



Projet d'Ingénierie 014

Liaison filaire USB sous Android

- PING 2020 -

Alexandre ANASTASSIADES, Élise BRUCHET, Arnaud GUIBERT, Quentin MALLÉN,
Alice SHAH HOSSEINI, Julien VERNAY, Florian VUITON

Sommaire



- Attentes
- Composantes du projet
- Résultats obtenus
- Méthodologie & Pilotage
- Retour d'expérience

Attentes : la problématique

Notre client : CIO Systèmes Embarqués

- Tuteur projet : M. Selso Liberado (selso.liberado@ciose.fr, tel std : 04.77.93.34.32)



Nos référents pour le projet :

- Tuteur académique : M. Yves Bringer (yves.bringer@univ-st-etienne.fr, tel std : 04.77.91. 58.88)
- Chef de projet : Alexandre Anastassiades (alexandre.anas98@gmail.com)

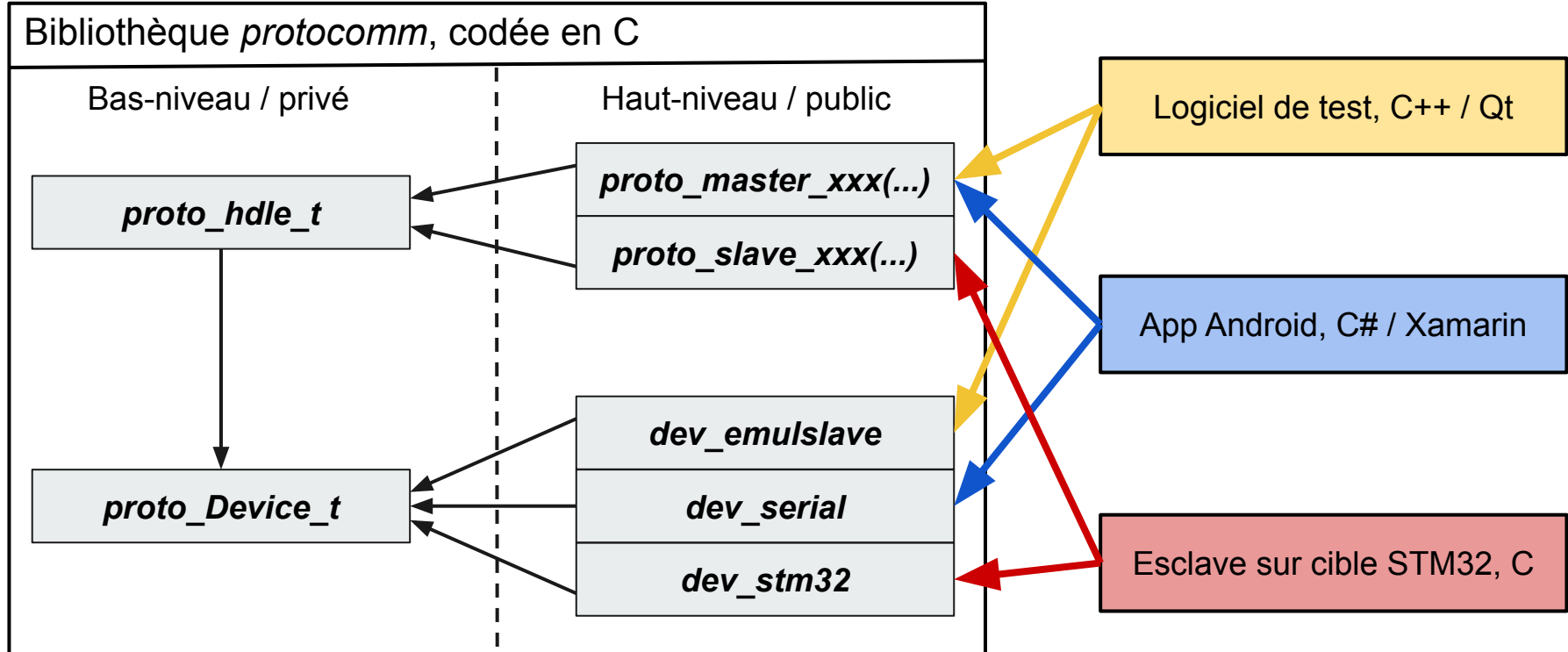
Notre sujet : Liaison filaire USB sous Android

Attentes : les engagements



- Documentation technique sur un protocole de communication basé sur RS232 et son implémentation sous Linux
- Développement d'une interface tablette Android
- Réception et affichage d'information de la carte sur l'interface Android
- Développement d'une interface PC
- Détection de l'USB, envoi et réception de trames en filaire

Composantes du projet



DLL *protocomm* : les devices

Projet cross-plateforme :

- GNU/Linux (Android) : utiliser l'API Linux avec les FileDescriptor
- (Windows : API différente de Linux)
- STM32 : pas d'OS → pas d'API directe
- Mocking : faux device pour réaliser des tests

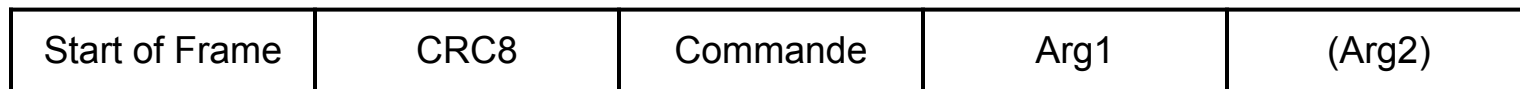
```
proto_Device_t {  
    int (*open)(proto_Device_t _this, ...);  
    int (*close)(proto_Device_t _this);  
    int (*read)(proto_Device_t _this, ...);  
    int (*write)(proto_Device_t _this, ...);  
    void (*destroy)(proto_Device_t _this);  
    void* user;  
}
```



Prog Orientée Objet :

- Encapsulation
- Polymorphisme
- Abstraction

DLL *protocomm* : les trames



Commandes : GET / SET REPLY / ERR_CRC / ERR_ARG

Lecture : on reçoit des blobs d'octets → machine d'état : ***proto_hdlc_t***

API bas-niveau : gestion des erreurs et envoi des trames restent à faire
Symétrique (mêmes fonctions pour maître et esclave)
Asynchrone (non-bloquant, lire la réponse plus tard)

DLL *protocomm* : Slave, implém. haut-niveau

Callback de réception à implémenter :

```
int slave_receive(void* userdata, proto_Command_t command, proto_frame_data_t* args_inout)
```

Utilisation :

```
proto_hdl_t* hdl = proto_slave_create(devXXX_create(), &slave_receive, userdata);
proto_slave_open(hdl, "nom de la connexion (ex: /dev/ttyS0)");
/* ... */
while (1) { /* boucle principale */
    proto_slave_main(hdl);
    /* ... */
}
proto_slave_destroy(hdl);
```


DLL *protocomm* : Master, implém. haut-niveau

Utilisation :

```
proto_hdl_t* hdl = proto_master_create(devXXX_create());
proto_master_open(hdl, "nom de la connexion (ex: /dev/ttyS0)");

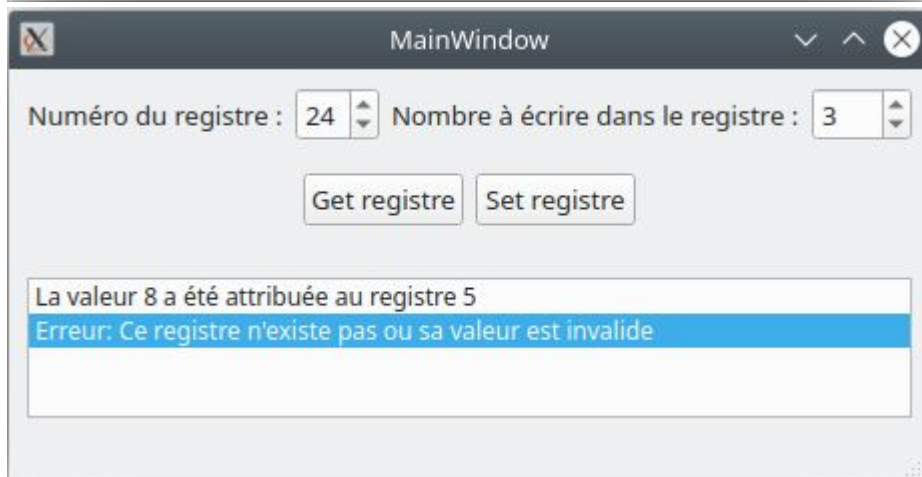
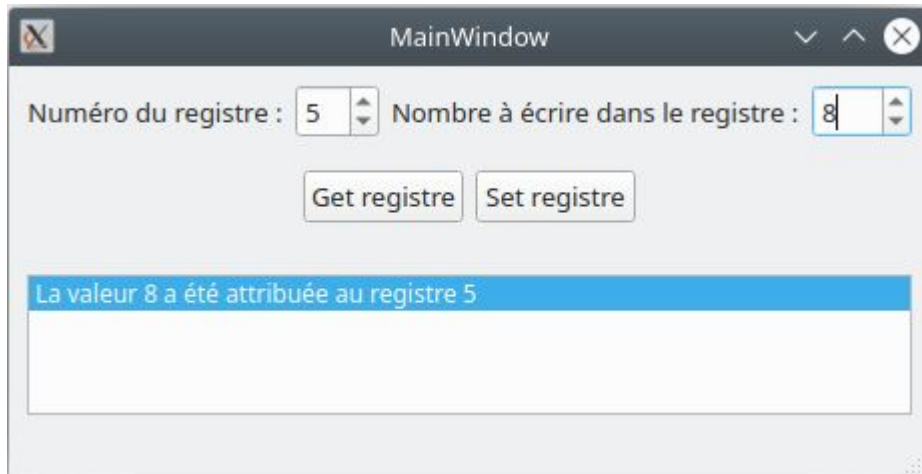
/* ... */
proto_Status_t status = proto_master_set(hdl, 5, 10); /* registre 5 vaut 10 */
/* ... */
uint8_t output;
status = proto_master_get(hdl, 5, &output); /* lire le registre 5 */
/* ... */

proto_master_destroy(hdl);
```

→ Synchrone : on attend la réponse à chaque fois

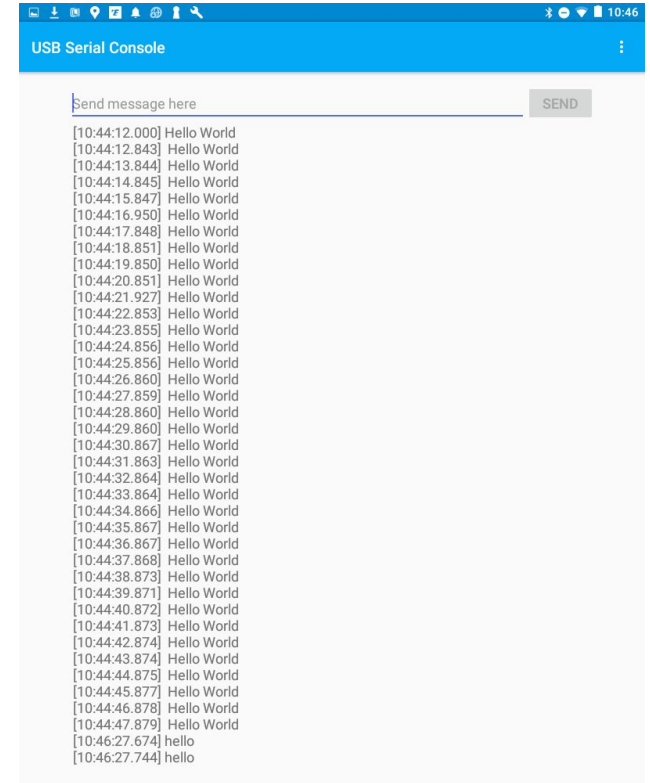
IHM sur PC

- Utilisation de la DLL protocomm
- Interface graphique de la DLL



Développement cible (NUCLEO STM32F439ZI)

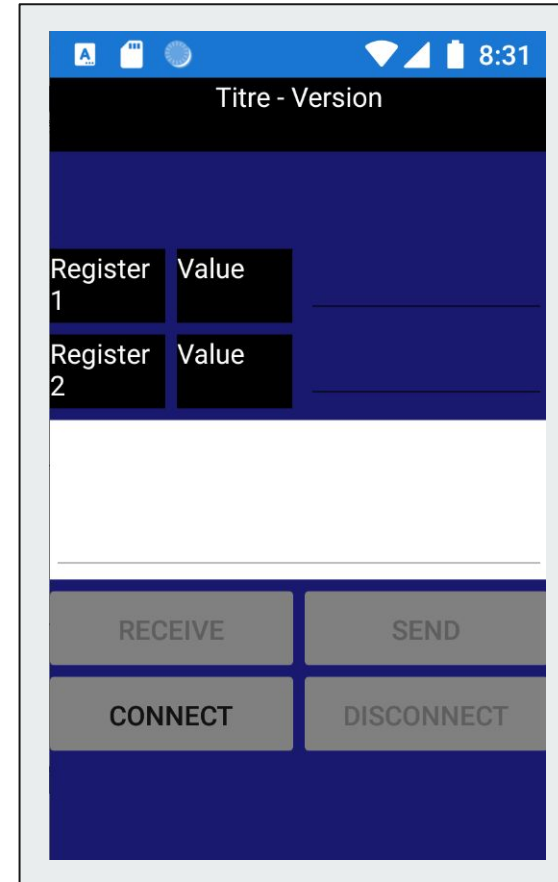
- Création d'un projet avec le logiciel STM32 CubeMX
- Compléter certaines fonctions
- Réalisation de différents test : écho (sur la tablette avec l'application USB Serial Console), envoi en continu d'un message, ...
- Intégration du protocole "protocomm" dans le projet : le service STM32 permet de lire dans une FIFO et écrit dans l'USB



```
[10:44:12.000] Hello World
[10:44:12.843] Hello World
[10:44:13.844] Hello World
[10:44:14.845] Hello World
[10:44:15.847] Hello World
[10:44:16.950] Hello World
[10:44:17.848] Hello World
[10:44:18.851] Hello World
[10:44:19.850] Hello World
[10:44:20.851] Hello World
[10:44:21.927] Hello World
[10:44:22.853] Hello World
[10:44:23.855] Hello World
[10:44:24.856] Hello World
[10:44:25.856] Hello World
[10:44:26.860] Hello World
[10:44:27.859] Hello World
[10:44:28.860] Hello World
[10:44:29.860] Hello World
[10:44:30.867] Hello World
[10:44:31.863] Hello World
[10:44:32.864] Hello World
[10:44:33.864] Hello World
[10:44:34.866] Hello World
[10:44:35.867] Hello World
[10:44:36.867] Hello World
[10:44:37.868] Hello World
[10:44:38.873] Hello World
[10:44:39.871] Hello World
[10:44:40.872] Hello World
[10:44:41.873] Hello World
[10:44:42.874] Hello World
[10:44:43.874] Hello World
[10:44:44.875] Hello World
[10:44:45.877] Hello World
[10:44:46.878] Hello World
[10:44:47.879] Hello World
[10:46:27.674] hello
[10:46:27.744] hello
```

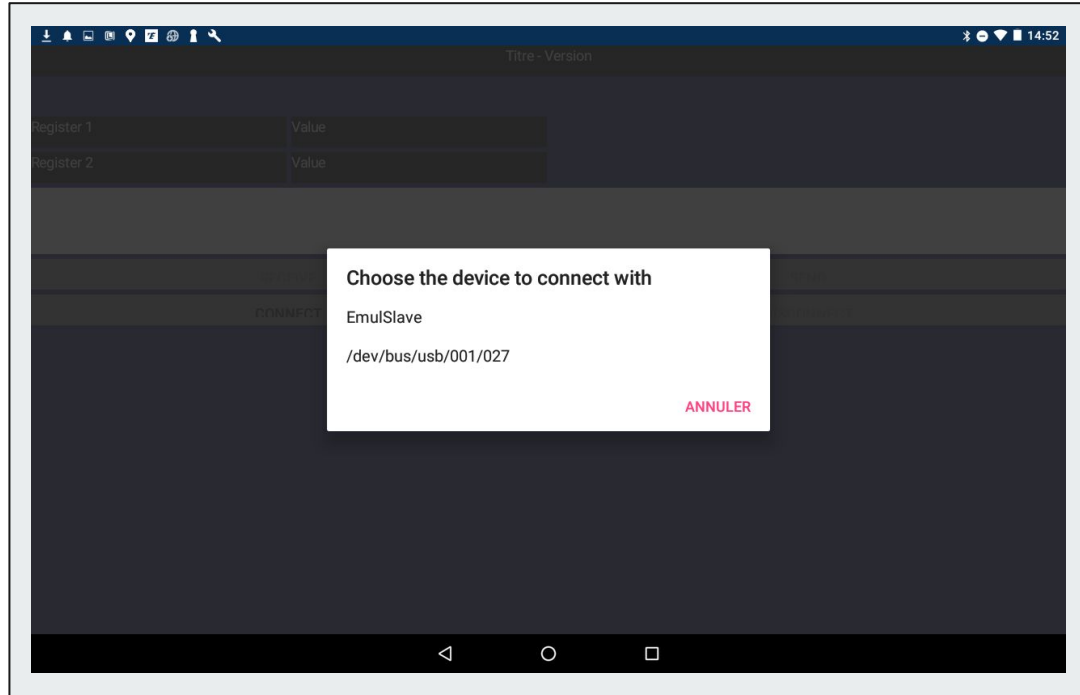
IHM Android

- L'IHM Android gère l'ouverture, la fermeture et l'accès aux ports sous Android
- Choix de UsbManager de Android
- Elle utilise ensuite la DLL protocomm pour réaliser l'envoi et la réception des trames de données à la cible (carte Nucleo) connectée en filaire



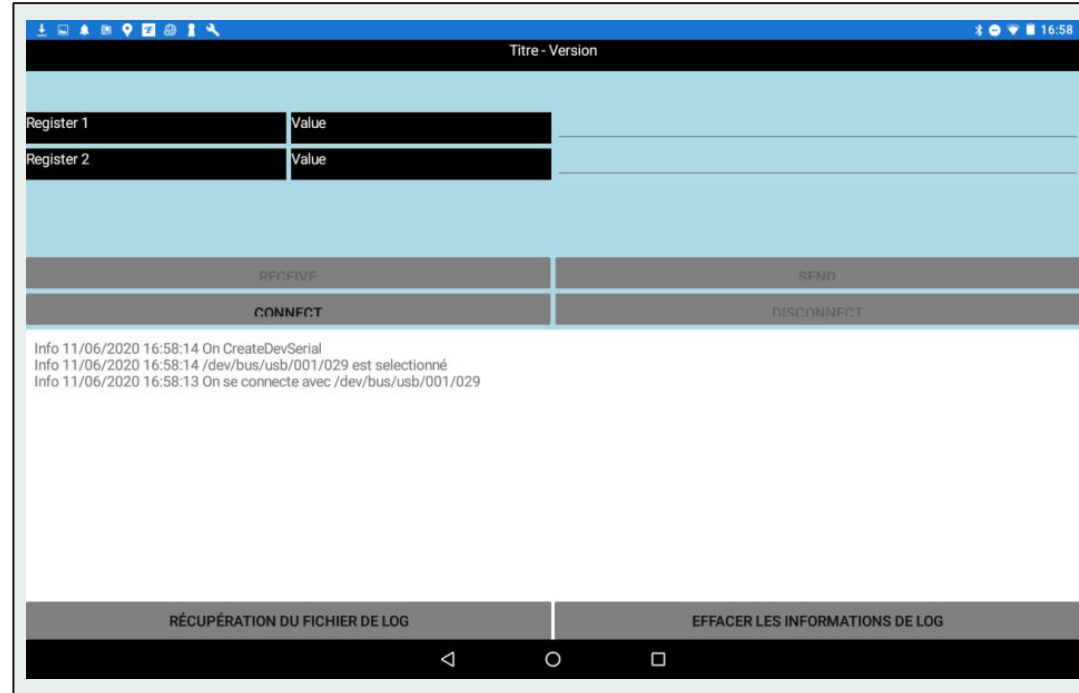
Résultats obtenus

- Déploiement de l'application sous Android
- Détection de la carte



Résultats obtenus

- Crash de l'application
- Ajout de logs pour pouvoir déboguer



Méthodologie & Pilotage : Organisation



Application de la méthode agile.

Séparation en trois sous-groupes (avec changement à mi-projet) :

- interface Android : Alexandre Anastassiades
- DLL protocomm et interface PC : Julien Vernay
- développement cible: Elise Bruchet

Méthodologie & Pilotage : les Risques

- Risques majeurs liés à la découverte de nouveaux langages et IDE
- Dans l'ensemble, les risques ont été correctement établis (avec le plus gros point bloquant lié à l'utilisation du C#)

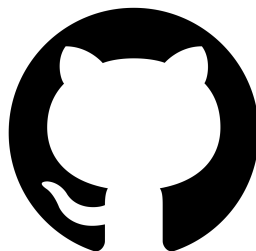
Risque	Sévérité
Covid-19	$3 \times 2 = 6$
Risques matériels	$2 \times 2 = 4$
Détection filaire	$2 \times 2 = 4$
“Attolic True Studio”	$3 \times 2 = 6$
Visual Studio Xamarin	$3 \times 2 = 6$
Découverte C#	$3 \times 3 = 9$

Méthodologie & Pilotage : les Outils



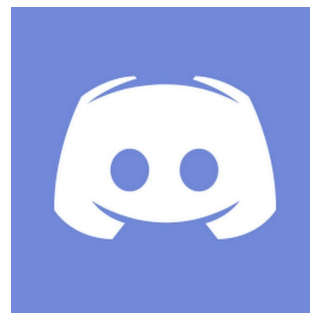
Webex :

- Réunions avec PO



Github :

- Versioning
- Workflow



Discord :

- Réunions en interne
- Avancement des sous-groupes



Google Drive :

- Partage de documents
- Collaboration temps réel

Retour d'expérience



Fin



N'hésitez pas à poser vos questions !