TRABAJO FINAL DE GRADO EN ADMINISTRACIÓN Y DIRECCIÓN DE EMPRESAS

Un modelo probabilístico para predecir el valor del subyacente IAG (International Airlines Group) del IBEX-35. Diseño de una página web dinámica para mostrar las predicciones.

Validación y aplicación en Python del activo subyacente IAG.MC con representación en página web: http://cotizaccion.imm.upv.es

Autor:

D. Miguel Beltrán Sanz

Directores:

Dr. Juan Carlos Cortés López
Dr. Rafael Jacinto Villanueva Micó

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

FACULTAD DE ADMINISTRACIÓN Y DIRECCIÓN DE EMPRESAS. UPV

Índice

Índice del tablas

Índice de gráficos

Índice de figuras

Justificación de asignaturas relacionadas

El Trabajo Final de Grado es la culminación a más de cuatro años de esfuerzo. En este último gran proyecto antes de la obtención del título de grado es necesario poner de manifiesto las asignaturas que a lo largo de la carrera han contribuido a la correcta ejecución de este trabajo.

Modelos Matemáticos Para ADE (Primer Curso): En esta asignatura aprendimos el valor de los modelos y las matemáticas aplicados a un ámbito como es la economía. Esta asignatura ha sido vital en el correcto entendimiento del problema y en el desarrollo de la solución.

Econometría (Segundo Curso): Esta asignatura defiende el valor de la estadística en diferentes campos de la administración de empresas. Es vital para tener las nociones básicas en términos estadísticos. Lo que más desatacaría es el concepto que se transmite de la adecuación de un cálculo estadístico en la realidad.

Análisis y Consolidación Contable (Tercer Curso): Si bien es cierto que esta asignatura estaba focalizada en el análisis contable de las empresas creo que transmite con rigor el concepto del análisis de una empresa. Ya no solo a nivel de los libros contables si no a un nivel estratégico que permite entender el camino que sigue una empresa.

Resumen del Trabajo Final de Grado

Una de las herramientas fundamentales para el desarrollo de la economía en el mundo globalizado en el que vivimos es la existencia de bolsas donde las acciones de las empresas son cotizadas.

Estas bolsas juegan un papel fundamental para los inversores y para las empresas que participan en ellas. En un periodo económico donde los tipos de interés son del 0% debido a las directrices marcadas por el BCE es fundamental para los inversores, acudir a este tipo de mercados para obtener rentabilidad por su dinero y vencer a la inflación.

Es necesario, por tanto, el proveer a esos inversores de información relevante sobre el desarrollo de las acciones que tienen en sus carteras. Debido a la, también, creciente uso de las nuevas tecnologías este trabajo tiene como objetivo dar información relevante a los inversores sobre el desarrollo de la cotización del grupo IAG.

Primero, haremos un análisis a nivel estratégico del grupo IAG. Posteriormente analizaremos, desde un punto de vista tecnológico y de ejecución, los cambios a acometer en el software que proporcionará valor a los inversores. Más adelante desarrollaremos los modelos matemáticos que nos ayudarán a dar la predicción de los valores futuros y finalmente haremos un breve repaso al esfuerzo acometido en este Trabajo Final de Grado.

Objetivos del Trabajo Final de Grado

El objetivo fundamental de este proyecto es el dotar a cualquier interesado de una herramienta financiera que realizará predicciones sobre el activo IAG.MC. Para ello realizaremos lo siguiente:

- Un análisis estratégico del grupo IAG para conocer su estado en la actualidad y su desarrollo futuro a nivel corporativo.

- Aplicar el modelo Log-Normal para obtener predicciones acertadas sobre el comportamiento en bolsa del activo.

- Automatizar el proceso de cálculo por medio de un sistema de computación en *Python*.

- Mostrar los resultados a través de una web.

Capítulo 1 | International Consolidated Airlines Group, S.A

En esta parte del trabajo analizaremos la compañía a desde una perspectiva económica y de negocio. Posteriormente pasaremos a describir el desarrollo de la aplicación. Al final del capítulo habremos descrito la situación general de la compañía a nivel estratégico y seremos capaces de ver que compañia estamos analizando.

1.1 | Historia y evolución

Antes de entrar en materia propiamente dicha debemos entender cuales son los motivos históricos por los cuales IAG hoy es IAG y no son diez empresas diferentes. Para ello daremos un rápido paseo por la historia del grupo.

1.1.1 | Historia de Iberia

Quizá el nombre del grupo no nos indique mucho pero bajo esta denominación se encuentran dos de las compañías de aviación más relevantes del mundo, como son Iberia y British Airways.

Iberia, cuya razón social es *Iberia Líneas Aéreas de España, S. A. Operadora Unipersonal,* es una compañía de transporte aéreo de pasajeros española. Habiéndose fundado en 1927 es la más antigua compañía de este tipo en nuestro país así como una de las más longevas a nivel mundial. Fundada durante la dictadura de Primo de Rivera, operó hasta 1929 cuando el gobierno militar de entonces fundó una compañía estatal llamada CLASSA a la cual dicho gobierno otorgó el monopolio en materia de aviación. Tendrían que pasar 8 años, para que en el transcurso de la guerra civil española, volviese a operar como parte del bando franquista que más adelante consiguió el poder en España.

Iberia, inició una expansión global en materia de destinos parapetada bajo el monopolio de la hegemonía franquista instaurada durante más de 35 años. Iberia pertenecía al estado y tenía el monopolio del transporte lo que le confirió una gran ventaja competitiva en un

periodo donde las barreras de entrada eran altas y la competencia anecdótica. Con el paso de las décadas, multitud de rutas y aviones han pasado por sus filas. Desde los viejos Junkers 52 a los modernos europeos Airbus A330 pasando por los primeros Douglas DC-4 transoceánicos o los primeros aviones de dos pisos como los estadounidenses Boeing 747.

En el año 1991 debido a una mala gestión tanto de la compañía matriz como algunas de sus filiales, la empresa llegó una bancarrota para lo cual el estado tuvo que realizar dos ampliaciones de capital así como una reestructuración global de la compañía para evitar el desmantelamiento de la empresa.

Finalmente, después de 10 años de estancamiento a nivel estratégico la propietaria pública, el Instituto Nacional de Industria, decidió acabar con el carácter público de la compañía realizando así una privatización. Por medio de la salida a bolsa en 2001.

En la historia de Iberia siempre ha tenido un peso particular no solo el desempeño de la matriz en general si no de las filiales en particular. Iberia, entendida como el grupo ha tenido compañías regionales e internacionales desde las recientes Iberia Express o Vueling en modelo de filiales de bajo coste a la valenciana Air Nostrum en modelo de franquicia o la ya vendida Binter Canarias cuyo ámbito de actuación son las islas. No podemos hablar de Iberia sin mencionar las relaciones con los países latinoamericanos del otro lado del atlántico. Aerolíneas Argentinas, Viasa o Ladeco, compañías argentinas, venezolanas y chilenas respectivamente estuvieron en propiedad de Iberia, parcial o totalmente.

1.1.2 | Historia de British Airways

Si antes hemos mencionado a Iberia como la compañía de bandera española -la compañía de más relevancia de un país-, tenemos que hacer la propio con British Airways. British Airways es la compañía de bandera inglesa, fundada en 1974 con la fusión de varias compañías británicas British Overseas Airways Corporation y British European Airways Corporation así como Cambrian Airways o Northeast Airlines. Después de un par de años de pésimo desarrollo tuvo que ser nacionalizada por el gobierno de Reino Unido hasta que más tarde en 1981 empezó un proceso de privatización que culminó en 1987 con la salida a bolsa.

Con una historia similar a su homóloga española, tuvo desde su creación una vocación global incluyendo, de la misma manera, una cantidad de modelos de avión modernos focalizándose en rutas transoceánicas.

British Airways y Air France las únicas compañías aéreas del mundo con un avión supersónico entre sus filas aparte de Aeroflot -la compañía rusa- que también operó un avión supersónico. Estamos hablando del Concorde considerado por muchos como el cúlmen de la ingeniería aeronáutica ya no solo en su desarrollo en los años 70, hasta hoy en día.

La compañía inglesa ha tenido una larga lista de adquisiciones de terceras compañías. En 1987 compró a la segunda compañía aérea más grande del Reino Unido, British Caledonian. 5 años más tarde realizó lo propio con Dan Air. Así como la adquisición parcial de la australiana Qantas.

1.1.3 | Historia de IAG

Con el escueto resumen del compañias podemos apreciar las similitudes entre ambas compañías, una fundación de vocación pública pero con un desarrollo ulterior privado.

En 1999 ambas compañías fueron socias fundadoras de una alianza de aerolíneas comerciales conocida como One World por las cuales una serie de compañías aúnan sus ventajas estratégicas para conseguir un incremento del volumen de negocio con la operación de nuevas rutas y la creación de sinergias aprovechando el carácter estatal de cada una de las empresas integrantes.
En 2010, se creó una compañía en modelo holding, esto es una empresa cuya razón de ser es administrar otras empresas las cuales posee, como resultado de la fusión estratégica de Iberia y British Airways. La nueva empresa conocida como International Airlines Group cuya razón social en España es International Airlines Group S.A y muchas veces referida como IAG cotiza en la bolsa española como en la inglesa y pertenece al IBEX 35.

Las razones de la fusión entre las compañías es principalmente la necesidad de un gran volumen de negocio para poder ofrecer un servicio de calidad a precios mayores en comparación con las incipientes compañías de bajo coste.

Hay que entender esta fusión el el marco empresarial de la época. La fusión de la francesa Air France con la holandesa KLM y la compra de las líneas aéreas austriacas por parte de la alemana Lufthansa junto con la gran crisis mundial de 2008 hace que ambas compañías se vean abocadas a una fusión en orden de preservar la hegemonía aérea nacional y transnacional.

Es sencillo atisbar el porqué de Iberia con British Airways y no con otra compañía. Ambas compañías han participado conjuntamente en negocios como la ya citada alianza One World o la creación conjunta del sistema de reservas Amadeus junto con otras compañías europeas.

La fusión no estuvo ausente de críticas tanto sindicatos españoles e ingleses que acusaban a las contrapartes de haberse beneficiado de las rutas con más beneficios. La principal razón de estas críticas fue la ausencia del crecimiento en términos de contratación por parte de British Airways así como el despido de cerca de 4.000 empleados por parte de Iberia.

La manera en la que operan las compañías sigue siendo bajo la misma marca comercial. No hay una marca IAG si no que se mantienen las imágenes comerciales de las filiales. La fusión supuso una repartición de las rutas quedándose la española con las rutas de sudamérica y la británica las de norteamérica.

Con unas condiciones de mercado similares a las de su fundación, IAG sigue intentando su expansión por parte de la adquisición de terceras compañías como la oferta lanzada en 2015 para la compra de la aerolínea de bandera irlandesa Aer Lingus.

1.2 | Estructura del grupo

El grupo IAG es mucho más que Iberia y British Airways. Este grupo es un conglomerado de empresas y personas. Para entender la posición estratégica del grupo en la economía global debemos entender qué empresas pertenecen al mismo, cual es la flota con la cual mueven personas y mercancías a lo largo y ancho del globo, que personas dirigen la institución así como dónde están las diferentes sedes y que resultados tiene la compañía.

1.2.1 | Empresas del grupo

El grupo controla las siguientes compañías organizadas por países:

España:

- Iberia participada al 100%.

  Filiales:
  - Iberia Express participada al 100%.
  - Iberia Regional en modelo de franquicia.
  - Amadeus IT holding participada al 7,49%.

- Vueling participada al 100%.

Reino Unido

- British Airways (participada al 100%)

  Filiales:
  - BA Cityflyer participada al 100%.

- FlyBe participada al 15%.

Irlanda:

- Aer Lingus participada al 95%.

Francia:

- Openskies participada al 100%.

Sudáfrica:

- Comair participada al 18%.

Dinamarca:

- Sun-Air en modelo de franquicia.

Marruecos

- RAM (Royal Air Maroc) participada al 0,95%.

Europa:

- Eurostar participado al 10%

Estos datos de participaciones no proporcionan una información relevante sobre el interés de IAG a nivel estratégico en las compañías en las cuales participa.

Iberia

Iberia es la compañía de bandera española. Con base en España y sus principal centro de operación en Madrid vuela a más de 100 destinos en más de 40 países. A través de su base en Madrid vertebra todo el territorio nacional dotando de cobertura internacional a más de 20 aeropuertos españoles.

Entre las filiales que integran Iberia podemos destacar:

Iberia Express

La propuesta de bajo coste de la compañía española lanzada en 2012 como respuesta a la incipientes críticas sobre su estructura de costes y como una manera de competir con compañías internacionales cuyas rutas competían directamente con rutas históricas.

Con 20 aviones en la flota opera cerca de 40 destinos en la península, Canarias y Europa.

Air Nostrum

La compañía valenciana propiedad de la familia Serratosa a través de su grupo de inversión NEFINSA opera bajo un acuerdo de franquicia con la matriz. Por este acuerdo los vuelos de la compañía son puestos a la venta por Iberia y además los aviones pasan a llevar los colores corporativos de esta así como la denominación de Iberia regional.

Es importante notar la importancia de Air Nostrum en el plano nacional de Iberia la cual conecta las principales españolas ya que vuela a todas las comunidades autónomas así como a Melilla. Con más de 40 aviones la compañía opera cerca de 400 vuelos al día.

Amadeus IT Holding

Pese a que Iberia es una empresa que se dedica al transporte de pasajeros por vía aérea posee una significativa participación en esta empresa tecnológica. Fundada en 1987 entre las grandes aerolíneas europeas, la francesa Air France, la alemana Lufthansa, y la noruega SAS junto con la empresa española es la principal operadora tecnológica de reservas del mundo.

Vueling

La también española compañía aérea de bajo coste con base en Madrid fundada en 2004 y fusionada en 2009 con Clickair fue comprada por IAG en 2013. Con más de 105 aviones y 120 destinos a nivel europeo y norte de África.

Si antes hemos hablado de la importancia de Air Nostrum a nivel español, Vueling tiene un papel semejante a nivel europeo donde compite con otros operadores de bajo coste para alimentar las bases de Madrid, Barcelona o Londres.

British Airways

British Airways es la compañía de bandera británica. Tiene su principal base de operaciones en el aeropuerto de Londres Heathrow. Con 240 aviones vuela a 170 destinos en los 5 continentes.

Entre las filiales que integran Iberia podemos destacar:

BA Cityflyer

Desde 2007 esta subsidiaria perteneciente a British airways opera 28 destinos europeos con 18 aviones. Esta compañía tiene diversas funciones estratégicas entre las que destacan las operaciones de aeropuertos europeos con el aeropuerto de London City con aviones de reducida capacidad -menos de 100 pasajeros-.

Su principal ventaja competitiva es la de la cercanía a la *city* de Londres principal centro internacional financiero de Europa.

FlyBe

Pese a no estar integrada en el grupo IAG apenas tienen un 15% de participación. Esta compañía posee una importante relación con el grupo ya que ambas compañías tienen un

acuerdo de código compartido. Esto es que el grupo comercializa plazas y enlaces con las rutas de esta compañía.

El modelo de negocio es un modelo de bajo coste con bases en aeropuertos del Reino Unido.

Aer Lingus

La compañía de bandera Irlandesa fue adquirida por medio de una OPA en 2015 por parte de IAG. La compañía tiene un cincuentena de aviones dando servicio a 80 aeropuertos distintos en tres continentes.

La función de esta compañía en el grupo es cubrir el segmento de pasajeros irlandeses que buscan un servicio de calidad. Pese a un viraje hacia un modelo de bajo coste para competir con la también irlandesa Ryanair, después de su adquisición por parte del grupo IAG se busca cubrir el segmento exigente.

Openskies

Esta pequeña compañía francesa opera únicamente desde el aeropuerto francés de Paris Orly a Nueva York. La peculiaridad es que esta compañía solo oferta plazas en clase business cubriendo así el segmento más exclusivo. Este esfuerzo surge como alternativa al desaparecido concorde aunque no en tiempo si no en comodidad.

Apenas hay 4 aviones en la flota todos B757 de medio-largo alcance como luego veremos.

Comair

El grupo IAG posee una pequeña participación en esta compañía sudafricana. Esta compañía tiene un acuerdo con British Airways en modelo de franquicia. Algo similar a Air Nostrum pero en Sudáfrica. Estratégicamente esta compañía sirve como apoyo de los

vuelos en el país y dota de conexiones desde varios países de la zona Europa por medio de las conexiones que tienen Iberia y British Airways las bases de Ciudad del Cabo y Durban.

Sun-air

Esta compañía danesa funciona bajo un acuerdo de franquicia en Dinamarca y Escandinavia. Bajo la marca British Airways y 20 aviones provee de servicio a 14 destinos.

Al igual que la compañía sudafricana arriba mencionada, el sentido estratégico es mejorar las conexiones del grupo en el norte de Europa.

Royal Air Maroc

La compañía de bandera marroquí es poseída en una pequeña cantidad por IAG. Esta pequeña relación se traduce en una relación cordial donde RAM tiene rutas de código compartido con Iberia y donde permite a sus pasajeros emplear el sistema de recompensas de la compañía española.

Eurostar

No sorprende los intereses del grupo en otras compañías no directamente relacionadas con el transporte de pasajeros como actividad principal (véase Amadeus).

Eurostar es una compañía de transporte por vía férrea -trenes- gestionada por un consorcio de tres países (Bélgica, Francia y Reino Unido). En el caso de IAG, a su fundación British Airways poseía un 10% de las participaciones de la compañía que cuando la creación del grupo permanecieron en él.

1.2.2 | Flota

En la Tabla 1.1 se detalla la flota de IAG a fecha 31 de diciembre de 2015 y las futuras adquisiciones comprometidas.

Modelos	Activos fijos en balance	Arrendamientos operativos fuera de balance	Total 31 de diciembre de 2015	Futuras entregas	Opciones
Airbus A318	2	-	**2**	-	-
Airbus A319	34	34	**68**	2	-
Airbus A320	66	156	**222**	101	128
Airbus A321	26	17	**43**	18	-
Airbus A330-200	4	-	**4**	12	7
Airbus A330-300	3	9	**12**	2	-
Airbus A340-300	7	-	**7**	-	-
Airbus A340-600	4	13	**17**	-	-
Airbus A350	-	-	**-**	43	57
Airbus A380	10	-	**10**	2	7
Boeing 737-400	-	-	**-**	-	-

Boeing 747-400	40	-	**40**	-	-
Boeing 757-200	1	2	**3**	-	-
Boeing 767-300	12	-	**12**	-	-
Boeing 777-200	41	5	**46**	-	-
Boeing 777-300	9	3	**12**	-	-
Boeing 787-800	8	-	**8**	1	-
Boeing 787-900	5	-	**5**	16	18
Boeing 787-1000	-	-	**-**	12	-
Embraer E170	6	-	**6**	-	-
Embraer E190	9	3	**12**	2	15
Total del Grupo	**287**	**242**	**529**	**211**	**232**

Tabla 1.1: Flota de IAG

Fuente: http://www.es.iairgroup.com/phoenix.zhtml?c=240950&p=aboutfleet ; Elaboración propia

Familia Airbus

Airbus es una compañía europea. Su misión es fabricar aviones comerciales. Su avión más pequeño es el A318 y el más grande es el A380, el cual es el avión de pasajeros más grande del mundo.

Actualmente el grupo IAG opera 385 aviones de la compañía. Distribuidos en los siguientes grupos.

Corta y Media distancia:

A318: El avión más modesto de la familia de corto rango de Airbus. Puede albergar hasta 132 pasajeros. Sin embargo British Airways posee 2 de estos aviones con solo 32 asientos. La razón es poder operar vuelos desde el aeropuerto de Londres - City directamente a Norteamérica (Nueva York esencialmente) con una configuración de solo asientos de primera clase.

A319: Con 68 aviones IAG tiene la capacidad de transportar hasta un máximo de 156 pasajeros por avión. Estamos hablando de una versión más corta del A320. Hay diferentes versiones, siendo la más común la empleada para rutas nacionales o cortas (menos de 3 horas de duración). Sin embargo hay una versión de largo alcance con un autonomía de vuelo de 8.000 kilómetros que puede ser empleada para vuelos en versión ejecutiva con algo menos de 50 plazas.

A320: Con 222 unidades es el avión más numeroso dentro de la familia airbus en IAG y también el más numeroso en términos globales. Se emplea para rutas de media distancia a nivel europeo fundamentalmente. Hay diferentes variantes aunque el diseño original permanece prácticamente intacto. Con hasta 220 pasajeros constituye la piedra angular del grupo.

A321: La versión alargada del A320 tiene un mayor radio de alcance que sus hermanos pequeños dotándolo además de una capacidad de hasta 220 pasajeros. En su variante 200 es capaz de llegar a las 5700 kilómetros de distancia lo que la hace la versión con más alcance de la gama de media distancia de Airbus.

Los aviones arriba mencionados son variantes del modelo A320 siendo el A318 y A319 más cortos y el A321 más alargados. Solo tienen dos motores. Se consideran aviones de fuselaje

angosto porque solo tienen un pasillo. Están pensados para viajes cortos, menos de 4 horas y para un rango medio de 2500 kilómetros.

Larga distancia:

A330-200 y A330-300: Es la apuesta bimotor de Airbus para la larga distancia. Con la capacidad de transportar hasta 335 pasajeros. Están pensados para vuelos largos, generalmente transatlánticos y están certificados con la certificación ETOPS 180 lo cual les permite volar a 3 horas del aeropuerto más cercano. También se usa como avión de carga.

A340-300 y A340-600: Si antes hemos dicho que el A330 es la apuesta bimotor de Airbus el A340 es la apuesta cuatrimotor para la larga distancia. Con más capacidad que el A330 con algo más de 370 pasajeros de capacidad en su máxima versión. Al tener cuatro motores no está sometido a la certificación ETOPS.

Hay que notar que los aviones arriba mencionados (A330 y A340) son aviones desarrollados y con su apogeo en los 90'. Los aviones de los que ahora hablaremos son las propuestas de Airbus para el mismo segmento pero con un desarrollo moderno.

A350: Lanzado en 2015, nació con el propósito de sustituir el A330 como estandarte de la larga distancia bimotor. Su eficiencia en el motor le permite estar certificado para la certificación ETOPS 370 lo cual supone más del doble que su predecesor. Además permite transportar hasta 336 pasajeros.

Las aerolíneas del grupo IAG están esperando la recepción de la centena de pedidos realizados.

A380: El avión de pasajeros más grande del mundo y el segundo después del ruso Antonov An-225 permite transportar hasta 853 pasajeros con sus cuatro motores y 73 metros de longitud. Con sus 18.000 kilómetros de alcance permite recorrer las rutas más largas del mundo sin parar.

Los aviones de larga distancia de la familia Airbus son aviones de fuselaje ancho, es decir con más de un pasillo.

El grupo IAG opera 335 aviones de corta y media distancia por una parte y 40 aviones de larga distancia de Airbus.

En el Gráfico 1.1 se muestra la distribución de aviones tipo Airbus por tipo de rango de IAG.

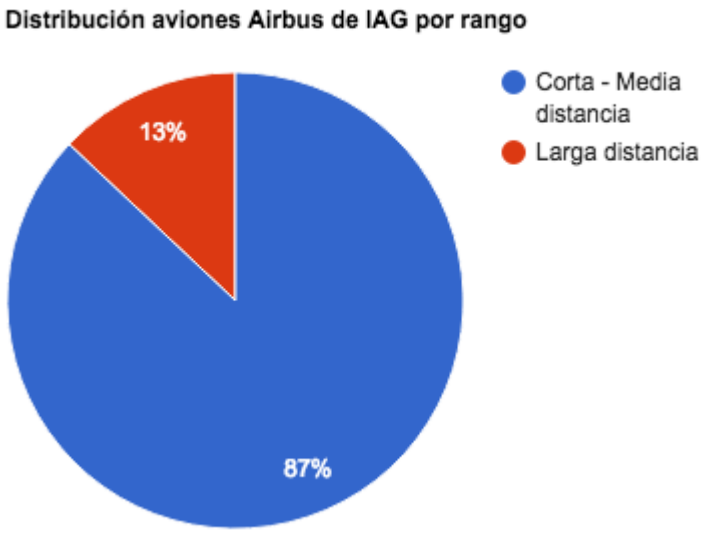

Gráfico 1.1: Distribución flota Airbus IAG por rango
Fuente: http://www.es.iairgroup.com/phoenix.zhtml?c=240950&p=aboutfleet ; Elaboración propia

Familia Boeing

Si antes hemos mencionado la familia Airbus como un fabricante de aviones comerciales generalistas, lo mismo tenemos que hacer con Boeing. Boeing es una compañía estadounidense que se fabrica aviones de todo rango.

Corta y Media distancia:

B737-400: Perteneciente al modelo 737, el modelo de avión de pasajeros más vendido del mundo, estamos ante el caso de la versión 400 la cual es un competidor directo de A320.

Puede transportar hasta 168 pasajeros. Y se emplea para rutas cortas. Es de fuselaje angosto. Ya no son utilizados.

B757-200: Pese a que el B757 se considera un avión de fuselaje angosto al solo tener un solo pasillo, tiene un rango de alcance elevado -hasta 7.200 kilómetros- lo cual le permite operar rutas transatlánticas sin ningún tipo de problemas. Puede transportar hasta 240 pasajeros. Es competidor directo del A321. Está certificado con ETOPS 120

Los modelos de corta y media distancia de Boeing son bimotores.

Larga distancia:

B767-300: Este avión de larga distancia con hasta casi 12.000 kilómetros de rango permite cruzar cualquier océano con hasta 375 pasajeros abordo. Está certificado con ETOPS 120. Compite con el A330.

B777-200 y B777-300: Este avión surgió de la necesidad de cubrir el rango entre el B767 y el B747. Puede transportar hasta 550 pasajeros en un rango de hasta 17.000 kilómetros. Es un avión ampliamente utilizado en rutas masificadas y/o largas. Es competencia directa del A380 y el A350. Está certificado con ETOPS 120

B787-900 y B787-1000: Desde el 2011, nacido del esfuerzo en encontrar la eficiencia surge este modelo donde puede transportar hasta 323 pasajeros durante 15.000 kilómetros con un 20% de lo que gasta en combustible un avión del mismo segmento. Compite directamente con el A330 y A350. Está certificado ETOPS 330.

B747-400: “La reina”, “Jumbo”, conocido con esos nombres, es un avión de hace 40 años que hoy en día sigue surcando el cielo. Hasta la entrada en servicio del A380 fue el avión de pasajeros más grande del mundo pudiendo transportar hasta 460 pasajeros durante 14.000 kilómetros. Este modelo es el único cuatrimotor de larga distancia de Boeing utilizado por las aerolíneas del grupo IAG. Compite directamente con los modelos de larga distancia de Boeing.

El grupo IAG opera 3 aviones de corta y media distancia por una parte y 111 aviones de larga distancia de Airbus.

En el Gráfico 1.2 se muestra la distribución de aviones tipo Boeing por tipo de rango de IAG.

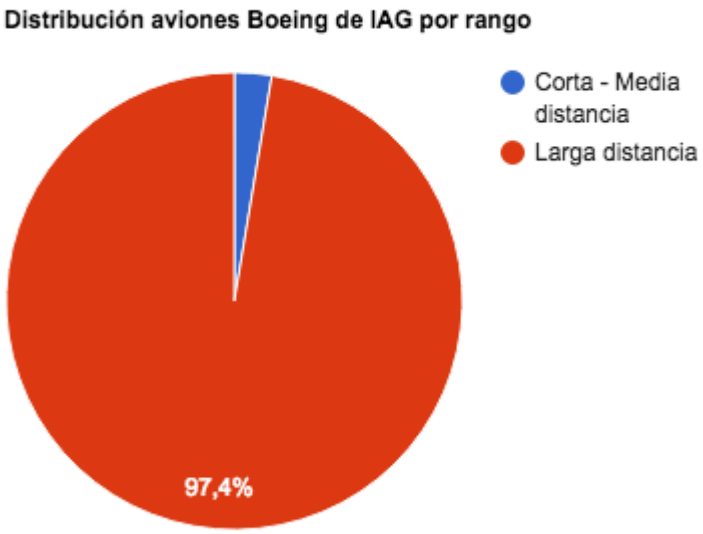

Gráfico 1.2: Distribución flota Boeing IAG por rango
Fuente: http://www.es.iairgroup.com/phoenix.zhtml?c=240950&p=aboutfleet ; Elaboración propia

Familia Embraer

A diferencia de Airbus y Boeing, Embraer es una compañía que solo fabrica aviones de corta y media distancia. Es una compañía brasileña y es la tercera compañía de fabricación de aviones del mundo por detrás de las ya citadas Airbus y Boeing.

E170: Es un avión pequeño que puede transportar hasta 80 pasajeros durante 3500 kilómetros. El grupo IAG posee 6 los cuales utiliza para rutas cuya ocupación no es especialmente elevada.

E190: Es la versión alargada del E170, puede transportar hasta 114 pasajeros durante 4200 kilómetros. El grupo IAG posee 12 los cuales utiliza para cubrir los segmentos donde el A319 es demasiado grande para su utilización.

Ambos modelos son bimotores.

Análisis estratégico por rango de alcance

En el Gráfico 1.3 se muestra la distribución de aviones por alcance de IAG.

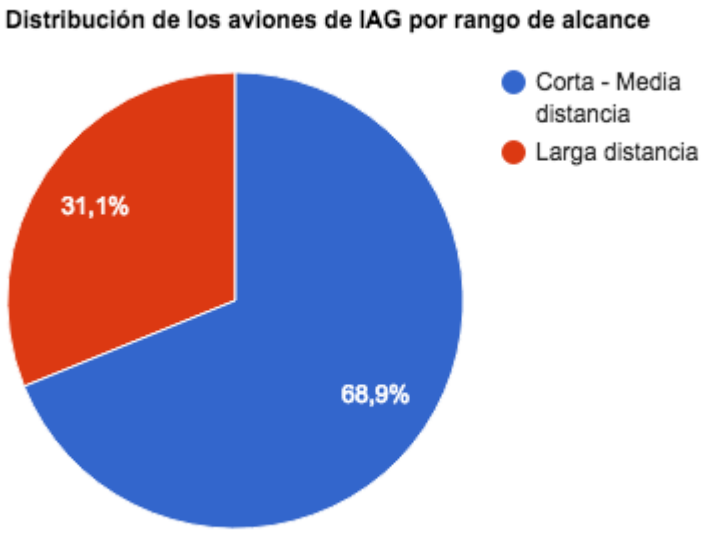

Gráfico 1.3: Distribución flota IAG por rango
Fuente: http://www.es.iairgroup.com/phoenix.zhtml?c=240950&p=aboutfleet ; Elaboración propia

Como podemos observar casi un 69% de los aviones del grupo están destinados a rutas de corta y media distancia -rutas menores de 4 horas- mientras que un 31% están destinados a rutas de larga distancia -rutas de más de cuatro horas-.

Esta dualidad refleja la dualidad de las verticales de negocio en el transporte de pasajeros que son rutas largas o cortas. Aunque suene obvio IAG tiene intereses y un gran volumen de negocio en ambas rutas pero hay compañías también grandes que no. Por ejemplo, Ryanair solo tiene aviones del segmento de corta y media distancia porque solo opera en ese

mercado, en cambio marcas del grupo IAG como OpenSkies o compañías como Virgin Atlantic operan solo en el mercado de larga distancia.

También podemos ver que la preferencia de los aviones en rutas de corta distancia se decanta hacia la europea Airbus mientras que la preferencia en larga distancia se decanta hacia la americana Boeing.

1.2.3 | Gobierno corporativo

La composición del Consejo de Administración se detalla en la Tabla 1.2.

Cargo	**Nombre**	**Nacionalidad**	**Primer nombramiento**	**Número de participaciones**
Presidente	Antonio Vázquez	España	Mayo de 2010	0.025%
Vicepresidente	Sir Martin Broughton	Reino Unido	Mayo de 2010	0.09%
Consejero Delegado	Willie Walsh	Irlanda	Mayo de 2010	0.081%
Consejero no Ejecutivo	César Alierta	España	Septiembre de 2010	0.049%
Consejero no Ejecutivo	Patrick Cescau	Francia	Septiembre de 2010	0%
Consejero no Ejecutivo	Enrique Dupuy De Lome	España	Junio de 2013	0.023%
Consejero no Ejecutivo	Baroness Kingsmill CBE	Reino Unido	Septiembre 2010	0%
Consejero no Ejecutivo	James Lawrence	Estados Unidos	Septiembre 2010	0.016%
Consejero no Ejecutivo	María Fernanda Mejía	México	Febrero 2014	0%

Consejero no Ejecutivo	Kieran Poynter	Reino Unido	Septiembre 2010	0%
Consejero no Ejecutivo	Dame Marjorie Scardino	Estados Unidos	Diciembre 2013	0%
Consejero no Ejecutivo	Alberto Terol	España	Junio 2013	0%

Tabla 1.2: Consejo de Administración IAG
Fuente: http://www.es.iairgroup.com/phoenix.zhtml?c=240950&p=irol-govboard2 ; Elaboración propia

El consejo de administración de una empresa, en este caso grupo, es el encargado de representar la sociedad. Sus miembros deberán velar por el correcto gobierno del grupo al que representan así como por el velado de los intereses de los accionistas.

Los miembros del consejo de administración son personas con una trayectoria importante dentro del mundo de los negocios. En este caso IAG cuenta con antiguos consejeros delegados de compañías varias como pueden ser Telefónica, Unilever, Rothschild North America, Altadis o Kellog. Además de antiguos altos cargos de Iberia y British Airways como son sus consejeros delegados o su director financiero.

Los miembros del consejo de administración poseen entre todos un 0,284% de las participaciones de la empresa. Para ser una gran empresa cotizada públicamente es un porcentaje sano que incita a la neutralidad de sus miembros para la buena marcha del grupo.

La composición del Comité de Dirección se detalla en la Tabla 1.3.

Cargo	**Nombre**	**Primer nombramiento**
Consejero Delegado	Willie Walsh	Mayo 2010
CEO de British Airways	Alex Cruz	Abril 2016
Presidente ejecutivo	Luis Gallego	Marzo 2013
Director de Estrategia	Robert Boyle	Enero 2011

Director Financiero	Enrique Dupuy De Lome	Junio de 2013
Director de Servicios Globales	Ignacio de Torres	Enero 2011
Jefa de Gabinete	Julia Simpson	Enero de 2011
Asesor Jurídico	Chris Haynes	Enero de 2011
CEO de Vueling	Javier Sanchez-Prieto	Abril 2016
CEO de IAG cargo	Andrew Crawley	Enero 2016
CEO Aer Lingus	Stephen Kavanagh	Marzo 2015

Tabla 1.3: Comité de dirección IAG
Fuente: http://www.es.iairgroup.com/phoenix.zhtml?c=240950&p=irol-mgt ; Elaboración propia

El comité de dirección es el órgano de la empresa que se encarga del gobierno de la misma. Se encarga del desarrollo de las operaciones del día a día y de la visión a largo plazo.

El comité de dirección de IAG está formado por **Willie Walsh** antiguo consejero delegado de British Airways y Aer Lingus así como por **Alex Cruz** actual consejero delegado de British Airways y antiguo ejecutivo de American Airlines , nombrado recientemente. También forman parte **Luis Gallego** antiguo consejero delegado de Iberia y ahora presidente ejecutivo así como **Enrique Dupuy De Lome** de Iberia, **Robert Boyle** de British Airways e **Ignacio de torres** antiguo consejero de Iberia los directores de Finanzas, Estrategia y Servicios globales. A nivel de asesoría forman parte la jefa del gabinete de comunicación **Julia Simpson** directora de comunicación de British Airways en el pasado y el de asesoría jurídica, **Chris Haynes,** abogado.

Finalmente también entran dentro de este comité los CEOs de Vueling, IAG cargo y Aer Lingus, **Javier Sanchez-Prieto**, **Andrew Crawley** y **Stephen Kavanagh**.

1.2.4 | Estructura de las sedes y cotización

Cuando Iberia y British Airways se fusionaron surgió la duda de donde establecer la sede de la empresa. Esto es de vital importancia puesto que en una empresa como esta debe haber

un tipo de balanza entre los accionistas de cada una de las empresas. Finalmente la decisión fue salomónica, la sede corporativa se situaría en Londres, Reino Unido y la sede social en Madrid.

La diferencia entre sede social y corporativa radica en la naturaleza de las actividades llevadas en cada uno de los lugares. Se denomina a sede social al lugar donde ocurren las decisiones del día a día mientras que se denomina a sede corporativa donde se derivan las obligaciones de ámbitos tributario-financieros.

Aparte, como ya hemos mencionado cada empresa del grupo mantiene una estructura social propia. Por lo que, por ejemplo, Vueling tiene su sede social en Barcelona, España o Aer Lingus tiene su sede social en Dublín, Irlanda.

Desde 2011, IAG cotiza en las bolsas de Barcelona, Bilbao, Madrid, Londres y Valencia. Pertenece al IBEX 35 y al FTSE UK Index Series. En España las acciones se negocian en el mercado continúo.

1.2.5 | Resultados y análisis financiero.

En este apartado realizaremos un análisis financiero sobre las cifras más significativas del grupo IAG.

Facturación (millones de euros)

En la Tabla 1.4 observamos los datos de facturación de IAG en los años 2014 y 2015.

2015	**2014**	**Diferencia**
22.858	20.170	2.688 (13,3%)

Tabla 1.4: Datos de facturación IAG

Fuente: http://www.iairgroup.com/phoenix.zhtml?c=240949&p=irol-reportsannual ; Elaboración propia

Gastos (millones de euros)

En la Tabla 1.5 observamos los datos de gastos de IAG en los años 2014 y 2015.

2015	2014	Diferencia
20.523	19.141	1.382 (7,2%)

Tabla 1.5: Datos de facturación IAG

Fuente: http://www.iairgroup.com/phoenix.zhtml?c=240949&p=irol-reportsannual ; Elaboración propia

Beneficio después de impuestos (millones de euros)

En la Tabla 1.6 observamos los datos de beneficio de IAG en los años 2014 y 2015.

2015	2014	Diferencia
1.539	1.003	536 (53,4%)

Tabla 1.6: Datos de facturación IAG

Fuente: http://www.iairgroup.com/phoenix.zhtml?c=240949&p=irol-reportsannual ; Elaboración propia

La facturación creció un 13.3%, a diferencia de los gastos que solo crecieron en un 7.2%. El beneficio dio un salto del 50% gracias a la diferencia de los dos indicadores ya nombrado. La contención del gasto hizo que el beneficio después de impuestos se disparase permitiendo a la compañía dar un dividendo.

División de la facturación por compañía (millones de euros)

En la Tabla 1.7 observamos los datos de facturación por compañía de IAG.

Compañia	British Airways	Iberia	Vueling	Aer Lingus
2015	15.862	4.412	1.962	622
2014	14.456	3.989	1.725	-
Diferencia	1.406 (9,7%)	423 (10.6%)	237 (13,7%)	-

Tabla 1.7: Datos de facturación por compañia IAG

Fuente: http://www.iairgroup.com/phoenix.zhtml?c=240949&p=irol-reportsannual ; Elaboración propia

En el Gráfico 1.4 observamos esa distribución porcentualmente en un gráfico.

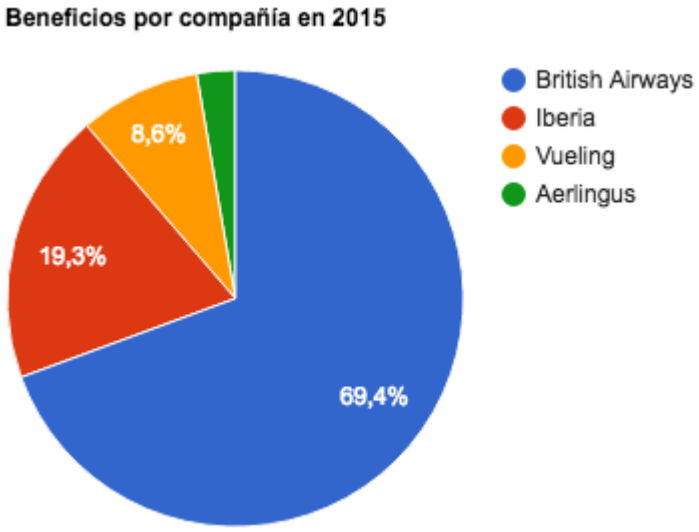

Gráfico 1.4: Beneficios por compañía en 2015

Fuente: http://www.iairgroup.com/phoenix.zhtml?c=240949&p=irol-reportsannual ; Elaboración propia

En cada una de las compañías están las subsidiarias de las mismas. Aer Lingus no tiene datos en el año 2014 porque fue comprada en el año 2015.

Como podemos observar casi el 70% de los ingresos vienen de British airways, esto refleja la importancia de tener la sede financiera de la compañía en Reino Unido.

Facturación por área geográfica

En la Tabla 1.8 observamos los datos de facturación por área geográfica de IAG en los años 2014 y 2015.

Área	**2015**	**2014**	**Diferencia**
Reino Unido	8.256	6.931	1.325 (19,11)
España	3.462	3.203	259 (8.08%)
USA	3.447	2.893	554 (19.14%)
Otra	7.693	7.143	550 (7.69%)

Tabla 1.8: Facturación por área geográfica

Fuente: http://www.iairgroup.com/phoenix.zhtml?c=240949&p=irol-reportsannual; Elaboración propia

En el Gráfico 1.5 observamos esa distribución porcentualmente en un gráfico para el año 2015.

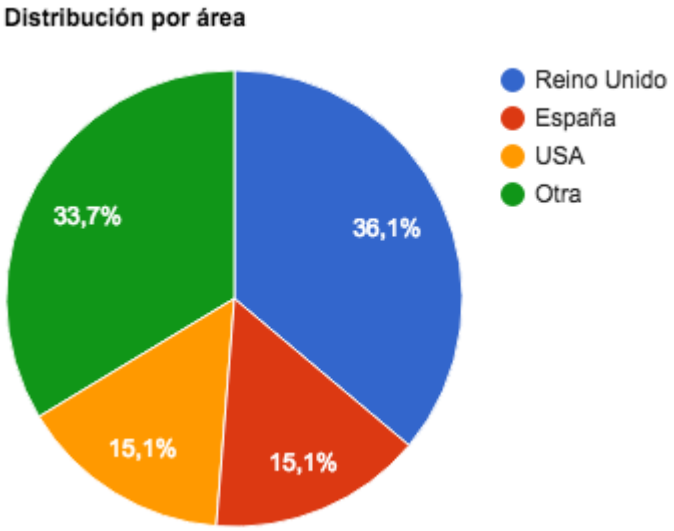

Gráfico 1.5: Distribución por área geográfica

Fuente: http://www.iairgroup.com/phoenix.zhtml?c=240949&p=irol-reportsannual ; Elaboración propia

Viendo los datos de la distribución de ingresos por compañía no es de extrañar que la facturación por área geográfica tengo sus principales focos en Reino Unido y España ya que es la sede de las principales bases de operaciones del grupo

Empleados

En la Tabla 1.9 observamos los datos sobre el número de empleados de IAG en los años 2014 y 2015.

2015	**2014**	**Diferencia**
60.862	59.484	1.378 (2.3%)

Tabla 1.9: Empleados IAG

Fuente: http://www.iairgroup.com/phoenix.zhtml?c=240949&p=irol-reportsannual ; Elaboración propia

Gastos de personal

En la Tabla 1.10 observamos los datos sobre los gastos de personal de IAG en los años 2014 y 2015.

2015	2014	Diferencia
4.905	4.585	320 (6.97%)

Tabla 1.10: Gastos de personal IAG

Fuente: http://www.iairgroup.com/phoenix.zhtml?c=240949&p=irol-reportsannual ; Elaboración propia

Por lo que respecta a la plantilla de IAG los empleados han crecido de 2014 a 2015 un 2.3% mientras que los costes asociados a ellos han crecido cerca del 7%.

Retorno sobre el capital invertido (ROIC)

En el Gráfico 1.6 observamos la evolución del ROIC de IAG desde 2011 hasta 2015.

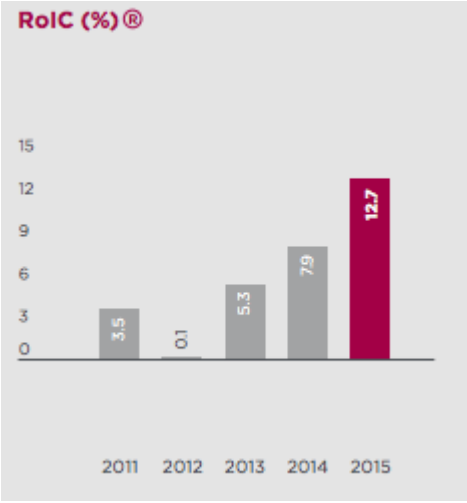

Gráfico 1.6: ROIC

Fuente: http://www.iairgroup.com/phoenix.zhtml?c=240949&p=irol-reportsannual

El ROIC es la rentabilidad que han conseguido los inversores sobre el dinero invertido en la compañia. Un 12.7% en 2015 es un excelente dato, muy por encima de la media del mercado. Este dato es especialmente relevante ya que IAG es cotiza en diversas bolsas

europeas y para ser atractiva necesita, naturalmente, tener un ROIC alto que atraiga a inversores.

EBITDAR

En el Gráfico 1.7 observamos la evolución del EBITDAR de IAG desde 2011 hasta 2015.

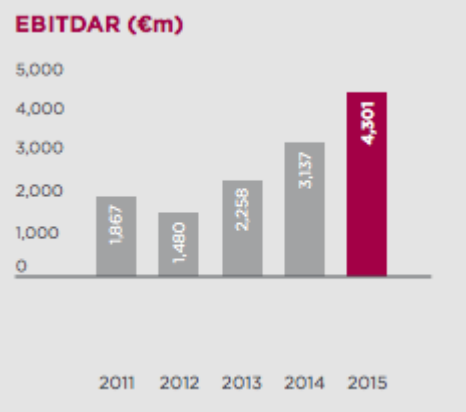

Gráfico 1.7: EBITDAR IAG

Fuente: http://www.iairgroup.com/phoenix.zhtml?c=240949&p=irol-reportsannual

El EBITDAR es el EBITDA que incluye gastos adicionales como por ejemplo el de arrendamiento de aeronaves en el caso de grupo. El EBITDAR de IAG en 2015 es de 4.300 millones y está en constante crecimiento desde 2012.

Este indicador es especialmente usado entre las aerolíneas y grupos hoteleros.

Flujo de caja libre del accionista

En el Gráfico 1.8 observamos la evolución del flujo de caja de IAG desde 2011 hasta 2015.

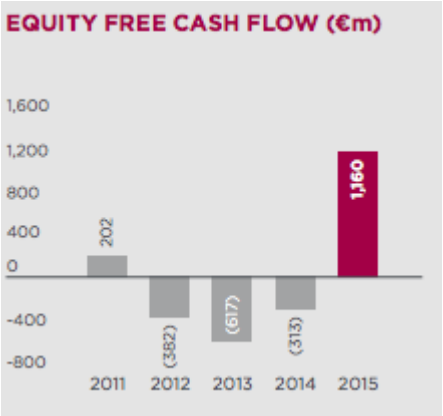

Gráfico 1.8: Flujo de caja libre del accionista IAG
Fuente: http://www.iairgroup.com/phoenix.zhtml?c=240949&p=irol-reportsannual

El flujo de caja libre del accionista es el dinero que hay a final del año para repartir dividendos o para repagar la deuda. Este indicador intenta reflejar el desacople de la capacidad de una empresa de su estructura financiera.

En las empresas cotizadas esta cantidad se emplea en repartir dividendos para los accionistas para como ya hemos dicho generar atractivo hacia los inversores.

El año 2015 ha sido un año especialmente bueno para IAG en este sentido obteniendo más de 1.000 millones.

A pesar de no realizar un extenso análisis financiero, que quedaría fuera del rango de este trabajo, podemos decir que estamos ante una empresa próspera a la vista de los indicadores selecciones y el recorrido en bolsa que revisaremos a continuación.

1.3 | Recorrido en bolsa

Para entender el desarrollo de la compañía en bolsa debemos observar cómo evolucionaron las empresas que hoy lo componen. Por ello, analizaremos tanto Iberia desde 2001 hasta 2011 (el periodo de cotización) como British Airways desde 1997 a 2010 y finalmente IAG desde su fundación en 2011 hasta actualmente.

1.3.1 | Iberia

En el Gráfico 1.9 observamos la evolución de la cotización de Iberia.

Gráfico 1.9: Cotización en bolsa Iberia
Fuente: http://www.invertia.com

En el Gráfico 1.9 puede verse que desde su creación Iberia tuvo un crecimiento sostenido hasta el pico más alto en 2007 antes de una grave caída a niveles de 2001 por la crisis financiera y global del 2008. Después comenzó a crecer hasta su fusión con British Airways a finales de 2010.

1.3.2 | British Airways

En el Gráfico 1.10 observamos la evolución de la cotización de British Airways.

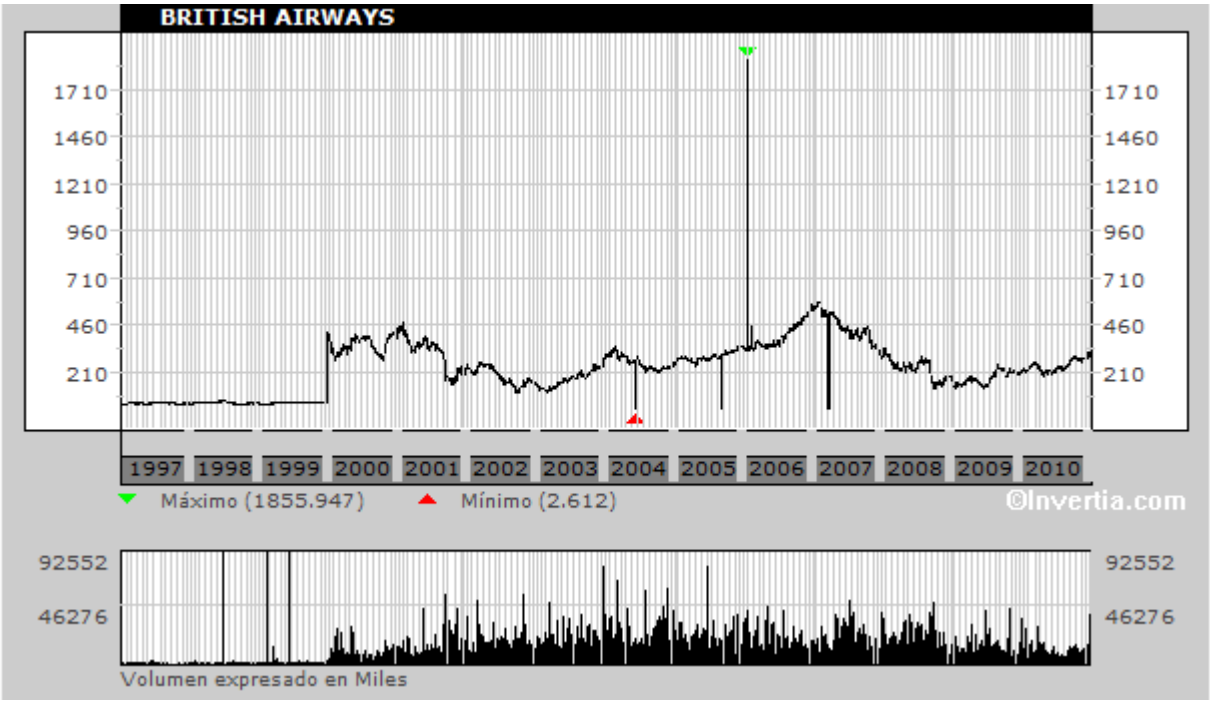

Gráfico 1.10: Cotización en bolsa British Airways
Fuente: http://www.invertia.com

En el Gráfico 1.9 puede verse que la trayectoria de British Airways en bolsa es bastante similar a la de Iberia. Aunque con una salida a bolsa temprana a la de Iberia, un máximo histórico alrededor del 2007 antes de una caída estrepitosa por la crisis y un crecimiento moderado antes de la fusión.

1.3.3 | IAG

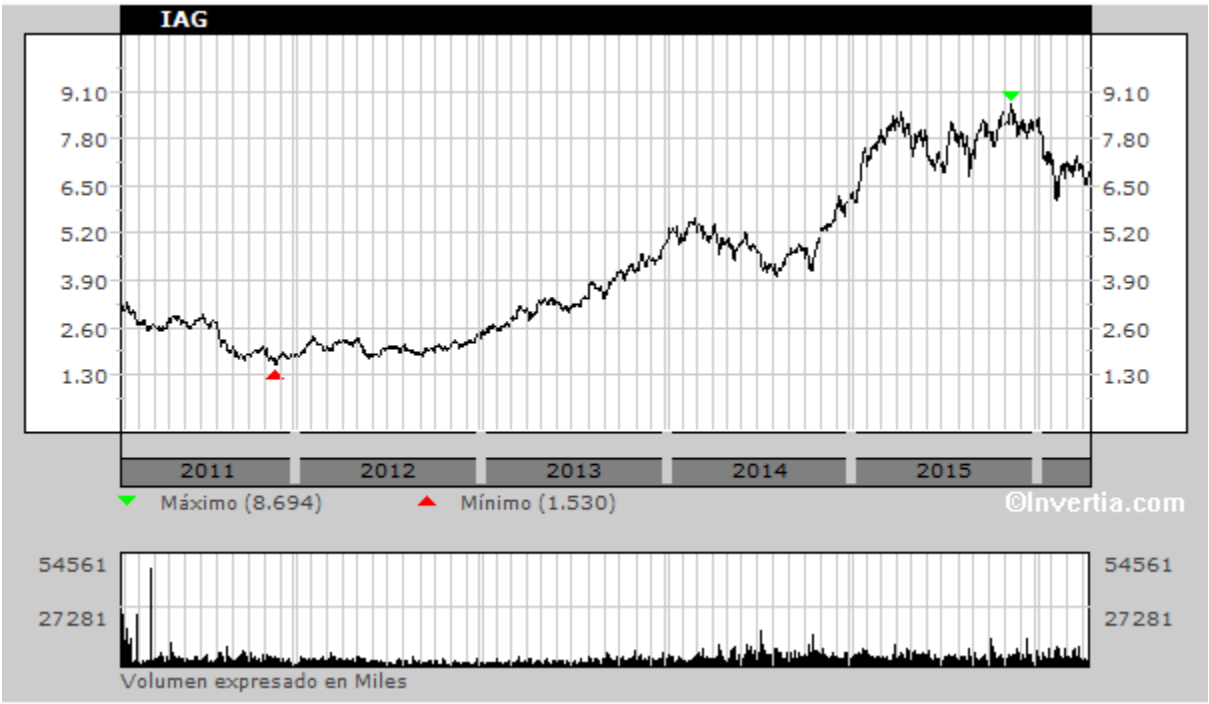

Gráfico 1.11: Cotización en bolsa IAG
Fuente: http://www.invertia.com

En el Gráfico 1.11 puede verse que IAG, cotizada desde 2011, sufrió un severo revés en bolsa que hizo que perdiese casi la mitad de su precio de cotización debido al escepticismo inicial de la alianza angloespañola aunque después de resolver esas dudas ha crecido imparablemente en bolsa hasta casi triplicar su precio de salida.

1.4 | Desarrollo estratégico

En el mundo hay infinidad de aerolíneas. La principal razón de esa cantidad de compañías dedicadas al transporte de pasajeros reside en que la mayoría de líneas aéreas se centran en el país donde tienen sus hubs -centros donde coliden los vuelos, también llamados bases-.

Por ejemplo, Iberia. Iberia tiene hubs en ciudades españolas pero no tiene hubs en Latinoamérica, esto es que sus aviones siempre vuelan desde las mismas ciudades

españolas. Sí, es cierto que tienen rutas fuera de España pero, por ejemplo, un avión de media distancia como el A320 con base en Madrid realiza vuelos de la siguiente forma:

- Vuelo de Madrid a Londres y de Londres a Madrid
- Vuelo de Madrid a Milán y de Milán a Madrid

El vuelo siempre pasa por, en este caso, Madrid y en las horas que no opera -de noche-, las pasa en el aeropuerto base que es Madrid. Pese a que hay compañías internacionales como Ryanair que pese a ser Irlandesa ponen un gran énfasis en abrir bases en múltiples países -Ryanair tiene 81 bases en más de 20 países- no es lo habitual en las conocidas como compañías de banderas. En la Figura 1.1 se detallan las rutas de Iberia desde su aeropuerto base en Madrid.

Figura 1.1: Rutas de Iberia desde Madrid

Fuente: https://www.oneworld.com/flights/where-we-fly

La imagen superior son las rutas que tiene Iberia desde Madrid -la principal base- al resto del mundo. Como podemos observar hay un gran flujo de vuelos a Centroamérica y América

del sur mientras que el continente asiático y oceanía son inexistentes así como la mayor parte de Estados Unidos.

En la Figura 1.2 se detallan los vuelos de British Airways desde su principal base que es Londres Heathrow, apreciamos como apenas hay vuelos a Centroamérica y América del sur mientras que hay una ingente cantidad de vuelos a Asia y Oceanía.

Figura 1.2: Rutas de British Airways desde Londres
Fuente: https://www.oneworld.com/flights/where-we-fly

En ambos casos hay una gran cantidad de vuelos a Europa y unos pocos a África los cuales ambas compañías operan desde Madrid y Londres respectivamente.

No es casualidad la repartición de rutas, cuando ambas compañías pasaron a ser una hubo una restructuración de rutas donde los destinos asiáticos y oceánicos pasaron a ser operados por la marca inglesa mientras que los latinoamericanos pasaron a ser operados por la marca española.

Esto ocurre porque los modelos de desarrollo de las compañías aéreas funcionan operando rutas donde el porcentaje de asientos vendidos es alto y donde la competencia perjudica a las empresas que operan estas rutas. El número de pasajeros no aumenta linealmente con respecto a la cantidad de oferta y esto desemboca en una guerra de precios que hace reducir el margen de beneficio.

A este efecto se le suma el modelo hexagonal de atracción de usuarios. Poniendo otra vez el ejemplo de Iberia. Iberia tiene multitud de rutas nacionales por ejemplo Valencia - Madrid con un 75% de capacidad media -esto es que de 100 plazas que tiene el avión, 75 van ocupadas y 25 libres- además la misma compañía tiene una ruta Madrid - Nueva York con un 85% de capacidad media.

Iberia quiere aumentar la capacidad media de ambos vuelos así como satisfacer las demandas de los consumidores valencianos que actualmente están volando a París para coger un vuelo a Nueva York. Por sí misma una ruta Valencia - Nueva York no sería rentable así que Iberia lanza un ofrece la ruta Valencia - Madrid - Nueva York con escala de dos horas en la capital para poder aumentar la capacidad media y satisfacer las demandas de los consumidores. Al asegurarse que estos clientes van a realizar dos rutas. Valencia - Madrid y Madrid - Nueva York Iberia puede permitirse lanzar una oferta especial del trayecto entero con un precio inferior al de ambos vuelos por separado.

Capítulo 2 | Desarrollo del Software

En este capítulo hablaré sobre los principios del desarrollo de software en general que he empleado durante la realización de esta aplicación. Además daré más información sobre la cultura del desarrollo de aplicaciones informáticas para dar una mayor visibilidad y justificación a mis decisiones.

2.1 | Modelos de desarrollo de Software

Aunque hay variantes, generalmente los modelos de desarrollo de software son dos, Desarrollo en cascada también conocido como *Waterfall* -cascada en inglés- o BDUF siglas de Big Design Up Front (Primero un gran diseño) o el conocido como desarrollo ágil. La razón por las que solo voy a explicar estos dos modelos es porque los demás modelos conocidos se agrupan entorno a variantes de estos dos o incluso mezclas. Por ejemplo, el desarrollo iterativo es considerado por algunos como un modelo pero la realidad es que el desarrollo iterativo puede ser empleado por el desarrollo en cascada y por el desarrollo ágil.

2.1.1 | Desarrollo en cascada (BDUF o Waterfall)

El desarrollo en cascada es el sistema tradicional de orientación a proyectos. Se compone de fases sucesivas que empiezan en un análisis de los requisitos y acaba con el despliegue de la aplicación así como un mantenimiento por los errores realizados en las fases anteriores.

El proceso de desarrollo habitual se muestra en la Figura 2.1.

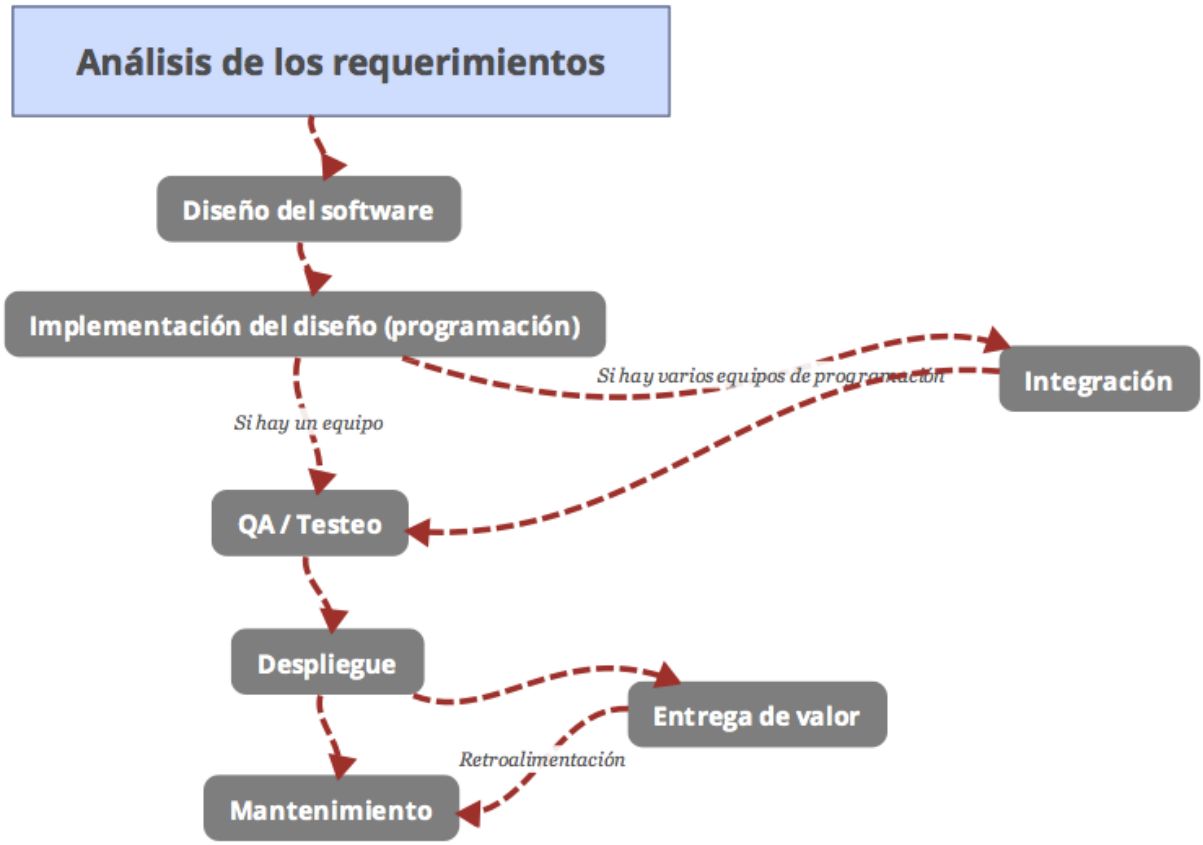

Figura 2.1: Flujo de trabajo BDUF
Fuente: Elaboración propia

El proceso empieza con el cliente diciéndole a un equipo de analistas sus necesidades. Los analistas transformarán esas necesidades en requerimientos que podrán ser ejecutados por los arquitectos y los programadores.

Esas piezas ejecutables serán recogidas por los arquitectos que diseñarán el software. Generarán piezas más pequeñas, en teoría, orquestadas que permitirá a los programadores escribir un código. Llegados a este punto puede haber un equipo de programadores o varios trabajando en paralelo. En caso de que haya varios equipos, habrá un paso más que es el

del proceso de integración. En este proceso se junta las piezas ejecutadas, el código escrito por los diferentes equipos.

Una vez desarrollado y si hiciera falta integrado el siguiente paso es tarea de los testers también llamados aseguradores de calidad. Este equipo se encarga de verificar que lo desarrollado por parte de los programadores se ajusta a lo especificado por los analistas. Además tienen como también como función principal preservar la calidad general del producto así como encontrar posibles fallas en el análisis efectuado por los analistas.

Finalmente un equipo de sistemas, generalmente ayudado por algún programador que haya participado en el proceso de desarrollo ayuda a efectuar lo que se conoce como despliegue que es básicamente la puesta en producción -en el aire- del producto desarrollado. Este es el conocido como punto de entrega de valor, ya que es el momento en el cual el producto puede ser disfrutado por el cliente.

Una vez se ha lanzado el producto y se ha verificado que el proyecto funciona según lo indicado el cliente ejercita la retroalimentación hacía un equipo de mantenimiento que se encargará de ajustar las imperfecciones que el cliente aprecia.

El desarrollo en cascada entendido como una sola iteración más el mantenimiento posterior tiene una serie de problemas los cuales nombraré:

- El cliente solo participa al principio y al final del proyecto, esto hace que no haya una retroalimentación y que el más mínimo fallo en la evaluación de los requisitos, en las pruebas o en el desarrollo desemboque en la no satisfacción de las necesidades finales del cliente.

- El proceso es vertical y de arriba a abajo. Cada eslabón solo puede comunicarse con su antecesor inmediato. Esto hace que haya una falta de comunicación. Los arquitectos son los que hablarán con los analistas, los programadores hablarán con los arquitectos y los testers hablarán con los programadores.

- No hay una compartimentación del trabajo. Al ser grupos de trabajo los que hacen cada paso (analistas, arquitectos, programadores, testers, sistemas y soporte) hacen que el conocimiento no permanezca en el desarrollo del proyecto ya que una vez finalizado cada paso el equipo de trabajo es reasignado a otro proyecto lo cual hace que la ventana temporal usualmente solo permite la comunicación entre el eslabón superior.

- El cliente no recibe valor hasta la entrega del producto final. Al entregar toda la aplicación se crea un espacio entre que el proyecto empieza hasta que el mismo es desplegado en los cuales el cliente no participa y no puede sacarle partido a la aplicación porque la misma solo existe en el marco del proyecto.

- Los proyectos tardan más por lo general, al realizar cada paso una única vez hace que cada paso sea visto con un mayor temor al error y por tanto cada fase tome más tiempo.

- El diseño del software y la programación lo hacen equipos diferentes. Esto suele conllevar desalineamiento y código rocambolesco debido a que hay problemas a los que solo puedes ver lidiando con ello.

- Si los equipos de programación lo están bien balanceados y con una cultura única de la programación puede dar un desarrollo asimétrico en cuanto a calidad y unos silos de conocimientos que pueden resultar fatales para el equipo de mantenimiento.

2.1.2 | Desarrollo ágil

El desarrollo ágil es una orientación más moderna al desarrollo de proyectos de software. Nacido y popularizado en los 90 es el estilo de desarrollo de proyectos más popular hoy en día por las empresas de software más importantes del mundo.

Ciertamente tiene una estructura similar al desarrollo en cascada pero tiene un gran cambio en cuanto al flujo de comunicación y a la ventana temporal de ejecución.

En la Figura 2.2 se muestra el flujo de trabajo ágil.

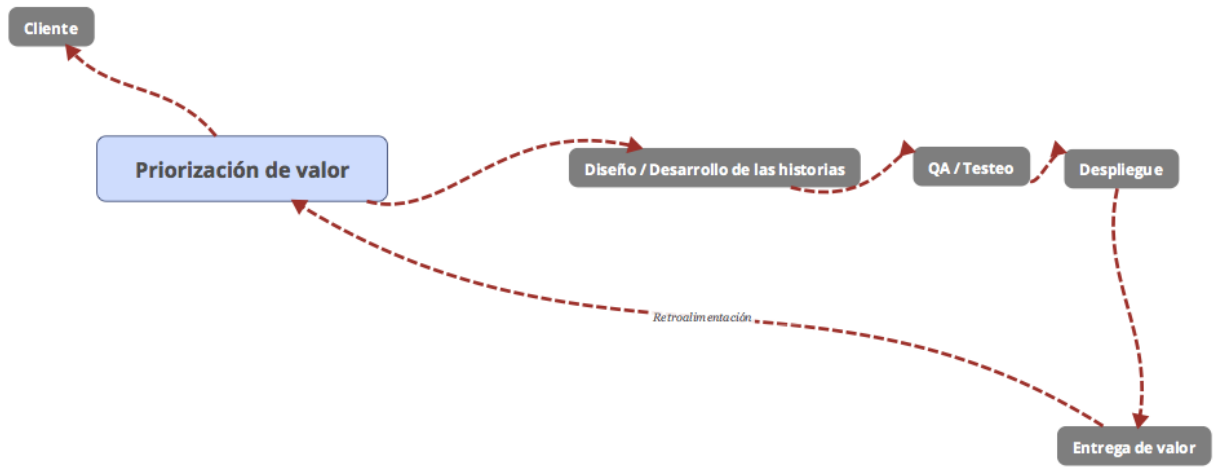

Figura 2.2: Flujo de trabajo ágil
Fuente: Elaboración propia

Generalmente hay un cliente que tiene una serie de necesidades, este cliente transmite esas necesidades a un equipo de producto que será el encargado de plasmar las necesidades de forma en la cual los desarrolladores podrán ejecutarlas y transformarlas en valor. Podríamos pensar que este equipo de producto tiene las funciones del equipo de analistas que vimos en el otro modelo de desarrollo pero no. Este equipo de producto se encarga de verificar que las necesidades son las correctas y de averiguar cuáles son más prioritarias.

Las conocidas como historias de usuario que son las mínimas partes accionables que generan valor para el cliente se trasladan a los desarrolladores. Por lo general este equipo de desarrolladores lo forman personas cuyo rol podría ser definido como arquitecto-desarrollador. Las mismas personas que diseñan el software son las que lo programan. Esas pequeñas historias pasan al siguiente eslabón.

El equipo de testers se encarga de verificar que el valor intencional del equipo de producto ha sido ejecutado correctamente por el equipo de desarrollo.

Finalmente se produce un despliegue que coincide con el punto de entrega de valor.

Hay que notar que aquí no hablamos de un proceso donde hay un todo si no como un proceso vivo donde ese todo se parte en las llamadas como historias de usuario que siguen un flujo corto e iterativo.

Los equipos permanecen ejecutando constantemente. Estos mismos equipos al estar en iteraciones continuas no se van a otro proyecto.

Algunas desventajas son:

- Si el proyecto solo tiene sentido como un todo, por ejemplo, lanzar un cohete a la luna. La entrega de valor solo tendrá sentido una vez se despliegue la totalidad de historias.

- Las personas tienen una gran responsabilidad al no tener unas estructuras burocráticas y rígidas. Depende mucho de la profesionalidad de los trabajadores.

- Al haber un gran flujo de comunicación hay que filtrar los mensajes y tener buenos métodos de comunicación para que no haya un desalineamiento en los equipos.

Generalmente los procesos de desarrollo ágil se dividen en los conocidos como **sprints** los cuales duran entre 1 y 4 semanas dependiendo de la cadencia de cada equipo y las necesidades del cliente. Los sprints son grupos de historias que serán trabajadas durante el periodo determinado.

Las historias se trabajan en un modelo de columnas llamado **kanban**, las historias en el proceso de desarrollo tienen por lo general tres estados (por hacer, trabajando en ello y hecho).

En la Figura 2.3 se detalla el proceso kanban.

Figura 2.3: Kanban
Fuente: Elaboración propia

Una historia se considera hecho cuando está desplegada.

La ventaja del kanban reside principalmente en que todos los miembros del equipo saben el estado actual de las historias del sprint.

2.1.3 | Aproximación del desarrollo de la aplicación dependiendo de la metodología

Para visualizar las diferencias entre las aproximaciones al desarrollo de software desarrollaré a grandes rasgos cómo atajar la ejecución del programa informático del cual versa este trabajo.

Al hacer el proyecto yo solo habrá etapas en las cuales tenga que hacer yo mismo varias cosas.

Desarrollo en cascada

Primera fase. Análisis de los requerimientos. En esta fase hablaría con mis tutores para ver las necesidades que tiene la aplicación. Descubriremos cosas como:

- Los datos de la cotización tienen que estar sacados dinámicamente de otro servicio en lugar de la alternativa que sería cada día insertar los datos a mano.

- Necesitaríamos tener una página web donde mostrar los datos de la predicción.

- Los modelos se tendrían que ajustar a unas modelizaciones matemáticas pre existentes.

- La página web mostrada debería de tener una serie de gráficas y tablas con los resultados.

- Sería interesante tener un almacenamiento a largo plazo de las predicciones.

- El sistema debería ser flexible para añadir nuevos modelos en el futuro.
-

Segunda fase. Diseño del software. En esta fase analizaría los requisitos y haría un diseño a grandes rasgos. En este caso me plantearía toda la estructura de la web. Generaría un esquema como en el que se muestra en la Figura 2.4.

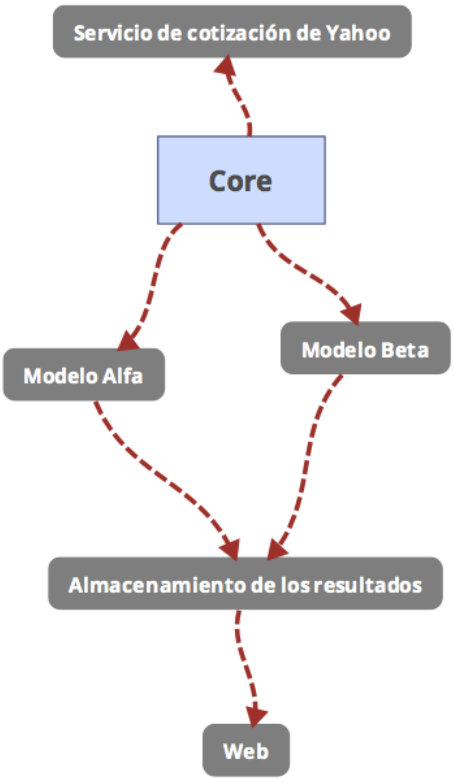

Figura 2.4: Esquema de la aplicación
Fuente: Elaboración propia

Este esquema nos dice que hay un servicio llamado *core* cuya función es la de orquesta los cálculos de los modelos alfa y beta. Una vez calculado esos datos se almacenarán en una base de datos de resultados. Esos datos se mostrarán luego por la web.

Aparte de las generalidades también especificaría los estándares, patrones y funcionalidades de cada parte de los subsistemas.

En la Figura 2.5 se ilustra el caso de cotización especificaría la forma de recoger los resultados.

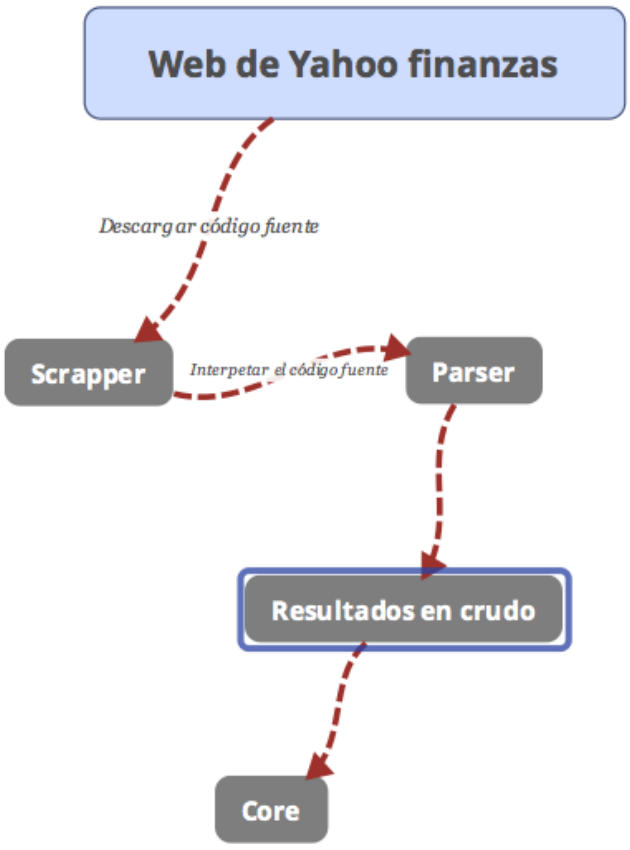

Figura 2.5: Esquema del servicio de recogida de datos
Fuente: Elaboración propia

En este caso habría un subservicio llamado *scrapper* que se encargaría de descargar el código fuente de la página de Yahoo finanzas. Ese código sería analizado por otro subsistema llamado *parser* que interpretaría el código fuente. Una vez hecho eso tendríamos los datos en crudo que serían enviados al servicio de *core*.

Después de realizar un análisis estructural de toda la aplicación pasaríamos a la siguiente etapa.

Tercera fase. Programación. Ejecutaría lo arriba estructurado. Siguiendo con el servicio de coger los datos de las cotizaciones haría programaría algo así. En pseudocódigo:

```
clase Scrapper
  funcion descargar_codigo_fuente
    web = 'http://yahoo.es/cotizacion/IAG.MC'
    retornar LibreriaHttp.get(web).codigo_fuente
  fin
fin

clase Parser
  function datos_en_crudo(codigo_fuente)
    retornar codigo_fuente.busca_texto_en('table tr td')
  fin
fin

clase ServicioCotizacionYahoo
  funcion ejecutar
    codigo_fuente = Scrapper.descargar_codigo_fuente
    datos = Parser.datos_en_crudo(codigo_fuente)
    retornar datos
  fin
fin
```

Una vez realizado lo propio con todos los sistemas y subsistemas pasaremos a la siguiente fase.

Cuarta fase. Aseguración de la calidad / Testeo. En esta fase con todo ya programado revisaría que todo lo programado se ajusta a los requerimientos. Una vez comprobado que todo está correcto y habiendo hecho los cambios necesarios pasaríamos a la fase de despliegue.

Quinta fase. Despliegue. En este caso sería hacer dos cosas:

- Subir la web al servidor para que esté disponible, configurando el dominio si hiciera falta.

- Configurar los conocidos como *cronjobs* para que el sistema se ejecute periódicamente.

Sexta fase. Mantenimiento. El cliente diría si aprecia alguna funcionalidad con un mal funcionamiento.

Desarrollo ágil

Una vez hemos visto cómo sería el desarrollo de este proyecto en cascada, vamos a a ver como sería con un desarrollo ágil.

Primera fase. Determinar con el cliente las funcionalidades que aportan valor. En esta primera fase entenderíamos las necesidades del cliente. En este caso listaríamos todas las funcionalidades que requiere el cliente.

Una vez tuviésemos todas las funcionalidades claras, haríamos historias de usuario para los desarrolladores. Estas historias serán las funcionalidades más pequeñas que aporten valor al cliente. En nuestro proyecto las podríamos partir así:

- Un servicio que saque el código fuente de la página de la página de yahoo

- Un servicio que saque los datos del código fuente de la página de yahoo.
- Un servicio que transmita los datos a otro servicio que procese los datos del modelo.
- Un servicio que procese los datos del modelo Alfa.
- Un servicio que procese los datos del modelo Beta.
- Una página web con una presentación.
- Añadir el resultado de los datos del modelo Alfa a la web.
- Añadir el resultado de los datos del modelo Beta a la web.

Luego pondríamos éstas historias en un kanban con una descripción y con lo indispensable para que los desarrolladores puedan trabajar. En la Figura 2.6, se muestra el kanban real.

Figura 2.6: Kanban real en trello
Fuente: http://trello.com ; Elaboración propia

Podríamos seguir añadiendo historias conforme el cliente nos fuese indicando mediante la retroalimentación.

Para la organización del tiempo, como he indicado anteriormente se trabaja en sprints. Lo bueno que tienen los sprints es que se puede organizar la fuerza del trabajo dependiendo

del contexto temporal en el que nos encontremos. Supongamos que tengo un plazo orientativo de tres semanas para hacer esto.

- La primera semana calculo que voy a tener mucho tiempo para trabajar en esto y mucha motivación.
- La segunda semana coincide que tengo que ir a una conferencia el Jueves y el Viernes.
- La tercera semana voy a tener mucho tiempo para trabajar en esto y no sé cuánta motivación tendré.

Sabiendo eso puedo clasificarme el trabajo en sprints. En la Figura 2.7 se muestra el kanban.

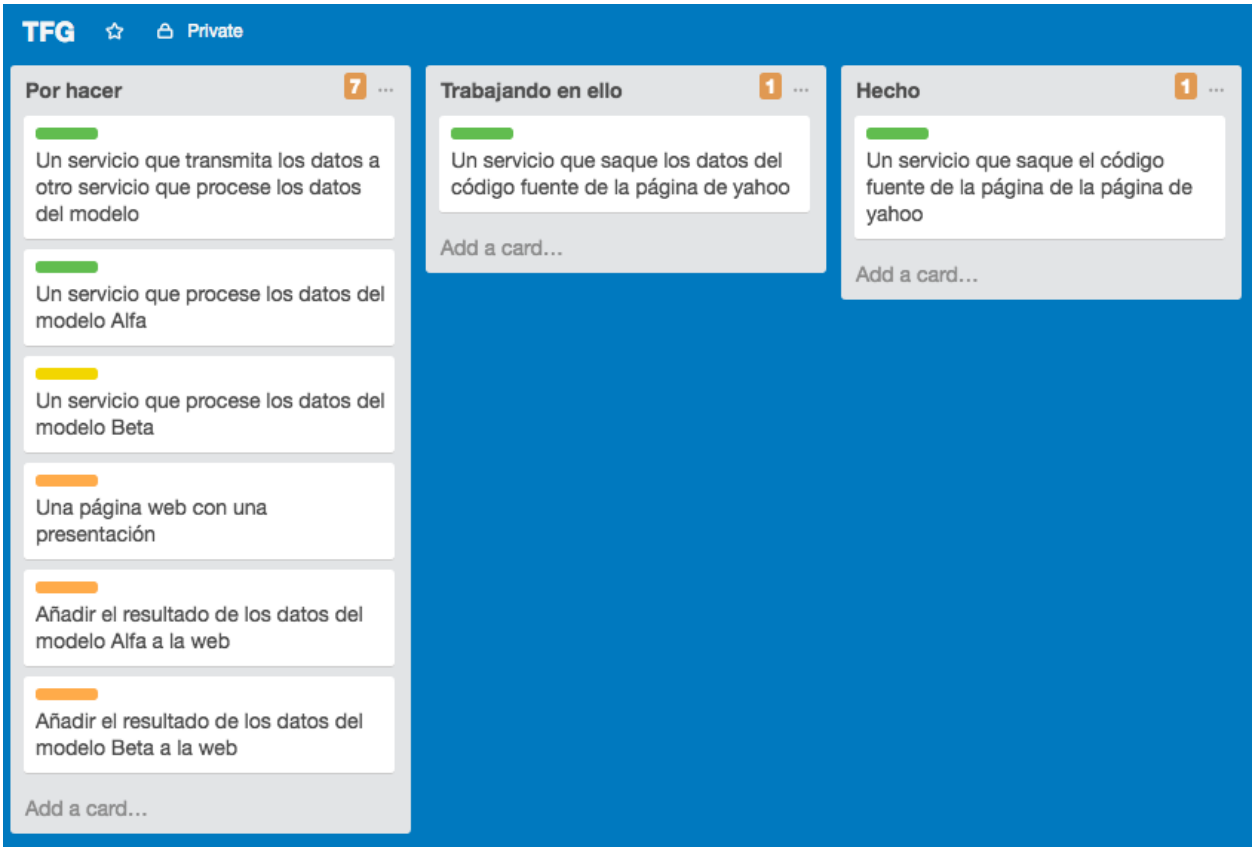

Figura 2.7: Kanban real en trello con etiquetas

Fuente: http://trello.com ; Elaboración propia

Las historias con el color verde serán trabajadas la primera semana, las que tienen el color amarillo serán trabajadas en la segunda semana y finalmente las de naranja la tercera semana.

La ventaja del kanban es el poder visual. Cualquiera que vea esa imagen sabe en que se está trabajando, que hay hecho, en que se va a trabajar y cuando.

Cada tarjeta se introduce cuando se reconoce esa necesidad. Los sprints están sujetos a cambios, lo cual dota de flexibilidad al proceso.

Segunda fase. Desarrollo. Una vez con el kanban montado no hay duda en lo que hay que trabajar. La siguiente tarjeta es la siguiente tarea a realizar.

Tercera fase. Testeo. Una vez se ha acabado el desarrollo de la tarjeta, el trabajo de los testers es el de verificar que efectivamente se ha hecho todo lo que se requería. No desde una perspectiva de control pero desde una perspectiva de aportar el valor dividida en dos fases:

- Verificación: verificar que efectivamente el comportamiento es el esperado por el cliente.
- Búsqueda: buscar casos extremos que no han sido valorados ni por el cliente, ni por el equipo de producto, ni por los desarrolladores.

Además para darle más valor es recomendable que el equipo de testeo tenga conocimientos de lo que va a testear previamente para tener una idea de las posibles carencias del software.

Cuarta fase. Despliegue. El equipo de sistemas ayudado del de desarrollo despliega cada historia. Es recomendable hacerlo mediante un sistema de **integración continúa** el cual se emplea para no tener grandes problemas en el despliegue de grandes historias. Básicamente es el momento de la generación de valor y hay que hacerla lo antes posible.

Finalmente la parte de retroalimentación conecta directamente al cliente con el departamento de producto los cuales tienen acceso a un flujo vivo que hará que los errores en el diseño, los requerimientos o el funcionamiento sean solucionados por un flujo habitual y conocido.

2.2 | Identificación de las carencias del software legado

En el mundo del desarrollo hay una frase popular que dice algo como: “El mejor software escrito es el que funciona hoy”. Esta frase justifica las decisiones tomadas en el desarrollo de una aplicación. No hay mal software por definición, hay que entender las aplicaciones en su contexto de desarrollo. Hay que entender quién las hizo, qué conocimiento tenía, cuál era su propósito y saber que las decisiones que tomó fueron fruto de ese contexto.

El software es una cosa que cuanto tienes la tarea de ampliar tiene implícito la tarea de mejorar. Se considera *código legacy* es decir código o software legado a la aplicación que no has programado tú pero tienes la tarea de trabajar con ella.

2.2.1 | Análisis del código legado

Cuando tienes ante ti un código que no has programado tu, tienes que entender la idiosincrasia en esa base de código y para ello tienes que analizar la forma en la que se ejecutó dicho proceso de desarrollo. Eso te permite saber a que te estás enfrentando y que vas a ser capaz de hacer.

Personalmente me gusta seguir un simple proceso para trabajar con código legado:

- Estructura de las archivos. La estructura de los archivos y las carpetas permite de un rápido vistazo ver cómo está estructurado el código. Puedes ver si el equipo que lo hizo era ordenado o no y te da una idea de que patrones se han empleado. En la programación web es realmente efectivo para ver si se ha utilizado el patrón MVC -modelo, vista, controlador-.

- Patrones empleados. Hay un libro conocido llamado “Design Patterns” que es la referencia en cuanto a los patrones más utilizados en programación. Algunos patrones como Template, Factory o Façade son fácilmente reconocibles viendo el código. Si el código está desarrollado con patrones con los cuales estás familiarizado será más sencillo el futuro desarrollo.

- Estilo de sintaxis. Para cada lenguaje de programación hay una guía de estilos más o menos oficial que es aceptada por la comunidad de ese lenguaje como la forma de escribir código *de facto*. Si el código sigue la guía de estilos oficial, las variaciones que introduzcas tendrán que seguir la misma guía. Si por otra parte el creador emplea otro estilo tendrás que adaptarte al estilo existente.

- Paradigmas empleados. Más adelante veremos los paradigmas en el mundo de la programación. Hay lenguajes que solo aceptan un paradigma como puede ser C o Java -imperativo y orientado a objetos respectivamente- pero en cambio hay otros como Python que son multiparadigma, en los estadios iniciales del análisis es necesario saber qué paradigma se ha empleado.

- Nombrado empleado. El nombrado puede ser descriptivo o no descriptivo. Un ejemplo de nombrado descriptivo es:

  HORAS_EN_UN_DIA = 24
  DIAS_EN_UNA_SEMANA = 7
  *HORAS_EN_UNA_SEMANA = HORAS_EN_UN_DIA * DIAS_EN_UNA_SEMANA*

  Cualquiera entiende que las horas en una semana son las horas en un día por las el número de días en una semana. Esto es un nombrado descriptivo. En cambio:

  A = 24
  B = 7
  *C = A * B*

No es un nombrado descriptivo y necesitarás mucho más contexto para saber de donde salen esos cálculos.

Después de la revisión del código se pueden observar las siguientes carencias:

- Nombrado no consistente. Las variables, métodos y clases tienen un nombre que no se corresponden con la realidad.

- Mejora de la legibilidad a través de comentarios. Esto es una carencia porque los comentarios no se actualizan con la misma frecuencia que con respecto al código que hacen referencia. A largo plazo esto provoca inconsistencias entre los comentarios y el código lo cual confunde al desarrollador.

- Ausencia de patrones definidos. Las partes de la aplicación no tienen un lógica definida y el contenido es altamente procedural.

- Ausencia de reglas de estilos oficiales. Todos los lenguajes de programación tienen unas convenciones por las cuales los desarrolladores escriben el código en ese lenguaje con una serie de reglas.

- Ausencia de tests. Eso hace difícil la mantenibilidad del código. En caso de fallo hay que probar manualmente todas la combinaciones para saber el punto de fallo.

- Ausencia de documentación. Generalmente las aplicaciones tienen una serie de textos llamada documentación en la cual explican los razonamientos de la implementación realizada.

- Acoplamiento entre las diferentes partes de la aplicación.

2.2.2 | Identificación de antipatrones

Los antipatrones en el diseño de software son una serie de errores en el diseño y la programación del software. Son errores recurrentes a todos los niveles bien sea por desconocimiento, bien sea porque las prioridades eran otras -por ejemplo, acabar lo antes posible-.

Algunos de los antipatrones reconocidos en la base de código legada son:

- Objeto todopoderoso. Se conoce a este antipatrón como el de un objeto que tiene la mayoría de responsabilidades de la aplicación.

- Clases con demasiado comportamiento. Toda la lógica de los modelos desde la abstracción a la generación del resultado están en el mismo fichero.

- Re-dependencia hay una codependencia entre componentes en los cuales no debería de haber.

- Acoplamiento. El abuso del paradigma imperativo hace que el correcto desarrollo de las funciones básicas de la aplicación estén en un riguroso orden.

- Ancla de barco. No eliminar las partes de la aplicación que ya no se necesitas.

- Números mágicos. Variables que están acopladas a los requerimientos y no se entiende por el contexto lo que representan

- Falta de un diseño consistente.

2.2.3 | Seteo de expectativas

Cuando desarrollamos software tenemos que tener claro lo que vamos a realizar. Es necesario tener en cuenta lo que el cliente espera del valor que aportamos.

Por ello antes de empezar el desarrollo hay que realizar un planificación sobre el alcance del valor que vamos a entregar.

El valor en el caso de este proyecto consiste fundamentalmente en esta serie de acciones:

- Eliminar la duplicación en la creación del modelo. Habilitar la plataforma para que en el caso de un tercer modelo no haya que modificar el comportamiento de los otros dos.

Seguir el principio de diseño llamado “abierto, cerrado” en el cual se basa en el que el software tiene que estar abierto a la extensión y cerrado a modificaciones.

- Rehacer el diseño de la web dotándolo de un diseño más vistoso.

- Desacoplar el *frontend* (la web que se ve) del backend *(los procesos de cálculo).*

- Dotar de tests para asegurar la mantenibilidad de las partes más sensibles de la plataforma.

- Desacoplar la información variable (el símbolo de la empresa seleccionada) a un único punto de cambio.

- Separar la parte contextual (como sacamos los datos y como los procesamos) de la parte matemática (como aplicamos el modelo). En definitiva la idea es que alguien que solo sepa hacer modelos matemáticos pueda añadir un modelo sin tener que preocuparse del resto de cosas.

- Modularización del código a patrones.

- Hacer que el código siga la guía de estilos PEP 8 que es el estándar *de facto* para Python. https://www.python.org/dev/peps/pep-0008/

- Eliminar las partes no usadas del código legado.

- Hacer gráficos dinámicos para mejorar su visualización.

- Eliminar los comentarios del código sin perder la legibilidad, mejorando el naming.

2.3 | Paradigmas de la programación y tipado del lenguaje

Los lenguajes de programación siguen uno o más paradigmas de programación. Esto son las formalizaciones de los procesos mentales plasmados en la forma en la que los lenguajes se estructuran.

Fundamentalmente hay tres: imperativa, orientada a objetos y funcional.

Imperativa

El exponente más claro es el lenguaje C. En este paradigma las distintas partes del código se ejecutan de forma sucesiva. Se entiende el funcionamiento como un proceso que empieza en un punto A y que acaba en un punto B.

Orientada a objetos

El exponente más claro es el lenguaje Java. Las distintas partes del código se encapsulan en los llamados objetos que tienen un estado. Posteriormente esos objetos son llamados al empezar la ejecución para obtener un resultado. Se entiende el funcionamiento como un proceso que empieza en un punto A el cual es un objeto y termina en un punto B habiendo entre medias objetos que llaman a objetos que tienen las diferentes responsabilidades.

Funcional

Está basado en la representación de las matemáticas a nivel de código con funciones que llaman a funciones. Se representa como un punto A que hace operaciones sobre sí mismo para llegar a un punto B.

En el caso de Python, estamos hablando de un lenguaje multiparadigma que tiene la posibilidad de ser ejecutado bajo la confluencia de los tres paradigmas. En el caso de otro de los lenguajes que hemos empleado, Javascript, también estamos ante un lenguaje multiparadigma.

Al igual que los lenguajes se dividen según su paradigma también se dividen según el tipado. El tipado de un lenguaje es la forma en la que se etiquetan sus variables. Desde una explicación no técnica podríamos decir que los lenguajes de programación cuando los lee el ordenador realiza una secuencia de procesos donde dice el tipo de cada componente y dependiendo de eso realizará una operación o dará un error.

La diferencia entre un tipado dinámico y uno estático es que en el primero no debemos de explicitar el tipo de las variables y en el segundo sí. Un ejemplo sencillo de un lenguaje dinámico:

```
SUMANDO_PRIMERO = 1
SUMANDO_SEGUNDO = 2

SUMA(SUMANDO_PRIMERO, SUMANDO_SEGUNDO) => 3
```

El proceso interno con el cual el ordenador lee este código es el siguiente:

```
Analizo SUMANDO_PRIMERO y descubro que es un número de cantidad 1
Analizo SUMANDO_SEGUNDO y descubro que es un número de cantidad 2
```

Mi función de suma solo admite números. Ambos son números, sumo las cantidades.

```
SUMANDO_PRIMERO = (NÚMERO) 1
SUMANDO_SEGUNDO = (NÚMERO) 2

SUMA(SUMANDO_PRIMERO, SUMANDO_SEGUNDO) => 3
```

El proceso interno con el cual el ordenador lee este código es el siguiente:

Sé que SUMANDO_PRIMERO es un número de cantidad 1
Sé que SUMANDO_SEGUNDO es un número de cantidad 2

Mi función de suma solo admite números. Ambos son números, sumo las cantidades.

Es una sutil diferencia pero tiene implicaciones en cuestiones de rendimiento. En el caso de ambos Python y Javascript estamos hablando de lenguajes dinámicos donde no hay que explicitar el tipo de cada variable.

2.4 | Principios utilizados

Los principios son unas reglas mínimas por las cuales los desarrolladores intentan manejar la complejidad con una serie de reglas aceptadas por ellos.

DRY

Del inglés *Don't Repeat Yourself* consiste en reutilizar las partes de la aplicación que se puedan. No hay ninguna necesidad de volver a escribir el código dos veces. Esto es realmente importante por dos cosas. La primera es el coste que tiene volver a escribir algo que ya has hecho y la segunda es el peligro que corres al hacer lo mismo en dos sitios diferentes.

SOLID

Es el acrónimo de Single Responsibility (Responsabilidad Única), Open - Closed (Abierto - Cerrado), Liskov Substitution (Sustitución de Liskov), Interface Segregation (Segregación de Interfaz) y Dependency Inversion (dependencia inversa).

Es el principio por el cual se basa la programación orientada a objetos. Cada uno de los componentes del acrónimo hace referencia a un principio.

- **Principio de responsabilidad única**. Una clase representa un componente mínimo que tiene un contexto en la realidad. En este caso el modelo matemático sería una

clase o una serie de clases, el extractor de los datos otro y así sucesivamente. Este principio dice que cada clase solo debe tener una responsabilidad.

Sería confuso que hubiese una clase que tuviese responsabilidades sobre cómo extraer los datos y cómo hacer el modelo matemático porque no en la realidad son responsabilidades de clases diferentes. Si hiciésemos lo que hace esta aplicación a mano por una parte iríamos a ver los datos de la cotización y luego haríamos los cálculos del modelo, pero no podríamos hacer eso en el mismo tiempo pensando de la misma manera.

- **Principio Abierto Cerrado.** Como hemos mencionado anteriormente este principio se basa en que las clases, los componentes, deben estar abiertos a la extensión pero cerrados a la modificación. Debemos de pensar en que es común en que el contexto de un software cambia pero debemos trabajar de forma en que los cambios que añadamos en el futuro extiendan el código que ya tenemos y no que lo modifiquen.

Un ejemplo de esto es si viene alguien a añadir otro modelo matemático en la aplicación no debe cambiar lo que ya está hecho, debe de ampliar la funcionalidad.

- **Principio de sustitución de Liskov.** Este principio está relacionado con el concepto de diseño por contrato por el cual los objetos de una aplicación pueden ser reemplazados por objetos que respondan a la misma interfaz.

Este concepto es un poco técnico pero básicamente representa la certeza de la ejecución. Es decir que si dos objetos son iguales y se comportan igual el comportamiento esperado en la aplicación debe ser igual.

Esto está relacionado con el concepto de **Duck Typing**, este principio se relaciona con un razonamiento que dice:

"Cuando veo un ave que camina como un pato, nada como un pato y suena como un pato, a esa ave yo la llamo un pato."

Es decir cuando un objeto tiene un comportamiento idéntico a otro eso significa que son el mismo objeto.

- **Principio de segregación de la interfaz.** Este principio dice que los objetos solo deberían poder usarse de la forma en la que se usan en este momento. Esto se relaciona con YAGNI que veremos luego.

Cuando estamos creando un objeto que representa una pieza de realidad, por ejemplo, el servicio que extrae las cotizaciones de la web de cotizaciones solo deberíamos conocer la forma en al que se utiliza en la aplicación. Esto es el punto de entrada al servicio que se le da una empresa y devuelve la cotización de los últimos treinta días. Para hacer eso tiene que ir a la web y buscar los resultados (dos pasos) si la aplicación no necesita específicamente ir a la web como un proceso individual entonces el único que debe saber cómo ir a la web es el servicio.

- **Principio de dependencia inversa.** Este principio dice que las dependencias entre los módulos deben de ser lo más finas posibles. Al estar las responsabilidades separadas apenas queda un hilo de interacción. En dos módulos claramente separados el cambio en los mismos no afectan mutuamente. En cambio en dos módulos altamente dependientes el cambio en uno sí que afecta al otro lo cual evita que podamos trabajar cómodamente en solo uno de ellos.

YAGNI

Es el acrónimo de la frase en inglés “You Ain’t Gonna Need It”. Esta frase refleja la forma de pensar en la cual las cosas solo se deben hacer en el estado ulterior en el que las vayas a necesitar.

Por ejemplo, uno puede imaginar cuánto quiera los requerimientos de la aplicación. Podríamos pensar que quizá necesitemos comparar los resultados de dos empresas y por ello decidimos hacer la aplicación para que en el futuro esté preparada para esa comparación. El tiempo pasa y nunca nos llegan a pedir esa comparación, en cambio nos

piden que implementemos otra cosa que no tiene nada que ver. Entonces se nos hace más complicado el implementar la nueva funcionalidad porque tenemos un código con funciones innecesarias.

KISS

Es el acrónimo de la frase en inglés “Keep It Simple, Stupid”. Esta frase refleja que debemos hacer los sistemas lo más sencillos posible. La sencillez es la razón por la que tenemos facilidad de implementar el cambio y es la principal razón de entrada a la comprensión del código.

Parce un principio sencillo pero muchas veces, junto con YAGNI perdemos el foco de lo que queremos hoy y empezamos a hacer cambios pensando en el futuro.

Tell Don’t Ask

Traducido al español significa: “Pregunta y no digas”. Es un principio que refleja el hecho de que los objetos deben responder a las preguntas que otros objetos le hacen en lugar de ser los otros objetos los que den información.

2.5 | Patrones empleados

A continuación enumerare y daré una breve explicación de los patrones empleados en el proyecto. Los patrones son estructuras organizativas del código ampliamente conocidas por la comunidad de desarrolladores.

Constructor

El patrón constructor es un patrón que permite separar la construcción de objetos complejos a través de diferentes representaciones. Se compone de un director que es el encargado de orquestar la construcción a través de un constructor. El director tiene la responsabilidad de ejecutar la creación de las distintas partes (responsabilidades) y devolver un resultado a través de una interfaz común. El constructor al final devuelve un objeto de un tipo diferente a

la clase que orquesta la construcción. En las Figuras 2.8 y 2.9 se muestran dos esquemas del patrón Constructor UML empleado.

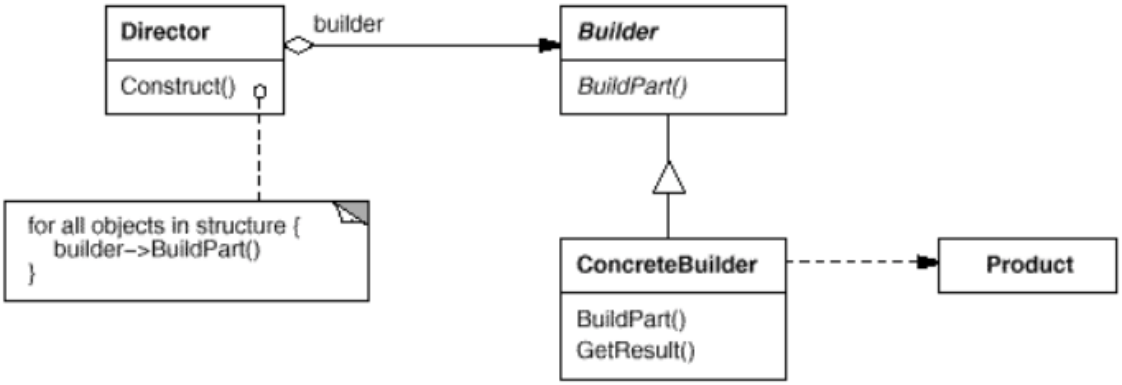

Figura 2.8: UML patrón constructor
Fuente: Design Patterns: Elements of Reusable Object-Oriented Software

Así por ejemplo un ModeloConstructor devolvería un ModeloConstructor cada vez hasta que se requiriese el modelo en una llamada tipo: ModeloConstructor.modelo en el cual se devolvería el modelo construido hasta ese momento.

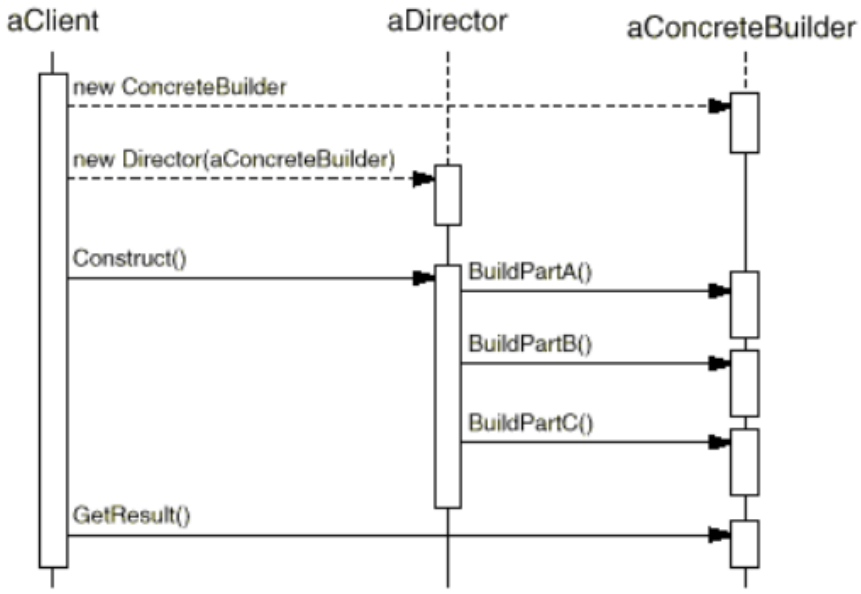

Figura 2.9: UML vertical patrón constructor
Fuente: Design Patterns: Elements of Reusable Object-Oriented Software

Un ejemplo de aplicación en nuestro sistema ha sido la construcción del gráfico. El gráfico se realiza a través de un patrón constructor. El director del gráfico hace la instanciación de las características básicas de un gráfico, por ejemplo en nuestro caso, un gráfico siempre tiene dos ejes y se trata de un gráfico lineal. Las sucesivas representaciones como la cotización, la predicción y los intervalos de confianza se separan en diferentes fases de la construcción en lugar de hacerlas todas juntas. Esto enlaza con el principio Abierto - Cerrado donde si quisiesemos añadir una representación no deberíamos modificar la clase si no extenderla añadiendo un elemento de construcción.

Fachada

La motivación de este patrón es definir una serie de interfaces en el sistema que puedan ser accesibles solo desde una interfaz común.

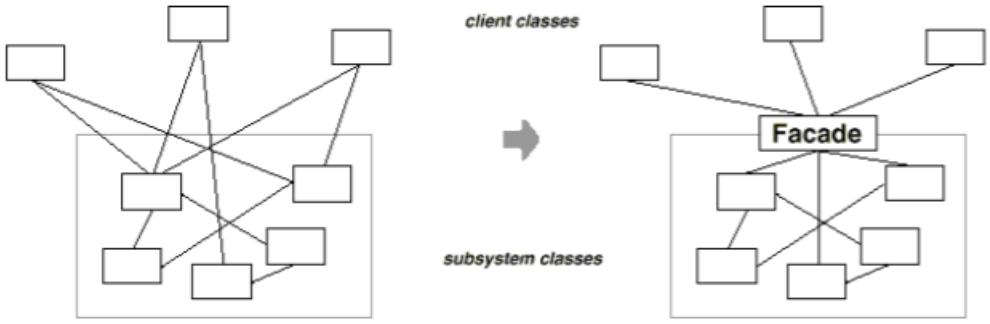

Figura 2.10: UML patrón fachada
Fuente: Design Patterns: Elements of Reusable Object-Oriented Software

En la Figura 2.10, a la izquierda tenemos 3 interfaces accediendo directamente a 4 servicios en la de la derecha las mismas tres interfaces acceden a una único servicio que es el encargado de determinar las relaciones entre los servicios.

Este patrón es realmente útil para reducir la complejidad en la aplicación. El ejemplo más claro en el que este patrón me ha ayudado a reducir la complejidad es en el tener un solo punto de entrada de cálculo.

Antes el punto de cálculo era un archivo de 500 líneas el cual debías ejecutar modificando ciertas variables, como el símbolo de la empresa, para obtener los resultados. Esto hacía complicado el entendimiento del sistema. En la Figura 2.11 se muestra el esquema del patrón fachada.

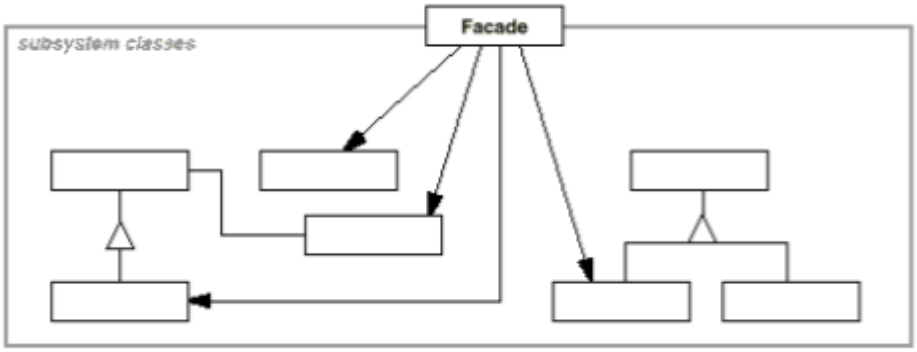

Figura 2.11: Representación patrón fachada
Fuente: Design Patterns: Elements of Reusable Object-Oriented Software

Una vez aplicada la fachada el punto de entrada de cálculo es un archivo de apenas diez líneas donde se ven las partes del proceso (ir a la web, coger los datos, limpiarlos, aplicar el modelo, formatearlos). De esta forma he atomizado la complejidad reduciendo la interrelación de los componentes.

Cadena de responsabilidad

Este patrón ayuda a manejar la necesidad del orden entre los componentes de la aplicación. En esta aplicación el orden es importante. De nada nos valdría calcular el modelo si no tenemos los datos actualizados.

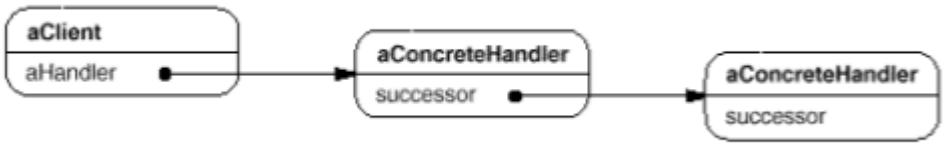

Figura 2.12: UML patrón cadena de responsabilidades
Fuente: Design Patterns: Elements of Reusable Object-Oriented Software

Así pues con el esquema mostrado en la Figura 2.12 podemos ver cómo funciona. Un cliente pasa su resultado a un servicio sucesivo que se encarga de saber cual es el siguiente paso. Como si de una cadena de montaje se tratase cada servicio solo sabe cual es el siguiente servicio que tiene que ejecutar.

Este patrón nos ayuda a evitar un antipatrón ya reconocido en la base del código que son servicios que conocen demasiado del proceso. El objeto todopoderoso que conoce cada una de las partes del proceso de cálculo puede ser evitado con la utilización de este patrón.

Este patrón es empleado en la creación de las tablas de resultados. Una vez tenemos el resultado de los modelos debemos de llenar las tablas de datos para que sean representados en la página web. Estos datos, en crudo, deber ser organizados por ello el servicio de creación de tablas recibe los datos y cada tabla sabe que necesita coger de los resultados para llenar la información. Una vez este subsistema tiene lo que necesita se lo pasa al siguiente y así sucesivamente hasta el final de la ejecución.

Estrategias

La motivación de utilizar estrategias es la de encapsular una familia de algoritmos para manejar la variabilidad de la aplicación. Esto ayuda a reducir la complejidad ciclomática.

La intención de emplear estrategias es hacerlo en el punto último donde diverge un proceso común. En la Figura 2.13 se muestra el esquema del patrón estrategias.

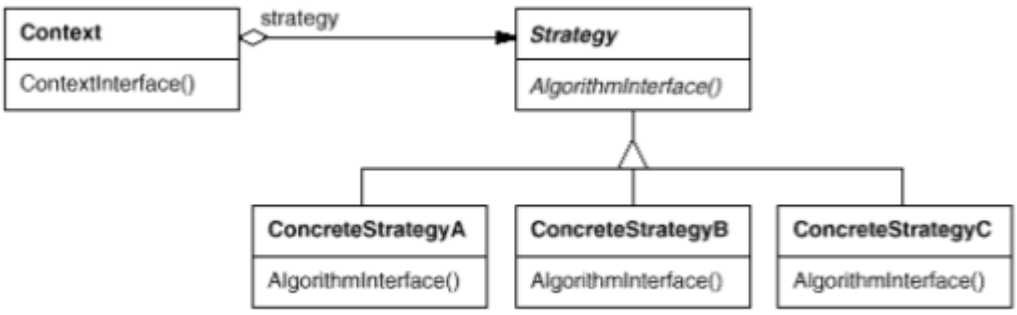

Figura 2.13: UML patrón estrategias

Fuente: Design Patterns: Elements of Reusable Object-Oriented Software

Para entender cómo aplicar este patrón debemos notar que existe un contexto y sobre ese contexto se emplea una estrategia. La variabilidad viene del contexto es decir si el contexto es X se aplica la estrategia A, si el contexto es Y se aplica la estrategia B…

Es ciertamente útil para eliminar las estructuras de control condicionales.

En el caso de nuestra aplicación, este patrón ha sido empleado para las distintas páginas de la web. Como al final solo difiere el contenido se emplea una estrategia dependiendo del modelo que se quiera ver.

Plantillas

El patrón de plantillas es útil para extraer la parte común de operaciones parecidas en diferentes objetos. Así se define un esqueleto que extrae la parte común de ciertas operaciones. En la Figura 2.14 se muestra el esquema del patrón plantillas.

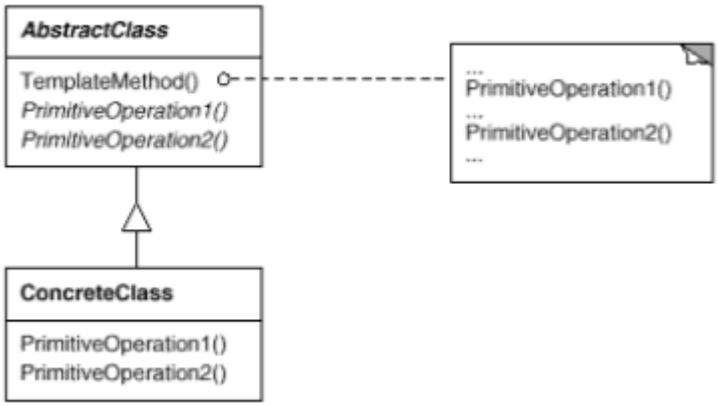

Figura 2.14: UML patrón plantillas
Fuente: Design Patterns: Elements of Reusable Object-Oriented Software

Hay un objeto que hereda de otro objeto del cual es tipo. Dando un ejemplo metafórico, imaginemos que tenemos que representar la la extracción de datos de distintas webs. Al final todas tienen una parte común (el acceso) a la web y una parte variable, la estructura en la cual esos datos se hayan. Bien, podemos extraer la parte común a una plantilla.

En el caso de nuestra aplicación se emplean las plantillas en el caso de los modelos, es decir, se ha extraído la parte común de los modelos, así cuando venga la siguiente persona a añadir un modelo ya sabe cómo tiene que utilizarlo aislando la carga matemática.

Comandos

El patrón comando sirve para representar las peticiones en forma de código, es decir para encapsular las acciones y peticiones.

Fundamentalmente hay un objeto comando con un método que dispara la ejecución del mismo. Este método es potente porque se puede adaptar al dominio del contexto así por ejemplo al servicio de Modelo lo podemos ejecutar a través de una interfaz simple llamada calcular así se mejora la legibilidad del código al tener un instrucción que se articula de la siguiente forma: Modelo.calcular. En la Figura 2.15 se muestra el esquema del patrón comandos.

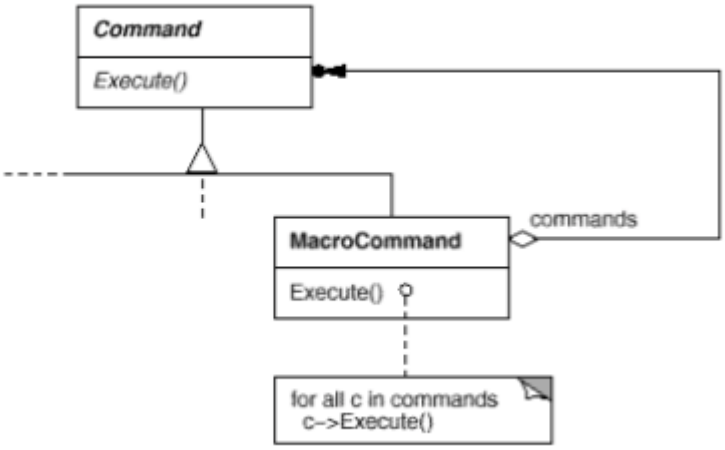

Figura 2.15: UML patrón comandos
Fuente: Design Patterns: Elements of Reusable Object-Oriented Software

En el caso de nuestra aplicación empleamos el cálculo de los modelos en forma de comandos donde dotamos al servicio de cálculo de una independencia, agrupamos la complejidad en subsistemas a los cuales accedemos a través de una fachada y lo invocamos con la finalidad concreta de obtener una serie de resultados.

Composición

El patrón de composición maneja la complejidad al componer estructuras en forma de árbol. Cada componente tiene una serie de nodos. En la Figura 2.13 se muestra el esquema del patrón composición.

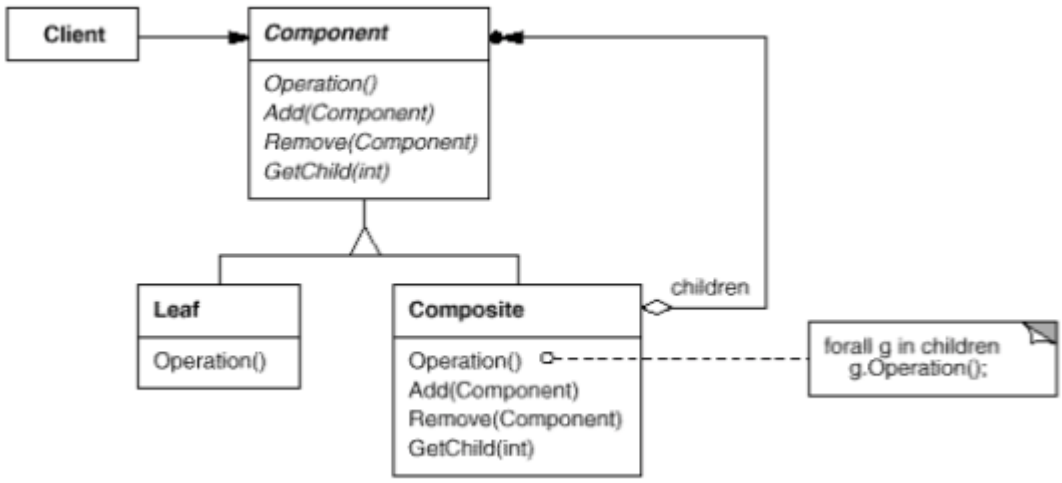

Figura 2.16: UML patrón composición
Fuente: Design Patterns: Elements of Reusable Object-Oriented Software

Esto hace posible que atajar la variabilidad representando los servicios complejos como caminos definidos. Es ciertamente útil cuando dicha variabilidad se torna en una gran cantidad de caminos que pueden tomar los elementos desde el principio. El aplicar este patrón permite que se puede ver la estructura de un subcomponente y que no haya caminos ocultos.

El caso más claro de composición es el caso del HTML. Al usar HTML estamos usando inherentemente composición ya que cada etiqueta HTML (nodo) tiene una serie de propiedades (hojas) que bien pueden ser del propio nodo o bien puede contenter otro nodo. Esa hoja sería hoja del nodo pero no en sí mismo que podría albergar otras.

Singularidad

El patrón singularidad proporciona una forma de tener un objeto inmutable durante toda la aplicación. Esto es una gran ventaja para objetos cuyo comportamiento es conocido pero no

se requiere su uso de diversas formas en la aplicación. En la Figura 2.17 se muestra el esquema del patrón singularidad.

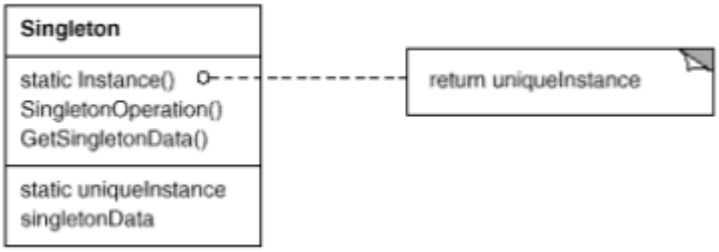

Figura 2.17: UML patrón singleton
Fuente: Design Patterns: Elements of Reusable Object-Oriented Software

Es un patrón muy sencillo cuya función es la devolver una instancia del mismo objeto (no la misma clase) cada vez que se emplea.

Este patrón es utilizado por la aplicación para el acceso a páginas web externas, en este caso Yahoo. Utilizamos el cliente HTTP de Python que es una singularidad.

Por último cabe mencionar que los patrones no son absolutos ni exclusivos. Diferentes patrones se pueden emplear para un mismo problema y diferentes patrones (más de uno) se pueden emplear para el mismo problema.

No es raro emplear diversos patrones. En el caso de nuestra aplicación se utilizan los patrones descritos de diversas formas. Por ejemplo para extraer los datos de las cotizaciones se emplea un patrón de cadena de responsabilidad y de fachada. Para aplicar los modelos se emplean estrategias, plantillas, fachadas y comandos.

La razón de empleo de estos patrones es la utilización de estructuras conocidas por los desarrolladores.

Aunque pueda parecer que los patrones atienden a un uso técnico, no es así, los patrones atienden a un propósito. El propósito es representar la realidad de forma en la que el

software generado tenga la máxima similaridad al contexto que el software intenta automatizar. Debemos entender la aplicación como la sustitución de un trabajo manual por uno automático. Y entonces podremos ver la utilidad que tienen estos patrones.

2.6 | Tecnologías empleadas

Para la realización de la plataforma hemos empleado diversos lenguajes de marcado y programación así como componentes los cuales describiremos brevemente en este apartado.

Es necesario notar que todas las tecnologías empleadas son tecnologías libres que no corresponden a una empresa y/o entidad. Esto es sumamente importante para mantener un código neutro y perdurable en el tiempo ya que no te ata a las decisiones que una entidad privada toma.

2.6.1 | Python

Es un lenguaje de programación creado por el holandés Guido van Rossum en la década de los 80. Es uno de los lenguajes de programación más populares de todos los tiempos. Es un lenguaje libre ya que admite cambios por cualquier interesado y puede ser utilizado sin ningún tipo de límite.

Como ya hemos mencionado con anterioridad es un lenguaje multiparadigma ya que admite programación orientada a objetos, funcional e imperativa. Además tiene un tipado dinámico y se considera multiplataforma porque puede ser ejecutado en múltiples sistemas operativos.

Tiene una clara intención de proporcionar una sintaxis sencilla que haga que haga fácil la lectura del código.

Este lenguaje ha sido empleado para la modelización y extracción de los datos en los procesos internos. El 90% del trabajo ha sido empleando este lenguaje.

2.6.2 | JavaScript

Al igual que Python es un lenguaje interpretado, multiparadigma y de tipado dinámico. Nacido en el año 1995 desde el seno de la ya difunta Netscape fue empleado durante muchos años como el principal lenguaje de navegador. Esto es que el navegador interpreta el código para ejecutarlo en tu ordenador en lugar de en un servidor.

Con el paso de los años su uso se fue popularizando y es el lenguaje más popular de los últimos años.

Se emplea en la aplicación para renderizar todos los datos y mostrarlos por la web. Se emplea un motor de plantillas que hace posible reutilizar todo el contenido.

2.6.3 | HTML

Es el acrónimo de HyperText Markup Language que traducido al castellano es: “Lenguaje de marcado de hipertexto” como su nombre indica es un lenguaje de marcado que se emplea para organizar el contenido de las páginas web.

Todas las páginas web utilizan HTML para mostrar sus datos y es el estándar que leen todos los navegadores. Hay diferentes versiones siendo la más reciente ls 5.

En la plataforma se emplea HTML5 para servir todos los datos haciendo posible que todos los ordenadores sean capaces de visualizar dichos datos con corrección.

2.6.4 | CSS

CSS es el acrónimo de Cascading Style Sheets que en español significa Hoja de estilos en cascada.

Si antes hemos mencionado al HTML como el lenguaje de marcado que organiza el contenido de las páginas web con el CSS estamos ante el lenguaje de marcado que da los colores y la forma a todo el contenido de las páginas web.

Es el estándar que emplean todos los navegadores para dotar de estilo al contenido que las páginas muestran.

Desde su creación en 1995 se han publicado tres versiones siendo la más actual la conocida como CSS3. En la plataforma empleamos la última versión para dar estilo al contenido mostrado.

2.6.5 | SQL

SQL es el acrónimo de Structured Query Language se considera un lenguaje de tipo declarativo. Se emplea para realizar operaciones sobre una base de datos.

Creado en 1974 su principal función es poder insertar y extraer información de una base de datos.

Hay variantes dependiendo de qué base de datos emplees. En el caso de la aplicación empleamos MySQL que es una base de datos relacional.

En la aplicación se usan la base datos para persistir los datos de los cálculos y para mostrarlos luego.

Capítulo 3 | Modelos matemáticos

En este capítulo se presentan las herramientas estocásticas que se requerirán en el siguiente capítulo para calcular la solución del Modelo Log-Normal y sus principales propiedades estadísticas. En primer lugar introduciremos un tipo particular de proceso estocástico que jugará un rol muy importante en todo el desarrollo teórico del Modelo Log-Normal, el proceso de Wiener o también denominado Movimiento Browniano. Partiremos de su definición y enunciaremos únicamente las propiedades estadísticas que posee y que se requerirán posteriormente. A continuación, se explicará cómo se pueden obtener simulaciones del proceso de Wiener, lo cual se necesitará cuando se aplique el Modelo Log-Normal en la modelización de un activo subyacente para poder implementar un método tipo Monte Carlo para realizar predicciones. Como el Modelo Log-Normal se basa en una ecuación diferencial estocástica de tipo Itô, su resolución pasa por el manejo de integrales estocásticas de tipo Itô y de sus propiedades estadísticas, el capítulo se cierra estudiando estos tópicos.

3.1 | El Movimiento Browniano o Movimiento de Wiener

Un proceso estocástico o función aleatoria describe la evolución temporal de una variable aleatoria. En esta memoria, haremos uso de uno de los procesos estocásticos más importantes en Estadística, y que se denomina el Movimiento Browniano. Este proceso toma valores continuos y depende de la variable tiempo, la cual también se considera continua. El Movimiento Browniano resulta adecuado para describir el comportamiento de variables económico-financieras, como es el caso de los activos subyacentes.

El Movimiento Browniano es un proceso estocástico de tipo *gaussiano*. Su introducción se realizó de forma intuitiva en 1827 por el botánico escocés Robert Brown quien lo utilizó para describir el movimiento aleatorio de las partículas de polen en el agua debido a la interactuación molecular. A este fenómeno se le denominó “Movimiento Browniano”.

En el siglo XX, se descubrió la utilidad de dicho instrumento matemático en múltiples campos; en particular en el campo de las finanzas, en el que se utilizó dicho instrumento para la modelización del comportamiento de los precios bursátiles. Louis Bachelier (1900)

hizo uso de las mismas en su tesis doctoral “La Teoría de la Especulación” para modelizar ciertos activos financieros. No obstante, el trabajo de L. Bachelier no fue comprendido en su época y durante mucho tiempo permaneció ignorado. Con posterioridad, fue Norbert Wiener quien consiguió formalizar matemáticamente el concepto de Movimiento Browniano y de ahí que, en ocasiones, se le denomine también proceso de *Wiener.* A lo largo de esta memoria el Movimiento Browniano o proceso de Wiener se denotará por:

$$\{B(t;\omega): t \geq 0, \omega \in \Omega\} \quad \text{ó} \quad \{B(t): t \geq 0\}$$

aunque en muchos textos se utiliza indistintamente la notación:

$$\{W(t): t \geq 0\}.$$

Como ya se ha indicado, el objeto de esta memoria es el estudio y aplicación de un modelo estocástico de activos subyacentes denominado Modelo Log-Normal, cuya solución es un proceso estocástico denominado Movimiento Browniano Geométrico (MBG). El ingrediente que dota de aleatoriedad al Movimiento Browniano Geométrico es el Movimiento Browniano cuyas trayectorias muéstrales son muy irregulares. A continuación damos la definición de este importante proceso estocástico, la cual se da no a través de una fórmula matemática, sino enunciando una serie de propiedades estadísticas que lo caracterizan.

El Movimiento Browniano:

$$\{B(t): t \geq 0\} \quad \text{ó} \quad W \equiv \{W(t), t \in R^{+}\}$$

es un proceso estocástico que cumple:

- **MB.1.** Comienza en el origen con probabilidad 1:

$$P[B(0)=0]=1.$$

- **MB.2.** Los incrementos del Browniano dados por,

$$B(t)-B(s),$$

son variables aleatorias independientes:

$$B(t_1)-B(t_0); B(t_2)-B(t_1); \ldots; B(t_{n+1})-B(t_n), \qquad 0 \leq t_1 \leq t_2 \leq \ldots \leq t_{n+1} \leq +\infty.$$

- **MB.3.** Tiene incrementos estacionarios:

$$B(t+\Delta t)-B(t)=B(s+\Delta t)-B(s), \qquad \forall\ s,t:\ 0 \le s \le t \in [\ 0, +\infty [\ ,$$

  donde la igualdad anterior es en distribución.

- **MB.4.** Los incrementos del proceso son *gaussianos* de media 0 y varianza $t-s$:

$$B(t)-B(s)\sim N(0;\sqrt{t-s}), \qquad \forall\, s,t\, 0\le s\le t \quad .$$

Considerando la propiedad MB.4., en el caso particular en que

$$s=0,$$

se deduce que

$$B(t)\sim N(0;\sqrt{t}),$$

es decir, que fijado t entonces la variable aleatoria $B(t)$ sigue una distribución normal o *gaussiana* de media 0 y desviación típica la raíz cuadrada del instante temporal.

3.2 | Propiedades estadísticas del Movimiento Browniano

A continuación, se especifican las principales propiedades estadísticas del proceso estocástico Movimiento Browniano. Para algunas de estas propiedades no daremos la demostración, la cual puede verse por ejemplo en el libro de Øksendal, Bernt K.

P.1. *Función Media*: A partir de las condición MB.4, se deduce que la función media del Movimiento Browniano es idénticamente nula:

$$\mu_{S(t)}=0, \qquad \forall\, t\ge 0.$$

En la Gráfica 3.1 se ilustra esta propiedad.

P.2. *Función Covarianza*: Mide el grado de relación lineal entre las variables aleatorias,

$$B(s) \quad \text{y} \quad B(t),$$

que se obtienen al fijar dos instantes s y t respectivamente. En efecto, veamos que se cumple

$$Cov[B(t),B(s)]=min(s,t), \qquad \forall\, s,t\geq 0.$$

En efecto, si tomamos $0\leq s\leq t$ entonces utilizando la propiedad P.1. y las propiedades básicas del operador esperanza se tiene:

$$\begin{aligned}
Cov\,[B(t),B(s)] &= E\,[B(t)B(s)] - E[B(t)]E[B(s)] \\
&= E\left[B(t)B(s) - \left(B(s)\right)^2 + (B(s))^2\right] \\
&= E\left[\left(B(t) - B(s)\right)B(s) + \left(B(s)\right)^2\right] \\
&= E\left[\left(B(t) - B(s)\right)B(s)\right] + E[(B(s))^2] \\
&= E[(B(t) - B(s))(B(s) - B(0))] + E[(B(s))^2] \\
&= E[B(t) - B(s)]\,E[B(s) - B(0)] + E[(B(s))^2] \\
&= (E[B(t)] - E\,[B(s)])(E[B(s)]) - E[B(0)] + E[(B(s))^2] \\
&= E[(B(s))^2] \\
&= Var[B(s)] \\
&= s.
\end{aligned}$$

Obsérvese que si en la relación anterior tomamos $s=t$ se obtiene la propiedad MB.4, es decir, que la varianza del Movimiento Browniano es t.

P.3. $B(t)$ es $\frac{1}{2}$- autosemejante: Esta es, únicamente, una propiedad geométrica que formalmente se denota de la siguiente manera.

$$B(T\cdot t)=\sqrt{T}\cdot B(t), \qquad \forall\, T\geq 0, \forall\, t\geq 0\,.$$

P.4. Las trayectorias muestrales de $B(t)$ son continuas, pero no son diferenciables en ningún punto. Se puede demostrar que $B(t)$ tiene trayectorias que no son de variación acotada, lo que significa que no son derivables (las trayectorias del Movimiento Browniano tienen puntos angulosos, es decir, con pico para todo instante). Este comportamiento muestral se observa en la Gráfica 3.1.

3.3 | Simulación del Movimiento Browniano

Como se verá posteriormente cuando se estudie y resuelva el Modelo Log-Normal para modelizar los activos subyacentes, la solución quedará expresada en términos del Movimiento Browniano. Para poder realizar predicciones de subyacentes será necesario poder realizar simulaciones del Movimiento Browniano. A continuación, se indica una forma de simular dicho proceso, aunque cabe señalar que el software especializado disponible tiene implementados otros algoritmos para realizar dicha simulación.

Una manera sencilla de simular el Movimiento Browniano es a través de variables aleatorias normales tipificadas, haciendo uso para ello de la identidad:

$$B(T)=\sqrt{T}\cdot B(t),\qquad \forall\, t\geq 0, \forall\, T\geq 0. \qquad (3.3.1)$$

Para justificar esta identidad en distribución es suficiente con probar que ambos miembros de la identidad tienen la misma distribución. Más específicamente, es sencillo ver que dicha distribución es *gaussiana* y justificar que la media y varianza de ambos miembros coinciden. Esto se muestra en la Tabla 3.1.

	$B(t)$	$\sqrt{t}\,Z$
Distribución	Gaussiana	Transformación lineal de una variable Gaussiana
Media	0	$E[\sqrt{t}\,Z]=\sqrt{t}\cdot E[Z]=0$
Varianza	t	$Var[\sqrt{t}\,Z]=(\sqrt{t})^2\cdot Var[Z]=t$

Tabla 3.1: Propiedades del movimiento browniano
Fuente: Elaboración propia

En el Gráfico 3.1 se muestra una simulación de Movimiento Browniano sobre la ventana temporal $[0,1]$. En esta gráfica se ilustra la propiedad MB.1 y la propiedad estadística P.4 introducidas anteriormente. No obstante, cabe puntualizar que, pese a que el Movimiento Browniano es el ingrediente que introduce la componente de aleatoriedad en el Modelo de Log-Normal, éste no se corresponde directamente con el Movimiento Browniano, sino con

su diferencial. El diferencial de éste, $\{dB(t), \forall t \geq 0\}$, genera un nuevo proceso estocástico, también de tipo *gaussiano* y denominado Ruido Blanco.

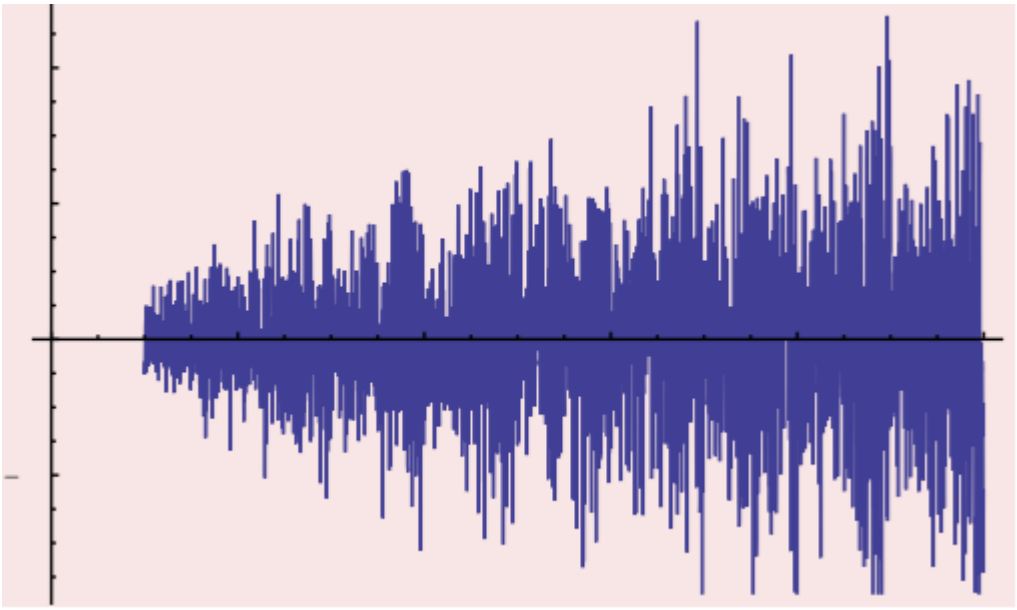

Gráfico 3.1: Ruido del teorema central del límite
Fuente: Elaboración propia

En base al Teorema Central del Límite, la diferencial del Movimiento Browniano se empleará para modelizar la aleatoriedad o "ruido" en el modelo. Son numerosos y diferentes los factores que en el mundo real pueden llegar a determinar el valor de un activo financiero. En condiciones generales, este teorema indica que, si S_n es la suma de n variables aleatorias independientes, la función de distribución de S_n se aproxima a una distribución normal (o *gaussiana*). En definitiva, modelizando el "ruido" con el Teorema Central del Límite, éste sigue una distribución *gaussiana* o normal, lo que otorga consistencia a la decisión de modelizar la aleatoriedad mediante un proceso *gaussiano*, como lo son el Movimiento Browniano y su diferencial, el Ruido Blanco.

3.4 | El Cálculo de Itô

Kiyoshi Itô fue un matemático japonés que desarrolló una teoría para la diferenciación e integración de procesos estocásticos. Esta teoría se conoce como el Cálculo de Itô. El

concepto básico de este cálculo es la Integral de Itô y el más importante de sus resultados, es el Lema de Itô. Cabe puntualizar que la Integral de Itô es el corazón del análisis estocástico, facilita la comprensión matemática de sucesos aleatorios y difiere de la teoría matemática clásica de integración y diferenciación.

La teoría y fórmula de K. Itô tiene numerosas aplicaciones de interés, no obstante, en este apartado se detalla la versión del cálculo de Itô que se adapta al objeto de la presente memoria. Se centra en el cálculo exacto de la solución de una ecuación diferencial estocástica definida de la siguiente forma:

$$dX(t)=f(t,X(t))dt+g(t,X(t))dB(t). \quad (3.4.1)$$

Dicha solución se obtiene, en algunos casos que dependen de la forma específica de los coeficientes $f(t,X(t))$ y $g(t,X(t))$, utilizando el Lema de Itô. El Lema de Itô es una versión estocástica de la regla de la cadena para procesos estocásticos $X(t)$. Para el Modelo Log-Normal, que es el caso que ocupa el presente trabajo, se verá como ello sí es factible. A continuación enunciamos el Lema de Itô:

Lema de Itô (versión integral)

Hipótesis: Sea $X(t)$ un proceso estocástico que satisface la siguiente ecuación diferencial tipo Itô con condición inicial determinista x_0:

$$\begin{aligned} dX(t) &= f\left(t,x(t)\right)dt + g\left(t,x(t)\right)dB(t), \quad t \geq 0, \\ X(0) &= x_0, \end{aligned}$$

y sea $F(t,x)$ una función $F\colon [0,T]\times\mathbb{R} \longrightarrow \mathbb{R}$ tal que las siguientes derivadas parciales existen y son continuas:

$$\frac{\partial F(t,x)}{\partial t} = F_1(t,x), \qquad \frac{\partial F(t,x)}{\partial x} = F_2(t,x), \qquad \frac{\partial F(t,x)}{\partial x^2} = F_{22}(t,x).$$

Tesis: Entonces para $t > 0$ se cumple

$$\begin{aligned} F\left(t,x(t)\right) - F\left(s,x_0\right) &= \int_0^t \{F_1\left(r,x(r)\right) + f\left(r,x(r)\right)F_2\left(r,x(r)\right)\}dr \\ &+ \int_0^t \frac{1}{2}\left(g(r,(x(r))^2 F_{22}\left(r,x(r)\right)dr\right. \\ &+ \int_0^t g\left(r,x(r)\right)F_2\left(r,x(r)\right)dB(r). \end{aligned}$$

3.4.1 | La Integral de Itô

En esta sección introducimos el concepto de Integral de Itô de un proceso estocástico. La definición de la Integral de Itô difiere de la teoría matemática clásica de integración y diferenciación, definiendo lo que debería entenderse por integración de un proceso estocástico con respecto al Movimiento Browniano (u otro proceso estocástico). El objetivo de esta sección, por tanto, es dar una interpretación a la siguiente expresión:

$$\int_0^t X(s)dB(s), \qquad (3.4.2)$$

donde $X(s)$ es un proceso estocástico que cumple determinadas condiciones que a continuación se detallarán. Se dice que la expresión (3.4.2) es la Integral de Itô con respecto al Movimiento Browniano. Primero daremos la interpretación de dicha integral si el integrando y el integrador $X(s)$ y $B(s)$ no fueran procesos estocásticos, sino funciones deterministas.

Supongamos que $f(s)$ y $g(s)$ son dos funciones deterministas suaves respecto de la variable tiempo s y consideremos la siguiente integral:

$$\int_0^t g(s)df(s). \qquad (3.4.3)$$

Cuando $f(s)$ tiende a una función diferenciable, se escribe $\frac{df(s)}{ds}=f'(s)$, o equivalentemente, $df(s)=f'(s)ds$. Sustituyendo esto en la integral (3.4.3), se llega a:

$$\int_0^t g(s)df(s)=\int_0^t g(s)f'(s)ds,$$

quedando ésta definida como una integral estándar (de tipo Riemann). Si $f(s)$ no fuese diferenciable, todavía se puede utilizar la teoría determinista para definir la integral anterior. En efecto, cuando $f(s)$ no es demasiado irregular como función del argumento es decir, cuando $f(s)$ es lo que se denomina una función de variación acotada, se puede probar que la integral está bien definida como el siguiente límite:

$$\int_0^t g(s)df(s)=\lim_{n\to\infty}\sum_{i=1}^{n-1} g(s_i)(f(s_{i+1})-f(s_i)).$$

Puesto que $f(s)$ es de variación acotada, $f(s_{i+1})$ está próximo a $f(s_i)$. A partir de esto es posible probar que el límite anterior existe siempre que $g(s)$ no varíe demasiado. Por supuesto, si la función $g(s)$ es extremadamente fluctuante en diferentes puntos en el tiempo, el límite puede ser divergente.

Se define la integral (3.4.2) de forma análoga mediante el siguiente límite:

$$\int_0^t X(s,\omega)dB(s,\omega)=\lim_{n\to\infty} X(s_i,\omega)(B(s_{i+1},\omega)-B(s_i,\omega)), \quad \omega\in\Omega. \quad (3.4.4)$$

Nótese que se toma el límite para cada ω fijo en el espacio muestral Ω de la variable aleatoria $B(s)$. El problema aquí es que el límite para cada ω en general no existe (empieza en $\pm\infty$) para muchos procesos estocásticos $X(s)$. Para cada ω, la función

denotada como $s \to B(s,\omega)$ es extremadamente volátil. Como se indicaba con anterioridad (véase la propiedad estadística P.4), el Movimiento Browniano es un ejemplo de un proceso estocástico con trayectorias muestrales continuas, pero no diferenciables en ningún punto. Todavía peor, el Movimiento Browniano como función del tiempo no es de variación acotada para cada ω, tal y como se requiere para $f(s)$. Para compensar la irregularidad de las trayectorias del Movimiento Browniano, se tienen que exigir dos condiciones, que detallaremos después, en el proceso integrador $X(s)$. Bajo estas condiciones el límite existirá a pesar de la irregularidad de las trayectorias del Movimiento Browniano. La primera condición es que se asume que $X(s)$ es independiente de los incrementos del Movimiento Browniano. La segunda condición guarda relación con la variación del integrando (similar a la condición de que $g(s)$ en (3.4.3) no debe variar demasiado).

A partir de la propiedad MB.2 del Movimiento Browniano se sabe que la variación de un incremento del mismo, está dada por:

$$E[(B(s_{i+1})-B(s_i))^2]=s_{i+1}-s_i.$$

Si $X(s_i)$ es independiente del incremento $B(s_{i+1})-B(s_i)$, se obtiene

$$E[(X(s_i)(B(s_{i+1})-B(s_i)))^2]=E[(X(s_i)^2)]E[(B(s_{i+1})-B(s_i))^2]$$

$$=E[(X(s_i)^2)](s_{i+1}-s_i).$$

Considerando el segundo momento de la variable aleatoria

$$\sum_{i=1}^{n-1} X(s_i)(B(s_{i+1})-B(s_i)), \qquad (3.4.5)$$

y asumiendo que $X(s_i)$ es independiente de $B(s_{i+1})-B(s_i)$ para cada $i=1,..,n$, puede verse por la independencia de los incrementos del Movimiento Browniano que:

$$E[(\sum_{i=1}^{n-1} X(s_i)(B(s_{i+1})-B(s_i)))^2]=\sum_{i=1}^{n-1} E[(X(s_i)^2)](s_{i+1}-s_i).$$

La suma del miembro derecho es una aproximación de la integral $\int_0^t E[(X(s))^2]ds$. Por tanto, si esta integral existe, se deduce:

$$\lim_{n\to\infty} E[(\sum_{i=1}^{n-1} X(s_i)(B(s_{i+1})-B(s_i)))^2]=\int_0^t E[(X(s))^2]ds,$$

lo cual conduce a la conclusión de que la varianza de la suma en (3.4.5) converge a

$\int_0^t E[(X(s))^2]ds$. Asumiendo que esta integral existe, se demuestra:

$$E[(\lim_{n\to\infty} \sum_{i=1}^{n-1} X(s_i)(B(s_{i+1})-B(s_i)))^2]=\int_0^t E[(X(s))^2]ds. \quad (3.4.6)$$

Cabe subrayar que la integral que aparece en el lado derecho de la relación (3.4.6), en algunos casos, puede no existir. Existirá siempre que sea finita, es decir, siempre que el proceso estocástico $X(s)$ sea tal que su segundo momento estadístico puede ser integrado de 0 a t.

Por ejemplo, para el proceso $X(s)=s^{-1}B(s)$, por la condición MB.2 del Movimiento Browniano, se tiene

$$\int_0^t E[(X(s))^2]ds=\int_0^t s^{-1}ds=\ln(t)-\ln(0)=+\infty.$$

Por otro lado, si tomamos $X(s)=B(s)$, fácilmente se puede reconocer que $X(s)$ satisface la condición de integrabilidad. Afortunadamente, esto también se cumple para una larga clase de procesos estocásticos.

Retomando la relación (3.4.6), y considerando que el término $X(s_i)$, que aparece en el miembro izquierdo, debe ser independiente de incrementos $B(s_{i+1})-B(s_i)$ para todos los valores de $i=1,\ldots,n-1$. Esto motiva la introducción de la denominada adaptabilidad del proceso integrador:

Definición 1. *Una variable aleatoria* X *es llamada* F_s *-adaptada si* X *puede ser escrita como (limite de una sucesión de) funciones de* $B(\tau)$ *para uno o más* $\tau \leq s$, *pero no como función de cualquier* $B(u)$ *con* $u>s$. *Un proceso estocástico* $X(s)$ *se dice que es adaptado si se cumple que para cada instante temporal* $s\in[0,t]$, *la variable aleatoria* $X(s)$ *es* F_s*- adaptada.*

Puntualizamos algunos aspectos de la Definición 1. En primer lugar, cabe señalar algunos procesos que se derivan de composiciones simples del Movimiento Browniano tales como el denotado como $X(s)=f(s,B(s))$, los cuales son adaptados; mientras que procesos tales como $X(s)=B(s+1)$, no lo son.

La integral $X(s)=\int_0^s B(\tau)d\tau$, también define un proceso estocástico adaptado, puesto que la integral es el límite de sumas del Movimiento Browniano en diferentes tiempos menores que s. En definitiva, por la definición de la integral de Itô, se tiene

$$\int_0^t X(s,\omega)dB(s,\omega)=\lim_{n\to\infty}\sum_{i=1}^{n-1} X(s_i,\omega)\big(B(s_{i+1},\omega)-B(s_i,\omega)\big).$$

Por tanto, focalizando sobre la Integral de Itô, obsérvese que siempre que el proceso integrando $X(s)$ sea adaptado, la integral

$$\int_0^t X(s)dB(s)$$

tiene sentido como el límite puntual en (3.4.4). Además, este límite (3.4.4) puede probarse que converge en media cuadrática y por lo tanto, también para cada $\omega\in\Omega$. Se concluye la exposición con la definición rigurosa de la Integral de Itô.

Definición 2. *Un proceso estocástico* $X(s)$, *es integrable en el intervalo* $[0,t]$ en el sentido de Itô si

1. $X(s)$ *es adaptado para* $s\in[0,t]$, *y*
2. $\int_0^t E[(X(s))^2]ds<\infty$.

La Integral de Itô se define como la variable aleatoria

$$\int_0^t X(s,\omega)dB(s,\omega)=\lim_{n\to\infty}\sum_{i=1}^{n-1}X(s_i,\omega)(B(s_{i+1},\omega)-B(s_i,\omega)), \qquad (3.4.7)$$

donde el límite es considerado por cada $\omega\in\Omega$.

3.4.2 Propiedades de la Integral de Itô

Introducido el concepto de Integral de Itô, a continuación enunciaremos las propiedades estadísticas de dicha integral de Itô, algunas de las cuales serán utilizadas en el próximo capitulo. Estas propiedades se citan distinguiendo si el integrando es una función determinista o un proceso estocástico $X(t)$.

Sea $h(t)$ una función determinista tal que $\int_0^t (h(s))^2 ds$, entonces se cumplen las siguientes propiedades:

I.1. Media: $E[\int_0^t h(s)dB(s)]=0.$

I.2. Varianza: $Var[\int_0^t h(s)dB(s)]=\int_0^t (h(s))^2 ds.$

I.3. Covarianza: $Cov[\int_0^t h_1(\tau),\int_0^s h_2(\tau)dB(\tau)]=\int_0^{t\wedge s} h_1(\tau)h_2(\tau)d\tau$ para $t\wedge s=min(t,s).$

En particular,

$$E[(\int_0^t h_1(\tau)dB(\tau))(\int_0^s h_2(\tau)dB(\tau))]=\int_0^{t\wedge s} h_1(\tau)h_2(\tau)d\tau.$$

I.4. Normalidad: $\int_0^t h(s)dB(s)\sim N(0;\int_0^t (h(s))^2 ds).$

Si el integrando es un proceso estocástico $X(t)$ que satisface las condiciones de la Definición 2 anterior, entonces se verifican las siguientes propiedades:

I.5. Media: $E[\int_0^t h(s)dB(s)]=0.$

I.6. Varianza: . $V[\int_0^t X(s)dB(s)]=\int_0^t E[(X(s))^2 ds].$

Obsérvese que a partir de las propiedades I.5 e I.6 se deduce

$$E[(\int_0^t X(s)dB(s))^2]=\int_0^t E[(X(s))^2]ds,$$

llamada isometría de Itô.

3.5 | Modelo estocástico Log-Normal

El objetivo de este apartado es presentar el Modelo Log-Normal que constituye el fundamento matemático sobre el cual se realizarán las predicciones del activo subyacente IAG (IAG.MC). Este modelo se basa en la siguiente ecuación diferencial estocástica tipo Itô

$$dS(t)=\mu S(t)dt+\sigma S(t)dB(t),\quad S(0)=s_0, \qquad (3.5.1)$$

donde:

- $S(t)$ es el valor del subyacente en el instante t.
- s_0 representa el valor del subyacente en el instante inicial $t=0$ y se supone conocido, por ello se denota en minúsculas.
- $\mu \in R$ es un parámetro del modelo denominado *drift* o tendencia.
- $\sigma>0$ es un parámetro del modelo denominado volatilidad local.
- $B(t)$ es un proceso estocástico de Wiener o Movimiento Browniano.

En lo que sigue de capítulo, en primer lugar motivaremos la consideración del modelo (3.5.1) a partir de los modelos clásicos (deterministas) de subyacentes y posteriormente, utilizando el Cálculo de Itô presentado anteriormente, obtendremos la solución del modelo Log-Normal y sus principales propiedades estadísticas tales como las funciones media y varianza.

Posteriormente, se detallará cómo se pueden estimar los parámetros μ y σ utilizando técnicas estadísticas, de modo que dispongamos de resultados que nos permitan realizar estimaciones (predicciones) puntuales y por intervalos de confianza de un activo subyacente cuando se lleva el modelo a la práctica.

3.5.1 | Motivación de un modelo estocástico para un subyacente en ambiente de certidumbre

En este apartado vamos a presentar un modelo determinístico clásico para representar la trayectoria temporal de una inversión en ambiente de certidumbre, es decir, sin riesgo. Este modelo constituye el punto de partida para motivar, en el siguiente apartado, un modelo estocástico para subyacentes cotizados que se denomina en la literatura financiera Modelo Log-Normal.

Sea S_0 un principal que se invierte (capitaliza) a un interés μ compuesto continuo durante un intervalo temporal $[0,T]$ dividido en K periodos de longitud $\Delta t>0$ cada uno (véase, Figura 3.1).

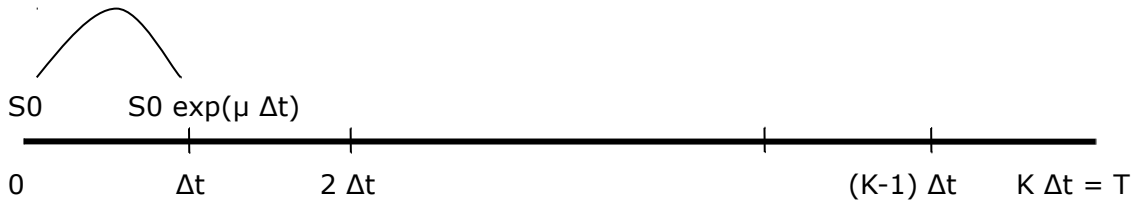

Figura 3.1. Capitalización a interés compuesto continuo.
Fuente: Elaboración propia.

Denotemos por $\hat{S}^{(J)}$, $j=0,1,\ldots,K$, el capital al cabo de j períodos (por convenio de notación, implícitamente se supone que $\hat{S}^{(0)}=S_0$, entonces sabemos que

$$\hat{S}^{(1)}=S_0 e^{\mu\Delta t}, \qquad \hat{S}^{(2)}=S^{(1)}e^{\mu\Delta t} \quad , \qquad (3.5.2)$$

y en general,

$$\hat{S}^{(j)}=\hat{S}^{(j-1)}e^{\mu\Delta t}, \qquad j=0,1,\ldots,K, \qquad (3.5.3)$$

y por tanto, razonando recursivamente se tiene

$$\hat{S}^{(j)}=\hat{S}^{(j-1)}e^{\mu\Delta t}=\hat{S}^{(j-2)}e^{2\mu\Delta t}=\ldots=S_0e^{\mu j\Delta t},\qquad j=0,1,\ldots,K. \qquad (3.5.4)$$

Obsérvese que cuando $j=K$, se cumple que: $\hat{S}(T)=\hat{S}^{(K)}=S_0e^{\mu K\Delta t}=S_0e^{\mu T}$, donde se ha utilizado que $K\Delta t=T$ y se ha introducido la notación siguiente: $\hat{S}(T)$ representa el capital al cabo de K períodos. Es decir, el efecto de K pasos de longitud Δt equivale a un paso de tamaño $T=K\Delta t$. En realidad, este modelo de capitalización resulta de resolver el problema de valor inicial (ecuación diferencial ordinaria junto a una condición inicial) siguiente:

$$\frac{d\hat{S}(t)}{dt}=\mu\hat{S}(t),\ \ \hat{S}(0)=S_0, \qquad (3.5.5)$$

cuya solución es

$$\hat{S}(t)=S_0e^{\mu t}, \qquad (3.5.6)$$

la cual, para $t=T$ captura la solución del problema anterior: $\tilde{S}(T)=S_0e^{\mu T}$. Es importante observar que el modelo (3.5.1), el modelo estocástico de partida, contiene al modelo clásico determinista de capitalización continua, pues basta hacer $\sigma=0$ en (3.5.1) para obtener a partir de la ecuación diferencial estocástica la ecuación diferencial ordinaria (3.5.5).

3.5.2 | Motivación del modelo estocástico para un subyacente cotizado: El Modelo Log-Normal

El modelo presentado anteriormente es un modelo para un subyacente en ambiente de certidumbre en el cual no se contempla ninguna aleatoriedad. En la práctica, el valor de un activo subyacente depende un gran número de factores económicos, sociales, etc. que influyen en los mercados donde la acción cotiza. Estos factores pueden contener una elevada incertidumbre dada la complejidad de su determinación.

Considerando el grado de incertidumbre existente, es más conveniente y adecuado introducir aleatoriedad en el modelo determinista para un subyacente. El parámetro μ pasa a ser una cantidad aleatoria en lugar de una cantidad determinística. Haciendo uso de

las herramientas estocásticas presentadas, la aleatoriedad en el modelo se introducirá vía el Movimiento Browniano $B(t)$, más concretamente, con su derivada o diferencial, el denominado proceso de Ruido Blanco $dB(t)$. Todo ello motiva la siguiente expresión estocástica del parámetro μ:

$$\mu \to \mu + \sigma B'(t), \quad \sigma > 0. \qquad (3.5.7)$$

En la expresión (3.5.7) se sigue denotando al parámetro μ como el rendimiento medio del subyacente, pero en este caso, se le ha añadido fluctuaciones aleatorias, siendo además $B'(t)$ el proceso estocástico Ruido Blanco y $\sigma > 0$ su intensidad.

Introduciendo en el modelo la notación diferencial de la derivada

$$S'(t) = \frac{dS(t)}{dt},$$

y considerando la aleatoriedad en el parámetro μ, el modelo se reescribe de la siguiente forma:

$$\frac{dS(t)}{dt} = (\mu + \sigma B'(t) S(t)) \Rightarrow dS(t) = (\mu + \sigma B'(t)) S(t) dt.$$

Agrupando los diferentes parámetros se obtiene:

$$dS(t) = \mu S(t) dt + \sigma S(t) B'(t) dt, \qquad (3.5.8)$$

donde se puede diferenciar claramente una parte determinista, $\mu S(t)$, y otra parte estocástica, $\sigma S(t) B'(t) dt$, en la cual se recoge el Ruido Blanco. Formalmente, dado que $dB(t) = B'(t) dt$, nuevamente el modelo se puede reescribir como en (3.5.1), conocido en literatura financiera, como el modelo de activos subyacentes Modelo Log-Normal.

3.5.3 | Solución del Modelo Log-Normal

En este apartado vamos a resolver la ecuación diferencial estocástica (3.5.1). Para ello utilizaremos dos enfoques. El primero de ellos, evita el uso del Cálculo de Itô y resulta más artificial en su presentación por la forma en la que se introduce la aleatoriedad. Sin embargo,

este enfoque no requiere conocer herramientas complejas como son la integración y diferenciación de procesos estocásticos. El segundo enfoque, más técnico, hace uso del Cálculo de Itô, y en particular del Lema de Itô presentado en el presente capítulo.

3.5.4 | Solución del Modelo Log-Normal sin el Cálculo de Itô

El valor de una acción está gobernado por multitud de factores y variables que deben considerarse inciertos y no deterministas, por ello (3.5.5) no es un buen modelo, si bien, como se ha motivado en el Apartado 3.5.2, el modelo que aspire a ser adecuado debe recoger en su formulación (y el modelo (3.5.1), así lo hace, como se verá después) la parte del valor de la evolución del precio de la acción que se comporta de forma determinista como cualquier otra inversión libre de riesgo, y la parte, incierta o aleatoria, que hace que el valor de ese activo financiero no sea predecible de forma determinista.

Partamos de la expresión (3.5.3) con $j=1,$ introduzcamos la aleatoriedad como sigue:

$$\widetilde{S}^{(1)}=\widetilde{S}^{(0)}e^{\mu\Delta t}e^{cZ_1}, \qquad (3.5.9)$$

siendo c una constante libre y no nula (que más tarde fijaremos) y Z_1 una variable aleatoria normal o gaussiana tipificada, i.e., $Z_1\sim N(0;1),$ y en general, introduciendo la aleatoriedad en (3.5.3) del mismo modo se tiene

$$\widetilde{S}^{(j)}=\widetilde{S}^{(j-1)}e^{\mu\Delta t}e^{cZ_j}, \qquad (3.5.10)$$

siendo $Z_j\sim N(0;1).$ Entonces, si hacemos esto para cada incremento, $j=1,2,\ldots,K$ de modo que, $Z_1,\ldots,Z_j,\ldots,Z_k$ sean variables aleatorias $N(0;1)$ e independientes, desde (3.5.10) con $j=K,$ obtenemos por recursividad que

$$\widetilde{S}^{(K)}=S_0e^{\mu K\Delta t}e^{c(Z_1+\ldots+Z_K)}, \qquad (3.5.11)$$

o equivalentemente,

$$\widetilde{S}(T)=S_0e^{\mu T}e^{c(Z_1+\ldots+Z_K)}.$$

La aleatoriedad introducida así en (3.5.9) (o más generalmente en (3.5.10)), parece coherente porque garantiza que $\widetilde{S}^{(1)}>0$ $(\widetilde{S}^{(j)}>0)$; pero no es totalmente satisfactoria, ya que, es de esperar que cuando no haya incertidumbre, $\widetilde{S}^{(1)}$ en (3.5.9) se comporte como en (3.5.4) con $j=1$ al menos en media. Sin embargo, se sabe:

$$E[e^{cZ}]=e^{\frac{c^2}{2}},\ \ Z\sim N(0;1), \qquad (3.5.12)$$

y por tanto

$$E[\widetilde{S}^{(1)}]=E[S_0e^{\mu\Delta t}e^{cZ_1}]=S_0e^{\mu\Delta t}E[e^{cZ_1}]=S_0e^{\mu\Delta t}e^{\frac{c^2}{2}}\neq S_0e^{\mu\Delta t}=\widetilde{S}^{(1)}.$$

Para lograr el objetivo marcado, vamos a introducir la aleatoriedad como sigue

$$S^{(1)}=S_0e^{\mu\Delta t}e^{cZ_1-\frac{c^2}{2}}, \qquad (3.5.13)$$

ya que, claramente de (3.5.13) se obtiene que

$$E[S^{(1)}]=S_0e^{\mu\Delta t},$$

y en general en (3.5.3) introduciremos la aleatoriedad del mismo modo:

$$S^{(j)}=S^{(j-1)}e^{\mu\Delta t}e^{cZ_j-\frac{c^2}{2}},\ \ j=1,2,\ldots,K. \qquad (3.5.14)$$

Razonando recursivamente como en (3.5.14) obtenemos

$$S(T)=S^{(j-1)}e^{\mu\Delta t}e^{c(Z_1+\ldots+Z_K)}e^{-K\frac{c^2}{2}}. \qquad (3.5.15)$$

Denotemos por

$$B_K=Z_1+\ldots+Z_K, \qquad (3.5.16)$$

entonces dado que $Z_j\sim N(0;1)$, $j=1,2,\ldots,K,$ son variables aleatorias independientes se tiene $B_K\sim N(0;1)$ y como $K\Delta t=T$ y $S^{(K)}=S(T),$ (3.5.15) equivale a

$$S(T)=S_0 e^{\mu T} e^{c B_K - K\frac{c^2}{2}}, \qquad (3.5.17)$$

donde S_0 es el precio o subyacente de la acción en el instante inicial; $e^{\mu T}$ es la componente determinista del valor de la acción (ligado al valor μ del tipo de interés); y donde B_K representa la introducción de la aleatoriedad en el modelo y $e^{-K\frac{c^2}{2}}$ es un término (determinista) de corrección.

La modelización dada (3.5.17) tiene una gran ventaja frente a otro tipo de modelos como los basados en árboles binomiales, y es que permite que el valor de la acción no sólo tome dos valores posibles, sino cualquier valor positivo. Sin embargo, sigue conteniendo un defecto importante respecto de nuestros intereses; si fijamos T y vamos considerando subintervalos más pequeños (particiones más finas del intervalo $[0,T]$), es decir, hacemos $K \to +\infty$ ó equivalentemente $\Delta t \to 0$, con T fijo, para la varianza del término que introduce la aleatoriedad en el modelo (3.5.17) se obtiene por (3.5.16) que

$$Var[c\,B_K]=c^2\,Var[B_K]=c^2 K \to +\infty, \qquad (3.5.18)$$

esto es, la varianza de este término que forma parte del modelo de la acción aumentará cuantas más discretizaciones tomemos del intervalo objeto de análisis, con independencia del valor T que define su extremo superior, el cual está fijo. Esto carece de sentido desde el punto de vista financiero, ya que fijado un intervalo durante el cual se observa una acción, la volatilidad de la misma es un valor fijo que no debería cambiar aunque en lugar de observar la acción intra-semana lo hiciéramos intra-día. Para resolver este inconveniente, recordemos que la constante c introducida en (3.5.14) es libre y por tanto está a nuestro servicio. Tomemos entonces c de modo que

$$c^2 K=\sigma^2 T, \qquad (3.5.19)$$

siendo σ^2 un parámetro fijo identificativo del modelo particular que manejemos (y que en la práctica se calculará a partir de los datos reales observados). De esta forma,

$$Var[c\,W_K]=\sigma^2 T<+\infty. \qquad (3.5.20)$$

Obsérvese que esto nos indica algo coherente que se aprecia en la práctica: a mayor longitud T del intervalo $[0,T]$, tendremos mayor volatilidad, es decir, variabilidad de la acción.

Vamos ahora a relacionar B_K dada en (3.5.16) con el Movimiento Browniano. Para ello, obsérvese que por la propiedad P.3 del Movimiento Browniano sabemos que:

$$B(T)=\sqrt{\Delta t}\,B_K. \qquad (3.5.21)$$

Entonces despejando de (3.5.19) se tiene:

$$c=\sigma\frac{\sqrt{T}}{K}\rightarrow c=\sigma\sqrt{\Delta t}, \qquad (3.5.22)$$

y entonces por (3.3.21) se tiene

$$c\,B_K=\sigma\sqrt{\Delta t}\,B_K=\sigma\,B(T) \qquad (3.5.23)$$

Finalmente, sustituyendo (3.5.19) y (3.5.23) en (3.5.17) llegamos a:

$$S(T)=s_0 e^{(\mu-\frac{\sigma^2}{2})T}e^{\sigma B(T)}. \qquad (3.5.24)$$

Por lo que desde una motivación basada en la introducción de la aleatoriedad en el modelo determinista, hemos obtenido la solución del Modelo Log-Normal.

3.5.5 | Solución del Modelo Log-Normal mediante el Cálculo de Itô

En este apartado resolveremos la ecuación diferencial estocástica de tipo Itô (3.5.1) con condición inicial:

$$dS(t)=\mu S(t)\,dt+\sigma S(t)\,dB(t), \quad S(0)=s_0,$$

aplicando el Lema de Itô introducido en el Apartado 3.4. Para ello, escribimos (3.5.1) en forma integral:

$$\int_0^t dS(r) = \int_0^t \mu S(r)\, dr + \int_0^t \sigma S(r)\, dB(r),$$

o equivalentemente:

$$S(t) - S(0) = \int_0^t \mu S(r)\, dr + \int_0^t \sigma S(r)\, dB(r). \qquad (3.5.25)$$

A continuación, aplicamos el Lema de Itô con la siguiente identificación:

$$X(t) = S(t),\ \ f(t, x(t)) = f(t, S(t)) = \mu S(t),\ \ g(t, x(t)) = g(t, S(t)) = \sigma S(t),$$

eligiendo

$$F(t, x) = \ln(x).$$

Para la aplicación del Lema de Itô se necesitan calcular las siguientes derivadas parciales:

$$F_1(t, x) = \frac{\partial F(t, x)}{\partial t} = 0,\ \ F_2(t, x) = \frac{\partial F(t, x)}{\partial x} = \frac{1}{x},\ \ F_{22}(t, x) = \frac{\partial^2 F(t, x)}{\partial x^2} = \frac{-1}{x^2}.$$

Por tanto,

$$\ln(S(t)) - \ln(s_0) = \int_0^t \left(\mu S(r)\frac{1}{S(r)} + \frac{1}{2}(\sigma S(r))^2\left(\frac{-1}{(S(r))^2}\right)\right) dr + \int_0^t \sigma S(r)\frac{1}{S(r)}\, dB(r).$$

Simplificando tenemos:

$$\ln\left(\frac{S(t)}{s_0}\right) = \int_0^t \left(\mu - \frac{1}{2}\sigma^2\right) dr + \int_0^t \sigma dB(r).$$

$$\ln\left(\frac{S(t)}{s_0}\right) = \left(\mu - \frac{1}{2}\sigma^2\right) t + \sigma(B(t) - B(0)).$$

Observemos que por la definición del Movimiento Browniano, $B(0) = 0$ con probabilidad 1, por tanto obtenemos:

$$\ln\left(\frac{S(t)}{s_0}\right)=\left(\mu-\frac{1}{2}\sigma^2\right)t+\sigma B(t),$$

$$\frac{S(t)}{s_0}=e^{\left(\mu-\frac{1}{2}\sigma^2\right)t+\sigma B(t)},$$

$$S(t)=s_0 e^{\left(\mu-\frac{1}{2}\sigma^2\right)t+\sigma B(t)}, \qquad (3.5.26)$$

que representa el proceso estocástico solución, llamado Movimiento Browniano Geométrico del Modelo Log-Normal**.**

3.5.6 Propiedades estadísticas del proceso estocástico solución del Modelo Log-Normal

Desde el punto de vista práctico, la solución (3.5.26) hallada en el apartado anterior se utiliza para realizar predicciones probabilísticas, en un tiempo digamos $t=T$, de tipo puntual del subyacente a partir de la función media y, se completan mediante intervalos de confianza. Se hace por tanto necesario el cálculo de las funciones media y varianza de (3.5.26) en $t=T$.

Función media

Para el cálculo de la función media de (3.5.26), se requiere de la aplicación de la siguiente relación, denominada propiedad de $\frac{1}{2}$- autosemejante del Movimiento Browniano:

$$B(T\cdot t)=\sqrt{T}\cdot B(t), \quad t\geq 0, \quad T\geq 0,$$

aplicada al caso particular $t=1$, es decir,

$$B(T)=\sqrt{T}\cdot B(1), \qquad (3.5.27)$$

donde recordamos que por definición el Movimiento Browniano:

$$B(1)\sim N(0;1).$$

Por otra parte, también se requiere de la siguiente propiedad:

$$E[e^{\lambda Z}]=e^{\frac{\lambda^2}{2}}, \qquad Z\sim N(0;1), \qquad (3.5.28)$$

que se deduce del cálculo de la siguiente integral:

$$E(e^{\lambda Z})=\frac{1}{\sqrt{2}\pi}\int_{-\infty}^{+\infty} e^{\lambda z}e^{-\frac{z^2}{2}}dz$$

Tomando el operador esperanza en (3.5.26) se tiene:

$$E[S(T)]=S_0e^{(\mu-\frac{\sigma^2}{2})T}E[e^{\sigma B(T)}]=S_0e^{(\mu-\frac{\sigma^2}{2})T}E[e^{\sigma\sqrt{T}B(1)}]=S_0e^{(\mu-\frac{\sigma^2}{2})T}E[e^{\sigma B(T)}]e^{\frac{(\sigma\sqrt{T})^2}{2}}, \qquad (3.5.29)$$

por tanto,

$$E[S(T)]=S_0e^{\mu T}. \qquad (3.5.30)$$

Obsérvese que en el segundo paso de (3.5.29) se ha aplicado la propiedad (3.5.27) y en el segundo paso se ha aplicado la relación (3.5.28) con $\lambda=\sigma\sqrt{T}$ y $B(1)=Z\sim N(0;1)$.

La fórmula (3.5.30) nos indica que el modelo Log-Normal goza de la deseable propiedad de que el comportamiento medio del precio de la acción es el mismo que el valor en el caso determinista, véase (3.5.6) con $t=T$.

Función varianza

Para saber cómo evoluciona la volatilidad del precio de la acción según el Modelo Log-Normal observemos que, razonando como para el caso de la media, la varianza está dada por:

$$\begin{aligned}Var[S(T)]&=E[(S(T))^2]-(E[S(T)])^2\\&=E[(S_0)^2e^{2\sigma B(T)+2(\mu-\frac{\sigma^2}{2})T}]-(S_0)^2e^{2\mu T}\\&=(S_0)^2e^{2\mu T}(E[e^{2\sigma B(T)}]e^{-\sigma^2T}-1),\end{aligned}$$

es decir,

$$\begin{aligned}Var[S(T)]&=(S_0)^2e^{2\mu T}(E[e^{2\sigma\sqrt{T}B(1)}]e^{-\sigma^2T}-1)\\&=(S_0)^2e^{2\mu T}(e^{\frac{4\sigma^2T}{2}}e^{-\sigma^2T}-1)\\&=(S_0)^2e^{2\mu T}(e^{\sigma^2T}-1)\end{aligned}$$

la cual nos indica (como cabe esperar desde la experiencia bursátil) que la varianza crece a medida que lo hace el horizonte temporal T.

3.6 Calibración de los parámetros del Modelo Log-Normal

En este apartado explicaremos dos técnicas estadísticas diferentes para estimar los parámetros μ y σ del Modelo Log-Normal:

- Método de Máxima Verosimilitud.
- Método no paramétrico.

El motivo para utilizar dos técnicas diferentes para calibrar los parámetros está justificado porque queremos que dichas estimaciones no dependan del método utilizado, con lo cual si los dos métodos proporcionan valores iguales o similares podemos asegurar que se trata de estimaciones robustas.

3.6.1. Método de Máxima Verosimilitud

En este apartado vamos a realizar la estimación de los parámetros μ y σ del Modelo Log-Normal, usando para ello una técnica estadística llamada, Método de Máxima Verosimilitud.

Para comprender mejor cómo se aplica el Método de Máxima Verosimilitud a el contexto que nos ocupa, para mayor claridad vamos en primer lugar a recordar una serie de conceptos y definiciones que se seguirán para aplicar este método de estimación.

El método de Máxima Verosimilitud esta basado en el concepto de función de verosimilitud, debido R.A. Fisher, y es uno de los conceptos más importantes en Inferencia Estadística.

Supondremos, para fijar ideas, que hemos obtenido una muestra $\{x_K : 0 \leq K \leq N\}$ de una población descrita por una variable aleatoria continua con una función de densidad de probabilidad $p(x;\vec{\theta})$ dependiente del vector de parámetros $\vec{\theta} \in R$. Los resultados son análogos si la población es discreta. Si consideramos muestras aleatorias simples, la probabilidad de que ocurra la muestra $(x_1, x_2, \ldots, x_N)$ para un vector dado del vector de

parámetros $\vec{\theta}$ viene descrita por la función de densidad de probabilidad conjunta (f.d.p) de la muestra y que denotamos por L.

$$L(x_1, x_2, \ldots, x_n; \vec{\theta}) = \prod_{i=1}^{N} p(x_i; \vec{\theta}). \qquad (3.6.1)$$

El problema que nos planteamos ahora es, dada una muestra fija pretendemos estimar el valor del vector de parámetros $\vec{\theta}$ que es desconocido.

Recordaremos que si $(x_1, x_2 \ldots x_N)$ es una muestra dada, se llama *función de verosimilitud* a la función de $\vec{\theta}$ dada por:

$$L(x_1, x_2, \ldots, x_n; \vec{\theta}) \quad = \quad \prod_{i=1}^{N} p(x_i; \theta).$$

Recalcamos que esta función se considera ahora como función solo de $\vec{\theta}$ y no de la muestra, que ya es fija, al construirse esta función después de haber obtenido la muestra. En este sentido podríamos utilizar simplemente la notación $L(\vec{\theta})$, aunque lo más usual es mantener la muestra de la notación de L.

El Método de Máxima Verosimilitud está basado en una idea muy simple, y quizás por eso es tan importante, que consiste en considerar como mejor estimación del vector de parámetros $\vec{\theta}$ el valor que haga máxima L para la muestra dada. Es decir, dada la definición $(x_1, x_2, \ldots, x_N)$ consideremos mejor estimación $\vec{\theta_1}$ que $\vec{\theta_2}$ si

$$L(x_1, x_{2,\ldots,} x_n; \vec{\theta}_1) > L(x_1, x_2, \ldots, x_n; \vec{\theta}_2),$$

lo que significa que el valor de $\vec{\theta_1}$ es más probable que el valor de $\vec{\theta_2}$ dada la muestra reflejada en $(x_1, x_2, \ldots, x_N)$. En otras palabras, es este método de estimación la idea consiste en elegir la moda de la distribución a posteriori de $\vec{\theta}$.

Teniendo en cuenta que la función de verosimilitud es mayor o igual que cero para todo valor de $\vec{\theta}$ y que la función logaritmo es monótona creciente resulta que:

$$max[L(x_1,x_2,\ldots,x_N;\vec{\theta})]=max[\ln(L(x_1,x_2,\ldots,x_N;\vec{\theta}))].$$

Se utiliza en la práctica el $\ln L$ en lugar de L por ser más fácil, en aras de calcular el máximo de la función de verosimilitud o equivalentemente de la función de log-verosimilitud, pues la derivación de una suma es más sencilla que la de un producto.

Si la muestra depende únicamente de un parámetro, θ entonces $\hat{\theta}=\hat{\theta}(x_1,x_2,\ldots,x_n)$ es un estimador de máxima verosimilitud de θ para la distribución poblacional $p(x;\theta)$ si el estimador $\hat{\theta}$ hace máximo el valor de la función de verosimilitud, o lo que es lo mismo, hace máximo:

$$\ln L(x_1,x_2,\ldots,x_n;\theta)=\sum_{i=1}^{N}\ln p(x_1,\theta).$$

En consecuencia, el calculo efectivo de $\hat{\theta}$ se reduce al resolver la ecuación:

$$\frac{dln\,L}{d\theta}=0,$$

de forma que si $\hat{\theta}$ es solución de esta ecuación $\hat{\theta}$ será máximo si la segunda derivada es negativa.

Si la distribución poblacional depende de N parámetros $\theta_1,\theta_2,\ldots,\theta_N$ habrá que resolver el sistema

$$\frac{\partial \ln L(\theta_1,\theta_2,\ldots,\theta_N)}{\partial\theta_1}=0,\ldots,\frac{\partial \ln L(\theta_1,\theta_2,\ldots,\theta_N)}{\partial\theta_N}=0.$$

Comprobando luego que si $\hat{\theta}=(\hat{\theta}_1,\hat{\theta}_2,\ldots,\hat{\theta}_n)$ verifica este sistema, además hace el hessiano correspondiente definido negativo.

Vamos a adaptar la definición anterior a nuestro contexto, donde dada una muestra

$$\{x_k:0\le K\le N\}$$

de un proceso estocástico $x(t)$ en los momentos $\{t_k : 0 \leq K \leq N\}$ y suponiendo que el proceso $x(t)$ depende de un vector de parámetros $\vec{\theta} \epsilon R^n$, deseamos determinar el vector $\vec{\theta}$ tal que

$$max\, L(\theta) = f(t_0, x_0, t_1, x_1, \ldots, t_n, x_n; \vec{\theta}),$$

siendo $f(t_0, x_0, t_1, x_1, \ldots, t_n, x_n; \vec{\theta})$ la función de densidad conjunta de la muestra. En general, nosotros no podemos asumir que la muestra sea simple, es decir, que existe independencia por lo que no será posible legitimar la factorización dada en (3.6.1). Sin embargo, para poder aplicar el Método de Máxima Verosimilitud, necesitamos tener una expresión de la función de densidad conjunta que sea preferiblemente algebraica.

Para ello, vamos en primer lugar a recordar que el modelo de subyacente que tenemos es de la forma (véase (3.5.1))

$$d\,x(t) = f(t, x(t); \vec{\theta})\, dt + g(t, x(t); \vec{\theta})\, dB(t), x(0) = x_0, \qquad (3.6.2)$$

con

$$x(t) = S(t); \quad x_0 = s_0; \quad \vec{\theta} = (\mu, \sigma);$$

$$f(t, x(t); \quad \vec{\theta}) = \mu S(t); \quad g(t, x(t); \vec{\theta}) = \sigma S(t).$$

Se puede demostrar que la solución de una ecuación de Itô de la forma (3.6.2) es un proceso de Markov de primer orden. Esto permite, utilizando el Teorema de la Probabilidad Total, reescribir la función de densidad de probabilidad conjunta de la muestra en términos de la función de verosimilitud del siguiente modo:

$$\begin{aligned} p(t_0,x_0,t_1,x_1,t_2,x_2,\ldots,t_N,x_N;\,\vec{\theta}) &= p(t_0,x_0,\vec{\theta})\,p\,(t_1,x_1|t_0x_0;\,\vec{\theta}) \\ &\times\ p(t_2,x_2|t_1x_1;\,t_0x_0;\,\vec{\theta})\ldots \\ &\times\ \vdots \\ &\times\ p(t_N,x_N|t_{N-1}x_{N-1},\ldots,t_2,x_2,t_1,x_1.t_0,x_0;\vec{\theta}) \\ &= p\,(t_0,x_0;\,\vec{\theta})p\,(t_1,x_1|t_o,x_o,\vec{\theta})p\,(t_2,x_2|t_1,x_1;\,\vec{\theta}) \\ &\times\ p(t_N,x_N|t_{N-1},x_{N-1};\vec{\theta}), \end{aligned}$$

donde $p(t_K,x_K\vee t_{K-1},x_{K-1};\vec{\theta})$ es la función de densidad de transición del proceso estocástico $x(t)$ que empieza en (t_{K-1},x_{K-1}) y pasa a (t_k,x_k). Como se ha señalado anteriormente, para mayor comodidad en el manejo computacional trabajaremos con la función del log-verosimilitud, es decir,

$$\ln(p(t_0x_0,t_1x_1,\ldots,t_Nx_N;\vec{\theta}))=\ln(p(t_0,x_0;\vec{\theta}))+\sum_{K=1}^{N}\ln p(t_K,x_K\vee t_{K-1},x_{K-1};\vec{\theta}).$$

Como los valores $p(t_kx_k\vee t_{k-1},x_{k-1};\vec{\theta}),\quad 1\le K\le N$, suelen estar entre 0 y 1, el logaritmo neperiano de dichos valores es negativo y el problema de maximización de la función de log-verosimilitud es equivalente a obtener el mínimo de su función opuesta, por ello consideraremos el siguiente programa de optimización:

$$Min\,D(\vec{\theta}) \quad = - \quad \ln(p(t_0,x_0,\vec{\theta}))-\sum_{K=1}^{N}\ln(p(t_{K-1},x_{K-1},\vec{\theta})). \qquad (3.6.3)$$

Ahora vamos a explicitar el valor de $p(t_{K-1},x_{K-1},\vec{\theta})$. Para ello, partimos de la ecuación diferencial estocástica de Itô dada en (3.6.2) y le aplicaremos el esquema discreto de Euler-Maruyama para aproximar dicha ecuación en los instantes de la muestra que supondremos equiespaciados en el tiempo siendo Δt el paso temporal. Para ello, consideremos un subintervalo de tiempo genérico, $[t_{K-1},t_K]$ y vía la discretización de Euler-Maruyama obtenemos

$$x(t_K)-x(t_{K-1})=f((t_{K-1},x(t_{K-1});\vec{\theta})\Delta t+g(t_{K-1},x(t_{K-1});\vec{\theta})\{B(t_K)-B(t_{K-1})). \qquad (3.6.4)$$

Por tanto, si denotamos por x_K la aproximación de la solución obtenida por el esquema anterior en el instante t_K, es decir,

$$x_K \approx x(t_K),$$

el esquema (3.6.4) se reescribe como

$$x(t_K) \approx x_{K-1} + f(t_{K-1}, x_{K-1}; \vec{\theta})\Delta t + g(t_{K-1}, x_{K-1}; \vec{\theta}) \quad \{B(t_K) - B(t_{K-1})\}, \quad k = 1, \ldots, N. \qquad (3.6.5)$$

Recordemos que por las propiedades del Movimiento Browniano se tiene que

$$B(t_K) - B(t_{K-1}) \quad N(0; \sqrt{\Delta t}), \qquad k = 1,2,\ldots,N.$$

Por lo tanto de (3.6.5) se sigue que la aproximación, vía Euler-Maruyama de x_K dado x_{K-1}, sigue la siguiente distribución normal

$$x_K | x_{K-1} \sim N(\mu_K, \sigma_K), \qquad \begin{array}{ll} \mu_K = x_{K-1} + & f(t_{K-1}, x_{K-1}; \theta)\Delta t, \\ \sigma_K = & g(t_{K-1}, x_{K-1}; \vec{\theta})\Delta t. \end{array} \qquad (3.6.6)$$

Por lo tanto,

$$p(t_k, x_k | t_{K-1}, x_{K-1}, \vec{\theta}) = \frac{1}{\sqrt{2\pi}\sigma_k} e^{-\frac{1}{2}\left(x_k - \frac{\mu_k}{\sigma_K}\right)^2}, \qquad k = 1,2,\ldots,N, \qquad (3.6.7)$$

siendo μ_K y σ_K las expresiones dadas en (3.6.6). Esto nos permite escribir explícitamente la función de log- verosimilitud dada en (3.6.3) de la siguiente forma:

$$
\begin{aligned}
D(\vec{\theta}) &= -\ln\left(p(t_0, x_0;\, \vec{\theta})\right) \\
&- \sum_{K=1}^{N} ln\left(\frac{1}{\sqrt{2\pi}\,\sigma_K}\, e^{-\frac{1}{2}\left(\frac{x_K-\mu_K}{\sigma_K}\right)^2}\right) \\
&= -ln\left(p(t_0, x_0;\, \vec{\theta})\right) \\
&- Nln\left(\frac{1}{\sqrt{2\pi}\,\sigma_K}\right) + \sum_{K=1}^{N} \frac{1}{2}\left(\frac{x_K-\mu_K}{\sigma_K}\right)^2 \\
&= -\ln\left(p(t_0, x_0;\, \vec{\theta})\right) + \frac{N}{2}\ln(2\pi) \\
&+ \sum_{K=1}^{N} \ln(\sigma_K) + \frac{1}{2}\sum_{K=1}^{N}\left(\frac{x_K-\mu_K}{\sigma_K}\right)^2.
\end{aligned}
\qquad (3.6.8)
$$

Observemos que en nuestro modelo de subyacente $x_0 = s_0$ es determinista, por tanto:

$$p(t_0, x_0; \vec{\theta}) = p(0, s_0; \mu, \sigma) = 1, \qquad (3.6.9)$$

además las funciones μ_K y σ_K dadas en (3.6.6) toman la forma:

$$\mu_K = S_{K-1} + \mu S_{K-1}\Delta t\,; \quad \sigma_K = \sigma S_{K-1}\sqrt{\Delta t}\,. \qquad (3.6.10)$$

Entonces sustituyendo (3.6.9) y (3.6.10) en (3.6.8) obtenemos

$$
\begin{aligned}
D(\mu,\sigma) &= \frac{N}{2}\ln(2\pi) + \sum_{K=1}^{N} \ln\left(\sigma S_{K-1}\sqrt{\Delta t}\right) + \frac{1}{2} \sum_{K=1}^{N} \left(\frac{S_K - S_{K-1} - \mu S_{K-1}\Delta t}{\sigma S_{K-1}\sqrt{\Delta t}}\right)^2 \\
&= \frac{N}{2}\ln(2\pi) + \frac{N}{2}\ln(\Delta t) + N\,ln(\sigma) \\
&+ \sum_{K=1}^{N} \ln(S_{K-1}) + \frac{1}{2\sigma^2\Delta t}\sum_{K=1}^{N}\left(\frac{S_K}{S_{K-1}} - 1 - \mu\Delta t\right)^2 .
\end{aligned}
$$

Por lo tanto el programa de minimización de la función de Log-Verosimilitud es:

$$
\begin{aligned}
&Min\ D(\mu,\sigma) \\
&(\mu,\sigma)\epsilon\ \mathbb{R}\times\ ]0,+\infty[
\end{aligned}
\qquad
\begin{aligned}
&\frac{N}{2}\ln(2\pi) + \frac{N}{2}\ln(\Delta t) + N\,ln(\sigma) \\
&+ \sum_{K=1}^{N} \ln(S_{K-1}) + \frac{1}{2\sigma^2\Delta t}\sum_{K=1}^{N}\left(\frac{S_K}{S_{K-1}} - 1 - \mu\Delta t\right)^2 .
\end{aligned}
$$

Para calcular el mínimo de $D(\mu,\sigma)$ en primer lugar calculamos los puntos críticos, que son aquellos en los que el gradiente es nulo:

$$
\begin{aligned}
\frac{\partial D(\mu,\sigma)}{\partial \mu} &= \frac{-1}{\sigma^2}\sum_{K=1}^{N}\left(\frac{S_K}{S_{K-1}} - 1 - \mu\Delta t\right) = 0, \\
\frac{\partial D(\mu,\sigma)}{\partial \sigma} &= \frac{N}{\sigma} - \frac{1}{\sigma^3\Delta t}\sum_{k=1}^{N}\left(\frac{S_K}{S_{K-1}} - 1 - \mu\Delta t\right)^2 = 0.
\end{aligned}
\qquad (3.6.11)
$$

Resolviendo el sistema de ecuaciones (3.6.11) obtenemos como puntos críticos:

$$
\hat{\mu} = \frac{1}{N\Delta t}\sum_{k=1}^{N}\left(\frac{S_K}{S_{K-1}} - 1\right), \quad \hat{\sigma} = \frac{1}{N\Delta t}\sum_{K=1}^{N}\left(\frac{S_K}{S_{K-1}} - 1 - \mu\Delta t\right)^2 . \qquad (3.6.12)
$$

Se puede comprobar que la matriz hessiana formada por las cuatro derivadas parciales segundas de la función $D(\mu,\sigma)$, es decir,

$$H(D(\mu,\sigma))=[\frac{\partial^2 D(\mu,\sigma)}{\partial \mu^2},\frac{\partial^2 D(\mu,\sigma)}{\partial \mu \partial \sigma},\frac{\partial^2 D(\mu,\sigma)}{\partial \sigma \partial \mu},\frac{\partial^2 D(\mu,\sigma)}{\partial \sigma^2}],$$

evaluada en el punto crítico $(\hat{\mu},\hat{\sigma})$ dado en (3.6.12) tiene valores propios positivos, lo que muestra que $(\hat{\mu},\hat{\sigma})$ es un mínimo de la función opuesta de Log-Verosimilitud y por tanto son los estimadores máximos verosímiles del Modelo Log-Normal**.**

3.6.2 | Método no paramétrico

En esta sección se complementa la estimación de parámetros del Modelo de Log-Normal realizada por el Método de Máxima Verosimilitud mediante un método no paramétrico[1]. Para describir con mayor generalidad este método de estimación, se parte de una ecuación diferencial estocástica tipo Itô general dada en (3.6.2), donde $\vec{\theta}\in R^n$ el vector de parámetros que se desea estimar. En el caso del Modelo Log-Normal, $\vec{\theta}\in R^2$. A continuación se discretiza la ecuación (3.6.2) por algún método numérico, por ejemplo, mediante un esquema de Euler-Maruyama:

$$x(t+\Delta t)-x(t)=f(t,x(t);\vec{\theta})\Delta t+g(t,x(t);\vec{\theta})\Delta B(t),$$

$$x(t+\Delta t)=x(t)+f(t,x(t);\vec{\theta})\Delta t+g(t,x(t);\vec{\theta})\Delta B(t). \qquad (3.6.13)$$

Fijada una partición del intervalo de tiempo $[0,T]$ donde se quiere aproximar la ecuación (3.6.13):

$$0=t_0<t_1<t_2<\ldots<t_{N-1}=T,$$

con paso fijo $\Delta t=\frac{T}{N}$, obsérvese que se está dividiendo el intervalo $[0,T]$ en N subintervalos de la forma:

$$[t_i,t_{i+1}],\ \ 0\le i\le N-2,$$

siendo $\Delta t=t_{i+1}-t_i$, y denotando por x_i la aproximación de $x(t)$ en el punto t_i, por ejemplo, $x_i\cong x(t_i),0\le i\le N-1,$ según (3.6.13) se obtiene:

1

$$x_{i+1}=x_i+f(t_i,x_i;\vec{\theta})\,\Delta t+g(t_i,x_i;\vec{\theta})\sqrt{\Delta t}\,Z_i,$$

donde se ha utilizado que (véase propiedad MB.4 del Movimiento Browniano)

$$B(t_i+\Delta t)-B(t_i)\ \ N(0;\sqrt{\Delta t}).$$

Ahora utilizando que la solución de la ecuación diferencial estocástica (3.6.13) es un proceso de difusión, se cumplen las siguientes relaciones:

$$E[\frac{x_{i+1}-x_i}{\Delta t}-f(t_i,x_i;\vec{\theta})]=\vartheta(\Delta t),\qquad (3.6.14)$$

$$E[\frac{(x_{i+1}-x_i)^2}{\Delta t}]-(g(t_i,x_i;\vec{\theta}))^2=\vartheta(\Delta t),\qquad (3.6.15)$$

que, en su notación de la derecha, indican que el tamaño de los órdenes de los momentos de los términos que aparezcan en los miembros de la izquierda.

Para estimar el vector de parámetros a partir de una muestra $\{x_0,x_1,\ldots,x_{N-1}\}$ se utilizan las relaciones (3.6.14)-(3.6.15) en su versión muestral:

$$\sum_{i=0}^{N-1} f(t_i,x_i;\vec{\theta})=\frac{1}{\Delta t}\sum_{i=0}^{N-1}(x_{i+1}-x_i),\qquad (3.6.16)$$

$$\sum_{i=0}^{N-1}(g(t_i,x_i;\vec{\theta}))^2=\frac{1}{\Delta t}\sum_{i=0}^{N-1}(x_{i+1}-x_i)^2.\quad (3.6.17)$$

En el caso del Modelo Log-Normal se escribe a partir de la muestra $\{x_0,x_1,\ldots,x_{N-1}\}$ que en el Modelo Log-Normal son las cotizaciones $\{s_0,s_1,\ldots s_{N-1}\}$, resolviendo el sistema de ecuaciones (3.6.16) y (3.6.17), se estimaría el vector de parámetros $\vec{\theta}\in R^2$. Obsérvese que el contexto del Modelo Log-Normal,

$$f(t_i,x_i;)=\mu\,s_i,g(t_i,x_i;)=\sigma\,s_i,$$

con la identificación $x_i=s_i$.

3.7 Validación del Modelo Log-Normal

Con objeto de analizar el buen ajuste del modelo se pueden emplear diversas herramientas. No obstante, en esta memoria las medidas de bondad de ajuste que se emplearán para la validación del Modelo Log-Normal aplicado al subyacente IAG (IAG.MC) serán:

- Error cuadrático medio (Root Mean Squared Error, RMSE).
- Error porcentual absoluto medio (Mean Absolute Percentage Error, MAPE).
- Construcción de Intervalos de Confianza del 95%.
- Gráfico de observaciones y predicciones (puntuales y por intervalos).

En base a los resultados obtenidos según las medidas de bondad del ajuste anteriormente especificadas, se evalúa la adecuación y ajuste del modelo, para proceder así a realizar las predicciones del subyacente en cuestión.

3.7.1 Medidas de bondad de ajuste

Error cuadrático medio (RMSE)

La medida de bondad de ajuste RMSE, mide la distancia euclídea por término medio entre los valores observados y los estimados. Cuanto menor sea el resultado del RMSE, entonces se podrá decir que el error en el modelo presentado será menor. Su valor está definido de la siguiente manera:

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N} (S_i - \hat{S}_i)^2}{N}}.$$

El cálculo del RMSE se realizará considerando las predicciones puntuales (función media) que se obtengan de aplicar el modelo teórico Log-Normal con ambos métodos de estimación de los parámetros explicados el Apartado 3.6. Calculados y contrastados los resultados del RMSE que muestre cada método, será posible tener una primera idea de cuál de ellos muestra un mejor ajuste.

Error porcentual absoluto medio (MAPE)

La segunda de las medidas de precisión de pronóstico a emplear, el MAPE, se calcula como el promedio de las diferencias absolutas entre los valores pronosticados y los observados dividido por las observaciones correspondientes, quedando expresado como un porcentaje de los valores reales. Es decir, si se pronostican k periodos y los valores reales corresponden a k periodos, el MAPE se calcula como:

$$MAPE = \frac{100}{N} \sum_{i=1}^{N} \frac{|S_i - \hat{S}_i|}{\hat{S}_i}.$$

Del mismo modo que con el RMSE, el MAPE será calculado considerando las predicciones puntuales dadas a partir de la aplicación del modelo teórico Log-Normal con cada uno de los métodos de estimación de los parámetros explicados. Además de realizar el correspondiente contraste de los resultados que esta medida presente con cada uno de los métodos, individualmente, se considerará un error global de diagnóstico tolerable, cuando el MAPE obtenido sea inferior al 5%.

3.7.2. Validación por intervalos de confianza

La construcción y estudio de intervalos de confianza (IC) del 95% es otra de las herramientas a emplear de cara a validar la aplicación del modelo Modelo Log-Normal al activo subyacente IAG (IAG.MC). En la construcción de los IC, hay dos elementos fundamentales a considerar. Por un lado, la amplitud del intervalo da la precisión de la estimación y por tanto, deberá ser la menor posible y, por otro lado, la probabilidad de que el intervalo contenga el verdadero valor del parámetro a estimar (IAG (IAG.MC)), que se llama nivel de confianza e interesará que sea la mayor posible. En base a esto, es cierto que se puede ganar en precisión a costa de perder confianza en la estimación. En la presente memoria, como se ha indicado, los intervalos de confianza construidos atienden a un nivel de confianza del 95%.

Este apartado se centra en la estimación de la variable en cuestión (IAG (IAG.MC)) a partir del intervalo que tiene por extremos los valores de dos funciones para una muestra, siendo esta una estimación de la variable por intervalos de confianza.

Para centrar ideas, supongamos que se selecciona una muestra aleatoria simple de una población descrita por la función $f(x;\theta)$ dependiente del parámetro θ que se pretende estimar. El problema se plantea como sigue: se fija un nivel de confianza, que se denota por $1-\alpha$, en donde $0<\alpha<1$, y se trata de determinar la función $\theta_1(x_1,x_2,\ldots,x_n)$ y la función $\theta_2(x_1,x_2,\ldots,x_n)$ de forma que:

$$P[\theta_1(x_1,x_2,\ldots,x_n)\leq\theta\leq\theta_2(x_1,x_2,\ldots,x_n)]=1-\alpha.$$

Al intervalo $[\theta_1(x_1,x_2,\ldots,x_n),\theta_2(x_1,x_2,\ldots,x_n)]$ se le denomina *intervalo de confianza* del parámetro θ al nivel de confianza del $(1-\alpha)100\%$. Es importante observar que sería un error afirmar que la probabilidad indicada anteriormente, es la probabilidad de que θ esté entre los números reales $\theta_1(x_1,x_2,\ldots,x_n)$ y $\theta_2(x_1,x_2,\ldots,x_n)$, puesto que θ no es una variable aleatoria sino un parámetro que tendrá un valor concreto. Las variables aleatorias son $\theta_1(x_1,x_2,\ldots,x_n)$ y $\theta_2(x_1,x_2,\ldots,x_n)$ al variar la muestra, luego la probabilidad anterior debe considerarse como la probabilidad de que el intervalo aleatorio denotado como $[\theta_1(x_1,x_2,\ldots,x_n),\theta_2(x_1,x_2,\ldots,x_n)]$ contenga el verdadero valor de θ.

Dicho en términos de frecuencias esto significa que de cada 100 muestras aleatorias que se tomen, cabe esperar que el $(1-\alpha)100\%$ de ellas contenga el verdadero valor de θ entre $\theta_1(x_1,x_2,\ldots,x_n)$ y $\theta_2(x_1,x_2,\ldots,x_n)$.

Suponiendo que $(x_1,x_2,\ldots,x_n)$ es una muestra aleatoria simple y que $\bar{x}$ es la media muestral. La variable aleatoria $\bar{x}$ media muestral se distribuye según una normal

$$N(\mu;\frac{\sigma}{\sqrt{n}}),$$

suponiendo que la población es $N(\mu;\sigma)$ con σ conocida. Esta propiedad permite asegurar que la variable tipificada $\frac{\bar{x}-\mu}{\sigma/\sqrt{n}}$ sigue una $N(0;1)$, y por lo tanto, fijando un nivel de confianza $1-\alpha$, obtener el valor $\lambda_{\alpha/2}$ tal que

$$P\left[\left|\frac{\overline{x}-\mu}{\sigma/\sqrt{n}}\right|<\lambda_{\alpha/2}\right]=1-\alpha\,.$$

Operando, resulta:

$$1-\alpha=P\left[\overline{x}-\lambda_{\alpha/2}\frac{\sigma}{\sqrt{n}}\leqslant\mu\leqslant\overline{x}+\lambda_{\alpha/2}\frac{\sigma}{\sqrt{n}}\right].$$

Con lo que las funciones buscadas son:

$$\Theta_1=\overline{x}-\lambda_{\alpha/2}\frac{\sigma}{\sqrt{N}},\quad \Theta_2=\overline{x}+\lambda_{\alpha/2}\frac{\sigma}{\sqrt{N}}.$$

El intervalo de confianza para la media poblacional al nivel de confianza $(1-\alpha)100\%$ es:

$$\left[\overline{x}-\lambda_{\alpha/2}\frac{\sigma}{\sqrt{N}},\overline{x}+\lambda_{\alpha/2}\frac{\sigma}{\sqrt{N}}\right].$$

Nótese que se puede ganar en precisión de dos formas, bien perdiendo confianza lo que, en general, no interesa, o bien aumentando el tamaño n de la muestra seleccionada.

3.7.3. Gráficos comparativos

Finalmente, calculada la función media (predicción puntual) y los intervalos de confianza (predicción por intervalos) para cada uno de los instantes de tiempo que comprenden el horizonte temporal observado y bajo cada uno de los métodos de estimación dados para la validación del modelo, se realizarán sus representaciones gráficas.

A través de estas, se realizará un estudio comparativo mediante el cual, visualmente, se podrá observar por un lado, el grado de ajuste existente entre las observaciones de la muestra (valor real de la cotización de IAG (IAG.MC)) y las predicciones puntuales (medias) obtenidas en cada uno de los métodos. Por otro lado, este análisis cualitativo se reforzará cuantitativamente calculando las medidas de bondad de ajuste anteriormente señaladas y una comparativa de los intervalos de confianza obtenidos en ambos casos.

3.8 Predicción

Uno de los objetivos que se persigue con los modelos económicos es el poder realizar predicciones de la variable en cuestión. El modelo ajustado debe permitir predecir el valor medio (estimación puntual) , y encontrar un intervalo que contenga el valor real con una determinada probabilidad (estimación por intervalos).

Obtener una estimación puntual será posible mediante el valor esperado de la variable estudiada Y, asociado a determinados valores de las variables explicativas del modelo ajustado. Para ello, se calcula la estimación de Y correspondiente a su valor medio condicionado $E(Y|x_1,\ldots,x_k)$, como se expresa:

$$\hat{Y}=s_0 e^{(\hat{\mu}-\frac{1}{2}\hat{\sigma}^2)t+\hat{\sigma}\sqrt{t}Z} \quad , \quad Z\sim N(0;1).$$

Por otro lado, la predicción puntual debe completarse con la varianza de la estimación, conociéndose así el error de la predicción puntual y obteniendo con ambos valores, el intervalo de confianza.

Capítulo 4 | Aplicación y contexto

4.1 | De los modelos matemáticos a la aplicación

En este apartado describiremos como hemos adaptado los modelos anteriormente desarrollados para tener una herramienta útil.

4.1.1 | Procesos internos de cálculo (*backend*)

El objetivo de la aplicación es automatizar el cálculo de los modelos descritos en el capítulo anterior.

En la Figura 4.1 se muestra el flujo de la aplicación.

```
class Method:
  @staticmethod
  def base(symbol, graphic_name, graphic_location, historical_data_location, data_location, data_name, model):
    scraper = YahooScraper(symbol)
    prices, dates = scraper.get_prices_and_dates()
    model = eval(model).run(prices)
    GraphicsCreator.create(graphic_name, graphic_location, DatesManager(), model)
    HistoricalData.save_file(model, historical_data_location, dates)
    FileGenerator.save_file(data_location, data_name, dates, DatesManager(), model, historical_data_location)
```

Figura 4.1: Flujo de la aplicación

Fuente: Elaboración propia (Código Python)

La imagen superior muestra el flujo del cálculo de un modelo. Es común para ambos modelos, solo cambia cuando realiza los cálculos matemáticos. Esto es especialmente útil para cuando queramos añadir un tercer modelo ya que solo deberemos desarrollar la parte matemática de este.

Podemos leer el contenido del método y sabremos lo que hace.

Para aplicar un modelo concreto lo que haremos será pasar una configuración, en la Figura 4.2 se muestra, el código para el primer modelo.

```
class MethodOne:
  @staticmethod
  def run(symbol):
    graphic_name = 'METODO 1'
    graphic_location = 'web_app/public/data/model_one/graphic.png'
    historical_data_location = 'historico.txt'
    data_location = 'web_app/public/data/model_one/model_data.js'
    data_name = 'model_one_data'
    model = 'ModelOne'

    Method.base(symbol, graphic_name, graphic_location, historical_data_location, data_location, data_name, model)
```

Figura 4.2: Flujo del modelo uno
Fuente: Elaboración propia (Código Python)

Lo que hacemos es pasarle la siguiente configuración:

- *Graphic_name* es el título del gráfico.
- *Graphic_location* es la ruta donde se almacenará el gráfico
- *Historical_data_location* es el archivo donde guardaremos cierta información sobre los precios
- *Data_location* es el archivo donde guardaremos en JSON los datos que calculamos a tráves del modelo
- Data_name es la referencia que le daremos a los cálculos del modelo para luego poder mostrarlo en la página web.
- *Model* es el nombre de la clase que tiene la lógica del cálculo matemático del modelo

En la Figura 4.3 instanciamos el scraper que cogerá los datos de Yahoo.

```
scraper = YahooScraper(symbol)
```

Figura 4.3: Detalle del scraper de Yahoo
Fuente: Elaboración propia (Código Python)

Luego en la Figura 4.4 con ese objeto se llama al método de obtener los precios y los datos.

```
prices, dates = scraper.get_prices_and_dates()
```

Figura 4.4: detalle de los precios y las fechas
Fuente: Elaboración propia (Código Python)

En la Figura 4.5 podemos ver en la clase de Yahoo lo que hace ese método.

```
import urllib
import re

class YahooScraper:
  PRICES_REGEX = '<td class="yfnc_tabledata1" align="right">(.+?)</td>'
  DATES_REGEX  = '<td class="yfnc_tabledata1" nowrap align="right">(.+?)</td>'

  def __init__(self, symbol):
    self.symbol = symbol

  def get_prices_and_dates(self):
    plain_text = self.__get_page(self.__url)
    return (self.get_prices(plain_text), self.get_dates(plain_text))

  def get_prices(self, plain_text):
    return self.__find_prices(plain_text)

  def get_dates(self, plain_text):
    return self.__find_dates(plain_text)

  def __url(self):
    return "http://finance.yahoo.com/q/hp?s={0}+Historical+Prices".format(self.symbol)

  def __get_page(self, url):
    htmlfile = urllib.urlopen(self.__url())
    return htmlfile.read()

  def __find_prices(self, plain_text):
    price_matches = re.compile(self.PRICES_REGEX)
    return re.findall(price_matches, plain_text)

  def __find_dates(self, plain_text):
    date_matches = re.compile(self.DATES_REGEX)
    return re.findall(date_matches, plain_text)
```

Figura 4.5: Código de la clase del scraper de Yahoo

Fuente: Elaboración propia (Código Python)

Si vemos el contenido del método *get_prices_and_dates* entenderemos como saca los precios y las fechas. Lo que hace es entrar en la página web a través de la librería de peticiones HTTP *urllib* (ver métodos privados *url* y *get_page*) y luego aplica una expresión regular para sacar los datos que necesita (ver métodos privados *find_prices* y *find_dates*).

Una vez tenemos los datos de los precios de la acción y las fechas, lo siguiente es aplicar el modelo (véase la Figura 4.6).

```
model = eval(model).run(prices)
```

Figura 4.6: Detalle del código que evalúa el modelo

Fuente: Elaboración propia (Código Python)

En el caso del primer modelo, la representación matemática en forma de código se encuentra en representada en la Figura 4.7.

```
from lib.model.helpers import *
from lib.model.model import *

class ModelOne(Model):
  NAME = 'model_one'

  @staticmethod
  def run(prices):
    model = ModelOne(prices)
    model.estimations()
    return model

  def __init__(self, prices):
    self.close_prices = PricesManager.get(prices)
    self.model_run = False
    self.number_of_close_prices = len(self.close_prices)

  def estimations(self):
    if self.model_run:
      return self.predicitions

    ultimoshare = self.close_prices[29]
    control = True

    while control:
      ini = numpy.random.rand(2)
      steps = 1
      res = minimize(
        L,
        ini,
        args=(self.close_prices, self.number_of_close_prices, steps),
        method = 'Nelder-Mead'
      )
      estmu = res.x[0]
      ultest = media(30, self.close_prices, estmu)

      if abs(ultimoshare - ultest) < 3:
          control = False

    estsigma = res.x[1]

    predicitions = [estmu, estsigma]
    self.predicitions = predicitions
    self.model_run = True

    return self.predicitions
```

Figura 4.7: Clase que contiene la lógica del modelo uno

Fuente: Elaboración propia (Código Python)

Como vemos hay una interfaz común donde sacamos las estimaciones dadas por el modelo (ver el método *estimations*). Los resultados del segundo modelo los resultados se muestran en la Figura 4.8.

```
from lib.model.helpers import *
from lib.model.model import *

class ModelTwo(Model):
  NAME = 'model_two'

  @staticmethod
  def run(prices):
    model = ModelTwo(prices)
    model.estimations()
    return model

  def __init__(self, prices):
    self.close_prices = PricesManager.get(prices)
    self.model_run = False
    self.number_of_close_prices = len(self.close_prices)

  def estimations(self):
    if self.model_run:
      return self.predicitions

    drift_denominator = numpy.sum(numpy.array(self.close_prices))
    drift_numerator = numpy.sum(numpy.array([self.close_prices[i + 1] - self.close_prices[i] for i in range(1, (self.number_of_close_prices - 1))]))
    drift = drift_numerator / drift_denominator

    volatility_denominator = numpy.sum(numpy.array(self.close_prices ** 2))
    volatility_numerator = numpy.sum(numpy.array([(self.close_prices[i + 1] - self.close_prices[i]) ** 2 for i in range (1, (self.number_of_close_prices - 1))]))
    volatility = numpy.sqrt(volatility_numerator / volatility_denominator)

    predicitions = [drift, volatility]
    self.predicitions = predicitions
    self.model_run = True
    return self.predicitions
```

Figura 4.8: Clase que contiene la lógica del modelo dos

Fuente: Elaboración propia (Código Python)

Vemos que es igual, hay un método *run* que saca las estimaciones.

Ya sabemos cómo sacar las estimaciones pero en la web mostramos una serie de datos variados, desde un gráfico a una tabla con la bondad de los errores. Para sacar esos datos hay un modelo común que tiene la lógica de calcular cosas como el MAPE, los intervalos de confianza o los precios medios (véase la Figura 4.9).

```
from lib.model.helpers import *

class Model:
  def mu(self):
    return self.estimations()[0]

  def sigma(self):
    return self.estimations()[1]

  def mean_prices(self):
    if self.model_run == False:
      raise ValueError('The model should be run')

    return numpy.array([media(i, self.close_prices, self.mu()) for i in range(0, self.number_of_close_prices + 5)])

  def inferior_confidence_interval(self):
    if self.model_run == False:
      raise ValueError('The model should be run')

    return numpy.array([mediamenosdt(i, self.close_prices, self.sigma(), self.mu()) for i in range(0, self.number_of_close_prices + 5)])

  def superior_confidence_interval(self):
    if self.model_run == False:
      raise ValueError('the model should be run')

    return numpy.array([mediamasdt(i, self.close_prices, self.sigma(), self.mu()) for i in range(0, self.number_of_close_prices + 5)])

  def graphic_prediction(self):
    if self.model_run == False:
      raise ValueError('the model should be run')

    return numpy.array([media(i, self.close_prices, self.mu()) for i in range(self.number_of_close_prices, self.number_of_close_prices + 5)])

  def error_estimation(self):
    return numpy.array([media(i, self.close_prices, self.mu()) for i in range(0, self.number_of_close_prices)])

  def absolute_error(self):
    return self.error_estimation() - self.close_prices

  def cuadratic_error(self):
    cuadratic_close_prices_error = self.absolute_error() ** 2
    cuadratic_error = sum(cuadratic_close_prices_error) / len(cuadratic_close_prices_error)
    return two_decimals_float(cuadratic_error)

  def mape(self):
    close_prices_error = self.absolute_error() / self.close_prices
    mape_error = sum(close_prices_error) / len(close_prices_error) * 100
    return two_decimals_float(mape_error)
```

Figura 4.9: Abstracción de modelo

Fuente: Elaboración propia (Código Python)

Con este objeto ya tenemos todos los cálculos necesarios para mostrarlos en la interfaz. Algunos cálculos se han extraído a una utillaje que permite su reutilización. El archivo helpers puede verse en la Figura 4.10.

```
import numpy

from scipy.optimize import minimize
from lib.model.error_calculator import *
from lib.data_manager.helpers import *
from lib.data_manager.prices_manager import *

def f(x):
  return x[0] * x[1]

def mu(x, tmp_close_prices, tmp_incremento):
    ux = tmp_close_prices[x[0] - 1]
    return ux + f([ux, x[1]]) * tmp_incremento

def sigma(x, tmp_close_prices, tmp_incremento):
  return f([tmp_close_prices[x[0] - 1], x[1]]) * numpy.sqrt(tmp_incremento)

def L(teta, tmp_close_prices, tmp_number_of_close_prices, tmp_incremento):
    c0 = - 0.5
    c1 = 1 / numpy.sqrt(2 * numpy.pi)
    s = numpy.array([sigma([i, teta[1]], tmp_close_prices, tmp_incremento) for i in range(1, tmp_number_of_close_prices)])
    dif = numpy.array([tmp_close_prices[i] - mu([i, teta[0]], tmp_close_prices, tmp_incremento) for i in range(1, tmp_number_of_close_prices)])
    ter = c0 * numpy.power(dif / s, 2)

    L1 = c1 / s
    L2 = numpy.exp(ter)
    L3 = - numpy.product(numpy.multiply(L1, L2))
    return L3

def media(t, tmp_close_prices, tmp_estmu):
  auxiliar = numpy.exp(tmp_estmu * t)
  return tmp_close_prices[0] * auxiliar

def varianza(t, tmp_close_prices, tmp_estsigma, tmp_estmu):
    au1 = tmp_close_prices[0] ** 2
    b = 2 * tmp_estmu * t
    au2 = numpy.exp(b)
    c = tmp_estsigma ** 2
    au3 = numpy.exp(c * t) - 1
    return au3 * au2 * au1

def mediamenosdt(t, tmp_close_prices, tmp_estsigma, tmp_estmu):
    aux = 2 * numpy.sqrt(varianza(t, tmp_close_prices, tmp_estsigma, tmp_estmu))
    return media(t, tmp_close_prices, tmp_estmu) - aux

def mediamasdt(t, tmp_close_prices, tmp_estsigma, tmp_estmu):
    aux = 2 * numpy.sqrt(varianza(t, tmp_close_prices, tmp_estsigma, tmp_estmu))
    return media(t, tmp_close_prices, tmp_estmu) + aux
```

Figura 4.10: Detalle del utillaje

Fuente: Elaboración propia (Código Python)

Por ejemplo, cálculos como la media y la varianza no son exclusivos de un modelo en concreto. Por esa razón hemos extraído esa lógica a una serie de métodos agnósticos.

Con los datos ya obtenidos, el siguiente paso es el de crear y guardar los gráficos (véase Figura 4.11).

```
GraphicsCreator.create(graphic_name, graphic_location, DatesManager(), model)
```

Figura 4.11: Detalle de la creación del gráfico

Fuente: Elaboración propia (Código Python)

Para ello lo que hace se muestra en la Figura 4.12.

```
import matplotlib
matplotlib.use('Agg')
import numpy as np
import pylab as pl

class GraphicsCreator:
  @staticmethod
  def create(title, filename, dates_manager, model):
    graphic = GraphicsCreator(dates_manager, model)
    top_limit = max(model.superior_confidence_interval())
    graphic.create_and_save_full(title, filename, top_limit)

  def __init__(self, dates_manager, model):
    self.close_prices = model.close_prices
    self.mean_prices = model.mean_prices()
    self.prediction = model.graphic_prediction()
    self.superior_confidence_interval = model.superior_confidence_interval()
    self.inferior_confidence_interval = model.inferior_confidence_interval()
    self.data_dates = dates_manager.DATA_DATES_RANGE
    self.full_model_dates_range = dates_manager.FULL_MODEL_DATES_RANGE
    self.prevision_dates_range = dates_manager.PREVISION_DATES_RANGE

  def create_and_save_full(self, title, filename, y_top_limit):
    self.print_real_quotation(pl)
    self.print_mean_quotation(pl)
    self.print_predictions(pl)
    self.print_confidence_intervals(pl)
    self.set_title(pl, title)
    self.set_axis_labels(pl)
    self.set_axis_limits(pl, y_top_limit)
    self.set_legend(pl)
    self.save_graphic(pl, filename)
    pl.clf()

  def print_real_quotation(self, pl):
    pl.plot(self.data_dates, self.close_prices, '-bo', label='Cotizacion real')
    return pl

  def print_mean_quotation(self, pl):
    pl.plot(self.full_model_dates_range, self.mean_prices, 'g', label='Media')
    return pl

  def print_predictions(self, pl):
    pl.plot(self.prevision_dates_range, self.prediction, 'yo', label='Predicciones')
    return pl

  def print_confidence_intervals(self, pl):
    pl.plot(self.full_model_dates_range, self.inferior_confidence_interval, 'r', label='IC 95% inferior')
    pl.plot(self.full_model_dates_range, self.superior_confidence_interval, 'r', label='IC 95% superior')
    return pl

  def set_title(self, pl, title):
    pl.title(title)
    return pl
```

Figura 4.12: Detalle de la clase encargada de crear el gráfico

Fuente: Elaboración propia (Código Python)

Primero sacar todos los datos que necesita y luego instanciar un gráfico vacío con la librería de *matplotlib* y luego va dibujando cada curva con una etiqueta (véase Figura 4.13).

```
def print_real_quotation(self, pl):
  pl.plot(self.data_dates, self.close_prices, '-bo', label='Cotizacion real')
  return pl
```

Figura 4.13: Detalle de la impresión de las líneas del gráfico

Fuente: Elaboración propia (Código Python)

Con el gráfico ya sacado lo siguiente será guardar tanto los datos históricos como los datos del modelo como representa la Figura 4.14.

```
HistoricalData.save_file(model, historical_data_location, dates)
FileGenerator.save_file(data_location, data_name, dates, DatesManager(), model, historical_data_location)
```

Figura 4.14: Detalle de la creación de los ficheros

Fuente: Elaboración propia (Código Python)

Para ello a través de los respectivos objetos *HistoricalData* y *FileGenerator* guardaremos en una base de datos y en archivos que más tarde utilizaremos para mostrar los datos en la web.

```
import numpy
import datetime
import MySQLdb

from lib.data_manager.dates import *

class HistoricalData:
  @staticmethod
  def save_file(model, historical_data_filename, dates):
    to_file = HistoricalData(historical_data_filename, dates)
    to_file.save(model)

  def __init__(self, historical_data_filename, dates):
    self.historical_data_filename = historical_data_filename
    self.dates = dates
    self.database_connection = MySQLdb.connect(
      host="localhost",
      user="root",
      passwd="12345678",
      db="ITX"
    )
    self.database_interface = self.database_connection.cursor()

  def save(self, model):
    self.__generate_historical_data(model)

  def get_hist(self, model):
    date_to_insert = DatesManager().next_quotation_days(self.dates, 1)[0]
    unformatted_date = datetime.datetime.strptime(date_to_insert, '%b %d, %Y')
    self.database_interface.execute("""SELECT mean_price, inferior_confidence_interval, superior_confidence_interval FROM predictions WHERE model_name = %s AND date < %s ORDER BY date DESC LIMIT 5"""
, [model.NAME, unformatted_date])
    results = [i for i in self.database_interface.fetchall()]
    return numpy.asarray(results)
```

Figura 4.15: Clase que contiene la lógica de los datos históricos

Fuente: Elaboración propia (Código Python)

En la Figura 4.15 podemos ver la interfaz de los datos históricos para los cuales empleamos una base de datos para luego ver el rendimiento del modelo.

Finalmente guardamos los datos a mostrar en un archivo en la ruta indicada (véase 4.16).

```
import time
import datetime

from lib.model.model import *
from lib.data_manager.historical_data import *
from lib.data_manager.helpers import *

class FileGenerator:
  @staticmethod
  def save_file(file_name, variable_name, dates, dates_manager, model, historical_data_filename):
    to_file = FileGenerator(historical_data_filename)
    to_file.load_data(dates, dates_manager, model)
    to_file.save(file_name, variable_name)

  def __init__(self, historical_data_filename):
    self.model_data = {}
    self.historical_data_filename = historical_data_filename

  def save(self, file_name, variable_name):
    f = open(file_name, 'w')
    data_parsed = "%s = %s;" % (variable_name, self.model_data)
    f.write(data_parsed)
    f.close()

  def load_data(self, dates, dates_manager, model):
    self.with_prediction_table(model)
    self.with_errors_table(model)
    self.with_performance_table(dates, model)
    self.with_graphic_table(dates_manager, model)
    self.with_last_updated()
    self.with_dates_references(dates)
```

Figura 4.16: Detalle de la clase encargada de generar los archivos
Fuente: Elaboración propia (Código Python)

Los procesos internos terminan en un archivo con aspecto similar al representado en la Figura 4.17.

```
model_one_data = {'errors_table': {'cuadratic_error': '0.61', 'mape': '10.53'}, 'prediction_table': {'in
ferior_ic': ['1.87', '1.80', '1.73', '1.67', '1.61'], 'prediction': ['5.06', '5.02', '4.99', '4.96', '4.
92'], 'superior_ic': ['8.25', '8.25', '8.25', '8.24', '8.24']}, 'last_updated_time': '25/06/2016 16:01:3
5', 'performance_table': {'close_prices': [6.33, 6.28, 6.35, 6.59, 4.82], 'dates': ['Jun 20, 2016', 'Jun
 21, 2016', 'Jun 22, 2016', 'Jun 23, 2016', 'Jun 24, 2016'], 'predictions': [28.34207, 28.39017, 5.79905
, 5.89381, 6.74552], 'inferior_ic': [21.89265, 22.05243, 4.54784, 4.56531, 4.74933], 'superior_ic': [34.
79149, 34.7279, 7.05027, 7.22231, 8.7417]}, 'days_span': {'next_quotation_days': ['Jun 27, 2016', 'Jun 2
8, 2016', 'Jun 29, 2016', 'Jun 30, 2016', 'Jul 01, 2016'], 'last_quotation_days': ['Jun 20, 2016', 'Jun
21, 2016', 'Jun 22, 2016', 'Jun 23, 2016', 'Jun 24, 2016']}, 'graphic_table': {'inferior_confidence_inte
rval': [[6.24], [5.499372266506236], [5.173540115430675], [4.917891299857176], [4.6993469298700745], [4.
50500050259316], [4.328185220843263], [4.1649120428545165], [4.012562816057255], [3.869305406380327], [3
.733795738245973], [3.6050113290812553], [3.4821515997013663], [3.3645748402993814], [3.251756570740911]
, [3.1432610239379737], [3.038721016854235], [2.937823371524049], [2.8402981188562215], [2.7459103476152
9], [2.6544539449635387], [2.5657467166156467], [2.4796265310158625], [2.3959482356163226], [2.314581163
587262], [2.2354070978397913], [2.1583185933931412], [2.0832175835308377], [2.010014212898807], [1.93862
58537177166], [1.8689762709725848], [1.8009949097384843], [1.7346162833520222], [1.669779445403195], [1.
606427531829003]], 'prediction': [[5.060218571686791], [5.024992668224838], [4.9900119842642034], [4.955
274812748491], [4.920779458504704]], 'prevision_dates_range': [30, 31, 32, 33, 34], 'superior_confidence
_interval': [[6.24], [6.893750206569343], [7.133309614749691], [7.303286261349747], [7.436754855485738],
 [7.5466177483383925], [7.639537614296771], [7.7194994010328015], [7.789117195522595], [7.85021909454572
15], [7.904145164493327], [7.951913906661046], [7.9943219466714055], [8.032007068291147], [8.06548985294
4975], [8.09520219615143], [8.121507436328724], [8.14471493358807], [8.165090865743615], [8.182866379146
141], [8.19824384795733], [8.211401753812666], [8.222498541463082], [8.231675702316725], [8.239060267550
08], [8.244766843907781], [8.248899291159177], [8.251552115767192], [8.252811637616492], [8.252756973628
61], [8.251460872400997], [8.248990426711192], [8.245407685176385], [8.240770180093786], [8.235131385180
406]], 'mean_prices': [[6.24], [6.196561236537789], [6.153424865090183], [6.110588780603462], [6.0680508
92677906], [6.025809125465776], [5.983861417570017], [5.942205721943659], [5.900840005789925], [5.859762
2504630245], [5.81897045136965], [5.778462617871151], [5.738236773186386], [5.698290954295264], [5.65862
3211842943], [5.619231610044702], [5.58011422659148], [5.541269152556059], [5.502694492299918], [5.46438
8363380715], [5.426348896460435], [5.388574235214157], [5.351062536239472], [5.313811968966524], [5.2768
20715568671], [5.2400869708737865], [5.20360894227615841, [5.1673848496490145], [5.13141292525765], [5.0
95691413673164], [5.060218571686791], [5.024992668224838], [4.9900119842642034], [4.955274812748491], [4
.920779458504704]], 'data_dates_range': [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 1
8, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29], 'full_model_dates_range': [0, 1, 2, 3, 4, 5, 6, 7, 8, 9,
_10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34]}};
```


Figura 4.17: Ejemplo de archivo resultado

Fuente: Elaboración propia (Código JSON)

Que es fundamentalmente, los datos de las tablas en forma de archivo.

4.1.2 | Visualización de los datos a través de la web (*frontend*)

Para mostrar los datos, utilizaremos una librería de plantillas llamada *handlebars.js* que permite reutilizar las plantillas y tener solo un punto de divergencia en los datos repetidos. Así pues definiendo una simple configuración podemos reutilizar la cabecera.

En el caso de la pantalla principal, la carga se hace a través del código de la Figura 4.18.

```
<!-- Templating -->
<script type="text/javascript">
  views = [
    {
      route: 'templates/header.hbs',
      context: undefined
    },
    {
      route: 'templates/about.hbs',
      context: undefined
    }
  ]

  load_views(views);
</script>
```

Figura 4.18: Ejemplo de generación de página principal

Fuente: Elaboración propia (Código JavaScript)

En el caso de un modelo se carga a través del código específico del modelo (véase Figura 4.19).

```
<!-- Templating -->
<script type="text/javascript">
  views = [
    {
      route: 'templates/header.hbs',
      context: undefined
    },
    {
      route: 'templates/model.hbs',
      context: {
        model_name: 'Método 1',
        company_name: 'INDITEX',
        data: model_one_data,
        image: 'public/data/model_one/graphic.png'
      }
    }
  ]

  load_views(views);
</script>
```

Figura 4.19: Ejemplo de generación de página del modelo

Fuente: Elaboración propia (Código JavaScript)

Además la carga de los datos en las tablas de visualización es agnóstica y se cargan los datos con plantillas.

```
<!-- About Model -->
<section id="model">
    <br/>
    <br/>
    <br/>
    <div class="container">
        <div class="row">
            <div class="col-lg-12 text-center">
                <h3>{{ model_name }}</h3>
            </div>
        </div>
        <br/>
        <br/>
        <div class="row">
            <div class="col-lg-8 col-lg-offset-2">
              <h4>Tabla de predicciones</h4>
              <table id="predictions-table" class="table table-striped" border="2">
                <thead>
                  <tr class="next-days">
                    <td>{{ company_name }}</td>
                    {{#each data.days_span.next_quotation_days }}
                      <td>{{ this }}</td>
                    {{/each}}
                  </tr>
                </thead>
                <tbody>
                  <tr class="superior-ic">
                    <td>I.C. 95%</td>
                    {{#each data.prediction_table.superior_ic }}
                      <td>{{ this }}</td>
                    {{/each}}
                  </tr>
                  <tr class="prediction">
                    <td>PREDICCIÓN</td>
                    {{#each data.prediction_table.prediction }}
                      <td>{{ this }}</td>
                    {{/each}}
                  </tr>
                  <tr class="inferior-ic">
                    <td>I.C. 95%</td>
                    {{#each data.prediction_table.inferior_ic }}
                      <td>{{ this }}</td>
                    {{/each}}
                  </tr>
                </tbody>
              </table>
```

Figura 4.20: Ejemplo de plantilla

Fuente: Elaboración propia (Código JavaScript)

En la Figura 4.20 podemos ver el código HTML con el cual se muestra el resultado.

4.2 | Diseño

Uno de los objetivos del trabajo era el de modernizar el diseño. Para ello se ha realizado un proceso de renovación a varios niveles.

4.2.1 | Visual

En el caso de la pantalla principal. Podemos ver el diseño de la versión previa en la Figura 4.21.

Figura 4.21: Pagina principal (previa)
Fuente: Captura de pantalla ; Elaboración propia

Posteriormente, en la Figura 4.22 podemos ver el resultado final.

Figura 4.22: Pagina principal (posterior)
Fuente: Captura de pantalla ; Elaboración propia

En el caso de la página principal, el objetivo ha sido dotar al texto de una mayor importancia en la web, siendo este lo fundamental en esta página.

En la Figura 4.23 podemos ver el diseño previo del resultado de un modelo.

Figura 4.23: Página del modelo (previa)
Fuente: Captura de pantalla ; Elaboración propia

En la Figura 4.24 podemos ver el diseño modificado del resultado de un modelo.

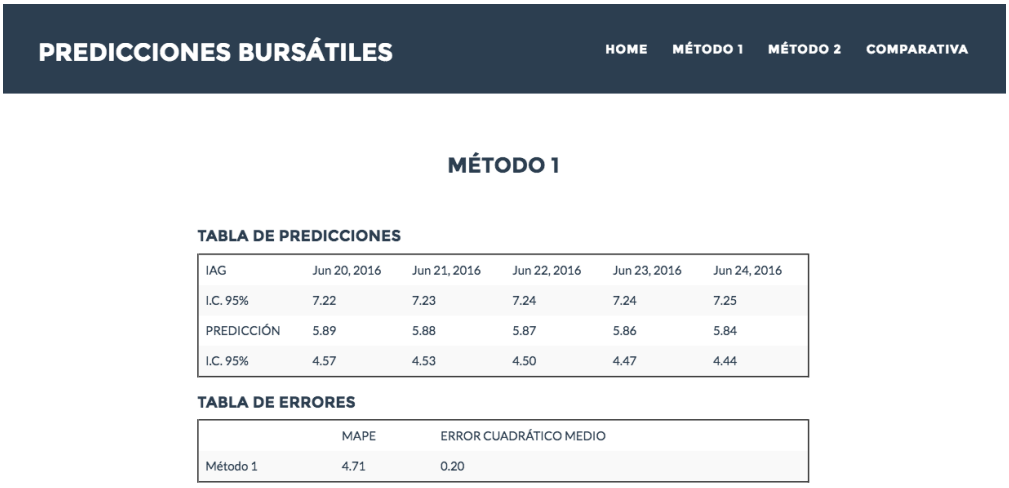

Figura 4.24: Página del modelo (posterior)
Fuente: Captura de pantalla ; Elaboración propia

A nivel del resumen del modelo el objetivo ha sido el de que se pueda ver la efectividad y el trabajo realizado mediante los modelos matemáticos descritos en el Capítulo 3 de un solo vistazo siendo el gráfico y las tablas, los ejes centrales de lo que el usuario ve.

4.2.2 | Adaptación a móvil

En los últimos años los *smartphones* se han vuelto populares y cada vez más personas los emplean para navegar por Internet. Hoy en día es esencial el soportar a dichos usuarios adaptando la web cuando un usuario entra desde un móvil.

En el caso de la pantalla principal para la versión móvil podemos ver el diseño en la Figura 4.25.

Figura 4.25: Página principal, vista móvil (previa)
Fuente: Captura de pantalla ; Elaboración propia

Una vez modificado podemos ver en la Figura 4.26 el diseño posterior con adaptación a móvil.

Figura 4.26: Página principal, vista móvil (posterior)
Fuente: Captura de pantalla ; Elaboración propia

En la Figura 4.27 podemos ver la versión móvil previa del resultado del modelo en versión móvil.

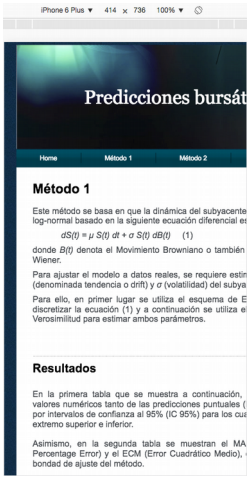

Figura 4.27: Página del modelo, vista móvil (previa)
Fuente: Captura de pantalla ; Elaboración propia

En la Figura 4.28 podemos ver la versión móvil posterior del modelo en versión móvil.

Figura 4.28: Página del modelo, vista móvil (posterior)
Fuente: Captura de pantalla ; Elaboración propia

4.3 | Regularidad de los procesos internos

Cada día (laborable) cambia la cotización del valor seleccionado, por ello deberemos de realizar el cálculo de los modelos cada día. Para ello emplearemos los denominados como *cronjobs* que es una forma de configurar el sistema informático para que realice una tarea. Podemos verlo en la Figura 4.29.

```
01 00 * * * python /home/cotizaccion/runner.py
```

Figura 4.29: Detalle del crontab
Fuente: Elaboración propia (Crontab)

Con este comando en el crontab (*crontab -e* para abrirlo) cada día a media noche se ejecutará el archivo *runner.py* que contiene el cálculo de ambos modelos (véase la Figura 4.30).

```
from lib.facades import *

symbol = 'IAG.MC'

MethodOne.run(symbol)
MethodTwo.run(symbol)
```

Figura 4.30: Detalle del runner.py
Fuente: Elaboración propia (Código Python)

4.4 | Despliegue

Finalmente, la aplicación ya está acabada. El último paso es el despliegue que consiste en poner la web en el servidor para que sea accesible públicamente.

Lo primero que haremos será subir el contenido de la web a un repositorio GIT. GIT es un control de versiones que permite mantener en múltiples ordenadores un control ordenado de los cambios que se producen sobre una serie de archivos.

Una vez tenemos el repositorio con el código podemos clonar ese repositorio en la carpeta habilitada para ese fin. Esto permite prescindir de otras formas de copiado de archivos como puede ser SCP o SFTP. Para hacer el clonado deberemos acceder al servidor por SSH.

Una vez que ya tenemos el contenido de la carpeta en el servidor podemos configurar Apache.

Apache es el servidor HTTP que permite que el contenido de la web sea accesible por vía web. Para ello configuramos Apache como en la Figura 4.31.

```
<VirtualHost *:80>
    ServerAdmin webmaster@localhost
        ServerName cotizaccion.imm.upv.es

    DocumentRoot /var/www/html/Ki/
    <Directory />
        Options FollowSymLinks
        AllowOverride None
    </Directory>
    <Directory /var/www/html/Ki/>
        Options Indexes FollowSymLinks MultiViews
        AllowOverride All
        Order allow,deny
        allow from all
    </Directory>

    ScriptAlias /cgi-bin/ /usr/lib/cgi-bin/
    <Directory "/usr/lib/cgi-bin">
        AllowOverride None
        Options +ExecCGI -MultiViews +SymLinksIfOwnerMatch
        Order allow,deny
        Allow from all
    </Directory>

    ErrorLog ${APACHE_LOG_DIR}/error.log

    # Possible values include: debug, info, notice, warn, error, crit,
    # alert, emerg.
    LogLevel warn

    CustomLog ${APACHE_LOG_DIR}/access.log combined

    Alias /doc/ "/usr/share/doc/"
    <Directory "/usr/share/doc/">
        Options Indexes MultiViews FollowSymLinks
        AllowOverride None
        Order deny,allow
        Deny from all
        Allow from 127.0.0.0/255.0.0.0 ::1/128
    </Directory>

</VirtualHost>
```

Figura 4.31: Configuración de Apache

Fuente: Elaboración propia

Una vez realizada dicha tarea, la web ya será accesible.

Capítulo 5 | Conclusiones del Trabajo Final de Grado

Después de realizar un esfuerzo activo durante meses para desarrollar una aplicación que nos facilite información financiera de un activo financiero podemos hablar de las conclusiones llegadas en el proceso del Trabajo Final de Grado.

En el primer capítulo nos adentramos en un análisis de la empresa. En él descubrimos las singularidades de la empresa a nivel general y a nivel sectorial.

Habiéndonos centrado en el desarrollo del software en el segundo capítulo, hemos habilitado la aplicación para recibir nuevos modelos sin suponer esto un perjuicio en la implementación de las mismas. Esto es ciertamente esencial para la continuación del proyecto.

Durante el presente documento hemos hablado del valor que genera el software al cliente que lo usa, bien, este cliente requiere una continua aportación de valor y para ello es necesario una plataforma que impulse ese valor.

A los dos modelos explicados e implementados en el tercer capítulo se les podrían sumar varios más, como puede ser el modelo Log-Normal con simulaciones de Monte Carlo o un proceso discontinuo tipo *Poisson*. El trabajo desarrollado en este Trabajo Final de Grado facilita el análisis cuantitativo de las cotizaciones de activos financieros mediante modelos matemáticos estocásticos, proporcionado diferentes técnicas de predicción puntuales y probabilísticas, lo que supone una herramienta más de utilidad para potenciales inversores.

Finalmente en el cuarto capítulo hemos dado un repaso a la ejecución de los cambios que ambicionábamos desde las perspectivas de las buenas prácticas de diseño y desarrollo de aplicaciones informáticas.

Bibliografía

Libros:

Gamma, E.; Helm, R.; Johnson, R; Vlissides, J (1994) Design Patterns: Elements of Reusable Object-Oriented Software

BAXTER, M.; REMIE, A. (2012) Financial Calculus: An Introduction to Derivate Pricing. Cambridge Univ. Press, 20th Edition.

Feathers, M. (2004) Working Effectively with Legacy Code

ITÔ, K. (1944) Stochastic Integral. Tokio. Proc. Imperial Acad.

ITÔ, K. (1961) Lectures on Stochastic Processes. Bombay. Tata Institute.

GONZÁLEZ DUQUE, R. (2014) Python para todos.

Lutz, M. (2003) Learning Python

ØKSENDAHL, B. (1980) Stochastic Differencial Equations. Nueva York. Springer.

ØKSENDAHL, B. (1998) Stochastic Differential Equations: An Introduction with Applications. Springer.

Morales, R. (2015) Gestión de Tareas con Kanban: Introducción a la gestión visual del trabajo

Artículos:

ITÔ, K. (1951) On stochastic differential equations. Memoirs, American Mathematical Society, fascículo 4, pp. 1–51.

MERTON, R.C. (1973) An inter temporal capital asset pricing model. Econometrica, fascículo 41 pp. 867-887.

Webs:

http://www.eleconomista.es (Última Fecha de Acceso: 11 de Abril 2016)

http://cotizaccion.imm.upv.es ; (Última Fecha de Acceso: 10 de Abril 2016)

http://www.scipy.org (Última Fecha de Acceso: 4 de Abril 2016)

http://www.iairgroup.com/phoenix.zhtml?c=240949&p=aboutoverview (Última Fecha de Acceso: 21 de Abril 2016)

http://www.iairgroup.com/phoenix.zhtml?c=240949&p=irol-irhome (Última Fecha de Acceso: 9 de Abril 2016)

http://www.ubuntu.com (Última Fecha de Acceso: 20 de Abril 2016)

https://www.stackoverflow.org (Última Fecha de Acceso: 10 de Abril 2016)

http://www.iairgroup.com/phoenix.zhtml?c=240949&p=irol-reportsannual (Última Fecha de Acceso: 14 de Marzo 2016)

http://www.yahoo.finance.com (Última Fecha de Acceso: 3 de Marzo 2016)

https://www.python.org (Última Fecha de Acceso: 2 de Marzo 2016)