores del proyecto competir con un software comercial en este ámbito. La ausencia de interfaz gráfica para el usuario puede representar una restricción en su uso como herramienta destinada a objetivos docentes, o de investigación y desarrollo. En contraste a este argumento, al facilitar la simulación desde Python, PyChrono permite un dominio directo en el diseño del algoritmo para construir el modelo del sistema mecánico cuya dinámica se requiere simular. A su vez, este dominio implica que el usuario debe evaluar, seleccionar y analizar, por ejemplo, la rutina para la solución del sistema de ecuaciones del modelo matemático del sistema, así como el método de integración numérica. Esto es relevante cuando el usuario está interesado en un mayor control del modelo numérico para su necesario análisis de convergencia.

REFERENCIAS

Alessandro Tasora. (2019). *Example: FEA of the Jeffcott rotor passing through resonance* (Versión 2019) [Python]. demo_FEA_beamsIGA.py.

Alsaleh, A., Sedighi, H. M., & Ouakad, H. M. (2020). Experimental and theoretical investigations of the lateral vibrations of an unbalanced Jeffcott rotor. *Frontiers of Structural and Civil Engineering*, *14*(4), 1024-1032.

Anaconda | The World's Most Popular Data Science Platform. (2021). Anaconda. https://www.anaconda.com/

Buffoni, L., Ochel, L., Pop, A., Fritzson, P., Fors, N., Hedin, G., Taha, W., & Sjölund, M. (2021). Open source languages and methods for cyber-physical system development: Overview and case studies. *Electronics*, *10*(8), 902.

Childs, D. (1993). *Turbomachinery rotordynamics: Phenomena, modeling, and analysis*. John Wiley & Sons.

Flores, P., & Ambrósio, J. (2010). On the contact detection for contact-impact analysis in multibody systems. *Multibody System Dynamics*, *24*(1), 103-122.

Fraanje, R., Koreneef, T., Le Mair, A., & de Jong, S. (2016). Python in robotics and mechatronics education. *2016 11th France-Japan & 9th Europe-Asia Congress on Mechatronics (MECATRONICS)/17th International Conference on Research and Education in Mechatronics (REM)*, 014-019.

Gelles, L. A., Lord, S. M., Hoople, G. D., Chen, D. A., & Mejia, J. A. (2020). Compassionate flexibility and self-discipline: Student adaptation to emergency remote teaching in an integrated engineering energy course during COVID-19. *Education Sciences*, *10*(11), 304.

Irrlicht Engine—A free open source 3D engine. (2021). https://irrlicht.sourceforge.io/

Liu, M. B., & Liu, G. R. (2010). Smoothed particle hydrodynamics (SPH): An overview and recent developments. *Archives of computational methods in engineering*, *17*(1), 25-76.

McKusick, M. K., Neville-Neil, G. V., & Watson, R. N. (2015). *The design and implementation of the FreeBSD operating system*. Pearson Education.

Pazouki, A., Serban, R., & Negrut, D. (2014). A high performance computing approach to the simulation of fluid-solid interaction problems with rigid and flexible components. *Archive of Mechanical Engineering*, *61*(2), 227-251.

Project Chrono—An Open-Source Physics Engine. (2018). https://projectchrono.org/

Serban, R., Tasora, A., & Negrut, D. (2018, junio 24). *Chrono: An Open-Source Multi-physics Simulation Package*. The 5th Joint International Conference on Multibody System Dynamics, Lisboa, Portugal.

Serban, R., Taylor, M., Negrut, D., & Tasora, A. (2019). Chrono: Vehicle: template-based ground vehicle modelling and simulation. *International Journal of Vehicle Performance*, *5*(1), 18-39.

Sunday, C., Murdoch, N., Tardivel, S., Schwartz, S. R., & Michel, P. (2020). Validating N-body code chrono for granular DEM simulations in reduced-gravity environments. *Monthly Notices of the Royal Astronomical Society*, *498*(1), 1062-1079. https://doi.org/10.1093/mnras/staa2454

Tasora, A., Benatti, S., Mangoni, D., & Garziera, R. (2020). A geometrically exact isogeometric beam for large displacements and contacts. *Computer Methods in Applied Mechanics and Engineering*, *358*, 112635.

Tasora, A., Serban, R., Mazhar, H., Pazouki, A., Melanz, D., Fleischmann, J., Taylor, M., Sugiyama, H., & Negrut, D. (2020). Chrono: Multi-physics simulation engine. *Astrophysics Source Code Library*, ascl-2009.

Tasora, A., Serban, R., Mazhar, H., Pazouki, A., Melanz, D., Fleischmann, J., Taylor, M., Sugiyama, H., & Negrut, D. (2015). Chrono: An open source multi-physics dynamics engine. *International Conference on High Performance Computing in Science and Engineering*, 19-49.

SUGERENCIAS FACTIBLES PARA LA EDUCACIÓN REMOTA 2021-II

Luis Hevia, luis.hevia@usm.cl académico Informática, Universidad Técnica Federico Santa María
Sebastián Jara, sebastian.jarac@sansano.usm.cl memorista Ingeniería Civil Informática

Resumen: Se propone una guía de orientación que entregue sugerencias para el proceso formativo online en INF UTFSM, que se estima puede ser aplicado, con adaptaciones a otras carreras y departamentos de la Universidad. Se recolectó información sobre las opiniones y necesidades de estudiantes, profesores y ayudantes a través de encuestas y entrevistas, además de considerar varios documentos recientes. Se presenta un conjunto de propuestas que son descritas en veinte y cinco sugerencias (25) que se pueden aplicar prontamente por los profesores, así como trece (13) recomendaciones a la Institución. Se tuvo el propósito que sean sencillas de adoptar y aplicar, pero tienen su base en un modelo denominado UVA (Unidades Virtuales de Aprendizaje) que se venía aplicando desde antes de la contingencia y son consistentes con las exigencias del CALED (Instituto Latinoamericano y del Caribe de CALidad en Educación a Distancia).

Palabras Claves: educación remota, no presencialidad, metodología, evaluación.

1. CONTEXTO

El objetivo es avanzar en disponer de un Modelo de Educación a Distancia que permita contar con un Sistema de Aseguramiento de la Calidad, SAC (2020), que coherentemente sea ofrecido por nuestro Departamento de Informática (INF) de la Universidad (UTFSM). Los proponentes estiman que varias de las medidas propuestas son factibles de poder aplicar en otras Unidades, reconociendo que si bien existen realidades diferentes (lo que ocurre también al interior de INF), existen patrones similares que, con algunas variaciones, se pueden adaptar a otras realidades.

Como sabemos, hemos estado casi exclusivamente en modo educación online desde el 19 de octubre del 2019 por el estallido social y desde marzo 2020 por pandemia, se continuará al menos paralelamente, durante el segundo semestre de este año 2021, con la incógnita sobre el futuro, ya que estaremos sujetos a la evolución de la pandemia. Se reconocen los esfuerzos realizados por la Universidad DEO (2021), pero dada la magnitud del desafío, estudiantes y profesores encuestados expresan son insuficientes.

Creemos que esta nueva realidad sobre el modo online llegó para quedarse, pues existen numerosas ventajas (y también desventajas obviamente) que obligarán a mantener formatos combinados en algunas asignaturas y segmentos de las carreras: un ejemplo, asignaturas Electivas, que simultáneamente se pueden dictar para distintos Campus o Sedes, que no requieren uso de laboratorios, donde los profesores especialistas son escasos y/o los estudiantes con esas preferencias se pueden sumar para obtener un número adecuado de inscritos; también es relevante el desarrollo que puede tener en Postgrado y Postítulos, con estudiantes que en su mayoría trabajan por lo que valoran la flexibilidad para acceder en cualquier momento y desde cualquier lugar a estos programas. Es muy importante además considerar que algunas universidades nacionales y varias extranjeras, aprovechando la validación implícita del modelo, están evaluando expandirse a otros territorios y países, de modo que estamos ante un desafío estratégico de dimensiones.

Indudablemente nadie en la humanidad estaba preparado para la magnitud del desafío global que hemos vivido. Nos hemos adaptado, se han generado recomendaciones y realizado capacitaciones, se ha tratado de hacer en la medida de lo posible y con buena voluntad. Se han logrado éxitos y también, debemos reconocerlo, malas experiencias. Antes de entrar a repetir la situación por quinta vez, debemos generar las bases de un sistema de educación a distancia, que gestione el conocimiento, obtenido desde la experiencia vivida y siendo coherente con modelos reconocidos existentes desde antes de la pandemia.

Se propone un conjunto asertivo de sugerencias, en el marco de modelos conceptuales, como CALED&CREAD (2021) y HeviaL&GonzálezR (2016), documentos generados por entidades reconocidas Instituto de Ingenieros (2021) y Prince,Felder&Brent (2021) y/o criterios que ya han sido experimentados con buenos reconocimientos hacia profesores, ayudantes y/o alumnos, algunas ya cuantificadas vía encuestas internas HeviaL&JaraS (2021)y externas, otras mediante entrevistas a profesores. Es importante tener presente que, existiendo libertad de cátedra, es el profesor quien toma la decisión final, luego las propuestas que se les formulen deben ser factibles que éste las incorpore sin requerir cambios reglamentarios que generalmente toman demasiado tiempo, y tener presente que los procesos de acreditación exigen brindar un proceso formativo avalado por un sistema que provea garantías de calidad. Complementariamente se presentan una serie de medidas de mayor alcance que la Institución debiera desarrollar.

2. DIAGNOSTICO

Se han realizado 2 encuestas, la primera en el mes de abril para detectar problemáticas y casos de éxito surgidos, y la segunda en agosto para validar el propósito de las sugerencias a implementarlas, además de entrevistar a profesores. Paralelamente, la dirección del departamento creo un equipo de trabajo [1] que trabaja en levantar un modelo de educación a distancia con la misión de rescatar los mejores elementos de la no presencialidad, que sumados a la forma tradicional permitan darnos viabilidad futura, ya que estimamos que será finalmente el b-learning el formato global que prevalecerá, con las graves consecuencias que ello representará para las universidades. En síntesis, debemos reformular nuestra Propuesta de Valor, esa es la magnitud del desafío que las universidades y carreras enfrentamos.

Las figuras # 1,2,3 4 sintetizan algunos de los principales resultados obtenidos:

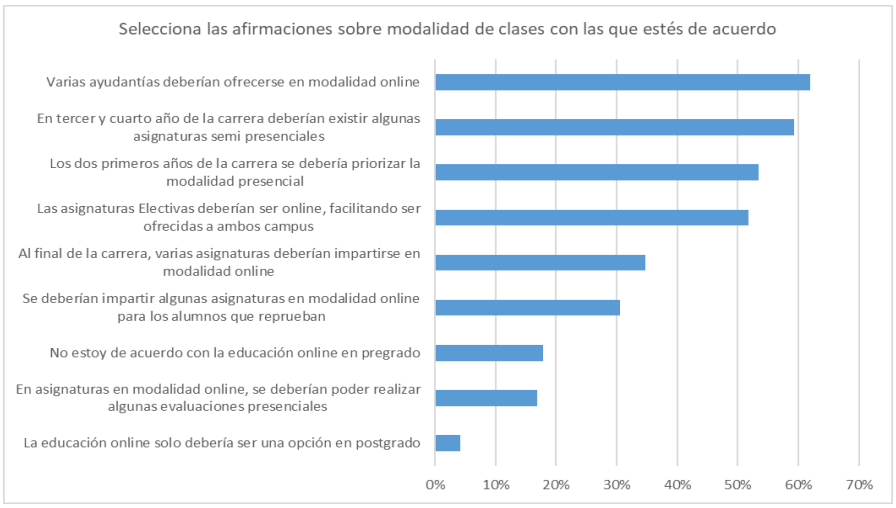

Figura #1: Segunda encuesta a estudiantes sobre la educación online en el DI.
Fuente: Elaboración propia.

Figura #2: Segunda encuesta a estudiantes sobre la educación online en el DI.

[1] Equipo integrado por el director del Departamento (Carlos Castro), la subdirectora de Pregrado (Cecilia Reyes), el subdirector de Postgrado (Marcelo Mendoza), la Coordinación de Programación de los 3 Campus (Andrea Vásquez por Valparaíso, Federico Meza por San Joaquín y Pedro Godoy por Vitacura), mas los profesores Claudio Lobos y Luis Hevia, junto al secretario técnico Sebastián Jara (alumno memorista=. Se agradece el aporte realizado por las y los colegas para el enriquecer este trabajo.

Fuente: Elaboración propia

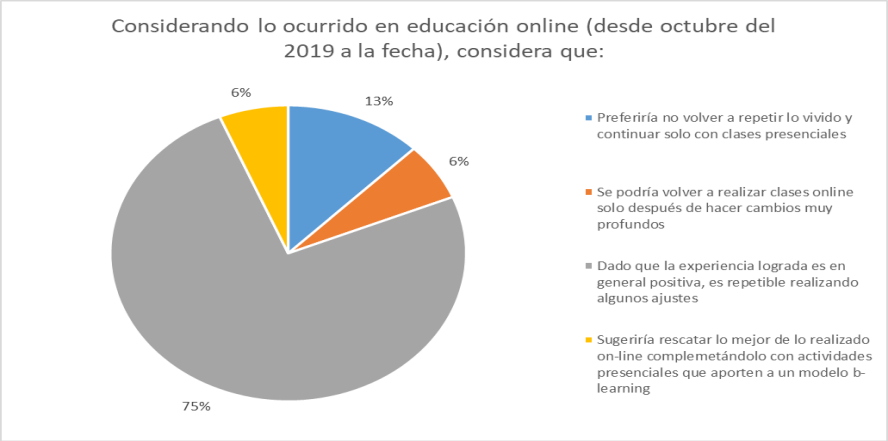

Figura #3: Segunda encuesta a profesores sobre la educación online en el DI.
Fuente: Elaboración propia.

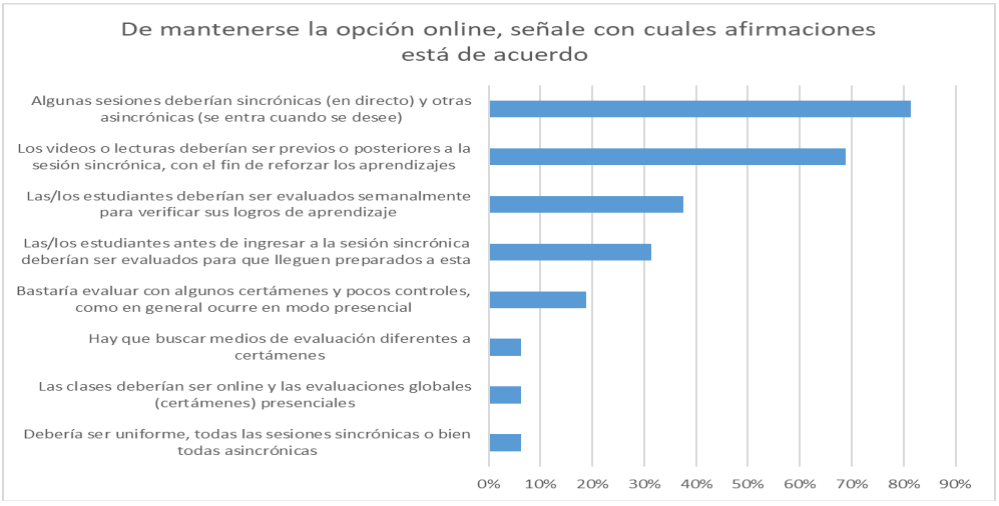

Figura #4: Segunda encuesta a profesores sobre la educación online en el DI.
Fuente: Elaboración propia.

3. UN MODELO DE EDUCACIÓN ONLINE

Es indudable que el diseño de una asignatura en modalidad online debe ser distinto al que se tendría en modalidad presencial y ello toma tiempo. Ante la urgencia que vivimos, se propone adaptar un modelo basado en las Unidades Virtuales de Aprendizaje (UVA), que se ha aplicado desde hace un tiempo (desde antes de la pandemia) en una asignatura masiva, dictada para miles de estudiantes en 5 campus o sedes, “Programación (IWI 131)”. Consiste básicamente en organizar el contenido de la asignatura en unidades temáticas asociadas a los resultados de aprendizajes, actividades para lograrlas y formas de evaluación coherentes, todo ello en periodos de tiempo determinados y breves (por ejemplo, en una o 2 semanas). Es importante destacar la facilidad de su implementación, pues es natural al quehacer de una asignatura, que prácticamente no implica problemas reglamentarios, pues es el profesor quien debe organizar el contenido de la asignatura, más aún ante la emergencia que se vive. La siguiente imagen (figura #5) relaciona las 3 ideas principales que lo sustentan y son presentados por VásquezA, MezaF & GodoyP (2021),

Figura #5: Modelo Constructivista.
Fuente: VásquezA, MezaF & GodoyP (2021).

4. SOBRE EVALUACIONES

Como reconocen los profesores, la evaluación es uno de los aspectos más críticos y cuestionados para aceptar la educación a distancia. Según comunicado reciente en diciembre las asignaturas tendrían calificaciones presenciales si así las condiciones sanitarias lo permiten. Por lo cual, es necesario tener previstas formas de evaluación no presenciales que el profesor debería:

1. Repensar el tipo y oportunidad de las evaluaciones que realiza en una asignatura, si están correctamente diseñadas para ser realizadas en el tiempo establecido y con los medios disponibles.
2. Teniendo muy presente que las conexiones sincrónicas a internet pueden fallar, se recomienda extender los tiempos que históricamente duraban los certámenes, a fin de prevenir el perjudicar a los estudiantes por esa causal.
3. Establecer desde el inicio del semestre un calendario de evaluaciones y una descripción básica del formato de estas, considerando la diferencia entre oportunidades sincrónicas (que se toman en un mismo momento) y asincrónicas (que se desarrollan en tiempos diferentes y con fecha y hora de entrega máxima). Es conveniente consultar a los estudiantes si tienen otros certámenes el mismo día, procurando que ello no ocurra por temas de salud mental, que se han agravado con la pandemia.
4. Notificar a las y los estudiantes que, si bien el departamento permite una asistencia voluntaria a las clases, recomienda la asistencia a estas, y en el caso de que existan evaluaciones, la obligatoriedad es exigida.
5. Analizar si los formatos de evaluación que utiliza que requieren vigilancia son completamente necesarios para la asignatura y revisar si es posible cambiarlos. Tener presente que pruebas con cámara y/o micrófono abierto genera molestias y en ocasiones es muy difícil de lograr por las limitaciones de las conexiones de internet de los hogares (incluso está prohibido en algunos países realizarlo así, ElPaís (2021). Priorizar las evaluaciones en las que sea menos probable el fraude académico, aunque no exista vigilancia, como ocurre con los proyectos
6. Innovar en evaluaciones que vayan más allá del test o preguntas estandarizadas, enfatizando en vías alternativas que permitan el desarrollo más libre de ideas o supuestos, como son los casos u otras formas; por ejemplo, colegas han solicitado a los estudiantes que expliquen el desarrollo de una pregunta en un video o una interrogación virtual (tener presente sí que toma tiempo al profesor).

7. Observar que las evaluaciones seleccionadas midan los resultados de aprendizaje y sean adecuadas para un entorno de aprendizaje en línea (es un tema crítico según estándares de educación a distancia).
8. Generar herramientas de autoevaluación que entreguen retroalimentación rápida a los estudiantes.
9. Tener presente que en el caso de las evaluaciones presenciales puede ocurrir que por temas de aforo no existan salas suficientes, de manera que esas fechas son propuestas a la universidad, que las podría alterar.

5. SOBRE ACTIVIDADES Y METODOLOGIA

Son relevantes las características de cada asignatura, si forman parte del bachiller, licenciatura, electivas, son convalidables con el posgrado, de tipo conceptual o mayoritariamente prácticas, campus donde se ofrecen, etc. Por lo anterior, el profesor debería:

10. Actuar con flexibilidad y comprensión, dado que seguirán existiendo situaciones impredecibles.
11. Identificar los contenidos prioritarios y contenidos secundarios según la importancia que tengan para el logro de los objetivos de aprendizaje, destinándolos a la modalidad más adecuada (sincrónica o asincrónica).
12. Disminuir la proporción de los tiempos de clase que son destinados a la explicación directa de contenidos teóricos (que pueden ser descritos en cápsulas de videos que el estudiante puede repasar hasta comprender) y aumentar el tiempo dedicado a lo práctico y metodologías activas, según sea posible en la asignatura.
13. Generar videos de corta duración (entre 6 y 8 minutos) que permitan enfatizar en aspectos claves de la asignatura, que deben ser subidos a Aula o Moodle al menos 24 horas antes de la clase.
14. Subir las grabaciones de las clases en un plazo máximo de 48 horas después de la clase (este es uno de los aspectos mejor evaluados de la modalidad online). Dejar con acceso público a los estudiantes del curso (esto implica también de los demás paralelos posibles) sin requerir solicitud para ello.
15. Definir como estructura de comunicación la provista por Aula para dejar mensajes y anuncios y, Zoom para las clases (evitando confusiones), de modo que el mail es para situaciones que requieren mayor privacidad.
16. Indicar el tiempo máximo de respuestas a correos y consultas en Aula, notificando a los estudiantes cuándo no podrán responder mensajes durante un periodo mayor de tiempo (este es uno de los aspectos que genera gran ansiedad y se refuerza dado que no hay interacción física ni atención en oficina).
17. Exigir aspectos básicos que mejoren la integración de los estudiantes, como es que cada uno de ellos deje visible en Aula una foto con su nombre (no se puede exigir conexión con cámara pues no todo estudiante tiene el ancho de banda que lo posibilita).
18. Favorecer la participación e interacción en clases, que permite a su vez verificar que la clase es seguida por los estudiantes, lo que puede ser vía encuestas en directo o preguntas dirigidas a algunos presentes (como se señaló antes no puede exigirse cámara y micrófono), pero si está disponible el chat (es un tema crítico según estándares de educación a distancia).
19. Comunicar el horario semanal de atención de los estudiantes vía Zoom o Discord (que ha sido muy valorado en las encuestas), lo cual ayuda tremendamente a resolver dudas puntuales, evitar confusiones y el estrés asociado, así como ofrecer la oportunidad de atender a otros estudiantes que necesitan dialogar con las y los profesores (es un tema critico según estándares de educación a distancia).
20. Cerrar el ciclo del aprendizaje, que se produce cuando el estudiante recibe *feedback* de sus evaluaciones. La plataforma Aula facilita la emisión de esos comentarios, al detectarse patrones o *cluster* en las respuestas emitidas, de un modo hasta mejor que en las pruebas rendidas en papel.

21. Evaluar la experiencia vivida. Insistir en el Informe de Cierre de cada asignatura que debe realizar cada profesor como exige INF profundizando las observaciones, los comentarios y los aportes de los profesores, así como de los estudiantes (vía Encuesta Docente), concentrándose en cuanto a la calidad de los contenidos y a la eficacia de la educación remota (es un tema crítico según estándares de educación a distancia).
22. Hacer un diagnóstico al inicio de una asignatura para medir las competencias de entrada de los estudiantes.
23. Estimar, en conjunto con los estudiantes, el tiempo que dedican a sus actividades académicas relacionadas con la asignatura (una crítica enunciada es la recarga que están teniendo). Por ejemplo, si usa la metodología de proyectos, es conveniente trabajar en proyectos coordinados con otras asignaturas, de manera que los estudiantes se concentren en pocos problemas, disminuyendo así la recarga académica y posibilitando una mayor profundidad en las temáticas abordadas.
24. Utilizar una estructura en Aula que facilite la comprensión del estudiante, y para ello incorporar las siguientes pestañas horizontales: Videos, Presentaciones, Apuntes, Entregas.
25. Valorar el aporte de los Ayudantes. En reiteradas ocasiones son mencionados por los estudiantes como colaboradores que han sido claves para mejorar sus aprendizajes y que en general dominan mejor el uso de las tecnologías ante la emergencia que se vive. Varias de las indicaciones y sugerencias aquí vertidas son totalmente extensibles a las ayudantías y deberían ser incorporadas en su quehacer.

.

6. TEMAS INSTITUCIONALES

Que por su magnitud son de responsabilidad central.

6.1 Difusión de información relevante

I. Se debe centralizar la siguiente información a estudiantes y profesores de una plataforma única, que incluya:
   i. Enlaces a servicios de la universidad, con una descripción de lo que puede obtener el estudiante en él.
   ii. Contactos de encargados de diversos temas, indicando a quién recurrir ante ciertas situaciones. Un aspecto que le debemos a nuestro Fundador, don Federico Santa María, es la trascendencia que tiene el desvalido meritorio, es contar con apoyo de asistentes sociales (y psicológico por la pandemia).
   iii. Tutoriales sobre temas relacionados al funcionamiento de la universidad.
   iv. Respuestas a preguntas frecuentes.
   v. Horas de operación para cada servicio prestado.
   vi. Incluir documentos relevantes.

II. Mejorar el acceso al material de ayuda sobre las plataformas existentes.

III. Aumentar la difusión de los servicios de orientación y asesoría (es un tema critico según estándares de educación a distancia).

IV. Ofrecer a las y los alumnos un medio para interactuar con sus pares en una comunidad en línea (fuera de asignaturas). En este sentido, INF ha generado una instancia, el "Patio Virtual" descrito en OrtegaO & ReyesC (2021) que debería ser realzada (es un tema critico según estándares de educación a distancia).

6.2 Apoyo en Temas Docentes

V. Fomentar la capacitación "puntual" a los profesores y ayudantes (que han sido colaboradores destacados por sus interesantes aportes) sobre el uso de herramientas, incluyendo la creación de videotutoriales.

VI. Aumentar la generación de documentos y videotutoriales que puedan ser accedidos para resolver dudas puntuales sin la necesidad de estar dentro de cursos de capacitación, los cuales deben explicar la función de cada tecnología, cómo se usan, y una explicación de cómo podrían beneficiar al proceso de aprendizaje. Esta es también una necesidad para con los estudiantes.

VII. Estimular la asistencia a las capacitaciones, así como para acceder al material producido.
VIII. Contar con un sistema de seguimiento de la capacitación docente e impacto real provocado en las asignaturas.
IX. Crear material informativo sobre el uso legítimo de obras, plagio y otros conceptos legales y éticos relevantes, que debe difundirse a todos los docentes (es un tema crítico según estándares de educación a distancia).
X. Diseñar un plan de apoyo para la adaptación de las clases a una menor duración (desde este año las sesiones bajaron de 90 a 70 minutos).
XI. Realizar de manera regular jornadas de retroalimentación y evaluación entre integrantes claves de la comunidad educativa, como jefes de carrera y representantes de los estudiantes.
XII. Disponer de equipamiento de calidad, esencial para la entrega de una educación de calidad. Las segundas pantallas, micrófonos, elementos de iluminación y audífonos con aislación de ruido, entre otros, permitirán aumentar el nivel de comodidad del profesor y facilitar su trabajo, y los estudiantes percibirán que reciben una educación de mayor nivel. En cuanto a los ayudantes, como son un elemento importante del proceso formativo, la mejora en sus condiciones con los implementos sugeridos tendría un impacto significativo. Imaginamos que, a esta altura, la universidad ha detectado las condiciones de internet de todas y todos sus estudiantes.
XIII. Evaluar los resultados obtenidos vía modificaciones en al menos algunas de las preguntas en la Encuesta Docente. Obviamente el cambio de modalidad provocado y prolongado ya por 2 años requiere medir aspectos claves que han sido impactados.

7. CONCLUSIONES

Considerando que siempre se debe evaluar para mejorar nuestro quehacer (*kaizen*) y que en este segundo semestre del año 2021 se promete el retorno a clases presenciales y también remotas, que será por quinto semestre, que las y los profesores hemos desarrollado experiencias con buenas (y también malas) prácticas, que existen desde hace años modelos de educación a distancia casi estandarizados y que por la premura de las contingencias no pudieron ser implementados, que debemos avanzar a consolidar nuestro quehacer online como base para asegurar la calidad, que existen universidades tanto nacionales como extranjeras que desean “invadir” el territorio de la virtualidad, se propusieron 25 sugerencias actualmente a ser implementadas y rápidas de incluir por los profesores a nivel de una Unidad (Departamento y/o Carrera), y 13 medidas bajo responsabilidad institucional.

8. REFERENCIAS BIBLIOGRAFICAS

CALED & CREAD (2021). Tarjeta de puntuación (SCCQAP) de Evaluación de Programas de Pregrado en línea. Sloan-C. Recuperado 5 de agosto de 2021, http://www.caled-ead.org/es/tarjeta-OLC-CALED.

ElPaís (2021). “Las universidades no podrán usar reconocimiento facial para vigilar exámenes online”. Recuperado 16 de agosto de 2021 https://elpais.com/tecnologia/2021-08-15/las-universidades-no-podran-usar-reconocimiento-facial-para-vigilar-examenes-online.html

DEO (2021). Dirección de Educación Online - Quiénes Somos. Recuperado 5 de agosto de 2021, de http://deo.usm.cl/somos.html

Hevia L & González R (2016). “Gestión del conocimiento y aprendizaje colaborativo utilizando Wikis: Adaptación de una metodología para generar una base de conocimientos usando Wiki como principal herramienta tecnológica”. Editorial VDM Verlag (Amazon). España

Hevia L & Jara S (2021). “Experiencia virtual en Informática de la UTFSM: Impacto y Mejoras”. Documento de trabajo enviado a 100 profesores.

Instituto de Ingenieros (2021). “Enseñanza de la Ingeniería en Pandemia. Percepciones y Desafíos de la Educación Remota en Chile”. Documento compartido a nivel universitario en Chile

OrtegaO & ReyesC (2021). “PATIO VIRTUAL: Un servido de Dicord para los estudiantes de pregrado del Departamento de Informática de la UTFSM”. Memoria para optar al titulo de Ingeniero Civil en Informatica. Gabiel Ortega (memorista) y Cecilia Reyes (prof. Guía)

Prince, Felder & Brent (2021). "Active Student Engagement in Online STEM Classes: Approaches and Recommendations" Advances in Engineering Education.

SAC (2020). Sistema de Aseguramiento de la Calidad en la UTFSM. Documento institucional.

VásquezA, MezaF & GodoyP (2021). “GoEmergency Remote Teaching Model for Massive. Programming Classes”. Versión "in press" del artículo aceptado en CLEI 2021.