

L'Instrumentation Virtuelle comme outil pédagogique

Philippe BAUCOUR¹

¹National Instruments France, Centre d'Affaires Paris -Nord, BP 217, 93153 Le Blanc-Mesnil, philippe.baucour@ni.com

RÉSUMÉ L'Instrumentation Virtuelle est un concept selon lequel l'utilisateur associe des outils informatiques et des moyens de mesures afin de concevoir l'instrument qui correspond à ses besoins. Qu'il s'agisse d'automatiser un banc de test électronique, de réaliser un système de supervision temps réel ou de déployer un algorithme d'asservissement au cœur d'un composant FPGA, il est possible dorénavant de concevoir des Instruments Virtuels aux formes multiples. Les avantages offerts par l'Instrumentation Virtuelle sont essentiels pour l'industrie : diminution des temps de développement, optimisation des moyens et réduction des coûts ne sont que quelques exemples. Dans le cadre spécifique de l'enseignement des sciences de l'ingénieur, l'Instrumentation Virtuelle permet de faire évoluer la démarche pédagogique avec des outils et des méthodes largement diffusés dans l'industrie. Enfin, pour l'étudiant, c'est l'assurance de posséder des connaissances et un savoir-faire utilisables dans une très large gamme de domaines d'application.

Mots-clés : Instrumentation Virtuelle, LabVIEW, mesure, contrôle, commande, conception, programmation, analyse, temps réel, embarqué.

1 INTRODUCTION

Généralement, le terme d'Instrumentation Virtuelle est mal compris. Un rapide sondage révèle que ce terme est très souvent associé à la notion d'instruments simulés, ou de réalité virtuelle et qu'à de très rares exceptions près, l'ingénieur ou le scientifique est capable d'en donner une définition exacte. Nous commencerons donc par définir clairement ce qu'est l'Instrumentation Virtuelle.

Par la suite, un résumé de l'historique sera l'occasion de démontrer que l'Instrumentation Virtuelle n'est pas "un phénomène de mode" mais bien une tendance de fond qui tend à révolutionner la façon dont les systèmes informatiques industriels sont conçus, réalisés, déployés et maintenus.

Ensuite, nous nous attarderons sur les aspects pédagogiques en faisant l'inventaire de quelques contraintes puis en montrant à l'aide d'exemples très précis les avantages qu'offre l'Instrumentation Virtuelle dans le cadre de l'enseignement.

Enfin, avant de conclure, les tendances et les perspectives de l'Instrumentation Virtuelle seront examinées.

À la fin de cet article, le lecteur enseignant aura donc une idée claire de ce qu'est l'Instrumentation Virtuelle, des possibilités qu'offre cette dernière du point de vue de l'enseignement et des évolutions à venir.

2 DÉFINITION

2.1 Définition

L'Instrumentation Virtuelle, c'est l'association d'outils informatiques (ordinateurs, logiciels, réseaux...) et de

moyens d'entrée/sortie (interface Série, GPIB, USB, modules d'acquisition, instruments de mesure...) afin de réaliser des systèmes définis par l'utilisateur.

2.2 Ce que dit le dictionnaire

Avant d'aller plus loin, il est sans doute utile de revenir sur la définition des termes utilisés. Ainsi, pour le Larousse, un instrument "*est un objet fabriqué servant à un travail, à une opération ou considéré par rapport à sa fonction, son usage (instrument oratoire, instrument de mesure)*". En ce qui concerne l'adjectif "virtuel", le dictionnaire nous apprend qu'il "*vient du mot latin virtus (force), utilisé pour décrire quelque chose qui n'est qu'en puissance ; potentiel, possible (les débouchés virtuels d'un nouveau produit)*".

Il faut donc se pénétrer de l'idée qu'un Instrument Virtuel ne peut et ne doit pas être limité à une application de mesure sur PC. En effet, la définition du dictionnaire laisse entendre qu'en tant qu'instrument, l'association matériel-logiciel peut être utilisée pour faire, certes, de la mesure mais aussi de la supervision, du contrôle, de l'analyse, etc. Le champ des applications est donc beaucoup plus vaste que la simple automatisation de prise de mesures sur un PC.

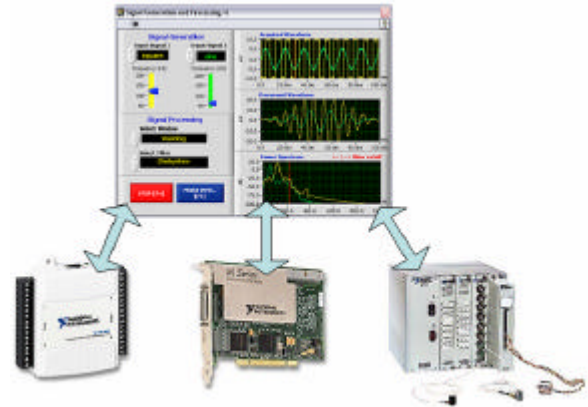
Ainsi, le scientifique qui conçoit un filtre numérique sur PC et qui le déploie dans un composant FPGA fait de l'Instrumentation Virtuelle. Il en va de même pour l'ingénieur qui acquiert des mesures, les analyse et les présente à l'écran. Enfin, l'électronicien qui compare des données issues d'un simulateur SPICE avec les mesures prises sur un prototype fait, lui aussi, de l'Instrumentation Virtuelle.

2.3 Systèmes définis par l'utilisateur

La définition que nous avons donnée plus haut est très générale. Ceci dit, certains termes sont importants. Par exemple, le fait que le système soit "*défini par*

l'utilisateur" est tout à fait essentiel puisque cela place l'Homme au cœur des préoccupations. Il n'est pas question de se lancer dans un débat philosophique mais il faut quand même noter que par rapport à des systèmes fermés, prêts-à-l'emploi, propriétaires, l'Instrumentation Virtuelle propose une alternative dans laquelle le concepteur peut utiliser son libre arbitre. Certes, cela impose de faire des choix et des responsabilités mais en contre partie, l'Instrumentation Virtuelle est la seule solution qui permette de concevoir des systèmes dont les coûts sont optimaux (aucune fonction en trop) et qu'il est possible de faire évoluer.

Par exemple, avec un même matériel d'E/S, il est possible de concevoir différents Instruments Virtuels. Outre les économies qui en découlent, c'est surtout un formidable exemple de ce que signifie "*systèmes définis par l'utilisateur*". Un jour, l'ensemble matériel-logiciel peut être utilisé comme un analyseur de spectre alors que le lendemain, il se comportera comme un enregistreur de données.

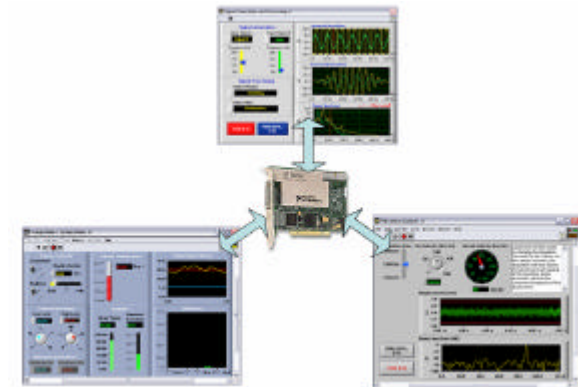


Le même Instrument Virtuel peut être utilisé avec différents matériels d'E/S.

2.4 Le tout vaut plus que la somme des parties

La deuxième notion importante qui est présente dans la définition est le fait que l'Instrumentation Virtuelle résulte d'une "*association*" entre le logiciel et le matériel. Fondamentalement, cela signifie que dès leur conception, les logiciels et les matériels utilisés dans un Instrument Virtuels doivent avoir été pensés pour travailler ensemble.

Cela ne peut pas être le cas d'un fournisseur qui limiterait son offre à la fourniture de matériels et de quelques bibliothèques afin d'être utilisés avec des environnements de développement généralistes. En effet, dans ce cas, qu'observe-t-on ? En pratique, l'utilisateur parvient généralement à compiler et à mettre en œuvre les quelques exemples qui accompagnent le matériel. Ceci dit, par la suite, le développeur se retrouve confronté à des problèmes de conception d'applications, de manque de fonctions de traitement, d'analyse, etc. Typiquement, les environnements généralistes ne possèdent pas (et c'est normal) les objets graphiques ni les fonctions d'analyse nécessaires aux applications d'Instrumentation Virtuelle. L'utilisateur doit donc chercher, trouver et intégrer ses dernières via des bibliothèques dont la maintenance n'est pas toujours aussi bien assurée que celle de leur environnement de développement.



Trois Instruments Virtuels différents utilisent le même matériel d'E/S.

La réciproque est vraie aussi. En effet, une application peut et doit être utilisée avec différents types de matériel. Par exemple, une première version d'un Instrument Virtuel met en œuvre un boîtier externe USB d'acquisition de données alors que la seconde utilise un module d'acquisition PCI ou tout un système de conditionnement du signal.

La réciproque, même si à notre connaissance elle n'existe pas, poserait aussi des problèmes. Imaginons qu'un fournisseur limite son offre à un environnement de développement spécialisé dans la création d'applications de test et mesure. Les problèmes proviendraient alors des difficultés à faire fonctionner ensemble différents modules d'E/S provenant de différents constructeurs. Pensez par exemple aux problèmes de synchronisation entre modules ou tout simplement aux difficultés pour l'utilisateur d'employer des API de programmation spécifiques et non

homogènes lorsqu'il utilise le module d'un fournisseur puis le module d'un autre.

Une troisième alternative, qui existe et qui n'est pas entièrement satisfaisante non plus, voit le jour lorsqu'un fournisseur décide de rendre ses matériels d'E/S utilisables dans des environnements de développement spécialisés dans la création d'Instruments Virtuels. Dans ce cas, si le module est très spécialisé, la situation tourne généralement à l'avantage de l'utilisateur. En effet, le périphérique est spécifique, son utilisation n'est pas tout à fait homogène avec celle des autres mais l'utilisateur s'en accommode car le type de mesures qu'il est possible de réaliser avec ce matériel le rend indispensable. Par contre, dans un marché très concurrentiel comme celui des cartes d'acquisition par exemple, utiliser des modules qui n'ont pas été conçus dès le départ pour fonctionner avec l'environnement de développement, pose de réels problèmes. Ceci est surtout vrai au niveau de l'API (Application Programming Interface) et des outils connexes tels que configurateurs, assistants, générateurs de code, etc. qui doivent être parfaitement intégrés et homogènes avec l'environnement dans lequel ils sont utilisés. En effet, si ce n'est pas le cas, l'utilisateur se trouve alors avec d'un côté un très bon environnement, de l'autre une bonne carte mais aucun moyen de faire diminuer le temps qu'il faut pour acquérir les premiers points de mesure. Nous l'avons dit, le marché des cartes d'acquisition est très concurrentiel. Cela signifie que ce type de module est devenu une vraie commodité et que l'utilisateur n'accepte plus de "perdre son temps". Il veut, et il a raison, obtenir très rapidement ses mesures afin de pouvoir les analyser et les présenter.

Un quatrième cas existe et a été résolu avec succès au bénéfice des utilisateurs. Il concerne les instruments de mesure classiques qui sont pilotés par des interfaces Série, GPIB, USB, 1394 ou TCP/IP. Ici, les différents fournisseurs se sont entendus pour définir ce que l'on a appelé un "driver d'instrument". Typiquement, il s'agit d'une bibliothèque de fonctions que les différents constructeurs écrivent pour leurs instruments respectifs mais qui sont reconnues par les environnements de développement. L'utilisateur peut donc en toute sérénité concevoir ses Instruments Virtuels même si ces derniers mettent en œuvre des instruments classiques très divers.

Autrement dit, le fait que dans un Instrument Virtuel les matériels et le logiciel doivent par définition avoir été conçus pour travailler ensemble est un atout pour l'utilisateur. C'est la prise en compte d'un phénomène naturel qui veut que le tout vaut toujours plus que la somme des parties.

Dans le monde de l'Instrumentation Virtuelle, cela se traduit par la réduction des temps de développement ou de mise en œuvre, autrement dit par une baisse des coûts. Attention, cette dernière est à différencier de la baisse des prix. Pour l'industrie, il suffit d'imaginer le surcoût associé à un projet si l'ingénieur met plus d'une journée à mettre en œuvre un module d'E/S dont le prix est bas. Quel est l'intérêt d'investir dans un module à 100 € si il faut dépenser une journée ingénieur (1000 €) avant de faire la moindre mesure? Dans le cadre de l'enseignement, le surcoût sera synonyme d'étudiants qui n'arrivent pas à terminer leur TP à temps ou d'heures de préparation supplémentaires pour l'enseignant afin que, justement, les élèves ne rencontrent pas trop de difficultés lors du TP.

2.5 Comparaison avec les instruments classiques

Si l'on observe les éléments constitutifs d'un instrument de mesure moderne, un oscilloscope par exemple, on remarque que ce dernier comprend une carte mère de PC (avec une ROM, de la RAM, un processeur, un OS, des applications, etc.). De plus, il comprend un frontal d'acquisition oscilloscopique, un écran, une alimentation et un ensemble de boutons qui constituent l'interface que met le constructeur de l'instrument à la disposition de l'utilisateur. Il est intéressant de noter que les options sont dorénavant si nombreuses que la plupart des instruments évolués n'ont pas d'autre choix que d'utiliser un logiciel propriétaire, embarqué dans l'appareil pour afficher les différents menus et l'ensemble des options disponibles.

Ceci dit, plusieurs remarques peuvent être faites. Par exemple, dans le cas de l'utilisation de l'oscilloscope dans un banc de mesure automatisé, quel est l'avantage de cet ensemble de boutons (ou de l'écran) si personne ne s'en sert ?

Par la suite, dans le cas où plusieurs instruments sont utilisés simultanément, quel intérêt y a-t-il à "payer" pour plusieurs alimentations, plusieurs écrans... alors qu'en fait, tout est piloté par un ordinateur ?

Par ailleurs, sachant que de plus en plus d'instruments sont architecturés autour d'un PC (avec système d'exploitation, interface réseau, etc.) il est naturel de se demander si l'ingénieur qui va en utiliser plusieurs ne va pas finir par "perdre son âme" de spécialiste de la mesure tout simplement parce qu'il va devoir passer plus de temps à gérer le parc informatique des PC embarqués plutôt qu'à faire de la mesure.

Enfin, le fait d'embarquer un PC dans l'instrument pose la question de la gestion de l'obsolescence. En effet, dans trois ans, dans cinq ans, même si les performances de mesures sont toujours suffisantes, il y a fort à parier que les besoins en traitement ou les moyens de

communication auront largement évolué. La puissance du PC embarqué sera-t-elle toujours suffisante ? Est-ce que les interfaces de communication seront toujours compatibles et interopérables (IEEE 1394, USB, Interface Ethernet Gigabit...). Si ce n'est pas le cas, quelles sont les alternatives offertes mise à part le remplacement pur et simple de l'instrument ? Ne serait-il pas plus efficace, dès le départ, de désolidariser la fonction de mesure de l'instrument du reste ou bien carrément d'insérer la fonction de mesure dans un ordinateur que l'on pourra facilement faire évoluer à moindre coût. Notons que cette réflexion n'est pas propre à l'instrumentation puisque les nouvelles possibilités offertes par le bus PCI Express permettent aux concepteurs de PC d'envisager la mise sur le marché de stations "éclatées" dont les éléments constitutifs seraient reliés par un bus PCI Express en version câblée (mémoire, stockage, unité centrale...).

2.6 Les contraintes qui découlent de la définition

Sans présager du reste de l'article, on peut donc déjà imaginer certaines des contraintes que doivent s'imposer les fournisseurs d'outils d'Instrumentation Virtuelle. Elles sont au nombre de trois.

2.6.1 Une offre d'E/S modulaire

Primo, en terme de moyens d'entrées/sorties (au sens large), ils doivent disposer d'une offre modulaire. Cette dernière est essentielle puisque c'est elle qui permet d'éviter les redondances (écrans, alimentations...). De même, ces fournisseurs doivent disposer d'une offre très large qui permet à l'utilisateur d'intégrer dans son système les modules dont il a besoin (vision, commande d'axes, acquisition de données, oscilloscope, multimètre...).

2.6.2 Des logiciels productifs

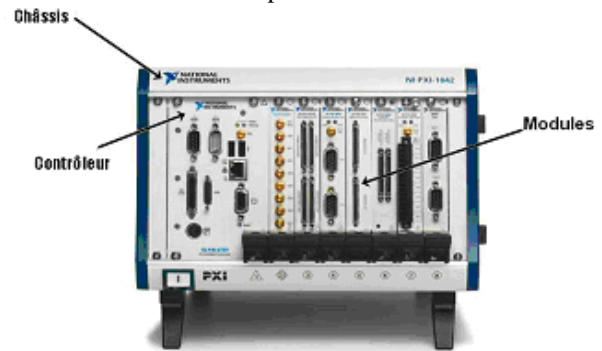
Secundo, les fournisseurs doivent disposer d'outils logiciels qui évitent à l'utilisateur de "perdre son âme" de spécialiste de la mesure en le forçant à devenir programmeur. Cet aspect sera développé ultérieurement en détail ce qui permettra véritablement de faire ressortir les avantages qu'offre la démarche d'Instrumentation Virtuelle du point de vue pédagogique.

2.6.3 Des plates-formes pour la synchronisation et le déploiement

Tertio, les fournisseurs d'Instrumentation Virtuelle doivent développer leur offre autour de "plates-formes". Ces dernières doivent permettre la synchronisation des moyens d'E/S et le déploiement des Instruments Virtuels dans des facteurs de formes correspondant aux contraintes spécifiques à tel ou tel domaine. Le PC représente une telle plate-forme mais ce n'est pas la seule.

En effet, dans le cadre d'applications embarquées ou utilisées dans des environnements durs (applications

marines ou spatiales, par exemple), le PC n'est pas la meilleure des plates-formes. Par exemple, le fait de devoir utiliser de nombreux modules d'E/S peut inciter l'utilisateur à choisir une plate-forme PXI.



PXI, un PC industriel sur bus PCI et sous forme de châssis industriel offrant de 4 à 18 emplacements

De même si l'Instrument Virtuel doit posséder un comportement déterministe ou supporter des contraintes environnementales très dures, c'est peut-être sur une plate-forme CompactRIO qu'il faudra le déployer.



La plate-forme CompactRIO associe un cœur FPGA, un contrôleur temps réel et des modules d'E/S.

À propos de la notion de plate-forme, il y a encore deux points à retenir. La capacité à tirer parti des plates-formes existantes ou à en créer est une caractéristique qui fait que tel ou tel fournisseur devient rapidement un standard industriel. Ensuite, dans un monde en constante évolution, le concepteur d'Instruments Virtuels ne peut plus se permettre le luxe d'être prisonnier de telle ou telle plate-forme. Il faut que son fournisseur lui offre les moyens de déployer ses outils d'une plate-forme à une autre : aujourd'hui, sur PC, demain sur PDA, puis sur DSP ou FPGA.

On le voit, l'étude de la définition de l'Instrumentation Virtuelle implique et sous-entend beaucoup de choses. Ceci étant, en simplifiant à l'extrême, on peut dire que l'Instrumentation Virtuelle est aux entrées/sorties et à la mesure et ce que fut le traitement de texte pour les machines à écrire.

Aujourd'hui, plus personne n'utilise de machine à écrire. La flexibilité qu'offre l'association du traitement de texte

et de l'ordinateur est trop importante pour que l'on revienne en arrière. Le même phénomène s'opère dans le monde des entrées/sorties (E/S). Les quantités d'informations sont trop grandes, les temps de traitement sont à la baisse, les avantages qu'offrent les réseaux et Internet sont trop importants pour que les mesures continuent à être prises à la main. Les moyens informatiques sont donc dorénavant systématiquement associés aux E/S.

3 HISTORIQUE

Afin de bien comprendre ce qu'il est possible de faire aujourd'hui et d'envisager les possibilités de demain, il est toujours utile d'étudier l'Histoire et c'est ce que nous allons faire ici. L'histoire de l'Instrumentation virtuelle débute dans les années 70 et peut être décomposée en quatre grandes phases.

3.1 Les années 70

Il s'agit essentiellement de contrôler des instruments de mesure à partir de micro-ordinateurs via une liaison GPIB. Le PC n'existe pas et la notion d'Instrumentation Virtuelle n'est pas encore complètement définie. Ceci dit, les problèmes liés au contrôle des instruments ne cessent d'inciter les ingénieurs à trouver "une meilleure façon" de travailler.

3.2 Les années 80

Le contrôle des instruments se fait dorénavant via le PC qui se démocratise. Les logiciels d'Instrumentation Virtuelle tels que LabVIEW et LabWindows apparaissent respectivement en 86 et 87. La première interface pour les instruments modulaires VXI est mise sur le marché en 88 en même temps que les premières cartes d'acquisition de données pour PC. À cette période, l'Instrumentation Virtuelle est clairement définie même si les contraintes liées aux performances des PC et aux possibilités de l'électronique de l'époque ne permettent pas la mise sur le marché de tous les produits souhaités.



Exemple de châssis et de modules VXI

Par exemple, dans les années 80, l'instrumentation modulaire de type VXI utilise des modules de grande taille (6U) parce qu'il faut beaucoup de place (de surface) pour pouvoir intégrer l'électronique d'un instrument. Les prix des modules sont très élevés et seules quelques applications militaires ou de traitement de très grande quantité de produits peuvent justifier de tels investissements. De même, les processeurs ne sont pas à la hauteur ce qui limite les possibilités d'analyse ou de traitement et à la volée (FFT ou filtrage numérique par exemple). Sur PC, à cette époque, le bus ISA a une bande passante faible (16 Mo/s). Cette dernière limite la quantité d'informations que l'on peut y faire circuler et le nombre de modules que l'on peut y connecter. Toujours à la même époque, les convertisseurs analogique/numérique proposent des fréquences maximales et des résolutions relativement faibles (100 Kéch./s, 12 bits typiquement).

3.3 Les années 90

On retient surtout l'apparition de la plate-forme PXI qui n'aurait pu exister sans la notion d'Instrumentation Virtuelle. Le format PXI est beaucoup plus compact que le format VXI (3U au lieu de 6U). C'est une conséquence directe des progrès de l'électronique qui, à surface égale, permet d'intégrer dorénavant beaucoup plus de fonctionnalités. PXI est aussi un standard qui s'appuie sur un bus PC de type PCI. Ce dernier offre une bande passante 10 fois supérieure à celle du bus ISA (132 Mo/s contre 16 Mo/s). C'est aussi à cette époque que les outils logiciels pour le déploiement d'Instruments Virtuels sur cible temps réel font leur apparition.

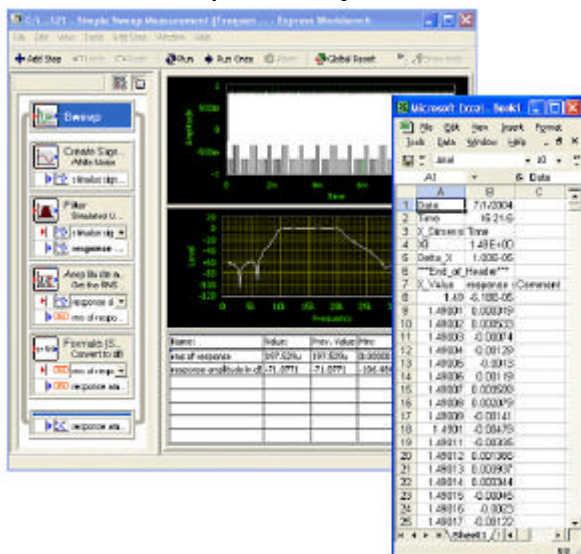
3.4 Les années 2000

Des premières années 2000 on retient l'extension de la gamme des instruments modulaires (aux formats PCI ou PXI) et l'explosion des options pour l'usage de l'Instrumentation Virtuelle dans le domaine du

contrôle/commande (temps réel). Du point de vue des grandes orientations logicielles, on note l'apparition d'options pour les domaines de la simulation, de la conception et de l'embarqué ainsi que la mise sur le marché de logiciels tels que SignalExpress.

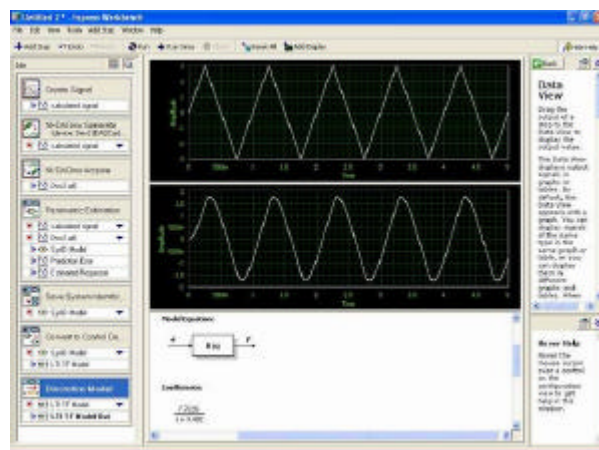
Ce dernier est sans doute le premier représentant de ce que l'on peut appeler les outils d'Instrumentation Virtuelle de seconde génération. SignalExpress est une application prête-à-l'emploi qui permet à l'utilisateur de "faire l'expérience" de l'Instrumentation Virtuelle sans avoir à programmer. Au choix, une application comme SignalExpress peut être vue comme :

- ? un outil complémentaire permettant à un programmeur de rapidement prototyper une application. A cette fin, SignalExpress offre la possibilité d'exporter ses scripts sous forme de code LabVIEW qui peuvent être alors complétés par le développeur.
- ? un outil à part entière qui permet au non-programmeur de tirer parti des ressources d'E/S qu'il a à sa disposition. De manière interactive, il va créer un script et automatiser ses prises de mesures ainsi que leur analyse.



SignalExpress est la troisième génération d'outils logiciels d'Instrumentation Virtuelle.

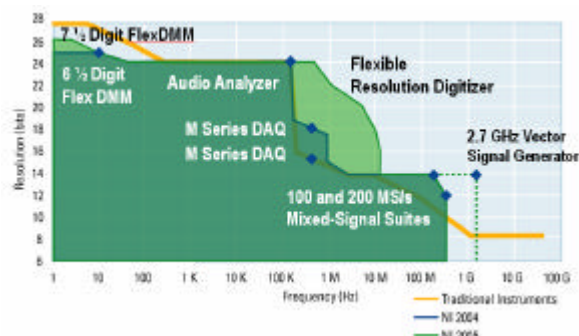
Ceci dit, SignalExpress va plus loin. En effet, cette application est en fait une application hôte dans laquelle des "plug-in" peuvent venir s'insérer afin de spécialiser son usage. Par exemple, l'illustration suivante montre comment le "plug-in" d'identification de système peut être utilisé dans SignalExpress pour étudier le comportement d'un système en boucle ouverte.



SignalExpress peut aussi être utilisé pour l'identification de système et la conception de systèmes de contrôle.

3.5 Quels enseignements en tirer ?

Que retenir de cette "très brève histoire du temps"? Cela fait maintenant plus de vingt ans que l'Instrumentation Virtuelle existe. Sur la même période, en MIPS et à prix constant, la puissance des PC a été multipliée par 10 000 (80 db). De même, on est passé du modem à la liaison Gigabit et dorénavant les réseaux ou Internet décuplent les possibilités de contrôle à distance ou de distribution des systèmes informatiques. De plus, comme il n'y a aucune raison pour que les performances de l'électronique ralentissent, il est certain que les performances des modules d'E/S vont continuer à progresser. Typiquement, en 2005 sur plate-forme PC, il est possible d'acquérir des données jusqu'à 3 GHz, d'échanger 700 Mo/s, d'avoir des résolutions supérieures à 26 bits en basse fréquence et de 16 bits à 15 MHz.



L'enveloppe résolution-fréquence des Instruments Virtuels comparée à celle des instruments classiques

3.5.1 Une nouvelle génération d'instruments

Compte tenu de l'explosion des performances de l'informatique, de l'électronique et de ce qui a déjà été fait avec l'Instrumentation Virtuelle ces vingt dernières années, on peut tirer une première conclusion : on ne concevra plus au troisième millénaire les systèmes d'E/S comme on le faisait précédemment.

3.5.2 Des instruments plus évolutifs

Pour le reste, et comme le laisse entendre la définition, on remarque que l'évolution de l'Instrumentation Virtuelle est liée à l'évolution des technologies PC, des logiciels et des possibilités de l'électronique. Ceci amène une remarque importante : le fait que l'Instrumentation Virtuelle s'appuie sur les moyens informatiques (PC, OS, logiciels...) et des moyens d'acquisition (convertisseurs, bus...) issus de l'électronique grand public implique que son évolution sera toujours plus rapide que celle des instruments classiques. En effet, ces derniers sont élaborés autour de moyens d'acquisition et d'architectures complètement propriétaires que les sociétés doivent développer seules, sur fonds propres tout en sachant que les volumes produits seront sans commune mesure avec ceux de l'industrie du PC. Remarquons enfin que de toute façon, il n'y a pas d'autre choix puisque ces "innovations" sont le seul moyen à la disposition des différents constructeurs d'instruments pour se différencier les uns des autres vis-à-vis de leur clientèle.

3.5.3 Le logiciel, composant essentiel

Finalement, de cet historique, il faut aussi retenir que de l'association logiciel-matériel c'est le logiciel qui devient l'élément prépondérant. C'est lui en effet qui permet à l'ingénieur de se concentrer sur l'analyse alors que la puissance du PC et les capacités de l'électronique des moyens d'acquisition deviennent des outils, des moyens. C'est la puissance du logiciel qui permet à l'utilisateur de "garder la tête hors de l'eau" malgré la croissance exponentielle des quantités de données à traiter (via des traitements 64 bits par exemple). De même, c'est la flexibilité du logiciel qui permet d'innover, d'adapter et d'optimiser tel ou tel traitement. C'est enfin, toujours et encore, le logiciel qui permet le déploiement sur différentes plates-formes (FPGA, temps réel, PDA, processeur embarqué, DSP...) d'un code conçu, testé et validé sur une cible PC.

4 LA DÉMARCHE PÉDAGOGIQUE

4.1 Quelques points à prendre en compte

4.1.1 La baisse des effectifs

Entre 1995 et 2000, les effectifs de l'enseignement supérieur scientifique ont diminué de plus de 6%. Cette baisse s'élevait à 12% dans les premiers cycles et les derniers indices laissent à penser que depuis, la tendance est confirmée. Par exemple, on observe une baisse de -40% en physique, de -24% en mathématique entre 96 et 2002.

Les raisons de ces désaffections ont été analysées par de nombreux experts. Sans vouloir prétendre posséder la solution, il est peut-être un aspect qui a été sous-estimé. National Instruments, société qui dispose de

filiales dans plus de 40 pays et travaille avec de nombreuses universités, observe que globalement, quel que soit le pays ou la matière, "l'enseignement des sciences n'est pas fun". Un brin provocatrice, cette remarque n'a rien à voir avec une remise en cause de l'étude de la théorie qui est obligatoire. Ceci dit, elle suppose qu'au moment où enseignants et étudiants vont passer aux séances de travaux pratiques, tout soit mis en œuvre pour que les élèves aient entre les mains des moyens qui leur permettent véritablement de prendre du plaisir. C'est sans doute un point important à prendre en compte dans le choix des outils utilisés dans la démarche pédagogique.

4.1.2 "Ce que j'entends je l'oublie, ce que je vois je le comprends, ce que je fais je le retiens"

C'est cette remarque qui fait le succès des séances de travaux pratiques. Ceci dit, il faut aussi que les outils dont disposent les élèves et les professeurs permettent de mettre sur pied de réelles investigations, de réelles recherches. D'un point de vue pratique, il faut que le temps de prise en main soit très court, permette de focaliser sur l'essentiel et soit facilement adaptable afin que l'utilisateur passe plus de temps à tester telle ou telle idée plutôt qu'à se battre avec des contingences matérielles (problèmes de programmation, difficultés à configurer tels ou tels outils de mesure...). On remarquera que cette "contrainte" est très proche de la demande des ingénieurs de recherche et développement qui doivent concevoir de plus en plus vite de nouveaux produits. La nécessité de mettre entre les mains des élèves des moyens modernes, efficaces et suffisamment souples pour leur permettre de "faire" des choses est le deuxième point qu'il faudra donc garder en tête lors du choix des moyens pédagogiques.

4.1.3 S'appuyer sur les outils informatiques

Le troisième point à retenir est sans doute la nécessité de choisir des outils qui s'appuient très largement sur les moyens informatiques. Nous l'avons dit, l'industrie, pour des raisons de rentabilité a déjà fait ce choix. Il est donc essentiel que l'enseignement prenne en compte ce fait et forme les futurs ingénieurs avec des méthodes qu'ils retrouveront une fois sur le marché du travail. Cela a déjà été fait en grande partie. Ceci dit, si on pense aux "manipes" de TP qui ne sont toujours pas informatisées, on réalise qu'il y a là un formidable potentiel en termes de projets susceptibles d'être réalisés par les étudiants eux-mêmes avec trois objectifs :

- ? comprendre la "manipe" elle-même
- ? trouver et mettre en place les solutions informatiques permettant son automatisation
- ? réaliser que la prise en compte de l'informatisation des moyens de production ou de test doit se faire très en amont.

4.1.4 Des outils à usages multiples

Le quatrième point concerne essentiellement les enseignants. En effet, ces derniers doivent non seulement faire le choix de solutions "up to date" mais aussi, susceptibles d'évoluer au même rythme que celles utilisées dans l'industrie. De plus, il faut s'assurer que les choix technologiques sur lesquels une partie de la pédagogie reposera permettent à l'élève de capitaliser des connaissances et un savoir-faire utilisables dans un vaste champ d'applications : test, contrôle/commande, conception, supervision, informatique distribuée, etc.

Les outils et méthodes sélectionnés doivent aussi permettre à l'équipe pédagogique d'établir des programmes sur deux ou cinq ans dans lesquels, d'une année sur l'autre, l'élève, sans changer fondamentalement d'outils, passe d'un domaine à l'autre (algorithmique, E/S, contrôle d'instruments, asservissements, automatiques, systèmes embarqués...). À ce titre, il est intéressant de noter que l'Instrumentation Virtuelle est déjà utilisée dans la recherche, le développement, la production. En ce qui concerne les applications, on la retrouve autant dans les communications que le traitement des eaux, la chimie, l'électronique, le spatial ou l'automobile.

4.1.5 Le respect des budgets

Le dernier point à prendre en compte dans le choix des outils qui accompagneront la démarche des enseignants nous ramène à la dure réalité puisqu'il concerne le coût par poste.

Depuis quelques années, on observe en effet que, dans les matières scientifiques, le nombre d'élèves décroît alors que, dans le même temps, le nombre de PC par étudiant explose. Autrement dit, il faut absolument que les moyens informatiques ne fassent pas exploser l'enveloppe budgétaire. Ceci explique en partie le succès de Linux et des logiciels libres (GNU et consorts) dans l'enseignement puisqu'ils sont censés réduire à zéro le coût de possession des logiciels. En ce qui concerne les moyens d'E/S, on observe par exemple un très fort engouement pour les matériels USB qui peuvent être facilement utilisés sur un poste puis sur un autre.

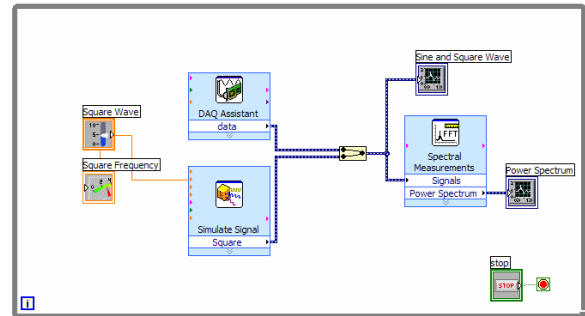
Il n'empêche... L'industrie utilise peu ou pas de logiciels libres et le besoin de mettre entre les mains des élèves des outils modernes, standardisés dans l'industrie entraîne bien souvent des problèmes de coût de licence ou de gestion des mises à jour. Ceci dit, une solution existe lorsque les outils logiciels sont multiplates-formes (Linux, Windows, Sun, HP, Mac...) puisque cela permet de former les étudiants sur une plateforme quitte à ce que, par la suite, ils utilisent ses mêmes outils sur un autre système d'exploitation.

4.2 LabVIEW comme outil pédagogique

LabVIEW est le logiciel phare de National Instruments. Dès le départ, il a été conçu comme l'outil de développement des Instruments Virtuels. De ce fait, il comporte tout un ensemble de caractéristiques qui en font l'outil de prédilection pour l'enseignement des sciences de l'ingénieur.

4.2.1 L'environnement LabVIEW

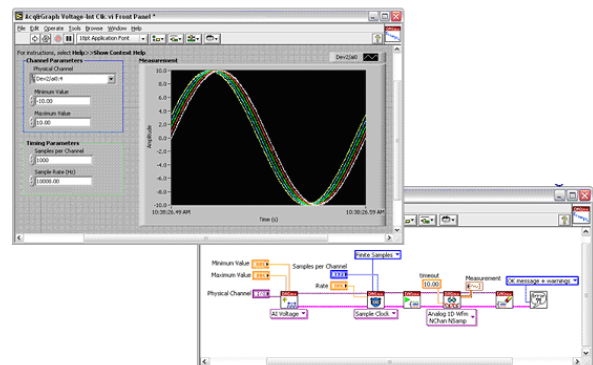
La caractéristique essentielle de l'environnement de programmation LabVIEW est qu'il s'appuie sur un langage de programmation graphique.



Exemple de code écrit avec LabVIEW

Que l'on ne se méprenne pas. Il s'agit d'un véritable langage de programmation qui s'appuie sur un parser, un compilateur et un éditeur de liens pour générer des applications exécutables (.exe sous Windows par exemple). Ces dernières supportent le multithread, les dernières technologies (.NET, .COM, ActiveX sous Windows par exemple) ou bien encore l'appel de code externe que ce soit sous forme de bibliothèques dynamiques ou statiques.

L'environnement, quant à lui, comprend tous les outils nécessaires à la création d'applications de professionnelle : éditeur d'interface graphique, débogueur, profiler, gestion de versions, etc. L'IDE est multiplates-formes et disponible par exemple sous Windows, Linux, Mac, Sun, HP, VxWorks, etc.



Les deux composants de toute application LabVIEW : l'interface graphique et le code source graphique

Toute application développée avec LabVIEW est composée de deux parties intimement liées. L'interface graphique correspond à la fenêtre que l'utilisateur final

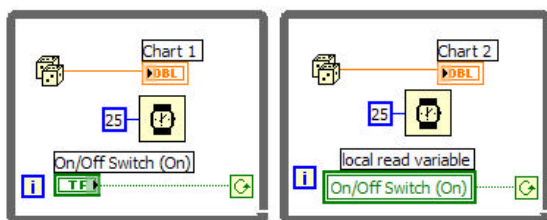
manipulera. Elle est conçue à l'aide d'un éditeur graphique qui permet de déposer tous les objets nécessaires à la création d'Instruments Virtuels. En effet, puisque LabVIEW "connaît" a priori le type d'application que le développeur a en tête, il comprend en standard des objets graphiques tels que des graphes, des graphes déroulants, des boutons rotatifs, etc. Avec ces composants, concevoir un oscilloscope, un analyseur logique, un superviseur ou un frontal de contrôle/commande est une affaire de minutes. Notons qu'à aucun moment, l'élève n'a besoin de se plonger dans l'API bas niveau de l'OS sur lequel il travaille pour gérer la fenêtre de l'application, il est donc très rapidement opérationnel.

4.2.2 La programmation graphique

Par la suite, associé à cette interface graphique, il va falloir coder la logique de l'application et pour cela, utiliser le langage de programmation graphique. Cette mise en œuvre a des implications pratiques très importantes.

Premièrement, le modèle de programmation graphique correspond très bien aux ingénieurs qui ont l'habitude de travailler avec des synoptiques. Intellectuellement, il n'y a donc pas de difficulté particulière et l'élève passe facilement de l'organigramme au code graphique.

Ensuite, par nature, le langage graphique est intrinsèquement parallèle. En effet, acceptons l'idée qu'une boucle "tant que" se retrouve dans le code sous forme d'un rectangle dont le contenu s'exécute tant que la condition de sortie n'est pas vérifiée (voir l'illustration suivante). Si c'est le cas, on imagine qu'il est très facile de mettre côte à côte deux boucles "tant que" dans le même diagramme.



Deux boucles "tant que" parallèles

Autrement dit, alors que le monde autour de nous est fondamentalement parallèle, on voit qu'avec LabVIEW, traduire la pensée du développeur (elle aussi massivement parallèle) ne pose pas de problèmes insurmontables au niveau du codage.

D'autres mécanismes existent afin de faciliter la prise en main du langage. Ainsi, l'élève n'est pas confronté à la barrière du langage. En effet, par nature, la syntaxe graphique évite l'usage des "#include" et autres

points-virgules qu'imposent des langages comme le C, le C++, C#, Java ou PHP, etc. Une fois le développement commencé, l'élève reste donc concentré sur la logique de l'application et ne perd pas de temps avec les détails de l'implémentation du code.

Toujours dans le même ordre d'idée, la notion de déclaration des variables n'existe pas en langage graphique. En effet, dès qu'un contrôle est instancié sur l'interface graphique de l'application, sa variable associée apparaît automatiquement dans le code. Cette méthode est beaucoup plus rapide que celle des langages textuels où les objets doivent être nommés puis associés à des variables qu'il faut avoir préalablement déclarées dans le code. En complément, des mécanismes existent pour gérer les cas où des objets graphiques sont créés dynamiquement sur la fenêtre de l'application et le langage permet de créer, côté code, toutes les constantes nécessaires à l'application.

En ce qui concerne la gestion de la mémoire (allocation, libération), LabVIEW dispose de son propre gestionnaire de mémoire ce qui réduit à néant les préoccupations qu'engendre la gestion de cette ressource.

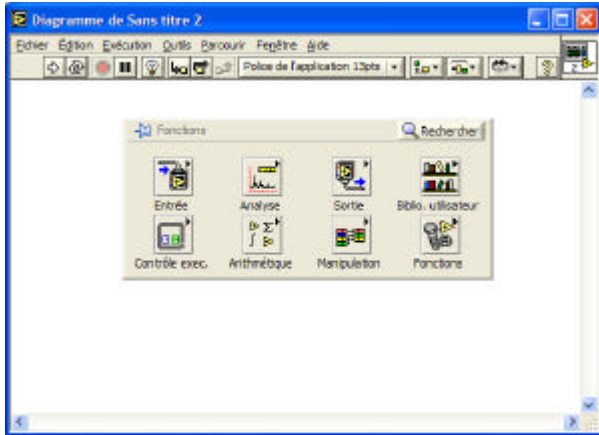
Concernant la gestion des codes source, l'usage veut que l'on applique les mêmes règles qu'en programmation textuelle à savoir : "dès qu'un diagramme (un code source) n'apparaît plus entièrement à l'écran, j'en fais une fonction". Autrement dit, l'utilisateur peut (et doit) créer ses propres fonctions afin de simplifier la lecture de son code source.

Cela conduit à parler des fonctions et des structures de données. Toutes les structures et tous les types de données classiques existent dans LabVIEW : boucle "for", boucle "tant que", etc. doubles, réels, entiers signés ou non, etc.

Pour le reste, toutes les fonctions de base (pensez aux fonctions de la bibliothèque C ANSI, par exemple) sont aussi disponibles. Ceci dit, en standard, l'environnement comprend tout un ensemble de fonctions supplémentaires spécifiques à la création d'Instruments Virtuels : E/S, GPIB, Série, TCP/IP, Mail, Analyse, etc.

Concernant les fonctions, leur utilisation est simplifiée à l'extrême. Contrairement à d'autres environnements où il faut se rappeler le nom des fonctions avant de les saisir dans le code source, l'éditeur de diagramme LabVIEW utilise un mécanisme de palettes graphiques afin d'assister l'utilisateur. Chaque palette et sous-palette regroupe sous forme de classes des ensembles de

fonctions spécifiques à telle ou telle type d'activité (gestion du temps, les fichiers, manipulation des chaînes de caractères, etc). L'utilisateur n'a donc pas besoin de se rappeler le noms de toutes les fonctions. Il lui suffit de parcourir les différentes palettes afin de trouver la ou les fonctions dont il a besoin.



La première palette qui apparaît dans le diagramme.

Ensuite, ayant trouvé une fonction, d'un simple clic l'élève va déposer cette dernière dans le diagramme. Elle apparaît alors comme un bloc fonctionnel avec des entrées et des sorties. L'invocation des fonctions est, elle aussi, simplifiée. En effet, à l'instar du C++, les fonctions graphiques possèdent des paramètres avec des valeurs par défaut. Autrement dit, il n'est pas nécessaire d'alimenter toutes les entrées et toutes les sorties des fonctions. Cela permet de gagner un temps précieux lors de l'édition du code.

Concernant les fonctions, il est intéressant de noter que nombre d'entre elles sont fournies sous forme de code source. Alors qu'une fonction se trouve dans un diagramme, un simple double clic sur le bloc fonctionnel permet d'accéder au code source (et à la face-avant) de la fonction en question. Du point de vue pédagogique, c'est un élément important puisque cela permet à l'utilisateur d'étudier par l'exemple et d'appliquer rapidement dans son code les techniques observées chez les autres.

Dernière remarque à propos des fonctions. Nombre d'entre elles sont polymorphes ce qui entraîne que leur comportement va changer en fonction du type des paramètres d'entrée. S'il n'est pas question ici de parler en détail de l'orientation objet que prend actuellement le langage graphique (GOOP, graphical object-oriented programming) il faut quand même faire remarquer qu'il s'agit d'une caractéristique du langage qui tend à simplifier l'écriture des codes source.

À propos de l'édition de code, cette dernière se fait logiquement avec une "bobine" que l'on utilise pour relier les sorties d'un bloc fonctionnel aux entrées d'un

autre. Modifier le code est une opération très rapide. Il suffit d'ajouter un fil ou d'en supprimer un autre. Comme les fonctions peuvent être invoquées avec une partie seulement de leurs paramètres d'entrée, on imagine la rapidité avec laquelle l'élève peut faire évoluer son code.

Nous l'avons dit, un diagramme LabVIEW est massivement parallèle et il peut être utile d'expliquer la façon dont les fonctions sont appelées. En fait, la règle est simple : une fonction est invoquée dès lors que tous ces paramètres d'entrée sont disponibles. Dans le cas où plusieurs fonctions ont toutes leurs paramètres d'entrée disponibles au même instant, si rien n'est fait, c'est le "scheduler" de LabVIEW qui décide d'invoquer telle fonction avant telle autre (techniquement un mécanisme interne à LabVIEW appelé Arbitrary Interleaving est utilisé). Ainsi, il est tout à fait possible qu'en mode "Debug", l'environnement exécute tel bloc fonctionnel avant tel autre alors qu'en mode "Run" l'ordre d'exécution soit différent. Bien sûr, des structures et des règles de codage existent et permettent de "sérialiser" les appels des fonctions si le besoin s'en fait sentir.

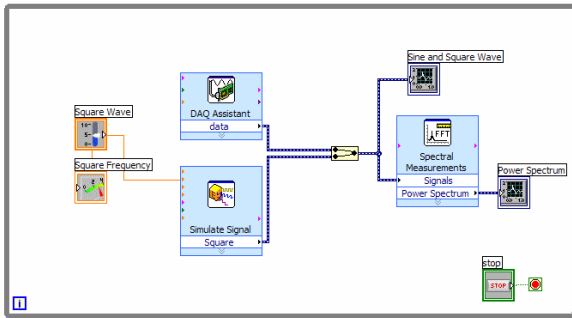
À un niveau plus élevé, à propos de la gestion de projet, il faut noter que cette dernière se réduit à sa plus simple expression : inexistant. Autrement dit, l'élève ne perd jamais de temps à résoudre des problèmes d'édition de liens, de bibliothèques manquantes, etc. Pour cela, LabVIEW s'appuie sur deux éléments. En règle générale, les bibliothèques standards qui sont incluses dans l'IDE suffisent à la majorité des applications typiques. Ceci dit, puisque LabVIEW est très ouvert, il existe de très nombreux compléments qui viennent enrichir le jeu de fonctions disponibles. Le mécanisme de prise en compte de ces bibliothèques dans l'environnement est alors très simple : "si la fonction est disponible dans l'environnement, alors l'application en cours de développement pourra être compilée et liée avec succès".

La dernière caractéristique du langage graphique sur laquelle nous souhaiterions insister est sa capacité à être facilement compris par les non-programmeurs. Il y a là deux aspects à mentionner.

Premièrement, il s'agit de la formidable collection d'exemples disponibles. À l'instar de ce que nous avons dit pour le code source des fonctions, les exemples évitent à l'élève de partir d'une feuille blanche et sont une très bonne source d'inspiration. De plus, il suffit d'un minimum de connaissances en LabVIEW pour pouvoir les étudier.

Deuxièmement, avec LabVIEW il est possible de travailler à deux sur un projet. "L'expert" explique ce qu'il fait, l'enchaînement des fonctions et la logique de l'application alors que le "novice", n'ayant aucune barrière liée à la syntaxe, suit et apprend très rapidement.

Que conclure de cette présentation de quelques-unes des caractéristiques de la programmation graphique ? En fait, il serait peut-être bon de tout oublier et d'étudier à nouveau le diagramme suivant :



Le code graphique permet vraiment à l'élève de se concentrer sur l'essentiel.

Quand on sait qu'une interface graphique est associée à ce code, on réalise de suite que l'élève va véritablement se concentrer sur l'essentiel. Faut-il utiliser telle fonction de filtrage plutôt qu'une autre ? Que se passe-t-il si l'on modifie ce paramètre dans ce sens ? Autrement dit, il ne va pas "perdre son âme" avec des problèmes de programmation. Au contraire, il va prendre plaisir à expérimenter, à modifier, à apprendre par la pratique.

C'est le constat qui a déjà été fait par de nombreux industriels et enseignants qui reconnaissent tous les avantages qu'offre la programmation graphique du point de vue pédagogique.

4.3 E/S et plateformes dans la démarche pédagogique

Soyons clair quitte à être excessif... Au troisième millénaire, dans le cadre d'un enseignement dispensé à de futurs ingénieurs nous sommes convaincus qu'il n'y a pas d'intérêt à étudier la programmation des moyens d'entrées/sorties au niveau des registres. La raison en est très simple : ce ne sont pas des connaissances que la majorité des élèves réutiliseront une fois sur le marché du travail.

Cette "bataille" ressemble beaucoup à ce que l'on connaissait en informatique il y a une quinzaine d'années et où les tenants du tout assembleur ferraillaient contre les programmeurs Turbo Pascal. Entre temps, la loi de Moore qui dit que la puissance des PC double tous les 18 mois est passée par là, les contraintes qu'impose la productivité ont légitimé l'usage d'outils de plus haut niveau et pratiquement

plus personnes ne travaille au niveau des registres. Oui, bien sûr, dans certaines applications (et LabVIEW en fait partie) il y a une ou deux routines très critiques, fortement optimisées. Pour le reste, même les programmeurs de jeux utilisent dorénavant des outils de haut niveau (C++, script de génération de scènes etc.).

Par contre, du point de vue matériel, il est sans doute de plus en plus important d'enseigner l'art et la manière de dimensionner un système d'Instrumentation Virtuelle (combien de voies, quels types, quelles vitesses, quelles résolutions...). De même, l'étude des moyens qui garantissent l'intégrité des données (tous les problèmes liés au conditionnement du signal, au calibrage, aux erreurs de mesure et à leur propagation) ne peuvent être négligées.

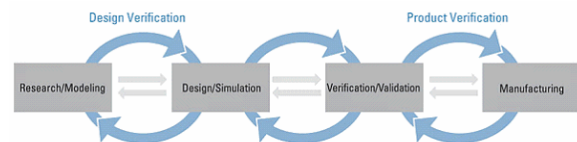
Enfin, l'étude des plates-formes, de leurs contraintes, de leurs performances est, elle aussi, nécessaire (quand faut-il déployer un code sur cible RT, quelles sont les contraintes liées à la bande passante des différents bus, quelle influence la plate-forme a-t-elle en termes de synchronisation, quelle plate-forme choisir pour une application distribuée).

5 TENDANCES ET PERSPECTIVES

En 2004, uniquement avec les produits National Instruments, plus de six millions de voies de mesures ou de contrôle ont été mises en œuvre dans des applications d'Instrumentation Virtuelle. De même, plus de 1000 produits multifournisseurs sont à l'heure actuelle disponibles au format PXI. Les machines équipées d'un bus PCI Express sont désormais largement disponibles. Il semble donc que toutes les conditions soient réunies pour que l'Instrumentation Virtuelle soit de plus en plus utilisée.

5.1 L'Instrumentation Virtuelle dans le cycle de développement des produits

Cependant, en complément du test et des applications de contrôle/commande, les industriels demandent à ce que l'Instrumentation Virtuelle soit dorénavant utilisée tout au long de la chaîne qui s'étend de la phase de Recherche-Modélisation à la phase de Production en passant par les phases de Conception-Simulation et de Vérification-Validation.



Le cycle de développement de n'importe quel produit

On peut légitimement se demander les raisons d'une telle tendance. En fait, si l'on accepte l'idée que l'Instrumentation Virtuelle est très largement utilisée dans le test de production, les industriels font alors le

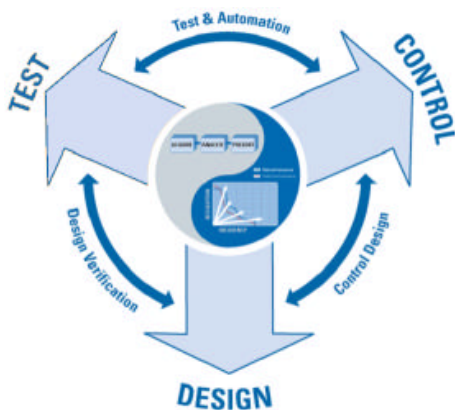
constat suivant : compte tenu du rythme auquel nous devons mettre de nouveaux produits sur le marché, nous avons deux leviers sur lesquels nous pouvons agir. On peut tenter de raccourcir le temps que prend chaque phase (de la Recherche à la Production) et on peut aussi essayer de diminuer le nombre d'itérations dans les cycles de test qui permettent de passer d'une phase à l'autre.

Pour y arriver, la première chose à faire, c'est sans doute de standardiser les méthodes et les outils utilisés. Il est évident que l'environnement LabVIEW n'est pas le bon outil pour concevoir le "layout" d'une carte électronique. Cependant, si LabVIEW peut collaborer avec les autres outils utilisés par l'ingénieur afin de récupérer un certain nombre de paramètres et faciliter le test d'un premier prototype, c'est autant de temps gagné. C'est dans cet esprit que LabVIEW peut d'ore et déjà collaborer avec toute une panoplie d'outils : SPICE, Simulink, SolidWorks, Multisim, Mathematica...

Dans un autre ordre d'idée, on sait que les ingénieurs de conception ou de simulation ne sont pas nécessairement prêts à développer une application pour automatiser la prise de mesures sur un prototype et comparer ces valeurs avec celles issues d'un simulateur SPICE. Ceci dit, une application telle que SignalExpress peut sans doute permettre d'augmenter la vitesse à laquelle un produit est mis sur le marché. En effet, non seulement la comparaison avec les données SPICE est possible et automatique mais en plus, le script développé par l'ingénieur de conception est utilisable par un ingénieur de test en production sous forme d'un code LabVIEW.

5.2 Les trois axes de développement de l'Instrumentation Virtuelle

On peut donc résumer les trois grands axes de développement de l'Instrumentation Virtuelle dans les années à venir avec l'illustration suivante.



Les trois grands axes de développement de l'Instrumentation Virtuelle dans les années à venir

Il est sans doute bon de remarquer qu'en plus des vecteurs Test, Contrôle/Commande et Conception, l'Instrumentation Virtuelle a les moyens de répondre à toutes les applications qui se situent entre le test et le contrôle, le test et la conception et enfin entre la conception et le contrôle/commande.

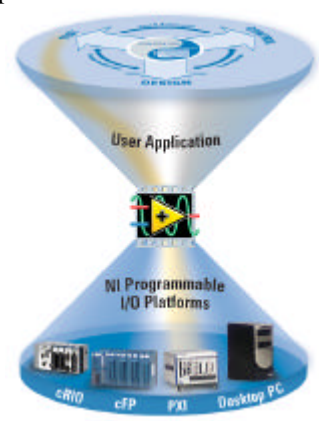
Dans le cas des efforts faits pour développer l'Instrumentation Virtuelle suivant l'axe de la Conception, rappelons qu'aujourd'hui il est possible d'intégrer des modèles Simulink au sein d'un code graphique LabVIEW. Ceci étant fait, il est alors possible de déployer ce modèle sur une cible temps réel afin d'élaborer, sur une base matérielle National Instruments, une application "hardware-in-the-loop".

Dans le même ordre d'idée, il est tout à fait envisageable de créer ses propres modèles directement au sein d'un code graphique. Pour cela, il suffit d'utiliser un nœud de simulation dans lequel on tape ses équations en s, et où l'on utilise des boucles de contre-réaction.

5.3 L'Instrumentation Virtuelle s'affranchit du PC

L'autre tendance de l'Instrumentation Virtuelle est de s'affranchir du PC et de permettre à l'utilisateur de déployer son code sur d'autres types de plates-formes.

Jusqu'à il y a peu, la situation pouvait donc être symbolisée par l'illustration suivante.



LabVIEW, outil de déploiement sur de multiples plates-formes

Typiquement, quel que soit le domaine d'application (test, contrôle/commande, conception), l'utilisateur développe son Instrument Virtuel et peut, parce qu'il a fait le choix de la programmation graphique en général et de LabVIEW en particulier, déployer son code sur tout un ensemble de plates-formes. Deux remarques cependant.

Jusqu'à aujourd'hui, la situation n'est pas aussi idyllique qu'il y paraît car, par exemple, le développement d'un code FPGA oblige à prendre en compte quelques spécificités. Ceci dit, c'est l'objectif des ingénieurs NI et

beaucoup d'efforts sont faits pour aplanir, côté développeur, les différences qui peuvent exister d'une plate-forme à l'autre.

Le deuxième point est lui beaucoup plus gênant. En effet, une étude des cibles disponibles révèle en fait que ces dernières sont toutes des cibles PC 32 bits. Or, la demande est de plus en plus forte pour que la programmation graphique permette de déployer des Instruments Virtuels sur d'autres types de cibles.

5.4 Des Instruments Virtuels dans le silicium

La raison de cette demande est relativement simple. En effet, les périphériques électroniques résultent de plus en plus d'une association entre l'électronique et le logiciel. Ce dernier est alors généralement exécuté au cœur d'un DSP, d'un FPGA ou d'un processeur embarqué. À partir du moment où il s'agit d'associer des E/S à du logiciel, il est naturel qu'un certain nombre d'industriels et d'universités travaillent avec NI afin que le modèle d'Instrumentation Virtuelle soit généralisé à ce type de cible.

L'illustration suivante schématise donc l'un des grands axes de développement de l'Instrumentation Virtuelle dans les années à venir.



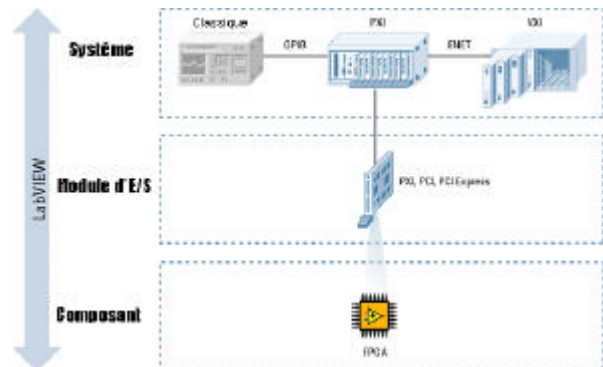
La programmation graphique va devenir un outil de déploiement de code dans le silicium.

Encore une fois, l'idée simple de départ consiste à réaliser qu'un DSP, qu'un processeur ARM, qu'un FPGA ou qu'un microcontrôleur associé à des moyens d'E/S et exécutant un code n'est ni plus ni moins qu'un Instrument Virtuel. À partir de là, il devient crucial de tout mettre en œuvre pour que LabVIEW soit capable de déployer du code sur ce type de cible.

Actuellement, alors que l'on en est encore aux prémices, LabVIEW est déjà capable de déployer du code sur n'importe quel processeur 32 bits, sur certains DSP Texas Instruments ainsi que sur les processeurs Blackfin.

5.5 Du système au composant

Afin de résumer toutes ces tendances on peut s'appuyer sur l'illustration suivante.



L'Instrumentation Virtuelle permet la programmation des systèmes, des modules d'E/S et des composants.

En fait, alors que l'idée qu'il est possible de concevoir des instruments entièrement définis par l'utilisateur est maintenant largement acceptée, on réalise que le marché tend à pousser encore plus loin le raisonnement.

"S'il est possible d'optimiser les systèmes avec l'Instrumentation Virtuelle, alors il doit être possible de faire de même avec les modules d'E/S. Or, ces derniers embarquent de plus en plus souvent des processeurs. Ceux-ci sont soit des processeurs qui exécutent des codes que l'utilisateur veut être capable d'optimiser soit des FPGA dont on doit être capable d'optimiser la programmation. Bref, il est impératif de faire quelque chose".

Nous avons mentionné quelques produits et technologies qui viennent étendre les possibilités de l'Instrumentation Virtuelle. Ceci dit, encore une fois, quand on regarde ce qui a été fait par le passé, on remarque qu'à chaque fois qu'une tendance ou une annonce a été faite, National Instruments a toujours exécuté ce qu'elle avait annoncé. Autrement dit, il n'y a aucun doute qu'à terme et à la demande des industriels, les utilisateurs/concepteurs d'Instrumentation Virtuelle auront la possibilité de configurer les systèmes, les modules et les composants.

Du point de vue de l'enseignement, être capable d'évaluer ces tendances et mettre en place les outils pédagogiques susceptibles de s'y adapter en temps voulu est très certainement un enjeu de taille auquel l'Instrumentation Virtuelle peut apporter des éléments de réponse.

6 CONCLUSION

L'Instrumentation Virtuelle, c'est l'association d'outils informatiques et d'E/S matérielles qui permettent la création de systèmes définis par l'utilisateur. Nous

avons montré que cette approche ne peut pas se résumer à l'automatisation de prise de mesures sur PC. Au contraire, après plus de vingt ans d'utilisation dans le monde industriel son usage s'étend du test de production aux autres phases de création de produits (validation, conception, modélisation). De plus, nous avons expliqué qu'à partir de la plateforme PC et grâce au logiciel qui permet au développeur d'avoir le niveau d'abstraction nécessaire, son usage se répand sur d'autres cibles telles que PDA, DSP, FPGA.

Autrement dit, de notre point de vue, l'Instrumentation Virtuelle a toute sa place dans l'enseignement des sciences de l'ingénieur. En effet, une fois sensibilisés aux outils et technologies sous jacentes, ces derniers posséderont alors des connaissances et un bagage applicable dans un très large champ d'applications et de domaines. Pour le reste, tout est affaire d'imagination...