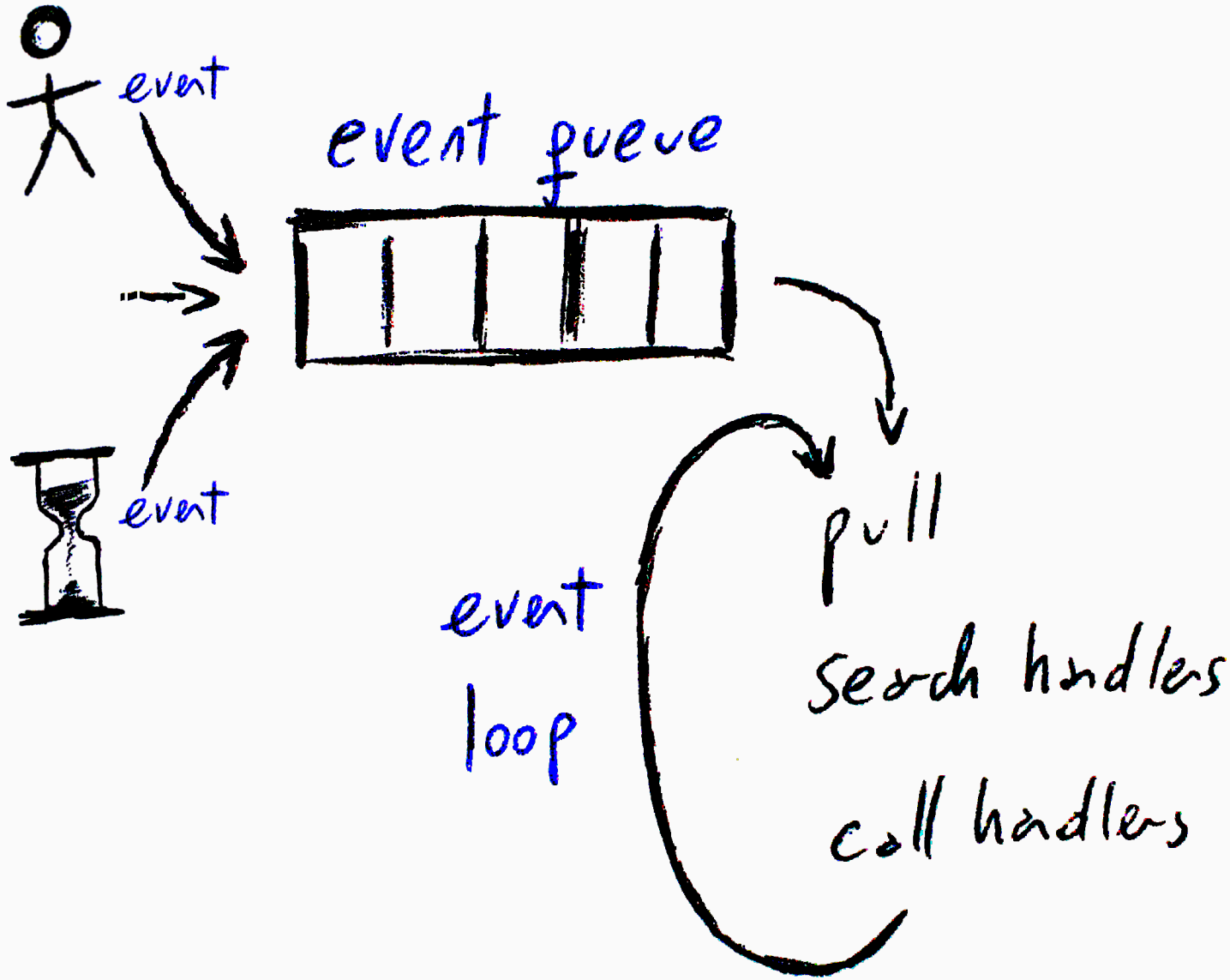
Programación orientada a eventos

Di Paola Martín

martinp.dipaola <at> gmail.com

Facultad de Ingeniería
Universidad de Buenos Aires

Programación orientada a eventos



Prohibido usar handlers lentos

```
1  void save_button_handler() {
2      FILE *f = fopen("data.txt", "wt");
3
4      /* ... */
5      fwrite(data, sizeof(char), data_sz, f);
6      /* ... */
7
8      fclose(f);
9  }
```

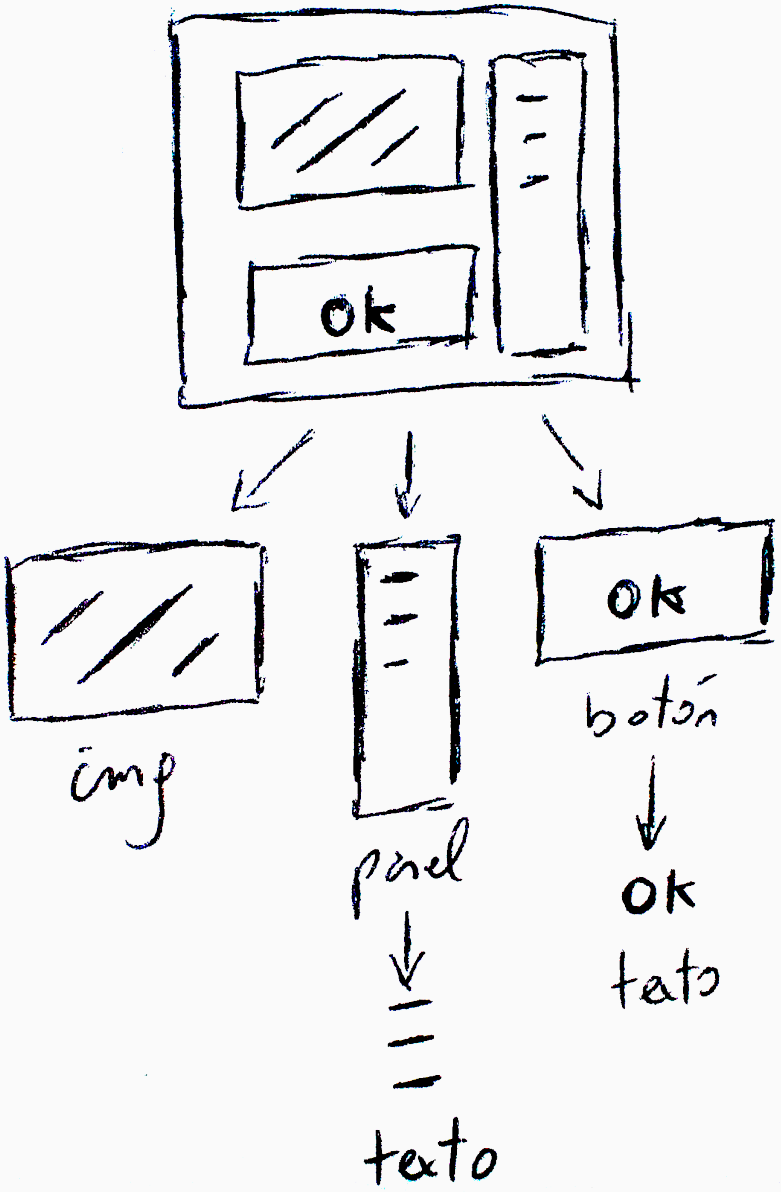
Handlers lentos: multithreading

```
1  void save_background() {
2      FILE *f = fopen("data.txt", "wt");
3
4      /* ... */
5      fwrite(data, sizeof(char), data_sz, f);
6      /* ... */
7
8      fclose(f);
9  }
10
11 void save_button_handler() {
12     std::thread t1 {save_background};
13 }
```

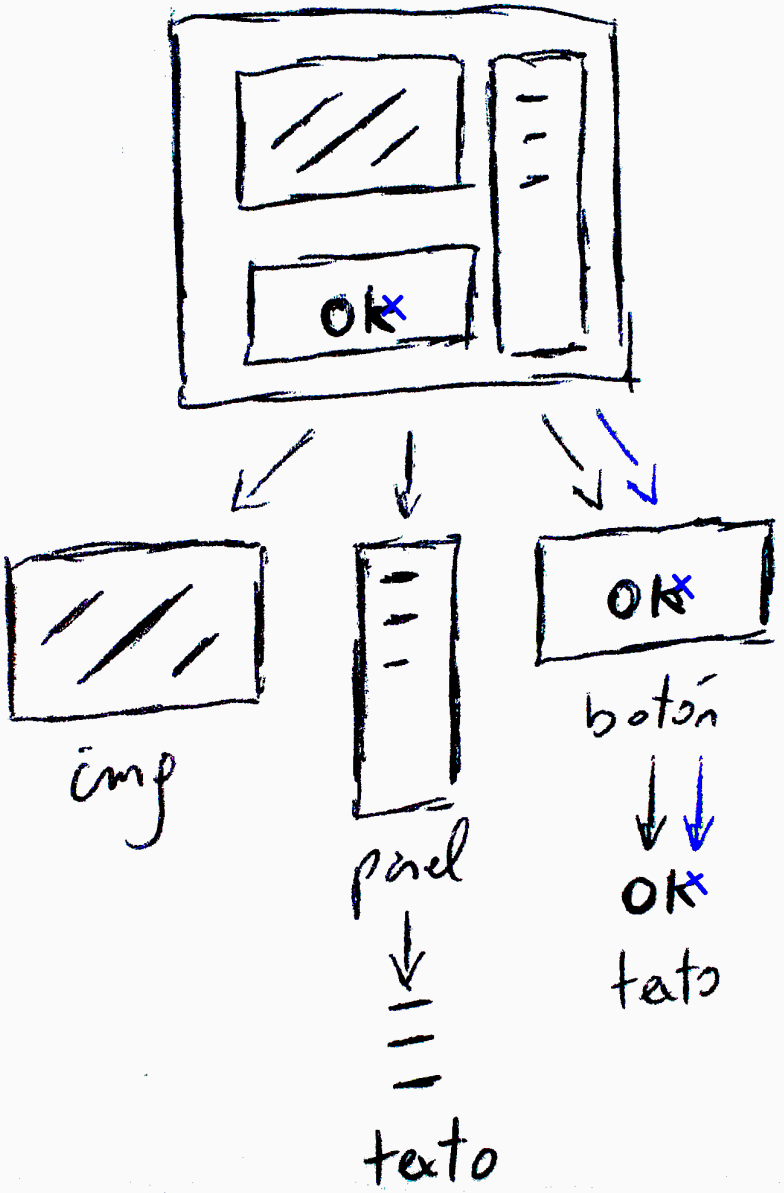
Handlers lentos: multithreading - join

```
1  void save_background() {
2     FILE *f = fopen("data.txt", "wt");
3
4     /* ... */
5     fwrite(data, sizeof(char), data_sz, f);
6     /* ... */
7
8     fclose(f);
9     emit_event("joinme", thread);
10 }
11
12 void save_button_handler() {
13    std::thread t1 {save_background};
14 }
15
16 void joinme_handler(thread) {
17     thread.join();
18 }
```

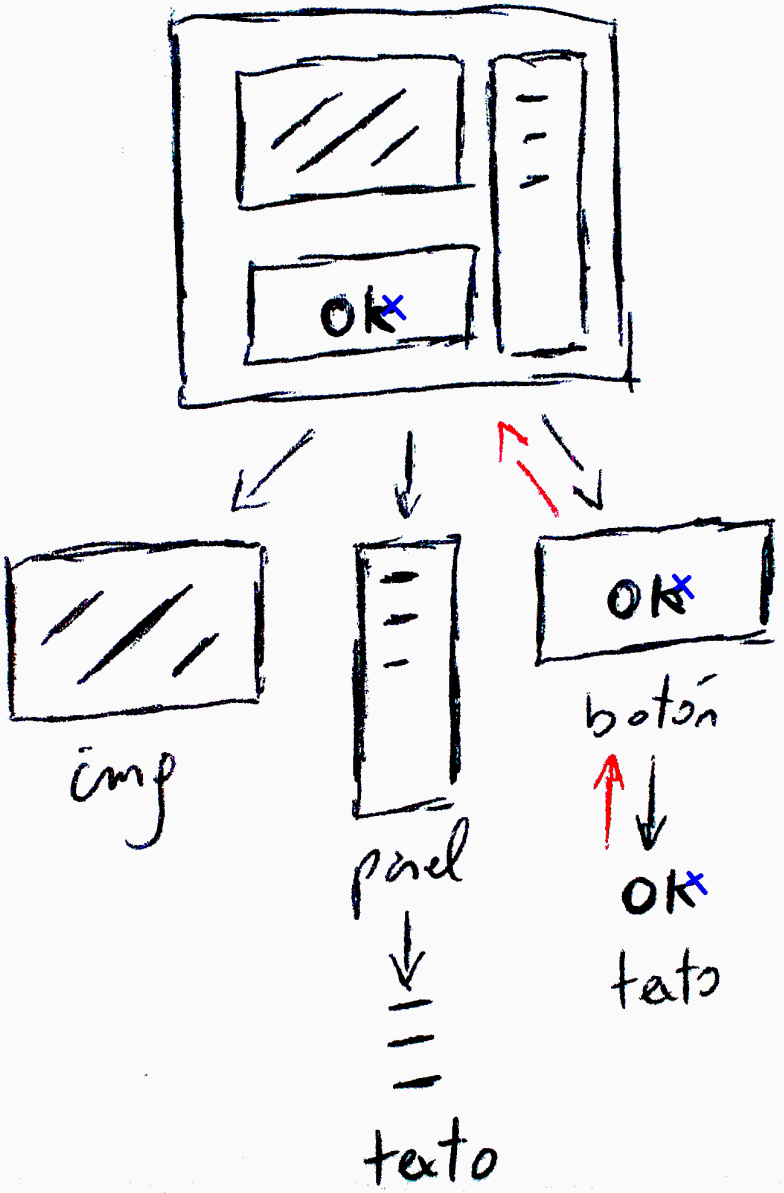
Búsqueda de los handlers (tree version)



Búsqueda de los handlers (tree version): capture phase



Búsqueda de los handlers (tree version): bubble phase



Búsqueda de los handlers (bit version): select

```
1  fd_set rfds;
2  struct timeval tv;
3
4  while (...) {
5     FD_ZERO(&rfds);
6     FD_SET(0, &rfds); /* 0 es la entrada estandar */
7
8     /* timeout de 5 segundos */
9     tv.tv_sec = 5;
10    tv.tv_usec = 0;
11
12    if (select(1, &rfds, NULL, NULL, &tv) == -1)
13        perror("select()");
14    else if (FD_ISSET(0, &rfds))
15        read(0, ...,); /* no deberia bloquearse */
16    else
17        /* time out */
18 }
```