



Pablo Alcain
pabloalcain@gmail.com

Herramientas de Diseño y
Deployment

Hagamos software científico

El único ejemplo que conozco: dinámica molecular

Cálculo de fuerzas

Evolución temporal

Cálculo de magnitudes

Condiciones iniciales

Visualización

Cálculo de fuerzas

Modelo de interacción

Distintos estilos de partículas

Distintos tipos de partículas

De qué depende la fuerza

Cómo afectan a las partículas

Fuerzas externas

Evolución temporal

Relación con la “fuerza”

Forma de la integración

Constraint de la evolución

Valor del intervalo temporal

¿Igual para todas las partículas?

Tareas durante la evolución

Cálculo de magnitudes

Sobre qué partículas

De qué dependen

Qué magnitudes

Cuándo calcularlas

¿De qué orden son?

Qué hacemos cuando las obtuvimos

Condiciones iniciales

Posiciones iniciales

Velocidades iniciales

¿Debería poder elegir las fuerzas?

Archivo de entrada

Estilo de partículas

Tipos de partículas

Visualización



Primera solución

Reformamos el problema original

Cálculo de fuerzas: un sólo potencial

Evolución temporal: mantengo energía constante

Cálculo de magnitudes: todas al final

Condiciones iniciales: una red simple, velocidad cero

Visualización: escribo un archivo y lo leo desde afuera

Primera solución

Qué pasa si...

Tenemos un **bug**

Queremos **cambiar**

Necesitamos **acoplarlo**

Podemos **reutilizarlo**

Queremos **compartirlo**

Primera solución

Qué pasa si...

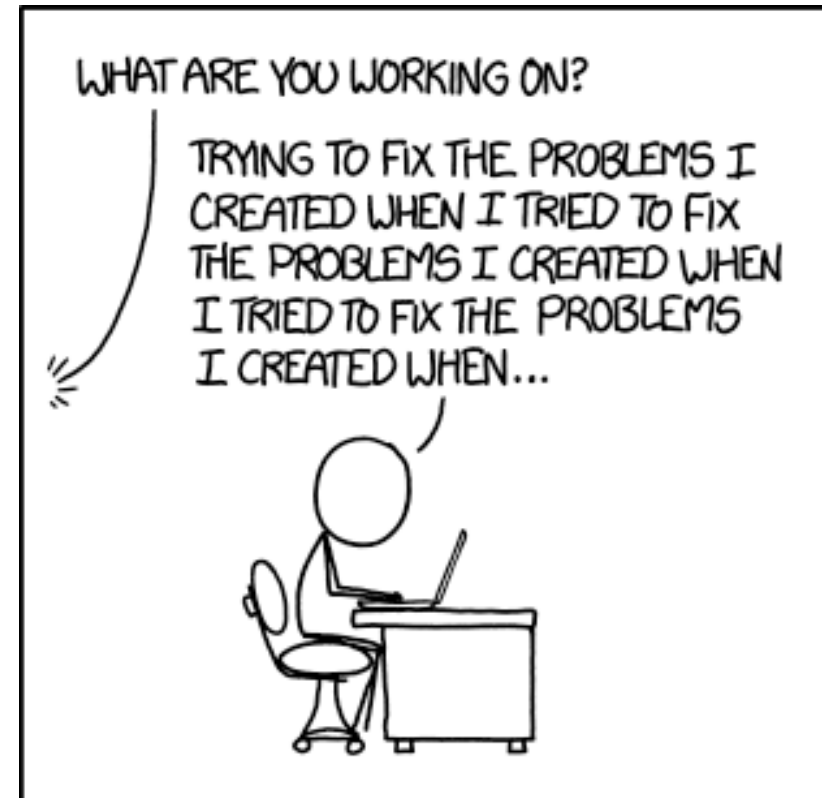
Tenemos un **bug**

Queremos **cambiar**

Necesitamos **acoplarlo**

Podemos **reutilizarlo**

Queremos **compartirlo**



Primera solución

Qué pasa si...

Tenemos un **bug**

gdb

Queremos **cambiar**

OOP

Necesitamos **acoplarlo**

Librerías

Podemos **reutilizarlo**

OOP

Queremos **compartirlo**

git

Primera solución

Qué pasa si...

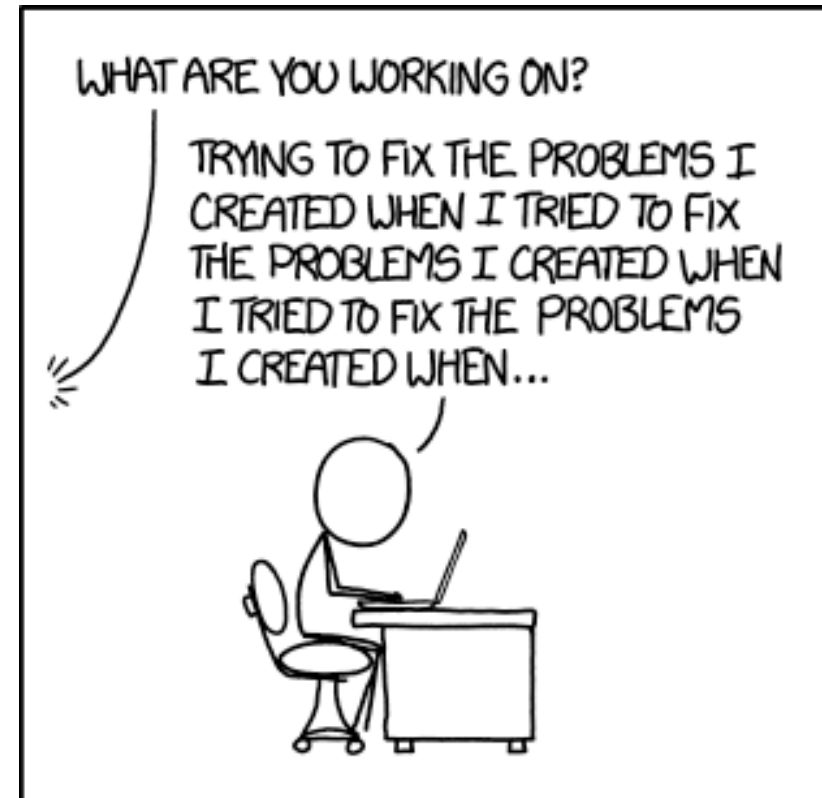
Tenemos un **bug**

Queremos **cambiar**

Necesitamos **acoplarlo**

Podemos **reutilizarlo**

Queremos **compartirlo**

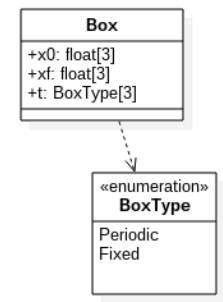
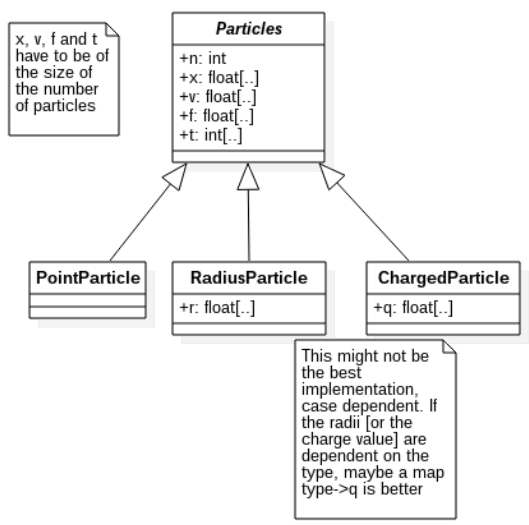
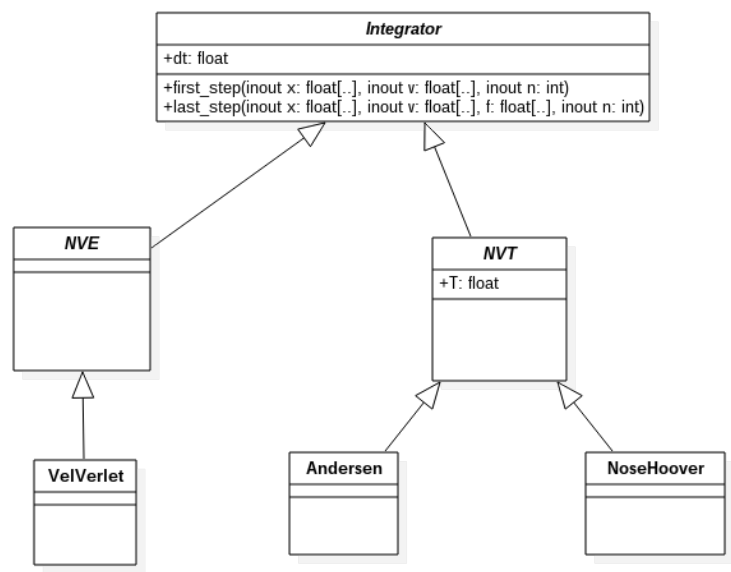


Desarrollo de software científico

Máxima: Nunca sabemos el tiempo de vida de un software

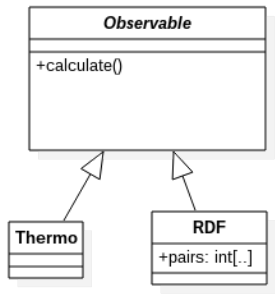
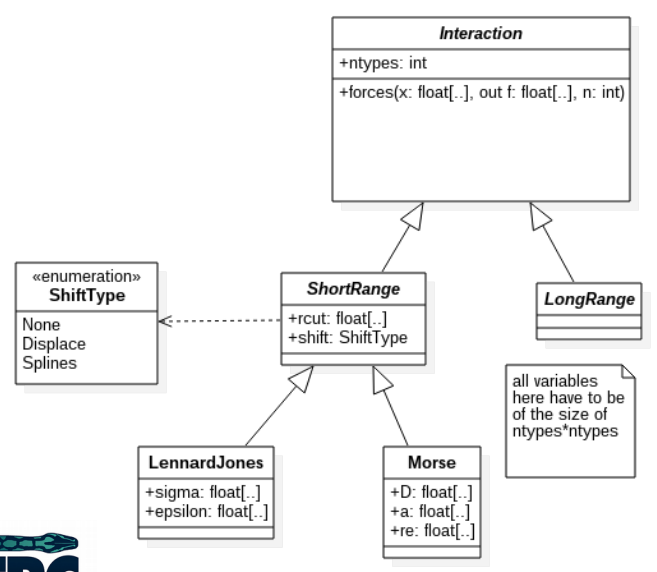
Cambio de mentalidad: primero el diseño, luego la implementación

Desarrollo de software científico



CAUTION:

1. We have not set here how the neighbours are treated.
2. It's not much of a problem to add observables that are $O(N)$, but those that depend on the pairs that interact somehow have to be inserted in the interaction loop. Not clear how and not clear how many actually do exist. LAMMPS only does this for pressure and total potential energy AFAIK.



Here we have decided to try to avoid, when possible, passing Particles [for example] as parameters. The reason is that we would like to be able to test and implement observables and/or integrators outside the ecosystem of the program. This, though, might probably change.

Desarrollo de software científico

Diferencias con el desarrollo “industrial”

No conocemos del todo el problema

El problema va cambiando

El diseño lo podemos hacer **incremental**

No nos casemos con un diseño

Implementación

¿Podemos hacer una implementación “incremental”?

Atomic commits en git tienen ese espíritu

Necesito que cada implementación sea **correcta**

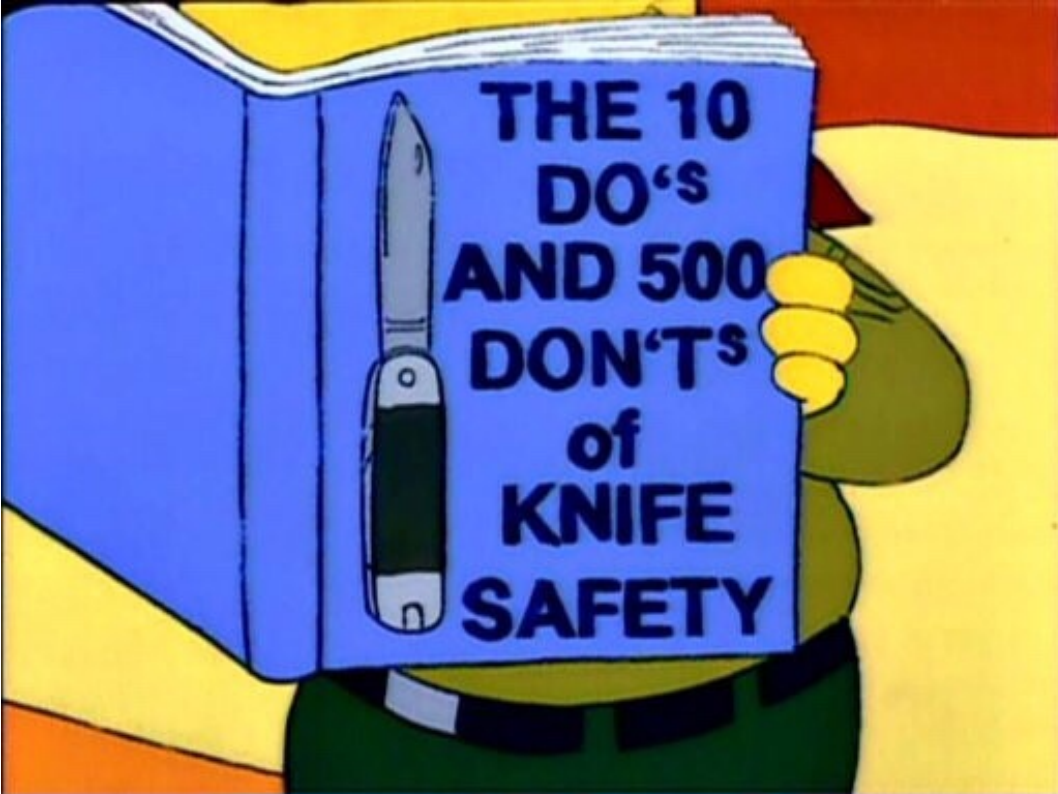
Unit testing

Unit testing

Supongamos que alguien nos da una función que divide:

```
/* file: div.h */  
#ifndef DIV_H  
#define DIV_H  
float divide(float x, float y);  
#endif
```

Y si ni sabemos la API?



Unit testing: ¿qué es y qué NO es?

Unit testing: ¿qué es y qué NO es?

NO ES “un paquete de X”

NO ES fácil de hacer bien

NO ES la forma de evitar todos los errores

NO ES una herramienta que “siempre está”

Unit testing: ¿qué es y qué NO es?

NO ES “un paquete de X”

NO ES fácil de hacer bien

NO ES la forma de evitar todos los errores

NO ES una herramienta que “siempre está”

SÍ ES algo que nos da confianza en nuestro código

SÍ ES una forma de desarrollar más rápido

SÍ ES algo que, una o dos veces, nos puede salvar

Unit testing: ¿qué es y qué NO es?

NO ES “un paquete de X”

NO ES fácil de hacer bien

NO ES la forma de evitar todos los errores

NO ES una herramienta que “siempre está”

SÍ ES algo que nos da confianza en nuestro código

SÍ ES importante para el desarrollo *muy colaborativo*

SÍ ES algo que, una o dos veces, nos puede salvar

Unit testing: ¿qué es y qué NO es?

NO ES “un paquete de X”

NO ES fácil de hacer bien

NO ES la forma de evitar todos los errores

NO ES una herramienta que “siempre está”

SÍ ES algo que nos da confianza en nuestro código

SÍ ES importante para el desarrollo *muy colaborativo*

SÍ ES algo que, una o dos veces, nos puede salvar

Unit testing: ¿qué es y qué NO es?

Continuous Integration








Compila el programa y ejecuta todos los tests automáticamente

pabloalcain / pexmd  build passing



















































Current **Branches** Build History Pull Requests

More options 

Default Branch

 master  14 builds	# 34 passed  4 days ago	 865b6ba   Pablo Alcain					
---	---	---	---	---	---	---	---

Active Branches

 particles  10 builds	# 33 passed  4 days ago	 dde556a   Pablo Alcain					
 interaction  4 builds	# 32 passed  8 days ago	 5e0e27e   Pablo Alcain					
 box  1 build	# 27 passed  20 days ago	 1fa3950   Pablo Alcain					
 integrator  1 build	# 24 passed  21 days ago	 48b2679   Pablo Alcain					
 uml  1 build	# 7 passed  about a month ago	 534df44   Pablo Alcain					
 pyup-config  2 builds	# 3 passed  2 months ago	 49347b2   pyup-bot					



Pablo Alcain
pabloalcain@gmail.com

Herramientas de Diseño y
Deployment