INDUSTRIALIZAR LA PRODUCCIÓN DE SOFTWARE CON LA RBSF

David Morera Merino
Jefe de Proyecto
ESI

Iratxe Gómez Susaeta
Ingeniero de Software
ESI

Ana R. Moya Cañas
Ingeniero de Software
ESI

David Morera Merino
Iratxe Gómez Susaeta
Ana R. Moya Cañas

Palabras clave

RBSF: Reuse Based Software Factory (Factoría de Software basada en Reutilización)
CF: Componente Flexible
OT: Orden de Trabajo
SDC: Sistema de Despliegue Controlado
CML: Component Management Lenguaje (Lenguaje de Gestión de Componentes)

Resumen de su Comunicación

Desde los coches hasta los juguetes, desde los teléfonos móviles y los ordenadores hasta los aviones, en todo el mundo los fabricantes realizan sus productos con componentes, mejorando así su calidad, tiempo-de-mercado y costes, su productividad y competitividad. Si esto es posible en la producción de los productos que nos rodean,¿ por qué no es posible en el software?

Mientras otros sectores han visto doblar en el tiempo su ratio de productividad, en el caso del desarrollo de software, el ratio de productividad se ha quedado estancado. Una de las razones del bajo éxito de las técnicas de Reutilización de Software dentro de la industria del software en general, es la percepción de que en algunos negocios todavía se ve el software como un objetivo y no como un medio. Se dedica mucho esfuerzo, tiempo y dinero al desarrollo de software, siendo éste una pequeña parte del negocio en sí mismo.

Este artículo presentará los principios para industrializar la producción de software para obtener ventajas financieras de este proceso. También se presentarán las herramientas GNSIS y SDC que proporcionan una solución completa para construir una Fábrica de Software Basada en Reutilización (RBSF).

David Morera Merino
Iratxe Gómez Susaeta
Ana R. Moya Cañas

INDUSTRIALIZAR LA PRODUCCIÓN DE SOFTWARE CON LA RBSF

1. Introducción

[1] En general e independientemente del sector industrial de aplicación del que hablemos, el desarrollo de software está caracterizado, en la mayoría de los casos, por una serie de problemas tales como: retrasos de tiempo, requisitos funcionales en muchos casos diferentes a los comprometidos, presupuestos sobrepasados, mala calidad de los productos y altos ratios de errores. Este ha sido el contexto en el que se ha enmarcado el desarrollo de software.

Analizando la situación actual mundial del desarrollo de software, se ve claramente que la calidad y la productividad de la industria no han estado creciendo al mismo ritmo que las necesidades de software de la sociedad en general, tal como se muestra en la Imagen 1. La informática ha llegado a ser, por tanto, un obstáculo al progreso.

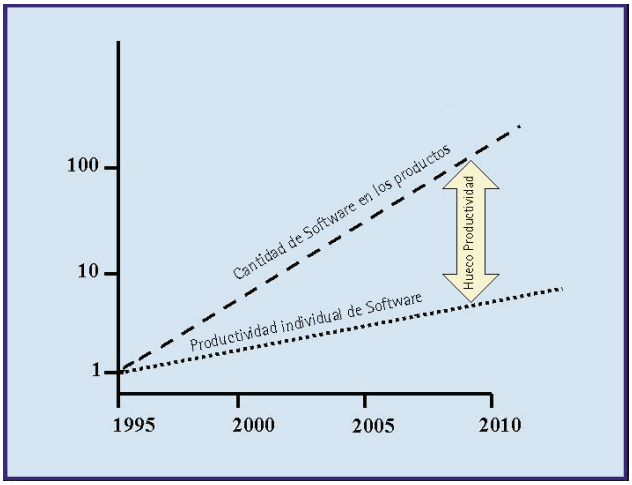


Imagen 1. Necesidades de Software en la Sociedad [2]

La disciplina de Ingeniería de Software ha estado intentando encontrar una solución a este problema durante mucho tiempo. Para dar la posibilidad de encontrar una solución final, tenemos que viajar en el tiempo un siglo atrás al norte de los Estados Unidos. [3] En Detroit, Henry Ford y 11 de sus socios, dieron vida a una de las mayores empresas fabricantes de coches en el mundo, la Empresa de Motor Ford. En aquel tiempo, sólo una parte limitada de la sociedad "rica" podía adquirir un coche considerado como artículo de lujo y un capricho demasiado caro. Quizás, la mayor contribución al sector de la fabricación de coches de la Empresa de Motor Ford fue la cadena de montaje móvil. Con esta técnica, los trabajadores podían estar en un mismo sitio realizando las mismas tareas repetidamente sobre múltiples vehículos. Esto hizo que la producción de coches fuera mucho más barata y por tanto mucho más al alcance de los consumidores.
El modelo de coche Ford T fue el primer vehículo fabricado con una cadena de montaje y también el primer coche que realmente fue accesible a todo el público. Este coche se empezó a fabricar en 1908 y fue el primer coche de la historia cuyo precio fue decreciendo cada año (desde $850 en 1908 hasta $380 en 1927 [4]). Esta disminución en el precio fue debida al hecho de que todos los coches se fabricaron de la misma manera, con las mismas piezas y con las mismas características [5].

Volvamos a l proceso de desarrollo de software y tratemos de encontrar similitudes con el caso del Ford T. ¿Qué ocurriría si construyésemos el software de forma similar a la fabricación del modelo Ford T? ¿Seríamos capaces de reducir el tiempo, coste y esfuerzo de desarrollo de ese software y por tanto el precio del producto final de nuestros clientes? Esta situación contrasta profundamente con el mundo de fabricación,

el reino de Ingeniería, donde la repetitividad de resultados determina y define la madurez de procesos[1]. La respuesta es "sí". El desarrollo de software se puede beneficiar de los conceptos y métodos de la Ingeniería de Fabricación, de forma que, construyendo una Reuse Based Software Factory (Factoría de Software basada en Reutilización), el desarrollo de software pasa de ser un arte a ser una industria basada en Ingeniería, intensivo en capital en lugar de en recursos humanos.

Los siguientes puntos explicarán el proceso para construir dicha factoría y mostrará las herramientas que permitirán construir una cadena de montaje de software.

2. Reuse Based Software Factory (RBSF)

Reorganizar el desarrollo de software en una organización a través de una Reuse Based Software Factory (RBSF) y así trabajar de una forma industrial, implica introducir innovación de forma intensiva en la empresa, a tres niveles:

Organización: La adopción de una RBSF necesita una organización que diferencie claramente el medio de producción de la producción real de software. Se trataría de una organización similar a cualquier empresa de fabricación. En este caso, la organización del desarrollo de software estaría separada en dos grupos: Oficina Técnica que define cómo se va a fabricar el software, los métodos, las herramientas, etc, y Producción que fabrican el software de acuerdo a los métodos definidos.

Métodos y Técnicas: A este nivel, la RBSF se centra en el método de producción de software en base a componentes y en las técnicas derivadas del uso de este método. Este tipo de producción hace posible alcanzar altas cuotas de estandarización con un alto grado de reutilización.

Herramientas: Esta nueva forma de fabricar el software requiere la integración de nuevas herramientas de producción que den soporte a la demanda industrial: ensamblado a demanda, trazabilidad, velocidad, calidad, repetitividad, fiabilidad y flexibilidad.

La clave de una RBSF es la reutilización: Reutilización de software alcanzada a través de componentes, previamente diseñados, que incluyen todos los estándares y normas de la organización y que han sido desarrollados, optimizados y probados para crear el mejor código software. Sin embargo la reutilización debe ser considerada el resultado de un conjunto de actividades realizadas para este propósito, de ahí la necesidad de innovar a nivel de organización, métodos y herramientas.

Si los programas de un dominio son diferentes, ¿cómo es posible construirlos con componentes estándar? Efectivamente, en la producción de productos manufacturados, todos los productos de una hornada son idénticos, algo que no ocurre en el caso del software, donde todos los programas son diferentes.

La tecnología de Componentes Flexibles (FC) resuelve este problema, logrando producir diferentes elementos desde componentes estándar. El uso de esta tecnología significa que es posible alcanzar un alto grado de reutilización con un número reducido de componentes.

Por tanto, ¿qué tipo de software soporta la RBSF? De la misma manera que conceptos como la cadena de montaje y componentes no son específicos de la industria o del producto, la RBSF es un concepto que no está ligado a ninguna tecnología específica. Una RBSF puede crear software en diferentes lenguajes de programación, accediendo a múltiples sistemas de gestión de bases de datos y trabajando bajo diferentes arquitecturas, abiertas o propietarias.

La implementación de una RBSF dura 6 meses y está compuesta de las siguientes fases: (Imagen 2).

David Morera Merino
Iratxe Gómez Susaeta
Ana R. Moya Cañas

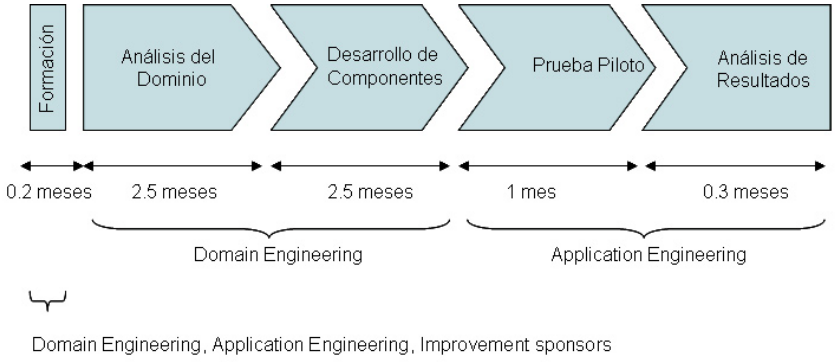


Imagen 2. Fases del proyecto RBSF

- **Formación:** Se va a formar en los siguientes temas:

    - Reutilización
    - Variabilidad y Similaridad (o Comunalidad)
    - Ingeniería de Dominio e Ingeniería de Aplicación
    - Factoría de Software
    - Desarrollo y utilización de componentes para construir programas

- **Análisis del dominio:** Fase del proyecto en la que la cooperación con el cliente es fundamental. El objetivo de esta fase es definir las diferentes funcionalidades software y el código fijo que se usa de forma constante en la empresa para desarrollar las diferentes aplicaciones. En otras palabras, se debe definir la variabilidad completa del dominio seleccionado en términos de qué es diferente y qué es siempre igual o similar en todas las aplicaciones existentes. Este análisis nos permitirá crear los Componentes Flexibles (FC).

- **Diseño de Componentes:** El resultado de esta fase son los Componentes Flexibles (FC) que se utilizarán en el desarrollo de las aplicaciones finales en la Factoría de Software. Podremos tener dos tipos diferentes de componentes flexibles, unos a nivel de diseño y otros a nivel de código. Estos componentes son las piezas clave a usar por los programadores para construir las aplicaciones del dominio seleccionado. Aunque la duración propuesta para esta fase es de 2.5 meses, similar a la fase de análisis del dominio, la duración de esta fase dependerá de los resultados de la fase anterior y del grado de variabilidad encontrado en el código fuente.

- **Prueba Piloto:** En esta fase, el cliente comprobará si la infraestructura recientemente creada se ajusta a sus necesidades y si los objetivos de la factoría definidos inicialmente han sido alcanzados (mejora de la calidad, descenso del tiempo de desarrollo, alto grado de reutilización, etc).

- **Análisis de Resultados:** En la última fase del proyecto se realiza un análisis detallado de los resultados finales alcanzados, se analizan las características de la factoría de software y se realiza un informe para uso interno o externo.

Tal y como se mencionó anteriormente en este punto, la implementación de una RBSF es imposible sin métodos y técnicas, pero tampoco sin un conjunto de herramientas que den soporte a su implementación, asegurando así un buen nivel de desarrollo de la factoría de software. La RBSF está compuesta de dos herramientas diferentes pero complementarias: GNSIS y SDC, que se explicarán en detalle en los siguientes puntos.

David Morera Merino
Iratxe Gómez Susaeta
Ana R. Moya Cañas

3. GNSIS

La herramienta GNSIS construye el código de las aplicaciones de la misma forma que se fabrican los coches en una cadena de montaje. Se utiliza para construir el código final de las aplicaciones partiendo de componentes reutilizables llamados Componentes Flexibles. Una característica clave de GNSIS es que no sabe lo que está construyendo. La herramienta es completamente independiente del lenguaje de programación de software, por lo que no sabe si está construyendo programas COBOL, Visual Basic, C, Java o cualquier otro lenguaje de programación. Lo único que hace es seguir las reglas escritas en un fichero plano denominado Orden de Trabajo.

Normalmente GNSIS está preparado para trabajar con no más de 30 o 40 Componentes Flexibles en cada dominio específico, lo cual puede depender de la complejidad de las aplicaciones a desarrollar y del lenguaje de programación elegido. Los Componentes Flexibles tienen la capacidad de comportarse de forma diferente dependiendo de en qué parte del código se usen y son lo suficientemente inteligentes como para saber cuándo deben ser usados y cómo. Yendo hacia atrás a la idea de la cadena de montaje de Henry Ford, GNSIS propone un esquema similar donde unos pocos ingenieros de software definen los Componentes Flexibles que después utilizarán los programadores para escribir las Órdenes de trabajo (OT). Después, dichas órdenes de trabajo serán generadas en GNSIS para obtener el código final de las aplicaciones. Estas órdenes de trabajo están preparadas también para incluir el código específico de cada aplicación que no lo proporcionan los componentes flexibles disponibles.

GNSIS normalmente trabaja en red, con directorios tanto locales como compartidos de Componentes Flexibles y Órdenes de Trabajo.

Principalmente, GNSIS tiene dos modos de funcionamiento:

• Se utiliza para construir las OTs, y proporciona los medios para crear y mantener todos los CFs que se utilizan en la Factoría de Software, teniendo en cuenta que los CFs son independientes de cualquier resultado que queramos obtener. GNSIS también permite probar cada uno de los componentes antes de añadirlos al repositorio de componentes.

• GNSIS proporciona los medios para construir el código final uniendo los CFs con las partes específicas de cada aplicación que se quiere construir. Este código final es totalmente independiente de la herramienta y una vez construido, estará preparado para su procesamiento.

Otra funcionalidad que ofrece GNSIS es la de analizar cada línea de código final generada. De esta forma se puede saber exactamente qué componentes se han utilizado, el modo de interacción de un CF con el resto de CFs, la cantidad de código generado por cada CF y el porcentaje que representa respecto al total.

Además, se puede garantizar la integridad de cada desarrollo realizado con GNSIS, ya que se puede controlar, de forma precisa, cada elemento utilizado en el sistema. Por ejemplo, se puede saber fácilmente qué código final se puede ver afectado si se cambia un CF. Así, GNSIS permitirá re-generar todas las OTs afectadas por el cambio de un componente.

Hay otras muchas características de GNSIS que no se mencionan en este punto pero que se mostrarán durante la presentación. Los siguientes puntos describen los principales subsistemas de los que está compuesta la herramienta GNSIS.

David Morera Merino
Iratxe Gómez Susaeta
Ana R. Moya Cañas

Construcción de Componentes Flexibles (CF)

GNSIS ofrece un editor inteligente que da soporte en la creación y edición de los CFs.

Un Componente Flexible (CF) es una pieza inteligente de código capaz de interpretar la variabilidad (conjunto de decisiones) de un dominio para adaptarlo al sistema final. Que un componente sea "inteligente" implica que es un pedazo ejecutable de código construido en un lenguaje específico que produce una parte del sistema final según las decisiones que han sido tomadas previamente, sin introducir ninguna línea de código adicional propia. Si sabemos qué resultados se quieren obtener, podremos construir los CFs que implementan esos resultados. Por ejemplo, podríamos crear un conjunto de CFs que generen un documento de especificación de requisitos en cualquier formato y también podríamos crear otros CFs que generen el diseño del sistema. Por lo tanto, los CFs no sólo se centran en generar código final de las aplicaciones, sino que también podrían cubrir por completo el ciclo de vida de un sistema (desde los requisitos de un cliente, hasta código y pruebas).

GNSIS trabaja con tres tipos diferentes de CFs:

• CF que define el dominio y la estructura básica del resultado final. Normalmente sólo hay un componente de este tipo para cada dominio.

• CF que define el sub-dominio o tipo de producto, es decir, componentes que son específicos para cierto tipo de resultados dentro del mismo dominio. Normalmente hay 5 o 6 componentes de este tipo por cada dominio.

• CF más específicos para cada funcionalidad. Se pueden tener varios componentes de este tipo, más pequeños y específicos unidos entre sí en lugar de tener uno más grande y genérico.
Los CFs están construidos en CML (Component Management Lenguaje), un lenguaje que es interpretado por GNSIS para producir el código final según unos parámetros de entrada que contienen las decisiones tomadas. CML es un lenguaje simple que contiene instrucciones para capturar los parámetros de entrada y realizar operaciones con ellos para establecer restricciones en los valores de entrada, para trabajar con tablas y ficheros, para comunicarse con otros CFs, etc.
Para poder utilizar los CFs, previamente hay que compilarlos con GNSIS, creando de esta forma un fichero de extensión "dll" para cada uno de ellos.

Pruebas de los CFs

Todos los CFs creados con GNSIS pueden ser probados de forma independiente. Se pueden probar una por una, todas las condiciones que están definidas para un componente para corregir o borrar el código que no es útil. Esto se podrá hacer a través de una interfaz que permitirá realizar varios casos de prueba a la vez definiendo únicamente un conjunto de valores de entrada. Los resultados de la prueba se mostrarán a través de un diagrama de barras que representará, desde una perspectiva global, el número de veces que cada condición del componente se ha cumplido.

Construcción de los elementos finales

Una vez se tengan todas las piezas para la cadena de montaje de software (FCs), es el momento para comenzar a definir las correspondientes Órdenes de Trabajo (OT) que producirán los resultados finales.
Una Orden de Trabajo GNSIS (OT) se compone de un conjunto de llamadas a los CFs con diferentes parámetros de entrada, junto con el Código de Negocio (código que es específico de cada programa que se está construyendo, y por tanto, no es posible incluirlo en los componentes). El lenguaje utilizado para manejar las llamadas a los CFs y la inserción del Código de Negocio es también lenguaje CML.

David Morera Merino
Iratxe Gómez Susaeta
Ana R. Moya Cañas

El editor de GNSIS guía al usuario en la construcción de las OTs en las llamadas a los CFs disponibles. Le indica las decisiones obligatorias, los valores posibles que esas decisiones pueden tomar (de acuerdo a la definición de parámetros realizada en los CFs), e incluye un mecanismo de control de errores muy preciso que permitirá mostrar gráficamente los errores en tiempo de ejecución.

Otra característica del editor de las OT GNSIS es que permite generar múltiples elementos finales a partir de un solo fichero de entrada. Por ejemplo, para una sólo OT se pueden crear todos los diferentes ficheros requeridos para un aplicación web completa construida en asp: login, acceso a base de datos, ayuda, etc.

Análisis de resultados

GNSIS tiene su propio subsistema de análisis para revisar y examinar el resultado final generado. Este subsistema ofrece las siguientes posibilidades:

- Visualizar una vista en forma de árbol de todos los CFs utilizados para generar cada resultado final.
- Visualizar las interdependencias entre todos los CFs.
- Visualizar las líneas que han sido insertadas en las diferentes secciones del resultado final.
- Visualizar estadísticas sobre la aportación de cada CF al resultado final.
- Saber exactamente el origen de cada línea generada, es decir, proporciona una trazabilidad completa.
- Identificar el origen de cada warning y de cada error. Se debe tener en cuenta que este subsistema no permitirá guardar un elemento que contenga errores.

Otra característica importante es la posibilidad de enlazar el subsistema con ejecutables externos al sistema que permitan realizar operaciones que no están incluidas en el juego de herramientas. Por ejemplo, se pueden compilar programas java que han sido generados, lanzando la aplicación de compilación con un solo clic a un botón añadido previamente al subsistema.

Administración del sistema

Cada elemento generado con GNSIS se almacenará en una base de datos que contendrá todos los CFs utilizados en cada OT, los parámetros de entrada con los que cada CF ha sido llamado y las partes específicas definidas en cada programa.

De esta forma, el administrador de la RBSF podrá obtener la siguiente información:

- Cada CF utilizado en una cierta Orden de Trabajo.
- Cada Orden de Trabajo que utiliza un cierto CF.
- Cada parámetro utilizado en una cierta Orden de Trabajo.
- Cada parte específica definida por cada Orden de Trabajo.

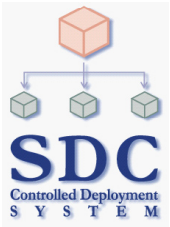
Todo esto puede ser muy útil para saber, por ejemplo, qué OTs están afectadas por el cambio de un CF o de uno de sus parámetros. También se podrán ver e imprimir informes si se requieren.

David Morera Merino
Iratxe Gómez Susaeta
Ana R. Moya Cañas

GNSIS Batch

Dos de las operaciones principales realizadas en GNSIS (compilar componentes y generar órdenes de trabajo) se pueden realizar en modo Batch. Esta característica es muy útil si se necesita re-generar el conjunto completo de OTs cuando un CF utilizado en todas ellas (como por ejemplo, el componente que define el dominio) ha cambiado; en lugar de generar cada una de las OTs una por una, se puede hacer todas de una vez. También es muy útil cuando se necesita re-compilar un conjunto de componentes que internamente utilizan otro componente que ha cambiado, por ejemplo, un conjunto de CFs que utilizan un componente de gestión de errores.

4. Sistema de Despliegue Controlado (SDC)



SDC (Sistema de Despliegue Controlado) es la herramienta para la construcción de aplicaciones en base a componentes de diseño. Está compuesta de dos subsistemas:

• Asistente para construcción de Componentes de Diseño: Asistente para el desarrollo de componentes de diseño de una forma controlada y guiada. Los componentes contienen la información necesaria para construir cualquier diseño de aplicación que esté definido dentro del dominio. Dichos componentes proporcionarán la arquitectura de componentes necesaria crear los diseños de las aplicaciones.

• Editor de Aplicaciones: Toma los componentes de diseño y genera las órdenes de trabajo estándar que GNSIS necesita para desarrollar una aplicación, guiando en la toma de decisiones para la construcción de aplicaciones.

El objetivo principal de esta herramienta es realizar el mejor diseño que un analista de sistemas podría pensar de las aplicaciones del dominio seleccionado. Este diseño también está basado en los estándares de diseño utilizados por la organización. La construcción del diseño de una aplicación con el SDC se hará en diferentes niveles funcionales que guían al usuario en todas las posibles decisiones que podrían ser tomadas en cada nivel. Una vez resueltas esas decisiones, el nivel queda implementado. La función de estos niveles de diseño, es "pensar" cómo debería ser un sistema nuevo haciendo mucho más fácil el trabajo del analista de sistemas.

El SDC irá pidiendo la información necesaria para crear el sistema completo. Se trabaja por niveles, ya que se empieza a preguntar desde lo más genérico hasta llegar a lo más específico. El SDC creará los diferentes subsistemas en base a las decisiones que se vayan tomando y así irá bajando de nivel hasta llegar al final, es decir, a las órdenes de trabajo de construcción que son la entrada a la herramienta GNSIS explicada en el punto anterior.

Por lo tanto, el SDC proporciona un medio para elevar un paso más el nivel de reutilización y comenzar a realizar las aplicaciones desde el nivel de diseño y no desde el propio código. Este instrumento es la parte de la puesta en práctica de la RBSF y directamente está unido a la herramienta GNSIS.

5. Experiencia con la RBSF

La RBSF va dirigida a organizaciones que desarrollen software para dar soporte a su propio negocio, por ejemplo, organizaciones de los sectores: Financiero, Seguros, Distribución, Administración, Energético, Transportes y también a organizaciones con equipos grandes de desarrollo en dominios tecnológicos específicos.

Nuestra experiencia en implantación de Factorías de Software data del año 1995 con más de 15 implan-

taciones realizadas, en los sectores: Banca, Administración Pública, Transporte, Energético, tanto de España como en Latino América. Recientemente se ha abierto una oficina en Estados Unidos para la entrada al mercado americano. Los dominios (lenguajes de programación) en los que se ha trabajado han sido: Cobol, asp, java, html, resaltando el hecho de que la Factoría de Software puede trabajar en cualquier otro dominio.

Como dato importante a resaltar, decir que, en todas las implantaciones realizadas en cada uno de los clientes de la RBSF, el número de componentes desarrollados de media, no ha superado en ningún caso el valor de 40, alcanzándose de media, un ratio de reutilización de entre el 60% y el 70%, habiendo desarrollado entre 2.500 y 3.000 programas.

6. Referencias

[1] Manu Prego, Reuse Based Software Factory, ESERNET Method Book v3.0, Enero 2003.

[2] ITEA - EC Internal Reflection Group on Software Technologies, April 2002

[3] Ford Motor Company, "Ford Motor Company, History" - http://www.ford.com/en/heritage/history/default.htm

[4] The Model T Ford Club, "Original Model T Ford Prices by Model and Year", http://www.modelt.org/articles/art_tprices.htm

[5] Autos Clásicos, "Ford T", http://autosclasicos.espaciolatino.com/archivos/ford-t.htm