



*Una introducción a  
«Berkeley Open Infrastructure for Network  
Computing»*

**Álvaro Rodríguez Pinilla**  
**I.T.Informática de Sistemas 2003**  
**Sistemas Distribuidos**

## ¿Para que?

- Mayor poder de computación.
- Relación precio/rendimiento.
- Reutilización de infraestructuras.
- Escalabilidad.

## ¿Qué es Boinc?

- Boinc es una plataforma software para la computación distribuida con ordenadores voluntarios.
- Gestiona proyectos independientes.
- Características avanzadas de seguridad, computación y tolerancia a fallos.

## Aplicaciones objeto de BOINC

- Aplicaciones de interés público.
- Paralelismo independiente.
- Relación información/computación pequeña.
- Tolerantes a fallos.

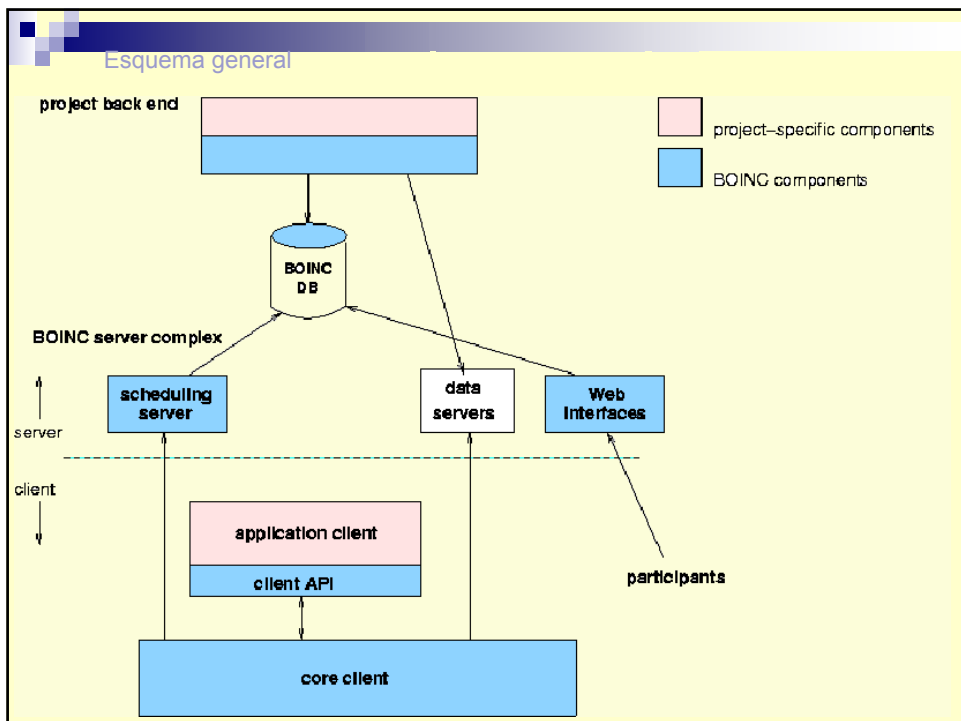
## Ejemplos:

- SETI@home, Search for ExtraTerrestrial Intelligence.

<http://setiathome.ssl.berkeley.edu/>

- PatriotGrid, Cáncer, Ántrax,...

<http://www.grid.org/projects/patriot.htm>



## Conceptos a aclarar

1. Plataformas.
2. Aplicaciones.
3. Unidades de trabajo.
4. Resultados.
5. Lotes (Batches).
6. Distribución del trabajo.
7. Secuencias de trabajo.
8. Acceso remoto a ficheros.
9. Seguridad.
10. Créditos.

## Plataformas

- Plataformas=CPU+S.O.
- Los host participantes no son todos iguales, no todos tienen igual potencia de cálculo y facilidades para la misma.
- Se intenta obtener el máximo rendimiento.
- Se deben diferenciar distintas plataformas.
- Lo ideal es una aplicación x plataforma.

## ¿Con que fin?

- El sistema debe evitar mandar código a un host que no puede ejecutarlo.
- Las aplicaciones deben ser capaces de explotar las características específicas de cada arquitectura.
- Información de las distintas arquitecturas se guarda en el servidor.
- Siempre considerando la simplicidad.

## Aplicaciones

- Una aplicación es un programa dedicado a una computación concreta, con sus unidades de trabajo y sus resultados.
- Una aplicación puede tener diferentes versiones.
- Cada aplicación tiene una versión mínima para garantizar la compatibilidad.
- Las versiones y las aplicaciones se mantienen en tablas de la Base de datos.

## Unidades de trabajo

- Una unidad de trabajo describe una computación que debe realizarse.
- Se mantienen en tablas de la Base de datos.
- Una unidad de trabajo está asociada a una versión mínima de una aplicación.

## Atributos de una unidad de trabajo

- Nombre.
- Aplicación.
- XML con Entradas y otros parámetros.
- La cantidad estimada de recursos requerida (Cobblestones, espacio, memoria).
- Tiempo máximo para ser procesada.

## Resultados

- Un resultado es una instancia de una computación, en cualquiera de sus estados posibles.
- Los resultados son almacenados en una tabla de la base de datos de Boinc.

## Atributos de un resultado

- Nombre (único en todo el proyecto).
- Unidad de trabajo asociada.
- Tiempo en que debería ser completado.
- XML con Nombre de los ficheros de salida.
- El tiempo en que se ha servido.
- Su estado: Inactivo, Preparado, Realizando, Terminado bien, Terminado con errores, tiempo excedido.

## Otros atributos de un resultado

- XML con tamaño y checksums de los ficheros de salida.
- Salida de errores.
- El host que lo ha mandado.
- El momento en que se ha recibido.
- El código de terminación.
- El tiempo de CPU usado.

## Lotes (Batches)

- Las unidades de trabajo y los resultados pueden agruparse en lotes.
- Cada lote es representado por un entero.
- Estas asociaciones permiten gestionar grupos de unidades de trabajo y resultados, de forma conjunta.



## Distribución de trabajo

- Los host prevén cuando terminarán su cuota de trabajo.
- High-water mark y low-water mark.
- Cuando se alcanza el low se pide todo el trabajo necesario hasta llegar al high.
- El servidor puede servir hasta 4 semanas de trabajo, al menos que el proyecto lo limite.

## Secuencias de trabajo

- ¿Como determinar el tamaño de una unidad de trabajo?
- Se usan secuencias de trabajo.
- Las secuencias se intentan ejecutar enteras en un mismo host, pudiendo ser recolocadas o extendidas de forma dinámica.
- Sus salidas podemos dividir las en: estado y respuestas.

## Planificar secuencias

- Una secuencia es referenciada por su primera unidad de trabajo.
- Esta cabecera contiene un tiempo de respuesta máximo, que determina a quien se asigna.
- Si un host no tiene secuencias asignadas, el planificador le asigna una, si es posible.

## Finalizar o extender una secuencia

- El manejador de resultados del proyecto, genera trabajo adicional o especifica que una secuencia ha terminado, cuando procesa un resultado de una secuencia.
- Si se especifica que la secuencia ha terminado, se le notifica al host y se le quita la asignación.

## Reubicando secuencias.

- Periódicamente se comprueba que ningún resultado de una secuencia ha alcanzado su tiempo máximo.
- En caso afirmativo se elimina su asignación y se prepara para reasignarla.
- Se crea una nueva secuencia desde la última unidad actualizada.
- Se reinicia el indicador de secuencia.

## Verificación redundante para secuencias

- Se crean varias secuencias con idénticos estados iniciales, escogiendo una como maestra.
- Cuando los resultados de una parte total de las secuencias se ha recibido, se comparan.
- Cuando se encuentra un error se reasigna.

## Acceso remoto a ficheros.

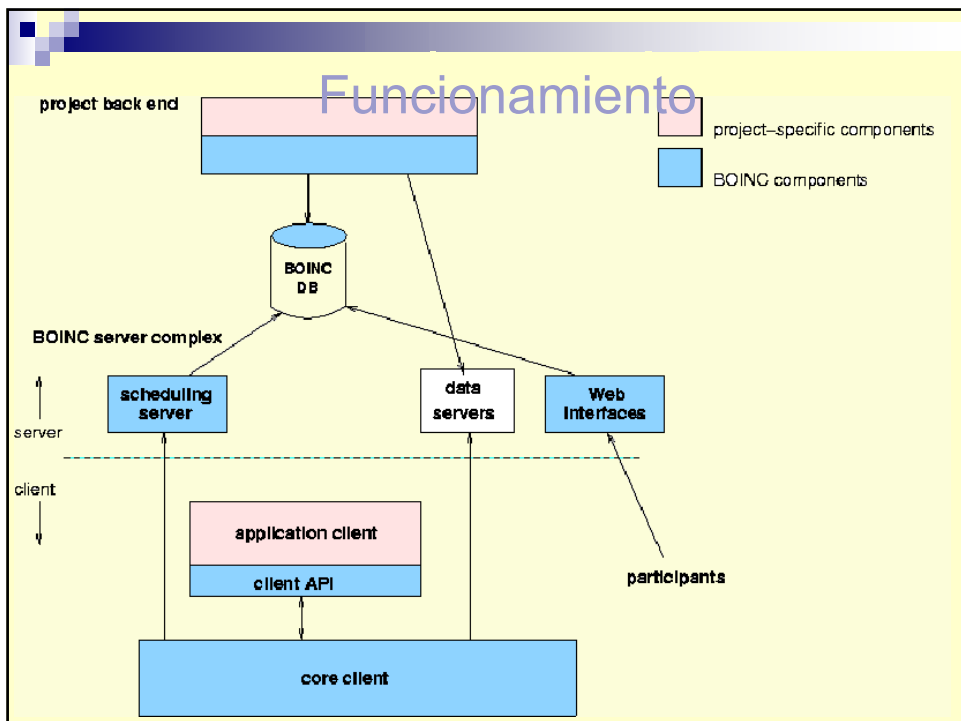
- Hay ficheros de salida, que bien por su tamaño o por ser fuente de entrada para otros procesos, no son subidos al servidor hasta que se conecta con el servidor de planificación.
- Esto se lleva a cabo manteniendo una petición de acceso a fichero en la base de datos.

## Seguridad

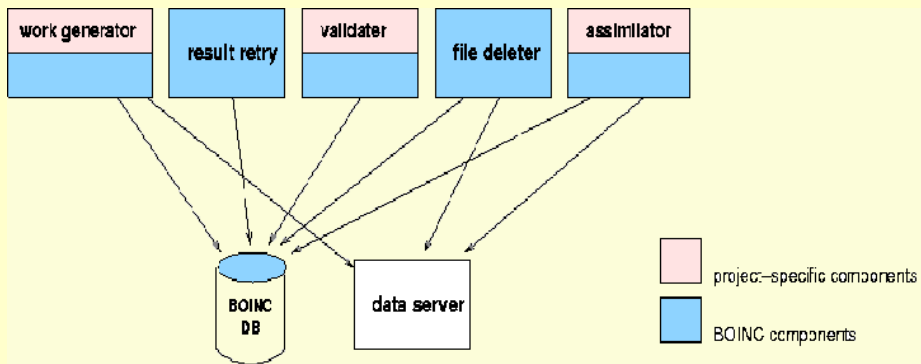
- Falsificación de resultados.
- Falsificación de créditos.
- Aplicaciones maliciosas.
- Desbordamiento del servidor de información.
- Robo de cuenta cliente por ataque al servidor.
- Robo de una cuenta cliente por ataque de red.
- Robo de archivos del proyecto.
- Abuso intencionado de hosts por el proyecto.
- Abuso accidental de hosts por el proyecto.

## Créditos

- Los créditos BOINC están basados en una máquina de referencia que puede efectuar:
  - 1000 millones de multiplicaciones de punto flotante por segundo.
  - 1000 millones de multiplicaciones de enteros por segundo.
  - 4000 millones de bytes de tráfico por segundo con la memoria.
- Cobblestone es un día de computación de esa máquina de referencia.



## Project Back end



```
bool ESTADO_CLIENTE::Hacer_algo() {
```

```
    bool accion=false;
```

```
    if (verificar_suspension_actividades()) return false;
```

```
    acción |= net_xfers->poll();
```

```
    action |= http_ops->poll();
```

```
    acción |= file_xfers->poll();
```

```
    acción |= active_tasks->poll();
```

```
    action |= scheduler_rpcs->poll();
```

```
    acción |= start_apps();
```

```
    acción |= pers_xfers->poll();
```

```
    acción |= handle_running_apps();
```

```
    acción |= handle_pers_file_xfers();
```

```
    acción |= recolector_de_basura();
```

```
    escribe_estado_fichero_si_es_necesario();
```

```
    return acción;
```

```
}
```

## Core client