UNIVERSIDAD

REY JUAN CARLOS

ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA INFORMÁTICA

INGENIERÍA INFORMÁTICA

Curso Académico 2010/2011

Proyecto Fin de Carrera

SISTEMA CENTRAL DE DATOS, CONTROL DE TRAZABILIDAD Y GENERADOR DE INFORMES

Autor: Eduardo Sánchez-Cabezudo Pérez

Tutor: Jaime Urquiza Fuentes

AGRADECIMIENTOS

Quiero dar las gracias especialmente a mis padres, sin su ayuda y sacrificio nunca podría haber conseguido este gran reto.

Gracias a mi madre por la insistencia que mostró en que me esforzase para terminar el Proyecto Fin de Carrera. También gracias a mi padre que intentaba ser más sutil a la hora de preguntar por el proyecto.

No me puedo olvidar de Patricia, que lleva mucho tiempo dándome ánimos y es la persona que más ha sufrido este último periodo.

Muchas gracias al equipo de Proin, S.L., que sin duda sin ellos este proyecto no sería posible, y con ello las ganas de hacer el trabajo lo mejor posible.

Gracias a Manuel por sus consejos y aclaraciones en todas mis dudas.

Me gustaría dar la gracias a Jaime, que sin él la entrega de este Proyecto Fin de Carrera no sería posible. Muchas gracias también por los ánimos y el entusiasmo que has mostrado acerca del proyecto desarrollado, así como por aclararme y asesorarme en todo lo que he necesitado.

Por último, gracias a Duna y Coco por haberme acompañado todos los días en el desarrollo de este documento, y por permitirme, a cierta hora, darme un respiro y cambiar de aires.

RESUMEN

El objeto de esta memoria es dar cuenta de los trabajos de desarrollo realizados por el autor en relación con el proyecto DataServer. Este proyecto ha sido desarrollado durante año y medio en la empresa Procedimientos Integrados, S.L (Proin, S.L.), con esfuerzo total aproximado de cuatro hombres/año. Hace tres años, y en el contexto de los sistemas automatizados de la producción, la empresa Proin, S.L. detectó la necesidad, y por tanto la oportunidad de negocio, de implantar sistemas de trazabilidad, dándose aquí forma de proyecto.

La empresa Proin, S.L. ha proporcionado los recursos humanos y materiales necesarios para que el autor haya podido realizar satisfactoriamente su labor.

Objetivos principales del proyecto:

- Cubrir una necesidad en el sector (necesidad legal)
- Mejorar la calidad de los procesos de fabricación, dotando de trazabilidad a los procesos de producción.
- Optimización de costes.
- Algún requerimiento técnico importante puede elevarse al rango de objetivo, como por ejemplo, escalabilidad y adaptabilidad del DataServer a multitud de organizaciones y arquitecturas de cliente.

ÍNDICE

AGRADECIMIENTOS i

RESUMEN ii

1. VISIÓN GENERAL DEL NEGOCIO 1

1.1. Sistema central de datos 2

1.2. Gestión de trazabilidad 13

1.3. Generador de informes 16

1.4. Adaptación de las aplicaciones existentes 17

2. OBJETIVOS 22

2.1. Metodología 23

2.2. Tecnologías 24

2.3. Plan de trabajo 31

3. DESCRIPCIÓN INFÓRMATICA 35

3.1. Especificación de requisitos 35

3.1.1. Requisitos funcionales 35

3.1.2. Requisitos no funcionales 58

3.2. Diseño y arquitectura 59

3.2.1. Decisiones de diseño 60

3.2.2. Condiciones de contorno 61

3.2.3. Arquitectura 62

3.3. Pruebas 78

4. CONCLUSIONES Y TRABAJOS FUTUROS 80

5. REFERENCIAS 82

6. ANEXOS 84

6.1. Contenido del CD 84

6.2. Manual de usuario 85

6.3. Enumerado del intérprete de comandos 104

6.4. Resultado de las pruebas 128

6.4.1. Informe de ejecución de pruebas del protocolo de comunicación entre el cliente y el sistema 128
6.4.2. Informe de ejecución de pruebas de funcionalidad del sistema 131

Índice de figuras

Ilustración 1. Fábrica de hormigón 4
Ilustración 2. Fábrica de mortero 5
Ilustración 3. Fábrica de prefabricados 6
Ilustración 4. Acopio 10
Ilustración 5. Esquema de trazabilidad 14
Ilustración 6. Conex M2 18
Ilustración 7. Sistema de gestión (IU) 19
Ilustración 8. Sinóptico de planta (IG) 20
Ilustración 9. Diagrama de Gantt 34
Ilustración 10. Arquitectura cliente/servidor 63
Ilustración 11. Arquitectura del sistema 64
Ilustración 12. Diagrama de paquetes del modelo de dominio 65
Ilustración 13. Diagrama de clases del paquete almacenes 67
Ilustración 14. Diagrama de clases del paquete artículos 68
Ilustración 15. Diagrama de clases del paquete centros 69
Ilustración 16. Diagrama de clases del paquete fabricación 69
Ilustración 17. Diagrama de clases del paquete medidores 70
Ilustración 18. Diagrama de clases del paquete medidas 70
Ilustración 19. Diagrama de clases del paquete meteorología 71
Ilustración 20. Diagrama de paquetes del modelo de control 72
Ilustración 21. Diagrama de clases del paquete servidor 74
Ilustración 22. Diagrama de colaboración del módulo Servidor 76
Ilustración 23. Diagrama de paquetes del Generador de informes 77

1. VISIÓN GENERAL DEL NEGOCIO

Hoy en día en el sector industrial ha adquirido una gran importancia el control de la historia de las materias primas en el proceso de producción, siendo necesario conocer en todo momento el origen y destino final de las mismas. Este proceso de control es lo que se conoce como control de trazabilidad.

El proceso de control de trazabilidad se ha vuelto en determinados casos un proceso obligatorio para la producción y venta, el más conocido es el sector alimenticio aunque también existen otros como el de producción de hormigón. En la actualidad es requisito imprescindible para poder exportar los productos a otros países y poder así incrementar el área de negocio de una empresa.

Este proyecto nace precisamente por la necesidad surgida con la nueva instrucción de hormigón estructural (EHE-08), aprobada el 18 de julio 2008 con el 14167 Real Decreto 1247/2008, de llevar un control de la trazabilidad de las materias primas en el proceso de fabricación del hormigón estructural. La conformidad de una estructura con esta Instrucción requiere de una trazabilidad adecuada entre los productos que se colocan en la obra con carácter permanente (hormigón, armaduras o elementos prefabricados) y cualquier otro producto que se haya empleado para su elaboración. Por tanto el constructor debe llevar un registro de todos los proveedores y de las partidas o remesas de los productos para cada elemento estructural de la obra.

El control de trazabilidad resulta de gran utilidad por ejemplo:

- Para detectar el origen de problemas de calidad. Seremos capaces de saber que material o materiales son los causantes de un problema.
- Nos permite adelantarnos a futuros problemas de calidad. Si un proveedor nos avisa de un problema con algún lote de sus productos, podremos ser capaces de saber los hormigones fabricados con dichos productos y nosotros a su vez avisar a las obras que pueden verse afectadas.
- Nos permite reducir los daños económicos en caso de detectarse un problema de calidad, ya que no será necesario retirar todo el producto sino únicamente los lotes afectados.

Para poder iniciar un proyecto de estas características es necesario modelar el entorno en el que se va a trabajar, en este caso debemos modelar un Centro de Producción en el que poder disponer de almacenes, materias primas, productos intermedios y finales almacenados, así como distintos componentes específicos que dependerán del tipo de centro de producción. Este modelo representará nuestro Sistema central de datos, con el cual podremos interactuar para crear, editar, borrar y pedirle información acerca de cualquiera de sus componentes. En el punto 1.1 veremos con más detalle el Sistema central de datos a implementar.

Como todo sistema de datos es imprescindible disponer de algún servicio de informes que nos permita obtener de manera visual, en papel o en formato electrónico información acerca de los mismos. Como parte del proyecto se ha desarrollado un servidor de informes para dar servicio a los distintos clientes. En el apartado 1.3 veremos con más detalle este módulo.

Este proyecto nace como consecuencia de nuevas necesidades en el sistema anterior de la empresa Procedimientos Integrados, S.L (Proin, S.L.). El antiguo sistema, llamado Conex, necesita de esta nueva funcionalidad, por tanto es necesario su adaptación para poder trabajar como cliente del Sistema central de datos. En el apartado 1.4 veremos con más detalle los distintos módulos del sistema Conex que han tenido que adaptarse para poder trabajar con el nuevo sistema.

1.1. Sistema central de datos

Ante cualquier sistema nuevo siempre debemos analizar y tener claro el ámbito en el que se va a mover, de tal forma que podamos determinar sus fronteras. Pongamos el típico ejemplo de un videoclub, el ámbito en el que se moverá será el de un videoclub, con las películas, música, proveedores, clientes, empleados,… pero no entrarán dentro de su ámbito los coches, los animales, etc. En este sentido el sistema a desarrollar centra su ámbito en torno a un centro de producción, por tanto debemos ser capaces de modelar un centro de producción con todos sus componentes. El sistema central de datos, que llamaremos DataServer, será capaz de controlar el ciclo de vida de cualquier componente dado de alta en él. A continuación explicaremos con más detalles cada uno de los componentes que puede tener un centro de producción, así como las peculiaridades que pueden surgir dependiendo de su tipo.

Un centro de producción lo podemos entender como el lugar físico donde se producen determinados materiales o productos, es decir, el terreno en el que se encuentra nuestro sistema de producción. Los centros pueden contener almacenes y planta de producción, y estos a su vez pueden contener más almacenes. En los almacenes tendremos guardadas materas primas, productos intermedios y productos finales. Y cada uno de estos productos o materias primas tienen unas características técnicas y comerciales específicas.

Centros de producción

Vamos a ver con más detalle distintos tipos de centro de producción para tener claro cómo debemos trabajar en cada uno de ellos. Como hemos explicado un centro de producción se compone de almacenes, ya sean almacenes exclusivos o almacenes agregados que pueden contener a su vez más almacenes, siendo el centro de producción a su vez es un almacén agregado.

Vamos a distinguir los siguientes tipos de almacén:

- *Almacén agregado* → son aquellos almacenes que pueden contener a otros y que pueden tener distintos stocks de material activos al mismo tiempo. Como ejemplos:
    - Centro de producción: contienen plantas de producción y almacenes.
    - Planta: contiene almacenes.
    - Almacén de materias primas y productos: estos almacenes los podemos entender como la nave en la que se almacenan los productos y materias primas para su uso en el proceso de fabricación o para su venta.
- *Almacén exclusivo* → son aquellos que sólo pueden tener un stock activo al mismo tiempo y no pueden contener a otros almacenes. Viéndolo como un árbol de componentes, este tipo de almacenes serian hojas. Como ejemplos:
    - Tolvas: almacenes para áridos y grabas.
    - Silos: almacenes para cementos.
    - Depósitos de aditivo: almacenes para los aditivos.

A continuación vamos a ver distintos tipos de centro de producción, estos ejemplos están basados en el mundo del hormigón, pero debemos ser conscientes que el modelo

de datos implementado podrá ser aplicado a cualquier tipo de campo, como puedan ser las fábricas de pienso compuesto, fábricas de pintura, fábricas de asfalto,…

- Fábrica de hormigón.

En la Ilustración 1 podemos ver el esquema típico de una planta de hormigón sencilla. Estaría compuesta por un centro de producción, que a su vez contiene una planta, y esta contiene cinco tolvas de árido, tres silos de cemento, tres depósitos de aditivo y un depósito de agua.

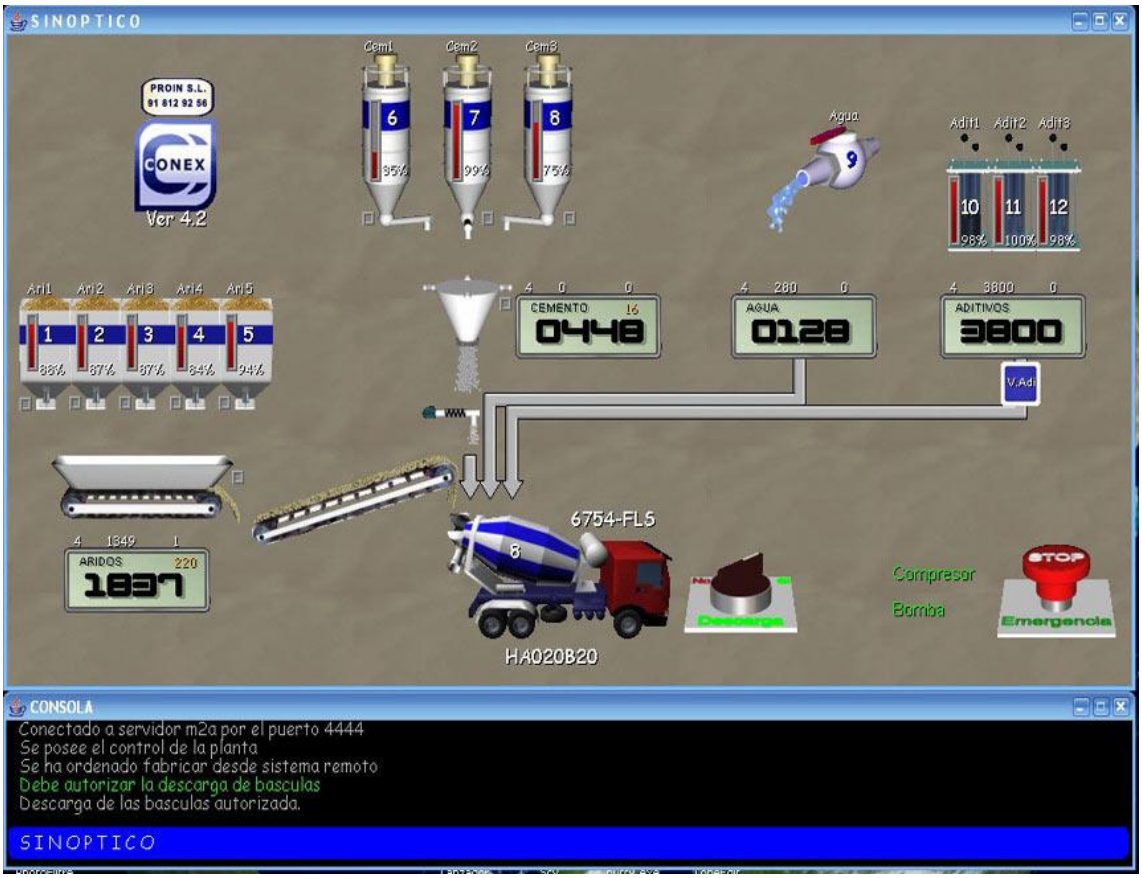


Ilustración 1. Fábrica de hormigón

El funcionamiento de este tipo de centros consiste en entradas de material a cada uno de los almacenes exclusivos (tolvas, silos y depósitos de aditivo), que nos proporcionan el stock de los mismos. Cuando se fabrica, el producto final no se almacena en planta, sino que se descarga directamente al camión hormigonera que lo suministra a la obra inmediatamente, ya que el hormigón tiene un tiempo límite de uso.

En la actualidad las plantas de hormigón tienden a ser más complejas, pudiendo incluso tener 2 o 3 plantas de producción dentro del mismo centro.

- Fábrica de mortero

Como podemos observar en la Ilustración 2 este tipo de fábricas resultan mucho más complejas que las fábricas de hormigón. En este caso está compuesta de un centro de producción que contiene una planta y esta a su vez contiene seis tolvas de árido, seis silos de cemento, seis depósitos de aditivos mayoritarios, seis silos de aditivos minoritarios y silos de producto terminado.

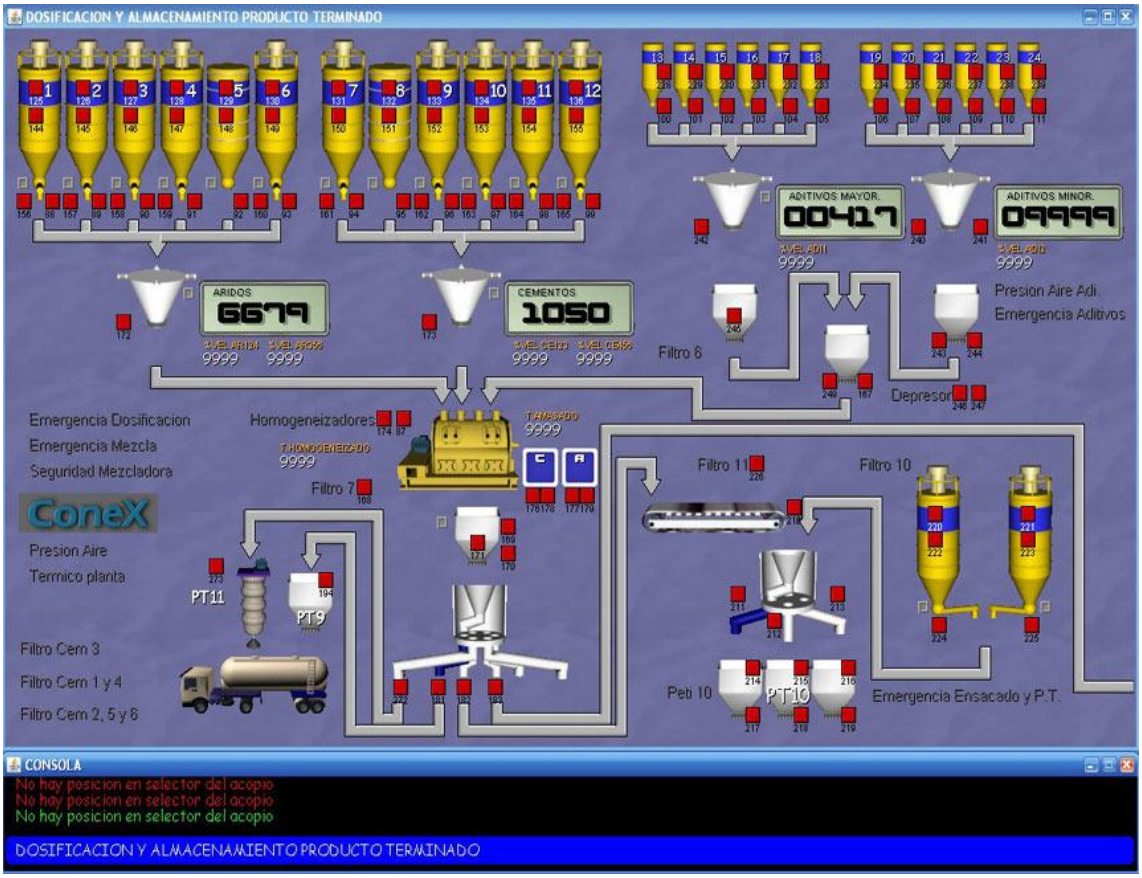


Ilustración 2. Fábrica de mortero

En este tipo de fábricas, el producto terminado puede descargarse directamente en el camión y ser entregado al cliente o pasar a los silos de producto terminado de los cuales se irá descargando producto a camiones o silos de obra según pida el cliente.

- Fábrica de prefabricados

En la Ilustración 3 podemos ver un esquema típico de fábrica de prefabricados la cual se compone de un centro de producción con una planta, y esta está compuesta de dos tolvas de árido, un silo de cemento, cuatro depósitos de aditivo y dos depósitos de agua.

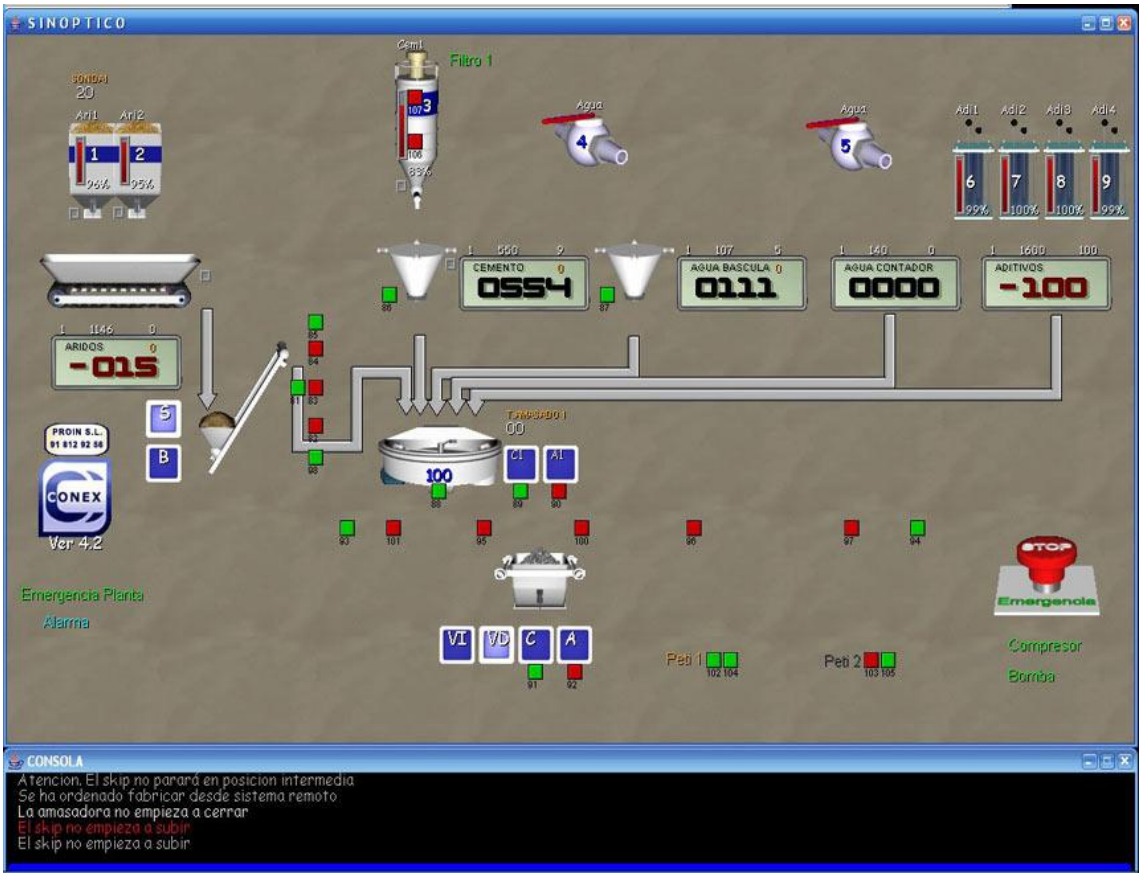


Ilustración 3. Fábrica de prefabricados

El funcionamiento de este tipo de fábricas consiste en fabricar hormigón para realizar piezas prefabricadas que una vez listas serán entregadas al cliente. El proceso de elaboración es exactamente igual al de una fábrica de hormigón pero en lugar de realizar la descarga en un camión, suele ir a un contenedor o vagoneta que irá depositando su contenido en los distintos moldes. Una vez preparadas las piezas estas se guardan en un almacén a la espera de ser entregadas al cliente.

Hasta ahora hemos hablado de los centros de producción y los almacenes que lo componen, sin embargo hay muchas otras cosas que debe tener en cuenta nuestro sistema para modelar y tener contempladas todas las peculiaridades que pueden tener los centros de producción.

Máquinas, bocas y medidores

Las fábricas en las que nos movemos, por lo general están automatizadas, eso quiere decir que hay un sistema que controla todas las señales para poder controlar en todo momento el estado de cada una de las partes implicadas en la automatización. Con la automatización debemos ser capaces de activar y desactivar señales, así como leer su estado y valores en el caso de las básculas. Para tener controladas las señales y poder indicar al sistema de automatización el trabajo a realizar, debemos compartir con él ciertos datos de cara a indicarle las cantidades objetivo de cada uno de los almacenes de los que debe dosificar y que nos pueda devolver el valor de las cantidades reales dosificadas. Es por esto que debemos crear máquinas, bocas y medidores para compartir información con el sistema de automatización.

- Máquinas: son componentes de la planta del cual debemos recibir información, sin embargo no son almacenes. Como ejemplo:
    - Amasadora. Se trata de una máquina que amasa el producto para hacerlo más homogéneo. En este caso debemos indicar al sistema de automatización el tiempo que debe amasar el producto. El sistema una vez amasado nos enviará el tiempo exacto que duro el proceso de amasado.
    - Sondas de humedad. Son unos dispositivos que permiten saber la humedad del árido. El sistema de automatización nos enviará el valor de la humedad del árido concreto.

- Bocas: Los almacenes exclusivos disponen de bocas de descarga por las que sale el material a la hora de ser dosificadas. Es por esto que a la hora de fabricar debemos indicarle al sistema de dosificación la boca por la que debe descargar, así como la cantidad teórica deseada. Un vez dosificado el material por la boca y leído el valor de la báscula, el sistema de automatización nos enviará el valor teórico y real dosificado a través de esa

boca. De esta manera seremos capaces de descontar del stock del almacén la cantidad real dosificada.

- Medidores: Nos permiten saber qué es lo que queremos medir y su unidad. En el caso de la amasadora queremos leer el tiempo de amasado en segundos, en el caso de las humedades nos interesa el % de humedad del árido.

Artículos y materiales

Los artículos son la cara comercial de las materias primas. Los artículos se pueden comprar (entradas de material) y se pueden vender (salidas de material). A un sistema de facturación lo que le interesará serán los artículos ya que son estos los que se venden.

Hemos diferenciado distintas facetas que puede tener un artículo de cara a tener posibilidad de realizar o no tareas con ellos:

- Comprable: teniendo esta faceta se pueden realizar entradas de dicho artículo en los almacenes del centro.
- Almacenable: indica que un artículo puede almacenarse en los almacenes.
- Vendible: especifica que un artículo puede venderse.
- Material técnico: especifica que un artículo dispone de características técnicas que deben tenerse en cuenta a la hora de fabricar.

Una materia prima usada en el proceso de fabricación será un artículo con la faceta de comprable y almacenable, ya que debemos tener stock para poder fabricar con él, y con la faceta de material técnico, ya que debemos conocer sus características técnicas a la hora de saber la cantidad a dosificar en la dosificación.

En el caso del producto final fabricado será un material técnico que además tendrá la faceta de vendible.

Stock, capacidad y niveles

Para saber el material o materiales disponibles en un almacén es necesario crear el stock del material para el almacén. Como ya se explicó anteriormente, dependiendo del tipo de almacén podemos tener cero, uno o más stocks asociados a un almacén.

A la hora de dar de alta un almacén exclusivo es importante indicar la capacidad de dicho almacén, ya que en el momento de dar una entrada superior a la capacidad del almacén, podremos avisar al usuario. Es de utilidad también para calcular el porcentaje de material disponible en el almacén y poderlo representar con una barra de nivel en el sinóptico de planta (ilustraciones 1, 2 y 3).

Es importante llevar correctamente el stock de un almacén ya que nos ayuda en la decisión de pedir más material y resulta imprescindible para llevar una correcta trazabilidad del material.

En los almacenes podemos disponer de señales de control, llamadas niveles, que nos avisarán en el caso de que el stock llegue a ese nivel, pudiendo realizar automáticamente un ajuste del stock.

Entradas

Para poder llevar un control correcto de la trazabilidad así como tener actualizado el stock es necesario registrar las distintas entradas de material de los distintos proveedores. Al registrar una entrada se incrementará el stock y se creará el lote de entrada correspondiente.

Consumos, apuntes y salidas

La aplicación debe registrar en todo momento la información de cantidad enviada por el sistema de automatización. En este punto distinguimos:

- Consumos: representan los datos teóricos del gasto, es decir, el objetivo de la dosificación
- Apuntes: representan la cantidad leída por los medidores (tiempo de amasado, humedades,…), su valor no implica un cambio en los stocks.
- Salidas: representan las cantidades reales del gasto. Esta cantidad es la que se restará del stock.

Acopios

El acopio es un sistema de distribución del árido a las distintas tolvas de la planta. En la ilustración 4 podemos ver un ejemplo típico, en el cual el árido viaja, a través de cintas transportadoras, desde una tolva inicial en la que se deposita el árido, mediante descarga de un camión o por una pala excavadora, hasta la tolva destino seleccionada mediante un distribuidor.

El sistema de acopio puede ser manual o automático. En el caso manual es activado por un operador que indicará la tolva destino. En el caso de un sistema de acopio automático, mediante niveles en las tolvas que indican el stock actual, puede iniciarse el proceso eligiendo como destino la tolva en la que se ha activado el nivel.

Estos niveles pueden ser físicos o software. En el caso de un nivel físico se trata de dispositivo que envía una señal cuando se llegue e nivel. En el caso de un nivel software se crea un nivel en base a la capacidad y stock de un almacén, de tal manera que el sistema notificará un aviso de nivel cuando el stock llegue al % de la capacidad indicado en el nivel.

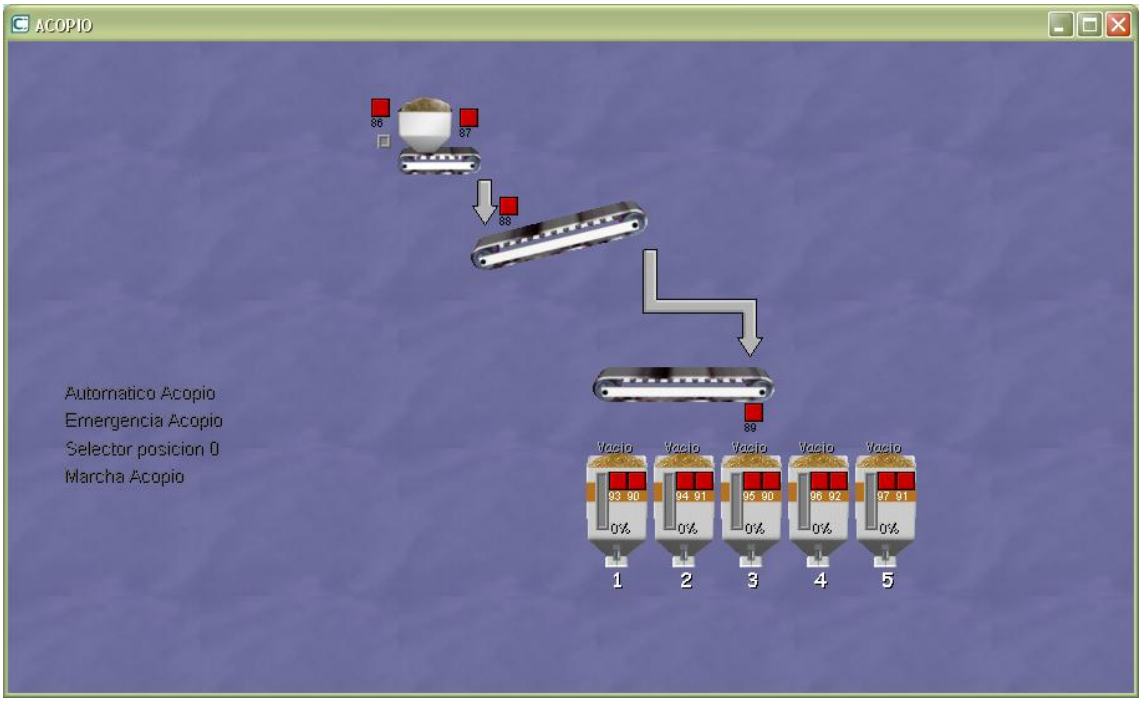


Ilustración 4. Acopio

Juegos de fabricación

En ocasiones para una misma planta tenemos distintos caminos de producción, siendo necesario poder crear estas rutas. Esto es lo que hemos definido como juegos de fabricación. Según el juego de fabricación se activarán las máquinas y dispositivos específicos de ese juego.

Destinos

A la hora de fabricar debemos indicar cuál es el destino del producto a fabricar. Este destino, por lo general, es un camión o un silo de producto terminado, pero existen casos, como en los prefabricados, que el destino son moldes.

Estación meteorológica

La fabricación de hormigón en condiciones climáticas extremas, ya sea con temperaturas altas o bajas, influye de una manera directa en las características del hormigón en cualquiera de sus etapas: amasado, transporte, puesta en obra y curado.

Es por este motivo que sea imprescindible, bajo condiciones climáticas extremas, la aplicación estricta de la normativa de toma de probetas de control, ya que la resistencia obtenida se verá reducida, y por tanto, se ocasionarán reclamaciones de calidad.

Según la instrucción EHE-08:

> 71.5.3.2. Hormigonado en tiempo caluroso
>
> Cuando el hormigonado se efectúe en tiempo caluroso, se adoptarán las medidas oportunas para evitar la evaporación del agua de amasado, en particular durante el transporte del hormigón y para reducir la temperatura de la masa. Estas medidas deberán acentuarse para hormigones de resistencias altas.
>
> Para ello los materiales constituyentes del hormigón y los encofrados o moldes destinados a recibirlo deberán estar protegidos del soleamiento. Una vez efectuada la colocación del hormigón se protegerá éste del sol y especialmente del viento, para evitar que se deseque.
>
> Si la temperatura ambiente es superior a 40ºC o hay un viento excesivo, se suspenderá el hormigonado, salvo que, previa autorización expresa de la

Dirección de Obra, se adopten medidas especiales.

71.5.3.1. Hormigonado en tiempo frío

La temperatura de la masa de hormigón, en el momento de verterla en el molde o encofrado, no será inferior a 5ºC.

Se prohíbe verter el hormigón sobre elementos (armaduras, moldes, etc.) cuya temperatura sea inferior a cero grados centígrados.

En general, se suspenderá el hormigonado siempre que se prevea que, dentro de las cuarenta y ocho horas siguientes, pueda descender la temperatura ambiente por debajo de los cero grados centígrados.

En los casos en que, por absoluta necesidad, se hormigone en tiempo de heladas, se adoptarán las medidas necesarias para garantizar que, durante el fraguado y primer endurecimiento de hormigón, no se producirán deterioros locales en los elementos correspondientes, ni mermas permanentes apreciables de las características resistentes del material.

El empleo de aditivos aceleradores de fraguado o aceleradores de endurecimiento o, en general, de cualquier producto anticongelante específico para el hormigón, requerirá una autorización expresa, en cada caso, de la Dirección de Obra. Nunca podrán utilizarse productos susceptibles de atacar a las armaduras, en especial los que contienen ión cloro.

Debido a esta peculiar circunstancia necesitamos un sistema que nos proporcione información de la temperatura y viento actual, y que sea capaz de evaluar un periodo de tiempo para determinar correctamente las condiciones climáticas en el momento de producción del hormigón.

1.2. Gestión de trazabilidad

La Real Academia Española define la trazabilidad [1]:

1. Posibilidad de identificar el origen y las diferentes etapas de un proceso de producción y distribución de bienes de consumo.
2. Reflejo documental de estas etapas.

Se entiende la trazabilidad como un conjunto de acciones, medidas y procedimientos técnicos que permite identificar y registrar cada producto desde su nacimiento hasta el final de la cadena de comercialización. La trazabilidad permite rastrear la cadena de producción y otorga a los productores la posibilidad de colocar sus productos en mercados específicos más rentables, que exigen la certeza del origen y de las distintas etapas del proceso productivo.

La trazabilidad se aplica en distintas áreas e industrias, de hecho en los países de mayor desarrollo se han publicado normativas específicas. Su aplicación se da principalmente por razones de mejora en el negocio o por imposición.

La trazabilidad nos ayuda a mejorar nuestro negocio ya que nos permite tener una mayor eficiencia en el proceso de producción. Otro beneficio es el ahorro de costes ante fallos, ya que al tener trazado el ciclo de vida del producto, si se detecta un problema podremos saber el lote o lotes afectados, retirando sólo los productos afectados.

En determinados sectores como pueden ser el sector alimenticio y actualmente tras la nueva instrucción EHE-08 en el sector de la construcción, se debe llevar obligatoriamente un correcto control de trazabilidad.

El uso de sistemas de trazabilidad es factible de obtener certificados, como puede ser el caso de sistemas de gestión de calidad, sistemas de control o sistemas de gestión medioambiental.

En el área de la construcción, la instrucción EHE-08 especifica los criterios de trazabilidad a tener en cuenta durante el ciclo de vida del producto fabricado. Es por este motivo, por el que Proin, S.L. se ve en la necesidad de desarrollar un sistema que permita llevar un correcto control de la trazabilidad de todos los productos con los que trabaja el sistema, desde su entrada en el centro de producción hasta su entrega final al cliente.

En la ilustración 5 podemos ver un esquema del proceso de trazabilidad.

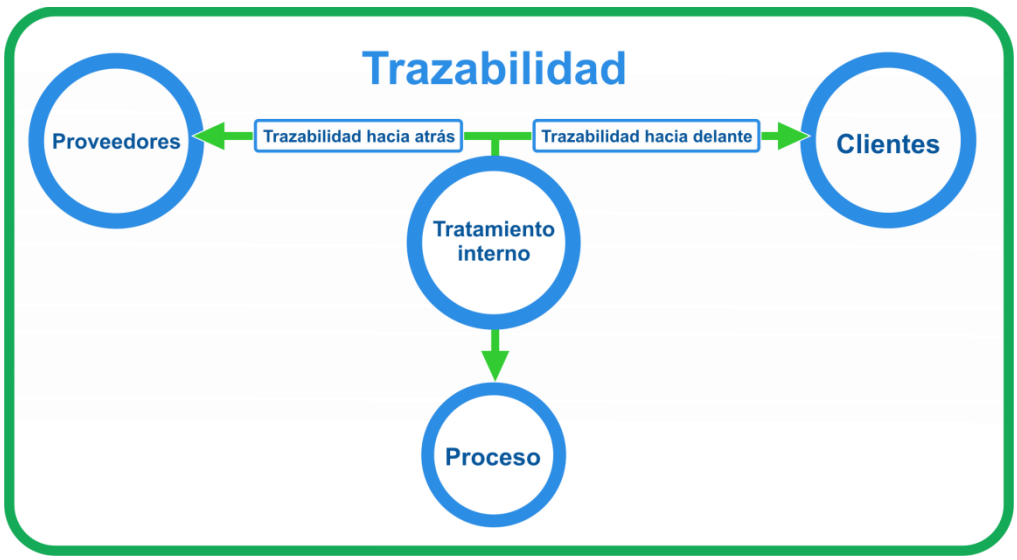


Ilustración 5. Esquema de trazabilidad

Trazabilidad hacia delante

La trazabilidad hacia delante consiste en conocer todos los lotes de producto final fabricados, y el destino de los mismos, dado un lote de entrada.

Trazabilidad hacia atrás

La trazabilidad hacia atrás consiste en conocer los lotes de entrada o lotes de origen dado un lote de producto final.

Trazabilidad interna

Aunque el objetivo principal de la trazabilidad es conocer los orígenes o destinos de un producto dado por su lote, no debemos olvidarnos que para fabricar un producto se ha podido necesitar fabricar productos intermedios. El sistema de trazabilidad debe ser capaz de navegar por todos los ciclos de la fabricación para obtener la información deseada. La trazabilidad interna consiste en trazar también los productos intermedios.

En la actualidad no todas las fábricas están automatizadas, por tanto no se puede saber con certeza el material utilizado de cada lote. Es por este motivo por el que tenemos que diferenciar entre trazabilidad basada en datos reales y trazabilidad basada en datos teóricos.

Trazabilidad basada en datos reales

Cuando el sistema está automatizado, nos llegan los datos leídos de las básculas automáticamente, pudiendo apuntarlos en nuestro sistema. Para que estos datos sean lo más reales posibles las básculas deben calibrarse cada cierto tiempo. Es en estos casos en los que disponemos de una trazabilidad más aproximada a la realidad. Debemos tener presente que es fundamental disponer de básculas correctamente calibradas, ya que si no es así la gestión de stocks y lotes pendientes de gasto no será fiable. El error en básculas es algo inevitable, pero por lo menos se puede controlar llevando una política de tarados y calibraciones adecuado. Con respecto a los errores en la lectura de las básculas, el problema común con el que nos encontramos son los descuadres en los stocks, y por tanto descuadres en los lotes pendientes de gasto. Una solución que reduce en parte el problema es el uso de niveles físicos que notifiquen al sistema cuando el stock llegue al nivel, de tal forma que podemos ajustar automáticamente el stock a la cantidad marcada por el nivel.

Trazabilidad basada en datos teóricos

En el caso de no disponer de automatización y básculas, para poder llevar un control de trazabilidad debemos llevarla con datos teóricos. El mayor problema que encontramos con este tipo de sistemas es que no tendremos fiabilidad en los stocks de cada uno de los almacenes, desconociendo en ese caso el estado de los lotes pendientes de gasto.

El entorno en el que se mueve el sistema tiene ciertas peculiaridades que deben ser controladas:

- Traspasos entre almacenes: consiste en traspasar una cantidad determinada de material de un almacén a otro.
- Regularizaciones: es habitual que debido a los errores de las básculas los stocks poco a poco tengan descuadres, por tanto se hace necesario disponer de un mecanismo para poder ajustar el stock a la cantidad correcta. La regularización puede provocar la creación de un lote de entrada, en el caso de tener que añadir material para ajustar el stock a la cantidad deseada, o de disminuir el nivel actual de los lotes pendientes de gasto en el almacén, para ajustar su stock a la cantidad deseada.

- Transmutaciones: existen casos en los que al realizar una entrada de material en un almacén, este tiene todavía una pequeña cantidad de material. Por lo general, antes de realizar una entrada de material en un almacén que previamente tenía otro material, se suele vaciar, pero en ocasiones esto no es así, por lo cual, el sistema debe contemplar esta peculiaridad. Esto suele ocurrir con los restos de material. En estos casos para no perder la trazabilidad de las pequeñas cantidades restantes del material anterior, el sistema convierte ese resto en el nuevo material, pero manteniendo una referencia con su lote original.

Disciplinas de consumos

Dependiendo del tipo de almacén, los materiales se consumen siguiendo una disciplina. En el sistema trabajamos con tres tipos de disciplina, pero según necesidades podrían ampliarse:

- FIFO: primer lote en entrar será el primero en consumirse.
- LIFO: último lote en entrar será el primero en consumirse.
- Fecha de caducidad: se consumirán primero los lotes con fecha de caducidad más cercana.

Como hemos visto un sistema de trazabilidad puede ayudar de manera importante a mejorar nuestro negocio. Es fundamental tener claro el tipo de control que deseamos llevar, siendo conscientes en todo momento de las distintas problemáticas que puedan surgir, de tal forma que podemos minimizarlas.

1.3. Generador de informes

Dado que el sistema es grande y complejo el usuario necesita de una herramienta que le suministre la información que necesita en cada momento y de una manera sencilla. Es por este motivo que sea necesario proveer al usuario de un sistema de informes que le ayude a evaluar el estado de su negocio, y de esta manera estar en condiciones de tomar decisiones oportunas y eficaces.

Esta herramienta debe ser eficaz y fácil de usar, permitiendo al usuario encontrar la información sin necesidad de comprender la estructura de datos.

Dada la complejidad en los datos se hace imprescindible, para obtener la trazabilidad de los lotes, disponer de informes útiles que muestren al usuario la información de forma directa y rápida.

Gracias a las comunicaciones actuales es posible suministrar información sin necesidad de estar físicamente en el centro que dispone de los datos, por ello nuestro sistema debe aprovechar estas ventajas y permitir ejecutar los informes de manera remota. De tal forma que, por ejemplo, el jefe de producción sea capaz de obtener la información que necesita para desarrollar su trabajo sin necesidad de desplazarse a los distintos centros de producción. Este mismo caso se puede aplicar con el personal de calidad, que podrá consultar los datos de las fabricaciones y de la trazabilidad de lotes, analizarlos y tomar decisiones en base a ellos.

Dado que según el momento se puede necesitar una información u otra, con un formato u otro, el sistema debe permitir añadir nuevos informes según la necesidad de los usuarios.

1.4. Adaptación de las aplicaciones existentes

Como hemos indicado anteriormente este proyecto surge de la necesidad por parte de Proin S.L. de disponer de un nuevo sistema que le ayude a ajustarse a la nueva instrucción EHE-08. Es por este motivo, por el que debemos realizar la adaptación de las aplicaciones existentes en Proin, S.L. que son afectadas por este nuevo sistema.

Conex es el sistema desarrollado por Proin, S.L., orientado a la automatización de fabricación industrial. Es un sistema desarrollado en java y multiplataforma, lo que le permite ejecutarse en distintos sistemas operativos.

Se adapta a cualquier tipo de planta de hormigón y se configura en función de la planta a informatizar. Es un sistema muy configurable que permite además personalizar la aplicación de gestión, en el caso de quererlo el cliente.

Actualmente el sistema Conex ha sido aplicado en los siguientes campos:

- Hormigón
- Prefabricados
- Morteros
- Áridos
- Mezclas bituminosas y asfaltos
- Alimentación animal y fabricas de piensos compuestos para animales

El sistema está compuesto principalmente por 4 componentes:

- Conex M2
- IU o Sistema de gestión
- IG o Sinóptico de planta
- Servidor de materiales o DataServer

Conex M2

Es el módulo principal, se ocupa de gestionar los distintos procesos de automatización. Es capaz de enlazar con los módulos de entrada/salida a través del bus CAN, y con el ordenador a través de una conexión Ethernet.

Para controlar la planta el M2 es capaz de recibir mensajes de los módulos de entradas, conociendo así el estado de la planta controlada, y es capaz de enviar mensajes a los módulos de salida, provocando acciones en la planta.

En la ilustración 6 podemos ver una imagen del dispositivo Conex M2.

Ilustración 6. Conex M2

IU o Sistema de gestión

El sistema de gestión nos permite a nivel de usuario gestionar una planta. A través de él se dan de alta los centros de producción, las plantas, los almacenes, las materias primas, los clientes, las obras, etc. Gracias a esta herramienta somos capaces de controlar las dosificaciones de los productos a fabricar, así como ordenar al Conex M2 que fabrique con la dosificación adecuada. Es a través de este módulo donde podemos obtener los distintos informes (clientes, producción, trazabilidad, etc.).

El IU es capaz de comunicar con el M2 tanto para lanzar una fabricación, como para esperar mensajes determinados, los cuales nos indican el estado de la fabricación en curso, nos piden que saquemos el albarán de la fabricación, en fabricas de mortero y prefabricado puede indicarnos que lanzamos en automático una nueva fabricación, etc.

También es capaz de comunicar con el IG, que veremos a continuación, para indicarle que muestre información de la fabricación (cantidad, fórmula y camión).

En la ilustración 7 podemos ver el menú del IU.

Ilustración 7. Sistema de gestión (IU)

Teniendo en cuenta que es a través de este módulo desde el que vamos a crear cada uno de los componentes de la planta se hace necesario adaptarlo para ser capaz de comunicar con el DataServer. Necesitaremos crear un cliente de conexiones con el DataServer, así como un módulo que permita enviar a través del cliente los comandos específicos y esperar la respuesta de DataServer para actuar en consecuencia.

IG o Sinóptico de planta

El sinóptico nos permite ver la planta y su estado en todo momento. Nos muestra un esquema de la planta y el estado de cada uno de los componentes en tiempo real. Gracias a él somos capaces de activar o desactivar máquinas de la planta, ya que se conecta como cliente al Conex M2, recibiendo las señales de entrada y de salida, y pudiendo enviarle mensajes específicos para que active o desactive señales, consiguiendo así interactuar con la planta.

Como hemos explicado en el punto anterior, el IU se conecta como cliente del IG, pero este no como cliente del IU.

En la ilustración 8 podemos ver una imagen del sinóptico de planta.

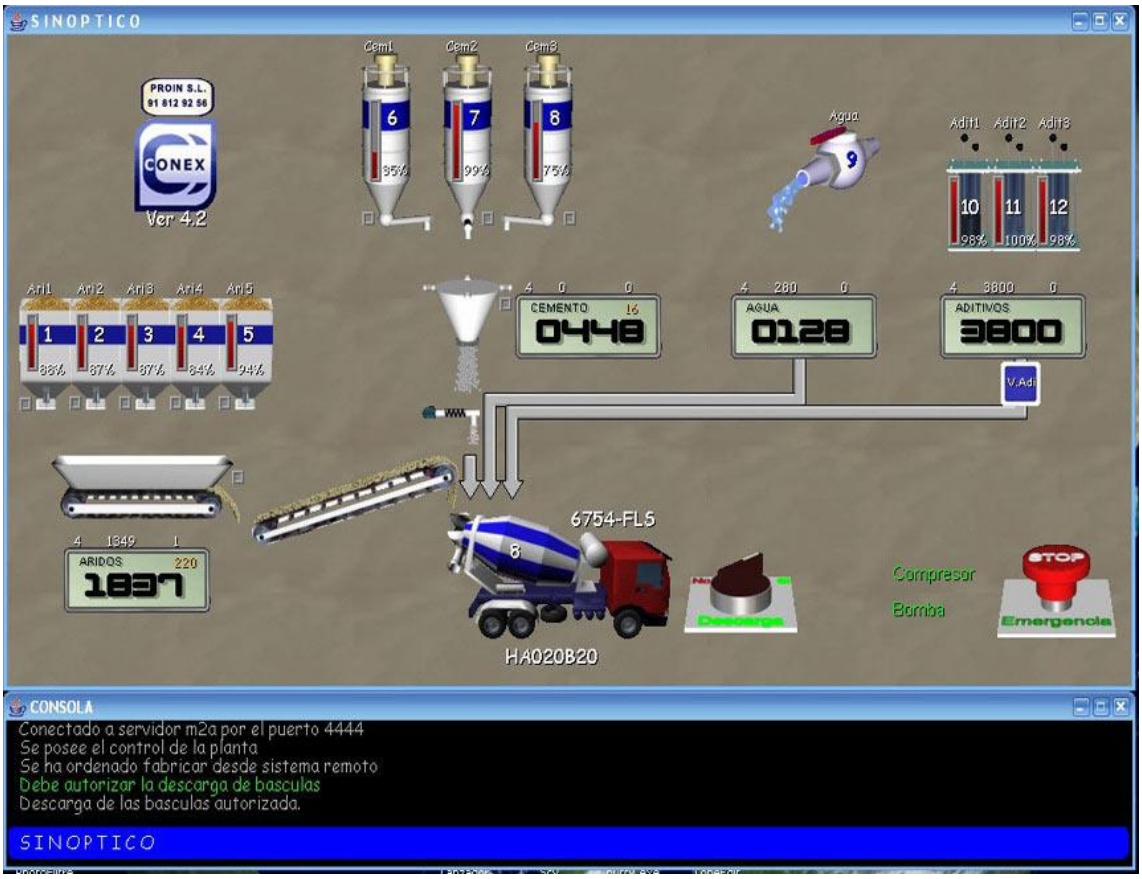


Ilustración 8. Sinóptico de planta (IG)

El sinóptico cuenta con dos componentes: el sinóptico y la consola. El sinóptico muestra un esquema de la planta y permite interactuar con la planta, sin embargo, la consola va indicando el estado del IG (conexiones como cliente con el resto de módulos) y de la fabricación en curso. En caso de notificaciones de error o avisos del Conex M2, este los muestra.

El IG se conecta como cliente al Conex M2 para así obtener en tiempo real las distintas señales y poder actualizar su estado en las imágenes. Por otro lado se conecta como cliente del DataServer, ya que debe recibir información instantánea del material de cada almacén, del nivel del stock y si algún almacén ha sido desactivado por el usuario para no usarlo en fabricación. Para recibir los mensajes de actualización del estado de los almacenes, el IG debe suscribirse en el DataServer al centro de producción que representa.

Nos vemos en la necesidad de adaptar el IG para poder comunicarse con el DataServer y así poder registrarse para el centro de producción que representa, recibiendo de esta manera mensajes instantáneos de actualización.

Servidor de materiales o DataServer

Es el módulo que nace de este proyecto.

2. OBJETIVOS

Una vez realizada una introducción, en líneas generales, sobre el contexto del negocio en el que se engloba nuestro proyecto, detallaremos los objetivos principales a conseguir al término del proyecto.

En este proyecto se plantea la elaboración de un sistema que permita crear y controlar el entorno de un centro de producción, junto con todos sus componentes y peculiaridades. El sistema debe permitir crear dicha estructura, así como interactuar con los objetos con el objetivo de poder remodelar el centro de producción en caso de cambios.

El sistema debe permitir la conexión de clientes, de cara a poder interactuar con él. Para ellos debe disponer de un protocolo de conexión conocido por los clientes para su conexión. Los clientes, usando dicho protocolo de conexión, podrán conectarse al sistema, para de esta manera interactuar con él mediante un protocolo de comunicación conocido por ambos.

El sistema debe tener trazados en todo momento los materiales que en él habitan, y debe permitir trabajar con ellos, de cara a poder controlar de manera eficaz los datos de trazabilidad.

El sistema debe suministrar al usuario una herramienta para la obtención de informes, de forma que estos puedan ayudar al usuario a mejorar y controlar su trabajo. Los informes deben ser accesibles desde cualquier parte, siempre y cuando haya acceso al sistema desde el equipo cliente.

El sistema debe permitir su arranque con tres perfiles diferenciados:

- Arranque completo: este modo arrancará todo el sistema.
- Arranque sin informes: este modo arrancará todo el sistema menos el servidor de informes.
- Arranque en modo informes: este modo sólo arrancará el servidor de informes.

Dado que este proyecto surge de la necesidad por parte de Proin, S.L. de desarrollar un sistema que complete la funcionalidad que le falta, dadas las nuevas especificaciones, debemos realizar la adaptación de los módulos cliente de Proin, S.L. para su correcto funcionamiento con el nuevo sistema.

Teniendo en cuenta que la información con la que trabajará el sistema es de máxima importancia en el negocio en el que se mueve, es imprescindible que el sistema sea robusto y fiable.

Dado que el negocio en el que se mueve o podría moverse el sistema puede requerir de la implementación de nuevas especificaciones, necesitamos un sistema lo más estable posible, y que permita su escalabilidad, de cara a su crecimiento según necesidades del negocio.

2.1. Metodología

En el desarrollo del software se hace necesario seguir una metodología de trabajo que nos ayude a mejorar el proceso de desarrollo. La metodología nos ofrece un conjunto de métodos, técnicas y herramientas que establecen un camino para desarrollar software de manera sistemática, proporcionándonos así un estándar de trabajo para la empresa.

El uso de una metodología nos guiará en el proceso de desarrollo de software. Nos aportará métodos apropiados para cada una de las fases de desarrollo, desde la fase inicial en la que se especifica el sistema a desarrollar hasta la fase de mantenimiento una vez implantada.

Actualmente existen diferentes metodologías, pero las más usadas son el Proceso Unificado de Desarrollo Software y la Programación Extrema.

El Proceso Unificado de Desarrollo Software [2] guía al equipo de desarrollo a administrar el desarrollo iterativo de un modo controlado manejando los requisitos del negocio, el tiempo y los riesgos. Nos suministra un marco de trabajo que se caracteriza por estar centrado en una arquitectura, dirigido por casos de uso y por ser iterativo e incremental. Nos suministra patrones de diseño y es soportado por herramientas que automatizan, entre otras cosas, el modelado, la administración de cambios y las pruebas.

La Programación Extrema [3] es la metodología más destacada de los procesos ágiles de desarrollo de software. Se basa en el desarrollo y entrega de incrementos de funcionalidad muy limitada. La Programación Extrema se diferencia de las metodologías tradicionales por poner más énfasis en la adaptabilidad que en la previsibilidad.

Estudiando las distintas metodologías se ha optado por El Proceso de Desarrollo Software, ya que se adapta mucho mejor a las exigencias del proyecto, teniendo en cuenta que no necesitamos ir realizando entregas de funcionalidad pequeñas ,y que los requerimientos no deberían ir cambiando con el tiempo, sino que necesitamos un sistema funcional grande al que irle añadiendo, en iteraciones, nuevas funcionalidades según se vayan recibiendo nuevos requerimientos, respetando por lo general los requerimientos ya implementados.

2.2. Tecnologías

A la hora de desarrollar un proyecto hay que analizar y decidir que tecnologías se usarán durante su desarrollo. La decisión de usar unas u otras dependerán de factores como:

- Conocimiento previo de la tecnología. Elegir una tecnología conocida y usada con anterioridad facilita el desarrollo, dado que la curva de aprendizaje será mucho menor y dispondremos de la experiencia previa en la resolución de incidencias surgidas por su uso.
- Uso de tecnología estándar. Elegir una tecnología que se ajuste al estándar permite desarrollar software con mayores garantías de buen funcionamiento, estabilidad y robustez, es decir, permiten desarrollar software de calidad.
- Empleo de tecnología de uso extendido. Trabajar con tecnologías de uso común facilitan la integración del sistema desarrollado en el entorno en el que debe funcionar.
- Uso de tecnología que a priori tendrán un ciclo de vida largo. El uso de tecnologías que a priori están pensadas para permanecer de manera duradera en el mercado e ir creciendo y mejorando, resulta imprescindible de cara al mantenimiento del software a desarrollar, ya que si una tecnología entra en desuso, es probable que también deje de dar soporte ante posibles problemas.

- Uso de la tecnología que se adapte mejor al objetivo definido. Es importante buscar la tecnología justa que se adapta a un requerimiento u objetivo marcado, ya que si no nuestro sistema puede estar consumiendo recursos de manera innecesaria.
- Uso de tecnología que se ajuste a nuestra arquitectura. Elegir una tecnología que se ajuste al tipo de arquitectura del sistema a desarrollar facilita la integración entre componentes.
- Uso de tecnologías compatibles. Usar tecnologías compatibles las unas con las otras facilita la conectividad entre ellas, ayudando así en el desarrollo del sistema.
- Uso de tecnología con licencia comercial o libre. El uso de tecnologías comerciales o libres dependen del presupuesto del que se dispone y de la necesidad concreta del uso de una de las dos.

A continuación se detallarán las distintas tecnologías a utilizar en el desarrollo del sistema.

Java

La plataforma Java [4] es un entorno de computación originaria de Sun Microsystems. Dicha plataforma se compone principalmente de una máquina virtual, encargada de la ejecución de aplicaciones desarrolladas en el lenguaje de programación Java, y de un conjunto de herramientas conocidas como JDK (Java Development Kit) que suministran un conjunto de herramientas de desarrollo para Java.

El lenguaje de programación Java [6] es un lenguaje orientado a objetos que fue desarrollado por Sun Microsystems a principios de los años 90. Se creó con los siguientes objetivos:

- Debería usar la metodología de programación orientada a objetos.
- Debería permitir la ejecución de un mismo programa en distintos sistemas operativos o plataformas.
- Debería incluir por defecto soporte para el trabajo en red.
- Debería diseñarse para ejecutar, de manera segura, código en sistemas remotos.
- Debería ser sencillo de usar y tomar lo mejor de otros lenguajes de programación orientados a objetos, como puede ser C++.

Las razones principales para el uso de la plataforma Java en el desarrollo del proyecto son: la compatibilidad con los sistemas actuales de la empresa, dado que Proin, S.L. desarrolla con tecnología Java, los programadores tiene una formación avanzada en el desarrollo con el lenguaje de programación Java, la posibilidad de migrar el sistema de plataforma o sistema operativo, y viendo la evolución de la plataforma desde su creación consideramos que tendrá un ciclo de vida largo.

Eclipse

Eclipse [7] es un entorno de desarrollo integrado (IDE) de código abierto y multiplataforma. Este entorno de desarrollo, a través de módulos, proporciona toda su funcionalidad, ayudando al usuario en el proceso de construcción, desarrollo y mantenimiento del software durante su ciclo de vida. Se diferencia de los entornos de tipo monolíticos en que estos incluyen toda la funcionalidad, la necesite o no el usuario, sin embargo, Eclipse permite al usuario incorporar los módulos que necesita. Eclipse permite crear aplicaciones de muy diversa naturaleza, como pueden ser aplicaciones web o aplicaciones de escritorio.

Existen dos entornos de desarrollo en Java ampliamente extendidos, estos son Net Beans [8] y Eclipse [7]. Es decisión del programador elegir el entorno a usar durante el desarrollo, siempre y cuando las tareas a realizar sean compatibles en los dos entornos.

JDO

JDO (Java Data Objects) [9] es un API que proporciona una forma estándar y sencilla de conseguir la persistencia de objetos con la tecnología Java. JDO permite realizar de forma sencilla la correlación entre el sistema de objetos Java y el sistema relacional de la base de datos. Establece un interfaz estándar desarrollado bajo el control de la Java Community Process [10].

JDO se caracteriza por ser fácil de usar, permite portabilidad de proveedores de implementación del estándar JDO, trabaja con independencia del tipo de base de datos y permite optimizar los patrones de acceso a los datos delegando los trabajos de persistencia a la implementación JDO con la que se trabaje.

Existen tecnologías similares a JDO como son Hibernate [11] o Enterprise Java Beans (EJB) [12]. Esta última cubre el mismo área que JDO, por ello han existido conflictos

de estándares entre ambas. Al final la persistencia ha ido a parar a Java Persistence API o JPA [13] que se ha convertido en el estándar y como parte de la arquitectura Java (paquete javax.persistence).

En el momento de iniciar el proyecto y elegir las distintas tecnologías, JDO iba a convertirse en el estándar Java para manejo de la persistencia y JPA estaba en sus comienzos. Además la empresa Proin, S.L. ya tenía experiencia en el desarrollo con JDO, es por ello que se optó por esta tecnología.

Como proveedor de implementación del estándar JDO se decidió por Versant [14] debido a que era la implementación que seguía de manera más fiel el estándar JDO, y era las más completa. Mejorando así la calidad del sistema a desarrollar.

JPA

JPA o Java Persistence API [13] proporciona un modelo de persistencia para mapear bases de datos relacionales en Java. JPA es una abstracción sobre JDBC que nos permite realizar de forma sencilla la correlación entre el sistema relacional de la base de datos y el sistema orientado a objetos de java representado por entidades. Establece un interfaz común que es implementado por los distintos proveedores de persistencia como son Hibernate [11] o EclipseLink [15], de forma que podemos decidir el proveedor a utilizar dependiendo del que mejor se adecue a nuestras necesidades. Es el proveedor el que realizará el trabajo, pero siempre funcionando bajo el API de JPA.

JPA se usará para el mapeo de la base de datos de gestión del Sistema Conex de Proin, S.L. de cara a poder obtener informes que relacionen los datos de gestión con los datos del centro de producción mantenidos en el DataServer.

Versant OpenAccess

Versant OpenAccess [14] es una herramienta visual que permite generar la estructura de tablas basándose en el modelo de objetos del sistema, o viceversa, a partir de la estructura de tablas de la base de datos generar el modelo de datos.

Resulta una herramienta de gran utilidad de cara a migraciones del sistema por cambio en el modelo de objetos, ya que analiza la estructura de tablas antigua y obtiene un script de migración al nuevo modelo de objetos.

SQL Server

Microsoft SQL Server [16] es un motor de base de datos fabricado por Microsoft y está basado en el modelo relacional.

Las características principales son:

- Soporte de transacciones.
- Escalabilidad, estabilidad y seguridad.
- Soporta procedimientos almacenados.
- Incluye un entorno visual de administración.
- Permite trabajar en modo cliente-servidor
- Permite administrar información de otros servidores de datos

Existen otras alternativas de uso común en el mercado como son motores de base de datos Oracle, PostgreSQL o MySQL. Durante el desarrollo del proyecto se ha decido utilizar un motor SQL Server debido a que es el motor utilizado en el resto de aplicaciones de la empresa Proin, S.L. Es importante indicar que como disponemos de un sistema de persistencia, el sistema podría también trabajar con otro motor de base de datos simplemente cambiando la librería o driver de conexión a la base de datos.

HSQLDB

HSQLDB (Hypethreaded Structured Query Language Database) [17] es un motor de base de datos escrito en Java, basado en un modelo relacional.

Sus principales características son:

- Sintaxis SQL estándar.
- Trabaja con integridad referencial.
- Permite trabajar con procedimientos almacenados en Java.
- Dispone de triggers.
- Permite tablas en disco de hasta 8 Gb.
- Permite trabajar con bases de datos en memoria.

El uso principal para esta tecnología es usar una base de datos en memoria para el módulo de pruebas, de tal manera que no quede grabada una vez finalizado el proceso.

Como alternativa a HSQLDB está Apache Derby [18], que tiene características similares, o incluso un motor de base de datos del tipo SQL Server u Oracle, sin embargo, para estos dos últimos no es factible su uso debido a que para poner en marcha una base de datos es preciso montar un motor de base de datos permanente, y por tanto las bases de datos serán persistentes, cuando realmente se necesita trabajar con bases de datos en memoria

RMI

RMI (Java Remote Method Invocation) [19] es un mecanismo ofrecido por Java para la invocación remota de un método, es decir, a través de RMI, un programa es capaz de exportar un objeto, quedando este accesible a través de la red, el programa permanece a la espera de peticiones en un puerto TCP, de tal forma que un cliente puede conectarse e invocar los métodos que proporciona el objeto.

RMI forma parte del estándar de ejecución de Java y proporciona un mecanismo simple para la comunicación de servidores en aplicaciones distribuidas basadas exclusivamente en Java. Si es necesaria la comunicación entre tecnologías distintas de Java debemos utilizar alternativas como CORBA [20] o SOAP [21]. La principal característica de RMI es su facilidad de uso en la programación dado que está específicamente diseñada para Java.

En el desarrollo de proyecto se escoge el uso de RMI frente a otras tecnologías por estar trabajando con tecnologías Java, proporcionando así una integración rápida y sencilla con el resto de tecnologías empleadas.

Jetty

Jetty [22] es un servidor http basado en Java. Es un proyecto de software libre disponible para su distribución y uso comercial. El objetivo de Jetty es disponer de un servidor web sencillo, eficiente y ligero. Una de las principales ventajas es su tamaño reducido, que permite disponer de un servidor web en una aplicación Java empotrada.

Como alternativa a Jetty encontramos Apache Tomcat [23], sin embargo, esta tecnología resulta mucho más pesada, aunque potente, no siendo necesaria tanta potencia en el sistema a desarrollar.

Jython

Jython [24] es un lenguaje de programación de alto nivel, interpretado y orientado a objetos que está basado en Python [25] e implementado en Java. Al tratarse de un lenguaje de programación interpretado permite escribir y ejecutar código sin necesidad de compilar. Otra de las principales ventajas de Jython se encuentra en el hecho de que las variables son genéricas y dinámicas, pudiendo utilizarse indistintamente para representar enteros, cadenas, clases, etc., y su tipo puede ser definido o cambiado en cualquier momento de la ejecución del programa.

La elección de esta tecnología para su uso se debe a la necesidad de poder ejecutar código Java, tener acceso a los datos del DataServer a través de RMI, y poder cambiar el diseño de las ventanas sin necesidad de compilar el módulo.

JasperReport

JasperReports [26] es un motor de informes desarrollado en Java capaz de trabajar con distintos tipos de origen de datos, como pueden ser una consulta SQL o un objeto. Los informes generados pueden ser mostrados por pantalla, impresos o exportados a formatos de uso común como son PDF, Excel, OpenOffice, Word y HTML. Al estar desarrollado en Java puede ser embebido en cualquier aplicación Java.

Se elige JasperReport como motor de informes por su fácil integración con el resto de tecnologías usadas en el proyecto, así como por la facilidad de trabajar no sólo con consultas SQL, sino también con los propios objetos del DataServer.

IReport

IReport [27] es un diseñador de informes visual, potente y fácil de usar. Esta implementado en Java y se apoya en el motor de informes JasperReport. Permite trabajar con subinformes y está integrado con la librería de gráficas JFreeChart [28].

Omondo

Omondo [29] es un módulo o plug-in de Eclipse que permite modelar un sistema mediante diagramas UML [2] y a partir de estos es capaz de generar la estructura de clases del proyecto. Se elige este plug-in para modelado UML principalmente por el hecho de haber trabajado con él con anterioridad.

JUnit

JUnit [30] es un framework que facilita la realización de pruebas unitarias en Java. Este framework es útil para controlar las pruebas de regresión, necesarias ante cambios en el código, de cara a comprobar que el sistema sigue cumpliendo con los requerimientos especificados. JUnit se encuentra disponible como módulo o plug-in de Eclipse.

JBossRules

JBossRules [31] es un motor de reglas realizado en Java. Permite evaluar un juego de reglas para los objetos de negocio del sistema. El juego de reglas puede ser leído de un fichero, el cual puede ser modificado y puesto en producción sin necesidad de compilar una nueva distribución.

2.3. Plan de trabajo

Como se ha explicado en puntos anteriores, el sistema a desarrollar supone un complejo y sofisticado software que pretende controlar un centro de producción, así como dar servicio a información, complicada de tratar sin un preprocesado, a los clientes que así lo requieran.

Durante el desarrollo será necesario abordar distintos ámbitos de la informática, como pueden ser la persistencia, servicios web, sockets, hilos o interfaces. Es por este motivo por el que es necesario dividir el proyecto en distintas tareas y personal.

El proyecto lo dividimos en distintos módulos, que se explicarán en el punto 3:

- Main: módulo de arranque del sistema
- API y modelos: API para trabajar con el sistema de datos y la implementación del modelo de datos
- Servidor: módulo de conexión con clientes, interpretación de mensajes de cliente para la generación de comando, gestos de la cola de comandos y envío de mensajes de vuelta a los clientes.
- Generador de informes: módulo servidor de informes.

Como partes específicas desarrolladas en este Proyecto Fin de Carrera se encuentran el módulo Main, el módulo Servidor, el servidor de informes, el desarrollo de todas las pruebas de validación del sistema, la adaptación de las aplicaciones de existentes de Proin, S.L., la generación del cliente de evaluaciones meteorológicas con su interfaz de usuario, y la implantación en los clientes.

Como partes comunes en la que he intervenido se encuentran la etapa de especificación de requerimientos y de análisis para todos los módulos, y diseño e implementación del API y modelos de datos, aunque en este último punto la intervención fue menor.

Se tienen los hitos de entrega siguientes:

1. Hito 1. Planificación global del proyecto.
2. Hito 2. Arquitectura operativa del sistema para un conjunto reducido de comandos. Incluye:
    a. Implementación de la arquitectura.
    b. Implementación de un comando o conjunto reducido de comandos.
    c. Implementación básica del módulo Servidor
    d. Implementación básica del módulo de arranque Main
    e. Prueba de la arquitectura
3. Hito 3. Fase de construcción de la primera iteración. Incluye:
    a. Implementación del módulo Main.
    b. Implementación del módulo Modelos y API.
    c. Implementación del módulo Servidor.
    d. Implementación del módulo Servidor de informes.
    e. Adaptación de las aplicaciones de Proin, S.L.
    f. Pruebas del sistema.
    g. Pruebas en clientes reales.
4. Hito 4. Fase de despliegue de la primera iteración, según concluye la iteración de la fase de construcción correspondiente a la primera iteración. Incluye:
    a. Pruebas de validación.
    b. Puesta en marcha del sistema en clientes.
5. Hito 5. Fases de construcción de la segunda iteración. Incluye:
    a. Implementación incremental del módulo Main.
    b. Implementación incremental del módulo Modelos y API.

    c. Implementación incremental del módulo Servidor.
    d. Implementación incremental del módulo Servidor de informes.
    e. Adaptación de las aplicaciones de Proin, S.L.
    f. Pruebas del sistema.
    g. Pruebas en clientes reales.
6. Hito 6. Fase de despliegue de la segunda iteración, según concluye la iteración de la fase de construcción correspondiente a la segunda iteración. Incluye:
    a. Pruebas de validación.
    b. Puesta en marcha del sistema en clientes.
7. Hito 7. Fases de construcción de la tercera iteración. Incluye:
    a. Implementación incremental del módulo Main.
    b. Implementación incremental del módulo Modelos y API.
    c. Implementación incremental del módulo Servidor.
    d. Implementación incremental del módulo Servidor de informes.
    e. Adaptación de las aplicaciones de Proin, S.L.
    f. Pruebas del sistema.
    g. Pruebas en clientes reales.
8. Hito 8. Fase de despliegue de la tercera iteración, según concluye la iteración de la fase de construcción correspondiente a la tercera iteración. Incluye:
    a. Pruebas de validación.
    b. Puesta en marcha del sistema en clientes.

En la ilustración 9 podemos ver el diagrama de Gantt con el plan de desarrollo previsto.

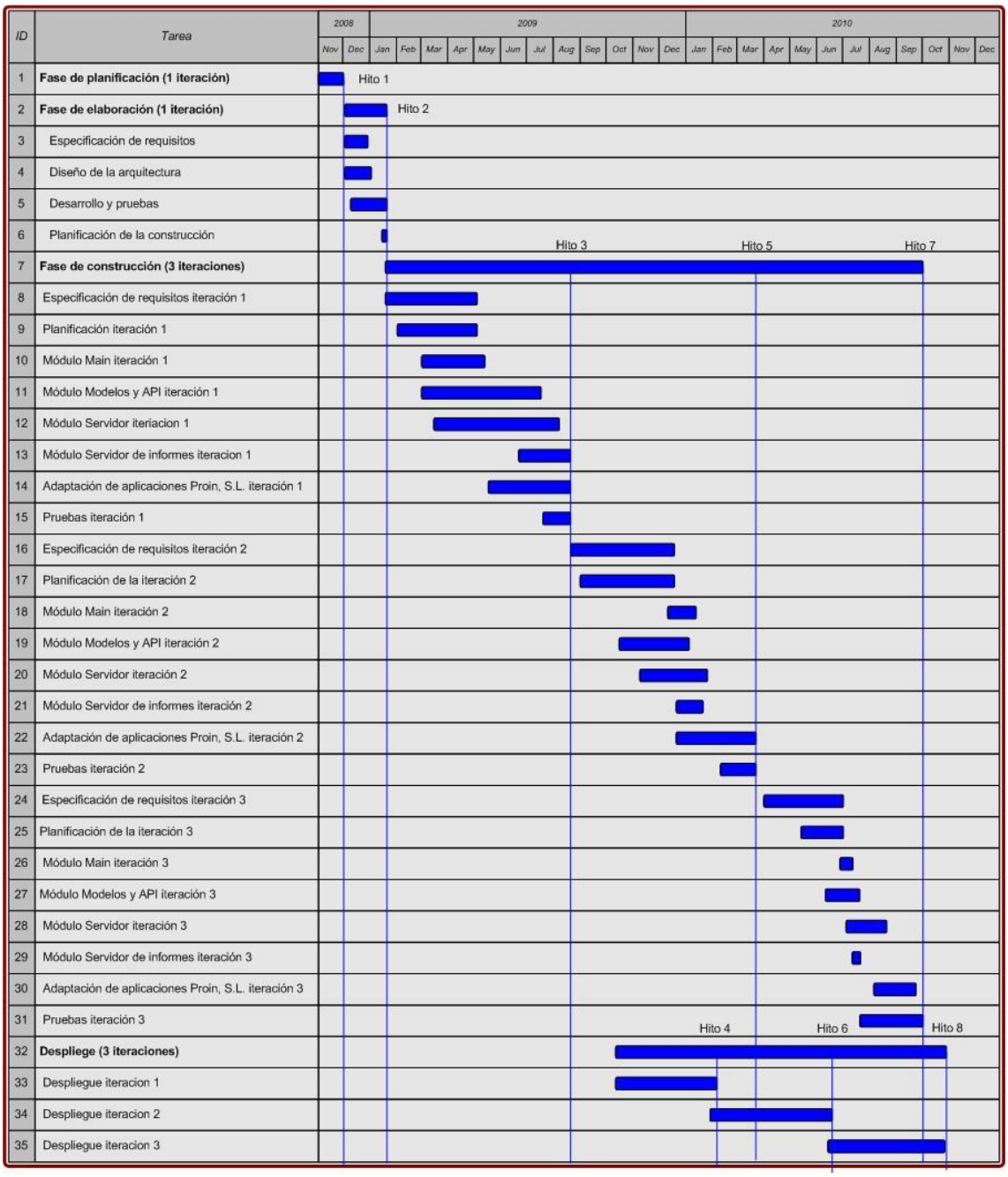


Ilustración 9. Diagrama de Gantt

3. DESCRIPCIÓN INFÓRMATICA

En esta sección veremos los aspectos técnicos del desarrollo del sistema. Comenzaremos describiendo la especificación de requisitos, a continuación nos centraremos en los aspectos de la arquitectura y el diseño de los módulos principales del sistema, y por último hablaremos sobre las pruebas realizadas sobre el sistema.

3.1. Especificación de requisitos

La identificación y especificación de requisitos supone una de las etapas más relevantes a la hora de desarrollar un sistema ya que nos proporciona una descripción detallada del comportamiento que debe tener este. Un requisito es una propiedad que el sistema debe poseer. Los sistemas se diseñan desde una perspectiva de uso, por ello la especificación de requisitos incluye un conjunto de casos de uso que describen las interacciones entre el sistema y una colección de actores produciendo un resultado observable y útil para ellos. El establecimiento de casos de uso permite capturar los requisitos funcionales con los que el sistema debe contar. Un requisito funcional es un servicio que el sistema ha de saber prestar. Se deben tener en cuenta también los requisitos no funcionales, que son aquellos que imponen restricciones que el sistema debe satisfacer.

3.1.1. Requisitos funcionales

3.1.1.1. Requisitos de Usuario

Para obtener los requisitos funcionales de usuario nos basaremos en los casos de uso del sistema.

Se pueden diferenciar los casos de uso por tipo, es por este motivo que definimos tres tipos de representación abstractos: Call, CallBack y Callback de actualización y consulta.

- Casos de uso Call. Esta representación de caso de uso se caracteriza por que el cliente manda una petición de comando al sistema, pero el sistema no notificará al cliente el resultado de la operación. A continuación puede verse el flujo

estándar y flujo alternativo que comparten todos los casos de uso pertenecientes a esta representación.

Flujo estándar	1. El cliente envía la petición de comando al sistema. 2. El sistema realiza la operación. 3. El sistema modifica el estado de la base de datos.
Flujo alternativo	1. El cliente envía la petición de comando al sistema. 2. El sistema lanza una excepción controlada 3. El sistema deja el estado de la base de datos en el estado previo al procesado de la petición.

- Casos de uso CallBack. Esta representación de caso de uso se caracteriza por que el cliente manda una petición de comando al sistema, y el sistema enviará al cliente el resultado de la ejecución de la operación, devolviendo "OK" o "ERROR". A continuación puede verse el flujo estándar y flujo alternativo que comparten todos los casos de uso pertenecientes a esta representación.

Flujo estándar	1. El cliente envía la petición de comando al sistema. 2. El sistema realiza la operación. 3. El sistema modifica el estado de la base de datos. 4. El sistema devuelve al cliente "OK" como resultado de la operación.
Flujo alternativo	1. El cliente envía la petición de comando al sistema. 2. El sistema no puede realizar la operación. 3. El sistema deja el estado de la base de datos en el estado previo al procesado de la petición. 4. El sistema devuelve al cliente "ERROR: Mensaje de error", como resultado de la operación.

- Caso de uso Callback de actualización y consulta. Esta representación de caso de uso se caracteriza por que el cliente manda una petición de comando al sistema, y el sistema enviará al cliente el resultado de la ejecución de la operación, devolviendo "true", "false" o "ERROR". A continuación puede verse el flujo

estándar y flujo alternativo que comparten todos los casos de uso pertenecientes a esta representación.

Flujo estándar	1. El cliente envía la petición de comando al sistema. 2. El sistema devuelve “true” o “false”.
Flujo alternativo	1. El cliente envía la petición de comando al sistema. 2. El sistema no puede realizar la petición. 3. El sistema devuelve al cliente “ERROR: Mensaje de error”, como resultado de la petición.

Casos de uso Call

- Caso de uso nº 1.

Nombre	Alta de muestra meteorológica.
Descripción	Permite agregar una muestra meteorológica al sistema.
Precondiciones	1. El código externo de la estación no puede ser null. 2. La estación meteorológica debe existir.

- Caso de uso nº2:

Nombre	Registro de consumo semiautomático
Descripción	Permite dar de alta un consumo semiautomático para el almacén indicado, dados la fabricación, cantidad y unidades.
Precondiciones	1. La unidad no puede ser null y debe existir. 2. La cantidad no puede ser null. 3. El material debe existir. 4. El almacén debe existir.

- Caso de uso nº 3:

Nombre	Registro de apunte
Descripción	Permite dar de alta un apunte asociado a una boca, M2, fabricación, ciclo de fabricación y motivo, con la cantidad teórica y final dadas.
Precondiciones	1. El motivo no puede ser null. 2. Debe existir un medidor con el código de boca, M2 y centro de producción indicados. 3. Debe existir una máquina con el código de boca, M2 y centro de producción indicados.

- Caso de uso nº 4:

Nombre	Aviso de nivel
Descripción	El sistema recibe un aviso de nivel de un almacén en un estado determinado. El sistema realizará las regularizaciones oportunas para ajustar el stock teórico al real.
Precondiciones	1. El nivel no puede ser null. 2. Debe existir un medidor con el código de boca, M2 y centro de producción indicados. 3. Debe existir una máquina con el código de boca, M2 y centro de producción indicados. 4. El almacén al que hace referencia el aviso de nivel debe existir.

- Caso de uso nº 5:

Nombre	Notificación de fin de ciclo
Descripción	Permite hacer un acumulado de consumos de una fabricación para un ciclo, y en el caso de ser una fabricación en la que se indica un material y almacén de destino, realizará una entrada automáticamente con la cantidad acumulada en los consumos del ciclo al almacén de destino, cambiando el material del almacén destino si no corresponde con el fabricado.
Precondiciones	1. La fabricación debe existir.

- Caso de uso nº 6:

Nombre	Mensaje de fabricación
Descripción	Permite registrar una anotación recibida del M2 durante una fabricación.
Precondiciones	1. La fabricación debe existir.

Casos de uso CallBack

- Caso de uso nº 7:

Nombre	Añadir nivel a un almacén
Descripción	Permite añadir un nivel a un almacén.
Precondiciones	1. El nivel debe existir. 2. El almacén debe existir. 3. El almacén debe ser exclusivo.

- Caso de uso nº 8:

Nombre	Añadir almacén a un agregado
Descripción	Permite agregar un almacén del tipo que sea a un almacén agregado.
Precondiciones	1. El almacén agregado y el almacén a agregar no pueden ser el mismo. 2. El almacén agregado debe existir. 3. El almacén a agregar debe existir. 4. El almacén agregado debe ser de tipo agregado. 5. El almacén a agregar no debe estar ya asociado al almacén agregado.

- Caso de uso nº 9:

Nombre	Alta de almacén agregado simple.
Descripción	Permite dar de alta un almacén agregado simple.
Precondiciones	1. El nombre del almacén no puede ser null o vacía. 2. El almacén contenedor debe existir. 3. No puede existir otro almacén con el mismo código perteneciente al almacén contenedor.

- Caso de uso nº 10.

Nombre	Alta de almacén exclusivo
Descripción	Permite dar de alta un almacén exclusivo.
Precondiciones	1. El nombre del almacén debe ser distinto de null y vacío. 2. La unidad debe ser distinta de null y debe existir. 3. El código externo del almacén debe ser único. 4. No puede existir otro almacén con el mismo código perteneciente al almacén contenedor.

- Caso de uso nº 11:

Nombre	Añadir destino a un acopio.
Descripción	Permite añadir un almacén como destino de un acopio.
Precondiciones	1. El acopio debe existir 2. El almacén destino debe existir. 3. El almacén destino no puede estar vinculado.

- Caso de uso nº 12:

Nombre	Añadir origen a un acopio.
Descripción	Permite añadir un almacén como origen de un acopio.
Precondiciones	1. El acopio debe existir 2. El almacén origen debe existir.

- Caso de uso nº 13:

Nombre	Alta de estación meteorológica
Descripción	Permite dar de alta una estación meteorológica.
Precondiciones	1. No puede existir otra estación con el mismo código.

- Caso de uso nº 14:

Nombre	Alta de centro de producción
Descripción	Permite dar de alta un centro de producción.
Precondiciones	1. El código no puede estar vacío. 2. La descripción no puede estar vacía. 3. No puede existir ningún centro o almacén con ese código.

- Caso de uso nº 15:

Nombre	Alta de acopio.
Descripción	Permite dar de alta un acopio.
Precondiciones	1. El nombre del acopio no puede ser null. 2. El nombre del m2 no puede ser null. 3. No puede existir otro acopio con el mismo nombre y m2. 4. El almacén contenedor del acopio debe existir.

- Caso de uso nº 16:

Nombre	Alta de artículo
Descripción	Permite dar de alta un artículo.
Precondiciones	1. El código externo no puede ser vacío. 2. La unidad debe ser distinta de null y debe existir. 3. El código externo del artículo debe ser único. 4. La descripción no puede estar vacía.

- Caso de uso nº 17:

Nombre	Alta comprable para un artículo
Descripción	Permite dar de alta la faceta de comprable a un artículo.
Precondiciones	1. El artículo debe existir.

- Caso de uso nº 18:

Nombre	Alta de boca
Descripción	Permite dar de alta una boca.
Precondiciones	1. El centro debe existir. 2. El almacén debe existir, ser exclusivo y pertenecer al centro. 3. El código de boca debe ser único para un centro, almacén y m2. 4. La unidad debe ser distinta de null y debe existir.

- Caso de uso nº 19:

Nombre	Alta vendible para un artículo
Descripción	Permite dar de alta la faceta de vendible a un artículo.
Precondiciones	1. El artículo debe existir.

- Caso de uso nº 20:

Nombre	Alta de juego de fabricación
Descripción	Permite dar de alta un juego de fabricación.
Precondiciones	1. No puede existir un juego con el mismo código asociado al centro de producción indicado.

- Caso de uso nº 21:

Nombre	Alta de fabricación
Descripción	Permite dar de alta una fabricación.
Precondiciones	1. Si se indica almacén de destino este debe existir, la unidad no puede ser null y el material de destino no puede ser null. 2. Si se indica material de destino este debe existir, el almacén de destino no puede ser null y la unidad no puede ser null. 3. Si se indica unidad, el almacén y material de destino no pueden ser null.

- Caso de uso nº 22:

Nombre	Alta de máquina
Descripción	Permite dar de alta una máquina.
Precondiciones	1. El código debe ser único para el centro indicado. 2. La unidad no puede ser null y debe existir. 3. No puede existir ningún centro o almacén con ese código.

- Caso de uso nº 23:

Nombre	Alta de medidor
Descripción	Permite dar de alta un medidor.
Precondiciones	1. No puede existir un medidor con el mismo código de dosificación, m2 y centro. 2. Debe existir la máquina a la que pertenecerá el medidor. 3. La unidad debe existir.

- Caso de uso nº 24:

Nombre	Alta de materia prima
Descripción	Permite dar de alta una materia prima.
Precondiciones	1. El artículo debe existir. 2. No puede existir un material ya asignado al artículo.

- Caso de uso nº 25:

Nombre	Asignar fabricación
Descripción	Permite asignar una fabricación a un consumo viudo.
Precondiciones	1. El consumo debe existir. 2. La fabricación debe existir. 3. El consumo no puede estar asignado a otra fabricación.

- Caso de uso nº 26:

Nombre	Alta de planta
Descripción	Permite dar de alta una planta.
Precondiciones	1. El nombre de la planta no puede ser null o vacío. 2. El código externo no puede ser null. 3. No puede existir ninguna planta o almacén en el mismo nivel con el mismo código.

- Caso de uso nº 27:

Nombre	Anular entrada de material
Descripción	Permite anular una entrada de material realizada.
Precondiciones	1. La entrada debe existir.

- Caso de uso nº 28:

Nombre	Asignar juego de fabricación
Descripción	Permite asignar un medidor a un juego de fabricación.
Precondiciones	1. El juego de fabricación debe existir. 2. El medidor debe existir.

- Caso de uso nº 29:

Nombre	Asignar material
Descripción	Permite asignar un material a un almacén.
Precondiciones	1. El almacén debe existir. 2. El material debe existir.

- Caso de uso nº 30:

Nombre	Consigna material
Descripción	Permite reservar una cantidad de material, de cara a una validación posterior de la cantidad real final.
Precondiciones	1. La cantidad no puede ser menor o igual a cero. 2. La unidad no puede ser null y debe existir.

- Caso de uso nº 31:

Nombre	Baja vendible para un artículo
Descripción	Permite dar de baja la faceta de vendible a un artículo.
Precondiciones	1. El artículo debe existir.

- Caso de uso nº 32:

Nombre	Conversión de unidad
Descripción	Permite convertir una cantidad de una unidad a otra dado un material.
Precondiciones	1. La unidad origen debe existir. 2. La unidad destino debe existir. 3. El material debe existir.

- Caso de uso nº 33:

Nombre	Desasignar juego de fabricación
Descripción	Permite desasignar un medidor de un juego de fabricación.
Precondiciones	1. El juego debe existir. 2. El medidor debe existir.

- Caso de uso nº 34:

Nombre	Desasignar material
Descripción	Permite desasignar un material de un almacén.
Precondiciones	1. El material debe existir. 2. El almacén debe existir.

- Caso de uso nº 35:

Nombre	Edición de artículo y material
Descripción	Permite editar conjuntamente un artículo y su material.
Precondiciones	1. El artículo debe existir. 2. El código de artículo debe ser único. 3. El material debe existir. 4. El material debe estar asociado al artículo indicado. 5. La unidad debe existir.

- Caso de uso nº 36:

Nombre	Deshabilitar acopio
Descripción	Permite deshabilitar un acopio.
Precondiciones	1. El acopio debe existir.

- Caso de uso nº 37:

Nombre	Desvincular almacén
Descripción	Permite desvincular un almacén de su almacén delegado.
Precondiciones	1. El almacén vinculado debe existir y debe ser exclusivo.

- Caso de uso nº 38:

Nombre	Edición de acopio
Descripción	Permite editar los datos de un acopio.
Precondiciones	1. El acopio a editar debe existir. 2. No puede existir otro acopio con el mismo nombre, código, m2 y almacén contenedor.

- Caso de uso nº 39:

Nombre	Edición de boca
Descripción	Permite editar los datos de una boca.
Precondiciones	1. La boca a editar debe existir. 2. Existe un medidor para la boca. 3. El nuevo centro existe. 4. El nuevo almacén existe. 5. No puede existir otra boca con mismo código para el mismo centro y almacén.

- Caso de uso nº 40:

Nombre	Edición de artículo
Descripción	Permite editar los datos de un artículo.
Precondiciones	1. El artículo debe existir. 2. El código debe ser único.

- Caso de uso nº 41:

Nombre	Edición de almacén exclusivo.
Descripción	Permite editar los datos de un almacén exclusivo.
Precondiciones	1. El nombre debe ser distinto de null y vacío. 2. La cantidad debe existir. 3. La capacidad debe ser mayor a cero. 4. El almacén a editar debe existir. 5. No puede existir otro almacén con el mismo nombre, para el mismo almacén contenedor. 6. No puede existir otro almacén con el mismo código para el mismo almacén contenedor.

- Caso de uso nº 42:

Nombre	Edición de máquina
Descripción	Permite editar los datos de una máquina.
Precondiciones	1. La máquina a editar debe existir. 2. El código de máquina debe ser único para el centro indicado. 3. La descripción de la máquina debe ser única.

- Caso de uso nº 43:

Nombre	Edición de boca y medidor
Descripción	Permite editar los datos de la boca y el medidor.
Precondiciones	1. Debe existir el medidor de la boca a editar. 2. La unidad debe existir. 3. No puede existir otra boca con el mismo código para el mismo centro y almacén.

- Caso de uso nº 44:

Nombre	Edición de materia prima
Descripción	Permite editar los datos de una materia prima.
Precondiciones	1. El material debe existir. 2. El artículo debe existir. 3. No puede existir un material ya asignado al artículo.

- Caso de uso nº 45:

Nombre	Edición de centro de producción
Descripción	Permite editar los datos de un centro de producción.
Precondiciones	1. El centro debe existir. 2. No puede haber ningún almacén con el mismo nombre en el mismo nivel. 3. El código debe ser único.

- Caso de uso nº 46:

Nombre	Edición de juego de fabricación
Descripción	Permite editar los datos de un juego de fabricación.
Precondiciones	1. El juego debe existir. 2. El código de centro no puede ser null. 3. El código de juego no puede ser null. 4. El código debe ser único.

- Caso de uso nº 47:

Nombre	Edición de medidor
Descripción	Permite editar los datos de un medidor.
Precondiciones	1. El medidor debe existir. 2. No puede existir un medidor con el mismo código de dosificación, m2 y centro. 3. Debe existir la máquina a la que pertenecerá el medidor. 4. La unidad debe existir.

- Caso de uso nº 48:

Nombre	Modificación de la disciplina de gasto por defecto
Descripción	Permite establecer la disciplina de gasto que tendrá un almacén: FIFO, LIFO o fecha de caducidad.
Precondiciones	1. El almacén debe existir. 2. La disciplina debe ser válida.

- Caso de uso nº 49:

Nombre	Edición de planta
Descripción	Permite editar los datos de una planta.
Precondiciones	1. La planta debe existir. 2. El nombre de la planta no puede ser null o vacío. 3. El código externo no puede ser null. 4. No puede existir ninguna planta o almacén en el mismo nivel con el mismo código.

- Caso de uso nº 50:

Nombre	Edición de fabricación
Descripción	Permite editar los datos de una fabricación.
Precondiciones	1. La fabricación debe existir. 2. Si se indica almacén de destino este debe existir, la unidad no puede ser null y el material destino no puede ser null. 3. Si se indica material de destino este debe existir, el almacén destino no puede ser null y la unidad no puede ser null. 4. Si se indica unidad, el almacén y material destino no pueden ser null.

- Caso de uso nº 51:

Nombre	Fin de fabricación manual
Descripción	Permite hacer un acumulado de consumos de una fabricación manual, en el caso de ser una fabricación en la que se indica un material y almacén de destino, realizará una entrada automáticamente con la cantidad acumulada de los consumos manuales de la fabricación al almacén de destino, cambiando el material del almacén destino si no corresponde con el fabricado.
Precondiciones	Sin precondiciones.

- Caso de uso nº 52:

Nombre	Evaluación meteorológica
Descripción	Permite recibir del sistema la última evaluación meteorológica registrada.
Precondiciones	1. La estación debe existir.

- Caso de uso nº 53:

Nombre	Poner acopio en reposo
Descripción	Permite poner un acopio en reposo.
Precondiciones	1. El m2 debe ser distinto de null y de vacío. 2. El acopio para el m2 debe existir.

- Caso de uso nº 54:

Nombre	Entrada de material
Descripción	Permite realizar una entrada de material.
Precondiciones	1. La cantidad no puede ser menor o igual a cero. 2. La unidad debe existir. 3. El material debe existir. 4. El almacén debe existir. 5. El almacén no puede estar vinculado.

- Caso de uso nº 55:

Nombre	Modificación de la disciplina de gasto por almacén y material
Descripción	Permite establecer la disciplina de gasto que tendrá un almacén para un material: FIFO, LIFO o fecha de caducidad.
Precondiciones	1. El almacén debe existir. 2. El material debe existir. 3. La disciplina debe ser válida.

- Caso de uso nº 56:

Nombre	Poner acopio en uso
Descripción	Permite poner el acopio en marcha.
Precondiciones	1. La estación debe existir. 2. El almacén de origen debe existir 3. El almacén destino debe existir. 4. Los almacenes origen y el destino no pueden ser el mismo. 5. El acopio debe tener un almacén de origen y otro de destino. 6. El acopio debe estar en estado de reposo.

- Caso de uso nº 57:

Nombre	Quitar destino a un acopio
Descripción	Permite quitar un almacén como destino de un acopio.
Precondiciones	1. El nombre del acopio debe ser distinto de null y de vacío. 2. El nombre del m2 debe ser distinto de null y de vacío. 3. El acopio debe existir. 4. El almacén de destino debe existir.

- Caso de uso nº 58:

Nombre	Quitar origen a un acopio
Descripción	Permite quitar un almacén como origen de un acopio.
Precondiciones	1. El nombre del acopio debe ser distinto de null y de vacío. 2. El nombre del m2 debe ser distinto de null y de vacío. 3. El acopio debe existir. 4. El almacén de origen debe existir.

- Caso de uso nº 59:

Nombre	Regularización
Descripción	Permite realizar una regularización de stock en un almacén.
Precondiciones	1. La cantidad no puede ser menor o igual a cero. 2. La unidad debe ser existir. 3. El almacén debe existir. 4. El material debe existir. 5. Debe existir un stock para el almacén y material indicados. 6. El almacén no puede estar vinculado.

- Caso de uso nº 60:

Nombre	Salida de material
Descripción	Permite registrar una salida de material de un almacén.
Precondiciones	1. La cantidad no puede ser menor o igual a cero. 2. La unidad debe existir. 3. El material debe existir. 4. El almacén debe existir.

- Caso de uso nº 61:

Nombre	Salida de consigna
Descripción	Permite confirmar la salida de una consigna.
Precondiciones	1. La cantidad no puede ser menor o igual a cero. 2. La unidad debe existir. 3. La salida consignada debe existir. 4. La unidad debe ser compatible con la unidad de la consigna.

- Caso de uso nº 62:

Nombre	Transmutación
Descripción	Permite realizar una transmutación de material en un almacén.
Precondiciones	1. El almacén debe existir. 2. El almacén no debe estar vinculado. 3. El almacén debe tener stock activo. 4. El stock no puede ser menor o igual a cero.

- Caso de uso nº 63:

Nombre	Set habilitada
Descripción	Permite establecer como habilitada o deshabilitada una boca de un almacén.
Precondiciones	1. El almacén debe existir. 2. El nombre del m2 debe ser distinto de null y de vacío. 3. El almacén debe ser exclusivo. 4. La boca para el almacén y m2 indicados debe existir.

- Caso de uso nº 64:

Nombre	Vincular almacén
Descripción	Permite vincular un almacén con otro que ejercerá de almacén delegado, de tal forma que cualquier actividad sobre los lotes del almacén vinculado serán tratados en el almacén delegado.
Precondiciones	1. El almacén a vincular y el delegado no pueden ser el mismo. 2. El almacén a vincular debe existir. 3. El almacén delegado debe existir. 4. El almacén a vincular no puede estar ya vinculado. 5. El almacén a vincular debe ser exclusivo. 6. El almacén a vincular no puede pertenecer a un acopio.

- Caso de uso nº 65:

Nombre	Traspaso de material
Descripción	Permite realizar un traspaso de material de un almacén a otro.
Precondiciones	1. El material debe existir. 2. El almacén origen debe existir. 3. El almacén destino debe existir. 4. El almacén de destino no puede estar vinculado. 5. Los almacenes origen y destino deben tener stock activo del mismo material.

Casos de uso CallBack de actualización y consulta

- Caso de uso nº 66:

Nombre	Existe almacén
Descripción	Permite saber si un almacén existe.
Precondiciones	1. Sin precondiciones.

- Caso de uso nº 67:

Nombre	Existe artículo
Descripción	Permite saber si un artículo existe.
Precondiciones	Sin precondiciones.

- Caso de uso nº 68:

Nombre	Existe boca
Descripción	Permite saber si una boca existe.
Precondiciones	Sin precondiciones.

- Caso de uso nº 69:

Nombre	Existe centro de producción
Descripción	Permite saber si un centro de producción existe.
Precondiciones	Sin precondiciones.

- Caso de uso nº 70:

Nombre	Existe juego de fabricación
Descripción	Permite saber si un juego de fabricación existe.
Precondiciones	Sin precondiciones.

- Caso de uso nº 71:

Nombre	Existe comprable
Descripción	Permite saber si un artículo tiene la faceta de comprable.
Precondiciones	Sin precondiciones.

- Caso de uso nº 72:

Nombre	Existe máquina
Descripción	Permite saber si una máquina existe.
Precondiciones	Sin precondiciones.

- Caso de uso nº 73:

Nombre	Existe materia prima
Descripción	Permite saber si una materia prima existe.
Precondiciones	Sin precondiciones.

- Caso de uso nº 74:

Nombre	Existe medidor
Descripción	Permite saber si un medidor existe.
Precondiciones	Sin precondiciones.

- Caso de uso nº 75:

Nombre	Existe planta
Descripción	Permite saber si una planta prima existe.
Precondiciones	Sin precondiciones.

- Caso de uso nº 76:

Nombre	Existe stock activo
Descripción	Permite saber si un almacén tiene stock activo para un material.
Precondiciones	1. El almacén existe. 2. El material existe.

- Caso de uso nº 77:

Nombre	Existe vendible
Descripción	Permite saber si un artículo tiene la faceta de vendible.
Precondiciones	Sin precondiciones.

- Caso de uso nº 78:

Nombre	Habilitar acopio
Descripción	Permite habilitar un acopio.
Precondiciones	1. El m2 debe ser distinto de null y de vacío. 2. El acopio debe existir.

Resto de casos de uso

- Caso de uso nº 79:

Nombre	Suscripción
Descripción	Permite suscribirse a un almacén con el objetivo de recibir notificaciones del cambio en el stock de todos los almacenes pertenecientes al él.
Precondiciones	Sin precondiciones.
Flujo estándar	1. El cliente envía petición de suscripción a un almacén. 2. El sistema registra al cliente como suscriptor del almacén.
Flujo alternativo	1. El cliente envía petición de suscripción a un almacén. 2. El sistema no puede realizar la suscripción y lanza una excepción controlada.

- Caso de uso nº 80:

Nombre	Petición de material
Descripción	Permite recibir información relativa al stock de un almacén dado el centro, m2 y código de boca.
Precondiciones	Sin precondiciones.
Flujo estándar	1. El cliente envía petición de material para un centro, m2 y boca. 2. El sistema devuelve al cliente información del estado habilitado/deshabilitado de la boca del almacén. 3. El sistema devuelve al cliente información del material asignado al almacén correspondiente. 4. El sistema devuelve al cliente información del nivel actual del stock del almacén.
Flujo alternativo	1. El cliente envía petición de suscripción a un almacén. 2. El sistema no puede realizar la suscripción.

- Caso de uso nº 81:

Nombre	Baja de suscripción
Descripción	Permite dar de baja la suscripción del cliente a un almacén.
Precondiciones	Sin precondiciones.
Flujo estándar	1. El cliente envía petición de baja como suscriptor de un almacén. 2. El sistema elimina al cliente como suscriptor del almacén.
Flujo alternativo	1. El cliente envía petición de baja como suscriptor de un almacén. 2. El sistema no puede realizar la baja de la suscripción y lanza una excepción controlada.

- Caso de uso nº 82:

Nombre	Informe de trazabilidad de entradas
Descripción	Permite obtener un informe de trazabilidad desde las entradas de material hacia sus salidas como producto final.
Precondiciones	Sin precondiciones.
Flujo estándar	1. El cliente pide el informe. 2. El sistema devuelve el informe relleno.
Flujo alternativo	1. El cliente pide el informe. 2. El sistema no puede general el informe. 3. El sistema informa al usuario del error.

- Caso de uso nº 83:

Nombre	Informe de trazabilidad de lotes de fabricación
Descripción	Permite obtener un informe de trazabilidad desde los lotes de fabricación hacia los lotes de entrada.
Precondiciones	Sin precondiciones.
Flujo estándar	1. El cliente pide el informe. 2. El sistema devuelve el informe relleno.
Flujo alternativo	1. El cliente pide el informe. 2. El sistema no puede general el informe. 3. El sistema informa al usuario del error.

3.1.1.2. Requisitos de interfaz

El usuario debe disponer de un interfaz gráfico a base de ventanas para realizar las tareas de gestión del centro de producción. Serán necesarias las siguientes ventanas:

- Alta y edición de artículos y materias primas.
- Alta y edición de centros de producción, plantas, almacenes exclusivos, almacenes agregados, máquinas, bocas, niveles y juegos.
- Asignación de bocas a juegos.
- Asignación de materiales.
- Asignación de orígenes y destinos a los acopios.
- Regularización de stocks.
- Vinculación y desvinculación de almacenes.
- Creación de fabricaciones.
- Entradas de material.
- Control de entradas para consultar y anular entradas.

3.1.2. Requisitos no funcionales

Los requisitos no funcionales son las restricciones de las funcionalidades que ofrece el sistema y que se aplicarán a su totalidad. Los requisitos no funcionales surgen de las necesidades que notifica el usuario sobre cómo debería funcionar el sistema, es decir, nos indican las cualidades del sistema.

3.1.2.1. Requisitos de rendimiento

El sistema deberá soportar una carga importante de datos, aunque esta carga dependerá del nivel de producción que tenga cada centro. El servidor debe trabajar de manera fluida, respetando en todo momento los recursos computacionales y de almacenamiento.

3.1.2.2. Requisitos de seguridad

El sistema contará con un protocolo de conexión conocido entre el servidor y sus clientes, el cual sólo validará usuarios pertenecientes a los perfiles autorizados en el DataServer.

3.1.2.3. Requisitos de fiabilidad

El sistema será el responsable de la información de producción, calidad y trazabilidad, por tanto debe ser fiable ante fallos en su ejecución. Realizará reintentos de procesos y será capaz de identificar fallos grabes en el sistema que pueden provocar el reinicio de alguno de los módulos de cara a dejar operativo de nuevo el sistema.

3.1.2.4. Requisitos de mantenibilidad

El sistema debe ser capaz de mantener de manera correcta, coherente y eficiente los datos que maneja. El sistema contará con un monitor de borrados que chequeará y borrará cada cierto tiempo los datos temporales generados en él.

3.1.2.5. Requisitos de disponibilidad

El sistema debe estar operativo el 100% del tiempo. Se podrán programar reinicios teniendo en cuenta si se encuentra en uso recientemente o no.

3.1.2.6. Requisitos de usabilidad

El sistema contará con un protocolo de comunicación que deben conocer los clientes. En el caso de contar con un Sistema de Gestión de Proin, S.L., esté aportará una interfaz usable e intuitiva para su utilización con el DataServer.

3.1.2.7. Requisitos de portabilidad

El sistema debe poder migrarse a distintos motores de base de datos, así como poder ejecutarse en distintos sistemas operativos.

3.1.2.8. Requisitos de estabilidad

El sistema deberá controlar situaciones inesperadas, siendo capaz de recuperarse a ellas.

3.1.2.9. Requisitos de escalabilidad

El sistema deberá estar diseñado para permitir de manera fácil, segura y estable, incrementos en su funcionalidad.

3.2. Diseño y arquitectura

El objetivo del diseño arquitectónico es desarrollar una estructura modular y representar las relaciones de control entre los módulos. El diseño de la arquitectura combina la

estructura del sistema y las estructuras de datos, definiendo interfaces que permiten el flujo de datos a través del sistema. Este proceso permite al desarrollador centrarse primero en las características generales del sistema y posteriormente en los detalles del mismo.

El diseño es el proceso por el que se determinan soluciones técnicas eficientes para implementar el análisis. En él debemos:

- Descomponer el sistema en subsistemas (diseño arquitectónico)
- Descomposición de cada subsistema en otros subsistemas
- Descomposición de cada subsistema en paquetes
- Descomponer cada paquete en clases
- Identificar nivel de concurrencia
- Decidir qué es software y qué es hardware

- Identificar recursos globales
- Identificar las condiciones de contorno del sistema
- Establecer prioridades de la implementación
- Definir la implementación de clases

3.2.1. Decisiones de diseño

En la programación orientada a objetos se entiende por patrón una solución probada que puede ser aplicada con éxito a determinados tipos de de problemas que aparecen de forma repetida en el desarrollo software de sistemas.

Para el desarrollo del sistema no nos basaremos en patrones de diseño [38], sin embargo, sí los usaremos para el diseño de aquellas partes en las que se determina de forma clara que pueden aplicarse. En este sentido los principales patrones utilizados son:

- Factory Method [39]: define un interfaz para crear un objeto, permitiendo además a las subclases decidir qué clase instanciar. Permite a las clases crear una instancia diferente para cada una de las subclases.

- Observer [40]: define dependencias entre objetos para que, cuando se experimente un cambio de estado en uno de ellos, el resto pueda enterarse de dicho cambio.
- Proxy [41]: suministra un intermediario de un objeto para controlar su acceso.
- Iterator [42]: aporta un modo de acceder a los elementos de una colección de objetos de forma secuencial.
- Adapter [43]: convierte el interfaz de una clase a otro interfaz, permitiendo a varias clases trabajar juntas, cuando de otro modo, por incompatibilidad de los interfaces, no podrían.
- Command [44]: encapsula una petición como un objeto, permitiendo gestionar colas o registro de mensajes y deshacer operaciones.

Debemos tener en cuenta en el diseño que trabajamos con un modelo de objetos, sin embargo, por requerimientos de administración de la base de datos, la capa de persistencia debe trabajar con una base de datos relacional y no orientada a objetos.

Aunque el sistema tiene un control distribuido, permitiendo acceder a sus servicios desde distintas ubicaciones y por distintos clientes, cuenta con una restricción que debemos tener en cuenta en el diseño, ésta es que los comandos deben ejecutarse en orden de llegada.

3.2.2. Condiciones de contorno

Ante el desarrollo de un sistema debemos analizar con detalle las condiciones de contorno del mismo, esto es, las características que el sistema debe cumplir en su inicialización, su finalización, y ante situaciones anómalas.

Con el fin de controlar las condiciones de contorno se ha desarrollado un módulo de inicialización del sistema llamado Main.

El módulo Main permite arrancar los distintos módulos del sistema:

- ProinModel
- Servidor
- AtendedoresM2
- Generador de informes

- Monitores

Hablaremos de todos ellos con detalle en el punto 3.2.3. Podemos indicar al módulo Main los distintos módulos y funcionalidades que debe iniciar, para ello se utiliza un fichero de configuración o propiedades del sistema. En él podemos indicar si queremos iniciar todo el sistema, si queremos iniciar sólo el generador de informes o si queremos iniciar todo el sistema pero sin el servidor de informes. Para ello el módulo Main cuenta con una factoría de arrancadores, denominada AbstractStarter, de forma que si necesitamos en un futuro añadir nuevas maneras de arrancar el sistema, podemos añadir de manera fácil una nueva clase de arranque en la factoría.

Resulta importante determinar cómo debe finalizar o cerrarse el sistema, así como tener claro qué debe hacer el sistema ante situaciones inesperadas. El sistema arranca cada módulo en un hilo independiente, por tanto para controlar el estado y ciclo de vida de cada uno de ellos estos quedan registrados, pudiendo así el sistema detenerlos en el orden apropiado. Los hilos hijos quedan apuntados a sus padres y viceversa, por lo cual cuando un hilo hijo finaliza (correcta o incorrectamente), el padre será consciente y podrá decidir si debe iniciarlo de nuevo o por el contrario debe finalizarse él también.

Dado que el sistema dispone de una tarea que permite la finalización del sistema de forma programada, el sistema debe saber cerrar los distintos módulos e hilos en el orden adecuado, es por ello que el disponer de una lista de los hilos disponible, y que los hilos padre entiendan de los hilos hijo y viceversa, facilita esta tarea, ya que un hilo puede ir indicando a sus hijos que se terminen de manera correcta, y este puede terminar una vez hayan finalizado todos sus hilos hijo.

3.2.3. Arquitectura

La arquitectura del software representa el diseño de más alto nivel de la estructura de un sistema. Corresponde con una vista o perspectiva que nos permite tener una visión global del sistema en todo momento. La arquitectura define, de manera abstracta, los componentes del sistema, sus interfaces y la comunicación entre ellos.

El sistema aquí tratado está basado en el patrón cliente/servidor. En la ilustración 10 se muestra de manera simbólica la arquitectura cliente/servidor aplicada a nuestro sistema.

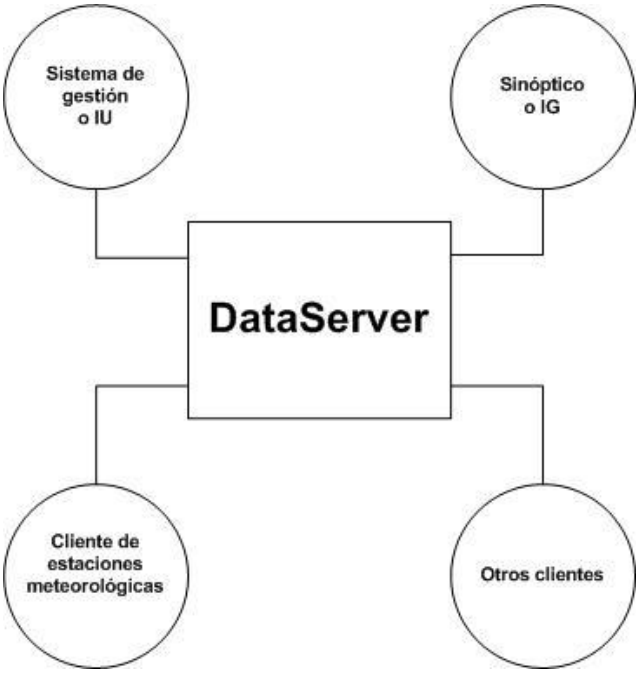


Ilustración 10. Arquitectura cliente/servidor

Para poder entender de manera rápida y clara la estructura de la arquitectura del sistema DataServer, en la ilustración 11se puede observar la arquitectura horizontal del sistema. Dentro de esta arquitectura podemos identificar los distintos componentes, así como la relación entre cada uno de ellos. En este punto veremos con detalle cada uno de los componentes que han sido desarrollados, dejando de lado las plataformas o componentes en los que se apoya el sistema como son:

- Sistema operativo.
- Java SDK 6. Esta tecnología ha sido descrita en el punto 2.2.
- JasperReport. Esta tecnología ha sido descrita en el punto 2.2.
- JBossRules. Esta tecnología ha sido descrita en el punto 2.2.
- Persistencia JDO. Aunque supone un punto importante dentro de la arquitectura, queda fuera de los límites del proyecto, dado que únicamente nos apoyamos en esta tecnología a través del API JDO que ha sido descrito en el punto 2.2. En el recurso Base de datos del CD adjunto a la memoria puede consultarse el script de la base de datos manejada por el sistema.

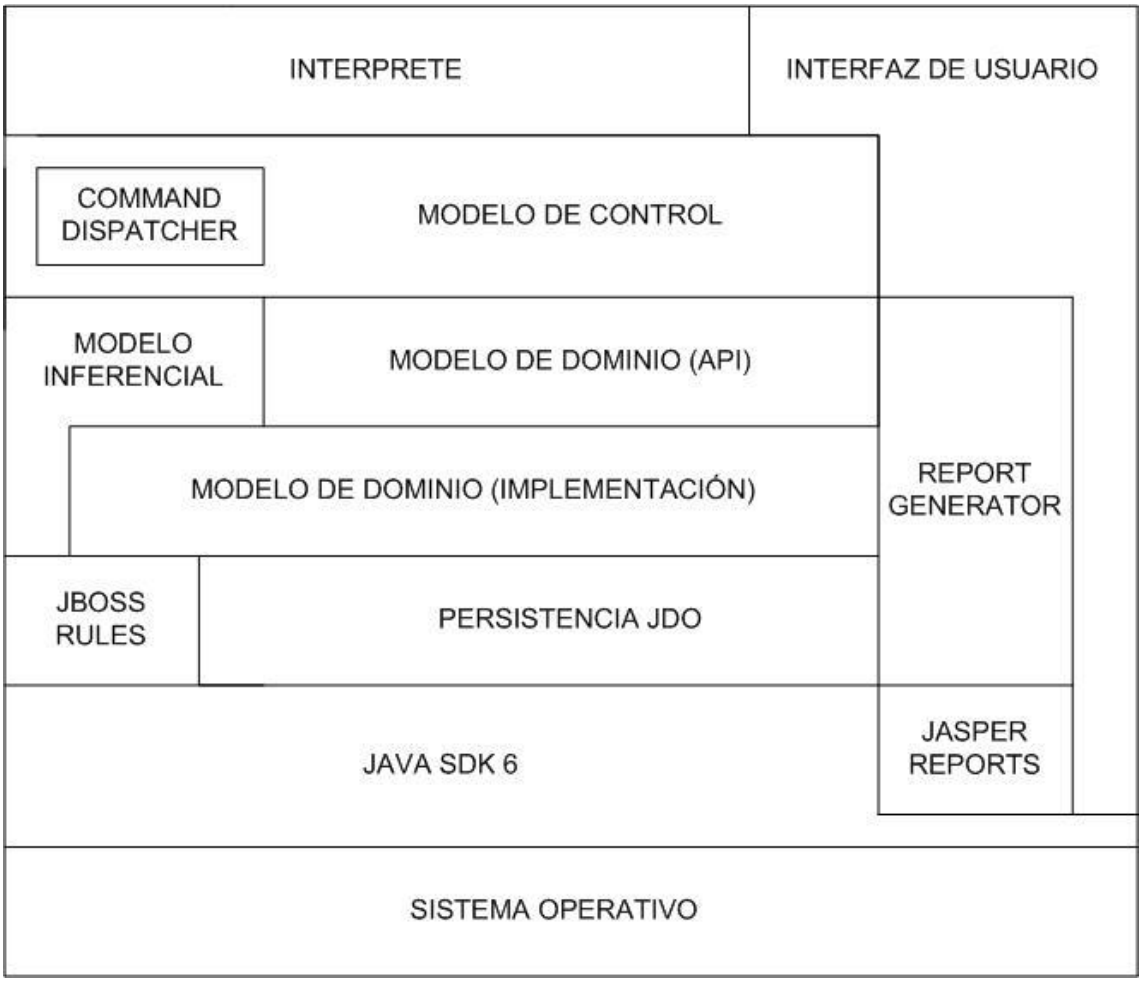


Ilustración 11. Arquitectura del sistema

Modelo del dominio

El modelo de dominio ha sido desarrollado con una fachada o API y con una implementación. Es a través del API como los distintos subsistemas podrán interactuar con la capa de dominio. La implementación será la encargada de manejar los objetos del dominio y quién comunicará con la capa de persistencia JDO.

En la ilustración 12 se puede ver el diagrama de paquetes del modelo del dominio, así como las dependencias entre ellos.

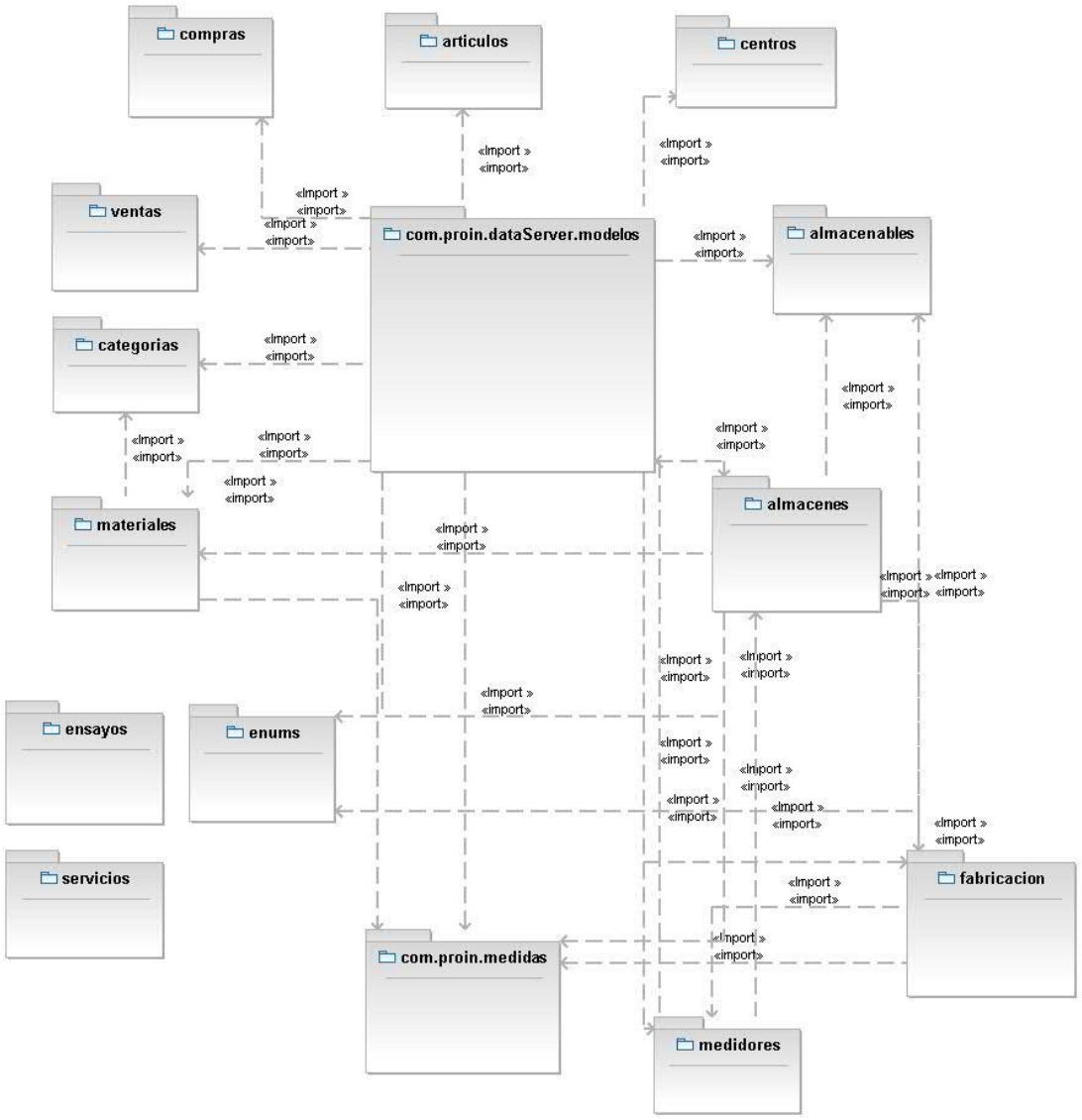


Ilustración 12. Diagrama de paquetes del modelo de dominio

Paquete principal de modelos

En este paquete se encuentran modelados los distintos objetos que manejará el sistema: almacenes, artículos, lotes, stocks, etc.

Paquete de almacenes

Este paquete contiene el modelado de los objetos relacionados de manera directa con los almacenes. Podemos dividirlo en:

- Órdenes y movimientos: hace referencia a los distintos movimientos que pueden sufrir los materiales en su ciclo de vida dentro de un almacén. Es aquí donde se realiza el modelado de las entradas, salidas, regularizaciones, traspasos y transmutaciones.
- Almacenes: hace referencia al modelado de los distintos tipos de almacenes disponibles en el sistema, así como la relación entre ellos. Los dos tipos principales con los almacenes agregados y los almacenes exclusivos. Se han modelado también dos tipos de almacén exclusivo que son la cisterna y el camión, sin embargo, actualmente no se trabaja con ellos ya que consideramos que están fuera del entorno controlado por el sistema, pero no se descarga una futura necesidad de ampliar el entorno añadiendo el control de este tipo de almacenes.
- Lotes y stocks: hace referencia al modelado de los lotes, cola de lotes y stocks. Recordemos que los stocks, dependiendo del tipo de almacén, puede tener cero, uno o más stocks. Los stock están formados por colas de lote, y los lotes pueden ser de entrada, de fabricación o virtuales. Como ya hemos hablado anteriormente, para el gasto de stock es necesario conocer la disciplina de gasto, es por ello que se necesita modelar un objeto de disciplina, el cual entiende de la estrategia de consumo a aplicar en los stocks.
- Acopios: hace referencia a los objetos modelados para representar y manejar los acopios de material. Dado que el sistema debe ser capaz de poner un acopio en marcha, este debe comunicar con el Conex M2 y enviarle la activación de las señales de control correspondientes al acopio a poner en marcha (origen y destino), es por ello que necesitemos modelar los objetos semáforo que corresponden con las señales de control del Conex M2.

En la ilustración 13 podemos ver el diagrama de clases del paquete de almacenes.

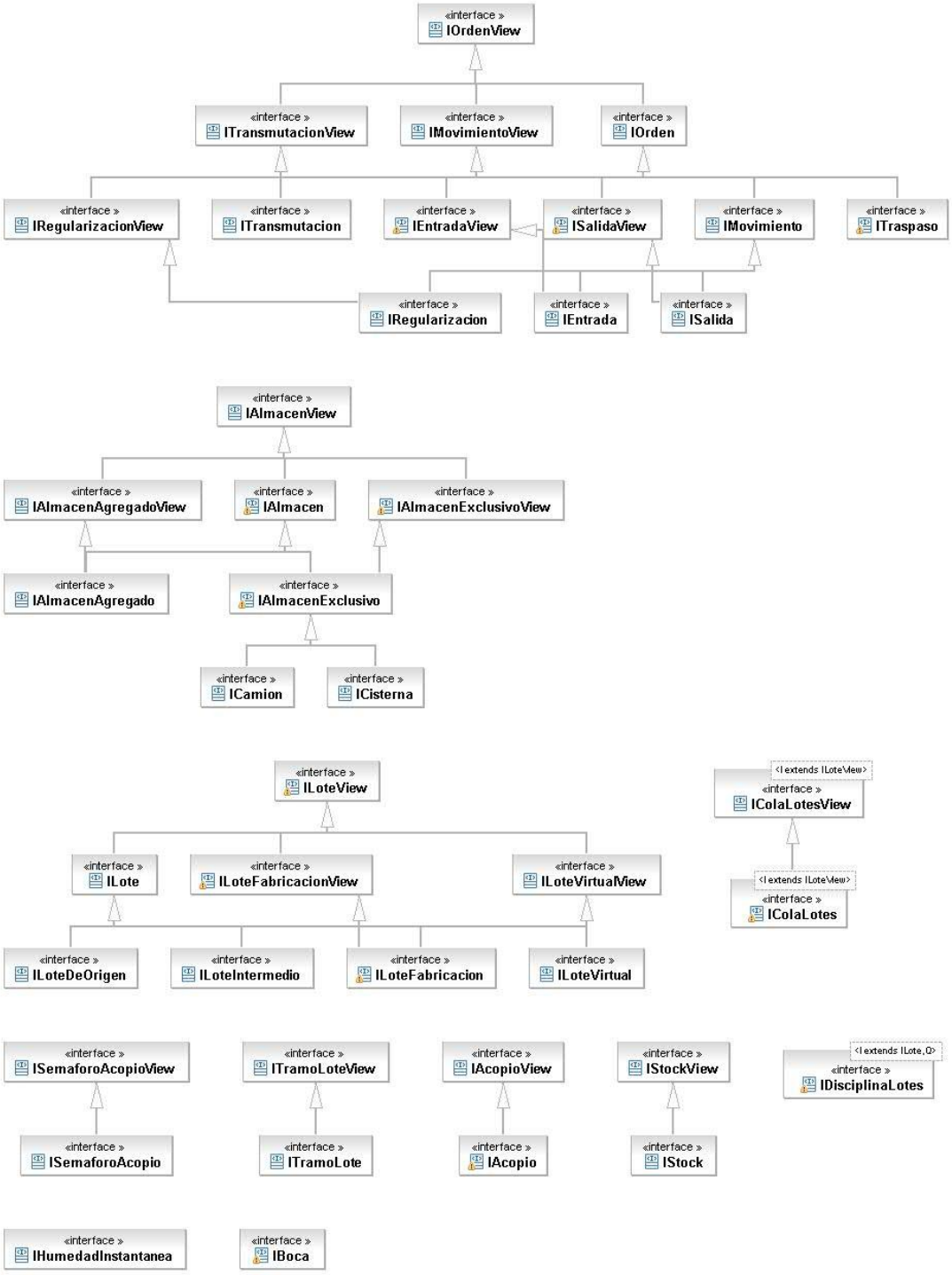


Ilustración 13. Diagrama de clases del paquete almacenes

Paquete de artículos

Este paquete contiene el modelado de los artículos y los objetos relacionados con él. Los artículos se pueden ver de distinta manera según quien los necesite, es por ello que se han modelado distintas facetas a los que los artículos se pueden suscribir: almacenable, comprable, vendible, servicio, ensayo y categoría. Los materiales técnicos representan las características técnicas que un artículo puede tener.

En la ilustración 14 podemos ver el diagrama de clases del paquete de artículos.

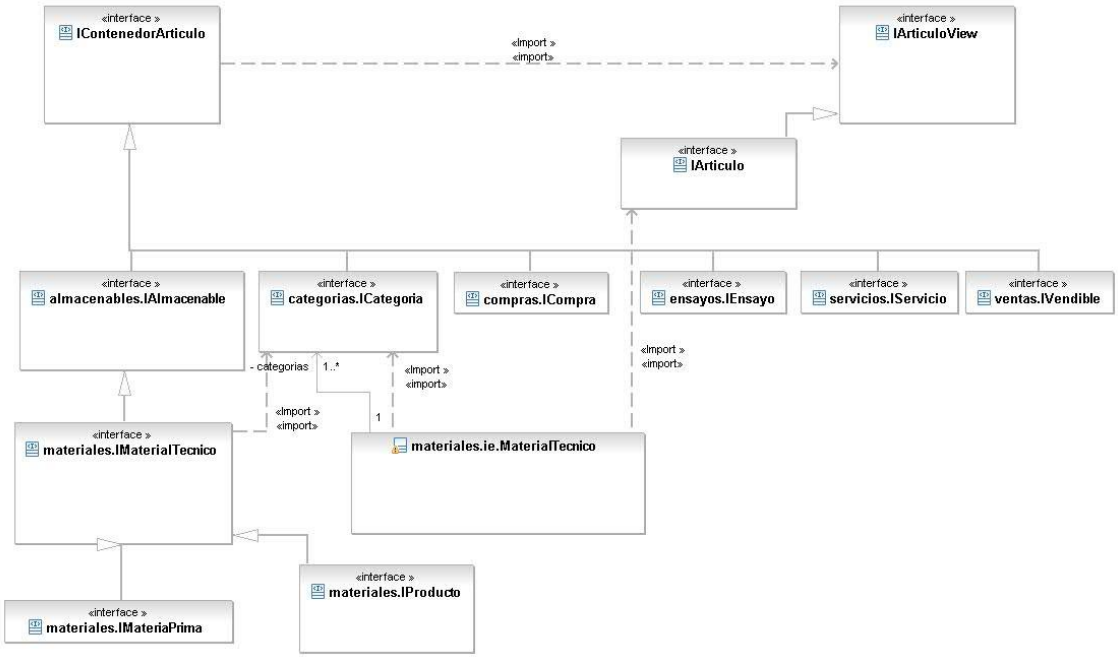


Ilustración 14. Diagrama de clases del paquete artículos

Paquete de centros

Este paquete contiene el modelado de los centros de producción y las plantas que este puede contener. Recordemos que los centros de producción y las plantas corresponden a almacenes agregados y que el centro de producción es el almacén principal a controlar.

En la ilustración 15 se puede ver el diagrama de clases del paquete centros.

Ilustración 15. Diagrama de clases del paquete centros

Paquete de fabricación

El paquete de fabricación contiene el modelado de la fabricación y los objetos ligados a ella. Cuando el sistema recibe un mensaje de consumo, este genera un apunte con su consumo y medida, apuntando a la fabricación de la que proviene. De igual forma, el sistema puede recibir mensajes de fabricación, que aunque no representan ninguna medida, informan de algo concreto que el sistema debe registrar. Dado que hay fabricaciones con destino a un almacén de producto terminado, debemos registrar, ante un mensaje de fin de ciclo, la entrada de material al silo de destino cuya cantidad corresponde con el acumulado de consumos de la fabricación para el ciclo indicado.

En la ilustración 16 se puede ver el diagrama de clases del paquete fabricación.

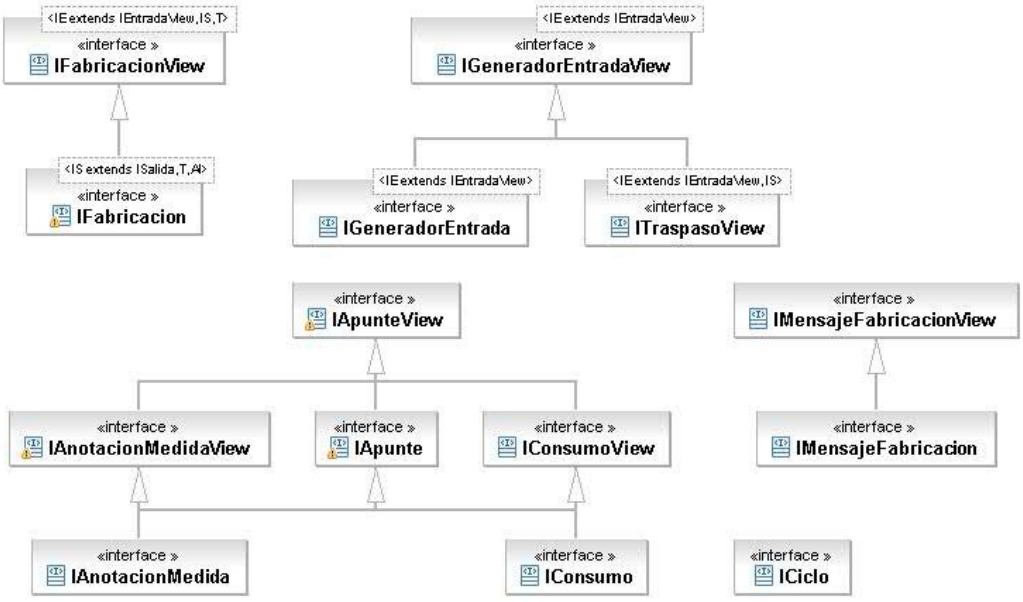


Ilustración 16. Diagrama de clases del paquete fabricación

Paquete de medidores

El paquete de medidores contiene el modelado de los objetos máquinas, bocas, medidores, juegos de fabricación y topologías.

En la ilustración 17 se puede ver el diagrama de clases del paquete medidores.

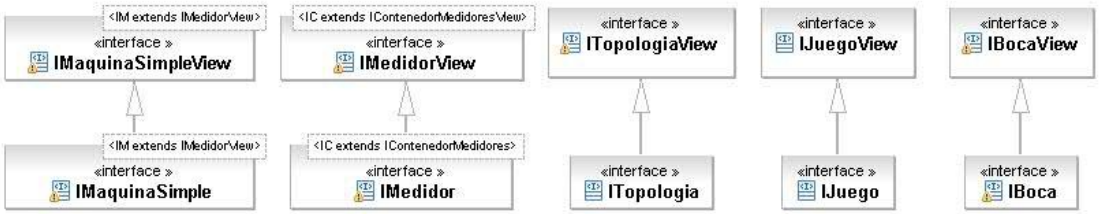


Ilustración 17. Diagrama de clases del paquete medidores

Paquete de medidas

Este paquete surge de la necesidad de disponer de un sistema de unidades para anotar cada una de las medidas que va generando el sistema (consumos, salidas,…). Este sistema permite representar las cantidades con su unidad. La unidad pertenece a una magnitud (masa, volumen,…), pudiendo crear fórmulas de conversión entre magnitudes, de tal forma que seamos capaces de realizar conversiones de unidad y de magnitud. El sistema permite añadir nuevas unidades sin necesidad de generar una nueva distribución.

En la ilustración 18 se puede ver el diagrama de clases del paquete de medidas.

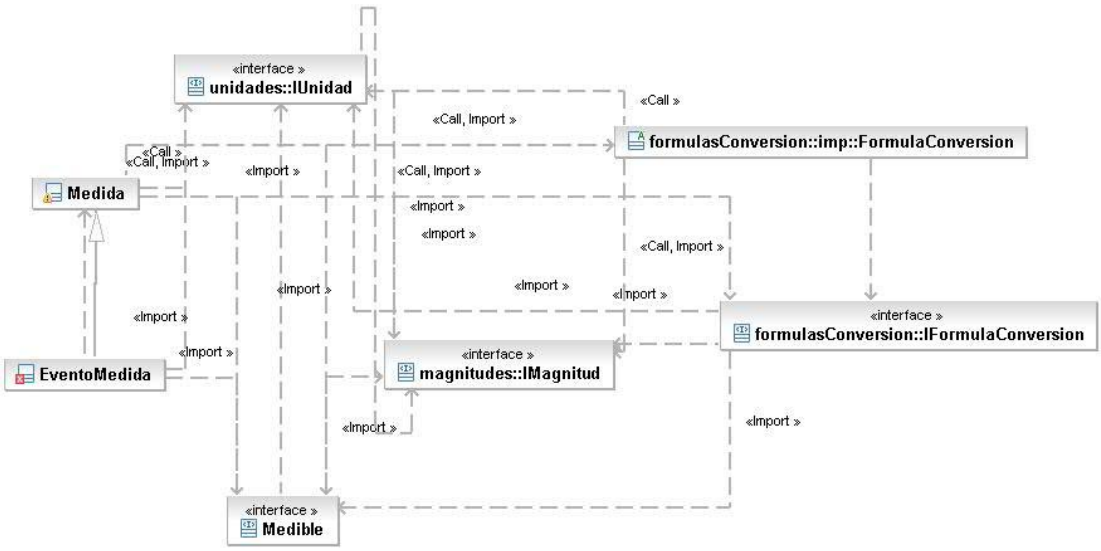


Ilustración 18. Diagrama de clases del paquete medidas

Paquete de meteorología

Este paquete modela los objetos que representan a los datos meteorológicos. Será a través de estos objetos como el sistema realizará las evaluaciones meteorológicas gracias al motor de reglas.

En la ilustración 19 se puede ver el diagrama de clases del paquete meteorología.

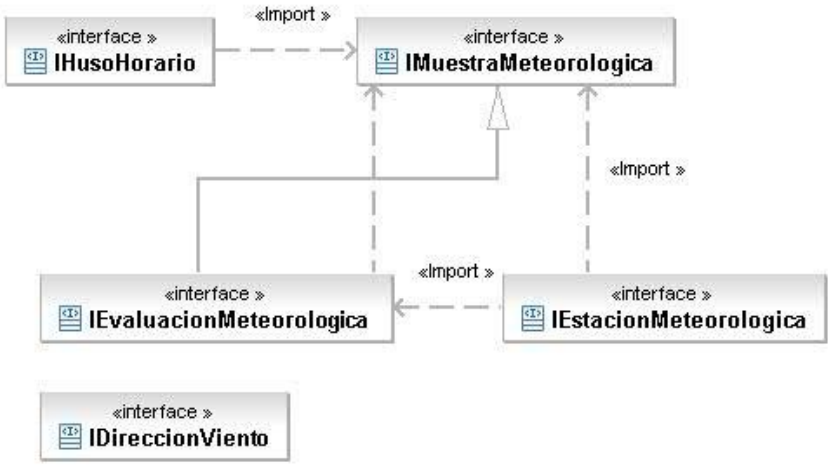


Ilustración 19. Diagrama de clases del paquete meteorología

Modelo de control

El modelo de control está compuesto de componentes relacionados entre sí, que permiten interactuar con el modelo de dominio a través del API de éste.

En la ilustración 20 podemos ver el diagrama de paquetes del módulo Servidor que corresponde con la capa del modelo de control.

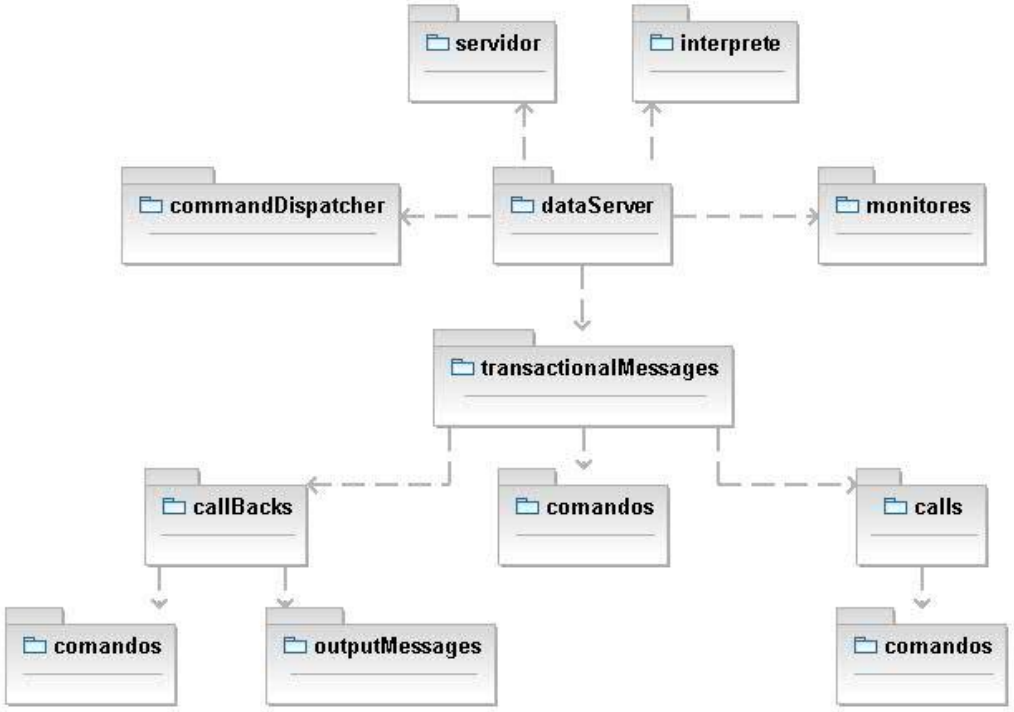


Ilustración 20. Diagrama de paquetes del modelo de control

Paquete principal dataServer

En este paquete se agrupan los distintos componentes del módulo Servidor.

Paquete servidor

En la ilustración 21 se puede ver el diagrama de clases del paquete servidor. Éste se encuentra compuesto de los siguientes componentes:

- Servidor: este componente se encarga de mantener abierto un servidor de conexiones o ServerSocket que acepte conexiones. Una vez aceptada una conexión cliente, el Servidor crea un InterlocutorM2 para tramitar la nueva conexión y continúa con su cometido de aceptar conexiones. El resto del trabajo lo realizará el InterlocutorM2.
- AtendedorM2: es mediante este componente con el que nos comunicamos con el Conex M2. Existe una peculiaridad en el sistema DataServer que es que aun siendo un servidor, necesita conectarse como cliente de un Conex M2, para así poder recibir los distintos consumos y mensajes de este, de cara a controlar los stocks y las fabricaciones. Para poder interactuar con el

sistema debemos conectarnos como cliente al DataServer, para así poder enviarle los distintos mensajes recibidos del Conex M2. Este módulo se encarga de mantener viva la conexión como cliente con el DataServer, en el caso de detectar un fallo en la conexión, es capaz de reconectarse. Dada la importancia de los datos que se reciben del Conex M2 el AtendedorM2 debe mantener en lo posible la conexión activa con el Conex M2, es por ello que se ejecuta en un hilo siempre activo, mientras no se notifique una condición de parada por parte del DataServer.

- InterlocutorM2DataServer: En ocasiones el DataServer debe enviar mensajes al M2 para que este ejecute ciertas tareas. Este es el caso la puesta en marcha de los acopios, en el cual el DataServer debe indicarle el almacén de origen y de destino. Para poder realizar esta tarea necesitamos un componente que sea capaz de enviar mensajes al Conex M2 siguiendo el protocolo de comunicación que este impone. El InterlocutorM2DataServer realiza esta tarea.
- ControladorM2: dado que la información que envía el Conex M2 al DataServer son datos de producción, los cuales permiten llevar un control de los stocks y por tanto de la trazabilidad, resulta imprescindible disponer de un componente que permita comprobar si la conexión con el Conex M2 sigue viva, ya que la pérdida de mensajes de fabricación de éste puede causar daños importantes en el trazabilidad, stocks y los datos de producción. El ControladorM2 se encarga de chequear que la conexión con el M2 sigue operativa. Su funcionamiento es el siguiente: el ControladorM2 chequea cada segundo si se ha recibido algún mensaje del M2, si no es así, envía el mensaje “00E” al Conex M2, para preguntarle si sigue vivo. El Conex M2, a partir de recibir el mensaje “00E”, enviará periódicamente un mensaje para confirmar que está vivo. En el caso de superar un tiempo máximo, configurado en el fichero de propiedades del sistema, el ControladorM2 le notificará a su AtendedorM2 que reinicie la conexión con su Conex M2.
- Interlocutor: este componente es el interlocutor entre el cliente y el DataServer. Se encarga de validar la conexión, mantiene un bucle para la recepción de mensajes del cliente, los cuales interpreta a través del módulo Intérprete, el cual devuelve el comando a ejecutar, y por último manda el comando al CommandDispatcher para que lo ejecute.

- Proxy: este componente se encarga de enviar las respuestas, que el DataServer genera como resultado de la ejecución de los comandos, a los clientes. Es a través de los proxies como el DataServer notifica los cambios en los elementos a los que el cliente se ha suscrito. El proxy es creado por el Interlocutor mediante una factoría, instanciando el proxy adecuado dependiendo del perfil de conexión del cliente. El proxy se crea una vez el Interlocutor ha validado la conexión con el cliente. En el caso de detectar varios fallos seguidos en el envío de mensajes de respuesta al cliente, el sistema elimina automáticamente el proxy con el cliente, entendiendo que la conexión ya no está activa.

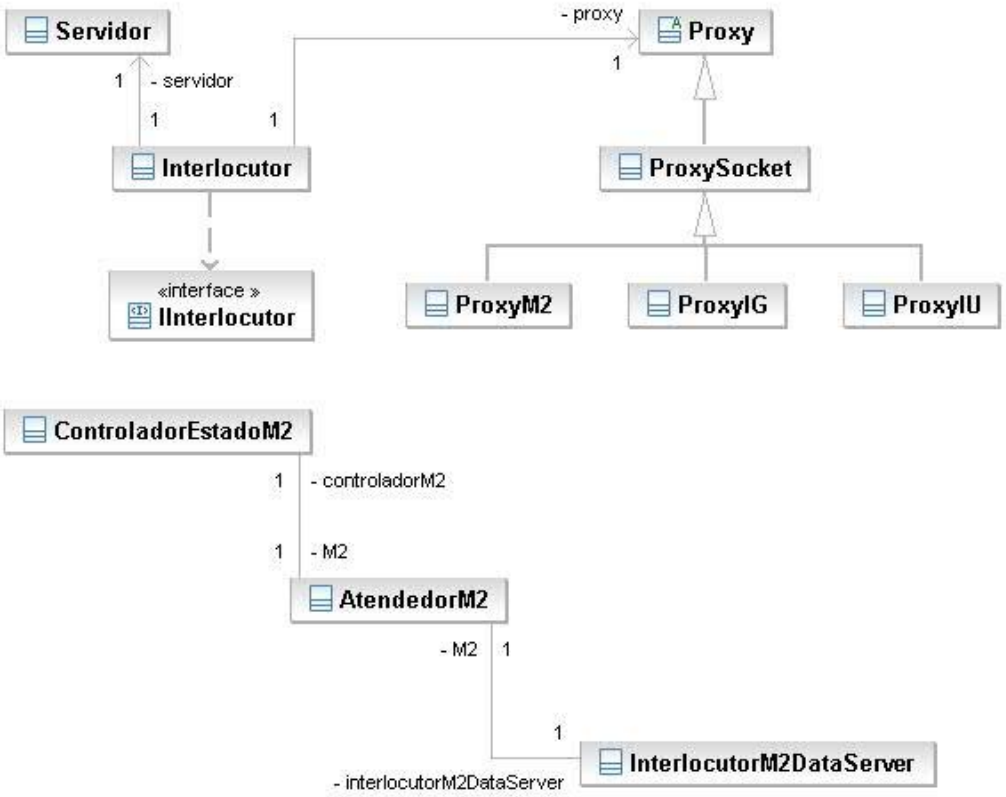


Ilustración 21. Diagrama de clases del paquete servidor

Paquete interprete

Este paquete contiene al Intérprete de comandos que se encarga de interpretar los mensajes de los clientes y de generar el comando apropiado para cada mensaje. Para realizar la interpretación se han usado expresiones regulares y un enumerado con los posibles mensajes que el Interprete es capaz de interpretar. En el Anexo 6.3 se puede consultar el enumerado con los distintos mensajes que es capaz de interpretar.

Paquete monitores

Los monitores permiten ejecutar tareas determinadas periódicamente. Se instancia mediante una factoría de monitores, y es en el fichero de propiedades del sistema donde se indica qué monitores arrancar y con qué configuración. Actualmente el sistema dispone de tres monitores:

- Monitor de acopio: permite comprobar cada n segundos si hay que notificar al Conex M2 la activación del sistema de acopio para un origen y un destino.
- Monitor de borrados: de cara a que el sistema se mantenga por sí solo, se ha generado una tarea de borrado de datos temporales, es por ello que dicho monitor se ejecuta periódicamente en busca de objetos temporales a borrar.
- Monitor de reboot: el sistema en ocasiones trabaja como servicio en servidores que no se apagan o reinician. De cara a que el sistema se mantenga por sí solo, se puede programar un reinicio del DataServer especificando hora del día a partir de la cual se puede reiniciar y condiciones de tiempo mínimo sin uso del sistema antes del reinicio, evitando así que el sistema se reinicie con carga de trabajo.

Paquete commandDispatcher

El CommandDispatcher trabaja con una cola de comandos que va procesando y ejecutando secuencialmente. Al crearse una instancia del CommandDispatcher, se crea un contexto JDO en modo transaccional. Los comandos, al ejecutarse, disponen de la capacidad de abrir y cerrar transacciones, así como de trabajar cómodamente con la capa de persistencia.

Paquete transactionMessages

Representan los distintos tipos de mensaje con los que trabaja el DataServer, cada mensaje se trata como una transacción, eso quiere decir que si un mensaje referente a la realización de un comando no se realiza correctamente, el sistema hará un rollback para dejar el estado del mismo en el estado previo a la ejecución del comando.

Existen dos tipos de transactionalMessages:

- CallBack: corresponde a mensajes transaccionales que deben generar una respuesta o confirmación para su envío al cliente.
- Call: corresponde a mensajes transaccionales sin respuesta.

En la ilustración 22 se puede ver el diagrama de colaboración del módulo Servidor. Mediante este diagrama podemos entender cómo el sistema trabaja e interactúa con el cliente para realizar una tarea concreta. El diagrama se ha realizado para el comando "Alta de artículo".

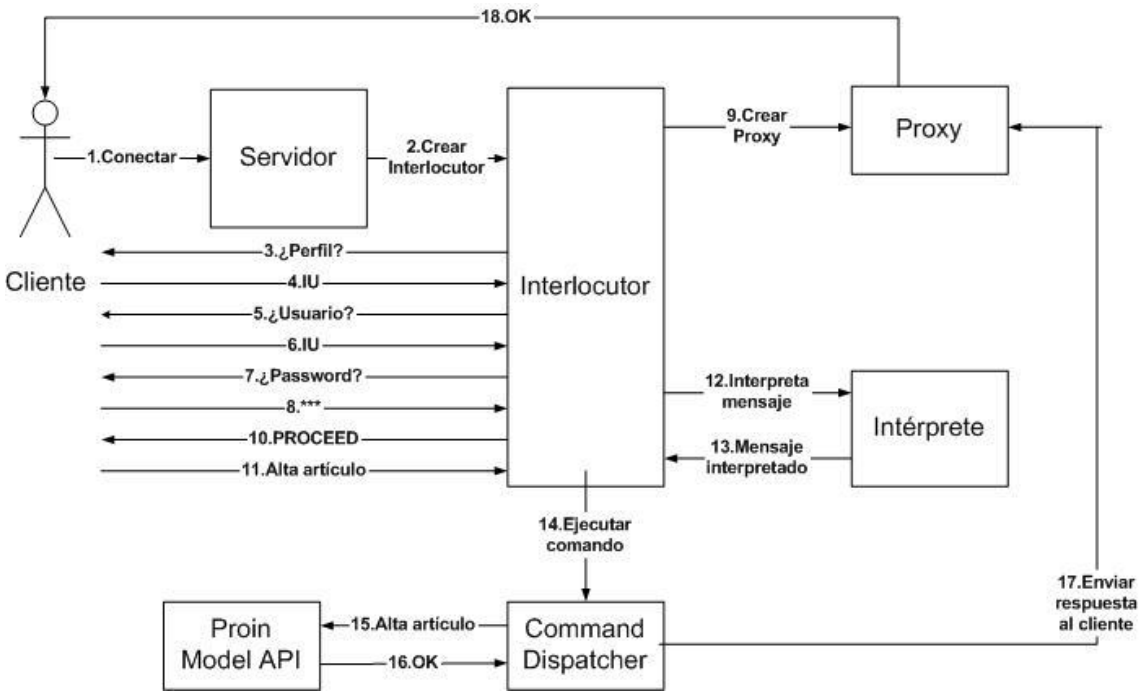


Ilustración 22. Diagrama de colaboración del módulo Servidor

Generador de informes

Debido a la necesidad de unir dos entornos distintos: DataServer y Conex, nos hemos visto en la necesidad de realizar un mapeo de la base de datos del sistema Conex mediante JPA, generando entidades JPA para cada uno de sus objetos. Para obtener los objetos del DataServer el generador de informes hace uso del API del modelo de dominio y mediante JPA es capaz de realizar el enlace entre los objetos Conex y los objetos DataServer.

El generador de informes en su inicialización levanta un servidor Jetty [22] que permite disponer de un portal web para la petición de informes desde el mismo.

Dada la necesidad de tener que obtener informes desde el sistema Conex se añaden al Jetty las interfaces RMI de los objetos a consultar en los informes, que proveerán los métodos a los que el cliente llamará para generar el informe deseado. Para poder realizar la llamada a los métodos de los objetos, es necesario que todos los objetos del generador de informes a los que se quiere acceder extiendan de java.rmi.Remote.

El sistema dispone de informes de trazabilidad (hacia delante y hacia atrás) realizados con la herramienta iReport [27]. La ruta donde se ubican queda determinada en el fichero de propiedades del sistema.

En la ilustración 23 se puede ver el diagrama de paquetes del módulo Generador de informes.

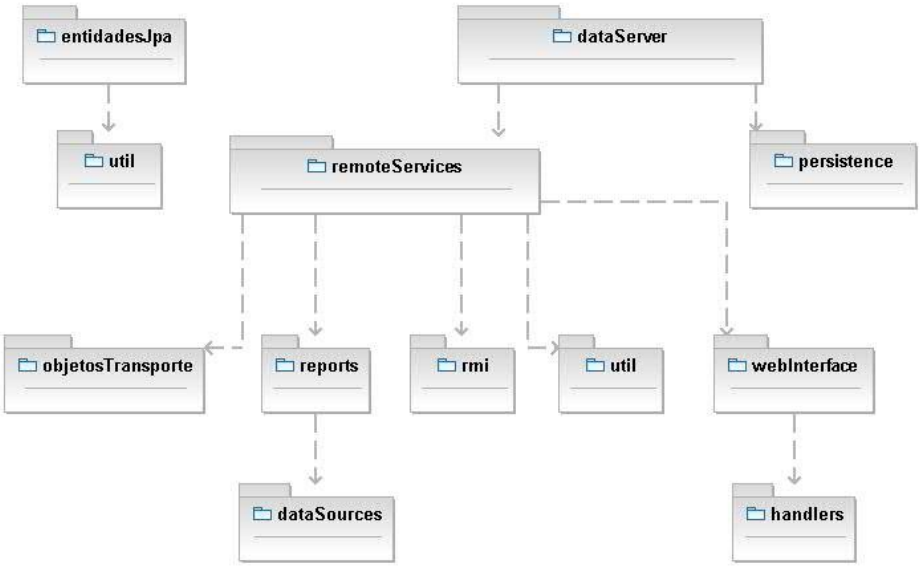


Ilustración 23. Diagrama de paquetes del Generador de informes

Modelo inferencial

Este subsistema corresponde con el conjunto de reglas de las que dispone el sistema para realizar evaluaciones del mismo. Actualmente se dispone de un paquete de reglas para realizar evaluaciones meteorológicas. El funcionamiento es el siguiente: el sistema, tras recibir la petición de realizar una evaluación, apoyándose en el motor de reglas JBossRules, cargado con las reglas de nuestro modelo inferencial, será capaz, gracias a los objetos que le suministra el modelo de dominio, de realizar la evaluación meteorológica, anotando en la base de datos y devolviendo al cliente el resultado de la misma.

Interfaz de usuario

Este componente hace referencia a la interfaz de ventanas, el Sistema de gestión o IU, del cual se ha hablado en el punto 1.4.

3.3. Pruebas

Realizar pruebas que determinen si el sistema cumple con los requerimientos especificados resulta de gran utilidad a la hora de preparar nuevas distribuciones, ya que el hecho de que un sistema pase la batería de pruebas indica que el sistema es fiable y que la probabilidad de fallo será muy baja, claro está, siempre y cuando la batería de pruebas abarque todos o la gran mayoría de los requisitos.

Para la realización de las pruebas unitarias se ha levantado el modelo de datos en una base de datos en memoria (HSQLDB [17]) y se ha utilizado el módulo o plug-in de Eclipse JUnit [30]. Las pruebas realizadas han intentado abarcar todas las posibles situaciones, de cara a evaluar de la forma más completa la correcta funcionalidad del sistema.

Se han realizado dos tipos de pruebas unitarias:

- Pruebas del protocolo de comunicación entre el cliente y el sistema: estas pruebas comprueban la correcta interpretación del mensaje enviado por el cliente, la creación del comando correspondiente y la correcta generación del mensaje de respuesta al cliente. Se han realizado un total de 90 test unitarios. En

el Anexo 6.4.1. se puede consultar el informe de ejecución de la batería de pruebas del protocolo de comunicación entre el cliente y el sistema.

- Pruebas de funcionalidad del sistema en base a los requisitos funcionales de usuario correspondientes a los casos de uso del número 1 al 87: estas pruebas permiten comprobar la correcta funcionalidad del sistema para cada uno de los requisitos de usuario especificados. Se han realizado un total de 316 test unitarios. En el Anexo 6.4.2 puede consultarse el informe de ejecución de la batería de pruebas de funcionalidad del sistema.

En referencia a los requisitos de interfaz, se han realizado pruebas del interfaz de usuario tanto por los técnicos de Proin, S.L. como por los propios clientes, evaluando y validado el correcto funcionamiento del mismo.

Es importante realizar pruebas de comportamiento del sistema en los posibles entornos de trabajo y bajo situaciones anormales de cara a cumplir con los requisitos no funcionales. En este sentido, para el sistema aquí tratado se han realizado pruebas de rendimiento, carga, fiabilidad y estabilidad ante fallos y situaciones anómalas.

4. CONCLUSIONES Y TRABAJOS FUTUROS

El objetivo principal del proyecto consistía en desarrollar un sistema capaz de controlar el entorno de un centro de producción. Como se ha explicado a lo largo de este documento, se ha creado una arquitectura y modelo de datos específico para el entorno del centro de producción, al mismo tiempo que se han facilitado mecanismos para la interacción de los clientes con el sistema, de cara a crear y mantener los centros de producción. Este objetivo ha sido cumplido siguiendo las especificaciones marcadas, y desarrollando pruebas específicas que confirman el correcto funcionamiento del sistema.

Otro de los objetivos marcados fue la capacidad del sistema de controlar la trazabilidad de los productos almacenados y fabricados. En este sentido, gracias al modelo implementado el sistema registra en todo momento la traza de los materiales que en él residen, por tanto ya sólo es necesario saber interpretar y buscar la información de trazabilidad que necesitamos en cada momento. Para ello contamos con la ayuda del Generador de informes, que apoyándose en el API, es capaz de mostrar al usuario informes de trazabilidad. El Generador de informes corresponde con otro de los objetivos marcados en el inicio del proyecto y que ha sido cumplido.

El sistema cumple con otros tres de los objetivos y requisitos marcados, que son que el sistema debe ser robusto, fiable y escalable. En este sentido, gracias al uso de patrones en el diseño, el sistema generado resulta más robusto y escalable. La generación de las pruebas realizadas ayuda a validar el sistema ante el desarrollo de nuevas funcionalidades, haciendo este más fiable. Como se indica en la planificación, el desarrollo se ha realizado en 3 iteraciones, en las cuales se han ido incrementando los objetos y las funcionalidades del sistema. Para las distintas iteraciones incrementales realizadas, el sistema se ha adaptado correctamente, esto se debe a la correcta elección de la arquitectura que resulta idónea para la escalabilidad del DataServer.

Dado que el sistema DataServer se corresponde con uno de los módulos que componen el sistema Conex de Proin, S.L. nos marcamos como objetivo adaptar los distintos módulos de Proin, S.L. para su trabajo con el DataServer. En este sentido, siguiendo la

planificación marcada, se han ido adaptando las aplicaciones según se disponía de funcionalidades en el DataServer.

Actualmente el sistema DataServer se encuentra instalado en más de 60 instalaciones, ejecutándose como servicio en equipos servidor, o ejecutándose como servicio del sistema Conex en equipos de sobremesa. Las incidencias surgidas con el mismo se han debido principalmente a problemas con el motor de base de datos, a la necesidad de nuevas funcionalidades o a la incorrecta gestión de los lotes de los stocks por parte de los usuarios, que generan descuadres en los datos de trazabilidad.

Se han detectado problemas de rendimiento por la gran cantidad de datos que se van generando en los procesos de fabricación y con los que el sistema debe trabajar. Estos problemas han sido resueltos en la última iteración.

Una vez que tenemos un sistema estable y escalable nos podemos plantear nuevos retos de funcionalidad del sistema:

- Generar nuevos paquetes de reglas:
    - Analisis de la calidad del producto fabricado. De esta forma el usuario podría ser avisado por el sistema, una vez terminada la fabricación, de un posible problema de calidad en el producto que lleva el camión.
    - Módulo que avise al usuario que deben tomarse muestras del producto fabricado, según ciertos criterios.
- Añadir un módulo de mantenimiento del centro de producción. Este módulo podría gestionar con avisos y alarmas el mantenimiento ha realizar de los distintos componentes del centro de producción, como pueden ser las cintas, las calibraciones de las básculas, el mantenimiento de los camiones, etc.
- Añadir un monitor de mantenimiento que analizase los datos de trazabilidad en busca de posibles problemas causados por la mala gestión de los stocks.
- Crear un sistema de centralización de datos en el cual un DataServer central gestionaría la información de los distintos DataServer de cada centro de producción. Este DataServer central sería el encargado de distribuir los mensajes a los DataServer de cada centro de producción en el caso de ser un mensaje destinado a él.

5. REFERENCIAS

[1] Real Academia Española, http://www.rae.es/rae.html, 2011

[2] UML, http://es.wikipedia.org/wiki/Lenguaje_Unificado_de_Modelado, 2011

[3] Programación extrema, http://es.wikipedia.org/wiki/Programación_extrema, 2011

[4] Java, http://www.oracle.com/technetwork/java/index.html, 2011

[5] Plataforma Java, http://es.wikipedia.org/wiki/Plataforma_Java, 2011

[6] Lenguaje Java, http://es.wikipedia.org/wiki/Lenguaje_de_programación_Java, 2011

[7] Eclipse, http://www.eclipse.org, 2011

[8] NetBeans, http:// www.netbeans.org, 2011

[9] JDO, http://www.oracle.com/technetwork/java/index-jsp-135919.html, 2011

[10] Java Community Process, http://www.jcp.org/en/home/index, 2011

[11] Hibernate, http://www.hibernate.org, 2011

[12] Enterprise Java Beans,
http://www.oracle.com/technetwork/java/javaee/ejb/index.html, 2011

[13] JPA, http://es.wikipedia.org/wiki/Java_Persistence_API, 2011

[14] Versant, http://www.versant.com/index.aspx, 2011

[15] EclipseLink, http://www.eclipse.org/eclipselink, 2011

[16] SQL Server, http://www.microsoft.com/spain/sql/2008/default.aspx, 2011

[17] HSQLDB, http://www.hsqldb.org, 2011

[18] Apache Derby, http://db.apache.org/derby/index.html, 2011

[19] RMI, http://www.oracle.com/technetwork/java/javase/tech/index-jsp-136424.html, 2011

[20] CORBA, http://www.corba.org, 2011

[21] SOAP, http://www.w3.org/TR/soap, 2011

[22] Jetty, http://jetty.codehaus.org/jetty, 2011

[23] Apache Tomcat, http://tomcat.apache.org, 2011

[24] Jython, http://www.jython.org, 2011

[25] Python, http://www.python.org, 2011

[26] JasperReport, http://jasperforge.org/projects/jasperreports, 2011

[27] IReport, http://jasperforge.org/projects/ireport, 2011

[28] JFreeChart, http://www.jfree.org/jfreechart, 2011

[29] Omondo, http://www.ejb3.org, 2011

[30] JUnit, http://www.junit.org, 2011

[31] JBossRules, http://community.jboss.org/wiki/JBossRules, 2011

[32] Procedimientos Integrados, S.L., http://www.proin.es, 2011

[33] ContretOnline, http://www.concretonline.com, 2011

[34] EHE-08,
http://es.wikipedia.org/wiki/Instrucción_Española_del_Hormigón_Estructural, 2011

[35] EHE-08 y Proin, S.L.,
http://www.concretonline.com/pdf/00hormigon/art_tec/PROIN_EHE_2011.pdf, 2011

[36] Java socket, http://download.oracle.com/javase/6/docs/api/java/net/Socket.html, 2011

[37] Java thread, http://download.oracle.com/javase/6/docs/api/java/lang/Thread.html, 2011

[38] Patrones de diseño software, http://es.wikipedia.org/wiki/Patrón_de_diseño, 2011

[39] Factory Method, http://es.wikipedia.org/wiki/Factory_Method_(patrón_de_diseño), 2011

[40] Observer, http://es.wikipedia.org/wiki/Observer_(patrón_de_diseño), 2011

[41] Proxy, http://es.wikipedia.org/wiki/Proxy_(patrón_de_diseño), 2011

[42] Iterator, http://es.wikipedia.org/wiki/Iterador_(patrón_de_diseño), 2011

[43] Adapter, http://es.wikipedia.org/wiki/Adapter_(patrón_de_diseño), 2011

[44] Command, http://es.wikipedia.org/wiki/Command_(patrón_de_diseño), 2011

6. ANEXOS

6.1. Contenido del CD

Junto a la memoria se deja un CD con la siguiente estructura de directorios:

- Base de datos
- Documentos
    - Informes de pruebas
    - Informes de trazabilidad
    - Manual de usuarios
- Fuentes
- Programa

6.2. Manual de usuario

A continuación se muestra una versión reducida del manual de usuario del sistema Proin Conex. Se deja una copia del manual de usuario completo en el recurso Manual del CD adjunto.

El módulo de gestión de datos Conex IU

El IU es el programa Conex encargado de administrar la información que necesita Conex para funcionar y la que se obtiene de su funcionamiento. En cierto modo, el funcionamiento general del sistema Conex se realiza bajo este programa, y sus funciones principales se llevan a cabo mediante las utilidades que incorpora. Entre ellas está la de lanzar fabricaciones, crear fórmulas, gestionar pedidos, supervisar albaranes y fabricaciones, crear informes y exportar la información a otros sistemas externos.

Funcionamiento utilidades comunes a todo el IU

A continuación describiremos el funcionamiento de algunas de las utilidades más importantes que el IU incorpora con la finalidad de facilitar el acceso a la información

Las barras de navegación de datos: Visualizar, crear, modificar y eliminar datos básicos

Se trata de una serie de botones que nos permitirá navegar por la información (llamada registros) mostrada en la pantalla en la cual nos encontremos, por ejemplo clientes. También nos permite modificar, crear y eliminar los datos. Observese que el espacio que ocupa cada cliente (o camión, obra, proveedor) en la base de datos se denomina 'registro'.

La disponibilidad de los botones depende del modo o situación en el cual nos encontremos. De esta manera, continuando con el ejemplo de clientes, no dispondremos de la posibilidad de pasar al cliente siguiente si estamos modificando los datos de otro. Estas situaciones se llaman modos y son los siguientes:

- **Modo navegación**. El predeterminado nada más aparecer en pantalla sirve sólo para ver los registros. En este modo no podremos ni modificar, ni añadir, ni eliminar registros.
- **Modo creación**. Cuando estamos creando un registro. En este modo los campos con la información del registro se habilitan para proceder a introducir sus datos.
- **Modo edición**. Cuando estamos modificando un cliente. En este modo los campos con la información del registro se habilitan para proceder a modificar sus datos.

Continuando con el ejemplo de clientes, la función de cada uno se detalla brevemente situando el cursor del ratón encima de los botones. No obstante los detallaremos a continuación:

- **Primer registro**. Nos presenta en pantalla el primer cliente. Sólo disponible en modo navegación.
- **Anterior registro**. Nos presenta en pantalla el cliente anterior. Sólo disponible en modo navegación.
- **Editar registro.** Modifica los datos del cliente que visualizamos actualmente en pantalla. Activa el modo edición. Sólo disponible en modo navegación.
- **Siguiente registro.** Sólo disponible en modo navegación.
- **Último registro.** Sólo disponible en modo navegación.
- **Añadir nuevo registro**. Activa el modo creación de cliente con el que podemos añadir un nuevo cliente a la base de datos. Sólo disponible en modo navegación.
- **Borrar registro.** Eliminar el cliente que visualizamos actualmente en pantalla. Tras pulsarlo siempre nos aparecerá un mensaje de confirmación para evitar borrados accidentales. Sólo disponible en modo navegación.
- **Salvar cambios.** Acepta y guarda los datos introducidos al cliente actual. Activa el modo creación. Sólo operativo en modo edición o creación.
- **Cancelar cambios**. Ignora los datos introducidos tras pulsar el botón azul de modificación de datos o el botón de añadir cliente. Sólo operativo en modo edición o creación.

Para crear un nuevo registro o proveedor, seguiríamos el siguiente proceso. Pulsar el botón 'Añadir':

Observamos que ahora todos los controles que muestran los datos se hacen operativos: estamos en modo creación. Es el momento de introducir los datos en cada campo:

No es obligatorio rellenar todos los campos, excepto los marcados en rojo (Código y razón social en la ilustración adjunta) que **SÍ SON OBLIGATORIOS**.

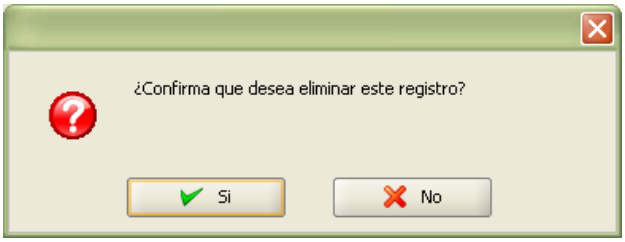
Una vez que hayamos terminado, si creemos que los datos introducidos son correctos y deseamos guardarlos deberemos pulsar el botón 'Aceptar':

Ahora tenemos un nuevo registro proveedor creado. Si por el contrario no deseamos guardarlos por cualquier motivo debemos pulsar el botón 'Cancelar' para ignorar la información que hubiésemos introducido y volver al modo navegación:

Si lo que deseamos es modificar los datos de un proveedor (por ejemplo su código, nombre, teléfono...) deberemos proceder de la misma forma que para crearlo exceptuando que inicialmente pulsaremos el botón azul 'Modificar' en lugar del botón 'Añadir':

Para eliminar un proveedor, primero debemos presentarlo en pantalla por medio de las barras de navegación 'Primero', 'Anterior', 'Siguiente' y 'Último' o buscándolo mediante las herramientas de búsqueda que detallaremos a continuación. Una vez tengamos el cliente a borrar en pantalla, pulsar el botón 'Eliminar':

Lo que nos presentará un mensaje de que nos preguntará si confirmamos borrar el registro (proveedor en este caso):



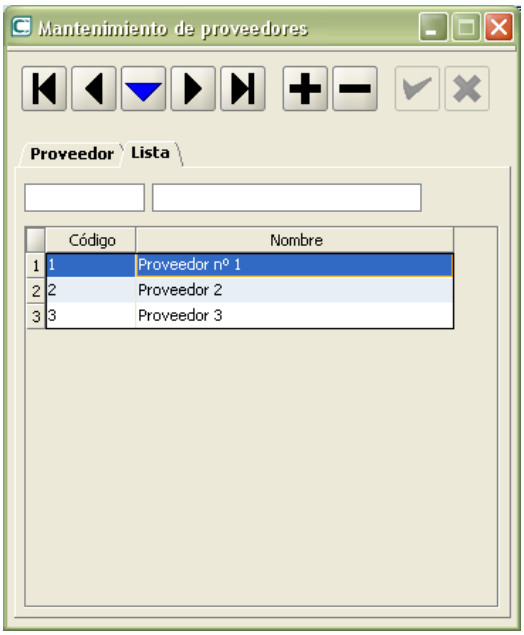
Si pulsamos 'Si' el proveedor se borrará automáticamente y en pantalla observaremos el proveedor que iba a continuación. Si pulsamos 'No' el proveedor no se borrará y continuará en pantalla.

La misma explicación que hemos desarrollado aquí para la barra de navegación de la ventana de proveedores es aplicable a todas aquellas ventanas en las que aparezca la barra de navegación, sólo que referido, claro está, al tipo de dato que estemos visualizando, modificando, creando o eliminando en ese momento.

Paneles de lista: acceso más fácil a la información.

Se trata de un panel adicional en la pantalla que nos situemos que nos permitirá un acceso más rápido y fácil a todos los registros creados. Se presenta mediante una pestaña adicional titulada 'Lista':

Al pulsar sobre esta pestaña nos aparecerá una lista de los registros de la base de datos en la que nos encontremos. El aspecto de lista de clientes por ejemplo sería:



La ventaja fundamental de esta vista es que podemos ver varios registros a la vez (en este caso proveedores) que desde la vista normal (pestaña Proveedor) sería imposible. Además de esto, la lista cuenta con una barra de desplazamiento a la derecha con la que podremos desplazarnos incluso a los registros que por cuestiones de espacio no aparecen en ella. Mientras tengamos seleccionado un registro

en esta lista podemos hacer cualquiera de las operaciones habituales: inspeccionar todos los detalles del registro seleccionado (pestaña Proveedor), modificarlo pinchando en el botón 'Modificar' o eliminarlo pulsando el botón 'Eliminar', ambos en la barra de navegación superior.

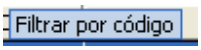
Esta utilidad podremos encontrarla prácticamente en todas las pantallas de captura de datos.

Las barras de búsqueda y filtrado: para encontrar lo que buscamos.

Es un conjunto de controles en pantalla que nos ayudan a encontrar la información rápidamente evitando tener que buscarla navegando de registro en registro con la barra de navegación. Piense que buscar, por ejemplo un proveedor entre 10 pulsando el botón 'Siguiente' cada vez que no encontremos el proveedor deseado no representa ninguna dificultad: con nueve pulsaciones como máximo lo encontraremos. Si en lugar de 10 tenemos 1000, deberíamos pulsar varios centenares de veces en la mayor parte de los casos, con la pérdida de tiempo y agilidad que esto conlleva.

Esta barra se presenta siempre sobre una vista de lista, en la cual los registros se presentan unos encima de otros formando filas superpuestas.

Se trata de una a varias cajas de texto rectangulares (dentro del rectángulo señalador negro en la ilustración adjunta) donde el usuario introduce la información a buscar dentro del campo al cual representa dicha caja de texto. El campo correspondiente a cada caja se especifica situando el cursor del ratón encima de ella. En el caso de la caja de la izquierda nos permite buscar un proveedor por su código porque situando el cursor del ratón encima nos seña 'Filtrar por código':



Así por ejemplo, para comenzar el filtrado introduciríamos el código en la caja de texto y pulsamos la tecla [INTRO] tras introducir el código deseado (1), lo que ocasionará que se omitan todos los proveedores cuyo proveedor no sea '1.

Además de filtrar por código, tenemos la posibilidad de filtrar por nombre de proveedor en la caja de texto de la derecha. El modo de actuar es el mismo que en caso anterior con el código, sólo que ahora basaremos nuestra condición de filtrado en el nombre de proveedor. La única diferencia entre

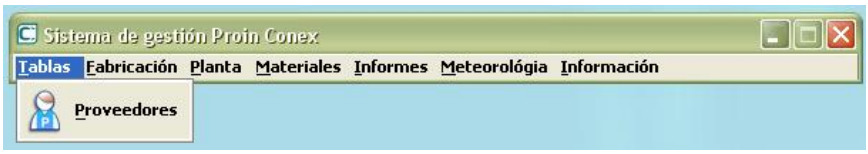
ambos filtros es que para filtrar por nombre de proveedor basta con especificar una parte de éste y en el código es necesario que se especifique la totalidad.

La operativa aquí explicada para búsqueda y filtrado de registros es aplicable a otras áreas del programa, como fórmulas, fabricaciones, etc.

Los menús

La estructura del IU se basa en menús desplegables, agrupados en áreas relacionadas entre sí para una mejor comprensión y comodidad por parte del usuario. A continuación describiremos la función de cada una de las opciones de estos menús.

Menú Tablas



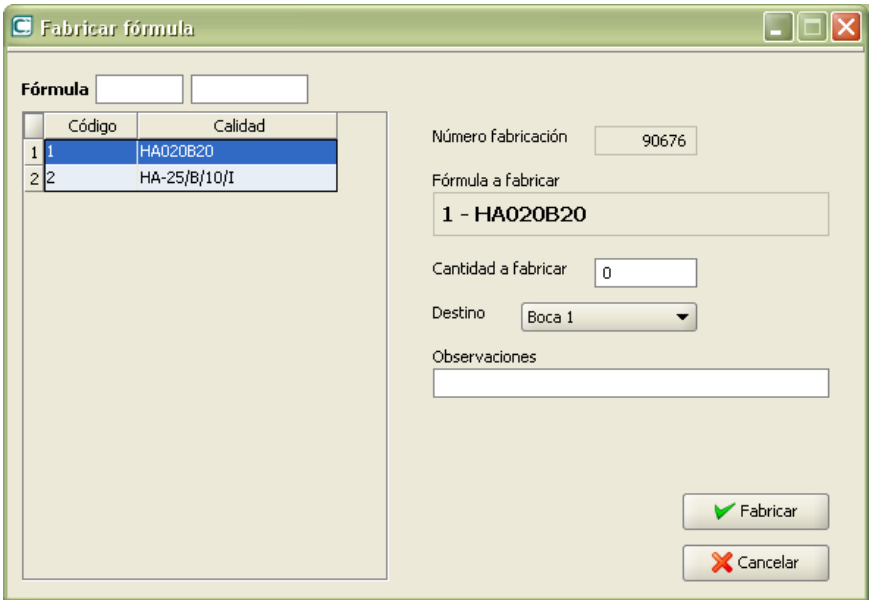
Este menú agrupa las funciones para crear y mantener prácticamente todos los datos que necesita Conex para realizar las operaciones principales. Según las opciones activas puede disponer de más o menos componentes, en este caso sólo tiene la opción de Proveedores.

- **Proveedores**. Creación y mantenimiento de proveedores. Aquí es donde crearemos los proveedores de materias primas y servicios de la planta. Observe que un transportista (dato necesario para crear camiones, como veremos a continuación) se considera un proveedor de transporte, por lo que sería aquí donde habría que darlo de alta.

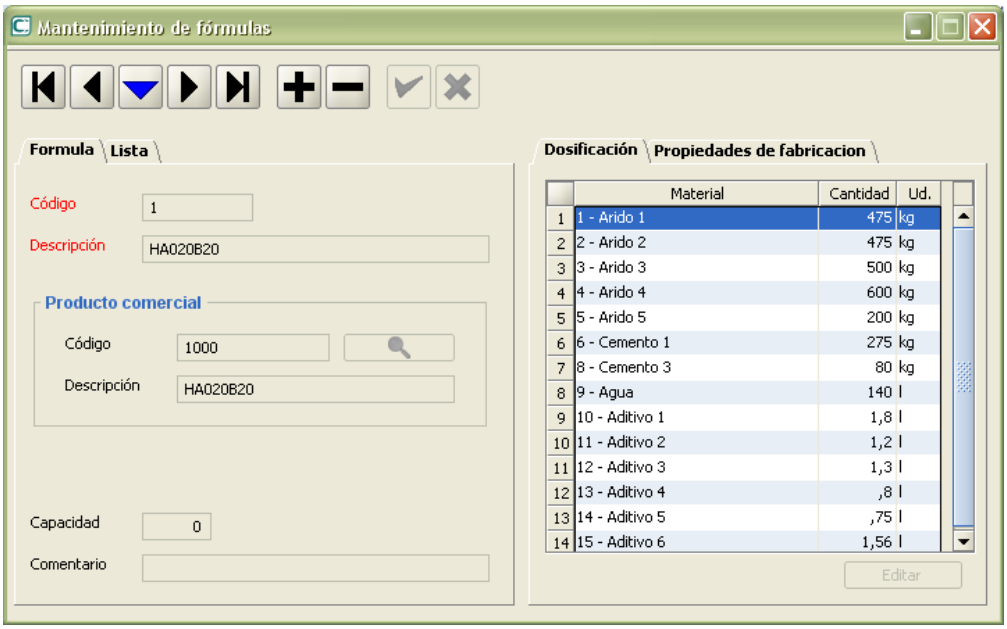
Menú Fabricación

Bajo este menú encontraremos todas las opciones necesarias para fabricar o para inspeccionar los resultados de las fabricaciones.

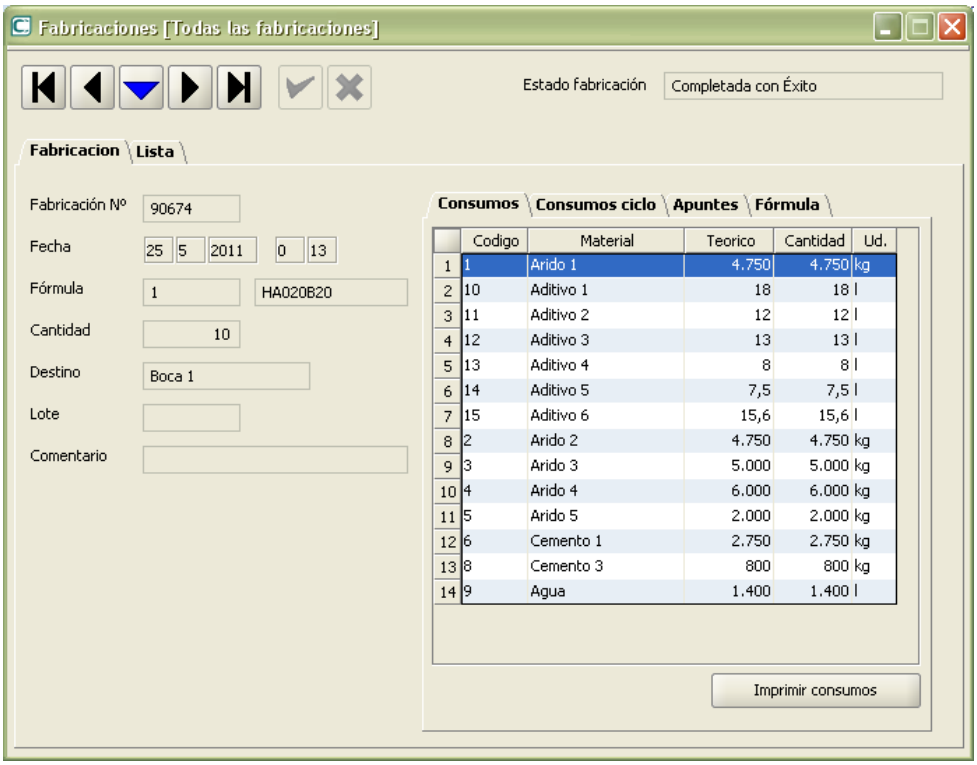
- **Fabricar.** Con esta opción se inicia el proceso de lanzar una fabricación.



- **Fórmulas.** Creación y mantenimiento de fórmulas. Su funcionamiento se detallará más adelante.

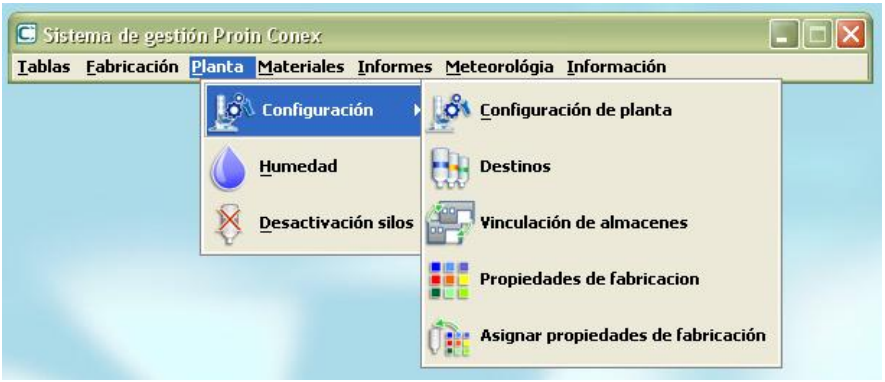


- **Fabricaciones.** Muestra en pantalla los albaranes fabricados junto con sus consumos de materiales. Su funcionamiento se detallará posteriormente.

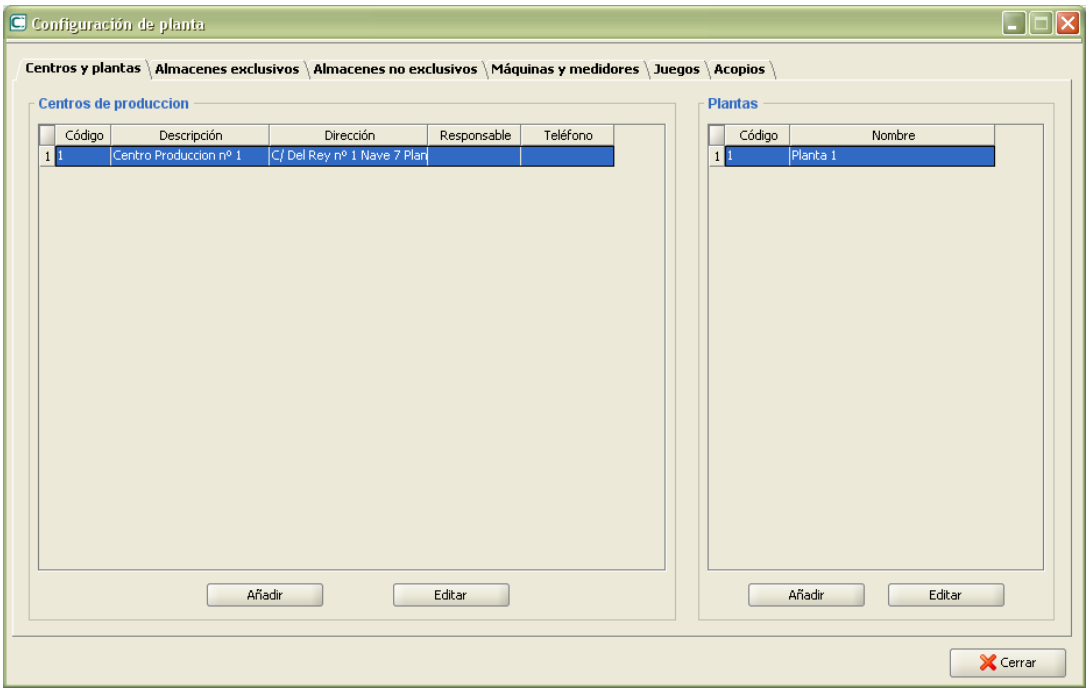


Menú Planta

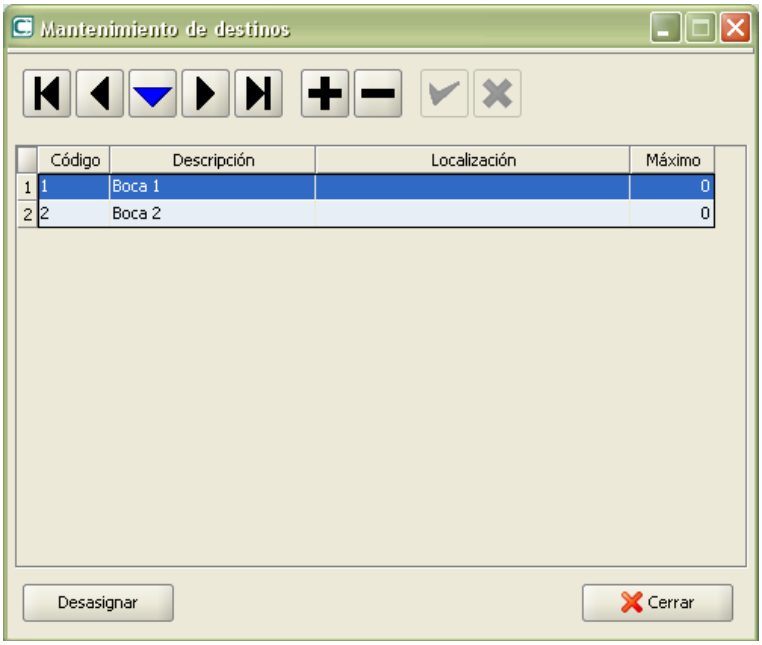
Este menú agrupa las opciones relativas a la configuración básica de la instalación en la cual nos encontremos. Los datos que aquí gestionaremos están muy relacionados con la operativa de fabricación diaria.



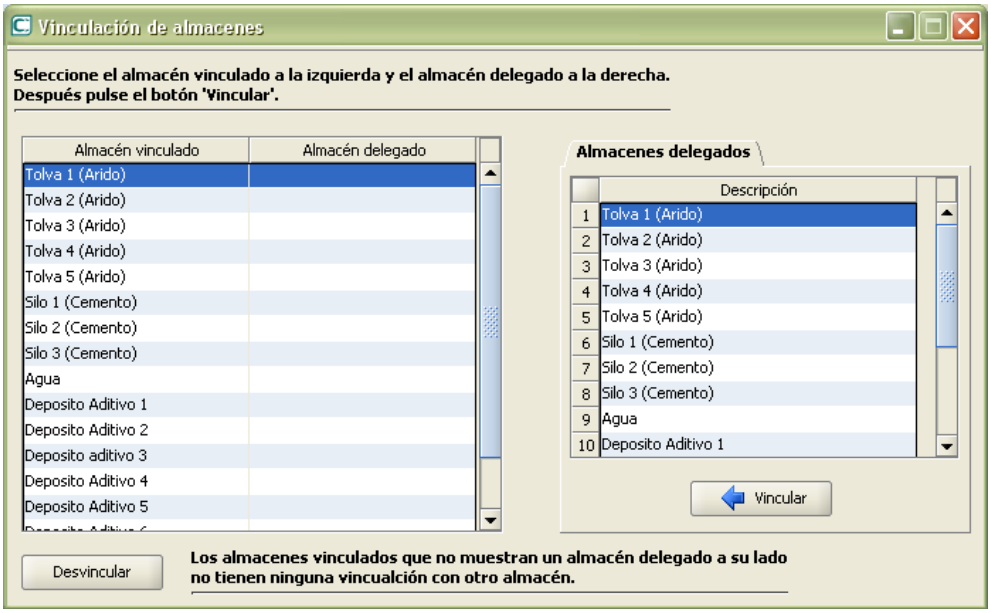
- **Configuración de planta.** Desde aquí se administra toda la maquinaria de planta encargada de almacenar materias primas, como silos y tolvas. Los datos que contienen ya vienen configurados en la puesta en marcha del sistema Conex para la instalación en la que nos encontremos, por lo que no es necesario ni recomendable manipularlos, a no ser que sea bajo la supervisión de un técnico Conex cualificado.



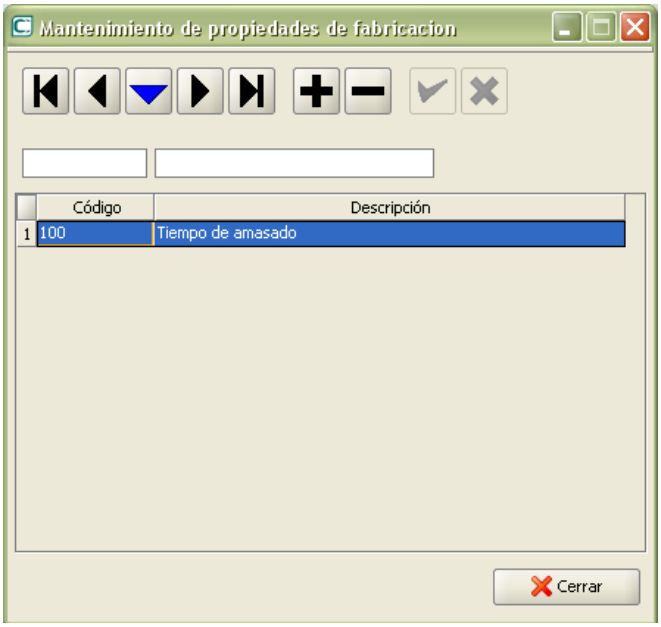
- **Destinos.** Esta opción permite configurar los almacenes de destino de las fabricacines.



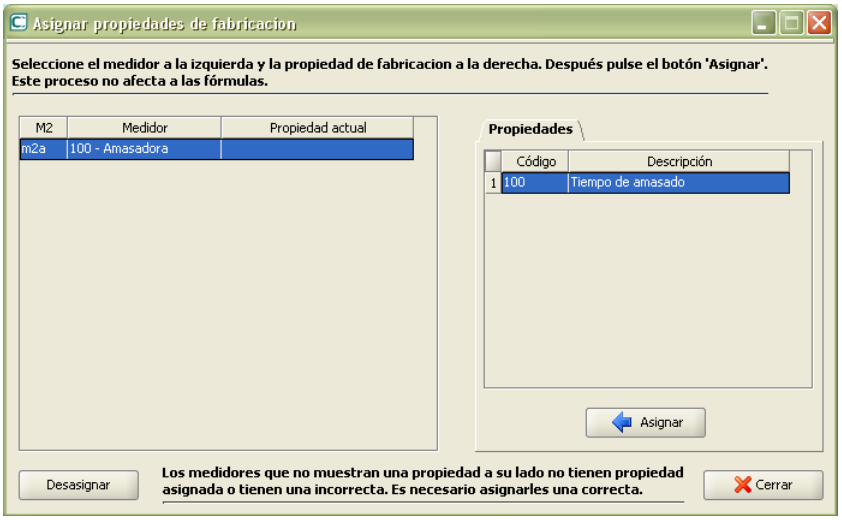
- **Vinculación de almacenes.** Esta opción permite vincular un almacén con otro que trabajará con el rol de delegado, siendo en este en el que se reflejaran todos los cambios de stock referentes al almaén vinculado.



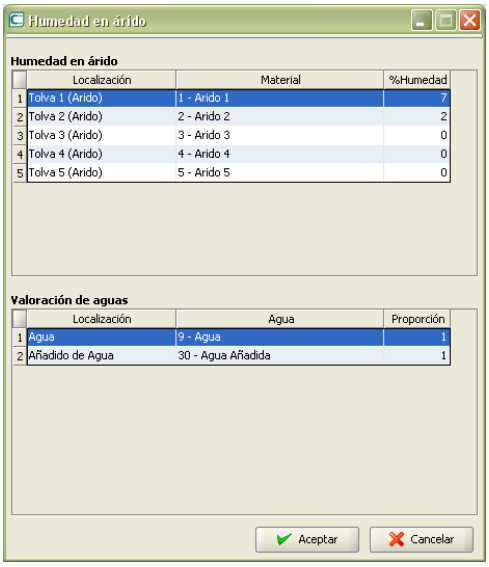
- **Propiedades de fabricación.** Esta opción permite dar de alta propiedades de fabricación, como puede ser el tiempo de amasado.



- **Asignar propiedades de fabricación.** Permite asignar a una máquina la propiedad de fabricación a medir.



- **Humedad.** Esta opción sólo está disponible si trabajamos en una planta de hormigón. Aquí mantenemos la información relativa al porcentaje de humedad de los áridos con los que estamos fabricando. Dependiendo del grado de humedad que configuremos, la proporción agua/árido a aplicar según fórmula variará.



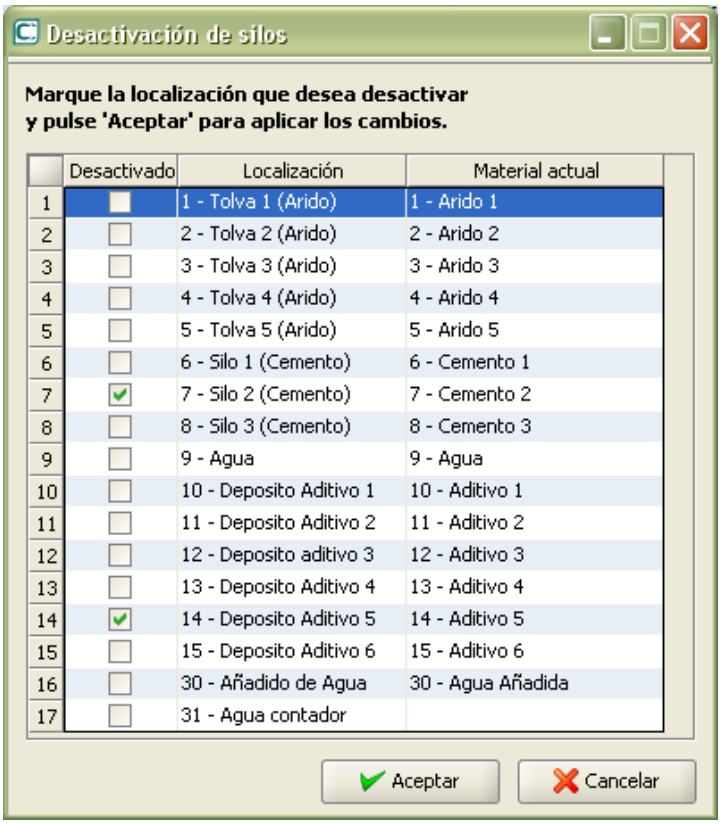
Conex aplica las humedades por localización, lo que quiere decir que configuraremos un porcentaje de humedad a la tolva lo que repercutirá en la dosificación final del material que contiene. No obstante, para facilitar las cosas basta con rellenar la humedad deseada en el la tabla superior, donde junto a cada localización aparece el material que contiene en ese momento. En el ejemplo de la ilustración, hemos configurado una humedad del 7 % en la tolva 1 que contiene actualmente Arido 1.

El agua o las aguas que se verán afectadas por estos porcentajes serán las que tengan en la columna 'Proporción' del panel inferior 'Valoración de aguas' un valor distinto de 0. Si hay más de una, los valores que introduzcamos serán la proporción de ellas con las que se añadirá o quitará agua a la hora de fabricar.

Para aplicar y guardar los datos que introduzcamos deberemos pulsar el botón 'Acepta' tanto del panel superior como del inferior. Es importante señalar que durante la carga **cualquier cambio efectuado en las humedades no se aplicará hasta la siguiente fabricación**.

- **Desactivación de silos.** Con esta utilidad controlaremos que silos queremos utilizar en cada momento a la hora de fabricar. Está pensado en las situaciones en las cuales dos silos contienen el mismo material. Su funcionamiento se basa en marcar en la columna 'Desactivado' los silos que queremos desactivar. Quiere esto decir que todos los que **NO** están marcados, **SI** están activos.

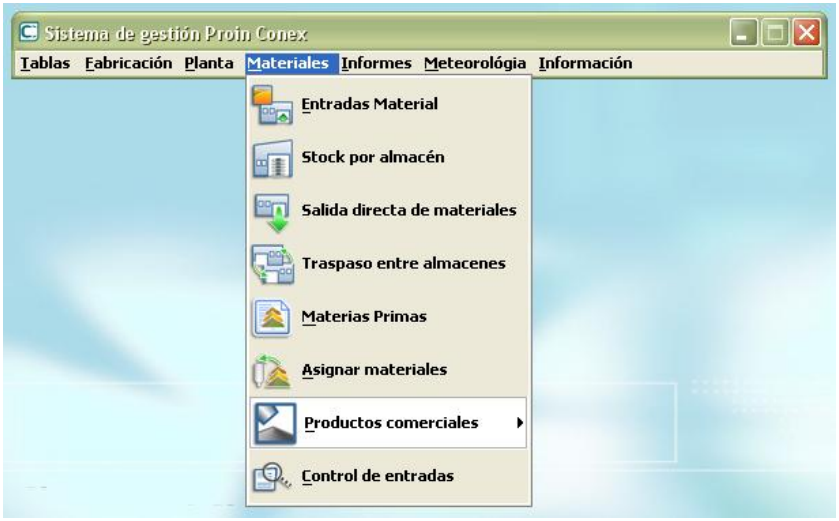
Si hay dos o más silos activos con el mismo material, Conex repartirá equitativamente entre ellos la dosificación. Si están todos desactivados, el programa generará un error a la hora de fabricar.



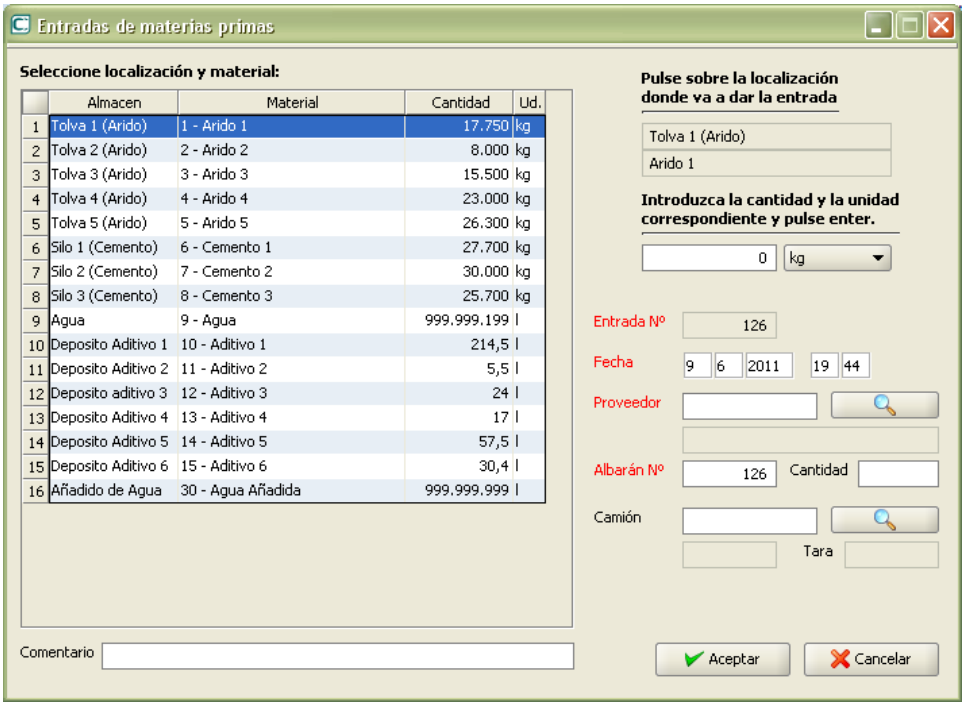
Al igual que sucedía con las humedades, es importante señalar que **durante la carga cualquier cambio efectuado en la desactivación de silos no se aplicará hasta la siguiente fabricación**.

Menú Materiales

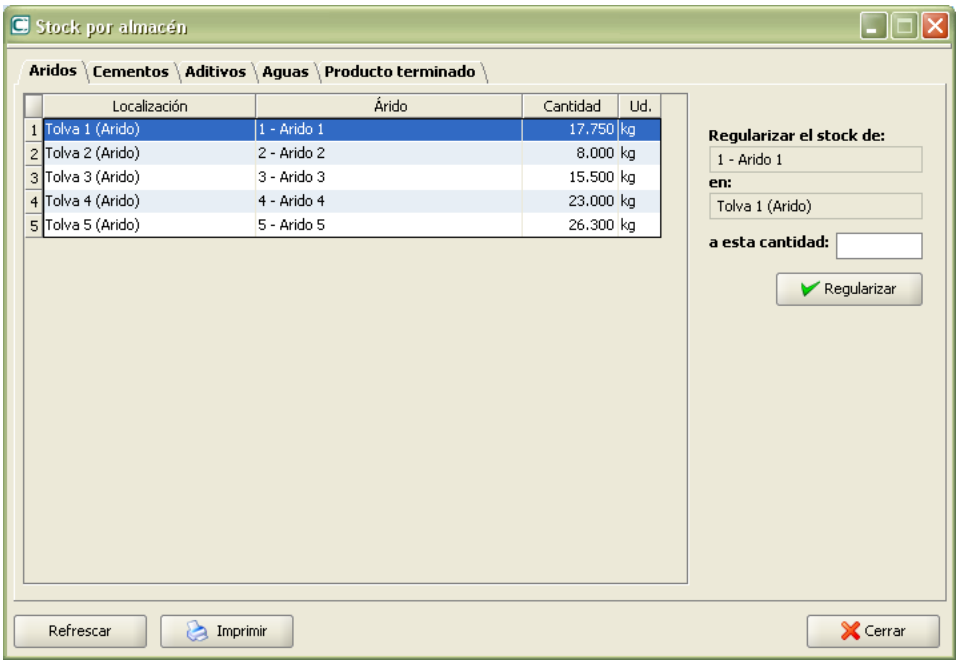
Este menú reúne todas las utilidades Conex relacionadas con las materias primas. Desde su mantenimiento, su gestión de entradas o control de stock



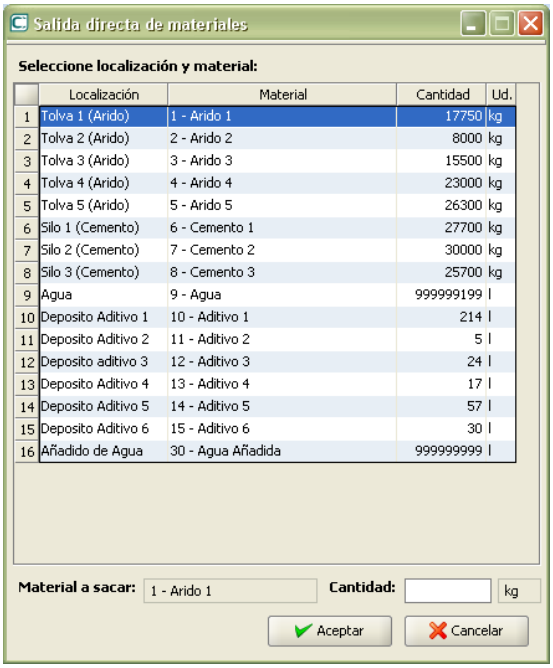
- **Entradas Material.** Desde aquí daremos de alta nuevas existencias de cualquier materia prima cuyo stock deseemos controlar. Para ello seleccionamos la localización que lo contiene, le introducimos la cantidad y datos como proveedor o camión que nos lo trae. Su funcionamiento se detallará en capítulos posteriores.



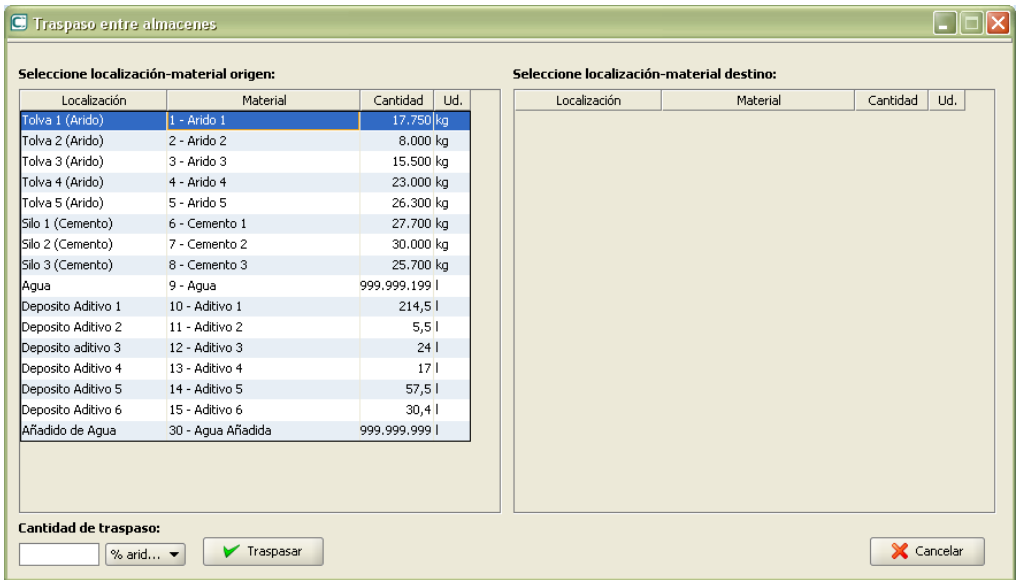
- **Stock por almacén.** Se trata de una pantalla en la que podremos visualizar de una sola vez las existencias de material en cada una de las localizaciones de la planta. Para que las cantidades sean correctas es imprescindible generar entradas de materias primas. Desde esta ventana podremos realizar regularizaciones para ajustar el stock a la realidad en caso de descuadres.



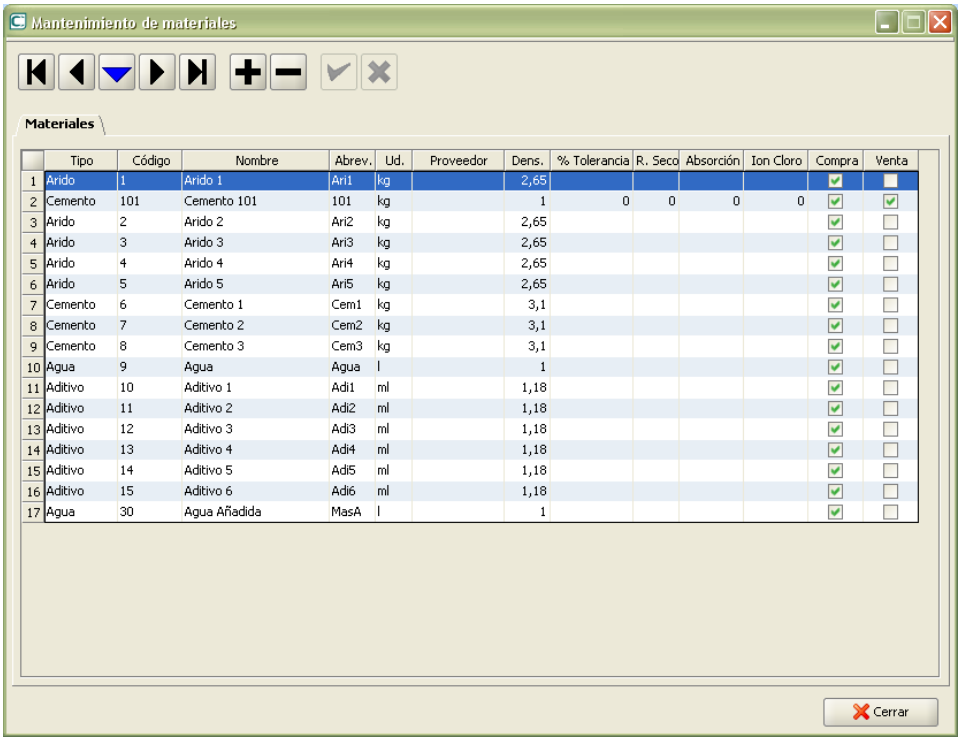
- **Salida directa de materiales.** Aquí podemos generar la operación inversa a las entradas: en lugar de dar de alta nuevas existencias, las daremos de baja. Puede ser útil para aquellos casos en que ciertas cantidades de materiales salen de planta manualmente y no queremos perder control de stock con ellas. Para ello la seleccionamos en la lista, introducimos la cantidad a sacar en el campo 'Cantidad' y pulsamos el botón 'Aceptar'.



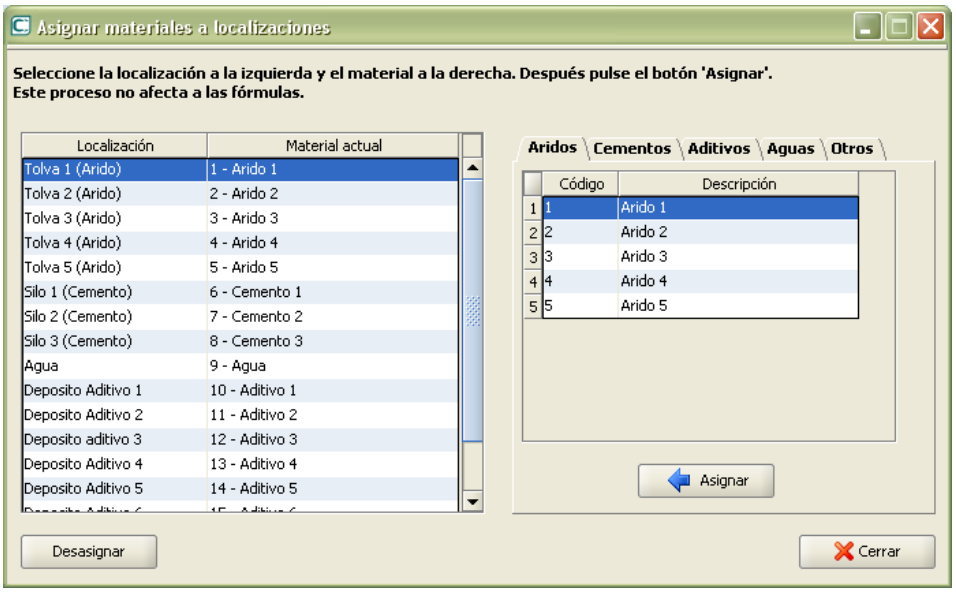
- **Traspaso entre almacenes.** A través de esta ventana podemos realizar traspasos de material desde un almacén a otro, manteniendo así correctamente el stock y la trazabilidad.



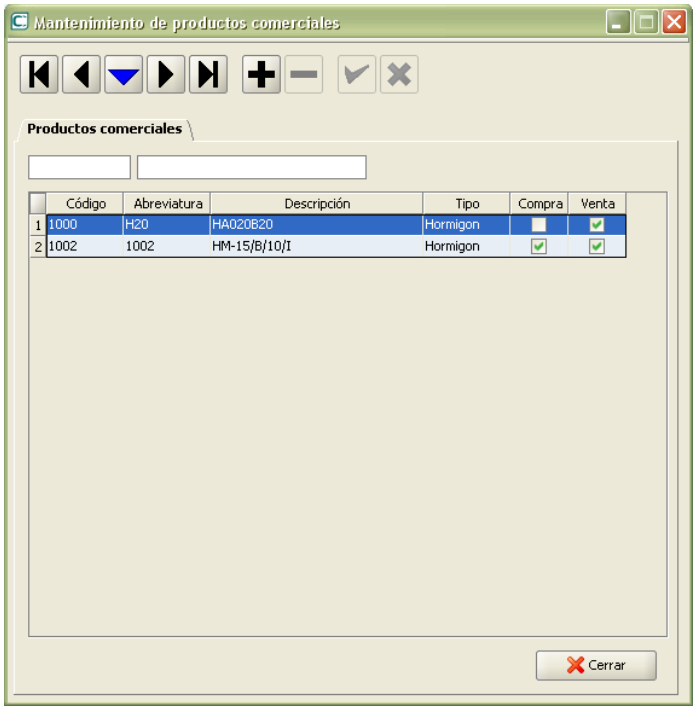
- **Materias primas.** Aquí es donde daremos de alta y mantendremos las materias primas con las cuales fabricaremos. Es de vital importancia que todas estén correctamente mantenidas, de lo contrario, el stock, los consumos y el control de calidad podrían verse afectados.



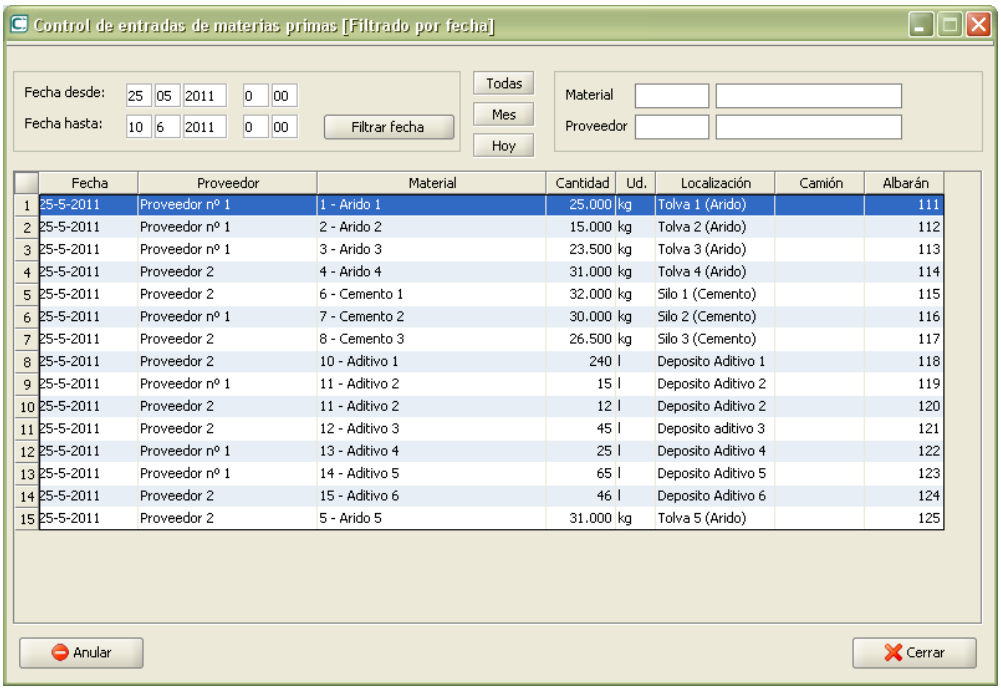
- **Asignar materiales.** Conex gestiona las materias primas en base a la localización en la cual estén contenidos. Para especificar que material contiene cada una tenemos esta opción del menú materiales. Esta ventana permite tanto asignar un material como desasignarlo. Al asignar o desasignar se genera o elimina el stock respectivamente.



- **Productos comerciales.** Los productos comerciales son aquellos que se obtienen de fabricar una fórmula. En este apartado es donde los daremos de alta y mantendremos. De cara a enlazar la producción de la planta con la facturación, un buen mantenimiento de estos productos resulta fundamental.



- **Control de entradas.** Desde esta utilidad gestionaremos todas las entradas que vayamos creando. Por defecto, esta pantalla sólo muestra las entradas realizadas durante el día en curso. Si deseamos ver las que se han realizado anteriormente habría que rellenar el intervalo de fechas 'desde' y 'hasta' y pulsar seguidamente el botón 'Filtrar fecha'. Así mismo podremos ver todas pulsando el botón 'Todas', sólo las del mes actual (botón 'Mes' o las del día en curso (botón 'Hoy'). Desde esta ventana también podemos anular entradas de material realizadas por error.



	Fecha	Proveedor	Material	Cantidad	Ud.	Localización	Camión	Albarán
1	25-5-2011	Proveedor nº 1	1 - Arido 1	25.000	kg	Tolva 1 (Arido)		111
2	25-5-2011	Proveedor nº 1	2 - Arido 2	15.000	kg	Tolva 2 (Arido)		112
3	25-5-2011	Proveedor nº 1	3 - Arido 3	23.500	kg	Tolva 3 (Arido)		113
4	25-5-2011	Proveedor 2	4 - Arido 4	31.000	kg	Tolva 4 (Arido)		114
5	25-5-2011	Proveedor 2	6 - Cemento 1	32.000	kg	Silo 1 (Cemento)		115
6	25-5-2011	Proveedor nº 1	7 - Cemento 2	30.000	kg	Silo 2 (Cemento)		116
7	25-5-2011	Proveedor 2	8 - Cemento 3	26.500	kg	Silo 3 (Cemento)		117
8	25-5-2011	Proveedor 2	10 - Aditivo 1	240	l	Deposito Aditivo 1		118
9	25-5-2011	Proveedor nº 1	11 - Aditivo 2	15	l	Deposito Aditivo 2		119
10	25-5-2011	Proveedor 2	11 - Aditivo 2	12	l	Deposito Aditivo 2		120
11	25-5-2011	Proveedor 2	12 - Aditivo 3	45	l	Deposito aditivo 3		121
12	25-5-2011	Proveedor nº 1	13 - Aditivo 4	25	l	Deposito Aditivo 4		122
13	25-5-2011	Proveedor nº 1	14 - Aditivo 5	65	l	Deposito Aditivo 5		123
14	25-5-2011	Proveedor 2	15 - Aditivo 6	46	l	Deposito Aditivo 6		124
15	25-5-2011	Proveedor 2	5 - Arido 5	31.000	kg	Tolva 5 (Arido)		125

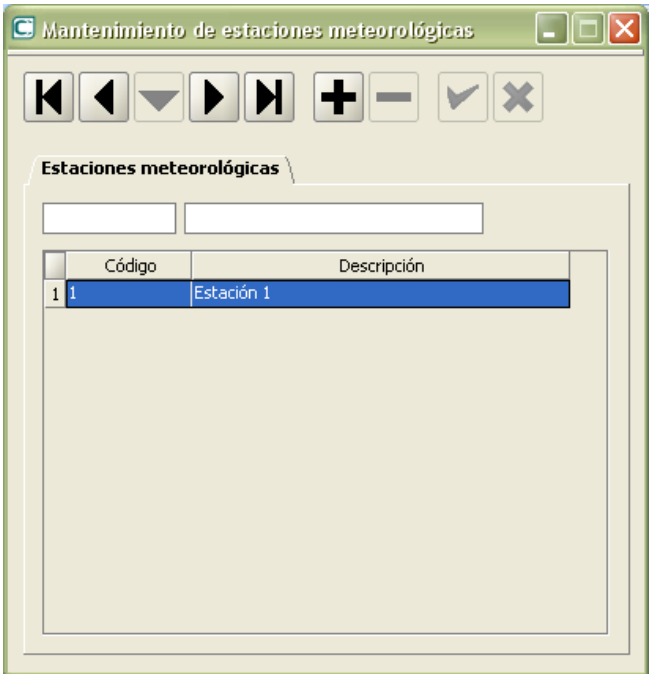
Menús de Informes

Este menú reúne todos los informes de trazabilidad de los que dispone el sistema Conex IU. Permite una generación en vista previa que puede ser exportada a varios formatos y puede ser impreso.

Menú Meteorología

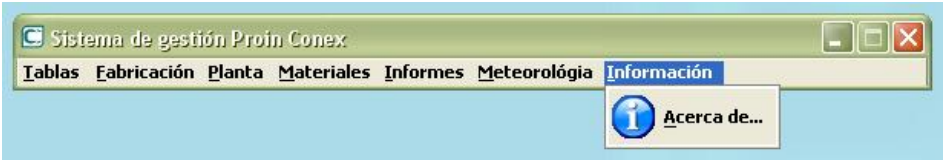
Este menú contiene las opciones para mantener las estaciones meteorológicas y mostrar información actualizada de las condiciones leidas por las mismas.

- **Estaciones.** Esta ventana permite realizar el mantenimiento de las estaciones meteorológicas.



- **Evaluación meteorológica estación 1.** Permite ver los datos de la evaluación meteorológica más cercana realizada por el sistema.

Menú Información



Este menú nos proporciona información acerca del programa.

- **Acerca de.** Muestra datos sobre la versión del programa.

6.3. Enumerado del intérprete de comandos

```
public enum Interpretado {
   ACTUALIZA_NUMERO_CICLOS (
         "07L", ActualizaNumeroCiclos.class,
         new Param("idFabricacion",      int.class,      ent),
         new Param("numeroCiclos",       int.class,      ent)),

   ADD_ALMACEN_TO_AGREGADO (
         "20A", AddAlmacenToAgregado.class,
         new Param("idAlmacenAgregado",  int.class,      ent),
         new Param("idAlmacenAnadido",   int.class,      ent)),

   ADD_NIVEL_ALMACEN (
         "20N", AddNivelAlmacen.class,
         new Param("idAlmacen",          int.class,      ent),
         new Param("nivel",              int.class,      ent),
         new Param("capacidad",          double.class,   comill)),

   AGREGAR_CATEGORIA_A_MATERIAL (
         "27M", AgregarCategoriaAMaterial.class,
         new Param("idMateriaPrima",    int.class,  ent),
         new Param("idCategoria",       int.class,      ent)),

   AGREGAR_DESTINO (
         "23F", AgregarDestino.class,
         new Param("nombreAcopio",       String.class,   comill),
         new Param("m2",                 String.class,   comill),
         new Param("idContenedor",       int.class,      ent),
         new Param("idAlmacenDestino",   int.class,      ent),
         new Param("idSemaforoAcopio",   int.class,      ent)),

   AGREGAR_ORIGEN (
         "23H", AgregarOrigen.class,
         new Param("nombreAcopio",       String.class,   comill),
         new Param("m2",                 String.class,   comill),
         new Param("idContenedor",       int.class,      ent),
         new Param("idAlmacenOrigen",    int.class,      ent),
         new Param("idSemaforoAcopio",   int.class,      ent)),
```

```
ALTA_ACOPIO (
    "23A", AltaAcopio.class,
    new Param("nombreAcopio",      String.class,   comill),
    new Param("numeroAcopio",      int.class,      ent),
    new Param("m2",                String.class,   comill),
    new Param("idContenedor",      int.class,      ent)),

ALTA_ALMACEN_EXCLUSIVO (
    "20X", AltaAlmacenExclusivo.class,
    new Param("nombreAlmacen",     String.class,   comill),
    new Param("unidades",          String.class,   comill),
    new Param("capacidad",         double.class,   comill),
    new Param("idAlmacenContened", int.class,      ent),
    new Param("codigoExterno",     int.class,      ent)),

ALTA_ARTICULO (
    "20R", AltaArticulo.class,
    new Param("codigoExterno",     String.class,   comill),
    new Param("codigoExternoProveedor",     String.class,   nullcomill),
    new Param("descripcion",       String.class,   comill),
    new Param("descrAbreviada",    String.class,   nullcomill),
    new Param("descrAlternativa",     String.class, nullcomill)),

ALTA_BOCA (
    "20B", AltaBoca.class,
    new Param("codigoCentro",      String.class,   comill),
    new Param("idAlmacen",         int.class,      ent),
    new Param("codigoBoca",        String.class,   comill),
    new Param("m2",                String.class,   comill),
    new Param("unidades",          String.class,   comill)),

ALTA_CATEGORIA (
    "20I", AltaCategoria.class,
    new Param("codExtArticulo",    String.class,   comill),
    new Param("descripcion",       String.class,   nullcomill, true),
    new Param("descAbreviada",     String.class,   nullcomill, true),
    new Param("tipoCategoria",     Integer.class,  nullcomill, true),
    new Param("idCategoria",       Integer.class,  nullcomill, true)),
```

```
ALTA_CENTRO_PRODUCCION (
        "20C", AltaCentroProduccion.class,
        new Param("codExternoCentro",   String.class,   comill),
        new Param("nombreCentro",       String.class,   comill),
        new Param("direccion",          String.class,   nullcomill),
        new Param("responsable",        String.class,   nullcomill),
        new Param("telefono",           String.class,   nullcomill)),

ALTA_COMPRABLE (
        "24X", AltaComprable.class,
        new Param("codigoExternoArticulo", String.class,   comill)),

ALTA_DESTINO (
        "21H", AltaDestino.class,
        new Param("codExterno",         String.class,   comill),
        new Param("descripcion",        String.class,   nullcomill, true),
        new Param("localizacion",       Integer.class,  nullcomill, true),
        new Param("cargaMaxima",        Double.class,   nullcomill, true),
        new Param("premezcla",          boolean.class,  bool)),

ALTA_ESTACION (
        "21A", AltaEstacionMeteorologica.class,
        new Param("codExterno",         String.class,   comill),
        new Param("descripcion",        String.class,   nullcomill, true)),

ALTA_FABRICACION (
        "20F", AltaFabricacion.class,
        new Param("idMaterial",         Integer.class,  nullcomill, true),
        new Param("idAlmacen",          Integer.class,  nullcomill, true),
        new Param("unidades",           String.class,   nullcomill, true),
        new Param("idFormula",          int.class,      ent),
        new Param("ajuste",             int.class,      ent),
        new Param("cantidad",           double.class,   comill),
        new Param("capacidad",          double.class,   comill),
        new Param("descripcion",        String.class,   comill),
        new Param("destino",            int.class,      ent),
        new Param("entrada",            int.class,      ent),
        new Param("estadoFabricacionId",int.class,      ent),
        new Param("intervencion",       int.class,      ent),
        new Param("momento",            double.class,   comill),
```

```
        new Param("caducidad",          Double.class,  nullcomill, true)),

ALTA_FORMULA (
        "21N", AltaFormula.class,
        new Param("idFormula",            int.class,      ent),
        new Param("descripcion",        String.class,   comill)),

ALTA_GRUPO_PETICIONES (
        "21K", AltaGrupoPeticiones.class,
        new Param("nombre",             String.class,   comill),
        new Param("idFormula",          Integer.class,  nullcomill, true),
        new Param("idRuta",             Integer.class,  nullcomill, true),
        new Param("cantidad",           Double.class,   nullcomill, true),
        new Param("uniCantidad",        String.class,   nullcomill, true),
        new Param("cantidadTotal",      Double.class,   nullcomill, true),
        new Param("uniCantidadTotal",   String.class,   nullcomill, true),
        new Param("capacidad",          Double.class,   nullcomill, true),
        new Param("uniCapacidad",       String.class,   nullcomill, true),
        new Param("semaforoBloqueo",    Integer.class,  nullcomill, true),
        new Param("semaforoConfirmacion",Integer.class, nullcomill, true),
        new Param("bloqueo",    Boolean.class,  nullcomill, true),
        new Param("dividir",    Boolean.class,  nullcomill, true),
        new Param("comentario",         String.class,   nullcomill, true)),

ALTA_JUEGO (
        "20J", AltaJuego.class,
        new Param("codigoJuego",        String.class,   comill),
        new Param("descripcion",        String.class,   nullcomill, true),
        new Param("codigoCentro",       String.class,   comill)),

ALTA_MAQUINA (
        "20Q", AltaMaquina.class,
        new Param("codigoCentro",       String.class,   comill),
        new Param("codigoExterno",      String.class,   comill),
        new Param("codigoDosificacion", int.class,      ent),
        new Param("m2",                 String.class,   comill),
        new Param("unidades",           String.class,   comill),
        new Param("descripcion",        String.class,   nullcomill)),
```

```
ALTA_MATERIA_PRIMA (
        "20M", AltaMateriaPrima.class,
        new Param("codExternoMaterial", String.class,   comill),
        new Param("tipoMaterial",       int.class,      ent),
        new Param("densidadReal",       double.class,   comill),
        new Param("proveedor",          Integer.class,  nullcomill, true),
        new Param("residuoSeco",        Double.class,   nullcomill, true),
        new Param("idCategoria",        Integer.class,  nullcomill, true),
        new Param("unidad",             String.class,   nullcomill, true),
        new Param("numeroOrden",        int.class,      ent)),

ALTA_RUTA (
        "21G", AltaRuta.class,
        new Param("codigoExterno",      String.class,   comill),
        new Param("idBoca", int.class,      ent),
        new Param("idDestino", int.class,      ent),
        new Param("nombreRuta",                 String.class,   comill)),

ALTA_VENDIBLE (
        "24V", AltaVendible.class,
        new Param("codigoExternoArticulo", String.class,   comill)),

ALTA_VENDIBLE_ID_ARTICULO (
        "25V", AltaVendible.class,
        new Param("idArticulo",         Integer.class,  nullcomill)),

ALTA_MATERIA_PRIMA_CON_ID (
        "20L", AltaMateriaPrima.class,
        new Param("codExternoMaterial", String.class,   comill),
        new Param("tipoMaterial",       int.class,      ent),
        new Param("densidadReal",       double.class,   comill),
        new Param("proveedor",          Integer.class,  nullcomill, true),
        new Param("residuoSeco",        Double.class,   nullcomill, true),
        new Param("idCategoria",        Integer.class,  nullcomill, true),
        new Param("unidad",             String.class,   nullcomill, true),
        new Param("numeroOrden",        int.class,      ent),
        new Param("idMateriaPrima",     Integer.class,  nullcomill)),
```

```
ALTA_MEDIDOR (
    "20D", AltaMedidor.class,
    new Param("codigoCentro",       String.class,   comill),
    new Param("codigoMaquina",      String.class,   comill),
    new Param("codigoDosificacion", int.class,      ent),
    new Param("m2",                 String.class,   comill),
    new Param("unidades",           String.class,   comill)),

ALTA_MUESTRA_METEOROLOGICA (
    "24B", AltaMuestraMeteorologica.class,
    new Param("codigoExternoEstacion",      String.class,   comill),
    new Param("gradosTemperatura",      double.class,   comill),
    new Param("milibaresPresion",       double.class,   comill),
    new Param("tpcHumedad",     double.class,   comill),
    new Param("litrosMetroHoraLuvia",       double.class,   comill),
    new Param("kmhoraVelocidadViento",      double.class,   comill),
    new Param("prevision",      String.class,   comill),
    new Param("direccionViento",        String.class,   comill)),

ALTA_PLANTA (
    "20P", AltaPlanta.class,
    new Param("nombreCentro",       String.class,   comill),
    new Param("idContenedor",       int.class,      ent)),

ALTA_PETICION (
    "21I", AltaPeticion.class,
    new Param("codigoExterno",      String.class,   comill),
    new Param("idGrupo",        Integer.class,  nullcomill, true),
    new Param("idFormula",      Integer.class,  nullcomill, true),
    new Param("idRuta",         Integer.class,  nullcomill, true),
    new Param("idJuego",        Integer.class,  nullcomill, true),
    new Param("cantidad",       Double.class,   nullcomill, true),
    new Param("uniCantidad",        String.class,   nullcomill, true),
    new Param("cantidadConsumida",      Double.class,   nullcomill, true),
    new Param("uniCantidadConsumida",       String.class,   nullcomill, true),
    new Param("cantidadTotal",      Double.class,   nullcomill, true),
    new Param("uniCantidadTotal",       String.class,   nullcomill, true),
    new Param("capacidad",      Double.class,   nullcomill, true),
    new Param("uniCapacidad",       String.class,   nullcomill, true),
    new Param("serie",      int.class,      ent),
```

```
        new Param("autoLote",         boolean.class,  bool),
        new Param("bloqueo",    Boolean.class,  nullcomill, true),
        new Param("dividir",    Boolean.class,  nullcomill, true),
        new Param("semaforoBloqueo",       Integer.class,  nullcomill, true),
        new Param("semaforoConfirmacion",       Integer.class,  nullcomill, true),
        new Param("stop",         boolean.class,  bool),
        new Param("lote",       String.class,   nullcomill, true),
        new Param("comentario",       String.class,   nullcomill, true)),

ALTA_ALMACEN_AGREGADO_SIMPLE (
        "20K", AltaAlmacenAgregadoSimple.class,
        new Param("nombreAlmacen",      String.class,   comill),
        new Param("idContenedor",       int.class,      ent)),

ANULA_ENTRADA (
        "22N", AnulaEntrada.class,
        new Param("idEntrada",          int.class,      ent)),

APUNTE (
        "07C", Apunte.class,
        new Param("numeroPlanta",       null,           ent),
        new Param("codigoFabricacion",  int.class,      ent),
        new Param("codigoMaquina",      int.class,      comill),
        new Param("cantidadReal",       double.class,   espcomill),
        new Param("motivo",             int.class,      ent),
        new Param("ciclo",              int.class,      ent),
        new Param("momentoInicio",      null,           nullcomill),
        new Param("momentoFin",         null,           comill),
        new Param("cantidadTeorica",    double.class,   espcomill)),

ASIGNAR_FABRICACION (
        "20Y", AsignarFabricacion.class,
        new Param("idConsumo",                  int.class,              ent),
        new Param("idFabricacion",                      int.class,      ent)),

ASIGNAR_JUEGO (
        "20V", AsignarJuego.class,
        new Param("codigoJuego",        String.class,   comill),
        new Param("codigoDosificacion", int.class,      ent),
        new Param("m2",                 String.class,   comill),
```

```
        new Param("codigoCentro",       String.class,   comill)),

    ASIGNAR_MATERIAL (
        "20S", AsignarMaterial.class,
        new Param("idAlmacen",          int.class,      ent),
        new Param("codExternoMaterial", String.class,   comill),
        new Param("disciplina",                 Integer.class,  nullcomill, true)),

    AVISO_NIVEL (
        "07N", AvisoNivel.class,
        new Param("numeroPlanta",       null,           ent),
        new Param("nivel",              int.class,      ent),
        new Param("idAlmacen",          int.class,      ent)),

    BAJA_VENDIBLE (
        "26V", BajaVendible.class,
        new Param("idArticulo",                 int.class,      ent)),

    CONSIGNA_MATERIAL (
        "23J", ConsignaMaterial.class,
        new Param("idAlmacen",          int.class,      ent),
        new Param("codigoMaterial",     String.class,   comill),
        new Param("cantidadTeorica",    double.class,   comill),
        new Param("unidad",             String.class,   comill),
        new Param("codigoExterno",      String.class,   comill)),

    CONSULTA_MASA (
        "07F", ConsultaMasa.class,
        new Param("numeroPlanta",       null,           ent),
        new Param("idFabricacion",      long.class,     ent),
        new Param("ciclo",              int.class,      ent),
        new Param("material",           int.class,      ent)),

    GET_ARTICULO_FROM_MATERIA_PRIMA (
        "07G", GetArticuloFromMateriaPrima.class,
        new Param("idMateria",      int.class,      ent)),

    GET_ARTICULO_FROM_CATEGORIA (
        "07H", GetArticuloFromCategoria.class,
        new Param("idCategoria",      int.class,      ent)),
```

```
CONSUMO_SEMIAUTOMATICO (
        "07c", ConsumoSemiautomatico.class,
        new Param("idAlmacen",          int.class,      ent),
        new Param("idFabricacion",      int.class,      ent),
        new Param("idMaterial",         int.class,      ent),
        new Param("cantidad",           double.class,   comill),
        new Param("unidad",             String.class,   comill)),

CONVERSION_UNIDADES (
        "20U", ConversionUnidad.class,
        new Param("valor",              double.class,   comill),
        new Param("unidad",             String.class,   comill),
        new Param("unidadDeseada",      String.class,   comill),
        new Param("idMaterial",         Integer.class,  nullcomill, true)),

DESACTIVA_ELEMENTO (
        "22K", DesactivaElemento.class,
        new Param("idElemento",         int.class,      ent)),

ACTIVA_ELEMENTO (
       "23C", ActivaElemento.class,
       new Param("idElemento",          int.class,      ent)),

DESASIGNAR_JUEGO (
        "22V", DesasignarJuego.class,
        new Param("codigoJuego",        String.class,   comill),
        new Param("codigoDosificacion", int.class,      ent),
        new Param("m2",                 String.class,   comill),
        new Param("codigoCentro",       String.class,   comill)),

DESASIGNAR_MATERIAL (
        "22S", DesasignarMaterial.class,
        new Param("idAlmacen",          int.class,      ent),
        new Param("idMaterial",         int.class,      ent)),

DESHABILITAR_ACOPIO (
        "21U", DeshabilitarAcopio.class,
        new Param("numeroAcopio",       int.class,      ent),
        new Param("m2",         String.class,   comill)),
```

```
DESUSCRIPCION (
    "06U", Desuscripcion.class,
    new Param("idAlmacen",          int.class,      comill),
    new Param("zonaAreaLocaliz",    null,           ent)),

DESVINCULAR_ALMACEN (
    "23V", DesvincularAlmacen.class,
    new Param("idAlmacenExclusivo",  int.class,      ent)),

EDIT_ACOPIO (
    "23B", EditAcopio.class,
    new Param("nombreAcopioAnt",    String.class,   comill),
    new Param("numeroAcopio",       int.class,      ent),
    new Param("m2Ant",              String.class,   comill),
    new Param("idContenedorAnt",    int.class,      ent),
    new Param("nombreAcopioModif",  String.class,   comill),
    new Param("nuevoNumeroAcopio",  int.class,      ent),
    new Param("m2Modif",            String.class,   comill),
    new Param("idContenedorModif",  int.class,      ent)),

EDIT_ALMACEN_EXCLUSIVO (
    "22X", EditAlmacenExclusivo.class,
    new Param("idAlmacen",          int.class,      ent),
    new Param("nombreAlmacen",      String.class,   comill),
    new Param("unidades",           String.class,   comill),
    new Param("capacidad",          double.class,   comill),
    new Param("idContenedor",       int.class,      ent),
    new Param("codigoExterno",      int.class,      ent)),

EDIT_ARTICULO (
    "22R", EditArticulo.class,
    new Param("idArticulo",         int.class,      ent),
    new Param("descripcion",        String.class,   comill),
    new Param("codigoExterno",      String.class,   comill),
    new Param("codigoExternoProveedor",     String.class,   nullcomill),
    new Param("descAbreviada",      String.class,   comill),
    new Param("descAlternativa",    String.class,   comill)),
```

```
EDIT_ARTICULO_Y_MATERIA_PRIMA (
        "22T", EditArticuloYMateriaPrima.class,
        new Param("idArticulo",         int.class,      ent),
        new Param("descripcion",        String.class,   comill),
        new Param("codigoExterno",      String.class,   comill),
        new Param("codigoExternoProveedor",      String.class,   nullcomill),
        new Param("descAbreviada",      String.class,   nullcomill),
        new Param("descAlternativa",    String.class,   nullcomill),
        new Param("tipoMaterial",       int.class,      ent),
        new Param("densidadReal",       double.class,   comill),
        new Param("proveedor",          Integer.class,  nullcomill, true),
        new Param("residuoSeco",        Double.class,   nullcomill, true),
        new Param("idCategoria",        Integer.class,  nullcomill, true),
        new Param("unidad",             String.class,   comill),
        new Param("numeroOrden",        int.class,      ent)),

EDIT_ARTICULO_Y_MATERIA_PRIMA_NO_ALT (
        "23T", EditArticuloYMateriaPrimaNoAlternativa.class,
        new Param("idArticulo",         int.class,      ent),
        new Param("descripcion",        String.class,   comill),
        new Param("codigoExterno",      String.class,   comill),
        new Param("codigoExternoProveedor",      String.class,   nullcomill),
        new Param("descAbreviada",      String.class,   nullcomill),
        new Param("tipoMaterial",       int.class,      ent),
        new Param("densidadReal",       double.class,   comill),
        new Param("proveedor",          Integer.class,  nullcomill, true),
        new Param("residuoSeco",        Double.class,   nullcomill, true),
        new Param("unidad",             String.class,   comill),
        new Param("numeroOrden",        int.class,      ent)),

EDIT_ARTICULO_Y_CATEGORIA (
        "22U", EditArticuloYCategoria.class,
        new Param("idArticulo",         int.class,      ent),
        new Param("descripcion",        String.class,   comill),
        new Param("codigoExterno",      String.class,   comill),
        new Param("descAbreviada",      String.class,   nullcomill),
        new Param("tipoCategoria",      Integer.class,  nullcomill)),

EDIT_BOCA (
        "22B", EditBoca.class,
```

```
        new Param("codigoCentroActual", String.class,   comill),
        new Param("idAlmacenActual",          int.class,      ent),
        new Param("codigoBocaActual",   String.class,   comill),
        new Param("m2Actual",           String.class,   comill),
        new Param("codigoCentroModif",  String.class,   comill),
        new Param("idAlmacenModif",           int.class,      ent),
        new Param("codigoBocaModif",    String.class,   comill),
        new Param("m2Modif",            String.class,   comill)),

    EDIT_BOCA_Y_MEDIDOR (
        "22Y", EditBocaYMedidor.class,
        new Param("codigoCentroActual", String.class,   comill),
        new Param("idAlmacenActual",          int.class,      ent),
        new Param("codigoBocaActual",   String.class,   comill),
        new Param("m2Actual",           String.class,   comill),
        new Param("codigoCentroModif",  String.class,   comill),
        new Param("idAlmacenModificado",          int.class,      ent),
        new Param("codigoBocaModif",    String.class,   comill),
        new Param("m2Modif",            String.class,   comill),
        new Param("unidades",           String.class,   comill)),

    EDIT_CATEGORIA (
        "23L", EditCategoria.class,
        new Param("idCategoria",        int.class,      ent),
        new Param("idArticulo",         int.class,      ent),
        new Param("descripcion",        String.class,   nullcomill, true),
        new Param("descAbreviada",      String.class,   nullcomill, true)),

    EDIT_CENTRO_PRODUCCION (
        "22C", EditCentroProduccion.class,
        new Param("idCentro",           int.class,      ent),
        new Param("codExternoCentro",   String.class,   comill),
        new Param("nombreCentro",       String.class,   comill),
        new Param("direccion",          String.class,   nullcomill, true),
        new Param("responsable",        String.class,   nullcomill, true),
        new Param("telefono",           String.class,   nullcomill, true)),

    EDIT_DESTINO (
        "22G", EditDestino.class,
        new Param("id",           int.class,      ent),
```

```
        new Param("codExterno",         String.class,   comill),
        new Param("descripcion",        String.class,   nullcomill, true),
        new Param("localizacion",       Integer.class,  nullcomill, true),
        new Param("cargaMaxima",        Double.class,   nullcomill, true),
        new Param("premezcla",          boolean.class,  bool)),

    EDIT_FABRICACION (
        "22F", EditFabricacion.class,
        new Param("idFabricacion",      int.class,      ent),
        new Param("idMaterial",         Integer.class,  nullcomill, true),
        new Param("idAlmacen",          Integer.class,  nullcomill, true),
        new Param("unidades",           String.class,   nullcomill, true),
        new Param("idFormula",          int.class,      ent),
        new Param("ajuste",             int.class,      ent),
        new Param("cantidad",           double.class,   comill),
        new Param("capacidad",          double.class,   comill),
        new Param("descripcion",        String.class,   comill),
        new Param("destino",            int.class,      ent),
        new Param("entrada",            int.class,      ent),
        new Param("estadoFabricacionId",int.class,      ent),
        new Param("intervencion",       int.class,      ent),
        new Param("momento",            double.class,   comill),
        new Param("caducidad",          Double.class,  nullcomill, true)),

    EDIT_GRUPO_PETICIONES (
        "22O", EditGrupoPeticiones.class,
        new Param("id",                 int.class,      ent),
        new Param("nombre",             String.class,   nullcomill, true),
        new Param("idFormula",          Integer.class,  nullcomill, true),
        new Param("idRuta",             Integer.class,  nullcomill, true),
        new Param("cantidad",           Double.class,   nullcomill, true),
        new Param("uniCantidad",        String.class,   nullcomill, true),
        new Param("cantidadTotal",      Double.class,   nullcomill, true),
        new Param("uniCantidadTotal",   String.class,   nullcomill, true),
        new Param("capacidad",          Double.class,   nullcomill, true),
        new Param("uniCapacidad",       String.class,   nullcomill, true),
        new Param("semaforoBloqueo",    Integer.class,  nullcomill, true),
        new Param("semaforoConfirmacion",Integer.class, nullcomill, true),
        new Param("bloqueo",    Boolean.class,  nullcomill, true),
        new Param("dividir",    Boolean.class,  nullcomill, true),
```

```
        new Param("comentario",        String.class,   nullcomill, true),
        new Param("overwrite", boolean.class, bool)),

    OVERWRITE_FROM_GRUPO_PETICIONES (
        "22I", OverwriteFromGrupoPeticiones.class,
        new Param("id",                int.class,      ent),
        new Param("formula",           boolean.class,  bool),
        new Param("ruta",              boolean.class,    bool),
        new Param("cantidad",          boolean.class,    bool),
        new Param("cantidadTotal",     boolean.class,    bool),
        new Param("capacidad",         boolean.class,    bool),
        new Param("semaforoBloqueo",   boolean.class,    bool),
        new Param("semaforoConfirmacion",boolean.class,  bool),
        new Param("bloqueo",           boolean.class,   bool),
        new Param("dividir",           boolean.class,   bool)),

    EDIT_JUEGO (
        "22J", EditJuego.class,
        new Param("idJuego",           int.class,      ent),
        new Param("codigoJuego",       String.class,   comill),
        new Param("descripcion",       String.class,   nullcomill, true),
        new Param("codigoCentro",      String.class,   comill)),

    EDIT_MAQUINA (
        "22Q", EditMaquina.class,
        new Param("codigoCentroAct",   String.class,   comill),
        new Param("codigoMaquinaAct",  String.class,   comill),
        new Param("codigoCentroModif", String.class,   comill),
        new Param("codigoMaquinaModif", String.class,  comill),
        new Param("descripcion",       String.class,   comill)),

    EDIT_MAQUINA_Y_MEDIDOR (
        "22Z", EditMaquinaYMedidor.class,
        new Param("codigoCentroAct",   String.class,   comill),
        new Param("codigoMaquinaAct",  String.class,   comill),
        new Param("codigoDosifAct",    int.class,      ent),
        new Param("m2Act",             String.class,   comill),
        new Param("codigoCentroModif", String.class,   comill),
        new Param("codigoMaquinaModif", String.class,  comill),
        new Param("codigoDosifModif",  int.class,      ent),
```

```
        new Param("m2Modif",          String.class,   comill),
        new Param("unidades",         String.class,   comill),
        new Param("descripcion",      String.class,   comill)),

    EDIT_MATERIA_PRIMA (
        "22M", EditMateriaPrima.class,
        new Param("idMaterial",       int.class,      ent),
        new Param("codExternoArticulo", String.class,   comill),
        new Param("tipoMaterial",     int.class,      ent),
        new Param("densidadReal",     double.class,   comill),
        new Param("proveedor",        Integer.class,  nullcomill, true),
        new Param("residuoSeco",      Double.class,   nullcomill, true),
        new Param("unidad",     String.class,   comill),
        new Param("numeroOrden",      int.class,      ent)),

    EDIT_MEDIDOR (
        "22D", EditMedidor.class,
        new Param("codigoDosifAct",   int.class,      ent),
        new Param("m2Act",            String.class,   comill),
        new Param("codigoCentroAct",  String.class,   comill),
        new Param("codigoDosifModif", int.class,      ent),
        new Param("m2Modif",          String.class,   comill),
        new Param("codigoCentroModif", String.class,   comill),
        new Param("codigoMaquina",    String.class,   comill),
        new Param("unidades",         String.class,   comill)),

    EDIT_PETICION (
        "22A", EditPeticion.class,
        new Param("id",               int.class,      ent),
        new Param("codigoExterno",    String.class,   comill),
        new Param("idGrupo",          Integer.class,  nullcomill, true),
        new Param("idFormula",        Integer.class,  nullcomill, true),
        new Param("idRuta",           Integer.class,  nullcomill, true),
        new Param("idJuego",          Integer.class,  nullcomill, true),
        new Param("cantidad",         Double.class,   nullcomill, true),
        new Param("uniCantidad",      String.class,   nullcomill, true),
        new Param("cantidadConsumida", Double.class,   nullcomill, true),
        new Param("uniCantidadConsumida",String.class,  nullcomill, true),
        new Param("cantidadTotal",    Double.class,   nullcomill, true),
        new Param("uniCantidadTotal", String.class,   nullcomill, true),
```

```
        new Param("capacidad",          Double.class,   nullcomill, true),
        new Param("uniCapacidad",       String.class,   nullcomill, true),
        new Param("serie",              int.class,      ent),
        new Param("autoLote",           boolean.class,  bool),
        new Param("bloqueo",    Boolean.class,  nullcomill, true),
        new Param("dividir",    Boolean.class,  nullcomill, true),
        new Param("semaforoBloqueo",    Integer.class,  nullcomill, true),
        new Param("semaforoConfirmacion",Integer.class,  nullcomill, true),
        new Param("stop",               boolean.class,  bool),
        new Param("lote",               String.class,   nullcomill, true),
        new Param("comentario",         String.class,   nullcomill, true)),

EDIT_PLANTA (
        "22P", EditPlanta.class,
        new Param("idPlanta",           int.class,      ent),
        new Param("nombrePlanta",       String.class,   comill),
        new Param("idContenedor",       int.class,      ent)),

EDIT_RUTA (
        "21L", EditRuta.class,
        new Param("id", int.class,      ent),
        new Param("codigoExterno",      String.class,   comill),
        new Param("idBoca", int.class,      ent),
        new Param("idDestino", int.class,   ent),
        new Param("nombreRuta",         String.class,   comill)),

ENTRADA_MATERIAL (
        "07E", EntradaMaterial.class,
        new Param("idAlmacen",          int.class,      ent),
        new Param("codigoMaterial",     int.class,      ent),
        new Param("cantidad",           double.class,   comill),
        new Param("unidad",             String.class,   comill),
        new Param("motivo",             null,           ent),
        new Param("momento",            null,           comill)),

ENTRADA_MATERIAL_LOTE_EXTERNO (
        "20E", EntradaMaterialLoteExterno.class,
        new Param("idAlmacen",          int.class,      ent),
        new Param("codigoMaterial",     int.class,      ent),
        new Param("cantidad",           double.class,   comill),
```

```
        new Param("unidad",             String.class,   comill),
        new Param("loteExterno",        String.class,   comill),
        new Param("caducidad",          Double.class,   nullcomill, true)),

    EVALUACION_METEOROLOGICA (
        "24A", EvaluacionMeteorologica.class,
        new Param("codigoExternoEstacion",String.class, comill)),

    ES_BORRABLE (
        "22H", EsBorrable.class,
        new Param("id",     int.class,      ent)),

    ES_EHE (
        "22L", EsEHE.class,
        new Param("idArticulo",     int.class,      ent)),

    EXISTE_ALMACEN (
        "21X", ExisteAlmacen.class,
        new Param("idAlmacen",          int.class,      ent)),

    EXISTE_OBJETO (
        "22E", ExisteObjeto.class,
        new Param("id",         int.class,      ent),
        new Param("nombreClase", String.class,   comill)),

    BORRA_PROIN_MODEL (
        "21F", BorrarProinModel.class,
        new Param("id",         int.class,      ent)),

    CLEAN_UP (
        "CLEAN", CleanUp.class),

    SE_PUEDE_FABRICAR (
        "21Z", SePuedeFabricar.class,
        new Param("idAlmacen",          int.class,      ent),
        new Param("m2",                 String.class,   comill),
        new Param("codigoDosificacion",         int.class,      ent)),

    EXISTE_ALMACEN_POR_NOMBRE (
        "21Y", ExisteAlmacenPorNombre.class,
```

```
        new Param("nombreAlmacen",      String.class,   comill),
        new Param("idContenedor",       int.class,      ent)),

    EXISTE_ARTICULO (
        "21R", ExisteArticulo.class,
        new Param("codExternoArticulo", String.class,   comill)),

    EXISTE_BOCA (
        "21B", ExisteBoca.class,
        new Param("idAlmacen",          int.class,      ent),
        new Param("codigoBoca",         String.class,   comill),
        new Param("codigoCentro",       String.class,   comill),
        new Param("m2",                 String.class,   comill)),

    EXISTE_CATEGORIA (
        "21E", ExisteCategoria.class,
        new Param("codExtCategoria", String.class,   comill)),

    EXISTE_CENTRO_PRODUCCION (
        "21C", ExisteCentroProduccion.class,
        new Param("codExternoCentro",   String.class,   comill)),

    EXISTE_COMPRABLE (
        "21W", ExisteComprable.class,
        new Param("codigoExternoArticulo", String.class,   comill)),

    EXISTE_JUEGO (
        "21J", ExisteJuego.class,
        new Param("codigoJuego",        String.class,   comill),
        new Param("codigoCentro",       String.class,   comill)),

    EXISTE_MAQUINA (
        "21Q", ExisteMaquina.class,
        new Param("codigoMaquina",      String.class,   comill),
        new Param("codigoCentro",       String.class,   comill),
        new Param("descripcion",        String.class,   comill)),

    EXISTE_MATERIA_PRIMA (
        "21M", ExisteMateriaPrima.class,
        new Param("codExtMateriaPrima", String.class,   comill)),
```

```
EXISTE_MEDIDOR (
        "21D", ExisteMedidor.class,
        new Param("codDosificacion",    int.class,      ent),
        new Param("m2",                 String.class,   comill),
        new Param("codigoCentro",       String.class,   comill),
        new Param("codigoMaquina",      String.class,   comill)),

EXISTE_PLANTA (
        "21P", ExistePlanta.class,
        new Param("nombrePlanta",       String.class,   comill),
        new Param("idContenedor",       int.class,      ent)),

EXISTE_STOCK_ACTIVO (
        "21S", ExisteStockActivo.class,
        new Param("idMaterial",         int.class,      ent),
        new Param("idAlmacen",          Integer.class,      nullcomill, true)),

EXISTE_VENDIBLE (
        "21V", ExisteVendible.class,
        new Param("codigoExternoArticulo", String.class,   comill)),

FIN_CICLO (
        "0AF", FinCiclo.class,
        new Param("ciclo",              int.class,      ent),
        new Param("idFabricacion",      long.class,     ent),
        new Param("codDosificacion",    null,           ent)),

FIN_FABRICACION_MANUAL (
        "23M", FinFabricacionManual.class,
        new Param("idFabricacion",      int.class,      ent)),

LIMPIAR_LOTES (
        "23Q", LimpiarLotesVirtuales.class,
        new Param("idAlmacen",          Integer.class,      nullcomill, true),
        new Param("codExternoMaterial", String.class,   nullcomill, true)),

HABILITAR_ACOPIO (
        "21T", HabilitarAcopio.class,
        new Param("numeroAcopio",       int.class,      ent),
        new Param("m2",         String.class,   comill)),
```

```
MENSAJE_FABRICACION (
        "07M", MensajeFabricacion.class,
        new Param("numeroPlanta",       null,          ent),
        new Param("idFabricacion",      long.class,    ent),
        new Param("ciclo",              int.class,     ent),
        new Param("modulo",             null,          ent),
        new Param("descripcion",        String.class,  comill)),

MENSAJE_M2_VIVO(
        "00e", Nop.class,
        new Param("mensaje",            String.class,  cadena)),

MODIFICACION_UNIDADES (
        "23N", ModificacionUnidades.class,
        new Param("idAlmacen",          int.class,     ent),
        new Param("codigoMaterial",     int.class,     ent),
        new Param("unidad",             String.class,  comill)),

MODIFICACION_DISCIPLINA (
        "23O", ModificacionDisciplina.class,
        new Param("idAlmacen",          int.class,     ent),
        new Param("codExternoMaterial", String.class,  comill),
        new Param("disciplina",         int.class,     ent)),

MONITORIZAR_ACOPIOS (
        "0AN", MonitorizarAcopios.class),

NO_OPERACION (
        "20O", Nop.class,
        new Param("mensaje",            String.class,  comill)),

NOP (
        "NOP", NopCallBack.class),

NOP2 (
        "nop", NopCallBack.class),

NOP3 (
        "Nop", NopCallBack.class),
```

```
DEL (
    "del", DeleteCallBack.class),

DEL2 (
    "DEL", DeleteCallBack.class),

PETICION_MATERIAL (
    "06M", PeticionMaterial.class,
    new Param("codigoCentro",      String.class,   comill),
    new Param("nombreM2",          String.class,   comill),
    new Param("codigDosificacion", int.class,      comill)),

PONER_ACOPIO_EN_REPOSO (
    "23D", PonerAcopioEnReposo.class,
    new Param("numeroAcopio",      int.class,      ent),
    new Param("m2",      String.class,   comill)),

PONER_ACOPIO_EN_USO (
    "23E", PonerAcopioEnUso.class,
    new Param("numeroAcopio",      int.class,      ent),
    new Param("m2",      String.class,   comill),
    new Param("idAlmacenOrigen",   int.class,      ent),
    new Param("idAlmacenDestino",  int.class,      ent)),

QUITAR_DESTINO (
    "23G", QuitarDestino.class,
    new Param("nombreAcopio",      String.class,   comill),
    new Param("m2",                String.class,   comill),
    new Param("idContenedor",      int.class,      ent),
    new Param("idAlmacenDestino",  int.class,      ent)),

QUITAR_CATEGORIA_A_MATERIAL (
    "28A", QuitarCategoriaAMaterial.class,
    new Param("idMateriaPrima",    int.class,  ent),
    new Param("idCategoria",       int.class,      ent)),

QUITAR_ORIGEN (
    "23I", QuitarOrigen.class,
    new Param("nombreAcopio",      String.class,   comill),
    new Param("m2",                String.class,   comill),
```

```
        new Param("idContenedor",       int.class,      ent),
        new Param("idAlmacenOrigen",    int.class,      ent)),

QUITAR_TODAS_CATEGORIAS_A_MATERIAL (
        "28B", QuitarTodasCategoriasAMaterial.class,
        new Param("idMateriaPrima",     int.class,  ent)),

REGULARIZACION (
        "20G", Regularizacion.class,
        new Param("idAlmacen",          int.class,      ent),
        new Param("codigoMaterial",     int.class,      ent),
        new Param("cantidad",           double.class,   comill),
        new Param("unidad",             String.class,   comill)),

REINIT_COMMAND_DISPATCHER (
        "ini", ReiniciaCommandDispatcher.class),

REINIT2_COMMAND_DISPATCHER (
        "INI", ReiniciaCommandDispatcher.class),

REINIT3_COMMAND_DISPATCHER (
        "Ini", ReiniciaCommandDispatcher.class),

SALIDA_MATERIAL (
        "07S", SalidaMaterial.class,
        new Param("idAlmacen",          int.class,      ent),
        new Param("codigoMaterial",     int.class,      ent),
        new Param("cantidad",           double.class,   comill),
        new Param("unidad",             String.class,   comill),
        new Param("motivo",             null,           ent),
        new Param("momento",            null,           comill)),

SALIDA_CONSIGNA (
        "23K", SalidaConsigna.class,
        new Param("idConsigna",         int.class,      ent),
        new Param("cantidadReal",       double.class,   comill),
        new Param("unidad",             String.class,   comill),
        new Param("fidedigno",          int.class,      ent)),
```

```
SET_EHE (
        "22W", SetEHE.class,
        new Param("idArticulo",        int.class,      ent),
        new Param("cumpleEHE",         boolean.class,  bool)),

SET_HABILITADO (
      "06I", SetHabilitada.class,
      new Param("idAlmacen",         int.class,      ent),
      new Param("m2",           String.class,   comill),
      new Param("codigoDosificacion",         int.class,      ent),
      new Param("habilitado",        int.class,  ent)),

SET_GESTIONA_LOTES (
      "23P", SetGestionLotes.class,
      new Param("idAlmacen",         Integer.class,      nullcomill, true),
      new Param("codExternoMaterial", String.class,   nullcomill, true),
      new Param("criterio",        int.class,  ent)),

SET_HUMEDAD_INSTANTANEA (
      "20H", SetHumedadInstantanea.class,
      new Param("idAlmacen",         int.class,      ent),
      new Param("idMaterial",        int.class,      ent),
      new Param("cantidad",          double.class,   comill),
      new Param("unidades",          String.class,   comill)),

SUSCRIPCION (
      "06S", Suscripcion.class,
      new Param("idAlmacen",         int.class,      comill),
      new Param("zonaAreaLocaliz",    null,          ent)),

TRANSMUTACION (
      "20T", Transmutacion.class,
      new Param("idAlmacen",         int.class,      ent),
      new Param("codigoMaterial",    int.class,      ent)),

TRASPASO (
      "20W", Traspaso.class,
      new Param("idAlmacenOrigen",    int.class,      ent),
      new Param("idAlmacenDestino",   int.class,      ent),
      new Param("idMaterial",        int.class,      ent),
```

```
        new Param("cantidad",          double.class,   comill),
        new Param("unidad",            String.class,   comill)),

JDO_FATAL_EXCEPTION (
        "jfe", JDOFatalException.class),

JDO_CAN_RETRY_EXCEPTION (
        "jcr", JDOCanRetryException.class),

VINCULAR_ALMACEN (
        "20Z", VincularAlmacen.class,
        new Param("idAlmacenExclusivo", int.class,      ent),
        new Param("idAlmacenDelegado",  int.class,      ent));
```

6.4. Resultado de las pruebas

6.4.1. Informe de ejecución de pruebas del protocolo de comunicación entre el cliente y el sistema


```
<?xml version="1.0" encoding="UTF-8"?>
<testrun name="TestInterpretado" project="dataServer.servidor" tests="90" started="90" failures="0" errors="0" ignored="0">
  <testsuite name="com.proin.dataServer.interprete.TestInterpretado" time="1.453">
    <testcase name="testGetInstance" time="0.016"/>
    <testcase name="testActivaElemento" time="0.703"/>
    <testcase name="testApunte" time="0.016"/>
    <testcase name="testConsumoSemiautomatico" time="0.015"/>
    <testcase name="testFabricar" time="0.047"/>
    <testcase name="testConsultaMasa" time="0.0"/>
    <testcase name="testAvisoNivel" time="0.016"/>
    <testcase name="testMensajeFabricacion" time="0.0"/>
    <testcase name="testSalidaMaterial" time="0.031"/>
    <testcase name="testEntradaMaterial" time="0.0"/>
    <testcase name="testSuscripcion" time="0.016"/>
    <testcase name="testDesuscripcion" time="0.0"/>
    <testcase name="testPeticionMaterial" time="0.015"/>
    <testcase name="testSetHabilitado" time="0.0"/>
    <testcase name="testAddNivel" time="0.016"/>
    <testcase name="testAltaCentroProduccion" time="0.0"/>
    <testcase name="testAltaPlanta" time="0.015"/>
    <testcase name="testAltaAlmacenExclusivo" time="0.0"/>
    <testcase name="testAddAlmacenToAgregado" time="0.016"/>
    <testcase name="testAltaArticulo" time="0.0"/>
    <testcase name="testAltaBoca" time="0.016"/>
    <testcase name="testAltaFabricacion" time="0.0"/>
    <testcase name="testAltaMateriaPrima" time="0.015"/>
    <testcase name="testAltaMateriaPrimaConId" time="0.0"/>
    <testcase name="testTransmutacion" time="0.016"/>
    <testcase name="testNoOperacion" time="0.0"/>
    <testcase name="testAsignacionMaterial" time="0.016"/>
    <testcase name="testEsEHE" time="0.0"/>
    <testcase name="testExisteAlmacen" time="0.015"/>
    <testcase name="testExisteArticulo" time="0.0"/>
    <testcase name="testExisteBoca" time="0.016"/>
```

<testcase name="testExisteMateriaPrima" time="0.0"/>
<testcase name="testExisteComprable" time="0.0"/>
<testcase name="testExisteVendible" time="0.015"/>
<testcase name="testEditArticuloYMateriaPrima" time="0.0"/>
<testcase name="testAgregarDestino" time="0.016"/>
<testcase name="testAgregarOrigen" time="0.0"/>
<testcase name="testAltaAcopio" time="0.016"/>
<testcase name="testAltaCategoria" time="0.015"/>
<testcase name="testAltaJuego" time="0.0"/>
<testcase name="testAltaMaquina" time="0.016"/>
<testcase name="testAltaMedidor" time="0.0"/>
<testcase name="testAnulaEntrada" time="0.016"/>
<testcase name="testAsignarFabricacion" time="0.0"/>
<testcase name="testAsignarJuego" time="0.0"/>
<testcase name="testConsignaMaterial" time="0.015"/>
<testcase name="testConversionUnidades" time="0.0"/>
<testcase name="testDesactivaElemento" time="0.016"/>
<testcase name="testDesasignarJuego" time="0.0"/>
<testcase name="testDesasignarMaterial" time="0.015"/>
<testcase name="testEditAcopio" time="0.016"/>
<testcase name="testEditAlmacenAgregado" time="0.0"/>
<testcase name="testEditAlmacenExclusivo" time="0.0"/>
<testcase name="testEditArticulo" time="0.016"/>
<testcase name="testEditBoca" time="0.0"/>
<testcase name="testEditBocaYMedidor" time="0.015"/>
<testcase name="testEditCategoria" time="0.0"/>
<testcase name="testEditCentroProduccion" time="0.0"/>
<testcase name="testEditFabricacion" time="0.016"/>
<testcase name="testEditJuego" time="0.0"/>
<testcase name="testEditMaquina" time="0.0"/>
<testcase name="testEditMaquinaYMedidor" time="0.016"/>
<testcase name="testEditMateriaPrima" time="0.0"/>
<testcase name="testEditMedidor" time="0.031"/>
<testcase name="testEditPlanta" time="0.0"/>
<testcase name="testEntradaMaterialLoteExterno" time="0.0"/>
<testcase name="testExisteAlmacenPorNombre" time="0.015"/>
<testcase name="testExisteCentroProduccion" time="0.0"/>
<testcase name="testExisteJuego" time="0.0"/>
<testcase name="testExisteMaquina" time="0.032"/>
<testcase name="testExisteMedidor" time="0.0"/>

```
    <testcase name="testExistePlanta" time="0.031"/>
    <testcase name="testFinCiclo" time="0.0"/>
    <testcase name="testPonerAcopioEnReposo" time="0.016"/>
    <testcase name="testPonerAcopioEnUso" time="0.015"/>
    <testcase name="testQuitarDestino" time="0.0"/>
    <testcase name="testQuitarOrigen" time="0.0"/>
    <testcase name="testRegularizacion" time="0.016"/>
    <testcase name="testSalidaConsigna" time="0.0"/>
    <testcase name="testSetEHE" time="0.0"/>
    <testcase name="testSetHumedadInstantanea" time="0.015"/>
    <testcase name="testTraspaso" time="0.0"/>
    <testcase name="testVincularAlmacen" time="0.0"/>
    <testcase name="testAltaSello" time="0.016"/>
    <testcase name="testAltaTipoSello" time="0.0"/>
    <testcase name="testAltaTarifa" time="0.0"/>
    <testcase name="testAltaTipoTarifa" time="0.016"/>
    <testcase name="testEditTipoSello" time="0.0"/>
    <testcase name="testEditTarifa" time="0.015"/>
    <testcase name="testEditTipoTarifa" time="0.0"/>
  </testsuite>
</testrun>
```

6.4.2. Informe de ejecución de pruebas de funcionalidad del sistema


```
<?xml version="1.0" encoding="UTF-8"?>
<testrun name="TestsDataServer" project="dataServer.main" tests="316" started="316" failures="0"
errors="0" ignored="4">
  <testsuite name="com.proin.dataServer.comandos.TestDesasignarJuego" time="3.109">
    <testcase name="testDesasignarJuego" time="2.984"/>
    <testcase name="testDesasignarJuegoRepetido" time="0.047"/>
    <testcase name="testAsignarJuegoMedidorInexistente" time="0.031"/>
    <testcase name="testAsignarJuegoJuegoInexistente" time="0.031"/>
  </testsuite>
  <testsuite name="com.proin.dataServer.comandos.TestRegularizacion" time="1.953">
    <testcase name="testRegularizacionEntrada" time="1.375"/>
    <testcase name="testRegularizacionEntradaDesdeStockNegativo" time="0.282"/>
    <testcase name="testRegularizacionSalida" time="0.156"/>
    <testcase name="testRegularizacionSinMovimientos" time="0.14"/>
  </testsuite>
  <testsuite name="com.proin.dataServer.comandos.TestEditMateriaPrima" time="0.516">
    <testcase name="testEditMateriaPrima" time="0.094"/>
    <testcase name="testEditMateriaPrimaMismoArticulo" time="0.047"/>
    <testcase name="testEditMateriaPrimaSinCambios" time="0.078"/>
    <testcase name="testEditMateriaPrimaMateriaYaExiste" time="0.125"/>
    <testcase name="testEditMateriaPrimaBorraCompra" time="0.078"/>
    <testcase name="testEditMateriaPrimaArticuloInexistente" time="0.032"/>
    <testcase name="testEditMateriaPrimaMaterialInexistente" time="0.062"/>
  </testsuite>
  <testsuite name="com.proin.dataServer.comandos.TestEditCategoria" time="0.172">
    <testcase name="testEditCategoria" time="0.063"/>
    <testcase name="testEditCategoriaSinDescripcion" time="0.062"/>
    <testcase name="testEditCategoriaArticuloInexistente" time="0.047"/>
  </testsuite>
  <testsuite name="com.proin.dataServer.comandos.TestAltaVendible" time="0.344">
    <testcase name="testAltaVendible" time="0.031"/>
    <testcase name="testAltaVendibleBorrado" time="0.047"/>
    <testcase name="testAltaVendibleYaExistente" time="0.063"/>
    <testcase name="testAltaVendibleArticuloInexistente" time="0.046"/>
    <testcase name="testAltaVendibleId" time="0.032"/>
    <testcase name="testAltaVendibleIdYaExistente" time="0.031"/>
    <testcase name="testAltaVendibleIdArticuloInexistente" time="0.0"/>
    <testcase name="testAltaVendibleIdBorrado" time="0.094"/>
  </testsuite>
```

```
  <testsuite name="com.proin.dataServer.comandos.TestQuitarDestino" time="0.203">
    <testcase name="testQuitarDestino" time="0.062"/>
    <testcase name="testQuitarDestinoQueNoEsDestino" time="0.078"/>
    <testcase name="testQuitarDestinoAlmacenInexistente" time="0.016"/>
    <testcase name="testQuitarDestinoAcopioInexistente" time="0.047"/>
  </testsuite>
  <testsuite name="com.proin.dataServer.main.TestDataServerAntesCommandDispatcher"
time="0.015">
    <testcase name="dummy" time="0.015"/>
  </testsuite>
  <testsuite name="com.proin.dataServer.main.TestDataServer" time="0.0">
    <testcase name="dummy" classname="com.proin.dataServer.main.TestDataServer" time="0.0"/>
  </testsuite>
  <testsuite name="com.proin.dataServer.comandos.TestAvisoNivel" time="2.266">
    <testcase name="testAvisoNivelSinRegularizacion" time="0.172"/>
    <testcase name="testAvisoNivelMinimoRegularizacionEntrada" time="0.219"/>
    <testcase name="testAvisoNivelMinimoRegularizacionSalida" time="0.406"/>
    <testcase name="testAvisoNivelMaximoRegularizacionEntrada" time="0.141"/>
    <testcase name="testAvisoNivelMaximoRegularizacionSalida" time="0.141"/>
    <testcase name="testAvisoNivelMinimoConAcopioUsable" time="0.187"/>
    <testcase name="testAvisoNivelMinimoConAcopioEnUso" time="0.219"/>
    <testcase name="testAvisoNivelMaximoFinAcopio" time="0.312"/>
    <testcase name="testAvisoNivelMaximoConAcopioUsable" time="0.266"/>
    <testcase name="testAvisoNivelNivelInexistente" time="0.203"/>
  </testsuite>
  <testsuite name="com.proin.dataServer.comandos.TestAltaJuego" time="0.031">
    <testcase name="testAltaJuego" time="0.016"/>
    <testcase name="testAltaJuegoMismoCodigoDistintoCentro" time="0.0"/>
    <testcase name="testAltaJuegoCodigoRepetido" time="0.015"/>
  </testsuite>
  <testsuite name="com.proin.dataServer.comandos.TestSalidaMaterial" time="0.766">
    <testcase name="testSalidaMaterialUnaEntradaParcial" time="0.203"/>
    <testcase name="testSalidaMaterialUnaEntradaTotal" time="0.141"/>
    <testcase name="testSalidaMaterialVariasEntradasParcial" time="0.203"/>
    <testcase name="testSalidaMaterialUnaEntradaNegativo" time="0.125"/>
    <testcase name="testSalidaMaterialSinEntradaNegativo" time="0.094"/>
  </testsuite>
  <testsuite name="com.proin.dataServer.comandos.TestEditBocaYMedidor" time="0.734">
    <testcase name="testEditBocaYMedidor" time="0.094"/>
    <testcase name="testEditBocaYMedidorSinModificaciones" time="0.078"/>
```

```
  <testcase name="testEditBocaYMedidorMedidorYaExiste" time="0.094"/>
  <testcase name="testEditBocaYMedidorBocaYaExiste" time="0.203"/>
  <testcase name="testEditBocaYMedidorBocaNoExiste" time="0.14"/>
  <testcase name="testEditBocaYMedidorMedidorNoExiste" time="0.125"/>
</testsuite>
<testsuite name="com.proin.dataServer.comandos.TestAltaMateriaPrimaConId" time="0.266">
  <testcase name="testAltaMateriaPrimaConId" time="0.032"/>
  <testcase name="testAltaMateriaPrimaConIdMismoArticulo" time="0.015"/>
  <testcase name="testAltaMateriaPrimaConIdCompra" time="0.016"/>
  <testcase name="testAltaMateriaPrimaConIdMismoId" time="0.109"/>
  <testcase name="testAltaMateriaPrimaConIdConCategoria" time="0.094"/>
</testsuite>
<testsuite name="com.proin.dataServer.comandos.TestAltaBoca" time="0.266">
  <testcase name="testAltaBoca" time="0.219"/>
  <testcase name="testAltaBocaCodigoDosificadorRepetido" time="0.047"/>
</testsuite>
<testsuite name="com.proin.dataServer.comandos.TestAsignarFabricacion" time="0.656">
  <testcase name="testAsignarFabricacionNull" time="0.39"/>
  <testcase name="testAsignarFabricacion" time="0.266"/>
</testsuite>
<testsuite name="com.proin.dataServer.comandos.TestExisteStockActivo" time="0.359">
  <testcase name="testExisteStockActivo" time="0.078"/>
  <testcase name="testExisteStockActivoStockOtroMaterial" time="0.109"/>
  <testcase name="testExisteStockActivoSinStockActivo" time="0.094"/>
  <testcase name="testExisteStockMaterialInexistente" time="0.063"/>
  <testcase name="testExisteStockAlmacenInexistente" time="0.015"/>
</testsuite>
<testsuite name="com.proin.dataServer.comandos.TestEditBoca" time="0.406">
  <testcase name="testEditBoca" time="0.125"/>
  <testcase name="testEditBocaAlmacenInexistente" time="0.063"/>
  <testcase name="testEditBocaBocaInexistente" time="0.015"/>
  <testcase name="testEditBocaRepetida" time="0.203"/>
</testsuite>
<testsuite name="com.proin.dataServer.comandos.TestAsignacionMaterial" time="0.25">
  <testcase name="testAsignacionMaterial" time="0.188"/>
  <testcase name="testAsignacionMaterialAlmacenInexistente" time="0.062"/>
</testsuite>
<testsuite name="com.proin.dataServer.comandos.TestConsumo" time="1.844">
  <testcase name="testConsumoEntradaLoteUnico" time="0.266"/>
  <testcase name="testConsumoEntradaVariosLotes" time="0.391"/>
```

```
    <testcase name="testConsumoLoteVirtual" time="0.25"/>
    <testcase name="testConsumoEntradaLoteVirtual" time="0.328"/>
    <testcase name="testConsumoFabricacionInexistente" time="0.14"/>
    <testcase name="testConsumoBocaInexistente" time="0.204"/>
    <testcase name="testConsumoEntradaVariosLotesFabricacionInexistente" time="0.265"/>
  </testsuite>
  <testsuite name="com.proin.dataServer.comandos.TestAltaArticulo" time="0.016">
    <testcase name="testAltaArticuloIdGenerado" time="0.0"/>
    <testcase name="testAltaArticuloCodigoExternoRepetido" time="0.016"/>
  </testsuite>
  <testsuite name="com.proin.dataServer.comandos.TestAltaCentroProduccion" time="0.062">
    <testcase name="testAltaCentroProduccionCorrecto" time="0.015"/>
    <testcase name="testAltaCentroProduccionDescripcionRepetida" time="0.032"/>
    <testcase name="testAltaCentroProduccionCodigoExternoRepetido" time="0.015"/>
  </testsuite>
  <testsuite name="com.proin.dataServer.comandos.TestDesactivaElemento" time="0.125">
    <testcase name="testDesactivaElemento" time="0.063"/>
    <testcase name="testDesactivaElementoYaDesactivado" time="0.047"/>
    <testcase name="testDesactivaElementoInexistente" time="0.015"/>
  </testsuite>
  <testsuite name="com.proin.dataServer.comandos.TestFinCiclo" time="2.688">
    <testcase name="testFinCicloUnConsumo" time="0.297"/>
    <testcase name="testFinCicloVariosConsumos" time="0.485"/>
    <testcase name="testFinCicloVariosConsumosDeDistintosAlmacenes" time="0.453"/>
    <testcase name="testFinCicloVariosConsumosDeDistintosMateriales" time="0.5"/>
    <testcase name="testFinCicloVariosConsumosDeDistintosMaterialesConConsumosDeOtrosCiclos"
time="0.5"/>
    <testcase name="testFinCicloSinConsumos" time="0.14"/>
    <testcase name="testFinCicloFabricacionInexistente" time="0.0"/>
    <testcase name="testFinCicloDistintosFactoresMedidores" time="0.313"/>
  </testsuite>
  <testsuite name="com.proin.dataServer.comandos.TestAltaMateriaPrima" time="0.156">
    <testcase name="testAltaMateriaPrimaCompra" time="0.031"/>
    <testcase name="testAltaMateriaPrima" time="0.031"/>
    <testcase name="testAltaMateriaPrimaMismoArticulo" time="0.032"/>
    <testcase name="testAltaMateriaPrimaConCategoria" time="0.062"/>
  </testsuite>
  <testsuite name="com.proin.dataServer.comandos.TestAddNivelAlmacen" time="0.313">
    <testcase name="testAddNivelAlmacenVariasPruebas" time="0.234"/>
    <testcase name="testAddNivelAlmacenInexistente" time="0.016"/>
```

```
  <testcase name="testAddNivelAlmacenCentroProduccion" time="0.016"/>
  <testcase name="testAddNivelAlmacenIncorrectoPlanta" time="0.047"/>
</testsuite>
<testsuite name="com.proin.dataServer.comandos.TestPonerAcopioEnReposo" time="0.328">
  <testcase name="testPonerAcopioEnReposo" time="0.14"/>
  <testcase name="testPonerAcopioEnReposoYaEnReposo" time="0.063"/>
  <testcase name="testPonerAcopioEnReposoAcopioInexistente" time="0.125"/>
</testsuite>
<testsuite name="com.proin.dataServer.comandos.TestAltaPlanta" time="0.359">
  <testcase name="testAltaPlantaNormal" time="0.062"/>
  <testcase name="testAltaPlantaDescripcionRepetida" time="0.297"/>
</testsuite>
<testsuite name="com.proin.dataServer.comandos.TestAgregarDestino" time="0.688">
  <testcase name="testAgregarDestino" time="0.063"/>
  <testcase name="testAgregarDestinoVariosAlmacenes" time="0.375"/>
  <testcase name="testAgregarDestinoAlmacenInexistente" time="0.015"/>
  <testcase name="testAgregarDestinoAcopioInexistente" time="0.156"/>
  <testcase name="testAgregarDestinoExistenteComoOrigen" time="0.079"/>
</testsuite>
<testsuite name="com.proin.dataServer.comandos.TestPonerAcopioEnUso" time="1.468">
  <testcase name="testPonerAcopioEnUso" time="0.218"/>
  <testcase name="testPonerAcopioEnUsoYaEnUso" time="0.328"/>
  <testcase name="testPonerAcopioEnUsoNoEnReposo" time="0.297"/>
  <testcase name="testPonerAcopioEnUsoOrigenNoAgregado" time="0.157"/>
  <testcase name="testPonerAcopioEnUsoDestinoNoAgregado" time="0.14"/>
  <testcase name="testPonerAcopioEnUsoDestinoInexistente" time="0.094"/>
  <testcase name="testPonerAcopioEnUsoOrigenInexistente" time="0.109"/>
  <testcase name="testPonerAcopioEnUsoAcopioInexistente" time="0.125"/>
</testsuite>
<testsuite name="com.proin.dataServer.comandos.TestSetHabilitado" time="0.391">
  <testcase name="testSetHabilitado" time="0.188"/>
  <testcase name="testSetHabilitadoDesactivacion" time="0.203"/>
  <testcase name="testSetHabilitadoHabilitarSinAlmacen" time="0.0"/>
  <testcase name="testSetHabilitadoDeshabilitarSinAlmacen" time="0.0"/>
</testsuite>
<testsuite name="com.proin.dataServer.comandos.TestConversionUnidad" time="0.156">
  <testcase name="testConversionUnidadMismaMagnitud" time="0.016"/>
  <testcase name="testConversionUnidadMismaMagnitudConIdMaterial" time="0.031"/>
  <testcase name="testConversionUnidadMismaUnidad" ="0.0"/>
  <testcase name="testConversionUnidadMismaUnidadConIdMaterial" time="0.015"/>
```

```
  <testcase name="testConversionUnidadVolumenAMasa" time="0.032"/>
  <testcase name="testConversionUnidadMasaAVolumen" time="0.031"/>
  <testcase name="testConversionUnidadVolumenAMasaSinIdMaterial" time="0.016"/>
  <testcase name="testConversionUnidadMasaAVolumenSinIdMaterial" time="0.0"/>
  <testcase name="testConversionUnidadMaterialInexistente" time="0.015"/>
  <testcase name="testConversionUnidadMagnitudesIncompatibles" time="0.0"/>
</testsuite>
<testsuite name="com.proin.dataServer.comandos.TestDesasignacionMaterial" time="0.391">
  <testcase name="testDesasignacionMaterial" time="0.172"/>
  <testcase name="testDesasignacionMaterialSinCambios" time="0.109"/>
  <testcase name="testDesasignacionMaterialAlmacenInexistente" time="0.016"/>
  <testcase name="testDesasignacionMaterialMaterialInexistente" time="0.094"/>
</testsuite>
<testsuite name="com.proin.dataServer.comandos.TestAltaComprable" time="0.109">
  <testcase name="testAltaComprable" time="0.047"/>
  <testcase name="testAltaComprableYaExistente" time="0.047"/>
  <testcase name="testAltaComprableArticuloInexistente" time="0.015"/>
</testsuite>
<testsuite name="com.proin.dataServer.comandos.TestAltaCategoria" time="0.188">
  <testcase name="testAltaCategoria" time="0.094"/>
  <testcase name="testAltaCategoriaSinDescripcion" time="0.047"/>
  <testcase name="testAltaCategoriaArticuloInexistente" time="0.015"/>
  <testcase name="testAltaCategoriaDuplicada" time="0.032"/>
</testsuite>
<testsuite name="com.proin.dataServer.comandos.TestAltaAcopio" time="0.093">
  <testcase name="testAltaAcopio" time="0.031"/>
  <testcase name="testAltaAcopioNombreDuplicado" time="0.031"/>
  <testcase name="testAltaAcopioNombreCodigoYM2Duplicado" time="0.031"/>
</testsuite>
<testsuite name="com.proin.dataServer.comandos.TestAltaFabricacion" time="1.11">
  <testcase name="testAltaFabricacion" time="0.204"/>
  <testcase name="testAltaFabricacionAlmacenConMaterialYaAsignado" time="0.203"/>
  <testcase name="testAltaFabricacionAlmacenConOtroMaterialYaAsignado" time="0.234"/>
  <testcase name="testAltaFabricacionSinAlmacenNiMaterial" time="0.016"/>
  <testcase name="testAltaFabricacionSinAlmacen" time="0.031"/>
  <testcase name="testAltaFabricacionSinMaterial" time="0.125"/>
  <testcase name="testAltaFabricacionAlmacenInexistente" time="0.016"/>
  <testcase name="testAltaFabricacionMaterialInexistente" time="0.093"/>
  <testcase name="testAltaFabricacionSinUnidad" time="0.188"/>
</testsuite>
```

<testsuite name="com.proin.dataServer.comandos.TestAsignarJuego" time="0.39">
<testcase name="testAsignarJuegoMedidorInexistente" time="0.015"/>
<testcase name="testAsignarJuegoJuegoInexistente" time="0.016"/>
<testcase name="testAsignarJuego" time="0.063"/>
<testcase name="testAsignarJuegoRepetido" time="0.296"/>
</testsuite>
<testsuite name="com.proin.dataServer.comandos.TestQuitarOrigen" time="0.235">
<testcase name="testQuitarOrigen" time="0.079"/>
<testcase name="testQuitarOrigenQueNoEsOrigen" time="0.062"/>
<testcase name="testQuitarOrigenAlmacenInexistente" time="0.031"/>
<testcase name="testQuitarOrigenAcopioInexistente" time="0.063"/>
</testsuite>
<testsuite name="com.proin.dataServer.comandos.TestConsignaMaterial" time="0.156">
<testcase name="testConsignaMaterial" time="0.156"/>
</testsuite>
<testsuite name="com.proin.dataServer.comandos.TestEditMaquinaYMedidor" time="0.875">
<testcase name="testEditMaquinaYMedidor" time="0.203"/>
<testcase name="testEditMaquinaYMedidorSinModificaciones" time="0.188"/>
<testcase name="testEditMaquinaYMedidorMedidorYaExiste" time="0.125"/>
<testcase name="testEditMaquinaYMedidorMaquinaYaExiste" time="0.109"/>
<testcase name="testEditMaquinaYMedidorMaquinaNoExiste" time="0.141"/>
<testcase name="testEditMaquinaYMedidorMedidorNoExiste" time="0.109"/>
</testsuite>
<testsuite name="com.proin.dataServer.comandos.TestPeticionMaterial" time="0.344">
<testcase name="testPeticionMaterial" time="0.344"/>
</testsuite>
<testsuite name="com.proin.dataServer.comandos.TestExisteArticulo" time="0.031">
<testcase name="testExisteArticulo" time="0.015"/>
<testcase name="testExisteArticuloInexistente" time="0.016"/>
</testsuite>
<testsuite name="com.proin.dataServer.comandos.TestEditCentroProduccion" time="0.328">
<testcase name="testEditCentroProduccionCompleto" time="0.125"/>
<testcase name="testEditCentroProduccionANulos" time="0.109"/>
<testcase name="testEditCentroProduccionCodigoRepetido" time="0.063"/>
<testcase name="testEditCentroProduccionNombreRepetido" time="0.031"/>
</testsuite>
<testsuite name="com.proin.dataServer.comandos.TestExisteMaquina" time="0.016">
<testcase name="testExisteMaquina" time="0.016"/>
<testcase name="testExisteMaquinaInexistente" time="0.0"/>
</testsuite>

<testsuite name="com.proin.dataServer.comandos.TestAltaAlmacenExclusivo" time="0.265">
<testcase name="testAltaAlmacenExclusivo" time="0.125"/>
<testcase name="testAltaAlmacenExclusivoDescripcionRepetida" time="0.14"/>
</testsuite>
<testsuite name="com.proin.dataServer.comandos.TestExisteMateriaPrima" time="0.063">
<testcase name="testMateriaPrima" time="0.047"/>
<testcase name="testMateriaPrimaInexistente" time="0.016"/>
</testsuite>
<testsuite name="com.proin.dataServer.comandos.TestModificacionUnidades" time="0.312">
<testcase name="testModificacionUnidades" time="0.141"/>
<testcase name="testModificacionUnidadesSinCambios" time="0.171"/>
</testsuite>
<testsuite name="com.proin.dataServer.comandos.TestAltaAlmacenAgregadoSimple" time="0.438">
<testcase name="testAltaAlmacenAgregadoSimple" time="0.125"/>
<testcase name="testAltaAlmacenAgregadoSimpleContenidoEnAgregado" time="0.157"/>
<testcase name="testAltaAlmacenAgregadoSimpleNombreDuplicado" time="0.14"/>
<testcase name="testAltaAlmacenAgregadoSimpleContenedorInexistente" time="0.016"/>
</testsuite>
<testsuite name="com.proin.dataServer.comandos.TestSalidaConsigna" time="0.203">
<testcase name="testSalidaConsigna" time="0.203"/>
</testsuite>
<testsuite name="com.proin.dataServer.comandos.TestEntradaMaterial" time="0.844">
<testcase name="testEntradaMaterial" time="0.25"/>
<testcase name="testEntradaConStockNegativoResultadoNegativo" time="0.328"/>
<testcase name="testEntradaConStockNegativoResultadoPositivo" time="0.266"/>
</testsuite>
<testsuite name="com.proin.dataServer.comandos.TestBajaVendible" time="0.062">
<testcase name="testBajaVendible" time="0.031"/>
<testcase name="testBajaVendibleYaBorrado" time="0.031"/>
<testcase name="testBajaVendibleArticuloInexistente" time="0.0"/>
</testsuite>
<testsuite name="com.proin.dataServer.comandos.TestEditArticuloYMateriaPrima" time="0.766">
<testcase name="testEditArticuloYMateriaPrima" time="0.266"/>
<testcase name="testEditArticuloYMateriaPrimaSinCambios" time="0.234"/>
<testcase name="testEditArticuloYMateriaPrimaCreaCompra" time="0.235"/>
<testcase name="testEditArticuloYMateriaPrimaArticuloSinMateria" time="0.031"/>
<testcase name="testEditArticuloYMateriaPrimaArticuloInexistente" time="0.0"/>
</testsuite>
<testsuite name="com.proin.dataServer.comandos.TestModificacionDisciplina" time="0.969">
<testcase name="testModificacionDisciplina" time="0.187"/>

```
  <testcase name="testModificacionDisciplinaSinCambios" time="0.188"/>
  <testcase name="testModificacionDisciplinaStockNoActivo" time="0.266"/>
  <testcase name="testModificacionDisciplinaAlmacenSinStock" time="0.171"/>
  <testcase name="testModificacionDisciplinaMaterialInexistente" time="0.125"/>
  <testcase name="testModificacionDisciplinaAlmacenInexistente" time="0.032"/>
</testsuite>
<testsuite name="com.proin.dataServer.comandos.TestTraspaso" time="2.781">
  <testcase name="testTraspaso" time="0.562"/>
  <testcase name="testTraspasoUnidadesCompatibles" time="0.344"/>
  <testcase name="testTraspasoMagnitudesCompatibles" time="0.344"/>
  <testcase name="testTraspasoStockOrigenFinalNegativo" time="0.359"/>
  <testcase name="testTraspasoStockDestinoExistente" time="0.328"/>
  <testcase name="testTraspasoStockDestinoConOtroMaterial" time="0.344"/>
  <testcase name="testTraspasoOrigenSinStock" time="0.297"/>
  <testcase name="testTraspasoMismoAlmacen" time="0.203"/>
</testsuite>
<testsuite name="com.proin.dataServer.comandos.TestConsultaMasa" time="1.328">
  <testcase name="testConsultaMasaUnConsumo" time="0.437"/>
  <testcase name="testConsultaMasaVariosConsumos" time="0.438"/>
  <testcase name="testConsultaMasaVariosConsumosCiclosDistintos" time="0.453"/>
</testsuite>
<testsuite name="com.proin.dataServer.comandos.TestEditPlanta" time="0.594">
  <testcase name="testEditPlanta" time="0.156"/>
  <testcase name="testEditPlantaMismoCentro" time="0.157"/>
  <testcase name="testEditPlantaSinCambios" time="0.156"/>
  <testcase name="testEditPlantaCentroInexistente" time="0.125"/>
</testsuite>
<testsuite name="com.proin.dataServer.comandos.TestVincularAlmacen" time="0.0">
  <testcase name="testVincularAlmacenExclusivoConExclusivo" time="0.0" ignored="true"/>
</testsuite>
<testsuite name="com.proin.dataServer.comandos.TestEditAcopio" time="0.14">
  <testcase name="testEditAcopioTodosLosCampos" time="0.047"/>
  <testcase name="testEditAcopioTodoIgual" time="0.031"/>
  <testcase name="testEditAcopioYaExistente" time="0.062"/>
</testsuite>
<testsuite name="com.proin.dataServer.comandos.TestEditMedidor" time="1.219">
  <testcase name="testEditMedidorMismaMaquina" time="0.297"/>
  <testcase name="testEditMedidorCambioMaquina" time="0.313"/>
  <testcase name="testEditMedidorRepetido" time="0.234"/>
  <testcase name="testEditMedidorMedidorInexistente" time="0.172"/>
```

```
  <testcase name="testEditMedidorMaquinaDestinoInexistente" time="0.203"/>
</testsuite>
<testsuite name="com.proin.dataServer.comandos.TestDesuscripcion" time="0.0">
  <testcase name="testDesuscripcion" time="0.0"/>
</testsuite>
<testsuite name="com.proin.dataServer.comandos.TestAddAlmacenToAgregado" time="0.0">
  <testcase name="testMovimientosAlmacenExclusivoCorrectos" time="0.0" ignored="true"/>
  <testcase name="testMovimientosAlmacenAgregadoCorrectos" time="0.0" ignored="true"/>
  <testcase name="testMovimientosNoPermitidos" time="0.0" ignored="true"/>
</testsuite>
<testsuite name="com.proin.dataServer.comandos.TestConsumoSemiautomatico" time="2.922">
  <testcase name="testConsumoEntradaLoteUnico" time="0.485"/>
  <testcase name="testConsumoEntradaVariosLotes" time="0.593"/>
  <testcase name="testConsumoLoteVirtual" time="0.641"/>
  <testcase name="testConsumoEntradaLoteVirtual" time="0.547"/>
  <testcase name="testConsumoFabricacionInexistente" ime="0.297"/>
  <testcase name="testConsumoEntradaVariosLotesFabricacionInexistente" time="0.359"/>
</testsuite>
<testsuite name="com.proin.dataServer.comandos.TestTransmutacion" time="0.391">
  <testcase name="testTransmutacion" time="0.391"/>
</testsuite>
<testsuite name="com.proin.dataServer.comandos.TestEditArticulo" time="0.297">
  <testcase name="testEditArticulo" time="0.109"/>
  <testcase name="testEditArticuloSinCambios" time="0.125"/>
  <testcase name="testEditArticuloInexistente" time="0.0"/>
  <testcase name="testEditArticuloYaExiste" time="0.063"/>
</testsuite>
<testsuite name="com.proin.dataServer.comandos.TestExisteBoca" time="0.156">
  <testcase name="testExisteBoca" time="0.14"/>
  <testcase name="testExisteBocaInexistente" time="0.016"/>
</testsuite>
<testsuite name="com.proin.dataServer.comandos.TestMensajeFabricacion" time="0.297">
  <testcase name="testMensajeFabricacion" time="0.297"/>
  <testcase name="testMensajeFabricacionFabricacionInexistente" time="0.0"/>
</testsuite>
<testsuite name="com.proin.dataServer.comandos.TestAltaMedidor" time="0.312">
  <testcase name="testAltaMedidor" time="0.312"/>
</testsuite>
<testsuite name="com.proin.dataServer.comandos.TestExisteCentroProduccion" time="0.078">
  <testcase name="testExisteCentroProduccion" time="0.047"/>
```

```
  <testcase name="testExisteCentroProduccionInexistente" time="0.031"/>
</testsuite>
<testsuite name="com.proin.dataServer.comandos.TestAltaMaquina" time="0.078">
  <testcase name="testAltaMaquina" time="0.063"/>
  <testcase name="testAltaMaquinaCodigoDosificadorRepetido" time="0.015"/>
</testsuite>
<testsuite name="com.proin.dataServer.comandos.TestAnulaEntrada" time="1.954">
  <testcase name="testAnulaEntradaEntradaEliminada" time="0.329"/>
  <testcase name="testAnulaEntradaEntradaEliminadaStockPositivo" time="0.296"/>
  <testcase name="testAnulaPrimeraEntradaEntradaEliminadaStockPositivo" time="0.329"/>
  <testcase name="testAnulaEntradaTrasSalidaEntradaEliminadaStockPositivo" time="0.328"/>
  <testcase name="testAnulaEntradaRegularizacionStockPositivo" time="0.343"/>
  <testcase name="testAnulaEntradaRegularizacionStockNegativo" time="0.329"/>
</testsuite>
<testsuite name="com.proin.dataServer.main.TestComandos" time="0.0">
  <testcase name="testComandos" time="0.0"/>
</testsuite>
<testsuite name="com.proin.dataServer.comandos.TestExisteAlmacen" time="0.171">
  <testcase name="testExisteAlmacenID" time="0.171"/>
  <testcase name="testExisteAlmacenIDInexistente" time="0.0"/>
</testsuite>
<testsuite name="com.proin.dataServer.webInterface.TestConsumosViudos" time="0.829">
  <testcase name="testConsumosViudos" time="0.829"/>
</testsuite>
<testsuite name="com.proin.dataServer.comandos.TestEditAlmacenExclusivo" time="1.468">
  <testcase name="testEditAlmacenExclusivo" time="0.312"/>
  <testcase name="testEditAlmacenExclusivoSinCambios" time="0.234"/>
  <testcase name="testEditAlmacenExclusivoMismaPlanta" time="0.25"/>
  <testcase name="testEditAlmacenExclusivoAlmacenYaExiste" time="0.375"/>
  <testcase name="testEditAlmacenExclusivoPlantaInexistente" time="0.204"/>
  <testcase name="testEditAlmacenExclusivoAlmacenInexistente" time="0.093"/>
</testsuite>
<testsuite name="com.proin.dataServer.comandos.TestExistePlanta" time="0.125">
  <testcase name="testExistePlanta" time="0.11"/>
  <testcase name="testExistePlantaInexistente" time="0.015"/>
</testsuite>
<testsuite name="com.proin.dataServer.comandos.TestEditFabricacion" time="3.11">
  <testcase name="testEditFabricacion" time="0.219"/>
  <testcase name="testEditFabricacionAlmacenConMaterialYaAsignado" time="0.266"/>
  <testcase name="testEditFabricacionAlmacenConOtroMaterialYaAsignado" time="0.297"/>
```

```
    <testcase name="testEditFabricacionSinAlmacenNiMaterial" time="0.0"/>
    <testcase name="testEditFabricacionSetAlmacenYMaterial" time="0.218"/>
    <testcase name="testEditFabricacionCambioAlmacenYMaterial" time="0.422"/>
    <testcase name="testEditFabricacionCambioAlmacen" time="0.391"/>
    <testcase name="testEditFabricacionCambioMaterial" time="0.344"/>
    <testcase name="testEditFabricacionCambioUnidad" time="0.265"/>
    <testcase name="testEditFabricacionAlmacenInexistente" time="0.219"/>
    <testcase name="testEditFabricacionMaterialInexistente" time="0.219"/>
    <testcase name="testEditFabricacionSetAlmacenConMaterialNull" time="0.203"/>
    <testcase name="testEditFabricacionSetMaterialConAlmacenNull" time="0.047"/>
    <testcase name="testEditFabricacionSetUnidadConAlmacenYMaterialNull" time="0.0"/>
  </testsuite>
  <testsuite name="com.proin.dataServer.comandos.TestEditJuego" time="0.078">
    <testcase name="testEditJuego" time="0.015"/>
    <testcase name="testEditJuegoSinCambios" time="0.016"/>
    <testcase name="testEditJuegoMismoCodigoDistintoCentro" time="0.016"/>
    <testcase name="testEditJuegoCodigoRepetido" time="0.015"/>
    <testcase name="testEditJuegoInexistente" time="0.016"/>
  </testsuite>
  <testsuite name="com.proin.dataServer.comandos.TestAgregarOrigen" time="0.672">
    <testcase name="testAgregarOrigen" time="0.109"/>
    <testcase name="testAgregarOrigenVariosAlmacenes" time="0.297"/>
    <testcase name="testAgregarOrigenAlmacenInexistente" time="0.031"/>
    <testcase name="testAgregarOrigenAcopioInexistente" time="0.11"/>
    <testcase name="testAgregarOrigenExistenteComoDestino" time="0.125"/>
  </testsuite>
  <testsuite name="com.proin.dataServer.comandos.TestExisteJuego" time="0.0">
    <testcase name="testExisteJuego" time="0.0"/>
    <testcase name="testExisteJuegoInexistente" time="0.0"/>
  </testsuite>
  <testsuite name="com.proin.dataServer.main.TestDataServerComandoCommandDispatcher"
time="0.0">
    <testcase name="dummy" time="0.0"/>
  </testsuite>
  <testsuite name="com.proin.dataServer.comandos.TestSuscripcion" time="0.015">
    <testcase name="testSuscripcion" time="0.015"/>
  </testsuite>
  <testsuite name="com.proin.dataServer.comandos.TestSetHumedadInstantanea" time="1.157">
    <testcase name="testSetHumedadInstantanea" time="0.266"/>
    <testcase name="testSetHumedadInstantaneaExistente" time="0.25"/>
```

```
    <testcase name="testSetHumedadInstantaneaMaterialInexistente" time="0.172"/>
    <testcase name="testSetHumedadInstantaneaAlmacenInexistente" time="0.234"/>
    <testcase name="testSetHumedadInstantaneaAlmacenSinStock" time="0.235"/>
  </testsuite>
  <testsuite name="com.proin.dataServer.comandos.TestEditMaquina" time="0.031">
    <testcase name="testEditMaquina" time="0.031"/>
  </testsuite>
  <testsuite name="com.proin.dataServer.comandos.TestExisteAlmacenNombre" time="0.219">
    <testcase name="testExisteAlmacenNombre" time="0.219"/>
    <testcase name="testExisteAlmacenNombreInexistente" time="0.0"/>
  </testsuite>
</testrun>
```