Make

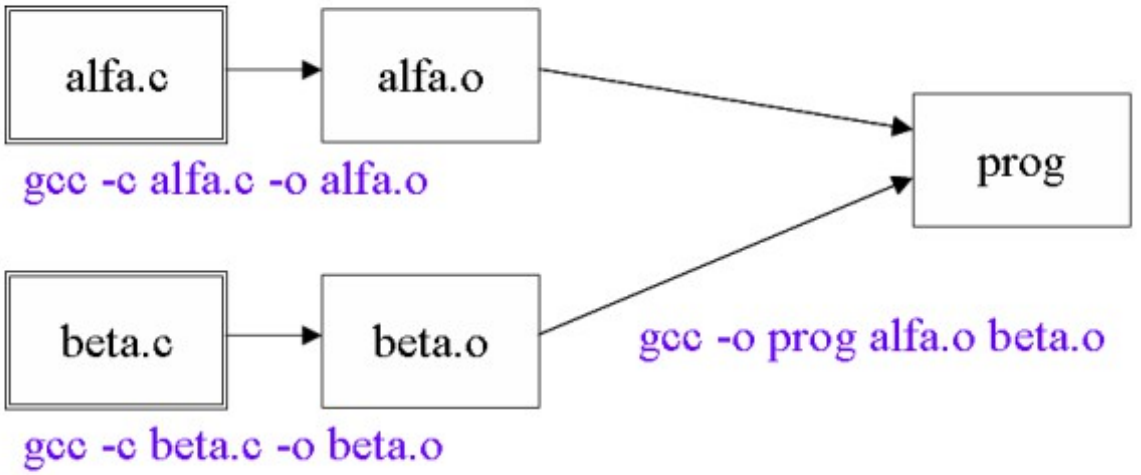
Herramienta para automatizar el proceso de generación y/o actualización de un conjunto de archivos (objetivo) que se construyen a partir de otros (fuente)

Prof. Angela Di Serio

Make

El proceso clásico de construcción de un programa ejecutable a partir de los archivos fuente consiste en compilar cada archivo fuente por separado,
para obtener el correspondiente archivo objeto, y luego combinar los archivos objetos (junto con funciones tomadas de las librerías) para obtener el programa ejecutable.

Prof. Angela Di Serio



Prof. Angela Di Serio

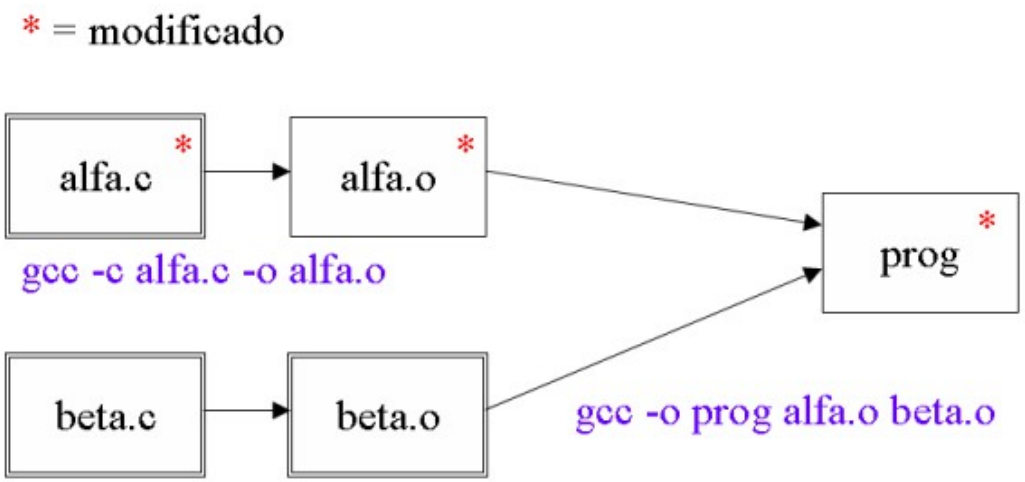
Qué buscamos con esta herramienta?

Recompilación selectiva

Tras construir el programa por primera vez y conservar los archivos objeto intermedios, sólo será necesario recompilar los archivos fuente que se hayan modificado desde la última construcción.

Siempre será necesario repetir el paso final de combinar los objetos para obtener el ejecutable.

Prof. Angela Di Serio



Prof. Angela Di Serio

Recompilación automática mediante comparación de fechas

El proceso de recompilación selectiva puede automatizarse mediante una herramienta que compare las fechas de actualización de los archivos fuente y las de los archivos objeto y ejecutable

Sólo hay que regenerar los archivos cuya fecha de actualización sea anterior a la de aquellos a partir de los cuales se reconstruye

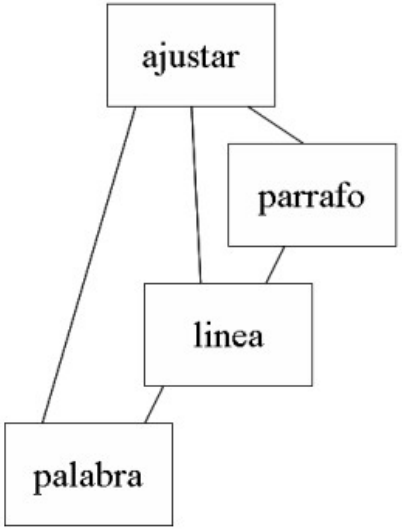
Prof. Angela Di Serio

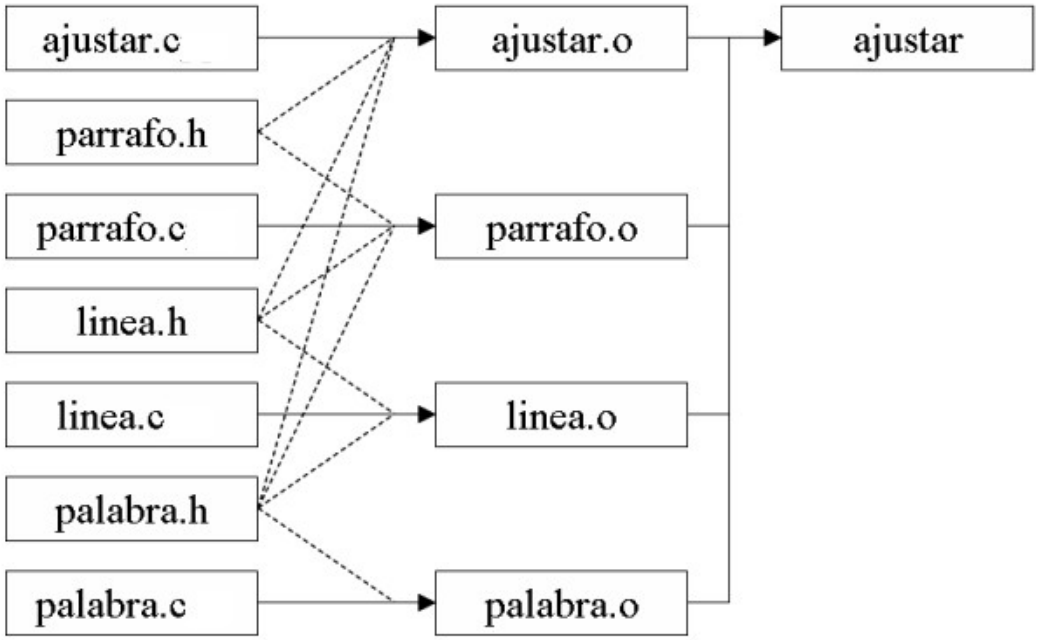
En casos muy sencillos no hace falta ninguna herramienta especial que ayude a construir el programa

Se podrían invocar las operaciones manualmente

En la mayoría de los casos reales, con un mayor número de archivos fuente y dependencias complejas entre ellos, es necesario disponer de alguna herramienta de ayuda

Prof. Angela Di Serio





Prof. Angela Di Serio

Detecta los archivos no actualizados mediante la comparación de fechas

Necesita información de las dependencias entre archivos y las acciones para regenerarlos

Esta información se suministra en un archivo denominado genéricamente ‘makefile’, usando una notación específica, basada en reglas

Cada regla contiene las dependencias de un archivo respecto a los demás y las acciones para regenerarlo

La herramienta make opera de forma recursiva, partiendo del objetivo final.

Para cada posible archivo a regenerar (archivo objetivo), primero se reconstruyen, si es necesario, los archivos de los que depende de manera inmediata, y a continuación se comprueban las fechas de actualización (que pueden haber cambiado si se ha regenerado alguno de ellos) para ver si el objetivo está al día.

Prof. Angela Di Serio

Los makefile deben contener la información que necesita make para operar:

- Archivos que intervienen en el proceso de reconstrucción
- Dependencias de unos archivos respecto a otros
- Acciones a realizar para regenerar cada archivo dependiente

Prof. Angela Di Serio

```
objetivo: dependencia dependencia ...
⇒ accion
⇒ accion
⇒ . . .
objetivo: dependencia ...
⇒ accion
⇒ . . .
```

Prof. Angela Di Serio

```
ajustar: ajustar.o parrafo.o linea.o palabra.o
    gcc -o ajustar ajustar.o parrafo.o linea.o palabra.o

palabra.o: palabra.c palabra.h
    gcc -c palabra.c -o palabra.o

linea.o: linea.c linea.h palabra.h
    gcc -c linea.c -o linea.o

parrafo: parrafo.c parrafo.h linea.h palabra.h
    gcc -c parrafo.c -o parrafo.o
```

make [-f makefile] [opciones] [objetivos]

Prof. Angela Di Serio

En un makefile se pueden utilizar variables

Las variables permiten almacenar valores de texto

Se crean automáticamente cuando se les asigna valor por primera vez

variable = valor

Las variables pueden usarse en cualquier parte del makefile (dependencias, acciones, o asignación de variables), usando la notación:

... $(variable) ...

Prof. Angela Di Serio

Reglas implícitas

Son plantillas genérica de reglas para reconstruir determinados tipos de archivos

La línea de dependencias de la regla hace referencia a los tipos de archivos por la extensión del nombre

%.ext1: %.ext2

Esta notación corresponde a que un archivo objetivo con extensión .ext1 depende de otro archivo con el mismo nombre y extensión .ext2

Prof. Angela Di Serio

Las líneas de acciones pueden contener códigos especiales (macros) para hacer referencia a determinados fragmentos de información que varían al aplicar la regla:

$@ - objetivo
$< - primera dependencia
$? - dependencias modificadas
$^ - dependencias (todas)
$* - lo que se ajusta al patrón %.

Prof. Angela Di Serio

```
OBJS = ajustar.o parrafo.o linea.o palabra.o

ajustar: $(OBJS)
   gcc -o ajustar $(OBJS)

%.o: %.c
   gcc -c $< -o$@

palabra.o: palabra.c palabra.h
linea.o: linea.c linea.h palabra.h
parrafo: parrafo.c parrafo.h linea.h palabra.h

clean:
   rm *.o
```

Prof. Angela Di Serio