UNIÓN INTERNACIONAL DE TELECOMUNICACIONES

UIT-T Z.352

SECTOR DE NORMALIZACIÓN
DE LAS TELECOMUNICACIONES
DE LA UIT

(03/93)

LENGUAJE HOMBRE-MÁQUINA

TÉCNICAS DE ESPECIFICACIÓN DE LA INTERFAZ HOMBRE-MÁQUINA ORIENTADA A DATOS – ALCANCE, MÉTODO Y MODELO DE REFERENCIA

Recomendación UIT-T Z.352

(Anteriormente «Recomendación del CCITT»)

PREFACIO

El Sector de Normalización de las Telecomunicaciones de la UIT (UIT-T) es un órgano permanente de la Unión Internacional de Telecomunicaciones. El UIT-T tiene a su cargo el estudio de las cuestiones técnicas, de explotación y de tarificación y la formulación de Recomendaciones al respecto con objeto de normalizar las telecomunicaciones sobre una base mundial.

La Conferencia Mundial de Normalización de las Telecomunicaciones (CMNT), que se reúne cada cuatro años, establece los temas que habrán de abordar las Comisiones de Estudio del UIT-T, que preparan luego Recomendaciones sobre esos temas.

La Recomendación UIT-T Z.352, preparada por la Comisión de Estudio X (1988-1993) del UIT-T, fue aprobada por la CMNT (Helsinki, 1-12 de marzo de 1993).

NOTAS

1 Como consecuencia del proceso de reforma de la Unión Internacional de Telecomunicaciones (UIT), el CCITT dejó de existir el 28 de febrero de 1993. En su lugar se creó el 1 de marzo de 1993 el Sector de Normalización de las Telecomunicaciones de la UIT (UIT-T). Igualmente en este proceso de reforma, la IFRB y el CCIR han sido sustituidos por el Sector de Radiocomunicaciones.

Para no retrasar la publicación de la presente Recomendación, no se han modificado en el texto las referencias que contienen los acrónimos «CCITT», «CCIR» o «IFRB» o el nombre de sus órganos correspondientes, como la Asamblea Plenaria, la Secretaría, etc. Las ediciones futuras en la presente Recomendación contendrán la terminología adecuada en relación con la nueva estructura de la UIT.

2 Por razones de concisión, el término «Administración» se utiliza en la presente Recomendación para designar a una administración de telecomunicaciones y a una empresa de explotación reconocida.

ÍNDICE

RESUMEN

Esta Recomendación proporciona un marco para la especificación de HMI armonizados.

La Recomendación señala la importancia de identificar el área de aplicación correcta y la gran repercusión que esto tendrá en el diseño final del HMI. En consecuencia, se propone un enfoque orientado a datos, en lugar de un enfoque funcional.

Se presenta y explica un modelo de referencia de tres capas para el HMI. Este modelo de referencia permite la centralización de definiciones de datos para HMI y la derivación de presentaciones externas de una manera armonizada.

El Anexo A proporciona directrices sobre la forma de llevar a cabo el desarrollo de los HMI de acuerdo con el modelo de referencia y el formalismo elegidos. El método de desarrollo no es un proceso de desarrollo de tipo «caída de agua». Este anexo también ofrece un marco para el correcto diseño de los datos desde el punto de vista del HMI.

El Apéndice I contiene directrices adicionales para el diseño de datos.

La correspondencia entre el modelo de referencia del HMI y la arquitectura funcional de la RGT está en estudio.

Entre las áreas que guardan relación con esta Recomendación cabe citar:

- correspondencia con las especificaciones RGT;
- normalización del soporte lógico.

Recomendación Z.352

TÉCNICAS DE ESPECIFICACIÓN DE LA INTERFAZ HOMBRE-MÁQUINA ORIENTADA A DATOS – ALCANCE, MÉTODO Y MODELO DE REFERENCIA

(Helsinki, 1993)

1 Alcance de la técnica de especificación

El objetivo de esta técnica de especificación es especificar los datos vistos en el HMI, así como la gramática de estos datos. La Recomendación no tiene por objetivo definir el modo de funcionamiento interno de las redes de telecomunicación.

Los usuarios finales utilizan la especificación de los datos antes de la realización:

- para evaluar en qué consiste la aplicación;
- para asegurar que su gramática se comprende correctamente;
- para asegurar que su terminología está bien elegida; y
- como una parte genérica de un contrato para las realizaciones.

Las especificaciones deben ser también aplicables para:

- dar orientaciones a los usuarios sobre la estructura admisible u obligatoria de los datos, y las explicaciones de los datos;
- acceder a las «instancias» de los datos para esa aplicación.

Por esta razón, las especificaciones deben ser legibles para los diseñadores y los usuarios finales, y aplicables a ambos.

Los diseñadores pueden ser diseñadores de HMI y diseñadores de soporte lógico. Los usuarios finales pueden ser operadores OA&M dentro de las administraciones y clientes externos que administran sus propios servicios.

2 Método

2.1 Datos por oposición a funciones

Las especificaciones funcionales identifican y descomponen las tareas que se ejecutan en diferentes áreas funcionales. Esta descomposición de actividades no distingue los objetos a que son aplicables las funciones ni la manera en que estos objetos son manipulados.

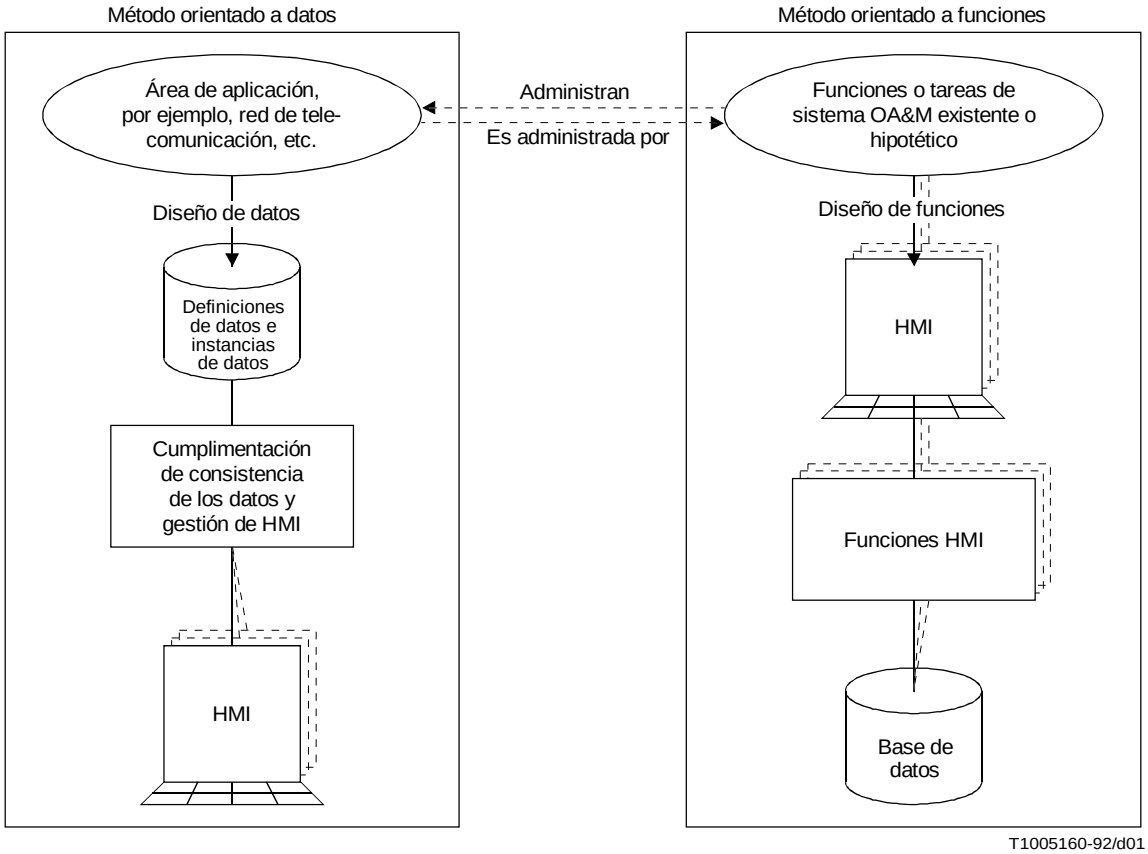
Los métodos orientados a datos se basan principalmente en la identificación de los datos a que son aplicables las funciones, independientemente de las propias funciones. Los medios para la manipulación de los datos pueden usualmente definirse de una manera genérica, independientemente de los diferentes tipos de datos que son identificados. Es muy importante reconocer que los dos métodos tienden a estudiar dos universos diferentes de acontecimientos. Véase la Figura 1. Los métodos orientados a funciones estudian y descomponen las tareas que van a ser ejecutadas en una organización. Los métodos orientados a datos estudian y descomponen el área de aplicación que es administrada por la organización. Como resultado del método orientado a datos se puede obtener una visión más coherente de los datos disponibles para la presentación al ser humano.

2.2 Consecuencias

Las implicaciones de la elección de uno de estos dos métodos son profundas. Las funciones o tareas de cualquier organización pueden cambiar más rápidamente que los datos manipulados por dichas funciones o tareas.

Los métodos orientados a funciones tienden a analizar un sistema de información antiguo o hipotético. Esto hace que, inevitablemente, las características existentes y las preconcepciones de los diseñadores sean conservadas y reflejadas en el diseño del futuro sistema.

Esta observación tiene varias implicaciones adicionales.



NOTAS

1 En el método orientado a funciones, los analistas estudian y descomponen las tareas realizadas por la organización. Para cada tarea se identifican y especifican interfaces hombre-máquina. Esto se ilustra por el «diseño de funciones» señalado con una flecha en la parte derecha de la figura. El sistema final orientado a funciones es realizado desarrollando un proceso para cada función HMI individual, mediante el empleo de una o más bases de datos comunes.

2 En el método orientado a datos, los analistas estudian y diseñan definiciones de datos comunes para la totalidad del sistema de información. Esto se ilustra por el «diseño de datos» señalado con una flecha en la parte izquierda de la Figura. Los procesos que cumplimentan la coherencia de los datos pueden hacerse comunes a todos los datos. El HMI no tiene que estar separado para cada tarea, sino que puede ser común a varias tareas. Obsérvese que los dos métodos tienden a estudiar dos diferentes universos de acontecimientos, que se representan por los dos óvalos en la parte superior de la Figura.

FIGURA 1/Z.352

Métodos alternativos para el desarrollo de HMI

En los métodos orientados a funciones, las secuencias de tareas están fijadas de manera firme (dícese «congeladas») en el sistema realizado, en tanto que los métodos orientados a datos proporcionan más flexibilidad para los cambios en las secuencias de tareas.

Tanto la capacitación para el manejo como la utilización de sistemas diseñados de acuerdo con estos dos métodos diferentes tienden a ser muy diferentes. En los diseños orientados a funciones, los usuarios aprenden a utilizar las diferentes funciones del sistema, y aprenden solo implícitamente el universo de acontecimientos, es decir, el área de aplicación, y la estructura de los datos que están siendo manipulados por las funciones. En los métodos orientados a datos se hace hincapié en la comprensión del área de aplicación que está siendo administrada por el sistema y definida por la estructura de los datos del sistema; se concede un menor nivel de prioridad a la capacitación sobre las tareas a ejecutar.

Si bien el análisis de funciones puede ser útil para identificar necesidades de datos, este análisis no ayuda a identificar datos para funciones adicionales, para nuevos usuarios, ni tampoco a identificar datos inconexos. En consecuencia, el análisis de funciones puede constreñir, más bien que favorecer, la innovación y el estudio crítico de definiciones de datos existentes. Igualmente, si bien la descomposición de funciones puede ser útil para la identificación de necesidades de datos, ella no ofrece interés para la documentación final de los datos. No obstante, puede ser útil para tomar nota de las necesidades funcionales y de los motivos que han conducido a la identificación de datos en el proceso de definición de esos datos.

En los métodos orientados a funciones, los datos se identifican en el nivel más bajo de descomposición de las funciones. Los datos identificados pueden ser introducidos en diferentes funciones u obtenidos a la salida de las mismas. La identificación de datos por separado en cada función no trata ni afecta a la estructura general, el diseño y la definición de los datos. En consecuencia, las definiciones de datos pueden seguir existiendo no armonizadas, y seguir estando formuladas de tal manera que sean ineficaces en lo que respecta a su empleo por los usuarios finales, sin que éstos lo perciban.

En los métodos orientados a funciones, distintas fases de diseño tratan la agrupación de datos en estructuras de fichero lógicas, apropiadas para las funciones que van a ser ejecutadas. Estas fases de diseño tienen en cuenta el acceso a, y la manipulación de, los datos, pero por lo general no son capaces de reconsiderar el diseño global de datos para esa aplicación.

2.3 Diseño de datos

En los métodos orientados a datos, los datos se identifican independientemente de las funciones. Los analistas y diseñadores

- observan el área de aplicación que está siendo administrada por la organización,
- diseñan definiciones de datos para esa área de aplicación.

El objetivo primario de los métodos basados en datos es diseñar definiciones de datos eficientes para el área de aplicación elegida. Para el diseño de datos se requiere, y hay que adquirir, un conocimiento profundo del área de aplicación que se está administrando.

Las definiciones de datos son importantes porque pueden ayudar a la organización a comprender la manera de realizar las tareas.

3 Modelo de referencia

3.1 Visión de conjunto

El método orientado a datos permite particionar todas las especificaciones HMI y todo el soporte lógico en una arquitectura estratificada (es decir, organizada en capas):

- La capa externa se hace cargo de la presentación y manipulación de datos. Se ocupa también de la correspondencia con los datos de aplicación, y de formación de subconjuntos de los datos de aplicación.
- La capa de aplicación es la capa del modelo de referencia HMI que se ocupa de la definición de datos y su comportamiento.
- La capa interna está fuera del ámbito del HMI. Se supone que esta capa se encarga del almacenamiento interno, acceso, realización y comunicación de datos, y de su comportamiento.

Cada capa de la arquitectura estratificada está particionada en esquemas, procesos y poblaciones. Los datos de cada capa son mapeados a datos de capas adyacentes solamente. Un esquema contiene las definiciones de datos, incluidas las limitaciones y reglas de derivación para los correspondientes datos de población. Una población contiene las instancias de datos que son cumplimentadas por un proceso de acuerdo con las reglas expresadas en un esquema correspondiente. En la Figura 2 sólo se representan algunos aspectos de las poblaciones [HMI y Base(s) de datos]. Un proceso lleva a efecto la cumplimentación de las reglas encontradas en un esquema sobre instancias de datos en una población correspondiente.

La capa de aplicación es un recurso centralizado de datos de aplicación y de su comportamiento. La especificación de la terminología y la gramática comunes de todos los datos HMI no está dispersada en varias funciones internas, sino que se define de manera no redundante en el esquema de aplicación.

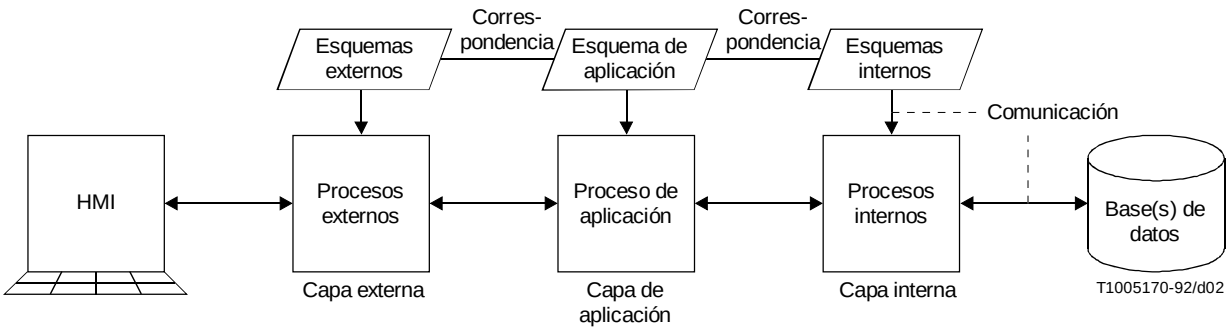


FIGURA 2/Z.352

La arquitectura de tres capas

3.2 Alcance

El alcance del modelo de referencia del trabajo HMI se ilustra en la Figura 3. Se utiliza exclusivamente la arquitectura de tres capas para definir e ilustrar el alcance del HMI, sin que se pretenda imponer ninguna restricción a la realización.

Las tres capas son:

- la capa externa HMI;
- la capa de aplicación HMI;
- la capa interna HMI.

La capa de aplicación HMI es controlada por un esquema de aplicación, que, por ser un recurso centralizado, contiene la especificación de la estructura y del comportamiento dinámico de todos los datos HMI.

La capa externa HMI puede contener varios esquemas externos. Los esquemas externos especifican las sintaxis concretas detalladas (layout) en el HMI y los derechos de acceso para cada presentación.

Las correspondencias entre el esquema de aplicación y los esquemas externos enuncian la formación de subconjuntos y derivación para cada presentación.

La capa interna está fuera del ámbito del HMI. Se supone que esta capa se encarga del almacenamiento interno, acceso, realización y comunicación de datos, y de su comportamiento.

Los usuarios finales utilizan primordialmente (la presentación de) (instancias de) datos de población HMI, de acuerdo con las especificaciones del esquema externo.

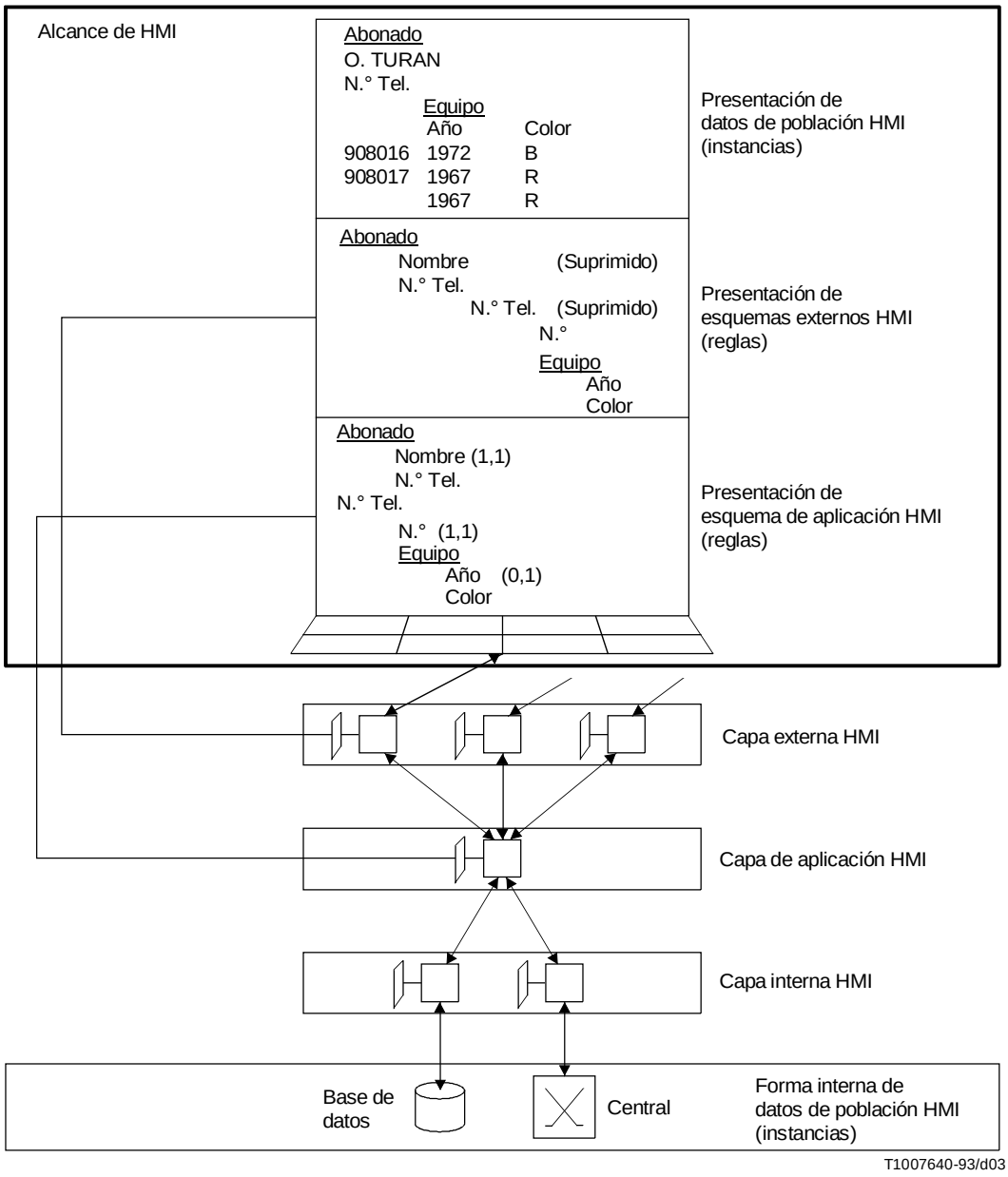
A fin de saber qué presentaciones están disponibles, los usuarios finales tienen que acceder a esquemas externos y sus contenidos.

Sin embargo, los usuarios finales necesitan también acceder a la definición de los datos, su comportamiento e interrelaciones. Esta información se obtiene del esquema de aplicación.

Los usuarios finales normalmente no tienen que acceder a los aspectos internos de los sistemas. Por tanto, el alcance del trabajo HMI se define como constituido por la presentación y la manipulación:

- de los datos de población HMI;
- de los esquemas externos HMI;
- del esquema de aplicación HMI;

Las exigencias de los usuarios finales en cuanto a la sintaxis y al comportamiento de la presentación de los datos de población HMI hace que surjan exigencias en cuanto a (la presentación de) los esquemas externos. Los (la presentación de los) esquemas externos hace que surjan exigencias en cuanto al (a la presentación del) esquema de aplicación, el cual tiene que ser capaz de especificar HMI (es decir, la unión de todos los HMI).



NOTAS

1 El alcance de HMI se ilustra en el interior del rectángulo superior. Además, de los datos de población, se muestra el alcance de HMI para incluir la presentación de las especificaciones en los esquemas externos (para la presentación de los datos de población HMI) y las especificaciones comunes en el esquema de aplicación (para la estructura y comportamiento de los datos de población HMI).

2 La forma interna y el almacenamiento de datos se muestran en el rectángulo inferior. Los usuarios finales no necesitan acceder a la forma interna de los datos.

3 La organización de la pantalla y la partición en ventanas se presentan a título de ejemplo para ilustrar el alcance de HMI solamente, y no se pretende que ilustren recomendaciones existentes ni que sugieran recomendaciones futuras. La pantalla puede contener también gráficos, pero éstos no se han incluido en el ejemplo. De la misma forma, la base de datos y la central se han incluido como ejemplos únicamente.

4 Dentro de los rectángulos que representan las capas, los pequeños paralelogramas representan esquemas y los pequeños cuadrados representan procesos. Las flechas bidireccionales representan datos de población que fluyen en ambos sentidos.

FIGURA 3/Z.352

Alcance del modelo de referencia

3.3 Subdivisión de la capa externa

El esquema externo se ocupa de la especificación de datos en el HMI para una presentación específica. Se proporciona la siguiente subdivisión del esquema externo (véase la Figura 4):

- El esquema de contenido especifica la estructura de los datos seleccionados y su relación para una presentación específica.
- El esquema de disposición (o esquema de layout) especifica la forma en que los datos se presentan al ser humano.

El contenido de los esquemas y las correspondencias entre los esquemas proporcionan especificaciones desde el punto de vista del ser humano (es decir, del diseñador y del usuario).

Estos contenidos y correspondencias no hacen ningún enunciado sobre la realización final del sistema. Por ejemplo, todas las especificaciones pueden ser compiladas en un solo bloque funcional.

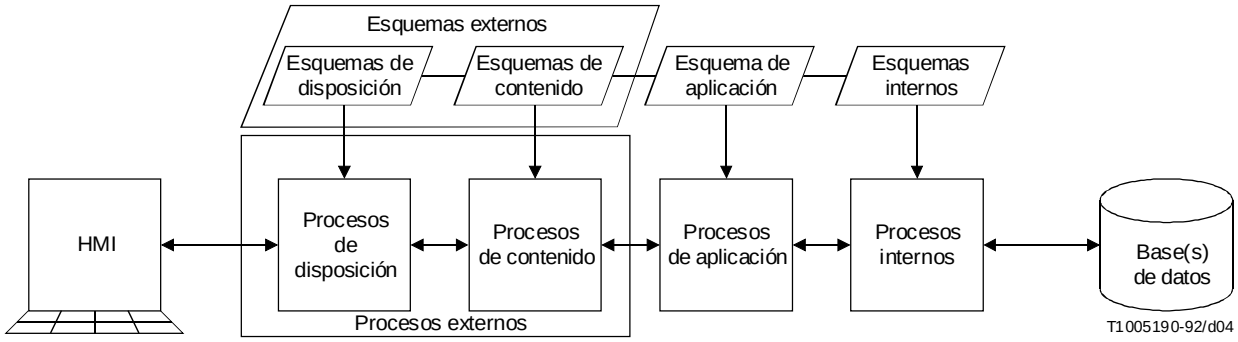


FIGURA 4/Z.352

Detalle de los esquemas internos

Anexo A

Directrices para los diseñadores de HMI

(Este anexo es parte integrante de la presente Recomendación)

A.1 Introducción

Este anexo contiene:

- subcláusula A.2: Método;
- subcláusula A.3: Diseño de datos.

A.2 Método

A.2.1 Finalidad

Este método tiene por finalidad

- identificar el alcance de especificaciones de la interfaz hombre-máquina;
- identificar las actividades estrictamente necesarias para diseñar interfaces hombre-máquina de acuerdo con el modelo de referencia de esta Recomendación;
- identificar las secuencias necesarias de estas actividades;
- dar algunas directrices para la obtención de diseños de datos apropiados.

Para información sobre el alcance, véase la cláusula 1. Las directrices sobre diseño de datos se presentan en A.3.

A.2.2 Proceso de desarrollo global

Esta subcláusula describe el proceso de desarrollo global de interfaces hombre-máquina. La arquitectura de tres capas del modelo de referencia presupone el proceso de desarrollo global, pero éste no es afectado por el formalismo elegido de la técnica de especificación de interfaz hombre-máquina.

El proceso de desarrollo global puede dividirse en las actividades siguientes:

1) estudio de la situación y análisis de las repercusiones;
2) planificación y coordinación de sistema;
3) desarrollo del esquema de aplicación;
4) desarrollo del esquema interno;
5) realización y creación de prototipos.

Se considera que la introducción y utilización de especificaciones y las realizaciones de acuerdo con las especificaciones están fuera del alcance del proceso de desarrollo global.

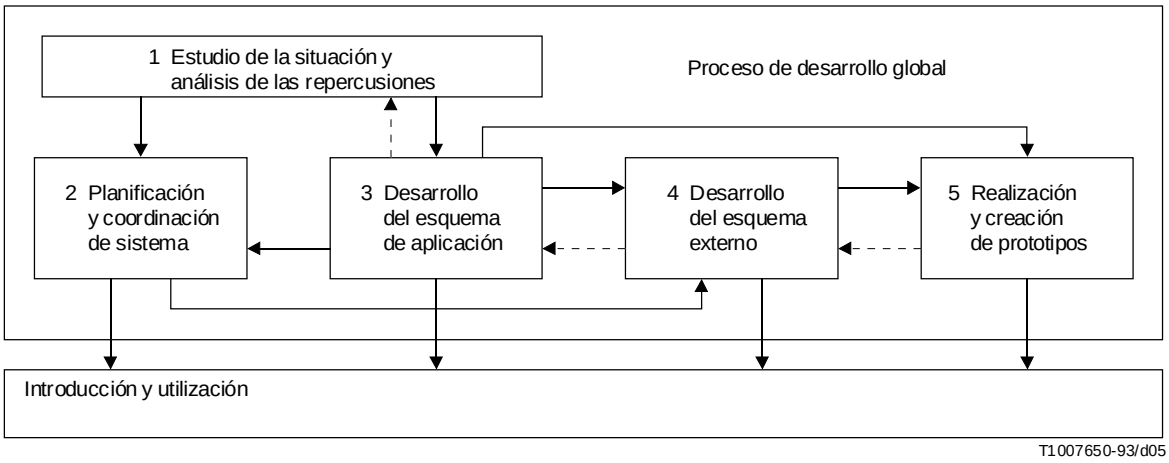
A continuación se hace una breve reseña de estas actividades:

- Actividad 1, Estudio de la situación y análisis de las repercusiones, no es objeto de esta serie de Recomendaciones. Puede incluir:
  - análisis del área de aplicación;
  - análisis de sistemas conexos, por ejemplo las organizaciones que intervienen y sus tareas;
  - desarrollo de diseños esquemáticos;
  - enunciado de objetivos del (de los) nuevo(s) sistema(s);
  - análisis de las consecuencias económicas, organizacionales, y en materia de personal, así como de otro tipo;
  - definición del alcance, límites e introducción del (de los) futuro(s) sistema(s).
- Actividad 2, Planificación y coordinación de sistema, no es objeto de esta serie de Recomendaciones. Puede incluir:
  - desarrollo de la organización propuesta de procedimientos de trabajo en la comunidad de usuarios.
- Actividad 3, Desarrollo de esquema de aplicación, es el punto principal de la técnica de especificación HMI orientado a datos. El proceso de desarrollo propuesto para el esquema de aplicación se describe en A.2.3.
- Actividad 4, Desarrollo de esquema externo, ha quedado en estudio. Incluirá:
  - especificación del contenido de vistas (esquemas de contenido) del esquema de aplicación;
  - specificación de HMI (Esquemas de disposición) concretos.

– Actividad 5, Realización y creación de prototipos, no es objeto de esta serie de Recomendaciones. Puede incluir:

  – desarrollo de soporte lógico de aplicación HMI;

  – verificación y validación del HMI desarrollado.



NOTA – Las flechas de trazo continuo indican el flujo principal de datos. Las flechas de trazo discontinuo indican retroalimentaciones. Las actividades 1, 2 y 5 no son objeto de esta serie de Recomendaciones.

FIGURA A.1/Z.352

Proceso de desarrollo global

A.2.3 Desarrollo del esquema de aplicación

Una consideración importante en el desarrollo de esquema de aplicación es la identificación del área de aplicación correcta. Este tema se discute en 2.

En un método orientado a datos, los analistas y diseñadores deben comenzar por observar y modelar las entidades fundamentales que existen en el área o dominio de aplicación que va a ser gestionado. Es probable que las definiciones basadas en estas entidades permanezcan estables, pues esas definiciones no pueden cambiar tan rápidamente como las tareas y rutinas que manejan datos relativos a las entidades.

Otra consideración importante es la identificación y comprensión de la perspectiva desde la que se está definiendo el área de aplicación, ya que diferentes perspectivas conducirán a diferentes esquemas de aplicación. Como ejemplo de esto cabe citar el área de aplicación gestión de tráfico de red. La gestión de tráfico puede visualizarse desde la perspectiva de la red (un administrador que observa el tráfico entre las centrales) o desde la perspectiva de la central (un administrador que observa el tráfico que está siendo manipulado por una sola central, y ve los circuitos con otras centrales desde la perspectiva de la primera central).

En el desarrollo de un esquema de aplicación para un área (dominio) de aplicación grande y compleja, es inevitable que haya que dividir el área de aplicación en subáreas, ya que no es factible desarrollar un solo esquema de aplicación que lo comprenda todo. La manera en que se efectúa la división del área de aplicación probablemente influya en la eficacia del proceso de desarrollo. Se sugiere que la división se efectúe de manera que tienda a reducir la duplicación de datos entre esquemas de aplicación para diferentes subáreas. El proceso de dividir el área de aplicación en subáreas debe verse como un proceso iterativo. Probablemente este proceso será afectado por los resultados de otras actividades posteriores.

La actividad 2, Desarrollo del esquema de aplicación, puede dividirse en las subactividades que se indican más abajo y se ilustran en la Figura A.2:

2.1 desarrollo de clases de objeto y referencias;

2.2 desarrollo de clases de atributo;

2.3 desarrollo de comportamiento;

2.4 coordinación de datos.

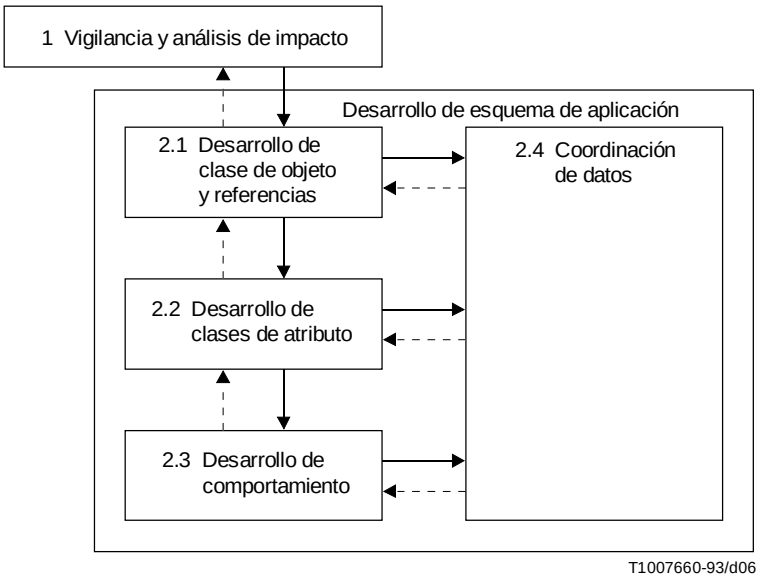


FIGURA A.2/Z.352

Desarrollo del esquema de aplicación

El objetivo de desarrollo de esquema de aplicación es definir las clases de objeto, referencias (relaciones), clases de atributo y comportamiento. El método de diseño consiste en identificar estos elementos en el orden dado, pero acepta que el proceso puede requerir varias iteraciones. Estas iteraciones permiten un proceso de refinamiento paso por paso. Por ejemplo, items de datos que son percibidos inicialmente como atributos pueden ser percibidos como objeto después de una investigación.

Actividad 2.1, Desarrollo de clases de objeto y referencias: el objetivo de esta actividad es analizar un área de aplicación particular para identificar las clases de objeto que van a ser gestionadas desde la perspectiva del HMI, y las referencias entre estas clases. Las clases de objeto pueden contener clases de objeto en una jerarquía. La documentación gráfica del esquema de aplicación es una valiosa herramienta para esta actividad.

El proceso de identificar clases de objeto y sus referencias implica la observación de instancias de objeto y sus relaciones. Las clases son creadas por un proceso de generalización, que implica la identificación de instancias con características comunes. El diseñador tiene que tomar decisiones en cuanto al grado de generalización que debe aplicarse.

Los atributos identificadores deben considerarse dentro de esta actividad. Véase el Apéndice I.

Algunas veces un esquema de aplicación sólo necesita contener especificaciones para un solo momento de tiempo. Sin embargo, a menudo, un esquema de aplicación puede necesitar también contener especificaciones relacionadas con estados pasados y/o futuros. En este caso, cuando la terminología independiente-del-tiempo ha sido adoptada, el manejo de los sucesos pasados (historia) o los previstos para el futuro puede tratarse como una extensión de las definiciones de datos independientes-del-tiempo. Como otra posibilidad, el manejo del tiempo puede incluirse en los diseños iniciales de datos. La adición de prestaciones dependientes-del-tiempo a las definiciones de datos probablemente influya en la definición de limitaciones que se efectúa dentro de la actividad 2.3.

Actividad 2.2, Desarrollo de clases de atributo: cuando se hayan identificado las clases de objeto, se definirán las siguientes propiedades:

- clases de atributo, que definen las características peculiares de una clase de objeto, por ejemplo las características operativas que controlan la manera de operar el recurso, la aptitud de un recurso para proporcionar un servicio a un usuario final;
- clases de atributo que contienen clases de atributo subordinadas, por ejemplo un nombre de abonado constituido por varias partes;
- clases de valor y gamas de clases de valor, admisibles;
- sintaxis admisibles para clases de valor complejo, de atributos.

Actividad 2.3, Desarrollo de comportamiento: en esta actividad se define lo siguiente:

- limitaciones sobre los valores y otros datos;
- reglas de derivación para nuevos datos a partir de datos existentes.

Esto puede efectuarse utilizando el constructivo función del formalismo.

Actividad 2.4, Coordinación de datos: la coordinación de datos ha quedado en estudio. Puede incluir

- comparación de definiciones de datos de diferentes subáreas; identificación y resolución de superposiciones e incoherencias;
- actualización de un diccionario común.

A.3 Diseño de datos

A.3.1 Introducción

Esta subcláusula ofrece un marco para el diseño de datos, y en el Apéndice I se ofrecen directrices adicionales.

A.3.2 Marco

Debe destacarse que el diseño de datos es a menudo un trabajo de carácter original, a diferencia de la realización que a menudo puede ser un trabajo de rutina. En consecuencia, la cantidad de trabajo que se necesita para alcanzar cierto grado de calidad de diseño de datos puede ser relativamente imprevisible. El proceso de desarrollo no es una transformación mecánica de requisitos que han sido fijados y convenidos, sino una investigación de un entorno abierto. El éxito de esta tarea depende de que se tenga un conocimiento adecuado del área de aplicación, de que se pueda prever el impacto de la elección de distintas alternativas, y de que se tenga la aptitud para revisar, generalizar y sistematizar diversos fenómenos en el área de aplicación.

La técnica de especificación del HMI especifica la forma común que deben tener todas las especificaciones, pero no proporciona el fundamento para el área de aplicación particular. Los diseños de datos pueden ajustarse a la forma requerida y, no obstante eso, ser diseños defectuosos desde el punto de vista de los usuarios finales.

Los diseñadores de esquemas de aplicación/definiciones de datos para HMI deben tener presente:

a) las instancias de las definiciones que están autorizadas y las que están implicadas;

b) las presentaciones HMI de estas instancias que están autorizadas y las que están implicadas.

Las clases de instancias de datos definidas describen entidades en un área de aplicación. En consecuencia, los diseñadores tienen que observar e investigar esta área de aplicación cuando diseñen los datos. La identificación del área de aplicación correcta no es una tarea trivial, por ejemplo, es posible confundir la modelación del sistema de información antiguo con la modelación de su área de aplicación.

Las instancias y clases de datos serán utilizadas por algunos usuarios. En consecuencia, los diseñadores tienen que observar e investigar las necesidades de estos usuarios cuando vayan a diseñar datos. La identificación de los usuarios reales no es una tarea trivial, por ejemplo, los operadores de terminales sólo pueden ser comunicadores de información a los usuarios finales.

Además de la investigación de

- necesidades de usuarios;
- área de aplicación apropiada,

los diseñadores deberán tener

- un conocimiento del formalismo para el esquema de aplicación;
- directrices para diseños de datos apropiados, que satisfagan los apartados a) y b).

Este último punto incluirá también el conocimiento sobre el tratamiento del tiempo, la coordinación de aplicaciones, los modelos de referencia para sistemas de información, etc.

Las necesidades de los usuarios proporcionarán una orientación (informal) sobre la parte del área de aplicación que deberá ser descrita y administrada. La selección de fenómenos en el área de aplicación y la comprensión del área de aplicación dependen de los constructivos de lenguaje utilizados por el observador cuando registra sus observaciones. Los

fenómenos seleccionados finalmente, descritos en la capa de aplicación dependen del formalismo utilizado para dicha descripción. Los constructivos de lenguaje para la observación original y la descripción final pueden ser los mismos.

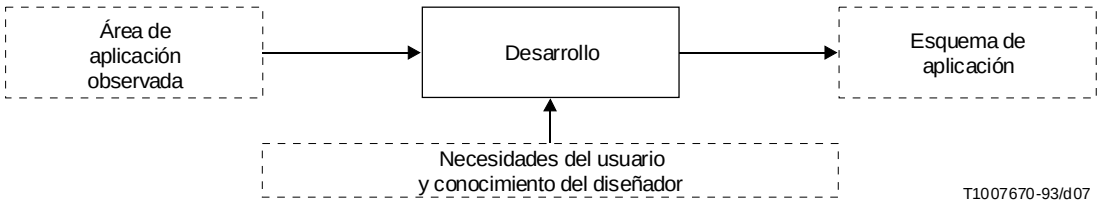


FIGURA A.3/Z.352

Desarrollo de datos HMI

Aunque las necesidades del usuario se toman como material de entrada para el desarrollo, se considera que el desarrollo es una función (correspondencia de muchos-a-uno) desde el área de aplicación observada al esquema de aplicación final. En consecuencia, no hay descripciones alternativas del mismo hecho. Este enunciado no tiene en cuenta el permiso para añadir nombres sinónimos alternativos en los pasos 7-10 que se indican más adelante.

El carácter de la función de desarrollo depende del formalismo utilizado para observar el área de aplicación y escribir el esquema de aplicación. Desafortunadamente, la función puede depender también de las decisiones tomadas en el método de desarrollo utilizado, y también de las definiciones de datos ya contenidas en el esquema de aplicación.

Una forma de realizar la correspondencia del área de aplicación observada al esquema de aplicación puede ser la siguiente:

1) seleccionar ejemplos de instancias de entidad del área de aplicación, incluida sus referencias;
2) diseñar propuestas de esquema de aplicación para cada ejemplo de instancia;
3) estudiar propuestas de esquema para tener en cuenta sus similitudes y tratar de unificarlas;
4) redefinir los ejemplos de acuerdo con nuevas propuestas de esquema;
5) crear un esquema integrado para todas las propuestas y finalizar la iteración;
6) seleccionar la porción pertinente de esquema global que habrá de proseguirse;
7) asignar nombres adecuados de usuario final al esquema final (objetos y referencias);
8) añadir y diseñar atributos de identificador;
9) añadir y diseñar otros atributos;
10) añadir y diseñar valores, limitaciones y reglas de derivación.

El último paso incluye la definición de datos derivados, que proporcionan una visión de conjunto de otros datos.

Obsérvese que los pasos 1) y 6) son funciones de selección puras (correspondencias de uno-a-uno) y no son «abstracciones»/«generalizaciones» de muchos-a-uno. Si permitimos que algunas instancias sean esquemas (es decir, prototipos) para otras instancias, el paso 2) puede considerarse vacío o, también como una correspondencia de uno-a-uno.

La razón para considerar la función de desarrollo desde el área de aplicación como constitutiva, básicamente, de un proceso de selección es que queremos ajustarnos a una correspondencia isomórfica entre una descripción y su mundo descrito. De no ser así, tenemos que conocer la función de desarrollo en su totalidad para interpretar correctamente los datos con respecto al mundo descrito pretendido; si sólo dispusiéramos de datos, no tendríamos un conocimiento cierto de la estructura del mundo descrito.

En algunas aplicaciones no hay un área de aplicación descrita, por ejemplo, en un sistema de correo electrónico, los datos pueden constituir el mundo real en sí. El tratamiento del tiempo puede estar integrado en los pasos anteriormente mencionados o ser añadidos como un paso separado.

Si bien la mayor parte de las definiciones de datos definen datos sobre un área de aplicación del sistema de información, por ejemplo sobre la red de telecomunicación, no hay nada que prohíba que la correspondencia de la descripción está anidada. Por consiguiente, el propio sistema de información puede estar descrito dentro del sistema EDP. En consecuencia, podemos describir procesos de tratamiento de información, flujos de información, flujos de control, etc. Algunos de los procesos descritos pueden ser automáticos, y otros manuales.

De acuerdo con lo expuesto en el párrafo anterior, reconocemos que, en principio, el análisis de un flujo de información no se diferencia en nada del diseño de definiciones de datos sobre el área de aplicación del sistema de información. Definir flujos de información no es más que describir otra área de aplicación. Las dos áreas de aplicación están relacionadas. Además, las dos áreas de aplicación pueden definirse utilizando la misma especie de formalismo. Esto no prohíbe el desarrollo de definiciones de datos relativamente genéricas que son aplicables a la definición de aspectos de los sistemas de información. Sin embargo, como los sistemas de información tienen que ser capaces de describir cualquier área de aplicación, y de tratar información sobre todos los aspectos en las áreas de aplicación, los constructivos de lenguaje para definir sistemas información sólo tienen que ser tan generales como los de un área de aplicación cualquiera.

Apéndice I

Directrices adicionales para el diseño de datos

(Este apéndice no es parte integrante de la presente Recomendación)

I.1 Generalidades

En lo que sigue se dan algunas directrices prácticas para el diseño de datos, sin entrar en los aspectos teóricos y filosóficos de las definiciones de datos. Las directrices se dividen en tres categorías:

A Diseño de clases (véase I.2)

B Diseño de identificadores (véase I.3)

C Datos derivados y nivel de detalle (véase I.4)

Los formatos de pantalla y las definiciones de datos consiguientes se presentan únicamente a título de ejemplo. Las etiquetas de clase están subrayadas, las etiquetas de atributo no lo están. Si las etiquetas de clase están representadas en la misma línea, los objetos pueden existir independientemente uno de otro, pero puede haber una referencia de un objeto de clase más a la izquierda a objetos de la clase más a la derecha. Si la etiqueta de clase más a la derecha está representada en la línea que sigue a la de la clase anterior, el cambio de línea indica contenencia. La contenencia de atributos dentro de clases de objeto y dentro de grupos de atributos se indica de forma similar mediante cambios de línea.

I.2 Diseño de clases

A1 Independencia existencial. La identificación de la existencia de objetos debe hacerse observando qué objetos pueden existir independientemente de cada uno de los demás.

Ejemplo

Una Línea de abonado por lo general está asociada a un Número de teléfono. En este ejemplo podemos suponer que existe un solo Número de teléfono para cada línea de abonado, y viceversa. En esta situación, lo primero que se nos ocurre es identificar la Línea de abonado mediante el Número de teléfono, utilizándolo como un atributo de identificación; esta solución, sin embargo es desaconsejada:

Línea de abonado

Numero de teléfono

06 809100

Sin embargo, la Línea de abonado puede ser asignada (y deberá ser identificada) antes de la asignación del Número de teléfono. En consecuencia, la Línea de abonado y el Número de teléfono deben tratarse como dos clases de objeto independientes, pero relacionadas. La relación puede tener la cardinalidad 1:1:

Línea de abonado	*Número de teléfono*
Nombre	Nombre
1234	06 809100

Las razones por las cuales se ha adoptado esta solución pueden encontrarse cuando se estudia la historia de la vida de las instancias de objeto. Si la Línea de abonado fue identificada solamente por el Número de teléfono, como se muestra en el ejemplo precedente, tendríamos que introducir dependencias, secuencias y trabajos de redenominación innecesarios, en las rutinas para administrar la red de telecomunicación.

Una redenominación de la Línea de abonado cuando se le asigna un Número de teléfono o cuando se le retira dicho Número de teléfono, puede también anular la posibilidad de llevar un historial de los objetos.

Otras razones para la adopción de esta solución podrán encontrarse cuando se estudien las referencias a los objetos. Por ejemplo, si se enuncia una referencia (por ejemplo de Par en cable al Nombre de Número de teléfono), la redenominación (de la Línea de abonado, identificada por diferentes Números de teléfono) puede dar lugar a una gran cantidad de trabajo manual o automático para actualizar las referencias cruzadas. Obsérvese que la discusión precedente no prohíbe la existencia de una relación derivada directa del Número de teléfono al Par (o los Pares) en cable en la Línea de abonados. Véase el ejemplo C3.

Teniendo en cuenta las preocupaciones señaladas, recomendamos que la referencia de Par en cable a Número de teléfono no se tome como una referencia primitiva (no derivada):

Número de teléfono	*Línea de abonado*	*Par en cable*
Nombre	Nombre	Nombre
06 809100	1234	A-B 1 15 B-C 3 1

A2 Atributos por oposición a clases de objeto. Cuando asignamos atributos a objetos, tenemos que asegurarnos de que esta información no pueda existir independientemente del objeto.

Ejemplo

Si queremos registrar, para cada Circuito, qué Grupo de circuitos está relacionado con él, y varios circuitos pueden ser relacionados con el mismo Grupo de circuitos, deberá considerarse el Grupo de circuitos como una clase de objeto separada (a), y no como un atributo solamente b)

a)

Circuito	*Haz de circuitos*
Nombre	Nombre
A-B 1	A-B 1
A-B 2	A-B 1
A-B 3	A-B 2
B-E 1	B-E 1

b)

Circuito	
Nombre	Grupo de circuitos
A-B 1	A-B 1
A-B 2	A-B 1
A-B 3	A-B 2
B-E 1	B-E 1

Obsérvese que en la alternativa a) podemos consultar directamente un Grupo de circuitos sin (tener que enunciar) una búsqueda a través de los circuitos.

A3 Desacoplamiento. Los hechos que sean diferentes deberán mantenerse aparte, en atributos separados, y no codificarse con los mismos valores.

Ejemplo

Supóngase que unas personas pueden cambiar con el tiempo, y pasar a ser, de personas de poca estatura a personas altas, y de personas delgadas a gruesas. En este caso, la información debe codificarse en dos atributos diferentes:

Estatura (con conjunto de valores)

– bajo;

– alto.

Corpulencia (con conjunto de valores):

– delgado;

– grueso.

Esto permitirá cambiar una (pieza de) información y escribir una lógica separada sin afectar a la otra. Un diseño de datos menos apropiado utilizaría un solo atributo:

Tamaño (con conjunto de valores):

– pequeño (bajo y delgado);

– redondo (bajo y grueso);

– largo (alto y delgado);

– grande (alto y grueso).

A4 Areas de aplicación interrelacionadas. Es muy importante identificar y delimitar el área de aplicación correcta para el Esquema de Aplicación; sin embargo, no está prohibido definir e interrelacionar datos de diferentes áreas de aplicación.

Ejemplo

Supóngase que se desea definir datos sobre encaminamiento de tráfico en la red de telecomunicación. Las clases de objeto del área de aplicación pueden ser Destino, Ruta, Haz de circuitos, etc. Esto no prohíbe definir una clase de objeto «Conjunto de parámetros» perteneciente al dominio del sistema de información o de la organización que administra la red. El «Conjunto de parámetros» puede identificarse por un estampado de hora y/o un nombre de usuario. Los objetos de esta clase pueden identificar las combinaciones de Destino, Ruta, Haz de circuitos, etc., que han sido puestos en servicio o registrados simultáneamente.

Obsérvese que, de esta manera puede definirse el sistema de información completo, incluidos, los procesos de tratamiento de información, las colas de información, las relaciones entrada-salida, el flujo de información, etc.

A5 Número de clases. Remplazando una clase por varias clases de objeto se imponen más limitaciones a los datos de población; fusionando varias clases en una sola clase se mitigan las limitaciones.

Ejemplo

a) Si Número de teléfono y Línea de abonado son fusionados para formar una clase, Número y la referencia entre las dos clases es remplazada por una referencia recursiva de Número a Número, el usuario está en libertad para decidir si relacionará Pares en cable directamente con Número (de teléfono) o indirectamente vía Número (de Línea de abonado) y la relación recursiva.

b) Si tanto el Número de teléfono como la Línea de abonado se mantienen/introducen como clases separadas, el usuario está obligado a relacionar los objetos en la forma deseada.

No haremos ninguna recomendación en cuanto a la solución que deba elegirse. Cualquiera de las dos soluciones puede ser adecuada en casos diferentes. Sin embargo, el diseñador de HMI debe ser consciente de las implicaciones de sus diseños.

A6 Presentaciones fusionadas. Las definiciones de clase deben hacerse de tal manera que los objetos requeridos puedan ser presentados convenientemente en una lista integrada.

Ejemplo

Una compañía tiene dos categorías de clientes: Personas y Organizaciones. En esta situación, la única presentación disponible de todos los clientes es:

Compañía	*Persona*	
Nombre	Nombre	
	Nombre de pila	Apellido
A	C	A
	H	A
	B	B
	R	J
	Organización	
	Nombre	
	AA	
	BB	
	CC	

Ejemplo

Si la compañía desea crear una lista ordenada integrada de todos sus Clientes, puede hacerlo de la siguiente forma:

Compañía		*Persona*	
	Cliente		
A	1	C	A
		Organización	
	2	AA	
	3	BB	
		Persona	
	4	H	A
	5	B	B
	6	R	J
		Organización	
	7	CC	

Obsérvese que las etiquetas de atributo se han suprimido y que solamente se muestran las etiquetas de clase de objeto.

Vemos que la lista se hace compleja debido a los diferentes diseños de identificadores de Personas y Organizaciones. Si Personas y Organizaciones se fusionan para formar una sola clase PersOrg, el problema desaparece. Para esto se necesita también que el primer nombre y el apellido se fusionen en un solo atributo largo.

Ejemplo

Compañía		*PersOrg*	
	Cliente		
A	1	C	A
	2	AA	
	3	BB	
	4	H	A
	5	B	B
	6	R	J
	7	CC	

Todos estos ejemplos representan posibilidades que pueden elegirse. Sin embargo, el diseñador debe ser consciente de las necesidades de los usuarios y de las implicaciones de cada solución.

I.3 Diseño de identificadores

B1 Identificadores locales. Si una instancia de objeto está contenida en un objeto superior, este objeto subordinado se identifica con relación al objeto superior.

Ejemplo

La clase de objeto Circuito está, por definición (en este ejemplo solamente) subordinada a la clase de objeto Grupo de circuitos. En consecuencia, en este ejemplo, un Circuito subordinado no puede existir independientemente de un Grupo de circuitos superior.

Circuito tiene el atributo identificante Nombre. Grupo de circuitos tiene un atributo identificante diferente denominado también Nombre.

El Nombre de Circuito tiene valores numéricos dentro del Grupo de circuitos superior. El Nombre de Grupo de circuitos tiene valores alfanuméricos. Por ejemplo:

Haz de circuitos	*Circuito*
Nombre	Nombre
A-B 1	1
A-B 2	2
A-C 1	1
B-E 1	1

Obsérvese que el Haz de circuitos A-B 1 tiene dos circuitos y que el Haz de circuitos A-C no tiene ningún circuito. En consecuencia, la presentación tiene forma de árbol y no de una tabla plana.

B2 Identificadores independientes. Si una clase de objeto no está definida de modo que está subordinada a otra, pero está relacionada con ella (quizás en el mismo nivel), los atributos identificantes tienen también que ser independientes uno de otro.

Ejemplo

Haz de circuitos	*Circuito*
Nombre	Nombre
A-B 1	A-B Z1
	A-B Z3
A-B 2	A-B Z2
A-C 1	
B-E 1	B-E Z1

Obsérvese que

– el Nombre de Circuito tiene que hacerse único añadiéndole los nombres de las dos centrales conectadas, por ejemplo A-B;

– en principio, las dos centrales en el Nombre de Circuito podrían ser diferentes de las que están en Nombre de Haz de circuitos;

– los números en el Nombre de Circuito son diferentes de los números en el Nombre de Haz de circuitos, aunque, por coincidencia, puedan parecer similares;

– los números en el Nombre de circuito ya no son asignados localmente al Nombre de Haz de circuitos; en consecuencia, al circuito en la tercera línea del ejemplo había que asignarle un número diferente del asignado en el ejemplo anterior, a fin de distinguir los circuitos en la primera y tercera líneas;

– se ha añadido una Z para indicar que todos estos circuitos son circuitos de tráfico; el carácter de par o impar de un número indica el sentido del tráfico.

Obsérvese que definir datos localmente a otros datos (como por ejemplo en B1) puede ser sumamente útil si los objetos pueden existir solamente cuando exista el objeto superior. Sin embargo, en B2, el nombre de un circuito deberá cambiarse cuando se cambie su utilización, lo que hace difícil rastrear el historial de un objeto determinado. Véanse también B8 y B9.

B3 Alcance de identificadores. Incluso los «identificadores globales» son locales a algún objeto «sistema». El objeto «sistema» debe identificarse y el alcance de los «identificadores globales» debe fijarse, para poder comunicar datos con el entorno del sistema.

Ejemplo

Este ejemplo ilustra dos perspectivas. En la primera se ha elegido una Central que será 'el sistema', y, en la segunda, la red completa será 'el sistema'. La central es en esta segunda perspectiva un componente de este 'sistema' más grande.

En la primera opción, todos los objetos fueron observados solamente desde una Central, o desde cada Central, por lo cual estos objetos (por ejemplo Haces de circuitos) pueden identificarse localmente a esta Central. En consecuencia, no es necesaria una coordinación para la asignación de identificadores de objetos fuera de la Central. En la segunda opción, Haz de circuitos es etiquetado y observado directamente desde la 'Red', por lo cual esta clase tiene que ser diferente de la clase Haz de circuitos vista desde la Central, ya que ninguna clase o instancia puede ser etiquetada o identificada y vista desde dos o más objetos o clases de objeto superiores. La referencia del Haz de circuitos de Red al Haz de circuitos de Central se denomina un Haz de circuitos terminados.

Sistema

Nombre	*Central*		
	Nombre	*Haz de circuitos*	
		Nombre	*Circuito*
			Nombre
Red	A	B 1	1
			2
		B 2	1
		C 1	
		E 1	1

	Haz de circuitos		
	Nombre	*Haz de circuitos terminados*	
		Nombre	
		Haz de circuitos de Central	
	A-B 1	A	B 1
	A-B 2	A	B 2
	A-C 1	A	C 1
	A-E 1	A	E 1

Como otra posibilidad, las referencias a Haces de circuitos de Central podrían hacerse utilizando 'denominación local regresiva':

Red

Nombre	*Haz de circuitos*		
	Nombre	*Haz de circuitos terminados*	*Central*
		Nombre	Nombre
Red	A-B 1	B1	A etc.

Esta presentación puede derivarse del mismo Esquema de Aplicación. Sin embargo, mientras el nombre Central en el ejemplo precedente es seleccionado desde el Haz de circuitos terminados, este Nombre de Central es seleccionado desde la propia Central. Esto permite el direccionamiento y la modificación del Nombre de la Central (no solo la referencia al mismo, como en el primer ejemplo), así como modificaciones de otras informaciones sobre la Central en esta presentación.

B4 Utilización de subclases. Se recomienda no utilizar la noción de subclase, sino considerar que diferentes clases tienen diferentes instancias y diferentes identificaciones.

Ejemplo

Compañía		*Persona*	*Individuo*
Nombre	*Cliente*		
	Nombre	Nombre	Nombre
IBM	1	John Smith	A
	2	John Smith	A
	3	Bill Jones	B
ND	1	Bill Jones	B

Los identificadores de la clase Persona no se aplican a todos los objetos de la clase Individuo, que puede incluir también Organizaciones, etc. En consecuencia, al Individuo se asigna un Nombre que armoniza los valores de identificador para todas sus instancias de objeto. Sin embargo, para cada Persona se requiere que haya un Individuo. Sin embargo, no se enuncia que los Roles Persona y Organizaciones de un Individuo se excluyan mutuamente.

Un Cliente puede considerarse una percepción de Persona desde el punto de vista de una Compañía, a condición de que sólo esté registrado un Cliente para cada Persona. En esta perspectiva podría existir la tentación de utilizar el Nombre de Persona para identificar el Cliente correspondiente. Sin embargo, si permitimos que el Cliente y la Persona existan independientemente uno de otra, deberá entonces asignárseles identificadores diferentes, incluso si están conectados por una referencia 1:1. Véase también el ejemplo A1.

Si extendemos las definiciones para permitir que haya varios clientes para cada Persona, sería muy desafortunado que hubiésemos antes utilizado el mismo identificador para ambas clases y tuviésemos que redenominar todos los clientes. Véase que el Cliente 1 y 2 corresponden a una misma Persona John Smith.

Si permitimos que cada Cliente sea visualizado como una compañía (filial) distinta (después de una fusión) se hace evidente la necesidad de identificar los clientes independientemente. Véase que el Cliente IBM 3 y ND 1 corresponden a una misma Persona Bill Jones.

Reconocemos que es importante diseñar los datos de tal manera que estén preparados para evolucionar, y que no provoquen confusiones.

Obsérvese que de acuerdo con lo anteriormente expuesto Persona no puede definirse de modo que sea una subclase de Individuo, porque el Nombre de Individuo no se hereda. Sin embargo, si los atributos heredables de Individuo son recogidos en una clase contenida (cardinalidad 1.1) de Individuo, Persona puede heredar de esta clase contenida. Lo mismo ocurre respecto a Cliente y Persona.

NOTA – La noción de subclases/herencia se define y utiliza de diferentes maneras en diferentes soluciones:

a) Un enunciado para copiar especificaciones de una clase a otra, mientras que las instanciaciones de las subclases y superclases son independientes; este es el método elegido en la Técnica de Especificación de HMI Orientado a Datos.

b) Cada instancia de la subclase es también un miembro de todas sus clases superiores, en forma prescrita por la teoría de conjuntos.

c) Un Individuo (por ejemplo John Smith) de una (super)clase (por ejemplo Persona) puede pertenecer simultáneamente a varias de sus subclases (por ejemplo Cliente y Empleado).

d) La pertenencia a una subclase (por ejemplo Mujer) puede excluir la pertenencia a otra subclase (por ejemplo Hombre) de una superclase común (por ejemplo Persona).

Para expresar las alternativas, un formalismo necesitaría diferentes enunciados o combinaciones de enunciados.

En B4 se recomienda no utilizar subclases para clases de objeto cuando se especifican HMI. Sin embargo, las subclases para copiar especificaciones pueden ser útiles para atributos.

En la solución a), es fácil que copiemos demasiado, si no se crean clases artificiales para ser copiadas.

En la solución b), tenemos la aptitud para relacionar dos instancias de una subclase con una instancia de la superclase.

La solución c) puede tratarse de una manera más apropiada por la noción de referencia, que falta en muchos métodos orientados a objeto.

La solución d) requiere enunciados condicionales, que tendrán que ser elaborados en futuras extensiones del formalismo.

B5 Atributos locales a clases de objeto. Si un atributo se utiliza dentro de una clase de objeto solamente, no hay necesidad de coordinar la asignación de valores de atributo a objetos fuera de esta clase. En los HMI no permitiremos que los identificadores abarquen varias clases de objeto.

Ejemplo

Si Haz de circuitos y Circuito son dos clases diferentes, sus atributos identificantes deben ser independientes entre sí, lo que permite utilizar los mismos valores para identificar diferentes objetos en clases distintas.

Haz de circuitos	*Circuito*
Nombre	Nombre
A-B 1	A-B 1
	A-B 2
A-B 2	A-B 3
A-C 1	
B-E 1	B-E 1

Obsérvese que cuando se hacen referencias entre dos objetos independientes, no relacionados por contenencia, hay que proporcionar el Nombre completo del objeto referenciado, incluso si una parte del Nombre referenciado es idéntica al Nombre de objeto actual.

B6 Identificadores complejos. En B1 hemos visto objetos que son denominados localmente a un objeto superior. Sin embargo, debemos reconocer que el mecanismo de identificación puede ser más complejo.

Ejemplo

Un Haz de circuitos puede ser identificado por las dos centrales (en orden alfabético) que él conecta.

Haz de circuitos	*Central*
Nombre	Nombre
A-B 1	A
	B

Vemos que esa información puede (y a veces tiene que) definirse de forma aparentemente redundante.

Obsérvese que hemos suprimido el atributo de referencia en el Haz de circuitos a Central porque una columna separada para el atributo de referencia no añadiría ninguna nueva información a la columna para el Nombre de las Centrales referenciadas. Esta supresión puede tener consecuencias para el diálogo de los usuarios finales. Véase el final de B4.

B7 Grupos de atributos. El atributo identificante de un objeto puede ser un grupo de atributos. En el HMI no estamos obligados a identificar objetos utilizando un solo atributo.

Ejemplo

Sistema

Nombre	*Circuito*		
	Nombre		
	Central 1	Central 2	N.°
PTN	A	B	1

Se recomienda clasificar las diferentes piezas de información en atributos separados y no codificar mucha información en un valor complejo.

B8 Estabilidad de identificadores. Los atributos de identificador pueden contener datos que dan información sobre las propiedades de los objetos, pero sólo deben contener datos que no puedan ser cambiados en el transcurso de la vida del objeto.

Ejemplo

Circuito

Nombre
A-B Z1

Obsérvese que, a diferencia de B7, donde el Nombre de atributo se descompone en partes, en este ejemplo lo que se descompone en partes es el valor del Nombre de atributo.

Si la utilización de un Circuito puede cambiarse haciéndola pasar de un circuito de tráfico (Z) a un circuito arrendado, un circuito de tráfico en otro sentido, un circuito utilizado para otros servicios, etc., sería muy poco inteligente tener esta información modificable en el atributo identificante.

El ejemplo muestra una desafortunada codificación de información en el valor de un solo atributo, mientras que Z informa sobre la utilización del Circuito, y la calidad de par o impar de los números informa sobre el sentido de tráfico. Véase también B2.

B9 Inestabilidad de identificadores. Algunas veces tenemos que incluir diseños de datos que son malos; esto sucede porque no podemos cambiar una utilización que es comúnmente aceptada.

Ejemplo

Persona

Nombre			
Nombre de pila	Apellido	Dirección	
		Calle	N.°
John	Smith	Kings road	11

En este caso la Dirección de la Persona puede cambiar, mientras que la Dirección es una parte del atributo identificante.

B10 Duplicados. Algunas veces puede ser necesario y conveniente, en el HMI, permitir el registro de datos que en ese momento no están identificados unívocamente.

Ejemplo

Dirección	*Persona*
Nombre	Nombre
Calle K, 5	John Smith John Smith

En un momento ulterior podemos recibir información que permita que las entradas lleguen a ser únicas. Es posible que, para tratar ficheros de textos y de gráficos, tengamos las mismas necesidades de tratar duplicados significativos.

Otro ejemplo lo constituye la presentación de iconos similares que representan diferentes instancias de central, sin proporcionar datos que distingan las centrales en el mapa, excepto por el posicionamiento de los iconos.

I.4 Datos derivados y nivel de detalle

C1 Granularidad por oposición a eficiencia. Cuando asignamos atributos a objetos, tenemos que asegurarnos de que los atributos proporcionan la flexibilidad necesaria y la eficiencia requerida.

Ejemplo

a)

Haz de circuitos

Nombre	*Circuito*	
	Nombre	Fecha
A-B 1	1	900223
	2	900224
A-B 2	1	900224
A-C 1		
B-E 1	1	900224

b)

Haz de circuitos

Nombre	Fecha	*Circuito*
		Nombre
A-B 1	900223	1
		2
A-B 2	900224	1
A-C 1	900224	
B-E 1	900224	1

En el ejemplo a), se puede registrar una Fecha diferente para el momento de que cada Circuito fue puesto en servicio. De esta forma se obtiene más flexibilidad para el registro de fechas individuales, pero se requiere mucho trabajo para registrar todas las fechas.

En el ejemplo b), las fechas de comienzo pueden registrarse para cada Haz de circuito solamente. Esta forma da menos flexibilidad para registros de informaciones individuales para cada Circuito, pero exige mucho menos trabajo de los operadores del HMI.

C2 Atributos derivados. Las definiciones de datos HMI no contendrán datos básicos solamente, sino también todos los datos derivados mostrados en los HMI.

Ejemplo

Cuando se consulta un Haz de circuitos, puede ser conveniente para los usuarios ver el «# circ. reposo» contenido en el Haz de circuitos, en lugar de tener que buscar en todos los circuitos para determinar los que están en reposo.

Haz de circuitos

Nombre	# circ. reposo	*Circuito*	
		Nombre	Situación
A-B 1	1	1	
		2	Reposo
A-B 2	0	1	
A-C 1	0		
B-E 1	1	1	Reposo

C3 Referencias derivadas. Las definiciones de datos HMI pueden contener referencias derivadas.

Ejemplo

Número de teléfono	*Línea de abonado*	*Par en cable*
Nombre	Nombre	Nombre
06 808011	9876543	A-B 31 B-C 15
	Par en cable	
	Nombre	
	A-B 31 B-C 15	

Para más detalles véase A1.

Una referencia derivada es una referencia que se ha derivado de otros datos por ejemplo, si hay un trayecto clase-1, ref-2, clase-2, ref-3, ..., clase-n, se puede introducir una referencia derivada directa ref-1-n desde la clase-1 hasta la clase-n (si está así definida), y de manera similar para las instancias correspondientes.

Obsérvese que los datos derivados no tienen que estar almacenados físicamente en la capa interna, sino que todos los datos utilizados en la capa externa tienen que estar definidos en la capa de aplicación. En consecuencia, las tres capas se comportan como un todo coherente.

C4 Objetos derivados. Las definiciones de datos HMI pueden contener objetos derivados.

Ejemplo

Enlace	*Circuito*
Nombre	Nombre
A-B	A-B 1 A-B 2 A-B 3
A-C	
B-E	B-E 1

La clase de objeto Enlace puede introducirse en forma de un objeto derivado para ofrecer una visión general de todos los circuitos entre dos centrales únicamente. Se supone que puede haber un solo Enlace entre dos centrales.