

Grado en Ingeniería Informática - Ingeniería del Software


Evolución y Gestión de la Configuración



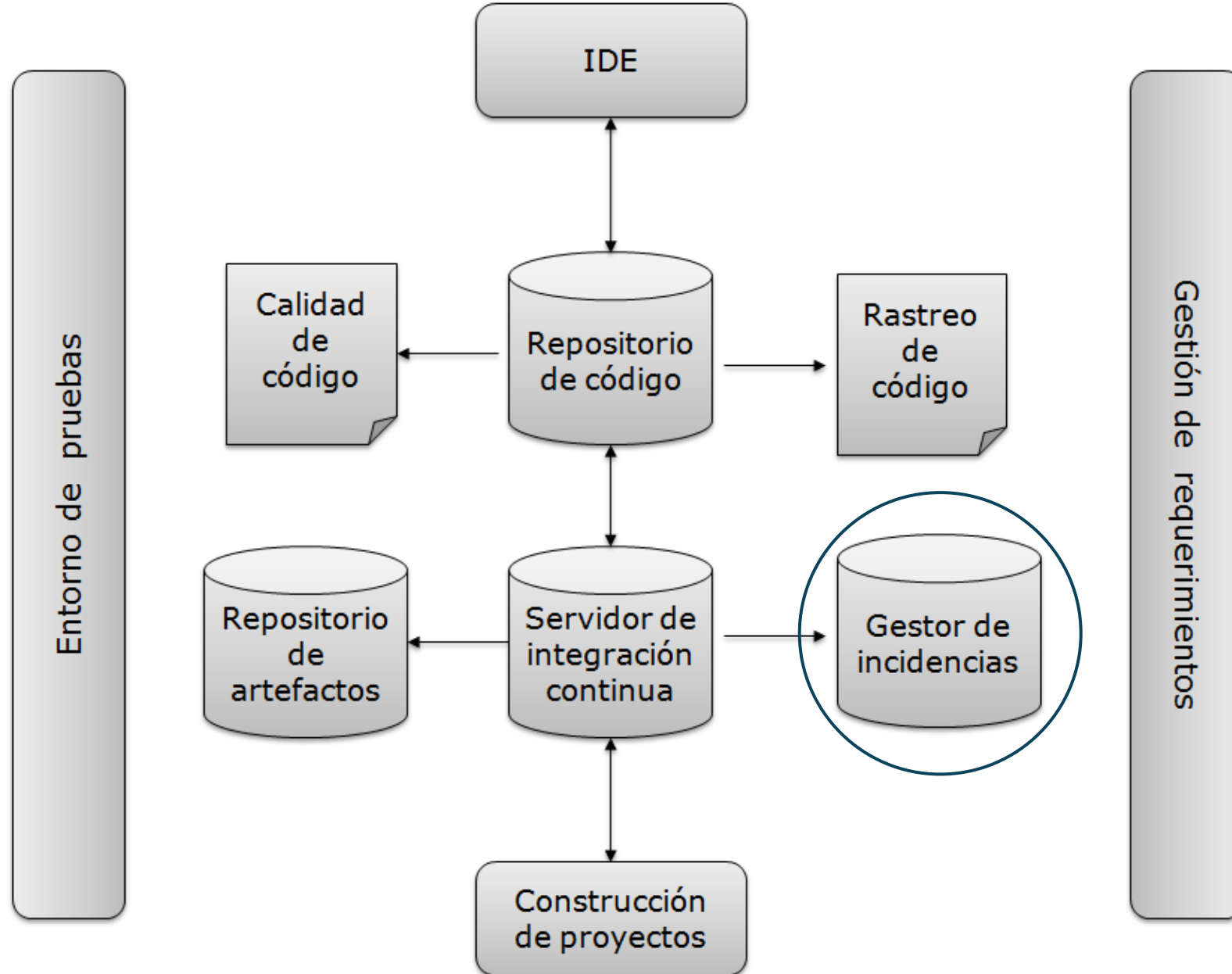
Escuela Técnica Superior de
Ingeniería Informática

Gestión de tareas e incidencias en el ciclo de integración continua

Objetivo principal:
Saber cómo organizar
las tareas relacionadas
con los *work-items* y las
incidencias

- 1. Conceptos básicos**
 - 2. Tipos de cambio**
 - 3. Depuración**
 - 4. Resumen**
 - 5. Bibliografía**
- 

1. Conceptos básicos | ecosistemas de desarrollo



1. Conceptos básicos | proceso de petición de cambios

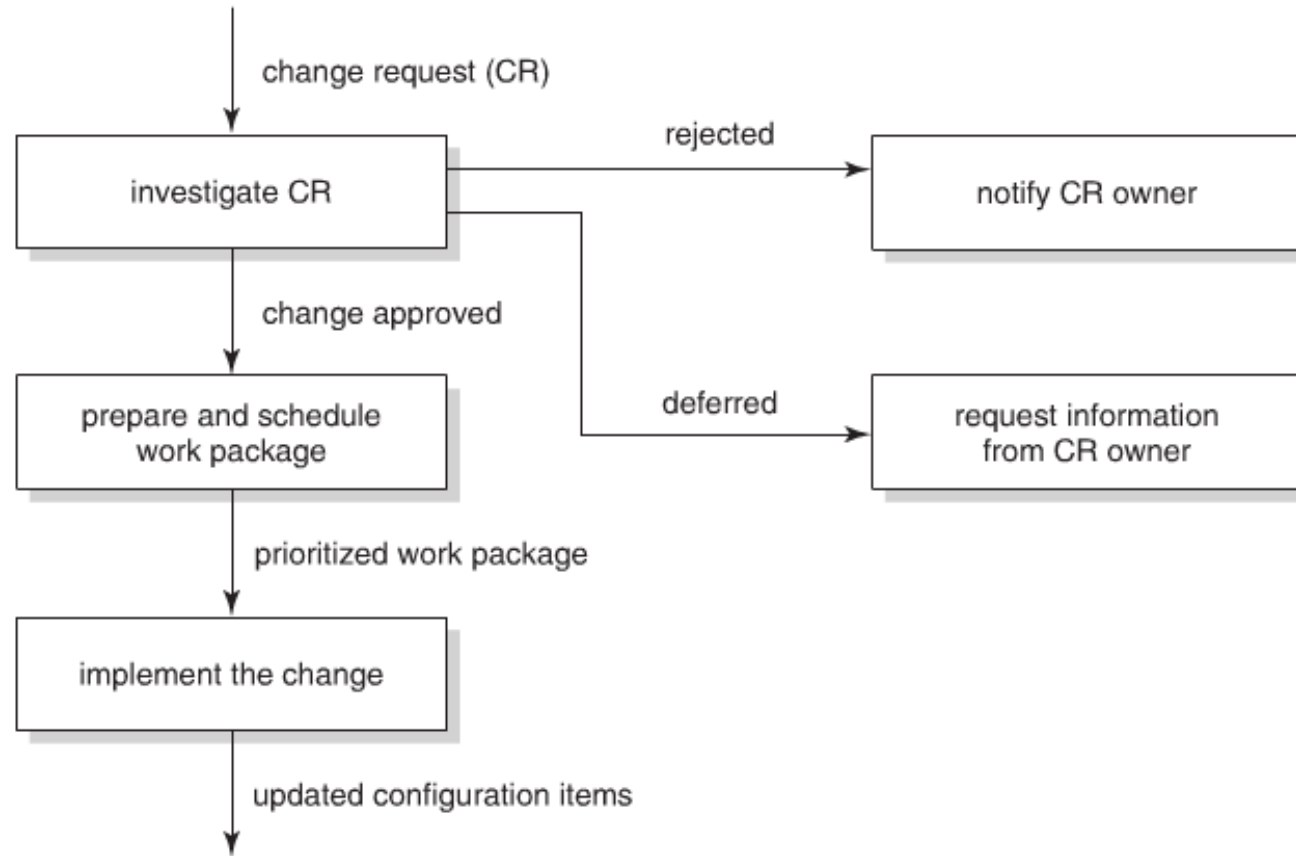
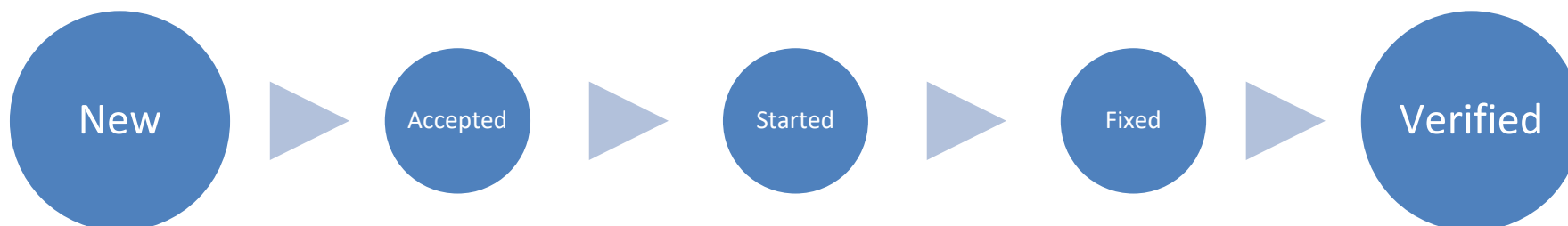
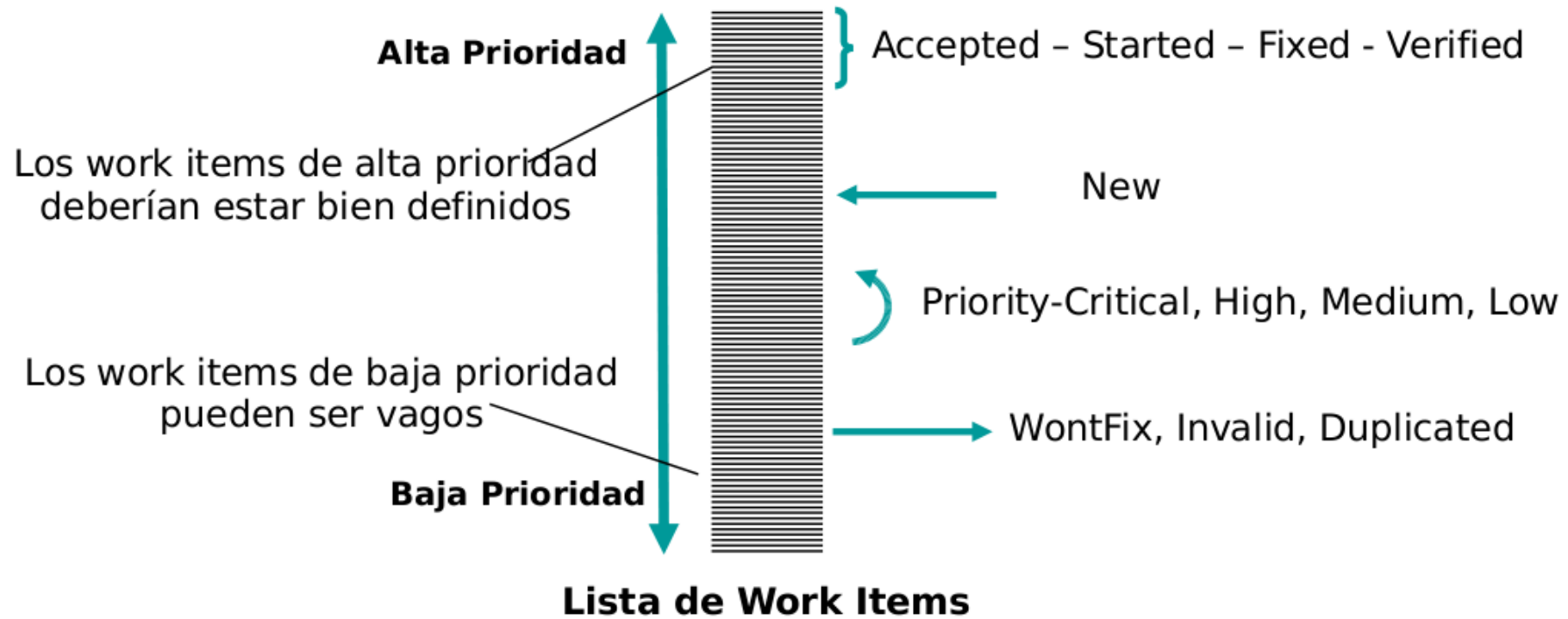


Figure 4.1 Workflow of a change request

1. Conceptos básicos | proceso de petición de cambios



1. Conceptos básicos | ejemplo de tablero

The image shows a Jira board for the project "OctoArcade Invaders". The board is organized into four columns representing different stages of the workflow: "Not Started", "Planning", "Building", and "Complete". Each column contains several issues (tasks) with their respective IDs, descriptions, and assignees. The "Not Started" column has 3 issues, "Planning" has 3, "Building" has 4, and "Complete" has 3. The board also features a navigation bar at the top with tabs for "The Plan", "Game loop Backlog", "Standup", and "New view".

OctoArcade Invaders

The Plan | Game loop Backlog | **Standup** | + New view

Not Started 3

- OctoArcade #10: Integrate with Leaderboard Service (Need help)
- OctoArcade #183: Interviews with media outlets
- OctoArcade #45: Save score across levels

Planning 3

- OctoArcade #42: Creative design update to aliens for variety
- OctoArcade #79: Alpha go-no-go meeting
- OctoArcade #12: Easter egg with high score unlocking a new paid level on map 8

Building 4

- OctoArcade #19: Updates to alien, beam, and cannon sprites (Design)
- OctoArcade #3: New start screen and multiplayer selection
- OctoArcade #38: Updates to velocity of the ship and alien movements (Bug)
- OctoArcade #17: Update to collision logic (Bug)

Complete 3


- OctoArcade #56: Game brief and go-no-go
- OctoArcade #21: Engine prototype (physics, rendering) (Need help)
- OctoArcade #31: Initial concept art

1. Conceptos básicos | lo básico

Todo proceso de gestión del cambio debe tener al menos tener:

- Prioridad del cambio
- Estado en la que se encuentra
- Tipo de cambio
- Roles en la gestión de dicho cambio



1. Conceptos básicos
 2. Tipos de cambio
 3. Depuración
 4. Resumen
 5. Bibliografía
- 

2. Tipos de cambio | posibles taxonomías

Por quien los origina

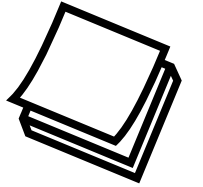
Por el impacto

Por la intención

Por el tiempo

2. Tipos de cambio | nuestra taxonomía

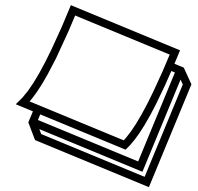
Work items y sus
derivados



Incidencias



2. Tipos de cambio | Work ítems (WIs)



SPRINT

JUST 5 DAYS

JAKE KNAPP
CREATED THE 5-DAY PROCESS

TO HELP PROTOTYPE AND VALIDATE IDEAS

⌚ RAPIDLY AND EFFECTIVELY

THEIR OUTCOMES WON'T BE REALIZED UNTIL MONTHS HAVE PASSED

TOUGH DECISIONS

5 DAY SPRINT

S M T W T F S

ITERATE AND TEST

IN A VERY COMPRESSED TIME FRAME

SPRINT METHODOLOGY
SHORTCUTS MONTHS OF DEBATE & DEVELOPMENT

EXAMPLE: ONLINE STORE WOULD TAKE

⌚ A LONG TIME
⌚ YEARS OF REFINEMENT

BLUE BOTTLE COFFEE

INITIAL DIRECTION: **CRUCIAL**

THAT'S WHERE THE 5 DAY SPRINT CAME IN

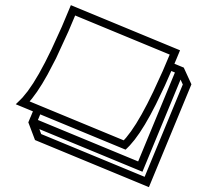
SPRINT
el método para

RESOLVER PROBLEMAS Y TESTAR NUEVAS IDEAS EN SÓLO CINCO DÍAS

JAKE KNAPP
CON JOHN ZERATSKY Y BRADEN KOWITZ
de **GOOGLE VENTURES**

connecta

2. Tipos de cambio | Design sprint



day 1



understand

- who are the users
- what are their needs
- what is the context
- competitor review
- formulate strategy

2



diverge

- envision
- developlots of solutions
- ideate

3



decide

- choose the best idea
- storyboard the idea

4



prototype

- build something quick and dirty to show to users
- focus on usability not making it beautiful

5

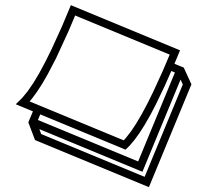


validate

- show the prototype to real users outside the organisation
- learn what doesn't work

M1

2. Tipos de cambio | Prototipado realista



4



prototype

- build something quick and dirty to show to users
- focus on usability not making it beautiful

5



validate

- show the prototype to real users outside the organisation
- learn what doesn't work

Download in different formats

Download in different formats #20

[Open](#) drorganvidez opened this issue on May 8 · 0 comments



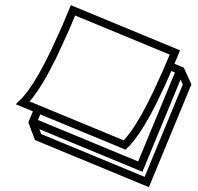
drorganvidez commented on May 8

Member ...

As a user, I want to be able to download the models or datasets in different formats (e.g., dimacs), for use in my own projects.

M1

2. Tipos de cambio | ejemplo



The screenshot displays the uvlhub.io website interface. On the left is a dark sidebar with navigation links: Home, Explore, Team, Login, and Sign Up. The main content area features a search bar at the top and a 'Latest datasets' section. Three dataset cards are visible: 'Smart Governance Product Line' (3.81 KB), 'Software Strategies' (1.48 KB), and 'Feature modelling book' (Book). A right-hand sidebar contains 'Hub statistics' (24 datasets, 1489 feature models, etc.) and 'uvlhub related publication' information. A 'Let's get started!' banner is at the bottom right.

uvlhub.io

Search datasets...

Get all datasets! Login Sign up

Latest datasets

Smart Governance Product Line

None

April 27, 2024 at 11:09 AM

This dataset models a (service-oriented) software product line that represents the features and variability of collaborative e-governance systems and services that could be deployed in public institutions.

Muñoz-Hermoso, Salvador

<http://uvlhub.io/doi/10.5281/zenodo.12697539>

e-governance smart governance e-collaboration e-government

[View dataset](#) [Download \(3.81 KB\)](#)

Software Strategies

None

April 23, 2024 at 09:16 AM

This model is the first version of the model for defining Software Strategies

Olivero, Miguel (Universidad de Sevilla) (0000-0002-6627-3699)
Francisco José Domínguez Mayo (University of Seville) (0000-0003-3502-8858)
David Benavides (University of Seville) (0000-0002-8449-3273)
Ernest Teniente (Universitat Oberta de Catalunya) (0000-0001-8890-9638)

<http://uvlhub.io/doi/10.5281/zenodo.12697511>

[View dataset](#) [Download \(1.48 KB\)](#)

Feature modelling book

Book

March 22, 2024 at 12:37 PM

Hub statistics

- 24 datasets
- 1489 feature models
- 2432 datasets viewed
- 150 feature models viewed
- 1335 datasets downloaded
- 15544 feature models downloaded

uvlhub related publication

David Romero-Organvidez, José A. Galindo, Chico Sundermann, Jose-Miguel Horcas, David Benavides. *UVLHub: A feature model data repository using UVL and open science principles*, Journal of Systems and Software, 2024, 112150, ISSN 0164-1212, <https://doi.org/10.1016/j.jss.2024.112150>

[Copy in BibTex](#) [Copy in RIS](#)

Let's get started!

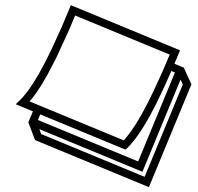
What are you waiting for to upload your fantastic feature models in UVL format?

[Sign up](#)

[I am already registered!](#)

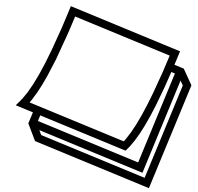
Compilation build v1.13

2. Tipos de cambio | ejemplo



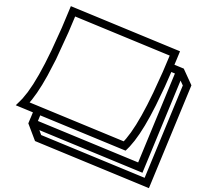
The screenshot displays the uvlhub.io website interface. On the left, a dark navigation sidebar contains the logo and menu items: Home, Explore, Team, and options for Login and Sign Up. The main content area features a search bar at the top with the text "Search datasets...". Below this, a "Latest datasets" section highlights a dataset titled "Smart Governance Product Line" with a "None" tag, dated April 27, 2024. To the right, a "Hub statistics" box lists: 24 datasets, 1489 feature models, 2432 datasets viewed, and 150 feature models viewed. A search bar at the bottom contains the text "uvlhub_bulk_2024_10_11.zip" and a red "Guardar" button. At the bottom of the page, a "Feature modelling book" is advertised with a "Book" tag and a "Sign up" button.

2. Tipos de cambio | ejemplo



The screenshot shows the uvlhub.io website interface. On the left is a dark navigation sidebar with the logo and links for Home, Explore, Team, Login, and Sign Up. The main content area features a search bar at the top, a 'Latest datasets' section with a card for 'Smart Governance Product Line', and a 'Hub statistics' box. A dark download progress bar is overlaid on the page, showing the file 'uvlhub_bulk_2024_10_11.zip' with a ZIP icon, a progress bar, and the text '59s restantes — 267 de 502 MB (3,9 MB/seg.)'. Below the progress bar is a search bar containing the same filename and an orange 'Guardar' button. At the bottom, there are banners for 'Compilation build v1.13' and 'Feature modelling book'.

2. Tipos de cambio | ejemplo



uvlhub.io Search datasets... Get all datasets! Login Sign up

Ubicación: /dataset_11/

Nombre	Tamaño	Tipo
uvl	14,8 MB	Carpeta
splot	129,4 MB	Carpeta
glencoe	168,5 MB	Carpeta
dimacs	26,1 MB	Carpeta

as >



Compilation build v1.13

Feature modelling book

March 22, 2024 at 12:37 PM

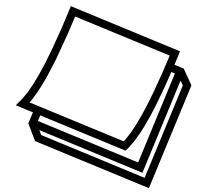
Book

feature models in UVL format?

Sign up

I am already registered!

2. Tipos de cambio | ejemplo



uvlhub.io Search datasets... Get all datasets! Login Sign up

Ubicación: /dataset_11/spot/

Nombre	Tamaño	Tipo
integrator_arm9.splx	11,2 MB	desconocido
excalibur_arm9.splx	11,2 MB	desconocido
aaed2000.splx	11,2 MB	desconocido
smdk2410.splx	11,2 MB	desconocido
innovator.splx	11,2 MB	desconocido
se77x9.splx	8,4 MB	desconocido
hs7729pci.splx	8,4 MB	desconocido
sh7708.splx	8,4 MB	desconocido
cq7708.splx	8,4 MB	desconocido


Compilation build v1.13

Feature modelling book
March 22, 2024 at 12:37 PM

feature models in UVL format?
Sign up
I am already registered!

2. Tipos de cambio | Incidencias



1. **Conceptos básicos**
 2. **Tipos de cambio**
 3. **Depuración**
 4. **Resumen**
 5. **Bibliografía**
- 

3. Depuración | Proceso general en caso de error

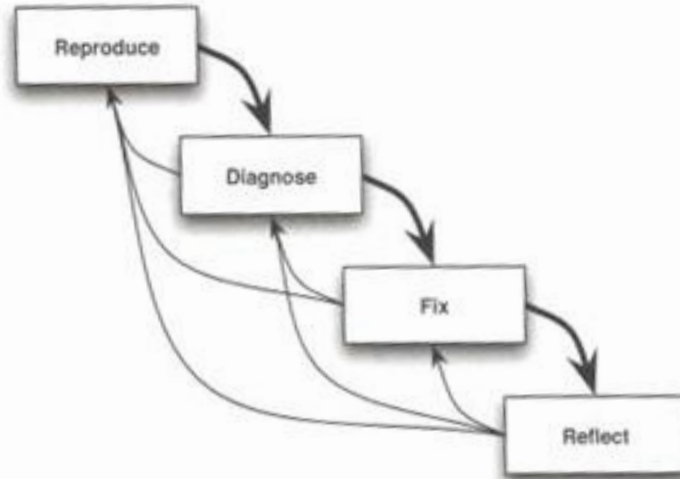


Figure 1.1: CORE DEBUGGING METHOD

- Reproducir: buscar una manera de reproducir el problema de una manera sistemática
- Diagnosticar: Construir hipótesis y validarlas haciendo experimentos
- Reparar: Diseñar e implementar cambios para solucionar el problema
- Analizar: aprender de las lecciones del error

3. Depuración | Distintos pasos

Depuración

- **Informar de la incidencia**
- Reproducir
- Diagnosticar
- Arreglar
- Analizar

Ley de Linux:

"dado un número suficientemente elevado de ojos, todos los errores se convierten en obvios"

3. Depuración | Informar de la incidencia

- Antes de empezar a intentar reproducir debemos estar seguros de entender qué es lo que está pasando:
 - Necesitamos un *bug-tracking system*:



3. Depuración | *bug tracking system*

- Nos permite:
 - No olvidar la incidencia
 - Forma estándar de informar sobre incidencias
 - Para auditar *bugs* (e.g en las entregas)
 - Para priorizar y construir la “*work item list*”
 - Para comunicación en el equipo
 -

Todo proyecto de software debe contar con un sistema de gestión de incidencias

3. Depuración | *bug tracking system*

Pero un sistema de gestión de incidencias es el medio, no el fin. Debe haber buenas prácticas al usarlo.

3. Depuración | ¿Cómo informar de una incidencia?

- Cuántos más detalles mejor
- Intentar que sea específico, sin ambigüedades, detallado.
- Debe ser único (si ya se reportó otra vez, ignorarlo o mezclarlo)

Así no	Mejor así
Salta una excepción cuándo pulso el botón de descargar un data set	Al pulsar el botón de descargar un dataset en un dataset con más de un modelo, con el navegador Firefox versión XX, salta la excepción “Overflow Exception” (compio mensaje de error)
Cuando entro en la sección de visualizar un modelo, uno de los elementos a visualizar me sale con símbolos raros	Al intentar visualizar el model M del dataset XXX, sale con un número de caracteres así: C%%%&&&

3. Depuración | ¿Cómo informar de una incidencia?

Información de los usuarios

- Usar una plantilla
- Automatizar al máximo

Asumir que de cada error que reporte un usuario, habrá entre 10 y 100 que lo hayan experimentado y no lo reporten [butcher]

3. Depuración | ¿Cómo informar de una incidencia?

New issue | Search for Search | [Advanced search](#)

Summary:

Description:
1.
2.
3.

What is the expected output? What do you see instead?

What version of the product are you using? On what operating system?

Please provide any additional information below.

3. Depuración | ¿Cómo informar de una incidencia?

What Makes a Good Bug Report?

Thomas Zimmermann, *Member, IEEE*, Rahul Premraj, Nicolas Bettenburg, *Member, IEEE*, Sascha Just, *Member, IEEE*, Adrian Schröter, *Member, IEEE*, and Cathrin Weiss

Abstract—In software development, bug reports provide crucial information to developers. However, these reports widely differ in their quality. We conducted a survey among developers and users of APACHE, ECLIPSE, and MOZILLA to find out what makes a good bug report. The analysis of the 466 responses revealed an information mismatch between what developers need and what users supply. Most developers consider steps to reproduce, stack traces, and test cases as helpful, which are, at the same time, most difficult to provide for users. Such insight is helpful for designing new bug tracking tools that guide users at collecting and providing more helpful information. Our CUEZILLA prototype is such a tool and measures the quality of new bug reports; it also recommends which elements should be added to improve the quality. We trained CUEZILLA on a sample of 289 bug reports, rated by developers as part of the survey. The participants of our survey also provided 175 comments on hurdles in reporting and resolving bugs. Based on these comments, we discuss several recommendations for better bug tracking systems, which should focus on engaging bug reporters, better tool support, and improved handling of bug duplicates.

Index Terms—Testing and debugging, distribution, maintenance, and enhancement, human factors, management, measurement.

3. Depuración | Distintos pasos

Depuración

- Informar de la incidencia
- **Reproducir**
- Diagnosticar
- Arreglar
- Analizar

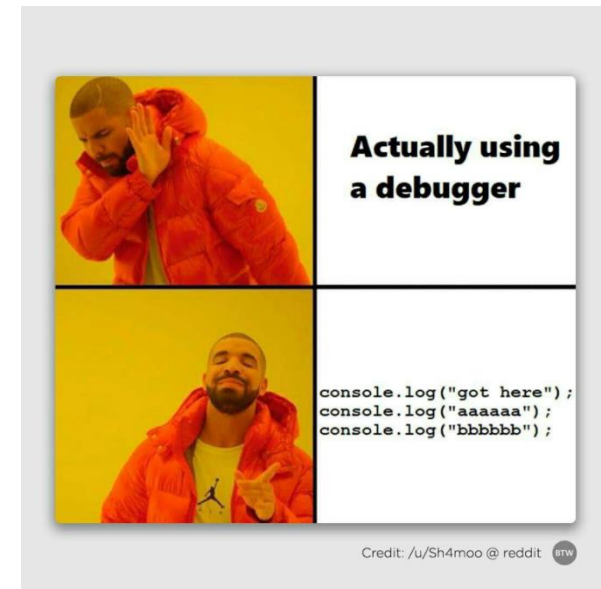
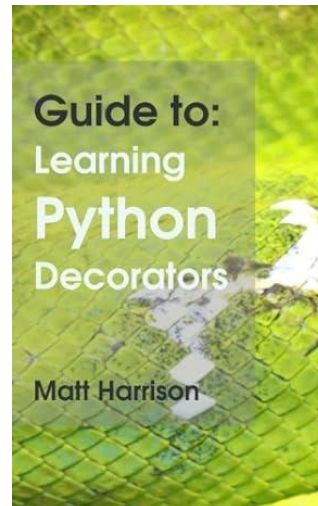
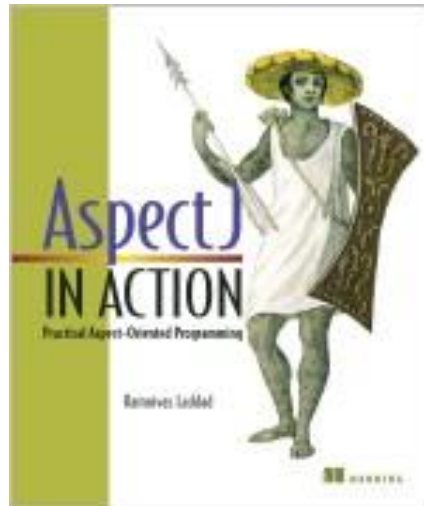
3. Depuración | Reproducir

- Dos escenarios: tengo toda la información, no la tengo
- Algunas recomendaciones si no la tengo:
 - Intentar reproducir sin preguntar
 - Empezar por lo simple
 - Intentar reproducirlo en el mismo entorno (máquinas virtuales)
 - Al menos tres escenarios: desarrollo, preproducción y producción.
 - Usa técnicas de pruebas
 - Si el fallo ha sido detectado usando casos de prueba, reproducir dichos casos de prueba
 - Hacer lo que aparentemente es no determinista, determinista.
 - Usar información de *log* (*log* sí vs *log* no)

¿Logging sí o no?

3. Depuración | ¿logging sí o no?

- ¿Argumentos a favor del logging?
- ¿Argumentos en contra?
 - Ensucia el código impidiendo diferenciar “las hojas de las ramas”
 - Puede tener los mismos problemas que los comentarios: tienen que tener un sincronización con los cambios en el código que según la experiencia, no siempre es así [butcher]
 - Conjetura: “No importa la cantidad de información de *log* que añadas, nunca será la que necesitas”
- Posible solución:



3. Depuración | Distintos pasos

Depuración

- Informar de la incidencia
- Reproducir
- **Diagnosticar**
- Arreglar
- Analizar

3. Depuración | Diagnosticar

- Para diagnosticar software hace falta método y agilidad mental.
- Un experimento debería poder probar algo.
- Un experimento debe introducir un solo cambio.
- Anotar la traza de experimentos
- Pedir una mano al compañero/a
- Ejemplos:

Hipótesis: El error se debe a que nuestro software no soporta la versión del driver de selenium

Experimento: cambiar el driver selenium

Hipótesis: Hay una variable que no se inicializa o se inicializa a null.

Experimento: inicializar esa variable con un valor distinto de null.

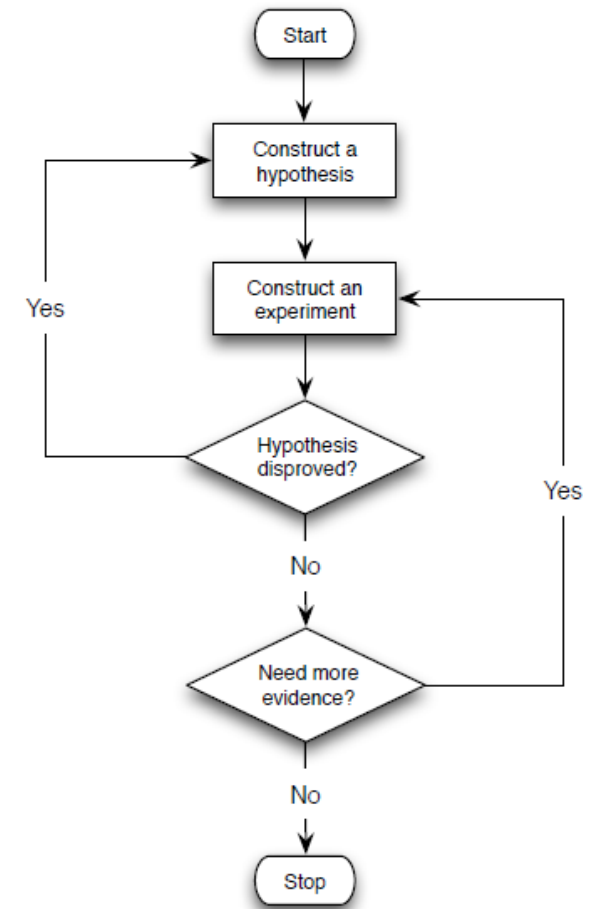


Figure 3.1: A Debugging Method

3. Depuración | Un método para construir hipótesis



Me debugging



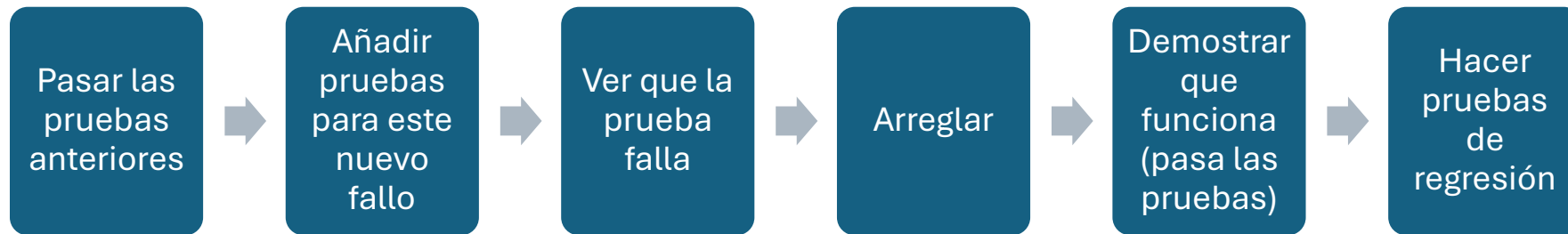
3. Depuración | Distintos pasos

Depuración

- Informar de la incidencia
- Reproducir
- Diagnosticar
- **Arreglar**
- Analizar

3. Depuración | Arreglar

- Arreglar no significa dejar el sistema en el último estado del último experimento
- Primero hay que volver a empezar en el estado inicial.
- Usemos la función *diff* y *cherrypick*
- Pasos para arreglar:



- Subir el cambio a “preproducción” o enviar como un “*patch*” y documentar y gestionar en el “*work item list*”

3. Depuración | Arreglar

IMPORTANTE

- Arreglar las causas no los síntomas: si hay un valor de una variable que parece ser incorrecto, no cambiar el valor de la variable, pensar por qué ha llegado ese valor incorrectamente.
- No intentar arreglar y refactorizar al mismo tiempo

3. Depuración | Distintos pasos

Depuración

- Informar de la incidencia
- Reproducir
- Diagnosticar
- Arreglar
- **Analizar**

3. Depuración | Analizar


- Intentar aprender de lo corregido.

“Tu mejor maestro es tu último error.”

Ralph Nader (1934-?) Activista y abogado de USA

- Las etapas de corrección de errores:
 1. No puede ser
 2. No sucede en mi máquina
 3. No debería pasar
 4. ¿Por qué está pasando?
 5. Ah, ¡ya!
 6. **¿Cómo podía estar funcionando?**

“The six stages of debugging”

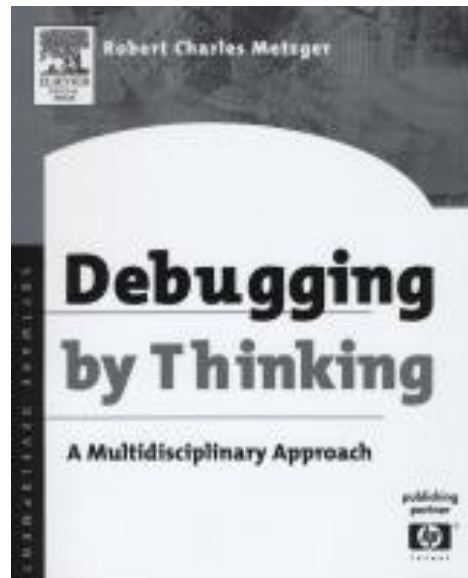
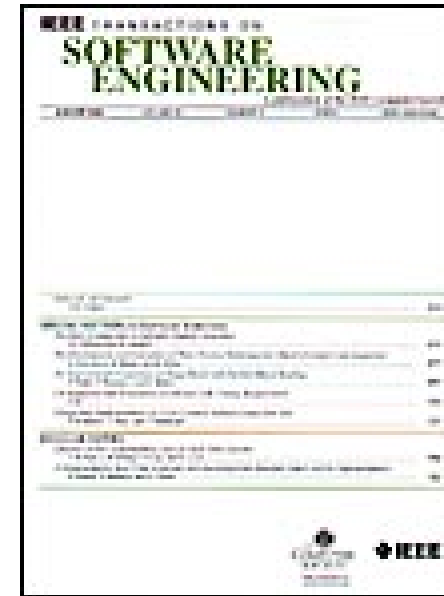
1. **Conceptos básicos**
 2. **Tipos de cambio**
 3. **Depuración**
 4. **Resumen**
 5. **Bibliografía**
- 

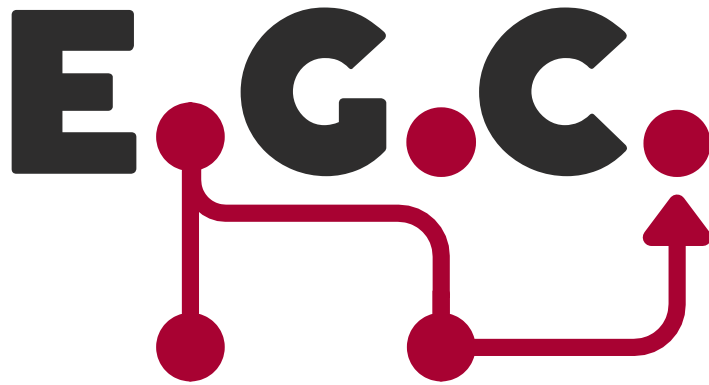
4. Resumen

- ¿Qué hemos aprendido?
 - El cambio es inevitable
 - Si no se gestiona bien puede haber muchos problemas
 - Hay que gestionar los cambios de manera ordenada con estados predeterminados, con tipos de incidencias, con prioridades y con roles
 - Informar de un error es una tarea crucial
 - Para depurar hace falta método y estrategias

1. **Conceptos básicos**
2. **Tipos de cambio**
3. **Depuración**
4. **Resumen**
5. **Bibliografía**

5. Bibliografía





Grado en Ingeniería Informática - Ingeniería del Software

Evolución y Gestión de la Configuración



Escuela Técnica Superior de
Ingeniería Informática

¡Gracias!