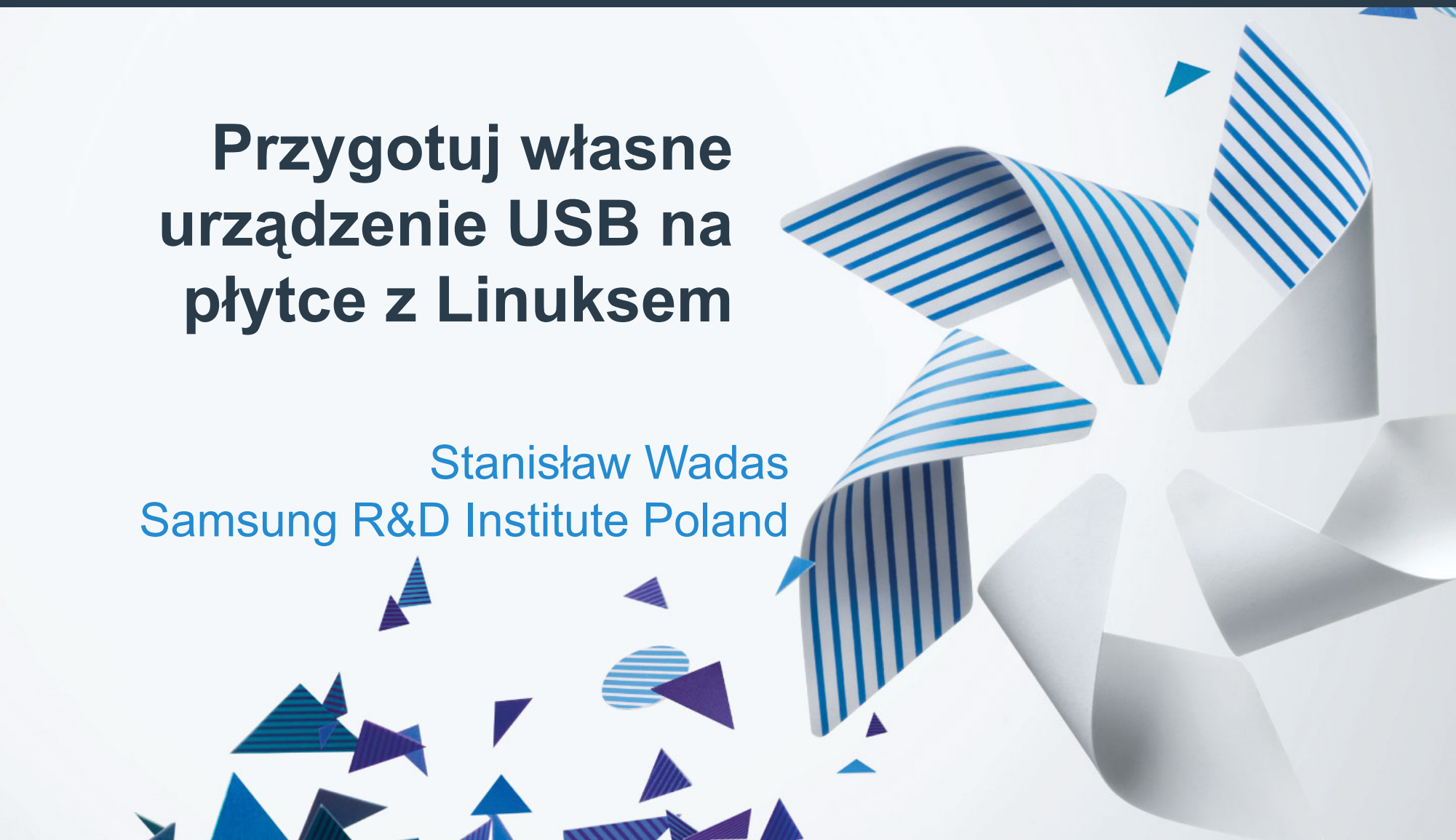
jesień linuksowa

7-9 Listopada 2014 - Szczyrk, Polska

Przygotuj własne urządzenie USB na płytce z Linuksem

Stanisław Wadas
Samsung R&D Institute Poland



Agenda

- USB
  - Hardware
  - Funkcja
  - FunctionFS
  - ConfigFS
- ConfigFS composite gadget
- libusbg & gt
- gadgetd & gadgetctl
  - Dalszy rozwój
  - gadgetctl
- Q&A

TIZEN™

USB

Host vs Devices

Host - *Może zostać wzbogacony o funkcje dostarczone przez Device'a*

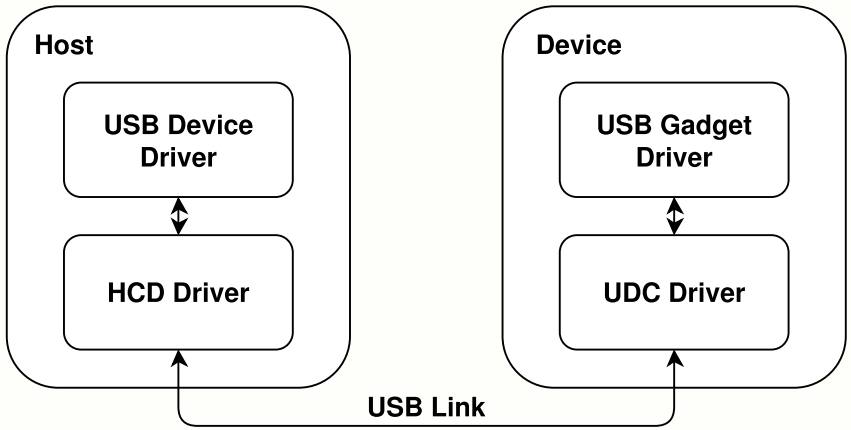
HCD driver - Sterownik dla kontrolera hosta

USB device driver - Sterownik dla urządzenia USB

Device - *Rozszerza hosta o pewne funkcje*

UDC driver Sterownik dla kontrolera urządzenia USB

USB Gadget driver Sterownik implementujący logikę



Hardware

PC

DUMMY_HCD (kmod)

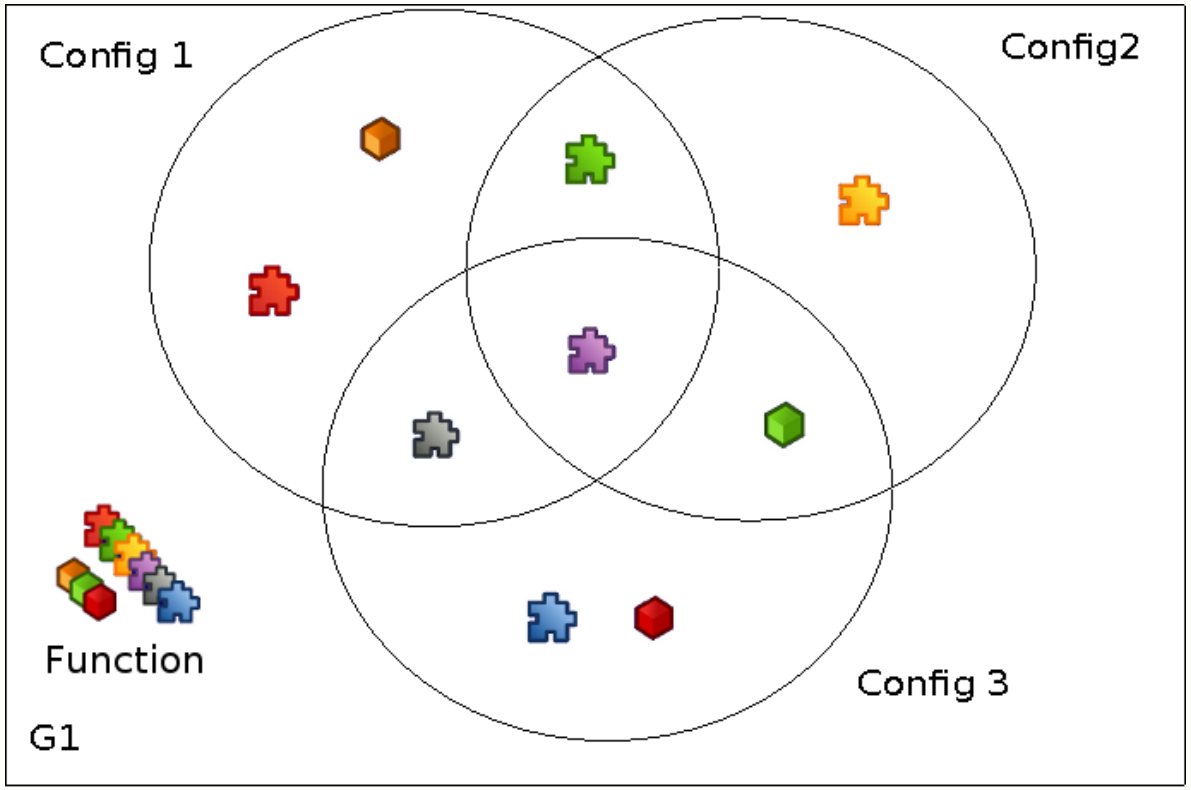
USB2380 (PCIe)

Embedded/mikrokomputery jednopłytkowe

TIZEN™

USB Composite device

Konfiguracje i funkcje

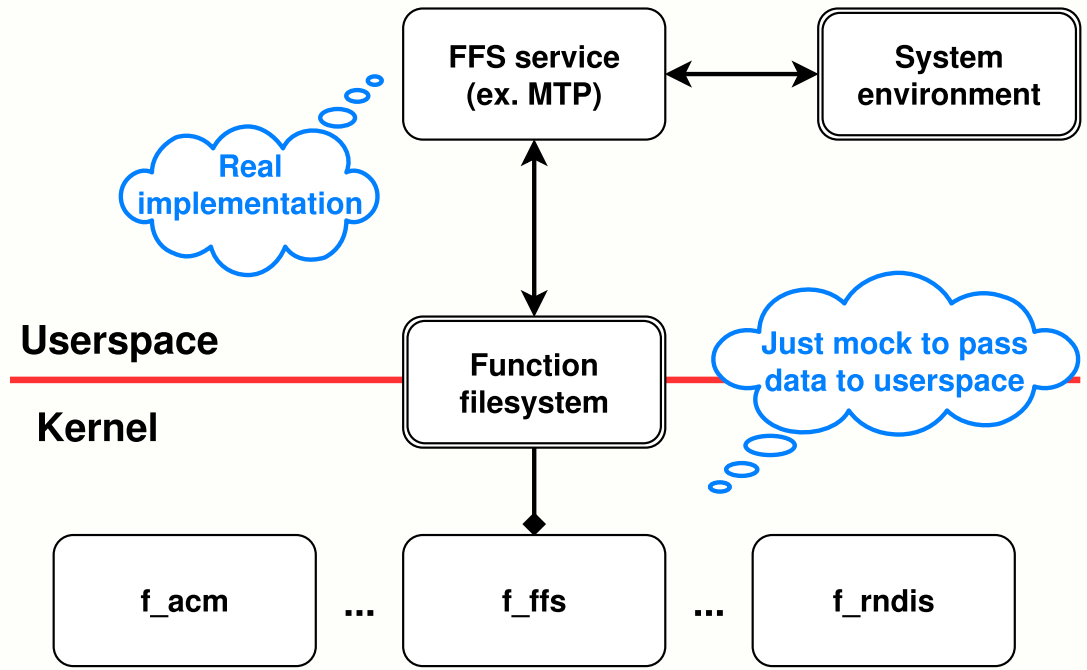


USB - Funkcje w kernelu

- **Serial : ACM, OBEX, Serial**
- **Ethernet : ECM, EEM, NCM, Subset, RNDIS**
- **Phonet**
- **Mass Storage**
- **Loopback**
- **SourceSink**
- **UVC and HID - WIP**

FunctionFS

- **Możliwość implementacji logiki funkcji USB w przestrzeni użytkownika**
- **Funkcja prostsza w implementacji**
- **Własna funkcja USB**
- **Opakowuje operacje IO na pliku w usb_requests**



USB gadget - dotychczas

- **Ładowanie modułu np. modprobe g_ether**
- **Jeden gadget == jeden moduł**
- **Modyfikowanie jedynie przez parametry - podczas ładowania modułu**

TIZEN™

ConfigFS

- **Dostarcza interfejs do konfiguracji**
- **Utworzenie gadgeta z wykorzystaniem systemu plików**
- **Brak konieczności rekompilacji**
- **Kod odseparowany od konfiguracji**
- **Elastyczność**

ConfigFS USB Gadget

- **Podajemy jego nazwę**
- **Wypełniamy**
  - Vendor ID
  - Product ID
  - Strings (manufacturer, product and serial)
  - Device Class details
- **Tworzymy funkcję**
- **Tworzymy konfigurację**
- **Przypisujemy funkcję do konfiguracji**

ConfigFS composite gadget

Jak zacząć

- **Hadware**
- **Kernel + funkcje USB konfigurowane przez ConfigFS**
- **(...)**

```
$ modprobe libcomposite
$ mount none -t configfs /sys/kernel/config
$ cd /sys/kernel/config/usb_gadget
```


TIZEN™

Przykład

```
$ mkdir g1
$ cd g1
$ echo "0x04e8" > idVendor
$ echo "0x6865" > idProduct
$ mkdir strings/0x409
$ echo "A123BC1" > strings/0x409/serialnumber
$ echo "Samsung" > strings/0x409/manufacturer
$ echo "Galaxy S3" > strings/0x409/product
$ mkdir functions/eem.usb0
$ mkdir configs/c.1
$ mkdir configs/c.1/strings/0x409
$ echo "Config 1" > \
        configs/c.1/strings/0x409/configuration
$ ln -s functions/eem.usb0 configs/c.1/
```


TIZEN™

Przykład

```
$ ls /sys/class/udc
12480000.hsotg
$ echo "12480000.hsotg" > UDC
```


TIZEN™

Przykład

```
$ lsusb –v
Bus 001 Device 009: ID 04e8:6865 Samsung Electronics Co., Ltd GT–I9100 Phone[Galaxy S III]
Device Descriptor:
  bLength                18
  bDescriptorType         1
  bcdUSB               0.00
  bDeviceClass            0 (Defined at Interface level)
  bMaxPacketSize0        64
  idVendor          0x04e8 Samsung Electronics Co., Ltd
  idProduct         0x6865 GT–I9100 Phone [Galaxy S III]
  bcdDevice            3.17
  iManufacturer           1 Samsung
  iProduct                2 Galaxy S3
  iSerial                 3 A123BC1
  bNumConfigurations      1
```


TIZEN™

ConfigFS i FunctionFS

```
$ echo "" > UDC
$ mkdir functions/ffs.my_func_name
$ ln -s functions/ffs.my_func_name configs/c.1/
$ mount my_func_name -t functionfs /tmp/mount_point
$ run_function_daemon
$ wait_for_daemon_initialization
$ echo "12480000.hsotg" > UDC
```


TIZEN™

Problemy

- **Około 20 poleceń by zbudować prosty gadget**
- **Użytkownik musi znać typ dostępnej funkcji**
- **Wiele zależności, konieczność znajomości pewnych wartości specjalnych np dla języka**
- **Bezpieczeństwo - konieczność wykonywania poleceń jako root/sudoer**

Problemy

- **Konieczność używania unikalnych nazw instancji funkcji FunctionFS**
- **Ręczne uruchamianie demona FunctionFS oraz przekazanie punktu montowania funkcji do demona**
- **Śmierć demona spowoduje odłączenie gadgeta od UDC(OOM - Killer)**

TIZEN™

libusbg & gt

libusbg

- **Projekt rozpoczęty przez Matta Portera we wrześniu 2013**
- **Biblioteka napisana w języku C pomaga w szybkim i prostym tworzeniu gadgetów**
- **Oficjalne repozytorium:**

  `https://github.com/libusbg/libusbg`

- **Wersja rozwojowa Krzysztofa Opasiaka:**

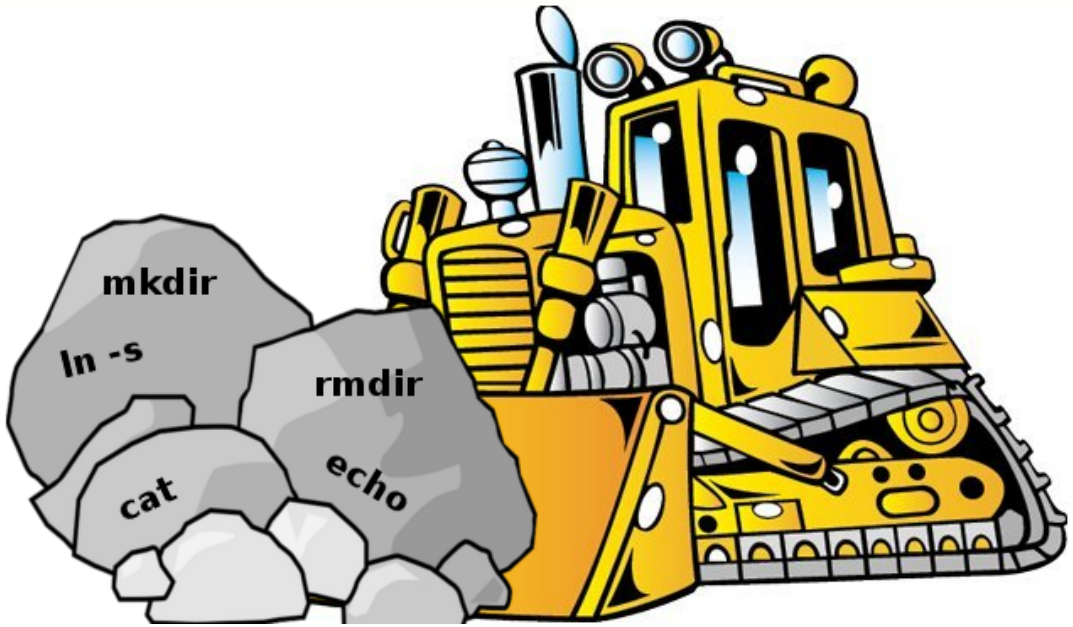
  `https://github.com/kopasiak/libusbg`

- **Wsparcie dla prawie wszystkich funkcji USB**

TIZEN™

libusbg

- **C API - pozwala na zbudowanie gadgeta "z kodu"**
- **Dostarcza warstwę abstrakcji dla ConfigFS**
- **Redukuje liczbę potencjalnych błędów**
- **Pozwala na szybkie i proste zbudowanie gadgeta**



TIZEN™

libusbg

- **Samo API niewystarczające**
- **Kod imperatywny, potrzeba deklaratywnego w postaci konfiguracji**

TIZEN™

Schematy gadgetów

Przykład

```
attrs = {idVendor = 0x04e8; idProduct = 0x6865;}

strings = ({
  lang = 0x409;
  manufacturer = "Samsung"
  product = "Galaxy S3"
  serialnumber = "A123BC1"
})

configs = ({
  id = 1
  name = "c"
  strings = ({lang = 0x409; configuration = "Config 1";})
  functions = ({function = {instance = "usb0"; type = "eem";}})
})
```


TIZEN™

Schematy gadgetów

- **Użycie plików do tworzenia gadgetów**
- **usbg_import_gadget(), usbg_export_gadget()**
- **Składnia libconfig'a**
- **Deklaratywny**
- **Zbliżamy się do** *modprobe g_ether* **!!!**

TIZEN™

gt

- Command line tool do zarządzania gadgetami
- Używa libusbg
- Projekt powstał z inicjatywy Krzysztofa Opasiaka
- Rozwijany na githubie:

  https://github.com/kopasiak/gt
- Zaimplementowane kilka podstawowych komend
- W trakcie intensywnego rozwoju

UNDER CONSTRUCTION

TIZEN™

gt - Gadget tool

- **C API niewystarczające**
- **Zapewnia dostęp do libusbg z command line**
- **Upraszcza administrację gadgetami**
- **Umożlwia zbudowanie gadgeta za pomocą jednego polecenia**

Przykład

Jedna konfiguracja, jedna funkcja

```
$ gt create g1 \
    idVendor=0x04e8 \
    idProduct=0x6865 \
    manufacturer="Samsung" \
    serialnumber="A123BC1" \
    product="Galaxy S3"

$ gt func create g1 eem usb0
$ gt config create g1 label 1
$ gt config add g1 1 eem usb0
$ gt enable g1
```


TIZEN™

gt - Gadget tool

- **gt load/save**
- **Deklaratywny sposób tworzenia gadgeta**
- **modprobe g_ether ? Jakiś zysk ?**
- **Co z FunctionFS?**

TIZEN™

gadgetd & gadgetctl

gadgetd

- **Rozwijane na githubie:**
  `https://github.com/gadgetd/gadgetd`
- **Projekt powstał na początku 2014 z inicjatywy Krzysztofa Opasiaka**
- **Używa libusbg**
- **WorkInProgress**

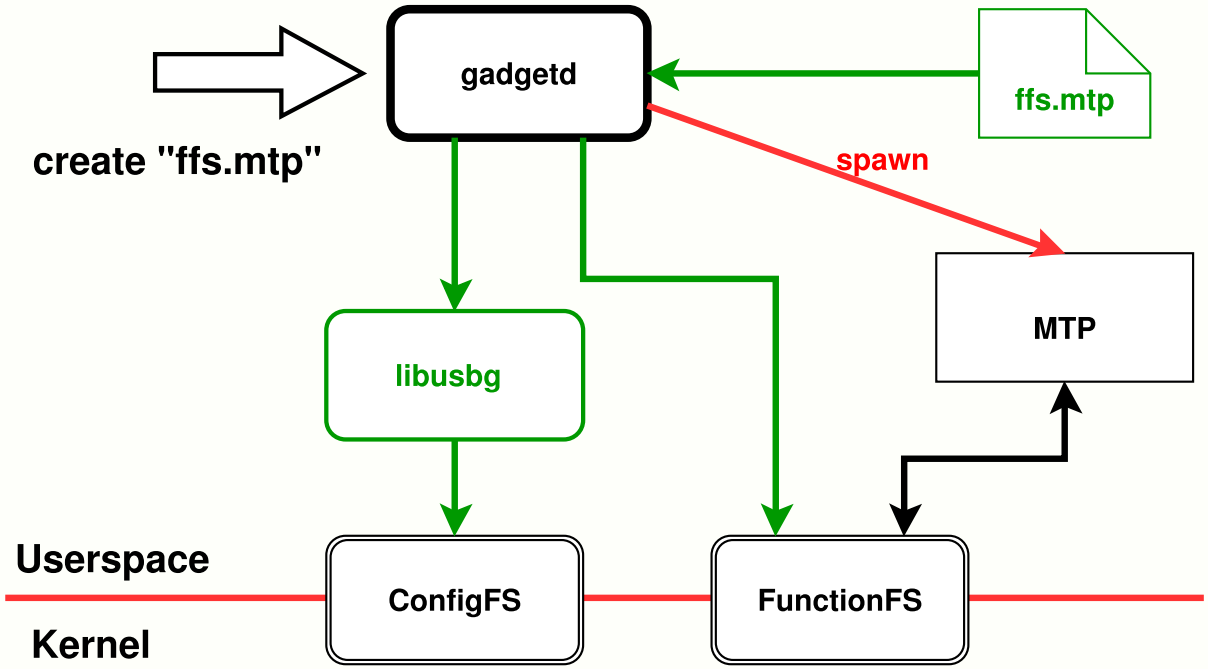
TIZEN™

gadgetd

- **Wysokopoziomowe API via D-Bus**
- **Jednolite API dla funkcji kernelowych i userspace'owych**
- **Reprezentacja w postaci obiektów D-Bus'owych**
- **Zwiększone bezpieczeństwo**

Gadgetd i FunctionFS

- **Kod odseparowany od konfiguracji**
- **Funkcja posiada odrębny plik konfiguracyjny, np dostarczony przez użytkownika**
- **Odrębny serwis dla pojedynczej funkcji**
- **Dbałość o nazewnictwo i montowanie**



Dalszy rozwój

- **Profile USB**
  ładowarka, prywatny, publiczny…
- **Różne funkcje dla różnych użytkowników?**
- **Gadget dla każdego?**
- **Bezpieczeństwo: Polkit || Cynara**

TIZEN™

gadgetctl

- **Referencyjny klient dla gadgetd**
- **Kod współdzielony z gt**
- **Narzędzie command line dla gadgetd**
- **Używa D-Bus'a**
- **Rozwijany jako backend dla gt**
- **WorkInProgress**

TIZEN™

Przykład

Jedna konfiguracja, jedna funkcja FunctionFS

```
$ gadgetctl create g1 \
    idVendor=0x04e8 \
    idProduct=0x6865 \
    manufacturer="Samsung" \
    serialnumber="A123BC1" \
    product="Galaxy S3"

$ gadgetctl func create g1 ffs.sample i_name
$ gadgetctl config create g1 label 1
$ gadgetctl config add g1 1 ffs.sample i_name
$ gadgetctl enable g1
```

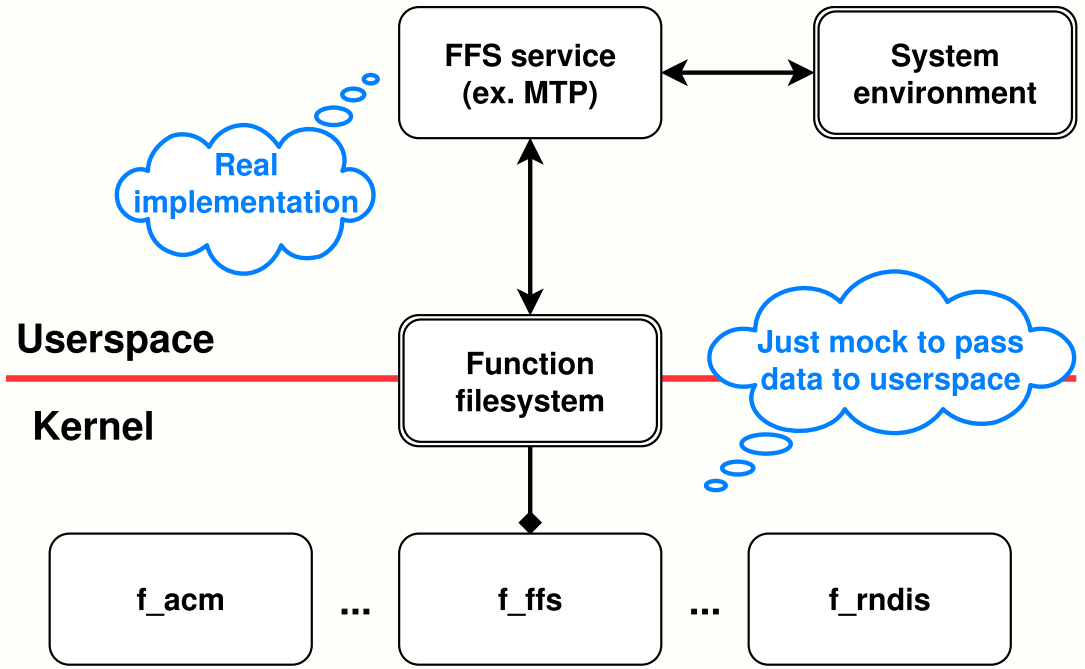

TIZEN™

Podsumowanie

- **Zredukowana liczba poleceń jakie należy wywołać by utworzyć gadget**
- **Zależności i mechanizmy**
- **Unikalne nazwy funkcji FunctionFS**
- **Bezpieczeństwo**

Dygresja

- **Dlaczego każdy komputer(z USB) może być dziurawy?**



Q&A

References

- **Andrzej Pietrasiewicz, Make your own USB gadget**
- **Krzysztof Opasiak, Tame the USB gadgets talkative beast**
- **Matt Porter, Kernel USB Gadget ConfigFS Interface**
- https://github.com/gadgetd/gadgetd/wiki
- https://github.com/libusbg/libusbg
- https://github.com/kopasiak/libusbg
- https://github.com/kopasiak/gt
- https://github.com/hyperrealm/libconfig
- http://lwn.net/Articles/395712/
- https://wiki.tizen.org/wiki/USB/Linux_USB_Layers/Configfs_Composite_Gadget
- https://wiki.tizen.org/wiki/Security:Cynara
- http://www.freedesktop.org/wiki/Software/polkit/

TIZEN™