Une possible réponse ludique pour les processus industriels communicants

Stéphane Begot, Florent Duculty, Manuel Avila, Pascal Vrignat, Jean-François Millet, Jean-Christophe Bardet

Une possible réponse ludique pour les processus industriels communicants

BEGOT S., DUCULTY F., AVILA M., VRIGNAT P., MILLET J.F., BARDET J-C
stephane.begot@univ-orleans.fr
IUT de l'Indre, 2 Avenue François Mitterrand 36000 CHATEAUROUX

RESUME : La communication sur Internet des processus, fantasme d'hier, est bien la réalité d'aujourd'hui. Dans cet article, nous présentons un processus réalisé par différents groupes d'étudiants de DUT Génie Electrique et Informatique Industrielle et de LP Supervision des Automatismes et des Réseaux. Ce processus illustre un système automatisé et supervisé par une interface distante sur Internet. Il regroupe un large éventail de technologies de communication au travers d'une solution ludique et attractive qui demeure à la hauteur des applications industrielles actuelles. Ainsi, nous souhaitons exposer le savoir-faire de nos étudiants tout en offrant une plate forme pédagogique pour des travaux pratiques en réseaux, automatismes, asservissement, Web 2 ou Interface Homme Machine. La démarche suivie (rédaction de tutoriaux, d'exemples et de documents techniques) doit permettre de reproduire à coût raisonnable une architecture logicielle et matérielle identique.

Mots clés : automatismes, Web 2, supervision, intelligence multi niveaux, réseaux, système pédagogique.

1 INTRODUCTION

Une Licence Professionnelle en Automatisme Réseaux et Internet (LP ARI) a été créée il y a 8 ans sur le site de l'IUT de l'Indre. Lors de sa création, nous avions trois objectifs : mettre à disposition des informations provenant d'une partie commande sur le réseau, superviser un processus via une interface « Web » et former des étudiants avec la double culture automatique-informatique industrielle. Les nombreuses solutions matérielles et logicielles disponibles aujourd'hui sur le marché, montrent que l'univers des solutions de contrôle-commande s'élargit au monde d'Internet même si, entre temps, l'intitulé de la formation a perdu son « I » pour devenir LP « Supervision des Automatismes et Réseaux » (LP SAR). Au sein de notre IUT, nous avons mis en place certains projets montrant l'intérêt de cet interfaçage. Le projet exposé dans cet article est dans la continuité de cette démarche.
Il est notable que le désengagement des jeunes pour la technologie est de plus en plus grand. Cela nous a donc conduit à réaliser un produit ludique qui garderait l'ensemble des caractéristiques technologiques professionnelles que l'on peut exiger dans l'industrie.
Pour garder une approche pédagogique dans ce travail, plusieurs équipes d'étudiants de DUT GEII[1] et LP SAR ont réalisé les différents éléments du système. De plus, des solutions ont été retenues afin de balayer un spectre plus large de technologies.
Dans un premier temps, nous ferons un exposé général du projet. Nous montrerons la place des différents protagonistes ainsi que leur rôle respectif. On verra aussi la gestion de l'intelligence à travers ce système ainsi que la gestion de la mémoire du jeu. Dans un deuxième temps, nous détaillerons les différents éléments du jeu que sont le plateau de jeu, l'API[2], le serveur de jeu ainsi que les différentes interfaces distantes. Enfin, nous conclurons sur l'avancée de ce travail ainsi que sur le rôle des étudiants dans la conduite de ce projet afin que l'expérience puisse être reproduite le cas échéant.

[1] GEII : Génie Electrique et Informatique Industrielle
[2] API : Automate Programmable Industriel

2 DESCRIPTION GENERALE

2.1 Préambule

Certaines entreprises ont d'importantes responsabilités en matière de sûreté. Bien qu'elles accordent un rôle important à l'homme pour la sécurité (l'opérateur en salle de commande est souvent considéré comme la « dernière barrière de sécurité »), elles se méfient des interactions susceptibles d'intervenir entre l'homme et le système technique. Elles attendent donc beaucoup de la technologie comme des procédures.
Le travail théorique des opérateurs est alors défini par rapport à des installations conçues avec un degré d'automatisation parfois élevé, et selon une vision des automatismes qui tente à minimiser l'erreur dans une perspective de fiabilité et de sûreté [1].
L'activité est donc étroitement encadrée par un système de règles mis en place pour répondre aux exigences de sûreté (conduite sur console ou sur tableau, interventions locales, rondes programmées, phases transitoires, recours à la maintenance, transmission d'informations...). Tout tend à être formalisé par des procédures précises.

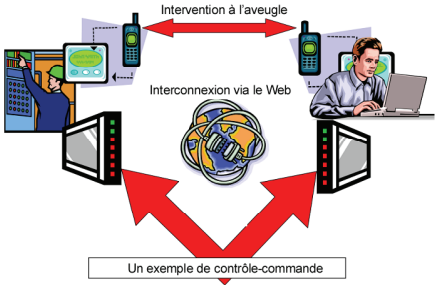

fig.1 : Exemple de contrôle-commande de processus actuel

Difficile de voir le futur, tant les outils de communication sur le sujet se sont focalisés pendant longtemps sur l'aspect graphique au détriment des autres composants. Ces informations sont transmises selon plusieurs moyens, le plus important étant d'avoir quelque soit la

donnée, à un instant *t,* une information unique, cohérente et identifiée parfaitement (fig.1). L'une des dernières évolutions reste le support de transmission, qui du câble série classique, s'est vu rajouter le réseau Ethernet TCP/IP et Internet [5], [10].

2.2 Application au contexte

En partant de cet état des lieux, nous avons entamé l'étude de la mise en place d'un jeu de pions contrôlable via Internet (incluant : les déplacements physiques des pions, les règles du jeu, le retour d'informations sur l'état en cours du plateau de jeu). La partie opérative (plateau de jeu décrit sur la fig. 3) est gérée par un API du constructeur Schneider Electric-Modicon. Les mouvements cartésiens X et Y des axes du robot sont réalisés par des moteurs de technologie Brushless. Le mouvement en Z est géré par un moteur Pas-à-Pas.

Si une personne souhaite jouer au jeu de pions, elle doit se connecter depuis une console locale ou depuis son navigateur à travers le réseau internet. Dans les deux cas, une application d'Interface Homme Machine spécialement développée, permet au joueur de disposer de l'ensemble des fonctions pour jouer et pour visualiser les informations sur le jeu en cours de partie (état du plateau…). De ce fait, un joueur peut être considéré comme « local » (situation où il est le plus proche du processus) alors que le deuxième joueur peut être « distant » en se trouvant sur le « Web ». Dans cette réalisation, il est également prévu qu'il puisse y avoir des spectateurs « distants » qui ne font que observer la partie depuis leur navigateur Internet.

Dans un contexte industriel, les joueurs seraient des opérateurs sur le processus et les spectateurs seraient des superviseurs.

Nous pouvons remarquer, dans ces conditions, que plusieurs besoins « clients » (joueurs ou spectateurs distants) doivent être gérés (fig. 2).

Le serveur de jeu coordonne les différentes demandes des joueurs et sollicite l'API pour qu'il gère correctement et avec cohérence les déplacements X, Y et Z souhaités (fig. 2). Cette contrainte apportée au cahier des charges fonctionnel a fait l'objet d'une séparation volontaire entre « le monde » du contrôle-commande dit de terrain et « le monde » d'Internet. Cette approche permet d'isoler les différents niveaux afin de mieux maîtriser la sécurité de la partie opérative. Cette orientation technologique peut également permettre d'envisager un tel interfaçage avec d'autres parties commandes existantes qui ne disposeraient que de connexion de type série (à condition qu'elles répondent aux exigences et notamment de rafraîchissement dans l'acheminement des informations). Cela a eu aussi l'avantage d'obliger les étudiants à gérer un système dont les capacités décisionnelles (« l'intelligence ») sont réparties sur plusieurs éléments (fig. 6) :

- les variateurs de vitesse de type LEXIUM,
- l'API (Modicon - M340),
- le serveur de jeu.

La nature du jeu n'a pas été arrêtée définitivement, mais la mise en œuvre actuellement opérationnelle s'articule autour d'un jeu de dames. D'ailleurs, il était souhaité que l'ensemble du système ne soit pas figé sur une règle spécifique.

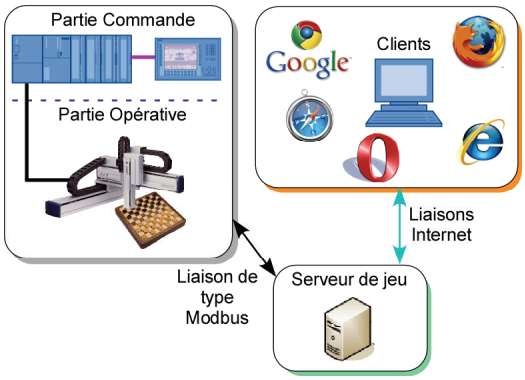

fig. 2 : Vue générale de l'architecture

2.3 Une intelligence du jeu

Comme nous l'avons précisé, cet ensemble pluritechnologique met en œuvre une répartition spécifique des rôles de chacun.

L'API n'est pas en mesure de s'occuper des règles du jeu de manières simples. En effet, pour simplifier la démarche de travail et renforcer sa sécurité, nous avons souhaité que la programmation de celui ci, soit indépendante des règles du jeu. A contrario, il est le garant de la cohérence et de la faisabilité des mouvements demandés à la partie opérative. Il est également le garant, du traitement de l'ensemble des informations remontant de la chaîne de sécurité du processus. Le serveur de jeu quant à lui, scrutant des données de l'API, gère les règles du jeu et documente l'interface avec les utilisateurs distants. De ce fait, l'API dispose d'une « vision » du déplacement des pions d'un niveau plus bas que celle du serveur de jeu. Ce dernier peut être vu comme le « grand chef » qui ordonne le déplacement de sa troupe (par l'API) sur le terrain de jeu. Ce serveur traite donc des mouvements stratégiques (pion A1 vers C3…) et l'API gère des mouvements cartésiens (prendre un pion en X1-Y1-Z, le déplacer en X2-Y2-Z…).

2.4 Mémoire du jeu

La programmation de chaque élément doit bien évidemment intégrer leur champ d'actions respectif tout au long d'une partie (incluant la distribution des pions et leur rangement). La question n'est pas qui sait quoi à l'instant *t*, mais plutôt, qui est la référence de ce savoir à cet instant ?

Après avoir pris en compte, les potentiels dysfonctionnements (électriques, informatiques...), il est apparu comme une évidence, que seul l'API était en mesure de posséder, à tout instant, en son sein, l'état du plateau de jeu (position des pions, état du jeu, …). Cette situation n'est bien évidemment pas unique. On retrouve ce constat dans bon nombre d'applications industrielles. Néanmoins, le serveur doit obligatoirement avoir une image de cette information. Il doit donc, en perma-

nence, se référer aux informations hébergées et rendues disponibles dans l'API, malgré une connaissance à priori de l'état de jeu, pour détecter des contradictions (dysfonctionnements).

API	Serveur de jeu
- Plateau de jeu (positions réelles des pions), - Etat du plateau (actif, en attente, en phase de rangement, en défaut…).	- Joueurs, - Etats logiques du jeu (qui demande, qui peut demander, quoi et pourquoi ?).

Tableau 1 : Répartition de la connaissance

Afin d'avoir une image du jeu accessible en permanence, il a été décidé de réaliser une sauvegarde des données par l'intermédiaire d'un serveur de données MySQL[3], sur la même machine que le serveur de jeu (cette orientation est un choix plus qu'une obligation). Ce serveur de données a son utilité pour les spectateurs « distants » du jeu (cf. AJAX[4]) ou pour reprendre des parties « sauvegardées » (fig. 6).

3 LES DIFFERENTS ELEMENTS

3.1 Le plateau de jeu

N'ayant pas de formation en mécanique dans notre composante, nous avons sous traité la conception et la réalisation de la partie mécanique du plateau de jeu incluant plans de conception et montages à un intervenant vacataire. Ce système confié à des étudiants de 2ème année de DUT GEII en projet tuteuré dès 2009, a été finalisé cette année par un nouveau groupe d'étudiants. Leur mission a consisté à concevoir et réaliser le coffret électrique, à programmer les variateurs de vitesse et l'application finale hébergée dans l'API (partie commande communicante via le réseau CANOpen[5]).

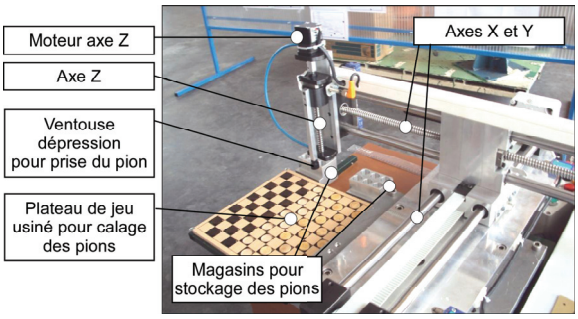

fig. 3 : Plateau de jeu

Ce système est constitué d'un damier où les cases ont été façonnées (creusées) pour assurer un bon calage du pion une fois déposé par la ventouse (élément de préhension et dépose de l'axe Z). Deux postes (magasins) de stockage ont été également insérés dans le système. Dans ces conditions de rangement, un empilage de six pions maxi par case est possible.

3.2 Communication API-serveur (ordinateur)

Comme nous le verrons par la suite, le serveur de jeu est installé sur un ordinateur de type PC. L'API dispose de plusieurs possibilités pour communiquer avec des éléments externes et il est en mesure de gérer plusieurs types de protocoles de communication : Modbus, CANOpen, RS232, RS485, Ethernet…

Nous avons choisi d'utiliser le protocole Modbus sur liaison RS232 en mode esclave côté API et en mode maître côté PC. Ce mode de communication permet au PC serveur d'écrire directement dans les variables de l'API suivant le respect de l'application de la norme IEC 61131-3 ([7]).

3.3 Le serveur ou plutôt les serveurs

Les serveurs sont installés sur un ordinateur PC sous Windows XP. Ils sont au nombre de trois, un serveur de jeu écrit en C++ par nos soins, un serveur web apache et un serveur de données MySQL.

Le serveur de jeu est le lien entre les clients-joueurs et l'API. Il a été écrit en C++ avec adjonction des librairies Winsock, Libmodbus et Libmysql. Il est en mesure de générer des trames Modbus pour communiquer avec l'automate et de dialoguer avec les clients-joueurs par l'intermédiaire de sockets TCP[6]. Enfin, il peut mettre à jour une base de données MySQL.

En dehors des échanges d'informations avec les joueurs, le serveur de données et l'automate, le serveur de jeu a en charge la gestion de la partie et donc des règles du jeu. C'est lui qui, au vu des demandes du joueur, établit si un mouvement est acceptable au sens des règles du jeu. Si c'est le cas, il envoie les ordres en conséquence à l'automate. Une fois ces ordres envoyés, il vérifie leur réception auprès de l'API et attend leur validation (mise à 1 d'une variable automate). Dès que l'automate accepte le mouvement (engagement du mouvement sur le plateau), le serveur prévient les joueurs que ce mouvement est « en cours » et il en profite pour vérifier l'image du plateau de pions de l'automate. Quand le mouvement « réel » est réalisé, l'état d'une variable de l'automate informe le serveur de jeu qui transmet l'information aux joueurs (avec changement de joueur). Par la suite, il met à jour la base de données en archivant le coup et en donnant la nouvelle configuration du plateau de jeu pour les spectateurs.

Le serveur web permet au client de se connecter en tant que joueur ou en tant que spectateur en se référant à des login et mot de passe situés dans la base de données MySQL. Si le client se connecte en tant que joueur, il charge l'applet JAVA et le serveur de jeu prendra le relais. Si le client se connecte en tant que spectateur, il renvoie la page web de spectateur et as-

[3] MySQL : My Structured Query Language
[4] AJAX : Asynchronous JavaScript And XML
[5] CANOpen : couche applicative (7 du modèle OSI), originellement pour les bus de terrain du type CAN (Controller Area Network) fonctionnant en temps réel

[6] mode de communication inter processus sur le réseau TCP/IP (entre autre) avec des sessions TCP

sure la mise à jour de cette interface en répondant aux requêtes AJAX. Pour ce faire, il y a appel à des scripts PHP qui interrogent le serveur de données pour connaître l'état de la partie.
La base de données du serveur MySQL contient un certain nombre de tables permettant la gestion du jeu. Ces données sont les identifiants des joueurs, l'état du jeu, l'état du plateau de jeu et une archive des coups déjà joués. Cette dernière table n'est pas encore complètement utilisée mais elle devrait permettre de reprendre une partie arrêtée (dans un contexte industriel, elle permettrait de savoir les événements précédents une panne ou un arrêt machine quelconque afin d'en connaître la cause).

3.4 Les interfaces distantes

Dans le cadre de la formation des étudiants de Licence Professionnelle SAR, nous abordons les spécifications techniques des RIA[7] du Web2. Connaissant les limitations d'une application de supervision de processus via une interface « Web » (http), les étudiants ont eu à rechercher une solution permettant d'avoir un contrôle et un suivi du système en « temps réel ». Rapidement, deux technologies ont immergé et les deux ont été exploitées afin d'en analyser les mécanismes (fig. 5).
La première utilise une applet[8] JAVA pour s'interfacer avec le client (fig. 4) et la seconde est constituée d'une interface XHTML en utilisant un échange asynchrone grâce à l'objet *XMLHttpRequest* du Javascript.
Par la suite, dans cet article, sans rentrer dans les détails, nous aborderons ces deux approches puis nous verrons leur rôle respectif dans le projet.

3.4.1 Version JAVA

En 2009, les étudiants de LP ont trouvé une interface JAVA permettant de jouer aux dames. Cette interface étant libre de droit, elle a été transformée pour répondre à notre besoin.

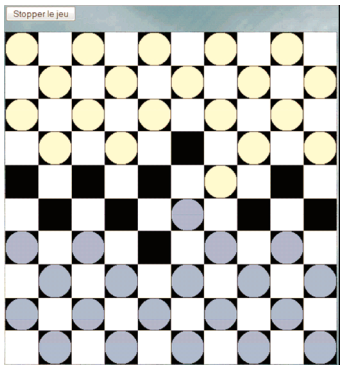

fig. 4 : Interface de jeu distante en Java

L'applet communiquant par socket TCP avec un serveur distant, les étudiants ont donc réalisé le serveur de jeu, écrit en C++, permettant d'interfacer le client à l'API. L'applet JAVA et le serveur de jeu échangent donc des informations par l'intermédiaire des sockets TCP. L'applet du client donne les directives du joueur et le serveur de jeu informe des changements sur le plateau.
Cette solution est efficace, mais elle nécessite le téléchargement de l'applet et l'acceptation du certificat. Il a été convenu que dans le cadre de ce travail, une telle solution ne peut être satisfaisante que pour un joueur (personne qui a confiance dans le jeu). Elle peut être un obstacle pour les spectateurs distants (personne souhaitant utiliser leur navigateur sans téléchargement spécifique).

3.4.2 Version AJAX

Depuis l'arrivée d'Internet, jusqu'au début des années 2000, la communication entre un navigateur et un serveur se faisait une fois et de manière globale (toute l'information était envoyée lors d'une requête [9], [3]).

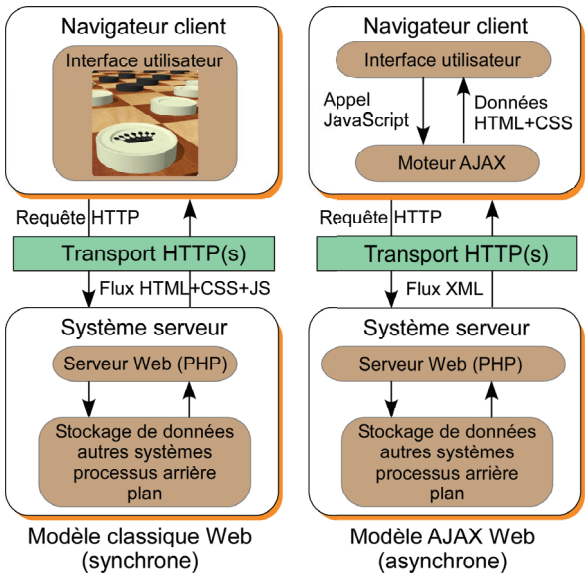

fig. 5 : Les modèles asynchrone et synchrone

Dans ce cadre, pour rafraîchir le contenu de l'information d'une page, il faut rafraîchir l'intégralité de la page (requête synchrone). Il n'était alors pas bien facile de suivre l'évolution d'un système automatisé dans ces conditions. En 2005, Jesse James Garrett comprend qu'il ne faut pas interrompre l'utilisation de la page web par des requêtes au serveur, qui entraînent un gel de la page et nuisent à son utilisation [4]. Dixit Jesse James Garrett, « *Il faut faire circuler sur le réseau le strict nécessaire d'informations pour la demande de la page du client et en tâche de fond, Magister dixit (le maître l'a dit)* ». L'utilisation de l'objet *XMLHttpRequest* de Javascript est un moyen d'avoir des requêtes asynchrones (sans rechargement complet) dans un navigateur (fig. 5). Par ces requêtes, les modifications à apporter à une page, sont envoyées du serveur au client et cela en toute transparence pour l'utilisateur suite à des événements côté client (click souris, timer …). Les calculs et traitements réalisés par le serveur semblent (ou plutôt doivent sembler) être réalisés

[7] Rich Internet Application : Concept d'utilisation d'applications distantes comme si elles étaient locales
[8] Application s'exécutant dans un navigateur web

en local et donc en « temps réel » [8]. C'est l'approche des RIA qui caractérisent le Web de 2ème génération (Web2)

Cette orientation technique répond parfaitement au besoin du cahier des charges fonctionnel préalablement établi (visualiser sans rechargement les évolutions du jeu donc du processus en cours). Ainsi nous pouvons visualiser « en direct » (pour ne dire en « temps réel ») la partie depuis un navigateur et sans opération particulière du spectateur puisque les routines Javascript (AJAX) sont exécutées périodiquement.
La programmation avec cet objet *XMLHttpRequest* devient rapidement laborieuse sans l'utilisation d'un framework[9]. En 2010, des étudiants de licence ont eu pour projet d'évaluer les performances des différents frameworks « libres » afin de trouver le meilleur cadre de développement. Au vu de sa facilité de mise en œuvre, de sa capacité d'évolution et de la réactivité de la communauté qui participe à son développement, le choix s'est porté sur le framework jQuery dans la version 1.3.1 ([6], [2]).
Pour notre « application », dès la validation de l'utilisateur, le client (navigateur) lance en permanence des requêtes AJAX au serveur (toutes les 500ms). Ces requêtes sont sous la forme d'appel à des scripts PHP accompagnées de variables en mode POST. Il existe plusieurs scripts différents qui ne seront pas détaillés ici mais globalement, ils récupèrent les informations de mise à jour du jeu depuis la table de données. Après un traitement adéquat, ces scripts construisent une réponse en publiant des « fragments de code Javascript ». « Ces fragments de code » sont transmis au client et récupérés par la fonction AJAX appelante. Cette dernière évalue (exécute) ces fragments de codes (fonction eval). Lors de cette évaluation, certaines propriétés de la page (l'image d'un pion se trouvant dans une case par exemple) peuvent être changées pour représenter l'état actuel du plateau de jeu.

3.4.3 Serveur MySQL

L'utilisation d'une base de données s'est imposée assez rapidement. La première raison est le besoin de consulter la partie depuis un « client léger » (navigateur web standard). Nous devions donc avoir une table de données archivant les positions des pions sur le plateau. Pour assurer un « minimum de sécurisation » du système, il a également été nécessaire d'avoir une table de personnes habilitées à intervenir (à jouer). Enfin, il est apparu important, d'avoir un historique du jeu afin de pouvoir :

- redémarrer une partie arrêtée volontairement ou non,
- revoir les coups précédemment joués.

[9] Bibliothèques de fonctions et d'outils pour le développement qui définissent un cadre de travail

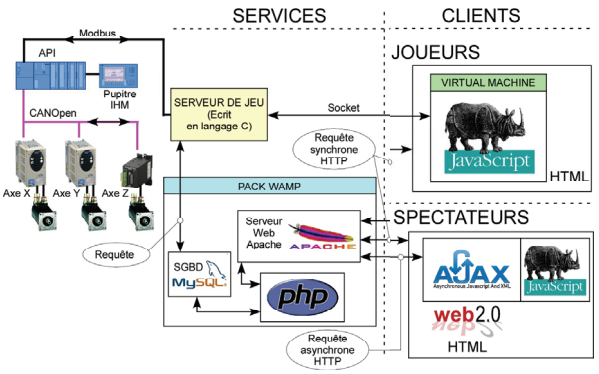

fig. 6 : Schéma de principe de la solution retenue

Ce dernier point nous est apparu comme indispensable pour animer la partie. Il est à noter que dans un contexte industriel, il est souvent primordial d'avoir accès à des données passées sur le système afin de comprendre un dysfonctionnement.

3.4.4 Structure finale

La structure finale et opérationnelle est représentée fig. 6. Cette solution englobe un grand nombre de moyens à la fois matériel et logiciel complexes. La pyramide CIM[10] initiée dans les années 80 et entièrement balayée et mise au « goût du jour ».

4 ETAT D'AVANCEMENT

Le serveur est opérationnel ainsi que le programme API permettant de gérer les modes (Photo 1) :

- automatique,
- semi-automatique.

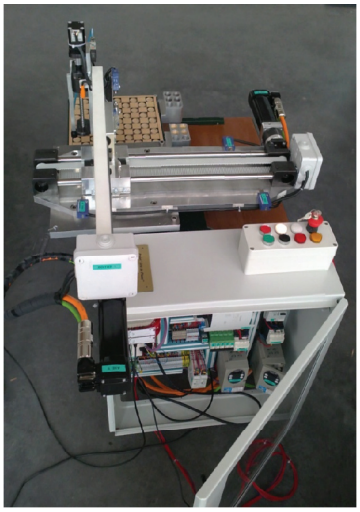

Photo 1 : Le système finalisé

Le serveur peut fonctionner même en cas de dysfonctionnement du système (partie opérative ou partie commande). Cette option très utile permettra de revoir l'historique des événements en cas de dysfonctionnements. L'interface superviseur (joueur) en JAVA est également opérationnelle.

[10] CIM : Computer Integrated Manufacturing

L'interface de l'observateur en AJAX n'est pas encore opérationnelle. Cela est dû à des choix sur l'organisation des données dans la base de données trop tardifs et une prise en main, tout aussi tardive, de cette technologie par les étudiants. Cette technologie n'est pas en soit très complexe, mais elle nécessite une organisation et une méthode de développement particulière.
De ce fait, le système est opérationnel en mode semi-automatique à l'heure de l'écriture de cet article. Le mode automatique est en cours de finalisation. Deux joueurs sont donc en mesure de jouer en demandant des déplacements sur le plateau de jeu.

5 OBJETS PEDAGOGIQUES

Bien sûr, la réalisation de ce système par des groupes d'étudiants de DUT GEII et de Licence Pro SAR était un objectif pédagogique en soi. La création d'un tel ensemble a également illustré les possibilités de superviser un processus via des interfaces web sans investissement logiciel. On peut également ajouter qu'un tel système, aura une place de choix dans la vitrine technologique du savoir faire de nos étudiants. Mais heureusement, cela ne s'arrête pas là. En effet, ce système a été conçu pour regrouper un maximum de technologies différentes comme nous avons pu le présenter. Ainsi, ce dispositif pourra servir de support pour des travaux pratiques dans les différentes disciplines de nos deux formations SAR et GEII. Le choix de la motorisation permet d'envisager des TP d'asservissement sans trop d'effort. Bien évidemment, la technologie de l'API permet d'imaginer des TP en automatismes. En résumé, un large spectre de sujets de formations dans différents domaines sera possible :

- automatismes : gestion de magasin ou de stock,
- réseaux : étude des protocoles, analyse de trames ou des temps de réponse des différentes structures,
- informatique : gestion de sockets en langage C et JAVA, gestion de BdD, gestion de serveur,
- Web : gestion d'IHM (Compréhension du DOM[11], écriture de script PHP, JS, HTML…), analyse du concept Web2 (AJAX), accès à une BdD,
- …

Pour revenir à la pédagogie mise en œuvre dans le cadre de ce projet, il est à souligner que l'implication d'équipes de niveaux de formations différents et de préoccupations différentes a initié explicitement une démarche différente. Ainsi les consignes ou les objectifs de certaines actions ou analyses, n'étaient pas en direction d'enseignant mais bien d'autres étudiants. Ainsi chacun d'eux était confronté au problème de la clarté du problème posé, de la pertinence de la réponse offerte (voire attendue) et du compte rendu du travail réalisé. En fait, ils étaient confrontés au problème de communication interne, problème récurrent en entreprise. Nous avons été satisfaits des réactions des étudiants, et nous pensons poursuivre cette démarche en y associant une démarche de conduite de projet plus stricte. Cette démarche n'est pas complexe en soi, mais ce n'est pas le sujet de cet article et surtout nous ne l'avons pas encore exploité au mieux.

[11] DOM : Domain Object Model

6 CONCLUSION ET PERSPECTIVES

Pendant deux années, nous avons donc pu guider les étudiants tout au long de la réalisation de ce système. Cet ensemble est bien à l'image des possibilités actuelles d'interfaçage possibles entre les univers de l'automatisme et d'Internet. Ainsi pour un budget de l'ordre de 15K€, nous avons une plate forme opérative « techniquement » très actuelle, attractive par son aspect ludique et riche pour de futurs développements pédagogiques.
De nombreux chantiers restent à ouvrir :

- démultiplier les types d'interfaces clients en utilisant de nouvelles technologies (HTML5 par exemple),
- éprouver la sécurité dans le transfert des données afin d'évaluer les limites de chacune de ces technologies,
- utiliser un serveur Linux,
- ouvrir la communication de l'API (interface TCP/IP directement sans le serveur de jeu avec accès MySQL par exemple).

Enfin, nous imaginons également pouvoir « cloner » ce retour sur expérience pour répondre à des besoins industriels ou de domotiques.

Bibliographie

[1] Blandin P, "Architecture logicielle type pour les applications de contrôle-commande", *Conservatoire National des Arts et Métiers, Paris, 2009.*

[2] Defrance J.M, "Premières applications Web avec Ajax, jQuery et PHP", *2ème édition, Eyrolles, 2009.*

[3] Dordoigne J and Atelin P, "TCP/IP et les protocoles Internet", *Eni Editions, 2008.*

[4] Garrett J.J, "Ajax: A New Approach to Web Applications", *Adaptative Path, 2005.*

[5] Gautreau S, "Solutions de communications et centralisation de données", *Centre de ressources et d'appui technologiques pour les entreprises régionales, Saintes, 2004.*

[6] http://jquery.com/.

[7] IEC 61131-3, "Programmable controllers – Part 3: Programming languages", INTERNATIONAL ELECTROTECHNICAL COMMISSION ISBN 2-8318-6653-7, 2003.

[8] phpsolutions, "AJAX et PHP", *PHP n°10/2010(46), 2010.*

[9] Pujolle G, "Les réseaux", *Editions Eyrolles, 2011.*

[10] Siemens, "Flexibilité dans toutes les applications IHM – du Micro Panel au PC", *Answers for industry, 2010.*