|  |
| --- |
| AceTeaM |
| Projet Arduino |
| Installation et Pilotage Domotique |
|  |
| **Thomas AUGUEY – Vincent AUGUEY – Michael Melkior** |
| **03/04/2013** |

|  |
| --- |
| Développement logiciel du pilotage d’une plateforme domotique basée sur l’architecture Arduino. |

Contenu

[Objectif 2](#_Toc381173519)

[Processus de communication 2](#_Toc381173520)

[Serveur 3](#_Toc381173521)

[Equipement 3](#_Toc381173522)

[Message 3](#_Toc381173523)

[Client 3](#_Toc381173524)

[Administration 4](#_Toc381173525)

[Sécurisation de fonctionnement 4](#_Toc381173526)

[Serveur 4](#_Toc381173527)

[Configuration 4](#_Toc381173528)

[Droits 4](#_Toc381173529)

[Applications 5](#_Toc381173530)

[Administration 5](#_Toc381173531)

[Client 5](#_Toc381173532)

[Serveur 5](#_Toc381173533)

[Matériel/Logiciel Requis 6](#_Toc381173534)

[Problèmes soulevés 6](#_Toc381173535)

[Applications Possibles 7](#_Toc381173536)

[Etude de marché 8](#_Toc381173537)

[Varnua3 (Hestia) 8](#_Toc381173538)

[Model de données 9](#_Toc381173539)

[Entités 9](#_Toc381173540)

[Outils 10](#_Toc381173541)

[Visual Studio (Plugin) 10](#_Toc381173542)

[Annexe 11](#_Toc381173543)

# Présentation Générale

## Objectif

Le système étudié doit permettre d’automatiser et de contrôler le fonctionnement des équipements électriques d’un habitat via l’informatique (en d’autres termes : faire de la domotique).

Le projet Arduino doit nous permettre d’automatiser et de contrôler le fonctionnement des équipements électriques d’un habitat via l’informatique (en d’autres termes : faire de la domotique).

## Arduino

Arduino fait référence à un modèle de carte électronique programmable, le projet est fondé sur ce matériel pour diverses raisons citées ci-dessous.

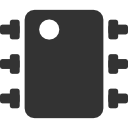
**Points importants :**

* L’application de control doit être simple et conviviale
* Le contrôle des équipements doit être centralisé en une seule application

**Avantages de l’Arduino :**

* Faible coût d’achat
* Programmation aisé en langage C++
* Outils de développement existants (IDE, Librairies, Documentation)
* Nombreux modules existants prêts à l’emploi (Wifi, Ethernet, Capteurs, …)
* Communauté active sur Internet
* Licence libre d’exploitation

# Environnement



**(5) BDD**

(3) Serveur de Communication

**Environnement technique**

## Serveur de communication

(6) Flash

Connexion physique

Connexion sans fil

(2) PINS

(1) Arduino

(4) WIFI

(5) GSM

(3) USB

**Communication de Communication**

Le **Serveur de communication** est une carte Arduino avec un nombre limitées d’entrées-sorties analogique/digitale que nous appellerons PIN. Une PIN Analogique peut transmettre des valeurs allant de 0 à 255. Une PIN Digital peut transmettre des valeurs binaires 0 ou 1.

L’Arduino embarque un programme chargé d’assuré la communication entre les équipements et les clients. La programmation de l’Arduino est réalisée via la connexion USB avec un PC exécutant l’IDE Arduino.

La communication est assurée par le module WIFI via le réseau informatique. Un module GSM pourra également être présent pour la communication avec les clients mobile.

Les équipements sont connectés sur les Pins de la carte. La configuration permet de retenir leurs types et numéro de Pin utilisé. Cette configuration est stockée dans la BDD sur le module Flash.

Enfin l’Arduino serveur devra offrir des performances optimales pour assurer la bonne communication entre les éléments de l’installation.

Communique avec :

* les équipements
* les applications clientes
* le PC de programmation

Modules embarqués :

* WIFI Shield
* SD Card (Flash)
* GSM

## Equipement

Les cartes Arduino secondaires sont appelées **Equipement**. Elles contrôlent un sous-ensemble de modules (ex : Leds, Sondes, Boutons, cartes esclaves, etc…). Le processus de communication avec les modules est indépendant de la communication avec le serveur.

Le serveur communique avec les équipements par **message.** La transmission est assurée par des trames réseau ou les E/S de la carte.

## Client

Un client est une application qui communique avec le serveur via le réseau. L’application est destinée à l’utilisateur final, un travail particulier doit être apporté à l’ergonomie et au visuel.

## Communication

Un message est une action ou une information transmise d’une application à une autre (ex : du serveur vers un équipement). Il peut par exemple ordonner l’allumage d’une lumière, le régalge d’une couleur de leds, etc…

Le système de messages joue un rôle d’abstraction la partie matériel et logiciel de chaque application. Pour ce faire, un standard de communication doit être définit. Ce standard assuré par l’utilisation d’un texte brut encodé au format **XARG** (voir annexe).

La syntaxe **XARG** définit un tableau associatif de paires nom/valeur. L’ensemble de ses valeurs informe sur l’objectif et la destination du message.

Un message comporte au moins 2 champs **EQUIP** et **TYPE**. L’exemple ci-dessous définit une commande permettant d’allumer la lumière de l’équipement LIGHT1.

|  |  |
| --- | --- |
| **Nom** | **Valeur (exemple)** |
| EQUIP | LIGHT1 |
| TYPE | CMD |
| ACTION | ON |

L’encodage et le décodage est assuré par la libraire **XARG** incluse dans le projet **NoyauPortable (NP)**.

## Administration

L’administration est l’unique application permettant de configurer le serveur. Son accès doit être limité aux personnes autorisées.

# Sécurisation de fonctionnement

## Serveur

Le Serveur Arduino est la pièce maitresse du fonctionnement de l’installation. Elle doit être indépendante de toutes autres applications du réseau.

Pour sécuriser et garantir le fonctionnement de l’installation, le serveur doit réagir convenablement en cas de problème :

* Si un redémarrage du serveur survient suite à une mauvaise manipulation ou une coupure d’électricité, le serveur doit être capable de reprendre son fonctionnement normal

## Configuration

La configuration « critique » des équipements doit être accessible à tout moment par le serveur Arduino. Pour ce faire, les données doivent êtres stockées dans la carte SD du serveur Arduino.

Les données critiques sont essentiellement la liste des équipements et plus précisément pour cahque équipement :

* L’identifiant, pour faire le lien entre la carte physique et son nom attribué
* La connectivité physique au niveau de la carte (numéros et rôles des PINS utilisés)

## Droits

Il existe 2 types de rôle dans l’application : L’Administrateur et L’utilisateur.

L’utilisateur ne peut pas exécuter certaines fonctions contrairement à l’administrateur qui lui, a tous les droits. Ainsi pour valider l’identité de l’application, l’administration joint un code de vérification à chacun des messages transmit. Ce code n’est connu que par le serveur qui a la responsabilité de le vérifier avant d’exécuter l’action.

# Applications

## Administration

**Fonctions :**

* Ajouter/Supprimer/Modifier un équipement
* Envoyer/Recevoir des messages avec le Serveur

## Client

**Fonctions :**

* Envoyer/Recevoir des messages avec le Serveur

## Serveur

**Fonctions :**

* Envoyer/Recevoir des messages avec l’Administration
* Envoyer/Recevoir des messages avec le Client

# Matériel/Logiciel Requis

1. Un contrôleur matériel (L’Arduino)
2. Un contrôleur logiciel (l’Application)
3. Un ou plusieurs équipements analogique/digital

# Problèmes soulevés

Si le contrôle des équipements passe par le tableau électrique (câble alimentation) :

1. Si un groupe d’équipements est relié au tableau en commun
   * **Problème** : Impossible de contrôler un équipement en particulier sans affecter les autres
   * **Solution** : Ajouter un multiplexeur sur l’Arduino
2. L’alimentation est reliée à un interrupteur (éclairage par exemple)
   * **Problème** : Même si un module est installé sur l’équipement, dans certains cas un interrupteur peut empêcher le courant de passer.
   * **Solution**: Installer le module dans l’interrupteur ?

# Applications Possibles

Voici une liste non exhaustive des diverses possibilités d’applications dans une maison :

* Piloter :
  + Donner des ordres aux équipements à distance (application web/mobile)
  + Piloter les équipements par la voie (application centrale)
* Sécuriser :
  + Aide aux personnes âgées *[à développer...]*
* Centraliser:
  + La température (Piloter la température des chauffages)
  + Le déclenchement des volets roulants
  + L’Eclairage
* Alerter:
  + Une présence est détectée
  + Un appareil potentiellement dangereux est allumé après la fermeture de la maison
* Portier high-tech :
  + Interphone / Vidéophone
  + Mise en contact direct par téléphone
  + Sonnerie MP3
  + Laisser un message

# Etude de marché

La plupart de la concurrence actuelle utilise des systèmes communs et onéreux.

*[A développer...]*

## Varnua3 (Hestia)

Propose des installations domotique complètes basées sur le **système KNX.**

SiteWeb : <http://www.hestia-france.com/>

Brochure et Prix : Voir le fichier **Etude/concurrencearnua3-tariffs.pdf**

Prix Installation Estimée

Centrale de contrôle : De **1 591,00 €** à **2 350,00 €**

Logiciel : De **145,00 €** à **694,00 €**

Equipements : De **10€** à **598,00 €** / Unité (sondes, contrôle d’accès, radio, etc…)

# Model de données

Définition.

## Entités

|  |  |  |  |
| --- | --- | --- | --- |
| **CLIENT** | Application un client | | |
| **Champ** | **Désignation** | **Format, Contraintes** | **Type** |
| IP | Adresse IP de l’application | IPv4, Unique | VARCHAR(15) |

|  |  |  |  |
| --- | --- | --- | --- |
| **EQUIPEMENT** | Représente un équipement | | |
| **Champ** | **Désignation** | **Format, Contraintes** | **Type** |
| Identifiant | Nom de l’équipement | Identifier, Unique | VARCHAR(32) |
| Type | Type d’équipement | Identifier | VARCHAR(32) |

|  |  |  |  |
| --- | --- | --- | --- |
| **SERVEUR** | Représente le serveur | | |
| **Champ** | **Désignation** | **Format, Contraintes** | **Type** |
| IP | Adresse IP de l’application | IPv4, Unique | VARCHAR(15) |
| MAC | Adresse Physique de l’application | MAC, Unique | VARCHAR( ?) |

|  |  |  |  |
| --- | --- | --- | --- |
| **PIN** | Représente un connecteur d’équipement | | |
| **Champ** | **Désignation** | **Format, Contraintes** | **Type** |
| IP | Numéro du connecteur physique | Integer | INTEGER |
| MAC | Type (Analogique/Digital) |  | CHAR( 1) |

# Messages

Un message est un texte formaté et structuré qui peut être transmit de diverses façons (Ethernet, Internet, Port Série, …). Le format est basé sur un model simple d’association Nom/Valeur (ou tableau associatif).

## Message d’envoi

|  |  |  |
| --- | --- | --- |
| **LIST** | Liste les équipements | |
| **Champ** | **Désignation** | **Format** |
| Count | Nombre d’équipements | Integer |
| **Retour** | **RESULT** | |

|  |  |  |
| --- | --- | --- |
| **REM** | Supprime un équipement | |
| **Champ** | **Désignation** | **Format** |
| Name | Nom de l’équipement | Identifier |
| Token | Champ de validation | Password |
| **Retour** | **RESULT** | |

|  |  |  |
| --- | --- | --- |
| **ADD** | Ajoute un équipement | |
| **Champ** | **Désignation** | **Format** |
| Name | Nom de l’équipement | Identifier |
| Token | Champ de validation | Password |
| **Retour** | **RESULT** | |

|  |  |  |
| --- | --- | --- |
| **GET** | Obtient les informations sur un équipement | |
| **Champ** | **Désignation** | **Format** |
| Name | Nom de l’équipement | Identifier |
| **Retour** | **RESULT** | |

|  |  |  |
| --- | --- | --- |
| **SWITCH** | Active / Désactive un équipement | |
| **Champ** | **Désignation** | **Format** |
| Name | Nom de l’équipement | Identifier |
| Value | True pour activer, false pour désactiver | Boolean |
| **Retour** | **RESULT** | |

|  |  |  |
| --- | --- | --- |
| **CHECK** | Obtient l’état d’un équipement | |
| **Paramètre** | **Désignation** | **Format** |
| Name | Nom de l’équipement | Identifier |
| **Retour** | **STATE** | |

|  |  |  |
| --- | --- | --- |
| **CMD** | Transmet une commande à un équipement | |
| **Champ** | **Désignation** | **Format** |
| Name | Nom de l’équipement | Name |
| Cmd | Nom de la commande | Name |
| … | Autres paramètres possibles suivant la commande définit | Text |
| **Retour** | **RESULT** | |

## Message de retour

|  |  |  |
| --- | --- | --- |
| **STATE** | Etat d’un équipement | |
| **Paramètre** | **Désignation** | **Format** |
| Switch | True si activer, false si désactiver | Boolean |

|  |  |  |
| --- | --- | --- |
| **RESULT** | Résultat de procédure | |
| **Paramètre** | **Désignation** | **Format** |
| Error | Contexte de l’erreur (ERR\_OK, ERR\_FAILED, ERR\_SYSTEM) | Identifier |
| Message | Code de l’erreur | Identifier |
| … | Autres paramètres possibles suivant le message définit | Text |

# Outils

* L’IDE **Visual Studio** propose un plugin de développement complet pour **Arduino**.
* **NetBeans** pour les applications **Android** et **JAVA**

## ****Visual Studio (****Plugin)

**Il existe un module permettant d’utiliser l’environnement Arduino avec Visual-Studio.**

**Site officiel :** [http://www.visualmicro.com](http://www.visualmicro.com/)

Un installeur pour Windows est disponible dans le dossier suivant : **"Arduino App/VisualStudio Plugin/ArduinoForVisualStudio.msi"**

# Annexe

**XARG** : Format de chaine utilisée pour représenter un tableau associatif de paires Nom/Valeur.

**Arduino** : Plateforme de développement rassemblant cartes mères, librairies et environnement de développement.