Le Raspberry PI sur les Unités Expérimentales

Jean-François Bompa[1], Julien Ancelin[2]

Résumé. Malgré leurs cures d'amaigrissement successives, nos PC se retrouvent, depuis peu de temps, face à de nombreuses petites cartes électroniques de développement au prix défiant toute concurrence. La modularité de ces cartes et leur appartenance au monde du logiciel libre ont attiré notre attention. Peut-être la fin du PC pour certains de nos projets ?

Mots clés : Raspberry, Docker, polyvalence, mono, automatisme

Introduction

La recherche expérimentale se nourrit d'une quantité croissante de données. Dans les prés ou les bâtiments d'élevage des UE (Unités Expérimentales) ou IE (Installations Expérimentales) de l'Inra, ces informations peuvent être saisies soit manuellement par le constat d'opérateurs, soit à l'aide de systèmes automatiques. Les PC, même s'ils ne sont pas toujours bien adaptés, sont souvent utilisés pour assumer ce rôle d'enregistrement de données. Nous allons vous présenter, ici, une alternative beaucoup plus personnalisable, ultra polyvalente, peu onéreuse et appartenant au monde de la communauté open source de Linux : le Raspberry PI (mini serveur linux) peut être utilisé aussi bien en tant que serveur «de poche» pour l'extérieur, ou en tant que système automatique de mesures.

Tout est bon dans la framboise

Le Raspberry PI (RPI) est une mini carte sous Linux, de la taille de celles qui encombrent nos portefeuilles (hormis l'épaisseur), il a un coût très faible : moins de 40 €. Il est développé par la Fondation Raspberry PI, une association caritative anglaise dont les créateurs font partie du laboratoire d'informatique de l'Université de Cambridge. Les premiers exemplaires sont apparus début 2012, et depuis 12,5 millions auraient été vendus, tous modèles confondus…

Caractéristiques du Raspberry PI

À l'heure actuelle, le RPI modèle B en est à sa troisième version (**Figure 1**).

Le port GPIO, comprend entre autres, des ports I²C, SPI et TTL.

L'OS est installé sur une carte SD. Plusieurs distributions linux sont disponibles mais la plus standard est Raspbian Jessie. Le RPI n'est pas le seul de sa catégorie ni le plus performant, mais il conserve un excellent rapport taille/équipement/prix et bénéficie d'une énorme communauté d'utilisateurs et d'un magazine dédié, MagPi. De plus, il existe de nombreuses cartes (shield) optionnelles en tout genre, qui étendent ses fonctionnalités (**Figure 2**) : lcd, écran tactile, écran E-inck, 3g, GPS, zig-Bee, Lora, Sigfox, CAN, RS232/485, RFID, caméra, relais, servomoteurs, Can/CNA…

1 *UMR GenPhySE, Inra, Chemin de Borde Rouge, 31326 Castanet -Tolosan, France*
2 *Plateforme DISA, Inra, La Rochelle, 545 rue Bois Maché,17450 Saint-Laurent-de-la-Prée, France*
jean-francois.bompa@inra.fr

Figure 1.
Raspberry PI V3 (fr.farnell.com).

– CPU ARMv8 quatre cœurs 64 bits à 1,2 GHz Videocore IV 3D graphique
– WIFI 802.11 n
– Bluetooth 4.1, Bluetooth BLE
– 1 GB RAM
– 4 ports USB2
– GPIO 40 broches
– 1 Port HDMI
– 1 port Ethernet 10/100
– 1 sortie jack audio + composite
– 1 pot interface camera (CSI)
– 1 pot interface écran (DSI)
– 1 slot µSD

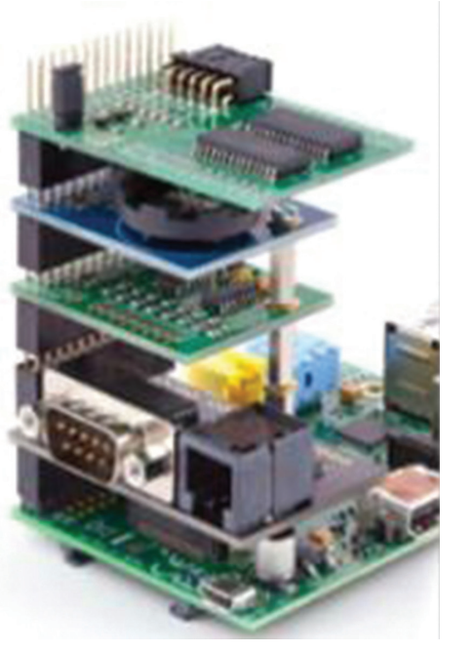


Figure 2. *RPI avec cartes optionnelles (www.framboise314.fr).*

Le RPI version serveur mobile avec Docker (utilisation en extérieur)

Il est assez aisé de déployer des solutions logicielles « métier » sur serveur et garantir la saisie au bureau et l'accessibilité des bases de données via le web. Sur le « terrain », il est indispensable de pallier les problèmes d'ergonomie, de polyvalence, d'accessibilité et de synchronisation des informations collectées. Les observateurs, pour des raisons évidentes de compétence, doivent également avoir des interfaces unifiées entre les différents outils de terrain et leur bureau. Le RPI étant l'équivalent, en miniature, d'un serveur, il est donc possible de déployer les mêmes webservices métiers et bases de données que sur un serveur standard et donc de pouvoir l'emmener simplement sur le terrain.

Les prérequis matériels

L'architecture matérielle tourne autour de trois composants physiques : une alimentation électrique portable, un Raspberry Pi 3 et une tablette. Un récepteur GNSS (global navigation satellite system) peut également être rajouté pour les besoins de géolocalisation haute précision.

La batterie doit avoir une capacité correspondant au temps d'utilisation sur le terrain, en plus d'être légère et peu encombrante. Une capacité de 10000 mA permet largement de tenir une journée. Il est recommandé d'utiliser une alimentation 5V 2.4A minimum pour le Raspberry Pi 3 car il faut alimenter le Wi-Fi (les pointes de courant pouvant atteindre 2 ampères par intermittence).

L'antenne Wi-Fi permet l'accessibilité et l'interconnexion avec des machines standards. Souvent utilisée pour se connecter à un réseau existant, il est possible d'inverser le processus pour en faire un point d'accès (Hotspot). Dans cette configuration, 1 à n matériels (smartphone, tablette, PC), sur une distance maximum de 20 mètres, peuvent se connecter sur le RPI (et donc accéder à une ou plusieurs interfaces de travail). Cette

possibilité renforce le travail coopératif sur le terrain et l'interaction entre les observateurs. La mise en place d'un point d'accès est possible grâce à un script Bash qui installe et paramètre automatiquement la mise en service. (https://github.com/jancelin/rpi_wifi_direct).

Côté utilisateur, une simple tablette servira d'écran (**Figure 3**). Il n'est pas nécessaire d'avoir un matériel puissant car la plupart des traitements informatiques sont effectués sur le RPI. Un navigateur web, comme Firefox ou Chromium, sera l'interface entre la machine et l'utilisateur.

Figure 3. *Principe du serveur mobile (Julien Ancelin Inra UE SLP).*

Gérer une usine logicielle

Une usine logicielle est un ensemble d'outils et de méthode(s) permettant aux développeurs et architectes logiciels de mener à bien leur mission afin de développer de manière simple et itérative. Dans notre cas, comment déployer, maintenir et mettre à jour, sur x machines d'un parc, des outils d'administration système simple, des bases de données d'une même architecture que notre serveur central et des interfaces métiers cartographiques ou tabulaires ?

Docker

Une solution est d'utiliser Docker, qui permet d'embarquer, dans un container virtuel, une application qui pourra s'exécuter sur n'importe quelle machine. Le container fait, en effet, directement appel à l'OS de sa machine hôte pour réaliser ses appels système et exécuter ses applications. Historiquement, Docker repose sur le format de containers Linux, alias LXC. À la différence de la machine virtuelle, le container est plus léger et n'a pas besoin d'activer un second système pour exécuter ses applications. Cela se traduit par un lancement beaucoup plus rapide, un empilement des applications, mais aussi par sa capacité à migrer plus facilement un container d'une machine physique à une autre.

Architecture logicielle

Il est important que chaque utilisateur puisse être libre d'installer l'usine logicielle correspondant à son besoin. Docker permettra grâce à des commandes simplifiées, de lancer les instances, lier les containers de données vers les containers d'applications et d'adresser les services.

Pour installer et lancer l'ensemble des briques logicielles sur une machine, nous utiliserons Docker-compose qui agit comme un chef d'orchestre. Une fois le fichier docker-compose.yml paramétré et chargé sur le RPI (https://github.com/jancelin/geo-poppy/blob/master/docker-compose.yml), la commande « docker-compose up » chargera les images et lancera l'ensemble des containers.

Figure 4. *RPI avec Sense Hat (http://www.newark.com).*

Extension

La carte Sense Hat (**Figure 4**) peut-être rajoutée sur le RPI afin de permettre à l'utilisateur d'interagir physiquement avec la machine et de profiter de quelques capteurs (accéléromètre,

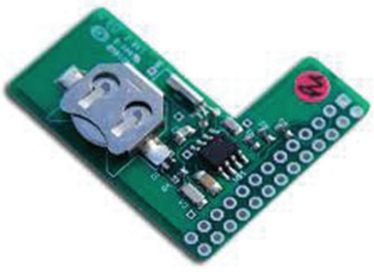
gyroscope, température, pression atmosphérique). Programmé en Python, ce script (https://github.com/jancelin/geo-poppy/blob/master/sense-hat/command.md) permet, grâce au joystick et à l'écran LED, de redémarrer ou d'arrêter le RPI, de connaître son adresse IP et de relancer le container.

Le RPI version PC, pour les automatismes

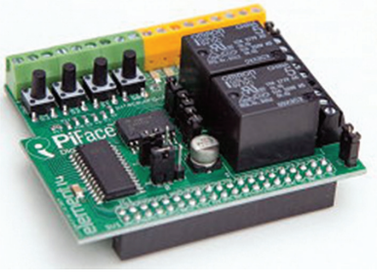
Les besoins d'automatismes sur les UE et IE animales sont variés : distributeurs d'aliments, pesée, tri, suivi par identification, contrôle laitier, études comportementales… Ces systèmes sont principalement commandés par des cartes électroniques propriétaires dédiées ou des automates qui gèrent les parties capteurs actionneurs. N'oublions pas notre indispensable PC qui supervise le tout et ajoute du confort d'utilisation. Le RPI avec son OS et ses cartes optionnelles, permet de réunir dans un même appareil, la gestion des entrées/sorties, la souplesse et la puissance d'un petit PC. De plus, on reste dans le monde open source communautaire et standard de linux.

Les accessoires indispensables

Pour moins de 300 € d'options, on fabrique un système hybride PC/automate (**Figure 5**).



(1) RTC Shym (fr.farnell.com)



(2) PiFace (fr.farnell.com)

(3) ADS1115 (www.adafruit.com)

(4) USB-COM232-Plus4 (fr.rs-online.com)

(5) Ecran 7" (fr.rs-online.com)

(6) DD Sony SLBG1 (www.focus-numerique.com)

Figure 5. *Les composants d'un système hybride.*

De gauche à droite. **(1)** La carte horloge temps réel RTC shym de PiFace avec pile lithium. **(2)** La PiFace 2 met à disposition sur des borniers, 8 entrées numériques, 4 boutons poussoirs, 8 sorties à collecteurs ouverts équipées de led (dont 2 sorties équipées de relais 10A). Elle se broche directement sur la RPI. **(3)** Le convertisseur analogique/numérique 4 entrées de 16 bit. **(4)** Le convertisseur USB/4x RS232 pour communiquer avec les lecteurs, RFID ou les systèmes de pesée. **(5)** L'écran tactile 7», 800x480 pixels, branché sur le connecteur DSI. **(6)** Le disque dur micro ssd 128 G.

Tous ces produits se trouvent chez les grands revendeurs qui ont pignon sur net.

Échange carte SD contre disque dur…

Un des points faibles du RPI est d'avoir son OS sur une carte SD, qui traîne la mauvaise réputation de fragilité dans le temps. Dans nos systèmes, une quantité très importante de données sont écrites sur cette carte SD, qui a un nombre limité d'écriture. Pour éviter le drame qui se profile, on transfère l'OS de la SD sur un disque dur USB externe qui, lui, supporte très bien tous ces accès mémoire et reste plus fiable dans le temps. Avec la dernière version du RPI, on peut maintenant booter directement sur un disque USB (après configuration spéciale) ce qui permet de se dispenser complétement de la carte SD. Une épine de ronce en moins.

Quels langages de programmation ?

Le Python aime-il les framboises ? *A priori* oui, puisque c'est le langage officiel de programmation installé par défaut sur le RPI. Mais on se doute bien que tout langage compilant sur un ARMv6 devrait fonctionner : Java, C, Scratch…

Les outsiders, Mono et C#

La plupart de nos logiciels sur PC et pocket PC sont développés en C# avec Visual Studio. Même si cela ne paraît pas évident, l'idéal est de pouvoir utiliser ce même langage et environnement de programmation pour le RPI. Au risque de choquer bien des linuxiens, c'est devenu possible grâce au framework Mono. Ce dernier est l'implémentation libre du framework .NET de Microsoft. Mono, qui appartient maintenant à la société Xamarin, est très performant, abouti et mis à jour. Framboise sur le gâteau, il nous permet de reprendre la quasi-totalité du code utilisé sur PC. Du coup, les programmes pour RPI sont développés et testés en majeure partie sur PC pour plus de confort, hormis bien sûr pour les parties de matériels spécifiques. Par ailleurs, nous avons développé une classe C# qui permet de simuler les entrées/sorties de la carte PiFace, directement sur PC.

Installer framework mono: apt-get install mono-runtime Projet C# Piface : https://github.com/mikeclayton/PiFaceSharp

Régime fruité pour tout le monde

La grande majorité de nos systèmes automatisés fonctionnent suivant ce principe (**Figure 6**) : les RPI + PiFace (ou PC + carte E/S) sont connectés aux capteurs/actionneurs, à un lecteur RFID et à un système de pesées. Les données collectées par le programme sont enregistrées dans une base de données. Ces appareils sont reliés au réseau Ethernet pour faire une copie en temps réel des données vers un serveur distant. Cette connexion permet aussi de contrôler et de mettre à jour le RPI à distance.

Coffret « universel »

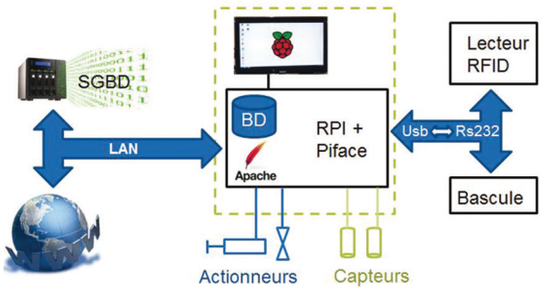


Figure 6. *Principe de fonctionnement des systèmes automatisés.*

Figure 7. *Coffret universel (Photo : JF Bompa).*

Un coffret dit « universel » (**Figure 7**) est en cours de finalisation. Il a pour but d'harmoniser le matériel et le câblage entre les différents systèmes d'automatisme pour le contrôle d'alimentation des animaux (foin, concentrés, eau, lait …). Il comprend une RPI, une carte PiFace, un écran tactile, un convertisseur USB/4xRS232, une RTC, un disque dur USB. Tout est relié sur connecteurs débrochables : alimentation électrique, E/S, Ethernet, USB, RS232. Ce coffret devrait permettre de réduire les coûts, l'encombrement et aussi être facilement interchangeable, ou adaptable, entre les différents automatismes.

Conclusion

La volonté de trouver une alternative au PC, plus économique et intégrable semble avoir porté « son fruit » avec le mini serveur Linux Raspberry. Grâce à sa grande polyvalence et à ses innombrables possibilités de customisation il est aussi bien à l'aise pour nos besoins de saisie en extérieur que pour nos automatismes d'élevage.

Depuis l'avènement du RPI, les mini-cartes de développement sont en plein essor dans les communautés de développeurs. De plus, le fait qu'elles possèdent un OS très répandu et communautaire les rend plus standards et très souples d'utilisation. Le choix d'intégrer ce type de technologie dans nos projets semble donc s'affirmer et nous offre de grandes possibilités d'évolution.