

# 1. Introducción

## 1.1. Identificación do proxecto

**Título:** Diseñador gráfico de interfaces de usuario, Winforms.

**Código:**

**Autor:** Adolfo Álvarez López

**DNI:** 53199109V

**Curso:** 2013-2014

**Titor do TFG:** Baltasar García Pérez Schofield

**Cotitora:** Lourdes Borrajo Diz

**Área de Linguaxes e Sistemas Informáticos**

**Departamento de Informática**

**Universidade de Vigo**

## 1.2. Organización da documentación

- **Memoria:** Documento formal no que se detallan todas as etapas seguidas no proceso de desenvolvemento do sistema incluíndo os contratempos sufridos e outras dificultades, as cales quedarán reflexadas na comparativa entre representación temporal do traballo e a planificación inicial.
- **Manual técnico:** Documento formal no cal se incluírá toda a información técnica do proxecto para facilitar as futuras labores de mantemento do sistema
- **Manual de usuario:** Documento informal no cal se fará unha explicación ó usuario de cómo instalar e utilizar a ferramenta entregada.

## 1.3. Marco da aplicación

Na actualidade o uso dos sistemas operativos de microsoft (desde windows XP ata Windows 8) e superior o 80% polo que non é de extrañar que se pretenda desenvolver software para esta plataforma.

O desenvolvemento de aplicacións para este sistema operativo baséase no Framework .NET, éste inclúe as linguaxes de programación c# y VisualBasic, el Common Language Runtime e unha gran biblioteca de clases. O equivalente a .NET de código aberto é Mono, o cal

permite desenvolver aplicacións para .NET noutras plataformas.

Nestes frameworks existen diversas APIS para a creación de interfaces gráficas, pero teñen en comun a API Winforms. Esta foi creada para .NET pero co paso dos anos Mono foi capaz de emulala completamente. Winforms permite crear interfaces mediante instrucións de código no interior do código dun proxecto. Pero ésta API ten dous problemas, o primeiro é a dificultade que existe para separar a parte de lóxica dunha aplicación da parte da vista que correspondería ó uso dos Winforms, e o segundo é que é difícil que pode ser nalgúns casos obter a interfaz tal é como a deseñamos. En interfaces un pouco complexas que se compoñene de diversos elementos diferentes, posicionar e dimensionalos da maneira que pensáramos pode ser difícil mediante as instrucións proporcionadas pola API, basadas todas en valores numéricos e coordenadas, o que fai que se teña que probar varias veces o código correspondente á interfaz para asegurarse de que se mostra tal e como se planea.

Debido a estas dificultades creemos que se vota de menos una ferramenta visual que permita deseñar, crear, posicionar e dimensionar os elementos que compoñen a nosa interfaz sen necesidade de ter que xogar con diferentes combinacións de instrucións, valores e coordenadas directamente sobre o código, se non que te permita ir vendo como a construes pouco a pouco hasta obter o que realmente queres sen precisar de moito tempo.

Visual Studio, un IDE desenvolto por Microsoft, para programar nos linguaxes soportados por .NET, conta con unha ferramenta que cubre esta necesidade. Pero presenta un problema importante, tan só está disponible para Windows, non é libre e precisa de licencia. En Mono tamén se propon o uso dunha ferramenta que si que é libre e esta disponible para linux pero tras probala nos damos de conta que o engadir poucos elementos ou en certos ordenes a aplicación deixa de funcionar, explicaión para esto podería ser que non se actualiza dende xa fai tres anos.

Neste proxecto propónse a creación dunha aplicación multiplataforma (Windows e GNU/Linux), visual e de uso sinxelo que permita deseñar e construír interfaces gráficas da API Winforms e exportarlas para logo podelas usar noutro proxecto de desenvolvemento.

## **2. Obxetivos da aplicación**

A finalidade principal deste proxecto é a creacion dunha aplicación libre para o deseño de

interfaces de usuario para proxectos basados en .NET ou Mono.

A aplicación está orientada para ser utilizada por desenvolvedores e que lles permita, dunha forma rápida e sinxela xerar interfaces gráficas para podelas usar nos seus proxectos. Por isto buscarase que a aplicación teña unha interfaz sinxela e directa para que o seu uso sexa o mais natural posible.

A interfaz comporase de tres áreas:

- Un treeview lateral no que se poderá observar de forma esquemática o conxunto de elementos que compoñen a interfaz que se está a desenvolver.
- Un datagridview no outro lateral, onde estará o listado dos atributos dos elementos engadidos, que se poderán modificar para darlle forma os compoñentes da interfaz.
- Unha zona de traballo central onde poderase observar como se vai construindo a interfaz en desenvolvemento.

Ésta ferramenta permitirá crear interfaces gráficas sinxelas cós elementos mais utilizados da API Winforms. Entre eles destacamos os seguintes:

- Paneles contenedores:
  - Border: Panel basicos da API Winforms
  - Grid: TableLayoutPanel da API Winforms
  - HBox e VBox: Casos especiais de TableLayoutPanel. No primeiro os elementos que se lle engadan o farán horizontalmente, mentres que no segundo caso será verticalmente.
  - TabControl, GroupBox.
- Menús:
  - MenuStrip
  - ToolBar
- Controles:
  - Button, Label, TextBox, PictureBox, DateTimePicker, MonthCalendar, RadioButton, CheckBox, ProgressBar, Splitter
  - Combobox, TreeView, StatusBar
  - DataGridView, ListView

Aparte de poder deseñar os controles, paneles e menús soportados, tamén se permitirá crear

ós eventos mais importantes para estes elementos. Éstes serían:

- Click, DoubleClick
- Enter, Leave, GotFocus, LostFocus
- KeyDown, KeyPress, KeyUp
- MouseClick, MouseDoubleClick, MouseWheel, MouseDown, MouseUp, MouseEnter, MouseLeave, MouseHover
- Resize

Ademais, aparte da ferramenta que permitirá o deseño e exportación das interfaces, preténdese crear unha biblioteca que permita importar estas interfaces no proxecto no que se precisen utilizar, e así mediante unha simple función, poder utilizar a interfaz aforrando todo o código correspondente á vista da aplicación.

### 3. Descripción técnica da aplicación

O sistema desenvolvido deberá permitir a creacion de interfaces gráficas da a API Winforms de .Net e Mono, dando a posibilidade de exportalas e importalas noutros proxectos en desenvolvemento. A API Winforms usáse en Visual Basic e C# e será neste ultimo no que se desenvolverá o noso sistema.

O sistema comporase de duas partes:

- **Ferramenta de deseño e creación de interfaces:** Codificada en C# e con ela poderase crear gráficamente a interfaz que se desexe.
- **Biblioteca .dll:** Con esta biblioteca engadida como referencia nun proxecto C# poderase importar e utilizar a interfaz deseñada coa ferramenta de deseño.

Na figura 1 móstrase a arquitectura do sistema proposto. A ferramenta xeradora de Winforms terá os propios Winforms formando a capa de presentación da aplicación. Como xa se mencionou a lóxica completa será creada en C# e por último a persistencia da aplicación será en ficheiros Xml. Éstes ultimos ficheiros son os que se poderán importar nunha aplicación en construción mediante o uso da biblioteca que tamén se aportará, e que creará a interfaz en Winforms constituindo a capa de presentación da nova aplicación, tal e como se indica no bloque da dereita da Figura 1.

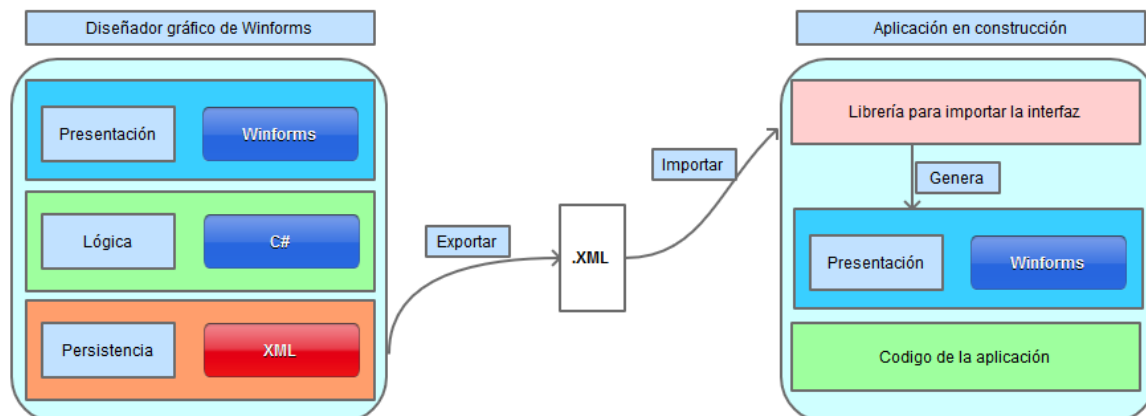


Figura 1. Estructura do sistema xerador de Winforms

A vista da aplicación é desenvolvida coa API Winforms e non con WPF ou outra API de interfaz para que a aplicación poida ser utilizada en sistemas Windows e GNU/Linux sen problemas. O mesmo ocorre coa persistencia que se fará en Xml e non en Xaml, debido a que este último non está soportado por Mono actualmente.

### 3.1. Winforms vs Windows Presentation Foundation vs GTK#

Cando se comezou a deseñar o sistema o primeiro que se plantexou foi con qué tecnoloxía se implementará a interfaz do sistema. Inmediatamente apareceron tres opcións: Winforms, WPF e GTK#.

A principal característica que diferencia estas dúas APIs é a súa relación co resto do código da aplicación. Mentres que en Winforms e GTK# o deseño da interfaz forma parte da lóxica da aplicación en C#, WPF permite separar completamente a capa de presentación da lóxica do sistema permitindo definir as interfaces de usuario en ficheiros Xaml. Esta opción permitiría crear un código moito mellor estruturado pero por desgracia Mono non admite este tipo de interfaz e unha das principais características do sistema é que poderá ser utilizado tanto en Windows .Net como en GNU/Linux Mono polo que queda totalmente descartado.

Nos encontramos ante a decisión de se utilizar GTK# o Winforms para a nosa aplicación. As dúas opcións a nivel de programación son moi similares pero finalmente utilizarase Winforms. A explicación desta decisión baséase simplemente en que xa que o sistema está

destinado a deseñar interfaces coa API Winforms, utilizar esta mesma na propia aplicación fará que o deseño e visualización das interfaces en creación sexa mais fiable e ademais moito mais sinxelo de desenvolver, e o non haber unha razón importante para utilizar GTK#, Winforms é sen dúbida a mellor opción.

### **3.2. XML vs XAML**

En canto á persistencia do sistema tamén nos encontramos cunha decisión en canto ás tecnoloxías a utilizar, Xml ou Xaml. Éste último foi creado para a especificación das interfaces de usuario en WPF. Xaml é un linguaxe declarativo baseado en Xml e optimizada para describir interfaces de usuario. Xaml representa directamente a creación de instancias de obxectos nun conxunto concreto de tipos de respaldo definidos nos ensamblados. Esta tecnoloxía sería perfecta para representar os elementos da interfaz creada coa nosa aplicación, pero de novo temos un problema de compatibilidade, Mono non soporta serialización en Xaml polo que nos vemos obrigados a utilizar Xml normal. Pero non todo é malo, en Mono e .Net existen as clases XmlReader e XmlWriter que permiten serializar obxectos de C# en ficheiros Xml permitindo polo tanto gardar instancias deses obxectos facendo a diferenza entre Xml e Xaml se reduce a temas de rendemento.

### **3.3. NUnit**

Para probar as distintas clases que compoñen o sistema desenvolvido utilizouse o framework NUnit e creáronse unha serie de probas de unidade. NUnit foi portado inicialmente de JUnit. Está escrito totalmente en C# e ha sido completamente rediseñado para aproveitar moitas das características dos linguaxes de .NET.

### **3.4. Git**

Git é un software de control de versións creado pensando na eficiencia e confiabilidade do mantemento de versións de aplicacións cando estas teñen un gran número de arquivos de código fonte. Na miña situación pensei que sería útil utilizar un sistema deste tipo xa que a metodoloxía de desenvolvemento que se utilizou, e da que se falará na seguinte sección, é unha metodoloxía baseada en prototipos e por tanto podería ser útil poder volver a unha versión anterior e estable do prototipo en desenvolvemento. Ademais utilizouse GitHub para aloxar o repositorio externamente o que facilitou poder compartilo co tutor do

proxecto, e o fixo disponible para poder traballar nel dende diferentes equipos e sistemas dunha forma cómoda.

## **4. Metodoloxías utilizadas**

Unha metodoloxía de desenvolvemento de software refírese a un framework que é usado para estruturar, planear e controlar o proceso de desenvolvemento nos sistemas de información.

Ó longo do tempo foron surxindo moitos métodos diferentes cada un con puntos fortes e débiles. Cada unha destas metodoloxías teñen o seu propio enfoque para o desenvolvemento de software. Entre os distintos enfoques destacamos catro como os mais xerais.

Primeriro o modelo en cascada, este consiste nun proceso secuencial de desenvolvemento no que os pasos do desenvolvemento son vistos cara abaixo a través das fases de análise das necesidades, o deseño, implementación, probas, a integración e o mantemento.

Outro enfoque destacado é o prototipado. O prototipado consiste en desenvolver modelos de aplicacións software que permiten visualizar a funcionalidade básica da mesma, sen precisar incluír toda a lóxica ou todas as características do modelo finalizado. O prototipado permite ó cliente avaliar de maneira temprana o produto e interactuar cos deseñadores e desenvolvedores para saber se se está a cumprir coas expectativas e as funcionalidades acordadas.

O enfoque incremental é unha mezcla dos dous enfoques anteriores. Este enfoque provee dunha estratexia para controlar a complexidade e os riscos, desenvolvendo unha parte de produto software reservando o resto de aspectos para o futuro. Conta cunha serie de pequenas cascadas, onde todas as fases da cascada se completan para unha pequena parte dos sistemas antes de proceder co próximo incremento.

Por último o enfoque de desenvolvemento en espiral, nel a atención céntrase na avaliación e redución de riscos do proxecto dividindo o proxecto en segmentos máis pequenos e proporcionar máis fiabilidade de cambio durante o proceso de desenvolvemento, así como ofrecer a oportunidade de avaliar os riscos co peso da consideración da continuación do proxecto durante todo o ciclo de vida. Cada viaxe ó redor da espiral atravesa catro fases, determinar obxectivos, alternativas e desencadenantes da iteración; avaliar as alternativas,

identificar e resolver resgos; desenvolver e verificar os resultados da iteración, e plan da próxima iteración.

Para desenvolver o noso sistema optaremos pola utilización dun **modelo de desenvolvemento baseado en prototipos**.

O modelo de baseado en prototipos é un modelo de desenvolvemento evolutivo que comeza coa definición dos obxetivos globais para o desenvolvemento do proxecto, posteriormente se identifican os requisitos coñecidos e as áreas do esquema. Tras isto plantexase con rapidez unha iteración de construción de prototipos e se presenta o modelado. O resultado desta iteración é un prototipo que se presentará ós clientes para que o evalúen e proporcionen información sobre os defectos do prototipo para poderlos correxir en prototipos futuros.

Con esta metodoloxía lógrase un mellor entendemento dos obxetivos e características da aplicación así como a posibilidade de refinar pouco a pouco o conxunto da aplicación para ó final obter como resultado o sistema desexado.

Existen dous tipos de prototipos, o prototipo desechable e o prototipo evolutivo. O primeiro é utilizado unicamente para desenvolver unha parte do sistema pouco clara para probala e eliminar dúbidas acerca dela. Tras isto o prototipo se descarta e esa funcionalidade se engade ao sistema completo ou a outro prototipo. E o prototipo evolutivo é un modelo parcialmente construído e que é mellorado en cada fase de prototipado engadíndolle máis funcionalidades, obtendo finalmente un prototipo que se corresponde ó sistema final e este prototipo deixa de serlo para ser o produto a entregar.

O desenvolvemento coa metodoloxía baseada en prototipos consta das seguintes fases:

- Investigación preliminar: Nela defínese o problema, os obxetivos xerais do sistema e un deseño preliminar e rápido do sistema completo.
- Fase de prototipado: Nela desenvólense os prototipos que se irán presentando ó cliente e irán refinándose ata obter o sistema final. Esta fase compoñese das seguintes subfases cíclicas:
  - Deseño rápido: Nesta fase defínense as funcionalidades que se van a implementar no prototipo e faise un deseño deste.
  - Construción do prototipo: Implementase o prototipo definido na fase anterior e



fanse as probas necesarias.

- Evaluación: Preséntase o prototipo ó cliente para que sexa evaluado e poder recibir indicaciones para mellorar o sistema para á próxima iteración.
- Desarrollo do produto: Creación do sistema final produto da evolución do sistema mediante a fase de prototipado.

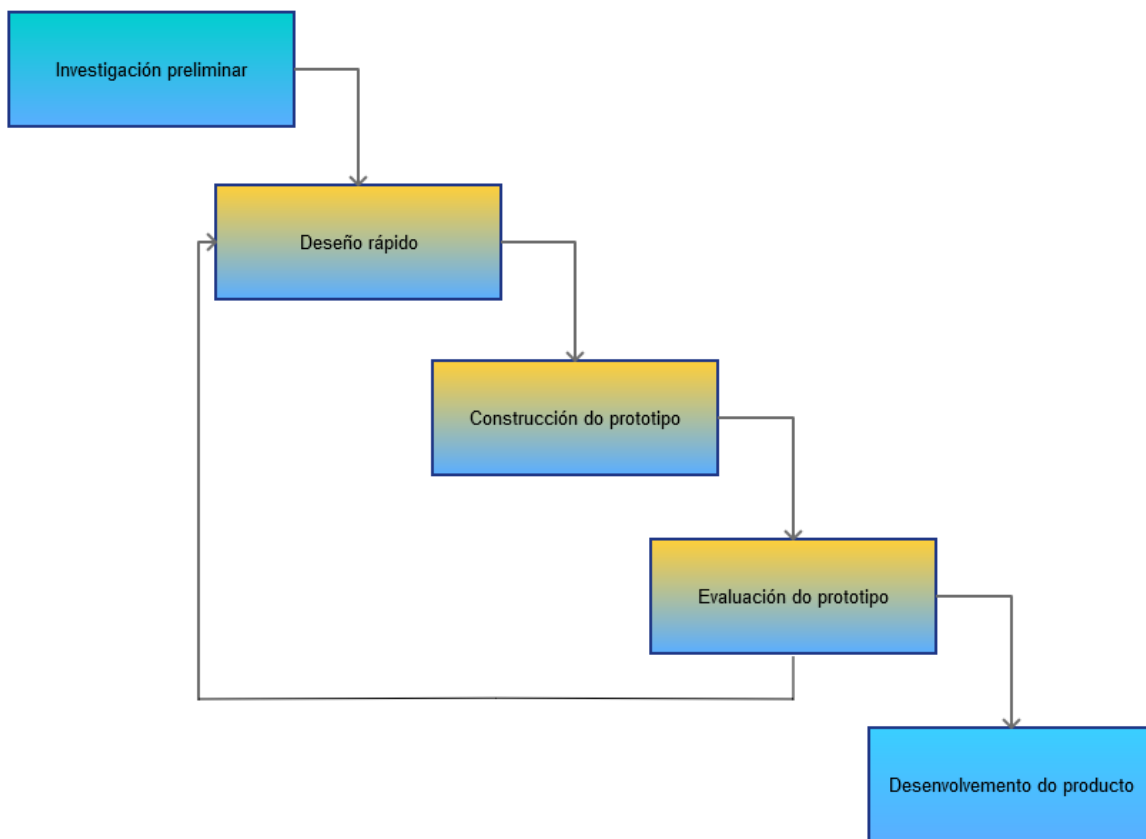


Figura 2. Modelo de Prototipos.

## 5. Planificación e presuposto

### 5.1. Planificación

Nesta sección móstrase unha comparativa entre a planificación estimada e a temporización final para este proxecto.




A planificación inicial do proxecto resúmese na Táboa 1. Nesta táboa pódese observar que a duración estimada do proxecto era de 300 horas, repartidas en 10 semana de traballo. Estes

cálculos estaban realizados en base a unha dedicación prevista de 5 días á semana, traballando 6 horas ó día. Para realizar a planificación inicial do proxecto decidiuse establecer unha duración de 2 semanas para cada un dos 3 prototipos previstos.

Dedicación semanal prevista (en horas/semanas): 30	
<b>Fase</b>	<b>Estimación temporal (en semanas)</b>
<b>Captura de requisitos xerais do sistema</b>	<b>1</b>
<b>Prototipo 1</b>	<b>2</b>
<b>Prototipo 2</b>	<b>2</b>
<b>Prototipo 3</b>	<b>2</b>
<b>Construcción final do sistema</b>	<b>1</b>
<b>Documentación</b>	<b>10</b>
<b>TOTAL PROXECTO</b>	<b>10</b>

*Táboa 1. Resumo da planificación estimada*

A Figura 3 representa a duración e as datas de planificación estimada. Como pode observarse, a data estimada de comezo foi o 07/04/2014 e a data estimada de finalización era o 20/06/2014.

		Nombre	Duración	Inicio	Fin
1		Captura de requisitos generales del sistema	1s	07/04/2014	11/04/2014
2		☐ <b>Prototipo 1</b>	2s	21/04/2014	02/05/2014
3		Definición de requisitos del prototipo	1d	21/04/2014	21/04/2014
4		Diseño del prototipo	2d	22/04/2014	23/04/2014
5		Construcción del prototipo	1s	24/04/2014	30/04/2014
6		Evaluación del resultado	2d	01/05/2014	02/05/2014
7		☐ <b>Prototipo 2</b>	2s	05/05/2014	16/05/2014
8		Definición de requisitos del prototipo	1d	05/05/2014	05/05/2014
9		Diseño del prototipo	2d	06/05/2014	07/05/2014
10		Construcción del prototipo	1s	08/05/2014	14/05/2014
11		Evaluación del resultado	2d	15/05/2014	16/05/2014
12		☐ <b>Prototipo 3</b>	2s	19/05/2014	30/05/2014
13		Definición de requisitos del prototipo	1d	19/05/2014	19/05/2014
14		Diseño del prototipo	2d	20/05/2014	21/05/2014
15		Construcción del prototipo	1s	22/05/2014	28/05/2014
16		Evaluación del resultado	2d	29/05/2014	30/05/2014
17		Construcción del sistema final	1s	02/06/2014	06/06/2014
18		Documentación	10s	07/04/2014	20/06/2014

*Figura 3. Planificación estimada*

Por último, na Figura 4 amósase o diagrama de Gantt correspondente á planificación estimada, xunto coas fases do proxecto.

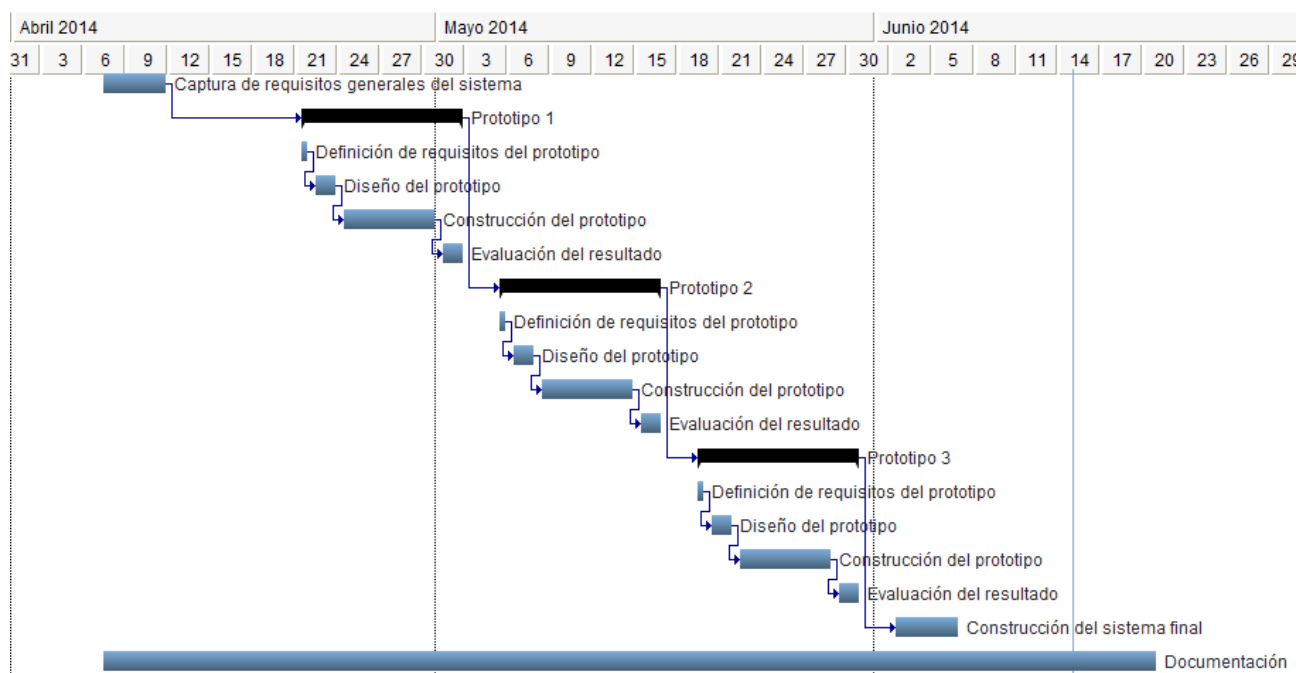


Figura 4. Diagrama de Gantt da planificación estimada

A variación máis importante que sufriu a planificación inicial foi na construción do terceiro prototipo. Éste tivo máis traballo que os outros dous prototipos e a súa construción durou unha semana máis, e como consecuencia, aumentou o número de horas dedicadas para o proxecto e o retraso da finalización do mesmo. Esta variación pódese observar na Táboa 2 e na Figura 5 e Figura 6 correspondentes ó diagrama de Gantt resultado da temporización real.

Dedicación semanal prevista (en horas/semanas): 30	
Fase	Estimación temporal (en semanas)
<b>Captura de requisitos xerais do sistema</b>	<b>1</b>
<b>Prototipo 1</b>	<b>2</b>
<b>Prototipo 2</b>	<b>2</b>
<b>Prototipo 3</b>	<b>3</b>
<b>Construción final do sistema</b>	<b>1</b>
<b>Documentación</b>	<b>10,4</b>
<b>TOTAL PROXECTO</b>	<b>10,4</b>

Táboa 2. Resumo da temporización final do proxecto

		Nombre	Duración	Inicio	Fin	Predecesoras
1		Captura de requisitos generales del sistema	1s	07/04/2014	11/04/2014	
2		▢ Prototipo 1	2s	21/04/2014	02/05/2014	1
3		Definición de requisitos del prototipo	1d	21/04/2014	21/04/2014	
4		Diseño del prototipo	2d	22/04/2014	23/04/2014	3
5		Construcción del prototipo	1s	24/04/2014	30/04/2014	4
6		Evaluación del resultado	2d	01/05/2014	02/05/2014	5
7		▢ Prototipo 2	2s	05/05/2014	16/05/2014	2
8		Definición de requisitos del prototipo	1d	05/05/2014	05/05/2014	
9		Diseño del prototipo	2d	06/05/2014	07/05/2014	8
10		Construcción del prototipo	1s	08/05/2014	14/05/2014	9
11		Evaluación del resultado	2d	15/05/2014	16/05/2014	10
12		▢ Prototipo 3	3s	19/05/2014	06/06/2014	7
13		Definición de requisitos del prototipo	1d	19/05/2014	19/05/2014	
14		Diseño del prototipo	2d	20/05/2014	21/05/2014	13
15		Construcción del prototipo	2s	22/05/2014	04/06/2014	14
16		Evaluación del resultado	2d	05/06/2014	06/06/2014	15
17		Construcción del sistema final	1s	09/06/2014	13/06/2014	12
18		Documentación	10.4s	07/04/2014	24/06/2014	

Figura 5. Temporización final do proxecto

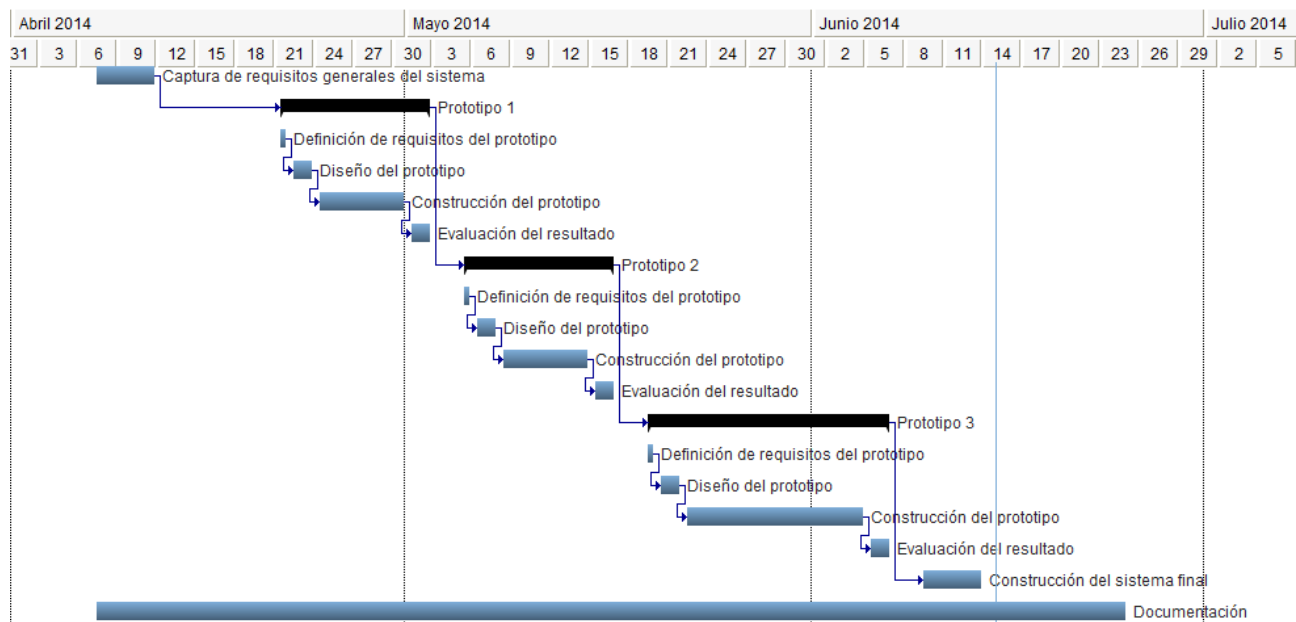


Figura 6. Diagrama de Gantt da temporización real

## 5.2. Presuposto

En canto ós custos asociados ó hardware, software e recursos humanos, amósase nas

seguintes seccións táboas a modo de resumo.

### 5.2.1 Custo do hardware

Na Táboa 3 detállase o hardware empregado para levar a cabo este proxecto e o seu custo.

Compoñente	Modelo	Custo
Portátil	Hp pavilion dv6 7003 ss	900 €

*Táboa 3. Custo do hardware*

Para calcular o custo real dos materiais utilizados, xa que o hardware descrito nonvai estar dedicado unicamente para este proxecto, fixéronse ás seguintes consideracións;

- Considerándose un uso posible do portatil de 5 días á semana durante ós 12 meses do ano e cunha xornada de 8 horas o que fai un numero de 2080 horas ó ano.
- Fíxase o tempo de vida util do portatil en 4 anos, e dicir, 8320 horas.

	Vida útil	Custo total	Custo por horas	Horas de uso	Total amortizado
Portatil	4 anos (8320 horas)	900,00 €	0,109 €	312	34,01 €

*Táboa 4. Amortizacion do proxecto*

### 5.2.2. Custo do software

Na Táboa 5 indicase o custo do software empregado.

Software	Custo
Linux mint 16 Petra	0 €
Monodevelop	0 €
Xamarin Studio	0 €
GitHub	0 €
Pencil	0 €
Ganttter.com	0 €
Visal Paradigm (Licenza aportada pola Universidade de Vigo)	0 €
Libre Office	0 €
<b>Total</b>	<b>0 €</b>

*Táboa 5. Custo do sobftware empregado*

Debido a que todo o software é de licenza libre ou gratuita, o custo total foi de 0 € e non é preciso indicar á amortización do mesmo.

### 5.2.3 Custo do persoal do proxecto

Para o cálculo do custo do persoal considerouse un prezo da hora de traballo de 20€ para unha duración total do proxecto de 312 horas.

Perfil	Horas	Prezo/Hora	Custo total
Analista/Programador	312 h	20 €/h	6240 €

Táboa 6. Custo do persoal

### 5.2.4 Custo total do proxecto

Tendo en conta os custos de hardware, software e de persoal, na Táboa 7 indícase o cálculo total dos custos do proxecto.

Concepto	Custo
Hardware	34,01 €
Software	0 €
Persoal	6240 €
<b>TOTAL</b>	<b>6274,01 €</b>

Táboa 7. Custos totais do proxecto

## 6. Problemas atopados e solucións aportadas

Un dos principais problemas atopados durante o desenvolvemento do sistema tivo orixe no obxectivo de facer un sistema multiplataforma e sinxelo. Aínda que Mono é a alternativa libre e para sistemas GNU/Linux de .Net existen moitas diferenzas de comportamento. Houbo obxectos da interfaz que nun sistema funcionaban dunha forma e no outro doutra, como por exemplo a hora de rechear un DataGridView, en windows tan só permite rechear as celdas con Strings e en Linux dá a posibilidades de utilizar obxectos. Finalmente descubrín que o máis restrictivo é windows e facendoo de forma que funcione ben ahi en Linux con Mono funciona correctamente tamen.

Con respecto ó mesmo tema houbo outro problema que se debe mencionar. Inicailemte a persistencia da ferramenta e a exportación das interfaces creadas con ela quería facerse co

formato Xaml pero debido a que este non está soportado por Mono finalmente houbo que decidir utilizare serialización con Xml.

Outro inconveniente sufrido foi ao principio da implementación decidir como crear a previsualización das interfaces en deseño. Non se sabia se se deberían utilizar imaxes ou outro sistema. Finalmente o mais sinxelo, práctico e efectivo foi renderizar directamente a interfaz coma se fora parte da interfaz do sistema.

## 7. Futuras ampliacións

Neste proxecto centrámonos na posibilidade de poder crear interfaces gráficas de usuario sinxelas, cos elementos máis coñecidos e utilizados da API Winforms, pero aínda quedan controles que non se incluíron e que, nun futuro, podería ser interesante incluír ata o punto de dar soporte a todos os elementos da API. Ademais da cantidade de elementos da API que non se soportan, nos elementos que si se incluíron non é posible personalizalos tanto coma se podería facer a través de código sen utilizar a aplicación, existen tamen moitas propiedades que non se lles dá a posibilidade de modificar. O mesmo ocorre cos eventos, cada elemento da API Winforms ten unha gran cantidade de eventos dos cales só os máis importantes e comúns están habilitados para a súa edición a través da ferramenta.

Outra ampliación que se podería contemplar é dar a posibilidade de crear e editar múltiples formularios de forma simultánea e así poder rapidamente e de forma cómoda construír a totalidade das interfaces da aplicación na que as queremos importar.

## 8. Bibliografía

- [1] – <http://msdn.microsoft.com/>. Documentación das librerías de C#
- [2] – <http://www.nunit.org/>. Documentación do framework de probas NUnit.
- [3] – <http://git-scm.com/>. Documentación acerca do sistema de control de versións Git.
- [4] -