Maestría en Ingeniería del Software

Universidad Nacional de La Plata

Facultad de Informática

Tesis: Metodología para construcción de procesos de migración de datos en contexto de sistemas en desarrollo

Director: Doctor Marcelo Martin Marciszack

Co director: Doctor Gustavo Rossi

Maestrando: Ingeniero Silvio Serra

Dedicatoria

Humildemente quisiera dedicar este trabajo a mis hijos, Salvador y Magdalena, por su constante apoyo y colaboración y por todos los momentos que les he quitado para poder concretar esta etapa.

También a mi papá Edgardo Serra y a mi mamá Leticia Ronca por haber sembrado en mí estas ganas enormes de aprender y descubrir todos los días algo más.

Agradecimientos

Quisiera agradecer muy especialmente a mi director de tesis Dr. Marcelo Martín Marciszack, no sólo por su constante apoyo y corrección de este trabajo sino por constituirse una guía en muchos aspectos profesionales y personales. Más allá de sus muchas ocupaciones, siempre con un tiempo para mí, para mis dudas, para leer una y otra vez lo que escribo y para explicarme lo que me cuesta entender.

También quiero agradecer al Ing. Sergio Quinteros que dirige el Centro de Investigación y Desarrollo de Sistemas (CIDS) de la Universidad Tecnológica Nacional, Facultad Regional Córdoba, en el que he podido realizar una importante implementación de la metodología y la arquitectura propuestas en este trabajo. En el mismo ámbito, el Ing. Felipe Steffollani ha constituido un gran apoyo durante esta tarea.

Resumen

Este trabajo propone una metodología especialmente diseñada para construir mantener e implementar en forma eficiente, migraciones de datos que ocurren en el contexto del desarrollo de un nuevo sistema de información. En ocasiones un nuevo sistema requiere para ponerse en marcha la incorporación de datos heredados de aquellos que reemplaza, esto da origen a un proyecto de migración que ocurre en paralelo al del desarrollo del sistema. El aspecto particular de dicho entorno es la aparición de frecuentes cambios en la nueva base de datos como consecuencia de la evolución en la construcción del sistema, lo que obliga a adaptar la migración. Se obtiene entonces, un proceso con las complejidades propias de una migración, pero en un entorno inestable.

Como respuesta, en este trabajo se define una arquitectura física que separa las transformaciones sintácticas de las semánticas y una metodología que delinea dos procesos iterativos que posibilitan implementar cambios frecuentes en la base de destino de manera precisa y ordenada.

Abstract

This work proposes a methodology specially designed to build, maintain, and implement in an efficient way, data migrations that occur in the context of the development of a new information system. Sometimes, a new system requires to start up the incorporation of data inherited from those it replaces, this gives rise to a migration project that occurs in parallel to the development of the system. The particular aspect of this environment is the appearance of frequent changes in the new database as a consequence of the evolution in the construction of the system, which forces to adapt the migration. You get a process with the complexities of a migration, but in an unstable environment.

In response, this paper defines a physical architecture that separates syntactic and semantic transformations and a methodology that delineates two iterative processes that make it possible to implement frequent changes in the destination base in a precise and orderly manner.

Índice

Índice de figuras

Capítulo I: Introducción y objetivos

Introducción

El presente trabajo de tesis aborda la problemática de los procesos de migraciones de datos en el contexto de sistemas en desarrollo. Realiza un análisis de técnicas y herramientas existentes y propone una arquitectura y una metodología a aplicar durante migraciones complejas y de gran volumen especialmente diseñadas para realizar en forma eficiente migraciones, cuando la base de datos de destino aún está sujeta a cambios estructurales o lógicos. Es importante aclarar que el entorno objeto de análisis se aplica al desarrollo de sistemas de información, abarcando tanto sus aspectos técnicos como funcionales, tal como se analiza en detalle en los capítulos siguientes.

En ocasiones, al abordar el desarrollo de un nuevo sistema, se requiere la migración de datos de uno o varios sistemas anteriores. En estos casos, la nueva plataforma que difiere no sólo en aspectos técnicos sino, más aún, en cuestiones funcionales debe incorporar desde su puesta en marcha información registrada y administrada por las herramientas informáticas que reemplaza. Como afirma Robert Alan “la migración de datos de sistemas heredados es un proceso complejo” (1).

Esto impone un desafío muy significativo en lo técnico porque pueden existir grandes diferencias entre las tecnologías que administran los datos actuales y las que serán puestas en marcha, pero además, genera la necesidad de incorporar en el nuevo modelo de datos información que responde a sistemas anteriores. En otras palabras, las diferencias de tecnología no son menores, pero aún más notables suelen ser las diferencias funcionales.

Puede ocurrir que el proceso de migración alcance un elevado nivel de complejidad. Algunos datos en la nueva aplicación pueden ser muy importantes y en los sistemas anteriores estar registrados de manera diversa y sin mismas validaciones. Algunos datos pueden estar en formato diferente. Otros pueden no existir en las soluciones anteriores y deben calcularse o deducirse.

Si el proceso de migración resulta ser complejo y además se requiere que la información esté disponible en el momento de la puesta en marcha, no es posible esperar a que la aplicación esté totalmente construida para comenzar su diseño y construcción. En estas situaciones, se debe trabajar en la migración en forma paralela al desarrollo del nuevo sistema, pero esto implica migrar datos hacia un destino en construcción que puede encontrarse sujeto a cambios. Como lo afirma Ian Sommerville, "La evolución de los requerimientos es inevitable" (2). Sin dudas, algunos cambios de requerimientos pueden afectar la estructura de datos.

Entonces, si el proceso de migración de datos heredados es complejo (1) y si a la vez los requerimientos evolucionan (2) y esto puede afectar la composición de la base de destino, se está en presencia de una situación compleja e inestable.

En estos casos se obtienen dos procesos de desarrollo que corren en paralelo. Uno es el que involucra la construcción del sistema en sí y otro es el que debe resolver el proceso de migración. Cada uno tiene sus propios desafíos técnicos y funcionales. Cambios de requerimientos en el sistema pueden implicar cambios en el proceso de migración y a la inversa.

Mantener ambos procesos coordinados y avanzando a la par con la menor cantidad de retrabajo posible puede resultar complejo y como se analiza más adelante, las metodologías clásicas se centran naturalmente en la complejidad inherente de la migración, pero no en este paralelismo.

Objetivo

Objetivo principal

- Proponer una metodología que permita construir, mantener e implementar en forma eficiente, procesos de migración de datos en el contexto de sistemas en desarrollo.

Objetivos secundarios

- Analizar las dificultades propias de las migraciones de datos cuando ocurren dentro de un proceso de desarrollo de sistemas.

- Explorar herramientas y modelos existentes para migraciones de datos. Analizar sus posibilidades de implementación en el contexto ya mencionado.
- Comparar técnicas y herramientas propuestas por los principales fabricantes de motores de bases de datos contra la metodología propuesta.
- Confirmar el desempeño de la metodología propuesta dentro de un caso de implementación real.

Motivación

Como se ha dicho antes, los procesos de migración de datos pueden resultar complejos. Es de esperarse que existan diferencias técnicas entre una plataforma y otra y también diferencias de diseño (3).

Las bases de datos de origen, aquellas donde reside la información a migrar, responden a una plataforma que puede ser muy diferente del destino en el que se pretenden almacenar los datos como resultado de la migración. El desarrollo tecnológico es muy veloz, los recursos cambian permanentemente y con el paso del tiempo surgen tecnologías nuevas que pueden incorporar elementos muy distintos.

En este aspecto, las herramientas de extracción y transformación de datos (ETLs) resultan de gran importancia en todo proceso de migración pero muchas veces son interpretadas como elementos muy simples cuando en realidad estos procesos rara vez son sencillos (4). Además, por sí mismas no resuelven el problema en el contexto de sistemas en desarrollo.

Por ejemplo, si se compara un sistema desarrollado recientemente, con otro que tenga apenas 10 o 15 años de uso, sin duda se encontrarán muchas diferencias técnicas. Desde los lenguajes de programación utilizados, hasta los dispositivos sobre los que se ejecuta pasando por los sistemas operativos y también las herramientas utilizadas para almacenar y recuperar datos.

Cuando un sistema va a reemplazar a otro, los sistemas operativos pueden cambiarse, también el equipamiento, así como muchos otros recursos y aunque esto implica un costo importante, puede calcularse e incluirse como parte del costo de desarrollo e implementación. En todo caso se tratar de una relación costo – beneficio que debe ser objeto de análisis dentro del proyecto. ¿Pero qué ocurre con la información? ¿Puede reemplazarse o descartarse? En muchos

casos NO. La información que reside en las plataformas anteriores puede tener gran valor y de allí la necesidad de transformarla para que siga disponible.

Los diferentes fabricantes de bases de datos y productos relacionados han desarrollado diversas herramientas que permiten exportar e importar datos de una plataforma a otra. Gracias a estas herramientas es posible solucionar muchos problemas técnicos relativos al formato de almacenamiento. En concreto, aunque puede resultar laborioso, es factible tomar información almacenada en archivos de texto, en hojas de cálculo, archivos de formatos específicos como DBFs (Dbase Data File), etc. y transportarla a otra plataforma como bases de datos relacionales o de objetos. También es posible pasar datos de una base de datos soportada por un producto a otra de otro fabricante. (5) - (6) - (7).

Pero el principal problema no es técnico sino funcional. Las características del negocio y de las organizaciones donde se implementan los sistemas son dinámicas, cambian con el paso del tiempo y fuerzan a actualizar los recursos informáticos para tener nuevos sistemas que se adapten a las necesidades del presente. Como consecuencia, la nueva estructura de datos no sólo tendrá marcadas diferencias técnicas, sino también de diseño.

Pueden existir datos que en los sistemas anteriores no están y en el nuevo si, o al revés, o datos que en la plataforma anterior no implicaban validaciones o relaciones y en el nuevo diseño están asociados a cuestiones de negocio muy centrales. Hacer que los datos existentes con anterioridad estén disponibles en el nuevo sistema, convivan con los datos "nuevos" y puedan ser procesados por el sistema como cualquier otro registro es todo un desafío.

Normalmente no es posible esperar a que el nuevo desarrollo esté terminado para diseñar y construir la migración porque eso retrasaría significativamente la conclusión del proyecto. Se debe ir diseñando y construyendo el proceso de migración en paralelo con el desarrollo del sistema, pero eso provoca que un cambio funcional, o de diseño, o de requerimientos, o un cambio técnico (que frecuentemente ocurren en los procesos de desarrollo) impacte en el proceso de migración. Gestionar y administrar esos cambios suele resultar complejo.

También ocurre a la inversa, a veces al indagar en el proceso de migración, se detectan requerimientos (ya sean funcionales o no) que no estaban previstos o que se contradicen con el

diseño actual de la estructura de datos por la necesidad de conservar determinada información. Esto puede disparar cambios en el nuevo sistema.

Finalmente, la motivación para realizar la presente tesis radica en la complejidad de los procesos de migración, el alto grado de interdependencia que existe entre éstos y los procesos de desarrollo y la necesidad de obtener un mecanismo que permita disminuir el retrabajo y organizar las migraciones de modo eficiente.

Estado del arte

En la actualidad diversos sistemas informáticos tienen aplicación en prácticamente la totalidad de las actividades comerciales, industriales, sociales, educativas etc. En otras palabras: fábricas, comercios, escuelas, universidades, hospitales, clubes, entidades gubernamentales, etc. utilizan diversos sistemas informáticos en el desarrollo de sus actividades y depositan en ellos información vital para todo tipo de operatoria.

Al mismo tiempo, el avance de la tecnología sigue siendo muy veloz. En las últimas décadas nuevas posibilidades de empleo de recursos tecnológicos han surgido y lo hacen cada vez con mayor frecuencia. El uso de internet con nuevos fines, los dispositivos móviles, las comunicaciones más veloces y económicas son sólo algunos de los elementos que empujan a las organizaciones a enfrentar frecuentes cambios en sus sistemas informáticos.

Ambas cosas, el alto grado de utilización de la tecnología informática por un lado y la necesidad de frecuentes cambios por el otro, conducen a numerosas situaciones en las que un sistema más moderno reemplaza a otro, pero la información que ha sido almacenada por el sistema reemplazado sigue siendo vital para la organización y debe seguir disponible. Tal como se analiza más adelante en este mismo capítulo, Según IBM, el 48% de las migraciones de datos se realizan en circunstancias de actualización de tecnología (8).

Más aún, según la firma Oracle, hasta el 75% de los nuevos sistemas no cumplen con las expectativas, a menudo por fallas en el proceso de migración (9). El mismo artículo afirma que las migraciones de datos generalmente resultan de la introducción de un nuevo sistema.

Como se mencionó anteriormente, diferentes fabricantes de motores de bases de datos han desarrollado herramientas (algunas serán analizadas más adelante) que posibilitan la migración de datos de una tecnología a otra. Por ejemplo, es posible poblar una base de datos desde archivos de texto, o una base de datos de objetos desde un modelo relacional, o una base de datos relacional soportada por un producto de un fabricante a otra de otro fabricante. Pero estas herramientas no pueden resolver las diferencias funcionales ni organizar el proceso en sí y allí radica la verdadera dificultad.

En resumen, la situación actual puede expresarse de esta manera:

- Es muy frecuente que la implementación de un nuevo sistema informático se lleve adelante para reemplazar a otro.
- En estos casos, la información almacenada tiene gran valor y no puede perderse.
- El nuevo sistema y su antecesor no difieren solamente en aspectos técnicos, sino que lo hacen en cuestiones funcionales lo que presenta entonces un desafío técnico y funcional al momento de migrar datos.
- Si bien existen herramientas para resolver aspectos técnicos, sus posibilidades de utilización dependen del contexto y con ellas no se llega a abarcar todo el problema. La migración de datos continúa siendo un proceso artesanal, poco estandarizado y que puede presentar una gran dificultad tanto en el desarrollo como en la implementación de sistemas.

Organización y contenido

La presente tesis se encentra organizada de la siguiente manera:

Capítulo I: Introducción y objetivos.

Capítulo II: Análisis de procesos de migración de datos en sistemas en desarrollo. Realiza un análisis de las necesidades y problemas que se presentan en procesos de migración de datos cuando el sistema a implementar aún se encuentra en desarrollo.

Capítulo III: Metodologías y herramientas existentes para procesos de migración de datos. Es una revisión de metodologías y herramientas que diferentes fabricantes de motores de

bases de datos proveen para realizar migraciones y que pueden ser utilizadas exitosamente en el contexto de sistemas en desarrollo.

Capítulo IV: Propuesta de una metodología y un modelo físico para procesos de migración de datos en sistemas en desarrollo. Es una propuesta de una metodología especialmente diseñada para las situaciones mencionadas anteriormente que se apoya en las herramientas analizadas en el capítulo III.

Capítulo V: Aplicación de la metodología propuesta. Es la implementación de la metodología propuesta en un caso real. Se describen las acciones realizadas en el proceso de migración a partir de aplicar el modelo propuesto y los resultados obtenidos.

Capítulo VI: Discusión de resultados, conclusiones y posibles trabajos futuros. Realiza una evaluación de los resultados obtenidos y enuncia conclusiones, expone un análisis de las ventajas y desventajas del modelo propuesto y define posibles investigaciones futuras.

Capítulo II: Análisis de procesos de migración de datos en sistemas en desarrollo.

Introducción

En este capítulo se analiza la problemática de los procesos de migración de datos que deben diseñarse, construirse, probarse y en ocasiones implementarse cuando el sistema que recibirá la salida de dicho proceso aún se encuentra en desarrollo.

Se entiende por migración de datos a un conjunto de procesos por medio de los cuales los datos son obtenidos, transformados y transferidos entre dos o más aplicaciones (10). De esta definición queda claro que los procesos de migración de datos son aquellos que persiguen como objetivo transformar y transportar datos de una plataforma a otra, pero esto puede ocurrir en contextos diferentes y con motivaciones y objetivos distintos.

En ocasiones el sentido de la migración es poder hacer una transformación en los datos para hacer posible un análisis con un enfoque diferente al previsto en el sistema que los genera. Como ejemplo podemos citar los entornos de data mining y data warehousing. En ellos los datos se migran a una base especialmente diseñada para poder hacer un análisis particular de los mismos. Esta base puede contener información de un solo origen o de varios.

En otros casos la necesidad viene dada por razones de rendimiento. Algunos sistemas, sobre todo los que trabajan con grandes volúmenes de datos, pueden requerir en algunos procesos tiempos de respuestas difíciles de lograr. Una opción suele ser transportar datos desde una base transaccional a otra que esté organizada de manera tal que agilice las consultas y las acciones de búsqueda.

También es posible encontrar migraciones por razones de distribución geográfica. Algunos sistemas mantienen bases de datos replicadas en lugares geográficamente separados con el objeto de garantizar el acceso eficiente a los mismos. En estos entornos pueden hallarse operaciones de traspaso de una fuente a otra, entendiendo por fuente, un servidor, un conjunto de servidores o la infraestructura que los contenga. Cabe aclarar que aquí si bien hay un pasaje de datos, no siempre

hay una transformación de estructura o de formato, por lo que el término "migración" no es totalmente aplicable, aunque suele utilizarse.

Hay situaciones en las que el traspaso de datos se realiza por razones de seguridad o de disponibilidad o de facilidad de acceso, etc. En fin, pueden encontrarse muchas razones y muchos contextos en los que se implementan pasajes de datos de una estructura a otra con mayor o menor grado de transformación y con diferentes niveles de complejidad.

En particular son objeto de estudio en esta tesis aquellas situaciones en las que el motivo para migrar datos radica en el reemplazo de un sistema por otro y éste último aún se encuentra en desarrollo. En este contexto pueden utilizarse muchas de las técnicas y herramientas empleadas en cualquier proceso de migración, pero teniendo en cuenta que el destino de la información (o sea, la base de datos que se pretende obtener) no es estable en su estructura ya que está sujeta a cambios.

Más allá del contexto y del tipo de migración de datos de que se trate, algo que es importante destacar y que sin duda es un factor clave que agrega complejidad, es que las migraciones rara vez son procesos meramente técnicos (10). Como ya se ha dicho en el capítulo anterior, aunque los aspectos técnicos como por ejemplo transformaciones de tipos de datos, cambio en relaciones de las entidades de la base de datos, cambio en las longitudes de campos etc. son de una complejidad relevante, los aspectos funcionales son los que verdaderamente determinan el grado de dificultad de todo el proyecto.

Justamente en el contexto objeto de estudio, estos factores más funcionales que técnicos, son particularmente volátiles por los propios cambios de requerimientos que frecuentemente se afrontan durante el desarrollo de un sistema.

La figura siguiente muestra las situaciones frecuentes por las que se realizan migraciones de datos según una publicación realizada por IBM (8). El caso de estudio de esta tesis se ubica dentro de lo que IBM considera en su análisis como migraciones por actualización de tecnología, la causa más frecuente.

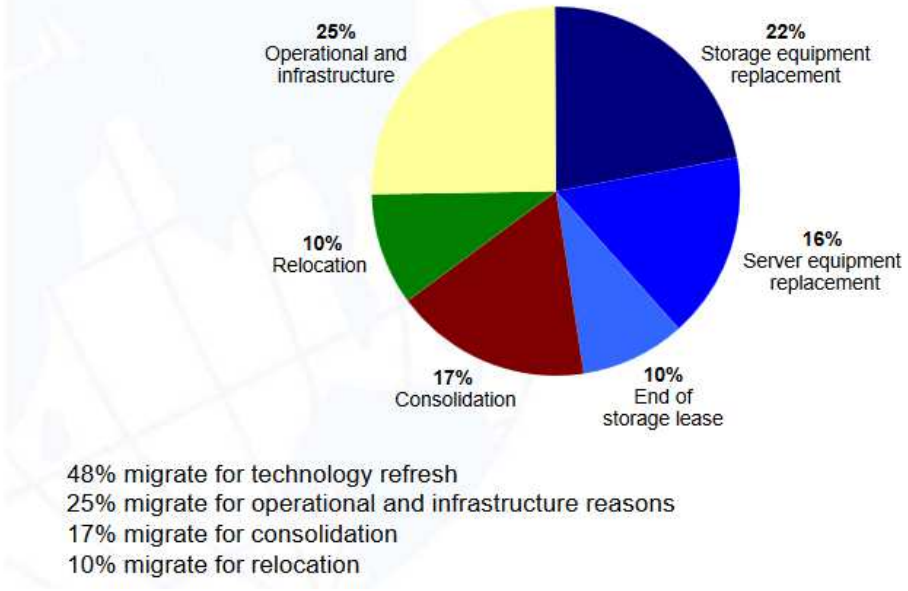


Figura 1: Razones para migrar datos (8).

Si bien el enfoque de IBM en este estudio es la actualización de tecnología en el hardware, muchas organizaciones renuevan su hardware y simultáneamente su software para mantener actualizada su infraestructura informática.

Según IBM, el 48% de las migraciones de datos se realizan en circunstancias de actualización de tecnología. Sin dudas, en muchos de esos casos, la base de datos destino de la migración será completamente nueva en su diseño o al menos habrá recibido modificaciones estructurales como parte de la misma renovación.

Elementos presentes en un proceso de migración de datos

Si bien no todas las migraciones son iguales ya que el contexto determina muchas de sus características, resulta frecuente encontrar al menos estos tres elementos (3):

Fuente de datos de origen: Es la base de datos (o el repositorio correspondiente) que contiene los datos en su formato actual. Es la fuente desde donde se desea tomar la información a ser migrada.

Fuente de datos de destino: Es la base de datos (o el repositorio correspondiente) que se pretende poblar como resultado del proceso de migración. Es el destino que se le quiere dar a la información.

Proceso de transformación: Es el proceso propiamente dicho. Aquí pueden intervenir herramientas diversas, algunas serán analizadas más adelante. Es posible encontrar elementos muy automatizados como por ejemplo los servicios de transformación de datos (DTS) de Sql Server o los servicios de integración de datos (BIDS) del mismo fabricante (11) u Oracle Warehouse Builder (OWB) (12) de Oracle, sólo por citar algunas herramientas de algunos fabricantes. También es factible que participen otros elementos no tan automatizados para dar paso a soluciones a medida como por ejemplo código escrito mediante lenguajes de programación como Transact SQL (13), PL SQL (14) y otros.

El siguiente gráfico ha sido extraído de una publicación realizada en International Journal of Engineering Science and Innovative Technology (IJESIT) (3).

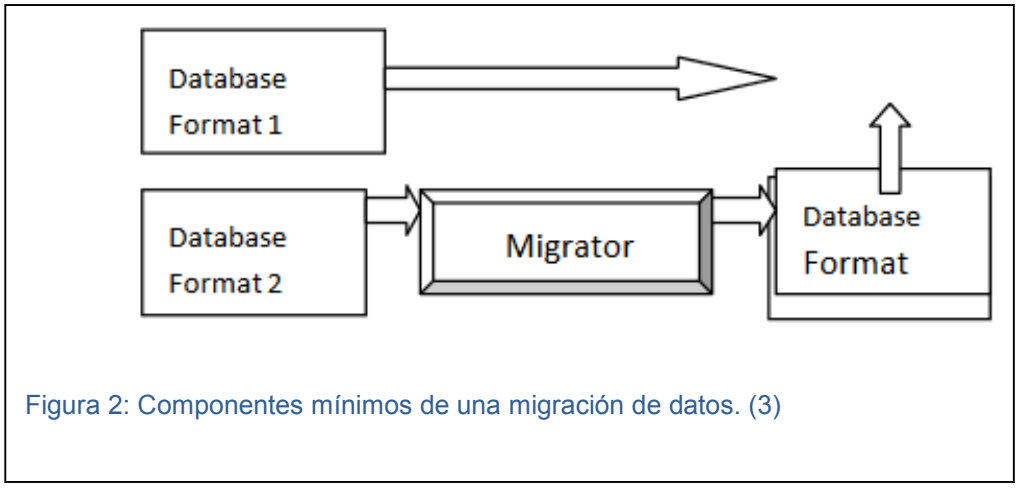


Figura 2: Componentes mínimos de una migración de datos. (3)

Las bases de datos de origen pueden ser más de una y normalmente se pretende obtener una sola base como resultado, aunque éste no es un factor excluyente, pueden existir casos en los que se desee obtener más de un destino.

En el contexto de estudio de esta tesis, la fuente de datos de origen normalmente es estable. Responde a un sistema que se encuentra funcionando y sobre el que no se realizan cambios estructurales significativos. Pero la base de datos de destino se encuentra en construcción, en constante movimiento por los cambios que implica el propio desarrollo del sistema y esto hace que el proceso de transformación deba ser dinámico.

Como se analiza más adelante, **el problema central es mantener actualizado el componente "proceso" frente a cada cambio del componente "destino", teniendo en cuenta que los cambios en la base de destino no son originados por el proceso ni por necesidades propias de la migración.**

Complejidad en los procesos de migración de datos y asignación de recursos

Independientemente del contexto, las migraciones de datos rara vez son simples. Las diferencias de tecnología, de estructuras de datos, de tipos de datos, de sintaxis en las diferentes plataformas de programación, etc. hacen de estos procesos algo complejo, de allí la importancia de hacer una correcta planificación y asignación de recursos.

En particular, en el contexto de estudio de esta tesis, tal como lo afirma Robert Alan, En muchas ocasiones los esfuerzos están más enfocados en el sistema nuevo y en sus requerimientos y muy pocos esfuerzos son enfocados en el proceso de migración. (1)

Esa tendencia es natural si se analiza la situación. El requerimiento inicial ha sido el de construir un sistema y como consecuencia se ha iniciado todo un proceso de desarrollo que seguramente incluye analistas funcionales, especialistas en requerimientos, desarrolladores, especialistas en testing, analistas de tecnología, gerentes, directores, etc. Todos enfocados en el desarrollo de la nueva plataforma porque éste es el objetivo del equipo de trabajo y es lo que ha dado origen al proyecto en sí.

La migración de datos es una consecuencia del cambio del sistema y en muchas ocasiones es vista como una dificultad a resolver más que como parte del proceso de desarrollo. De allí que la tendencia sea dedicarle menos recursos y atención que a otros aspectos del proyecto.

Al igual que en proyectos de desarrollo o implementación (o de cualquier otro tipo incluso de otras ramas de la ingeniería) la asignación de recursos es un factor clave. No debe sorprender que si no se han asignado los recursos apropiados la migración no llegue a buen término.

En 2005 la firma ESG (Enterprise Strategy Group) realizó un estudio de campo con empresas del medio y obtuvo resultados algo llamativos. La figura siguiente muestra la forma en que se planifican y utilizan tres recursos claves durante migraciones de datos, éstos son: El personal afectado, el tiempo de inactividad de la aplicación y el presupuesto. (15) (10).

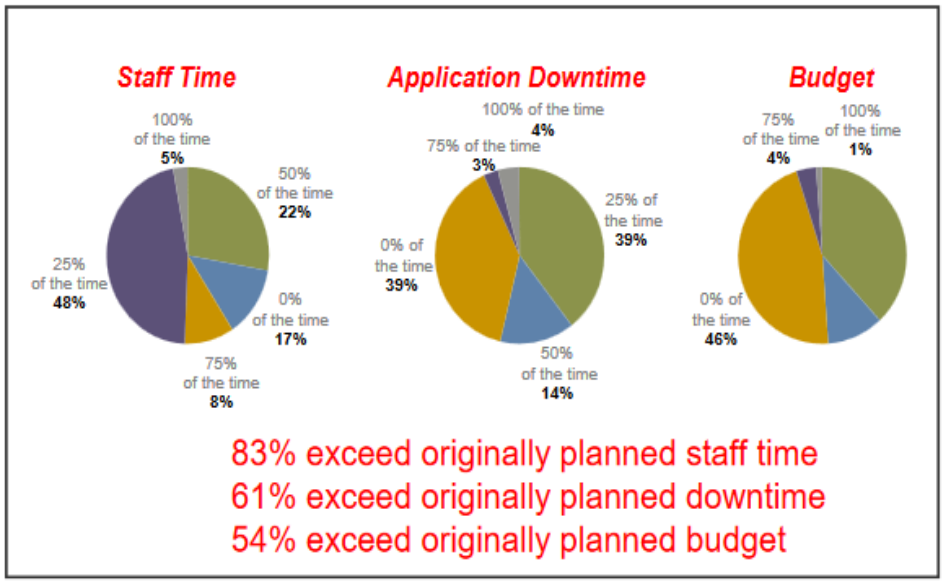


Figura 3: Recursos en los procesos de migración de datos. (15) (10).

Según este estudio, sólo el 17% de los procesos de migración de datos son realizados con la cantidad de horas de personal estimado originalmente, mientras que un 83% excede los límites planificados. En más de la mitad de los casos el exceso supera el 50%.

En el 61% de los casos las aplicaciones que utilizan esa base de datos han tenido que mantenerse fuera de línea más tiempo del planificado.

Menos de la mitad de las migraciones logran concluirse dentro del costo previsto.

Sin duda estos indicadores hablan, por un lado, de la complejidad propia del proceso, pero a la vez hacen notar las dificultades que existen en la planificación. Prever los recursos necesarios para realizar una migración es complejo.

El escenario resulta más complicado cuando la base de datos de destino no tiene aún su estructura definitiva. A las dificultades propias de toda migración hay que sumarle la inestabilidad de una base de datos que se encuentra en desarrollo y que recibe cambios de requerimientos que conllevan a modificaciones estructurales. De allí la importancia de contar con una metodología que permita asimilar los cambios con el costo más bajo posible y adaptar el proceso de migración de manera ordenada y eficiente.

Problemas frecuentes en los procesos de migración de datos

Las dificultades que se pueden presentar dentro de un proceso de migración de datos son muy variadas, sin embargo, hay algunas situaciones que son frecuentes y que requieren especial atención.

La figura siguiente muestra los problemas más frecuentes que se presentan en los proyectos de migración de datos, según el mismo estudio mencionado en el apartado anterior realizado por la firma ESG y publicado por Nancy Hurley (15) (10).

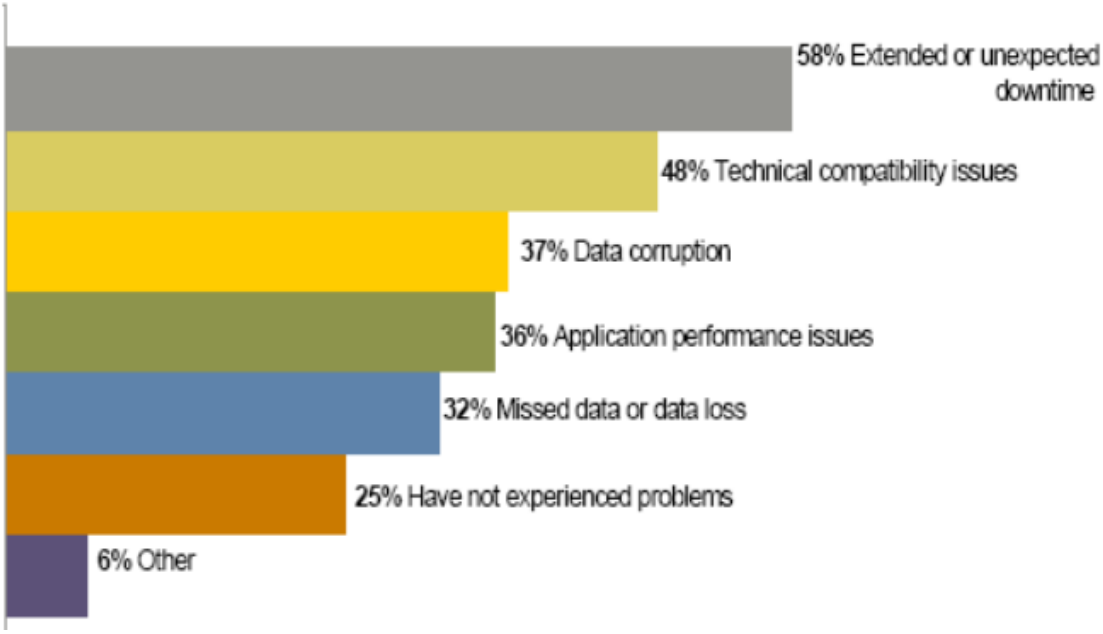


Figura 4: Problemas frecuentes en los procesos de migración de datos (15) (10).

Según este estudio, sólo en el 25% de los casos no se experimentaron dificultades. En el 58% de los casos la aplicación que utiliza los datos migrados ha tenido que estar fuera de línea más tiempo del previsto. En el 69% de los casos se presentaron situaciones de corrupción o de pérdida de datos. Además, hay que considerar que estas categorías no son mutuamente excluyentes, por lo que un proyecto puede (y de hecho es lo que ocurre) presentar combinaciones de ellas.

Tomando otra fuente, La figura siguiente expone una conclusión a la que arriba la firma Oracle. (16)

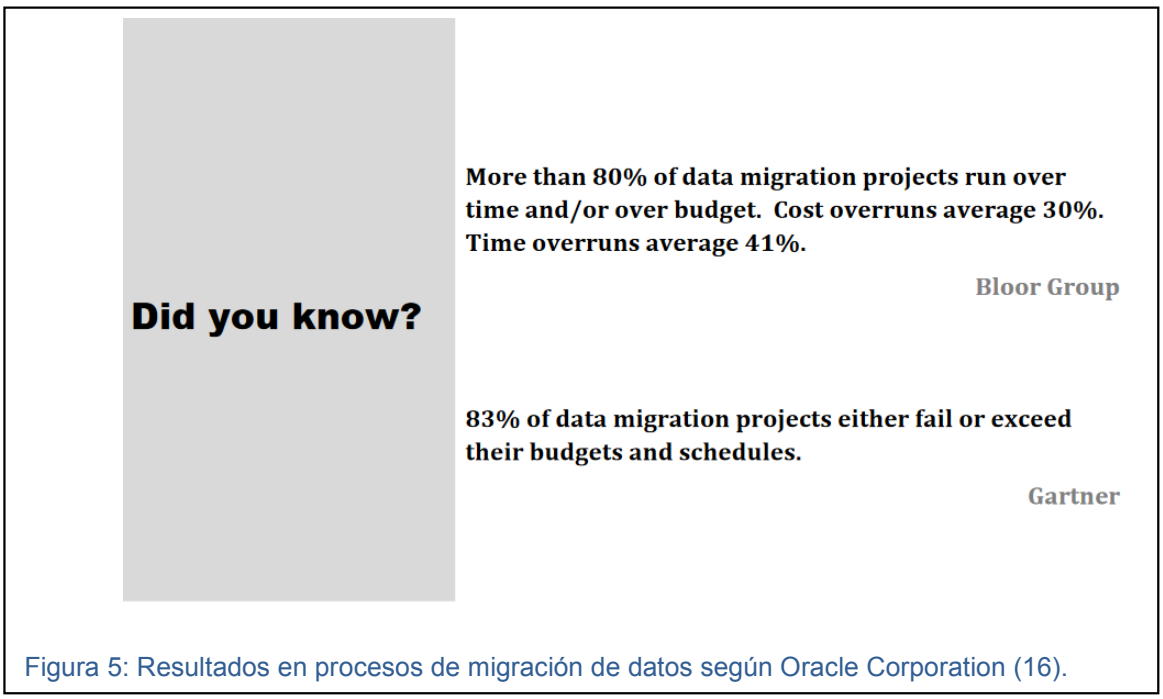


Figura 5: Resultados en procesos de migración de datos según Oracle Corporation (16).

Según el estudio realizado por Oracle, más del 80% de los proyectos de migración de datos exceden el tiempo y el costo originalmente estimado o directamente, no llegan a concluirse.

En el contexto de estudio, las causas que originan estas dificultades son particularmente difíciles de neutralizar y a la vez tienden a reaparecer. Los cambios en la base de destino llevan a modificar el proceso, si estas modificaciones no ocurren con rapidez y prolijidad, la propia migración comenzará a tomar un camino errático presentando cada vez más problemas y alejándose consecuentemente de los plazos y costos estimados.

En conclusión, la planificación de recursos y la correcta estimación de tiempos es central, pero en un porcentaje importante de los casos esta tarea no se realiza o bien, a la luz de los resultados, se hace deficientemente. Si la base de datos de destino va a recibir cambios durante el proceso de migración, realizar y mantener una correcta planificación es de mayor importancia, pero a la vez, es una tarea de significativa dificultad. Esto constituye una situación particularmente compleja.

Impacto de los cambios de requerimientos

Uno de los grandes desafíos que sin duda se enfrentan en la ingeniería de sistemas es realizar una correcta gestión de los requerimientos. No sólo relevarlos correctamente sino mantenerlos, validarlos y en general gestionarlos durante todo el ciclo de desarrollo.

Más allá de las particularidades de cada proyecto, en mayor o menor medida la gestión de requerimientos siempre presenta un gran desafío ya que éstos, por diferentes motivos, cambian durante el desarrollo.

Como dice Ian Sommerville, La evolución de los requerimientos durante el proceso de ingeniería de requerimientos y después de que un sistema esté en uso es inevitable (2). La figura siguiente ilustra el cambio de requerimientos conforme pasa el tiempo (2).

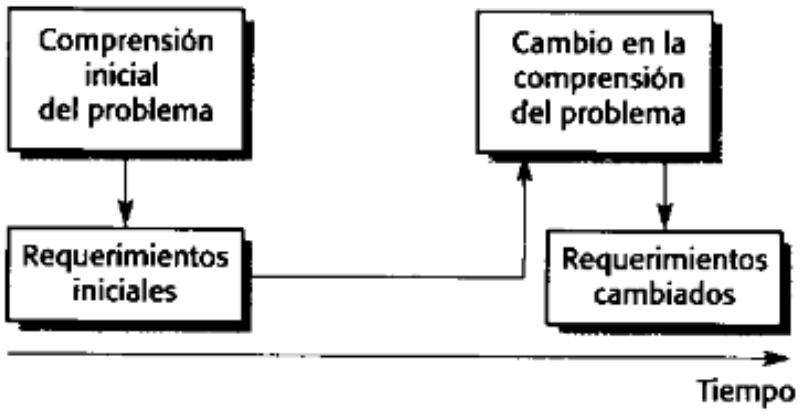


Figura 6: Evolución de los requerimientos. (2)

Por otro lado, en apartados anteriores se han analizado las dificultades que los procesos de migración de datos atraviesan normalmente, pero ¿Qué ocurre si además se considera un contexto en que los requerimientos cambian?

Durante el ciclo de desarrollo pueden aparecer cambios que no afectan la estructura ni las reglas generales de la base de datos. En estos casos, el proceso de migración no se ve alcanzado y dichos cambios pueden procesarse de manera independiente al desarrollo de la migración.

Pero hay otros casos en los que sí existe un impacto en la base de datos. Por ejemplo, cuando una modificación en requerimientos implica crear un nuevo campo o eliminar un campo existente, cambiar tipos de datos, modificar longitudes de campos, eliminar o crear tablas, cambiar reglas de validación de datos, eliminar o crear relaciones entre entidades, etc. En todos estos casos, un cambio de requerimiento implica cambios en la base de datos que es la salida de la migración y allí es donde el proceso debe adaptarse.

La figura siguiente ilustra la implicancia de los cambios en la base de destino y la necesidad de gestionar eficientemente las correspondientes modificaciones en el proceso.

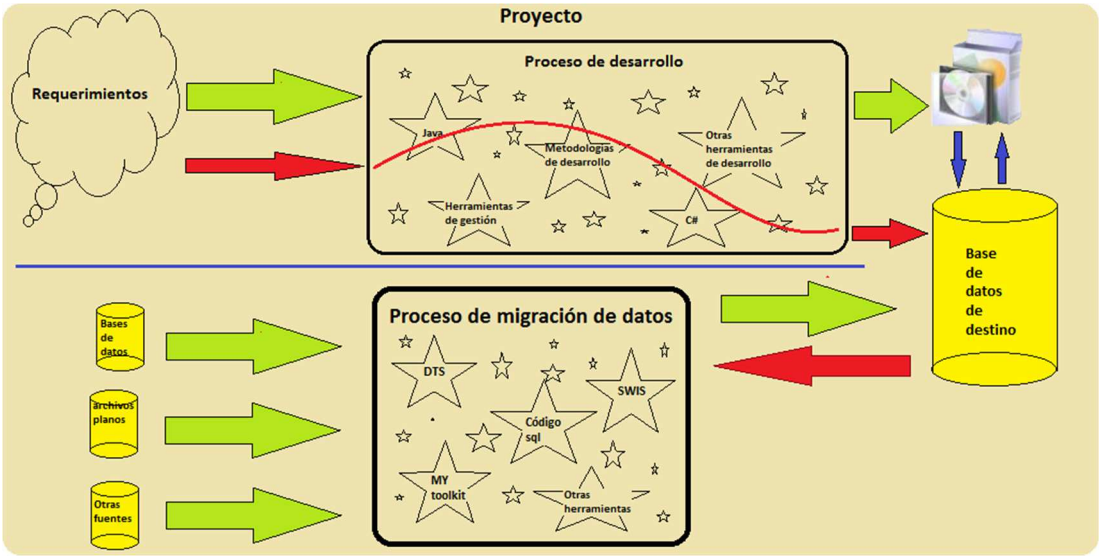


Figura 7: Impacto de los cambios de requerimientos en el proceso de migración de datos.

En el desarrollo de la migración de datos intervienen muchos actores. Además de los programadores, diseñadores, especialistas en bases de datos, etc. hay una variedad de herramientas y metodologías que se emplean para lograr el objetivo de obtener una base de datos que responda

a las necesidades del nuevo sistema pero que a la vez incorpore la información proveniente de las fuentes anteriores que, como ya se ha mencionado, pueden ser de variada tecnología.

En el proceso de desarrollo del sistema también intervienen recursos humanos con diferentes roles además de diversas herramientas y metodologías que persiguen un objetivo diferente. En este contexto se tiene como meta el desarrollo del sistema en sí, que grabará y leerá información de la base de datos destino de la migración.

Ambos procesos pueden llevarse adelante de forma totalmente independiente. Cada cual puede tener su propia planificación, objetivos, plazos, hitos, etc. pero en realidad no son absolutamente inconexos, sino que, por el contrario, pueden tener una fuerte interdependencia.

Cuando un nuevo requerimiento se presenta o surgen modificaciones de uno existente, el proceso de desarrollo absorbe ese cambio y dispara los mecanismos necesarios para que el sistema finalmente responda dichos requerimientos. Pero en algunos casos, para que esto ocurra, se requieren modificaciones en la base de datos.

El proceso de migración no recibe los cambios de requerimientos como una entrada natural porque sus entradas están constituidas por las estructuras de datos de origen. El nuevo requerimiento no afecta su entrada, sino que afecta su salida, porque impone cambios en la base de datos de destino.

De esta manera el desarrollo de la migración debe adaptarse a cambios que se imponen desde otro proceso y disparar sus mecanismos para absorber dichos cambios utilizando todas sus herramientas y actores, pero a partir de modificaciones en la salida no en la entrada.

En resumen, en apartados anteriores se han mostrado diferentes estadísticas que en definitiva hablan de la dificultad que frecuentemente se enfrenta en los proyectos de migraciones de datos para cumplir con los plazos y costos. Un número importante de este tipo de proyectos no logran concluirse dentro del plazo previsto y consumen muchos más recursos de los planificados. Incluso en un porcentaje importante, no llegan a terminarse. El escenario objeto de estudio incorpora a todas estas dificultades una variante más que tiene un fuerte impacto en la planificación y en la estimación de costos. La base de datos de destino, o sea, la salida del proceso de migración, está sujeta a cambios que no dependen de la entrada ni del proceso en sí. De allí la importancia de

contar con una metodología de trabajo que organice especialmente la migración para este tipo de situaciones.

Nuevos requerimientos que surgen del proceso de migración

Ya se ha analizado lo que ocurre en el desarrollo de la migración a partir de la evolución en los requerimientos del sistema, pero también puede ocurrir lo inverso. Necesidades del proceso de migración pueden tener un impacto en el diseño y construcción sistema.

Al profundizar en las características de la información que debe ser tomada como origen del proceso de migración, pueden aparecer nuevos requerimientos que el sistema debe contemplar. Frecuentemente se trata de requerimientos no funcionales, aunque también pueden detectarse requerimientos funcionales.

Cuando se analizan las fuentes de datos que van a ser reemplazadas, se explora la información que el nuevo sistema debe conservar y sobre la que debe proveer acceso. Puede ocurrir que parte de esa información ya no sea valiosa para las nuevas transacciones (las que se realizarán sobre el nuevo sistema) y por tanto no haya sido tenida en cuenta en el diseño actual. Pero al mismo tiempo puede pasar que la misma desee conservarse, al menos para poder consultarla, sobre todo si los sistemas antiguos van a ser sacados de línea.

También puede ocurrir que se prevean tipos de datos diferentes a los que luego se detectan en el proceso de migración. Por ejemplo, un código puede incluirse como un campo numérico en el nuevo sistema, pero al avanzar sobre el proceso de migración puede descubrirse que en alguna de las fuentes de origen fue tratado como alfanumérico. Aquí habría varias soluciones posibles. Una sería adaptar el proceso de migración para que transforme esos valores a números con alguna lógica determinada (aunque se perdería el valor original), otra sería adaptar el nuevo sistema para que prevea valores alfa numéricos, y otra sería incluir estos datos en un nuevo campo. Las dos últimas opciones requieren ingresar modificaciones en el desarrollo del sistema.

Más allá de los casos particulares y de los ejemplos y experiencias que se puedan citar, lo importante de destacar en este punto es que, del análisis de los datos a migrar pueden surgir

cambios en el desarrollo del nuevo sistema, que luego se materializan como requerimientos funcionales o no funcionales.

¿Cuándo comenzar con el desarrollo de la migración?

Otras de las características importantes de los casos de migración objetos de estudio es cuando comenzar su desarrollo.

Tal como lo afirma la compañía especialista en migración de datos Powerdata, "todo proceso de migración debe estar alineado con la estrategia empresarial y, por ello, el plan de migración de datos se debe ajustar a las prioridades del negocio". (17)

Aquí intervienen dos variables importantes, por un lado, la necesidad de terminar la migración a tiempo para no provocar demoras en la puesta en marcha del sistema y por otro lado la importancia de que el desarrollo esté lo más avanzado posible para que la base de datos de destino sea menos inestable.

En apartados anteriores se han analizado diversas estadísticas provenientes de distintas fuentes que muestran las dificultades frecuentes que tienen los proyectos de migración de datos para poder ser concluidos en el tiempo planeado. Independientemente de la fuente que se tome, lo relevante es que resulta muy notable la dificultad para cumplir los plazos planificados en los procesos de migración. En el caso particular de estudio de esta tesis, una demora en el desarrollo de la migración implica una demora en la puesta en marcha del nuevo sistema.

También se debe tener en cuenta que la migración de datos no puede terminarse con el tiempo justo para poner el sistema en marcha. En muchas ocasiones otras áreas necesitan disponer de una base de datos poblada con la información proveniente de la migración antes de la puesta en marcha del sistema. Por ejemplo, para las actividades de testing puede resultar relevante analizar el comportamiento de la aplicación cuando opera sobre una base de datos generada a partir del proceso de migración, no sólo porque sobre ella pueden aparecer nuevos errores y defectos, sino porque se pueden analizar muchos casos de estudio al tener un volumen de información mayor, dando de este modo una cobertura de test más amplia.

También se debe tener en cuenta que el propio proceso de migración debe ser analizado y testeado. Si el área de testing debe recibir dicho proceso para someterlo a pruebas, se debe considerar el tiempo que estas actividades demanden al momento de planificar la puesta en marcha del sistema.

Otro factor relevante es la necesidad de realizar pruebas de carga. Si el nuevo sistema va a ponerse en marcha con una base de datos poblada con una cantidad significativa de registros migrados, puede resultar importante analizar el tiempo de respuesta que se obtiene al operar la aplicación. Si bien es posible realizar este tipo de pruebas sin contar con la base de datos migrada, las conclusiones a las que se arriben serán mucho más certeras si el análisis se realiza sobre datos y volúmenes reales.

Existen también situaciones en las que resulta difícil estimar la infraestructura requerida para alojar el nuevo sistema. Si bien hay técnicas y herramientas específicas para realizar cálculos y aproximaciones, contar con la base de destino de la migración completa da un valor mucho más exacto y de gran utilidad. Hay que tener en cuenta aquí que, si la migración toma como orígenes varias fuentes de distinta tecnología, no es simple estimar parámetros como por ejemplo espacio de almacenamiento en disco, uso de memoria para operaciones de consulta típicas, uso de procesador, etc.

Todos los puntos anteriores llevan a pensar que lo más conveniente es comenzar con el desarrollo del proceso de migración lo antes posible, pero hay otros factores a tener en cuenta.

Mientas más cambios se realicen al proceso de migración, más lento será su desarrollo, más recursos harán falta y menor será la posibilidad de concluirlo dentro de los plazos estimados.

La base de datos de destino no tiene una estructura libre que puede amoldarse a las necesidades de la migración. Como ya se ha dicho, su estructura es parte del sistema que se está desarrollando y está sujeta a éste. La evolución en los requerimientos es inevitable (2) y muchos cambios en los requerimientos determinan modificaciones estructurales en la base de datos destino de la migración.

La propia gestión de modificaciones en la migración puede resultar compleja. Cada cambio debe procesarse de manera ordenada, utilizando las herramientas correctas, validando que se

obtiene el resultado esperado y documentando las acciones realizadas. Sin dudas esto insume recursos, principalmente tiempo.

Si se analizan estos últimos factores, resulta conveniente comenzar con el proceso de migración lo más tarde posible para que éste se vea menos afectado por los cambios en la base de destino.

Finalmente, algunas necesidades llevan a concluir que el desarrollo de la migración debe comenzar lo antes posible. De ser factible, al mismo tiempo que el desarrollo del sistema o incluso antes. Pero otros factores llevan a asegurar que mientras más tarde mejor. De ser posible, después de que el sistema se encuentre desarrollado.

La tabla siguiente muestra las ventajas de comenzar el desarrollo de la migración de datos con mayor o menor atraso respecto del comienzo del proceso de desarrollo del sistema.

Comienzo temprano del desarrollo de la migración	Mayor disponibilidad de tiempo para concluir el desarrollo del proceso de migración.
	Mayor disponibilidad de tiempo para probar el funcionamiento del proceso de migración.
	Posibilidad de contar con una base de datos resultante del proceso de migración para ser utilizada en actividades de testing.
	Menor riesgo de demorar la implementación del sistema a causa de la migración de datos
	Posibilidad de contar con una base de datos similar a la que se utilizará en la puesta en marcha del sistema para poder realizar pruebas de carga y dimensionar con mayor exactitud la infraestructura.

Comienzo tardío del desarrollo de la migración	Mayor evolución en el desarrollo del sistema. Puede esperarse un menor impacto por cambios de requerimientos.
	Menor riesgo de sufrir demoras en el proceso por contar con una estructura de salida más estable.
	Mayor exactitud en la estimación de tiempo y recursos por ser más previsible todo el proceso.

Algunos de los factores expuestos en la tabla anterior pueden tener mayor o menor peso en diferentes proyectos. En algunos contextos obtener en forma temprana una base de datos resultante de la migración para poder hacer estimaciones de hardware (como por ejemplo espacio de almacenamiento) puede ser muy importante, mientras que en otras situaciones puede ocurrir que esto no sea necesario. Lo mismo con cualquier otro factor.

Queda claro entonces que definir el momento en el que debe comenzar el desarrollo de la migración es algo muy relevante y a la vez muy propio de cada proyecto ya que depende fuertemente del contexto.

Más allá de las características particulares de cada situación, una opción puede ser la siguiente:

1) Definir cuál es el momento más tardío posible en el desarrollo del sistema en el que se debe disponer de una base de datos resultante de la migración. Este momento puede ser la propia puesta en marcha, una puesta en pre-producción para realizar pruebas o cualquier otro hito del desarrollo del sistema.
2) A partir de ese momento considerar el tiempo estimado de desarrollo de la migración.
3) Considerar el tiempo de pruebas y otras actividades que se decidan realizar más un margen por ajustes y error en la planificación.

4) Restar estos tiempos de desarrollo, pruebas y otras actividades al momento en el que se debe disponibilizar la base de datos migrada y de esta manera se obtiene el momento de inicio.

La figura siguiente muestra el riesgo asumido en función del momento del desarrollo del sistema en el que se decida comenzar con la migración.

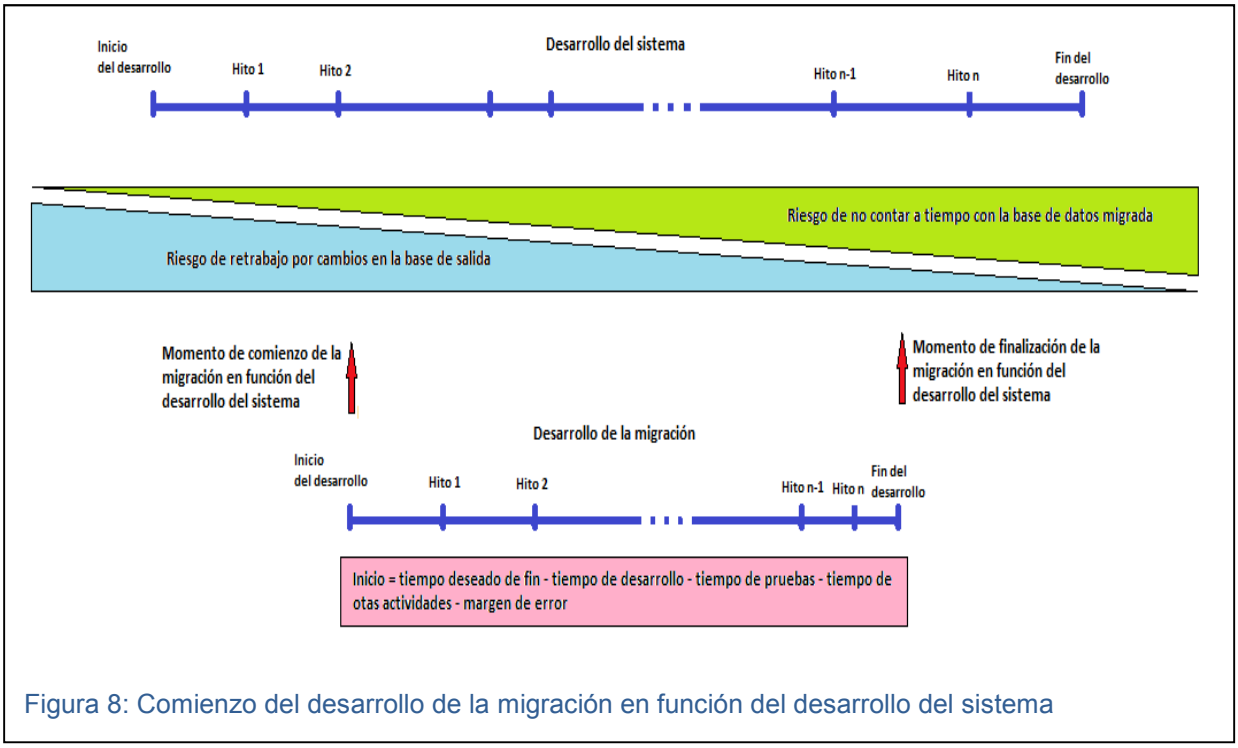


Figura 8: Comienzo del desarrollo de la migración en función del desarrollo del sistema

Conclusión

En este capítulo se ha analizado la situación general de los procesos de migración y se los ha situado en el contexto de estudio de esta tesis.

Según las diferentes fuentes consultadas, resulta notable la frecuente dificultad para concluir este tipo de proyectos dentro del tiempo planeado y consumiendo los recursos estimados.

En parte esto se debe a que muchas veces resultan más complejos de lo que aparentan ser y las herramientas de los diferentes fabricantes, si bien son de gran ayuda, no simplifican el proceso.

Luego se ha situado el análisis dentro del contexto de la construcción de un nuevo sistema. En este entorno, la migración constituye un proceso que se realiza en paralelo con el desarrollo del sistema, existiendo una fuerte interrelación entre ellos. Cambios de requerimientos en el sistema pueden implicar cambios en el desarrollo de la migración y, a la vez, algunos requerimientos planteados en la migración de datos pueden implicar cambios en el sistema.

Al mismo tiempo, la base de datos de destino de la migración no puede diseñarse y construirse libremente, sino que su estructura depende completamente del diseño y de los requerimientos del nuevo sistema. La migración debe adaptarse permanentemente a la estructura de la base de destino que no es estable, cambia en función del desarrollo.

Si resulta complejo estimar el tiempo de desarrollo y los recursos necesarios para una migración de datos, más complejo será aún en este contexto por la propia inestabilidad de la base destino. Además, hay que notar que una demora en la migración implica también un atraso en la puesta en marcha del sistema con lo que su impacto es importante.

Un factor muy relevante a tener en cuenta es el momento en el que se debe comenzar con el desarrollo de la migración. Si se lo inicia temprano respecto del desarrollo del sistema, se dispone de más tiempo, pero es más posible encontrar cambios en la base de salida. Si se lo inicia más tarde, la base de salida resulta más estable, pero se dispone de menos tiempo. La decisión depende del contexto de cada proyecto y es un factor muy relevante.

Finalmente, lo cierto es que los proyectos de migración frecuentemente resultan más complejos de lo previsto, a menudo se ven demorados y consumen más recursos que los planificados. Si ocurren dentro del desarrollo de un nuevo sistema resultan aún más complejos y difíciles de estimar. De allí la importancia de darles la atención necesaria y de contar con una metodología especialmente diseñada para estos casos.

Capítulo III: Trabajos relacionados, metodologías y herramientas existentes para procesos de migración de datos

Introducción

En este capítulo se analizan algunas metodologías y herramientas provistas por diferentes fabricantes para realizar migraciones de datos.

El objetivo es comprender su funcionamiento y arquitectura para observar que posibilidades tienen de ser aplicadas en migraciones que ocurren en el contexto de sistemas en desarrollo. Sus fortalezas y debilidades en este entorno.

También es importante conocerlas para analizar cómo pueden aplicarse dentro del modelo propuesto en el siguiente capítulo, ya que en definitiva son las herramientas disponibles para realizar procesos de migración, independientemente de la arquitectura que se utilice.

Los productos para gestión de bases de datos son muchos y diversos. A nivel mundial hay una cantidad muy significativa de fabricantes de motores de bases de datos y de herramientas relacionadas que pueden incorporarse en diferentes procesos. Ante la imposibilidad de revisarlos a todos, en este capítulo se centra el análisis en los productos más populares según diferentes fuentes.

Los diversos fabricantes proveen herramientas útiles para la realización de migraciones de datos de una plataforma a otra. En buena medida esto tiene un motivo comercial ya que posibilita a un usuario de una determinada marca portar sus datos a otra. Esta es una de las principales razones por la que es común encontrar una gran variedad de opciones al momento de migrar bases de datos.

Al existir tanta variedad, concluir cuales son los productos más populares no es tarea simple. El grado de utilización varía por la distribución geográfica, por los tipos y magnitud de los proyectos, por política de utilización de software libre o pago de las diferentes instituciones, etc. El objetivo de este capítulo no es arribar a una conclusión certera de cuáles son los productos y fabricantes más populares, sólo se necesita tener alguna tendencia o apreciación general como para

poder más adelante comparar las metodologías y hacer una implementación de la propuesta sobre una plataforma conocida.

Algunos productos de bases de datos disponibles

Como ya se ha dicho antes, determinar cuáles son las herramientas, productos y fabricantes más populares de motores de bases de datos no es una tarea simple. A continuación, se muestran algunos estudios realizados por compañías especializadas.

En noviembre de 2016 la compañía Austríaca Solid IT, dedicada principalmente a proyectos de Big Data (grandes volúmenes de datos) publicó un análisis de los motores de bases de datos y fabricantes más populares a nivel global (18). El mismo se realizó a partir de lanzar búsquedas de temas relacionados, en forma automatizada, en buscadores como Google, Bing y Yandex. También se analizaron temas relacionados en redes sociales como LinkedIn, Upwork y Twitter. (19)

La figura siguiente muestra los primeros 10 elementos del ranking obtenido por Solid IT y publicados por DB-Engines (18).

Rank			DBMS	Database Model	Score		
Nov 2016	Oct 2016	Nov 2015			Nov 2016	Oct 2016	Nov 2015
1.	1.	1.	Oracle	Relational DBMS	1413.01	-4.09	-67.94
2.	2.	2.	MySQL	Relational DBMS	1373.56	+10.91	+86.71
3.	3.	3.	Microsoft SQL Server	Relational DBMS	1213.80	-0.38	+91.48
4.	5.	5.	PostgreSQL	Relational DBMS	325.82	+7.12	+40.13
5.	4.	4.	MongoDB	Document store	325.48	+6.67	+20.87
6.	6.	6.	DB2	Relational DBMS	181.46	+0.90	-21.07
7.	7.	8.	Cassandra	Wide column store	133.97	-1.09	+1.05
8.	8.	7.	Microsoft Access	Relational DBMS	125.97	+1.30	-14.99
9.	9.	10.	Redis	Key-value store	115.54	+6.00	+13.13
10.	10.	9.	SQLite	Relational DBMS	112.00	+3.43	+8.55

Figura 9: Bases de datos más populares según Solid IT. (18)

Según este estudio, los motores de bases de datos relacionales más populares son Oracle, MySQL, SQL Server, PostgreSQL y DB2. Los demás resultados no son relevantes para el presente análisis ya que se trata de productos para bases de datos no relacionales o de servidores de archivos.

Otro análisis que puede tomarse como referencia es el realizado por la compañía Gartner, consultora en tecnología. Según este estudio, que fue publicado en enero de 2007 por ComputerWeekly (20) el 40.8% del mercado europeo está cubierto por la firma Oracle, el 29.4% por IBM y el 14.9% por Microsoft. Esto constituye más del 80% del mercado total, el resto se reparte entre todos los demás productos. Es importante aclarar que la compañía no ha publicado los mecanismos utilizados para obtener estos resultados.

En 2006 la firma española Marketing Directo publicó un estudio realizado por el IDC (International Data Corporation) según el cual a nivel global Oracle lidera el mercado de los motores de bases de datos ocupando un 44.4% del mismo. La firma IBM se encuentra en segundo lugar con su producto DB2 con un 21.2% y en tercer puesto Microsoft con su producto SQL Server con un 18,6% (21). Cabe destacar que este análisis se ha realizado evaluando niveles de facturación. Si bien es un indicador válido, los valores de licencias de una plataforma y otra son muy diferentes por lo tanto no necesariamente indica que un motor de bases de datos sea más popular o más utilizado que otro, aunque marca alguna tendencia. Otro elemento a tener en cuenta es que al evaluarse niveles de facturación quedan fuera del estudio los motores de bases de datos gratuitos como MySQL y Postgres entre otros.

Según los tres estudios de mercado citados, los motores de bases de datos relacionales más utilizados a nivel mundial son Oracle, DB2, SQL Server y MySql aunque como ya se ha dicho antes, estas conclusiones, si bien hablan de alguna tendencia, son relativas. A continuación, se analizan las herramientas y técnicas propuestas para estos cuatro productos, sus fortalezas y debilidades y la posibilidad de utilizarlas junto al modelo que se propone en el capítulo siguiente.

Metodología Oracle AIM

AIM (Oracle Application Implementation Methodology) es una metodología desarrollada por Oracle para realizar implementaciones de aplicaciones (22). Esta metodología propone una fase para la migración de bases de datos, sus etapas son las siguientes (23):

1. **Business Process Architecture [BP]**
2. **Business Requirement Definition [RD]**
3. **Business Requirement Mapping [BR]**
4. **Application and Technical Architecture [TA]**
5. **Build and Module Design [MD]**
6. **Data Conversion [CV]**
7. **Documentation [DO]**
8. **Business System Testing [TE]**
9. **Performance Testing [PT**
10. **Adoption and Learning [AP]**
11. **Production Migration [PM]**

Figura 10: Etapas AIM (23).

Si bien un análisis completo de la metodología se encuentra fuera de los objetivos de esta tesis, resulta importante conocer como está estructurada para comprender su aplicación en migraciones de datos.

En la primera etapa, se analizan las prácticas de negocio tanto actuales como futuras para luego en la segunda definir los requerimientos. En la tercera etapa, los requerimientos se vinculan (mapean) con las funcionalidades de la aplicación que se pretende poner en marcha. En la cuarta etapa se plantea la definición de la arquitectura técnica mientras que en la quinta se desarrolla la personalización de la aplicación según los requerimientos planteados anteriormente.

La sexta etapa plantea la conversión de datos desde aplicaciones anteriores a la nueva base de datos, ésta resulta particularmente interesante y se analiza en detalle más adelante.

En la séptima etapa se elabora toda la documentación que respalda las decisiones tomadas en las anteriores. En la octava se realizan las pruebas de la aplicación, es una fase completamente orientada al testing que tiene continuidad en la novena etapa donde se realizan pruebas de

rendimiento. En la décima se desarrolla un plan para sacar de línea las aplicaciones a ser reemplazadas, esto incluye planes de entrenamiento, análisis de impacto, realización de eventos de comunicación y entrenamiento entre otras actividades. Finalmente, en la última etapa se diseña una estrategia de transición, se dispone el entorno de producción, la aplicación se instala y es puesta en marcha.

Como puede advertirse la metodología es amplia, puede utilizarse para la puesta en marcha de aplicaciones de todo tipo (no sólo desarrollos de Oracle) y plantea un mecanismo extenso que da cobertura desde un análisis funcional inicial hasta la puesta en producción con el consecuente reemplazo de aplicaciones anteriores, pasando por etapas de conversión y migración de datos.

A los efectos del presente estudio, resulta particularmente interesante la etapa de conversión de datos que está organizada de la siguiente manera (23):

CV.010 Define data conversion requirements and strategy
CV.020 Define Conversion standards
CV.030 Prepare conversion environment
CV.040 Perform conversion data mapping
CV.050 Define manual conversion procedures
CV.060 Design conversion programs
CV.070 Prepare conversion test plans
CV.080 Develop conversion programs
CV.090 Perform conversion unit tests
CV.100 Perform conversion business objects
CV.110 Perform conversion validation tests
CV.120 Install conversion programs
CV.130 Convert and verify data

Figura 11: Etapas de conversión de datos de AIM [22]

La figura siguiente describe brevemente cada una de estas etapas en el proceso de conversión de datos según una publicación realizada por OracleApps Epicenter. (24)

Task	Description	Deliverable outcome
Define Data Conversion Requirements and Strategy	Analyze and document the conversion scope, objectives, approach, requirements, and the strategy for how the conversion will be performed.	Data Conversion Requirements and Strategy
Define Development Standards	Document the standards that should be followed throughout the conversion effort.	Development Standards
Prepare Conversion Environment	Prepare the conversion environment for the development and testing of the conversion programs.	Conversion Environment
Perform Conversion Data Mapping	Map legacy data files and elements to the Oracle Application table(s) and columns. This map includes an explanation of business, translation, foreign key rules, and default settings.	Conversion Data Mapping
Define Manual Conversion Procedures	Define procedures for manually converting applicable business objects through Oracle Application forms.	Manual Conversion Procedures
Design Conversion Programs	Design how the conversion programs should be coded using conventional programming techniques or built using an automated conversion tool.	Conversion Program Designs
Prepare Conversion Test Plan	Specify test procedures to be followed for performing conversion unit, business object, and validation tests.	Conversion Test Plans
Develop Conversion Programs	Develop conversion programs based on the Conversion Program Design that have been coded.	Conversion Programs
Perform Conversion Unit Tests	Test the performance of each of the individual conversion program modules.	Unit-Tested Conversion Programs
Perform Conversion Business Object Tests	Test how the converted data performs within the target application.	Business Object-Tested Conversion Programs
Perform Conversion Validation Tests	Test how the converted data performs within the entire suite of target applications.	Validation-Tested Conversion Programs
Install Conversion Programs	Install the conversion programs that have been coded and tested. If an automated conversion tool is being used, the tool should remain installed until the final conversion is performed	Installed Conversion Programs
Convert and Verify Data	Convert data that has been verified by the users before commencement of production operations.	Converted and Verified Data

Figura 12: Etapas del proceso de conversión de datos Oracle Aim. (24)

Como puede apreciarse, el planteo del proceso es muy amplio y prevé etapas que van desde la definición de estrategias a alto nivel, definición de requerimientos y construcción de estándares hasta la verificación de los datos migrados pasando por procesos de testing y de construcción y ejecución de los programas de migración.

Conclusiones metodología AIM

En general AIM es una metodología muy amplia que ha sido concebida para la realización de implementaciones. Uno de los procesos que la compone es el de conversión y migración de datos que puede utilizarse independientemente del contexto, o sea, que es posible utilizarlo más

allá de que se esté o no dentro de una implementación. En otras palabras, no es aplicable toda la metodología en el contexto de estudio, pero si se puede emplear el proceso de conversión de datos dentro de cualquier escenario de migración.

Como fortalezas del proceso de conversión de AIM pueden destacarse las siguientes:

- Es muy amplia. No abarca sólo aspectos técnicos, sino que da un marco de trabajo para elementos fundamentales y estratégicos que están más allá de los recursos técnicos que luego puedan emplearse. Su especificación va desde la definición de estrategias hasta la propia implementación pasando por varios procesos de testing y validación.
- Es flexible. Cada etapa del proceso es independiente de las demás. Pueden implementarse todas o un subgrupo o llevarse a cabo sólo algunas actividades dentro de una etapa y realizar otras con mayor profundidad. Por ejemplo, en algunos proyectos pueden tener gran importancia los procesos de testing y validación y en otros quizás ser menos relevantes. La metodología se adapta muy bien a diferentes situaciones.
- Es independiente de la tecnología. Si bien es un desarrollo de Oracle, puede utilizarse con cualquier producto y con cualquier conjunto de herramientas.

Como debilidades del proceso de conversión de AIM pueden destacarse las siguientes:

- Es extensa y muy abarcativa. Si lo que se pretende es resolver un problema puntual la metodología puede resultar demasiado extensa. Incluso aplicando sólo algunas de sus etapas, cada una individualmente, puede resultar en una cobertura muy amplia. Se debe tener cuidado de no implementar procesos muy complejos que superen las necesidades del caso particular que se esté abordando.
- Es poco adaptable a procesos de migración en sistemas en desarrollo. Para el caso particular del contexto de estudio de la presente tesis, esta metodología puede resultar costosa de implementar. Tras cada cambio en la base de destino puede ser necesario modificar en mayor o menor medida el proceso de migración. Repetir todos los pasos que define AIM será costoso. En este aspecto se puede decir que resulta poco ágil.
- Ninguna de sus etapas ni su planteo general incorpora un patrón o una recomendación ni ningún otro elemento que permita absorber los cambios frecuentes de los sistemas en

desarrollo que pueden afectar a la migración. El planteo parte de una base de destino estable y no prevé que en ella ocurran cambios que lleven a repetir parte del proceso o realizar iteraciones sobre el mismo.

Metodología DULCIAN

Dulcian es una empresa de los Estados Unidos dedicada al desarrollo de aplicaciones, proyectos de datawarehousing y consultoría en procesos de migración de datos principalmente con tecnología Oracle. En 1998 Joseph R. Hudicka, uno de los directivos de la compañía, desarrolló una metodología de migración de bases de datos que divide el proceso completo en siete etapas. (25) (26).

La metodología Dulcian ha sido pensada especialmente para migrar datos en situaciones de implementación de nuevos sistemas. Como lo dice Hudicka, "Uno podría pensar que ambos sistemas mantienen el mismo tipo de datos y deben estar haciendo cosas muy similares y, por lo tanto, debe mapear de uno a otro con facilidad. Esto casi nunca es así". (25)

Es una metodología sencilla y que, si bien pertenece a una compañía que realiza proyectos principalmente con Oracle, puede implementarse sobre cualquier tecnología.

Está dividida en siete fases que se enuncian a continuación. (25)

- Estrategia: En esta fase, se determina el enfoque del proyecto global. Es importante señalar que los proyectos de migración de datos no ocurren de forma independiente. Generalmente, se generan a partir de otros esfuerzos de desarrollo tales como la implementación de nuevos entornos ya sean OLTP (On Line Transaction Processing – Procesamiento de Transacciones en Línea) y / o OLAP (On Line Analytical Processing – Procesamiento Analítico en Línea).
- Análisis: La fase de análisis de la migración de datos debe programarse para que ocurra simultáneamente con la fase de análisis del proyecto principal. Desafortunadamente, en la mayoría de los casos se espera que las mismas personas realicen ambos análisis. Esto puede

hacerse, pero estas personas necesitan hacerse responsables de un conjunto de tareas claramente definido.

- Construcción: La fase de construcción de la migración coincide con la fase de diseño del proyecto central. La primera tarea de la fase de construcción es la generación de las nuevas estructuras de datos y su creación dentro de la base de datos.
- Testing e Implementación: Dado que estas dos fases son prácticamente inseparables, las pruebas y la implementación se combinan a menudo en una sola fase. Las pruebas se dividen en dos áreas temáticas principales: errores lógicos y errores físicos. Los errores físicos son típicamente de naturaleza sintáctica y pueden ser fácilmente identificados y resueltos. Los errores físicos no tienen nada que ver con la calidad del mapeo, en este caso se trata de errores lógicos.
- Revisión: La fase de revisión, previamente discutida durante la fase de prueba / implementación, es donde se administra la limpieza. Cada modificación del modelo de datos, el ajuste de las reglas de transformación y la modificación de los scripts se combinan esencialmente para formar la fase de revisión.
- Mantenimiento: La fase de mantenimiento es donde todas las asignaciones se validan y se implementan con éxito en una serie de secuencias de comandos que se han probado exhaustivamente. La fase de mantenimiento difiere dependiendo de si está migrando a un OLTP o un sistema OLAP. Si la migración es a un sistema OLTP, se está trabajando dentro del paradigma 'uno y hecho'. La meta es migrar con éxito los datos heredados al nuevo sistema, haciendo que los scripts de migración sean inútiles cuando ya se ha realizado la misma. Si se está migrando a un sistema OLAP, lo más probable es que se vuelva a cargar el nuevo sistema en intervalos oportunos.

Conclusiones metodología DULCIAN

Su aspecto más relevante es que ha sido pensada para ser utilizada en procesos de migraciones que ocurren dentro del contexto de un cambio de sistema, lo cual la hace particularmente interesante para el caso de estudio de esta tesis. Sin embargo, sus etapas, que

corren en paralelo a muchos aspectos de la construcción del nuevo sistema, pueden resultar difíciles de adaptar a los cambios propios del desarrollo.

En resumen, como fortalezas pueden destacarse los siguientes aspectos:

- Está pensada especialmente para ser utilizada en procesos de cambios de sistemas.
- Es sencilla, fácil de comprender y aplicar. En siete fases provee recomendaciones que van desde el diseño de la migración hasta la realización de pruebas y la implementación.
- Puede ser usada tanto en contextos OLTP como OLAP.
- Es independiente de la plataforma. Si bien la compañía DULCIAN se dedica principalmente a proyectos con tecnología Oracle, la metodología es independiente de esto y sus etapas pueden ponerse en práctica en cualquier contexto.

También pueden destacarse las siguientes debilidades:

- En ocasiones, los cambios en el sistema que está en desarrollo pueden ser numerosos y para estos casos la aplicación de la metodología DULCIAN puede resultar compleja. Repetir todas las etapas ante un cambio puede resultar inviable y a la vez delegar esto sólo a la etapa de revisión podría ser poco práctico.
- No define elementos de arquitectura del proceso de migración. Define las etapas, pero no determina la manera en que se debe construir el proceso en sí. En ese aspecto la metodología AIM es más extensa, pero da una cantidad mayor de elementos.
- No en todos los casos resulta posible llevar el proceso de migración en paralelo a determinadas etapas del proceso de desarrollo. A veces la complejidad de una migración es significativamente mayor o menor que la de la construcción e implementación de un nuevo sistema. Los tiempos y la disponibilidad de recursos no siempre posibilitan este paralelismo.
- Ninguna de sus etapas ni su planteo general incorpora un patrón o una recomendación ni ningún otro elemento que permita absorber los cambios frecuentes de los sistemas en desarrollo que pueden afectar a la migración. Si bien existe una etapa de revisión, no se definen mecanismos específicos para implementar las correspondientes modificaciones.

Metodologia Oracle relational migration maps

Esta metodología es un desarrollo de Oracle y está particularmente diseñada para migrar bases de datos de otros orígenes a la plataforma de este fabricante. Abarca también la migración de aplicaciones, por lo que algunos de sus elementos no son directamente aplicables cuando se trata sólo de bases de datos. Es posible decir que abarca estos dos aspectos

- Migración de bases de datos de otros fabricantes a la plataforma Oracle.
- Migración de bases de datos y aplicaciones de otros fabricantes a la plataforma Oracle.

Está basada en seis fases que se emplean para llevar adelante el proyecto. Estas fases son las siguientes: (10).

- **Fase de Definición**: En esta fase se reúne información sobre las aplicaciones y las bases de datos a ser migradas con el objetivo de realizar estimaciones para el resto del proyecto.
- **Fase de Análisis**: Es la que define el verdadero comienzo del proyecto. Durante ella se siguen los lineamientos de la fase anterior y se obtiene la planificación del proyecto.
- **Fase de Diseño**: En esta fase se realiza el diseño del proceso de migración tanto a nivel de la aplicación como de las bases de datos. Como salida se obtiene un plan de migración que expresa la solución para los problemas detectados en las etapas anteriores. En relación a los datos se define cuando comenzar con su migración y las estrategias de transformación y depuración. También se determinan los procesos de prueba y validación.
- **Fase de Migración**: El objetivo de esta fase es la generación de scripts y otras herramientas de transformación de datos que permitan crear los esquemas de destino y realizar el pasaje de datos y de los demás elementos de la base como ser procedimientos, funciones, secuencias, etc. Se utiliza una base de testing para poder realizar pruebas y hacer las modificaciones y ajustes necesarios.
- **Fase de Transición**: Esta etapa incluye la puesta en marcha del nuevo sistema sobre los datos migrados. También se ponen en marcha todos los demás elementos de la base de datos. Se realizan las pruebas desde la óptica del cliente de la aplicación recorriendo los casos de prueba desarrollados en las etapas anteriores.

- **Fase de Producción**: Aquí se realizan actividades relacionadas con el soporte posterior a la migración. Se resuelven problemas que pueden haber quedado pendientes y se refina la aplicación y la base de datos.

La metodología prevé un conjunto de tareas que pueden realizarse durante la ejecución de cada fase. Estas tareas son:

- Entrenamiento.
- Evaluación del Sistema Existente.
- Arquitectura técnica.
- Migración de la Base de Datos.
- Migración de la Aplicación.
- Migración de los Datos.
- Pruebas.
- Transición.
- Administración del Proyecto.

La figura siguiente muestra la relación entre las tareas y cada una de las etapas. En otras palabras, indica cuales tareas son incluidas en cada fase. (10)

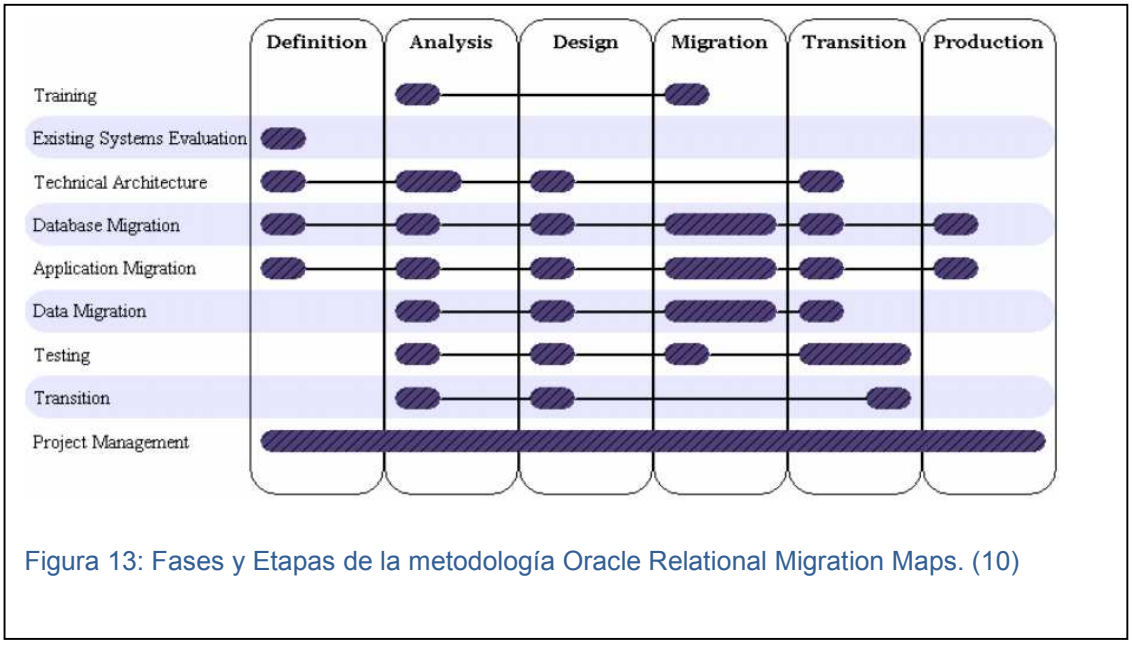


Figura 13: Fases y Etapas de la metodología Oracle Relational Migration Maps. (10)

Como puede observarse, la tarea de Project Management está presente en todo el proceso.

La tarea de migración de la base de datos comienza en la fase de definición, pero no así la de migración de datos, esto se debe a que la metodología prevé una primera etapa donde se analizan las estructuras con fines estratégicos y de planificación.

Se debe notar también que, en la fase de producción, además de la tarea de Project Management, sólo se encuentran Database Migration y Aplication Migration. Hay que tener en cuenta que la metodología está diseñada para migrar tanto bases de datos como aplicaciones. En producción se da soporte a la aplicación y a elementos estructurales de la base, no a los datos en sí ya que debieron migrarse en las fases anteriores.

Conclusiones metodología Oracle relational migration maps

Su característica más relevante es que ha sido creada para migrar tanto aplicaciones como bases de datos desde cualquier plataforma hacia la tecnología Oracle. Por esta razón contiene fases y tareas que no necesariamente guardan relación con las bases de datos en sí.

La tarea de Project Management está presente durante todo el proceso, este es un aspecto positivo sobre todo en casos complejos. Como se ha mencionado en capítulos anteriores, muchas migraciones atraviesan dificultades serias o incluso fracasan por falta de planificación y de organización de recursos.

Pueden destacarse las siguientes fortalezas:

- Comienza con una fase introductoria y de planificación estratégica donde se hace foco en las estructuras no con fines técnicos sino con el objetivo de realizar un análisis preliminar que sirva para marcar el rumbo general del proyecto.
- Plantea una serie de fases sencilla y con pocas etapas. Otras metodologías, como AIM por ejemplo, son mucho más complejas.
- Contiene una tarea de testing con fuerte presencia en casi todo el proceso de migración. Define la realización de pruebas y verificaciones en varias fases.

- Como ya se mencionó antes, la tarea de project management está presente en todas las fases incluso en la de producción cuando la nueva aplicación y base de datos ya se encuentran funcionando.

También es posible observar estas debilidades:

- Como está creada para migrar aplicaciones y bases desde otros orígenes hacia la plataforma Oracle, se basa fuertemente en la utilización de herramientas de este fabricante. Sería necesario hacer algunas adaptaciones para poder usarla en otro contexto.
- Está pensada para migrar tanto aplicaciones como bases de datos, por lo tanto, en el caso de que se trate sólo de bases de datos, no todas sus tareas son aplicables y también sería necesario realizar algunas adaptaciones para poder emplearla.
- La fase de migración prevé un fuerte uso de scripts para llevar adelante el proceso de transformación de datos. Se debe tener en cuenta que, si bien éste es un muy buen recurso, existen otras herramientas que requieren un esfuerzo de programación menor y que pueden resultar de gran ayuda.
- Al igual que en los casos anteriores, su planteo es etapa por etapa. Si aparecen cambios significativos en la base de destino pueden tener un alto impacto en el proceso de migración
- Ninguna de sus etapas ni su planteo general incorpora un patrón o una recomendación ni ningún otro elemento que permita absorber los cambios frecuentes de los sistemas en desarrollo que pueden afectar a la migración. El planteo parte de una base de destino estable y no prevé que en ella ocurran cambios que lleven a repetir parte del proceso o realizar iteraciones sobre el mismo.

Softek – IBM Data migration methodology

Softek es una empresa con gran experiencia en movilidad de datos. Junto a IBM han desarrollado una metodología que apunta a resolver los principales problemas que atraviesan con frecuencia este tipo de procesos.

La metodología hace fuerte foco en la planificación, siendo ésta su primera etapa. Luego plantea una etapa de migración y por último una de validación.

La figura siguiente muestra cómo está organizada. (27)

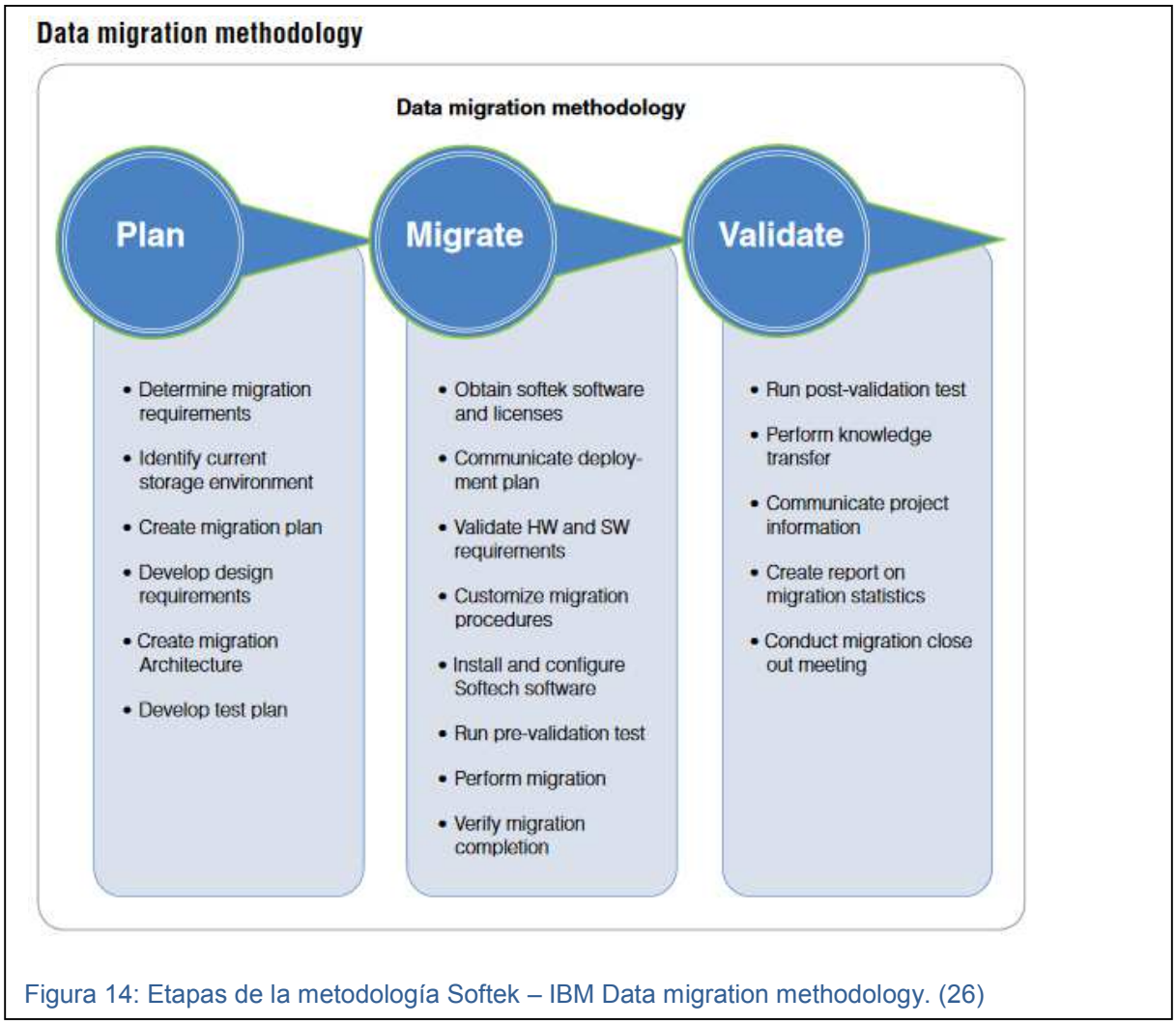


Figura 14: Etapas de la metodología Softek – IBM Data migration methodology. (26)

Como muestra esta figura, dentro de la fase de planeamiento se detectan los requerimientos de la migración, se analiza el entorno de almacenamiento actual y se diseña la arquitectura de la migración entre otras tareas. Lo más relevante es que aquí se construye tanto el plan de migración como el plan de testing.

En la segunda fase se debe obtener el licenciamiento y realizar la instalación de las herramientas de software utilizadas para la migración, previamente validar que se satisfacen los

requisitos de software y hardware para éstas. Desarrollar los procedimientos de migración utilizando estos recursos, ejecutarlos y verificarlos.

En la fase de validación se ejecutan los test y se reportan los resultados.

Conclusiones metodología Softek – IBM Data migration methodology

Si bien se trata de una metodología que puede extenderse a otras plataformas, su característica más relevante es que ha sido desarrollado por la firma Softek y se basa fuertemente en el uso de sus herramientas. Incluso el primer paso de la fase de migración es "obtener el software Softek y sus licencias".

Cuenta con pocas fases y en realidad la mayor parte del trabajo que demanda la migración se realiza en la segunda. La primera es de planificación y la tercera es de testing. El éxito o fracaso dependerá fuertemente de la utilización de las herramientas que provee Softek y de cuanto éstas se adapten al contexto en el que ocurre el proyecto.

Pueden detectarse las siguientes fortalezas:

- Para casos en los que el equipo de migración ya cuenta con experiencia en el uso de las herramientas de Softek, puede resultar muy conveniente su utilización ya que se basa fuertemente en ellas.
- El uso de un solo conjunto de herramientas acotado simplifica la comunicación y las definiciones de arquitectura como así también la capacitación de los miembros del equipo.
- En la primera etapa se diseña la arquitectura de la migración. Este es un análisis muy conveniente que no está previsto en otras metodologías.

También se pueden observar las siguientes debilidades:

- Como metodología guarda una estrecha relación con herramientas de un determinado fabricante. No puede emplearse en casos donde se pretenda usar otra tecnología. En este sentido es poco adaptable.

- Su estructura está basada en tres fases, pero en realidad la migración en sí ocurre en una sola. En migraciones complejas esto puede resultar poco conveniente.
- Más allá de la utilización de determinadas herramientas, como metodología hace un planteo superficial y divide a todo el proceso en pocas etapas con alta interdependencia entre ellas.
- Ninguna de sus etapas ni su planteo general incorpora un patrón o una recomendación ni ningún otro elemento que permita absorber los cambios frecuentes de los sistemas en desarrollo que pueden afectar a la migración. El planteo parte de una base de destino estable y no prevé que en ella ocurran cambios que lleven a repetir parte del proceso o realizar iteraciones sobre el mismo.

Microsoft Data Migration Methodology

Otro planteo metodológico que puede tenerse en cuenta es el que realiza la firma Microsoft (28). En general la propuesta está desarrollada para realizar migraciones de datos desde otros fabricantes a SQL Server. Hace un fuerte uso de herramientas de Microsoft y por ende es poco aplicable a escenarios donde la base de datos de destino es de otro fabricante.

Está basada en cuatro pasos tal como muestra la figura siguiente.

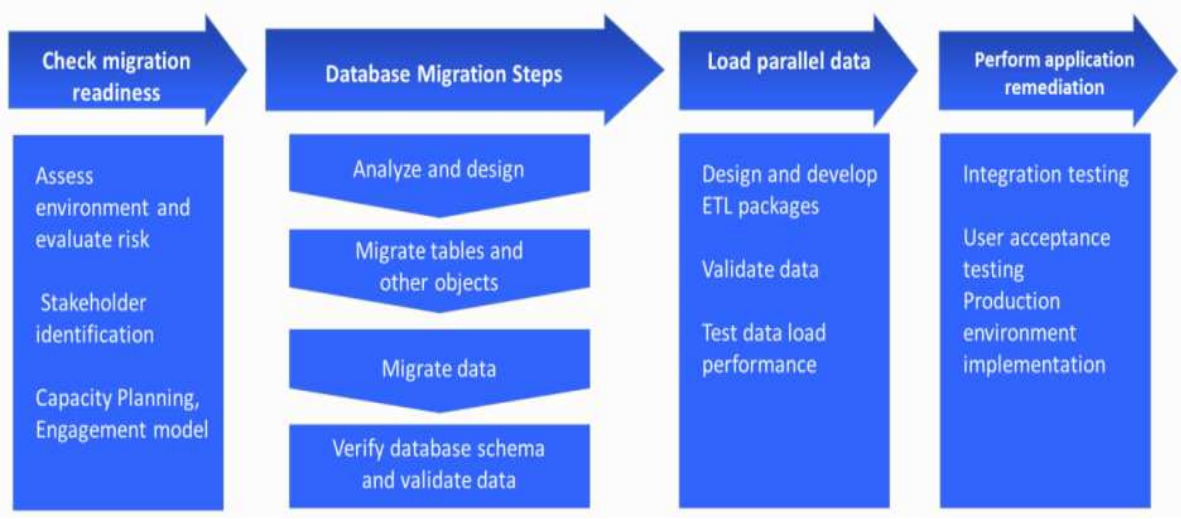


Figura 15: Metodología de Microsoft (28)

La propuesta de Microsoft se basa en cuatro etapas.

La primera etapa es una fase de preparación. En ella se analiza el entorno y se evalúan los riesgos, se definen los roles de los miembros del equipo y se obtiene el plan de migración.

La segunda etapa es la migración propiamente dicha y está compuesta de cuatro pasos fundamentales:

- Realización del análisis y diseño del proceso de migración.
- Migración de estructuras. Es decir, tablas sin registros. Migración de elementos de programación que puedan estar contenidos en la base como por ejemplo procedimientos almacenados, triggers, vistas, funciones, etc. Migración de cualquier otro objeto necesario que no contenga datos.
- Migración de datos. Aquí es donde se realiza la transformación y traslado de la información. Se pueblan con registros las estructuras contempladas en el paso anterior.
- Verificación. Se ejecutan los procedimientos definidos para validar los datos y verificar el correcto funcionamiento de todo el esquema.

La tercera etapa apunta a detectar si en la base origen de la migración existen procesos que toman información desde otras bases de datos. De ser así, deben adaptarse y validarse para que su salida impacte sobre la nueva base. En otras palabras, si en la base de datos de origen existen ETLs (procesos de lectura y transformación de datos) también deben tenerse en cuenta en la nueva estructura.

La última es una etapa de testing y soporte. Deben probarse las aplicaciones que utilizan la base de datos nueva y realizar test de aceptación. También realizar capacitaciones si corresponde. Montar el entorno de producción y dar soporte post producción.

Conclusiones metodología de Microsoft

Es una metodología simple, de pocos pasos que no requiere elementos complejos para su implementación. Como ya se ha dicho antes está muy fuertemente asociada a herramientas del

fabricante que la propone lo cual la hace poco aplicable en procesos que utilicen otras tecnologías. Algo muy particular es que prevé una etapa de adaptación de procesos ETLs, algo que muchas veces no es tenido en cuenta y que puede ser un elemento muy relevante.

En resumen, pueden detectarse las siguientes fortalezas:

- Su planteo es simple, fácil de comprender e implementar.
- Tiene una etapa completa destinada a la planificación, al análisis del entorno y al armado de un plan de migración.
- Prevé pasos específicos para la adaptación de procesos ETL.
- Define pasos específicos para realizar pruebas y validaciones en la base de datos de destino.
- Tiene una etapa completa dedicada al testing pero no sólo desde el punto de vista de la base de datos sino desde la utilización de las aplicaciones que leen y escriben en dicha base.

También pueden mencionarse las siguientes debilidades:

- Toda su definición se encuentra muy estrechamente relacionada con herramientas de Microsoft. Por ejemplo, la utilización de SSMA (Sql Server Migration Assistant) como elemento técnico principal. Esto la hace poco adaptable a otros escenarios.
- Realiza definiciones de etapas muy generales. Tiene una óptica muy abstracta del proceso de migración en sí y entonces su implementación en situaciones concretas puede contener elementos no contemplados. Es una metodología general, no contiene pasos detallados.
- La segunda etapa es el proceso de migración en sí. Prácticamente todos los elementos propios del proceso, se encuentran es esta única fase por lo que su implementación puede resultar extensa en casos complejos.
- Ninguna de sus etapas ni su planteo general incorpora un patrón o una recomendación ni ningún otro elemento que permita absorber los cambios frecuentes de los sistemas en desarrollo que pueden afectar a la migración. El planteo parte de una base de destino estable y no prevé que en ella ocurran cambios que lleven a repetir parte del proceso o realizar iteraciones sobre el mismo.

Herramientas para ejecución de migraciones

Más allá de los diferentes planteos metodológicos realizados por diversos fabricantes y compañías especializadas, existen también muchas herramientas que pueden utilizarse en diferentes etapas de un proceso de migración.

Algunas de estas herramientas plantean un nivel de automatización muy elevado y permiten con poco o ningún esfuerzo de programación, y haciendo uso sólo de interfaces gráficas, migrar datos entre plataformas diferentes.

Para los casos en los que se requiera un mayor control en la lógica empleada para las conversiones existen otros recursos como los lenguajes de programación de las diferentes bases de datos que permiten escribir procesos con alto nivel de detalle, aunque con mayor esfuerzo.

Se analiza a continuación un grupo de herramientas de diferentes fabricantes que pueden emplearse en distintas etapas de las metodologías antes mencionadas. Más adelante, en los capítulos siguientes, se sitúan algunas de ellas en la metodología propuesta.

Oracle SQL Developer Migration WorkBench

Oracle Migration WorkBench es una herramienta desarrollada para migrar bases de datos desde diferentes orígenes hacia una plataforma Oracle. (29)

Este software puede transferir hacia una base Oracle no sólo datos con determinadas conversiones sino también estructuras (tablas, índices, secuencias, etc) como así también elementos de programación como vistas, funciones, triggers y procedimientos almacenados.

Provee funciones que permiten comparar y traducir código fuente residente en la base de datos desde otros lenguajes a PL SQL. Por ejemplo, puede convertir código Transact SQL a código PL SQL con ninguna o muy poca intervención humana.

Toda su interfaz de usuario es gráfica y resulta muy intuitiva a la vez que permite integración con SQL Developer, una de las herramientas de programación y administración de bases de datos Oracle más populares.

La siguiente figura muestra las diferentes fuentes de origen soportadas por esta herramienta, siempre teniendo como destino una base de datos Oracle.

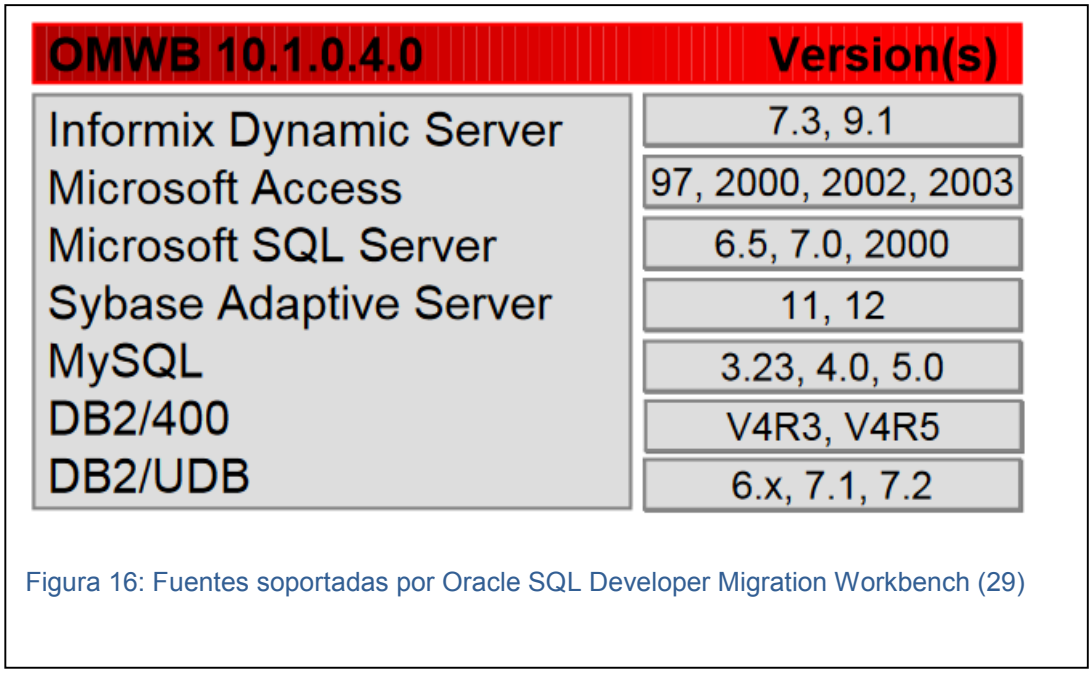


Figura 16: Fuentes soportadas por Oracle SQL Developer Migration Workbench (29)

Si bien es una herramienta muy potente y con un buen nivel de automatización ya que permite transferir objetos de muchos tipos distintos de manera totalmente gráfica y asistida, también hay que destacar que resulta difícil resolver con ella diferencias semánticas. En otras palabras, resulta sencillo transportar un objeto de una tecnología a otra, pero cuando hay diferencias estructurales significativas la situación ya no es tan simple.

La figura siguiente muestra los pasos que esta herramienta emplea para realizar una migración (10):

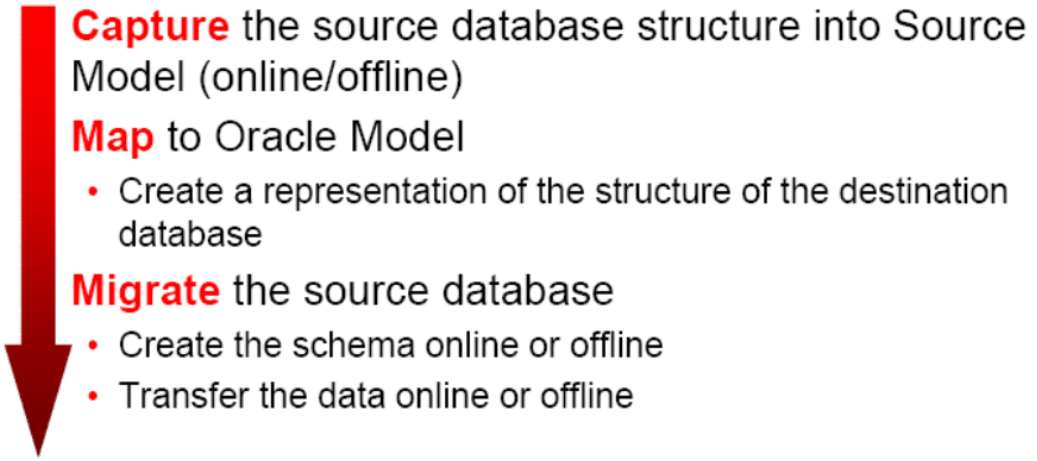


Figura 17: Proceso de migración Oracle SQL Developer Migration Workbench (10)

Como ya se ha dicho anteriormente, cuando un sistema reemplaza a otro, no sólo se deben esperar diferencias técnicas en sus bases de datos, por el contrario, las más significativas tienen relación con las nuevas funcionalidades que se incorporan. Esto lleva a grandes diferencias estructurales y este problema no puede resolverse sólo con el uso de esta herramienta gráfica ya que no se trata sólo de realizar un "mapeo" entre entidades de distinta tecnología.

Un aspecto también relevante es que esta herramienta sólo puede usarse si el origen de datos está comprendido dentro del grupo de los que puede interpretar y si el destino es únicamente una base de datos Oracle.

MySQL Workbench: Database Migration

Es una herramienta también desarrollada por Oracle y que guarda muchas similitudes con la analizada en los párrafos anteriores. En general permite diseñar, construir y administrar bases de datos MySql y tiene un módulo específico para realizar migraciones. (30)

Es fuertemente gráfica y provee un mecanismo muy asistido para migrar diferentes objetos de bases de datos de distintos fabricantes siempre que se tenga como destino MySql.

No sólo permite ejecutar un proceso de migración, sino que posibilita diseñarlo, editarlo y modificarlo y también planificar tareas. (31)

Pueden aplicarse sobre esta herramienta las mismas conclusiones que en el caso anterior. Si bien es ágil, gráfica y potente, no resultará suficiente como único elemento para resolver diferencias semánticas. Puede realizarse un buen "mapeo" gráfico de entidades y transportarlas de su origen a un destino MySql, pero no pueden resolverse de manera automatizada todas las diferencias estructurales.

SSMA – Sql Server Migration Assistant

SSMA (Sql Server Migration Assistant) es una herramienta semi-automática desarrollada por Microsoft para gestionar procesos de migración.

Puede tomar como entrada distintos orígenes y migra a SqlServer los datos transformando automáticamente sus tipos. También puede transferir funciones, procedimientos almacenados, triggers y otros objetos. Es posible traducir automáticamente código Pl-SQL a T-SQL. (32)

Luego de realizar el proceso de transformación, la propia herramienta puede proveer informes sobre la cantidad de objetos que ha migrado con éxito y un detalle sobre los que han fallado para que se tomen acciones especiales sobre ellos.

Una característica relevante es que pueden personalizarse los mecanismos de mapeo entre tipos de datos. La herramienta provee un modo de conversión de tipos desde los diferentes orígenes a SQL Server, pero éste puede modificarse para una migración en particular de manera que afecte tanto a todos los objetos como a alguno en particular.

SSMA prevé la realización de una etapa de testing y para ello permite definir casos de pruebas que pueden almacenarse y ejecutarse luego de generar la base de datos de destino. Las pruebas pueden involucrar diversos objetos, por ejemplo, si se han migrado procedimientos almacenados, pueden programarse pruebas para ejecutarlos con una determinada combinación de parámetros y verificar si se obtienen las salidas esperadas.

Otro aspecto relevante es que, si bien se trata de un recurso muy flexible, que puede tomar como origen bases de datos de distintos fabricantes, sólo puede utilizarse cuando el destino es SQL Server.

A modo de conclusión, ocurre algo similar que con las herramientas analizadas anteriormente. El hecho de poder resolver los complejos aspectos técnicos con la ayuda de una herramienta avanzada es sin duda de mucha utilidad y es un factor que hace un gran aporte a todo el proceso. Transformar de manera automatizada estructuras de datos, tipos de datos, elementos de programación como triggers, funciones, procedimientos, etc. es de muchísima utilidad.

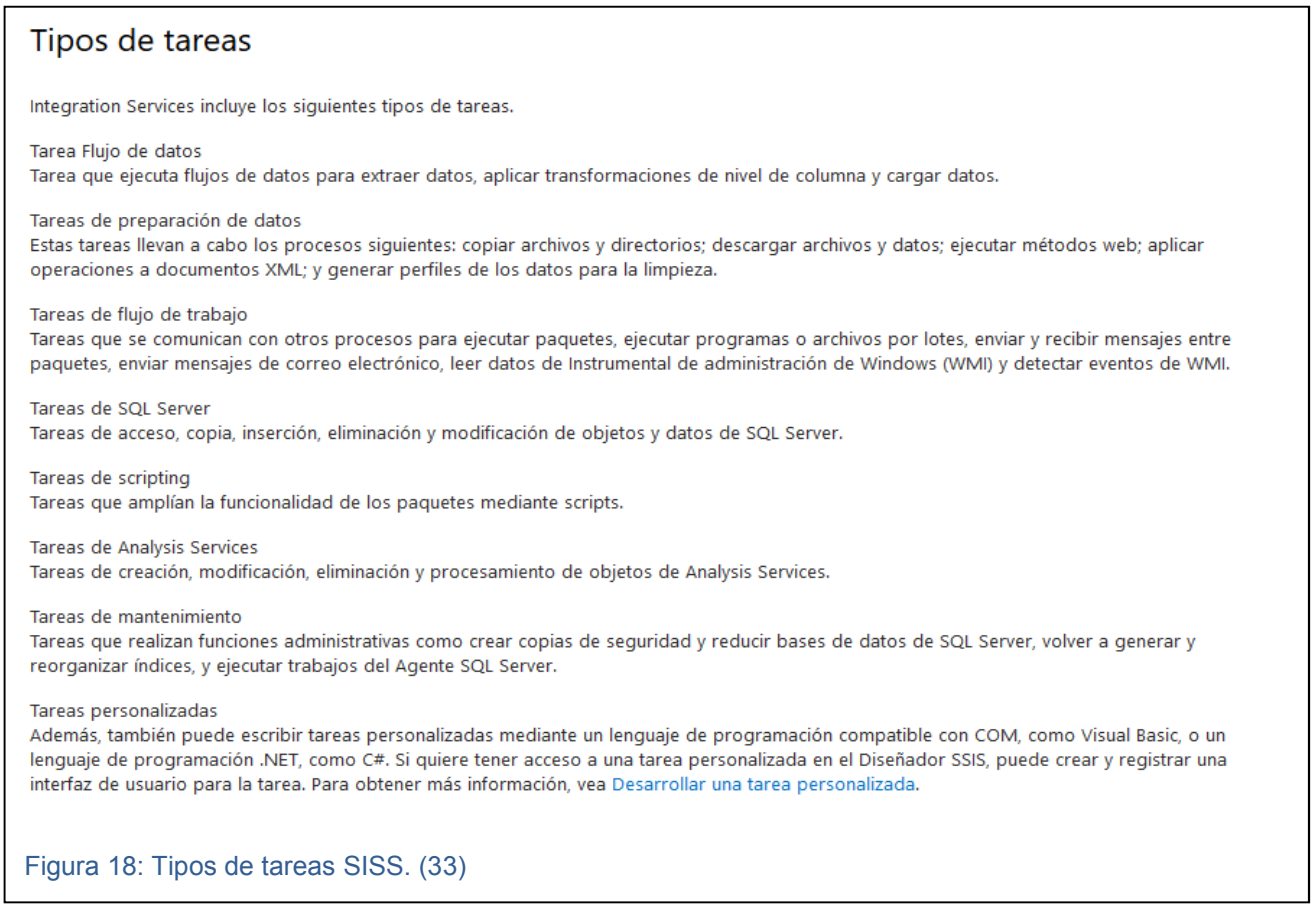
También se constituye un gran recurso poder programar pruebas y ejecutarlas de manera ágil y rápida obteniendo informes de los resultados para tomar las acciones correctivas en la construcción del proceso.

La resolución de las diferencias semánticas entre las bases de origen y la de destino queda alcanzada parcialmente por esta herramienta. Si bien muchos de sus elementos pueden ser de utilidad, en el caso en que un sistema reemplace a otro, la mayor dificultad seguirá siendo incorporar en la nueva base de datos información proveniente de otras anteriores que fueron generadas como respuesta a una lógica de negocio diferente. En este sentido, SSMA puede ser de gran ayuda, pero su utilización no resuelve por sí sola el aspecto más complejo de una migración.

Sql Server Integration Services (SISS)

Se denomina SISS a los servicios que provee Sql Server para integrar datos de diferentes orígenes. Normalmente esta integración consiste en leerlos, transformarlos y guardarlos en una base de datos Sql Server. Su funcionamiento se basa en la creación de paquetes que son conjuntos de tareas que pueden desempeñar diferentes funciones.

La figura siguiente muestra una clasificación y una descripción breve de cada tipo de tarea. (33)



Tipos de tareas

Integration Services incluye los siguientes tipos de tareas.

Tarea Flujo de datos
Tarea que ejecuta flujos de datos para extraer datos, aplicar transformaciones de nivel de columna y cargar datos.

Tareas de preparación de datos
Estas tareas llevan a cabo los procesos siguientes: copiar archivos y directorios; descargar archivos y datos; ejecutar métodos web; aplicar operaciones a documentos XML; y generar perfiles de los datos para la limpieza.

Tareas de flujo de trabajo
Tareas que se comunican con otros procesos para ejecutar paquetes, ejecutar programas o archivos por lotes, enviar y recibir mensajes entre paquetes, enviar mensajes de correo electrónico, leer datos de Instrumental de administración de Windows (WMI) y detectar eventos de WMI.

Tareas de SQL Server
Tareas de acceso, copia, inserción, eliminación y modificación de objetos y datos de SQL Server.

Tareas de scripting
Tareas que amplían la funcionalidad de los paquetes mediante scripts.

Tareas de Analysis Services
Tareas de creación, modificación, eliminación y procesamiento de objetos de Analysis Services.

Tareas de mantenimiento
Tareas que realizan funciones administrativas como crear copias de seguridad y reducir bases de datos de SQL Server, volver a generar y reorganizar índices, y ejecutar trabajos del Agente SQL Server.

Tareas personalizadas
Además, también puede escribir tareas personalizadas mediante un lenguaje de programación compatible con COM, como Visual Basic, o un lenguaje de programación .NET, como C#. Si quiere tener acceso a una tarea personalizada en el Diseñador SSIS, puede crear y registrar una interfaz de usuario para la tarea. Para obtener más información, vea Desarrollar una tarea personalizada.

Figura 18: Tipos de tareas SISS. (33)

Una característica importante es que resulta factible incorporar, como parte de un paquete, programas escritos en C# o en VB net lo que brinda un elemento de gran flexibilidad para desarrollar operaciones específicas.

Una vez definidas todas las tareas que forman parte del paquete, éstas pueden organizarse de manera gráfica según un flujo de control que expresa el proceso general y posteriormente ejecutarse.

Con este conjunto de tareas, SISS provee un completo mecanismo para integrar datos no sólo en el ámbito de una migración sino en procesos de datawarehouse o datamining. La posibilidad de construir cada paso del proceso intercalando recursos gráficos y de programación tanto en T-SQL como en C# o VB net hace de esta herramienta un muy buen recurso.

También cabe concluir que, al igual que las analizadas anteriormente, una migración compleja no podrá ser resuelta sólo por el empleo de módulos SISS. Particularmente en el contexto

de un sistema en desarrollo puede resultar costoso volver a configurar un paquete de migración y modificar tareas si se realizan cambios en la base de datos de destino.

Softek Transparent Data Migration Facility (TDMF)

TDMF es una herramienta desarrollada por Softek para IBM pero es multi-plataforma, es decir, puede ser empleada para tomar información desde múltiples orígenes y puede generar bases de datos de distintas tecnologías no sólo DB2 de IBM.

Tiene como principal objetivo facilitar los procesos de migración sin interrumpir la operatoria de los sistemas que utilizan las bases de origen.

Permite equilibrar las operaciones de entrada salida para que los procesos de lectura y escritura de datos durante la migración no afecten la operatoria normal de los sistemas transaccionales que utilizan las bases de datos involucradas. Ésta es su característica más distintiva y está relacionada con la razón de ser del producto.

También facilita la realización de validaciones durante la ejecución del proceso para evitar la corrupción de datos y garantizar una correcta transferencia desde los orígenes al destino.

La figura siguiente muestra sus principales características. (34)

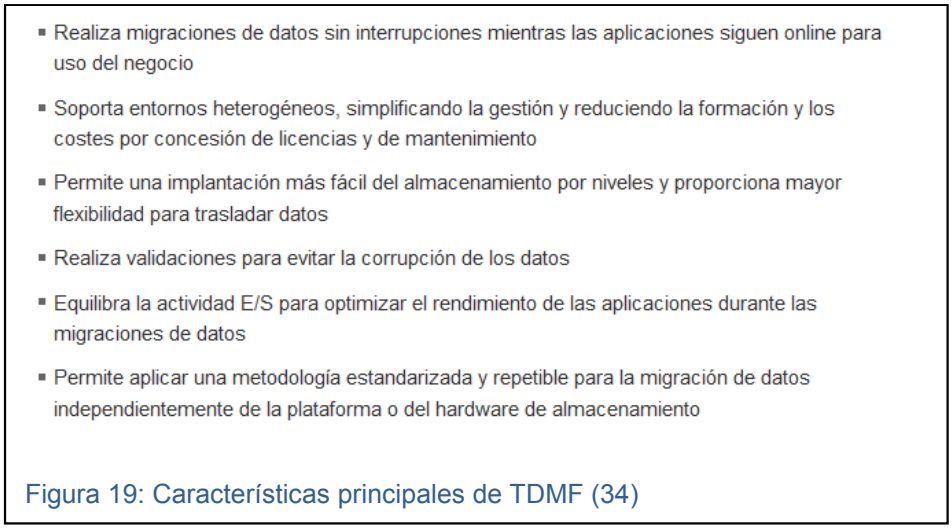


Figura 19: Características principales de TDMF (34)

Nuevamente se puede concluir que la herramienta resulta de gran ayuda en procesos de migración y que sus características facilitan el mismo sin interrumpir el funcionamiento de otras

aplicaciones, pero su utilización por sí sola no garantiza una migración exitosa. El empleo de este recurso debe ocurrir en el contexto de una metodología apropiada.

Otras herramientas

Como es de esperarse, existen tantas herramientas y recursos técnicos como fabricantes de productos de bases de datos puedan encontrarse, o incluso más, porque muchos de ellos tienen varias propuestas.

Así por ejemplo puede mencionarse a Softek Logical Data Migration Facility (LDMF™), los servicios de transformación de datos (DTS) de Sql Server, en general todas las herramientas de ETL propuestas por los diferentes fabricantes ya que, si bien están pensadas para procesos de datawarehousing o datamining, pueden emplearse para migrar datos y muchas otras herramientas más.

Incluso los lenguajes de programación que respetan el estándar SQL presentes en casi todos los fabricantes, constituyen una herramienta formidable porque permiten leer datos de un origen, transformarlos y guardarlos en un destino con lo cual puede llevarse adelante todo un proceso de migración, aunque hay que destacar que es laborioso utilizarlos como único recuso.

También se pueden emplear numerosas herramientas para realizar tareas de testing aunque en general no son específicas para procesos de migraciones de datos. En otras palabras, los recursos que pueden utilizarse para definir casos de prueba, ejecutarlos, analizar y registrar resultados son los mismos que están disponibles para proyectos de desarrollo de cualquier tipo.

Con las herramientas de gestión del proyecto ocurre algo similar. Existen diversos recursos que pueden emplearse para realizar planificaciones, definir roles, asignar tareas, controlar el avance, comparar lo planeado con lo realizado, etc. Dichas herramientas no son específicas para migraciones, sino que pueden emplearse en cualquier proyecto.

Por esta razón, tanto los recursos para realización de testing como para gestión del proyecto no se han incluido en el análisis realizado en el presente capítulo. Son elementos independientes de las características particulares del proceso de migración.

En términos generales todas las herramientas son de utilidad. Naturalmente algunas se adaptan más a determinados contextos, pero lo importante es concluir que ninguna por sí misma resuelve completamente y de manera satisfactoria un proceso de migración. La elección de determinado grupo de herramientas tiene relación con la tecnología empleada en las bases de origen y en las de destino, con el conocimiento del equipo de trabajo, con los costos, con la disponibilidad técnica, etc. pero en ningún caso debe esperarse que un elemento gráfico o de programación resuelva por sí mismo el complejo proceso de migración.

Comparación de las metodologías y herramientas expuestas

En los párrafos anteriores se han analizado varias metodologías propuestas por distintas compañías especializadas en procesos de migración de bases de datos. También se han analizado distintas herramientas como principales recursos técnicos para desarrollar las tareas involucradas en el proceso.

Ahora caben algunas preguntas importantes. ¿Qué aspectos comunes tienen esas metodologías? ¿Cuán utilizables son en el contexto de migraciones que ocurren en el marco del desarrollo de un nuevo sistema? ¿Cuál es el verdadero aporte de las herramientas en el mismo contexto?

La siguiente matriz de comparación expone las características más relevantes, dentro del contexto de estudio, de las metodologías analizadas.

Metodología	Tareas de organización y planificación	Nivel de detalle	Complejidad	Tareas de pruebas y validación de datos	Dependencia de la plataforma	Adaptabilidad a los cambios
Oracle AIM	La metodología es muy completa. En particular la etapa de migración de datos. Cuenta con etapas de planificación y definiciones de estrategia.	Plantea la totalidad del proceso con mucho detalle. Sólo la etapa de migración está dividida en doce pasos.	Complejidad alta. El planteo es muy detallado y la metodología abarca todo el proceso de cambio de una aplicación.	En la etapa de migración de datos propone pasos específicos para la realización de test unitarios, pruebas de rendimiento de los procesos de migración y verificación de los datos de salida.	Si bien es un desarrollo de Oracle, su planteo es genérico y puede usarse en cualquier plataforma.	Si durante alguna etapa del proceso cambian los objetivos y la base de destino de la migración sufre modificaciones, repetir todos sus pasos resulta costoso. La metodología no define acciones específicas para cambios en la base de salida y en consecuencia resulta poco adaptable.
DULCIAN	Plantea dos etapas previas a la construcción del proceso. Una de definición de estrategias y otra de análisis.	Está dividida en siete pasos que van desde la definición de una estrategia previa hasta la realización de mantenimiento posterior a la migración.	Su división en siete pasos es simple. No es una metodología compleja.	La etapa de testing está planteada junto con la implementación de la migración. Si bien propone tareas de pruebas éstas no forman parte de una etapa independiente. Luego define una fase de revisión en la que pueden realizarse correcciones.	Si bien la compañía DULCIAN trabaja principalmente con plataforma Oracle la metodología puede emplearse con otros motores de bases de datos.	En sus siete pasos no plantea estrategias para resolver cambios en la base de destino. La etapa de revisión es la anteúltima y repetir todos los pasos previos ante cada cambio en la base de destino puede resultar costoso. La metodología resulta poco adaptable.
Oracle RMM	En su definición de seis etapas destina las tres primeras a tareas de definiciones previas, análisis y planificación. En este sentido, la metodología hace un fuerte foco en la planificación y organización del proceso.	Tiene un planteo matricial. Propone seis etapas cada una dividida en nueve tareas. Si bien no todas las tareas están presentes en todas las etapas, el planteo es muy detallado y preciso.	La división en seis etapas no resulta compleja. La implementación de cada tarea dentro de las etapas puede variar según las necesidades de cada proyecto.	Da buena cobertura a la realización de pruebas. La tarea de testing está presente en cuatro de las seis etapas y comienza desde el análisis de la migración.	Esta especialmente diseñada para migrar bases de datos y aplicaciones desde otras tecnologías hacia la plataforma Oracle. En este aspecto es dependiente de la plataforma.	Si en la base de destino ocurren cambios frecuentes, volver a repetir sus seis etapas y cada una de las tareas que involucran, puede resultar lento y costoso. La metodología resulta poco adaptable.

IBM Data Migration Methodology	Plantea una primer etapa de planificación en la que se determinan los requerimien-tos de la migración y se arma el plan general.	Sólo establece tres etapas, la primera dedicada a la planifica-ción, la última a la realización de pruebas y todo el proceso de migración propia-mente dicho queda en una sola etapa. En este sentido resulta poco detallada y da una definición muy general.	Su planteo en tres etapas es muy simple. Cada implementa-ción requerirá interpretar y ajustar cada fase.	De las tres etapas que define, dedica una de ellas a la realización de pruebas y dentro de la fase de migración también contempla actividades de validación de datos.	Es dependien-te de la plataforma. Su plateo incluye la utilización de herramientas de la firma Softec.	Ninguna de las herramientas que propone utilizar ni el planteo general de la metodología prevén hacer frente a cambios frecuentes en los objetivos y en la estructura de la base de salida. Resulta poco adaptable a los cambios.
Microsoft Data Migration Methodology	Prevé una primera etapa de planificación y de preparación.	Abarca todo el proceso aunque sólo definiendo cuatro etapas. Es poco detallada.	Su estructura es relativamente simple. Define sólo cuatro etapas.	No define una etapa específica para la realización de pruebas pero si prevé pasos con verificaciones y validaciones de los datos y de las estructuras en diferentes fases.	Es dependien-te de la plataforma ya que su planteo propone la utilización de herramienas de Microsoft.	Ninguno de los elementos que propone apunta a la realización de modificaciones en la base de salida. En este sentido es poco adaptable a los cambios.

Sin duda podrían compararse otras características entre las diferentes metodologías, pero la matriz anterior analiza seis elementos que resultan particularmente importantes en el contexto de estudio.

La coincidencia más relevante es la dificultad de las diferentes metodologías para adaptarse a los cambios. Todas están planteadas en etapas que ocurren una después de otra y en ningún caso contemplan la posibilidad de tener que repetir algunos de sus pasos.

Esto es natural si se tiene en cuenta que en general todas parten de la necesidad de migrar datos desde una base en uso (y por ende estable en su estructura) a otra que también resulta estable. Eso es justamente lo que no ocurre cuando la base de destino responde a la lógica de un sistema en desarrollo.

También es notable que en todos los casos la propia transformación de los datos, es decir, el verdadero núcleo del problema es visto como un solo paso. Algunas metodologías proponen doce etapas y otras sólo tres, algunas hacen un fuerte foco en la planificación y otras no, algunas dan mayor o menor importancia a las actividades de prueba y validación, pero todas enfrentan la propia transformación de los datos como un solo paso cuando en realidad se trata de resolver dos problemas. Uno sintáctico (conversiones de tipos de datos, ajuste de longitudes de campos, usos de distintos formatos para campos de fecha y hora, etc.) y otro semántico que tiene que ver con las verdaderas diferencias estructurales entre las bases de datos de origen y la de destino como consecuencia de que ésta última responde a un nuevo sistema con otra lógica de negocio en sus procesos.

Otro aspecto relevante es la relación entre la complejidad y la cobertura del proceso. En general mientras más detallada es la estructura de trabajo propuesta, mayor cobertura se da a todo el proceso, pero también resulta mayor la complejidad. Las metodologías como AIM que hacen un planteo muy detallado dan buena cobertura a todas las etapas de la migración, pero a la vez pueden resultar más complejas de implementar. Es posible que en migraciones de pequeño tamaño resulten algo excesivas. Por otro lado, las metodologías como IBM Data Migration hacen un planteo simple de tres etapas lo cual facilita su comprensión e implementación, pero a la vez deja aspectos libres que deben resolverse en cada caso en particular.

Por último, debe notarse que algunas incluyen la utilización de determinadas herramientas como parte del proceso lo cual las hace dependientes de una plataforma en particular. Otras realizan una división de tareas y etapas independiente de cualquier tecnología y por lo tanto pueden implementarse más allá de los motores de bases de datos con los que se esté trabajando.

Respecto de las herramientas que pueden emplearse en el proceso de migración propiamente dicho, o sea, para transformar y transportar los datos y más allá de las actividades de planificación, control, gestión y testing, es posible dividirlas en dos grupos.

Por un lado, están los lenguajes de programación que posibilitan escribir procesos que transformen los datos con alto nivel de detalle. Esa es justamente su principal fortaleza, pero debe tenerse en cuanta que mientras más programas formen parte del proceso de migración o más

extensos o más complejos sean, mayor será el esfuerzo requerido no sólo para escribirlos sino para mantenerlos, probarlos y corregirlos.

En el contexto de estudio, si la base de destino cambia en su estructura y los programas deben adaptarse a esos cambios. Mientras más se haya utilizado este recurso, mayor será el esfuerzo para mantenerlo.

Por otro lado, existen las herramientas gráficas que los diferentes fabricantes ofrecen y que tienen distinto grado de utilidad. Algunas son simples y permiten configurar un mapeo de elementos para hacer alguna transformación entre objetos de la base de datos de origen y la de destino. Por ejemplo, convertir algunos tipos de datos, adaptar longitudes de campos y luego transportar registros. Otras son más complejas y abarcan diferentes etapas del proceso, incluso algunas metodologías como la que propone Microsoft (28) se basan fuertemente en ellas.

En el contexto de estudio este tipo de herramienta gráfica es de gran ayuda porque suelen adaptarse mejor a cambios frecuentes en la base de destino. Puede resultar significativamente más visual, intuitivo y simple modificar algunos elementos gráficos (flechas, figuras, pequeños scripts) que adaptar complejos programas.

Finalmente, el uso de determinadas herramientas depende de las características del proceso en sí, de la forma en que se organice el equipo, de los conocimientos previos que se tengan, de la metodología que se emplee entre otros factores.

Estructura general de las metodologías clásicas expuestas

Más allá de las coincidencias y diferencias y con mayor o menor detalle, en general en todas las metodologías clásicas analizadas se han observado tres etapas importantes:

- Una etapa de análisis y diseño previa a la construcción y ejecución del proceso de migración.
- Una etapa de construcción y ejecución del proceso en sí.
- Una etapa de pruebas sobre la base de datos que se genera como salida del proceso.

En la etapa de análisis y diseño, en términos generales las metodologías estudiadas plantean indagar en las bases de datos de origen tanto en su estructura como en la información que contienen y diseñar un proceso que transforme esos datos en función de los requerimientos de la base de datos que se pretende obtener como salida.

En la etapa de construcción y ejecución, en términos generales la propuesta es destinar esfuerzos a la elaboración de un proceso que pueda tomar los datos de las fuentes de origen y obtener como salida una base de datos que cumpla con el diseño planeado. Este proceso será de mayor o menor complejidad dependiendo de cada situación y se podrán usar diversas herramientas que van desde elementos muy automatizados hasta lenguajes de programación que permiten escribir rutinas de transformación y pasaje de datos.

En la etapa de prueba el enfoque general es realizar diferentes verificaciones para garantizar que la base de datos obtenida responde a las necesidades y a las reglas que fueron planteadas en la primera fase. En muchos casos se debe verificar que la aplicación o grupo de aplicaciones que utilizan la base de destino funcionen correctamente a partir de los datos migrados.

Como lo afirman Birgit Kreuz y Antony Higginson, la figura siguiente expone este formato general según el enfoque de la firma Oracle. (35)

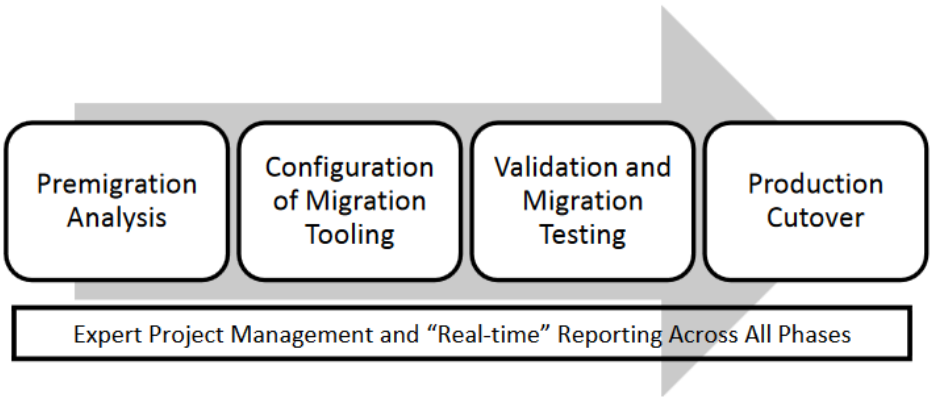


Figura 20: Fases de una migración Oracle. (35)

De la misma manera, la figura siguiente expone el enfoque de la firma especializada en consultoría y procesos de migración Startin Point. (36)

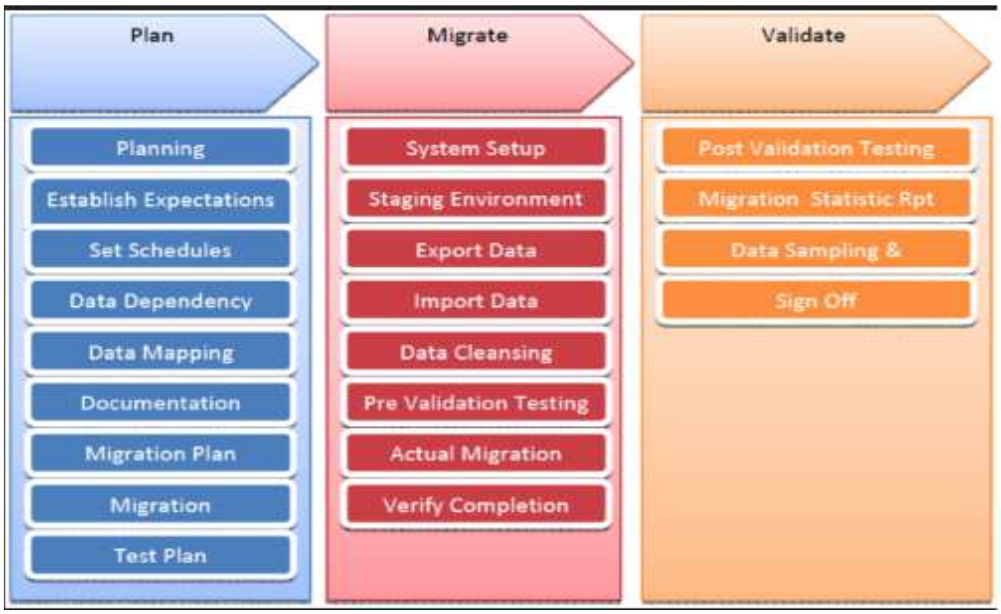


Figura 21: Fases de una migración Startin Point. (36)

Esta otra óptica pertenece a la compañía Black Blade Associates, Inc. (37)

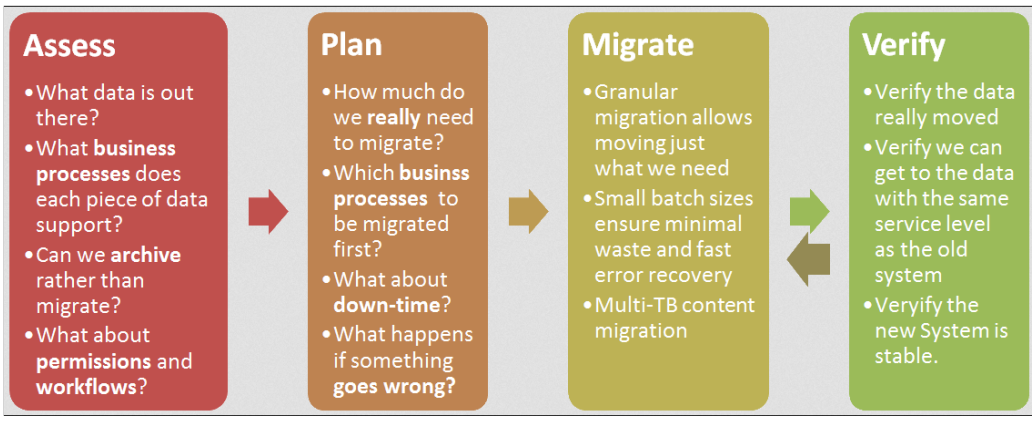


Figura 22: Fases de una migración Black Blade. (37)

Y podrían citarse muchas otras fuentes. Lo importante es destacar que si bien algunas metodologías plantean una división más granular (y por ese motivo tienen más etapas), otras hacen

una segmentación más general, pero como ya se ha dicho, en términos generales todas definen estos tres grandes pasos.

Conclusiones

En este capítulo se han analizado algunas metodologías y herramientas propuestas por compañías importantes, con el objeto de encontrar factores comunes que puedan ser relevantes en el contexto de estudio.

Se ha hecho foco en empresas con fuerte presencia en el mercado de los motores de bases de datos y con amplia experiencia en migraciones y procesos relacionados.

Respecto de los distintos planteos metodológicos puede concluirse que más allá de las diferencias estructurales y de los distintos niveles de complejidad**, en general se visualiza el proceso de migración como una actividad que debe transformar y transportar datos desde un origen estable hacia un destino también estable. Por este motivo las metodologías no prevén mecanismos que permitan gestionar cambios de manera eficiente**.

Como ya se ha dicho antes, en un proceso de migración realizado mientras la base de datos de destino se encuentra asociada a un sistema aún en desarrollo, los cambios estructurales pueden ser numerosos y en todas las metodologías analizadas, esto puede significar una frecuente repetición de algunas de sus etapas.

Otro aspecto relevante y común es que todas enfrentan el problema como un solo proceso cuando en realidad pueden visualizarse al menos dos. Por un lado, el problema sintáctico propio de realizar transformaciones de tipos, cambios en longitudes de campos, ajustes de diferencias de formatos y todo lo que se desprenda de un cambio de plataforma y por otro lado, el problema semántico que se presenta al migrar datos que fueron generados con una determinada lógica hacia una base que responde a un sistema diferente.

Las metodologías analizadas enfocan estos dos desafíos como uno solo y eso también dificulta el procesamiento de cambios ya que con cada modificación en la base de destino será necesario volver a analizar ambas cosas juntas dentro de los mismos procesos y herramientas. Si se gestionaran en forma separada, algunos cambios sólo impactarían en un área en particular e

incluso si llegaran a afectar ambos aspectos, igualmente se obtiene un proceso más ordenado y fácil de gestionar.

Respecto de las herramientas examinadas ocurre algo similar. Tanto las que emplean recursos gráficos para realizar algunas partes del proceso como las que se basan en elementos de programación, pueden utilizarse tanto para resolver el problema sintáctico como el semántico, pero no promueven una división de ambos enfoques.

Las distintas herramientas sin duda son de gran utilidad, algunas abarcan muchos aspectos de la migración y acompañan todo el proceso otras están dirigidas a elementos puntuales, pero en ningún caso se puede esperar que resuelvan la migración por sí mismas. En otras palabras, el empleo de determinadas herramientas no garantiza un proceso exitoso.

En definitiva, las metodologías existentes son muy variadas y más allá de las similitudes y diferencias que pueden encontrarse entre ellas, todas constituyen recursos probados y de gran utilidad sin duda, pero a la vez, resultan de difícil implementación en el contexto de estudio. De allí la necesidad de plantear un mecanismo de trabajo que permita utilizar las herramientas disponibles pero que a la vez sea flexible y se adapte a cambios frecuentes en la base de destino.

Capítulo IV: Propuesta de una metodología y un modelo físico para procesos de migración de datos en sistemas en desarrollo

Introducción

Como se menciona en las conclusiones del capítulo anterior, las diferentes metodologías analizadas hacen un aporte significativo y pueden ser aplicadas en distintas situaciones en las que se requiera realizar una migración de datos.

Algunas son más complejas otras tienen un planteo estructural más simple, unas dan mayor importancia a las etapas previas de análisis y diseño, otras están centradas en el proceso en sí o en las pruebas a la salida del mismo. Unas visualizan el proceso como un todo y lo dividen en pocas actividades, en otras se mantiene una visión más granular del trabajo y el proceso aparece dividido en pasos más puntuales.

Más allá de los diferentes enfoques, todas parten de la existencia de una base de destino y una de origen ya construidas, con una estructura definida y estable. Por esta razón incorporan muy pocos o ningún elemento que facilite la adaptación del proceso a cambios en la base de datos de salida.

Al mismo tiempo, todas utilizan un conjunto de herramientas para abordar la transformación sintáctica y la semántica dentro de un mismo núcleo lo que puede dificultar aún más el procesamiento de cambios frecuentes.

En este capítulo se analiza cómo operan las metodologías clásicas frente al procesamiento de un cambio. Cuáles son los problemas frecuentes, cuáles son las actividades que deben repetirse y que impacto tiene un cambio en todo el proceso.

Luego, como respuesta a los problemas encontrados, se propone primeramente un modelo físico que permite dividir la migración en dos etapas y abordar por separado el problema sintáctico y el semántico. A partir de este modelo, se construye una metodología que también divide la propia transformación, las realizaciones de pruebas y validaciones previendo etapas iterativas para lograr

que todo el proceso sea más flexible y resulte más adaptable a cambios en los objetivos, es decir, a cambios en la base de datos que se pretende generar.

El principal objetivo de este capítulo es exponer tal metodología junto a su arquitectura física y analizar su funcionamiento frente al procesamiento de cambios. También se realizan comparaciones con algunas metodologías clásicas expuestas en el capítulo anterior.

Problemática común de las metodologías clásicas en migraciones que ocurren durante el desarrollo de un nuevo sistema

Ahora que ya se han analizado algunas metodologías tomadas como referencia y también su estructura y aspectos comunes, pueden verse con más claridad cómo se utilizan en el contexto de estudio.

Más concretamente, en el caso particular de un proyecto de migración de datos que se lleva adelante en forma conjunta con un proceso de desarrollo de un nuevo sistema, el planteo metodológico clásico presenta algunas dificultades como consecuencia de los cambios de requerimientos que pueden ser frecuentes y de un impacto significativo, tal como se analizó en el capítulo dos.

Para expresarlo de manera gráfica, se puede tomar cualquiera de las figuras expuestas en el capítulo anterior y detenerse a analizar cuál es el proceso a seguir con cada cambio en la base de destino.

Por ejemplo, tomando el planteo genérico y sencillo de la firma Startin Point (36), puede reconstruirse una imagen como la siguiente.

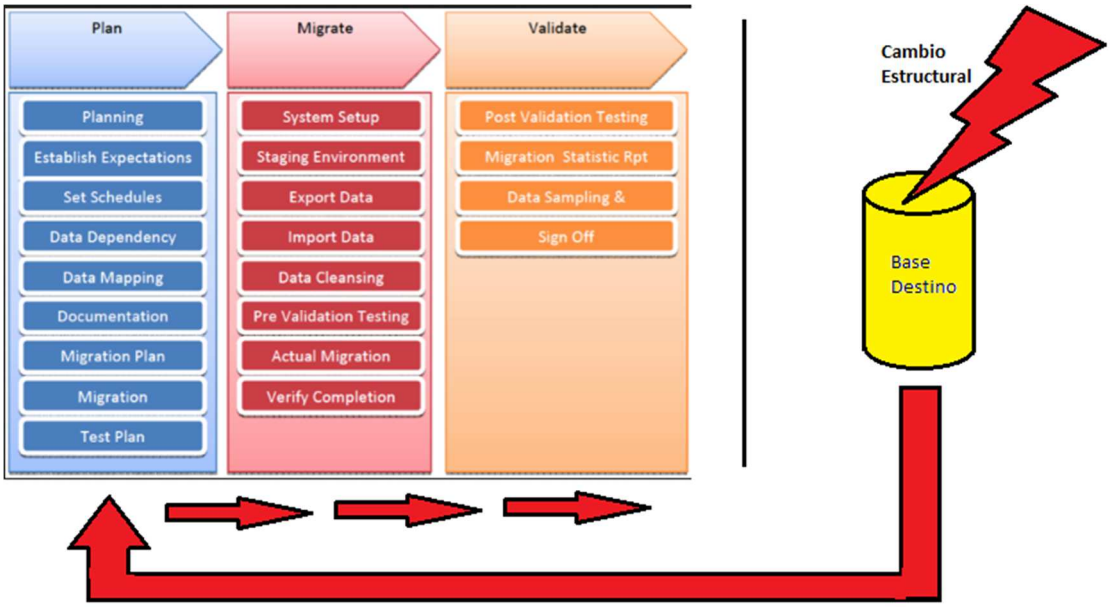


Figura 23: Cambio de la base de salida en una metodología clásica

A partir de la figura anterior, puede advertirse que, si un cambio en los requerimientos del sistema impacta en la estructura de la base de datos de salida, el proceso de migración deberá adaptarse y para ello será necesario pasar nuevamente por los tres pasos descriptos.

En otras palabras, si la base de destino cambia se deberá volver a hacer un análisis para modificar consecuentemente el proceso de migración, volver a obtener una base de destino y realizar nuevamente pruebas sobre ella. Si los cambios son frecuentes repetir todo el proceso puede resultar lento y costoso.

Metodologías que mantienen un enfoque más detallado con mayor granularidad en la división de tareas y mayor complejidad pueden verse más afectadas que aquellas en las que el enfoque es más simple. Mientras mayor sea la cantidad de pasos y documentos a modificar, mayor será el impacto.

Al mismo tiempo las metodologías más sencillas definen menos elementos y por ende son más adaptables pero muchas veces no pueden utilizarse en proyectos de migraciones complejas y de gran tamaño.

Está muy claro que cada metodología es diferente y puede adaptarse con mayor o menor eficiencia a distintas situaciones, pero mientras el planteo general sea el de planificar un proceso, implementarlo y por último probarlo, siempre ante un cambio en la base de destino será necesario volver a repetir estas etapas.

Esta situación no debe resultar sorprendente, de hecho, es natural que sea así porque todas las metodologías analizadas parten del supuesto de que la base de destino es estable, tal como se mencionó en el capítulo pasado. Es más, en muchas situaciones la migración es así, pero dentro del contexto de estudio esto no ocurre.

Por supuesto, en la implementación de cada metodología seguramente el equipo de trabajo encuentra los mecanismos para poder hacer frente a los cambios. Es factible hacer adaptaciones sobre la marcha y no repetir de manera innecesaria la totalidad de cada etapa, pero debe notarse que incluso en ese caso, se trata de adaptaciones empíricas y no de un planteo metodológico acorde que prevea esta situación y se anticipe. Las distintas fuentes citadas en el capítulo dos muestran con claridad las dificultades que con frecuencia atraviesan los procesos de migración.

Procesamiento de un cambio de requerimiento en una metodología clásica

Para exponer con mayor claridad la situación descripta en el apartado anterior, se plantea aquí un ejemplo de cómo sería procesar un cambio de requerimiento en una metodología clásica. Este caso es una abstracción que se desprende de una situación real dentro del proceso de migración presentado en el capítulo cinco como implementación.

Dentro del desarrollo de un nuevo sistema que reemplaza a otro existente se comienza paralelamente, en un determinado momento, el desarrollo del proceso de migración que tiene como objetivo incorporar a la nueva base de datos la información proveniente del sistema a ser sustituido.

Las diferencias son tanto técnicas como semánticas. Es decir, la plataforma que aloja los datos en la base de origen es diferente a la que se debe utilizar para soportar la base de destino y a la vez existen grandes diferencias estructurales como consecuencia del diseño del nuevo sistema.

Una de las entidades de la nueva estructura de datos que debe llenarse con información migrada es la tabla de personas que originalmente se diseñó (para el nuevo sistema) con la siguiente estructura:

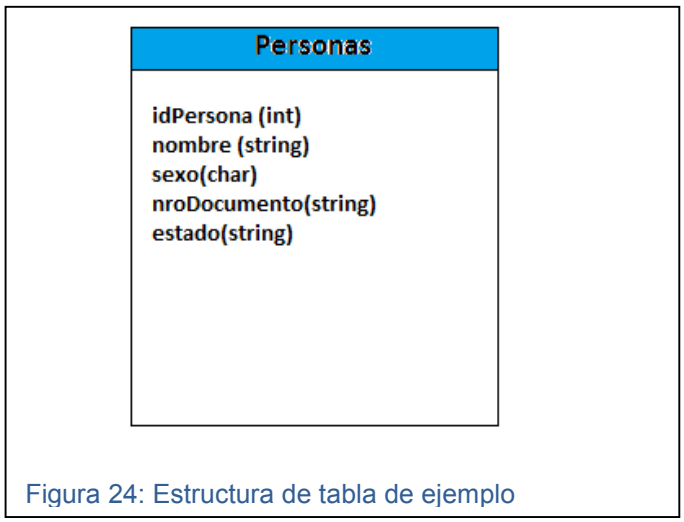


Figura 24: Estructura de tabla de ejemplo

El campo "estado" inicialmente se ha diseñado como un string y el desarrollo del sistema lo ha tratado de esa forma desde el comienzo. Por ende, el proceso de migración, también lo ha tratado de esta forma. La migración se ha planeado, diseñado, construido, ejecutado y probado con el campo "estado" como un string.

En un momento del desarrollo, un cambio de requerimiento ha de surgir la necesidad de tipificar los estados de las personas. Aunque en el sistema actual no existe este criterio, en realidad los estados responden una estructura de tipos que pretende incluirse en el sistema nuevo para que ya no sean escritos simplemente como una cadena de caracteres. Al procesar este cambio de requerimiento, se modifica la documentación correspondiente y se ingresa el cambio a desarrollo. Ahora la base de datos responde a esta estructura:

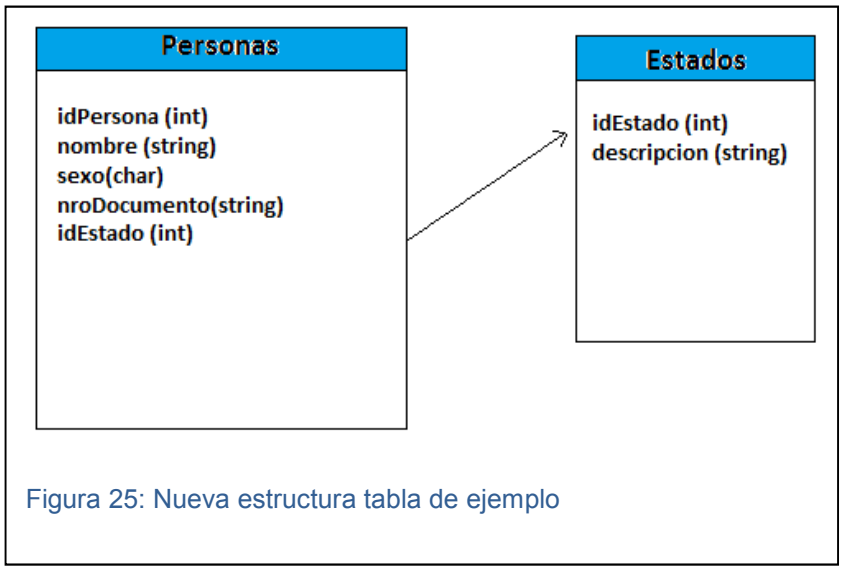


Figura 25: Nueva estructura tabla de ejemplo

Ocurre que la base que ha cambiado de estructura no es solamente objeto del proceso de desarrollo del sistema, sino que es también la base de destino de la migración. Consecuentemente ahora el proceso de migración debe adaptarse también al mismo cambio. Si en este momento se ejecutaran los programas que forman parte de la migración, éstos fallarían ya que el campo "estado" no existe, en su lugar hay un nuevo campo "idEstado" y se ha creado una nueva tabla que la migración ignora.

Para adaptar la migración será necesario:

1) Evaluar el cambio y volver a planificar el proceso. Un cambio pequeño puede tener un impacto importante. Puede resultar difícil identificar qué impacto tiene cada cambio.
2) Modificar los procesos que toman la información desde la base de origen para que puedan transformarla a la nueva estructura de tipos y puedan escribirla en la base de destino. Si se usaron programas para realizar la lectura y escritura tendrán que reescribirse y si se usaron herramientas gráficas habrá que reconfigurarlas.
3) Modificar los procesos de prueba y validación ya que ahora ha cambiado la estructura de la base de salida, la lectura de datos y la escritura.
4) Volver a ejecutar toda la migración para generar una nueva base de destino
5) Volver a ejecutar todos los test ya que se ha obtenido una base de destino totalmente nueva.

Todo esto ocurre mientras el propio proceso de migración sigue en desarrollo. Así que será necesario organizar el resto de las actividades para poder encadenarlas con este cambio y encontrar un punto en el que pueda ejecutarse y validarse correctamente.

Realizar este cambio resulta perfectamente factible, pero debe notarse que afecta a todas las etapas de la migración. Si se ha utilizado una metodología más compleja y detallada serán más pasos los que se tengan que revisar y probablemente modificar. Si esto ocurre una cantidad de veces relativamente pequeña puede absorberse el impacto, pero si ocurre con mucha frecuencia (como suele suceder dentro del desarrollo de sistemas) implica una carga de trabajo muy significativa.

Dentro de la misma dinámica del desarrollo surge luego otra necesidad. Ahora el nuevo sistema debe incorporar información de los domicilios de las personas, aunque esto no formaba parte del diseño original. La información existe y puede ser extraída de la base de datos de origen, pero no ha sido contemplada en los procesos de lectura.

Los pasos para procesar este cambio deben ser los mismos que en el caso planteado anteriormente. Es decir, debe analizarse el cambio, replanear el proceso, modificar los programas y las herramientas que realizan la lectura de datos de la base de origen y que llevan a cabo la escritura en la base de destino, ejecutar los procesos para obtener una nueva base y volver a probar y validar los resultados.

En este segundo caso cabe el mismo análisis que en el primero. Afecta a todo el proceso y las conclusiones serían las mismas: puede hacerse, pero implica un retrabajo importante.

Debe notarse que si bien ambos casos son abordados de la misma manera no son iguales. En el primero los datos se estaban leyendo correctamente, sólo que necesitaban ser tipificados en lugar de guardarse en una cadena de caracteres. En el segundo caso los datos no se estaban leyendo desde la base de origen.

Más tarde una versión del sistema en desarrollo es probada por el equipo de testing y dentro de los defectos reportados se descubre que los nombres de las personas en algunos casos no se visualizan en forma completa. El equipo analiza la aplicación y descubre que la longitud del campo nombre prevista en el nuevo sistema es inferior a la que preveía el anterior por lo tanto algunos

registros migrados simplemente exceden la cantidad de caracteres previstos. Como solución se planea una modificación en la base de datos y se agranda el campo nombre de la tabla de personas.

Nuevamente debe modificarse el proceso de migración, corregir los programas y herramientas que realizan las transformaciones, volver a ejecutarlo y probarlo. Pero debe notarse que aquí el problema no estaba en la lectura de los datos de origen, sino que sólo afectaba a la escritura en la base de destino. No hay dificultades en la transformación ni han surgido cambios funcionales que representen modificaciones semánticas en la base de datos, sólo se trata de un problema de longitud.

Por último, puede plantearse una situación más. Ahora en una de las validaciones previstas en la última etapa del proceso de migración se descubre que algunas personas no tienen el estado correcto. La lectura de la base de origen funciona bien porque la información es encontrada, la escritura en la base de destino también porque los registros se almacenan, pero la transformación intermedia tiene un error y bajo determinadas situaciones calcula incorrectamente el valor para este campo.

En este caso nuevamente debe analizarse una parte significativa del proceso, aunque en realidad la lectura de la base de origen funciona correctamente. El impacto es similar al del resto de los casos.

En todas las situaciones presentadas el impacto es similar, aunque en realidad son de naturaleza muy diferente. En algunos casos debió hacerse foco en los procesos de lectura de la base de origen, en otros en los de escritura y en otros en la propia transformación.

Tanto las diferencias sintácticas como semánticas son tratadas de la misma manera por las metodologías clásicas y el proceso de lectura transformación y escritura es planteado como uno solo. Esto aumenta la complejidad y provoca que cada modificación requiera una revisión completa o al menos extensa.

Lo mismo pasa con las pruebas, normalmente se ejecutan en la salida de la migración sobre la base de destino. Con un único conjunto de pruebas y validaciones se evalúan las transformaciones de todo tipo.

En migraciones complejas, la evaluación y realización de cada cambio puede requerir un esfuerzo importante y la ejecución de todo el proceso puede llevar mucho tiempo. Lo mismo ocurre con las validaciones, incluso tratándose de pruebas automatizadas, con cantidades muy grandes de registros y lógicas complejas pueden requerir mucho tiempo de ejecución.

Regresando sobre el capítulo dos y revisando algunas de las estadísticas publicadas por diferentes fuentes, se puede volver a citar por ejemplo lo que afirma Nancy Hurley y la firma ESG (15) (10): El 54% de las migraciones de datos excede el presupuesto planificado. O lo que afirma la compañía Oracle (16) : el 83% de los procesos de migración exceden el presupuesto y el tiempo panificado, sólo por mencionar algunos.

Indudablemente agregar a la complejidad inherente de este tipo de actividades un escenario de cambios frecuentes incrementa el riesgo de sobrepasar los parámetros estimados originalmente a la vez que dificulta la gestión y planificación de todo el proceso.

Problemática común de las herramientas automatizadas

En el capítulo anterior se analizaron herramientas que pueden utilizarse en varias etapas de cada metodología. Algunas están diseñadas para un fin específico, otras son de uso más general y pueden emplearse en todo el proceso. Mientras unas fueron creadas para una metodología o tecnología en particular otras son independientes de la plataforma y de la manera en que se organizan las etapas. Algunas son fundamentalmente gráficas, otras se basan en mecanismos de programación y en mapeos parametrizados.

Más allá de las similitudes y diferencias debe notarse que si bien su empleo es de suma importancia, no soluciona el problema que implica procesar cambios frecuentes en la base de destino. Desde luego un recurso gráfico que provea algún grado de automatización es de mucha ayuda porque simplifica la tarea de modificar los procesos, pero esto no significa que el propio procesamiento de cambios pueda automatizarse. No debe asumirse que, por contar con herramientas automatizadas y gráficas, procesar cambios frecuentes resulte sencillo.

Si el grado de complejidad y el volumen de la migración son altos, las herramientas alcanzan un nivel de configuración también complejo. Por ejemplo, si se mapean tablas con determinados campos y tipos de datos haciendo muchas transformaciones, incluso de manera gráfica, puede resultar difícil ubicar luego cada transformación y modificarla ante un cambio en la base de salida.

También debe tenerse en cuenta que independientemente de las herramientas empleadas, cada cambio conlleva a potenciales errores y a la necesidad de volver a realizar pruebas y posiblemente correcciones. Con las herramientas para realización de pruebas ocurre algo similar, sin duda son de mucha ayuda, pero cuando se tienen dispuestas muchas pruebas complejas, modificarlas, volver a ejecutarlas, evaluar e informar los resultados no es tarea simple.

En definitiva lo que ocurre es que las herramientas del tipo de las analizadas en el capítulo anterior proponen un alto grado de automatización que resulta muy útil para resolver aspectos técnicos como por ejemplo conversiones de tipos de datos, resolución de formatos diferentes para los mismos tipos (como suele ser el caso de campos tipo fechas), mapeo de campos con cambios de longitudes etc. pero naturalmente no solucionan aspectos semánticos relacionados con las diferencias de negocio de los sistemas reflejadas en las bases de datos de origen y de destino de la migración. Por este motivo, los cambios impactan fuertemente incluso en el uso de estos recursos.

Arquitectura clásica de un proceso de migración

Las dificultades mencionadas en los apartados anteriores no se deben al uso de una metodología en particular o de una herramienta en particular. Guardan estrecha relación con la arquitectura de los procesos de migración. Como ya se dijo antes, éstos están pensados para transportar datos entre orígenes y destinos estables en su estructura.

Como lo dice Anil Mahadev (38) una arquitectura clásica para migraciones de bases de datos es la que muestra la figura siguiente.

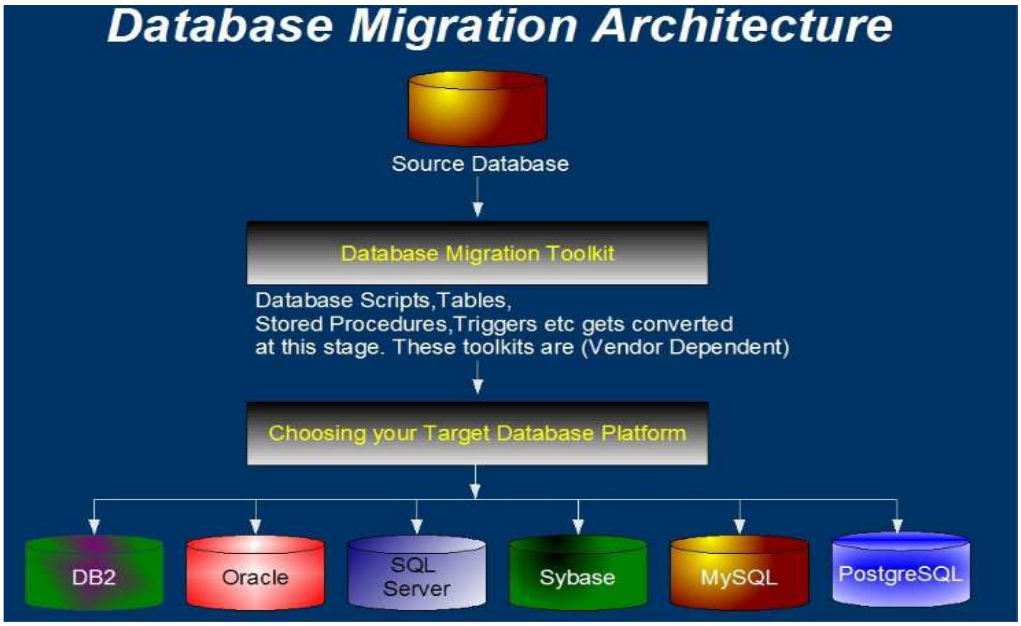


Figura 26: Arquitectura de migración de base de datos (38)

En figura se muestran como posibles destinos bases de datos que responden a productos de determinados fabricantes, pero el concepto es el mismo independientemente de esto.

Así, por ejemplo, y citando la misma fuente, un proceso que tome como origen una base de datos de SQL Server y como destino una de DB2 puede representarse con la figura siguiente (38)

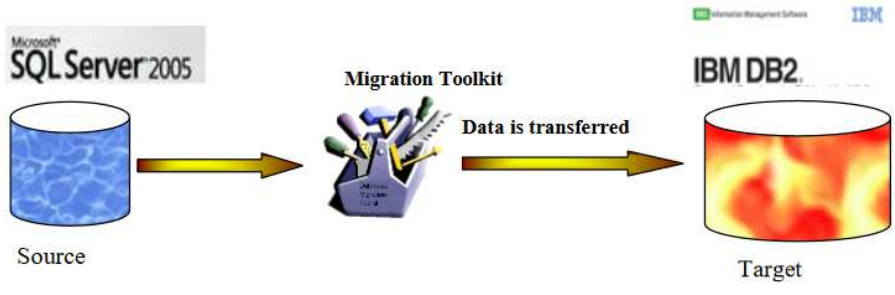


Figura 27: Migración desde SQL Server a DB2 (38)

Como puede advertirse en la imagen, existe una base de origen, una de destino y un proceso de transformación que se basa en un conjunto de herramientas.

Aunque el ejemplo habla de dos fabricantes en particular, el enfoque puede aplicarse a cualquier migración independientemente de la tecnología de las bases de origen y destino, de las herramientas que se utilicen y da la metodología empleada.

En términos generales, siempre existe un proceso de transformación que toma con entrada la base de origen y brinda como salida la de destino. En ese proceso debe hacer foco toda la planificación, toda la administración y asignación de recursos. Allí trabajan todos los miembros del equipo, se utilizan todas las herramientas y metodologías expuestas, se realizan todas las pruebas y validaciones y naturalmente, allí también impactan todos los cambios que se generen como consecuencia del desarrollo del sistema que utiliza la base de salida.

La figura siguiente ilustra en forma genérica la arquitectura clásica.

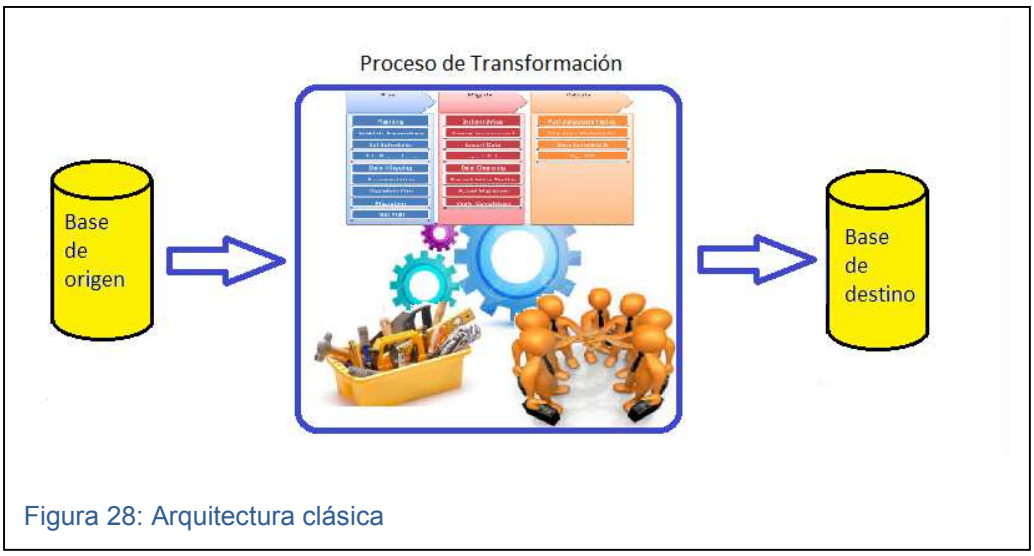


Figura 28: Arquitectura clásica

En definitiva, todos los recursos que se han analizado intervienen un único proceso de transformación. Las metodologías expuestas destinan todas sus etapas a planificar, realizar y validar este proceso. Con las herramientas ocurre algo similar, todas están diseñadas y se utilizan en el mismo contexto. Los miembros del equipo de trabajo también actúan sobre un mismo proceso independientemente del rol que le toque cubrir a cada uno.

Esto es natural, al fin y al cabo, todos los elementos forman parte del mismo proyecto, pero a la vez eleva la complejidad y hace que cualquier cambio impacte en muchos agentes intervinientes a la vez.

Modelo de bases y procesos divididos

Como ya se ha mencionado antes, existen dos grandes problemas a resolver dentro de una migración. Uno relacionado con las conversiones sintácticas y otro con las semánticas. Por un lado, si las plataformas son diferentes seguramente habrá que convertir tipos de datos, longitudes, formatos etc. Al mismo tiempo, si los datos son los mismos, pero responden a sistemas distintos, sin duda habrá que adaptar su significado, su estructura y relaciones.

Las metodologías y herramientas analizadas enfocan ambos aspectos como uno solo, pero pueden resolverse estos problemas por separado y así obtener dos procesos de transformación. Uno de transformación sintáctica y otro de transformación semántica.

En un escenario de cambios frecuentes esta división resulta muy conveniente. El modelo propuesto a continuación parte de dividir al proceso de migración físicamente en dos y en utilizar también dos bases de destino para realizar la migración en dos etapas. De esta manera ya no se tiene un único bloque complejo donde interactúan todos los agentes intervinientes y donde impactan todos los cambios, consecuentemente se disminuye la complejidad y se agiliza la gestión de las modificaciones.

La figura siguiente ilustra el modelo de bases y procesos divididos.

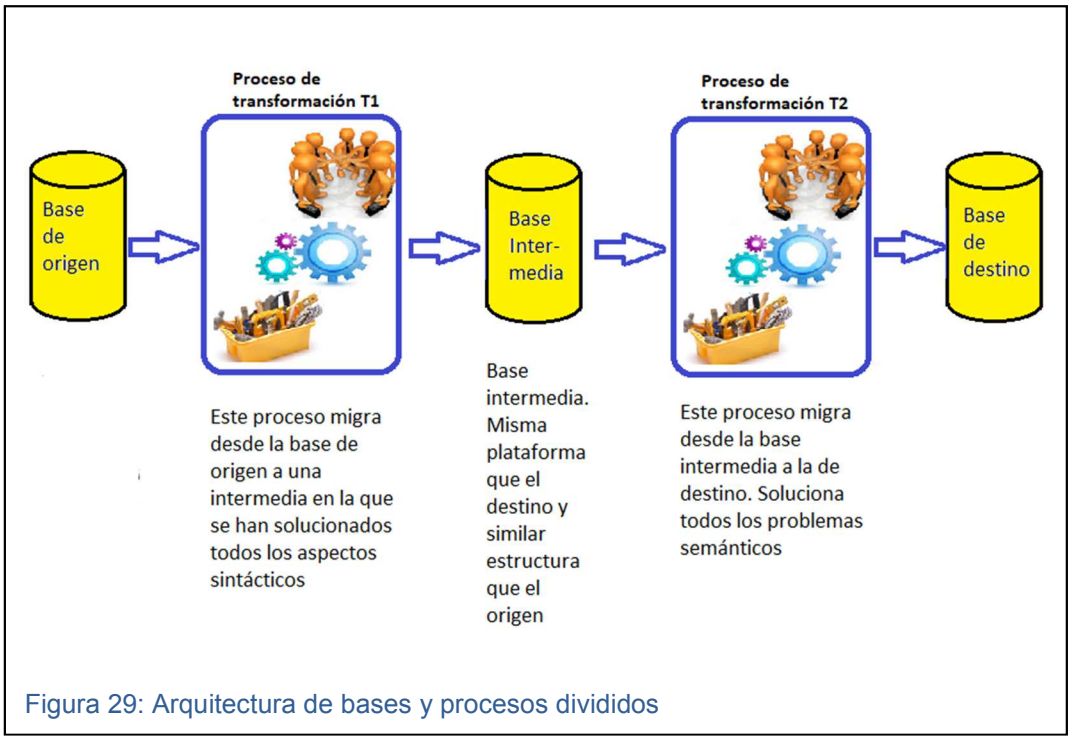


Figura 29: Arquitectura de bases y procesos divididos

Como puede advertirse en la imagen, la migración parte de una base de origen (se ha ilustrado solo una, pero podrían ser varias, el planteo sería idéntico). Un primer proceso en el que intervienen herramientas y recursos del tipo de los analizados anteriormente, realiza una primera transformación, pero haciendo solamente foco en las adaptaciones sintácticas.

El objetivo del primer proceso, llamado T1, es el de adaptar tipos de datos, longitudes, formatos y todo elemento sintáctico para obtener una única base de salida intermedia montada sobre la misma plataforma que la base de destino final, pero con las estructuras semánticas lo más similares que resulte posible a las bases de origen. De este modo el proceso T1 contendrá sólo elementos de transformación sintáctica. Todas las herramientas y recursos intervinientes tendrán ese único objetivo.

Un segundo proceso, llamado T2, toma la base intermedia generada por T1 y abstraído de cualquier diferencia de sintaxis (porque ya no existen) realiza todas las transformaciones estructurales para lograr que los datos de origen se adapten al diseño de la base de salida. En este módulo todos los recursos intervinientes se orientan a analizar y adaptar los datos estructuralmente.

A partir de esta división se disminuye la interdependencia entre las bases de origen y la de destino. Al desarrollar T1 todos los recursos se orientan a resolver un problema de plataforma, teniendo en cuenta la tecnología de la base de destino, pero no su diseño ni su estructura ni ninguno de sus aspectos funcionales. Consecuentemente, la dinámica del desarrollo del sistema, que en muchos casos implica cambios en el diseño de la base de datos, muchas veces resulta transparente y no tiene impacto en esta fase de la migración.

Al mismo tiempo, al trabajar sobre T2, ocurre lo contrario, todo cambio en la estructura de la base de datos de destino impacta en forma directa, pero como se tiene una única plataforma y una sola fuente de origen, será menos costoso realizar las modificaciones.

Al momento de procesar un cambio de requerimiento, dependiendo del tipo de cambio de que se trate, es probable que sólo impacte en uno de los procesos ya sea T1 o T2. Incluso si impacta en ambos, será más fácil medir su incidencia y procesarlo ya que cada módulo es individualmente más sencillo.

Es importante destacar que una división física y un enfoque de arquitectura no alcanzan para resolver los problemas típicos del contexto de estudio. Se requiere sin duda un planteo metodológico que permita organizar toda la migración en torno de esta estructura, pero hasta el momento (y sólo por hacer una división física) ya pueden advertirse algunas ventajas importantes. Más adelante en este mismo capítulo se propone una metodología acorde.

La enumeración siguiente resume las características y ventajas que implica la arquitectura propuesta.

1. T1 no guarda relación con la base de datos de destino, por lo tanto, la dinámica del proceso de desarrollo frecuentemente no lo afecta.
2. T2 no guarda relación con las plataformas de las bases de origen por lo tanto las diferencias de tecnología no lo afectan.
3. T1 es individualmente más simple que el proceso total por lo tanto desarrollarlo y mantenerlo es también más simple.
4. T2 es individualmente más simple que el proceso total por lo tanto desarrollarlo y mantenerlo es también más simple.

5. No todos los cambios impactarán en T1 yT2. En muchos casos sólo tendrán incidencia en uno de ellos.
6. El desarrollo y mantenimiento de ambos procesos puede realizarse en paralelo.
7. Los recursos técnicos y humanos afectados a T1 y T2 pueden o no ser los mismos.

Así como pueden advertirse varios aspectos positivos de esta arquitectura, también pude verse una desventaja significativa. Si bien T1 y T2 individualmente serán más simples que el proceso de transformación total, porque cada uno realiza una tarea específica, es muy probable que ambos juntos sean más complejos.

Al separarlos, internamente se obtendrán dos fases de lectura, dos transformaciones y dos escrituras. También aparece una base de datos intermedia a la que hay que dar soporte físico y administrar. Las herramientas utilizadas pueden no ser las mimas y consecuentemente los recursos humanos afectados deben tener un mayor conocimiento o bien desempeñarse sólo dentro de uno de los procesos.

Si se escriben programas como recursos de la migración, es probable que al realizar esta separación se escriban más líneas de código y se generen algoritmos separados. Si se utilizan herramientas con recursos gráficos es probable que se hagan más configuraciones y mapeos.

La arquitectura expuesta no apunta a generar una migración más pequeña en términos de configuraciones, líneas de código, uso de herramientas, cantidad de personas afectadas, etc. El objetivo es lograr un proceso más adaptable.

Bajando la arquitectura a un planteo práctico

Se expone a continuación un ejemplo que deriva de una abstracción o simplificación de uno de los casos de aplicación presentados en el capítulo siguiente.

Durante esta migración deben tomarse como origen tres fuentes. Dos de ellas responden a archivos de texto plano y otra a archivos con formato DBF. El destino es una base de datos Oracle.

Las diferencias no son sólo de tecnología (archivos de texto, tablas dbf y base Oracle) sino que hay fuertes cambios estructurales. El nuevo sistema difiere significativamente en sus reglas de

negocio respecto de aquellos que reemplaza y consecuentemente su base de datos es estructuralmente muy distinta.

Al mismo tiempo los datos generados por los sistemas a ser reemplazados no pueden perderse, son de gran valor y el nuevo sistema no puede ponerse en marcha sin incorporarlos a su base de datos.

Tampoco puede esperarse a finalizar el desarrollo del sistema para comenzar con la migración porque esto retrasaría significativamente la puesta en marcha.

La arquitectura planteada para llevar adelante el proceso de migración dentro del escenario descripto es la que se muestra en la figura siguiente.

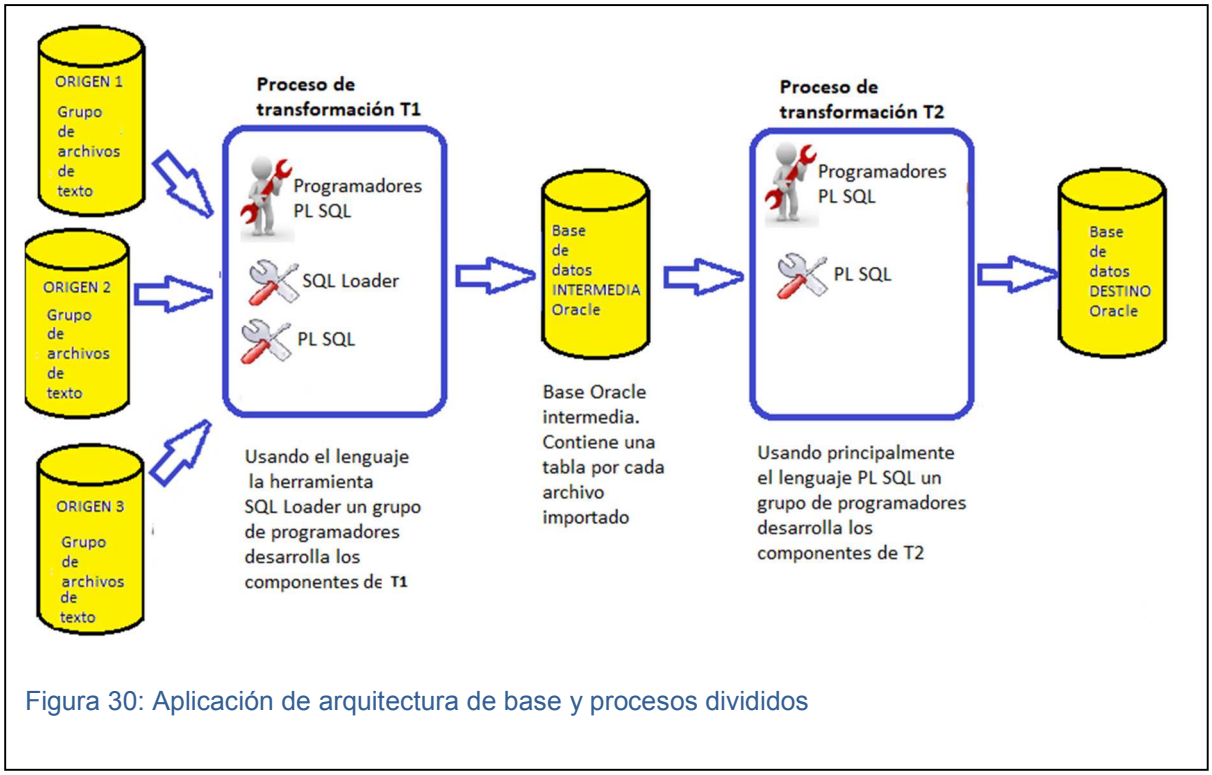


Figura 30: Aplicación de arquitectura de base y procesos divididos

En este caso en particular, se utilizan dos herramientas en el primer proceso. SQL Loader para hacer la lectura e importación de los archivos de origen y el lenguaje PL SQL para programar procesos con los que se realizan algunas transformaciones primarias.

También dentro de T1 se usa PL SQL para programar algunas pruebas relacionadas con validaciones propias de este nivel, como por ejemplo, verificar al cumplimiento de algunas premisas en los datos de entrada y controlar que la cantidad de registros de las tablas generadas coincidan con los que originalmente tienen los archivos de texto.

En el caso de T2 se utiliza como principal herramienta PL SQL para programar procesos de transformación que leen en las tablas de la base intermedia y escriben en la base de salida haciendo las transformaciones más complejas que son las que derivan de la lógica de negocio del nuevo sistema.

También se utiliza el mismo lenguaje para desarrollar las pruebas propias de este nivel.

Metodología de tres niveles iterativa

A partir del planteo anterior puede visualizarse la arquitectura y las razones para dividir físicamente el proceso de transformación. Pero también puede advertirse que sólo con esto no alcanza para llevar adelante toda la migración.

Quedan muchas preguntas por responder:

- ¿Cómo se procesa un cambio si los procesos están divididos?
- ¿Cómo y dónde se realizan las etapas de prueba y validación?
- ¿Cómo se gestiona la generalidad del proceso?
- ¿Cómo se planifica el proceso?
- ¿En qué momento se ejecuta la migración?

Se advierte claramente que éstas son las preguntas que responden las metodologías expuestas en el capítulo anterior. Lo que se requiere entonces es un planteo metodológico acorde a las premisas expuestas.

Retomando la estructura general de las metodologías analizadas, las tres etapas: planificar, migrar y probar no pueden aplicarse sobre la arquitectura propuesta. La migración en general requiere de estas fases, pero cada proceso individualmente también es una migración en sí misma.

Además, debe tenerse en cuenta el procesamiento de cambios que es la razón de ser de la propia metodología.

En síntesis, para poder utilizar este modelo, hay que contar con una metodología acorde.

Descripción de la metodología de tres niveles

Se describen a continuación las fases de la metodología propuesta. Debe tenerse presente que todas sus etapas se basan en el modelo de arquitectura expuesto y que está especialmente diseñada para gestionar un proceso de migración en forma paralela al desarrollo del sistema y de la base de datos destino.

Fase 1: Planificación y Análisis general

En esta etapa se realiza la planificación general de todo el proceso de migración. También se hace un primer análisis global, es decir, independiente de la división de los procesos de transformación. Los documentos que se construyen en esta fase son de alcance general y sirven como punto de partida del proyecto.

Más allá de que la base de destino se encuentre sujeta a cambios y de que seguramente éstos afectan el proceso de migración haciendo variar su ciclo de vida y los recursos involucrados, resulta de suma importancia alcanzar una visión general y obtener una estimación de los recursos afectados en cada etapa. También es muy importante desarrollar una primera planificación donde pueda estimarse el tiempo necesario para el desarrollo de toda la migración general.

En consecuencia, las tareas a realizar en esta etapa son las siguientes:

- **Análisis general de la migración**. Determinar su alcance y establecer parámetros generales de su volumen.
- **Análisis general de la tecnología**. Relevar las plataformas de las fuentes de origen y de la de destino. Establecer las posibles herramientas a utilizar durante todo el proceso.
- **Realización de planificación general**. Construir una primera planificación que contenga estimaciones de tiempos y recursos generales de alcance preliminar.

- **Tomar contacto con el proyecto de desarrollo del sistema**. Analizar el estado y la planificación del mismo. Determinar el momento en el que resulta conveniente comenzar con la migración y establecer los mecanismos de comunicación.

Los resultados que deben obtenerse a partir de esta fase se ven materializados en los siguientes documentos:

- **Planificación general**: Es un documento que establece las fechas en las que se comenzarán y culminarán las etapas siguientes. Es de alcance preliminar y general, sin dudas tendrá ajustes.
- **Afectación de recursos**: Es un documento que establece el personal que será necesario durante toda la migración y particularmente el que estará a cargo de desarrollar T1 y T2.
- **Determinación de recursos técnicos**: Es un documento que establece el equipamiento, herramientas de software, elementos de configuración, de comunicaciones etc.
- **Plan de comunicación**: Es un documento que establece la forma en que el equipo de migración se comunica de manera interna y la forma en que lo hará con el equipo de desarrollo del sistema. Este último aspecto es de vital importancia porque se debe establecer un mecanismo que garantice la comunicación formal de aquellas modificaciones en la base de destino porque son las que afectan a la migración y la propia razón de ser de esta metodología.

Los documentos mencionados no necesariamente tienen que ser elementos extensos y detallados. Las características de cada proyecto son la que determinan qué elementos requieren mayor detalle.

En algunos entornos estimar el personal afectado puede resultar complejo, en otros será suficiente con asignar formalmente un especialista en bases de datos con conocimientos en las herramientas a utilizar. En ocasiones identificar las herramientas a utilizar puede ser una ardua tarea mientras que en otros casos quizás vengan impuestas por la propia tecnología que se utiliza en el desarrollo del sistema.

Algo muy importante es que no debe visualizarse en ninguno de estos documentos un elemento final sino todo lo contrario, son un punto de partida. Todo lo que se estime está sujeto a cambios por lo tanto no debe perderse de vista que la fase uno tiene como objetivo dar una

orientación y planificación inicial y no "anclar" o "condicionar" la migración a determinados factores.

En este sentido, los documentos construidos deben ser flexibles, simples y adaptables. Es deseable que no tengan un gran volumen, mientras más resumidos y concretos mejor. También pueden agruparse los distintos aspectos en uno solo o incluir otros que sean necesarios según las características del proyecto particular que se trate.

El verdadero objetivo de la primera fase es otorgar una primer visión macro y general.

Fase 2: Análisis y Diseño de T1

En esta fase se toma contacto con las bases de origen que constituyen los datos de entrada de la migración y se diseña el proceso que tiene como salida la base intermedia. Las tareas de esta fase son las siguientes.

- **Realizar un análisis de la estructura física de los datos de entrada**. Esto es, identificar cuáles son las bases de origen y sobre qué tecnología están soportadas.
- **Realizar un análisis de la estructura lógica de los datos de entrada**. Resulta importante para tomar decisiones posteriores conocer como están estructurados los datos de origen. Cómo son sus objetos (tablas, archivos, relaciones, funciones, procedimientos, etc.) y a qué lógica responden.
- **Establecer la manera de conectarse con los servidores que dan soporte a las bases de origen para poder hacer la extracción**. Debe tenerse en cuenta que ésta puede no ser una tarea sencilla pues responden a sistemas en producción y pueden ser de variada tecnología.
- **Identificar el subconjunto de elementos a incluir como entrada de T1**. No todas las tablas o archivos contenidos en las bases de origen son objeto de migración. El nuevo sistema puede requerir la migración sólo de algunas entidades. Debe tenerse aquí un criterio amplio porque es probable que luego se presenten modificaciones que impliquen incorporar nuevas entidades, en este sentido es mejor tenerlas en cuenta desde un comienzo e incluirlas en la base intermedia, pero al mismo tiempo no es conveniente incorporar en T1 un gran volumen de transformaciones que no tengan utilidad real.

- **Diseñar la base de datos intermedia**. Respetando la misma tecnología que la base de destino, pero manteniendo una concepción lógica lo más similar posible a los datos de origen, se debe establecer la estructura de la base intermedia.
- **Especificar las herramientas a utilizar en el desarrollo de T1**. Se deben determinar los elementos tanto de hardware como de software para realizar la programación y en general toda la construcción del primer proceso.
- **Ajustar la planificación y asignación de recursos para T1**. En este punto se tiene un nivel de conocimiento mucho más detallado y profundo no sólo de las estructuras de origen sino del conjunto de entidades objeto de la migración y del alcance del primer proceso. Por ese motivo puede realizarse una planificación mucho más certera que la inicial.

Si bien la metodología no determina un conjunto de documentos obligatorios y formales, es deseable que al terminar esta fase se tengan al menos los siguientes.

- Diagrama de entidad relación (o una especificación similar) de las bases de origen identificando el subconjunto de los elementos a migrar.
- Diagrama de flujo, pseudo-código (o una especificación similar) de los procesos que componen T1 cuando éstos tengan una lógica de transformación compleja. No debe perderse de vista que el objeto principal de T1 es el de hacer una transformación sintáctica por lo tanto los programas, mapeos y demás recursos que lo compongan no deberían tener una lógica compleja. No obstante, de encontrarse algún caso de mayor dificultad es conveniente dejarlo documentado.
- Diagrama de entidad relación (o una especificación similar) de la base intermedia.
- Planificación detallada de T1. Debe tenerse en cuenta aquí que la planificación no puede abarcar todo el proceso pues está sujeto a cambios, pero al menos el documento debe contener los pasos a realizar y una estimación de tiempos hasta la primera versión ejecutable de T1.
- Otros documentos que se consideren necesarios. Puede elaborarse un documento o varios para determinar otros factores como por ejemplo los pasos a realizar para conectarse y extraer datos desde las bases de origen, el equipamiento necesario, las especificaciones de las herramientas de software a utilizar etc.

Nuevamente aquí hay que resaltar que T1 está sujeto a cambios por lo tanto no deben construirse demasiados documentos ni realizarlos de manera muy extensa. La documentación debe ser la estrictamente necesaria y estar realizada de la manera más simple posible.

Fase 3: Análisis y Diseño de T2

Con una visión clara de los alcances de T1 y el desarrollo de éste en marcha puede realizarse un análisis y diseño del proceso siguiente. Si bien T2 toma como entrada la base intermedia que es la salida de T1, no es necesario esperar a que éste se culmine y se ponga en marcha para comenzar con su análisis y diseño o incluso con su construcción.

Debe tenerse en cuenta que en T2 la dificultad técnica es menor porque las bases de origen y destino ahora se encuentran soportadas sobre la misma tecnología, pero la dificultad en las transformaciones lógicas puede ser muy significativa. Dentro de esta fase pueden aparecer procesos complejos desde el punto de vista lógico y sobre ellos debe tenerse especial atención.

Las tareas de esta fase son las siguientes:

- **Estudiar la estructura de la base de datos de salida** teniendo en cuenta que la misma está sujeta a cambios por la propia evolución del desarrollo de la aplicación que la utiliza. Se debe tomar contacto con su estructura, estudiar sus entidades e identificar los aspectos que aún están propensos a modificaciones y aquellos que resultan más estables.
- **Identificar el conjunto de entidades de la base de salida que deben poblarse con datos provenientes de la migración**. Analizar principalmente su estructura, tipos de datos, restricciones y relaciones.
- **Diseñar los procesos que van a realizar las transformaciones** leyendo datos de la base intermedia y escribiendo en las entidades de la base de destino. Ésta es la tarea central de esta fase. Nuevamente vale decir, que aquí es donde se enfrenta la verdadera transformación lógica de los datos provenientes de las fuentes originales para poder ser incluidos en la nueva aplicación. Esta tarea es el centro de la migración.
- **Realizar una planificación detallada de las actividades de construcción de T2**. Si bien en la primera fase se hace una planificación general, al igual que en el caso de T1, ahora al

tener un diseño detallado y una visión más clara de los alcances de T2 es posible elaborar una planificación más concreta y certera.

De esta manera, los documentos que se deben obtener a partir de esta fase son los siguientes:

- Diagrama de entidad relación (o un recurso similar) de la base de destino. Si este documento ya ha sido generado por el equipo de desarrollo de la aplicación, puede adaptarse a las necesidades de la migración marcando las entidades alcanzadas por ésta, agrupando por funcionalidades de la migración, aclarando tipos de datos y demás elementos que contribuyan con la construcción de T2.
- Documento de los procesos de T2. Pueden ser diagramas de flujo, pseudocódigo, especificaciones de mapeo y todo elemento que lleve a establecer la lógica de los procesos. Éstos son los documentos centrales de todo el proceso general y se les debe dar especial importancia.
- Plan detallado de T2. Con un criterio similar al aplicado en T1, no es necesario construir un plan extenso de todo el ciclo de vida de T2 porque el mismo se encuentra sujeto a cambios, pero sí deben planificarse las tareas necesarias como mínimo hasta la primera versión ejecutable de este proceso.
- También de una manera similar a lo expuesto en las fases anteriores, pueden incluirse otros documentos que se consideren relevantes. Por ejemplo, pueden detallarse los mecanismos de conexión con la base de destino, las herramientas utilizadas, el equipamiento, etc.

Fase 4: Fase Iterativa. Construcción Ejecución y Prueba de T1

Una vez diseñado y planeado T1 puede comenzarse con su construcción. El desarrollo inicialmente debe centrarse en alcanzar una primer línea base, o sea, su primera versión ejecutable para mantenerlo luego dentro de un ciclo iterativo de mejora y modificación.

La primera versión puede incluir todo el proceso de importación y generación de la base intermedia y todas las pruebas e informes de resultados planeados. En este caso el proceso iterativo posterior se limita a incluir modificaciones cuando surjan cambios en la base de salida que afecten

a T1. Como se analiza más adelante en este mismo capítulo, no todas las modificaciones en la base de datos implican alteraciones en T1.

También puede definirse la primer línea base como la migración de un subconjunto de entidades. Este grupo puede estar compuesto por elementos que resulten centrales en el resto del proceso, o que tengan un interés particular por su volumen o por su lógica u otro factor. Las características de cada proyecto son las que determinan si la primera versión de T1 incluye sólo un subgrupo de entidades y cuáles son las que componen dicho conjunto.

En este caso, el desarrollo de pruebas y la obtención de los informes también pueden realizarse por partes. Lo natural es incluir las pruebas y los informes relacionados a las entidades alcanzadas en la línea base.

Si para la primera versión de T1 se define un subconjunto, entonces el proceso iterativo posterior ya no puede limitarse a incorporar cambios, sino que debe abarcar gradualmente al resto de los objetos de origen para así hacer crecer la estructura de la base intermedia.

Lo mismo ocurre con las pruebas y los informes. A medida que se incluyen más entidades éstas deben contemplarse en los test.

A partir de lo dicho anteriormente, dentro de esta fase pueden identificarse las siguientes tareas:

- **Determinar el grupo de entidades que conforman la primer línea base**.
- **Construir la primera versión de T1**. Esta tarea puede incluir actividades de mapeo con herramientas gráficas, actividades de programación en lenguajes específicos y toda acción que lleve al desarrollo en sí del proceso.
- **Ejecutar la primera versión de T1 para obtener la primera base intermedia**.
- **Definir y Construir los informes de migración**.
- **Mantener un proceso iterativo en el que se actualice T1** para incorporar modificaciones o realizar correcciones en función de los errores detectados en los informes y según la evolución del desarrollo del sistema.

Como puede advertirse, las tareas enunciadas tienen carácter general ya que esta metodología no define la forma o los criterios que conllevan a establecer divisiones o subgrupos

dentro de T1. Tampoco determina la manera en que se organizan las actividades de desarrollo ni los documentos que se construyen. Estos factores están relacionados con las características particulares de cada proyecto, no obstante, se puede hacer esta recomendación general respecto de la organización de la secuencia de tareas:

- Incluir en la primer línea base, la migración de todas las entidades que se hayan contemplado como datos de origen al momento de comenzar el desarrollo de T1.
- Dejar el proceso iterativo como mecanismo para procesar modificaciones que surjan como consecuencia de la evolución del proceso de desarrollo o por corrección de errores relevados en los informes de migración.
- En el comienzo de cada iteración incluir los cambios y las funcionalidades nuevas que surgieron luego de comenzar la vuelta anterior y por ende no fueron incorporadas. Mantener este proceso como una tarea continua.
- Respecto de los test y los informes de migración, cada proyecto puede tener sus propias necesidades, pero en general es muy útil controlar en este nivel, como mínimo, con los siguientes controles:
    - Que todas las entidades tomadas como origen por T1 tengan su correspondiente salida en la base intermedia.
    - Que para cada una de estas entidades la cantidad de datos en la fuente de origen sea la misma que la obtenida en la base intermedia. Como las estructuras de datos iniciales pueden diferir en su arquitectura con la base intermedia, este control puede resultar complejo. Por ejemplo, es probable que la cantidad de datos de un archivo de texto separado por tabulaciones deba coincidir con la cantidad de registros de una tabla o incluso de varias.
    - Si hay registros rechazados, o sea, que formando parte del origen no se pasan a la base intermedia, verificar su estado y la razón por la que fueron excluidos. Es probable que no deban ser tenidos en cuenta en la migración, pero también puede ocurrir que los datos de origen requieran alguna corrección o que la lógica de T1 tenga que ser revisada.
    - Verificar posibles faltas de integridad que no pueden luego subsanarse en la base de destino. Si bien el pasaje a la base final la realiza T2, aquí pueden preverse algunos factores de manera anticipada, como por ejemplo faltas en futuras reglas de integridad,

tipos de datos que resultarán en T2 imposibles de convertir, longitudes que serán imposibles de adaptar, etc.

Por último, la documentación a elaborar dentro de esta fase depende principalmente de cómo se gestione el propio desarrollo de T1. Eso escapa al plateo metodológico actual que tiene su foco en la migración como proceso. Más adelante, en el capítulo seis se plantean posibles trabajos futuros, algunos de ellos incluyen un análisis de metodologías a emplear dentro de los procesos internos de la migración.

Fase 5: Fase Iterativa. Construcción Ejecución y Prueba de T2

A partir de la finalización de la Fase 3, se tiene un diseño del proceso T2 por lo tanto puede comenzarse con su construcción. Uno de los principales aspectos a definir es el momento en el que resulta más conveniente iniciar esta etapa.

Este proceso realiza principalmente la conversión semántica más compleja. Dentro de T2 ocurre la verdadera adaptación de los datos partiendo de las reglas de los sistemas de origen para llegar a las reglas del nuevo sistema. Desde este punto de vista conviene comenzar la etapa lo más temprano posible. En rigor, la fase 5 puede comenzar ni bien se finalice la fase 3, o sea, se puede dar inicio a la construcción, ejecución y prueba de T2 ni bien se termine con su diseño.

A la vez, debe tenerse en cuenta que la principal entrada de este proceso es la base intermedia generada por T1 y que, si no se dispone de la misma, no podrá ejecutarse ni probarse ningún aspecto de T2. Desde ese punto de vista, resulta conveniente comenzar con T2 luego de tener al menos una primera versión ejecutable de T1.

La metodología no establece un momento específico. Es este aspecto, las características propias de cada proyecto son las que determinan el punto de inicio correcto para esta etapa. En todos los casos, se trata de alcanzar un equilibrio entre la necesidad de anticipar el desarrollo de T2 y al mismo tiempo contar con una base de entrada para realizar pruebas en firme.

Al igual que con T1, la primera versión ejecutable de T2 no tiene que contener necesariamente todas las entidades que componen la base de salida. Pueden alcanzarse sólo un subgrupo en función de algunos criterios. Por ejemplo, se le puede dar prioridad a las entidades

más relevantes por su complejidad o por su volumen o incluir aquellas que resultan más importantes para realizar pruebas sobre el desempeño de la aplicación cuando opera con datos migrados, o simplemente incorporar las entidades que se encuentren disponibles en la base intermedia generada por T1.

A partir de lo dicho anteriormente, dentro de esta fase pueden identificarse las siguientes tareas:

- **Determinar el grupo de entidades de la base de datos intermedia que conforman la primera línea base**.
- **Construir la primera versión de T2**. Esta tarea puede incluir actividades de mapeo con herramientas gráficas, actividades de programación en lenguajes específicos y toda acción que lleve al desarrollo en sí del proceso.
- **Ejecutar la primera versión de T2 para obtener la primera base de salida.**
- **Definir y Construir los informes de migración**.
- **Mantener un proceso iterativo en el que se actualice T2** para incorporar modificaciones o realizar correcciones en función de los errores detectados en los informes y de la evolución del desarrollo del sistema.

Al igual que en el caso de T1 (y por las mimas razones) la metodología no determina criterios fijos para incluir o excluir entidades en el desarrollo de T2, pero pueden hacerse estas recomendaciones generales:

- Incluir en la primer línea base de T2 todas las entidades que se hayan contemplado en la construcción de la base intermedia hasta ese momento.
- Dejar el proceso iterativo como mecanismo para realizar modificaciones que surjan como consecuencia de la evolución del desarrollo o por corrección de errores relevados en los informes de migración o por la incorporación de nuevas entidades en la base intermedia.
- En el comienzo de cada iteración incluir los cambios y las funcionalidades nuevas que surgieron luego de comenzar la vuelta anterior y por ende no fueron incorporadas. Mantener este proceso como una tarea continua.

- Respecto de los test y los informes de migración, cada proyecto puede tener sus propias necesidades, pero en general es muy útil controlar en este nivel, como mínimo, los siguientes controles:
    - Que todas las entidades tomadas como origen desde la base intermedia tengan su correspondiente salida en la base de destino final. Ésta puede ser una tarea compleja pues no necesariamente hay una relación uno a uno. Un conjunto de datos de la base intermedia (una tabla por ejemplo) puede tener equivalencia en más de una entidad de la base final y viceversa.
    - Que, para cada una de estas entidades, la cantidad de datos en la base intermedia sea la misma que la obtenida en la base de salida. Como T2 realiza una conversión semántica, las estructuras pueden resultar completamente diferentes. Un registro en una tabla de la base intermedia puede implicar uno o varios en la de base de salida y también a la inversa. Los informes que se elaboren en este sentido deben controlar elementos conceptuales, por ejemplo, cantidad de personas, empleados, cargos, etc. independientemente de cómo se almacenen físicamente.
    - Si hay registros rechazados, o sea, que formando parte de la base intermedia no se pasan a la base de destino, verificar su estado y la razón por la que fueron excluidos. Es probable que no deban ser tenidos en cuenta en la migración, pero también puede ocurrir que los datos de origen requieran alguna corrección por no respetar reglas de la base de destino o que la lógica de T2 tenga que ser revisada.
    - Que más allá de las reglas que pueden fijarse en la base de datos de destino (integridad referencial, restricciones de unicidad, restricciones chek, etc.) los datos cumplen las reglas de negocio de la aplicación. Éstos pueden ser chequeos muy amplios dependiendo de la complejidad de dichas reglas de negocio. Si no es factible dar una cobertura del total de las reglas, al menos verificar el cumplimiento de las más importantes e ir agregando más validaciones con cada iteración en función de los problemas detectados.

Una vez alcanzada la primer línea base de este proceso, se puede ejecutar para obtener la base de datos de salida. En este punto se alcanza por primera vez un ciclo completo de toda la migración y se obtiene una base de salida que se ajusta a las necesidades del nuevo sistema, pero

conteniendo los datos generados por las aplicaciones anteriores. Es el momento correcto para realizar las pruebas y construir los informes de migración de esta etapa.

Al igual que como se dijo en el caso de T1, la documentación a elaborar dentro de esta fase depende principalmente de cómo se gestione el propio desarrollo de T2. Eso escapa al planteo metodológico actual que tiene su foco en la migración como proceso. La aplicación de metodologías de desarrollo dentro de T1 y T2 está planteada como posibles trabajos de investigación futuros en el capítulo seis.

Mapa de las fases de la metodología de tres niveles

A partir de las fases y grupos de tareas presentados en los apartados anteriores, puede construirse el siguiente diagrama que muestra la estructura general del a metodología.

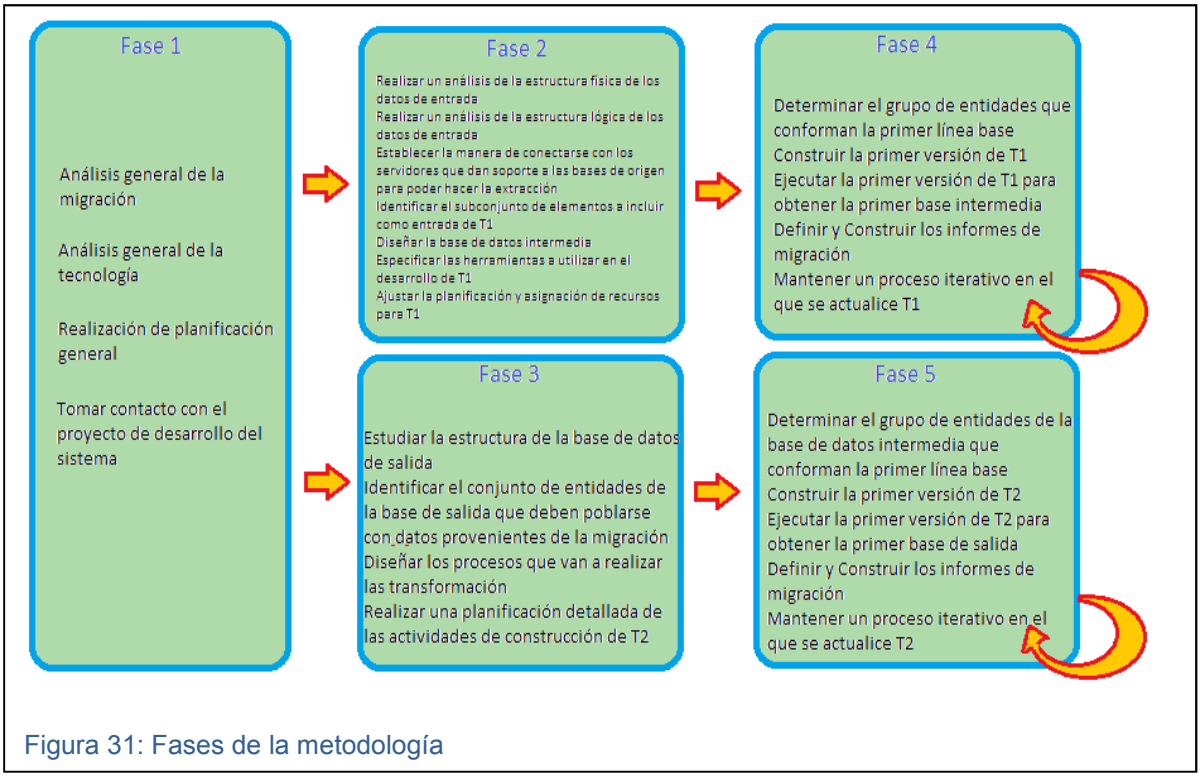


Figura 31: Fases de la metodología

Como puede advertirse en la figura, las fases dos y tres pueden desarrollarse en paralelo, aunque como ya se ha dicho, no necesariamente debe ser así y además puede traslaparse en algunos

momentos o en forma completa. Sí es importante que ambas comiencen luego de terminar la primera fase ya que ésta da el marco general.

Las fases cuatro y cinco deben comenzar luego de la dos y la tres respectivamente y pueden desarrollarse en paralelo también en forma completa o parcialmente al igual que en el caso anterior.

Estas dos últimas etapas son las que luego de la primera ejecución entran en un ciclo iterativo donde se incorporan nuevas funcionalidades y se adaptan a los cambios provenientes de la propia migración o del proceso de desarrollo del sistema.

Es importante notar que en la fase cuatro y cinco no se han incorporado explícitamente tareas de planificación y diseño, pero obviamente, dependiendo de las características de los cambios procesados puede ser necesario ajustar las planificaciones y los diseños originales. En este sentido, y como ya se ha dicho antes, la metodología no define un grupo de tareas estrictas, sino que da un marco que luego debe adaptarse a las necesidades de cada proyecto. Esta flexibilidad constituye un elemento fundamental pues la hace adaptable a los cambios y ésa es su razón de ser.

Relación entre la metodología de tres niveles y la arquitectura de procesos divididos

Hasta aquí en este capítulo, se ha presentado por un lado una arquitectura que propone dividir los procesos típicos de una migración y por otro una metodología que plantea fases especialmente diseñadas para el contexto de estudio. Ahora hay que analizar la relación entre ambas para visualizarlas como una única propuesta.

La relación viene dada por las siguientes reglas.

- Toda acción que lleve a planificar, diseñar, programar o realizar cualquier operación que guarde relación con la realización de una conversión sintáctica, debe realizarse sobre T1 y alcanza principalmente a las fases dos y cuatro.

- Toda acción que lleve a planificar, diseñar, programar o realizar cualquier operación que guarde relación con la realización de una conversión semántica, debe realizarse sobre T2 y alcanza principalmente a las fases tres y cinco.

De este modo se debe concluir que la fase uno es de planificación y organización general y, por ende, es transversal a toda la arquitectura. Las fases dos y cuatro se realizan sobre T1 tomando como entrada las fuentes de datos de origen y como salida la base intermedia y las fases tres y cinco se desempeñan sobre T2 tomando como entrada la base intermedia y como salida la base de destino.

Lo antedicho puede exponerse en la siguiente figura.

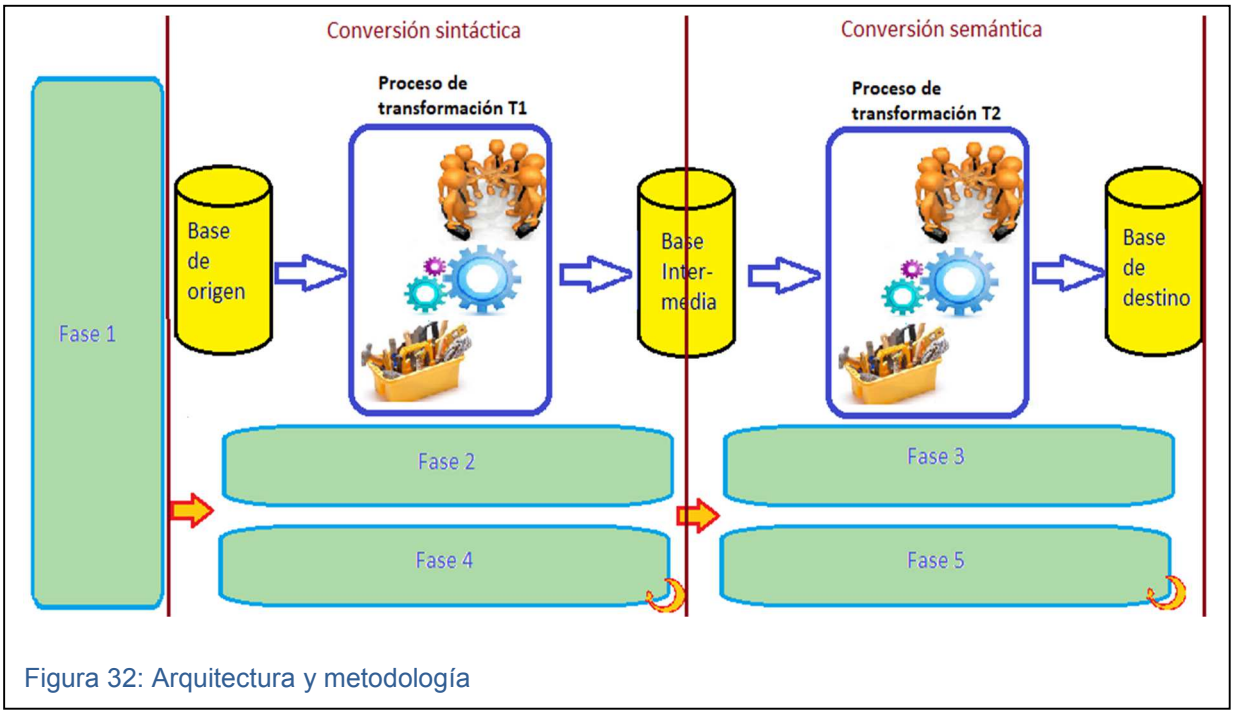


Figura 32: Arquitectura y metodología

En el gráfico anterior puede advertirse que las fases cuatro y cinco son las que realizan el proceso iterativo que constituye el aspecto más importante en la migración cuando se emplea esta metodología. Más adelante en este mismo capítulo se clasifican las posibles modificaciones según afecten a un proceso u otro.

En la práctica, las divisiones entre T1 y T2 pueden no ser muy claras. Por ejemplo, puede ocurrir que se incorpore como parte de T1 un informe de migración que estrictamente no guarda

relación con la sintaxis de los datos pero que ayuda a detectar en forma temprana la existencia de registros que serán rechazados por las reglas de T2.

Más concretamente y para citar un caso particular, podrían incluirse en T1 informes que muestren registros que desde el punto de vista sintáctico se pudieron leer, convertirse y grabarse en la base intermedia pero que no van a cumplir reglas de integridad referencial que son exigidas en T2.

El mismo informe podría formar parte de T2 ya que no se encuentra relacionado estrictamente con elementos sintácticos, pero debería ser ejecutado antes de lanzar el proceso de lectura y transformación para poder dar tratamiento a esos registros en la base intermedia y tomar acciones sobre ellos.

Las acciones pueden ser muy variadas. Podría tratarse de un error en las fuentes de origen reparable o no, podría tratarse de un error en la lectura de los datos de origen realizada por T1, podría tratarse de un problema en la lógica del propio proceso de transformación de T1, etc.

Como ya se ha dicho antes al analizar otros aspectos, cada proyecto tiene sus características y necesidades particulares, por esta razón la metodología no impone reglas estrictas sobre la forma en la que se dividen las tareas en sus fases y la relación entre éstas y la arquitectura física, pero si debe quedar perfectamente claro que las transformaciones sintácticas deben realizarse sólo en T1 y que, a su vez, este proceso no debe incluir transformaciones de otro tipo. Lo mismo es aplicable para T2 respecto de las transformaciones semánticas.

Elementos a incluir en cada iteración de las fases cuatro y cinco

En cada iteración se incluyen nuevos elementos en el desarrollo tanto de T1 como de T2. Se debe destacar que ambos ciclos no tienen relación, es decir, puede iterarse más rápidamente sobre T1 o sobre T2 y el lanzamiento de un nuevo ciclo sobre uno de los procesos de transformación no implica hacer lo mismo sobre el otro.

Los aspectos a incluir en cada iteración van a depender principalmente de la dinámica de la migración en cada implementación en particular y también de la dinámica del proceso de

desarrollo del sistema, pero siempre van a responder a alguno de estos tres tipos analizados a continuación.

Corrección

Una corrección ocurre cuando se detecta un error en el proceso de migración y ésta debe arreglarse ya sea dentro de la misma iteración o en alguna siguiente. En este sentido, los informes de migración son importantes porque, entre otros factores, permiten detectar posibles defectos en las lecturas, transformaciones o escrituras.

Por ejemplo, puede descubrirse que la cantidad de registros de una entidad de origen no coincide con la cantidad que se obtienen en la base de destino, o que algunos registros tienen un valor en un campo que no corresponde al dato de la fuente de origen, o también puede pasar que algunos registros de la base de salida no cumplen con restricciones de integridad, etc.

En todos esos casos es necesario **corregir** el proceso para que los datos en la base de destino alcancen el formato deseado y reflejen la misma información que contiene la base de origen.

Debe notarse que, aunque se deban hacer cambios para alcanzar la corrección, los mismos no provienen de modificaciones del sistema en desarrollo, sino que son consecuencias de errores de la propia migración. Dicho de otra manera, su origen es interno y de allí su nombre de "corrección".

Ampliación

Una ampliación ocurre cuando se incorporan en una iteración nuevas funcionalidades.

Puede ocurrir que las mismas ya hayan sido planificadas y tenidas en cuenta en el desarrollo de la migración y que como parte de la evolución natural sean incluidas en alguna determinada iteración. También puede pasar que sean consecuencia del avance en el desarrollo del sistema o de la incorporación de nuevas funcionalidades en éste.

Por ejemplo, si un cambio de requerimientos del sistema en desarrollo conlleva a agregar nuevas entidades en la base de datos y éstas deben llenarse con información proveniente de los

sistemas a ser reemplazados, el proceso de migración tiene que ser **ampliado** para incorporar los nuevos elementos.

En el caso de una ampliación, la razón puede ser tanto interna como externa. Es interna cuando surge del desarrollo de la migración como parte su ciclo iterativo natural y es externa cuando su origen radica en la evolución del desarrollo del sistema.

Modificación

Una modificación tiene lugar cuando la base de datos de salida ha sufrido cambios y la migración debe adaptarse a éstos.

Tal como se ha mencionado antes, el propio desarrollo del sistema suele imponer cambios en la base de datos de salida. Esto puede ocurrir como consecuencia de cambios de requerimientos, o por mejoras de rendimiento, o por correcciones en el diseño del sistema, o por correcciones en el diseño de la base de datos, etc.

Por ejemplo, puede pasar que se decida redundar un campo en una tabla para mejorar el rendimiento de algunas consultas, o que un cambio de requerimiento lleve a modificar un tipo de datos o cambie relaciones obligando a crear entidades intermedias.

Más allá de la razón que se tenga para modificar la base de destino, lo cierto es que si dicho cambio afecta entidades que están bajo el alcance de la migración, ésta tiene que **modificarse** en consecuencia.

En el caso de una modificación, la razón es siempre externa porque está íntimamente relacionada a cambios en la base de datos de destino cuya definición se encuentra fuera del alcance de la migración.

Proceso de un cambio de requerimiento en la metodología de tres niveles

Independientemente del tipo de cambio que se trate, sea una corrección, ampliación o modificación, lograr un procesamiento eficiente es la razón de ser de la metodología y de la

arquitectura propuesta. Más aún si se trata de ampliaciones o modificaciones donde el origen es externo.

Algunos cambios pueden afectar sólo a T1, otros a T2 y otros a ambos. En función de esto requieren un tratamiento diferente y por eso es importante clasificarlos tal como se hace a continuación.

Cambios tipo uno

Un cambio es tipo uno cuando sólo afecta al proceso T1 pero sin que se tenga necesidad de cambiar la estructura de la base de datos intermedia por lo tanto, T2 no sufre consecuencias.

Puede implicar incorporar más información de las fuentes de origen teniendo como destino entidades de la base intermedia que ya han sido creadas lo que constituye una ampliación, o alterar las reglas de transformación de algunos datos originales lo que puede verse como una modificación, o realizar correcciones por errores detectados en los informes de migración del mismo T1 y en ese caso se tratarse de una corrección, etc.

Este tipo de cambio debe procesarse en alguna iteración de T1. Puede ser la inmediata siguiente o alguna posterior dependiendo de la urgencia y de la forma en que se organice la migración en general.

Los cambios tipo uno, normalmente llevan a un replanteo de elementos sintácticos como alteraciones en longitudes de campos, modificaciones en conversiones de tipos, lectura y transformación de nuevas entidades desde las bases de origen y demás acciones similares. En consecuencia, suelen no requerir un análisis muy profundo y su dificultad se encuentra más relacionada a elementos técnicos que funcionales.

Cambios tipo dos

Un cambio es tipo dos cuando sólo afecta el proceso T2 y no tiene ninguna implicancia en T1. Son cambios que pueden procesarse tomando como origen la base intermedia sin que sea necesario hacer sobre ella ninguna modificación y por lo tanto T1 no se ve afectado.

Puede tratarse de la incorporación de nuevas funcionalidades en la migración que lleven a cubrir más entidades en la base de destino, lo que se enmarca dentro de una ampliación, o centrarse en la corrección de un error detectado por los informes de migración del propio T2 lo que sería una corrección o la razón del cambio puede estar relacionada con una modificación en los requerimientos del sistema que ha llevado a cambiar algunas entidades de su base de datos haciendo que T2 deba adaptarse y en este caso tratarse de una modificación, sólo por mencionar algunas situaciones posibles.

A la inversa del caso anterior, este tipo de cambios sólo afectan a T2 y por lo tanto pueden incorporarse en cualquier iteración de la fase cinco sin guardar sincronismo con el avance de T1. Esta independencia simplifica el proceso.

Un aspecto importante de los cambios tipo dos es que siempre se encuentran relacionados a elementos semánticos que frecuentemente son los más complejos de resolver y por eso no debe sorprender que su procesamiento sea más lento y demande una mayor cantidad de recursos.

Dicho de otro modo, puede esperarse que cambiar tipos de datos, rehacer lecturas de archivos de distintos orígenes o modificar conversiones de longitudes, etc., sin ser una tarea simple, resulte menos complejo que, por ejemplo, evaluar una nueva regla de negocio de la aplicación en construcción para lograr que determinadas entidades de la base de datos cumplan con ellas o modificar procesos para que sean capaces de identificar los datos de la base intermedia y clasificarlos según las definiciones del nuevo sistema.

Cambios tipo tres

Un cambio tipo tres es aquel que afecta tanto a T1 como a T2. Para procesarlo deben modificarse ambos procesos pues resulta necesario modificar la base intermedia para luego tomar de allí esas entidades e impactarlas en la base de destino.

Algunos casos posibles son los siguientes:

- Puede ocurrir que se incorporen nuevas funcionalidades al sistema como producto de un cambio de requerimientos y que esto lleve a la incorporación de nuevas entidades en la base de datos. Si éstas deben llenarse con información de sistemas anteriores pero las

fuentes no han sido tomadas dentro de los orígenes evaluados inicialmente, tendrá que modificarse T1 para hacer las correspondientes lecturas y transformaciones e incorporar esta información a la base intermedia y también T2 para realizar las conversiones semánticas y finalmente afectar a la base de destino. En este caso se trata de una ampliación.

- Pueden detectarse errores en los informes de migración de T2 que no pueden repararse sin realizar cambios en la base intermedia. Bajo esta situación tiene que modificarse primero T1 para lograr las correcciones en dicha base y luego T2 para mejorar su proceso en función de los nuevos resultados de T1. En este caso se trata de una corrección.
- También puede pasar que un cambio en el sistema implique agregar datos a entidades modificando su estructura, típicamente esto es, agregar nuevos campos en tablas ya existentes. Si alguna de estas entidades no existe en la base intermedia, debe leerse primero desde las fuentes de origen y eso afecta a T1. Una vez incorporados los datos en la base intermedia puede modificarse T2. Si no se escriben procesos nuevos, sino que se modifican algunos ya existentes, se está en presencia de una modificación.

Los tres ítems anteriores sólo nombran algunas posibilidades a modo de ejemplo, pero podrían encontrarse muchas más.

Los cambios tipo tres son los menos deseables ya que involucran a los dos procesos y por lo tanto, es necesario llevarlos a cavo guardando alguna sincronización entre las iteraciones de T1 y T2. En otras palabras, estos cambios no pueden resolverse en forma independiente como sí ocurre en los tipos uno y dos. Esto agrega aún más dificultad.

Si al desarrollar T1 se incorporan en la base intermedia sólo las entidades de las fuentes de origen que estrictamente son necesarias en la base de destino, muchos cambios resultan luego ser tipo tres pues cualquier acción que implique incorporar datos, por más mínimos que sean, requiere modificar la base intermedia.

En este sentido puede resultar conveniente tener un criterio amplio al momento de diseñar T1 e incluir en su tratamiento entidades relacionadas a las que deben ser tenidas en cuenta en la migración, aunque en rigor no sean necesarias. De este modo se tendrán más cambios tipo dos.

Al mismo tiempo, no es conveniente incluir en T1 cuanto conjunto de datos se pueda sin ningún tipo de control ni criterio porque esto agrega un volumen importante al proceso de transformación y genera una base intermedia grande y compleja. De este modo se tienen menos cambios tipo tres, pero a un alto costo.

En definitiva, lo conveniente es tener un criterio amplio al momento de diseñar T1, considerando alguna previsión sobre elementos que probablemente sean alcanzados por la migración, pero sin complicar demasiado su estructura ni agrandar irracionalmente la base intermedia.

En el capítulo siguiente se analizan algunos ejemplos de cambios de distintos tipos realizados durante la implementación que allí se expone.

La figura siguiente es un mapa conceptual de los tipos de cambios ya sean tipo uno, dos y tres y su relación con la inclusión de elementos de corrección, de ampliación o de modificación junto con los procesos que afectan.

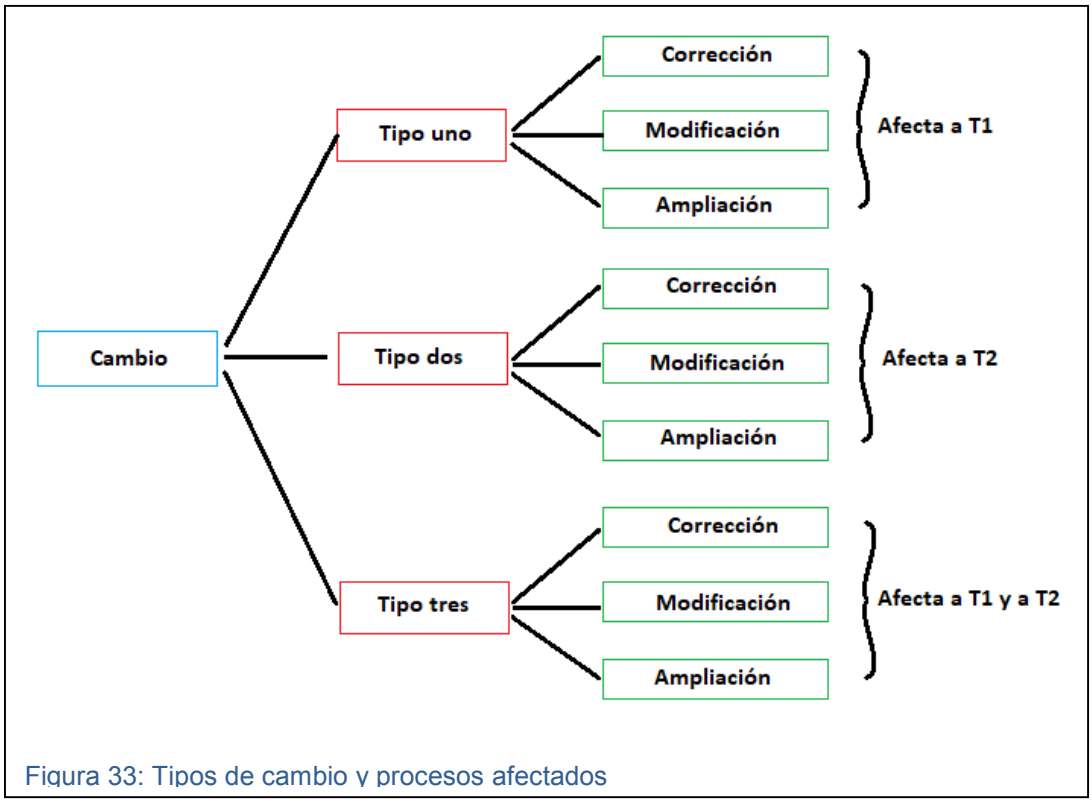


Figura 33: Tipos de cambio y procesos afectados

Herramientas que pueden utilizarse en las diferentes etapas

La metodología propuesta es independiente de la plataforma, es decir, no ha sido diseñada para un producto o un grupo de productos en particular ni de un determinado fabricante. Al mismo tiempo brinda una gran flexibilidad respecto de las herramientas a utilizar dentro del desarrollo de sus procesos como así también de la construcción de las pruebas.

Por estas razones es posible utilizar todo tipo de herramientas, ya sean las analizadas en el capítulo anterior u otras. También debe tenerse en cuenta que no necesariamente deben usarse las mimas para el desarrollo de T1 y para T2. Cada proceso puede tener sus propias necesidades y la metodología permite emplear las herramientas que mejor se adapten sin que exista relación entre ambos grupos.

Las características de cada proyecto de migración son las que determinan el grupo de herramientas a emplear en cada etapa. Entre otros factores, sin duda son de gran incidencia elementos como la tecnología de las fuentes de origen, la tecnología que soporta la base de destino, las destrezas y conocimientos del personal afectado, la complejidad de las conversiones, el volumen de datos a migrar, etc.

Así, por ejemplo, si la base de destino es DB2 y los orígenes son múltiples podría emplearse TDMF (Transparent Data Migration Facility) (34) tanto en T1 como en T2 haciendo uso de sus características gráficas en T1 y apoyándose en elementos de programación dentro de T2.

Si la base de destino fuera SQL Server y los orígenes respondieran a múltiples plataformas, podrían utilizarse paquetes SISS (Sql Server Integration Services) (33) para T1 y también para T2 apoyándose en rutinas programadas en lenguaje T-SQL.

Si toda la plataforma estuviera apoyada fuertemente en Oracle podría emplearse Oracle SQL Developer Migration Workbench (29) de punta a punta en todo el proceso, desarrollando algunas rutinas en lenguaje PL-SQL.

De este modo, se podrían plantear muchísimas configuraciones posibles porque tal como se analizó en el capítulo anterior, las herramientas existentes son numerosas. Lo importante es

hacer un análisis con un criterio amplio y optar por un conjunto que resulte útil en el desarrollo tanto de T1 como de T2.

Como en T1 se realizan conversiones sintácticas, se toman datos de fuentes que muchas veces son heterogéneas y se cuenta con libertad para diseñar la estructura de la base intermedia, suelen ser muy efectivas las herramientas con mayor uso de interfaces gráficas, es decir, las que permiten hacer mapeos, lecturas, conversiones y escrituras sin escribir rutinas programadas o desarrollando sólo algunas.

En el caso de T2 la situación es diferente. Como allí se hacen conversiones semánticas, la base de origen es una sola con la misma plataforma que la de salida y además no se tiene libertad para diseñar la base de destino, suelen ser muy efectivas las herramientas que se apoyan fuertemente en leguajes de programación y que permiten desarrollar rutinas que leen, transforman y graban datos.

En definitiva, tal como ya se ha dicho, son las características de cada migración y no la metodología las que determinan las herramientas a emplear. Lo importante es tener un criterio amplio y disponerse a emplear herramientas diferentes según las características de T1 y de T2.

Fortalezas y debilidades de la metodología

Al igual que se hizo en el capítulo anterior con todas las otras metodologías analizadas, deben considerarse las fortalezas y debilidades de la nueva propuesta.

Estas fortalezas y debilidades guardan estrecha relación con su razón de ser, debe siempre tenerse en cuenta que se trata de una metodología diseñada para ser utilizada en un contexto particular.

Fortalezas de la metodología de tres niveles.

- En general, toda la metodología está pensada para llevar adelante una migración en paralelo al desarrollo del sistema que emplea la base de datos de destino. Asume en todas sus fases y tareas que el destino es inestable.
- Divide al proceso en dos haciendo que toda la migración resulte más adaptable a los cambios.
- Propone fases iterativas que permiten hacerla evolucionar en función del avance del proceso de desarrollo del sistema.
- Las fases iterativas son independientes una de otra, por lo tanto, pueden evolucionar en paralelo sin condicionarse mutuamente.
- Tiene una primera fase de planificación general que posibilita alcanzar una visión global antes de entrar en detalles técnicos.
- Es independiente de la plataforma. Puede emplearse para migrar datos con orígenes y destinos de tecnologías heterogéneas.
- Permite emplear herramientas de diferentes fabricantes en cada uno de sus procesos. Ninguno de sus aspectos condiciona el conjunto de herramientas a utilizar en cada implementación.
- Prevé la realización de pruebas y la elaboración de informes de migración en sus dos procesos. También define la necesidad de incorporar los cambios que se deriven de dichas pruebas en las siguientes iteraciones.
- Las definiciones de las tareas en cada etapa constituyen una recomendación, con lo que pueden agregarse o quitarse tareas en función de las necesidades de cada proyecto.
- En las etapas iterativas, que sin dudas constituyen su aspecto más relevante, no define mecanismos para gestionar cada iteración. Esto permite emplear recursos de todo tipo dentro del proceso iterativo en función de las necesidades de cada proyecto.

Debilidades de la metodología de tres niveles

- Al estar especialmente diseñada para migraciones que se ejecutan en paralelo con el desarrollo de un sistema y con una base de destino inestable, su uso puede no ser

recomendable en otros contextos porque pierden sentido las tareas y las fases plateadas. En otras palabras, es una metodología para un uso específico.

- Sus fases iterativas imponen la necesidad de gestionar correctamente cada iteración definiendo los elementos que se incorporan y planificando reiteradamente sus alcances, recursos, tiempos de ejecución, etc.
- No es una metodología estándar ni muy difundida por lo que su empleo requiere capacitar especialmente al personal afectado. Si se incorporan recursos humanos en medio del proyecto será crucial destinar esfuerzos a su formación antes de que puedan intervenir eficazmente. Las personas involucradas deben comprender muy bien la manera en funciona el modelo.

Cuadro de comparación

En el capítulo anterior, al comparar las diferentes metodologías, se construyó un cuadro sobre algunas de sus características relevantes. Más allá de las ventajas y desventajas mencionadas, resulta conveniente situar la metodología de tres niveles iterativa dentro del mismo cuadro.

Por razones de simpleza en la lectura, no se reiteran aquí todas las otras metodologías, sino que sólo se incluye una fila en el cuadro de comparación a los efectos de analizar la propuesta.

Metodología	Tareas de organización y planificación	Nivel de detalle	Complejidad	Tareas de pruebas y validación de datos	Dependencia de la plataforma	Adaptabilidad a los cambios
De tres niveles iterativa	La metodología es muy completa. Prevé una fase de planificación y define tareas del mismo tipo, pero más detalladas tanto dentro de T1 como en T2.	Define todo el proceso en cinco fases. El nivel de detalle es importante aunque la necesidad de flexibilidad impide acotar las tares estrictamente	Complejidad alta. El plateo parte de una división de procesos e impone la utilización de una arquitectura especial, una base intermedia más dos procesos iterativos.	Da una buena cobertura en este aspecto. En sus dos procesos prevé la realización de pruebas y la evaluación de los informes de migración.	Es totalmente independiente de la plataforma. Puede emplearse con cualquier base de origen o de destino y permite utilizar herramientas de cualquier fabricante en sus dos procesos.	Muy adaptable. La adaptabilidad a los cambios constituye su propia razón de ser.

Conclusiones

En este capítulo se analizaron los problemas frecuentes que presentan las metodologías clásicas al ser utilizadas cuando la base de datos de destino es inestable en su estructura. Estas dificultades guardan estrecha relación con la manera en que se organizan las etapas y las tareas ya que resultan ser poco adaptables a los cambios.

También influye directamente en la escaza adaptabilidad la arquitectura física clásica de un proceso de migración ya que normalmente se resuelven juntos los problemas sintácticos y semánticos. En el contexto de estudio esto eleva la complejidad y hace más difícil el procesamiento de cambios.

Luego se presentó el modelo de procesos divididos que propone separar las conversiones físicamente a través de una arquitectura que identifica dos procesos llamados T1 y T2 vinculados por una base intermedia. T1 realiza las conversiones sintácticas y T2 las semánticas. De este modo los cambios no siempre afectan a ambos y pueden realizarse más simplemente haciendo que el proceso general sea más adaptable. En este sentido los cambios se han clasificado en tipo uno cuando afecta sólo a T1, tipo dos cuando afecta sólo a T2 y tipo tres cuando afecta a ambos.

A partir del empleo de este modelo físico puede definirse la metodología de tres niveles iterativa. Está compuesta por cinco fases cada una de ellas con sus respectivas etapas para dar cobertura desde las actividades de planificación y diseño hasta las de prueba y verificación pasando por todas las tareas de desarrollo técnico de conversiones sintácticas y semánticas.

Sobre el final de este capítulo pueden verse las ventajas y desventajas de la utilización de la metodología que se propone.

Con todo esto, la principal conclusión que puede obtenerse es que las metodologías clásicas no se acoplan a las migraciones que ocurren en el contexto de estudio por resultar poco adaptables a los cambios en la base de destino. De hecho, parten del supuesto de que tanto la base de origen como la de destino son estables, pero justamente eso es lo que no ocurre en el entorno analizado.

La metodología propuesta resulta más adaptable, aunque debe notarse que en algunos aspectos esto la hace más compleja, con mayor demanda de recursos técnicos para dar soporte a

la arquitectura física en que se basa y más dependiente de actividades de panificación y gerenciamiento para poder gestionar los procesos iterativos.

La propuesta no está dirigida a una metodología más simple o más liviana o económica, sino que apunta a lograr un proceso completo más adaptable, que permita gestionar los cambios eficientemente y de ese modo poder llevar a cabo una migración, aunque la base de destino resulte inestable. Está claro que la propuesta no es de uso general, es más, podría incluso afirmarse que no resulta conveniente su utilización fuera del contexto de estudio.

En varios párrafos de esta tesis, sobre todo en el capítulo dos, pero también en el presente, se citan estudios de diferentes fuentes que revelan cifras alarmantes de fracasos o excesos de consumo de recursos en los procesos de migración. Si a esas situaciones complejas se le agrega una base de destino inestable se obtiene una posibilidad de fracaso muy alta. De allí la importancia de la metodología propuesta.

Para terminar este capítulo, y a modo de conclusión final puede afirmarse que si una migración debe realizarse mientras se diseña y construye la base de destino, el equipo a cargo de ella no puede ignorar los cambios, no puede disminuir su complejidad, tampoco puede disminuir la complejidad propia del proceso de migración, no puede hacer que no haya cambios en los requerimientos del sistema ni anticiparse a ellos ni preverlos, pero si puede gestionar el proceso apoyándose en una estructura de trabajo que le permita absorber los cambios con el menor impacto posible y darles curso eficientemente. Esa es la metodología de tres niveles iterativa.

Capítulo V: Aplicación de la metodología propuesta

Introducción

En los capítulos anteriores se presentan diferentes metodologías y herramientas para migraciones de datos propuestas por diversos fabricantes. Luego se analiza la problemática común que éstas tienen dentro del contexto de estudio y se elabora una propuesta especialmente adaptada a las migraciones que ocurren paralelamente al desarrollo de un nuevo sistema.

En el presente capítulo se exponen las tareas realizadas y la experiencia general obtenida a partir de la implementación de la metodología propuesta en un caso real. Se analizan las ventajas de su utilización, los resultados obtenidos, las iteraciones realizadas, etc.

El caso de estudio se basa en una migración con alto grado de complejidad en sus transformaciones, un gran volumen de datos, más de una base de origen y un escenario de inestabilidad en la base de destino como consecuencia del desarrollo de un nuevo sistema de información complejo.

El objetivo de este capítulo es mostrar la metodología en marcha, validar su desempeño en un caso real para visualizar sus ventajas y desventajas, pero ya no desde su definición y aspectos teóricos sino desde un caso de implementación concreto.

Presentación del caso de implementación

Proyecto de desarrollo del nuevo sistema

La Cámara Nacional Electoral (CNE) tiene competencia en todo el territorio de la Nación. Este Tribunal electoral integra el Poder Judicial de la Nación y es la autoridad superior de aplicación de la legislación político-electoral, cumpliendo un rol esencial en todo lo relativo a la organización de los procesos electorales. Dentro de sus funciones registrales y de fiscalización específicas se encuentran: El Registro Nacional de Electores (RNE), el Registro de Cartas de Ciudadanía, el Registro de Electores Residentes en el Exterior, el Registro de Electores Privados

de Libertad e Inhabilitados, el Registro de Infractores al deber de votar, el Registro Nacional de Partidos Políticos, el Registro Nacional de Afiliados a los Partidos Políticos, el Registro de Empresas de Encuestas y Sondeos de Opinión, el Registro Público de Postulantes a Autoridades de Mesa, el Registro Nacional de Divisiones Electorales, el Registro de Delegados de la Justicia Nacional Electoral y el Cuerpo de Auditores Contadores. (39)

Para desempañar muchas de las funciones mencionadas, actualmente se cuenta con un sistema que podría llegar en los próximos años al límite de su capacidad y que opera de manera descentralizada en cada una de las 24 delegaciones del Juzgado Electoral. Por estas razones en el año 2014 la CNE encargó a la Universidad Tecnológica Nacional Facultad Regional Córdoba que por medio de su Centro de Investigación y Desarrollo de Sistemas (CIDS) realice una reingeniería de sus procesos y posteriormente desarrolle un nuevo sistema que se denominó Sistema de Gestión Electoral (SGE).

Este sistema tiene por objetivo la gestión de novedades que permiten mantener actualizado el Registro Nacional de Electores del país, en todos sus distritos, el cual almacena la siguiente información de los ciudadanos Argentinos (39):

- Datos personales.
- Documentos cívicos (ejemplares).
- Datos filiatorios.
- Datos de residencia y ubicación georeferencial.
- Datos de circuitos electorales.
- Situaciones o estados electorales.
- Documentación digital respaldatoria.
- Datos biométricos.

Toda esta información representa un gran volumen de datos que en muchos aspectos ya ha sido generada y almacenada por los sistemas que van a ser reemplazados cuando SEG se implemente. Naturalmente responde a la estructura propia de estos sistemas, pero debe ser incorporada a la nueva base de datos antes de la puesta en marcha.

No resulta factible poner en funcionamiento el nuevo sistema de gestión electoral sin migrar la información residente en las bases actuales. Como la migración debe estar concluida al momento de la puesta en marcha, se debe comenzar con ella de manera anticipada, sin que el desarrollo del sistema esté concluido. Esto da origen a un proyecto de migración de datos que se lleva adelante en paralelo al desarrollo del sistema y que debe adaptarse constantemente a la evolución de la base de datos de destino.

Proyecto de migración

Este proyecto comienza a poco tiempo de iniciar el desarrollo y actualmente se está llevando adelante en forma paralela con el mismo. Como se analizó en el capítulo dos, esto obliga a trabajar con una estructura de datos de destino inestable. Desde ese punto de vista hubiera sido más conveniente demorar el comienzo de la migración hasta un momento en el que el desarrollo del sistema se encuentre más avanzado, pero dada la complejidad de las transformaciones necesarias, el gran volumen de datos y la necesidad de contar con una base de datos migrada al momento de la puesta en marcha del sistema, se toma la decisión de comenzar las tareas propias de la migración en forma anticipada.

Las bases de datos de origen, en las que reside la información a ser migrada, responden a varios sistemas y son de una tecnología diversa, pero a los efectos de la migración se transforman en archivos de texto plano y son tomadas de esa manera como entrada del primer nivel.

La base de datos de destino es única y responde a la plataforma Oracle 11G, debiendo centralizarse en ella toda la información. Éste es un aspecto muy relevante del proyecto ya que los sistemas actuales funcionan de manera descentralizada en las 24 jurisdicciones electorales y debe unificarse toda la información en una base de datos centralizada.

En resumen, la entrada al proceso la componen 24 bases de datos expresadas todas bajo el formato de archivos de texto plano, un grupo más correspondiente a personas que residen en el exterior y 3 bases pertenecientes a otros sistemas. La salida se constituye como una única base relacional Oracle 11G. Más adelante en este mismo capítulo se brinda mayor detalle.

Cabe aclarar que inicialmente la migración se lleva adelante con alguna división física de procesos, pero la metodología de tres niveles iterativa comienza a aplicarse a partir de enero de 2017. De esta manera se ha podido gestionar el proceso y realizar numerosas iteraciones para lograr que ambos proyectos avancen en forma paralela pudiendo implementarse con éxito reiterados cambios lógicos y estructurales en la base de destino.

Al momento de escribir esta tesis, el desarrollo del sistema se encuentra en sus últimas etapas y se ha podido obtener una base de datos íntegramente poblada con información proveniente de las aplicaciones a ser reemplazadas, es decir, una migración completa. En este aspecto se puede considerar concluido el proyecto de migración, sólo resta aguardar el momento de la puesta en marcha para ejecutar los procesos de transformación de manera definitiva en los servidores de producción y eventualmente, procesar algún nuevo cambio que pueda surgir.

Mientras tanto, esta base de datos se utiliza para tareas de testing, para dar soporte a las actividades de desarrollo, para la realización de análisis de rendimiento y para la concreción de despliegues en los servidores de producción, que actualmente se emplean para realizar capacitaciones a los usuarios que utilizarán el nuevo sistema cuando esté concluido.

En su diseño actual la base de datos de destino tiene 456 tablas de las cuales 78 son alcanzadas por el proceso de migración y constituyen el núcleo de la estructura de datos a la vez que concentran la mayor complejidad lógica y el mayor volumen en cantidad de registros y campos. Cuenta también con 813 procedimientos almacenados y funciones, 470 secuencias, 1338 índices entre otras entidades.

Durante la última iteración realizada se logran migrar con éxito, usando la metodología propuesta, más de 38 millones de electores con toda su información periférica como por ejemplo sus estados, tipos de documentos, situaciones electorales, datos de inhabilitaciones etc. También se migran más de 127 millones de domicilios, más de 90 millones de trámites, más de 250 millones de registros de información de historiales de electores entre otras tablas. Todas ellas con información periférica que también impacta en otras entidades.

También hay que destacar que la lógica de negocio del nuevo sistema guarda diferencias muy significativas respecto de los que reemplaza, por lo tanto, no sólo se trata de grandes volúmenes de datos, sino que además las propias transformaciones resultan complejas. En otras

palabras, por cada entidad mencionada en el párrafo anterior se han debido realizar transformaciones lógicas que requieren un esfuerzo de análisis y de programación significativo.

En total en el proyecto de migración participan, cubriendo distintos roles, un coordinador general, un especialista en análisis funcional y seis administradores de bases de datos. También intervienen cuando es necesario programadores, diseñadores y directores afectados al proceso de desarrollo.

Respecto del equipamiento empleado se ha dispuesto de un servidor de bases de datos Oracle 11G en dependencias del Centro de Investigación y Desarrollo de Sistemas de la Facultad Regional Córdoba de la Universidad Tecnológica Nacional y de otro servidor de mayores prestaciones en cuanto a su hardware dentro del datacenter de Poder Judicial de la Nación.

Aplicación de la metodología de tres niveles iterativa

Como se expresa en párrafos anteriores, la necesidad de contar con una base de datos completamente migrada al momento de la puesta en marcha del sistema, obliga a comenzar el proyecto de migración en una etapa temprana del desarrollo y consecuentemente, lleva a trabajar con una estructura de destino inestable. A esto hay que sumarle el gran volumen de datos a migrar, la alta complejidad de las transformaciones y la necesidad de integrar varias bases de datos como origen para obtener un único destino soportado sobre una plataforma de tecnología muy distinta.

Estas razones motivan el empleo de la metodología de tres niveles iterativa ya que las características propias del proyecto de migración y del entorno en general, se ajustan a su definición y a las situaciones específicas para las que ha sido creada.

A continuación, se describen el modelo físico empleado, la implementación de cada una de sus cinco etapas, los informes de migración construidos, las herramientas empleadas y los mecanismos utilizados para el procesamiento de cambios dando algún ejemplo de cada uno de los tres tipos.

Construcción del modelo físico

Durante la primera fase (la de planificación y análisis general), entre otras tareas que se describen más adelante, uno de los primeros pasos es el planteo de los objetivos que debe cumplir el modelo físico. Los mismos se enumeran a continuación:

- Debe permitir tomar como entrada los datos provistos en archivos de texto plano.
- Debe permitir generar una única base intermedia como salida del primer proceso de transformación (T1) en Oracle 11G.
- Debe permitir tomar esa base intermedia como entrada del segundo proceso de transformación (T2) para dar como salida una base de datos soportada por Oracle 11G y que responda a las definiciones de formato y lógicas impuestas por el nuevo sistema.
- Debe resultar flexible y eficiente al momento de incorporar o cambiar elementos de programación o de configuración en cualquiera de sus niveles.

En función de estas premisas se dispone de un servidor Oracle 11G en el que se construyen dos esquemas respondiendo al diagrama expuesto en la figura siguiente. Más adelante en este mismo capítulo se dan detalles de las herramientas empleadas.

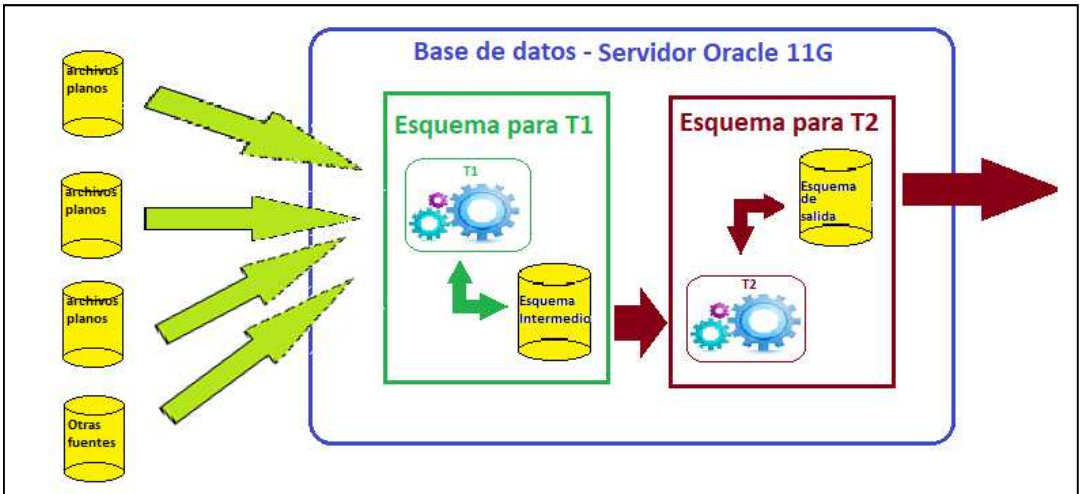


Figura 34: Estructura modelo físico para migración

Como lo muestra la figura, durante esta implementación, en una base de datos Oracle 11G se disponen dos esquemas, uno para el primer proceso de transformación y otro para el segundo.

En el primer esquema se alojan las herramientas para tomar la información de origen que es entregada en su mayor parte en formato de archivos de texto plano. Con determinadas herramientas se desarrollan las transformaciones sintácticas propias de T1 para generar la estructura de datos intermedia en Oracle 11G.

En el segundo esquema se toma como entrada la base intermedia y usando como principales herramientas procedimientos almacenados y funciones desarrolladas en PL-SQL, se hace la transformación semántica propia de T2. De ese modo se obtiene la estructura de datos de salida que puede ser utilizada por el nuevo sistema.

La definición este modelo físico se lleva adelante durante la primera fase de la metodología ya que hace a la planificación general. Cabe destacar que, en esta implementación en particular, podrían disponerse los recursos de diferentes maneras, por ejemplo, usar un solo esquema para alojar tanto T1 como T2 o usar dos bases de datos en lugar de dos esquemas o emplear dos servidores Oracle físicamente separados para ambos procesos, etc. La metodología no impone un determinado modelo, sino que da la flexibilidad suficiente para adaptar su implementación según las particularidades de cada caso siempre que se lleve a una división física de T1 y T2 con su respectiva base intermedia como único nexo.

Se opta por la arquitectura expuesta porque es simple y a la vez satisface las necesidades actuales y estimadas. De cualquier modo, pueden reubicarse los elementos intervinientes de manera distinta en una etapa posterior del desarrollo sin mayores inconvenientes.

Más adelante en este mismo capítulo, se brindan detalles de las herramientas empleadas en T1 y en T2.

Desarrollo de la fase uno (Planificación y análisis general)

Al terminar las tareas de reingeniería de procesos que forman parte de este proyecto, se comienza con las actividades de análisis y diseño del nuevo sistema y se obtiene un primer diagrama de entidades sobre el que se construye una versión inicial de la base de datos. Hasta ese momento no se inician actividades de migración.

Una vez obtenido el primer diseño de la base de datos se comienza con su construcción física en forma paralela a las actividades de desarrollo y de programación, todas éstas son ajenas al proyecto de migración.

Tal como se explica en el capítulo dos, en ocasiones existe una relación entre la estabilidad de la base de datos y el avance del desarrollo. Si bien no es una ley general, muchas veces ocurre (y es la situación en este proyecto) que a medida que el desarrollo del sistema avanza la base de datos recibe cada vez menos modificaciones. Así, en un comienzo resulta frecuente la realización de cambios estructurales o lógicos ya sea por el propio avance del desarrollo, por la incorporación de nuevas entidades, por errores en la construcción del modelo de datos, por cambios en requerimientos, etc. A medida que la construcción del nuevo sistema avanza la estructura de datos se vuelve más estable.

Si bien esto no es una regla formal ni algo que ocurre siempre, sí resulta algo frecuente y es la situación presentada en este caso particular. Desde este punto de vista hubiera sido conveniente dar comienzo a la primera fase de la migración de manera más tardía.

A la vez se detecta un gran volumen de datos, y sin haber hecho actividades formales de planificación de la migración, se advierte que la misma resulta compleja no sólo por este aspecto sino también por la complejidad de las transformaciones y por la significativa diferencia entre la plataforma de la base de destino y las de origen. También se determina que no es posible poner en marcha el nuevo sistema sin contar con la totalidad de la migración concluida y que es necesario realizar pruebas del funcionamiento de la aplicación sobre una base migrada antes de su implementación.

Por todas estas razones, se decide comenzar con las actividades de migración de manera muy temprana, aunque eso lleve probablemente a afrontar numerosos cambios en la base de destino. En este sentido, se hace una fuerte apuesta a la capacidad de la metodología para procesar cambios.

Durante el desarrollo de la primera fase del proyecto de migración se realizan numerosas reuniones en las que participan especialistas en bases de datos, analistas funcionales, programadores y directores tanto del Centro de Desarrollo como del Poder Judicial. Como resultado se generan los siguientes documentos:

- Documento de arquitectura general: Incluye las definiciones de arquitectura del proceso. (Anexo I)
- Documento de planificación general: Incluye el plan hasta la primera versión ejecutable del proceso de migración y la obtención de la primera base de datos de salida.
- Documento de definiciones técnicas: Incluye las definiciones de personal afectado, plataforma utilizada abarcando las definiciones de arquitectura y conectividad.
- Documento de plan de comunicación: se agregan definiciones en el plan de comunicación del proyecto de desarrollo que establecen los mecanismos a seguir cuando se modifique la estructura de la base de datos del sistema.

Debe recordarse que una de las premisas de la metodología es construir sólo los documentos que sean estrictamente necesarios y que los que se elaboren sean lo más sencillos posible a los efectos de favorecer la agilidad durante las siguientes fases.

Desarrollo de la fase dos (Análisis y diseño de T1)

Durante esta etapa se realiza primeramente el análisis de los datos de entrada para detectar las entidades sobre las que es necesario extraer información.

Se mantienen reuniones con especialistas en la materia integrantes del área de informática del Poder Judicial de la Nación y se acuerdan los siguientes puntos:

- Al ser diversas las fuentes de datos, la información a ser migrada se entrega en formato de archivos de texto plano para todos los casos. Esta definición simplifica el proceso ya que los sistemas actuales tienen un mecanismo que permite exportar los datos en dicho formato. Personal del Poder Judicial se responsabiliza de realizar la exportación que se comporta como única fuente de entrada del proceso de migración.
- Al estar en producción los sistemas actuales y dada la alta demanda que tienen, las necesidades de seguridad y la dispersión geográfica, se decide no conectar ningún proceso de migración en forma directa a ellos. Todos los datos de entrada son tomados a partir de los archivos de texto mencionados en el punto anterior.

- La entrega de los archivos con la información de entrada se realiza formalmente dejándolos disponibles en una unidad de disco destinada a tal fin dentro del servidor Oracle sobre el que se ejecutan los procesos de migración.
- Conforme al documento de arquitectura elaborado en la fase uno (Anexo I), se crea un esquema en el servidor Oracle dispuesto para la migración. Dicho esquema contiene el primer proceso de transformación (T1) y la base de datos intermedia.
- Todo el desarrollo de T1 se lleva adelante usando como única herramienta SQL Loader de Oracle (40) ya que permite importar archivos de texto con rapidez y simpleza, se adapta muy bien a entornos donde el origen de datos responde a formato de archivos de texto sobre todo si se realizan pocas transformaciones semánticas y muchas conversiones sintácticas.

Posteriormente se construye un diagrama de entidad relación que expresa las entidades alcanzadas por el primer proceso de migración y define la estructura de datos de la base intermedia.

No se elaboran diagramas de flujo ni se especifican los procesos de transformación porque no se encuentran conversiones complejas desde el punto de vista lógico. Tal como se explicó en el capítulo tres, esta situación es previsible en el primer nivel dado que se realizan aquí transformaciones sintácticas y no semánticas.

Se define también una planificación más detallada para las tareas técnicas de este nivel que contempla la reutilización de elementos de migración con los que se cuenta previamente y establece un plazo de sesenta días para la obtención de la primera versión ejecutable de T1.

Desarrollo de la fase tres (Análisis y diseño de T2)

Tal como indica la planificación original, el desarrollo de esta etapa se comienza en paralelo con la fase cuatro. En otras palabras, el diseño del segundo nivel puede llevarse adelante al mismo tiempo que la construcción del primero. Resulta conveniente hacerlo de este modo porque se ahorra tiempo y algunos elementos de T2 pueden probarse a medida que se concluya con los factores relacionados de T1.

Al igual que en la etapa anterior se realizan numerosas reuniones con personal especializado del Poder Judicial de la Nación y se acuerdan los aspectos centrales que pueden resumirse en los siguientes ítems:

- La única fuente de datos de entrada para T2 es la base intermedia generada por T1.
- Todas las transformaciones necesarias para obtener la base de salida a partir de la intermedia se realizan mediante procedimientos almacenados escritos en PL SQL que se alojan en el mismo esquema que la base intermedia y que se agrupan en:
  - Procedimientos de migración de electores.
  - Procedimientos de migración de historiales.
  - Procedimientos de migración de partidos políticos.
  - Procedimientos de migración de trámites en curso.
  - Procedimientos de migración de tablas satélites.

No se elaboran diagramas de entidad relación porque se cuenta con el construido para la base intermedia en la fase anterior y con el perteneciente a la base de destino definido por el equipo de bases de datos del proyecto de desarrollo del sistema. No obstante, se procede a distinguir en dicho diagrama las entidades alcanzadas por T2.

Se construyen algunos diagramas de flujo, descripciones en pseudo-código y especificaciones de los procesos que lo requieren por su complejidad. A modo de ejemplo puede verse el Anexo II que corresponde al proceso de migración de domicilios para los electores.

Al igual que en el caso anterior se construye una planificación detallada para la fase cinco que abarca hasta la primera versión ejecutable del proceso T2 y la obtención de la primera base salida. En términos generales, dicha planificación prevé comenzar con la programación de los procedimientos almacenados que migran el grupo de tablas satélites a principios de marzo de 2017 para poder ejecutarlos y probarlos inmediatamente después de obtener la primera base intermedia.

Luego se planea continuar con la migración de las entidades relacionadas a los electores y a sus domicilios para posteriormente incorporar los historiales. En octubre de 2017 estos grupos deben estar completamente migrados y encontrarse en fase de prueba.

A partir de noviembre de 2017 la planificación contempla comenzar a incorporar todas las entidades relacionadas al grupo de partidos políticos y finalizar con la migración de trámites en curso. Establece un plazo aproximado de 60 días para su culminación incluyendo las pruebas.

Desarrollo de la fase cuatro (Construcción ejecución y prueba de T1)

La construcción del primer nivel de migración se lleva adelante según lo planeado. A fines de marzo de 2017 se obtiene la primera versión ejecutable de los procesos desarrollados con Oracle SQL Loader y se crea físicamente la primera base intermedia conteniendo la información de los archivos de texto plano de origen en tablas del esquema Oracle destinado a tal fin.

No se registran retrasos significativos respecto de la planificación original y no se encuentran transformaciones de excesiva complejidad.

Al momento de escribir esta tesis se han realizado cuatro iteraciones de la fase cuatro generando una base intermedia compuesta por 507 tablas que en la mayoría de los casos guardan una relación uno a uno con los archivos de texto plano tomados como fuente de origen. Un pequeño grupo de tablas contiene información redundante a los efectos de simplificar algunos procesos de la fase cinco.

Debe tenerse en cuenta que estas tablas agrupan información proveniente de las 24 provincias más un grupo adicional compuesto por los datos de los electores residentes en el exterior. En esta implementación en particular, éste resulta un aspecto muy relevante del primer nivel porque no sólo permite realizar las transformaciones sintácticas (su verdadero objetivo) sino que posibilita obtener una primera base de datos centralizada.

En la iteración inicial se incluyen principalmente tablas satélites que en general tienen un volumen menor y que conforman la base para migrar otras más complejas. Como ejemplo se puede citar la tabla “bahra_distritos” que con sus 25 registros representa los distritos electorales de nuestro país y que resulta central en la migración de los electores. Otro ejemplo es la tabla “bahra_calles” que en 147710 registros contiene un grupo de calles correspondiente a uno de los sistemas actuales y que son necesarias para poder migrar los domicilios.

En la segunda iteración se incorporan las entidades relacionadas con los electores y sus domicilios. Por ejemplo, la tabla "PADR" que con sus 32854978 registros contiene información de los electores de nuestro país. Otro ejemplo es la tabla "PMIX" que almacena datos sobre domicilios de los mismos electores, sólo por mencionar algunas entre muchas otras que conforman la estructura.

En la tercera iteración se incluyen los datos correspondientes historiales que son los de mayor volumen. Por ejemplo, la tabla "MPAC" que está relacionada a los movimientos de trámites efectuados por los ciudadanos. Actualmente contiene 111022671 registros que responden a la información de los sistemas a ser reemplazados. Otro ejemplo que se puede citar es la tabla "HFMI" que con 109501759 registros guarda información relacionada con estructura familiar de los electores.

En la cuarta se incorporan las tablas relacionadas a partidos políticos y trámites en curso que tienen un volumen mucho menor y que en general no presentan ninguna dificultad relevante.

Es conveniente aclarar que los nombres de las tablas de la base intermedia se han hecho coincidir con los archivos de entrada provistos, que a su vez guardan relación con las denominaciones de las tablas de los sistemas originales. De allí la particularidad de dichos nombres, su corta longitud y escasa descripción.

En cada iteración no sólo se incluyen nuevas entidades por el propio avance del proceso de migración, sino que también se incorporan modificaciones según las necesidades del desarrollo del sistema. El protocolo para registrar los cambios estructurales en la base de datos, para que luego sean evaluados y eventualmente incluidos en el proceso de migración, se define en un documento específico mostrado en el Anexo III.

La definición de dicho protocolo corresponde al equipo de trabajo afectado al desarrollo del sistema y no está dentro del alcance del proceso de migración. En este sentido, se puede decir que resulta muy conveniente reutilizar instrumentos de comunicación ya establecidos porque de ese modo no se generan mecanismos excepcionales y se disminuye la cantidad de documentación.

Más adelante, en este mismo capítulo, se expone un ejemplo de procesamiento de cambio que afecta la base intermedia y que por ende conlleva a modificar convenientemente el primer nivel de transformación T1.

Respecto de las pruebas y los correspondientes informes de migración, en este nivel se desarrollan consultas programadas en PL-SQL sobre la base intermedia y programas que leen los archivos de texto de origen con el objeto de hacer comparaciones en los datos y posteriormente elaborar los informes que se enumeran a continuación:

- Cantidad total de registros de cada archivo de texto frente a cantidad total de registros de cada tabla correspondiente de la base intermedia.
- Por cada entidad, claves existentes en los archivos de texto que no se corresponden a claves en las tablas correspondientes.
- Por cada entidad, claves existentes en las tablas de la base intermedia que no existen en los archivos de texto.
- Detalle de registros descartados. Esta información se obtiene del archivo del log de resultados de la herramienta SQL Loader empleada en el desarrollo de T1.
- Para algunas entidades de mayor relevancia y complejidad como, por ejemplo, electores, domicilios, situaciones electorales, etc. se realizan algunos análisis específicos relacionados a la conversión de tipos de datos en algunos campos particulares.

Como es de esperarse, al hacer sólo conversiones sintácticas en T1, se encuentran pocos casos de registros rechazados o mal procesados. Mantener una equivalencia uno a uno en todos los casos posibles entre los archivos de origen y las tablas de la base intermedia, simplifica notablemente el proceso, facilita el análisis y comparación de los resultados para elaborar los informes de migración y disminuye las posibilidades de error.

Desarrollo de la fase cinco (Construcción Ejecución y Prueba de T2)

Tal como se había planeado durante la fase tres, se da comienzo a la construcción de los primeros procedimientos almacenados a fines de abril de 2017. Estos abarcan el grupo de entidades periféricas, como por ejemplo la tabla “Circuitos_electoral” que contiene todos los circuitos electorales de nuestro país con un total de 6645 registros o la “Localidades” que contiene todas las

localidades y de la que se migran 162737 registros. Durante esta iteración inicial se desarrollan en total 18 procedimientos almacenados, en su mayoría de lógica simple, para obtener una cantidad importante de tablas.

Aquí cabe aclarar que varios procedimientos fueron programados con anterioridad, a poco de comenzar el desarrollo del nuevo sistema y antes de la aplicación de la metodología propuesta. Si bien éstos no se ajustan a los requerimientos actuales ni al planteo metodológico, parte de su código puede reutilizarse con el consecuente ahorro de tiempo. Éste es un aspecto positivo de la metodología ya que su flexibilidad permite escoger las herramientas con mucha libertad y reutilizar artefactos existentes cuando esto resulte conveniente.

En el mes de mayo, luego de tener una primera versión ejecutable de T2, se comienza una segunda iteración que incorpora el grupo de entidades relacionadas con electores y sus domicilios y también incluye los cambios que sufre la base de destino como consecuencia del desarrollo del sistema.

Para migrar la totalidad de electores y domicilios se realizan cinco iteraciones, cada una incluye un nuevo grupo de tablas y la totalidad de las modificaciones pendientes en la base de destino.

En total durante esas cinco iteraciones, que transcurren entre mayo y setiembre de 2017, se desarrollan 12 procedimientos almacenados de una alta complejidad y se generan doce tablas en la base de destino. Cabe destacar que la coincidencia de estos dos números es meramente casual, no existe una relación uno a uno entre los procedimientos y las tablas generadas, más aun, algunos de estos procedimientos han sido reemplazados por otros. Al finalizar la última iteración, quedan 3 que incluyen las funcionalidades que originalmente se planearon para los 12.

A la vez se procesan los cambios estructurales y lógicos que ocurren como consecuencia del desarrollo del sistema que sigue evolucionando paralelamente a la migración. Al igual que en el caso de T1, el protocolo para registrar los cambios estructurales en la base de datos, para que luego sean evaluados y eventualmente incluidos en el proceso de migración, está definido en el mismo documento que se incluye en el Anexo III. Más adelante en este mismo capítulo se describen, a modo de ejemplo, algunos de estos cambios.

Entre las tablas afectadas por esta etapa se encuentran por ejemplo “Electores” de la cual se migran 38491761 registros. O la tabla “domicilios” con idéntica cantidad. Es el avance más significativo y el hito más importante desde el comienzo del proyecto ya que se ven alcanzadas las tablas más centrales, de mayor cantidad de registros y de gran complejidad en sus transformaciones.

A partir de setiembre de 2017 se incorporan las entidades relacionadas a la gestión de historiales, en este caso se desarrollan 5 procedimientos almacenados que resuelven la migración de 15 tablas. En el período mencionado ocurren 3 iteraciones que se gestionan con el mismo criterio que en el caso anterior, es decir, incorporar en cada una los cambios pendientes en la base de destino. Algunas de las tablas afectadas son por ejemplo “Tramites” con 89081227 registros o “Movimientos_Tramite” con 89086895 registros.

Durante noviembre de 2017 se da curso a la décima iteración en la que se incorporan 14 procedimientos almacenados para completar la migración del grupo de entidades relacionadas con partidos políticos y trámites en curso.

Actualmente la base de datos de destino se encuentra operativa y conectada a la aplicación. Se emplea para tareas del área de testing, para el área de desarrollo y para la realización de capacitaciones. También para pruebas de rendimiento de las que han surgido importantes ajustes y para ensayos de despliegue en los servidores de producción. En este aspecto, puede afirmarse que es muy positivo para todo el proyecto en general, contar con una base de datos poblada con una importante cantidad de registros migrados en forma temprana.

Durante toda la fase 5, el proceso se desarrolla con normalidad y la metodología demuestra ser muy efectiva al momento de incorporar nuevas funcionalidades y de procesar cambios en la base de salida, incluso en situaciones en las que los cambios afectan procesos de transformación complejos. Un aspecto muy positivo es que la migración en ningún momento implica demoras en el desarrollo del sistema ya que se adapta a los cambios con facilidad.

Como puede advertirse a partir de los párrafos anteriores, en esta implementación, T2 se compone en su totalidad de procedimientos almacenados y funciones desarrolladas en PL-SQL tal como se planifica inicialmente. Ésta es una particularidad del caso de aplicación, no es una necesidad de la metodología. Se escoge dicha herramienta dada la cantidad y complejidad de

transformaciones semánticas que deben realizarse sobre los datos de la base intermedia. Si bien esto lleva a un gran esfuerzo de programación, da la posibilidad de mantener mucho control. La metodología demuestra adaptarse muy bien al empleo de herramientas de programación dentro del desarrollo de los procesos de transformación.

Respecto de los informes de migración, se debe decir que en esta etapa alcanzan un alto grado de importancia por la confluencia de varios factores. La gran cantidad de entidades y registros migrados hace imposible una inspección visual de los resultados, al mismo tiempo la complejidad de las transformaciones lleva a la utilización de instrumentos de programación que son más susceptibles a fallos que las herramientas automatizadas y, además, se debe sumar la sensibilidad de la información que no admite ningún error y entonces hace muy necesario un control exhaustivo. En otras palabras, se trata de millones de registros, que representan datos de personas que se utilizan para la configuración de los actos electorales y que no permiten ningún margen de error y donde la estructura de origen tiene grandes diferencias con la de destino.

A partir de esta situación se programan diversos informes de migración, algunos se enumeran a continuación:

- Cantidad de registros migrados por entidad frente a cantidad de registros de las entidades de origen. Aquí se comparan campos claves porque no existe una relación uno a uno entre las entidades como ocurre en T1 ya que las diferencias estructurales entre la base de origen y la de destino son muy significativas. Así, por ejemplo, puede programarse un informe que compare la cantidad de electores en el origen y la cantidad en el destino, aunque no hay una relación registro a registro ni tabla a tabla.

  Algo similar ocurre con otras entidades, como los domicilios, los historiales, los trámites. En otros casos como, por ejemplo, las localidades, las profesiones, los circuitos electorales, etc. la comparación es más simple.
- Cantidad de electores por situación. Esto agrupa los electores pos su situación electoral (un atributo muy importante del modelo de negocio) lo que permite comparar dichas cantidades con los valores arrojados por los sistemas actuales.
- Estado de avance de la migración. Los distintos procedimientos escriben gradualmente su grado de avance en una tabla de auditoría sobre la que se elabora un informe que permite

visualizar su desempeño y evaluar la cantidad de registros tratados y el tiempo de respuesta para hacer eventuales ajustes en la programación.

- Log de errores. Los distintos procesos escriben en un log de errores los registros que descartan por situaciones anómalas que no logran resolver. Por ejemplo, imposibilidad de cumplir una regla de integridad referencial o una restricción unique. Su análisis resulta de suma importancia porque permite detectar posibles errores de programación, posibles transformaciones mal realizadas en T1 o incluso inconsistencias en los datos de entrada.

No es la intención de este apartado realizar una extensa descripción de los informes utilizados, sólo se mencionan algunos para tener una visión global de la manera en que se organizan las tareas. Lo importante es destacar que la metodología se ajusta con facilidad al empleo de diversos informes según las necesidades de cada caso y según las herramientas de las que se disponga.

Luego de contar con cada nueva versión en la base de salida se realizan pruebas del funcionamiento de la aplicación sobre esa estructura de datos. En este proyecto esa responsabilidad corresponde al equipo de testing de la aplicación, que en algunos casos emplea test manuales y en otros test automatizados. De cualquier manera, en esta implementación las actividades de testing están fuera del alcance del proyecto de migración. Nuevamente aquí resulta de mucha utilidad la flexibilidad de la metodología que no impone una determinada manera de hacer las pruebas adaptándose a cada caso particular.

Durante toda esta fase la metodología demuestra una gran capacidad para procesar cambios y permite desarrollar la migración en paralelo a la evolución del sistema sin que un proyecto implique demoras en el otro. Su gran flexibilidad y su comportamiento iterativo resultan claves para adaptarse a las modificaciones e incorporarlas gradualmente a la estructura de destino.

Respecto de las herramientas empleadas, debe decirse que para el desarrollo de T2 se utiliza el lenguaje PL-SQL como única herramienta tanto para programar los procedimientos almacenados y funciones que realizan las transformaciones como para elaborar los informes de migración. Al igual que se mencionó en el apartado anterior, la elección de la herramienta está relacionada con las características de las conversiones realizadas, la tecnología empleada, la

disponibilidad y los conocimientos y experiencia de las personas intervinientes. La metodología es multiplataforma y no condiciona el empleo de determinadas herramientas.

Procesamiento de cambios

Se ha insistido en que uno de los aspectos centrales de la metodología propuesta es justamente lograr un procesamiento de cambios eficiente. En este apartado se exponen algunos casos en los que se debieron incorporar modificaciones en el proceso de migración dentro de la implementación realizada.

No es la intención mostrar una larga lista de cambios. En rigor de verdad, las modificaciones que ocurren durante este desarrollo son numerosas, pero basta mencionar algunos ejemplos de cada tipo de cambio para analizar el desempeño y la mecánica empleada.

En el capítulo anterior se clasificaron los cambios en tres tipos. Recordando, tenemos que un cambio es de tipo uno cuando sólo afecta a T1, es de tipo dos cuando sólo afecta a T2 y es de tipo tres cuando afecta tanto a T1 como a T2. Esto no guarda relación con la razón del cambio, es decir, es independiente de que provenga de un cambio de requerimientos durante el desarrollo del sistema de información, de la corrección de un error en el proceso de migración o de la incorporación de nuevas entidades a la misma. Esto en el capítulo anterior se denomina modificación, corrección o ampliación respectivamente.

Se exponen a continuación algunos casos situados en distintas iteraciones y de razones y tipología diversos.

Ejemplo de cambio tipo uno por ampliación

Durante la segunda iteración de la etapa de migración de electores, se incorporan aquellos que tienen la característica de ser privados de la libertad. Inicialmente no son tenidos en cuenta debido a que el sistema aún no los incluía en sus procesos.

Conforme al avance del desarrollo del sistema, se incorpora el tratamiento de este tipo de electores y consecuentemente la migración debe tenerlos en cuenta. Del análisis surge que la

estructura intermedia no requiere ningún cambio, los electores en cuestión pueden incluirse en las tablas previstas y ser migrados a partir de los procedimientos ya desarrollados en T2.

Esta situación hace que dicho cambio pueda catalogarse como de tipo uno porque sólo afecta al primer proceso sin que sea necesario modificar el segundo. Al mismo tiempo se identifica que su origen responde a una ampliación ya que, como consecuencia del desarrollo del sistema, se incorporan entidades que anteriormente habían sido excluidas.

Las acciones que se realizan modifican algunos procesos de T1 y, empleando la herramienta SQL Loader, se añaden a la base intermedia los registros específicos. No es necesario modificar los informes de migración del primer nivel puesto que los recuentos de registros y los controles realizados siguen siendo los mismos.

El cambio puede resumirse de este modo:

- Tipo: Tipo uno pues sólo afecta al primer nivel.
- Origen: Ampliación.
- Momento: Segunda iteración migración de electores.
- Objetivo: Incorporar electores privados de la libertad.
- Acción realizada: Modificación de T1 utilizando SQL Loader para incluir en la base intermedia los registros requeridos.
- Resultado: Se procesa el cambio con éxito y rapidez ya que no fue necesario analizar ni modificar la estructura intermedia ni los procesos que realizan transformaciones a partir de ella. Los esfuerzos se concentran en la lectura de nuevos registros en las bases de origen y grabación de los mismos en la intermedia.

Ejemplo de cambio tipo uno por corrección

Luego de la primera iteración de la etapa de migración de tablas satélites, a partir de los informes de migración, se descubre que la cantidad de registros de la tabla intermedia que guarda información de las localidades no coincide con la cantidad de entradas del archivo de texto plano tomado como origen.

A partir de revisar la estructura del archivo, la estructura de la tabla intermedia y el log de errores generado por SQL Loader se concluye que algunos registros se descartan por contener accidentalmente en sus descripciones caracteres especiales que no pueden ser convertidos.

Se modifica el proceso para que pueda convertirlos y en la segunda iteración de esta misma etapa, los datos de las localidades pueden procesarse sin inconvenientes. Así lo revela el informe de migración elaborado oportunamente.

El cambio puede resumirse de este modo:

- Tipo: Tipo uno pues sólo afecta al primer nivel.
- Origen: Corrección.
- Momento: Segunda iteración migración de tablas satélites.
- Objetivo: Corregir la migración de localidades.
- Acción realizada: Corregir el proceso de SQL Loader y volver a ejecutar en la iteración siguiente.
- Resultado: El error se corrige con éxito y rapidez sin implicar modificaciones en ningún procedimiento almacenado ni en elementos de programación complejos ya que no fue necesario analizar ni modificar la estructura intermedia ni los procesos que realizan transformaciones a partir de ella.

Ejemplo de cambio tipo dos por ampliación

En la segunda iteración de la etapa de migración de electores, se decide incorporar información de las situaciones electorales. Éstas se comportan como un atributo del elector en las bases de origen y tienen una estructura de datos diferente en la base de destino, por lo que su transformación resulta compleja.

Inicialmente los conjuntos de datos de origen son tenidos en cuenta en el nivel uno por tratarse de atributos de los electores y consecuentemente se incorporaron a la base intermedia. Por esta razón, no es necesario realizar ninguna modificación a T1 y el trabajo se centra en T2.

Se desarrolla un procedimiento almacenado que lee la información correspondiente de la base intermedia y realiza las transformaciones semánticas grabando las situaciones electorales correspondientes a cada elector en la base de destino.

El cambio puede resumirse de este modo:

- Tipo: Tipo dos pues sólo afecta al segundo nivel.
- Origen: Ampliación.
- Momento: Segunda iteración migración de electores.
- Objetivo: Incorporar información de las situaciones electorales.
- Acción realizada: Se construye un nuevo procedimiento almacenado que toma información de la base intermedia y realiza las transformaciones semánticas escribiendo en la base de destino.
- Resultado: El procedimiento almacenado se desarrolla con las dificultades que impone la transformación en cuestión, pero sin que sea necesario leer archivos de texto ni revisar las conversiones sintácticas de la fuente de origen lo que simplifica su concreción.

Ejemplo de cambio tipo dos por corrección

Durante la implementación objeto de estudio, este tipo de cambio resulta algo frecuente. La razón es que las transformaciones de T1 se realizan con una herramienta automatizada, mientras que las de T2 se llevan a cabo mediante programación debido a su complejidad. Consecuentemente, un mínimo error en una línea de código, provoca que determinados registros no se transformen semánticamente de la manera correcta y que luego deba subsanarse este error.

Al finalizar la segunda iteración de la etapa de migración de electores, a partir de los resultados arrojados por los informes de migración, se descubre que algunos registros no tienen asociado de manera correcta su circuito electoral. Ésta es una información sumamente central dentro de la lógica de negocio del nuevo sistema, por lo que su corrección resulta urgente.

A partir del análisis realizado, se descubre un error en un procedimiento almacenado que forma parte de T2. Su código se modifica, se prueba con una cantidad pequeña de registros para verificar su correcto funcionamiento y es dejado disponible y listo para ejecutar en la siguiente iteración.

Durante la tercera iteración de la misma etapa, el error desaparece, los registros son procesados correctamente y así lo revelan los informes de migración.

El cambio puede resumirse de este modo:

- Tipo: Tipo dos pues sólo afecta al segundo nivel.
- Origen: Corrección.
- Momento: Tercera iteración migración de electores.
- Objetivo: Corregir un error al asignar circuitos electorales.
- Acción realizada: Corrección de un procedimiento almacenado que forma parte de T2.
- Resultado: El procedimiento almacenado se corrige, se prueba y se vuelve a ejecutar con éxito. Un aspecto muy relevante es que la corrección ha podido hacerse sin modificar la estructura de la base intermedia y sin destinar esfuerzos a interpretar información de la fuente de origen.

  Otro aspecto que simplifica el procesamiento de la corrección es que el procedimiento almacenado en cuestión no contiene transformaciones sintácticas y no realiza lecturas en archivos de texto ni ninguna otra operación que no esté estrechamente relacionada con la conversión semántica que resuelve. Una vez más, durante esta implementación, se manifiesta la gran ventaja que implica contar con una metodología que plantee las transformaciones en un modelo físico dividido y de manera iterativa.

Ejemplo de cambio tipo dos por modificación

Durante el desarrollo de la cuarta iteración de la etapa de migración de electores, un cambio en los requerimientos del sistema implica modificar la manera en que un elector se asocia a su domicilio.

Más concretamente, en el sistema de información, la forma en que se determina la localidad en la que el elector reside, cambia y esto impacta en el proceso de migración.

Se realiza un análisis de impacto y se llega a la conclusión de que la información necesaria está disponible en la base intermedia y que la modificación sólo alcanza a la transformación semántica, es decir, alcanza a T2 sin tener impacto en T1.

Consecuentemente se realizan modificaciones en el procedimiento almacenado que migra electores con sus domicilios y se lo ejecuta con resultados exitosos según lo revelan los informes correspondientes.

El cambio puede resumirse de este modo:

- Tipo: Tipo dos pues sólo afecta al segundo nivel.
- Origen: Modificación.
- Momento: Cuarta iteración migración de electores y domicilios.
- Objetivo: Cambiar la asociación con localidades respondiendo a un cambio de requerimiento del sistema.
- Acción realizada: Modificación de un procedimiento almacenado que forma parte de T2.
- Resultado: El procedimiento almacenado se modifica exitosamente sin que este cambio de requerimiento implique volver a leer y a transformar datos desde las bases de origen. En términos generales, cabe el mismo análisis de resultado que en el punto anterior.

Ejemplo de cambio tipo tres por corrección

Luego de la primera iteración de migración de tablas satélites, al analizar el log de errores, se descubre que algunos países no fueron incorporados en la base de destino. Del análisis surge que su descripción resulta muy larga para el campo previsto.

La primera conclusión es que dichos registros debieron incorporarse de todos modos, a partir de cortar el campo con excesiva cantidad de carateres, de este modo no se pierden registros y paralelamente informar la situación en el log de errores para que no pase inadvertida.

Al mismo tiempo se reconoce que la situación no debiera ocurrir, pues las transformaciones sintácticas se realizan dentro de T1 que debe dejar una base intermedia con campos ya convertidos sintácticamente, resolviendo problemas de tipos de datos, conversiones de formato y longitudes entre otros elementos.

Del análisis de T1 se desprende que también tiene un error, pues las descripciones con largo excesivo en la base intermedia están pobladas con espacios tanto al comienzo como al final de la

cadena de caracteres convertida. Revisando los datos de origen en profundidad se descubre que allí se encuentran dichos espacios y que el error de T1 es no quitarlos.

Finalmente se decide modificar T1 para que corte los espacios innecesarios y para que controle el largo de este campo al igual que lo hace con otros. También se modifica T2 para que, si vuelve a ocurrir una situación similar, informe el problema, pero no descarte el registro ya que puede importarse simplemente truncando los caracteres que queden fuera de rango.

Lamentablemente este pequeño error se torna costoso al implicar modificaciones en ambos procesos de transformación. Si los informes de migración de T1 hubieran detectado oportunamente la situación, se hubiese podido tener en cuenta en el desarrollo de T2, pero como el defecto paso inadvertido, ambos procesos tuvieron que ser modificados.

El cambio puede resumirse de este modo:

- Tipo: Tipo tres pues afecta tanto a T1 como a T2.
- Origen: Corrección.
- Momento: Primera iteración, migración de tablas satélites.
- Objetivo: Reparar un error para que determinados registros no sean rechazados cuando un campo en particular excede el largo previsto.
- Acción realizada: Modificación de un procedimiento almacenado que forma parte de T2 y de una configuración de SQL Loader que forma parte de T1.
- Resultado: El error se repara, aunque fue necesario modificar elementos de ambos procesos. Incluso en este caso, la metodología y la arquitectura física planteada sigue teniendo importantes ventajas pues posibilita hacer las correcciones en dos etapas afectando recursos humanos distintos y paralelizando las tareas.

Ejemplo de cambio tipo tres por ampliación

En la décima iteración, se realiza la incorporación de datos de partidos políticos. Es una iteración importante porque agrega al proceso de migración el último conjunto de datos de volumen relevante que aún no ha sido tenido en cuenta.

Esta incorporación implica tomar datos desde otra fuente de origen y, por lo tanto, hay que modificar T1 para que realice las transformaciones sintácticas correspondientes. Al mismo tiempo, también resulta necesario incluir nuevos procedimientos almacenados dentro de T2 para que resuelvan las transformaciones semánticas. Finalmente, esta ampliación tiene impacto tanto en T1 como en T2.

Un aspecto que debe destacarse, es que, si la información de origen de los partidos políticos hubiera podido incorporarse con anterioridad, se estaría ahora frente a una ampliación tipo dos pues el impacto sólo alcanzaría a T2 sin que sea necesario modificar T1, pero la dinámica del proyecto no permite en este caso anticiparse.

El análisis y el diseño como así también la construcción de todo el módulo de partidos políticos del sistema, se realiza con posterioridad lo que impide adelantar en algún aspecto el proceso de migración al no contar con una definición de alcance tanto en la estructura de datos de origen como de destino.

Durante esta iteración, es necesario incorporar a T1 las configuraciones necesarias de SQL Loader para que pueda realizar las conversiones sintácticas en las nuevas entidades, modificar la estructura de la base de datos intermedia para que pueda alojar las nuevas salidas de T1 y desarrollar 12 nuevos procedimientos almacenados que formen parte de T2 para tomar los datos de las nuevas entidades de la base intermedia y grabarlos en la base de destino según las conversiones semánticas que se realicen.

El cambio puede resumirse de este modo:

- Tipo: Tipo tres pues afecta tanto a T1 como a T2.
- Origen: Ampliación.
- Momento: Décima iteración de T2 y cuarta de T1.
- Objetivo: Incorporar al proceso de migración todas las entidades vinculadas al tratamiento de partidos políticos.
- Acción realizada: Desarrollo de nuevas configuraciones en SQL Loader dentro de T1, modificación de la base intermedia y programación de nuevos procedimientos almacenados dentro de T2.

- Resultado: Incorporación de partidos políticos al proceso de migración. La ampliación se desarrolla sin inconvenientes. Las operaciones sobre T1 se realizan en paralelo a las de T2 lo cual pone nuevamente de manifiesto las ventajas de la metodología y de su arquitectura al igual que en los casos anteriores.

Ejemplo de cambio tipo tres por modificación

Durante el desarrollo de la sexta iteración en la etapa de migración de electores y domicilios, nuevamente un cambio en los requerimientos impacta en la forma en que se relaciona un elector con la localidad en la que reside.

Desde el punto de vista del desarrollo del sistema, el cambio de requerimiento implica asociar la localidad al circuito electoral cuando resulte posible o inferirla mediante determinadas reglas de negocio cuando no. También implica quitar la localidad en algunos filtros de búsqueda y pasar a tratarla como un campo no obligatorio en la base de datos.

Desde el punto de vista del proceso de migración, el impacto es realizar la transformación semántica con las mismas reglas de negocio de la aplicación, por lo tanto, la transformación semántica de la localidad debe ser cambiada. Desde este enfoque el cambio sólo afecta a T2, pero ocurre que la información necesaria para la programación de algunas reglas que vinculan la localidad al circuito electoral no está disponible en la base intermedia, simplemente no fue tenida en cuenta al desarrollar T1 porque no se esperaba que sea necesaria.

Consecuentemente debe modificarse T1 para tomar la información de las fuentes de origen e incorporarla a la base intermedia y T2 para realizar las transformaciones semánticas y poder escribir en la base de destino según las reglas impuestas por el nuevo requerimiento del sistema.

El cambio puede resumirse de este modo:

- Tipo: Tipo tres pues afecta tanto a T1 como a T2.
- Origen: Modificación.
- Momento: Sexta iteración de T2 y tercera de T1.
- Objetivo: Incluir dentro de la migración una nueva lógica para migración de localidades que se desprende de un cambio de requerimiento del sistema.

- Acción realizada: Modificación de un procedimiento almacenado que forma parte de T2, Desarrollo de una nueva configuración de SQL Loader que forma parte de T1 y creación de una nueva tabla en la base intermedia.
- Resultado: El cambio de requerimiento es procesado exitosamente. Si bien este tipo de cambio de requerimiento resulta el más costoso porque afecta tanto a T1 como a T2, la metodología posibilita resolverlo en paralelo y sin mayores complicaciones.

Hipótesis de no implementación

Planteada la experiencia de implementación de la metodología propuesta, resulta conveniente hacer un breve análisis de cómo hubiera sido no utilizarla en el mismo caso de estudio. Desde luego que, al tratarse de una suposición, el análisis es meramente especulativo pero aun así, resulta interesante confrontar la experiencia realizada con una supuesta implementación de cualquier otra metodología clásica.

Partiendo de la base de la estructura general de las metodologías clásicas planteadas en el capítulo III (ver figuras 35, 36 y 37) puede hacerse alguna especulación respecto de cómo hubiera sido su empleo en el presente caso de estudio.

Primeramente, es importante destacar que las metodologías clásicas analizadas en el capítulo III son de mucha utilidad, están respaldadas por los principales fabricantes de motores de bases de datos y herramientas relacionadas y cuentan con una amplia difusión y solvencia. No es la intención de este apartado concluir que no son útiles o que no deben emplearse, simplemente resulta interesante analizar su comportamiento hipotético en el caso de estudio.

Como se observó en el capítulo anterior, la estructura clásica, con mayor o menor detalle, está basada en tres grandes etapas: planear, migrar y validar.

Para la primera, en el contexto particular estudio, hubiera sido muy difícil (o incluso imposible) elaborar una planificación detallada de todo el proceso porque buena parte del mismo resulta desconocido al momento de comenzar. Usualmente, los proyectos de migración cuentan con algún factor de incertidumbre, sobre todo en sus momentos iniciales, pero al encontrarse en

construcción la base de destino y también el sistema que la emplea, ese nivel de incertidumbre es mucho mayor porque está sujeto a las posibles variaciones que surjan durante el desarrollo.

En este aspecto, la metodología iterativa demuestra adaptarse muy bien ya que permite realizar una planificación general, suficiente para organizar la generalidad del proceso y comenzar con la primera iteración para luego hacer los ajustes correspondientes en función de los cambios en el desarrollo del sistema y realizar ajustes graduales en cada ciclo.

Otro elemento relevante es que en las metodologías clásicas la migración en sí, es decir, la lectura desde las bases de origen, la propia transformación y la escritura en las bases de destino, está vista como un único proceso. Como se analiza en el capítulo III, algunas metodologías hacen un planteo más detallado a partir de diversas etapas y otras tienen un punto de vista más general, pero todas lo definen como un único proceso. En el caso de estudio, hubiera resultado muy complejo llevar adelante el proyecto de migración bajo este enfoque.

Construir un solo proceso sin la posibilidad de incorporar modificaciones y nuevas funcionalidades de manera iterativa hubiera sido muy difícil. A la complejidad de las transformaciones, que llevan a la necesidad de construir procedimientos en PL-SQL también complejos, hay que sumarle la necesidad de procesar numerosos cambios consecuencia del desarrollo del sistema. Basta analizar los ejemplos de cambios tipo uno, dos y tres planteados anteriormente e imaginar que no se pueden hacer de manera gradual e iterativa para visualizar las dificultades que hubiera implicado el empleo de una metodología clásica.

Además de las ventajas que aporta un enfoque iterativo para la realización de cambios y la incorporación gradual de funcionalidades, el modelo físico que divide las transformaciones sintácticas de las semánticas también resulta de gran utilidad. Muchos cambios tienen impacto en uno solo de los procesos de transformación, algunos impactaron en T1 y otros en T2 y aunque también existen situaciones en las que deben modificarse ambos, éstas constituyen un número menor, e incluso en ellas, el poder hacerlas en paralelo y separando la problemática sintáctica de la semántica simplifica significativamente el proceso general.

En otras palabras, de haber empleado en el caso de estudio una metodología clásica, no sólo se hubiera carecido del planteo iterativo, sino que además, cualquier cambio ya sea por

ampliación, modificación o corrección se hubiese tenido que procesar como de tipo tres y sobre un proceso de mayor complejidad al resolver las transformaciones sintácticas y semánticas juntas.

Por último, para la realización de pruebas, validaciones e informes de migración, el empleo de una metodología clásica hubiera implicado desarrollarlos de manera acorde al proceso, es decir, trabajando tanto sobre los resultados de las conversiones sintácticas como semánticas y sin la posibilidad de incorporarlos gradualmente conforme evolucione el proyecto.

Sin dudas esta situación genera informes complejos, más difíciles de elaborar, con resultados más extensos, difíciles de interpretar y con poca flexibilidad. Debe tenerse en cuenta que durante toda la experiencia realizada resulta de vital importancia contar con informes de migración ágiles y que permitan evaluar rápidamente el impacto de las modificaciones procesadas (independientemente del tipo que sean) para darle continuidad y velocidad a todo el proyecto de migración.

Para finalizar, todo puede resumirse de este modo: Durante la implementación realizada, el empleo de una metodología clásica hubiera, sin dudas, quitado agilidad al procesamiento de cambios, elevado la dificultad técnica, dificultado las tareas de planificación y también las de pruebas y validaciones y consecuentemente, demorado todo el proyecto y aumentado las posibilidades de fracaso según se analiza en los capítulos uno y dos.

Conclusiones

Durante este capítulo se realiza una implementación de la metodología propuesta. El caso de estudio resulta particularmente interesante porque se trata de una migración que ocurre en el contexto del desarrollo de un nuevo sistema de información y además tiene una alta complejidad en las conversiones que realiza, gran diferencia de tecnología entre las bases de origen y la de destino, gran cantidad de entidades a migrar y cada una con volúmenes de datos también significativos y una importante sensibilidad en la información ya que se trata de datos de personas que se utilizan en los procesos electorales de nuestro país y por lo tanto no admiten ningún error.

En toda la implementación, la dinámica del desarrollo del sistema implica la realización de numerosos cambios en la base de destino lo que impacta de manera directa en el proceso de

migración. Sin embargo, estos cambios pueden procesarse en su totalidad y sin un empleo excesivo de recursos.

La posibilidad de realizar una planificación general en la primera etapa, para luego realizar los correspondientes ajustes conforme se avance en las siguientes, es un factor desequilibrante que las metodologías clásicas no proveen. Sin dudas, el éxito en este proyecto tiene mucho que ver con lo antedicho ya que se hace posible planificar y administrar los recursos gradualmente.

Otro factor muy relevante es la flexibilidad para incorporar diversas herramientas tanto de programación como para la elaboración de informes y la realización de pruebas. Durante esta implementación las herramientas no son muchas ni diversas, pero se eligen conforme a las necesidades y características del proyecto y no en función de la metodología lo cual es muy positivo. Visto más ampliamente, la metodología no condiciona en ningún aspecto las herramientas empleadas en el desarrollo del proceso de migración ni el del sistema de información.

En el caso de los informes de migración y de las realizaciones de pruebas, durante toda la implementación resultan de gran importancia. Poder analizar los resultados de las conversiones sintácticas por un lado y por otro los de las conversiones semánticas es una ventaja muy significativa ya que permite paralelizar tanto la generación de informes como la obtención de conclusiones. Debe notarse que la herramienta usada para desarrollar los informes del primer proceso es diferente de la empleada en el segundo y los objetivos de cada uno son también distintos. La flexibilidad que plantea el proceso iterativo y el modelo de división física resulta de mucha utilidad durante todo el caso de estudio porque permite desarrollar y analizar los informes gradualmente y conforme a las necesidades que se van presentando.

El empleo de la metodología permite contar con una base de datos poblada completamente con registros migrados en una etapa muy temprana del desarrollo del sistema de información. Más concretamente, a la fecha, el mismo no se encuentra concluido y sin embargo ya se cuenta con una migración completa que brinda la salida esperada. Esto no sólo admite utilizarla en diversas tareas que van desde el desarrollo hasta las pruebas de implementación pasando por el análisis de rendimiento (tal como se menciona en apartados anteriores) sino que posibilita realizar tareas de testing con el sistema y la base de datos migrada al mismo tiempo. De este modo, se prueba el sistema y la migración simultáneamente lo que redunda en un significativo ahorro de recursos y

aumenta las posibilidades de descubrir defectos, tanto en el sistema como en la migración, en forma temprana.

La arquitectura física planteada también es en un elemento muy positivo durante esta implementación. Tener los procesos divididos no resulta un factor complejo ni difícil de administrar y a la vez permite realizar las modificaciones y cambios solicitados en forma gradual y paralela asignando recursos diferentes.

Respecto de la documentación elaborada, nuevamente la flexibilidad resulta una importante ventaja. Los documentos necesarios para registrar aspectos relevantes de la migración se van construyendo según la necesidad puntual de cada caso sin que la metodología los imponga ni determine. Así, por ejemplo, algunos procedimientos almacenados del segundo nivel se documentan con diagramas de flujo y otros con una simple descripción narrada. Algunas configuraciones del primer nivel se documentan en planillas creadas a tal fin. En otros aspectos, como por ejemplo el documento de control de cambios, se reutilizan artefactos que provienen del desarrollo del sistema de información.

Más allá de los aspectos positivos mencionados en los párrafos anteriores, sin ninguna duda, la característica iterativa de la metodología es la de mayor relevancia. Es gracias a su iteratividad que los cambios pueden procesarse gradualmente sin condicionar el desarrollo del sistema de información e incorporando modificaciones, correcciones de errores y nuevas funcionalidades de manera eficiente. Durante toda la implementación no se encuentra ningún caso en el que un cambio, del tipo que sea, no pueda procesarse o resulte en un efecto cascada que afecte toda la migración y lleve a un uso excesivo de recursos. Tal como se describe durante el presente capítulo, los cambios son muchos y de diverso tipo, sin embargo, todos pueden llevarse a cabo en los tiempos y con los recursos planificados.

Así como pueden describirse muchos aspectos positivos, también surgen de la implementación realizada elementos con los que deben tomarse algunas precauciones.

Al no tratarse de una metodología estándar, muy difundida ni adoptada por los fabricantes de motores de bases de datos y productos relacionados más importantes, su utilización en este desarrollo requiere capacitar al personal afectado con el costo y el tiempo que ello implica. Incluso habiendo realizado dicho proceso de capacitación, al comienzo resultan frecuentes errores

conceptuales en el desarrollo tanto de T1 como de T2 producto de la escaza experiencia de los especialistas en bases de datos intervinientes.

La flexibilidad, que es uno de los factores claves de la metodología, también es un elemento que puede volverse en contra si no se administra correctamente. Contar con un proceso flexible no debe llevar a la carencia de planificación y control, ni a la falta de documentación que resulte necesaria para el desenvolvimiento del proyecto.

Durante la primera etapa se debe definir la arquitectura física y los recursos técnicos para darle soporte. Tal como se explica anteriormente en este mismo capítulo, se decide disponer de dos servidores Oracle con una configuración de esquemas específica. Tomar esta decisión no es tarea simple y a la vez es algo que acompaña al proyecto de migración durante todo su ciclo de vida. Se deben tener en cuenta factores técnicos, operativos y económicos.

En estos aspectos, que en definitiva constituyen tres riesgos que deben identificarse con claridad, resulta de vital importancia acompañar el empleo de la metodología con una fuerte tarea de gerenciamiento del proyecto.

Por último, debe destacarse que las conclusiones obtenidas a partir de esta implementación no pueden verse como algo absolutamente general. Naturalmente, obtener los resultados mencionados durante este proyecto en particular no puede llevar a una generalización porque cada implementación tiene sus propias características. No obstante, debe destacarse que son resultados muy positivos y que indican un buen desenvolvimiento de la metodología. En los hechos ocurre que tratándose de una migración compleja (por la coexistencia de factores que ya se mencionaron), inserta en el desarrollo de un sistema de información también complejo, desde marzo a diciembre de 2017 se realiza con éxito una migración que no había podido llevarse adelante de manera positiva anteriormente.

Capítulo VI: Discusión de resultados, conclusiones y posibles trabajos futuros

Introducción

En el capítulo final de esta tesis se pretende hacer un breve resumen de sus aspectos centrales a los efectos de arribar claramente a las conclusiones finales.

En muy pocas palabras se puede resumir el sentido de este trabajo diciendo que, inicialmente se plantea una situación particular, ésta es la migración de datos en el contexto de sistemas en desarrollo con una base de destino inestable, luego se analizan y comparan algunas metodologías pertenecientes a distintos fabricantes y compañías relacionadas y finalmente se propone una nueva alternativa especialmente diseñada para las condiciones objeto de estudio.

En el capítulo anterior se realiza una implementación a los efectos de probar y validar la propuesta. Se puede ver que el desempeño de la metodología es el esperado a partir de su definición teórica y se obtienen resultados muy positivos pues se logra resolver una migración compleja dentro de los parámetros de tiempo y utilización de recursos esperados.

En el capítulo IV, como parte de la propuesta de la nueva metodología, se identifican seis aspectos relevantes y se exponen en una matriz. Esos mismos factores se analizan de igual manera en el capítulo III para las otras metodologías relevadas, pero no se los ha cuantificado ni comparado formalmente. Durante este apartado se utiliza el método analítico de procesos (AHP) (41) y el software Expert Choice para obtener una visión cuantificada y de mayor exactitud.

A partir de todos estos elementos, pueden enunciarse las conclusiones finales y proponer algunos posibles trabajos futuros que dan continuidad y complementan la actual propuesta.

Revisión de aspectos relevantes de las metodologías en el contexto de estudio

Las diferentes metodologías analizadas tienen, sin dudas, numerosos factores positivos y grandes ventajas que favorecen su utilización. No debe perderse de vista que en todos los casos se trata mecanismos de trabajo bastamente probados y propuestos por las compañías más importantes.

No obstante, en el contexto de estudio, se han podido aislar seis elementos que resultan particularmente relevantes para las situaciones planteadas. Éstos son los que se expresan en las matrices expuestas en los capítulos III y IV y pueden resumirse a continuación de la siguiente manera:

1) **Adaptabilidad a los cambios o flexibilidad**: Para migraciones que ocurren en paralelo al desarrollo de un nuevo sistema y donde consecuentemente las bases de datos de destino son inestables, la capacidad de adaptarse a los cambios es un factor altamente desequilibrante. Poder implementar cambios en el proceso de migración de manera continua, controlada y a bajo costo es sumamente importante y puede marcar la diferencia entre el éxito y el fracaso de todo el proyecto e incluso, puede ser un factor determinante en la implementación del nuevo sistema.

2) **Dependencia de la plataforma**: Si se plantea una metodología cuya utilización es viable sólo en determinadas tecnologías, todos los proyectos de migración que no responden a la misma quedan fuera de alcance. Para compañías que trabajan en forma exclusiva con un fabricante esto no es problema, pero si se busca una propuesta que pueda ser utilizada en diversos proyectos más allá de los recursos técnicos que se utilicen, se transforma en un factor de peso.

   Particularmente en el contexto de estudio, es un elemento muy importante porque en el desarrollo de un nuevo sistema de información no siempre puede elegirse la tecnología que da soporte a sus bases de datos e incluso, si ese fuera el caso, definitivamente no puede escogerse el soporte técnico para los datos de origen, aquellos que pertenecen a los viejos sistemas que van a ser reemplazados. Si bien todas las metodologías analizadas, incluso las más fuertemente relacionadas a una determinada

compañía, brindan ciertas flexibilidades sobre todo en los datos de origen, disponer de un planteo totalmente independiente de la plataforma es muy relevante.

3) **Definición de tareas de pruebas y validación:** Se trata de otro factor importante en toda metodología. Si bien cada proyecto tiene sus particularidades y puede requerir pruebas de mayor o menor cobertura e intensidad, resulta conveniente que la propia metodología defina los momentos en las distintas etapas donde implementar dichas pruebas y verificar que los procesos de transformación estén arrojando resultados correctos.
   En el escenario objeto de análisis, como se esperan cambios frecuentes en la estructura de destino que impactan en las transformaciones, este ítem alcanza una significativa importancia.
   Al mismo tiempo, debe buscarse que la definición metodológica no sea rígida, es decir, que no imponga limitaciones (como por ejemplo el uso de herramientas particulares) para que no se pierda la flexibilidad que permite adaptarla a las necesidades de las distintas implementaciones.

4) **Complejidad**: Es deseable que una metodología abarque todos los aspectos del proceso de migración, pero que lo haga de la manera más simple posible. Definiciones de mecanismos de trabajo y de control más sencillas pueden implementarse rápidamente, con menor uso de recursos tanto técnicos como humanos y suelen ser más flexibles ya que dejan librado algunos aspectos a las características propias de cada proyecto de migración.

5) **Nivel de detalle**: Las metodologías que se definen en pocos pasos son fáciles de entender, pero no necesariamente simples de implementar. Como ya se ha mencionado antes, los procesos de migración son complejos y por ende resulta conveniente disponer de metodologías que abarquen con buen nivel de detalle todas las etapas del proyecto.

6) **Definición de tareas de organización y planificación**: Algunas metodologías tienen etapas específicas donde se contempla la realización de planes y la organización del proceso. Son planteos que hacen un fuerte énfasis en el management como recurso para lograr una implementación eficiente. Éste es un factor importante en todos los escenarios,

pero más aún en caso particular de estudio porque se requiere un continuo ajuste de los planes y una gestión permanente de los cambios en cada iteración.

Más allá de todos los aspectos abarcados en los capítulos anteriores, es importante poder cuantificar estos seis ítems y evaluar la propuesta en función de ellos. En los párrafos siguientes se realiza un análisis jerárquico de procesos para obtener resultados objetivos y precisos.

Análisis por método AHP

En este apartado se desarrolla un estudio por medio del método AHP creado por Thomas Saaty para análisis y toma de decisiones a partir de múltiples criterios (41).

A partir de los factores enumerados anteriormente, se puede realizar un análisis jerárquico tomando estos elementos y asignando prioridades de unos sobre otros al hacer una comparación de a pares.

La asignación de estas prioridades o pesos es una cuantificación subjetiva, cambiando los valores se pueden obtener resultados diferentes, pero incluso teniendo en cuenta esta característica del propio método, el análisis resulta muy interesante sobre todo si las cuantificaciones son lo más aproximadas y razonables posible.

Indudablemente cada proyecto tiene sus propias características, por tal motivo, no es factible realizar un análisis universal y válido para todos los casos. Por ejemplo, si en un proyecto se estima que pueden presentarse numerosos cambios, la flexibilidad resulta muy importante y al momento de realizar este estudio debe dársele mayor peso que a otros factores. De la misma manera pueden existir casos en los que las fuentes de origen de los datos sean muy variadas o incluso desconocidas, aquí cobra mayor importancia la independencia de la plataforma y, por el contrario, si se pretende trabajar sobre una tecnología única y claramente definida el mismo elemento pierde peso.

Asignación de pesos a los factores

A partir de las características generales del contexto objeto de estudio, se le asigna un fuerte peso a la flexibilidad ya que es el eje central de la metodología y constituye un factor decisivo durante las definiciones del capítulo IV y la implementación descripta en el capítulo V. La razón de ser de la metodología es justamente aquellos casos en los que la base de destino recibe cambios frecuentes y se requiere gran flexibilidad. En proyectos de este tipo, resulta ser un componente determinante.

Como segundo factor de importancia se considera a la independencia de la plataforma. Como ya se ha dicho, en los casos en los que un nuevo sistema reemplaza a otros no se tiene control sobre las plataformas de los datos de origen y consecuentemente resulta relevante que la metodología pueda emplearse independientemente de ello.

En tercer lugar, se colocan a la definición de tareas de prueba y de planificación general. No caben dudas de que éstas resultan sumamente importante y que es muy positivo que una metodología las defina, pero también debe considerarse que estas actividades pueden desarrollarse más allá de lo establecido por la metodología. En otras palabras, es muy conveniente que la metodología aporte sus propias definiciones, pero son las características de cada proyecto las que determinan la manera de implementarlas y la necesidad real de ejecutarlas.

Por último, se ha colocado la complejidad y el nivel de detalle. No es porque estos elementos resulten poco importantes sino porque en el contexto de estudio la implementación resulta compleja, independientemente del mecanismo de trabajo que se proponga, debido a la necesidad de procesar cambios como se ha expresado en varias oportunidades. El nivel de detalle de la metodología no resulta un factor decisivo ya que, con mayor o menor grado de definición, se debe encontrar una forma de interactuar con los cambios frecuentes.

Cabe destacar que todos los factores analizados son de gran relevancia. Justamente por esta razón son motivo de análisis en el capítulo IV. Que uno se considere en este apartado de mayor o menor importancia frente a otros es una apreciación relativa. En términos absolutos, todos son importantes.

La imagen siguiente muestra la matriz de asignación de prioridades de los factores y un cuadro comparativo. Para construirá se ha usado el software Expert Choice al igual que en todos los gráficos siguientes.

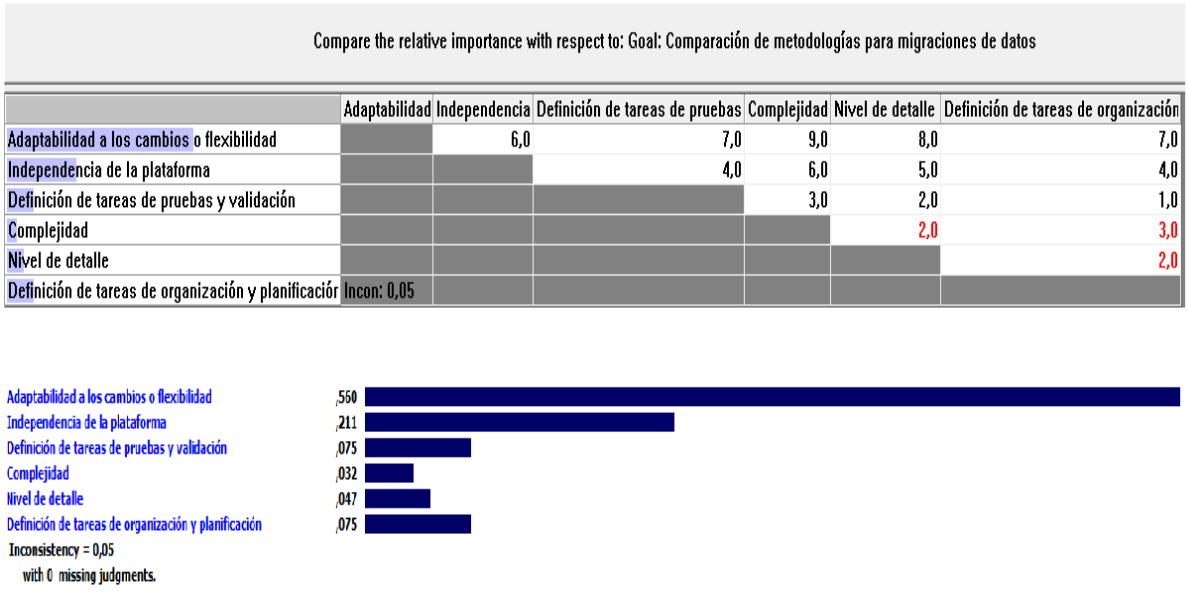


Figura 35: Matriz de Asignación de prioridades en Expert Choice

Comparación de alternativas

A partir de la definición de factores y su correspondiente asignación de prioridades el método AHP propone la definición de alternativas que deben compararse de a pares en función de cada factor, dando así origen a una serie de matrices (41) .

Nuevamente caben las mimas deducciones y aclaraciones. No es posible hacer en este sentido un estudio con conclusiones absolutas y universales ya que la manera en que cada metodología se adapta a cada factor depende de las características particulares del proyecto, pero en general y situándose en el contexto de estudio, en un caso con las características de la implementación realizada en el capítulo anterior, se pueden proponer los valores que se muestran a continuación.

La figura siguiente expresa la comparación de las distintas metodologías frente al factor de flexibilidad, utilizando el software Expert Choice.

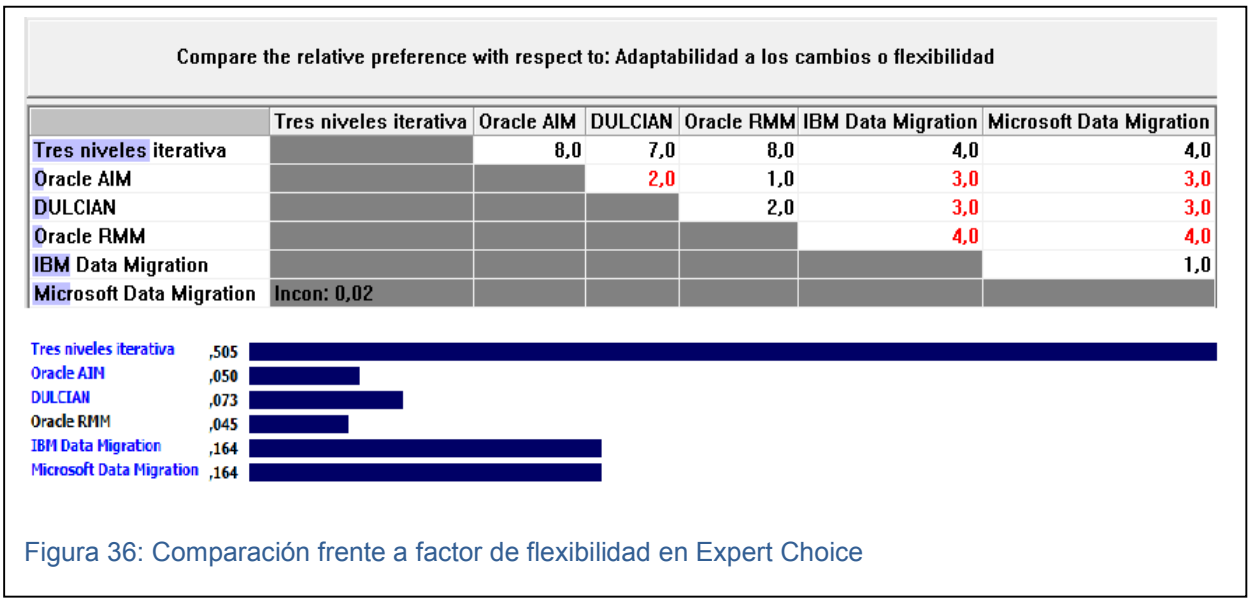
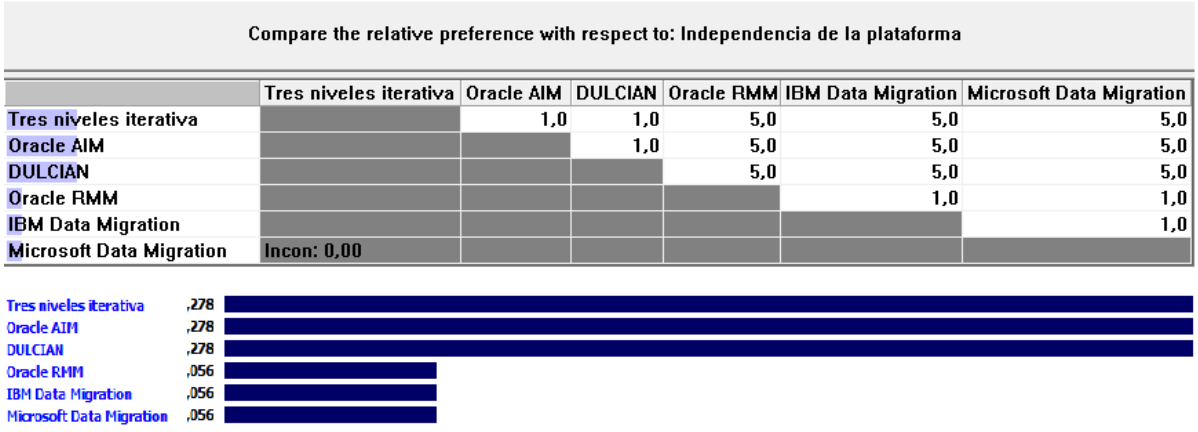


Figura 36: Comparación frente a factor de flexibilidad en Expert Choice

Como muestra la figura, en esta apreciación se considera a la metodología de tres niveles como muy flexible y adaptable. Luego las metodologías de IBM y de Microsoft porque al proponer pocos pasos y ser muy generales dejan un margen amplio para su implementación y por último quedan AIM, DULCIAN y RMM ya que son extensas y detalladas (lo cual constituye una ventaja en otros factores como se muestra más adelante) pero no definen ningún mecanismo de gestión de cambios frecuentes.

De la misma manera puede exponerse la comparación de todas las metodologías frente al factor de independencia de la plataforma, como lo muestra la figura siguiente.



Compare the relative preference with respect to: Independencia de la plataforma

	Tres niveles iterativa	Oracle AIM	DULCIAN	Oracle RMM	IBM Data Migration	Microsoft Data Migration
Tres niveles iterativa		1,0	1,0	5,0	5,0	5,0
Oracle AIM			1,0	5,0	5,0	5,0
DULCIAN				5,0	5,0	5,0
Oracle RMM					1,0	1,0
IBM Data Migration						1,0
Microsoft Data Migration	Incon: 0,00					

Figura 37: Comparación frente al factor de independencia de la plataforma en Expert Choice

En este caso se han considerado independientes de la plataforma a la metodología de tres niveles iterativa, AIM y DULCIAN según lo analizado oportunamente. Oracle RMM, Microsoft Data Migration e IBM Data Migration reciben un valor bajo en este factor ya que son para plataformas específicas y están definidas en función de determinadas herramientas.

Respecto de las tareas de prueba y validación, todas las metodologías analizadas tienen sus propias definiciones. En general son bien detalladas en este aspecto sólo que DULCIAN y Microsoft Data Migration no tienen definiciones específicas, sino que incluyen estas actividades con otras propias de la implementación y demás etapas.

La figura siguiente muestra la ponderación realizada.

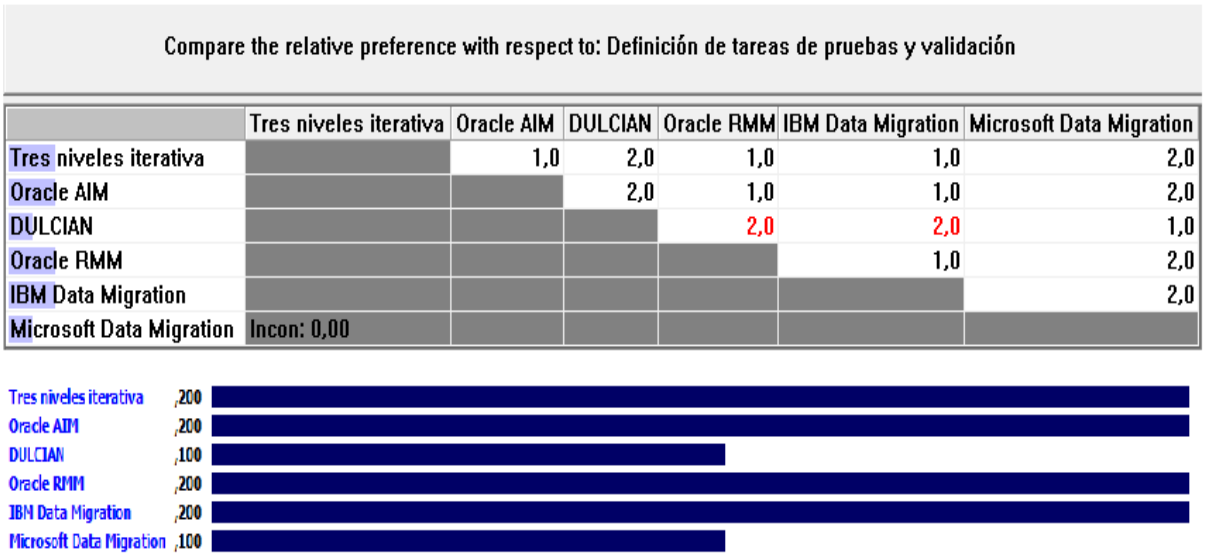


Figura 38: Comparación frente al factor de tareas de prueba y validación en Expert Choice

Con un criterio similar puede ponderarse el factor de complejidad. En general Microsoft Data Migration e IBM Data Migration son simples ya que definen pocas etapas y consecuentemente su comprensión e implementación son sencillas. Dulcian y Oracle RMM están en un punto intermedio mientras que Oracle AIM y Tres niveles Iterativa tienen una cantidad importante de etapas, son más difíciles de comprender a la vez que pueden requerir elementos adicionales en su implementación. Debe tenerse en cuenta que estas dos fueron desarrolladas para migraciones en contextos también más complejos.

La figura siguiente muestra la valoración realizada.

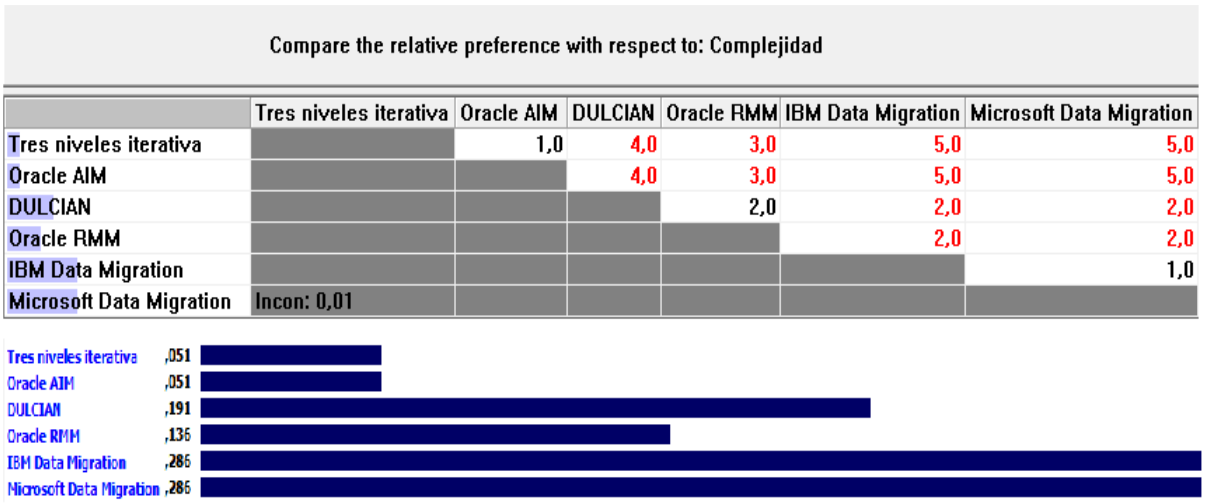


Figura 39: Comparación frente al factor de Complejidad en Expert Choice

El siguiente elemento a confrontar es el nivel de detalle. En general las metodologías más simples tienen un nivel de detalle menor y se adaptan mejor a situaciones sencillas mientras que las metodologías complejas definen sus pasos y etapas minuciosamente.

Por esta razón tanto Microsoft Data Migration como IBM Data Migration reciben una puntuación baja en este aspecto mientras que los otros planteos se consideran todos con buen nivel de detalle.

La figura siguiente ilustra dicha valoración.

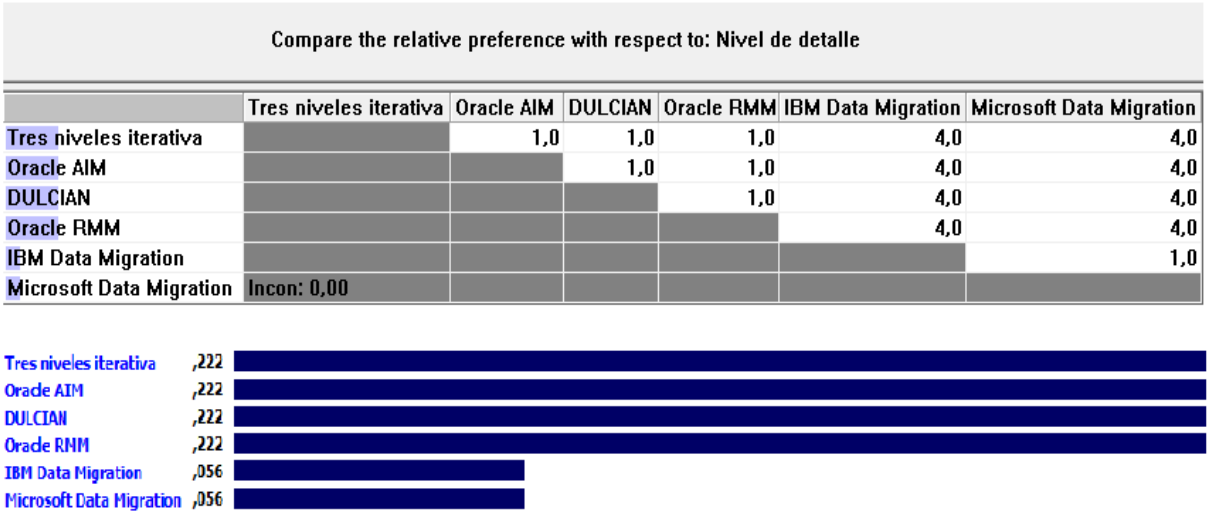


Figura 40: Comparación frente al factor de Nivel de Detalle en Expert Choice

Por último, resta ponderar la definición de tareas de organización y planificación. Todas las metodologías aportan sus definiciones, pero nuevamente ocurre que las más simples tienen escasos elementos de definición mientras que las más complejas describen etapas relacionadas a este tipo de actividades.

En el caso particular de DULCIAN define dos etapas afectadas a la planificación y organización general pero no como una tarea de fuerte presencia continua. Por esta razón recibe un nivel intermedio en la comparación.

La imagen siguiente muestra las ponderaciones.

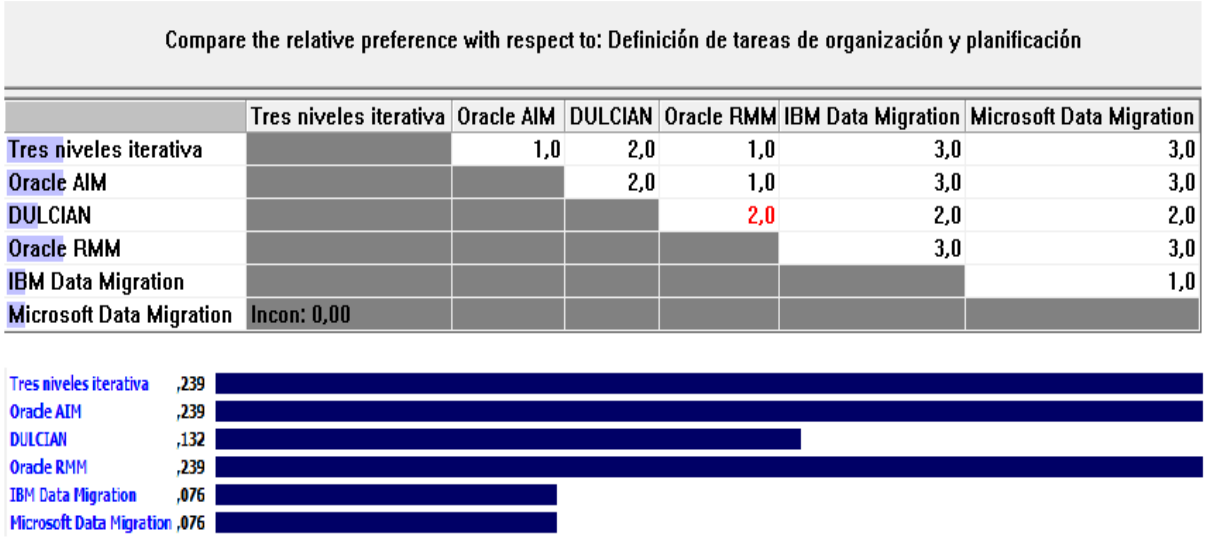


Figura 41: Comparación frente al factor de Tareas de Planificación y Organizacion en Expert Choice

Antes de analizar los resultados obtenidos por este método, una vez más, se debe resaltar el hecho de que las valoraciones pueden variar de un proyecto de migración a otro. En general se ha intentado mantener una postura objetiva y situada dentro del contexto de estudio, tomando a la vez, la experiencia de implementación realizada en el capítulo anterior. Además, como se expone más adelante, pequeñas variaciones en los criterios no afectan los resultados finales de manera significativa, sólo se obtienen resultados diferentes si se suponen situaciones con necesidades radicalmente distintas.

Obtención de resultados

A partir de las valoraciones y comparaciones realizadas, utilizando el método AHP se puede indicar cual metodología resulta más aplicable para el contexto de estudio.

La figura siguiente muestra los resultados obtenidos por este método utilizando el software Expert Choice.

Figura 42: Resultado final método AHP utilizando Expert Choice.

Como puede advertirse, a partir de las valoraciones realizadas y de los criterios empleados, la metodología propuesta resulta significativamente más conveniente dentro del contexto de estudio.

Estos resultados son coherentes con el análisis realizado en el capítulo IV y con la experiencia recogida a partir de la implementación descripta en el capítulo V. En un entorno donde la aparición de cambios es frecuente y la flexibilidad tiene una importancia tan predominante sobre los demás factores, un mecanismo de trabajo que se comporte en consecuencia es altamente preferible.

Incluso si se realizan variaciones en las cuantificaciones, por ejemplo, considerando a Microsoft Data Migration e IBM Data Migration con una puntuación más alta en el factor nivel de detalle y a DULCIAN con mejor puntuación en las tareas de organización y planificación y otras variantes similares, se obtienen resultados finales que conllevan a las mismas conclusiones. Esto se debe a la alta importancia que se le ha dado al factor de flexibilidad.

Por otro lado, si se disminuye fuertemente la ponderación del factor de flexibilidad, el de independencia de la plataforma y el del nivel de detalle y a la vez se aumenta muy significativamente el factor de complejidad, se obtienen resultados diferentes. Esta situación se corresponde con una migración en la que no se busca procesar cambios, tampoco se pretende trabajar con tecnologías diversas y, por el contrario, resulta muy importante que el planteo metodológico sea simple.

Éste también es un resultado esperado. Al analizar las desventajas de la propuesta se ha dicho que su razón de ser no es la simpleza. No se busca en mecanismo simple y económico en términos de uso general de recursos, sino que se busca uno muy flexible y adaptable.

La figura siguiente muestra esas variaciones y resultados utilizando el mismo software como herramienta.

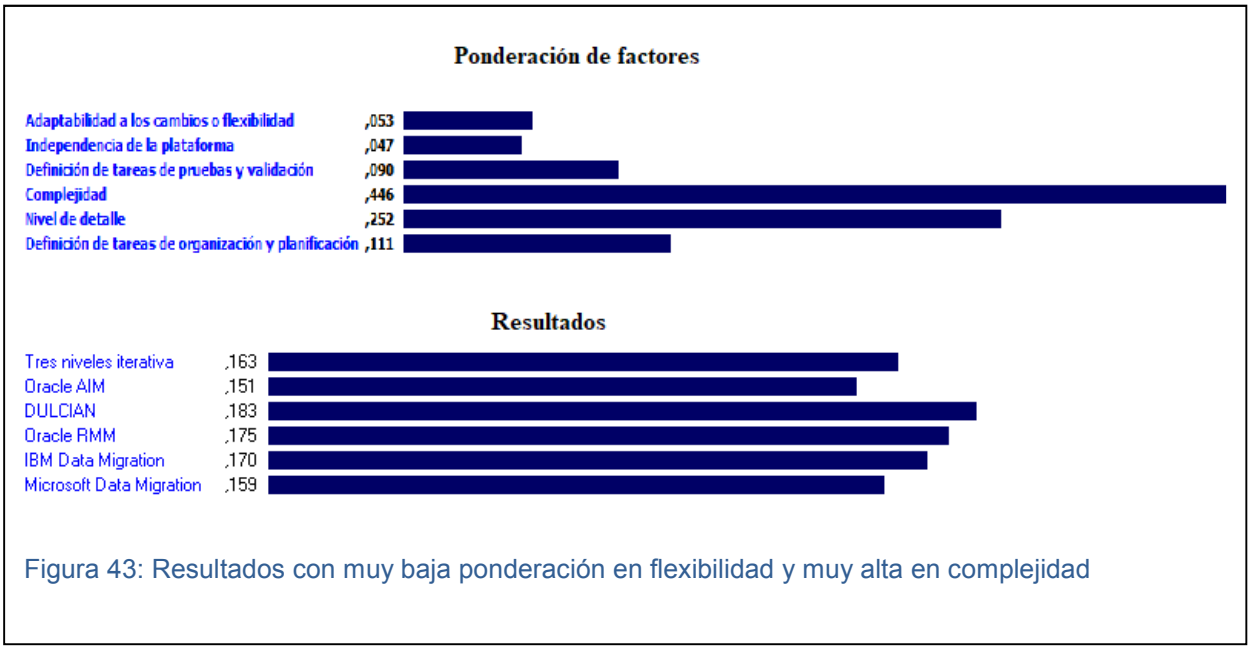


Figura 43: Resultados con muy baja ponderación en flexibilidad y muy alta en complejidad

Conclusiones finales

A modo de conclusión final de este trabajo de tesis resulta conveniente extraer en orden los elementos más relevantes de cada capítulo a los efectos de hacer, muy sintéticamente, un razonamiento final.

En el primer capítulo se presentó el contexto general y se hicieron notar las dificultades que frecuentemente atraviesa una migración que ocurre en el contexto del desarrollo de un nuevo sistema. Estas características especiales son la motivación de la propuesta que se elabora posteriormente.

En el segundo capítulo se realiza un análisis detallado de la situación objeto de estudio y se muestran estadísticas de diversas fuentes que, en definitiva, llevan a concluir que los procesos de migración son complejos, frecuentemente tienen dificultades y muy a menudo no se concluyen dentro de los planes originales. Si a esto se le suma inestabilidad en la base de destino, se obtiene un contexto general aún más desfavorable.

En el tercer capítulo se analizan algunas metodologías propuestas por diversos fabricantes de motores de bases de datos y productos relacionados, como así también de compañías especializadas en el tema. Es importante conocerlas para ver cómo se desempeñan en el marco de estudio y poder compararlas.

El cuarto capítulo resulta absolutamente central. Aquí se hace una propuesta metodológica especialmente adaptada para las migraciones que ocurren en paralelo a un sistema en desarrollo. Como principal resultado se obtiene una propuesta muy completa y detallada, con todas sus fases y etapas, además de la descripción del modelo físico necesario para su implementación. Se describen con precisión sus componentes, sus dos procesos iterativos, sus mecanismos para procesamientos de cambios y forma de trabajo general.

El quinto capítulo describe las experiencias obtenidas a partir de una implementación real. Se trata de un proyecto de migración con muchas dificultades y que ocurre dentro del contexto del desarrollo de un nuevo sistema de grandes proporciones y alta complejidad. Debe tomar datos de orígenes numerosos y diversos, realizar transformaciones tanto sintácticas como semánticas complejas y mantener actualizada una base de destino inestable y de gran volumen.

El uso de la metodología permitió el desarrollo de esta migración con muy buenos resultados en término de las transformaciones realizadas, los objetivos cumplidos y el uso de recursos.

Por último, en el actual capítulo, se ha utilizado el método AHP y particularmente el software Expert Choice para tomar los factores relevantes analizados con anterioridad y realizar una comparación cuantitativa. Si bien las cuantificaciones tienen un grado de subjetividad, se ha intentado que las mismas guarden relación con la implementación realizada y también se han explorado algunas variaciones.

Según estas comparaciones, la metodología propuesta resulta muy conveniente si la flexibilidad es un factor de alto peso, situación que concuerda con el planteo inicial y la motivación expuesta en el capítulo uno.

Más allá de este breve resumen, y de los valores obtenidos por el método AHP, lo más importante de resaltar y concluir es lo que se obtiene de la visión general de este trabajo. La metodología propuesta aporta ventajas significativas y constituye una buena alternativa si se debe abordar una migración dentro de las situaciones descriptas. Ha demostrado un funcionamiento óptimo dentro de la implementación realizada y esto concuerda con los análisis posteriores.

Definitivamente estas ventajas tienen su costo. Tal como se analizó en el capítulo IV y afirmó en el V, no se trata de una metodología simple. Implementarla y mantenerla en funcionamiento requiere recursos que se justifican si la base de destino es inestable y si resulta relevante procesar los cambios eficientemente.

En escenarios más simples, sin la existencia de cambios o incluso si se presentan algunos, pero las transformaciones son sencillas y el volumen de datos es pequeño, probablemente resulte conveniente tomar otro camino. No debe olvidarse que las otras alternativas analizadas y comparadas corresponden a las principales compañías relacionadas a la temática y por lo tanto puede esperarse que constituyan buenas opciones.

Pero si se requiere flexibilidad y adaptabilidad ante cambios en la base de destino en escenarios complejos, sin duda la metodología propuesta es la mejor opción.

En definitiva, como aporte de este trabajo se ha obtenido una metodología especialmente diseñada y que ha mostrado un buen desempeño en el contexto propuesto como objeto de estudio desde el comienzo, esto es, migraciones complejas que ocurren paralelamente al desarrollo de un nuevo sistema de información.

Propuesta de posibles trabajos futuros

Se mencionan a continuación algunos trabajos futuros que puede resultar particularmente interesantes y que darían continuidad a la actual propuesta.

Propuesta de trabajo futuro uno

Utilización del modelo físico como herramienta para la detección temprana de inconsistencias en los requerimientos. Durante la implementación descripta en el capítulo V, frecuentemente ocurre que un cambio en la base de datos, procesado en forma acorde con el planteo metodológico, deja expuestas contradicciones e inconsistencias en el modelo de negocio que define determinadas reglas que no pueden cumplirse con los datos migrados.

Más allá del empleo de esta metodología o cualquier otra, resulta importante descubrir estas situaciones de la manera más temprana posible para minimizar su impacto en todo el proceso. Probablemente el modelo de procesamientos de cambios puede usarse con este fin.

Propuesta de trabajo futuro dos

Utilización del modelo propuesto en bases de datos no relacionales y/o NOSQL. Si se observa la propuesta se advierte que todo su planteo, desde el comienzo hasta las conclusiones pasando por la implementación se basa fuertemente en bases de datos relacionales y bajo estándares SQL, pero en realidad no existen elementos que hagan pensar que su empleo no es factible en otros paradigmas.

Resulta interesante analizar el empleo de la metodología en migraciones que ocurran, por ejemplo, con bases de datos de objetos o incluso en migraciones hibridas, teniendo quizás un origen relacional y un destino de objetos o a la inversa.

Propuesta de trabajo futuro tres

Si se observa el planteo metodológico de los procesos de transformación T1 y T2 descriptos en el capítulo IV, debe notarse que se deja abierta la manera en la que se gestionan los ciclos iterativos. La propuesta no define un mecanismo particular y esto es algo positivo porque así puede adaptarse a las características de cada implementación.

Sin perjuicio de esta ventaja, que en definitiva aporta flexibilidad, puede investigarse el empleo de metodologías ágiles para gestionar los ciclos tanto de T1 como de T2. En su mayor medida, estos enfoques tienen su origen en el desarrollo de sistemas y no en las migraciones de

bases de datos, pero los procesos iterativos responden a las mimas características generales y de allí la importancia de su análisis.

Referencias

1. **Alan, R.** The Serials Data Migration Dilemma. [Online] Julio 17, 2002. [Cited: Julio 25, 2017.] http://dx.doi.org/10.1300/J124v20n01_03.

2. **Sommerville, I.** *Ingeniería del software séptima edición.* Madrid : Pearson Educación S.A., 2005.

3. **Vitthal, S., et al.** Data Migration System in Heterogeneous Database. [Online] Marzo 2013. [Cited: Julio 10, 2016.] http://www.ijesit.com/Volume%202/Issue%202/IJESIT201302_14.pdf.

4. **Reeves, L.** *A Manager's Guide to Data Warehousing.* Indianápolis : Wiley Publishing, 2009.

5. **IBM Corporation.** Migración a otra base de datos. [Online] 2013. [Cited: Julio 15, 2016.] http://www.ibm.com/support/knowledgecenter/SS69YH_6.0.0/com.ibm.spss.cads.config.doc/model_management/thin/migration_db_platfrom_switch.html?lang=es.

6. **Lee, B.** SQL Server: Administrar la Migración. [Online] 2011. [Cited: Julio 15, 2016.] https://technet.microsoft.com/es-es/magazine/hh334645.aspx.

7. **ORACLE Corporation.** Migrate to Oracle Database with SQL Developer. [Online] [Cited: Julio 15, 2016.] http://www.oracle.com/technetwork/database/migration/index-084442.html.

8. **Kengchon, C.** Helping clients deal with economic uncertainty: Getting more from existing hardware. [Online] Febrero 2009. [Cited: Setiembre 15, 2016.] http://www-07.ibm.com/th/events/ibmuni2009/downloads/Other/GTS_2009_Thailand_University_gts.pdf.

9. **Oracle Corporation.** Successful Data Migration . [Online] Octubre 2011. [Cited: Octubre 15, 2017.] http://www.oracle.com/technetwork/middleware/oedq/successful-data-migration-wp-1555708.pdf.

10. **Gonsalez Lau, C. and Aristiabal Moreno, C.** ESTADO DEL ARTE DE PRUEBAS DE BASES DE DATOS PARA LA MIGRACIÓN Y VALIDACIÓN DE DATOS. Tesis de grado Ingeniería de Sistemas, Escuela de Ingeniería Departamento de Informática y Sistemas, Universidad EAFIT, Medellin Colombia. *eafit.edu.co.* [Online] 2007. [Cited: Enero 30, 2016.] https://repository.eafit.edu.co/xmlui/bitstream/handle/10784/2399/GonzalezLau_ConsueloInes_2007.pdf?sequence=1&isAllowed=y.

11. **Hotek, M.** *SQL Server 2008 paso a paso.* Madrid : Ediciones ANAYA, 2009.

12. **ORACLE Corporation.** Oracle Warehouse Builder User's Guide. [Online] Enero 2009. [Cited: Septiembre 02, 2016.] https://docs.oracle.com/cd/B28359_01/owb.111/b31278/concept_overview.htm.

13. **Natarajan, J., et al.** *Pro T-SQL 2012 Programmer's Guide.* New York : Springer Science+Business Media, 2012.

14. **Perez, C.** *Oracle PL/SQL.* Madrid : RA-MA, 2008.

15. **Hurley, N.** Data Migration: Everyone is doing it but are they doing it right? [Online] Marzo 2005. [Cited: Enero 26, 2016.] http://www.enterprisestrategygroup.com/ESGPublications/BriefPopup.asp?ReportID=374.

16. **ORACLE Corporation.** Put Your Data First or Your Migration Will Come Last. [Online] [Cited: Julio 15, 2016.] http://www.oracle.com/us/products/middleware/data-integration/enterprise-data-quality/data-migration-wp-2345281.pdf.

17. **Powerdata.** Plan de migración de datos: Elementos imprescindibles. [Online] Enero 2015. [Cited: Octubre 15, 2016.] http://blog.powerdata.es/el-valor-de-la-gestion-de-datos/bid/397552/Plan-de-migraci-n-de-datos-elementos-imprescindibles.

18. **DB-Engines.** DB-Engine Ranking. [Online] Noviembre 2016. [Cited: Noviembre 30, 2016.] http://db-engines.com/en/ranking.

19. —. Method of calculating the scores of the DB-Engines Ranking. [Online] Noviembre 2016. [Cited: Noviembre 30, 2016.] http://db-engines.com/en/ranking_definition.

20. **Computer Weekly.** Choosing the right database management system. [Online] Enero 2007. [Cited: Noviembre 05, 2016.] http://www.computerweekly.com/feature/Choosing-the-right-database-management-system.

21. **Marketing Directo.** ORACLE LIDERA EL MERCADO DE LAS BASES DE DATOS EN 2006. [Online] Mayo 2007. [Cited: Diciembre 09, 2016.] https://www.marketingdirecto.com/marketing-general/marketing/oracle-lidera-el-mercado-de-las-bases-de-datos-en-2006.

22. **Crum, J.** *Using Oracle 11i.* United States of America : BOSS Corporation, 2002.

23. **Oracle ERP APPs Guide.** List of AIM Documents. [Online] Marzo 2011. [Cited: Diciembre 16, 2016.] http://www.oracleerpappsguide.com/2011/03/list-of-aim-documents.html.

24. **OracleApps Epicenter.** Conversion & AIM's Deliverables. [Online] Agosto 2007. [Cited: Marzo 15, 2017.] http://www.oracleappshub.com/conversion/conversion-aims-deliverables/.

25. **Hudicka, J. R.** The Complete Data Migration Methodology. [Online] Abril 1998. [Cited: Marzo 15, 2017.] http://dulcian.com/articles/overview_data_migration_methodology.htm.

26. **Caraguay Martinez, S.** METODOLOGÍA PARA MIGRACIÓN DE DATOS QUE PERMITA ASEGURAR Y CONSERVAR LA INTEGRIDAD Y CONSISTENCIA DE LA INFORMACIÓN ADMINISTRADA POR LA EMPRESA VSYSTEMS. . *Trabajo de investigación científica Previa a la obtención del Grado Académico de Magister en Gestión de Bases de Datos, Facultad de Ingeniería en Sistemas, Electrónica e Industrial, Universidad Técnica de Ambato, Ambato, Ecuador.* [Online] 2012. [Cited: Marzo 08, 2017.] http://repo.uta.edu.ec/bitstream/123456789/3003/1/Tesis_t776mbd.pdf.

27. **IBM Global Technology Services.** Best practices for data migration. Methodologies for planning, designing, migrating and validating data migration. [Online] Julio 2007. [Cited: Marzo 09, 2017.] https://www-935.ibm.com/services/us/gts/pdf/softek-best-practices-data-migration.pdf.

28. **Gajre, S., Bordia, B. and Soni, V.** Competitive Database Migration Tools and Methodology. [Online] Marzo 2015. [Cited: Marzo 29, 2017.] https://social.technet.microsoft.com/Search/es-AR?query=database%20migration%20methodology&ac=5.

29. **Daly, D.** SQL Developer Oracle Migration Workbench Taking Database Migration to the next level. [Online] [Cited: Abril 11, 2017.] http://www.oracle.com/technetwork/topics/sqldevmigrationworkbench-132899.pdf.

30. **Oracle Corporation.** MySql Workbench 6.3. [Online] 2017. [Cited: Abril 12, 2017.] https://www.mysql.com/products/workbench/.

31. **ORACLE Corporation.** MySql Workbench Database Migration. [Online] 2017. [Cited: Abril 12, 2017.] https://www.mysql.com/products/workbench/migrate/.

32. **Welly, L.** SQL Server: Manage the Migration. [Online] 2011. [Cited: Abril 15, 2017.] https://technet.microsoft.com/en-us/library/hh334645.aspx.

33. **MICROSOFT Corporation.** Tareas de Integration Services. [Online] 2017. [Cited: Abril 18, 2017.] https://msdn.microsoft.com/es-es/library/ms139892.aspx.

34. **IBM Corporation.** Softek Transparent Data Migration Facility (TDMF). [Online] [Cited: Abril 18, 2017.] https://www-935.ibm.com/services/es/es/it-services/softek-transparent-data-migration-facility-tdmf.html.

35. **Kreuz, B. and Higginson, A.** Migrating Oracle Databases. [Online] Julio 2014. [Cited: Mayo 23, 2017.] http://www.oracle.com/us/support/library/database-migration-service-wp-2220890.pdf.

36. **Startin Point Pet. Ltd.** Methodology of Data Migration. [Online] [Cited: Abril 18, 2017.] http://www.startinpoint.com/index.php/services/migration-services.

37. **Black Blade Associates, Inc.** Migration Process. [Online] [Cited: Abril 18, 2017.] http://www.blackbladeinc.com/en-us/services/PublishingImages/migration-process.png.

38. **Mahadev, A.** Database Migration : An In Depth look!! [Online] Marzo 2010. [Cited: Junio 06, 2017.] https://db2hitman.files.wordpress.com/2010/03/database-migration-word.pdf.

39. **Barale, Lorena, et al.** SEDICI - Sistema de Gestión Electoral. *Sociedad Argentina de Informática (SADIO) (45JAIIO).* [Online] Setiembre 2016. [Cited: Agosto 29, 2017.] http://sedici.unlp.edu.ar/handle/10915/58344.

40. **Oracle Corporation.** Database Utilities - SQL*Loader. [Online] 2017. [Cited: Septiembre 14, 2017.] http://docs.oracle.com/cd/B19306_01/server.102/b14215/part_ldr.htm.

41. **Saaty, Thomas L.** *Mathematical models for decision support.* Berlin : Springer, 1988. 978-3-642-83555-1.

Anexos

Anexo I: Documento de arquitectura de la implementación de la metodología.

Migración de datos. Documento de Arquitectura

Proyecto :	**SGE – Sistema de Gestión Electoral**	Versión :	1.0

Migración de datos. Documento de Arquitectura

SGE- Sistema de Gestión Electoral

Documento de arquitectura del proceso de migración de datos

Versión 3.0

1 -

Migración de datos. Documento de Arquitectura

Proyecto:	**SGE – Sistema de Gestión Electoral**	Versión:	1.0

Contenido

Historial de cambios

Registro de REVISIONES			
Revisión	**Fecha**	**Revisado por**	**Observación**
01	01/04/2017	Silvio Serra	Versión inicial
02	04/03/2018	Felipe Steffolani	Revisión
03	06/03/2018	Silvio Serra	Ampliación
04			

Registro de APROBACIÓN				
Entidad	**Apellido y Nombre**	**Cargo**	**Firma**	**Fecha**

Migración de datos. Documento de Arquitectura

Proyecto:	**SGE – Sistema de Gestión Electoral**	Versión:	1.0

Introducción

El presente documente tiene por objetivo describir la arquitectura general y las bases fundamentales del proceso de migración de datos utilizado para electores, historiales, partidos políticos y todas las entidades periféricas relacionadas.

El proceso es único para todas las entidades alcanzadas por la migración de datos, pero está dividido en grupos y en niveles para hacer factible su implementación y su monitoreo en tiempo de ejecución.

Precondiciones y elementos básicos de definición

Los siguientes son factores que condicionan la definición de arquitectura:

- Las fuentes de origen son las bases de datos de los sistemas SIE, RNAP, RENAPER y BAHARA. A los efectos de la migración se realiza sobre ellos una extracción de datos en formato de archivos de texto plano que constituyen el origen del proceso. De esta manera no se interfiere con el funcionamiento de los sistemas y no se acceden directamente a sus repositorios. El proceso de migración debe ser capaz de realizar todas las conversiones de tipos, longitudes y formatos desde estos archivos de texto de origen hasta la base de datos Oracle de destino final que es la que utiliza el nuevo sistema SGE. Este tipo de conversiones se denominan “conversiones sintácticas”. Debe destacarse que la obtención de dichos archivos de texto desde los sistemas de origen está fuera del alcance del proceso de migración.
- Los sistemas mencionados en el punto anterior, desde los que proviene la información, no solo difieren en su tecnología con el nuevo sistema SGE, sino que tienen grandes diferencias en su modelo de negocio, por lo tanto, la base de datos de destino de la migración es muy diferente estructuralmente. El proceso de migración debe realizar todas las transformaciones necesarias para almacenar la información migrada en la base de destino, pero respetando la estructura, relaciones, validaciones y todos los elementos necesarios para que SGE pueda funcionar a partir de ella. Este tipo de conversiones se denominan “conversiones semánticas”.
- No resulta factible esperar a que SGE se encuentre totalmente concluido para comenzar con el desarrollo de la migración porque esto retrasaría muy significativamente su puesta en marcha, consecuentemente, la migración debe

desarrollarse en paralelo al sistema. Esto provoca que los cambios en la base de datos que se encuentra en construcción impacten sobre el proceso. En este sentido, es una prioridad absoluta que la arquitectura y la metodología empleada permitan hacer las dos conversiones mencionadas en los puntos anteriores, pero con suficiente flexibilidad para incorporar los cambios gradualmente y al menor costo posible.

Descripción general de la arquitectura

Teniendo en cuenta las precondiciones del apartado anterior, se propone la utilización de una metodología iterativa apoyada en una arquitectura de tres niveles. La misma se describe a continuación.

El primer nivel está compuesto por los archivos de texto con la información original. A partir de ellos, un proceso de transformación sintáctica llamado T1, tiene por objetivo tomarlos como entrada y realizar todas las conversiones sintácticas necesarias para poder almacenarlos en una base de datos Oracle con una estructura similar a la original.

En otras palabras, T1 es responsable de incorporar toda la información disponible en los orígenes en una única base de datos Oracle intermedia. Al finalizar T1, no se obtiene aún la base de datos final para ser utilizada por SGE, pero se obtiene una intermedia, soportada sobre la misma tecnología que la base final y con todas las conversiones sintácticas resueltas. Esta base intermedia constituye el segundo nivel. Es la salida de T1 y la entrada de T2.

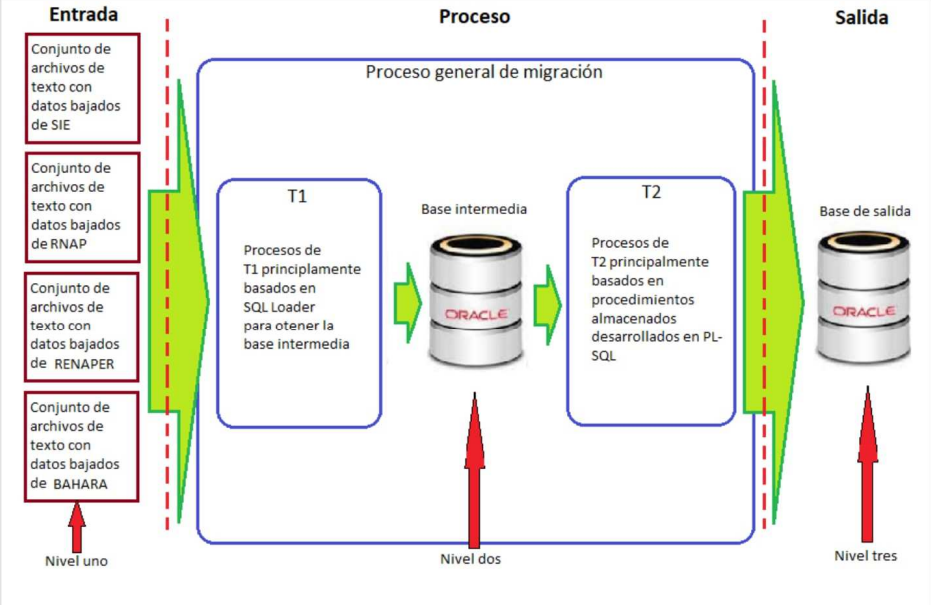
La herramienta utilizada para desarrollar T1 es Oracle SQL Loader que permite parametrizar conversiones con un buen nivel de automatización y control de errores.

En el segundo nivel, tomando la base intermedia, T2 realiza todas las conversiones semánticas. Para ello, un conjunto de procedimientos almacenados, mantienen conexión con la base intermedia, extraen los datos, aplican las reglas de transformación especialmente programadas para cada caso y escriben en la base de destino final del SGE que constituye el tercer nivel. Al finalizar T2, se obtienen una base de datos lista para ser utilizada por el nuevo sistema.

Cabe destacar que, por cuestiones de rendimiento, la base obtenida por T2 carece de índices, relaciones entre entidades y restricciones. Las mismas deben crearse y activarse antes de utilizar el sistema y luego de finalizar la ejecución de T2.

El tercer nivel, está compuesto por la base de salida, indexada, relacionada y con todos los elementos necesarios para que SGE pueda utilizarla.

El siguiente es un diagrama general de la arquitectura.



Descripción detallada de T1

Existen 4 Orígenes de datos distintos, SIE, BAHRA, RENAPER, RNAP, de estos 4 orígenes, los archivos de SIE están divididos por distrito (es decir existe un origen con todos sus archivos por distrito) 25 conjuntos de archivos independientes. Los otros 3 son planos (es decir existe un solo conjunto de archivos) independiente de la cantidad de distritos.

Cada origen es una carpeta independiente en el servidor que se ha dispuesto para alojar los archivos de texto plano.

Para importar los datos a la base de nivel uno se utiliza SqlLoader mediante un script de bash que automatiza el recorrido por los archivos de control.

Para cada origen existe una estructura de carpetas:

- bin: script de bash que ejecuta la carga,
- control: archivos de control de SqlLoader,
- config: archivos de configuración del script,
- data: archivos de datos,
- log: donde se escriben los logs de SqlLoader.

En el caso de SIE las carpetas: data y control están divididas en 24, una por cada distrito con sus archivos txt y ctl respectivamente.

SIE tiene aproximadamente 70 archivos distintos, pero estos se transforman en aproximadamente 100 por los casos de los archivos que están partidos en varias partes, como por ejemplo hpad del que existen hpad.txt, hpad1.txt, hpadN.txt. Esto hay que multiplicarlo por 25 distritos y sumarle los 6 archivos de Privados de Libertad.

BAHRA tiene 18 archivos, RENAPER tiene 31 archivos, y RNAP tiene 56 archivos.

Respecto de los procesos, el de SIE es el más complejo porque tiene que recorrer los distritos y dentro de cada distrito los archivos y además debe identificar si es un archivo único de datos o es el caso de un archivo con múltiples partes, para esto existen 2 scripts uno para los archivos monolíticos y otro para los multiparte y cada uno con su configuración más un 3er. script para privados de libertad. Además, hay un script base que ejecuta estos tres y redirecciona el log a un único archivo.

En el caso de los otros 3 orígenes, al ser solo un conjunto de archivos y ser todos monolíticos, solo existe un script por cada uno.

Descripción detallada de T2

El proceso T2 se encuentra compuesto de cuatro grupos de procedimientos que deben ejecutarse secuencialmente intercalando los controles que garanticen la estabilidad del proceso.

Estos cuatro grupos son:

1. Grupo de migración tablas periféricas.
2. Grupo de migración de electores.
3. Grupo de migración de historiales.
4. Grupo de migración de partidos políticos.

La división de T2 permite controlar la consistencia de los datos al finalizar una ejecución y continuar con la siguiente o deshacer los cambios (hacer un rollbak) de todo el grupo y reejecutar. Por ejemplo, se pueden migrar las tablas satélites, controlar su información y si no hay errores migrar los electores, controlar su información y si se detectan anomalías o el proceso se ve afectado por un problema técnico, deshacer electores sin afectar las tablas satélites que ya se habían controlado. Lo mismo es trasladable a todos los niveles.

De este modo, un error en parte del proceso no afecta a la totalidad de T2 sino solo al grupo en el que ocurre.

Procedimientos almacenados que componen cada grupo

Cada grupo mencionado en el apartado anterior está compuesto de un conjunto de procedimientos almacenados que realizan una tarea específica. A continuación, se los enumera dentro del grupo correspondiente. Una descripción detallada de cada uno puede encontrarse en el documento de definición de T2.

Procedimientos del grupo de migración de tablas periféricas:

- SP_COMPLETA_GEO_ELECTORAL
- SP_LOCALIDAD_CIRCUITO
- SP_MIGRA_CIRCUITOS_ELECTORAL
- SP_MIGRA_CIRCUITOS_GRAL
- SP_MIGRA_CUENTAS_USUARIO
- SP_MIGRA_DELEGACIONES
- SP_MIGRA_INSTITUCIONES
- SP_MIGRA_LOCALIDADES
- SP_MIGRA_LUGARES_RESIDENCIA
- SP_MIGRA_PAISES
- SP_MIGRA_PARTIDOS_DEPARTAMENTO
- SP_MIGRA_PROFESIONES
- SP_MIGRA_PROVINCIAS_DISTRITO
- SP_MIGRA_SECCIONES_ELECTORAL
- SP_MIGRA_SECCIONES_GRAL
- SP_MIGRA_TIPOS_EJEMPLAR

Migración de datos. Documento de Arquitectura

Proyecto:	**SGE – Sistema de Gestión Electoral**	Versión:	1.0

Procedimientos del grupo de migración de electores

- SP_INICIAR_MIGRACION_ELECTORES
- SP_MIGRA_ELECTORES
- SP_MIGRA_ELECTORES_FA
- SP_MIGRA_ELECTORES_INHAB
- SP_MIGRA_ELECTORES_PL

Procedimientos del grupo de migración de historiales

- SP_INICIAR_MIGRACION_HISTORIAL
- SP_MIGRA_HIST_ELECT

Procedimientos del grupo de migración de partidos políticos

- SP_ACTUAL_PP_PADRE_CONFNAC
- SP_ACTUAL_PP_PADRE_SIE
- SP_ACTUAL_PP_PADRE_TCARG_IND
- SP_EJECUCION_MIGRA_ADH_AFI
- SP_EJECUCION_MIGRA_PP
- SP_HISTORIAL_ADHESIONES
- SP_HISTORIAL_AFILIACIONES
- SP_HISTORIAL_CERTIFICANTES
- SP_MIGRA_ESTRUCT_APODER
- SP_MIGRA_ESTRUCT_AUTORID
- SP_MIGRA_MANDATARIOS
- SP_MIGRA_PP_ALIANZAS_NACIONAL
- SP_MIGRA_PP_SIE

Página 8

Anexo II: Especificación de lógica de proceso de migración de domicilios

PODER JUDICIAL DE LA NACION
PROYECTO S.G.E
CNE CÁMARA NACIONAL ELECTORAL

Universidad Tecnológica Nacional
Facultad Regional Córdoba

Proyecto SGE – Descripción de Lógica de Migración

1 INFORMACION DEL PROCESO

ID del Proceso: 19
Nombre: LM_DOMICILIO_ELECTOR
Objetivo: Migrar los domicilios de los electores, para los distintos estados definidos en SGE, provenientes de las tablas de padrón y bajas existentes en SIE.
Nivel de complejidad: ALTA
Descripción de la lógica: Se definen tres casos diferentes en la migración de electores que incluye sus domicilios. Estos casos son: - Padrón - Fallecidos - Privados de la libertad - Los SP utilizan como entrada las tablas de Padrón excepto para los fallecidos que utilizan las bajas existentes en SIE, y según el valor en la columna TTPA_CODIGO de la tabla de origen PADR. Para todos los estados y distritos Privado de Libertad y Residentes en el Exterior se crea un registro en la tabla Domicilios, para el distrito Residentes en el Exterior se crea también un registro en la tabla Exteriores.
Detalle de inconsistencias: Se crearon las tablas de origen: MIGRA_FA, MIGRA_PF y MIGRA_99 para poder identificar los PBAJ_NUMREG en las tablas de bajas del SIE.
Validado por Cámara: Sin fecha
Estadística de ejecución: Haga clic aquí para escribir texto.

Página 1 de 4

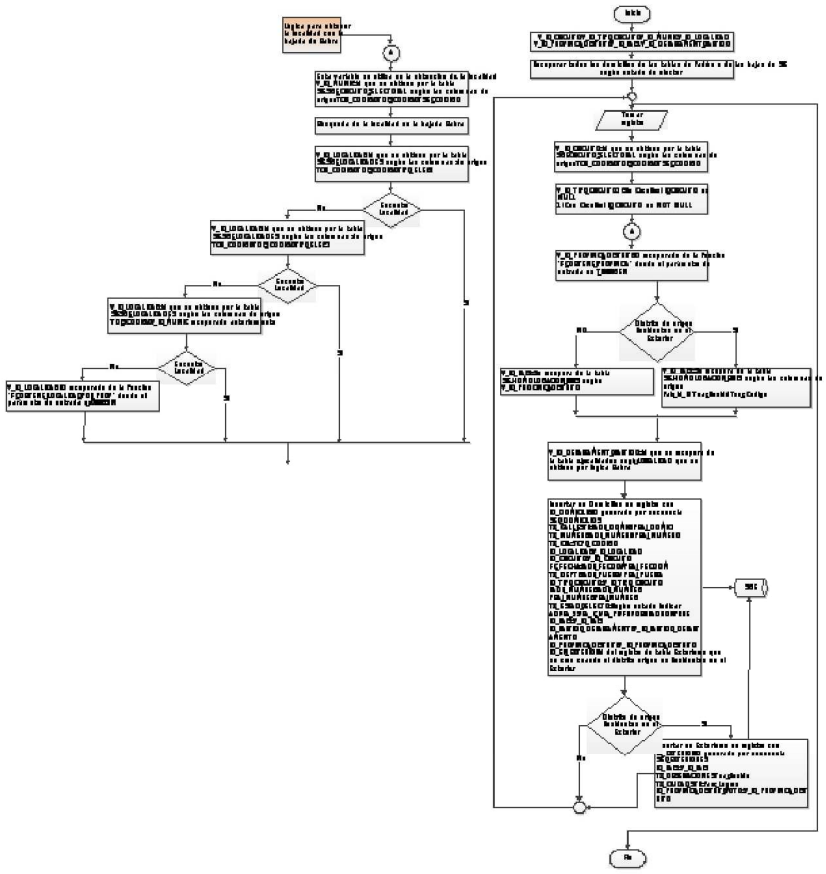
tmp

PODER JUDICIAL DE LA NACION
PROYECTO S.G.E
CNE
CÁMARA NACIONAL ELECTORAL

Universidad Tecnológica Nacional
Facultad Regional Córdoba

Proyecto SGE – Descripción de Lógica de Migración

2 DIAGRAMA



Página 2 de 4

tmp

PODER JUDICIAL DE LA NACION
PROYECTO S.G.E
CNE CÁMARA NACIONAL ELECTORAL

Universidad Tecnológica Nacional
Facultad Regional Córdoba

Proyecto SGE – Descripción de Lógica de Migración

3 DEFINICIÓN DE SP (STORE PROCEDURE)

La definición del código se encuentra en los siguientes SP del esquema SIE del entorno de producción:
SP_MIGRA_ELECTORES
SP_MIGRA_ELECTORES_FA
SP_MIGRA_ELECTORES_PL

4 MAPEO DE CAMPOS

Para el distrito Residentes en el exterior se crea un registro en la tabla Exteriores:

Tabla: SGE.EXTERIORES	**Lógica: Se obtiene de las tablas de origen Padrón y bajas según el estado.**
ID_EXTERIOR	ID generado por secuencia SEQ_EXTERIORES
ID_PAIS	Se recupera de la tabla SIE.HOMOLOGACIONES_PAIS según las columnas de origen: Pais_Id_D/ Tnac_Reside/ Tsec_Codigo.
TX_OBSERVACIONES	Columna de origen Tnac_Reside.
TX_CIUDAD_STR	Columna de origen Paex_Lugres
ID_PROVINCIA_DISTRITO_VOTO	ID recuperado de la función 'FC_OBTENER_PROVINCIA' donde el parámetro de entrada es TDIS_ORIGEN

Para todos los distritos se crea un registro en la tabla Domicilios:

Tabla: SGE. DOMICILIOS	**Lógica: Se obtiene de las tablas de origen Padrón y bajas según el estado.**
ID_DOMICILIO	ID generado por secuencia SEQ_DOMICILIOS
TX_CALLE_STR	PADR_DOMICI/ PBAJ_DOMICI
TX_NUMERO	PADR_NUMERO/ PBAJ_NUMERO
TX_CPA	TCPO_CODIGO
ID_LOCALIDAD	Id que se obtiene por bajada Bahra. **Ver lógica en diagrama de flujo**.
ID_CIRCUITO	Id que se obtiene por la tabla SGE.CIRCUITOS_ELECTORAL según las columnas de origen: TCIR_CODIGO/ TDIS_CODIGO/ TSEC_CODIGO.
FC_FECHA	PADR_FECDOM/PBAJ_FECDOM
TX_DEPTO	PADR_PUERTA/ PBAJ_PUERTA
ID_TIPO_CIRCUITO	3 (Sin Circuito) si ID_CIRCUITO es NULL.

Página 3 de 4

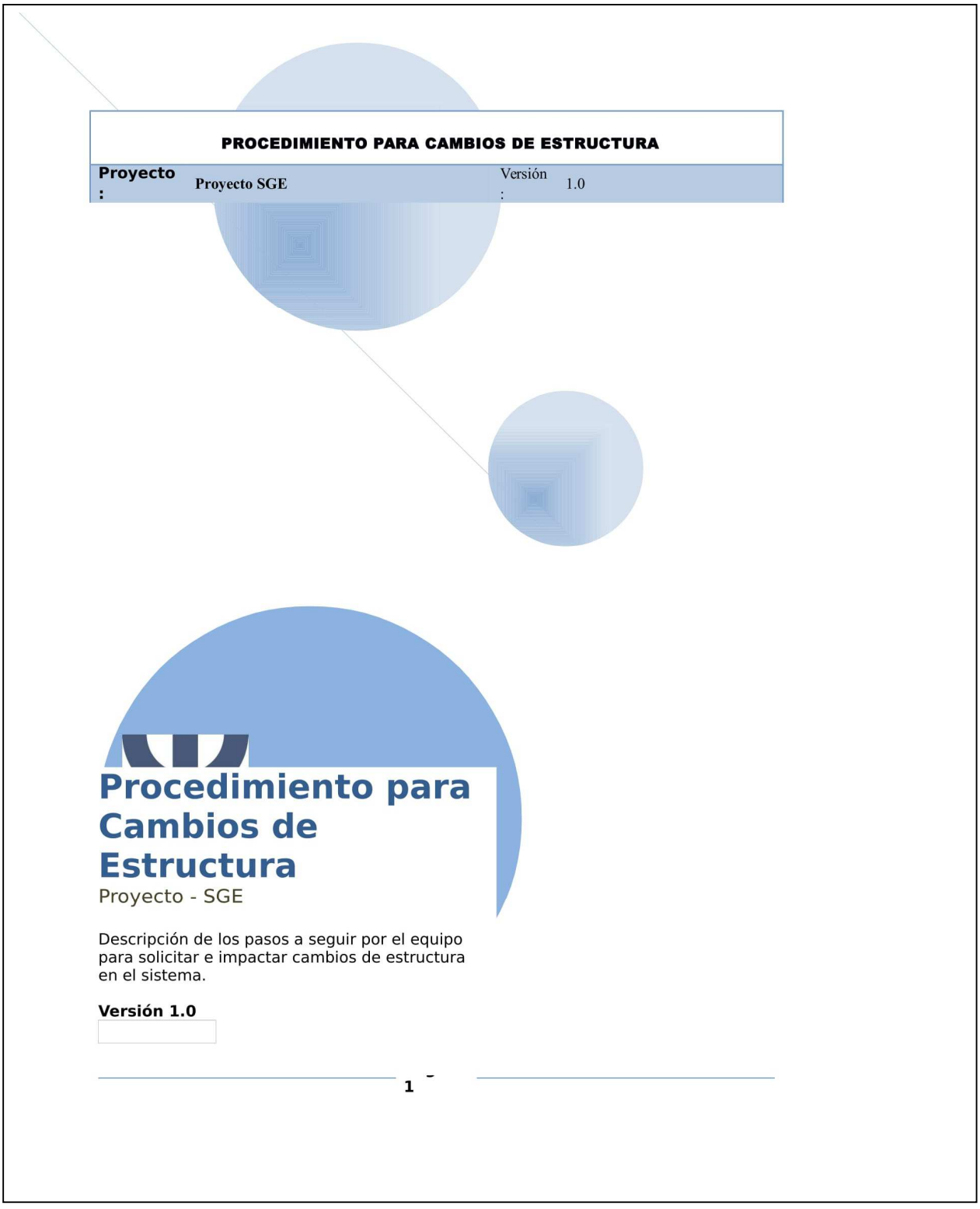
tmp

	1 (Con Circuito) si ID_CIRCUITO es NOT NULL.
PADR_NUMREG	PADR_NUMREG
PBAJ_NUMREG	PBAJ_NUMREG
TX_ESTADO_ELECTOR	Según estado indicar: ADF/FA_99/FA_C_I/FA_PR/FE/I/OBS/PADRON/PL/RE
ID_PAIS	Id que se recupera de la tabla SGE.PROVINCIAS_DISTRITO según id_provincia_distrito recuperado.
ID_PARTIDO_DEPARTAMENTO	Id que se recupera de la tabla sge.localidades según ID_LOCALIDAD que se obtiene por lógica Bahra.
ID_PROVINCIA_DISTRITO	ID recuperado de la función 'FC_OBTENER_PROVINCIA' donde el parámetro de entrada es TDIS_ORIGEN
ID_EN_EXTERIOR	Id correspondiente al registro de tabla Exterior que se crea cuando el distrito de origen es Residentes en el Exterior.

1 HISTORIAL DE CAMBIOS

Versión	Fecha	Breve	Nombre del Autor
1.0	30/09/2016	Definición inicial del documento	Gabriela Falcón

Anexo III: Documento de definición de control de cambios



PROCEDIMIENTO PARA CAMBIOS DE ESTRUCTURA

Proyecto :	Proyecto SGE	Versión :	1.0

Procedimiento para Cambios de Estructura

Proyecto - SGE

Descripción de los pasos a seguir por el equipo para solicitar e impactar cambios de estructura en el sistema.

Versión 1.0

1

PROCEDIMIENTO PARA CAMBIOS DE ESTRUCTURA			
Proyecto :	**Proyecto SGE**	Versión :	1.0

Contenido

Historial de cambios

Registro de REVISIONES			
Revisión	**Fecha**	**Revisado por**	**Observación**
1.0	02/08/2016	Gustavo Boiero	Creación de versión Inicial
1.1	09/08/2016	Gustavo Boiero	Se agrega nota de explicación al finalizar los pasos de cambios

Registro de APROBACIÓN				
Entidad	**Apellido y Nombre**	**Cargo**	**Firma**	**Fecha**

PROCEDIMIENTO PARA CAMBIOS DE ESTRUCTURA			
Proyecto :	**Proyecto SGE**	Versión :	1.0

Descripción del Proceso - Pasos a seguir

1. Ante un pedido de cambio del equipo de análisis que tenga impacto a nivel de estructura y/o datos de base de datos, un integrante del equipo de base de datos debe tomar la responsabilidad de implementar la modificación pertinente en un plazo no superior a las 24 horas de la notificación en cuestión. Un pedido de cambio se recibirá en la bandeja de entrada de correo de cada integrante del equipo de base de datos bajo el asunto de ***"LIBERO PLANILLA"*** junto con un nombre que indique la fuente de cambio, por ejemplo SGE 1.

2. Los integrantes del equipo de base de datos proceden a consultar la planilla de pedidos de cambios ubicada en repositorio en la ruta *'6.1 Análisis/Diagrama Clases Completo/ Diagrama de clases -Cambios solicitados'* dentro de la carpeta correspondiente a la fuente de cambio especificada. En dicho documento se especifica el detalle del cambio solicitado por el equipo de análisis.

3. Un integrante del equipo de base de datos tomará la responsabilidad de implementar el cambio solicitado en el plazo pertinente. Para ello se debe responder con COPIA A TODOS el mail recibido por el equipo de análisis correspondiente al cambio con el mensaje ***"Proceso el cambio"***.

4. Se crea una tarea para el cambio tomado en la herramienta TAIGA (https://tree.taiga.io/) asignándosela a sí mismo y agregando como *"Watchers"* al analista que solicitó el cambio y al responsable de Migración en CNE (Gabriela Falcón). En la descripción de la misma se especifica lo detallado en la/s fila/s correspondiente/s a la modificación solicitada en la planilla de cambios.

5. Una vez implementado el cambio sobre la base de datos correspondiente se crea un archivo de script de cambio en la carpeta de actualización de la versión actual del esquema donde se trabaja en el repositorio, por ejemplo *"Repositorio\6 Ingenieria de Producto\6.3 Desarrollo\BD\EN PROCESO BRANCH V2\Script_ACTUALIZA"*. En dicho script se debe especificar el código SQL necesario para llevar a cabo el cambio solicitado junto con una descripción de los pasos a llevar a cabo con una estimación de tiempo a insumir en las base reales.

PROCEDIMIENTO PARA CAMBIOS DE ESTRUCTURA			
Proyecto:	**Proyecto SGE**	Versión:	1.0

6. Commitear el archivo script creado en el paso anterior en el repositorio.
7. En la tarea de TAIGA respectiva al cambio solicitado se debe adjuntar el archivo script commiteado en el paso anterior.
8. Modificar el estado de la tarea de TAIGA a *"Closed".*
9. Finalmente, luego de impactados los cambios en nuestra BD local, la responsable de migración podrá impactar los cambios necesarios para el proceso de migración, si corresponde.

Nota: Si el equipo de base de datos detecta un cambio que afecte la estructura de la base de datos debe informar a través de un mail al grupo de base de datos sobre la implementación de las modificaciones pertinentes adjuntando el mismo archivo script que se usó en TAIGA. Se deben respetar los pasos desde el punto 4 al 9.