ESCUELA TÉCNICA SUPERIOR DE INGENIEROS
INDUSTRIALES Y DE TELECOMUNICACIÓN

UNIVERSIDAD DE CANTABRIA

Proyecto Fin de Carrera

MEJORA DEL ENTORNO DE VERIFICACIÓN DEL COMPUTADOR MIU DEL EUROFIGHTER TYPHOON: VIRTUALIZACIÓN DEL BANCO DE ENSAYOS

(Verification environment enhancement of the Eurofighter Thypoon MIU computer: Software bench virtualization)

Para acceder al Titulo de

INGENIERO DE TELECOMUNICACIÓN

Autor: José Rufo Martínez Rodríguez

Marzo -2013

Índice

1. Introducción 5
   1.1 EADS 5
   1.2 Proyecto Eurofighter 6
   1.3 Método de desarrollo software de comunicaciones en CASSIDIAN 8
   1.4 MIU 9
   1.5 Niveles de clasificación OTAN 10
   1.6 Banco de pruebas 10
   1.7 Descripción del proyecto 11
   1.8 Objetivos 11
   1.9 Estado del arte 12
2. Descripción de las herramientas de trabajo 13
   2.1 Visión global de MaTE 13
   2.2 Subsistemas de MaTE 13
      2.2.1 Data Repository 14
      2.2.2 Modelling Environment 15
      2.2.3 Test System 16
      2.2.4 Data Visualization 17
      2.2.5 Data Analysis 18
      2.2.6 Interfaz con el hardware 19
   2.3 TIGER 19
      2.3.1 Modos de funcionamiento de TIGER 20
      2.3.2 Software usado en TIGER 21
   2.4 MANDRIL 22
   2.5 PBA 23
3. Comunicación entre MIU y MIDS 24
   3.1 MIL-STD-1553 (STANAG 3838) 24
      3.1.2 Mensajes FxMs 27
      3.1.3 Mensajes BxMs 27
   3.2 STANAG 5516 29
      3.2.1 Mensajes J-Series 29
      3.3 EFEX y STANAG 3910 31
4. Proceso de migración 33

4.1 Configuración del banco de pruebas 33
4.2 Configuración del banco de pruebas virtual 34
4.3 Arquitectura software 35
4.3.1 PPSW y TSAP **35**
4.4 Desarrollo del trabajo 35
4.5 Migración del código de la MIU 36
4.6 Resultados de los test 39
4.7 PRC Format converter 39
4.8 Link 16 Simulator 39
4.9 Fase de test y verificación 39
4.9.1 Integración y test de cada subsistema **40**
4.9.2 Integración del sistema y pruebas de verificación **40**
4.9.3 Operación verificación y validación del sistema **40**
5. Conclusiones 42
6. Trabajo futuro 43
7. Acrónimos 44
8. Bibliografía y referencias 45

Índice de figuras

Figura 1 - Productos de EADS 6
Figura 2 - Reparto de fabricación 7
Figura 3 - Organización de venta 7
Figura 4 - modelo en V 8
Figura 5 - MIU 9
Figura 6 - MIDS-LVT 9
Figura 7 - MGUI de MaTE 14
Figura 8 - Data Repository 15
Figura 9 - Modelling Environment 16
Figura 10 - Test System 17
Figura 11 - Data Visualization 18
Figura 12 - Data Analysis 19
Figura 13 - Esquema de conexionado de TIGER 20
Figura 14 - Simulador en modo red 21
Figura 15 - Simulador en modo terminal 21
Figura 16 - Interfaz de TIGER 22
Figura 17 - PBA (Hardware) 23
Figura 18 - PBA (Software) 23
Figura 19 - Intercambio de mensajes en STANAG 3838 26
Figura 20 - Configuración del banco de pruebas 33
Figura 21 - Configuración del banco de pruebas virtual 34
Figura 22 – Orden de bits de MIU y MaTE 37

1. Introducción

1.1 EADS

El proyecto fin de carrera que tratamos ha sido realizado en la empresa EADS. European Aeronautic Defence and Space Company N.V. (EADS) es la unificación, formada en junio del 2000, de cuatro grandes compañías europeas del sector aeronáutico y aeroespacial;

- Aérospatiale-Matra (Francia)
- Dornier GmbH (Alemania)
- DaimlerChrysler Aerospace AG (DASA) (Alemania)
- Construcciones Aeronáuticas SA (CASA) (España)

EADS se compone de cuatro divisiones, cada una especializada en un mercado distinto, principalmente dentro del sector aeronáutico y aeroespacial.

Las divisiones que forman EADS son:

- Airbus: Dedicada a la fabricación de transporte aéreo comercial (bienes y personas) y militar. Uno de sus productos más conocidos es el A380, el avión de transporte de pasajeros más grande del mundo. Airbus cuenta con una división llamada Airbus military, encargada de los productos de uso militar. Destaca el modelo A400M.

- Eurocopter: Fabricante de Helicópteros tanto civiles, como el EC225 Super Puma, como militares. Uno de los productos más conocidos es el EC665 Tigre.

- Astrium: Especializada en sistemas espaciales. Orientada a satélites, servicios espaciales y transporte espacial.

- Cassidian: Participante del proyecto Eurofighter. Se dedica a la seguridad civil y militar, ofreciendo una amplia gama de productos, siendo el Eurofighter el más representativo e importante.

En la Figura 1 podemos ver productos representativos de cada filial.

Dentro de toda la organización, éste proyecto ha sido realizado en EADS-CASA, localizada en Getafe (Madrid). La división de la empresa en la que se ha desarrollado ha sido Cassidian.

Cassidian es la división de EADS que forma parte de las compañías desarrolladoras del Eurofighter (EF).

Figura 1 - Productos de EADS

1.2 Proyecto Eurofighter

El Eurofighter Typhoon se ha diseñado para ser el caza polivalente (multi-role) más avanzado del mundo. Se ha concebido tanto para misiones de supremacía aérea como acciones aire-superficie, incluso durante una misma misión (swing-role).

La compañía encargada de la coordinación del desarrollo del EF es Eurofighter Jagdflugzeug GmbH, llamada comúnmente Eurofighter GmbH[1]. El principal cliente es NATO Eurofighter and Tornado Management Agency (NETMA), que representa a los gobiernos de los cuatro países desarrolladores del EF (Alemania, Italia, Reino Unido y España).

Estos países fabrican de forma conjunta el avión. El trabajo de desarrollo del EF se acordó en el contrato previo a la fabricación de la forma:

- Reino Unido 33%
- Alemania 33%
- Italia 21%
- España 13%

El reparto del trabajo de producción está relacionado con el volumen de compra de aviones solicitado por cada país. La producción queda repartida:

- Reino Unido (BAE Systems) 37,5%

- Alemania (Cassidian Deutschland) 30%
- Italia (Alenia Aermacchi) 19,5%
- España (Cassidian España) 13%

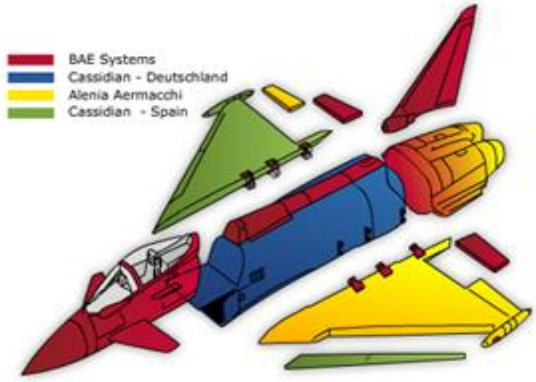

Figura 2 - Reparto de fabricación

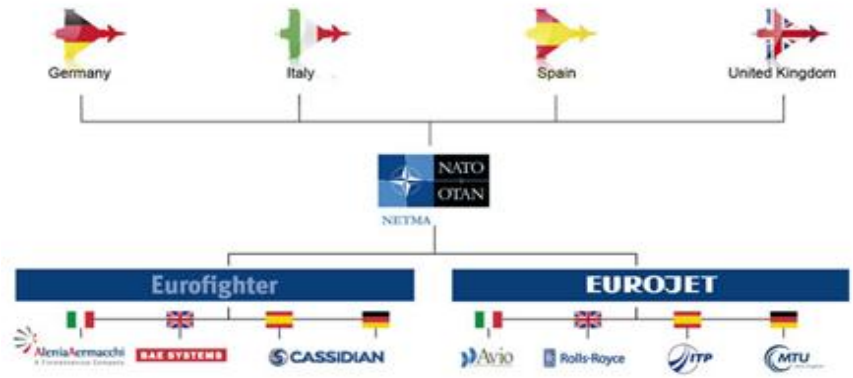

Figura 3 - Organización de venta

Además de los países involucrados en el desarrollo del avión (cada uno llamado Eurofighter Typhoon Partner Company (EPC)), el EF se ha vendido a otros países, como Austria y Arabia Saudí. Además participa en concursos y campañas de ventas de varios países.

En España se fabrica el ala derecha y los slats (pieza móvil de la parte delantera del ala). Además se desarrolla parte importante del software de las comunicaciones del avión. Es ahí donde se engloba este proyecto.

1.3 Método de desarrollo software de comunicaciones en CASSIDIAN

El método de desarrollo usado para el ciclo de vida del software es el llamado “Modelo en V”. Consiste en una planificación previa de las necesidades y una ejecución por niveles con sucesivos test de verificación.

Éste modelo se suele representar de forma grafica, donde se aprecia su forma de “V”. En el lado izquierdo está representada la planificación de necesidades y especificaciones. Al otro lado de la V se representa la integración y verificación de cada nivel.

Cada unidad se desarrolla de forma independiente y se puede probar su funcionalidad. Después de que la integración haya superado los test correspondientes, se evalúa el sistema completo y se comprueba que satisface los requerimientos iniciales.

Este modelo utiliza una serie de test estandarizados. Permite, por tanto, un ahorro de costes y asegura la calidad deseada. Además podemos hacer un seguimiento del proyecto en cada fase.

Esta planificación, seguimiento, control y evaluación se traduce, además de la reducción de costes señalada anteriormente, en un ahorro de tiempo y optimización del trabajo realizado.

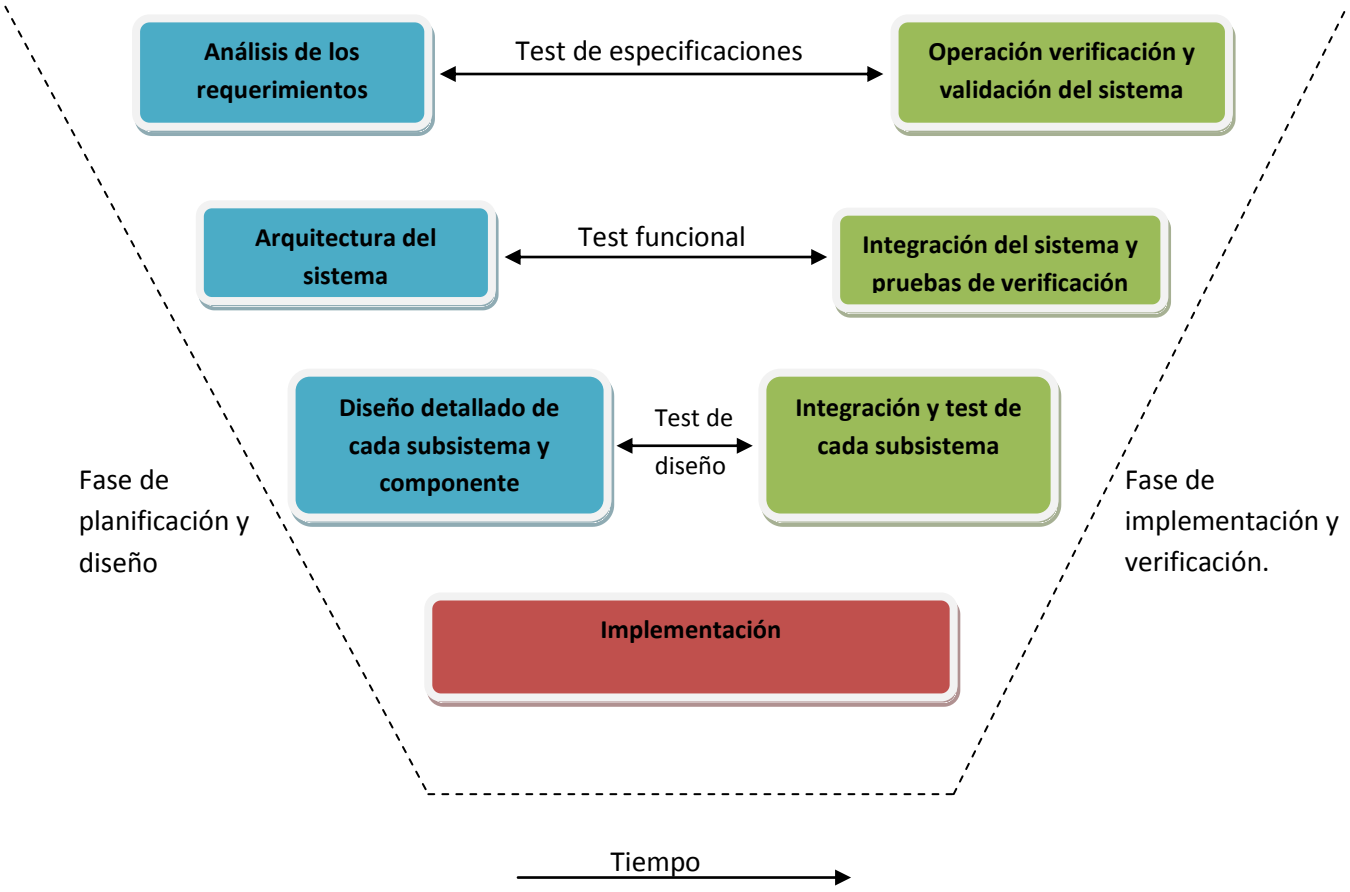

Figura 4 - Modelo en V

1.4 MIU

La MIDS Interface Unit (MIU) es la unidad de comunicaciones que vincula el Multifunctional Information Distribution System - System-Low Volume Terminal (MIDS-LVT) [2] con el resto de los sistemas de avión. Comenzaremos dando una breve descripción del MIDS-LVT.

Figura 5 - MIU

El MIDS-LVT es un dispositivo embebido en el avión. Es el encargado de enviar y recibir mensajes, pudiendo intercambiar en tiempo real información táctica, tanto con otros aviones como con tierra o mar. Para las comunicaciones utiliza los estándares de la OTAN "STANAG 5516" y "STANAG 3838", más conocido como MIL-STD-1553. Más adelante profundizaremos en estos estándares, ya que será necesario su manejo para el desarrollo del proyecto.

Cabe destacar que el MIDS-LVT no es único del EF, ya que es empleado en otros aviones como el Rafale, F16 y F18 además de portaaviones, cruceros y diversas unidades militares.

Figura 6 - MIDS-LVT

La MIU, por tanto, es capaz de comunicarse con el MIDS-LVT y procesar los mensajes tácticos. Gestiona la información y la distribuye a los distintos sistemas del avión de forma comprensible para éstos y por el puerto apropiado.

Para hacer esta función, la MIU dispone de un procesador y memoria. Es programable y está realizado con código multitarea Ada 95. Además dispone de varios puertos preparados para trabajar con varios tipos de buses, como el 1553 y el EFEX.

Englobado en el método de desarrollo mencionado en el apartado anterior, el computador MIU es un subsistema del proyecto Eurofighter. Por lo tanto se somete a una serie de pruebas de verificación, además de test de mejora y corrección de errores. Para su realización, las instalaciones de EADS-CASA disponen de un banco de pruebas (software bench) donde se realiza ese proceso. Dispone de varias MIUs reales, conectadas a varias computadoras que simulan el resto de elementos necesarios del avión.

1.5 Niveles de clasificación OTAN

La información sensible en materia de defensa está clasificada por niveles dependiendo su naturaleza y contenido. Para evitar la difusión de información que comprometa la seguridad y producción de los países miembros de la OTAN, existe un sistema de clasificación que limita ésta difusión.

Los niveles de seguridad son, en orden de importancia:

- COSMIC TOP SECRET
- NATO SECRET
- NATO CONFIDENTIAL
- NATO RESTRICTED
- NATO UNCLASSIFIED

La MIU, su código, los test de simulación, el banco de pruebas y gran parte de la documentación asociada están clasificados como NATO RESTRICTED. Esto implica que la información que se puede proporcionar y publicar está limitada. La memoria de este proyecto se ve afectada por esta limitación, impidiendo exponer información catalogada en ese nivel.

1.6 Banco de pruebas

Todas las modificaciones realizadas en el código de la MIU, para realizar una versión nueva, deben verificarse en el banco de pruebas. Es una plataforma que dispone del instrumental adecuado para llevar a cabo el test: ordenadores, analizador de buses, tarjetas de simulación, etc.

Cada vez que un desarrollador quiere verificar la viabilidad y funcionalidad del código realizado, debe acudir al banco de pruebas. Allí dispone de computadores con un entorno de simulación preparado para ejecutar el código realizado. Se conectan de esta forma al elemento real del avión con un entorno simulado. Se reproducen los escenarios en los que se quieren realizar las pruebas y se ejecuta el código. Durante la ejecución, la MIU devuelve las señales oportunas que son analizadas para verificar el éxito o fracaso de la prueba. Como no se disponen de más elementos reales del EF, es a través de un analizador de buses (PBA) donde se captura la respuesta. Una vez grabado el tráfico generado, se traduce mediante el software adecuado (programas que trataremos más adelante, “MANDRIL” para la información

relacionada con el MIDS-LVT y MaTE para la relacionada con el resto de computadores) a un formato más cómodo y eficaz para realizar el análisis.

De esta forma, un test realizado en el banco de pruebas corresponde a la fase de verificación del modelo en "V" usado para el desarrollo software. Se concluye con estas pruebas si el código cumple con las especificaciones requeridas.

Queda patente la gran importancia del banco de pruebas, ya que sin él sería inviable la realización de cualquier proyecto de esta envergadura. Este trabajo está íntimamente ligado al banco de pruebas, ya que supone realizar la misma funcionalidad en un entorno más accesible.

1.7 Descripción del proyecto

Este trabajo consiste en una alternativa al banco de pruebas en la que los desarrolladores de software puedan verificar sus modificaciones de código desde su PC.

Como estaba compuesto el banco de pruebas, se hacía imprescindible el uso de la MIU para realizar los test sobre la misma. Este proyecto consiste en modificar el código de la MIU para que pueda ser ejecutado desde un PC en un entorno de simulación, pudiendo generar una MIU migrada en la que cualquier PC sea capaz de simularla. Esta labor se realiza de forma que el sistema migrado mantenga el mismo comportamiento y funcionalidad que la MIU real.

Debemos tener presente, que en el banco de pruebas existen otros equipos necesarios para el desarrollo de los test. Por lo tanto, además de migrar el código de la MIU también será necesario proporcionar una alternativa, ejecutable desde un PC, a estos equipos.

1.8 Objetivos

Un problema del banco de pruebas es el elevado precio del material y de su mantenimiento. Por este motivo, el software bench dispone de una cantidad limitada de MIUs, bastante inferior al número de desarrolladores. Esto significa, que existe una limitación física para los ingenieros a la hora de hacer pruebas simultáneamente. Esto genera un cuello de botella en las fases de test. Además el continuo uso genera un desgaste en los dispositivos y cada recambio supone miles de euros. En definitiva, crear un puesto de simulación en el banco de trabajo supone un esfuerzo económico millonario.

El objetivo de éste trabajo es que cada ingeniero de software pueda disponer en su propio PC de un banco de pruebas. Esto supone un gran ahorro tanto de dinero como de tiempo.

Este proyecto supone un ahorro de tiempo en la corrección de fallos, así como una mejora de la calidad del nuevo código generado (disminuyendo los requerimientos de posterior modificación) y finalización del efecto cuello de botella en el laboratorio.

Cabe destacar que las simulaciones realizadas en el nuevo banco virtual serán para avanzar en el trabajo que realizan los desarrolladores, pero las pruebas definitivas necesarias para la

aceptación de las nuevas versiones de código de forma oficial por parte de la empresa, deberán realizarse en el banco de pruebas tradicional como indica el protocolo de la empresa.

Un objetivo prioritario en este trabajo es mantener, en todo lo posible, el mismo formato entrada/salida del banco real en el banco virtual. De esta forma los desarrolladores pueden usar el sistema virtual de la misma manera que el sistema real.

El último objetivo es que el sistema migrado proporcione otras ventajas, como nuevas formas de visualización de datos integrada en el código y herramientas paralelas generadas, así como mejoras en los test a realizar agilizando su proceso.

1.9 Estado del arte

Existen muchos simuladores que son utilizados como bancos de pruebas en distintas áreas. En el campo de la aviación, sin embargo, casi todos los bancos de pruebas conocidos se realizan sobre el componente real. Existe otro tipo de banco de pruebas, como es el Aircrew Synthetic Training Aids (ASTA). El ASTA es un banco de pruebas virtual, sin ningún elemento real, que simula todos los componentes del avión. Puede realizar en una misma simulación misiones completas como si se tratase de un avión real. La diferencia con nuestro propósito es que con ASTA no se puede analizar un componente de forma aislada. Además sigue siendo necesario un entorno distinto al ordenador del desarrollador. Así que aun siendo un banco de pruebas virtual, la intención y funcionalidad es distinta a nuestro proyecto.

Más cercano al trabajo que tratamos, actualmente se están desarrollando migraciones de algunos computadores de aviónica gracias a las nuevas funcionalidades de la última versión del entorno de simulación que utilizamos para este proyecto.

2. Descripción de las herramientas de trabajo

Antes de comenzar con el desarrollo del trabajo, es necesario dar una breve descripción de las herramientas usadas para el desempeño del mismo.

La aplicación fundamental que usaremos en este trabajo es MaTE (Modelling and Test Environment) [3]. En este entorno es donde se incluirán las simulaciones destinadas a estimular los interfaces de la aplicación y se analizarán los resultados.

También proporcionaremos una breve introducción a otros programas y herramientas que debemos conocer para entender el proceso de migración, como son los programas Tiger y Mandril además del analizador de buses o PBA. Todos ellos usados en el banco de pruebas (Software Bench).

2.1 Visión global de MaTE

Las cuatro compañías asociadas del Eurofighter (EF) han desarrollado un proyecto con el fin de mejorar las herramientas y los procesos que se utilizan para los sistemas embarcados del Eurofighter. Este proyecto proporciona, de forma integrada, todas las herramientas necesarias para el desarrollo y soporte de la aviónica del EF. El "Modelling and Test Environment" (MaTE) es una de estas herramientas.

MaTE reemplaza a las herramientas de test y simulación de aviónica usadas para testear e integrar las aplicaciones software y hardware. Además extiende su capacidad para abarcar fases de diseño y soporte en servicio del ciclo de vida de los sistemas aéreos.

Con esta herramienta podemos ejecutar una combinación de modelos, software migrado, simulaciones y componentes reales para el test y/o simulación de componentes, subsistemas y sistemas de aviación.

El sistema MaTE se compone de un número de subsistemas, *Master GUI*, *Modelling environment*, *Test system*, *Data visualization*, *Data repository* y *Data analysis*. Cada uno de los subsistemas se ha desarrollado por separado y cumple con una funcionalidad distinta. Estos subsistemas están intercomunicados, permitiendo el flujo de datos entre ellos.

2.2 Subsistemas de MaTE

La Master GUI (MGUI) es la interfaz de usuario de alto nivel. Permite acceder al resto de subsistemas de MaTE. Desde la Master GUI también podremos seleccionar el área de trabajo o "user work area" (uwa) donde tenemos definida la simulación. La información disponible en el área de trabajo abarca los modelos usados, sus características, el código a emplear y el test que se va a ejecutar.

Desde la Master GUI podemos crear el diccionario. El diccionario es la recopilación de señales que hace el sistema con la información asociada a ellas, esto es, sus mapeos, su tamaño y demás características.

Otra capacidad de este subsistema es la compilación. Para poder usar este entorno, debemos compilar el código generado, tanto de la MIU como de los test desde la Master GUI. Ésta compilación la realiza mediante el compilador GNAT [4]. Dispone de un log en el cual informa en cada momento del proceso realizado, así como de los errores, warnings y salidas de texto del código.

Finalmente podemos comenzar la simulación desde la Master GUI. Realiza una inicialización del código y permite seleccionar los módulos que se desean ejecutar. Además proporciona unos archivos de registro para controlar las operaciones realizadas tanto actualmente como en otras ejecuciones del programa.

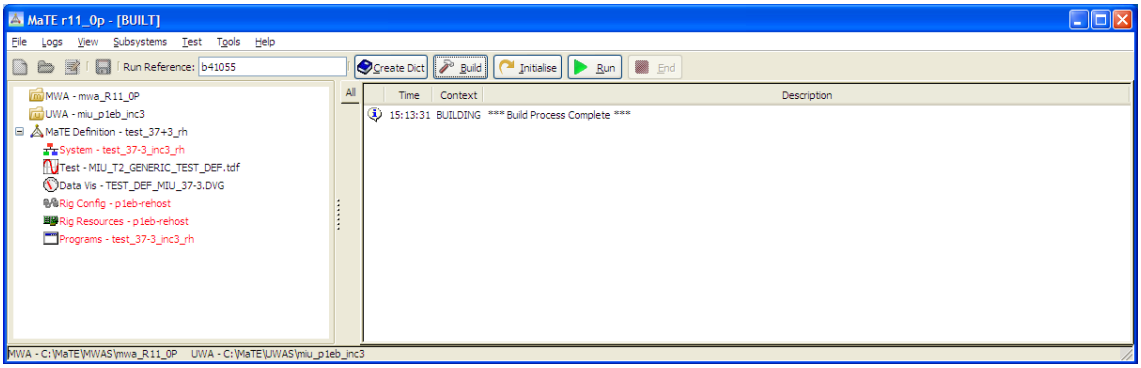

Figura 7 - MGUI de MaTE

2.2.1 Data Repository

El Data Repository (DR) almacena y maneja los datos requeridos por el sistema MaTE. Desde este subsistema se gestionan los datos del documento de configuración de interfaz (ICD). El ICD consiste en la información sobre todas las señales del sistema. Indica para cada señal que bus, puerto, dirección (address) y subdirección (subaddress) utiliza para la comunicación. Además incluye el tamaño de cada señal y sus propiedades. El ICD también contempla la tabla de transacciones.

Con la tabla de transacciones se arbitra la comunicación en el bus. Define la correspondencia de “address” y “subaddress” de transmisión/recepción del bus. Las transacciones están definidas en ciclos, es decir, existen ciclos de transmisión de paquetes de datos que son enviados y recibidos en un orden establecido. Cada ciclo se denomina ciclo mayor y está compuesto de otros ciclos, ciclos menores, que indican el orden de los paquetes.

A parte de la gestión del ICD, desde el Data Repository podemos definir los componentes de la simulación y sus propiedades. Podemos seleccionar qué test se usará y cómo está compuesto, definiendo si usaremos componentes reales o migrados y la plataforma física que deseamos utilizar.

Todas estas definiciones que realiza el Data Repository son traducidas a un formato legible para el sistema MaTE y que usarán otros subsistemas para realizar sus cometidos.

El Data Repository no está limitado al ámbito local, si no que está conectado a la base de datos del sistema. Las modificaciones y actualizaciones del ICD, los componentes y los test son accesibles por todos los usuarios que utilicen la base de datos. Debido al potencial peligro de pérdida de información por un mal uso, existe una gestión de permisos de lectura/escritura para controlar quién y de que que forma se conecta y modifica la base de datos. En caso de accidente siempre se podrá recuperar la información, gracias a la base de datos global, aunque supondría un retraso en los proyectos debido a la espera provocada por la restauración de la información.

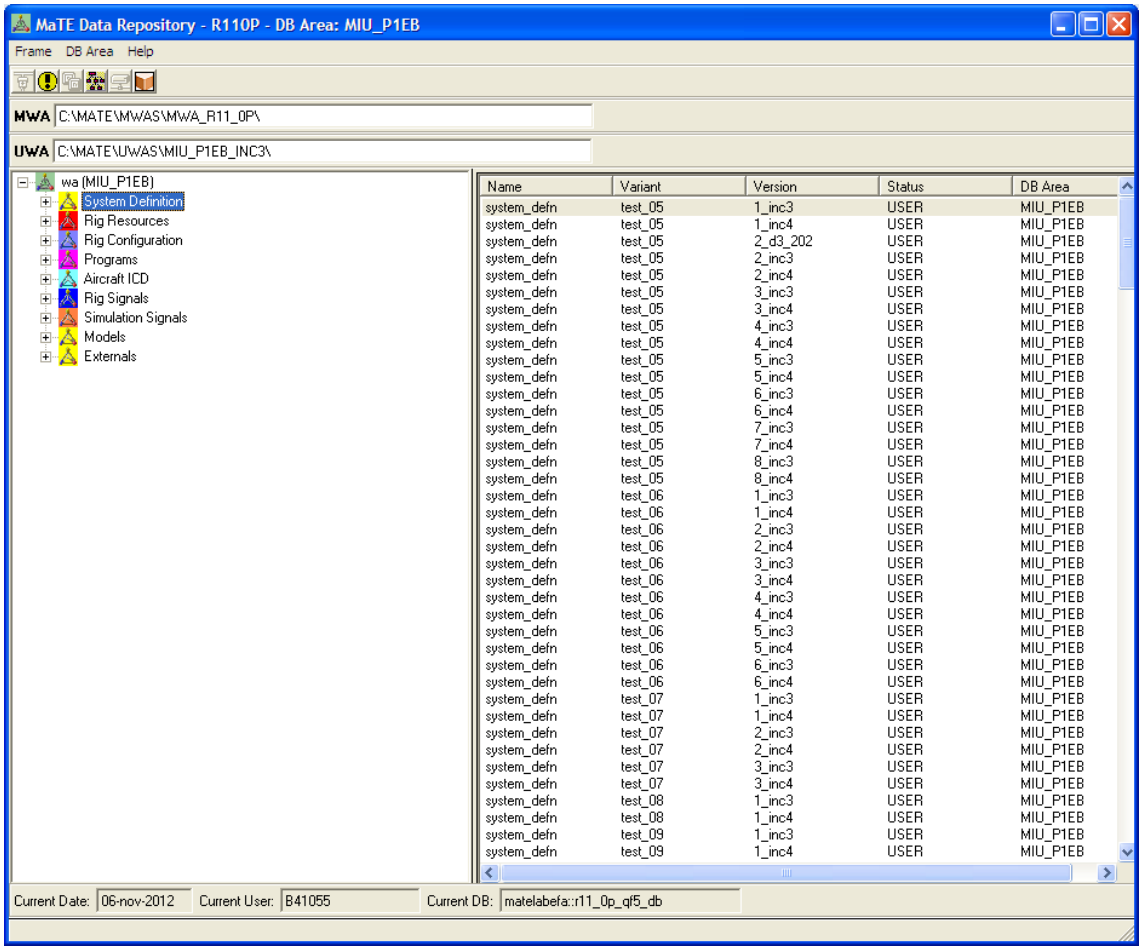

Figura 8 - Data Repository

2.2.2 Modelling Environment

El Modelling Environment (ME) proporciona la funcionalidad de ejecutar los elementos programados para la realización del test. El ME muestra qué módulos se han definido en el Data Repository y nos permite inicializarlos y ejecutarlos de forma controlada. Por lo tanto podemos controlar el acceso a los datos del test que realiza el resto de subsistemas desde el ME. Es desde este subsistema donde podemos acceder a la capacidad de debug del sistema. Para esta labor se apoya en el GPS (GNAT Programming Studio).

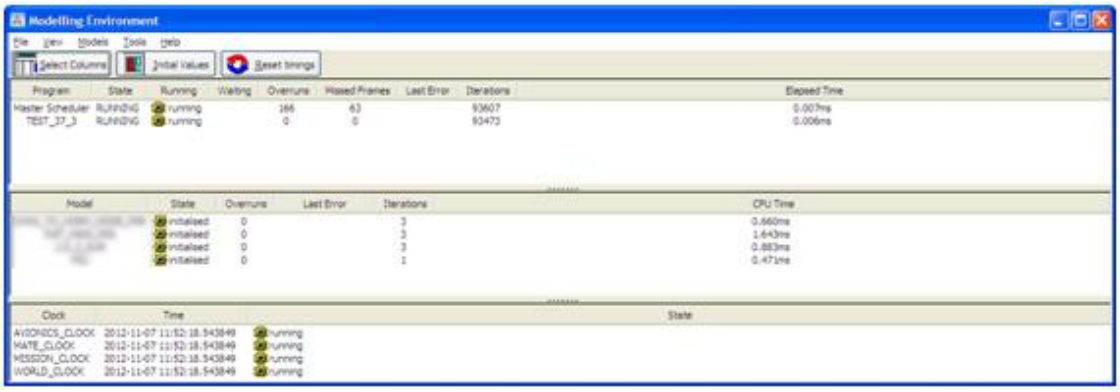

Figura 9 - Modelling Environment

2.2.3 Test System

El Test System (TS) proporciona las capacidades de grabación para los test, monitorear y simular las señales y datos de MaTE en tiempo real.

Desde el TS tenemos acceso a todas las señales definidas en el ICD. Nos proporciona la información asociada a cada señal, así como el valor que toma a lo largo de la simulación. Dispone de herramientas para realizar una visualización tanto numérica como gráfica de las señales. Nos muestra de esta forma la variación de valores de las señales y la capacidad de estimularlas en cualquier momento con los valores deseados.

Podemos acceder a las transacciones y modificarlas, adecuándolas a las necesidades del test o proceso que estemos realizando.

Otra funcionalidad del TS es la grabación de señales y mensajes. Podemos seleccionar qué información queremos grabar, indicando el tiempo de muestreo y el puerto de grabación dentro del sistema. Las grabaciones se pueden realizar en cualquier momento del test pudiendo ser pausadas y reanudadas en cualquier momento. El resultado es almacenado en un fichero interpretable por MaTE, concretamente por el subsistema Data Analysis. Estas grabaciones son imprescindibles para el posterior análisis de los test e interpretación de los resultados.

Todas las acciones que podemos realizar en el Test System las podemos automatizar gracias a la capacidad de definir scripts. Los scripts son pequeños programas realizables desde el test system, en un lenguaje cómodo e intuitivo, Automated Test Language (ATL), desde el que podemos programar acciones a realizar por el test system, así como operaciones y demás lógica. En este proyecto nos apoyaremos en estos scripts no sólo para realizar análisis, también los usaremos para automatizar los test a realizar en la MIU, ya que se ejecutan en tiempo real junto con los procesos activados desde el Modelling Environment.

Estas automatizaciones y ejecuciones en tiempo real también nos serán muy útiles para estimular señales de control periódicas pertenecientes al protocolo de comunicación de la MIU. Estas señales, llamadas DYNTAG (Dynamic Tag), son señales periódicas que verifican la

correcta conexión, en nuestro caso migrado de tipo virtual, del bus encargado de la comunicación con los computadores del avión.

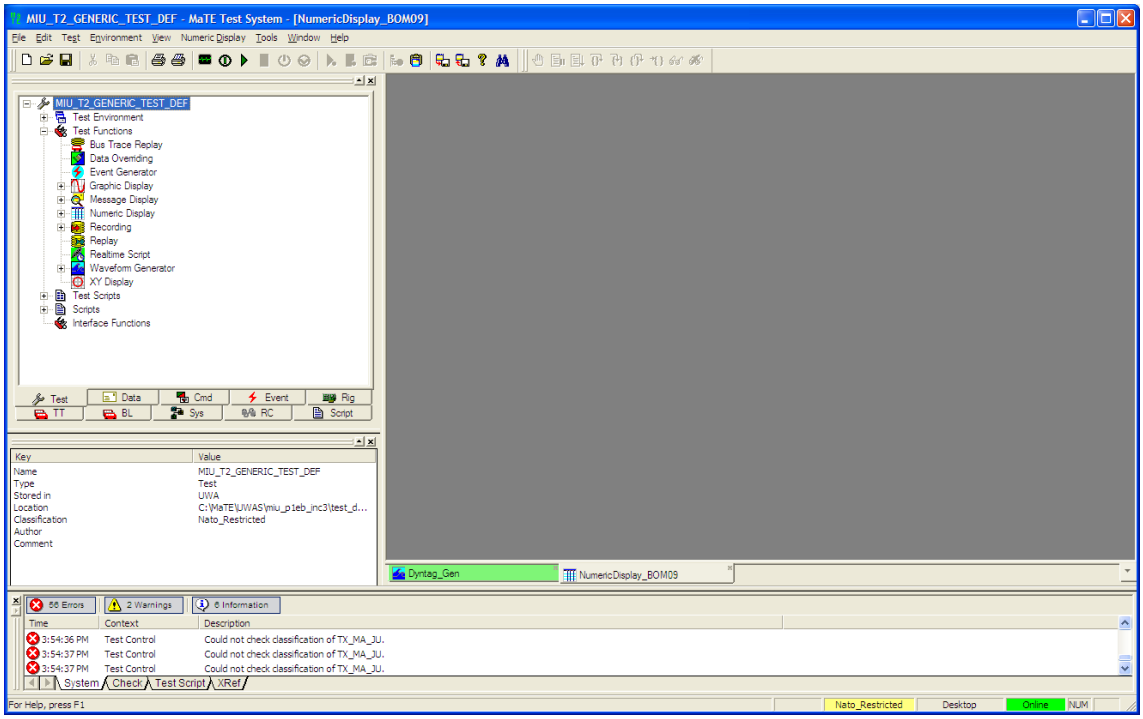

Figura 10 - Test System

2.2.4 Data Visualization

El subsistema Data Visualisation (DV) es una interfaz grafica que muestra señales y valores. Proporciona una herramienta basada en paneles gráficos que nos muestra la información de las señales, en tiempo real, en formato de valores de ingeniería. Es decir, no solo podemos ver qué valores adquiere cada señal, sino que muestra el significado asociado a ese valor, lo cual agiliza en gran medida el proceso de análisis.

Los paneles son adaptables a las necesidades del usuario. Se dispone de una serie de widgets, de una variedad de tipos y formas, asociables a las señales del ICD. Proporciona la capacidad de disponer de una serie de paneles personalizados que muestra la información en tiempo real de las señales involucradas en el test. De esta forma muestra una interfaz muy útil para analizar la evolución de un test. Además permite la funcionalidad de editar las señales y modificar su valor en tiempo real, así como realizar capturas temporales de los valores de las señales.

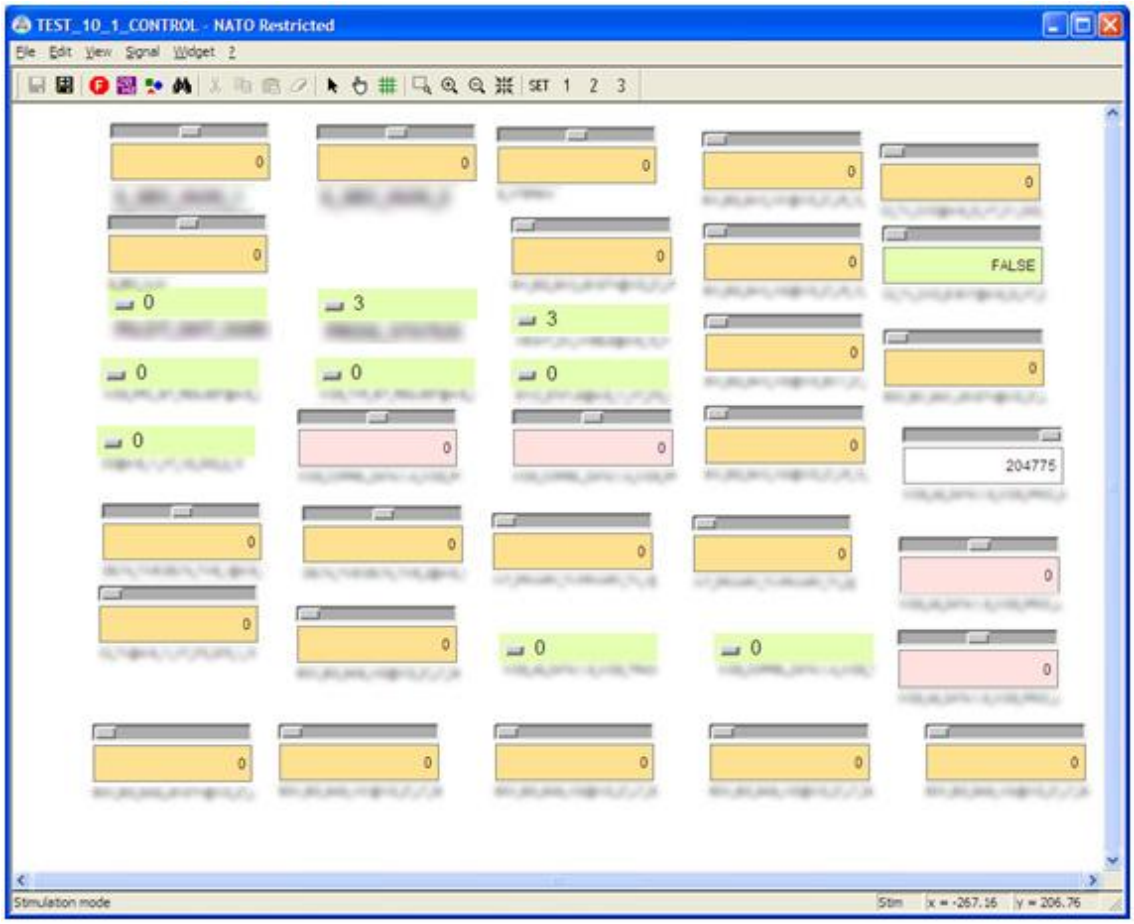

Figura 11 - Data Visualization

2.2.5 Data Analysis

El Data Analysis (DA) es un subsistema que permite la visualización y el análisis de simulaciones grabadas (RDFs).

Las grabaciones que realiza el Test System, en formato RDF (Recorder Data File) pueden ser visualizadas desde este subsistema. A diferencia de otros subsistemas, este análisis y visualización no se puede realizar en tiempo real.

El Data Analysis proporciona herramientas de análisis para visualizar las señales y mensajes grabados. Puede mostrar los resultados en gráficas con análisis matemático y comparación de señales. Otra visualización posible es la generación de reportes tabulados en texto plano e impresión de los datos guardados.

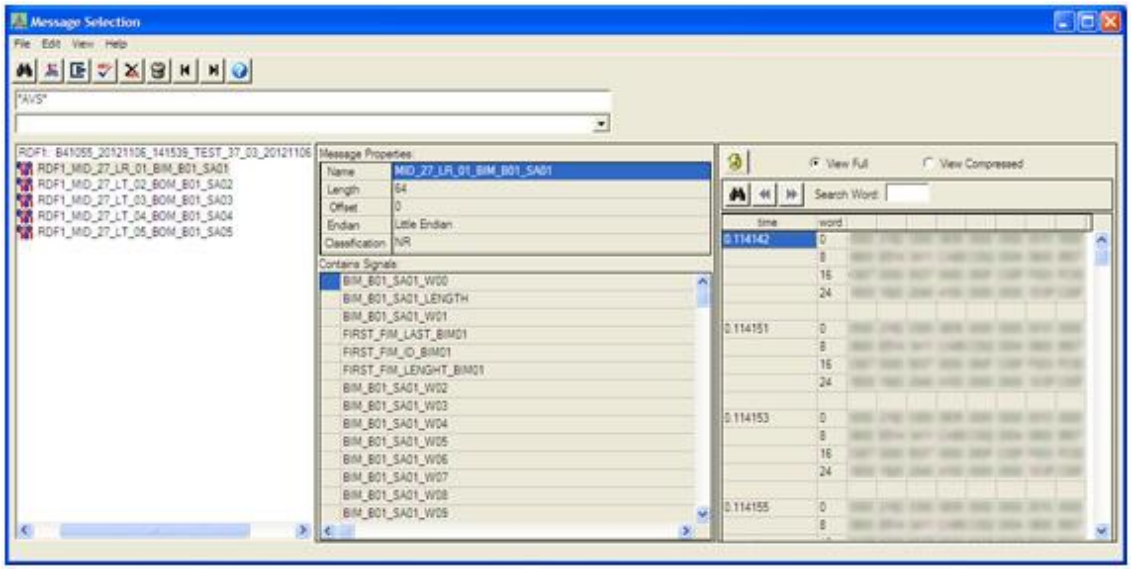

Figura 12 - Data Analysis

2.2.6 Interfaz con el hardware

El sistema MaTE es capaz de realizar comunicaciones con el hardware a través de distintas plataformas de trabajo, como son:

- EFABus Express Databus
- STANAG-3910 Databus
- MIL-STD-1553 Databus
- CES 8756AA CCDL
- FO-DDL
- Interfaz Analógica / Discreta para LRIs físicos.

Además MaTE se puede configurar para soportar otros tipos de interfaz. En el banco de pruebas real, se usan los buses EFABus Express (EFEX) y 1553.

2.3 TIGER

El Tiger [5] es un programa usado en una red de área local capaz de generar y recibir mensajes de información táctica “Data link” del Eurofighter. Esto significa que estando conectado a la MIU es capaz de interactuar como si se tratase del MIDS-LVT, enviándole la información en el mismo formato.

Además de esta capacidad, puede recibir datos del “Target Interface” (TIF). Ésta es la información proveniente del simulador de sensores del avión. Los paquetes TIF se trasmiten a través del bus 1553. También es posible transmitir y recibir datos SWIF (Software Interface) del

simulador de sensores por la interface de Ethernet. Estos paquetes contienen bloques de datos con la información del formato de datos test, escenario, tiempos de mensaje, etc.

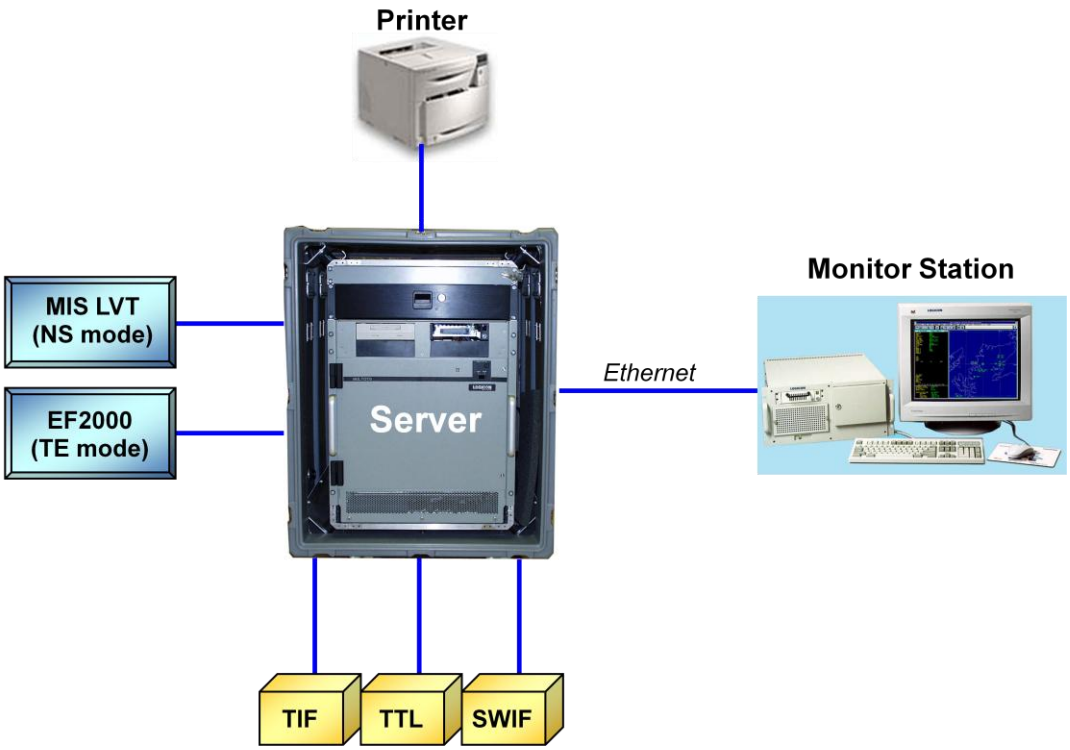

Figura 13 - Esquema de conexionado de TIGER

2.3.1 Modos de funcionamiento de TIGER

El TIGER puede funcionar de dos formas distintas. En modo de simulador de red y en modo de simulación de terminal [6].

El TIGER en modo de simulador de red consiste en interactuar con el MIDS-LVT usando el bus 1553 y antenas. El TIGER proporciona los datos tácticos que se deben intercambiar. Estos datos simulan los procedentes de otros vehículos (aéreos, terrestres y/o navales) o una estación fija. Se envían con el formato Link16 compuesto de BIM/BOMs (en la siguiente sección se explica en qué consiste este formato). El terminal MIDS y las antenas proporcionan las formas de ondas oportunas.

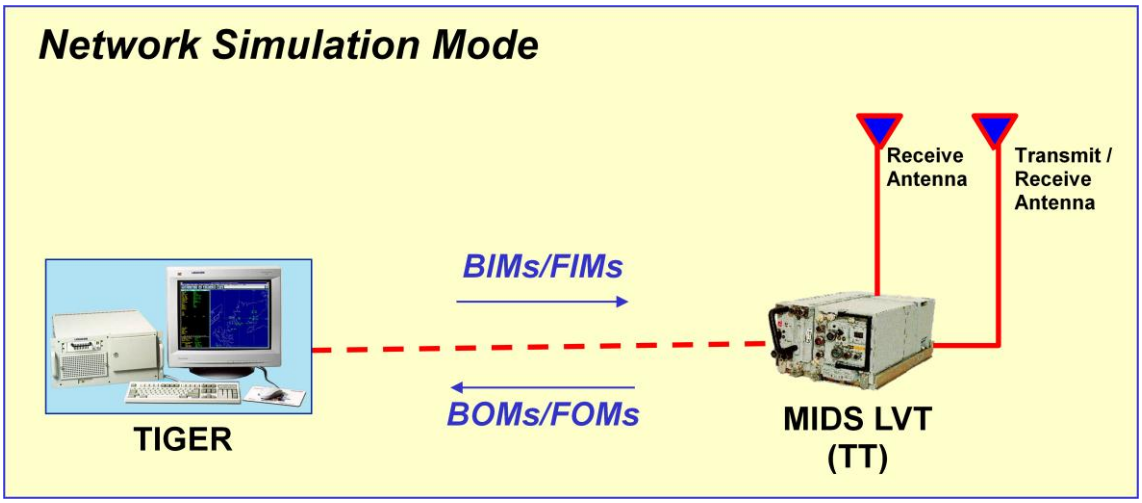

Figura 14 - Simulador en modo red

El TIGER en modo emulador de terminal consiste en el TIGER interactuando directamente con la MIU del EF usando el bus 1553.

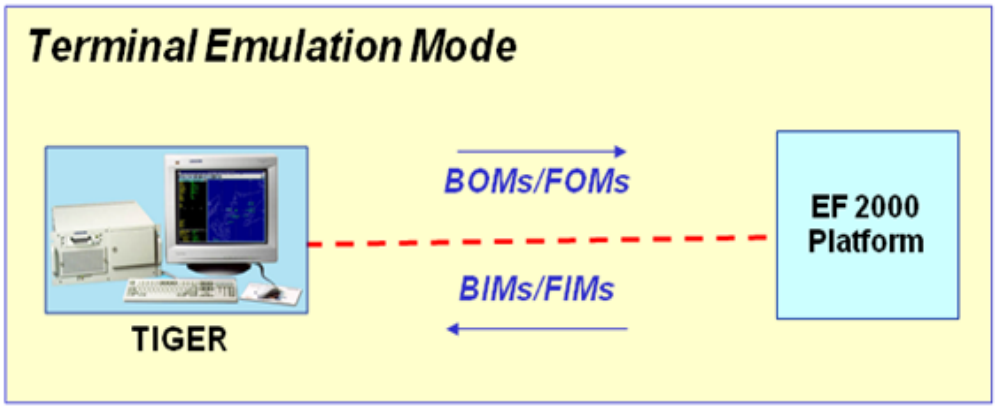

Figura 15 - Simulador en modo terminal

Con esta configuración, podemos enviar y recibir los mensajes tácticos conectados directamente a la MIU. De esta forma podemos realizar test sobre la MIU. Por ese motivo, en el banco de pruebas de usa el modo de emulador de terminal.

2.3.2 Software usado en TIGER

TIGER usa el protocolo "Transmission Control Protocol/Internet Protocol" (TCP/IP) para encaminar los paquetes de control y datos en la red, el protocolo "User Datagram Protocol" (UDP/IP) para los mensajes Link de la red y Ethernet para el control de medios de acceso.

Podemos diferenciar dos programas previos a la simulación encargados de la preparación de escenarios para la ejecución del programa en tiempo real.

Multi-Link Scenario Generation (MLSG): El programa de generación de escenario se usa para crear los archivos de escenario que se guardan para ejecutar durante el ejercicio. Todas las capacidades de generador de escenarios también están disponibles durante la realización del ejercicio.

Multi-Link Scenario Developer (MLSD): El programa de desarrollo de escenario es una interfaz gráfica que se utiliza junto con el Generador de escenarios (MLSG) para crear archivos de escenario. MLSD se puede utilizar para crear o actualizar archivos de código fuente existentes de escenarios.

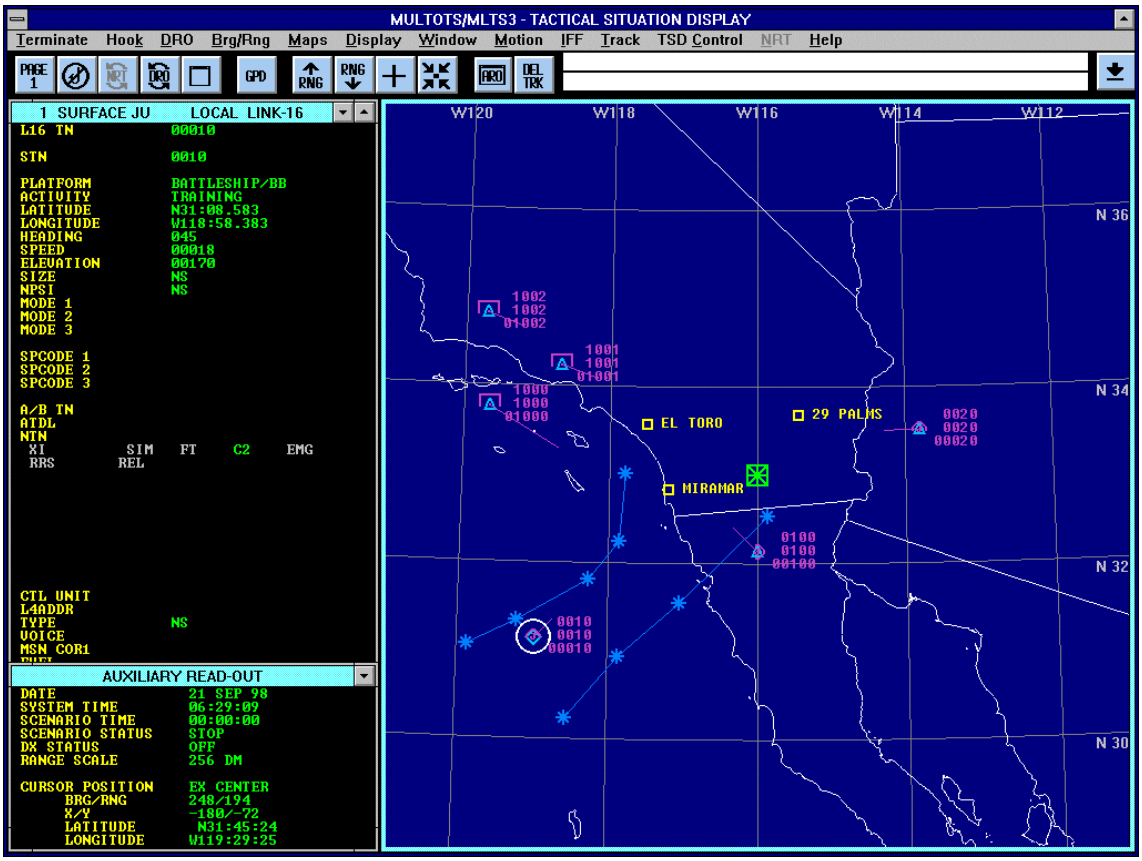

Figura 16 - Interfaz de TIGER

2.4 MANDRIL

MANDRIL [7] es un analizador de protocolos tácticos en excel. Puede decodificar y analizar trafico de diversos data link, como Link 16, Link 11 y Link 22. Es comercializado por Lockheed Martin UK IS&S Ltd.

Los data link son catálogos de organización de información táctica para los protocolos usados en la comunicación con el MIDS-LVT. MANDRIL es capaz de analizar los datos tácticos capturados por el analizador de datos disponible en el banco. Mediante este programa se presenta la información en un formato de trabajo cómodo, donde se puede ver con claridad la información transmitida.

2.5 PBA

El PBA [8][9] es un instrumento del banco de pruebas para testear, analizar y simular buses de datos de aviónica. Está preparado para soportar los estándares MIL-STD-1553 (STANAG 3838), STANAG 3910 y EFEX que son los que usamos en este proyecto.

Es sistema operativo PBA es Windows, desde donde se puede ejecutar la aplicación que realiza en análisis y la captura de datos. Es capaz de realizar una grabación del flujo de tráfico de los puertos y representarlo en un display numérico con los valores de los mensajes, su dirección e información de los tiempos. Estas grabaciones pueden grabarse para posteriores análisis y comparaciones.

Figura 17 - PBA (Hardware)

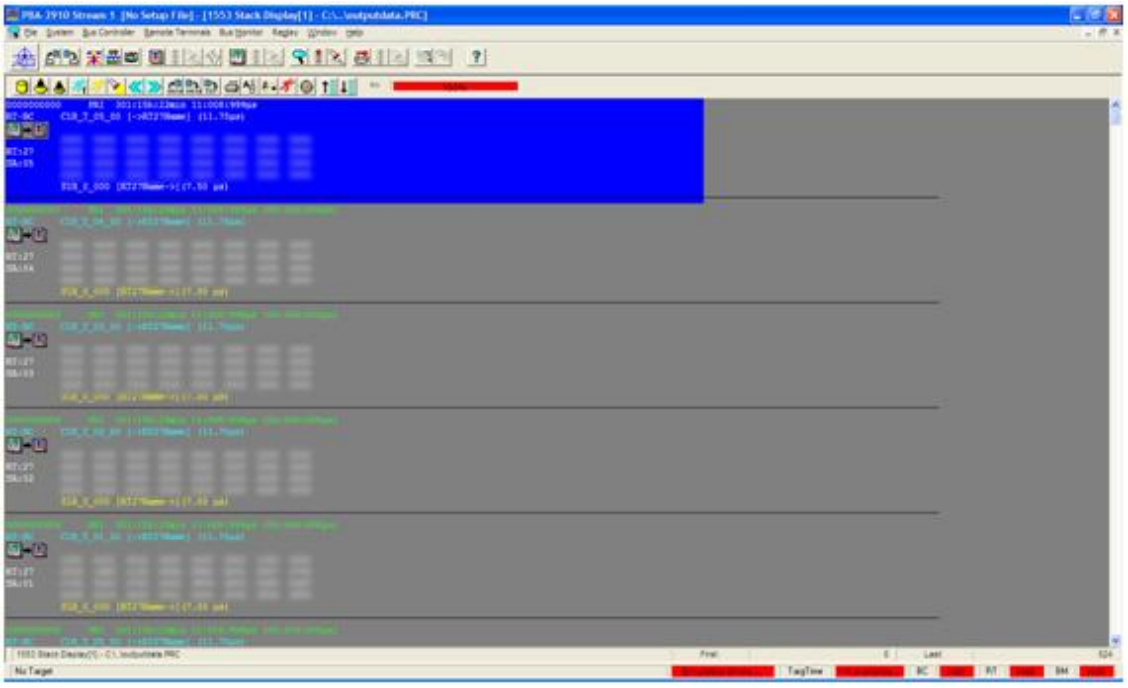

Figura 18 - PBA (Software)

3. Comunicación entre MIU y MIDS

Las comunicaciones de la MIU siguen los estándares de OTAN. Es necesario cumplir estos estándares ya que el EF debe ser capaz de comunicarse durante sus misiones con otros modelos de aviones, barcos, estaciones de tierra y en definitiva, con cualquier unidad aliada.

Los estándares de la OTAN para este cometido son los STANAG (Standardization Agreement). Los STANAG definen los procesos, condiciones y términos en los que se comunican los equipos de los países aliados.

La MIU, al formar parte de una unidad englobada en la OTAN, utiliza éstos estándares. La comunicación con los equipos del avión se realiza bajo el STANAG 3910 [10] y el EFEX [11]. Por otro lado, para la comunicación con el MIDS se usa el MIL-STD-1553 [12] y el STANAG 5516 [13]. Para la realización de este proyecto nos centraremos en el STANAG 5516 y el MIL-STD-1553. Para el uso del bus 1553 siempre se usa el MIL-STD-1553.

Estos estándares son muy generales y abarcan un amplio espectro de computadores. Existen numerosas plataformas con características apropiadas para diferentes tipos de dispositivos. En este proyecto describimos los estándares de la forma en que se aplica a nuestro propósito, es decir, a la comunicación MIU-MIDS. Dentro del estándar, para este proyecto vamos a usar la llamada “plataforma B”.

3.1 MIL-STD-1553 (STANAG 3838)

El STANAG 3838 o MIL-STD-1553 es un estándar ideado para el uso militar que describe el método de comunicación y los requisitos de la interfaz eléctrica para los subsistemas conectados al bus de datos 1553.

El Bus 1553 (1 Mbit / s) se compone de dos líneas bi-direccionales (para redundancia). La redundancia es “pasiva”, esto quiere decir que solo una línea se utilice en cada momento.

Cada una de estas líneas es un par trenzado apantallado. El bus controller (BC) transmite las palabras de comando por sólo una línea. El RT escucha las dos líneas y analiza las palabras de comandos de ambas simultáneamente.

En el conector de la aeronave, se utilizan pines para la asignación individual de la dirección física del terminal en el bus. Si la dirección física no es válida (la paridad), el terminal remoto no responderá a ningún mensaje. El terminal del MIDS-LVT debe tener la misma dirección en los dos buses redundantes del 1553.

La función del BC del bus 1553 la realiza el ordenador principal del avión, en nuestro caso la MIU. El MIDS debe actuar como terminal remoto (RT). Todas las comunicaciones son iniciadas por el BC. La comunicación puede ser directa entre BC – RT, RT – RT o en modo broadcast. Estas comunicaciones se hacen median palabras de 20 bits.

3.1.1 Formato de las palabras

Tenemos varios tipos de palabras. Palabras de comando (Command Word), palabras de datos (Data Word) y palabras de estado (Status Word). La composición de las palabras es la siguiente:

Command Word:

			15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
SYNC			ADDRESS					T/R	SUBADRESS /MODE					WORD COUNT/ MODE CODE					P

SYNC: Señal de sincronización.

ADDRESS: Direccion de destino. La dirección 31 se reserva para los mensajes broadcast.

T/R: Indica la dirección de la transferencia. ‘0’ para recepción y ‘1’ para transmisión.

SUBADDRESS/MODE: Este campo es usado tanto para indicar la subaddress del terminal remoto como para indicar que el campo Word count / mode code hay un comando de modo

WORD COUNT/MODE CODE: Dependiendo del campo anterior indica el número de palabras con datos involucradas en la transferencia (Word count) o un código que indica unan acción que debe llevar a cabo el RT.

Parity (P): Indica la paridad. Se usa paridad impar.

Data Word:

			15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
SYNC			MSB							DATA							LSB		P

SYNC: Señal de sincronización.

Data: Este campo contiene la información que debe ser intercambiada entre el terminal y el controlador.

Parity (P): Indica la paridad. Se usa paridad impar.

Status Word:

			15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
SYNC			ADDRESS					E	I	SR	RESERVED			D	B	S	DB	T	P

SYNC: Señal de sincronización.

ADDRESS: Direccion de destino. La dirección 31 se reserva para los mensajes broadcast.

Message Error(E): Este bit con valor ‘1’ indica que una o más palabras de datos asociadas a la palabra de comando anterior ha fallado. También indicara error si se especifica un comando ilegal o si la continuidad de transmisión no es mantenida.

Instrumentation (I): Para la plataforma B no está implementada. Siempre está a ‘0’.

Service Request (SR): Indica la petición de cesión de control del bus.

RESERVED: Siempre definida a 0.

Received Diffusion Command: Para la plataforma B no está implementada. Siempre está a ‘0’.

Bussy (B): Cuando está a ‘1’ indica que el terminal no puede enviar ni recibir información.

Subsystem Flag Bit (S): Indica la presencia de un fallo en la condición del terminal, detectado por el test de fabricación hardware o software.

Dynamic Bus (DB): Indica el fin de la cesión de control del bus.

Terminal Flag Bit (T): Indica la presencia de un fallo en el terminal detectado por el test de fabricación del acoplador.

Parity (P): Indica la paridad. Se usa paridad impar.

Un mensaje, como define MIL-STD-1553, consiste en el intercambio de palabras de comando, datos y estados como se muestra en la siguiente figura.

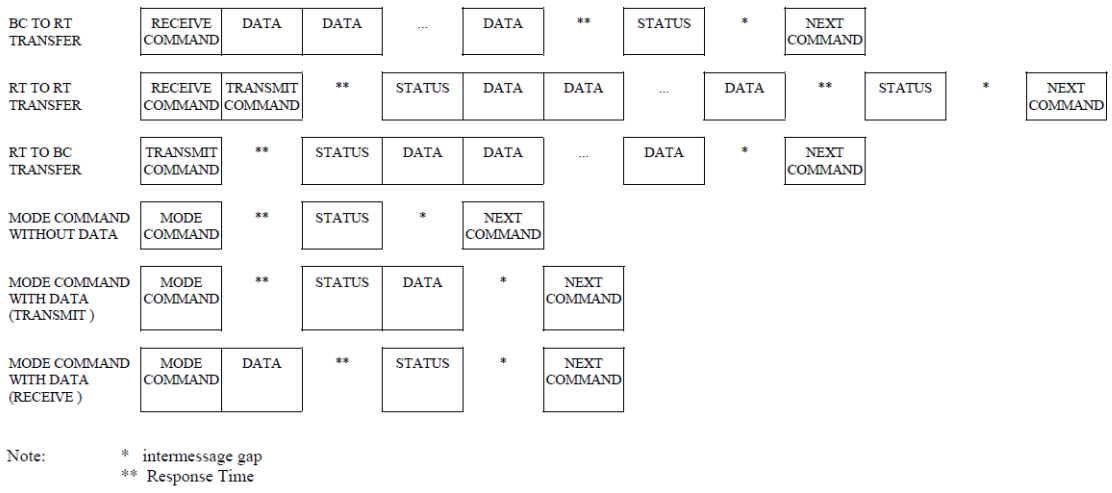

Figura 19 - Intercambio de mensajes en MIL-STD-1553

La información se envía en los llamados mensajes funcionales FIM/FOM (Functional Input Message/Functional Output Message). Los mensajes FxM transportan la información de entrada y salida entre la MIU y el MIDS-LVT. Estos mensajes están encapsulados dentro de los mensajes de bus (BIM/BOM), que pueden contener varios FxM.

3.1.2 Mensajes FxMs

Para tratar a los FIM y FOM de forma genérica, independientemente si son de entrada o salida, nos referiremos a ellos como FxM. Dentro de un mensaje de bus, ya sea de entrada como de salida (BxM) puede haber uno o varios FxM. Los mensajes FxM van precedidos de una cabecera donde indica su identificador y su longitud. Ésta cabecera se llama “presentation header”.

Presentation Header:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
P	FxM ID						FxM LENGTH								

Last Field (L): Es un booleano. Cuando tiene como valor el ‘1’ lógico, indica que es el último mensaje funcional. Si está con valor ‘0’, significa que no lo es.

FxM ID: Este campo proporciona el identificador del FxM. Dependiendo del identificador, proporciona un tipo de información u otra.

FxM Length: Indica la longitud útil del mensaje funcional, en números de palabras de 16 bits. La longitud de las cabeceras es excluida.

A continuación se muestra un ejemplo de cómo sería un FxM:

1	0	3	27
2 28	FxM 03 (27 palabras)		
29	0	11	10
30 .. 39	FxM 11 (10 palabras)		
40	1	7	15
41 55	FxM 07 (15 palabras)		

3.1.3 Mensajes BxMs

Como hemos indicado antes, los mensajes FxMs viajan dentro de mensajes BxM. Cada BxM puede transportar uno o varios mensajes funcionales. De las misma forma que los mensajes FxM usaban un identificador, los BxM también. Dependiendo de este identificador transporta unos tipos determinados de FxM. Por ejemplo, el BIM_B01 sólo transporta mensajes FIM01, el BIM_B03 en cambio, del tipo FIM04, FIM05 y FIM34. Antes del identificador incluimos una “B”, esto es porque trabajamos en la plataforma B. Cada variante lleva implementadas unas funcionalidades determinadas.

La longitud máxima de un mensaje BxM es de 32 palabras. Al transportar varios FxM es común que la información exceda ese número de palabras. Por eso, dependiendo del tipo de BxM, puede tener varias subdirecciones (subaddress). No todos los BxM tienen el mismo número de subaddress, por ejemplo el BIM_B01 tiene 5 subaddress, el BOM_B05 tiene dos y el BIM_B02 tiene 1. Los mensajes BxM incluyen una cabecera (Transfer Header Word) donde indican la longitud del mensaje:

Transfer Header Word:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1
RESERVED						USEFUL LENGTH								

Reserved: Los 6 bits de este campo estan reservados para un uso future, así que siempre se definen como ceros.

Useful Length: Especifica el número de palabras de 16 bits que contiene. La longitud de las cabeceras se excluye. Si este campo está definido en 0, el mensaje se descarta. En el caso de subdirecciones independientes, si la longitud es mayor que el campo “Word count” de la palabra “Command Word”, es ignorado. En todos los casos, los datos más allá de lo indicado por la longitud son descartados.

A continuación vemos un ejemplo de un mensaje BIM_B01:

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	TRANSFER HEADER WORD															
1	PRESENTATION HEADER WORD															
2 3	FIM01 #1															
..	PRESENTATION HEADER WORD															
..	FIM01 #2															
	PRESENTATION HEADER WORD															
N	FIM01 #M															
5*32-1	SPARE															

M = Número de FIM01s contenidos en este BIM

N+1 = Número total de palabras útiles en el BIM

3.2 STANAG 5516

Éste estándar define el “Tactical Data Link” (TDL) que utiliza la red de comunicación de la MIU con el MIDS. Concretamente utiliza el interfaz Link 16

Esta interfaz es la que utilizaremos y está destinada a proporcionar la toda la información necesaria de entrada/salida del avión durante su misión.

La interfaz Link 16 está compuesta por mensajes “J-series”. Estos mensajes J tienen una nomenclatura “Jn.m”. Cada mensaje Jn.m está compuesto de unos campos diferentes al resto y orientado a una finalidad distinta.

3.2.1 Mensajes J-Series

Los mensajes Jn.m incluyen una cabecera de 35 bits seguido de uno o más mensajes. Estos mensajes que siguen a la cabecera están compuestos por tres, seis o doce palabras de 70 bits de información, dependiendo del tipo de mensaje más 5 bits de paridad.

Estos mensajes Jn.m viajan dentro de los FxM. Como las palabras del bus son de 16 bits, los 75 bits del mensaje Jn.m se divide en 5 palabras de 16 bits. Los mensajes están compuestos por un “initial Word” y puede incluir una “extension Word” y/o una o varias “continuation Word”. Por ejemplo, el mensaje J-Series 3.5 está compuesto así: J3.5I – J3.5E0 – J3.5C1 – J3.5C2 – J3.5C3

Header:

24 23 22 21 20 19	18 17 16 15 14 13 12 11 10 9 8 7 6 5 4	3	2 1 0
<------	Track number, source	RI TM	Time Slot Type
	15	1	3

34 33 32 31 30 29 28 27 26 25
Secure data unit serial number
16

Time Slot Type: Define las características del mensaje.

RELAYED TRANSMISSION INDICATOR(RI): Indica cuando es o no un mensaje transmitido.

TYPE MODIFICATION(TM): Segun su valor, modifica la interpretación de Time Slot Type.

Track number, source: Numero identificador de la fuente del mensaje.

Secure data unit serial number: Indica el número de la “secure data unit” de la fuente.

Initial Word:

La “initial word” (Jn.mI) identifica el mensaje y el tipo. Contiene además datos básicos del mensaje.

24 23 22 21 20 19 18 17 16 15 14 13	12 11 10	9 8 7	6 5 4 3 2 1	0
<−−−−−−−	Message length indicator	Sublabel, J-series	Label, J-series	Word Format
	3	3	5	2

49 48 47 46 45 44 43 42 41 40 39 38 37 36 35 34 33 32 31 30 29 28 27 26 25
Campos de información
57

74 73 72 71 70	69 68 67 66 65 64 63 62 61 60 59 58 57 56 55 54 53 52 51 50
Paridad	−−−−−−−>
5	

Word Format indica de qué tipo es el mensaje (initial, extension o continuation).

Label informa de la etiqueta del mensaje Jn.m, es decir, la “n”.

Sublabel hace referencia a la sub-etiqueta, segun la nomenglatura corresponde a “m”.

Message length indicator indica la longitud del mensaje.

Extension Words:

Las “extension word” (Jn.mEx) contienen datos adicionales y normalmente se incluye en los mensajes, aunque no son necesarios. Si son transmitidos, deben hacerse en orden se serie a continuación de la “initial Word”.

24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1	0
<−−−−−−−	Word Format
	2

49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32	31	30	29	28	27	26	25
Campos de información																								
68																								

74	73	72	71	70	69	68	67	66	65	64	63	62	61	60	59	58	57	56	55	54	53	52	51	50
Paridad					−−−−−−−>																			
5																								

Continuation Words:

Las “continuation word” (Jn.mCx) contiene datos adicionales incluidos en el mensaje que dependen del protocolo y de la disponibilidad de información. Si se transmiten, deben hacerse a continuación de las “extensión Word”.

24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
<−−−−−−−																		Continuation Word label					Word Format	
																		5					2	

49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32	31	30	29	28	27	26	25
Campos de información																								
63																								

74	73	72	71	70	69	68	67	66	65	64	63	62	61	60	59	58	57	56	55	54	53	52	51	50
Paridad					−−−−−−−>																			
5																								

Continuation label Word indica el número de “continuation Word” del que corresponde.

3.3 EFEX y STANAG 3910

El MIDS-LVT y algunos computadores del avión se comunica con la MIU mediante los estándares descritos (MIL-STD-1553, STANAG 5516). Para el resto de computadores de aviónica existen otros dos protocolos, el EFEX, también llamado EFAbus Express y el STANAG 3910.

Estos computadores son simulados por MaTE. Debido a que en nuestro nuevo entorno seguimos trabajando con MaTE, como se hace en el banco de pruebas, no necesitamos trabajar con estos protocolos. Aun así resulta interesante conocer, aunque superficialmente, la existencia de estos protocolos.

En alto nivel EFEX y STANAG 3910 se comportan de la misma manera. Están compuestos por mensajes de comando y por mensajes de datos. Las comunicaciones las inicia le Bus Controller y pueden ser de tres tipos: BC-RT, RT-RT y BC broadcast.

La cabecera de los mensajes está compuesta por los DYNTAG (Dynamic Tag) y los STATTAG (Static Tag). Los STATTAG son un número fijo que identifica al terminal. Los DYNTAG son un número que aumenta secuencialmente de 0 a 255 que aumenta en una unidad cada vez que se transmite un mensaje. Una forma sencilla de averiguar si se están transmitiendo con éxito mensajes por el bus es observar si los DYNTAG están ciclando.

A bajo nivel sí existen diferencias entre EFEX y STANAG 3910. Para la comunicación bajo el STANAG 3910, la MIU dispone del bus 1553 y un bus de fibra óptica. Mediante el bus 1553 se tramitan los mensajes de comando. Esta comunicación se hace usando el estándar MIL-STD-1553. Los mensajes con los datos son enviados a través del conector de fibra óptica.

Los computadores que establecen comunicación con la MIU mediante el protocolo EFEX, lo realizan íntegramente por el conector de fibra óptica. La transmisión tanto de datos como de comandos se realiza por esa línea.

Por simplicidad, al canal de comunicación que opera bajo estos dos estándares lo llamaremos bus EFEX.

4. Proceso de migración

4.1 Configuración del banco de pruebas

Para poder migrar el banco, primero es imprescindible conocer cómo está conectada la MIU y de qué manera interactúa con los demás elementos del banco. Podemos ver un esquema de esta disposición en la Figura 20.

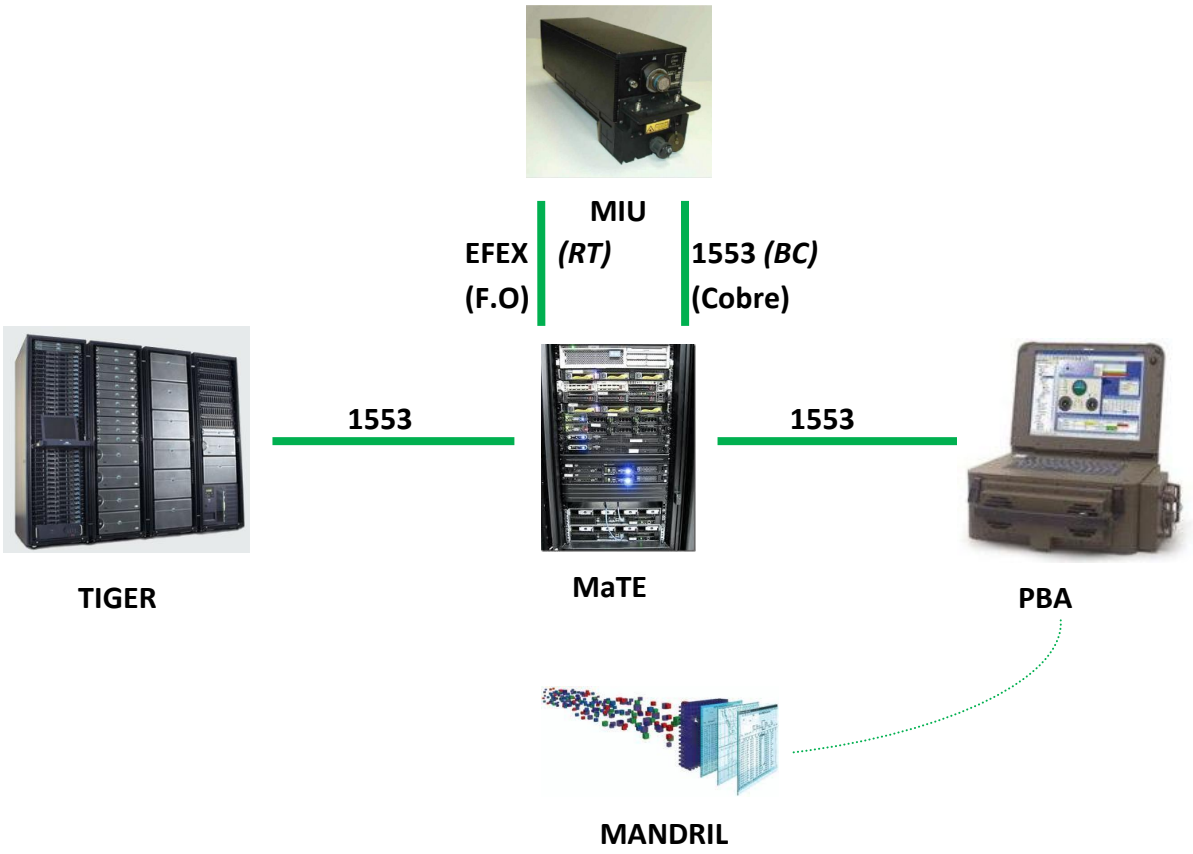

Figura 20 - Configuración del banco de pruebas

En primer lugar vemos que la MIU está conectada, mediante dos puertos distintos a un bloque llamado "MaTE". Como indicamos anteriormente, MaTE es el entorno, corrido bajo un computador, donde se ejecutan las simulaciones. De esta manera la MIU se comunica con MaTE como si estuviese en un entorno real. MaTE se comunica por el bus 1553 como Terminal Remoto y por el bus de fibra óptica como Bus Controller. El escenario creado para la realización de los test se genera con el TIGER (aunque alternativamente se puede realizar como una simulación directamente en MaTE). Éste escenario lo suministramos a MaTE por un bus 1553. Contiene la información con los mensajes Link 16 que queremos probar. Es decir, Tiger es un sistema ejecutado en un computador que sustituye al MIDS-LVT. Además por este puerto también recibe la respuesta que la MIU daría al MIDS.

Las simulaciones de la comunicación MIU-MIDS se realiza por el bus de cobre 1553, usando el protocolo de comunicaciones MIL-STD-1553, por eso MaTE tiene una conexión 1553 con la MIU. En esta conexión la MIU ejerce de BC ya que en el avión la MIU es BC del MIDS. Por otro lado, el computador de MaTE está conectado a la MIU por un conector de fibra óptica, que es usado bajo el protocolo EFEX o el STANAG 3910 (usando también un conector de cobre 1553). Mediante este puerto se vuelca el tráfico de datos correspondiente al resto de sistemas del avión que tenemos simulado.

Todos los resultados que queremos analizar, se envían mediante un bus 1553 al analizador de datos PBA. Finalmente el archivo guardado en el PBA podemos abrirlo en un ordenador con el programa "Mandril", que nos mostrará los resultados en un formato cómodo y legible.

4.2 Configuración del banco de pruebas virtual

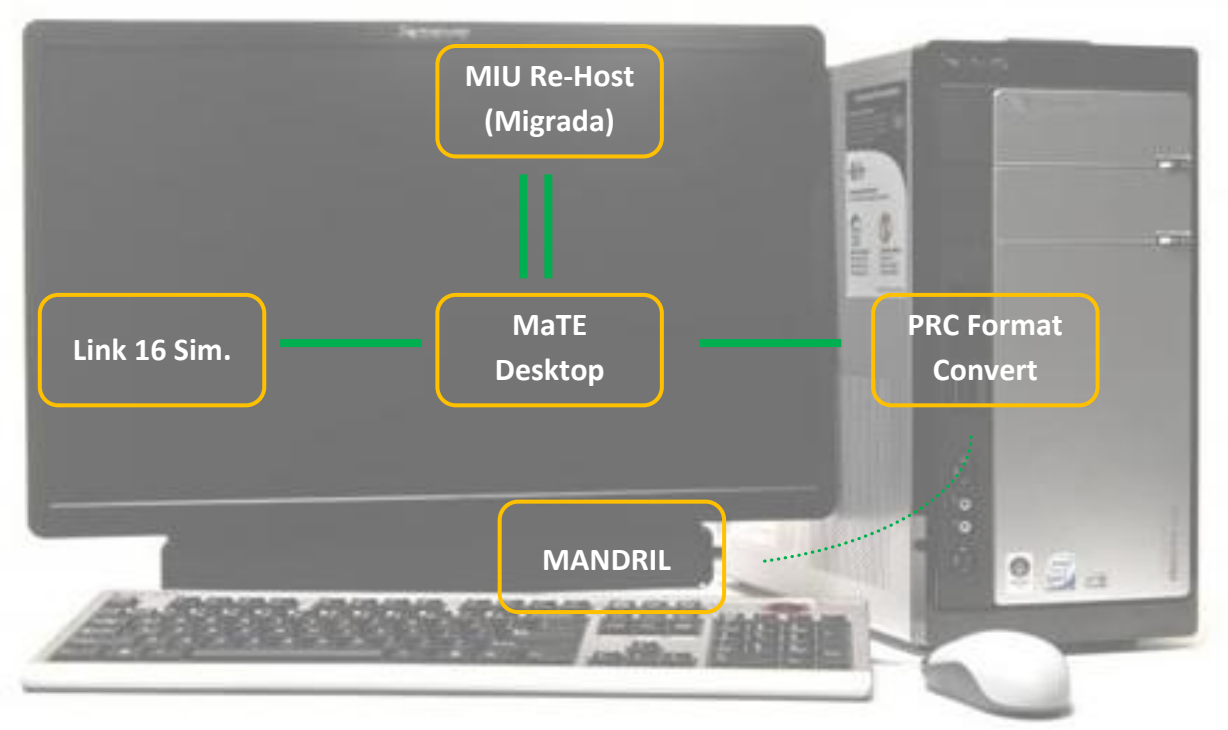

Figura 21 - Configuración del banco de pruebas virtual

Una vez conocido el funcionamiento *grosso modo* del banco de pruebas, podemos plantearnos una solución para lograr nuestro objetivo. Primero es importante conocer que existe una versión de MaTE, llamada MaTE Desktop, que es capaz de ejecutarse en un PC cualquiera. Esto nos proporciona la base para crear un entorno simulado. Debemos, por tanto, migrar el código de la MIU para que sea posible ejecutarla en un PC y de alguna forma, generar una plataforma que actúe como el Tiger y otra como el PBA. Podemos apreciar en la figura cómo resultaría.

Al sustituto del Tiger lo hemos llamado "Link 16 simulator" (Link 16 SIM.) y al del PBA "PRC Format convert". Usamos la nomenglatura "PRC" porque así se llama, dentro de todos los que hay, el formato de salida del PBA2000 que nos interesa. El programa MANDRIL no hace falta modificarle, ya que desde un principio es usado en un PC.

4.3 Arquitectura software

El código de la MIU está realizado en el lenguaje Ada95. Está compuesto por una serie de paquetes de código que podemos diferenciar en dos categorías. Los *Purchase Provides Software* (PPSW) y los *Target Specific Ada Packages* (TSAPs). Los PPSW son los paquetes que contienen la funcionalidad de la MIU. Gestionan el iniciado, cambio de estado de la MIU, manejo de interrupciones y el procesado de la información. Por otro lado, los TSAPs realizan la comunicación con el hardware. Serían los drivers de la MIU. Como ya no estamos en un banco físico, las llamadas a los elementos hardware se sustituyen por funciones equivalentes de MaTE. Estas funciones están incluidas en los llamados MaTE CTSAPs (Common TSAP).

4.3.1 PPSW y TSAP

Los paquetes pertenecientes a PPSW realizan las siguientes funciones:

- Manejo y gestión de interrupciones del bus EFEX y 1553
- Procesado y creación de mensajes I/O
- Almacenamiento de datos de misión
- Funcionalidades catalogadas NATO RESTRICTED

Los TSAPs manejan las siguientes funcionalidades:

- Inicialización de buffers y buses
- Gestión de los buses
- Producir interrupciones

4.4 Desarrollo del trabajo

Como indicamos en la introducción del proyecto, el método de desarrollo que usaremos será el modelo en V. Por lo tanto primero tenemos que analizar el proyecto para organizar el trabajo.

Comenzamos con la planificación;

Análisis de los requerimientos: Como explicamos en la introducción, el objetivo de este proyecto es tener la capacidad de realizar la misma funcionalidad que el banco de pruebas. El simulador debe ser capaz de simular todas las funcionalidades de la MIU, administrar sus salidas y entradas y mostrar los resultados. Además debe ser capaz de generar escenarios e interpretar la respuesta de la MIU. Todo esto debe ser realizado en un PC sin cambiar el entorno de trabajo del ingeniero.

Arquitectura del sistema: Podemos dividir el conjunto del proyecto en varios componentes. Por un lado tenemos la misión de migrar la MIU. Debemos modificar su código para que pueda funcionar en un PC. Además debemos simular los buses que utiliza, ya que en un PC normal no disponemos del 1553 ni de la F.O.

Otra tarea consiste en realizar un programa o método que pueda sustituir al TIGER. Debe ser una aplicación que pueda interpretar un escenario y convertirlo al formato Link 16 para ser simulado en MaTE.

Finalmente queda el diseño de un componente que capture las señales de respuesta de la MIU y las decodifique para que Mandril las interprete.

Ahora que tenemos dividido el trabajo en bloques, podemos comenzar a diseñar cada componente individualmente para implementarlo.

4.5 Migración del código de la MIU

En el proceso de migración se han tenido que realizar una serie de modificaciones en el código, tanto en los PPSW como en los TSAP.

Para adaptar el código debemos enfrentar las complicaciones debidas al cambio de entorno. Podemos desglosar las modificaciones en varias secciones.

1. **Formato de almacenamiento de datos.**

Es fundamental tener en cuenta que la MIU en el banco de pruebas funciona en un sistema del tipo “big endian”. Por el contrario, nuestro nuevo entorno de PC es “Little endian”. Esta diferencia provoca errores en la lectura de datos. Concretamente, debemos tenerlo en cuenta cuando realizamos una grabación desde MaTE. La grabación se diferenciará a la obtenida en el banco real en la ordenación de los bytes. Otra diferencia fundamental es la numeración de bits, que es difiere en cada entorno. El código está preparado para que se considere que la posición 0 esté situado “a la izquierda” de la palabra (MSB). Sin embargo, MaTE a la hora de manipular la información lo hace considerando que en ese lugar está la posición 15 (LSB). Debemos, por tanto, realizar modificaciones en el mapa de bits de las palabras. Cada palabra es de 16 bits, esto es, está definida del bit 0 al bit 15. Para cambiar el orden tenemos que realizar un complemento a 15.

Esta complicación entraña más complejidad de la aparente. Es así porque no tenemos libertad absoluta para realizar los cambios, todo lo contrario, estamos muy limitados. Las modificaciones del código deben ser las mínimas posibles. Además cada cambio debe ser discutido y aprobado, asegurando que no modifica o compromete el normal funcionamiento ni rendimiento de la MIU. Las modificaciones deben quedar justificadas y documentadas para su posterior manejo.

Por lo tanto, lo primero que debemos hacer es analizar qué mapa de bits deben ser cambiados. Esta situación afecta a los mensajes de lectura y escritura. Siguiendo el esquema del sistema, vemos que se realiza comunicación por dos vías. Una es el bus de aviónica (F.O.) y el otro es el bus MIDS (1553). Debido a las características del sistema y el objetivo de los test, no se realizan simulaciones con señales de aviónica. Es asó porque MaTE simula todos los computadores de aviónica. Esto implica que las señales usadas en este bus se usan directamente sin necesidad de realizar lectura/escritura.

En cuanto al bus MIDS, sí se realiza lectura y escritura de las señales. Por lo tanto los mapas de bits susceptibles al cambio son los que definen los BIM y los BOM. Dentro de esta comunicación, sólo se verán afectadas las señales cuyo valor es definido en bloque. Es decir, cuando indicamos, dentro de una palabra, el valor de determinados bits. Sin embargo si definimos directamente el valor asociado a una señal, MaTE lo interpreta correctamente.

Podemos ilustrar estos casos con el siguiente ejemplo:

Imaginemos que tenemos una señal "Señal_A" que está definida en los cinco primeros bits de la palabra "Palabra_1". Queremos asignarle el valor 3. Podemos definirla de dos formas:

- Señal definida en bloque:

Palabra_1 = 0001_1xxx_xxxx_xxxx

Como Señal_A está definida del 0 al 4, asignamos el valor 3 a esos bits. El problema está en que cuando procedemos a la lectura de esa palabra en nuestro entorno virtual, MaTE considera que los bits del 0 al 4 están situados al final de la palabra, produciendo un error de lectura/escritura.

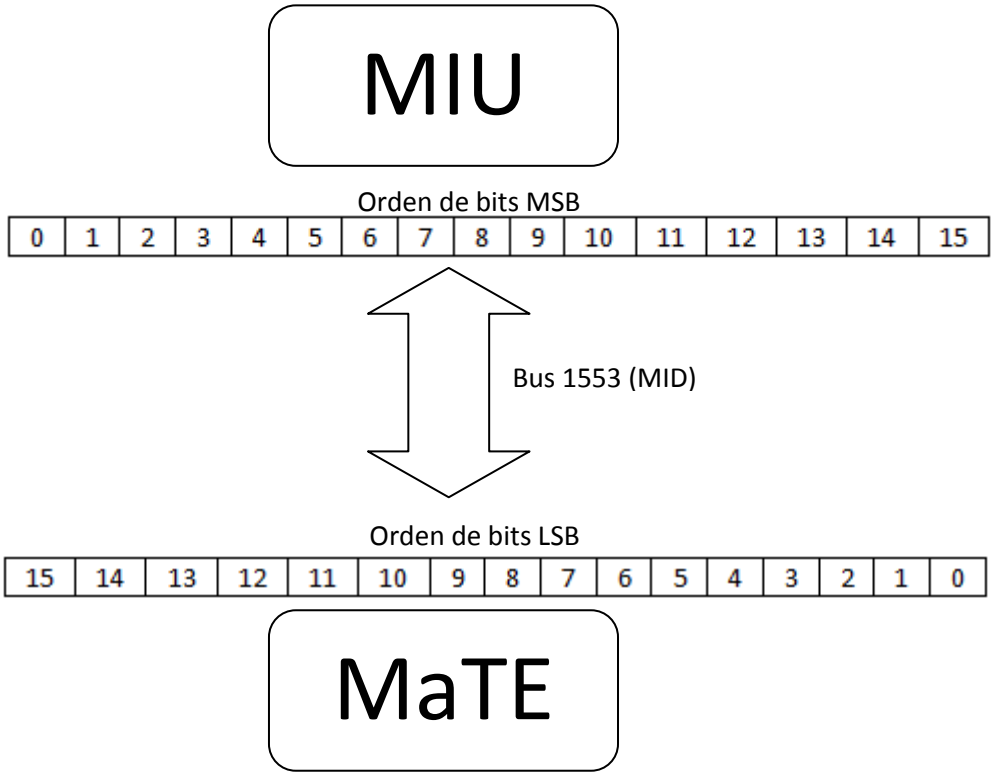

Figura 22 – Orden de bits de MIU y MaTE

- Señal escalada:

Señal_A = 3

De esta manera asignamos a la señal su valor directamente, independientemente de la posición que ocupe dentro de la palabra. Así no se produce error al acceder a la información.

No podemos aplicar la solución de asignar todas las señales de forma escalada, porque en la base de datos usada por MaTE, no están definidas. Tradicionalmente se ha accedido a estas señales usando la posición que ocupan en las palabras, así que nos vemos forzados a modificar el código para evitar este error en la lectura y escritura de las señales de los mensajes del bus (BxMs)

2. Gestión del bus MIDS.

En nuestro sistema usamos dos buses, el bus EFEX, también llamado bus de aviónica (AVS) y el bus 1553, que llamamos MIDS. Éste bus debe tener un trato especial. En el sistema real, la MIU es el BC. Es decir, el computador MIU inicia todas las comunicaciones. En nuestro entorno migrado no disponemos del LRI y debe MaTE asumir el rol de BC. Esto supone hacer unas modificaciones, especialmente en la tabla de transacciones.

La complicación de esta tarea es, principalmente, que MaTE no está preparada para ejercer de BC del bus MIDS, porque esta misión siempre la realiza la MIU. Debemos, por tanto, modificar la configuración de MaTE para que sea capaz de hacerlo. Para hacer esto es necesario modificar en el Data Repository la base de datos. Como indicamos en la descripción de las herramientas, desde el Data Repository de MaTE podemos definir las propiedades de la simulación. Debemos establecer en las propiedades de los modelos que no vamos a usar un elemento real, si no virtual. Ademas modificaremos la tabla de transacciones para establecer el cambio en el rol de control de bus entre MIU y MaTE. Ésta base de datos es común del banco de pruebas y nuestro sistema, así que los cambios deben hacerse de forma que no interfieran al funcionamiento del banco.

Para no interferir con el normal funcionamiento del banco de pruebas, no modificamos ningún modelo, sino que creamos nuevos modelos, similares a los ya existentes pero con las definiciones necesarias para el sistema migrado. En cuanto a la tabla de transacciones, la creamos pero la definimos en estado inactiva. De esta forma no afectará a los test del banco de pruebas y para nuestro nuevo entorno podemos ser capaces de activar las transacciones desde el código.

3. Prioridades

Otro aspecto a tener en cuenta es la asignación de prioridades del código. Al ser multitarea, debemos definir el orden de ejecución de las tareas estableciendo prioridades en el código. En el compilador AdaMULTI [14], usado en el banco de pruebas, las prioridades son asignadas hasta 255. Sin embargo, al cambiar de plataforma y trabajar en Windows con GNAT, las prioridades son hasta 30. Por lo tanto tenemos que escalar las prioridades del código para que GNAT lo pueda compilar.

4. Llamadas hardware/software

Las llamadas de los TSAPs a los elementos hardware deben ser eliminadas. Se sustituyen por las funciones MaTE, CTSAP, que simulan los elementos físicos del banco. Se debe, por tanto, modificar el código para estructurar los datos en el formato adecuado para su introducción en las funciones de los CTSAPs.

4.6 Resultados de los test

Una vez que el código funciona correctamente en nuestro nuevo entorno, tenemos que ser capaces de obtener los resultados de las simulaciones en el formato apropiado.

Cuando ejecutamos un test con sus simulaciones correspondientes, debemos utilizar la opción de grabación del Test System de MaTE. Con esta aplicación podemos grabar los mensajes que se envían y reciben por los buses.

Una vez tenemos una grabación, podemos ser capaces de visualizarla desde la herramienta Data Analisys de MaTE. Desde aquí no sólo podemos visualizar los resultados, si no que podemos guardarlos en un archivo en modo texto (.txt).

Desde el Data Analysis podemos examinar los mensajes de aviónica, es decir los mensajes que viajan en el bus EFEX y los mensajes del bus MIDS. Sin embargo, para visualizar los mensajes sujetos al catálogo Link 16 es conveniente el uso del PRC Format conveter y MANDRIL para poder analizar su contenido.

4.7 PRC Format converter

Una vez obtenida una grabación de MaTE, debemos diseñar e implementar otro componente. Es el llamado PBA Format converter que señalamos en la introducción. En este punto, tras averiguar cómo genera los archivos el analizador de buses PBA 2000, se crea un programa que tiene como entrada una grabación de MaTE en formato .txt y devuelve un fichero en formato PBA (.prc). Éste nuevo fichero puede ser visualizado tanto desde un simulador para PC del analizador como desde el Mandril. De esta forma cumplimos nuestro objetivo de simular un test y visualizar la salida en el mismo formato que en el sistema real.

4.8 Link 16 Simulator

El último componente que nos queda para completar nuestro esquema es un simulador de mensajes Link 16, que sustituya al TIGER en los test. El proceso para realizar esta tarea consiste en crear un programa que tenga como entrada una grabación de los BOM que genera el TIGER en cada test y devuelva un fichero con el formato de script de Test System con los BOM necesarios. De esta manera, la simulación de MaTE tiene los BOM con la información necesaria y los tiempos apropiados.

4.9 Fase de test y verificación

Continuando con nuestro modelo de desarrollo, se realizó una serie de verificaciones del código a distintos niveles. Primero comprobamos que cada subsistema funciona

correctamente. Cumplida esta etapa, se verificó que el conjunto del sistema se comportaba como esperábamos. Finalmente lo validamos al cumplir con las especificaciones requeridas

4.9.1 Integración y test de cada subsistema

En esta etapa de test, se trabaja en tres subsistemas: El código de la MIU, el Link 16 simulator y el PRC Format converter. Evaluamos cada componente de forma individual comprobando que cumple con las especificaciones que nos hemos marcado para cada uno.

En cuanto al código de la MIU, se realizaron varios test para comprobar su correcto funcionamiento. Cada test ponía a prueba una funcionalidad distinta. Los aspectos verificados fueron:

- Comunicación por el bus EFEX. Se transmiten y se reciben datos correctamente y en el tiempo debido.
- Recepción y procesamiento de los mensajes BOM que recibe la MIU
- Transmisión y creación de mensajes BIM.

Las pruebas realizadas en el PRC Format converter consistieron en usar grabaciones de test obtenidas en MaTE y analizar el fichero de salida. Este análisis se realizaba tanto en el propio PBA del banco de pruebas como visualizándolo desde MANDRIL. Comprobamos que los mensajes figuraban correctamente y en los tiempos debidos.

Finalmente tenemos el Link 16 Simulator. Para este componente actuamos de forma similar al PRC Format converter. Usamos grabaciones de la salida del TIGER de test formales de la MIU y analizamos el script generado por este simulador. Comprobamos mediante grabaciones de MaTE que se generan los mensajes BOM de forma correcta.

4.9.2 Integración del sistema y pruebas de verificación

El siguiente paso en la fase de verificación es poner a prueba el funcionamiento del sistema, es decir, que la unión de todos los componentes realizados funciona correctamente. Para realizar este proceso, generamos los mensajes BOM con el componente Link 16 Simulator. El script generado se ejecuta en el Test System de MaTE mientras simulamos la MIU. Grabamos la respuesta de la MIU y la usamos como entrada para el PRC Format converter. Finalmente procesamos ese resultado en MANDRIL.

Una vez comprobado que el proceso funciona correctamente, pasamos a evaluar si cumple con las especificaciones planteadas al inicio del proyecto.

4.9.3 Operación verificación y validación del sistema

Finalmente comprobamos que los test necesarios para validar el funcionamiento de la MIU se superan satisfactoriamente y analizamos si el sistema cumple con todos sus objetivos. Se realizan los test formales [15] establecidos por el equipo de la MIU que se deben superar para verificar que funciona correctamente. Una vez superados y por lo tanto comprobado el correcto funcionamiento, debemos analizar si el sistema cumple con los requerimientos establecidos al inicio del proyecto.

El sistema elaborado es capaz de realizar la misma tarea que el banco de pruebas real. La única necesidad de uso del banco es la grabación del escenario de TIGER. Ésta grabación se realiza una única vez por escenario, pudiendo ser reutilizados para varias versiones posteriores de código. Por lo tanto tenemos una independencia casi total del banco de pruebas. Por lo tanto damos como cumplidos los objetivos de este proyecto y tenemos una nueva herramienta preparada para ser usada por el equipo de trabajo de la MIU

5. Conclusiones

Una vez finalizado este trabajo, se ha obtenido una herramienta muy útil para el desarrollo de la MIU. Establecida la plataforma, el migrado de una versión software se realiza de forma rápida. Se ha redactado un documento de uso interno para el equipo de la MIU con los cambios a realizar en el código y el uso de las aplicaciones para que el proceso se agilice más.

De esta manera, a cambio de una breve dedicación inicial, se puede ahorrar mucho tiempo en las tareas de los desarrolladores. Por lo tanto hemos conseguido cumplir los objetivos que nos proponíamos al inicio de este proyecto.

Por un lado, evitamos el cuello de botella que supone acudir al banco de pruebas. Sólo será necesario ocupar los puestos para realizar los test formales para publicar una nueva release. Además de un ahorro de tiempo, también supone un desgaste menor del material, ahorrando una gran cantidad de dinero en recambios.

El segundo objetivo es proporcionar una plataforma que facilite el uso de debug en el código de la MIU. En el entorno generado, se pueden parar a la vez todas las simulaciones y tareas del código y usar GPS para depurarlo. Podemos, por tanto, analizar el código de una forma más sencilla y eficiente.

Finalmente, nuestro último objetivo, también superado con éxito es la realización de forma transparente para el usuario. Es decir, no es necesaria ninguna preparación previa o consideración a tomar para su uso al mantener el mismo formato de entrada. El formato de salida también se mantiene, siendo compatible con los procesos y herramientas usadas por el equipo de la MIU para analizar los resultados.

Durante la etapa de modificación de código se han incluido nuevas funciones que proporcionan reportes de datos de misión que facilitan la labor de los desarrolladores. Además se han creado herramientas externas que simplifican el análisis de los datos extraídos. Esto proporciona nuevas funcionalidades al entorno migrado que no existen en el banco real, reduciendo aun más el tiempo empleado en el desarrollo de código y corrección de errores.

6. Trabajo futuro

Una vez logrados los objetivos del proyecto, se nos plantean nuevas vías para mejorar el proceso y para proporcionar nuevas funcionalidades. Actualmente tenemos desarrollado un sistema migrado que imita al banco real. Aún así, se le ha proporcionado nuevas funcionalidades que no existen en el software bench. Ésta es la principal línea de desarrollo de mejoras que se plantea en este trabajo. Crear una serie de herramientas que proporcionen un acceso a la información que no existe en el banco de pruebas.

De esta forma se plantea el uso del Data Visualization de MaTE, en su forma programable, para crear un panel gráfico que simule el computador del piloto. De esta forma podríamos visualizar datos de misión de la misma forma que lo visualizaría un piloto, accediendo en tiempo real a la información durante el proceso de test y detectar ágilmente posibles fallos.

Otra línea futura es desarrollar una pantalla gráfica con las “huellas” o traks del avión durante una misión. De esta manera mejora la visualización de la situación del avión durante los test.

Finalmente, nos planteamos una herramienta orientada al uso requerido para el análisis de los mensajes tácticos del catálogo Link 16. Actualmente para analizar los mensajes con formato Link 16 usamos el programa MANDRIL. Este programa nos ofrece una funcionalidad mucho mayor de la necesaria. Sería interesante generar una herramienta orientada a los mensajes específicos que usa la MIU. De esta forma lograríamos un análisis de datos más eficiente.

7. Acrónimos

ATL	.....	Automated Test Lenguage
AVS	.....	Bus de avionica
BC	.....	Bus controller
BIM	.....	Bus Input Message
BOM	.....	Bus Output Message
DA	.....	Data Analysis
DR	.....	Data Repository
DV	.....	Data Visualisation
EF	.....	Eurofighter
EP/T	.....	Enhanced Process and Toolset
EPC	.....	Eurofighter Typhoon Partner Company
EFEX	.....	EFAbus Express
FIM	.....	Functional Input Message
FOM	.....	Functional Output Message
GUI	.....	Graphical User Interface
ICD	.....	Interface Control Document
LRI	.....	Line Replaceable Item
LVT	.....	System-Low Volume Terminal
MANDRIL	.....	Message Analysis and Data Reduction for the Integration of Links
MaTE	.....	Modelling and Test Environment
ME	.....	Modelling Environment
MGUI	.....	Master Graphical User Interface
MIDS	.....	Multifunctional Information Distribution System
MIU	.....	MIDS Interface Unit
MLSD	.....	Multi-Link Scenario Developer
MLSG	.....	Multi-Link Scenario Generation
RDF	.....	Recorder Data File
RT	.....	Remote Terminal
STANAG	.....	Standard agreement
SWIF	.....	Software Interface
TCP/IP	.....	Transmission Control Protocol/Internet Protocol
TDL	.....	Tactical Data LINK
TIF	.....	Target Interface
TIGER	.....	Tactical Data Link Integration Exerciser
TS	.....	Test System
UDP	.....	User Datagram Protocol
UWA	.....	User Work Area

8. Bibliografía y referencias

[1] www.eurofighter.com

[2] www.euromids.com

[3] MaTE Documentation

[4] www.adacore.com/gnatpro/toolsuite

[5] https://secure.lm-isgs.co.uk/TIGER/default.as

[6] TIGER Software User Manuals

[7] www.lockheedmartin.co.uk/uk/what-we-do/products/MessageAnalysisandDataReductionfortheIntegrationofLinks.html

[8] www.aim-online.com/products/software/pba2000.aspx

[9] PBA-2000 User's Manual

[10] STANAG 3910 Tutorial

[11] EFABUS Express Standard

[12] MIL-STD-1553 Tutorial and Reference

[13]STANAG 5516 Ed4

[14] www.ghs.com/products/AdaMULTI_IDE.html

[15] Software acceptance test description MIDS interface unit (P1Eb)