La mesure des écarts en Sciences de l'Ingénieur

Gil Sause, Dominique Laporte

La problématique

L'enseignement des sciences de l'ingénieur (SI) au lycée s'inscrit dans une continuité de la réforme de l'enseignement scientifique et technologique, depuis le collège jusqu'en classes préparatoires scientifiques puis en école d'ingénieurs. Sous l'appellation de sciences de l'ingénieur sont rassemblées des disciplines scientifiques en rapport avec le métier d'ingénieur, notamment dans les domaines du génie mécanique, de l'informatique, du génie civil, du génie électrique et de l'automatique.

Le métier de l'ingénieur consiste à poser et résoudre de manière performante et innovante des problèmes complexes, de création, de conception, de réalisation et de mise en oeuvre de produits, de systèmes ou de services.

Un ingénieur doit être capable de :

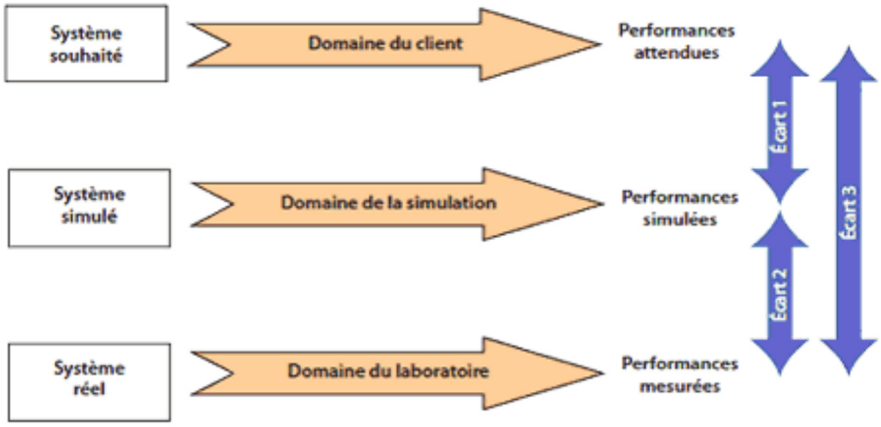
- vérifier les performances attendues d'un système complexe ;
- valider une modélisation à partir d'expérimentations ;
- prévoir les performances d'un système à partir d'une modélisation ;
- imaginer et concevoir un système complexe ;
- piloter des projets.

Les activités proposées visent à explorer comment :

- exploiter des modélisations et des simulations numériques pour prévoir les comportements d'un système pluri technologique ;
- concevoir ou optimiser une solution au regard d'un cahier des charges, dans le respect des contraintes de développement durable.

La spécificité des sciences de l'ingénieur vise donc à l'intégration des systèmes complexes, complexes dans le sens pluri disciplinaires et non dans le sens difficile.

Pour cela, l'enseignement de SI est basé sur la démarche de l'ingénieur en situation réelle que l'on peut représenter par le schéma suivant :



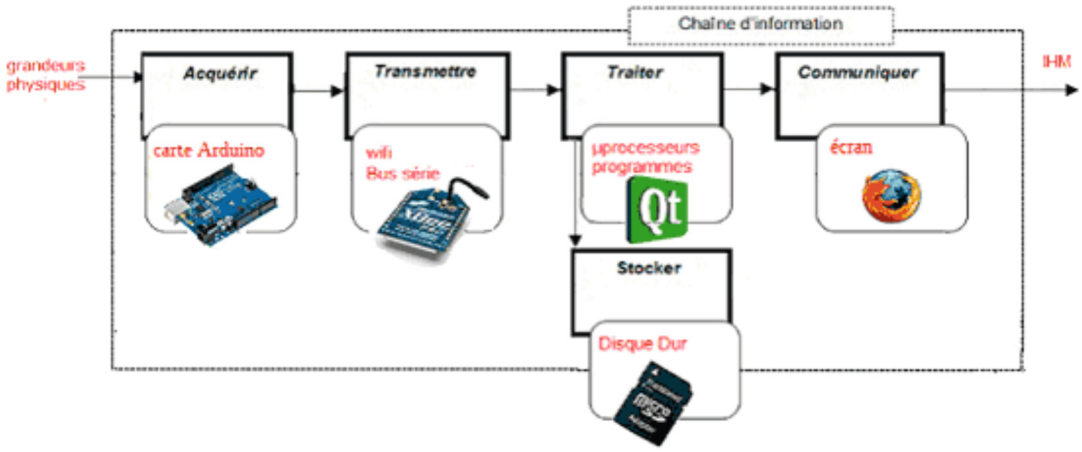
Deux étapes importantes consistent à mesurer les écarts entre le système simulé et le système réel (maquette) d'une part et le système réel et le système attendu (spécifié par le cahier des charges du client) d'autre part ; désignées respectivement par « étape 2 » et « étape 3 » sur le schéma.

L'importance de la mesure physique prend ici toute sa place avec une contrainte supplémentaire dans le cas du suivi de l'évolution des systèmes où la mesure doit être temps réel. Un appareillage spécifique est donc nécessaire qui doit à la fois acquérir une série de mesures, l'enregistrer et si possible l'analyser pour en faire émerger l'information utile.

Logiciel myDataLogger

Un enregistreur de données (ou data logger) est un dispositif qui enregistre des séries de mesures sur des périodes pouvant couvrir jusqu'à plusieurs heures ou plusieurs jours. Les grandeurs physiques sont mesurées automatiquement à intervalles de temps réguliers, numérisées, transmises et enregistrées sur un support en vue d'être traitées.

La figure ci-dessous représente la chaîne d'information du système.



Dans le cadre de l'enseignement en SI, nous avons retenu comme solution d'effectuer l'acquisition des mesures grâce à un nano automate Arduino puis de la transmettre par un module Xbee sur le port série RS232 [1] d'un nano ordinateur (Raspberry Py ou PCduino) ou d'un ordinateur de type PC. Les données sont ensuite lues et stockées sur disque à l'aide d'un logiciel graphique développé avec la librairie Qt. Ces dernières sont directement affichées par le datalogger ou peuvent être traitées ultérieurement à travers un client léger (navigateur internet).

Il existe sur le marché de nombreux datalogger qui vont du simple dispositif électronique d'acquisition programmable ou non, jusqu'au dispositif intégré avec traitement de données pour le monde professionnel ou éducatif.

Dans le milieu scolaire, les solutions proposées par les éditeurs ou les constructeurs sont souvent chères, mal adaptées et avec un format de fichier qui manque d'ouverture. Alors que le cahier des charges en SI, STI2D (sciences et technologies de l'industrie et du développement durable) ou BTS CRSA (Conception et réalisation de systèmes automatiques) est relativement simple : lire les données sur un port série RS232, d'afficher ces données sous forme de DEL [2], d'ACL [3] et/ou de graphes et de les enregistrer dans un format de fichier ouvert qui pourrait être lues directement par un tableur.

Pour remplir ce cahier des charges, nous avons décidé de développer un logiciel sous licence GNU/GPL [4] pour que nos travaux puissent être repris, adaptés et réutilisés selon les besoins de l'enseignant. Plusieurs choix s'offraient à nous, notamment le choix du langage et le choix de l'interface. Si le choix de l'interface en mode graphique s'est rapidement imposé pour une raison de simplicité et d'ergonomie, la question du langage de programmation a été plus complexe : fallait-il privilégier le critère économique, la mise en oeuvre, le « standard » de fait... ?

Après une courte étude de faisabilité, deux solutions possibles ont émergé : Qt ou Visual Studio.

- Qt est une bibliothèque multiplate-forme pour créer des GUI [5]. Elle est écrite en C++ mais il est possible de l'utiliser avec d'autres langages comme Java, Python, etc. La bibliothèque est tellement énorme et offre de telles possibilités que les développeurs la considèrent plus comme un cadriciel que comme une bibliothèque. Qt est développée actuellement par la société Digia et est distribuée sous licence propriétaire ou LGPL. On notera qu'elle est utilisée depuis 1991 par l'environnement de bureau KDE du système GNU/Linux.

- Visual Studio est développé par la société Microsoft et est également multiplate-forme grâce au langage C#, très proche de la syntaxe du langage Java, et à la plate-forme .NET. Des implémentations libres de ce langage et de sa plate-forme d'exécution ont été mises en oeuvre dans le projet Mono.

Les deux outils proposent un Environnement de développement intégré avec sans doute une meilleur ergonomie pour le produit de la société Microsoft.

La société Microsoft a signé un accord cadre avec l'Éducation Nationale qui autorise aux enseignants, comme aux élèves, de pouvoir utiliser sa plate-forme à un coût très raisonnable. Cependant, la politique commerciale de Microsoft nous a montré par le passé que rien est gravé dans le marbre. De plus, la communauté Qt, comme à l'image de la communauté « open source » est plus nombreuse et plus dynamique en terme de support.

Enfin, le langage C++ de Qt se rapproche énormément du langage utilisé par Arduino qui nous sert à programmer le nano automate.

Pour des raisons économique (gratuité de Qt), de maintenabilité (communauté), pédagogique (unicité du langage de programmation) et éthique (modèle du logiciel libre),

nous nous sommes orientés sur Qt.

La durée du développement du logiciel myDataLogger a totalisé une quarantaine d'heures échelonnées de janvier à février, jusqu'au lancement du site officiel vers la mi-mars (http://projet.eu.org/). Il est actuellement exploité en SI pour l'optimisation et la validation du modèle d'une mini voiture électrique à énergie solaire, en STI2D pour le pilotage d'une serre autonome et en BTS CRSA pour le suivi et la modélisation d'un chauffe eau.

La mini voiture solaire

Le projet de mini voiture solaire s'inscrit dans le « défis solaires » qui s'adresse aux jeunes de 10 à 25 ans, désireux de mieux comprendre leur environnement et de s'intégrer dans une démarche de développement durable. La construction de ce type de véhicule permet de mieux appréhender les problèmes d'énergies, de mécanique, d'aérodynamisme et de sensibiliser les élèves de SI et STI2D aux solutions énergétiques non polluantes, en montrant les avantages des énergies renouvelables.

C'est l'objectif que s'est fixée une équipe de professeurs de Sciences de l'Ingénieur du lycée en décidant de participer pour la cinquième année consécutive au défi solaire Méditerranée, organisé par l'association Planète Sciences (http://www.planete-sciences.org/), où participent plusieurs établissements de la région PACA.

Dans le cadre d'une démarche d'ingénierie, les élèves utilisent un modèle multi-physique à l'aide des logiciels Matlab [6] ou Scilab [7] afin de simuler le comportement du système dans des conditions extrêmes.

L'instrumentation du véhicule solaire est la seconde étape de la démarche qui permet d'effectuer des mesures physiques en vue de paramétrer correctement le modèle multi-physique.

L'objectif final étant de minimiser au maximum les écarts entre le simulé (le modèle numérique), le réel (la mesure physique) et l'attendu (l'optimisation par rapport au cahier des charges afin de gagner la compétition).

Le choix de la mise en oeuvre de l'instrumentation s'est porté sur le nano automate Arduino. Arduino est un circuit imprimé en matériel libre (même si certains composants comme le microcontrôleur par exemple, ne sont pas en licence libre) qui peut être programmé en langage objet C++ à travers une liaison série RS232. Ce type d'appareil est largement utilisé en domotique dans des projets en code ouvert (open source) ou en logiciel libre.

Le microcontrôleur de la carte utilise des entrées-sorties pour l'interfaçage avec des capteurs qui permettent, dans notre cas, de mesurer :

- le courant et la tension délivrés par le panneau photovoltaïque qui alimente la voiture ;
- la luminosité reçue par les cellules solaires ;
- l'accélération et la vitesse du véhicule.

Nous avons fait le choix d'équiper la carte d'un module de communication Xbee [8] afin de faire l'acquisition de mesures pour un suivi en temps réel du système ; la carte Arduino centralise les données des capteurs et les transmet à un PC distant où elles sont traitées par le logiciel myDataLogger.

La difficulté majeure dans le fonctionnement des mini-voitures est le démarrage. Nous disposons d'une source d'énergie électrique limitée (panneau photovoltaïque). Nous avons une consommation importante d'énergie que nous pourrons mesurer avec myDataLogger. Il nous faudra trouver les solutions techniques qui nous permettrons d'utiliser cette source d'énergie à bon escient (démarrer le véhicule et aller le plus vite possible).

Les élèves interviennent directement sur la configuration logicielle du programme à implémenter dans le microcontrôleur. Ils doivent programmer les entrées-sorties des capteurs (choix des broches, type des signaux, grandeurs et unités physiques), paramétrer les ports de communication, compiler le programme et le téléverser sur la carte.

En conclusion

L'intégration de systèmes complexes, dans un monde où les objets sont de plus en plus communicant, doit prendre en compte une chaîne d'information où l'informatique prend une place prédominante. Les choix qui nous ont amené à nous orienter vers une solution à base de matériel et de logiciel libres n'ont pas été dictés simplement par une raison économique, ou même technique, mais également par un modèle éthique qui se concrétise par l'ouverture, la diffusion et le partage du savoir.

Les élèves ont ainsi accès directement au code source qu'ils peuvent étudier et modifier pour l'adapter aux besoins propres à l'étude d'un système. Il en va de même pour les enseignants qui souhaiteraient faire évoluer le datalogger en fonction de leur enseignement et de leurs classes.

Manosque, avril 2015.

Gil Sause
professeur de Sciences de l'Ingénieur

Dominique Laporte
professeur de Sciences de l'Ingénieur

NOTES

[1] Norme de bus de communication de type série.

[2] Diode électroluminescente.

[3] Affichage à cristaux liquides.

[4] http://www.gnu.org/licenses/licenses.fr.html

[5] Graphic User Interface.

[6] http://fr.mathworks.com/

[7] http://www.scilab.org/

[8] http://www.digi.com/lp/xbee/

Association EPI
Mai 2015

Articles