

Buenas Prácticas en el Desarrollo de Proyectos

Hipólito Guzmán Miranda
Profesor Contratado Doctor
Departamento de Ingeniería Electrónica
Universidad de Sevilla

¿Por qué este seminario?

- Ya tenéis los conocimientos técnicos para realizar tareas de ingeniería
- Y los conocimientos de gestión para darles forma de proyecto
- Me gustaría daros cierta información que no se veía en la carrera y que hemos descubierto que nos hace la vida más fácil

El Ponente

- Ingeniero de Telecomunicación en 2006
- Profesor Sustituto Interino, 2008
- Doctor Ingeniero de Telecom., 2010
- Profesor Ayudante Doctor, 2011
- Profesor Contratado Doctor, 2015

Grupo de trabajo



- Subgrupo de trabajo dentro del Grupo de Investigación Grupo de Ingeniería Electrónica (GIE, TIC-192)
- GIE: microelectrónica / ASICs, sistemas empotrados, redes de sensores inalámbricas
- Subgrupo: FPGAs, efectos de la radiación en circuitos integrados

Experiencia en Proyectos

- Investigación:
RENASER, RENASER+,
RENASER3, EDELWEISS, ...
- Desarrollo:
INDI, SEA2ME, VISIR, ...
- Internacionales:
FT-UNSHADES2, Soporte FTU1,
VEGAS ...



Buenas prácticas

- Es común que en grupos de trabajo se pierda buena parte del tiempo en ‘overhead’:

“Pásame el fichero”

“¿Cuál de éstas 7 es la última versión?”

“¿Has integrado los cambios de ...?”

“Esto antes funcionaba”

Buenas prácticas

- Y os ocurrirá, si no lo evitáis:

“¿Quién escribió este código? ¿Cómo que ya no trabaja aquí?”

“Ah, sí, vimos ese bug antes pero con todo el lío se nos había olvidado”

“De eso no tenemos copia de seguridad”

“Habrá que rehacer ese trabajo”

Buenas prácticas

- Existe una situación 'ideal' en la que todo marcha bien en el desarrollo
- Si te alejas un paso o dos, puedes vivir con ello
- Si te alejas más...





« hipolito » Dropbox » RENASER_PLUS » NuevaPropuesta » Solicitud

Buscar Solicitud

Organizar Incluir en biblioteca Compartir con Grabar Nueva carpeta

- Favoritos
 - devel
 - Descargas
 - Escritorio
 - Sitios recientes
 - proj
 - Dropbox
- Bibliotecas
 - Documentos
 - Imágenes
 - Música
 - Videos
- Grupo en el hogar
- Equipo
 - OS (C:)
 - bq Curie 2
- Red

| Nombre | Fecha de modifica... | Tipo | Tamaño |
|-------------------------------------|----------------------|--------------------|-----------|
| Memoria_RENASER3_V0.docx | 11/11/2013 0:21 | Documento de Mi... | 41 KB |
| Memoria_RENASER3_V1_MGV.docx | 12/11/2013 14:32 | Documento de Mi... | 73 KB |
| Memoria_RENASER3_V1_MGV_YM.docx | 13/11/2013 19:35 | Documento de Mi... | 79 KB |
| Memoria_RENASER3_V02.docx | 14/11/2013 13:40 | Documento de Mi... | 58 KB |
| Memoria_RENASER3_V10.docx | 17/11/2013 23:00 | Documento de Mi... | 80 KB |
| Memoria_RENASER3_V11.docx | 17/11/2013 11:28 | Documento de Mi... | 71 KB |
| Memoria_RENASER3_V11_YM.docx | 18/11/2013 3:42 | Documento de Mi... | 6.603 KB |
| Memoria_RENASER3_V11_YM_MGV.docx | 18/11/2013 17:57 | Documento de Mi... | 12.683 KB |
| Memoria_RENASER3_V11_YM_MGV_CL.d... | 18/11/2013 21:09 | Documento de Mi... | 12.692 KB |
| Memoria_RENASER3_V20.docx | 19/11/2013 16:55 | Documento de Mi... | 72 KB |
| Memoria_RENASER3_V20_YM.docx | 19/11/2013 18:13 | Documento de Mi... | 71 KB |
| Memoria_RENASER3_V20_YM_CL.docx | 19/11/2013 19:38 | Documento de Mi... | 75 KB |
| Memoria_RENASER3_V21.docx | 20/11/2013 14:46 | Documento de Mi... | 76 KB |
| Memoria_RENASER3_V21_RPP.docx | 20/11/2013 14:44 | Documento de Mi... | 76 KB |
| Memoria_RENASER3_V22_conUSUA.docx | 20/11/2013 14:38 | Documento de Mi... | 81 KB |
| Memoria_RENASER3_V22_conUSUA_AM... | 20/11/2013 19:58 | Documento de Mi... | 93 KB |
| Memoria_RENASER3_V22_conUSUA_MG... | 20/11/2013 18:21 | Documento de Mi... | 85 KB |
| Memoria_RENASER3_V22_conUSUA_MG... | 20/11/2013 20:30 | Documento de Mi... | 108 KB |
| Memoria_RENASER3_V22_conUSUA_YM... | 20/11/2013 17:39 | Documento de Mi... | 83 KB |
| Memoria_RENASER3_V23.docx | 20/11/2013 20:29 | Documento de Mi... | 87 KB |

30 elementos



Buenas prácticas

- Control de versiones
- Bug tracking
- Generadores de documentación
- Builds en un paso
- Builds diarios
- Backups automáticos
- Estrategia de verificación
- Planificación realista
- Condiciones de trabajo

Buenas prácticas

- Control de versiones
- Bug tracking
- Generadores de documentación
- Builds en un paso
- Builds diarios
- Backups automáticos
- Estrategia de verificación
- Planificación realista
- Condiciones de trabajo

Control de Versiones...

... no es usar
Dropbox!



¿Qué es?

Un sistema de control de versiones es un

- software que permite gestionar los cambios que se van realizando sobre un conjunto de ficheros
- de forma que se puedan recuperar en cualquier momento versiones válidas del código
- etiquetar versiones específicas
- y permitir el trabajo simultáneo de diferentes desarrolladores.

No sólo en proyectos de SW

- Diseño de HW: VHDL, PCBs
- Sistemas empotrados y redes de sensores distribuidos mezclan componentes HW y SW
- Ficheros de texto plano y LaTeX
- Cualquier clase de fichero/código

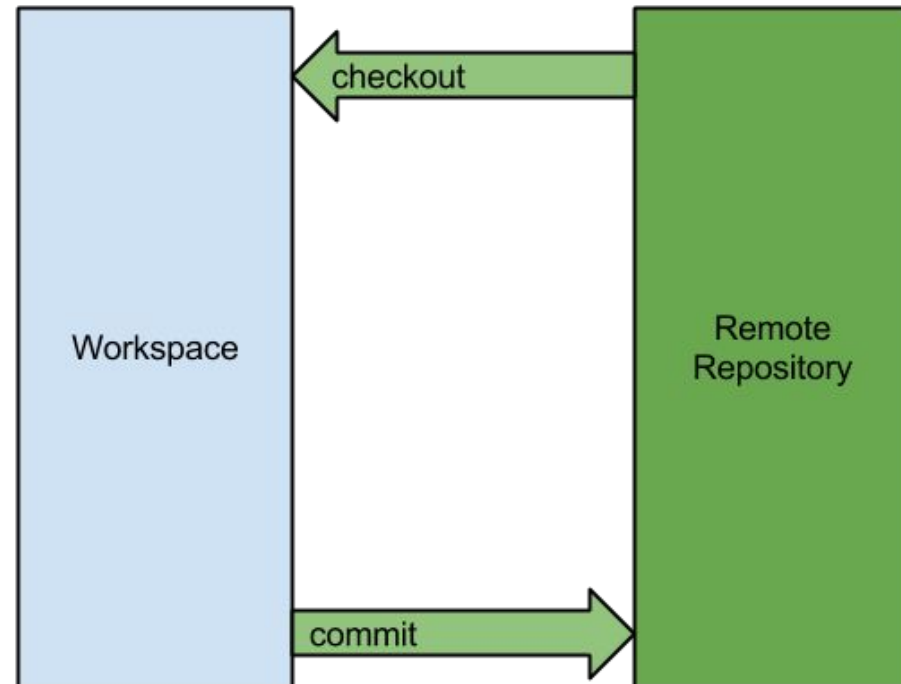
- ¡No se deben añadir binarios que se puedan generar a partir del código!

¿Cómo funciona?

- Existe un repositorio de código al que se van añadiendo conjuntos de cambios (**commits**) del código
- Un **commit** puede ser “añadí la funcionalidad Z” o “Arreglé el bug #248”
 - Puede implicar cambios en varios ficheros
- Todos los desarrolladores añaden sus cambios al repositorio

¿Cómo funciona?

- **Commit** añade cambios al repositorio
- **Checkout** descarga cambios del repositorio
- Puedes hacer **checkout** de una versión específica



Ventajas

- Historial de cambios
- Los proyectos que usan control de versiones no suelen perder código (sobre todo si es control de versiones distribuido)
- Posibilidad de revertir errores

Podréis borrar ese código que tenéis comentado desde hace semanas ;)



Centralizado vs Distribuido

Centralizado: svn



- **Commit y checkout** directamente al repositorio remoto
- Requiere conexión de red para poder trabajar!

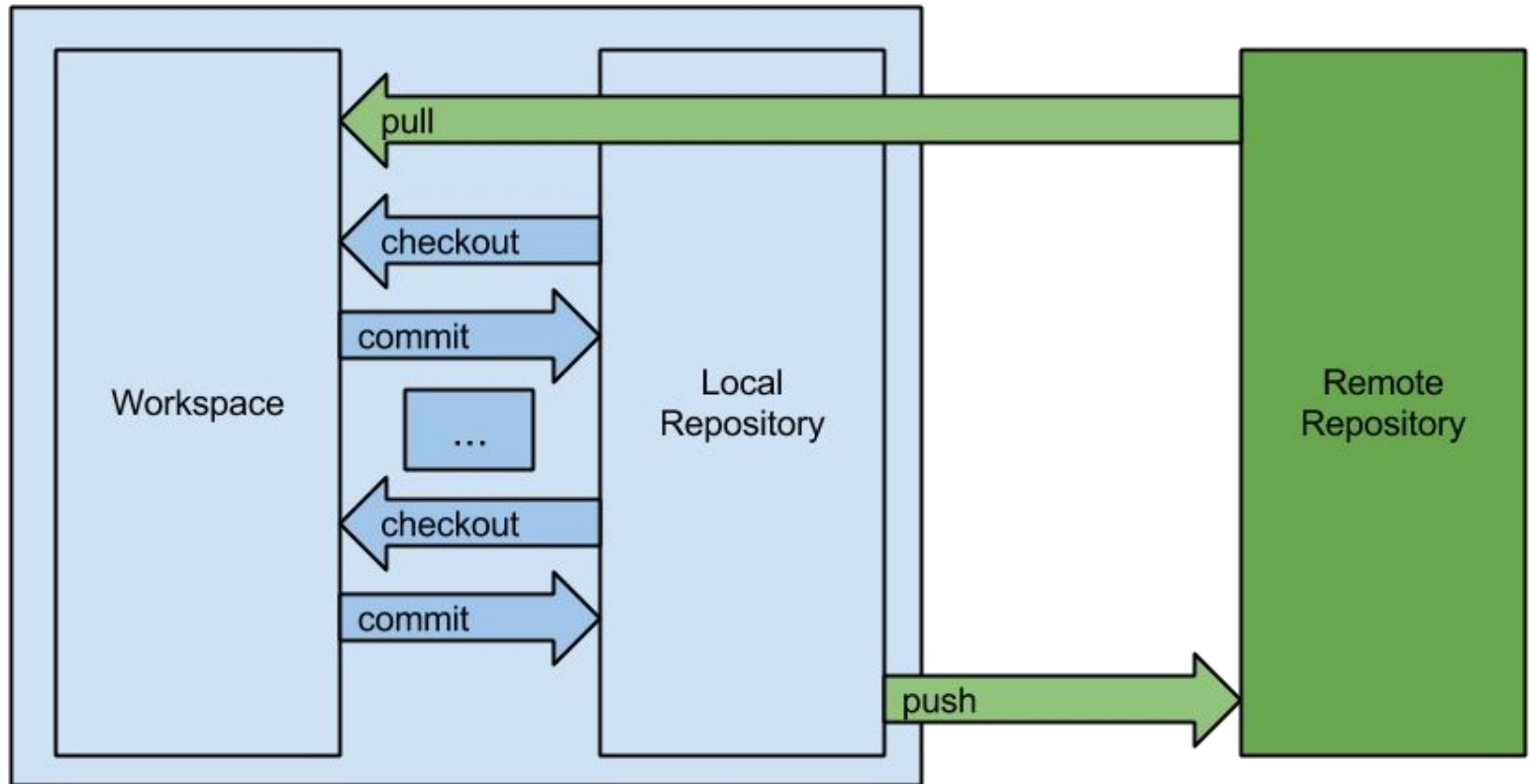
Distribuido:



- **Clone:** copia *completa* del repo remoto en local
- **Commit, checkout** trabajan con el repo local
- **Push, pull** trabajan con el repo remoto

Distribuido

Pull, checkout/commit, Push



Archivo Editar Vista Ayuda

- **master** — **remotes/origin/master** Added doxygen configuration file
- datagen now working correctly
- Added Patricio's tests for fadapt
- Added sample file for test I/O
- Merge branch 'master' of freyja.us.es:/var/git/edelweiss
- Merge branch 'master' of freyja.us.es:/var/git/edelweiss
- datagen 1st partially working version
- Added new module bit2syb to src/
- Added fadapt.vhd to src/
- Moved demonstrator testbenches to concepts/
- Added a preliminary test for datagen
- Added sim/tb_datagen.vhd
- Working on datagen
- Running in process...
- First classification of Edelweiss project
- Moved std_logic_hard.vhd to lib/

Hipolito Guzman <hipolito@gie.esi.us.es>
 Hipolito Guzman <hipolito@gie.esi.us.es>
 Hipolito Guzman <hipolito@gie.esi.us.es>
 Javier Barrientos <jbr@gie.esi.us.es>
 Javier Barrientos <jbr@gie.esi.us.es>
 Hipolito Guzman <hipolito@gie.esi.us.es>
 Hipolito Guzman <hipolito@gie.esi.us.es>
 Javier Barrientos <jbr@gie.esi.us.es>
 Javier Barrientos <jbr@gie.esi.us.es>
 Hipolito Guzman <hipolito@gie.esi.us.es>
 Hipolito Guzman <hipolito@gie.esi.us.es>
 Hipolito Guzman <hipolito@gie.esi.us.es>
 Hipolito Guzman <hipolito@gie.esi.us.es>
 Hipolito Guzman <hipolito@gie.esi.us.es>
 Javier Barrientos <jbr@gie.esi.us.es>
 Javier Barrientos <jbr@gie.esi.us.es>
 Hipolito Guzman <hipolito@gie.esi.us.es>

SHA1 ID:

ff42e349e90415f41fc2e81aeealce10b8febfc

Row

10/

22

Buscar

<<

>>

revisión que contiene:

Exacto

Todos los campos

Buscar

◆ Parche ◆ Árbol

◆ Diferencia ◆ Versión antigua ◆ Versión nueva

Líneas de contexto: 3

 Ignora cambios de espaciado

Line diff

Autor: Hipolito Guzman <hipolito@gie.esi.us.es> 2013-11-21 10:58:00
 Committer: Hipolito Guzman <hipolito@gie.esi.us.es> 2013-11-21 10:58:00
 Padre: [5ddfd58e50123a5dd62bcfd857bed94bfe79ded1](#) (Added a preliminary test for datagen)
 Hija: [91f3690fa6823be4084b1d03a76e1761199a5a93](#) (datagen 1st partially working version)
 Hija: [1ba2a3f45bc1e479192fbd8c48fe1ef9159b7ca0](#) (Added fadapt.vhd to src/)
 Rama: [master](#), [remotes/origin/master](#)
 Sigue-a:
 Precede-a:

Moved demonstrator testbenches to concepts/

----- concepts/tb_data_io.vhd -----

```
similarity index 100%
rename from sim/tb_data_io.vhd
rename to concepts/tb_data_io.vhd
```

----- concepts/tb_hello_world.vhd -----

```
similarity index 100%
rename from sim/tb_hello_world.vhd
rename to concepts/tb_hello_world.vhd
```

----- concepts/tb_simplefileio.vhd -----

```
similarity index 100%
rename from sim/tb_data_gen.vhd
rename to concepts/tb_simplefileio.vhd
```

Comentarios

concepts/tb_data_io.vhd
 concepts/tb_hello_world.vhd
 concepts/tb_simplefileio.vhd
 sim/datagen_test.vhd
 sim/tb_data_gen.vhd
 sim/tb_data_io.vhd
 sim/tb_hello_world.vhd

Necesito mi propio servidor?

Si no tienes un servidor online 24/7 en el que instalar git:

- Github (repos públicos gratuitos)
- Gitlab
- Bitbucket

Lee y *entiende* los términos y condiciones!

Se consciente de que tus datos están en 'hierro' que no es tuyo -> no los uses para proyectos realmente confidenciales

Desarrolladores sin \$\$, opciones:

- Repo local + copias de seguridad (sólo 1 persona)
- Raspberry Pi + Git + Dynamic DNS

Otras herramientas

“No importa cuál uséis, mientras uséis alguno”

- David Merodio Codinachs, European Space Agency

Mercurial

CVS

Bitkeeper

...

http://en.wikipedia.org/wiki/List_of_revision_control_software

Buenas prácticas

- Control de versiones
- Bug tracking
- Generadores de documentación
- Builds en un paso
- Builds diarios
- Backups automáticos
- Estrategia de verificación
- Planificación realista
- Condiciones de trabajo

Bug Tracking

**Mis aplicaciones
no pasan nunca
de la version 0.1**



**...porque no existen nuevas
funcionalidades que pudieran
tener, ni bugs que corregir**

¿Qué es?

Un sistema de bug tracking o seguimiento de errores es

- una aplicación software
- que permite anotar y seguir la evolución de fallos o bugs en sistemas en desarrollo
- ya sean software o hardware.

También podemos anotar funcionalidades por añadir

¿Por qué usarlo?

- En mitad de un desarrollo, en el día a día caben unos 2-3 bugs en una cabeza humana
- Sistematizar la 'captura' de los bugs para que no se olviden
- Permitir a los testers indicar bugs a los desarrolladores
- (Posibilidad de) abrir la herramienta a clientes para que informen bugs

¿Cómo se usa?

En un buen bug report es **INDISPENSABLE** que conste:

1. Pasos para reproducirlo
2. Qué esperabas que ocurriera
3. Qué ocurrió realmente

¿Cómo se usa?

- Normalmente tienen una interfaz web (algunos incluso apps de Android), para conectarse a la Base de Datos donde están almacenados los issues
- Vida de un bug:
 - Nuevo
 - Admitido
 - Confirmado
 - Resuelto
 - Cerrado

¿Cómo se usa?

- Los equipos de trabajo pueden poner reglas internas a la evolución de los bugs
- Las herramientas suelen ser configurables para forzar su cumplimiento
- Ejemplos:
 - Que sólo puedan abrir bugs los testers
 - Que sólo pueda cerrar un bug quien lo ha abierto
 - Que todo bug tenga siempre una persona asignada (responsable)

Logged in as: *hipolito* (Hipólito Guzmán - developer)

2013-11-28 03:41 CET

[Main](#) | [My View](#) | [View Issues](#) | [Report Issue](#) | [Change Log](#) | [Roadmap](#) | [My Account](#) | [Logout](#)

Issue #

Jump

Recently Visited: [000034](#)**Assigned to Me (Unresolved) [^] (1 - 6 / 6)**

- [0000164](#) Doxygen comments are missing from most of the embedded software code
- 2013-04-15 18:49
- [0000162](#) D4 document reports obsolete firmware command set
- 2013-04-15 18:32
- [0000031](#) Off-by-one error in get_last_input_vector
- 2013-04-02 00:40
- [0000019](#) Start-up sequence hangs randomly
- 2013-04-02 00:10
- [0000094](#) FTU2 run too slow for big vectors
- 2013-03-28 19:22
- [0000032](#) damage_per_run does not seem to stop a run in the Motherboard
- 2013-03-05 08:55

Unassigned [^] (1 - 10 / 18)

- [0000215](#) Sometimes FTU2 board hangs
- 2013-11-20 17:54
- [0000212](#) Wanted: FTU2nXX/UNKNOWN should be valid devices for "reboot" purposes
- 2013-11-04 13:59
- [0000209](#) UFF/TNT/FW do not check that target_cycles is between correct bounds
- 2013-10-22 14:36
- [0000186](#) Wanted: detect from .bit file if SelectMAP pins have persisted
- 2013-10-07 09:46
- [0000202](#) Wanted: check campaign options during setup
- 2013-05-28 13:22
- [0000130](#) If USB FIFO in CFPGA is full, FTU2 boards never receive REBOOT commands
- 2013-05-28 12:46
- [0000183](#) Wanted: commit version in firmware
- 2013-05-06 19:46
- [0000179](#) A small counter example that hungs de system
- 2013-05-06 19:21
- [0000161](#) Wanted: emuVSsim
- 2013-04-15 18:51
- [0000163](#) Doxygen comments are missing from most of the firmware code
- 2013-04-15 18:37

Reported by Me [^] (1 - 10 / 54)

- [0000215](#) Sometimes FTU2 board hangs
- 2013-11-20 17:54
- [0000212](#) Wanted: FTU2nXX/UNKNOWN should be valid devices for "reboot" purposes
- 2013-11-04 13:59
- [0000210](#) readb fpga_in/out returns empty strings
- 2013-10-28 15:29
- [0000211](#) Jump command ignores total number of clock cycles
- 2013-10-28 15:28
- [0000209](#) UFF/TNT/FW do not check that target_cycles is between correct bounds
- 2013-10-22 14:36

Resolved [^] (1 - 10 / 34)

- [0000214](#) Instrumentalizer: Variable names change
- 2013-11-20 11:25
- [0000213](#) Instrumentalizer: Impact emulation source model change
- 2013-11-07 13:53
- [0000210](#) readb fpga_in/out returns empty strings
- 2013-10-28 15:29
- [0000211](#) Jump command ignores total number of clock cycles
- 2013-10-28 15:28
- [0000208](#) Brackets [] break register names
- 2013-10-22 14:34

View Issue Details [[Jump to Notes](#)] [[Send a reminder](#)] [<<] [>>] [[Issue History](#)] [[Print](#)]

| ID | Project | Category | View Status | Date Submitted | Last Update |
|--------------------|-------------|-------------------|-------------|------------------------|------------------|
| 0000202 | FTUNSHADES2 | | public | 2013-05-28 13:22 | 2013-05-28 13:22 |
| Reporter | hipolito | | | | |
| Assigned To | | | | | |
| Priority | normal | Severity | feature | Reproducibility | have not tried |
| Status | feedback | Resolution | open | | |
| Platform | | OS | | OS Version | |

Summary 0000202: Wanted: check campaign options during setup

Description If an error in campaign options is found while running the campaign, the error will not be output to the console (it will instead be written in the campaign log).
A set of assertions for the campaign setup stage will allow to output information about errors in campaign options to the console instead of the campaign log.

Tags No tags attached.

Attach Tags (Separate by ",") Existing tags

Attached Files

Relationships

New relationship

Current issue

Upload File

Select File (Maximum size: 2,097k) No se ha seleccionado ningún archivo

Users monitoring this issue

User List There are no users monitoring this issue.

Username

Notes

There are no notes attached to this issue.

Add Note

Issue History

| Date Modified | Username | Field | Change |
|------------------|--------------------------|-------------------------|----------------------|
| 2013-03-29 18:56 | hipolito | Severity | major => minor |
| 2013-03-29 18:58 | luis | Note Added: 0000106 | |
| 2013-03-29 18:58 | luis | Assigned To | => luis |
| 2013-03-29 18:58 | luis | Status | new => resolved |
| 2013-03-29 18:58 | luis | Resolution | open => fixed |
| 2013-03-29 18:59 | luis | Note Edited: 0000106 | |
| 2013-03-29 19:00 | hipolito | Status | resolved => closed |
| 2013-03-30 00:44 | hipolito | New Issue | |
| 2013-03-30 00:44 | hipolito | File Added: slashes.png | |
| 2013-03-30 00:59 | luis | Status | resolved => feedback |
| 2013-03-30 00:59 | luis | Resolution | fixed => reopened |
| 2013-03-30 00:59 | luis | Status | feedback => resolved |
| 2013-03-30 00:59 | luis | Resolution | reopened => fixed |

¿Cómo se usa?

- Otra buena práctica es arreglar los bugs antes de seguir con el desarrollo
- “Fix bugs before writing new code” es aplicable no sólo a SW, sino también a HW
- Realmente, aplicable a cualquier desarrollo

Issue Tracking

Cualquier bug tracker puede usarse para seguimiento de:

- Nuevas funcionalidades (features)
- Wishlist
- Documentación que falta por hacer
- Código que falta por comentar
- etc ...

Una opción interesante: poblar el Issue Tracker con las tareas y entregables el día del kick-off del proyecto

Otras herramientas

Bugzilla

FogBugz

Bug tracker de Trac

...

http://en.wikipedia.org/wiki/Comparison_of_issue-tracking_systems

Buenas prácticas

- Control de versiones
- Bug tracking
- Generadores de documentación
- Builds en un paso
- Builds diarios
- Backups automáticos
- Estrategia de verificación
- Planificación realista
- Condiciones de trabajo



¿Qué son?

Un generador de documentación es

- Una herramienta software
- que genera documentación técnica automáticamente
- a partir de código fuente especialmente comentado

Ejemplo:

C

```

hipolito@challenger: ~/devel/ftu2/fw/mb/ppc440/src
/*****
 * \brief Data for each of the daughter boards.
 *****/
struct mbppc_daughter
{
    int id;                ///< Identifier code read from daughter
    struct ftu2_pcie_dev dev;    ///< PCIe device for communication
    int frame;            ///< Frame currently selected
    u32 last_output[32];    ///< Last output vectors read
};

/*****
 * \brief Configuration for target FPGA
 *****/
struct mbppc_config
{
    u8 * buf;                ///< Pointer to data
    size_t len;            ///< Size of data
};

/*****
 * \brief I/O vectors for target FPGA
 *****/
struct mbppc_vectors
{
    u8 * buf;                ///< Pointer to data
    size_t len;            ///< Size of data

    u8 i_mask[VECTOR_SIZE];    ///< Mask for input vectors
    size_t i_size;        ///< Size of input vectors in 32-bit words
    u8 * i_ptr;                ///< Pointer to the current vector must belong in [
buf,buf+len)

    u8 o_mask[VECTOR_SIZE];    ///< Mask for output vectors
    size_t o_size;        ///< Size of output vectors in 32-bit words.
    u8 * o_ptr;                ///< Pointer to the current vector must belong in [
buf+len,...)

    int count;                ///< Total number of cycles in run
    int remaining;            ///< Remaining cycles of compressed vector
};

/*****
 * \brief Campaign batches
 *****/

```

```
FTUNSHADES 2: Vector x
file:///home/hipolito/devel/ftu2/fw/html/c/html/group_vector.html
ftu.us.es/uff/ Google Keep Mantis Horario Poli 2o zipi roundcube Reserva CDC git - la guía sen... reformauniversi... Plan Propio US Git - Book
Default channels.
enum {
    FTU2_VECTOR_SET_STIMULI_MASK = 0x01,
    FTU2_VECTOR_SET_RESPONSE_MASK = 0x00,
    FTU2_VECTOR_SET_STIMULI_SIZE = 0x02,
    FTU2_VECTOR_SET_RESPONSE_SIZE = 0x03,
    FTU2_VECTOR_SET_CYCLES = 0x04,
    FTU2_VECTOR_RUN = 0x05,
    FTU2_VECTOR_FLUSH_VECTORS = 0x06
}
```

Initialization

ftu2_vector_init initializes the device pointed by **dev**, using **ctrl_dev_id** and **data_dev_id** as the identifiers for the control and data GPIOs respectively.

```
static int ftu2_vector_init (struct ftu2_vector_dev *dev, int ctrl_dev_id, int data_dev_id)
```

Status

The following methods check certain flags on the stream pointed by **dev**.

ftu2_vector_done indicates that the vector module has finished the current operation

ftu2_vector_busy indicates that the input vector module has not finished

ftu2_vector_status_full the stream is full and data may not be written.

ftu2_vector_status_almost_full less than **FTU2_VECTOR_OUTPUT_BUFFER_SIZE** bytes may be written to the device. When this happens, data must be written element by element and the full flag checked before each one.

empty and **almost_empty** are here for debugging purposes, will be removed

```
#define ftu2_vector_flags(dev) (ftu2_fifo_read (&(dev)->ctrl))
#define ftu2_vector_busy(dev) ((ftu2_fifo_read (&(dev)->ctrl) & 0x01) != 0)
#define ftu2_vector_done(dev) ((ftu2_fifo_read (&(dev)->ctrl) & 0x02) != 0)
#define ftu2_vector_full(dev) ((ftu2_fifo_read (&(dev)->ctrl) & 0x04) != 0)
#define ftu2_vector_almost_full(dev) ((ftu2_fifo_read (&(dev)->ctrl) & 0x08) != 0)
#define ftu2_vector_empty(dev) ((ftu2_fifo_read (&(dev)->ctrl) & 0x10) != 0)
#define ftu2_vector_almost_empty(dev) ((ftu2_fifo_read (&(dev)->ctrl) & 0x20) != 0)
```

Input/output

The fast methods are just that: fast, but at the cost of every known safety measure.

They are more or less guaranteed to eventually crash the system if used directly, and are called mostly by the other input/output methods.

Ejemplo: VHDL

```

hipolito@challenger: ~/devel/ftu2/fw/common/src
--! @addtogroup common
--! @{
--! @defgroup common_edgedetector Edge Detector
--! @{
--! @brief Outputs single-cycle pulses when input goes from low to high.

library IEEE;           --! Use IEEE standard definitions library
use IEEE.STD_LOGIC_1164.ALL; --! Use std_logic* signal types

--! @brief Detects edges on insignal, outputs a single-cycle pulse (clocked by clk)
. Reset is active low.
entity edgedetector is
  Port (
    clk : in STD_LOGIC;           --! Clock
    reset_N : in STD_LOGIC;       --! Active low asynchronous reset
    insignal : in STD_LOGIC;      --! Input signal
    pulse : out STD_LOGIC        --! A single-cycle active high pulse three clk cycles
    after insignal goes from low to high
  );
end edgedetector;

--! @brief Implementation is done by using three Flip-flops
architecture threeFFs of edgedetector is

  signal r1      : std_logic;    --! First delay for insignal
  signal r2      : std_logic;    --! Second delay for insignal
  signal p_output : std_logic;    --! Detection of rising edge in insignal
  signal output   : std_logic;    --! Extra delay to avoid glitches

begin

  pulse <= output;

  --! @brief Synchronizes data to clk
  sinc: process(reset_N, clk)
  begin
    if(reset_N='0')then
      r1 <= '0';
      r2 <= '0';
      output <= '0';
    elsif(clk='1' and clk'event)then
      r1 <= insignal;
      r2 <= r1;
      output <= p_output;
  
```

FTUNSHADES 2

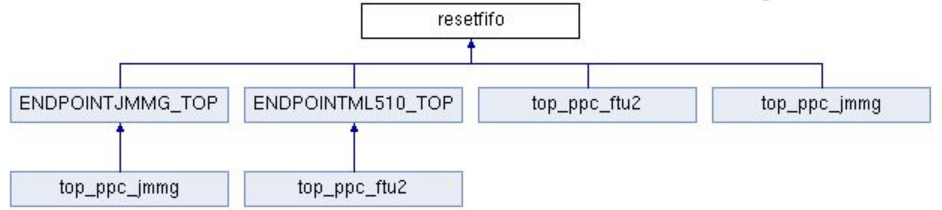
Generics Ports

Counts cycles of global reset with global clock, outputs a synchronous reset which lasts a number of clock cycles. [More...](#)

resetfif Entity Reference

Common

Inheritance diagram for resetfif:



Entities

counting architecture
 Implementation is done counting clock cycles. [More...](#)

Generics

ACTIVE_RST_VALUE **std_logic := '0'**
 Active vale of input reset.

RST_CYCLE_DURATION **integer := 10**
 Duration in input clock cycles of output reset signal.

Ports

sys_clk **in std_logic**
 Input clock.

rst **in std_logic**
 Input reset (configurable polarity)

locked **in std_logic**
 Indicates that the second FIFO clock is stable.

rst_fifo **out std_logic**
 Generated reset for FIFOs.

Otras herramientas

Natural Docs, ROBOdoc, Javadoc, doc-o-matic, JSdocs, ...

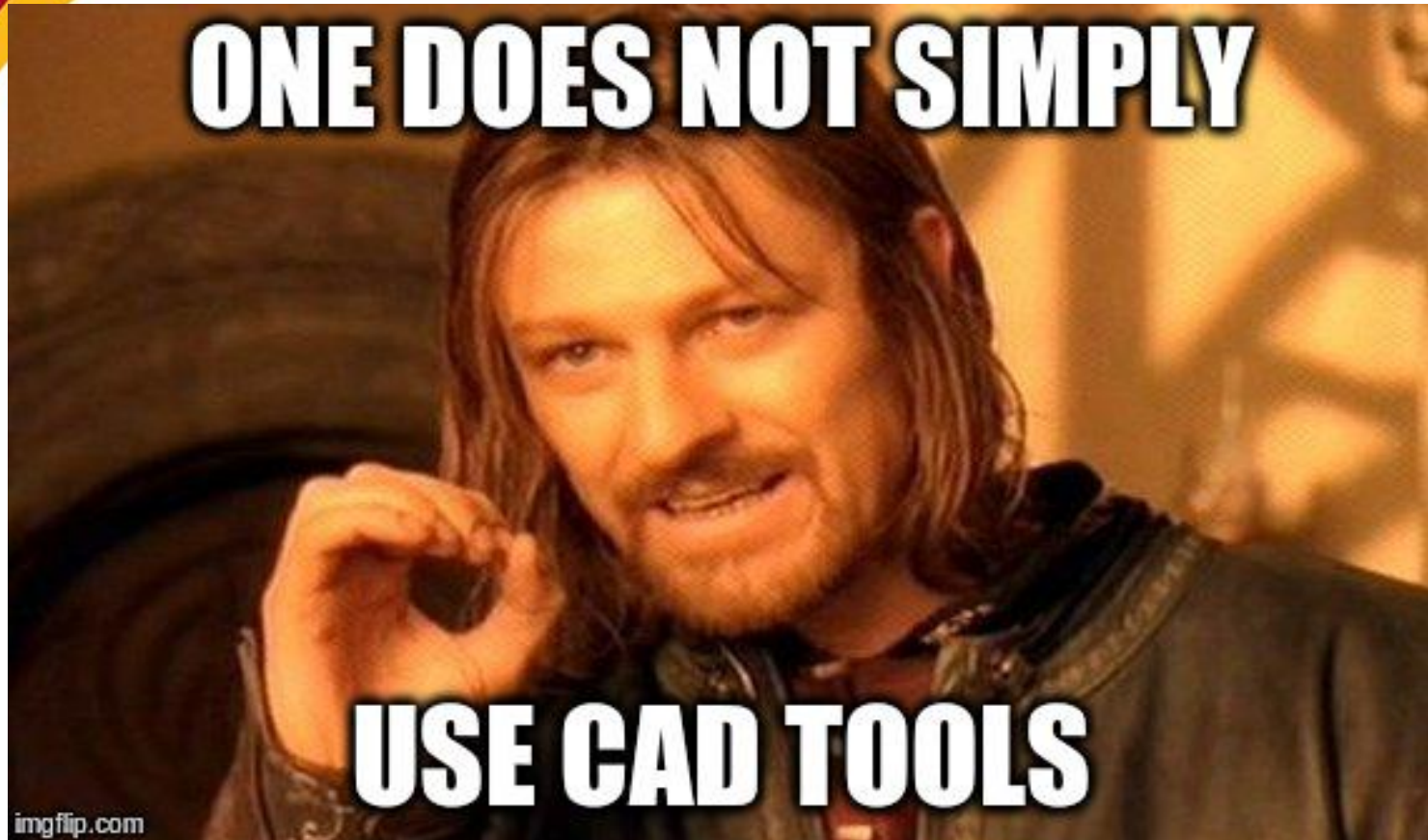
No todos soportan todos los lenguajes!

http://en.wikipedia.org/wiki/Comparison_of_documentation_generators

Buenas prácticas

- Control de versiones
- Bug tracking
- Generadores de documentación
- Builds en un paso
- Builds diarios
- Backups automáticos
- Estrategia de verificación
- Planificación realista
- Condiciones de trabajo

One-step builds



¿Qué son?

Cuando hablo de build, no sólo me refiero a compilación de SW

Desde la última versión del source a un ejecutable / tapeout / FPGA PROM file / etc

- Síntesis e implementaciones (FPGA)
- Design Rule Check (PCB, ASIC)
- SW empotrado
- Ficheros para impresión 3D
- ...

¿Por qué es importante poder hacer un 'build' en un único paso?

- Si hay más de un paso, es sensible a errores humanos
- No digamos si hay 20, con un IDE de por medio
- Si mañana tienes un cliente nuevo, tienes que dedicar horas de desarrollador a hacer un build

¿Cómo se hace?

- Olvida el IDE
- Línea de comandos
 - Scripting (bash)
- Muchas herramientas ofrecen posibilidad de scripting
 - En cada herramienta es diferente
- Una vez que puedes llamar a las herramientas desde línea de comandos:
 - Script / Makefile / CMake


```
hipolito@challenger:~/devel/ftu2/fw/mb/build$ cmake .. && make project && make
```

```
-- Configuring done
```

```
-- Generating done
```

```
-- Build files have been written to: /home/hipolito/devel/ftu2/fw/mb/build
```

```
Scanning dependencies of target project
```

```
[100%] Generating project.xise
```

```
. /opt/Xilinx/12.1/ISE_DS/EDK/settings32.sh
```

```
. /opt/Xilinx/12.1/ISE_DS/ISE/settings32.sh
```

```
. /opt/Xilinx/12.1/ISE_DS/PlanAhead/settings32.sh
```

```
. /opt/Xilinx/12.1/ISE_DS/
```

```
INFO:HDLCompiler:1061 - Parsing VHDL file
```

```
"/home/hipolito/devel/ftu2/fw/mb/src/ftinterf.vhd" into library work
```

```
INFO:HDLCompiler:1061 - Parsing VHDL file
```

```
"/home/hipolito/devel/ftu2/fw/mb/src/mig_in_padding.vhd" into library work
```

```
INFO:HDLCompiler:1061 - Parsing VHDL file
```

```
"/home/hipolito/devel/ftu2/fw/mb/src/mig_out_padding.vhd" into library work
```

```
INFO:HDLCompiler:1061 - Parsing VHDL file
```

```
"/home/hipolito/devel/ftu2/fw/mb/src/mig_ui.vhd" into library work
```

```
INFO:HDLCompiler:1061 - Parsing VHDL file
```

```
"/home/hipolito/devel/ftu2/fw/mb/src/ml510_endpoint.vhd" into library work
```

```
INFO:HDLCompiler:1061 - Parsing VHDL file
```

```
"/home/hipolito/devel/ftu2/fw/mb/src/top_mig.vhd" into library work
```

```
INFO:HDLCompiler:1061 - Parsing VHDL file
```

```
"/home/hipolito/devel/ftu2/fw/mb/src/top_ppc_ftu2.vhd" into library work
```

```
INFO:HDLCompiler:1061 - Parsing VHDL file
```

```
"/home/hipolito/devel/ftu2/fw/mb/src/uart_fsm.vhd" into library work
```

```
INFO:ProjectMgmt:656 - Parsing design hierarchy completed successfully.
```

```
Started : "Regenerate Core".
```

```
Release 12.1 - Xilinx CORE Generator M.53d (lin)
```

```
Copyright (c) 1995-2010 Xilinx, Inc. All rights reserved.
```

```
All runtime messages will be recorded in
```

```
/home/hipolito/devel/ftu2/fw/mb/build/xco/coregen.log
```

Integración Continua

Continuous integration:

- Otra forma de decir build + test automáticamente, a cada commit

Herramientas como Jenkins o Gitlab CI

Buenas prácticas

- Control de versiones
- Bug tracking
- Generadores de documentación
- Builds en un paso
- Builds diarios
- Backups automáticos
- Estrategia de verificación
- Planificación realista
- Condiciones de trabajo

¿Qué son?

Consiste en hacer

- una compilación automática de tu diseño
- todos los días (o noches) a la misma hora
- ya que estás, compila la documentación
- y ponla online / en la intranet
- y si tienes tests, ¡pásalos!

¿Por qué hacer una compilación automática todos los días?

- Sabes automáticamente si alguien se ha cargado el repositorio
- Podría haber un fichero vital para el proyecto que NO esté en el repositorio
- Tu compilación podría funcionar porque tienes un fichero temporal intermedio, pero no funcionaría desde cero
- Que en tu máquina funcione no signifique que funcione siempre

¿Cómo se hace?

- Debe hacerse en un directorio limpio, sin ficheros temporales de compilaciones/implementaciones/simulaciones anteriores
- Si ya usas control de versiones y puedes hacer one-step build es sencillo:
 - `rm -rf project/`
 - `git clone usuario@servidor:/var/git/project.git`
 - `cd project`
 - `make`
- Usad cron para que ocurra todos los días

Buenas prácticas

- Control de versiones
- Bug tracking
- Generadores de documentación
- Builds en un paso
- Builds diarios
- Backups automáticos
- Estrategia de verificación
- Planificación realista
- Condiciones de trabajo

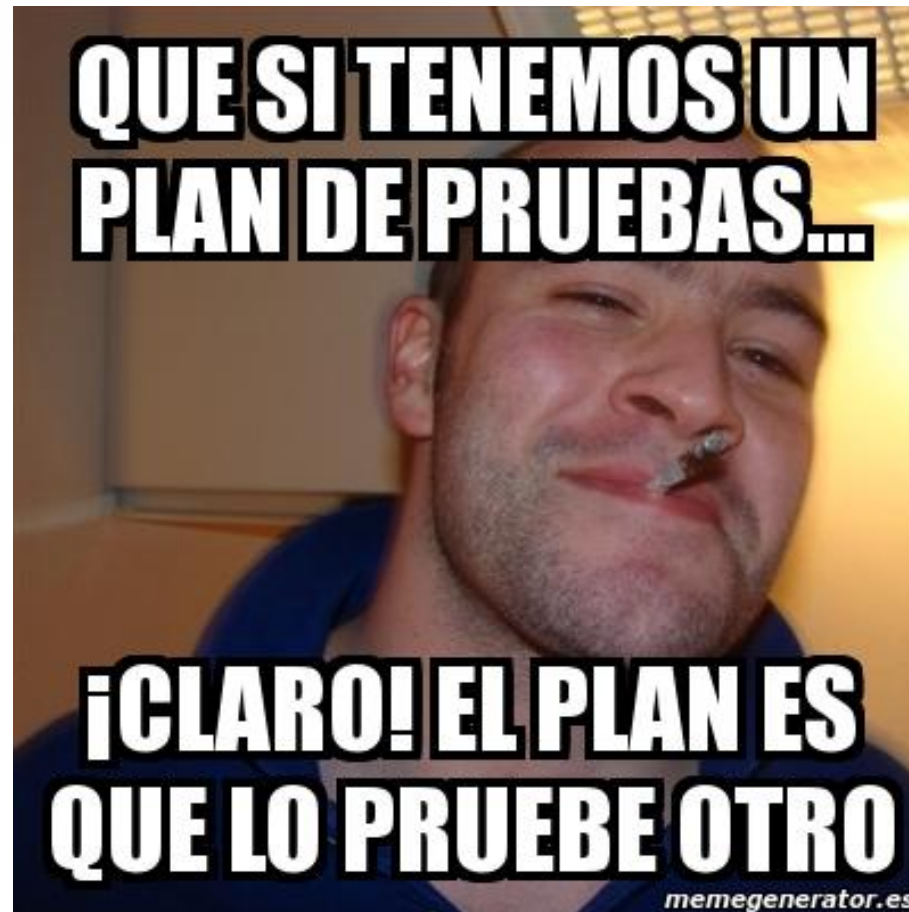
¿Por qué?

- Es más tarea del administrador de sistemas que del desarrollador
- Pero el gestor de proyecto debe estar pendiente de que el backup exista
- Un backup no vale **nada** si no se puede recuperar!
- Backups online: cron + rsync
- Opcional: Backups offline (cinta), para mayor redundancia

Buenas prácticas

- Control de versiones
- Bug tracking
- Generadores de documentación
- Builds en un paso
- Builds diarios
- Backups automáticos
- Estrategia de verificación
- Planificación realista
- Condiciones de trabajo

Típico plan de pruebas:

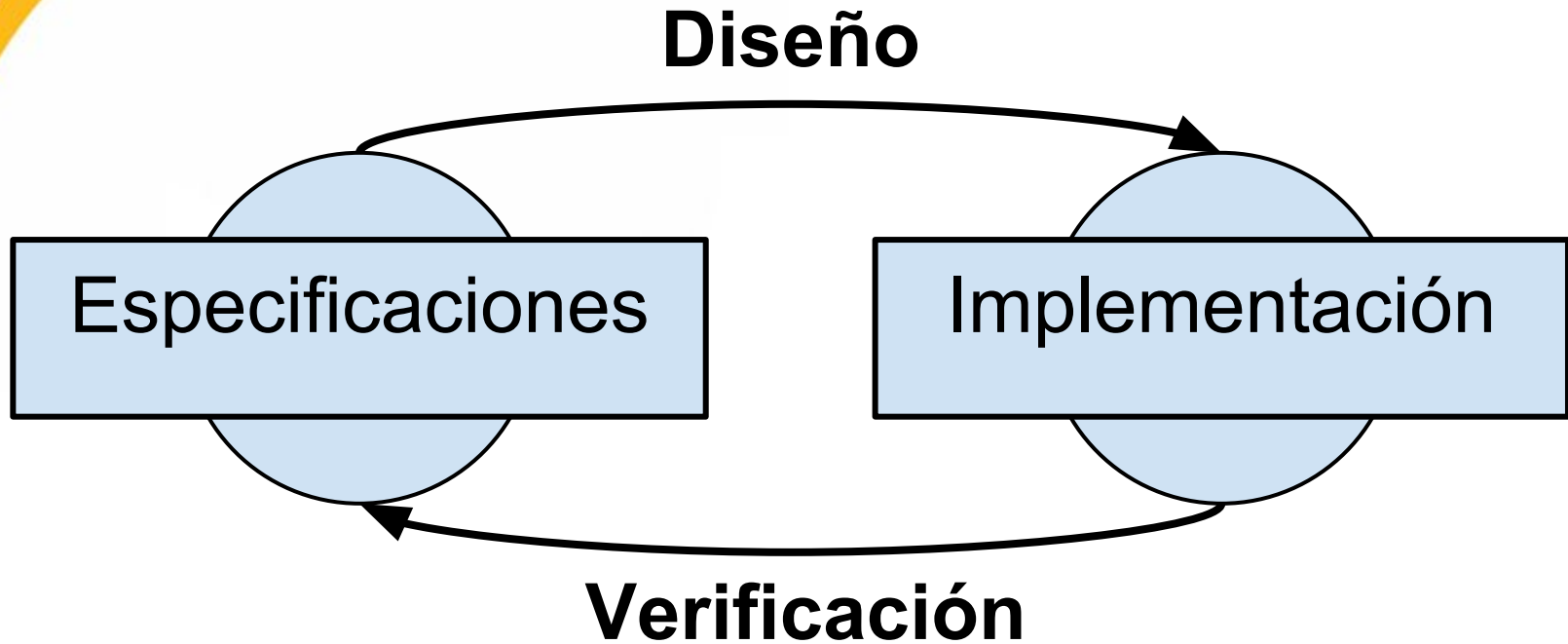


¿Cómo sabes que lo que has hecho funciona?

- En proyectos de mínima complejidad, más del 50-60% del tiempo se te va en verificación
- “Es más fácil hacer algo que depurarlo”

Conclusión: Cuando se hace un desarrollo, hay que definir una estrategia de verificación

¿Qué es verificar?



- Comprobar que los requisitos se cumplen
- Porque tenéis requisitos, ¿verdad?

Como mínimo

- Tests!
- Automatizar todos los tests que se puedan
- Ya que son automáticos, que se pasen tras los daily builds:
 - Tests del SW
 - Simulaciones (HDLs, PCBs, ...)
 - Pruebas en prototipo (FPGA, ...)

```
hipolito@challenger:~/devel/edelweiss/build$ make test
```

```
Running tests...
```

```
Test project /home/hipolito/devel/edelweiss/build
```

```
Start 1: 3x127byte_d_ff
1/11 Test #1: 3x127byte_d_ff ..... Passed      8.66 sec
Start 2: 3x127byte_fifo
2/11 Test #2: 3x127byte_fifo ..... Passed      3.53 sec
Start 3: 3x127byte_fadapt
3/11 Test #3: 3x127byte_fadapt ..... Passed     9.70 sec
Start 4: 3x127byte_bit2symb
4/11 Test #4: 3x127byte_bit2symb ..... Passed    7.83 sec
Start 5: 3x127byte_symb2chip
5/11 Test #5: 3x127byte_symb2chip ..... Passed   10.75 sec
Start 6: 3x127byte_upsampling
6/11 Test #6: 3x127byte_upsampling ..... Passed   8.69 sec
Start 7: 3x127byte_pulse_shaping
7/11 Test #7: 3x127byte_pulse_shaping ..... Passed 10.72 sec
Start 8: 3x127byte_Qdelay
8/11 Test #8: 3x127byte_Qdelay ..... Passed    8.96 sec
Start 9: 3x127byte_top_tx
9/11 Test #9: 3x127byte_top_tx ..... Passed   19.18 sec
Start 10: 3x127byte_dem_filter
10/11 Test #10: 3x127byte_dem_filter .....***Failed 0.04 sec
Start 11: 3x127byte_downsampling
11/11 Test #11: 3x127byte_downsampling .....***Failed 0.04 sec
```

```
82% tests passed, 2 tests failed out of 11
```

```
Total Test time (real) = 88.10 sec
```

```
The following tests FAILED:
```

```
10 - 3x127byte_dem_filter (Failed)
```

```
11 - 3x127byte_downsampling (Failed)
```

```
Errors while running CTest
```

```
make: *** [test] Error 8
```

```
hipolito@challenger:~/devel/edelweiss/build$
```

Para profundizar:

- Aprended las estrategias y herramientas de verificación de vuestra especialidad
 - Simulación mixta (electro-mecánico-térmico) puede ser útil para todos
- Electrónicos:
 - VHDL no sintetizable, assertions, self-checking testbenches
 - Simulaciones (PCB, RTL, post-layout, ...)
 - SystemVerilog, Metodologías de Verificación (UVM, OSVVM)

Buenas prácticas

- Control de versiones
- Bug tracking
- Generadores de documentación
- Builds en un paso
- Builds diarios
- Backups automáticos
- Estrategia de verificación
- Planificación realista
- Condiciones de trabajo

Riesgo: el proyecto tiene dos planificaciones

- Una en la pared del project manager
- Otra: la que ocurre en la realidad

La planificación de tareas es “fuertemente dependiente de datos históricos”

¿Cómo sé lo que voy a tardar?

Reglas útiles

- Empezad con herramientas sencillas (hoja de cálculo, pocos campos)
 - Ej: Tarea / Prioridad / Estimación original h / Estimación actual h / h Invertidas / h Quedan
- Cada funcionalidad ha de separarse en tareas
- El desarrollador que ejecuta la tarea es quien la divide y estima la duración (no el jefe!)

Reglas útiles

- Las tareas deben ser de ‘grano fino’
 - Deben medirse, por lo general, en horas
 - Si son demasiado grandes (más de 2 días) es que no has hecho el trabajo de dividirla correctamente
- Por supuesto, las Tareas deben ser ‘accionables’
- Includid tiempo (y, si podéis, personal) para la verificación!
 - (¡Aunque no se lo contéis al cliente!)

Plani... ¿qué?

Para tareas que ‘sabemos hacer’:

- Medirse el rendimiento y planificarse a partir de datos de uno mismo

Y si no he hecho nunca algo igual antes??

- Hay que tener un **‘plan B’**
 - No podemos dejar que el éxito del proyecto dependa de algo que no sabemos si conseguiremos terminar

Buenas prácticas

- Control de versiones
- Bug tracking
- Generadores de documentación
- Builds en un paso
- Builds diarios
- Backups automáticos
- Estrategia de verificación
- Planificación realista
- Condiciones de trabajo

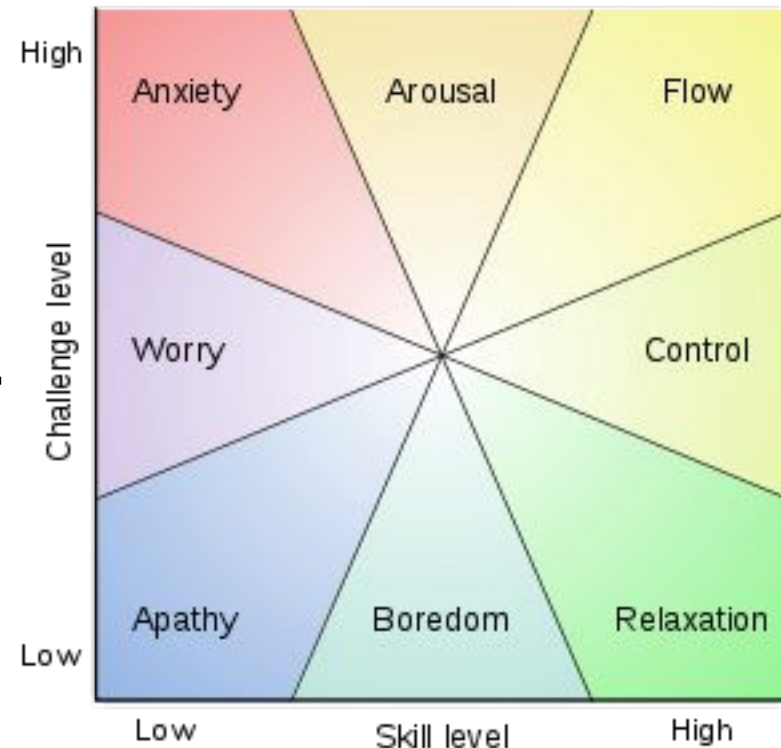
¿En qué condiciones trabajan los desarrolladores?

- Estado de Flow (Mihály Csíkszentmihályi)

Lo ideal sería estar ahí todo el tiempo...

... pero no es fácil entrar...

... y es muy fácil salir!



¿En qué condiciones trabajan los desarrolladores?

- Multitasking, papeleo excesivo, interrupciones, etc
- Decrementan la concentración
- Sacan a la gente del estado de 'flujo'
- Incrementan el tiempo que se tarda en desarrollar
- Y verificar!

Procurad que vuestro equipo pueda concentrarse en su trabajo!

¿En qué condiciones trabajan los desarrolladores?

- A unas malas, técnicas de gestión del tiempo
 - Timeboxing
 - 'Final Version' (Mark Foster)
 - Getting Things Done (David Allen)
 - ...

Desarrollador que puede concentrarse su trabajo = desarrollador motivado y feliz



Contenido

Conclusiones

Conclusiones

- Buenas prácticas ahorran quebraderos de cabeza durante el desarrollo
- No es lo mismo la buena práctica que la herramienta
 - las herramientas se pueden usar mal: hay que aprender cómo deben usarse
 - hay alternativas además de las que os he presentado aquí
- Las herramientas tienen su curva de aprendizaje, pero a la larga ahorran quebraderos de cabeza y tiempo

- Aprender las herramientas de una en una te permite adoptar la combinación que te sea más útil
- Existen sistemas que mezclan varias herramientas
 - Todo a la vez
 - Herramientas que no usas
 - Herramientas que te faltan
 - En algunos casos pueden ser muy útiles!
 - (tendréis que decidir)

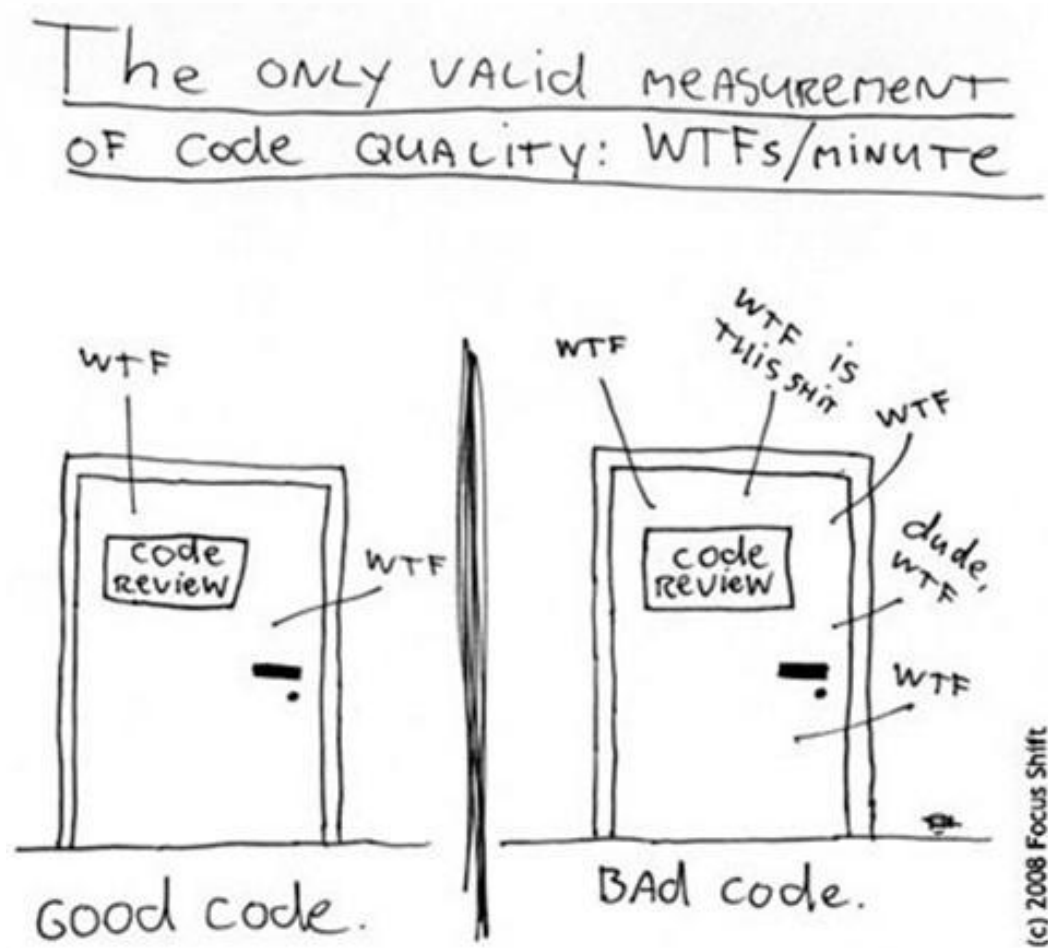
Os sugiero...

- Probad y familiarizaos con las herramientas
 - La mayoría son SW libre y se pueden montar en un ordenador con linux sin problema
 - Preguntadnos! Quienes las usamos estaremos encantados de echaros un cable
- Learn to love the commandline:
 - mayor control
 - automatización (builds, tests, lo que queráis, ...)
- Seguid aprendiendo y planteándoos: “*se puede hacer mejor*”?

¿Preguntas?

hguzman@us.es hipolitoguzman.net

Esto también aplica a las prácticas de trabajo!



Extra Slides

Builds diarios/automáticos

Sistemas de Integración Continua
(Continuous Integration) como Jenkins:


- No sólo para SW!

Por ejemplo en mi caso:

- Simulaciones de HDL
- Compilaciones de LaTeX

[New Item](#)[People](#)[Build History](#)[Manage Jenkins](#)[Credentials](#)[My Views](#)[add description](#)
 Todo

| S | W | Name ↓ | Last Success | Last Failure | Last Duration | |
|---|---|----------------------------------|-----------------------------------|-------------------------------|---------------|---|
|  |  | edelweiss | 9 days 8 hr - #42 | 13 days - #39 | 1 min 37 sec |  |
|  |  | vhd-verification | 2 min 30 sec - #2 | N/A | 6,4 sec |  |
|  |  | vunit_test1 | 13 days - #1 | N/A | 95 ms |  |

Icon: [S](#) [M](#) [L](#)
[Legend](#)
 [RSS for all](#)
 [RSS for failures](#)
 [RSS for just latest builds](#)
Build Queue

No builds in the queue.

Build Executor Status

1 Idle

2 Idle



search

Hipólito Guzmán | log out

Jenkins > edelweiss >

[ENABLE AUTO REFRESH](#)

[Back to Dashboard](#)

[Status](#)

[Changes](#)

[Workspace](#)

[Build Now](#)

[Delete Project](#)

[Configure](#)

[Test Results Analyzer](#)

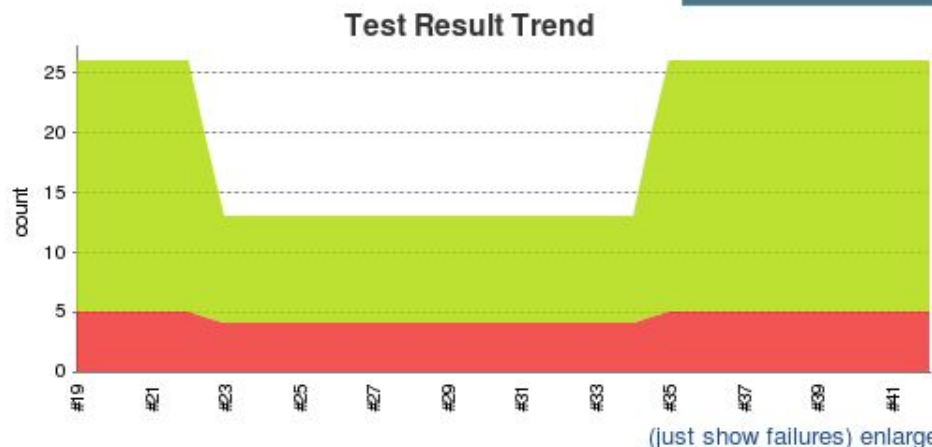
[DoxyGen HTML](#)

Project edelweiss

[add description](#)

[Disable Project](#)

- [DoxyGen HTML](#)
- [Workspace](#)
- [Recent Changes](#)
- [Latest Test Result \(5 failures / ±0\)](#)



Build History [trend](#)

find x

- [#42](#) Feb 17, 2016 12:49 PM
- [#41](#) Feb 17, 2016 12:47 PM
- [#40](#) Feb 13, 2016 3:56 PM
- [#39](#) Feb 13, 2016 3:47 PM
- [#38](#) Feb 13, 2016 3:42 PM

Permalinks

- [Last build \(#42\), 9 days 8 hr ago](#)
- [Last stable build \(#23\), 13 days ago](#)
- [Last successful build \(#42\), 9 days 8 hr ago](#)
- [Last failed build \(#39\), 13 days ago](#)
- [Last unstable build \(#42\), 9 days 8 hr ago](#)
- [Last unsuccessful build \(#42\), 9 days 8 hr ago](#)
- [Last completed build \(#42\), 9 days 8 hr ago](#)



search

Hipólito Guzmán | log out

Jenkins > edelweiss > #42

[ENABLE AUTO REFRESH](#)

[Back to Project](#)

[Status](#)

[Changes](#)

[Console Output](#)

[Edit Build Information](#)

[Delete Build](#)

[Git Build Data](#)

[No Tags](#)

[Test Result](#)

[Previous Build](#)

Build #42 (Feb 17, 2016 12:49:01 PM)

Started 9 days 8 hr ago
Took [1 min 37 sec](#)

[add description](#)



No changes.



Started by user [Hipólito Guzmán](#)



Revision: 51d1447720c17249c1d9ff8220d1352a3e5d81b8

- refs/remotes/origin/master



[Test Result](#) (5 failures / ±0)
[Show all failed tests >>>](#)



search

Hipólito Guzmán | log out

Jenkins > edelweiss > #42 > Test Results

[ENABLE AUTO REFRESH](#)

- [Back to Project](#)
- [Status](#)
- [Changes](#)
- [Console Output](#)
- [Edit Build Information](#)
- [History](#)
- [Git Build Data](#)
- [No Tags](#)
- [Test Result](#)**
- [Previous Build](#)

Test Result

5 failures (±0)



26 tests (±0)

Took 1 min 33 sec.

[add description](#)

All Failed Tests

| Test Name | Duration | Age |
|--|----------|--------------------|
| + projectroot.3x127byte_cfilter | 11 sec | 8 |
| + projectroot.32bit_tap | 0,2 sec | 24 |
| + projectroot.32bit_cfilter | 0,19 sec | 24 |
| + projectroot.32bit_downsampling | 0,2 sec | 24 |
| + projectroot.32bit_dem_filter | 0,18 sec | 24 |

All Tests

| Package | Duration | Fail | (diff) Skip | (diff) Pass | (diff) Total | (diff) |
|------------------------|--------------|------|-------------|-------------|--------------|--------|
| (root) | 1 min 33 sec | 5 | 0 | 21 | 26 | |