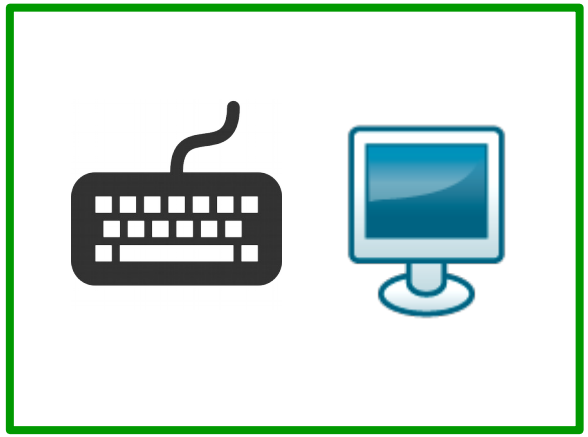
M4102 – Prog. réseau

AnnØe 2015/2016

Jeu Client - Serveur

Client - Serveur

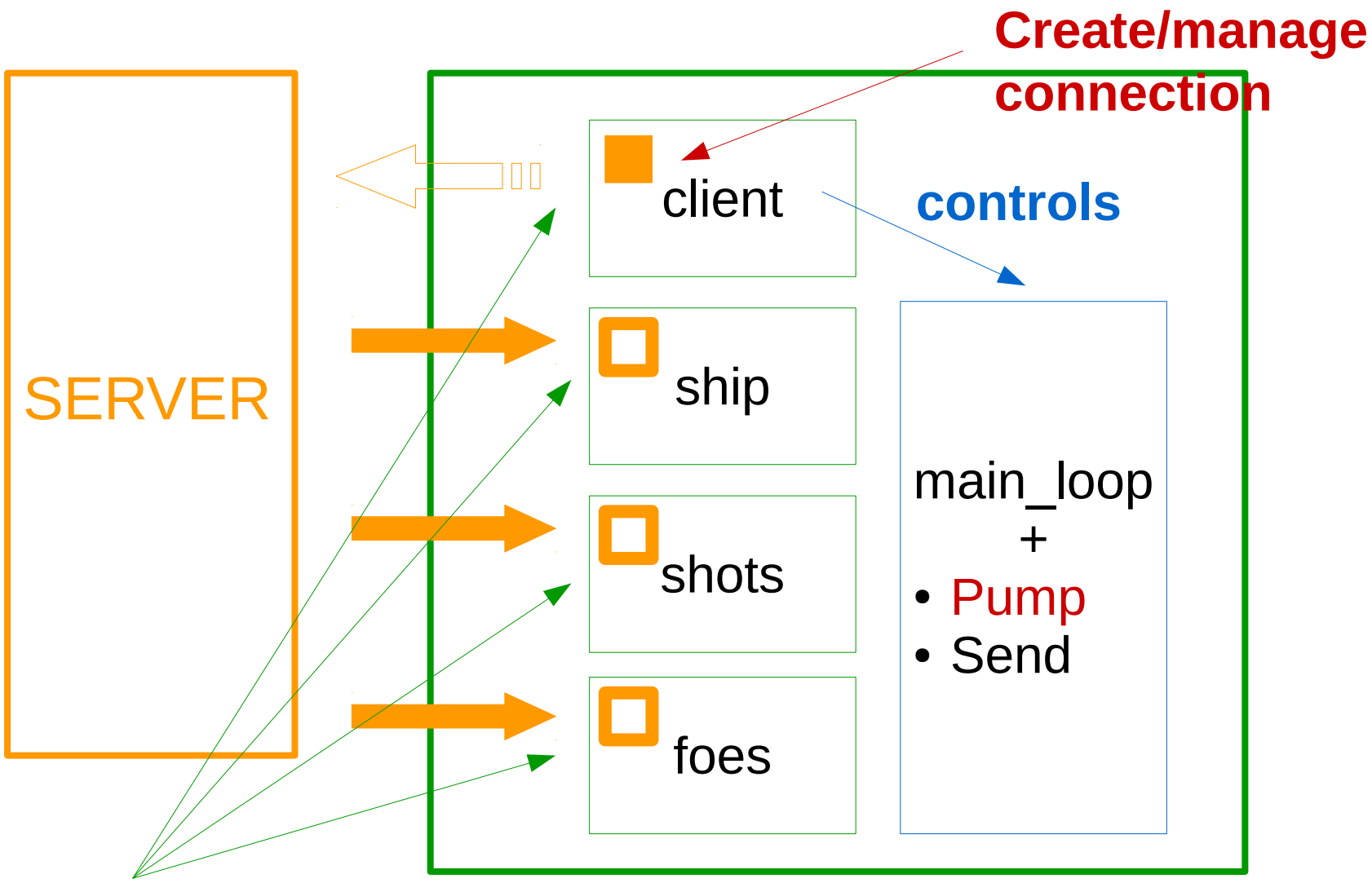
Simple game



Server

Client

Principe du client



```
"""Main function of the game"""

# PodSixNet init
game_client = GameClient(sys.argv[1],int(sys.argv[2]))
# Init Pygame
pygame.init()
...
# Game init
...

while True:
    clock.tick(60)
    connection.Pump()
    game_client.Pump()
    ...

    if game_client.run:
        keystrokes = pygame.key.get_pressed()
        connection.Send({'action':'keys','keystrokes':keystrokes})

        # updates
        ...
        # drawings
        ...

    else: # game is not running
        screen.blit(wait_image, wait_rect)

    pygame.display.flip()
```

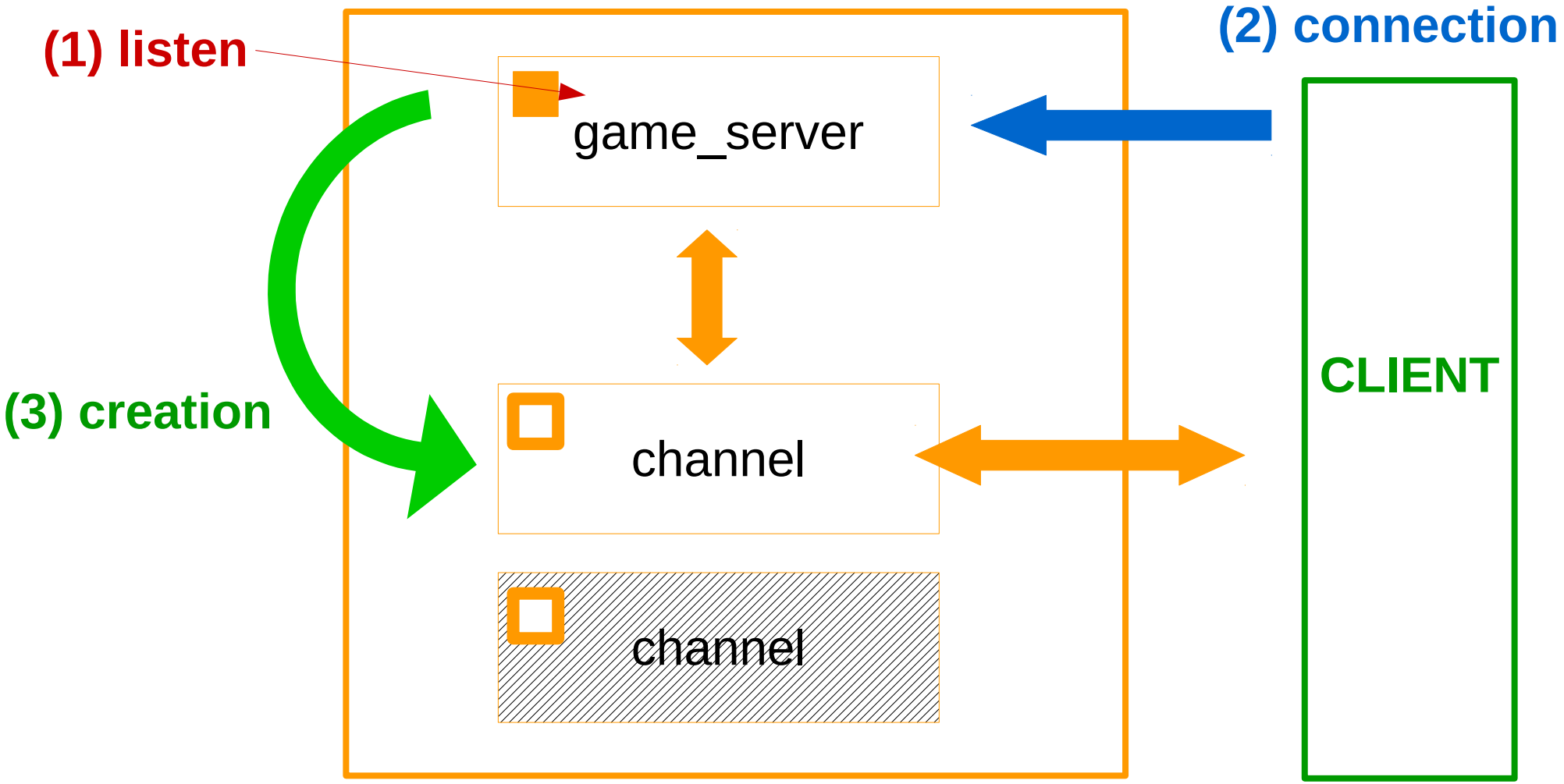
Principe du client (2)

```
# Game connection
class GameClient(ConnectionListener):

    def __init__(self, host, port):
        self.Connect((host, port))
        self.run = False

    ### Network event/message callbacks ###
    def Network_connected(self, data):
        ...
    def Network_error(self, data):
        ...
    def Network_disconnected(self, data):
        ...
```

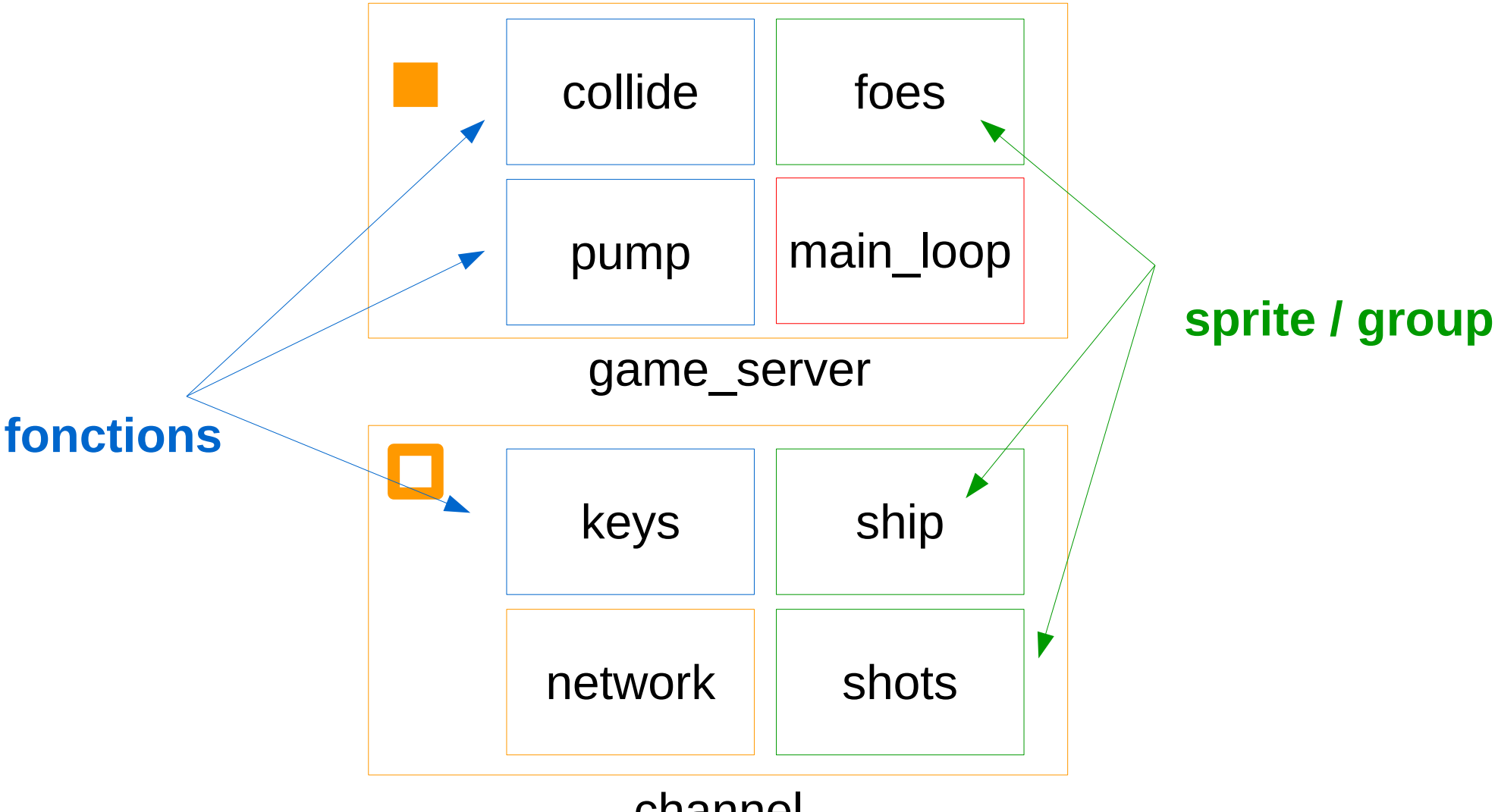
Principe du client (3)

```
# CLASSES
class Ship(pygame.sprite.Sprite, ConnectionListener):
    """Class for the player's ship"""

    def __init__(self):
        pygame.sprite.Sprite.__init__(self)
        self.image, self.rect = load_png('Pics/ship1.png')
        self.rect.center = [SCREEN_WIDTH/2, SCREEN_HEIGHT/2]

    def Network_ship(self,data):
        self.rect.center = data['center']

    def update(self):
        self.Pump()
```

Principe du serveur



Détails du serveur



```
class MyServer(Server):
    channelClass = ClientChannel

    def __init__(self, *args, **kwargs):
        Server.__init__(self, *args, **kwargs)
       ...
        pygame.init()
        ...

    def Connected(self, channel, addr):
       ...

    # SENDING FUNCTIONS
    def send_ships(self):
        for client in self.clients:
            client.send_ship()

    # UPDATE FUNCTIONS
    def update_ships(self):
        for client in self.clients:
            client.update_ship()

    def launch_game(self):
        # Init Pygame
        pygame.display.set_caption('Server')
        ...

        while True:
            clock.tick(60)
            self.Pump()

            if self.run:
               ...
                self.update_ships()
                self.send_ships()
                # no drawing
            pygame.display.flip()
```

Principe du serveur (2)

```
class ClientChannel(Channel):

    def __init__(self, *args, **kwargs):
        Channel.__init__(self, *args, **kwargs)
        self.number = 0
        self.is_shooting = 0
        self.ship = Ship()

    def Close(self):
        self._server.del_client(self)

    def Network_keys(self,data):
        touches = data['keystrokes']
        if(touches[K_LEFT]):
            self.ship.left()
       ...

    def send_ship(self):
        self.Send({'action':'ship','center':self.ship.rect.center})

    def update_ship(self):
        self.ship.update()
```