UNIVERSIDADE DE SANTIAGO DE COMPOSTELA

ESCOLA TÉCNICA SUPERIOR DE ENXEÑARÍA

Deseño e implementación dun software flexible para o xogo de rol en liña baseado en texto

Autor:
Martín Coego Pérez

Directores:
Paulo Félix Lamas
Tomás Teijeiro Campo

Grao en Enxeñaría Informática

Febreiro 2017

Traballo de Fin de Grao presentado na Escola Técnica Superior de Enxeñaría da Universidade de Santiago de Compostela para a obtención do Grao en Enxeñaría Informática

D. Paulo Félix Lamas, Profesor do Departamento de Electrónica e Computación da Universidade de Santiago de Compostela, e **D. Tomás Teijeiro Campo**, Profesor do Departamento de Electrónica e Computación da Universidade de Santiago de Compostela,

INFORMAN:

Que a presente memoria, titulada *Deseño e implementación dun software flexible para o xogo de rol en liña baseado en texto*, presentada por **D. Martín Coego Pérez** para superar os créditos correspondentes ao Traballo de Fin de Grao da titulación de Grao en Enxeñaría Informática, realizouse baixo a nosa dirección no Departamento de Electrónica e Computación da Universidade de Santiago de Compostela.

E para que así conste aos efectos oportunos, expiden o presente informe en Santiago de Compostela, a (Data):

O director, O codirector, O alumno,

Paulo Félix Lamas Tomás Teijeiro Campo Martín Coego Pérez

Agradecementos

Se se quere pór algún agradecemento, este vai aquí.

Resumo

Se se quere pór resumo, este vai aquí.

Índice xeral

Índice de figuras

Índice de cadros

Capítulo 1

Introdución

Introdución do documento

1.1. Obxectivos xerais

1.2. Documentación

1.3. Descrición do sistema

1.4. Información adicional de interese

Capítulo 2

Planificación e presupostos

Planificación e presupostos: debe incluír a estimación do costo (presuposto) e dos recursos necesarios para efectuar a implantación do Traballo, xunto coa planificación temporal do mesmo e a división en fases e tarefas. Recoméndase diferenciar os costos relativos a persoal dos relativos a outros gastos como instalacións e equipos.

Capítulo 3

Especificación de requisitos

Antes de comezar a desenvolver o software deberase facer unha especificación dos requisitos que este debe cumprir. Neste caso o primeiro que faremos será definir casos de uso do sistema, para posteriormente extraer a partir deles unha serie de requisitos funcionais e non funcionais.

3.1. Definicións

Nos casos de uso e requisitos faremos referencia a estas definicións, polo que é necesario explicalas aquí con precisión.

Partida

Instancia concreta deste software xestionada por un administrador. Cada partida debe ter unha base de datos propia e un directorio cos ficheiros que compoñen o mundo. Non se contempla a comunicación entre partidas distintas.

Mundo

O mundo de xogo é a abstracción de todos os obxectos e clases correspondentes a unha partida. Pódese distinguir entre a definición do mundo e o seu estado:

- Definición: O conxunto das clases descritas polos directores de xogo.
- Estado: O conxunto dos obxectos instanciados a partir das clases.

Xogador

Usuario que ten control dun personaxe e toma decisións en base a este. Os xogadores non poden modificar o mundo directamente, senón que as súas interaccións co mundo realízanse a través dos personaxes.

Director de xogo (DX)

Usuario con privilexios que pode alterar o mundo de forma directa, e resolver as decisións dos xogadores. Poden tanto alterar a definición do mundo (creando e modificando clases) como cambiar o estado do mesmo (instanciando obxectos e modificándoos). Os directores de xogo non teñen personaxe propio.

Clase

No ámbito do mundo de xogo, abstracción dun conxunto de obxectos con atributos e métodos comúns, seguindo o paradigma da programación orientada a obxectos. Os directores de xogo poden crear e modificar estas clases. Un exemplo de clase sería *espada*, *porta*, *cabalo* ou *guerreiro*.

Obxecto

No ámbito do mundo de xogo, cada unha das instancias dunha clase concreta. Os obxectos *existen* no mundo, e sitúanse nunha determinada estancia. É labor dos directores de xogo crealos, modificalos e destruílos. Todos os obxectos posúen un identificador global dentro do mundo. Un exemplo de obxecto sería *a segunda mesa da posada*, *o personaxe dun dos xogadores* ou *o can que acompaña ao grupo de personaxes*.

Personaxe

Obxecto especial que representa a un xogador no mundo. Toda a información percibida polo personaxe será posta en coñecemento do xogador correspondente, e toda decisión tomada polo xogador determinará as accións do personaxe.

Decisión

Descrición verbal dun xogador explicando a acción que pretende que realice o seu personaxe. As decisións están redactadas en linguaxe informal, pero con suficiente información como para que o director de xogo poda entender o que o xogador pretende. É labor do director de xogo xerar as accións a partir das decisións dos xogadores. Un exemplo de decisión sería: *"O meu personaxe achégase á espada incrustada na pedra, di en voz alta 'alá vou' e trata de arrincala."*

Acción

Conxunto de liñas de código que alterarán o mundo de xogo en función dunha decisión dun xogador. A acción é redactada por un director de xogo segundo o que interprete lendo a decisión correspondente.

Suceso

Conxunto de liñas de código que alterarán o mundo de xogo pero que non depende da decisión de ningún xogador. Os sucesos son provocados polos directores de xogo cando precisen alterar algo do mundo sen ser consecuencia directa das accións dos xogadores.

Rolda

No mundo de xogo, en cada estancia as accións se resolven divididas por roldas. Por cada rolda, cada personaxe pode facer como máximo unha acción. Estas roldas representan simbolicamente o paso do tempo dentro do mundo, aínda que o avance do tempo pode non coincidir exactamente en todas as estancias por igual.

Rexistro de personaxe

Todo o que perciba un personaxe determinado será almacenado nun rexistro. O xogador ten acceso a este rexistro, polo que poderá consultar en todo momento todo o que o seu personaxe puido oír, ver ou percibir de calquera xeito.

Estancia

Cada un dos recintos pechados nos que se divide o mundo de xogo. Tamén coñecidas como *habitacións*. Todos os obxectos están situados nunha estancia, incluídos os personaxes, e en calquera momento poden cambiar de estancia. As estancias pódense agrupar en *rexións* mediante nomes compostos. Exemplos de estancias poderían ser *Capital/mercado*, *montañas/mina abandonada/terceira sección* ou *castelo antigo/segunda planta/habitación 1*.

Tirada

Resultado aleatorio que representa unha tirada de dados. A tirada especifica o número de dados tirados e o número de caras de cada dado. Este resultado emprégase xeralmente para determinar o éxito ou o fracaso das accións dos personaxes ou doutros obxectos.

Recipiente

Calquera obxecto coa capacidade de almacenar outros obxectos. Estes últimos estarán vinculados ao obxecto recipiente, por tanto atoparanse sempre na mesma estancia. Un obxecto pode ser recipiente de varias listas distintas. Un exemplo de recipiente é *un cofre do tesouro*, *un barril* ou un *xogador co seu inventario*. Un exemplo de recipiente con varias listas pode ser *un xogador no que se diferencien os obxectos que leva nas mans dos que leva nas costas*.

3.2. Casos de uso

Mediante os casos de uso podemos representar situacións típicas do noso software, as cales suporán un bo xeito de extraer requisitos funcionais.

3.2.1. Descrición de actores

Inicialmente teremos que definir os actores que se relacionan co sistema nos nosos casos de uso. Identificamos tres actores diferenciados: director de xogo, xogador e administrador.

- **Director de xogo:** Este actor correspóndese coa definición descrita previamente; o seu cometido por tanto será dirixir a partida, recibindo decisións dos xogadores e transcribíndoas como accións dos seus personaxes.

- **Xogador:** Calquera usuario que participe na partida cun personaxe. Redacta decisións e envíaas ao DX.

- **Administrador:** Usuario especial que crea a partida e se encarga de administrala. As súas principais funcións será a asignación do rango de DX a outros usuarios, e a configuración básica da partida.

3.2.2. Descrición de casos de uso

A continuación atopamos os seguintes casos de uso, elaborados a partir de entrevistas con usuarios potenciais. Son posibles situacións que se poden dar durante a execución do software. Por simplificar daremos por feito que todos os casos de uso requerirán ao usuario en cuestión estar autenticado (salvo o caso de uso CU-01).

CU-01: Alta de xogadores

Actores Xogador

Escenario principal O caso de uso comeza cando un usuario anónimo se rexistra no sistema. Para isto emprega a opción *crear novo usuario*, onde introducirá os datos básicos da súa conta. Por defecto, o seu rol no sistema será o de xogador.

CU-02: Xestión de usuarios

Actores Administrador

Escenario principal O caso de uso comeza cando o administrador desexa asignar un usuario como DX. Neste caso, abre a sección de *xestión de usuarios* e escribe o nome do usuario cuxos permisos desexa cambiar. O administrador cambia entón o seu tipo de usuario de *xogador* a *director de xogo*. De ter xa un personaxe asignado, deixará de estalo neste mesmo momento.

CU-03: Definición dunha nova clase

Actores Director de xogo

Escenario principal O caso de uso comeza cando un director de xogo pretende crear unha nova clase na partida. Na interface gráfica deberá acoder á sección de *definición do mundo*. Unha vez aí, escollerá a opción de crear unha clase nova, e implementará a clase como desexe. Unha vez rematada, o sistema analizará a clase para comprobar se ten algún erro (léxico ou sintáctico), e en caso de non atopar ningún, a clase será almacenada e estará lista para o seu uso posterior.

Escenario alternativo A clase ten algún erro, por tanto o sistema non a garda nun ficheiro e, no seu lugar, mostra unha mensaxe de erro instando ao DX a reescribila.

CU-04: Creación de personaxe

Actores Director de xogo, xogador

Escenario principal O caso de uso comeza cando un xogador desexa crear o seu personaxe. O xogador accede á sección de *creación de personaxe*, onde atopará un cadro de texto que lle solicitará unha breve descrición do que desexa. O xogador redacta o texto e envíao ao sistema. O director de xogo trata este texto como unha decisión máis, e a acción consecuente é crear un obxecto personaxe ligado ao xogador que o solicitou, situado nunha estancia concreta.

Escenario alternativo O DX non ten información suficiente na descrición como para facer un personaxe. O DX elimina esta descrición e escribe unha resposta ao xogador, que se verá obrigado a repetir o proceso.

CU-05: Resolución de decisións en accións

Actores Director de xogo, xogador

Escenario principal O caso de uso comeza cun xogador desexando facer algo na partida. Accede á sección de *actuar*, onde poderá ler as últimas accións que sucederon na estancia na que se atopa o seu personaxe. O xogador redacta a súa decisión e envíaa ao DX. Unha vez feito isto, non poderá tomar máis decisións ata que a primeira se resolva. O director de xogo recibe unha notificación de que hai estancias con decisións non resoltas. Accese á sección de *resolver accións*, onde ve unha lista de estancias con decisións sen resolver. Abre unha das estancias e le a lista de decisións e os xogadores que as enviaron, ademais do estado actual da estancia. En consecuencia, escribirá as accións que mellor representen as decisións tomadas polos xogadores, que pasarán a ser executada polo sistema. Finaliza o proceso, e o sistema executa o código das accións introducidas polo DX. O xogador observa entón un breve texto describindo o sucedido.

Escenario alternativo O DX comete algún erro escribindo o código das accións, polo que non se poderán executar. O sistema devolve unha mensaxe de erro.

CU-06: Creación dunha estancia

Actores Director de xogo

Escenario principal O caso de uso comeza cando o director de xogo pretende engadir unha nova estancia ao mundo. Accede á sección de *lista de estancias*, onde aparecerán as estancias que xa existen debidamente ordenadas. Mediante a opción *crear nova estancia* poderá crear unha estancia nova de cero (sen obxectos). Finaliza o proceso e a estancia créase.

Escenario alternativo O DX pode optar por crear a estancia a partir dun modelo e non dende cero. Neste caso, executarase un código determinado no momento de crearse a estancia, que a encherá de obxectos predefinidos.

CU-07: Execución dun suceso

Actores Director de xogo

Escenario principal O caso de uso comeza cando o director de xogo pretende causar un suceso nunha estancia. O DX accede á sección de *lista de estancias*, e selecciona a estancia na que desexe causar o suceso. Unha vez no panel da estancia, selecciona a opción *causar suceso*, e escribe o código do suceso. Cando finaliza a opción, o sistema executa o código.

Escenario alternativo O DX comete algún erro escribindo o código do suceso, polo que non se poderán executar. O sistema devolve unha mensaxe de erro.

CU-08: Cambio de estancia dun obxecto

Actores Director de xogo

Escenario principal O caso de uso comeza cando o director de xogo pretende mover un obxecto dunha estancia a outra, sexa como consecuencia dunha decisión dun xogador ou non. O DX escribe o código do cambio de estancia, especificando o obxecto que se desprazará e a estancia de destino. Cando se executa a acción, o obxecto abandona a estancia de orixe e pasa a estar na estancia de destino. No momento de resolver as accións da estancia de destino, o sistema avisará da entrada de novos obxectos na mesma para que o DX o teña en conta.

3.3. Requisitos funcionais

3.3.1. Requisito FN.01

Título: Alta de usuarios
Descrición: A aplicación debe permitir que se creen novas contas de usuario.
Casos de uso relacionados: CU-01
Importancia: Esencial

3.3.2. Requisito FN.02

Título: Modificación de usuarios
Descrición: A aplicación debe permitir que se modifiquen contas de usuario existentes.
Casos de uso relacionados: CU-02
Importancia: Esencial

3.3.3. Requisito FN.03

Título: Identificación de usuarios
Descrición: A aplicación debe permitir que os usuarios se identifiquen correctamente.
Casos de uso relacionados: CU-02, CU-03, CU-04, CU-05, CU-06, CU-07, CU-08
Importancia: Esencial

3.3.4. Requisito FN.04

Título: Creación de clases
Descrición: A aplicación debe permitir ao director de xogo crear clases novas no mundo.
Casos de uso relacionados: CU-03
Importancia: Esencial

3.3.5. Requisito FN.05

Título: Modificación de clases **Descrición:** A aplicación debe permitir ao director de xogo modificar clases existentes.
Casos de uso relacionados: CU-03
Importancia: Esencial

3.3.6. Requisito FN.06

Título: Eliminación de clases
Descrición: A aplicación debe permitir ao director de xogo eliminar clases existentes. A aplicación debe garantir que non se borra unha clase da que hai obxectos instanciados.
Casos de uso relacionados: CU-03
Importancia: Esencial

3.3.7. Requisito FN.04

Título: Herdanza de clases
Descrición: A aplicación debe permitir facer que unhas clases sexan fillas de outras, herdando deste xeito os seus métodos.
Casos de uso relacionados: CU-03
Importancia: Esencial

3.3.8. Requisito FN.07

Título: Creación de obxecto
Descrición: A aplicación debe permitir crear novos obxectos dende as clases xa existentes. Estes obxectos créanse no contexto dunha estancia determinada.
Casos de uso relacionados: CU-04, CU-05, CU-06, CU-07 **Importancia:** Esencial

3.3.9. Requisito FN.08

Título: Modificación de obxecto
Descrición: A aplicación debe permitir modificar os atributos de obxectos existentes.
Casos de uso relacionados: CU-05, CU-07 **Importancia:** Esencial

3.3.10. Requisito FN.09

Título: Eliminación de obxecto
Descrición: A aplicación debe permitir eliminar obxectos nas estancias.
Casos de uso relacionados: CU-05, CU-07 **Importancia:** Esencial

3.3.11. Requisito FN.10

Título: Ligazón de obxecto personaxe a xogador
Descrición: A aplicación debe permitir que un determinado obxecto personaxe se asocie a un xogador, para deste xeito permitir ao xogador ver os resultados das súas accións e enviar as súas decisións á estancia axeitada.
Casos de uso relacionados: CU-04
Importancia: Esencial

3.3.12. Requisito FN.11

Título: Creación dunha estancia baleira
Descrición: A aplicación debe permitir ao director de xogo crear unha nova estancia de cero que non conteña obxectos.
Casos de uso relacionados: CU-06
Importancia: Esencial

3.3.13. Requisito FN.12

Título: Eliminación dunha estancia
Descrición: A aplicación debe permitir ao director de xogo eliminar unha estancia do mundo. Todos os obxectos que conteña a estancia serán eliminados. A estancia non pode conter personaxes.
Casos de uso relacionados: CU-06
Importancia: Opcional

3.3.14. Requisito FN.13

Título: Creación dun modelo de estancia
Descrición: A aplicación debe permitir ao director de xogo deseñar un modelo de estancia. Este modelo conterá código que se executará no momento de crear a estancia.
Casos de uso relacionados: CU-06
Importancia: Opcional

3.3.15. Requisito FN.14

Título: Creación dunha estancia dende modelo **Descrición:** A aplicación debe permitir ao director de xogo crear unha nova estancia empregando un modelo de estancia, executando o código que este modelo contén.
Casos de uso relacionados: CU-06
Importancia: Opcional

3.3.16. Requisito FN.15

Título: Cambio de estancia dun obxecto
Descrición: A aplicación debe permitir ao director de xogo mover un obxecto dunha estancia a outra. O identificador global do obxecto permanece igual, pero na nova estancia pode ter un identificador local diferente.
Casos de uso relacionados: CU-05, CU-07
Importancia: Esencial

3.3.17. Requisito FN.16

Título: Envío de decisións
Descrición: A aplicación debe permitir aos xogadores enviar as súas decisións para que o DX poda tratalas.
Casos de uso relacionados: CU-05
Importancia: Esencial

3.3.18. Requisito FN.17

Título: Visualización de resultados
Descrición: A aplicación debe permitir que os xogadores podan ver os resultados

das accións que involucren aos seus personaxes, sexan accións causadas polas súas decisións, ou accións e sucesos na estancia na que se atopa o seu personaxe. **Casos de uso relacionados:** CU-05, CU-07 **Importancia:** Esencial

3.4. Restricións de deseño

3.4.1. Requisito RD.01

Título: Uso de ferramentas libres
Descrición: A aplicación debe realizarse empregando unicamente ferramentas libres, como unha forma non só de reducir custos, senón tamén de permitir unha licencia libre do propios software resultante.
Importancia: Esencial

3.5. Requisitos non funcionais

3.5.1. Requisito NF.01

Título: Linguaxe: Lóxica de estancias
Descrición: A linguaxe debe proporcionar ao DX a capacidade de crear e eliminar estancias, mover obxectos dunha estancia á outra, obter a lista de personaxes nunha estancia, e asignar variables locais para nomear aos obxectos ou almacenar datos de tipos primitivos.O código executado nunha estancia non debe afectar nun principio a outras estancias diferentes, salvo no movemento de obxectos entre elas.
Importancia: Esencial

3.5.2. Requisito NF.02

Título: Linguaxe: Lóxica de visibilidade e ocultación
Descrición: A linguaxe debe proporcionar a capacidade de determinar que obxectos son visibles por un personaxe e cales non mediante o uso de condicións. Isto non ten un efecto real na xogabilidade (un personaxe pode interactuar cun obxecto cuxa existencia descoñece se así o determina o DX no seu código), mais é relevante
Importancia: Opcional

3.5.3. Requisito NF.03

Título: Linguaxe: Lóxica de inventario
Descrición: A linguaxe debe permitir que uns obxectos se conteñan aos outros. Na práctica isto ten varias consecuencias, sendo a máis importante o feito de que o obxecto dependente deba estar forzosamente na mesma estancia co obxecto recipiente. Tamén ten o seu efecto na visibilidade, xa que se un personaxe non pode ver ao recipiente nunca poderá ver os obxectos que contén. A linguaxe debe ofrecer a capacidade de especificar as clases que poden converterse en recipientes, métodos para introducir uns obxectos noutros, e tamén proporcionará a capacidade de recibir a lista de obxectos dun recipiente.
Importancia: Esencial

3.5.4. Requisito NF.04

Título: Linguaxe: Tipos de datos básicos
Descrición: A linguaxe debe implementar como tipos primitivos os números (enteiros ou en punto flotante) e as cadeas de texto. Tamén deben implementarse os arrays, sexan de números ou de obxectos.
Importancia: Esencial

3.5.5. Requisito NF.05

Título: Linguaxe: Lóxica de eventos
Descrición: A linguaxe debe ofrecer a opción de definir código de clase que se executa automaticamente ante determinados eventos, tales como o cambio de rolda, a chegada de novos obxectos á estancia ou o cambio de estancia do propio obxecto.
Importancia: Opcional

3.5.6. Requisito NF.06

Título: Linguaxe: Condicionais e tiradas de dados
Descrición: A linguaxe debe ofrecer a capacidade de empregar condicións lóxicas para alterar o fluxo do código. Polo outro lado, a linguaxe debe prover dun método de obter valores aleatorios para simular as tiradas de dados, nas que o DX poda especificar o número de dados lanzados e o número de caras de cada dado.
Importancia: Esencial

3.5.7. Requisito NF.07

Título: Linguaxe: Iteracións
Descrición: A linguaxe debe prover de funcións que permitan traballar en listas de elementos de forma iterativa, tratando cada elemento por separado.
Importancia: Esencial

3.5.8. Requisito NF.08

Título: Linguaxe: Operacións lóxicas e matemáticas
Descrición: A linguaxe debe ofrecer as operacións lóxicas e matemáticas básicas, tales como comparacións, sumas e multiplicacións. Tamén debe permitir a operación en arrays, sexa entre dous arrays ou entre un array e un número (elemento a elemento).
Importancia: Esencial

3.5.9. Requisito NF.09

Título: Linguaxe: Prioridade de decisións
Descrición: A linguaxe debe permitir definir a orde na que se resolverán as decisións dos xogadores mediante funcións de comparación. No momento de resolver as decisións, estas aparecerán ante o DX ordenadas como corresponda.
Importancia: Opcional

3.5.10. Requisito NF.10

Título: Resultado textual: Seccións comúns e particulares
Descrición: No momento de mostrar o texto resultante dunha acción, o DX debe ter a opción de escribir unha narración común en 3ª persoa para todos os personaxes, e narracións personalizadas en 2ª persoa para cada xogador por separado.
Importancia: Esencial

3.5.11. Requisito NF.11

Título: Resultado textual: Condicións
Descrición: No momento de mostrar o texto resultante dunha acción, o DX deberá ter a opción de definir condicións en determinados bloques de texto. Os

xogadores que non cumplan tales condicións non poderán ler eses bloques de texto. As condicións poderán estar determinadas por variables locais da estancia e por variables de obxectos concretos.
Importancia: Esencial

3.5.12. Requisito NF.12

Título: Resultado textual: Formato especial
Descrición: No momento de mostrar o texto resultante dunha acción, o DX poderá empregar formato especial, tal como letra en negriña ou cursiva, ou o uso de imaxes e enlaces web.
Importancia: Opcional

3.5.13. Requisito NF.13

Título: Resultado textual: Anonimato
Descrición: No momento de mostrar o texto resultante dunha acción, os xogadores non lerán directamente o nome dos personaxes mencionados, senón que verán o nome co que eles coñezan a tales personaxes.
Importancia: Opcional

3.6. Requisitos de proxecto

3.6.1. Requisito PR.01

Título: Data límite do proxecto
Descrición: O proxecto deberá rematarse antes do 8 de febreiro do 2017, data establecida no regulamento de traballos fin de grao para GREI da USC.
Importancia: Esencial

3.7. Requisitos de calidade

3.7.1. Requisito CA.01

Título: Internacionalización
Descrición: O software deberá estar deseñado de tal xeito que se poda traducir facilmente a distintos idiomas sen realizar cambios no código.
Importancia: Opcional

3.7.2. Requisito CA.02

Título: Manual de usuario
Descrición: O software resultante deberá ir acompañado dun manual de usuario que explique o funcionamento do mesmo, incidindo especialmente no uso da linguaxe propia.
Importancia: Esencial

3.7.3. Requisito CA.03

Título: Protección contra ataques
Descrición: O software resultante deberá estar protexido contra os principais ataques, como por exemplo a inxección de código SQL e a introdución de código HTML e Javascript non intencionado.
Importancia: Esencial

3.8. Requisitos de almacenamento

3.8.1. Requisito AL.01

Título: Persistencia da información
Descrición: A aplicación debe garantir que a información do mundo (clases, obxectos, estancias) se conservará cando a aplicación remate e volva a poñerse en funcionamento.
Importancia: Esencial

Capítulo 4

Deseño

Deseño: cómo se realiza o Sistema, a división deste en diferentes compoñentes e a comunicación entre eles. Así mesmo, determinarase o equipamento hardware e software necesario, xustificando a súa elección no caso de que non fora un requisito previo. Debe achegarse a un nivel suficiente de detalle que permita comprender a totalidade da estrutura do produto desenvolvido, utilizando no posible representacións gráficas.

Capítulo 5

Exemplos

5.1. Un exemplo de sección

Esta é *letra cursiva*, esta é **letra negrilla**, esta é letra subrallada, e esta é `letra curier`. Letra tiny, scriptsize, small, large, Large, LARGE e moitas más. Exemplo de fórmula: $a = \int_o^\infty f(t)dt$. E agora unha ecuación aparte:

$$S = \sum_{i=0}^{N-1} a_i^2. \tag{5.1}$$

As ecuaciones se poden referenciar: ecuación (5.1).

5.1.1. Un exemplo de subsección

O texto vai aquí.

5.1.2. Otro exemplo de subsección

O texto vai aquí.

Un exemplo de subsubsección

O texto vai aquí.

Un exemplo de subsubsección

O texto vai aquí.

Un exemplo de subsubsección

O texto vai aquí.

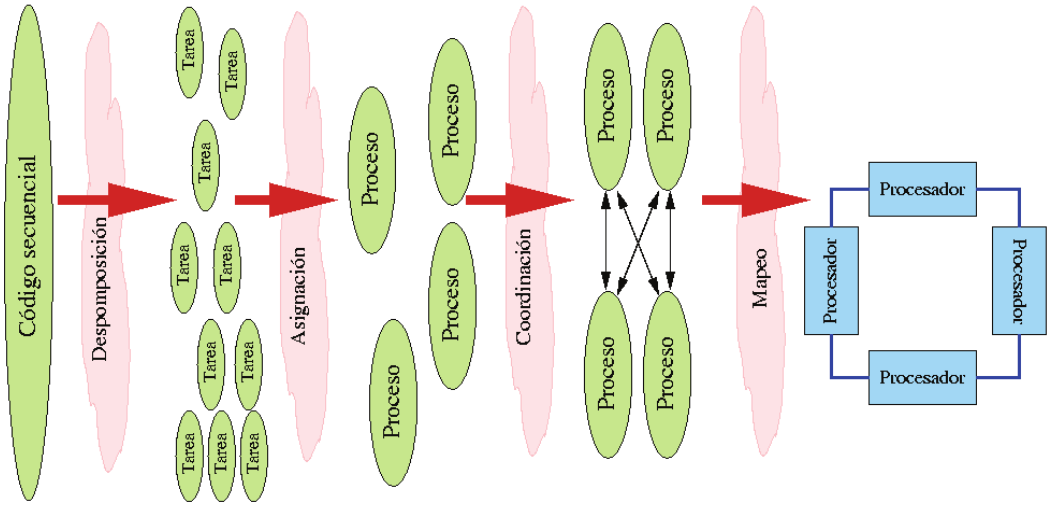

Figura 5.1: Esta é a figura de tal e cal.

Izquierda	Derecha	Centrado
ll	r	cccc
llll	rrr	c

Cadro 5.1: Esta é a táboa de tal e cal.

5.2. Exemplos de figuras e cadros

A figura número 5.1.
O cadro (taboa) número 5.1.

5.3. Exemplos de referencias á bibliografía

Este é un exemplo de referencia a un documento descargado da web [1]. E este é un exemplo de referencia a unha páxina da wikipedia [2]. Agora un libro [3] e agora unha referencia a un artigo dunha revista [4]. Tamén se poden pór varias referencias á vez [1, 3].

5.4. Exemplos de enumeracións

Con puntos:

- Un.
- Dous.

- Tres.

Con números:

1. Catro.
2. Cinco.
3. Seis.

Exemplo de texto verbatim:

```
O texto        verbatim
     se visualiza tal
            como se escribe
```

Exemplo de código C:

```c
#include <math.h>
main()
{   int i, j, a[10];
    for(i=0;i<=10;i++) a[i]=i; // comentario 1
    if(a[1]==0) j=1; /* comentario 2 */
    else j=2;
}
```

Exemplo de código Java:

```java
class HelloWorldApp {
    public static void main(String[] args) {
        System.out.println("Hello World!"); // Display the string.
    }
}
```

Capítulo 6

Conclusións e posibles ampliacións

Conclusións e posibles ampliacións

Apéndice A

Manuais técnicos

Manuais técnicos: en función do tipo de Traballo e metodoloxía empregada, o contido poderase dividir en varios documentos. En todo caso, neles incluirase toda a información precisa para aquelas persoas que se vaian a encargar do desenvolvemento e/ou modificación do Sistema (por exemplo código fonte, recursos necesarios, operacións necesarias para modificacións e probas, posibles problemas, etc.). O código fonte poderase entregar en soporte informático en formatos PDF ou postscript.

Apéndice B

Manuais de usuario

Manuais de usuario: incluirán toda a información precisa para aquelas persoas que utilicen o Sistema: instalación, utilización, configuración, mensaxes de erro, etc. A documentación do usuario debe ser autocontida, é dicir, para o seu entendemento o usuario final non debe precisar da lectura de outro manual técnico.

Apéndice C

Licenza

Se se quere pór unha licenza (GNU GPL, Creative Commons, etc), o texto da licenza vai aquí.

Bibliografía

[1] Nvidia CUDA programming guide. Versión 2.0, 2010. Dispoñible en *http://www.nvidia.com*.

[2] Acceso múltiple por división de código. Artigo da wikipedia (*http://es.wikipedia.org*). Consultado o 2 de xaneiro do 2010.

[3] R.C. Gonzalez e R.E. Woods, *Digital image processing*, 3ª edición, Prentice Hall, New York, 2007.

[4] P. González, J.C. Cartex e T.F. Pelas, "Parallel computation of wavelet transforms using the lifting scheme", *Journal of Supercomputing*, vol. 18, no. 4, pp. 141-152, junio 2001.