UNIVERSIDAD DE CHILE
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN

IMPLEMENTACIÓN DE UNA HERRAMIENTA DE APOYO A LA TOMA DE DECISIONES EN INSTITUCIONES DE EDUCACIÓN SUPERIOR EN BASE A LA INFORMACIÓN CURRICULAR DE UCAMPUS

MEMORIA PARA OPTAR AL TÍTULO DE
INGENIERO CIVIL EN COMPUTACIÓN

ROBERTO SEBASTIÁN TAPIA AVENDAÑO

PROFESOR GUÍA:
WILLY MAIKOWSKI CORREA

PROFESOR CO-GUÍA:
JOCELYN SIMMONDS WAGEMANN

MIEMBROS DE LA COMISIÓN:
JOSE PINO URTUBIA
RODRIGO ARENAS ANDRADE

SANTIAGO DE CHILE
2022

Resumen

En la actualidad, el uso de plataformas computacionales en gestión y administración de organizaciones ha llevado a la acumulación de datos por parte de éstas mismas, y subsecuentemente, al interés por utilizar estos datos para generar valor. Ucampus, centro tecnológico de la Universidad de Chile, desarrolla las plataformas Ucampus y U-Cursos, mediante cuya operación acumulan gran cantidad de información sobre las instituciones que las utilizan. Actualmente no existen herramientas en las plataformas que utilicen apropiadamente estos datos. A raíz de esto, en el año 2020, se realiza un trabajo de memoria que analiza las necesidad de información y presenta un boceto de herramienta que la pueda suplir. El presente trabajo de memoria busca utilizar este trabajo previo como base para efectivamente implementar una herramienta que permita utilizar los datos almacenados con fines analíticos y que ayude a la toma de decisiones.

En primer lugar, se estudió las capacidades similares a lo buscado, implementado por plataformas alternativas Ucampus. En conjunto, se revisaron herramientas de *Business Intelligence*, y cómo éstas podrían ser utilizadas para solucionar la situación planteada.

Posteriormente, en una etapa de análisis, se estudiaron los tipos de usuario y casos de uso relevantes para establecer los requisitos de la herramienta. Además, se consideraron elementos no funcionales que serían importantes para la herramienta a desarrollar, como la escalabilidad y la extensibilidad de la misma. Además, se establece que la herramienta será un desarrollo propio, no utilizando las herramientas de *Business Intelligence* estudiadas. Con esto en consideración, se procede a una etapa de diseño, donde se determina que la herramienta será implementada como un módulo en la plataforma Ucampus, siguiendo su arquitectura ya establecida. Una vez establecido el diseño, se procede a la implementación, llevada a cabo en dos iteraciones. En una primera etapa, se sigue el proceso completo de implementación de un único indicador, `TOTAL_INGRESOS`. Posterior a su implementación, se encuentran una serie de problemas que motivan un rediseño de la herramienta. Luego de éste rediseño, se da una segunda iteración, que sigue la nueva estructura buscando implementar la mayor cantidad de indicadores posibles, lo que incluye finalmente un total de 8 indicadores.

Con el módulo finalmente implementado, se procede a una etapa de validaciones. En ella se estudian principalmente la extensibilidad y escalabilidad del desarrollo. Además de esto, en reemplazo de validaciones mediante user testing, se presenta el trabajo a desarrolladores del centro, donde se obtiene diverso feedback. Finalmente se concluye que se cumplieron los objetivos, generando una herramienta que ayude a la toma de decisiones estratégicas. Por último se plantea posible trabajo futuro, dando prioridad a la realización de user testing.

Agradecimientos

A mis padres, Danilo y Silveria, por su amor incondicional, y por todas las oportunidades que me brindaron. A mi hermanos. A Javier, por abrir todas las puertas que necesité para llegar hasta aquí, y por ser como yo, pero 10 pasos delante. A Danilo, por creer siempre en mí, por su apoyo y ayuda incondicional.

A mis amigos más cercanos. A Felipe, por ser mi amigo desde hace mucho, por siempre tener el meme apropiado para la situación, y por siempre tener tiempo para ir a jugar lo que sea. A Constanza, por escuchar todos los dramas que he necesitado contar alguna vez. A Manuel, por apañar (casi) siempre, por llegar siempre que le dije que era importante. A Guido, por estar siempre ahí para las piscolas que fueran necesarias. A todos los demás, que no alcanzo a mencionar, pero que me acompañaron en algún momento.

A mis profesores guía, Willy y Jocelyn, por su ayuda, tiempo, y sobre todo paciencia, en el desarrollo de esta memoria.

Tabla de Contenido

Índice de Tablas

Índice de Ilustraciones

Capítulo 1

Introducción

1.1. Contexto y Oportunidad

En la actualidad, gracias al uso de plataformas computacionales utilizadas para gestión y administración, prácticamente cualquier organización genera constantemente un elevado flujo de datos. Esta cantidad de información existente en las organizaciones ha motivado el esfuerzo por hacer uso de estos datos para el mejoramiento de las organizaciones mismas. El área del conocimiento resultante se denomina *Business Intelligence* (BI), y comprende técnicas, estrategias, tecnologías y otros, dedicados a obtener conocimiento a partir de los datos existentes en las organizaciones [1].

Este tema podría ser de gran interés para Ucampus, un centro tecnológico perteneciente a la Universidad de Chile y a cargo de la Facultad de Ciencias Físicas y Matemáticas, y encargada del desarrollo de las plataformas U-Cursos[1] y Ucampus[2]. Este interés viene dado por los grandes volúmenes de datos que almacenan producto del uso de sus plataformas por distintas instituciones de educación superior a quienes prestan servicios. Entre la información presente en estas plataformas se encuentran datos como planes de estudios o cursos inscritos por alumnos, puntajes de ingresos de los mismos, y publicaciones de investigación de los académicos, entre varios otros. En general, esta información es obtenida y utilizada en distintos procesos de gestión, y no son aprovechados con fines analíticos.

Dado lo anterior, es lógico cuestionar qué información se puede obtener a partir de los datos existentes, y que sea relevante para las instituciones de educación superior. En este sentido, se destaca la necesidad de información generada por los procesos de acreditación de las instituciones. Dichos procesos son regidos por la Comisión Nacional de Acreditación (CNA), y requieren que las instituciones informen una serie de elementos, tales como la dotación académica, la cantidad de estudiantes, y resultados de proyectos de investigación como papers, solo por nombrar algunos [4].

Tomando esto en cuenta, se establece la oportunidad de desarrollar alguna herramienta

[1] `https://www.u-cursos.cl/`
[2] `https://ucampus.uchile.cl/`

que utilice la variedad de datos obtenidos mediante la operación de las plataformas U-Cursos y Ucampus, para obtener información útil para las instituciones, como sus estados con respecto a ciertas métricas establecidas para procesos de acreditación, o para seguimiento de metas estratégicas de las instituciones de manera general. Cabe notar que en la actualidad existen ciertas funcionalidades en las plataformas que se acercan a lo aquí propuesto, pero tienen ciertas carencias que les impiden ser consideradas como respuesta a la oportunidad aquí planteada, las cuales serán revisadas con detenimiento más adelante.

1.2. Trabajo de memoria previo

El trabajo en este informe explicado utiliza como base lo planteado por el estudiante de ingeniería civil industrial de la Universidad de Chile, Gabriel González, en su propio trabajo de memoria [8]. Dicho trabajo estudia la posibilidad de implementar una herramienta que permita realizar seguimiento de metas estratégicas para instituciones de educación superior, utilizando la variada información que estas instituciones tienen en la plataforma Ucampus. Esta herramienta se constituye como un panel compuesto de indicadores, los cuales pueden ser valores numéricos, tablas o gráficos que presentan alguna variable de importancia para las instituciones, como por ejemplo ingresos de estudiantes por género o por región, o cantidad de publicaciones de investigación realizadas por profesores.

Como producto de este trabajo, se generan 3 principales resultados de importancia para el trabajo explicado en este documento. Primero, se establece la necesidad y utilidad de una herramienta que permita el seguimiento de metas estratégicas en la plataforma Ucampus. Segundo, se genera una lista de indicadores a incluir en dicha herramienta, seleccionados dada su importancia y transversalidad para instituciones de educación superior. Por último, se presenta una serie de bocetos de interfaz sobre cómo se podría estructurar la herramienta y cómo los usuarios podrían interactuar con ella. De estos 3 resultados, el primero es importante como justificación de la presente memoria, mientras que los 2 últimos servirán como guía en este nuevo proyecto, que incluye un rediseño de ellos junto con su posterior implementación.

Es importante también considerar las limitaciones de este trabajo previo. Para efectos de la presente memoria, es importante considerar que el trabajo de González no verifica la presencia en la plataforma Ucampus de los datos necesarios para construir cada uno de los indicadores que propone, es decir, no verifica su factibilidad técnica. Por lo tanto, es posible que algunos de los indicadores seleccionados, considerados relevantes para el seguimiento de metas estratégicas, no sean implementables dado que podría no encontrarse una fuente de información adecuada.

Más adelante se presentará una explicación más detallada de este trabajo previo y como se utilizará a lo largo del presente trabajo.

1.3. Trabajo a realizar

Con el fin de responder a la oportunidad mencionada en la sección 1.1, el proyecto relatado en este informe consiste en el diseño, implementación, y validación de una herramienta que presente una visión global del estado de la institución y facilite la toma de decisiones estratégicas. Para esto se usarán los datos de todo tipo que se encuentran disponibles en las bases de datos de la plataforma Ucampus. Es decir, se buscará concretizar el trabajo presentado por González en su memoria [8]. Esto implicará encontrar fuentes de datos para los indicadores, establecer los procesos necesarios para consultarlos, y por último, mostrarlos a los usuarios.

Como se comentó anteriormente, el presente trabajo utiliza como base el trabajo previo de González [8], pero dados los alcances de este, se deben tomar en cuenta ciertas consideraciones. En primer lugar, dado que los indicadores presentados en dicho trabajo no incluyen fuentes de datos, será necesario buscarlas, en caso de no encontrarlas, el indicador no será considerado en la implementación. Por otro lado, las interfaces presentadas como resultado de dicho trabajo deberán ser modificadas según se estime conveniente en el transcurso del presente trabajo de memoria con el objetivo de presentar a los usuarios la información de la manera más clara posible. En general, los resultados presentados por Gónzalez serán útiles como una base, pero podrán ser modificados en las etapas de rediseño e implementación según se considere necesario.

Los resultados del trabajo a realizar podrían ser de amplia utilidad para las instituciones que la utilicen, pues permitiría conocer rápidamente su estado a grandes rasgos, ayudando así a los usuarios correspondientes tomar decisiones con el objetivo de acercarse a las metas respectivas. De esta manera, se apoya el mejoramiento constante de las instituciones en distintos ámbitos, impactando así indirectamente a todos sus miembros.

Por otro lado, el desarrollo de este proyecto también tendrá un impacto significativo en el centro Ucampus, pues implica acercamiento a un área de análisis de datos que actualmente no se encuentra tan avanzada como los datos que almacenan podrían permitirles. De esta manera, la implementación del proyecto en cuestión ayuda a convertir la plataforma Ucampus en un producto más atractivos para posibles nuevos clientes del centro, sin considerar los beneficios a las instituciones que ya hacen uso de la plataforma.

1.4. Objetivos

1.4.1. Objetivo General

La principal meta del proyecto planteado consiste en el diseño, desarrollo, y validación de una herramienta que permita a los usuarios de la plataforma Ucampus fácil y rápido acceso a información necesaria para la toma de decisiones estratégicas con el fin de avanzar según los objetivos establecidos por las respectivas instituciones de educación superior. Para esto se usará como base el trabajo de memoria anteriormente mencionado, del cual se determinará un subconjunto de los indicadores propuestos para implementar. Además de la usabilidad de

la herramienta, es necesario considerar la extensibilidad y escalabilidad de la misma, pues en un futuro debe ser posible alterar los indicadores contenidos en ella.

1.4.2. Objetivos Específicos

Para lograr el objetivo general señalado, se definen los siguientes objetivos específicos:

1. Definir el subconjunto de los indicadores del trabajo de memoria previo que sea representativo en cuanto a fuentes de información y visualización de datos.
2. Estudiar los indicadores seleccionados y generar una centralización de los datos necesarios para su cálculo.
3. Diseñar un módulo que permita presentar en Ucampus la información requerida.
4. Implementar el módulo diseñado.
5. Validar el trabajo realizado ante usuarios finales mediante la utilización de user testing.

1.5. Metodología

Para llevar a cabo el trabajo a realizar en el proyecto aquí descrito, se partió con un análisis y diseño inicial, a partir de los cuales se decide que el trabajo se estructurará en base a los indicadores a implementar. Dichos indicadores se implementarán de manera secuencial, con el objetivo de encontrar rápidamente problemas presentes en cualquier etapa del proceso. Con este planteamiento en consideración, se decide llevar el trabajo mediante una metodología iterativo incremental, principalmente en dos iteraciones. Una primera iteración que incluye el análisis y diseño ya mencionados, junto con la implementación de un único indicador completo, es decir desde la extracción de sus datos hasta la presentación al usuario. Luego se produce una segunda iteración, la cual incluye un rediseño y la implementación de un total de 8 indicadores.

En la primera iteración, se escoge un indicador específico para implementar, denominado `TOTAL_INGRESOS`, el cuál refleja los ingresos a las instituciones y será explicado con detalle más adelante en el documento. Durante la implementación de este indicador se lleva registro de los problemas encontrados y como estos surgen desde las decisiones tomadas en las etapas de análisis y diseño. Una vez implementado el primer indicador, y empezar la implementación de un segundo, se encuentran problemas que la impiden, motivando un rediseño casi completo, y empezando con este la segunda iteración.

En esta segunda iteración, que empieza con el rediseño de la herramienta, se modifican severamente variados aspectos de la solución, como la estructura de la base de datos y cómo se calculan los datos de cada indicador. Posterior a este rediseño, se implementan los cambios decididos, junto con 8 indicadores (incluyendo la reimplementación de `TOTAL_INGRESOS`). Una vez listos estos indicadores, se da por terminada la segunda iteración del desarrollo y se

procede a realizar pruebas de carga de los indicadores implementados, junto con validaciones de los desarrolladores del centro Ucampus.

En esta última etapa de pruebas y validaciones, se estudia cómo los cambios entre primera y segunda iteración afectan algunos aspectos de la herramienta desarrollada, tales como escalabilidad y extensibilidad. En conjunto con estas pruebas, se muestra lo implementado a desarrolladores del centro Ucampus con el objetivo de obtener feedback, tanto en cuanto a calidad de código como de facilidad de uso de las interfaces. Con esta etapa se da por terminado el proceso de implementación.

1.6. Estructura del documento

Por último, cabe mencionar la estructura de capítulos de la presente memoria y el contenido de dichos capítulos. Luego de esta introducción se encuentra el capítulo Estado del Arte, el cuál explica las herramientas similares a la que se busca desarrollar presentes en otras plataformas alternativas a Ucampus. Además, se consideran herramientas externas que podrían ser incorporadas a la plataforma como parte del proyecto aquí presentado.

A continuación se encuentra el capítulo Situación Actual, explicando el contexto en el cual se realiza la memoria en el centro Ucampus, en particular la arquitectura de la plataforma, y las fuentes de datos en ella presentes. Junto a esto, se explica en mayor detalle el trabajo de González ya mencionado.

Se prosigue con el capítulo Análisis y Diseño, el cual toma en consideración diversos aspectos del contexto y las necesidades de la solución para explicar el diseño elegido para implementar.

Posteriormente está el capítulo de Implementación, donde se da la explicación técnica de la solución desarrollada, explicando su arquitectura y complejidad, en conjunto con los problemas encontrados y cómo estos motivaron un rediseño, seguido por la implementación de este mismo.

Luego encontramos el capítulo de Validación, en el cual se analizarán distintos ámbitos de la solución implementada, principalmente usabilidad y extensibilidad, incluyendo los comentarios y opiniones de desarrolladores del centro Ucampus sobre el producto desarrollado.

Por último encontraremos el capítulo de Conclusión, donde se comentará sobre el trabajo realizado, tanto el proceso como sus resultados, incluyendo además menciones sobre trabajo futuro que podría mejorar la herramienta desarrollada.

Capítulo 2

Estado del Arte

En este capítulo se divide en dos secciones, primero, en la sección 2.1, se presentan algunas herramientas alternativas a Ucampus, utilizadas por diversas instituciones de educación superior, y como estas responden o no a la necesidad de centralización de información para ayudar a la toma de decisiones.

Posteriormente, en la sección 2.2, se presentan en mayor profundidad el área de *Business Intelligence*, detallando conceptos de particular relevancia para el desarrollo de la presente memoria, y además señalando algunas plataformas de esta área que podría ser utilizadas para cumplir los objetivos planteados.

2.1. Plataformas alternativas a Ucampus

Actualmente las universidades chilenas pueden elegir entre una amplia gama de alternativas para sus requerimientos de gestión académica aparte de Ucampus, como por ejemplo Banner y PowerCampus (desarrolladas por Ellucian) y PeopleSoft (de Oracle), sin contar aquellas instituciones de educación superior que han optado por desarrollar sus propios sistemas. Para el desarrollo de este trabajo resulta interesante observar que características ofrecen estos servicios que atiendan la necesidad que se busca cubrir para el centro Ucampus en el trabajo de memoria a desarrollar. Con este fin se estudiaron ciertas alternativas detalladas a continuación.Tanto la relación entre instituciones y software utilizado, como la cantidad de estudiantes en las instituciones fueron obtenidas desde una investigación de mercado realizada por el centro [3].

Antes de continuar, vale la pena comentar brevemente qué es un software de ERP (Enterprise Resource Planning), puesto que la mayoría de las alternativas a revisar son de este tipo. En general un ERP es un software orientado a centralizar información de todas las áreas de producción o servicios de la institución que hace uso de dicho software, principalmente con el objetivo de permitir la planificación y optimización de recursos [5].

2.1.1. Banner

Banner es una herramienta de ERP desarrollada por la empresa Ellucian [9], que declara estar enfocada en fortalecer el desempeño de las principales áreas de una institución educativa, en particular esto quiere decir que cuenta con módulos enfocados en:

- Estudiantes
- Recursos humanos
- Finanzas
- Ayudas estudiantiles

En particular tanto los módulos de Estudiantes y de Finanzas declaran en sus respectivas páginas la existencia de herramientas enfocadas en generar información relacionada a los objetivos institucionales y su progreso, permitiendo que cada institución defina sus propios indicadores de interés. En la figura 2.1 se muestra un panel de ejemplo obtenido del sitio web de Banner [9]. En la figura se presentan distintos gráficos, mostrando información relevante para la organización, como ingresos y gastos en un gráfico de barras (parte central izquierda de la figura) o el estado actual con respecto a un presupuesto (parte inferior izquierda). Además se presenta un botón para crear nuevas consultas y visualizaciones, con el texto 'New Query' (parte superior derecha).

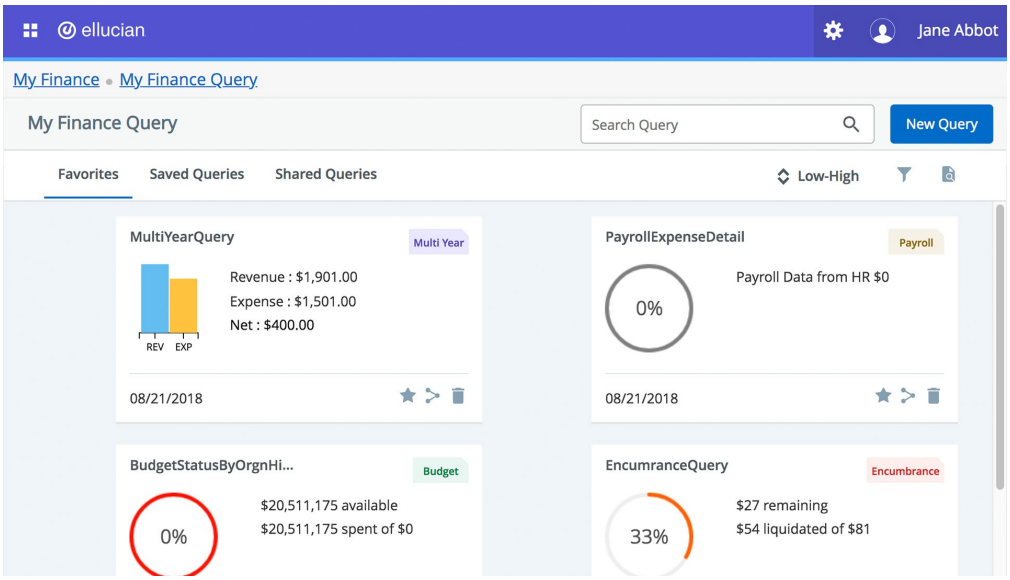


Figura 2.1: Panel Banner

Actualmente en Chile, Banner es utilizado por instituciones como Universidad Católica y Universidad San Sebastián, entre otras, y suma un total de 89.079 estudiantes basándose en el alumnado de dichas instituciones.

2.1.2. PowerCampus

Al igual que Banner, PowerCampus es un ERP desarrollado por Ellucian, con la diferencia que este último declara estar orientado para instituciones más pequeñas [12]. De manera similar a Banner, también presenta el uso de paneles para mostrar fácilmente información relacionada al seguimiento de objetivos institucionales, aunque no se especifica si se pueden generar indicadores según las necesidades de una institución específica o solo se pueden seleccionar indicadores prefabricados.

PowerCampus es utilizado en la Universidad Adventista de Chile y llega a un total de 2.274 alumnos.

2.1.3. Ellucian Analytics

Una última alternativa desarrollada por Ellucian, a diferencia de las anteriores, enfocada exclusivamente en obtener información desde los datos, no incluye sistemas de gestión académica y por lo tanto es un complemento a otras herramientas, como los mismos PowerCampus y Banner [10]. Permite amplia personalización en los indicadores a mostrar, tanto en como se crean, calculan y muestran indicadores a partir de todo tipo de información que se le entregue al sistema. En la figura 2.2 se muestra un panel de ejemplo obtenido del sitio web de Ellucian Analytics. Es de particular interés la presencia de filtros en la parte derecha de la figura, permitiendo reducir la información mostrada sólo a segmentos de interés, ya sea por año u otras categorías disponibles.

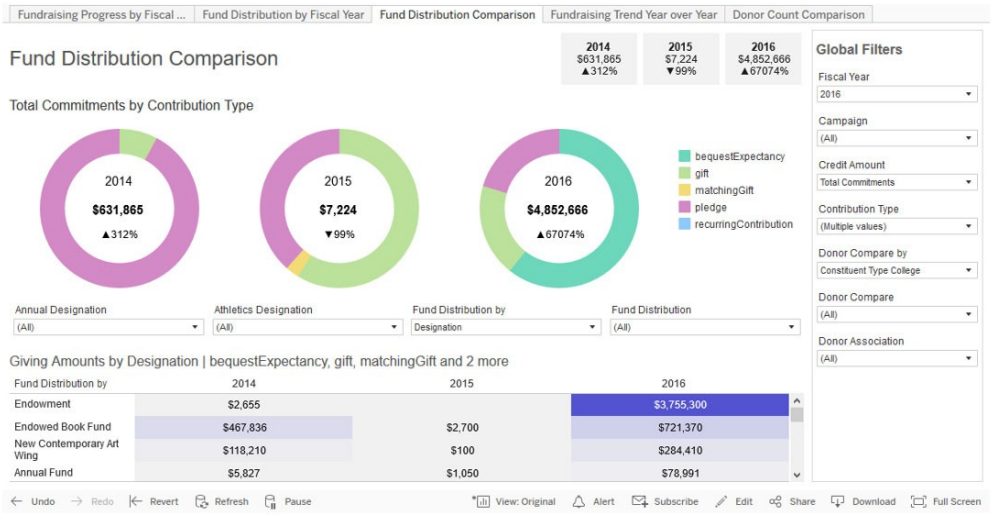


Figura 2.2: Panel Ellucian Analytics

2.1.4. PeopleSoft

PeopleSoft es un ERP desarrollado por Oracle que actualmente cuenta con un módulo denominado PeopleSoft Scorecard [11], el cual se centra en la definición y seguimiento de me-

tas mediante el uso de KPIs (key performance indicator). Este módulo permite definir metas estratégicas propias para cada organización que utiliza la plataforma, establecer los KPI apropiados para cada meta y como estos deben ser calculados, para posteriormente comunicar los resultados de estos indicadores a miembros relevantes de la organización. A continuación en la figura 2.3 se muestra el módulo Scorecard anteriormente mencionado. Aquí se aprecia en la parte central de la figura un gráfico de barras que muestra el porcentaje de logros de distintos KPIs, seguido por una lista de estos mismos, mostrando nombre, valor actual, valor objetivo y porcentaje de logro. Cabe mencionar que éste módulo presenta información similar a lo que se busca en este proyecto de memoria, pero de manera significativamente distinta, dando prioridad a los porcentajes de logro de las metas.

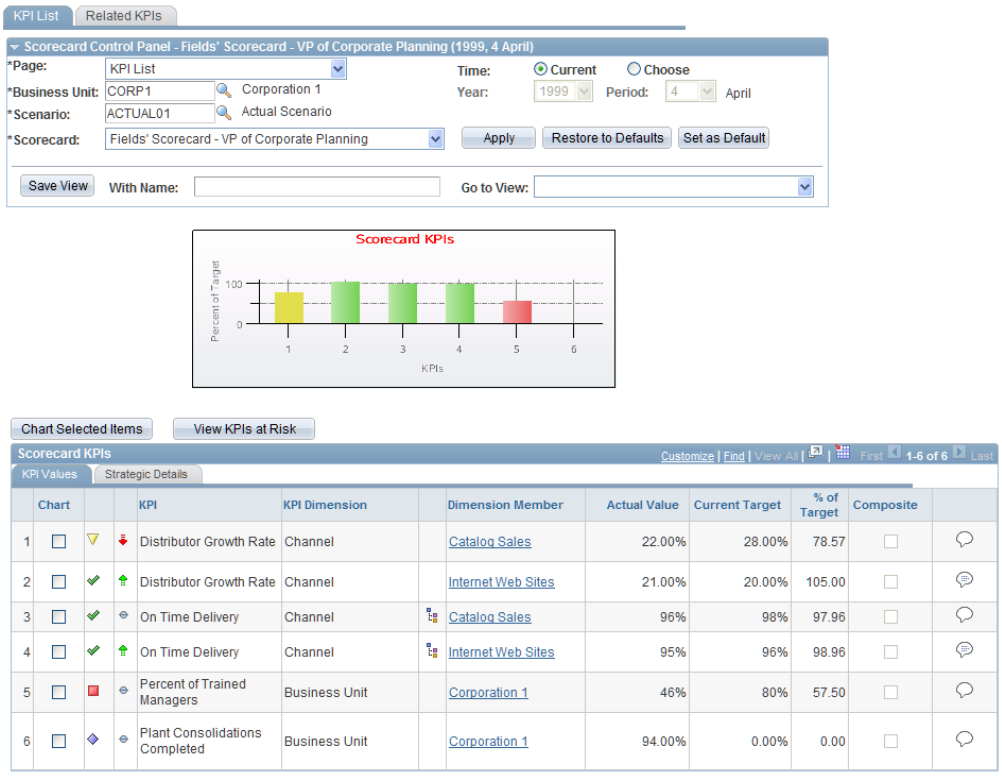


Figura 2.3: Módulo PeopleSoft Scorecard

Por último, cabe mencionar que actualmente en Chile PeopleSoft es utilizado por la Universidad de Santiago y por la Universidad Santo Tomás y llega en total a 46.107 estudiantes.

2.1.5. SISREL ERP Gestión Académica

Esta alternativa desarrollada por SISREL se divide en 9 módulos, entre los cuales los de Admisión y Matrícula presentan alguna capacidad analítica similar a lo que se busca desarrollar [13], llevada a cabo mediante informes y estadísticas. No es posible determinar si esta información se encuentra agrupada a modo de panel, o separada en los módulos respectivos. Nada indica la posibilidad de definir indicadores según las necesidades propias de una institución específica.

Esta opción es utilizada por instituciones como la Universidad de Los Lagos y llega a 9.384 estudiantes.

2.1.6. U+

Una última opción a mencionar es U+, desarrollada por la empresa Bettersoft, la cual cuenta con ciertos módulos enfocados en ayudar la gestión de la dirección superior de las instituciones de educación superior. Uno de dichos módulos es denominado U+ Scorecards [14], este está orientado a la planificación estratégica de la institución y permite la definición de indicadores por parte de las instituciones a partir de los datos que estas ya tienen en sus propios sistemas.

Actualmente U+ es utilizado en Chile por instituciones como la Universidad de Atacama y el Instituto Profesional Esucomex y llega a un total de 104.483 estudiantes.

2.1.7. Resumen

A modo de síntesis, podemos destacar que todas las alternativas estudiadas incluyen, en alguna capacidad, la posibilidad de mostrar estadísticas de interés con el fin de apoyar la gestión de administrativos, pero también es importante que solo algunas de las opciones revisadas incluyen la opción de que cada institución defina sus propios indicadores, en particular solo Ellucian Analytics, PeopleSoft de Oracle, y U+ de Bettersoft ofrecen con seguridad esta característica, mientras que no se sabe con certeza en los casos de Banner y PowerCampus (ambos de Ellucian) y en el caso de la alternativa desarrollada por SISREL, esta capacidad definitivamente no está presente.

Por último, la figura 2.4 muestra la alta presencia en el mercado de las herramientas de gestión con características similares a lo que se busca desarrollar. Es por esto que se considera que el desarrollo de este proyecto daría al centro Ucampus y su plataforma un atractivo extra a posibles nuevos clientes.

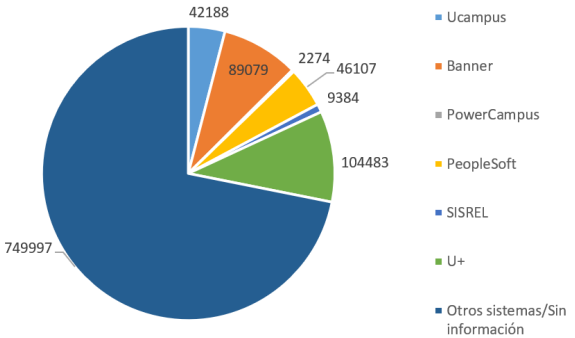


Figura 2.4: Estudiantes 2020 por plataforma

2.2. Business Intelligence

Como se comentó en el capítulo anterior, el proyecto aquí planteado se puede pensar como un acercamiento al área de *Business Intelligence*, y por lo tanto, será provechoso estudiar conceptos y herramientas relevantes pertenecientes a dicha área.

En primer lugar, es necesario establecer que dicha área no está estrictamente definida, sino que corresponde a una serie de procesos y métodos que pretenden entregar una visión clara sobre las organizaciones para ayudar a los usuarios a tomar decisiones más informadas [2]. Estos procesos incluyen, pero no se limitan, a algunos como minería de datos, generación de reportes, análisis estadístico y visualización de datos.

2.2.1. Conceptos relevantes

De toda la variedad de procesos y conceptos relacionados a *Business Intelligence*, para el desarrollo de esta memoria son de particular importancia dos en específico. Primero, los procesos de *Extract, Transform, Load* o ETL por sus siglas, traducido como Extracción, Transformación, Carga. Y segundo, la visualización de datos.

ETL

Se denomina como ETL a los procesos encargados de integrar datos de variadas fuentes a una única centralización. Esto puede ser necesario dadas las variadas bases de datos utilizadas por una organización, por ejemplo, una empresa podría tener bases de datos separadas según el origen de los datos, como información de ventas, de proveedores, contrataciones, etc, y querer utilizar toda esta información para obtener una visión global del negocio. Además, los datos obtenidos mediante la operación de la organización pueden no ser óptimos para fines análiticos, motivando así procesamiento posterior a los datos. Dado esto, se considera que los procesos de ETL son cruciales para el área de *Business Intelligence* [7]. Los procesos de ETL responden directamente a estas necesidades en sus 3 etapas, Extracción, Transformación y Carga.

En la primera de estas etapas, Extracción, se realizan las consultas pertinentes a las distintas fuentes de información necesarias, para posteriormente, en la etapa de Transformación, limpiar y validar las respuestas, asegurando que la información a guardar cumpla con los estándares que la organización pueda requerir. A continuación, y aún en la etapa de Transformación, la información de las distintas fuentes es combinada según sea requerido. Finalmente, en la etapa de Carga, se almacenan los resultados en una nueva base de datos, lista para ser consultada posteriormente.

Visualización de datos

Una vez la información ha sido preparada y almacenada, encontramos la relevancia del segundo concepto de importancia mencionado anteriormente, la visualización de datos. El resultado de los procesos de ETL puede ser toda la información necesaria para presentar el estado de una organización, pero esta es inútil si no puede ser apropiadamente mostrada a los usuarios para que tomen decisiones basadas en esta información. Es por esto que la visualización de datos es tan importante en el área de *Business Intelligence* y su prominencia en plataformas del área, como Power BI, Tableau o Qlik, que serán comentadas un poco más adelante.

Habiendo presentado estos conceptos y su importancia, es fácil entender cómo se relacionan con el trabajo a realizar. Por un lado, los datos presentes en Ucampus se encuentran distribuidos en diferentes bases, y son producto de la operación, por lo tanto probablemente requieran cierto nivel de procesamiento antes de poder ser usados con fines analíticos, lo que nos acerca a los procesos de ETL. Una vez estos datos sean procesados, será necesario presentarlos apropiadamente a los usuarios relevantes, lo cual corresponde directamente con lo comentado sobre visualización de datos.

2.2.2. Plataformas de Business Intelligence

Por último, es útil comentar sobre las plataformas del área ya mencionadas, es decir, Power BI, Tableau y Qlik, aunque estas no son únicas y existen otras alternativas. Estas 3 opciones son muy similares entre sí, presentando un enfoque en la visualización de datos mediante reportes y paneles. Todas son servicios privativos, es decir, requieren el pago de una licencia. Actualmente todas incluyen herramientas de ETL, aunque este no fue siempre el caso, por ejemplo, Tableau no incluía esta capacidad antes de 2018 [15].

A continuación, en las figuras 2.5, 2.6 y 2.7, se presentan reportes de ejemplo de Power BI, Tableau, y Qlik respectivamente. En estas figuras se presenta la amplia variedad de visualizaciones disponibles en estas plataformas, desde algunas relativamente sencillas y comunes, como gráficos de barra, hasta algunas más complejas como el mapa presente en la figura 2.7.

Figura 2.5: Reporte de ejemplo Power BI

Figura 2.6: Reporte de ejemplo Tableau

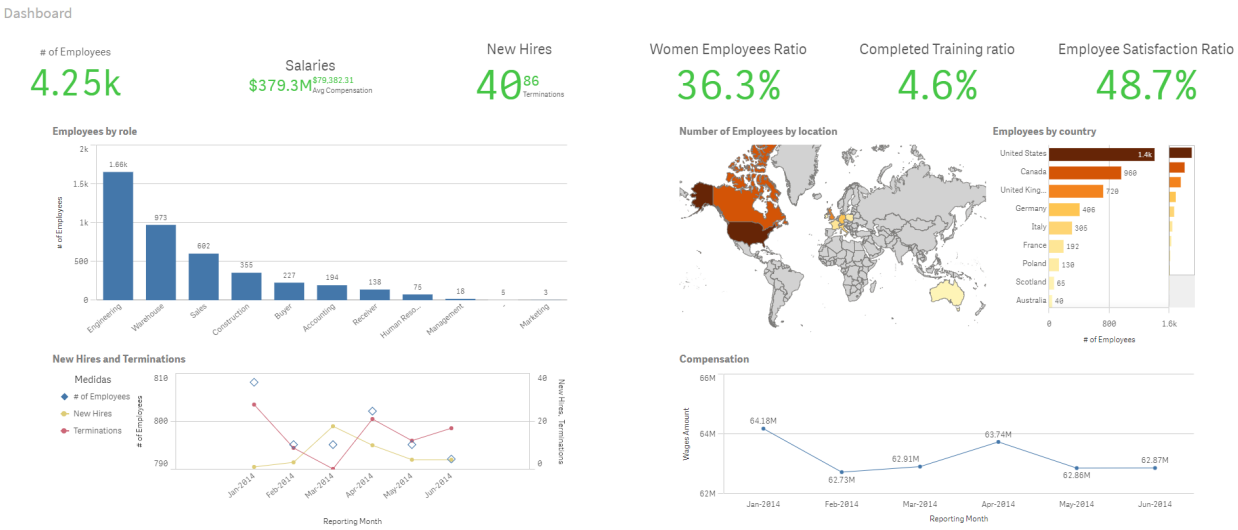


Figura 2.7: Reporte de ejemplo Qlik

Capítulo 3

Situación Actual

En este capítulo se entrega al lector el contexto necesario para comprender la situación previa al desarrollo del trabajo de memoria aquí presentado.En primer lugar se encuentra la sección 3.1, la que explica de manera general el funcionamiento de ésta en cuanto a su arquitectura modular y las funcionalidades que serán necesarias para entender posteriormente el capítulo de implementación del presente trabajo. Junto a esto, se mencionan y explican un pequeño conjunto de bases de datos que serán de particular utilidad como fuentes de información para los indicadores a construir.

A continuación, en la sección 3.2 se presentan los resultados de un trabajo de memoria previamente realizado en el centro Ucampus [8], los cuales incluyen, principalmente, un boceto de diseño y un conjunto de indicadores deseables.

3.1. Ucampus

La plataforma Ucampus es la herramienta de gestión curricular desarrollada por el centro Ucampus, contando con una gran variedad de funcionalidades, desde el catálogo de cursos y la inscripción académica, hasta emisión de certificados y creación de boletines. Dada su gran variedad de funciones, la plataforma Ucampus recopila extensos volúmenes de datos, los cuales se almacenan en bases de datos relacionales específicas según el módulo al cual correspondan. Cabe mencionar también que, dada la presencia de la plataforma en diversas instituciones, cada una de estas instituciones cuenta con su propia instalación, las cuales son independientes entre sí.

3.1.1. Arquitectura modular

Para entender el funcionamiento de Ucampus y el desarrollo de este proyecto es necesario entender, al menos de manera general, la arquitectura de la plataforma. Dicha arquitectura se compone en base a una serie de módulos. En primer lugar, el núcleo de la plataforma, llamado `KERNEL`, se encarga de administrar los grupos de personas presentes en la plataforma

y los permisos que estos tienen, almacenando esta información en su propia base de datos.

Fuera de este núcleo encontraremos módulos que aportan distintas funcionalidades, como por ejemplo aquellos para la inscripción académica (IA), los boletines de notas, y los catálogos de cursos. Estos módulos interactúan continuamente con el núcleo a la hora de verificar si un determinado usuario tiene acceso a ellos. Cada uno de estos módulos puede tener su propia base de datos, en la cual guardan información propia según sea necesario.

Por último, estos módulos pueden interactuar también con otras bases de datos, no relacionadas estrictamente a un módulo, como por ejemplo la base de datos `MUFASA`, la cual contiene información curricular de distintos tipos, necesaria para el correcto funcionamiento de una variedad de módulos, y será explicada más adelante en esta misma sección. Estas interacciones se presentan en la figura 3.1, donde los cilindros representan las bases de datos de cada módulo, los círculos representan la lógica de negocios de los mismos, las líneas continuas representan las interacciones con bases de datos, y las líneas punteadas muestran interacciones entre distintos módulos. Notemos que esta figura representa solo una pequeña fracción del total de módulos y bases de datos presentes en la plataforma.

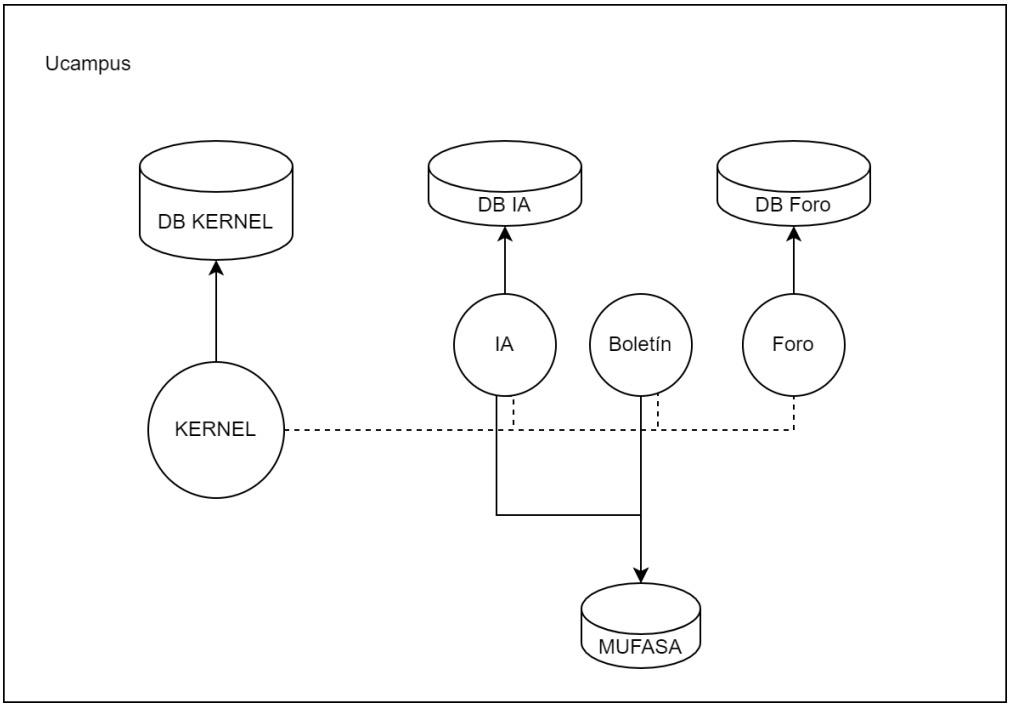


Figura 3.1: Diagrama arquitectura modular de Ucampus

Cabe mencionar que, como se puede ver en la figura 3.1, no todos los módulos tienen su base de datos propia, como es el caso del módulo Boletín, y no todos los módulos interactúan con otras bases de datos, como por ejemplo, el módulo Foro.

En cuanto a la estructura de estos módulos, cada uno está contenido en una carpeta propia dentro del código de la plataforma, la cual a su vez separa el código según su funcionalidad, intentando seguir una arquitectura de ModeloVistaControlador. Para el desarrollo de este trabajo de memoria, es importante considerar cuatro: *cron*, *include*, *template* y *web*. La

carpeta *cron* incluirá *scripts* que deberán ser ejecutados periódicamente, como por ejemplo, aquellos que busquen poblar bases de datos a partir de información recogida desde alguna otra fuente. En general *include* incluirá definiciones de funciones necesarias para el funcionamiento del resto del módulo. *template* contendrá archivos .html que serán utilizados para generar las interfaces con las que el usuario interactuará. Por último, *web* contendrá los *scripts* necesarios para el manejo de datos en las distintas secciones del módulo, por ejemplo, obteniendo los datos necesarios para presentar al usuario mediante los *templates*.

3.1.2. Bases de datos

En cuanto a las bases de datos presentes en la plataforma UCampus, para comprender el trabajo aquí presentado, es necesario tener al menos 3 en consideración, `MUFASA` y `KERNEL` (las cuales ya fueron mencionadas anteriormente) y `UCURRICULUM`. La importancia de estas bases de datos en específico surge desde la cercanía entre la información que almacenan y aquella que se quiere incluir en los indicadores como parte de la solución. Más información sobre estos indicadores se presentará en la sección 3.2.

En primer lugar, tenemos las bases de datos denominadas `MUFASA`. En general cada instalación de la plataforma UCampus tiene su propia instancia de `MUFASA`, las cuales serán muy similares en estructura, con algunas particularidades. Estas bases de datos contienen entre 85 y 90 tablas distintas, variando mínimamente según instalaciones, pues algunas instituciones tienen necesidades especiales en cuanto a qué información almacenar.

Una excepción importante en relación a estas tablas se encuentra en la instalación de la Universidad de Chile. En dicha instalación, cada facultad cuenta con su propia tabla `MUFASA`, teniendo así por ejemplo `MUFASA_AGRONOMIA`, `MUFASA_DERECHO`, `MUFASA_MEDICINA`, etc. Esta particularidad es importante principalmente porque significa que la información a recopilar estará distribuida en una mayor cantidad de tablas, a diferencia de otras instituciones que concentran toda su información en una sola instancia de `MUFASA`

En general todas estas bases de datos contendrán información curricular sobre la institución, como por ejemplo las carreras impartidas, los planes de estudio asociados a cada carrera, y a su vez, los ramos asociados a cada plan de estudio. En cuanto a información curricular de los alumnos, contienen información sobre los ramos tomados y sus estados finales (aprobación, reprobación, eliminación, etc), las carreras inscritas, y las solicitudes de inscripción académica. Al mismo tiempo, también presentan antecedentes e información personal de los alumnos, como sus puntajes de ingreso a las instituciones, RUTs, género, región de origen, entre otros. A continuación, en la figura 3.2 se presenta un diagrama obtenido desde la documentación de las API de `MUFASA` [6]. En este diagrama se presentan entidades presentes en las bases de datos `MUFASA`, con las distintas relaciones entre ellas representadas de manera genérica por flechas. Así se muestra por ejemplo la pertenencia de los planes a las carreras, y de las carreras a las instituciones, como también la relación entre cursos y personas mediante cursos_inscritos.

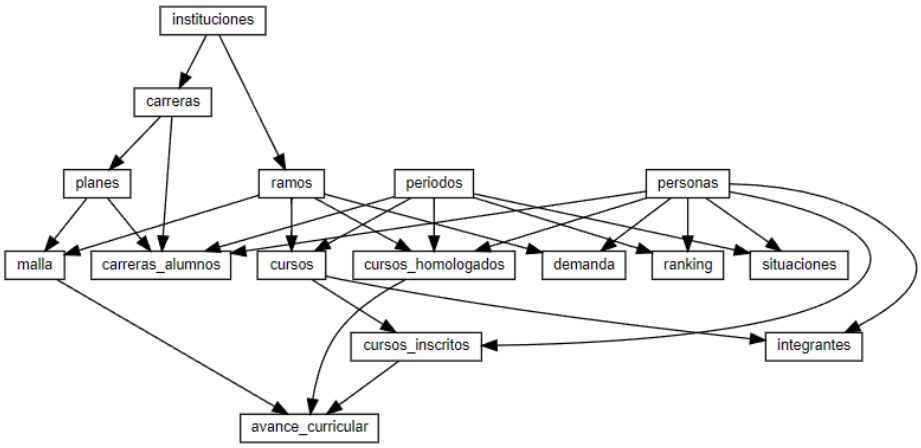


Figura 3.2: Diagrama API MUFASA

Por otro lado, cada instalación también debe tener una base de datos denominada `KERNEL`, la cual contiene información esencial para el funcionamiento de la plataforma, como por ejemplo información relativa a los módulos (nombre, identificador, etc), o también los grupos existentes en la plataforma y la pertenencia de personas a estos grupos. En general, los grupos son conjuntos de personas que comparten alguna característica, por ejemplo pertenencia a algún departamento o facultad. Estos grupos pueden indicar también los roles de las personas que contienen, como por ejemplo alumnos o académicos. Dado esto, los grupos serán de particular utilidad en el desarrollo de la memoria pues permitirán obtener fácilmente agrupaciones de personas asociadas a los distintos indicadores que se puedan querer obtener.

Por último, tenemos un tercer tipo de bases de datos presentes en cada instalación de la plataforma, este corresponde a las bases de datos denominadas `UCURRICULUM`. Estas bases de datos contienen información muy variada, y funciona en base a 3 tablas, `QUES`, `QUIENES` y `RELACIONES`, donde `QUES` representa distintos objetos, como proyectos, publicaciones, patentes o ponencias, `QUIENES` representa sujetos, y `RELACIONES` establece lazos de distinto tipo entre `QUES` y `QUIENES`, como por ejemplo autoría o participación. Estas bases de datos son utilizadas para relacionar a los usuarios con sus respectivos proyectos de investigación, y permiten, por ejemplo, mostrarles una lista de sus proyectos en la plataforma Ucampus.

3.1.3. Herramientas Analíticas

Como se mencionó en la sección 1.2, la plataforma Ucampus cuenta con herramientas que permiten generar valor a partir de la gran cantidad de datos allí almacenados, estos son el módulo Estadísticas y las APIs. Lamentablemente, ambas opciones presentan inconvenientes a la hora de generar información útil para la toma de decisiones estratégicas.

Estadísticas

El módulo Estadísticas nace a partir de constantes solicitudes de información por parte de distintos actores en las instituciones, como por ejemplo áreas de bienestar que solicitan la lista de alumnos eliminados, o la lista de estudiantes egresados por parte de secretarías de estudios. Si bien originalmente estos requerimientos se cumplían de manera manual, consultando las bases de datos y generando reportes, a partir de las mismas consultas, se pudo construir un acceso directo para los usuarios a través de enlaces concentrados en un módulo al que se llamó Estadísticas. Con el tiempo, este módulo fue creciendo en funcionalidades, agregando opciones de filtrado y agrupación, entre otras.

Actualmente el módulo Estadísticas ofrece una gran variedad de información, pero presenta dos dificultades principales. En primer lugar, utiliza el mismo modelo de datos que el resto de la plataforma, la cual está enfocada a la gestión y operación más que al análisis, y por lo tanto realizar consultas analíticas requiere cruzar información de distintas fuentes, lo que ralentiza el cálculo. Por otro lado, la información se presenta de manera poco amigable, pues todas las estadísticas entregan solamente tablas con datos, lo que dificulta el consumo de estos resultados. Si bien estos problemas se han intentado solucionar de distintas maneras, el primero de ellos, mediante el uso de memoria caché, y el segundo mediante agrupación de la información según distintos parámetros, esto no ha sido suficiente para que el módulo sea finalmente útil en términos estratégicos.

El resultado final entonces, es un módulo con mucha información, pero lento e incómodo de usar para los usuarios. En la figura 3.3 se muestra una estadística presente en el módulo, mostrando los filtros aplicables para ella y la representación única de los datos resultantes, una tabla.

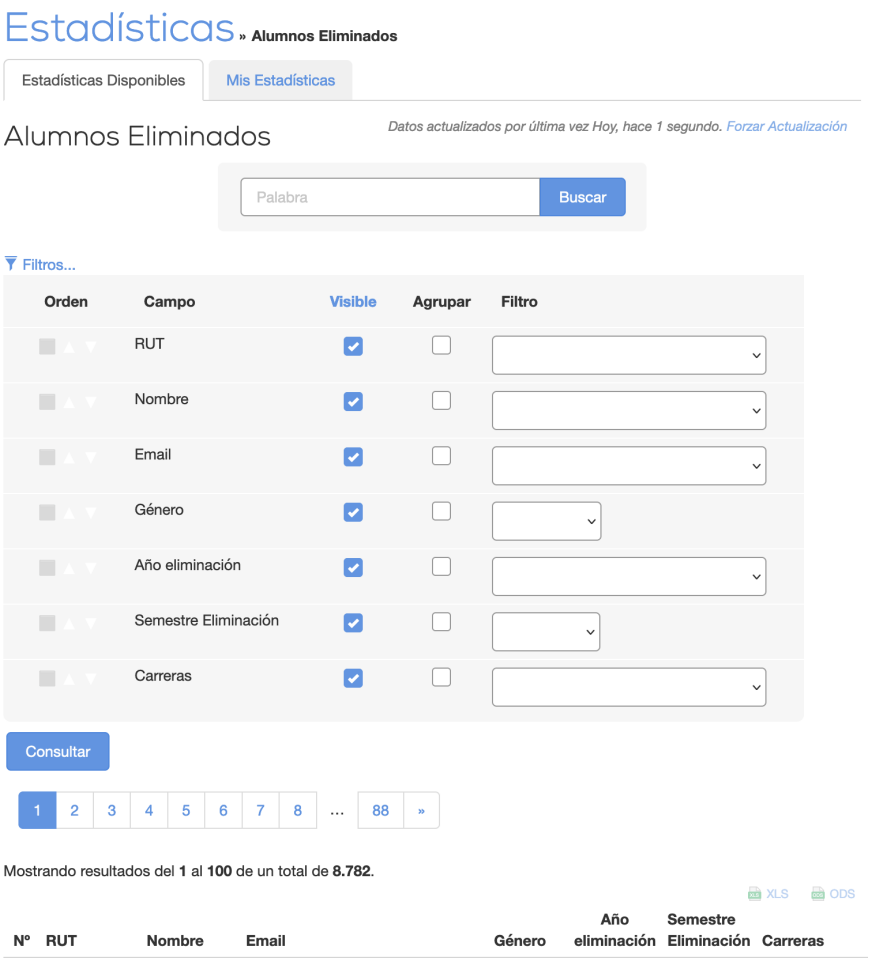


Figura 3.3: Estadística Alumnos Eliminados

APIs

De manera similar al módulo Estadísticas, las APIs surgen como respuesta a las facultades que solicitan funcionalidades mayores en la plataforma, que no persiguen los mismos objetivos que el centro Ucampus. Como por ejemplo, la emisión de ciertos tipos específicos de certificados o procesos de rebaja de carga académica. Ante esto, el centro decide exponer a las facultades su propia información mediante APIs para que estas puedan desarrollar sus propios procesos e incorporarlos en la plataforma.

En la actualidad, existen APIs para distintas fuentes de información, tales como las bases de datos `MUFASA` y `UCURRICULUM` ya mencionadas, las cuales mediante *endpoints* definidos permiten consultarlas. Por ejemplo, la API de `UCURRICULUM` cuenta con el *endpoint* `/persona` la cual entrega una lista con ítems del currículum de un único académico.

El problema con esta funcionalidad radica en que solo se entregan datos y siguiendo las reglas de los *endpoints*, los cuales no son en absoluto amigables a los usuarios y deben ser procesados por las instituciones interesadas para obtener un análisis que permita tomar decisiones. Para esto, las organizaciones necesitarán de sus propios equipos de desarrollo con

la capacidad técnica necesaria, lo que no es siempre el caso.

3.2. Propuesta inicial de indicadores

Como se ha mencionado anteriormente, el trabajo aquí explicado tiene como base un trabajo de memoria [8] realizado por un estudiante de ingeniería industrial, el cual se llevó a cabo en el centro Ucampus durante el año 2020. Dicho trabajo sirvió principalmente como base para el diseño visual del módulo a desarrollar, aunque este fue significativamente modificado durante su implementación, y para saber qué indicadores implementar, los cuales fueron un subconjunto de los incluidos en el diseño inicial.

3.2.1. Diseño de interfaces

El diseño planteado en la memoria mencionada consiste básicamente en un panel con distintos gráficos o números que muestran los valores de distintos indicadores, dichos indicadores se dividen en distintas secciones, dependiendo de a qué corresponde la información presentada, estas secciones son Ingresos, Estudiantes, Docentes, e Institucional.

En la figura 3.4 se muestra un bosquejo de cómo se presentarían las secciones en la vista inicial del módulo. Nótese que las figuras presentadas en esta sección (3.4, 3.5 y 3.6) son obtenidas del trabajo de memoria previo [8]. Además de las secciones (con un indicador representativo para cada una de ellas) se puede ver un grupo de selectores en la esquina superior derecha, los cuales tienen por objetivo filtrar la información mostrada en indicadores de distintas maneras, ya sea para años, grados, campus o carreras específicas. Al interactuar con los títulos de las secciones ya mencionadas se accede a estas, mostrando el conjunto de indicadores correspondiente.

Figura 3.4: Vista inicial, obtenida de trabajo previo

Por brevedad, aquí se presentará solo la sección Ingresos, pero las cuatro secciones compar-

ten muchos elementos, y su principal variación consiste en cómo se muestran los indicadores, ya sea mediantes valores numéricos, tablas o gráficos. Esta sección, cuyo diseño preliminar se muestra en la figura 3.5 presenta indicadores relativos a los estudiantes que ingresaron en un año específico, determinado por el selector de año, a las unidades académicas señaladas por el resto de los selectores. Entre estos indicadores encontramos información sobre la cantidad de ingresos por género, región de origen, tipo de establecimiento de educación media, tipo de ingreso, NEM promedio y puntaje ponderado promedio, indicados por las letras A, B, C, D, E y F respectivamente en la imagen.

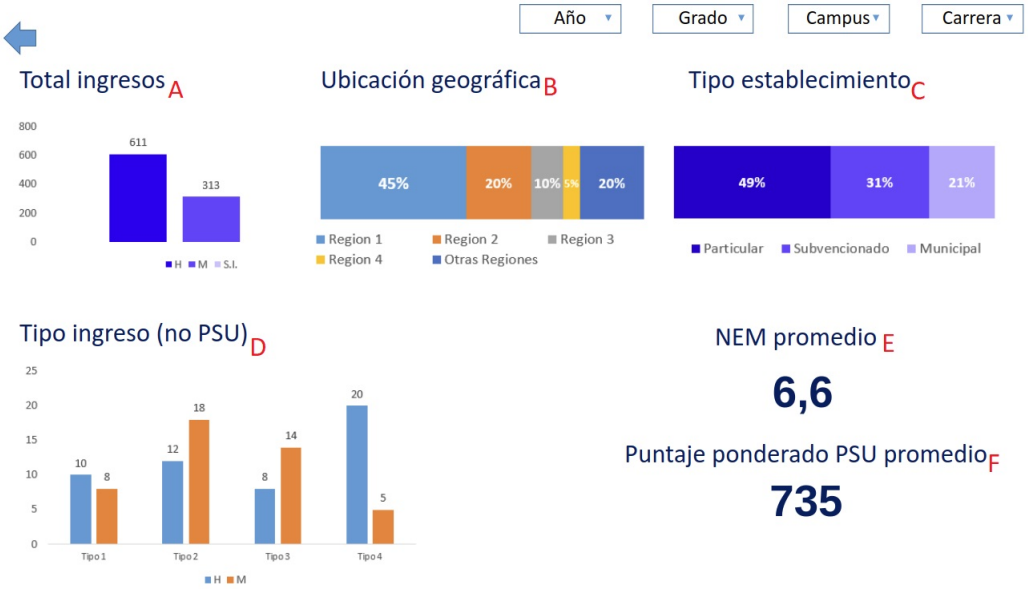
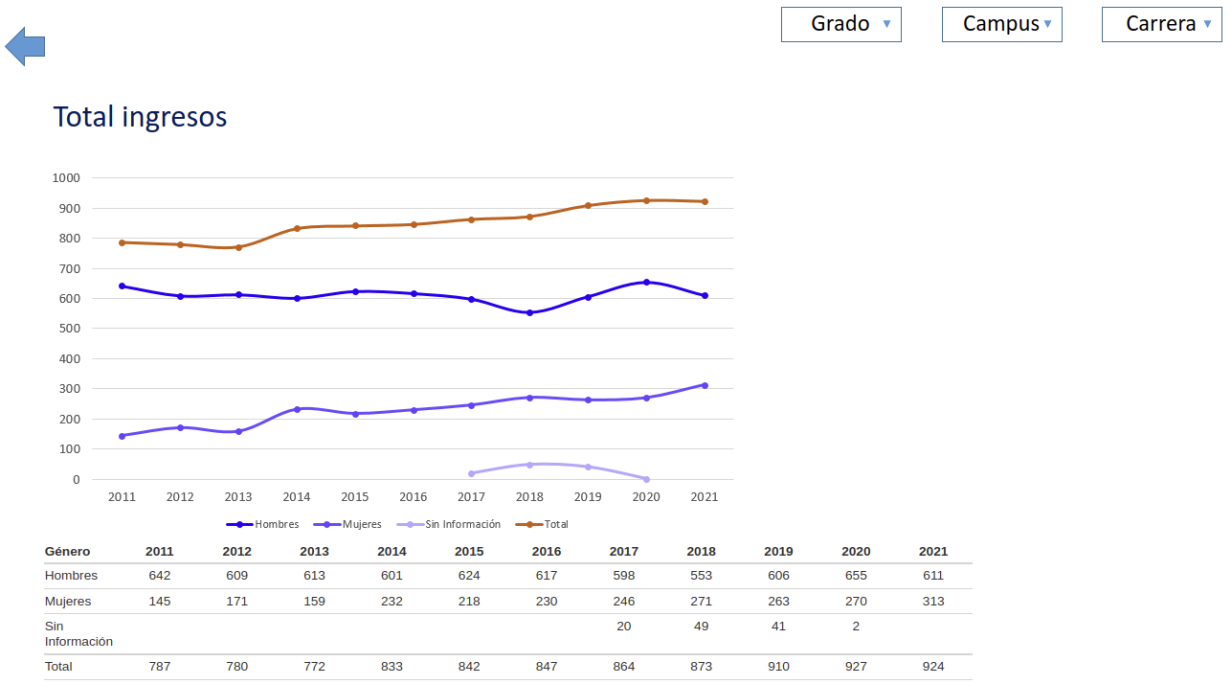


Figura 3.5: Sección Ingresos, obtenida de trabajo previo

En cuanto a las interacciones que podría tener un usuario, además de los selectores ya mencionados, se presenta una flecha, que sería utilizada para volver a la vista inicial descrita anteriormente. Por otro lado, también sería posible interactuar con un indicador, lo que redirigiría a una nueva vista presentando datos históricos para el indicador seleccionado.

En la figura 3.6 se muestra esta vista para el indicador Total ingresos, se presenta un gráfico de líneas, donde el eje X señala los años para los cuales se tiene disponible la información, y el eje Y señala los ingresos por género. Bajo el gráfico se encuentra una tabla presentando la misma información que el gráfico. Por último cabe mencionar que, dado que se quiere mostrar información para distintos años, el selector de año ha sido removido de la esquina superior derecha.



Género	2011	2012	2013	2014	2015	2016	2017	2018	2019	2020	2021
Hombres	642	609	613	601	624	617	598	553	606	655	611
Mujeres	145	171	159	232	218	230	246	271	263	270	313
Sin Información							20	49	41	2	
Total	787	780	772	833	842	847	864	873	910	927	924

Figura 3.6: Vista histórica Total ingresos, obtenida de trabajo previo

En general, la memoria previamente realizada [8] presenta un bosquejo de diseño, el cuál será útil como punto de partida a la hora de diseñar las interfaces que serán efectivamente implementadas en el transcurso del presente trabajo de memoria.

3.2.2. Indicadores

Además de presentar un boceto para el diseño, la memoria [8] también propone un total de 31 indicadores, divididos entre las 4 secciones. A continuación se presentan algunos de ellos, uno para cada sección, para ejemplificar qué clase de información contiene cada sección. Además el listado completo se presenta en el anexo.

- Ingresos - `TOTAL_INGRESOS`: Muestra cuántos nuevos estudiantes ingresaron a la unidad académica indicada, en el año seleccionado.
- Estudiantes - `TOTAL_ESTUDIANTES`: Cantidad total de estudiantes en una unidad académica, independiente de su año de ingreso.
- Docentes - `TOTAL_DOCENTES`: Cantidad de docentes asociados a una carrera/facultad.
- Institucional - `PUBLICACIONES_INVESTIGACION`: Cantidad de publicaciones realizadas a lo largo del año seleccionado por parte de académicos de la unidad seleccionada.

Como se mencionó anteriormente en la sección 3.1, la plataforma Ucampus cuenta con una variedad de bases de datos, las cuales ya fueron comentadas en la sección 3.1. En general,

cada indicador podría nutrirse de fuentes de datos distintas, por ejemplo, la mayoría de los indicadores de la secciones Ingresos y Estudiantes deberá ser obtenida desde las bases de datos `MUFASA`, mientras que la sección Institucional obtendrá la información necesaria principalmente desde las bases de datos `UCURRICULUM`.

3.3. Resumen

En este capítulo se revisó el contexto de Ucampus previo al desarrollo del trabajo de memoria, en particular la ausencia de una herramienta de apoyo a la toma de decisiones estratégicas, lo que plantea una necesidad de mejora para la plataforma. En conjunto con esto, se presentaron elementos importantes a tener en consideración sobre Ucampus, como su arquitectura y algunas bases de datos relevantes para este trabajo de memoria . Luego se comentaron ciertos aspectos de un trabajo de memoria previo [8], el cual plantea un boceto básico para suplir dicha necesidad, generando así una oportunidad para mejorar la plataforma Ucampus.

A continuación, en el capítulo de Análisis y Diseño, se toman en cuenta estas consideraciones para diseñar un solución concreta para ser implementada en el transcurso del trabajo de memoria.

Capítulo 4

Análisis y Diseño

En el siguiente capítulo se realizará un análisis del problema en el contexto planteado en el capítulo 3. Para esto, se plantean en la sección 4.1.1 los tipos de usuario que habrían de utilizar la herramienta a desarrollar y los casos de uso que le darían a esta. A continuación en la sección 4.1.2 se explican ciertas consideraciones no funcionales necesarias para poder considerar una solución exitosa en cuanto a la problemática planteada, principalmente en cuanto a escalabilidad. Luego, en la sección 4.1.3, se estudia la posibilidad de responder a las necesidades planteadas mediante la utilización de plataformas de *Business Intelligence*. Por último, en la sección 4.1.4 se detalla que partes de lo planteado en la memoria previa [8] se desea implementar en el presente trabajo de memoria

Una vez considerados los puntos anteriores de análisis, se procede a explicar la solución alcanzada en términos de diseño. Partiendo en la sección 4.2, se presentan arquitectura, modelo de datos, y bocetos de interfaces para la primera iteración llevada a cabo durante el trabajo de memoria.

4.1. Análisis

4.1.1. Tipos de usuario y Casos de uso

Como se mencionó anteriormente, los usuarios objetivos son personal administrativo de instituciones de educación superior que utilicen la plataforma Ucampus, pero dichos usuarios pueden ser muy variados y por lo tanto tener distintas necesidades de información. En particular, los clasificaremos según el nivel de la unidad académica a la cual pertenecen, es decir, serán administrativos de un departamento, facultad o universidad. Cada uno de estos usuarios requerirá ver información distinta para la toma de decisiones estratégicas en su unidad académica respectiva, pues no le interesa ver información ajena a su propia unidad.

Por otro lado, en cuanto a las interacciones de estos usuarios con el sistema, se identifican 2 casos de uso distintos. Primero, un usuario administrativo, de una unidad académica de cualquier nivel (Universidad, Facultad o Departamento), desea obtener alguna información

perteneciente a su unidad académica para un año específico. Denominaremos a este caso de uso como Caso de uso 1 o CU1. En el segundo caso de uso, un usuario administrativo (también de una unidad académica de cualquier nivel) desea obtener alguna información perteneciente a su unidad académica de manera histórica. Este caso será denominado Caso de uso 2 o CU2. Estas interacciones antes mencionadas se presentan en la figura 4.1

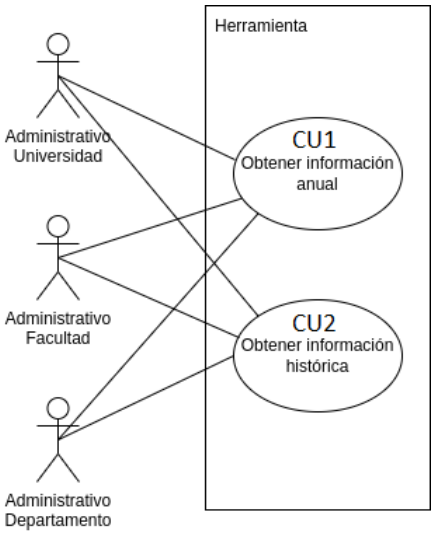


Figura 4.1: Diagrama tipos de usuario y casos de uso

Puede llamar la atención el hecho de que los 3 tipos de usuario tienen las mismas interacciones con los casos de uso, y por tanto podría pensarse que en realidad son un único tipo de usuario, pero es necesario considerar que cada tipo de usuario desea ver información filtrada y agrupada de manera distinta, y por tanto es necesario considerarlos separadamente.

4.1.2. Requisitos no funcionales

Dada las características de la herramienta a desarrollar, surgen principalmente dos preocupaciones. En primer lugar, la escalabilidad, pues la herramienta deberá obtener información a partir de los grandes volúmenes de datos presentes en las bases de datos del centro Ucampus, y deberá hacerlo en una escala de tiempo que no sea significativa para los usuarios. Esto es particularmente importante una vez se consideran las complejidades de los cálculos relacionados a algunos indicadores.

En segundo lugar, la extensibilidad, como se comentará en mayor detalle en la sección 4.1.4, no todos los indicadores señalados en el trabajo de memoria previo [8] serán implementados en el transcurso del presente trabajo, y por lo tanto deberían poder ser agregados a la herramienta a desarrollar en un futuro. Además, también es posible que surja la necesidad de implementar nuevos indicadores no considerados anteriormente. Por lo tanto el producto de esta memoria debería ser fácilmente extensible por desarrolladores de Ucampus y no solo por el autor de la presente memoria.

4.1.3. Uso de plataformas de Business Intelligence

Tomando en consideración lo presentado en la sección 2.2, es lógico cuestionar si la implementación de la herramienta deseada puede ser desarrollada mediante el uso de una de las plataformas presentadas, Power BI, Tableau o Qlik, en vez de un desarrollo a la medida. Para tomar esta decisión fue necesario considerar varios aspectos.

En primer lugar, debemos considerar razones para usar estas herramientas, donde la más destacable sería una reducción en el trabajo necesario, dadas las facilidades que estas plataformas entregan, principalmente en cuanto a las visualizaciones de datos. Como respuesta a eso, se plantea que en la plataforma Ucampus ya existen funcionalidades destinadas a la presentación gráfica de información, mediante el uso de distintos tipos de gráficos. Estos distintos gráficos, aunque muchos menos que los ofrecidos por las plataformas externas, pueden ser extendidos según las necesidades exactas de Ucampus, lo que no sucede en el caso contrario. Por último, se considera que el trabajo en cuanto a procesos de ETL no cambiará sustancialmente, dado que en cualquier caso es necesario generar una centralización de la información, y por lo tanto, las plataformas externas no presentan una ventaja.

Por otro lado, debemos considerar la integración de estas alternativas externas con la plataforma Ucampus. En este sentido, aunque ofrecen algunos métodos de integración, la creación de paneles o reportes y su posterior integración requieren de cierto nivel de conocimiento técnico, el cual actualmente no se tiene en el centro Ucampus, y será necesario para la mantención posterior a la finalización de este trabajo de memoria. En contraste, un desarrollo a la medida puede apegarse a los estándares del centro, facilitando así su mantención y expansión posterior.

Además, se estudiaron, aunque en menor profundidad, otros puntos, como el pago de licencias, dado que las plataformas estudiadas son privativas, o si estas permiten el flujo deseado, es decir, por secciones, y con una vista histórica. Finalmente, se decide que los dos puntos principales mencionados anteriormente, la poca reducción del trabajo, y el esfuerzo requerido para la integración, ya son suficiente para descartar el uso de estas plataformas, y decidir llevar a cabo el proyecto como un desarrollo a la medida.

4.1.4. Alcance con respecto a trabajo previo

El alcance del trabajo de memoria aquí presentado consiste en diseño, implementación y validación de una herramienta de ayuda a la gestión, basada en los resultados presentados en el trabajo de memoria previo [8]. Esto incluye desde diseñar la arquitectura y el modelo de datos, hasta la recopilación de los datos necesarios para el correcto funcionamiento de la herramienta y la implementación de las interfaces correspondientes.

Como se ha comentado con anterioridad, el presente trabajo de memoria utiliza como base una memoria previa [8], el cual plantea un diseño básico y un conjunto de indicadores. La presente memoria busca finalizar dicho diseño e implementarlo finalmente en la plataforma Ucampus. Para cumplir este objetivo, se hace necesario limitar el alcance, es decir, no se buscará implementar todo lo descrito en la memoria previa [8].

En particular, no se implementarán todos los posibles indicadores planteados, si no un subconjunto de estos que se considere representativo, tanto en origen de los datos, como en la forma en que estos serán finalmente presentados. A cambio, en esta implementación se prestará atención a otros factores, en particular, a los requisitos no funcionales mencionados anteriormente, es decir escalabilidad y extensibilidad. Estos dos factores mencionados serán de gran importancia para el diseño de la solución, mientras que no fueron considerados en el trabajo previo.

Por otro lado, en cuanto a interacción con el usuario, se buscará lograr una interfaz similar a la planteada en [8], a modo de panel, dividido en secciones, y con un grupo de selectores que permita filtrar la información según los distintos tipos de usuarios lo requieran.

4.2. Diseño

Tomando en consideración lo presentado en las secciones previas de este capítulo, se intenta abordar el problema en una primera iteración, en la cual se intenta implementar un pequeño subconjunto de los indicadores señalados en [8]. Para explicar el diseño de esta primera iteración, primero se presenta la arquitectura de esta, seguida del modelo de datos utilizado, y finalmente se muestran algunas de las interfaces generadas.

4.2.1. Arquitectura

En primer lugar, en cuanto a la arquitectura, la consideración más importante en esta primera iteración fue la necesidad de una base de datos que almacenase los indicadores listos para ser mostrados. Esto se debe al requisito no funcional comentado anteriormente de escalabilidad, pues dada la complejidad de los cálculos, y los grandes volúmenes de datos sobre los cuales estos se deben realizar, no sería posible consultar las fuentes originales de datos sin generar demoras significativas en los tiempos de respuesta al usuario.

Tomando esto en consideración, se decide crear una nueva base de datos en la plataforma Ucampus, la cual almacenará los valores precalculados de los indicadores para poder responder rápidamente a las solicitudes de los usuarios. Dicha base de datos deberá ser actualizada por *scripts* que consultarán las otras bases de datos presentes en la plataforma Ucampus, y almacenarán allí los resultados. Esta interacción se explica en la figura 4.2.

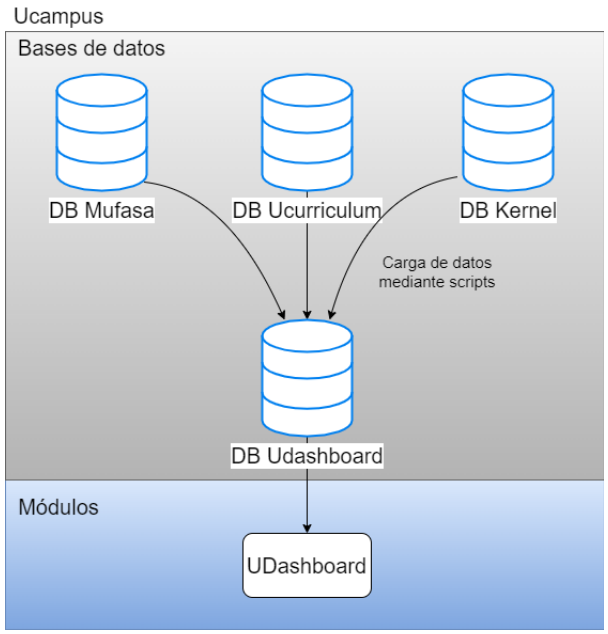


Figura 4.2: Diagrama arquitectura UDashboard

Además, es necesario considerar el proceso a seguir para la creación de un indicador. Este se diseña basado en 4 etapas: Extracción, almacenamiento, consulta y presentación de datos. En este proceso, las dos primeras etapas, extracción y almacenamiento de datos, se encuentran concentrados en los *scripts* de extracción de datos ya explicados anteriormente, y se ejecutan periódicamente, independiente de si el módulo es consultado o no. En estos *scripts*, uno para cada indicador, se establecen las conexiones a las bases de datos respectivas, tanto de origen como de destino de los datos. Luego se consulta la base de datos de origen, y se manipulan los datos para darles el formato que se estime conveniente, para finalmente ser almacenanadas en la base de datos `UDASHBOARD`. Cabe notar que, a pesar de la periodicidad anual de los datos, requerida por los casos de uso CU1 y CU2, algunos indicadores, como el previamente mencionado `PUBLICACIONES_INVESTIGACION`, pueden cambiar de valor durante el año, y por lo tanto se estima que el proceso de carga debería realizarse con una periodicidad semanal.

Por otra parte, las etapas de consulta y presentación de datos se ejecutan solamente cuando un usuario interactúa con el módulo. Cuando esto sucede, se consulta la base de datos `UDASHBOARD` y se manipulan los resultados para darles el formato necesario para generar las visualizaciones. Finalmente, el *template* de cada sección se encarga de utilizar estos resultados para presentar la información al usuario.

Este proceso se explica a grandes rasgos en la figura 4.3, la cual representa las etapas comentadas y cómo se agrupan las funcionalidades según sección o *script*. Se presenta en azul las actividades cuyo código es específico para cada indicador, y en rojo, el código que es específico a cada sección. Esto quiere decir que por cada indicador que se quiera agregar al sistema, será necesario crear nuevos *scripts* que realicen los pasos aquí presentados.

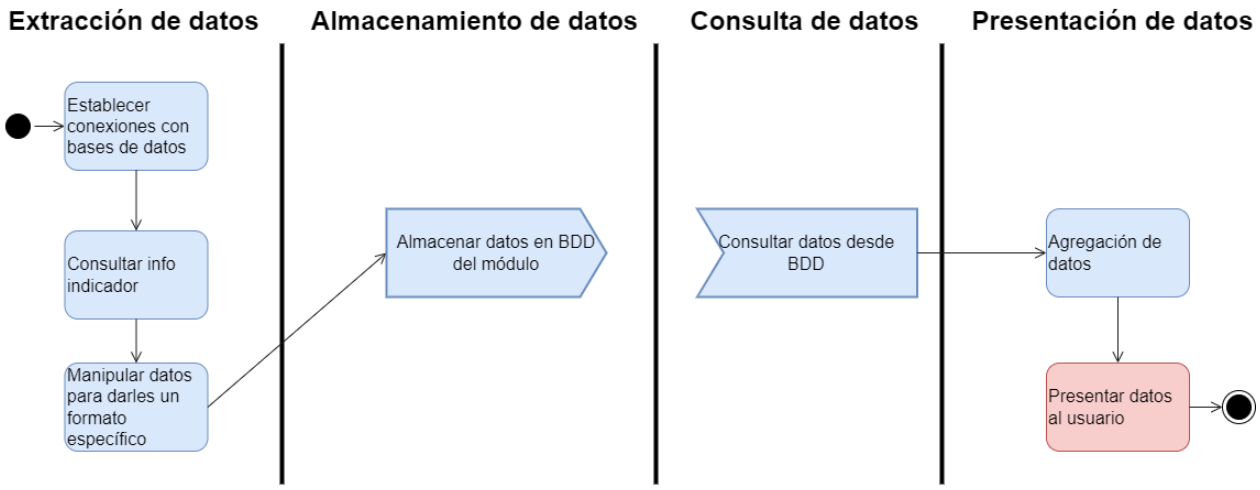


Figura 4.3: Proceso para agregar indicadores, diseño inicial

4.2.2. Modelo de datos

En cuanto al modelo de datos a usar, y qué atributos incluir en él, es necesario considerar algunos puntos. Primero, dado los casos de uso, es necesario incluir el año de cada valor, puesto que el CU1 requiere la obtención de información anual. Luego, dada la necesidad de filtrar la información según unidades académicas dependiendo del usuario que haga uso de la herramienta, se hace necesario incluir información que permita identificar dichas unidades, como el código de la carrera y la facultad a la cual pertenece. Además, dada la presencia de diversos indicadores, será necesario incluir información que identifique a los distintos indicadores de manera única. Por último, se requiere guardar el valor respectivo al indicador para la unidad respectiva.

Con estos elementos en consideración, se determina que el modelo de datos debe consistir de una única tabla, denominada `INDICADOR_CARRERA`, la cual contiene los siguientes atributos:

- `INDICADOR`: Nombre del indicador representado por la fila.
- `AÑO`: Año sobre el cual se calculó el indicador respectivo.
- `VALOR`: Valor asociado al indicador, puede ser valor único, un arreglo, o algún otro, dependiendo de qué indicador se trate.
- `CÓDIGO`: Código asociado a la unidad académica para la cual se calculó el indicador.
- `NOMBRE`: Nombre de la unidad académica respectiva.
- `FACULTAD`: Facultad a la cual pertenece la unidad académica respectiva.

Por último, cabe mencionar que en dicha tabla encontramos una llave primaria compuesta por `INDICADOR`, `AÑO`, `CÓDIGO` y `FACULTAD`.

Con esta arquitectura y modelo de datos presentados, se pretende almacenar la información necesaria para cumplir con los dos casos de uso mencionados en la sección 4.1.1, pues

cada fila de la base de datos contendrá el valor asociado anual a un indicador, permitiendo obtener la información respectiva al caso de uso CU1. En cuanto al caso de uso CU2, dado que cada fila de la tabla representa el valor de una año, para obtener la información requerida para este caso de uso, bastará con obtener todos los valores asociados a un indicador, junto con el año al que dicho dato pertenece.

Como último comentario con respecto a la arquitectura y modelo de datos, es necesario mencionar que ambos elementos consideran la necesidad futura de agregar nuevos indicadores señalada anteriormente como requisito no funcional. Es por esto que el modelo de datos puede nutrirse desde cualquier base de datos que sea necesario para los respectivos indicadores, solo hará falta crear los scripts correspondientes.

4.2.3. Interacciones con el usuario

En cuanto a las interacciones entre los distintos tipos de usuario y la solución, se mantienen ciertas ideas presentes en el trabajo previo [8]. En particular, el módulo diseñado se presenta como un panel con indicadores, valores numéricos, gráficos o tablas, con el objetivo de entregar la información relevante para la gestión de las unidades académicas de manera fácil de comprender. Esto se detalla a continuación, en conjunto con el uso de las figuras 4.5 y 4.4, de elaboración propia.

Dicho panel permitiría filtrar la información mostrada en los indicadores mediante un grupo de selectores, que permitirían fijar un año y unidad académica. Al entrar al módulo, se presentaría una vista con las secciones según las cuales se agrupan los indicadores, y un indicador representativo de cada una de ellas. Al acceder a una de las secciones se presentan los indicadores pertenecientes a ella (ver la figura 4.5).

Dada la presencia de un selector para fijar un año, estas interfaces responden directamente al caso de uso CU1. Por último, al interactuar con los indicadores, se accede a una vista que muestra los valores pertenecientes a todos los años para los cuales se tenga registro en el indicador seleccionado (ver la figura 4.4), generando así una vista histórica que responde directamente al caso de uso CU2.

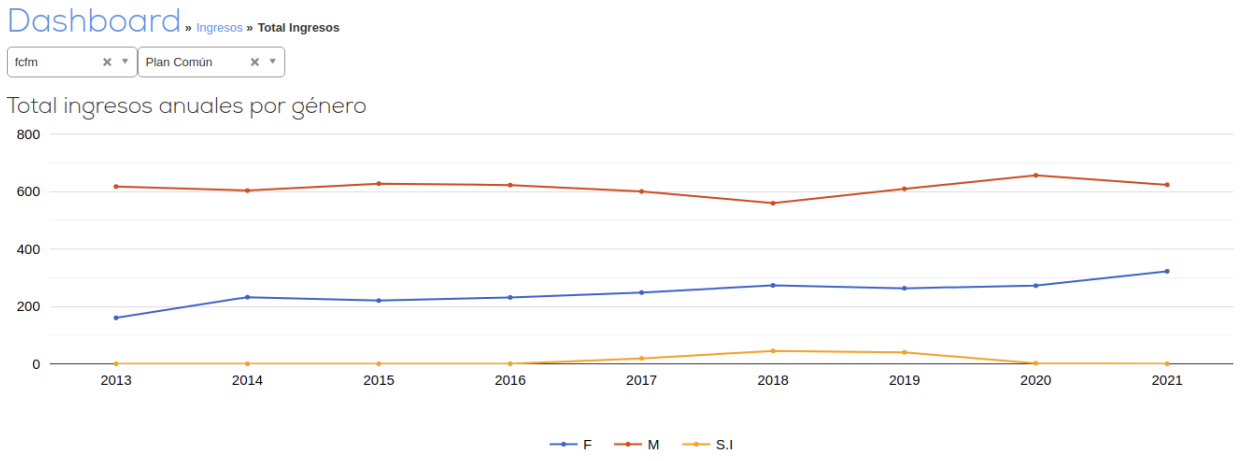


Figura 4.4: Interfaz histórico indicador

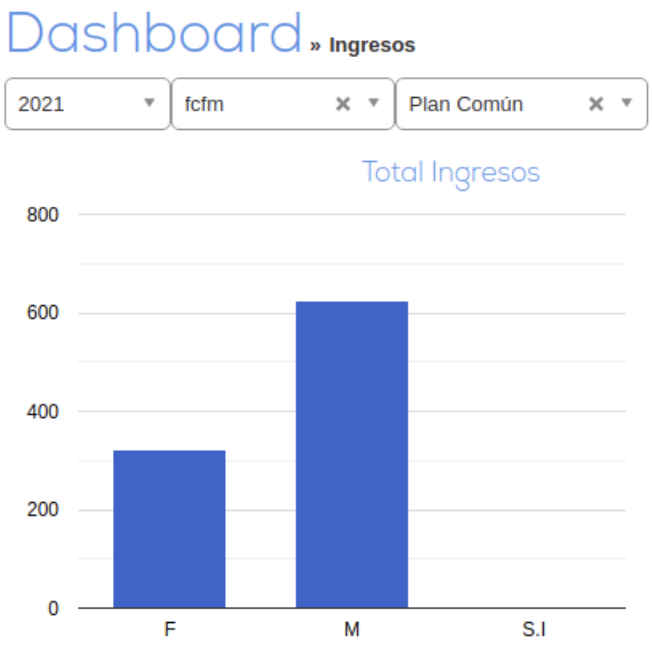


Figura 4.5: Interfaz sección ingresos

Capítulo 5

Implementación

En el siguiente capítulo se detallará la implementación de la herramienta previamente diseñada en el capítulo 4. Para ayudar a la comprensión de la complejidad del desarrollo, este capítulo se divide en tres secciones. Primero, la sección 5.1 describe una primera iteración de desarrollo, más apegada al análisis y diseño vistos en el capítulo 4. Luego, la sección 5.2 detalla los problemas encontrados en la implementación de este diseño y el rediseño resultante. Finalmente, la sección 5.3 explica la implementación de este último rediseño.

5.1. Primera Iteración

Como se comentó en la sección 4.2, no es posible consultar la fuente original de los datos constantemente, esto dada la complejidad de los cálculos y la demora que esto ocasionaría. A raíz de lo cual se hace necesaria la creación de una nueva base de datos, que será poblada mediante distintos *scripts* a partir de las fuentes originales de los datos, para finalmente ser consultada para poder mostrar información al usuario mediante un módulo de Ucampus. Dado esto, dividiremos esta sección en 3 subsecciones, análogas a las 3 partes mencionadas: Base de datos, *scripts* de extracción y módulo.

5.1.1. Base de datos

En esta primera iteración el trabajo en la base de datos se limitó a implementar la base de datos descrita en la sección 4.2, esto se hizo mediante el sistema estándar que ocupa la plataforma Ucampus, el cual corresponde a MariaDB. En dicho sistema se creó una nueva base de datos, denominada `DASHBOARD`, la cual contó con solo una tabla, `INDICADOR_CARRERA`, cuyas columnas ya fueron descritas anteriormente en la sección 4.2 y cuya estructura se puede apreciar en la figura 5.1

	Field	Type
☐	INDICADOR	varchar(255)
☐	AGNO	int(11)
☐	VALOR	text
☐	COD_CARRERA	varchar(255)
☐	NOM_CARRERA	varchar(255)
☐	FACULTAD	varchar(255)

Figura 5.1: Estructura tabla `INDICADOR_CARRERA`

5.1.2. Scripts de extracción

En esta primera iteración, cada indicador nuevo requiere la implementación de un nuevo *script* para extraer los datos correspondientes. Dado que deberán ser ejecutados periódicamente para mantener actualizada la información, y no interactúan directamente con los usuarios, estos *scripts* serán almacenados en la carpeta *cron* del módulo, según se explicó en la sección 3.1.

Para ejemplificar el proceso, en esta sección se muestra en concreto la implementación del *scripts* `TOTAL_INGRESOS`. El objetivo de este indicador es concentrar la información sobre los nuevos ingresos a las distintas unidades académicas, agrupándolos por sexo. Es importante también notar que este indicador será calculado sólo para un conjunto específico de carreras, denominadas en la plataforma Ucampus como carreras iniciables. Estas carreras se caracterizan por ser aquellas a las que un alumno puede ingresar directamente en circunstancias normales, es decir, sin pasar por otras carreras. Por ejemplo, en el caso de la Facultad de Ciencias Físicas y Matemáticas, la única carrera perteneciente a este grupo sería plan común.

Como se comentó en la sección 4.2, en esta primera iteración, los indicadores serán almacenados en la base de datos solo para las distintas carreras, mientras que para el resto de las unidades, es decir facultades y universidad, serán calculados posteriormente mediante las agregaciones correspondientes. En particular, en el caso del indicador `TOTAL_INGRESOS`, para calcular los ingresos a una facultad para un año específico, se sumarán los ingresos de sus respectivas carreras iniciables. De manera similar, para la universidad, el valor del indicador será calculado sumando los valores de las carreras iniciables pertenecientes a sus facultades.

En cuanto a los datos necesarios para calcular el indicador para las carreras ya mencionadas, estos se encuentran en la base de datos `MUFASA` repartidos en 3 tablas distintas `PERSONAS`, `CARRERAS`, `PLANES_ALUMS`. En la tabla `PERSONAS` se encuentran los datos personales de los usuarios de la plataforma, como RUT, nombre y sexo, mientras que en la tabla `CARRERAS` encontraremos los datos de estas, como código, nombre, y si es iniciable o no. Por último, en `PLANES_ALUMS` se señala la pertenencia de personas, representadas por su RUT, a las carreras, representadas por su código.

Una vez comprendido el indicador, podemos revisar su respectivo *script* de extracción, el cual se presenta a continuación en el listado 5.1.

Listado 5.1: Script extracción de datos indicador TOTAL_INGRESOS

```
1  def getIngresos( cursor, carrera, agno ):
2      sql = '''
3      SELECT PERS_SEXO, count(*) as CUENTA,CARR_NOMBRE
4      FROM PLANES_ALUMS, PERSONAS, CARRERAS
5      WHERE P_AL_RUT_ALUM=PERS_RUT
6      AND P_AL_C_CARRERA=CARR_CODIGO
7      AND P_AL_C_CARRERA=%(carrera)s
8      AND substring(P_AL_SEME_INI,1,4)=%(agno)s
9      group by PERS_SEXO
10     '''
11     cursor.execute(sql,{...})
12     resultados = cursor.fetchall()
13     return resultados
14
15 //... #Se omite parte del código que establece las conexiones con bases de datos
16 cursor_mufasa = mufasa.cursor() #Conexión con base de datos de origen
17 cursor_dash = uchile_dashboard.cursor() #Conexión con base de datos de destino
18 //...
19
20 sql_carreras = '''
21 SELECT CARR_CODIGO
22 FROM CARRERAS
23 WHERE CARR_INICIABLE =1
24 '''
25 cursor_mufasa.execute(sql_carreras)
26 carreras = cursor_mufasa.fetchall()
27
28 for annio in annios:
29     for carrera in carreras:
30         cod_carrera=carrera[0]
31         resultados = getIngresos(cursor_mufasa, cod_carrera, annio)
32         nom_carrera = ""
33         res_dic = {
34             "F":0,
35             "M":0,
36             "S.I":0
37         }
38         for tupla in resultados:
39             res_dic[tupla[0] if tupla[0]!="" else "S.I"] = tupla[1]
40             nom_carrera = tupla[2]
41         update_sql = '''
42         INSERT INTO INDICADOR_CARRERA (INDICADOR, AGNO, VALOR, COD_CARRERA, NOM_CARRERA,FACULTAD)
43         VALUES ( 'TOTAL_INGRESOS', %(annio)s, %(valor)s, %(cod_carrera)s,%(nom_carrera)s, %(facultad)s )
44         ON DUPLICATE KEY UPDATE
45         VALOR = %(valor)s
46         '''
47         cursor_dash.execute(update_sql,{...})
```

Notemos que el *script* a ser ejecutado parte en la línea 16, pues el código anterior a eso corresponde a la definición de una función que será utilizada para obtener los ingresos a una carrera en específico para un año en particular. En primer lugar, se establecen las conexiones necesarias a las bases de datos, en particular, `MUFASA`, la fuente de los datos, y `UDASHBOARD`, donde estos serán almacenados para su posterior consulta.

Una vez establecidas las conexiones, se procede a consultar la tabla `CARRERAS`, mencionada anteriormente con el objetivo de obtener las carreras iniciables pertenecientes a la facultad. Posterior a esto, se procede a iterar, primero sobre los años para los cuales se quiera calcular el indicador, y luego sobre las carreras iniciables obtenidas.

Para cada una de las carreras se utiliza la función definida al inicio del código, la cual se encarga de cruzar la información de las 3 tablas mencionadas anteriormente. En particular, se unen las tablas `PLANES_ALUMS` y `PERSONAS` mediante el atributo `RUT` presente en ambas, y

luego se une la tabla `CARRERAS` mediante el código de la carrera presente en esta última tabla y en `PLANES_ALUMS`. Posterior a esto se filtra según el código de la carrera y el año, ambos recibidos como entrada en la función, para finalmente entregar el resultado de la consulta.

Una vez la función retorna y se reciben los resultados de la consulta, se itera sobre estos para almacenarlos en un diccionario donde las llaves corresponden a los posibles sexos. Por último, se procede a almacenar los resultados en la base de datos `UDASHBOARD`.

Con esto, la información necesaria para mostrar el indicador queda lista para ser consultada rápidamente por el usuario mediante el módulo. Es importante mencionar que cada nuevo indicador que se quiera agregar al módulo deberá implicar el desarrollo de un *script* análogo al aquí presentado.

5.1.3. Consulta y Presentación de datos

Una vez los datos de los indicadores correspondientes hayan sido almacenados en la base de datos correspondiente, es necesario que el módulo los pueda consultar y presentar de manera amistosa al usuario. Con este objetivo, y siguiendo lo señalado en la sección 4.2, se genera en el módulo una interfaz a modo de panel, donde el usuario puede acceder a distintas vistas, las cuales presentan los indicadores. Cada una de estas vistas se compone de dos partes, un *script* escrito en PHP, el cual se encarga principalmente de consultar la base de datos y preparar la información necesaria, y un *template* html, el cual tomará estos datos y los presentará al usuario. Según se explicó en la sección 3.1, el *script* PHP será almacenado en la carpeta *web* del módulo, mientras que el *template* html lo hará en la carpeta *templates*.

Según se presentó en la sección 4.2, la interfaz del módulo funcionará en base a 3 vistas distintas:

- Primero, una vista principal, mostrando las distintas secciones según las cuales se dividen los indicadores (Ingresos, Estudiantes, Docentes e Institucional), con un único indicador representativo de cada una de ellas.
- Luego, una vista de sección para cada una de estas secciones, la cual muestra todos los indicadores implementados en la sección.
- Y por último, una vista histórica, mostrando la evolución en el tiempo de un único indicador seleccionado.

A continuación se explica lo implementado para cada una de estas secciones en esta primera iteración del desarrollo.

Vista principal

Como se comentó, la vista principal debiese contener las secciones, mostrando su título y un indicador representativo, junto con enlaces que lleven a las secciones correspondientes. Al mismo tiempo, debiese presentar selectores de facultad, carrera y año, que permitiesen al

usuario filtrar la información presentada en todo el panel según sus necesidades. Es importante notar que, por diseño, se pretende que, al no seleccionar facultad o carrera, se muestren los indicadores a nivel de universidad.

Dado el limitado alcance de esta primera iteración de desarrollo, esta vista solo contiene los títulos de las secciones, presentando el enlace correspondiente sólo en la sección ingresos, pues fue la única que vio uno de sus indicadores implementados. A continuación en la figura 5.2 se presenta la vista en cuestión. Dada su simplicidad y falta de contenido, no se presenta el código respectivo.

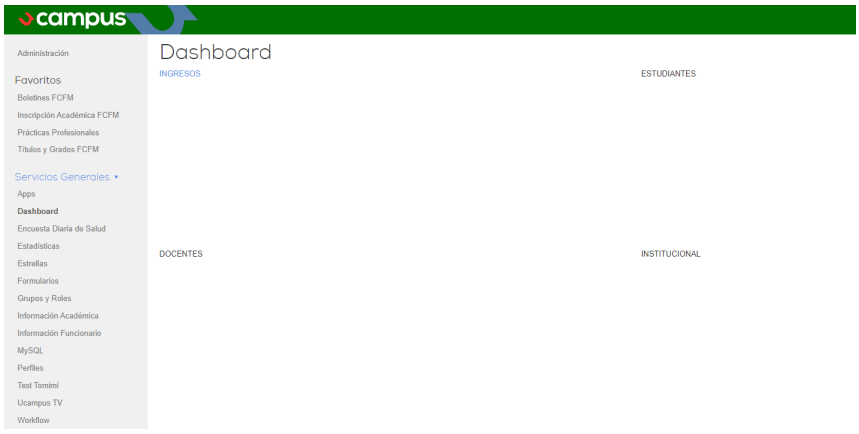


Figura 5.2: Vista principal del módulo

Por último, según se señaló anteriormente, al interactuar con el enlace presente en el título de la sección ingresos se redirige al usuario a la vista de la sección correspondiente.

Vista secciones

Al ingresar a cualquiera de las secciones presentes en la vista principal, se debiese acceder a una vista que contiene los indicadores pertenecientes a ella. Acompañados por selectores de facultad, carrera y año. Al igual que para la vista principal, el funcionamiento de esta vista depende de la creación de un *script* PHP en la carpeta *web*, y un *template* html en la carpeta *templates*. Notemos que cada sección debe tener su propio par de estos elementos, es decir, un *template* y un *script* para la vista de la sección ingresos, un *template* y un *script* para la vista de la sección docentes, etc.

En primer lugar, para el correcto funcionamiento de la vista se necesitan los datos correspondientes, para lo cual se crea el *script* respectivo en la carpeta *web*. Dicho *script* se puede dividir, a grandes rasgos, en dos partes. Primero, se obtienen los datos necesarios para poblar los selectores ya mencionados. Luego, utilizando la información que se reciba del usuario mediante los selectores, se realizan consultas a la base de datos para obtener la información solicitada.Como se comentó anteriormente, en esta primera iteración de desarrollo solo se implementó un indicador, `TOTAL_INGRESOS`, y por lo tanto solo se puede acceder a la sección a la que esta pertenece, Ingresos, la cual se usará como ejemplo del funcionamiento de estas

vistas. A continuación se muestra en el listado 5.2 la primera parte de este *script*, relacionada a poblar los selectores.

Listado 5.2: Obtención datos selectores

```
$facultad = $_REQUEST['facultad'];
$carrera = $_REQUEST['carrera'];

$sql_annos = "
select distinct AGNO
from INDICADOR_CARRERA
";
$res_annos = $link->query($sql_annos);
if( DB::isError($res_ingresos) ) KERNEL::error(ERR_SQL, $sql_annos, 'Error al obtener años');
$annos = [];
while ( $row = $res_annos->fetchRow() ){
    $annos[$row['AGNO']]=$row['AGNO'];
}
$sql_facultades = "
select distinct FACULTAD
from INDICADOR_CARRERA
";
/...
$facultades = [...]

$sql_carreras = "
select distinct FACULTAD, NOM_CARRERA, COD_CARRERA
from INDICADOR_CARRERA
";
/...
$carreras = [...]

if($facultad){
    $carreras=[
        0=>'',
        $facultad=>$carreras[$facultad]
    ];
}
if($facultad and $carrera and !$carreras[$facultad][$carrera] ){
    $carrera=0;
}
```

Primero, mediante una consulta SQL, se obtienen todos los años distintos para los que haya información de algún indicador en la tabla `INDICADOR_CARRERA`. De similar manera, se obtienen las distintas facultades para las cuales exista información de alguna de sus carreras, y luego se obtienen las carreras en sí mismas, con su nombre, código y facultad a la cual pertenecen. Una vez se tienen estos datos, se realizan dos modificaciones a los resultados obtenidos. Primero, si el usuario ha elegido una facultad mediante los selectores, entonces en el arreglo que contiene las carreras se mantendrán solo las que pertenecen a dicha facultad. Luego, si el usuario ha elegido una facultad y una carrera incompatibles, es decir, la carrera no pertenece a la facultad elegida, entonces se ignora la selección de carrera.

Luego se deben obtener los datos para los indicadores de la sección en cuestión. La obtención de los datos se puede dividir a su vez en dos partes, en la primera de las cuales se prepara y ejecuta una consulta SQL según la información que el usuario haya entregado, y en una segunda etapa, se manejan los datos según sea necesario para cada indicador. Al igual que antes, se mostrará el código destinado a obtener los datos del indicador `TOTAL_INGRESOS`. Dicho código se presenta a continuación en el listado 5.3.

Listado 5.3: Datos indicador

```
1  $annos = $_REQUEST['anno'];
2
3  $sql_ingresos = "
4  select VALOR
5  from INDICADOR_CARRERA
6  where INDICADOR='TOTAL_INGRESOS'
7  and AGNO =".$anno."
8  ";
9
10 $sql_ingresos = $sql_ingresos." and AGNO = ".$anno;
11 if ($facultad){
12     $sql_ingresos = $sql_ingresos." and FACULTAD = '".$facultad."'";
13 }
14 if ($carrera != 0 ){
15     $sql_ingresos = $sql_ingresos." and COD_CARRERA = ".$carrera;
16 }
17 $res_ingresos = $link->query($sql_ingresos);
18 if( DB::isError($res_ingresos) ) KERNEL::error(ERR_SQL, $sql_ingresos, 'Error al obtener indicadores');
19
20 $valores_ingresos = [];
21 while($row = $res_ingresos->fetchRow()) {
22     $valores_ingresos[] = $row['VALOR'];
23 }
24 $resultados_ingresos = [
25     "M"=>0,
26     "F"=>0,
27     "S.I"=>0
28 ];
29 foreach($valores_ingresos as $fila){
30     $fila = (array)json_decode($fila);
31     $resultados_ingresos["M"] += $fila["M"];
32     $resultados_ingresos["F"] += $fila["F"];
33     $resultados_ingresos["S.I"] += $fila["S.I"];
34 }
35
36 KERNEL::head(['.'=>'Ingresos']);
37 require( template( '../template/ingresos.html' ) );
38 KERNEL::foot();
```

Como se comentó, esta parte del *script* parte con una consulta SQL básica, que solicita los valores del indicador `TOTAL_INGRESOS` para el año que el usuario haya requerido, pero antes de ejecutarla, añade filtros respectivos a la facultad y carrera según lo que el usuario haya solicitado, esto entre las líneas 3 y 18. Esta primera parte debiese mantenerse similar para los demás indicadores que no fueron implementados.

Luego de ejecutar la consulta y recibir la respuesta, se procede a iterar sobre los resultados, mostrado entre las líneas 29 y 34. Esto es con la intención de generar las agregaciones necesarias para la facultad seleccionada, o en caso de no haber seleccionado ni facultad ni carrera, para la universidad entera. Dada la distinta naturaleza de los indicadores, no todos se debiesen procesar de igual manera, y por tanto esta parte del código sí cambiaría más entre distintos indicadores.

Una vez se han procesado los datos, estos pueden ser usados en el *template* respectivo, también almacenado en la carpeta *templates* del módulo. Para esto se utilizan las últimas 3 líneas de código. En ellas tenemos la función `KERNEL::head`, proporcionado por el núcleo de Ucampus, y la cual se encarga de generar las barras superior y lateral de la plataforma, presentes en la figura 5.2. Luego, en la línea 37 se utiliza el template respectivo a la sección, utilizando como parámetros las variables del *script* en ejecución. Finalmente se llama la función `KERNEL::foot`, la cual también pertenece al núcleo de la plataforma, y que gene-

rará en la vista una barra inferior propia de la plataforma Ucampus. Notemos que estas 3 líneas se repetirán de manera muy similar en todas las distintas vistas presentadas en esta implementación, tanto en esta primera iteración como en la versión posterior.

Idealmente, esta vista presentaría todos los indicadores correspondientes a la sección, acompañados de selectores para filtrar la información, y presentando links en los títulos de los indicadores para redirigir a las vistas históricas de estos mismos. Dado el alcance de esta primera iteración, la vista de la sección Ingresos, única sección implementada, contiene solo un indicador. La vista en cuestión se presenta a continuación en la figura 5.3.

Figura 5.3: Vista sección ingresos

Como se puede apreciar, solo el título del indicador `TOTAL_INGRESOS` contiene un enlace, el cual redirigirá a la vista histórica de dicho indicador.

Vista Histórica

Según se explicó en la sección 4.2, la vista histórica de un indicador debería presentar al usuario los valores del indicador para todos los años para los cuales se tenga registro en un gráfico de líneas. Al igual que en las vistas anteriores, también deberían presentarse selectores que permitan filtrar la información. Dado que se requiere mostrar todos los años, el selector de estos desaparece y se mantienen solo los de facultad y carrera. De manera idéntica a las vistas anteriores, el funcionamiento de esta depende de un *script* PHP y un *template* html. Cabe notar que en esta primera iteración, cada indicador debería tener su propio par de *script* php y *template* html para producir la vista histórica respectiva.

Análogamente a lo visto en secciones anteriores, dado el limitado alcance de la primera iteración, esta sección también se ejemplificará con el indicador `TOTAL_INGRESOS`. Así, en el listado 5.4 mostrado a continuación, se presenta el *script* PHP destinado a obtener los datos necesarios para la vista.

Listado 5.4: Script histórico ingresos

```
1  $sql = "
2  select AGNO, VALOR
3  from INDICADOR_CARRERA
4  where 1
5  ";
6  if ($facultad){
7      $sql = $sql." and facultad = '".$facultad."'";
8  }
9  if ($carrera){
10     $sql = $sql." and cod_carrera = '".$carrera."'";
11 }
12 $res = $link->getAssoc( $sql );
13 if( DB::isError($res) ) KERNEL::error( ERR_SQL, $sql );
14
15 $valores = [
16     'F'=>[],
17     'M'=>[],
18     'S.I'=>[],
19 ];
20 foreach( $res as $agno => $valor ) {
21     $v = UTIL::json_decode( $valor );
22     if( ! $v ) continue;
23     $valores['F'][ $agno ] += $v['F'];
24     $valores['M'][ $agno ] += $v['M'];
25     $valores['S.I'][ $agno ] += $v['S.I'];
26 }
```

Primero, entre las líneas 1 y 13, vemos una lógica similar a la presente en el *script* de la sección Ingresos, con la diferencia que ahora se debe obtener el año asociado a cada valor, dado que no se filtra por dicho parámetro. Luego, entre las líneas 15 y 26 se itera sobre el resultado de la consulta, sumando el valor de cada fila del resultado para obtener el resultado agregado según unidades académicas para cada año.

Una vez preparados los datos, y mediante el *template* html respectivo, se produce la vista presentada en la figura 5.4. En ella podemos ver los elementos mencionados anteriormente: selectores para facultad y año, seguidos de un gráfico de líneas que representa la evolución del indicador en el tiempo.

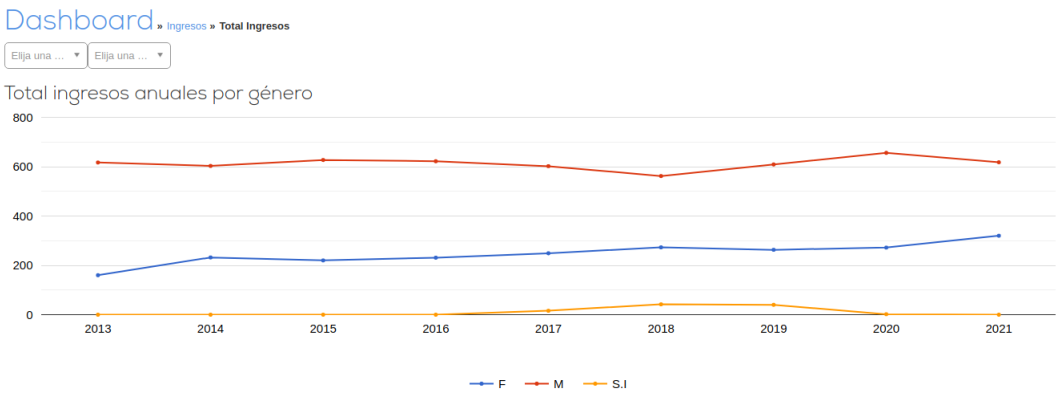


Figura 5.4: Vista histórica indicador total ingresos

5.2. Problemas y Rediseño

Una vez implementadas las vistas de sección e histórica para la sección Ingresos y su correspondiente indicador `TOTAL_INGRESOS` respectivamente, se decidió comenzar con el indicador `PUBLICACIONES_INVESTIGACION`. Dicho indicador fue elegido principalmente por proceder de una fuente de información ajena a la base de datos `MUFASA`, en particular los datos relevantes se encontraban almacenados en la base de datos `UCURRICULUM`. Fue al tratar de implementar este segundo indicador donde se encontraron problemas con cómo se calculan los indicadores y a raíz de esto, con el modelo de datos.

En el indicador previamente implementado, `TOTAL_INGRESOS`, los ingresos de un nivel superior de unidades académicas, como facultad o universidad, se obtenían simplemente sumando los valores asociados a las subunidades académicas respectivas. Se consideró que podían existir alumnos en más de una carrera y quienes por tanto podrían ser contados más de una vez al calcular los ingresos de la universidad, pero se decidió que dicho volumen era muy bajo y por tanto el error que producirían sería despreciable. Pero al usar este mismo método para calcular las publicaciones de investigación, se producirían errores significativamente mayores, pues es común que una publicación tenga autores de distintas carreras o facultades. Por tanto, se hace necesario que los indicadores sean calculados para cada unidad académica, independiente de su nivel.

Junto con esto, se encontró que el indicador `PUBLICACIONES_INVESTIGACION` no se relaciona fácilmente con las carreras, pues un profesor puede hacer clases para alumnos de muchas carreras distintas, y por tanto los trabajos de investigación de dicho profesor serían difíciles de determinar a qué carrera pertenecen. En cambio, se definió que este indicador (y otros a futuro), podrían agruparse en su nivel más bajo por departamento, los cuales, al igual que las carreras, están asociadas a una facultad, y por tanto dicho cambio no altera en gran medida el resto del diseño.

Por otra parte, se debe considerar la estructura jerárquica de las instituciones donde se utiliza la plataforma. En general estas instituciones son universidades, cada una de las cuales cuenta con una serie de facultades, las cuales a su vez imparten una serie de carreras, y, dado lo señalado en el párrafo anterior, también debemos considerar que cada facultad cuenta con una serie de departamentos. Notemos que, aunque resulte natural establecer relaciones de pertenencia entre carreras y departamentos, esta pertenencia no se ve reflejada en la tabla `CARRERAS` de la base de datos `MUFASA`.

Dado lo anterior, podemos establecer que cada carrera o departamento pertenece a una sola facultad, y estas a su vez pertenecen a una única universidad, generando así una estructura de árbol como se muestra a continuación en la figura 5.5.

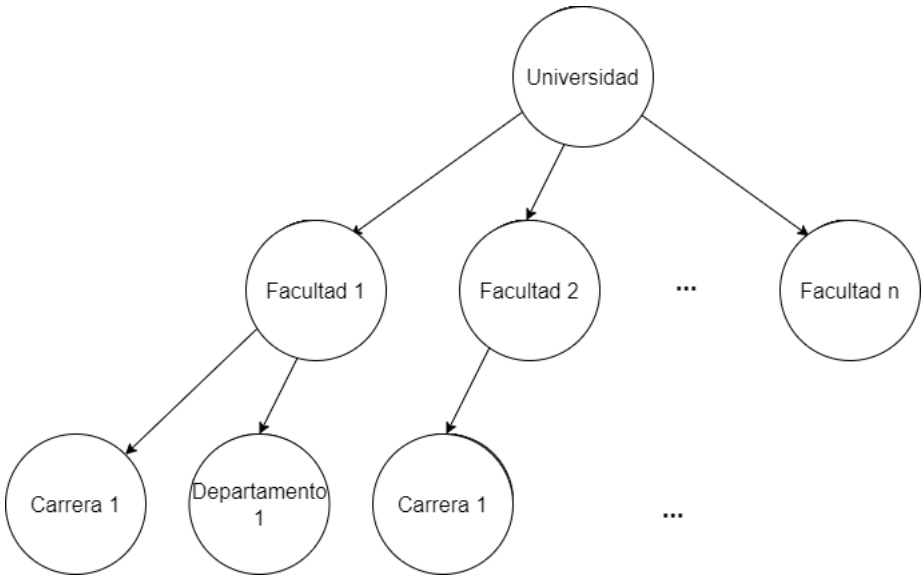


Figura 5.5: Árbol de unidades académicas

Tomando esto en consideración, se tomó la decisión de unificar los selectores de facultad y carrera anteriormente presentados por un único selector de unidades académicas. Para esto, fue necesario un rediseño del modelo de datos que permitiese construir el árbol señalado, para lo cual sería necesario que cada unidad académica en la base de datos guardase información sobre la unidad a la que pertenece.

Luego de todas estas consideraciones, se procede a analizar el proceso necesario para agregar nuevos indicadores, el cual fue originalmente explicado en la sección 4.2. Este diseño tiene, a grandes rasgos, dos problemas principales que van directamente en contra de los objetivos planteados para el proyecto, en particular en cuanto a extensibilidad.

En primer lugar, cada nuevo indicador define prácticamente todo lo necesario para su funcionamiento, pero en realidad hay diversos puntos en común entre los indicadores. Por ejemplo, en cuanto a establecer conexiones con las bases de datos, pues todos los indicadores deberán almacenar resultados en la misma base de datos. De manera similar, a la hora de obtener los valores de un indicador para mostrarlos, todos los indicadores funcionan de manera similar. Por lo tanto, el diseño presentado contiene duplicación de código, lo que en general será un problema si se quieren realizar cambios a cómo se realizan estas conexiones a las bases de datos, sin contar la eventual duplicación de bugs que podría resultar producto de esto. De la misma manera, si se quiere agregar un nuevo indicador, agregar código con un funcionamiento similar al ya existente, lo que complica extender el funcionamiento del módulo.

Con el objetivo de subsanar esta situación y en general mejorar la extensibilidad del código, se rediseña este proceso.

Primero, en cuanto a la extracción de datos y el almacenamiento de los mismos, se refactorizan los *scripts* de extracción de datos para cada indicador, a un solo *script* que se encargará de obtener los datos de todos los indicadores. Este *script* hará uso de funciones específicas a definir para cada indicador que se quiera agregar al sistema, con funciones distintas para cada

nivel de unidad académica que el indicador requiera, guardando de esta manera los datos ya agregados para las unidades académicas mas grandes.

Por otro lado, en cuanto a la consulta de los datos, esta utilizará una única función ya definida, para todos los indicadores, evitando así la duplicación de este código. Luego, en cuanto a la presentación de datos, dado que los datos se guardan listos para ser consultados en todos los niveles de unidades académicas, ya no será necesario agruparlos luego de consultarlos. Por lo tanto, la etapa de presentación de datos se aísla exclusivamente en los *templates* de cada sección, simplificando así los *scripts* de las distintas secciones.

Este rediseño se presenta en la figura 5.6. Notemos que, al igual que antes, se presenta en azul el código específico para cada indicador, y en rojo aquel que pertenece a cada sección. Además, ahora se presenta en verde el código que es utilizado por todos los indicadores. Finalmente, en cuanto a diferencias con la primera iteración, presentada en la figura 4.3, notemos que ahora se ha eliminado el paso de agregación de datos, dado que estos son guardados listos para consultar, y además, tres pasos que anteriormente eran específicos para cada indicador, ahora han sido generalizados para todos los indicadores.

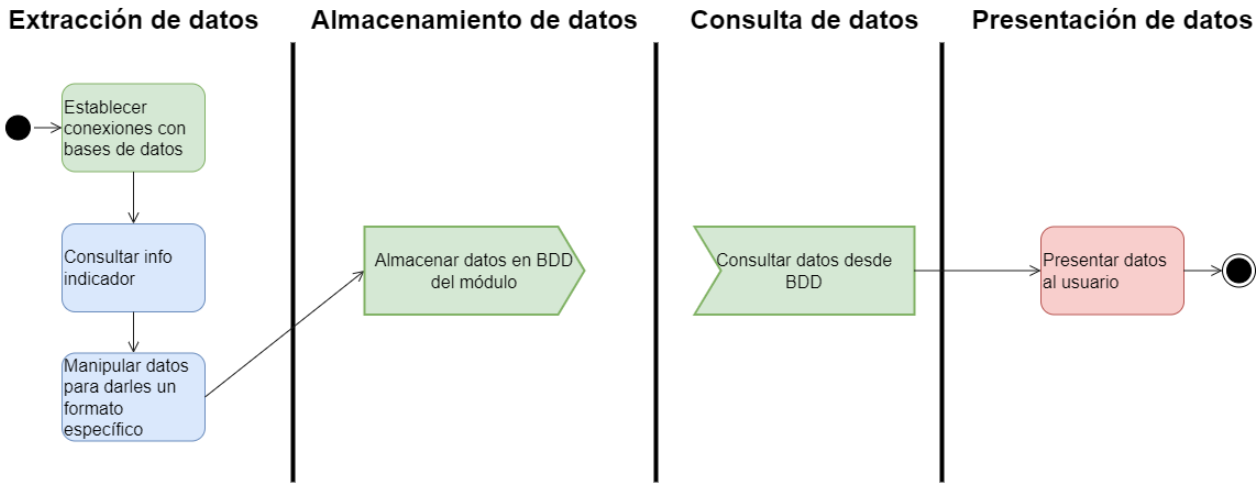


Figura 5.6: Diagrama Etapas segunda iteración

5.3. Nueva Iteración

Considerando el diseño original, los cambios que se le hicieron en la primera iteración, los problemas encontrados y el feedback recibido, se procede a rediseñar gran parte del sistema y a su posterior implementación, la cual se explica en las siguientes páginas, dividido de manera análoga a como se explicó la iteración original.

5.3.1. Base de datos

Para el modelo de datos, los cambios más importantes provienen de dos problemas señalados, primero la necesidad de tener las agrupaciones pre calculadas (a diferencia de la iteración

anterior), y de la necesidad de unificar los selectores a modo de árbol. A raíz de esto, el modelo anterior, que contenía solamente la tabla `INDICADOR_CARRERA`, la cual almacenaba toda la información necesaria, fue reemplazada por 2 tablas, `INDICADORES2` y `UNIDADES_ACADEMICAS2`, quedando así `INDICADOR_CARRERA` completamente obsoleta.

En primer lugar, la tabla `INDICADORES2` contendrá la información relativa a cada valor para los distintos indicadores, es decir, cada fila representará el valor de un año para un único indicador, para una unidad académica específica.

Luego, en la tabla `UNIDADES_ACADEMICAS2` encontraremos la información respectiva a las unidades académicas para las cuales se calculan los indicadores, en particular cada fila contendrá un identificador, un nombre, un nivel, y el identificador del padre de la unidad académica en cuestión, pues como ya se dijo, queremos construir un árbol a partir de las unidades académicas.

Por último, en la figura 5.7 encontramos una comparación entre el modelo de datos anterior y el nuevo diseño, donde se señala en verde la información nueva en el modelo, las líneas continuas indican qué información quedó en que tabla, y por último las líneas punteadas muestran que parámetros fueron reemplazados y por cuáles.

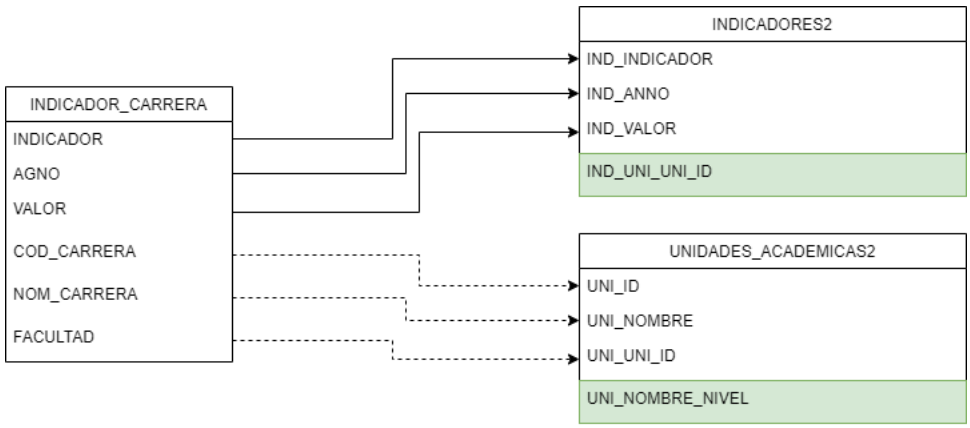


Figura 5.7: Cambios en el modelo de datos

5.3.2. Script de extracción

Como se explicó en la sección 5.2 y en la figura 5.6, en el rediseño de esta etapa del proceso se busca ayudar a la extensibilidad del módulo, principalmente reduciendo y estructurando el código que se necesita para agregar un nuevo indicador, de esta manera, los múltiples *scripts* (uno por cada indicador) fueron reemplazados por un *script* único, al cual se le deben hacer modificaciones mínimas para añadir nuevos indicadores.Al igual que antes, la extracción de datos debería realizarse de manera periódica, y por lo tanto el código respectivo debería mantenerse en la carpeta *cron* del módulo. En cuanto a la periodicidad con la que se extraen los datos, a pesar de la anualidad establecida por el modelo de datos, se estima que debiese ser semanal o al menos mensual. El motivo de esto consiste en la presencia de indicadores que varían constantemente durante el año, como por ejemplo, `PUBLICACIONES_INVESTIGACION` Este nuevo *script* se muestra a continuación en el listado 5.5.

Listado 5.5: Nuevo script de extracción de indicadores

```
1  $bases = InfoNucleo::getParametro( SISTEMA, 'bases_ucampus' );
2  $u = InfoNucleo::getParametro( SISTEMA, 'id' );
3  $titulo = InfoNucleo::getParametro( $u, 'titulo' );
4  if( $u == 'ucampus' ) $u = 'uchile';
5
6  $annio = date( 'Y' );
7  if( $_REQUEST['annio'] ) $annio = (int)$_REQUEST['annio'];
8  $universidad = getOrCreateUAcad( 'universidad.'.$u, $titulo, null, 'Universidad' );
9
10 $indicadores = [ 'TOTAL_INGRESOS', 'INGRESOS_REGIONES', 'NEM_PROMEDIO', 'PUBLICACIONES_INVESTIGACION',
       'TIPO_ESTABLECIMIENTO', 'PRUEBA_PROMEDIO', 'TIPO_INGRESO', 'RETENCION_ACTUAL_1' ];
11
12 foreach( $indicadores as $ind ) {
13    $_f = 'getIndicadorFuente_'.$ind;
14    $_f_u = $_f.'_Universidad';
15    $datos_u = call_user_func( $_f_u, $universidad, $annio );
16    insertIndicador( UTIL::json_encode( $datos_u ), $annio, $universidad['UNI_ID'] , $ind );
17
18    foreach( $bases as $base ) {
19       if( $base == 'ingenieria' ) $base = 'fcfm';
20
21       $fac = InfoNucleo::getParametros( $base );
22       $facultad = getOrCreateUAcad( 'facultad.'.$base, $base, 'universidad.'.$u, 'Facultad' );
23       $_f_fac = $_f.'_Facultad';
24       $datos_fac = call_user_func( $_f_fac, $base, $annio );
25       insertIndicador( UTIL::json_encode( $datos_fac ), $annio, $facultad['UNI_ID'] , $ind );
26
27       $_f_subuni = 'getSubunidades_'.$ind;
28       $subunidades = call_user_func( $_f_subuni, $base );
29       foreach( $subunidades as $subuni => $uacad ) {
30          $_f_sub = $_f.'_Subunidad';
31          $datos = call_user_func( $_f_sub, $base, $uacad['cod'], $annio );
32          insertIndicador( UTIL::json_encode( $datos ), $annio, $uacad['UNI_ID'], $ind );
33       }
34    }
35 }
```

En primer lugar, entre las líneas 1 y 4, se obtiene información sobre la instalación de Ucampus en la que se está corriendo el *script*, en particular se obtienen las facultades asociadas a la instalación ($bases), un identificador para la universidad ($u), y un título para la misma ($titulo). Posteriormente, en la línea 8, se crea u obtiene una fila para la universidad en cuestión en la tabla UNIDADES_ACADEMICAS2. Notemos que el parámetro null entregado en esta llamada representa al nodo padre, y por tanto señala a esta unidad académica como raíz de su respectivo árbol. Como último paso antes de comenzar a cargar los indicadores, se define en la línea 10 la lista de indicadores a calcular.

Desde la línea 12 y hasta el final del archivo, se itera sobre la lista de indicadores, y en las líneas 13 y 14 se obtiene para cada indicador el nombre de la función que se usará para calcularlo para las unidades académicas de nivel Universidad. En las líneas 15 y 16, se invoca esta función cuyo único parámetro será el año, y luego se guardan los resultados en la base de datos para su posterior consulta.

A continuación, desde la línea 18, se itera para cada facultad asociada a la universidad. Para cada una de estas facultades, se crea u obtiene una fila en la tabla `UNIDADES_ACADEMICAS2` mediante línea 22. Notemos que ahora el nodo padre no será nulo, sino que será aquel que representa a la universidad. A continuación, en la línea 23 se obtiene el nombre de la función que calculará el indicador respectivo para el nivel facultad. Esta misma función será llamada en línea 24, y sus resultados se almacenarán en la tabla `INDICADORES2` en la línea 25. Una vez

guardados estos datos, mediante las líneas 27 y 28 se obtiene una lista de subunidades pertenecientes a esta facultad, las cuales podrán ser carreras o departamentos según corresponda al indicador.

Por último, desde la línea 29, se itera sobre la lista de subunidades correspondientes a la facultad. Aquí se obtiene la función que calcula indicador para el nivel subunidad, y posteriormente se almacenan sus resultados con la línea 32.

Como se puede apreciar, los cambios necesarios en este *script* para agregar un nuevo indicador serán mínimos, pues solo se necesita agregar el nombre del nuevo indicador a la lista de la línea 10. Por otro lado, agregar un nuevo indicador requiere definir 4 funciones: `getIndicadorFuente_indicador_Universidad`, `getIndicadorFuente_indicador_Facultad`, `getIndicadorFuente_indicador_Subunidad`, y `getSubunidades_indicador`. De estas funciones, las 3 primeras se encargarán de obtener los datos para los distintos niveles de unidades académicas, mientras que la última indicará qué subunidades pertenecen a una facultad, y por tanto sobre qué unidades académicas se debe calcular el indicador. Para explicar de manera general su funcionamiento, en el listado 5.6 se muestran estas 4 funciones para el indicador `TOTAL_INGRESOS`.

Listado 5.6: Funciones TOTAL_INGRESOS

```
1  function getIndicadorFuente_TOTAL_INGRESOS_Subunidad($base, $subunidad, $annio){
2      $ingresos = getIngresos($subunidad, $annio, $base);
3      $res = [
4          'F' => 0,
5          'M' => 0,
6          'S.I' => 0
7      ];
8      foreach( $ingresos as $tupla ){
9          if($tupla['PERS_SEXO']=='') $tupla['PERS_SEXO']='S.I';
10         $res[$tupla['PERS_SEXO']] += $tupla['CUENTA'];
11     }
12     return $res;
13 }
14 function getSubunidades_TOTAL_INGRESOS( $base ){
15   return subunidadesCarrIniciables($base);
16 }
17 function getIndicadorFuente_TOTAL_INGRESOS_Facultad($base, $annio){
18     $res = [
19         'F' => 0,
20         'M' => 0,
21         'S.I' => 0
22     ];
23     $subunidades = getSubunidades_TOTAL_INGRESOS($base);
24     foreach($subunidades as $subuni => $uacad){
25         $ingresos_subuni = getIndicadorFuente_TOTAL_INGRESOS_Subunidad($base, $uacad['cod'], $annio);
26         foreach( $ingresos_subuni as $sexo=>$cuenta){
27             $res[$sexo] += $cuenta;
28         }
29     }
30     return $res;
31 }
32 function getIndicadorFuente_TOTAL_INGRESOS_Universidad($universidad, $annio){
33     $bases = InfoNucleo::getParametro( SISTEMA, 'bases_ucampus' );
34     $res = [
35         'F' => 0,
36         'M' => 0,
37         'S.I' => 0
38     ];
39     foreach( $bases as $base ){
40         if( $base == 'ingenieria' ) $base = 'fcfm';
41         $ingresos_fac = getIndicadorFuente_TOTAL_INGRESOS_Facultad( $base, $annio );
42         foreach( $ingresos_fac as $sexo=>$cuenta){
```

```
43              $res[$sexo] += $cuenta;
44          }
45      }
46      return $res;
47  }
```

En primer lugar tenemos la función `getIndicadorFuente_TOTAL_INGRESOS_Subunidad`, la cual recibirá `$base`, que representa la facultad a la que pertenece la subunidad en cuestión, `$subunidad`, que contiene el código en las bases de datos de Ucampus de la carrera cuyo indicador se quiere calcular, y por último, `$annio`, que contiene el año para el cual se quiere calcular el indicador. En cuanto a su funcionamiento, parte llamando a la función `getIngresos` en la línea 2 , la cual se encarga de realizar la consulta a la base de datos correspondiente para luego, entre las líneas 3 y 11, reordenar estos datos de manera análoga a lo que se vio en el *script* de extracción de datos de la primera iteración para este mismo indicador.

Entre las líneas 14 y 16 se define la función `getSubunidades_TOTAL_INGRESOS`, la cual recibe una `$base` representando una facultad, y entrega las subunidades de esa facultad. Esta función llama a `subunidadesCarrIniciables`, la cual realiza la consulta respectiva a la base de datos de Ucampus y retorna un listado de las subunidades académicas solicitadas. Cabe notar que esta misma función auxiliar será utilizada por todos los indicadores que se calculen en su nivel inferior para carreras iniciables.

Luego, entre las líneas 17 y 37, encontramos la función para obtener los datos del indicador al nivel de facultad, `getIndicadorFuente_TOTAL_INGRESOS_Facultad`, la cual recibirá solamente `$base`, que al igual que en la función anterior, representa la facultad cuyos datos se quieren obtener, y un año para el cuál se quiere calcular el indicador en cuestión. Esta función obtiene las subunidades correspondientes en la línea 23,mediante la función ya explicada, y posteriormente itera sobre estas, llamando a la función anteriormente presentada, `getIndicadorFuente_TOTAL_INGRESOS_Subunidad`, para luego sumar los resultados de esta según sexo, generando así la agrupación buscada.

Por último tenemos la función `getIndicadorFuente_TOTAL_INGRESOS_Universidad`, la cual recibe `$universidad` representando el nodo respectivo en la base de datos, y un año para el cual se quiere calcular el indicador. De manera análoga a la función anterior, esta utiliza `getIndicadorFuente_TOTAL_INGRESOS_Facultad` para obtener los datos de las facultades asociadas, para posteriormente iterar sobre ellas y sumar los resultados por sexo.

Cabe mencionar que todas las funciones análogas a las aquí mostradas para los demás indicadores deben compartir las mismas firmas que las aquí señaladas

5.3.3. Consulta y Presentación de datos

Al igual que en la sección 5.1.3, revisaremos vista por vista como se implementó el módulo, prestando especial atención a los *scripts* PHP y como estos se vieron simplificados en esta nueva iteración.

Vista principal

Al igual que antes, la vista principal debiese contener las secciones de indicadores, mostrando el título de estas y un indicador representativo, acompañados de enlaces que redirigen a las respectivas secciones. A diferencia de la primera iteración, los selectores que filtran la información del panel se han visto reducidos, pues ahora solo se cuenta con un selector de año, y uno de unidad académica, en contraste la versión anterior, donde deberían haber 3, año, facultad y carrera. Notemos que dada la disparidad en los indicadores representativos en esta vista, no es posibles agruparlos a todos por el mismo tipo de subunidad menor (carrera o departamento) y por lo tanto, se ha decidido que esta vista agrupara solamente por facultad o universidad, limitando así el selector solo a estas unidades académicas. Cabe también mencionar que, a diferencia de la primera iteración, ahora para agrupar por universidad, esta unidad debe ser elegida en el selector correspondiente, en vez de dejar este último en blanco.

Idénticamente a como fue en la iteración anterior, el funcionamiento de esta vista requiere de un *script* PHP en la carpeta web del módulo, y un *template* html en la carpeta *templates*. A continuación, en el listado 5.7, se presenta el código del *script* PHP respectivo para posteriormente ser comentado.

Listado 5.7: Script vista principal segunda iteración

```
1  $anno = (int)$_REQUEST['anno'];
2  $unidad = $_REQUEST['unidad'];
3  if( ! $anno ) $anno = date( 'Y' );
4  if( ! $unidad ) $unidad = 'universidad.uchile';
5  $annos = selector_annos();
6  $arbol = getArbol( 'TOTAL_INGRESOS' );
7  $lista = arbolToListFacultades( $arbol );
8
9  $ingresos = getIndicador( $unidad, $anno, 'TOTAL_INGRESOS' );
10 $retencion = getIndicador( $unidad, $anno, 'RETENCION_ACTUAL_1' );
11 $valor_publicaciones = getIndicador( $unidad, $anno, 'PUBLICACIONES_INVESTIGACION' );
12
13 KERNEL::head();
14 require( template( '../template/index.html' ) );
15 KERNEL::foot();
```

Notemos que en primer lugar se reciben el año y la unidad académica que el usuario ha solicitado ver, y en caso de no haberlo hecho, se asignan valores por defecto. Posteriormente se obtienen los años a mostrar en el selector mediante una función definida y estandarizada para ser usada en todas las vistas donde sea necesario. Luego se obtiene el árbol de unidades académicas mencionado en las sección 5.2 y 5.3.1. Dicho árbol consiste en un conjunto de listas anidadas, las cuales no pueden ser utilizadas directamente como entrada para el selector respectivo. Es por esto que se utiliza la función `arbolToListFacultades`, la cual convierte este conjunto de listas en una sola, donde la profundidad en el árbol se traduce en espacios en blanco antes del nombre de la unidad para mostrar la estructura jerárquica en el selector. Notemos que esta función responde directamente a lo señalado anteriormente sobre el selector de esta vista mostrando solamente la universidad y sus facultades, y no sus carreras o departamentos.

Una vez listos los selectores, se procede a obtener, en tan solo 3 líneas de código, los datos listos para ser presentados al usuario mediante el *template* correspondiente a la vista. La vista en cuestión se presenta en la figura 5.8. Podemos notar que ahora esta vista contiene

casi todo lo que se buscaba. Presenta dos selectores, para unidad académica y año, seguido de tarjetas que representan cada sección y contienen un indicador representativo en 3 de 4 casos, faltando solo la sección docentes.

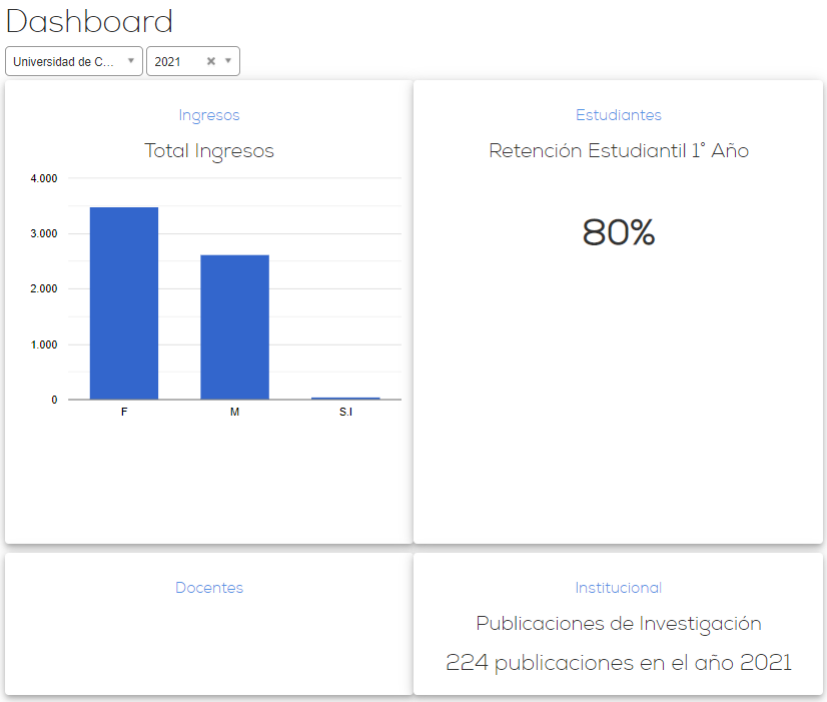


Figura 5.8: Vista principal del módulo, nueva iteración

Al igual que antes, los enlaces presentes en los títulos de las secciones redirigen a vistas dedicadas a dichas secciones.

Vista secciones

En primer lugar, debemos notar que, al igual que en la iteración anterior, cada vista de sección, es decir, Ingresos, Estudiantes, Docentes e Institucional, requieren su propio par de *script* PHP en la carpeta *web*, y *template* html en la carpeta *templates*.

En esta segunda iteración, cada vista dedicada a una sección debería contener una serie de indicadores relativos a ella, junto a los selectores ya comentados en la vista principal, permitiendo filtrar la información de todo el panel según las necesidades del usuario. Al igual que antes, los títulos de los indicadores debiesen contener enlaces que dirijan al usuario a la vista histórica de cada indicador.

En cuanto a los *scripts* requeridos para el funcionamiento de estas vistas, estos serán muy similares al presentado en la vista principal de esta segunda iteración, pues contienen los mismos pasos a grandes rasgos: obtención de entrada del usuario, obtención información selectores, y consultas a la base de datos para cada indicador perteneciente a la sección.

Dada la mayor cantidad de indicadores implementados para la sección Ingresos, esta será

utilizada para ejemplificar el funcionamiento de estas vistas de sección. En el listado 5.8 se presenta el *script* de la sección mencionada.

Listado 5.8: Vista sección Ingresos iteración 2

```
$anno = (int)$_REQUEST['anno'];
$unidad = $_REQUEST['unidad'];
if( ! $anno ) $anno = date( 'Y' );
if( ! $unidad ) $unidad = 'universidad.uchile';
$annos = selector_annos();
$arbol = getArbol( 'PRUEBA_PROMEDIO' );
$lista = arbolToList( $arbol );

$resultados_ingresos = getIndicador( $unidad, $anno, 'TOTAL_INGRESOS' );
$ingresos_regiones = getIndicador( $unidad, $anno, 'INGRESOS_REGIONES');
$nem_promedio = getIndicador( $unidad, $anno, 'NEM_PROMEDIO' );
$prueba_promedio = getIndicador( $unidad, $anno, 'PRUEBA_PROMEDIO' );
$tipo_ingreso = getIndicador( $unidad, $anno, 'TIPO_INGRESO' );
$tipo_establecimientos = getIndicador( $unidad, $anno, 'TIPO_ESTABLECIMIENTO' );

KERNEL::head( [ '.' => 'Ingresos' ] );
require( template( '../template/ingresos.html' ) );
KERNEL::foot();
```

Notemos que este *script* es casi idéntico al de la vista principal, con dos principales diferencias: la lista de indicadores es distinta, y la función para obtener la lista para el selector es distinta. Esta última diferencia se debe a que, dado que dentro de las secciones, todos los indicadores comparten las subunidades de menor nivel (carreras o departamentos), aquí no es necesario limitar el árbol, y por lo tanto se mostrarán en el selector todas las opciones disponibles para los indicadores. En este caso en particular, la sección de Ingresos utiliza carreras, y por lo tanto estas opciones serán mostradas en su selector de unidades.

Una vez se tienen los datos gracias al *script*, estos pueden ser presentados al usuario gracias al *template* respectivo. El resultado de esto, para la vista de sección Ingresos se presenta en 5.9, donde podemos apreciar la variedad de indicadores implementados en esta iteración para esta sección. Además en la figura 5.10 se presenta el selector de unidades académicas de esta sección, donde se puede apreciar el uso ya mencionado de espacios en blanco para señalar la jerarquía de las unidades académicas.

Figura 5.9: Vista principal del módulo, nueva iteración

Figura 5.10: Selector de unidades académicas en la sección ingresos

Como se puede ver en la figura 5.9, cada título de esta sección contiene un enlace, los cuales llevarán al usuario a la vista histórica del indicador en cuestión.

Vista Histórica de un indicador

Al igual que en la primera iteración, al acceder a la vista histórica de un indicador, se debe mostrar el selector de unidades académicas, ahora unificado, seguido de un gráfico de líneas que muestre la evolución del indicador en el tiempo. La principal diferencia ahora, se encuentra en el *script* y el *template* necesarios para esta vista. Anteriormente, se determinó que cada indicador requería su propio *script* y *template* para obtener y presentar los datos históricos. En contraste, en esta nueva iteración, se definió que todos los indicadores de una sección podrían utilizar un único par de *script* y *template*. En esta segunda iteración sólo se desarrollaron estos códigos para la sección Ingresos, y por lo tanto esta será usada para explicar el proceso. El *script* referido se presenta en el listado 4.4 y es explicado posteriormente.

Listado 5.9: Script vista histórica sección Ingresos

```
$indicador = $_REQUEST['ind'];
$unidad = $_REQUEST['unidad'];

$arbol = getArbol( $indicador );
$lista = arbolToList( $arbol );

$data = getIndicadorHistorico( $unidad, $indicador );
if( is_array( reset( $data ) ) ) {
  $res = [];
  foreach( $data as $anno => $array ) {
    foreach( $array as $key => $value ) {
      if( is_array( $value ) ) {
        foreach( $value as $key2 => $value2 ) {
          $res[$key2][$anno] += $value2;
        }
        continue;
      }
      $res[$key][$anno] = $value;
    }
  }
```

```
21    $data = $res;
22  }
23
24  KERNEL::head( [ './ingresos.php'=>'Ingresos', $indicador ] );
25  require( template( '../template/historico_ingresos.html' ) );
26  KERNEL::foot();
```

Podemos ver que las primeras cuatro líneas son similares a las presentes en las vistas ya revisadas, con la principal diferencia que ahora se recibe el nombre de un indicador como parámetro, en vez de un año. Luego se obtienen los datos del indicador, de manera similar a como se hizo anteriormente, con la diferencia que ahora se llama a la función `getIndicadorHistorico`, que recibe una unidad académica y un indicador, y retorna un arreglo de años con valores asociados. Pero dada la heterogeneidad en los indicadores, estos valores asociados no siempre tienen la misma forma. Por ejemplo, para el indicador `TOTAL_INGRESOS`, para cada año hay una lista de 3 opciones con valores asociados, mientras que para el indicador `NEM_PROMEDIO` cada año tiene un único valor asociado. Para lidiar con esto, se tiene el código entre las líneas 8 y 21, el cual formateará los datos de distinta manera dependiendo de si los valores para cada año son: valores numéricos únicos, listas de valores numéricos o listas de listas, permitiendo así que estos datos sean correctamente representados por el *template*.

Con los datos ya preparados, se genera la visualización mediante el *template*, la cual se presenta en la figura 5.11. En dicha figura se puede notar que la principal diferencia visual con la primera iteración consiste en la ya mencionada unificación de selectores.

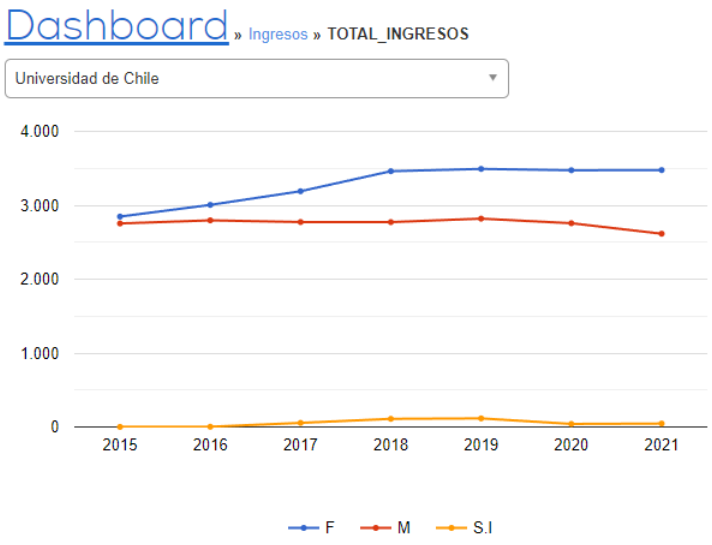


Figura 5.11: Vista histórica sección ingresos

Otras secciones

Dada la similitud y sencillez en los códigos, producto del rediseño, se han omitido las secciones restantes del módulo, pero en las figuras 5.12 y 5.13 se muestran las vistas de las

secciones Estudiantes e Institucional.

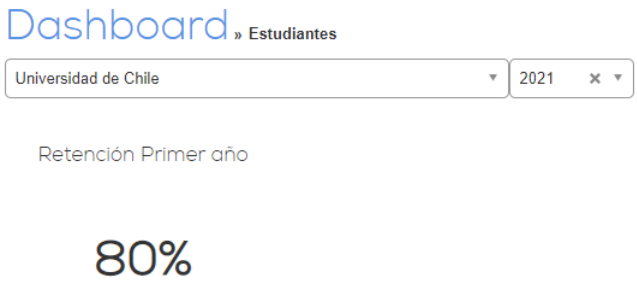


Figura 5.12: Vista sección Estudiantes

Figura 5.13: Vista sección Institucional

5.4. Resumen

En esta sección se explicó la implementación del diseño presentado anteriormente en la sección 4.2, seguido de los problemas que en ella surgieron y cómo estos influenciaron un rediseño, para finalmente explicar la implementación de este. Es también apropiado incluir una lista de todos los indicadores que fueron efectivamente implementados en esta etapa final del desarrollo.

- `TOTAL_INGRESOS`
- `INGRESOS_REGIONES`
- `NEM_PROMEDIO`
- `PRUEBA_PROMEDIO`

- TIPO_ESTABLECIMIENTO
- TIPO_INGRESO
- PUBLICACIONES_INVESTIGACION
- RETENCION_ACTUAL_1

Al mismo tiempo, es importante señalar aquellos indicadores cuya información se determina que sí esta presente en Ucampus, pero que no fueron implementados en el transcurso de esta memoria. Estos indicadores se muestran a continuación:

- TOTAL_ESTUDIANTES
- RETENCION_PROMEDIO_1
- TITULACION_OPORTUNA_ACTUAL
- TITULACION_OPORTUNA_PROMEDIO
- MOV_EST_INMIGRANTES
- DOC_JORNADA_COMPLETA
- DOC_DOCTOR
- PROYECTOS_INVESTIGACION
- ESTUDIANTES_POR_PROYECTO

Se debe recalcar que es posible que la información de algunos otros indicadores se encuentre disponible, pero no haya sido encontrada. De manera similar, es posible que la información necesaria para estos indicadores no sea en realidad suficiente como se plantea, lo que es difícil de detectar sin implementarlos.

De esta manera, de un total de 17 indicadores que se determinan como factibles por la existencia de sus datos, se implementa un total de 8.

Por otro lado, es útil comparar la herramienta implementada con las alternativas analizadas anteriormente, en el capítulo Estado del Arte. En este sentido, el resultado de este trabajo de memoria presenta capacidades similares a las descritas por las alternativas, con una diferencia significativa con respecto a las plataformas Banner, Ellucian Analytics, y PeopleSoft, pues aquí no se permite a cada institución definir directamente indicadores propios.

Finalmente, aunque el conjunto de indicadores presentes en el módulo es reducido en comparación a lo propuesto en el boceto preexistente [8], se considera que el proyecto presenta una base considerable y que puede seguir siendo desarrollada por el centro Ucampus.Si bien, el resultado de esta implementación no está listo para salir a una etapa de producción, sí se ha logrado producir un módulo integrado en Ucampus y extensible en el tiempo, que concreta la planteado sólo como un boceto en el trabajo previo [8].

Capítulo 6

Validación

Una vez terminada la implementación se procede a realizar una serie de validaciones sobre el resultado obtenido, las cuales serán explicadas en el siguiente capítulo. Partiendo por la sección 6.1, se analiza el costo de agregar un nuevo indicador, comparando entre las iteraciones explicadas, mostrando así qué tan extensible es el proyecto. A continuación, en la sección 6.2, se explica cómo se comportaría la herramienta desarrollada ante distintos cambios. Por ejemplo, en otras instalaciones de la plataforma Ucampus o ante una mayor carga de indicadores.

Por último, en la sección 6.3 se comenta sobre el user testing, la carencia de éste siendo originalmente planteado como un objetivo específico y cómo se subsana a partir de validación con desarrolladores del centro Ucampus.

6.1. Extensibilidad

6.1.1. Agregar nuevos indicadores

Para agregar nuevos indicadores es necesario modificar 3 elementos: extracción de datos, la obtención de éstos en la sección correspondiente y la presentación gráfica en el *template* respectivo.

Como ya se ha comentado, en la primera iteración, las dos primeras de estas etapas no se encuentran particularmente estructuradas. En la extracción de datos se debe definir un *script* que obtenga los datos necesarios desde la respectiva fuente y luego los guarde en la tabla `INDICADORES_CARRERA`, definiendo en este mismo *script* todo lo necesario para las conexiones a las bases de datos y para el guardado de la información. De similar manera, en cuanto a la posterior recuperación de esta información para mostrarla al usuario, es necesario agregar al *script* de la sección correspondiente todo el código necesario, desde las consultas SQL, hasta la manipulación de los datos ajustarlos a la forma necesaria para poder presentarlos.

Al contrario, en la segunda iteración, estas mismas dos etapas se encuentran claramente

estructuradas, y son significativamente más simples. En cuanto a la extracción de datos, se deben definir 4 funciones, cuyo objetivo y firma están claramente establecidos, y el único cambio al *script* que se encarga de traer los datos consiste en agregar el nuevo indicador a la lista de los indicadores a extraer. En cuánto a la posterior consulta de estos datos para mostrarlos al usuario, basta con agregar una única línea, llamando a una función ya definida, con el nombre del nuevo indicador para obtener la información desde la base de datos.

Dado esto, a continuación se comparan en concreto las modificaciones necesarias para agregar un nuevo indicador. Para esta comparación se utiliza el indicador `TOTAL_INGRESOS`. Se compara cuántos archivos y líneas se deben agregar o modificar para dejar el indicador funcionando.

Iteración	Archivos a modificar/agregar	Líneas a modificar/agregar
Primera	4	108
Última	4	81

Tabla 6.1: Comparativa indicador `TOTAL_INGRESOS`

En la primera iteración, se agrega el archivo script_ingresos.py, con 74 líneas de código, y se modifica el archivo ingresos.php, agregando 34 líneas de código, para un total de 2 archivos y 108 líneas. En cambio para la siguiente iteración, se definen las 4 funciones necesarias y una función auxiliar en un único archivo preexistente, agregando un total de 79 líneas, seguido de una linea en el *script* de la sección correspondiente para consultar la base de datos, y por último una línea más en el *template* para mostrar los datos al usuario, para un total de 81 líneas.

Si bien la diferencia entre ambas iteraciones en cuanto a líneas de código puede no parecer particularmente importante (aproximadamente un 20 %), la verdadera importancia recae en la presencia de una estructura clara a seguir en la segunda iteración, ausente en un inicio, que facilitará enormemente el trabajo a la hora de agregar nuevos indicadores en un futuro. Por ejemplo, `TOTAL_INGRESOS` fue un indicador que tomó semanas en ser creado considerando que fue uno de los primeros indicadores y no se tenía completa claridad de la estructura final. Aun así, `TIPO_ESTABLECIMIENTO` y `TIPO_INGRESO` tomaron cerca de 30 minutos en ser agregados comprendiendo y teniendo bien definido el objetivo de cada una de las funciones que debían ser implementadas.

6.1.2. Distintos indicadores en distintas instituciones

Por otro lado en cuanto a extensibilidad, se puede cuestionar la necesidad de utilizar distintos conjuntos de indicadores para distintas instituciones, esto dada la heterogeneidad de las organizaciones que hacen uso de las plataformas de Ucampus, desde la Universidad de Chile hasta la Escuela de Gendarmería. Con la implementación actual del proyecto, no es posible reutilizar el código perteneciente a las vistas, pero con un conjunto distinto de indicadores. A pesar de esto, sería posible implementar cambios menores a lo largo del proceso, que permitiesen el uso de archivos de configuración propios de cada institución, definiendo

en ellos las listas de indicadores de interés para cada instalación. Esta posibilidad se explica con mayor detalle más adelante en el presente capítulo.

6.2. Escalabilidad

Respecto a la escalabilidad del proyecto, se puede mirar desde dos perspectivas: Desde la cantidad de datos presentes en la instalación de Ucampus o desde el volumen de indicadores presentes en el módulo.

6.2.1. Respecto al volumen de datos

Para analizar la primera de estas 2 perspectivas, es decir, cómo escala el módulo desarrollado con respecto al volumen de datos en la instalación, se realizaron pruebas donde se generó la carga de datos y se midió el tiempo necesario para dicho proceso. Para contextualizar estas pruebas y el análisis posterior, es necesario primero tomar en consideración los tamaños de las instalaciones Ucampus. En particular, se utilizarán como referencia las bases de datos `MUFASA` y `UCURRICULUM`, dado que son utilizadas como fuentes de datos para los indicadores.

Como se mencionó en 3.1, cada instalación de la plataforma Ucampus cuenta con su propia instancia de `MUFASA`, con la excepción de la Universidad de Chile, que tiene un total de 23 instancias, una para cada facultad, más algunas otras entidades, como el campus Juan Gomez Millas. Entre estas instancias, la más pesada es la perteneciente a la Facultad de Ciencias Físicas y Matemáticas (FCFM), la cual tiene un peso total de 1.9 GB. El resto de las instancias tienen un peso promedio de aproximadamente 200 MB. Por otro lado, la instalación más pesada no perteneciente a la Universidad de Chile pertenece a la Universidad Metropolitana de Ciencias de la Educación (UMCE), con un peso de aproximadamente 470 MB. En la tabla 6.2 se muestran algunos datos sobre distintas instancias de `MUFASA`, incluyendo pesos totales.

Instancia de `MUFASA`	Alumnos (Registro total histórico)	Operaciones de inscripción académica	Peso total (GB)
`FCFM`	70.313	2.872.654	1,90
`MEDICINA_UCHILE`	36.229	1.324.644	0,61
`UMCE`	59.436	162.057	0,47
`UOH`	7.991	362.337	0,16

Tabla 6.2: Instancias de `MUFASA`

Por otro lado, tenemos la ya mencionada base de datos `UCURRICULUM`. Notemos que estas bases de datos sí son únicas por cada institución, es decir, a diferencia de `MUFASA`, todas las facultades de la Universidad de Chile comparten una única instancia, llamada `FCFM_UCURRICULUM`. El nombre de esta instancia tiene motivos históricos y no refleja apropiadamente su contenido. Para las pruebas de carga a realizar, se consideraron 3 instancias

de esta base de datos, FCFM_UCURRICULUM, UOH_UCURRICULUM y UMCE_UCURRICULUM. A continuación, en la tabla 6.3, se presenta un resumen de los tamaños de estas bases de datos, incluyendo cantidad de registros en tablas relevantes, y peso total.

Instancia de UCURRICULUM	Registros tabla QUIENES	Registros tabla RELACIONES	Peso Total (MB)
FCFM_UCURRICULUM	806.649	583.805	558
UOH_UCURRICULUM	16.096	37.242	19
UMCE_UCURRICULUM	4.185	4.062	9

Tabla 6.3: Instancias de UCURRICULUM

Con esta información en consideración, se realizaron pruebas de carga de indicadores para las 3 instituciones cuyo contexto se ha presentado, es decir, la Universidad de Chile, UMCE, y Universidad de O'Higgins (UOH). Dichas pruebas consisten en medir el tiempo que toma cargar el conjunto de indicadores implementado, para cada instalación. En particular, se mide el tiempo de carga de cada indicador específico, permitiendo así observar como varía el comportamiento entre las instituciones por indicador. Los resultados se presentan a continuación en la tabla 6.4.

	Tiempo de carga por cada institución (s)		
Indicador	UChile	UMCE	UOH
TOTAL_INGRESOS	6,2	0,6	0,3
INGRESOS_REGIONES	10,7	0,2	0,2
NEM_PROMEDIO	7,7	0,8	0,3
PUBLICACIONES_INVESTIGACION	130,6	0,1	0,2
TIPO_ESTABLECIMIENTO	4,3	0,4	0,2
PRUEBA_PROMEDIO	7,0	0,8	0,3
TIPO_INGRESO	4,7	0,4	0,2
RETENCION_ACTUAL_1	6,4	0,7	0,3

Tabla 6.4: Resultados pruebas de carga por indicador

En estos resultados vemos que los tiempos de carga son significativamente más altos para la Universidad de Chile, en todos los indicadores. Esto tiene directa relación con la información presentada en las tablas 6.2 y 6.3, donde se señala que esta institución tiene un volumen de datos mucho mayor. Además, es relevante considerar lo planteado en la sección 3.1.2, donde se señala la existencia de múltiples instancias de MUFASA dentro de la institución, una para cada facultad, a diferencia de las demás organizaciones.

Al mismo tiempo, es interesante notar que el indicador PUBLICACIONES_INVESTIGACION es, por un amplio margen, el más lento de calcular para la Universidad de Chile, lo que no se repite en las otras instituciones. Esto tiene directa relación con los resultados presentados en la tabla 6.3, donde se muestra que la instancia perteneciente a la Universidad de Chile es uno y dos ordenes de magnitud más voluminosa que sus contrapartes pertenecientes a la

UOH y UMCE respectivamente. Esta situación a su vez está estrechamente relacionada con la productividad científica de las distintas instituciones. En este sentido, es valioso tener en cuenta que, según la Agencia Nacional de Investigación y Desarrollo (ANID), desde 2008, la Universidad de Chile ha estado involucrada en un 18 % de los proyectos de investigación realizados en el país. En comparación, en el mismo período de tiempo, las otras instituciones presentadas en estas pruebas, UMCE y UOH, participan en aproximadamente un 0.3 % y 0.1 % respectivamente. Dado esto, podemos considerar que el tiempo que demora el indicador `PUBLICACIONES_INVESTIGACION` para la Universidad de Chile constituye el peor caso para las instituciones que podrían usar la herramienta.

Tomando en consideración la información hasta aquí presentada, podemos concluir que las pruebas, realizadas en la instalación de la Universidad de Chile, presentan el peor caso, dado los volúmenes de datos asociados a ella, y por lo tanto, cualquier otra institución que haga uso de la herramienta tendrá resultados aún mejores. Esto, en conjunto con los relativamente bajos tiempos de carga para la Universidad de Chile, y una periodicidad semanal para dicho proceso permiten concluir que la herramienta implementada escalará apropiadamente con el volumen de datos en el tiempo.

6.2.2. Respecto al volumen de indicadores

En cuanto a la segunda perspectiva mencionada, sobre el volumen de indicadores, debemos considerar las pruebas de carga ya planteadas, en particular, los resultados de estas pruebas sobre la instalación perteneciente a la Universidad de Chile, que ya fueron presentados en la tabla 6.4, con un tiempo total de aproximadamente 3 minutos, y diferencias muy marcadas entre los tiempos de cada indicador. Como se ha comentado anteriormente, la implementación actual del proyecto cuenta con un total de 8 indicadores, mientras que el trabajo de memoria previo propone un total de 31. Lamentablemente, dada la heterogeneidad de los indicadores, no es posible estimar cuánto se demorarían los indicadores no implementados en sus procesos de carga, pudiendo variar entre un par de segundos y un par de minutos, como los indicadores ya implementados. Afortunadamente, como ya se ha comentado, el proceso de carga tiene una periodicidad semanal, y por lo tanto, a menos que los indicadores no implementados generen un aumento explosivo en el de carga, no generarán problemas en el correcto funcionamiento de la herramienta implementada.

6.3. Validaciones mediante user testing

Como se señala la sección 1.4.2, un objetivo específico de este trabajo de memoria consistía en validar el resultado final con usuarios finales mediantes user testing. Dichos usuarios finales, como se señaló en la sección 4.1.1 corresponden en general a administrativos de las universidades que utilizan Ucampus para tomar decisiones estratégicas, de distintos niveles, tales como: rector, decanos, directores de área, jefes de departamento, entre otros. Por lo tanto, debido a la falta de implementación de todos los indicadores necesarios, y al complejo manejo de tiempos que pueden poseer estas autoridades se decidió no realizar validación junto a ellos.

Aun así, si bien no reemplaza la necesidad de testeo y validación, se realizó una reunión junto con el equipo del centro Ucampus para obtener feedback de lo realizado.

Como se comentó, el módulo desarrollado fue presentado ante miembros del equipo del centro Ucampus en una reunión grabada y realizada el día jueves 16 de noviembre de 2021. En dicha reunión se encontraron presentes Willy Maikowski, profesor guía y desarrollador del centro, Tomás Ahumada y Felipe Quintanilla, desarrolladores, y José Miguel Garrido, administrador de sistemas. Además, Javier Villanueva y Manuel Ortega, director y subdirector del centro Ucampus, observaron posteriormente el vídeo de la reunión para complementar la retroalimentación.

En esta reunión, se parte explicando el contexto y la motivación que llevan al desarrollo de esta memoria, en particular, a la necesidad y utilidad de una herramienta de apoyo a la toma de decisiones estratégicas. Posteriormente se muestra la vista principal del módulo desarrollado y se mencionan las secciones allí presentes. Luego se muestran los selectores que permiten filtrar los datos a mostrar, ya sea por año o por unidad académica. A continuación se muestra que se puede acceder a la sección Ingresos mediante el enlace en el título de dicha sección. Dentro de dicha sección se señalan las diferencias en los selectores, en particular la presencia de carreras, las cuales se encuentran ausentes en el selector de la vista principal. Posteriormente se muestra la posibilidad de acceder a la vista histórica de cada indicador, a la cual se llega mediante los enlaces en los nombres de los respectivos indicadores.

Una vez se terminaron de presentar las vistas disponibles en el módulo, se procede a explicar cómo este funciona, partiendo por las bases de datos. Se parte mencionando la existencia de dos tablas, `UNIDADES_ACADEMICAS2` y `INDICADORES2`. En particular, se señala que la primera de estas contiene las unidades académicas y permite ordenarlas como un árbol, donde las carreras y departamentos representan las hojas, las facultades serían nodos intermedios y la universidad sería la raíz. En cuanto a la segunda tabla mencionada, se señala que cada fila representa la medición de un indicador específico, para una única unidad académica en un año en particular.

Por último, se muestra la estructura del código, donde cada vista cuenta con un *script* que obtiene los datos desde la base de datos, y un *template* que utiliza dichos datos para producir los gráficos que serán finalmente presentados al usuario. Con esto concluye la presentación y se procede a recibir los comentarios de los oyentes, los cuales son explicados y resumidos a continuación.

En primer lugar, se señala la presencia de la información de toda la universidad para todos los usuarios con acceso al módulo, esto se problematiza si se considera, por ejemplo, que personal del departamento de ciencias de la computación de la facultad de ingeniería de una universidad probablemente no tenga ningún interés en visualizar información relativa a la facultad de derecho de la misma universidad. En general, se podría considerar que en algunos casos, la presencia de ciertas unidades académicas puede entorpecer la navegación de los usuarios. A grandes rasgos, debería ser posible utilizar la información disponible sobre los usuarios con respecto a cargo y departamento o facultad donde trabajan, para filtrar la información presentada en el módulo.

Otro comentario importante tiene relación con la navegación dentro del módulo, en parti-

cular, parece no ser claro el sentido de los enlaces presentes en los títulos, pues no se entiende que redirigen a secciones en el caso de la vista principal, o a vistas históricas en el caso de las vistas por sección. Una posible solución en este caso sería agregar pequeños iconos o textos que expliciten el destino de los enlaces presentados, aunque podrían haber otras maneras de afrontar esta problemática.

Además se señala la posibilidad de distintas instituciones requieran distintos conjuntos de indicadores, y se cuestiona cómo esto podría ser implementado. Ante esto, se llega a una idea de solución a grandes rasgos. En general, sería necesario implementar una configuración que permita asociar una lista de indicadores a cada institución, para posteriormente utilizar esta lista en lugar de directamente los indicadores en el código. Por ejemplo, esta lista sería utilizada en el *script* de obtención de indicadores, mostrado en 5.3.2, en lugar de la lista fija que funciona actualmente. También sería necesario modificar los *scripts* de las secciones mostrados en 5.3.3 para que, en vez de llamar a la función `getIndicador` sobre ciertos indicadores específicos mencionados, lo hicieran sobre los elementos de la lista respectiva. Con estos cambios, relativamente menores, se conseguiría responder a la necesidad planteada.

Finalmente, aunque se obtuvieron diversos comentarios, el módulo en general fue evaluado positivamente, considerando una clara entrega de la información, una buena arquitectura y diseño de la solución, y un tiempo de respuesta destacablemente rápido al cambiar vistas o selectores.

Capítulo 7

Conclusión

Como se ha comentado a lo largo de la memoria, este proyecto empieza desde una memoria anterior que entrega un bosquejo de una nueva funcionalidad para la plataforma Ucampus. La presente memoria, como se declara en la sección 1.4.1, tiene como objetivo generar el análisis, diseño, desarrollo y validación de esta funcionalidad, la cual es entregar a los usuarios una fuente centralizada de información desde la cual tomar decisiones estratégicas.

Siguiendo la arquitectura modular de Ucampus, la herramienta desarrollada se constituye como un módulo. Dicho módulo consiste en una especie de panel, dividido en secciones, cada una de las cuales cuenta con una serie de indicadores, los cuales presentan información sobre una serie de aspectos de las instituciones. Este módulo está construido utilizando el lenguaje PHP, y consulta una base de datos SQL la cual contiene los datos de los indicadores pre calculados y almacenados en ella mediante el uso de *scripts* que corren de manera periódica, los cuales también fueron implementados en PHP.

En general, los objetivos planteados fueron alcanzados. Se diseñó e implementó una herramienta que hiciese uso de los datos presentes en la plataforma Ucampus, con el fin de presentar a los usuarios relevantes información sobre el estado de sus instituciones.

Si bien se cumplieron la mayoría de objetivos específicos señalados en la sección 1.4.2, en particular no se cumplió el objetivo 5 que busca validar el trabajo con usuarios finales. Aun así, se concluye bajo criterios del centro Ucampus que el trabajo se finalizó de manera exitosa, cubriendo satisfactoriamente la mayoría de los componentes técnicos y cumpliendo en gran medida el objetivo general.

7.1. Conclusiones Técnicas

Desde un punto de vista técnico, resulta destacable la importancia de dos decisiones que ayudaron al exitoso desarrollo del proyecto. Primero, la selección de indicadores a implementar y su orden, y segundo, la perspectiva de priorizar la profundidad del desarrollo, antes que su alcance.

En cuanto a la selección de indicadores, en un principio se partió implementando el indicador `TOTAL_INGRESOS`, para luego agregar el indicador `PUBLICACIONES_INVESTIGACION`, donde este último fue elegido específicamente por proceder de una fuente de datos distinta a la utilizada anteriormente. Este cambio permitió notar tempranamente las falencias en el modelo de datos y su posterior mejora. En particular, gracias a intentar implementar este indicador, se encontraron 2 problemas, señalados en la sección 5.2, por un lado, no todos los indicadores son calculables para unidades académicas de niveles superiores mediante la suma de las subunidades académicas respectivas, y segundo, no todos los indicadores son calculables para unidades académicas del tipo Carrera, en particular, algunos deberán ser calculados para unidades académicas del tipo Departamento . En caso de haber continuado con otros indicadores de la sección ingresos, estos problemas no habrían sido notados a tiempo, lo que habría resultado en que los cambios necesarios se hubiesen realizado más tarde y requerido repetir más trabajo.

De similar manera, fue importante la decisión de priorizar la profundidad, esto se refiere en particular al desarrollo de los distintos indicadores, donde se prefirió implementar completamente cada parte del indicador antes de continuar con otros, en vez, de por ejemplo, implementar los *scripts* de extracción de datos para varios indicadores al mismo tiempo. Al igual que lo señalado anteriormente, esto también contribuyó a detectar los problemas anteriormente comentados y no volver a repetirlos en los indicadores posteriores.

7.2. Conclusiones Personales y Metodológicas

En cuanto a conclusiones personales a partir del trabajo desarrollado, parece importante la apropiada documentación del proceso, en particular, a lo largo del desarrollo del trabajo de memoria, en algunas partes del proceso se llevó una detallada bitácora del trabajo, mientras que en otras, la documentación fue nula o deficiente. A la hora de escribir la presente memoria, la diferencia entre qué partes fueron apropiadamente documentadas y cuales no se hizo evidente dada la mayor dificultad para plasmar en este documento el proceso y las decisiones tomadas.

7.3. Trabajo Futuro

Por otro lado resulta importante señalar las falencias del trabajo realizado, en particular, la más grande consiste en la ausencia de user testing. Esta validación es muy importante, pues hasta el momento toda apreciación sobre las interfaces y la experiencia de usuario proviene de desarrolladores del centro Ucampus, y por tanto podría ser muy distinta a la percepción de un usuario final. Por lo tanto, esta falta impide llevar el proyecto desarrollado a una fase de producción. Es por esto que se señala la realización de pruebas de usuario como primera prioridad para continuar el desarrollo de este proyecto a futuro.

Para finalizar, resulta interesante comentar cómo podría continuar este trabajo en un futuro. En primer lugar, como se ha señalado repetidas veces, deberían desarrollarse pruebas de

usuario con usuarios reales, para obtener una apreciación sobre las interfaces e interacciones desarrolladas y si estas son fácilmente entendibles por quienes deberán finalmente hacer uso de la herramienta. Es esperable que luego de llevar a cabo estas pruebas se obtenga valioso feedback, cuya implementación también debería ser considerada de alta prioridad.

A continuación se deberían seguir implementando indicadores del conjunto original planteado en [8]. Esto es de gran importancia, pues en el estado actual del proyecto, este presenta una imagen muy limitada del estado de las instituciones, dado la escasa cantidad de indicadores en las secciones Institucional y Estudiantes, y nula en el caso de la sección Docentes, lo cuál a su vez limita el potencial como fuente de información para la toma de decisiones estratégicas de la herramienta desarrollada. Sería recomendable priorizar estas últimas secciones, por sobre la sección Ingresos, que ya presenta una variedad de información. De similar manera, se debería priorizar fuentes de datos aún no utilizadas, es decir, distintas a los esquemas `MUFASA` y `UCURRICULUM`, para continuar verificando que el modelo de datos implementado sea suficiente en cualquier caso.

Por último, se podría llevar el módulo a una instalación distinta de la plataforma, idealmente con un conjunto distinto de indicadores, que permitan probar lo planteado en la sección 6.3 sobre cómo implementar listas distintas de indicadores para diferentes instituciones. Como se señaló anteriormente, debería ser posible generar una abstracción sobre los indicadores que permita trabajar sobre una lista almacenada para cada institución que haga uso del módulo y así evitar tener los indicadores hardcodeados en los distintos *scripts*, ya sea de extracción o de las distintas secciones.

Se considera que extender el trabajo ya realizado de estas 3 maneras agregaría amplio valor al proyecto y este al mismo tiempo agrega valor también al producto del centro Ucampus.

Bibliografía

[1] Frada Burstein and Clyde W. Holsapple. *Handbook on Decision Support Systems 2*. Springer, 2008.

[2] Business intelligence: What it is, how it works, its importance, examples, tools. `https://www.tableau.com/learn/articles/business-intelligence`, [En línea, consultada 08-02-2022].

[3] Centro Ucampus. Investigación ies chile y latinoamérica. documento interno no publicado., 2021.

[4] CNA. Guía para la autoevaluación interna interna acreditación institucional.

[5] Definición erp. `https://es.wikipedia.org/wiki/Sistema_de_planificaci%C3%B3n_de_recursos_empresariales`, [En línea, consultada 28-07-2021].

[6] Diagrama api mufasa. `https://ucampus.uchile.cl/api/0/mufasa`, [En línea, consultada 10-02-2022].

[7] Etl is not dead, it is still crucial for business success. `https://dataintegrationinfo.com/etl-is-not-dead/`, [En línea, consultada 09-02-2022].

[8] Gabriel González. *Diseño de una herramienta de apoyo a la toma de decisiones en instituciones de educación superior en base a la información curricular de Ucampus*. Universidad de Chile, 2021.

[9] Sitio web banner. `https://www.ellucian.com/solutions/ellucian-banner`, [En línea, consultada 28-07-2021].

[10] Sitio web ellucian analytics. `https://www.ellucian.com/solutions/ellucian-analytics`, [En línea, consultada 28-07-2021].

[11] Sitio web peoplesoft scorecard. `https://docs.oracle.com/cd/E41507_01/epm91pbr3/eng/epm/pbsc/concept_PeopleSoftScorecardOverview-991815.html`, [En línea, consultada 28-07-2021].

[12] Sitio web powercampus. `https://www.ellucian.com/es/soluciones/ellucian-powercampus`, [En línea, consultada 28-07-2021].

[13] Sitio web sisrel. `http://www.sisrel.cl/suite-academica.php`, [En línea, consultada 28-07-2021].

[14] Sitio web u+ scorecard. https://www.bettersoft.cl/servicio09.html, [En línea, consultada 28-07-2021].

[15] Tableau prep (project maestro) is here: Redefining your data prep experience. https://www.tableau.com/about/blog/2018/3/tableau-prep-project-maestro-here-redefining-your-data-prep-experience-84620, [En línea, consultada 09-02-2022].

[16] What is etl? (extract, transform, load). https://www.edq.com/blog/what-is-etl-extract-transform-load/, [En línea, consultada 09-02-2022].

Anexo

Ingresos:

- OCUPACION_VACANTES: Señala que porcentaje de las vacantes son efectivamente utilizadas para el año relevante.
- TOTAL_INGRESOS: Muestra cuántos nuevos estudiantes ingresaron a la unidad académica en el año seleccionado, agrupados por sexo.
- INGRESOS_REGIONES: Muestra la cantidad de alumnos que hicieron ingreso a la unidad académica según su región de origen.
- TIPO_ESTABLECIMIENTO: Indica la cantidad de alumnos que hicieron ingreso según el tipo de establecimiento donde terminaron su educación media, ya sea público, subvencionado, o privado.
- TIPO_INGRESO: Informa la cantidad de estudiantes que hicieron ingreso según tipo (excluyendo el ingreso tradicional), ya sea por bachillerato, intercambio, equidad de género, etc.
- NEM_PROMEDIO: Muestra el promedio de notas de educación media de una cohorte de ingreso.
- PRUEBA_PROMEDIO: Indica el promedio de los puntajes de ingreso de una cohorte, una vez ponderados según los requisitos de la carrera respectiva.

Estudiantes:

- TOTAL_ESTUDIANTES: Cantidad total de estudiantes en una carrera/facultad independiente de su cohorte de ingreso.
- RETENCION_ACTUAL_1: Dada la cantidad de alumnos que ingresaron a una unidad académica en el año recién pasado, qué porcentaje de estos sigue activo en la carrera en el año actual.
- RETENCION_PROMEDIO_1: Similar al indicador anterior, pero promediando alguna cantidad por definir de años.
- TITULACION_OPORTUNA_ACTUAL: Dada la duración ideal n de una carrera, que porcentaje de la cohorte de ingreso del año actual-n se encuentra actualmente titulada.

- TITULACION_OPORTUNA_PROMEDIO: Análogo al indicador anterior, pero promedio por una cantidad de años.
- MOV_EST_EMIGRANTES: Cantidad de estudiantes originarios de la unidad académica se encuentran en programas de movilidad estudiantil en instituciones externas.
- MOV_EST_INMIGRANTES: Cantidad de estudiantes originarios de instituciones externas realizando programas de movilidad estudiantil en la unidad académica.
- OCUPACION_REC_TITULADOS_ACTUAL: Porcentaje de recién titulados (hace un año o menos), que ya han ingresado al mercado laboral.
- OCUPACION_REC_TITULADOS_PROMEDIO: Similar al indicador anterior, pero promediado en una cantidad de años a determinar.

Docentes:

- TOTAL_DOCENTES: Cantidad de docentes asociados a una unidad académica.
- DOC_JORNADA_COMPLETA: Cantidad de docentes con jornada completa asociados a una unidad académica.
- DOC_JORNADA_MEDIA: Cantidad de docentes con media jornada asociados a una unidad académica.
- DOC_JORNADA_HORAS: Cantidad de docentes que trabajan por hora asociados a una unidad académica.
- DOC_DOCTOR: Cantidad de docentes con doctorado asociados a una unidad académica.
- DOC_MAGISTER: Cantidad de docentes con grado de magíster asociados a una unidad académica.
- RENTA_PROMEDIO_DOCENTES: Promedio de la remuneración de todos los docentes asociados a una unidad académica.
- PORCENTAJE_DOC_INVESTIGACION: Porcentaje de docentes que en el año en curso están involucrados en proyectos de investigación.

Institucional:

- AÑOS_ACREDITACION: Cantidad de años que le fueron concedidos a la carrera/institución en su último proceso de acreditación.
- ACT_VINCULACION_MEDIO: Este indicador no ha podido ser apropiadamente definido aún.
- CONVENIOS_INST_PRESTIGIO: Este indicador no ha podido ser apropiadamente definido aún.
- PROYECTOS_INVESTIGACION: Cantidad de proyectos actualmente en curso por académicos de la unidad académica.

- PUBLICACIONES_INVESTIGACION: Cantidad de publicaciones realizadas a lo largo del año en curso por parte de académicos de la carrera/facultad.
- ESTUDIANTES_POR_PROYECTO: Cantidad de estudiantes promedio entre los proyectos en curso.
- VENCIMIENTO_ACREDITACION: Fecha de vencimiento de la acreditación actual de la unidad académica correspondiente.