Accélérateur virtuel

Concept, implémentation
et
premier test

D. Uriot, R. Duperrier
CEA/DSM/DAPNIA/SACM/LEDA
Le 10 mai 2006

1 Introduction

Dans le cadre du projet SPIRAL2, l'éventualité de concevoir et d'implémenter un accélérateur virtuel a été évoquée. Il apparaît au regard de divers échanges, notamment avec l'équipe en charge du système de contrôle commande, qu'il était nécessaire d'expliquer ce concept dans une note technique. L'injecteur SILHI fait l'objet d'un premier test de ce type. Ce papier vise donc à présenter l'ensemble des applications et des avantages qu'apporterait la mise en œuvre de ce type de concept. Plusieurs développements furent menés au KEK [1,2] et d'autres sont encore en cours, notamment, pour l'accélérateur SNS [3]. Tous ont montré l'intérêt de cette approche.

Un accélérateur virtuel est le couplage d'un code de simulation avec le système de contrôle commande d'une machine réelle. Virtuel : parce que vue du système de contrôle commande, le code de simulation se comporte comme une machine réelle. Ce lien peut être de plusieurs natures et permettre plusieurs modes de fonctionnement différents. Ces différents aspects sont décrits ainsi que leurs applications et intérêts. La manière de les implémenter est montrée. Les outils utilisés ainsi qu'un premier exemple d'application sont exposés.

Finalement, ce rapport a pour objectif de faire d'une part la synthèse de nos réflexions sur ce sujet ainsi que le bilan de l'état d'avancement de leurs développements.

2 Limitations

Les limitations sont de deux ordres. Un accélérateur virtuel n'est évidement pas aussi rapide que la machine réelle. Patienter quelques secondes pour visualiser les résultats d'un changement de réglage se conçoit parfaitement. Quelques minutes deviennent rédhibitoires dans le cadre de fonctionnement en machine virtuelle. À l'heure actuelle, il paraît difficile d'exploiter par exemple les simulations d'un RFQ demandant au mieux quelques dizaines de minutes.
La seconde limitation est la qualité de sa réponse attendue qui est définie par la précision du modèle décrivant la physique de la machine réelle. Bien souvent, cette précision est fortement liée au temps de calcul.
Toutefois un des enjeux du concept d'accélérateur virtuel est de fournir un outil performant pour l'amélioration des codes de simulation. Ces limitations sont donc des bonnes raisons de s'intéresser à la conception d'un accélérateur virtuel.

3 Les modes de fonctionnement

Les différents modes de fonctionnement listés ci-dessous donnent un aperçu de l'apport de chacun d'entre eux du point de vue du physicien accélérateur ou de l'opérateur. Cette liste n'est pas exhaustive et peut bien entendu être enrichie.

3.1 Mode « surveillance »

Dans ce mode, toute action sur le système de contrôle commande de la machine, agit sur l'accélérateur réel et l'accélérateur virtuel. L'accélérateur virtuel est donc directement lié aux paramètres de réglage de la machine réelle et simule celle-ci. Cette simulation pouvant être automatiquement démarrée à chaque modification d'un réglage ou se faire périodiquement. La machine est sous la surveillance permanente de la machine virtuelle. Le code de simulation n'a besoin d'accéder qu'en lecture à la base de données décrivant l'accélérateur.

- **Intérêts pour la physique machine :** la comparaison entre les diagnostics des accélérateurs réel et virtuel permet de confronter le code de simulation avec la machine réelle. Au delà de l'amélioration des codes de simulation espérée, on attend une compréhension plus profonde de l'accélérateur, ce qui est essentiel à l'amélioration de celui-ci.
- **Intérêts pour l'opération machine :** la visualisation directe et complète (enveloppes, espace des phases, grandeurs physiques, diagnostics, etc.) de l'état du faisceau simulé en fonction des réglages permet de mieux comprendre le fonctionnement de l'accélérateur. Ce mode de fonctionnement a un intérêt évident pour la formation.

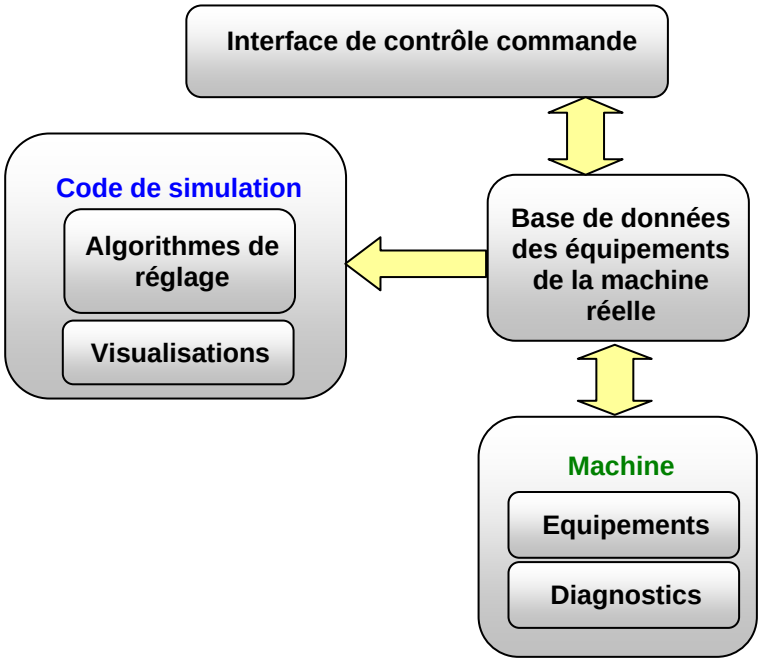

Figure 1 : Synoptique du mode surveillance

3.2 Mode « simulateur de vol »

Pour permettre ce mode, toute action sur le système contrôle commande de la machine, agit soit sur l'accélérateur réel soit l'accélérateur virtuel. L'utilisateur peut donc à volonté modifier via le système de contrôle commande soit la machine réelle soit la machine virtuelle. Dans ce dernier cas, par exemple, contrôler l'influence de réglages avant de les transmettre à la machine réelle. On dispose réellement de deux machines distinctes, l'accélérateur et son simulateur. Ce mode de fonctionnement suppose évidemment qu'il existe donc deux systèmes distincts de base de données représentant les deux accélérateurs. De plus, des commutateurs logiciels doivent permettre au système de contrôle commande de passer aisément et de manière sécurisée de l'une à l'autre, et éventuellement de copier les réglages d'une base de données à l'autre. Le code de simulation n'a besoin d'accéder qu'en lecture à la base de données virtuelle.

- **Intérêts pour la physique machine :** évaluer des réglages ou des procédures de réglage avant de les tester sur la machine réelle sans affecter son exploitation. Tester des modifications de la machine avant de les implémenter sur la machine réelle.
- **Intérêts pour l'opération machine :** prévisualiser ce qu'on s'apprête à commander sur la machine permet d'éviter des erreurs graves et de gagner du temps lors de réglages compliqués. Ce mode de fonctionnement a aussi un intérêt évident de formation en autorisant des entraînements malgré l'exploitation de la machine.

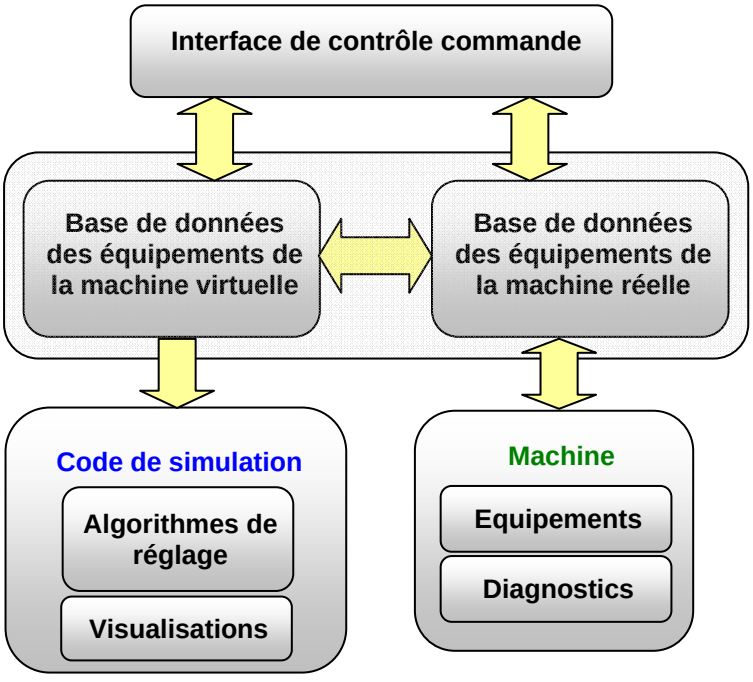

Figure 2 : Synoptique du mode simulateur de vol

3.3 Mode «pilote automatique »

Les codes de simulations modernes possèdent pour la plupart des capacités de réglage des accélérateurs simulés. Mieux encore, certains permettent de simuler un réglage automatique de la machine ou d'une partie en ne se basant que sur l'information issue de diagnostics virtuels représentants ceux réellement implémentés sur l'accélérateur réel.
Dans ce mode, on utilise les algorithmes de réglage automatique du code de simulation pour régler la machine réelle. Pour ce faire, le code utilise en entrée la description de la machine réelle. Il se base sur les informations issues des diagnostics réels et il agit cette fois directement sur les paramètres de réglage de la machine réelle. Il se comporte comme un système de contrôle commande de haut niveau qui met en quelque sorte la machine en pilotage automatique. Le code de simulation prend complètement la main sur la machine et doit avoir accès en lecture et en écriture à la base de données.

- **Intérêts pour la physique machine :** mise au point et tests des procédures et algorithmes de réglages automatiques de parties ou de l'ensemble de la machine. La partie simulation n'étant pas utilisée, il n'est pas nécessaire que toute la physique soit dans le code.
- **Intérêts pour l'opération machine :** les réglages complexes, délicats ou fastidieux peuvent être effectués de manière automatique et beaucoup plus rapide.

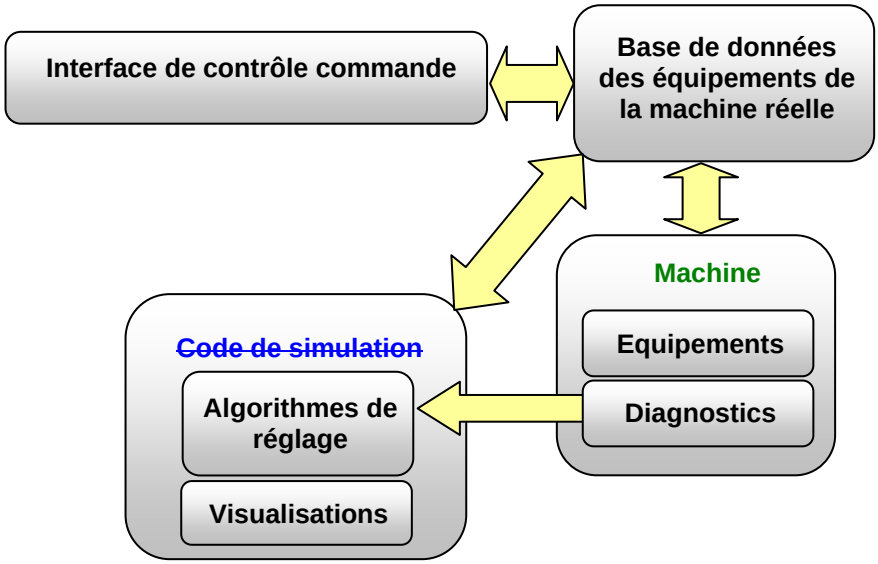

Figure 3 : Synoptique du mode pilotage automatique

3.4 Synoptique général

Pour permettre l'ensemble des modes de fonctionnement, il faudra implémenter 2 bases de données indépendantes ou une unique base contenant des champs séparés représentant les états et valeurs des équipements des machines virtuelle et réelle. Ces différents modes peuvent être résumés par un synoptique général où l'ensemble des liaisons nécessaires est présenté.

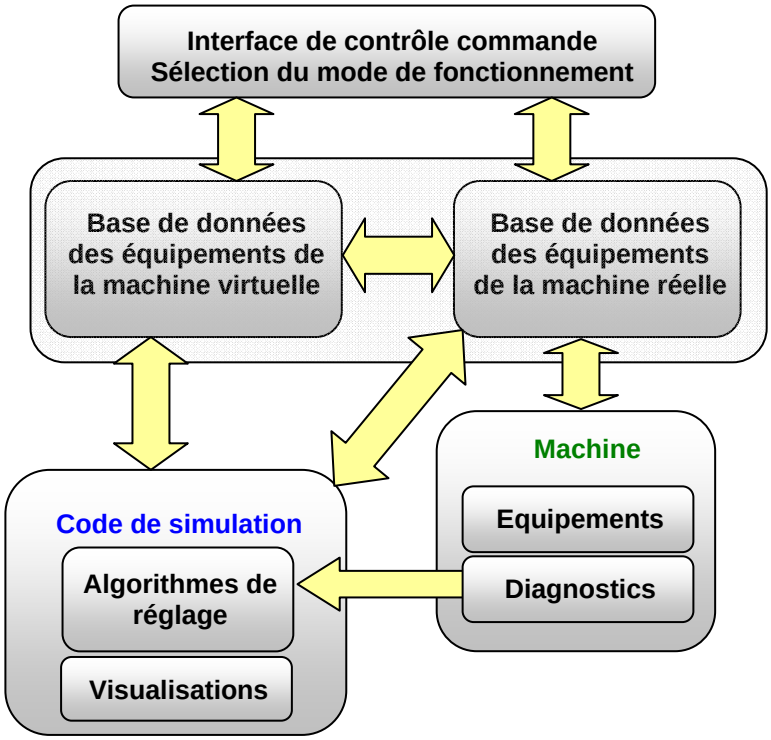

Figure 4 : Synoptique général

4 Implémentation

Beaucoup d'accélérateurs dans le monde ont basé leur système de contrôle commande sur EPICS [4], c'est aussi le cas d'IPHI et de l'injecteur de SPIRAL2. Les premiers essais de machines virtuelles sont pour certains basés sur ce système. Dans cette note le code de simulation utilisé est TraceWin [5] développé au SACM.

4.1 EPICS

EPICS (Experimental Physics and Industrial Control System) est ensemble d'outil de programmation et de programmes qui réalise une infrastructure logicielle pour utiliser et construire un système de contrôle distribué. EPICS présente plusieurs avantages pour la mise en œuvre d'un accélérateur virtuel. D'abord, il est utilisé par une large communauté et en particulier au CEA. De plus, il est portable sur la plupart des environnements (OS) courants. TraceWin ne fonctionne pour l'heure que sur système Windows ou sous Linux via un émulateur (Wine[6]). Avec EPICS est fournie la plupart des passerelles ou interface de programmation vers les langages de programmation évolués. L'accès à la base de données se fait à travers le protocole « Channel Access ». Pour finir, les interfaces graphiques d'utilisateurs sont variées et nombreuses. Moderne et d'une compréhension et utilisation aisée, EPICS semble pour l'heure l'outil idéal pour démarrer type de développement. La Figure 5 représente l'architecture d'EPICS.

EPICS repose sur une base de données « EPICS BD » composée d'enregistrements ou « RECORD » représentant des objets ou équipements. Chacun de ces RECORDs contenant de nombreux champs représentant les principaux paramètres de l'équipement considéré. Toutes ces informations sont accessibles en lecture et en écriture via le Channel Access.

Pour simplifier, on ne considérera pour un équipement donné que 3 champs appartenant au même RECORD :

- **La valeur de commande** : représente la grandeur que l'on désire transmettre à l'équipement. Par exemple, on désire envoyer 200 A dans la bobine d'un solénoïde.

- **La valeur virtuelle de commande** : représente la grandeur que l'on désire transmettre au code pour la simulation de cet équipement. Par exemple, on désire simuler la machine avec 200 A dans la bobine du solénoïde.

- **L'acquisition de cette grandeur** : représente l'acquisition de la valeur de cet équipement. Par exemple, on mesure 200.3 A la sortie de l'alimentation connectée à ce solénoïde.

Avec la structure des RECORDs telle qu'elle est définie dans EPCIS on peut donc se contenter d'une seule base de données pour représenter les deux accélérateurs. Il faut noter, que certains équipements tels que les diagnostics ne seront utilisés qu'à travers le dernier champ.
L'accès aux différents champs des RECORDs s'effectue via les « Real Process Variable » ou PV's variables.

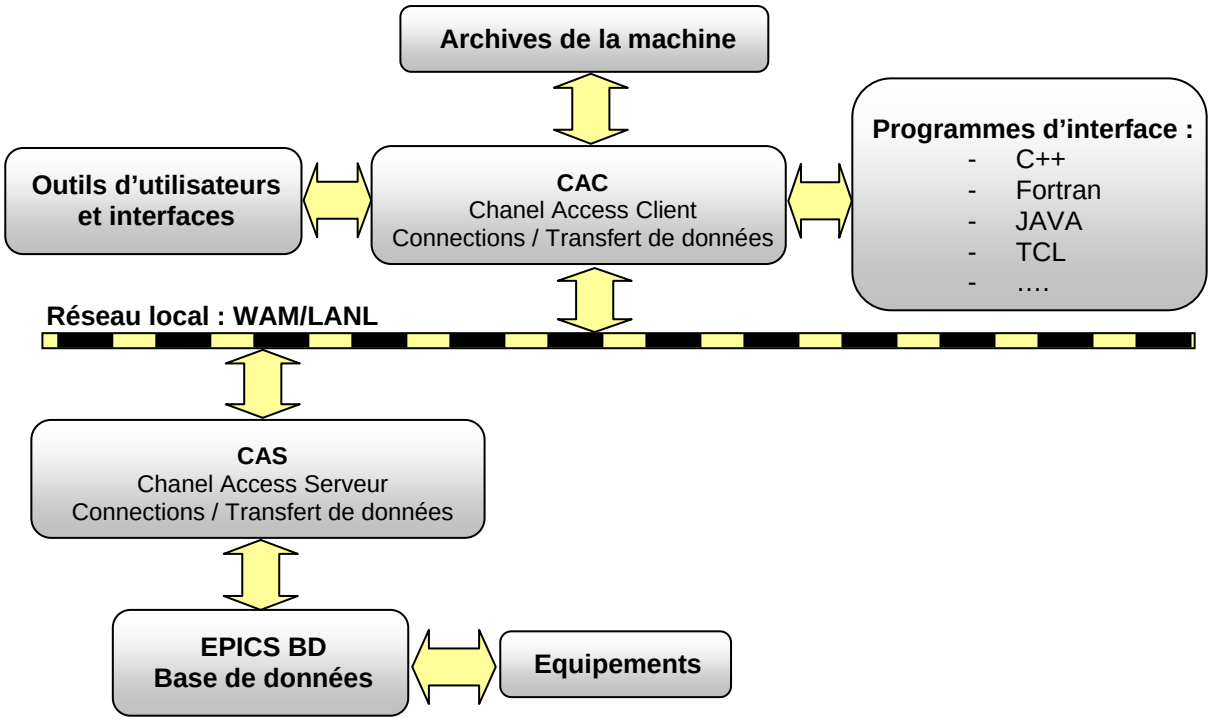

Figure 5 : Architecture d'EPICS

4.2 TraceWin

TraceWin [5] est un programme de calcul de l'enveloppe d'un faisceau de particules chargées en paquets ou continu à travers différents éléments d'un accélérateur linéaire. C'est un code matriciel incluant les effets linéaires des forces de charge d'espace. C'est un outil complet de développement, conçu pour les réglages automatiques d'une ou de plusieurs sections de

l'accélérateur en s'appuyant sur de nombreuses options. Le code permet de simuler les procédures de démarrage et de réglage de la machine en incluant les différents diagnostics et leurs correcteurs associés. Tous les transports effectués par le code peuvent être doublés à l'aide des codes multi particulaires Partran [5] et Toutatis [7] pour une confrontation (non-linéarités). Grâce à son interface graphique, l'utilisateur peut changer n'importe quel paramètre du design de la machine et observer rapidement les effets. Elle permet de visualiser la plupart des paramètres pertinents concernant ce type de développement tels que les enveloppes, les espaces de phase, les émittances, les avances de phase…
Ecrit en C++ ANSI, la limitation du code à Windows n'est due qu'à son interface graphique construite autour de VCL (bibliothèque des composants visuels). Il est parfaitement envisageable de la remplacer par des outils graphiques d'EPICS ou non portables sur d'autres systèmes comme Linux habituellement utilisé dans ce type d'application.

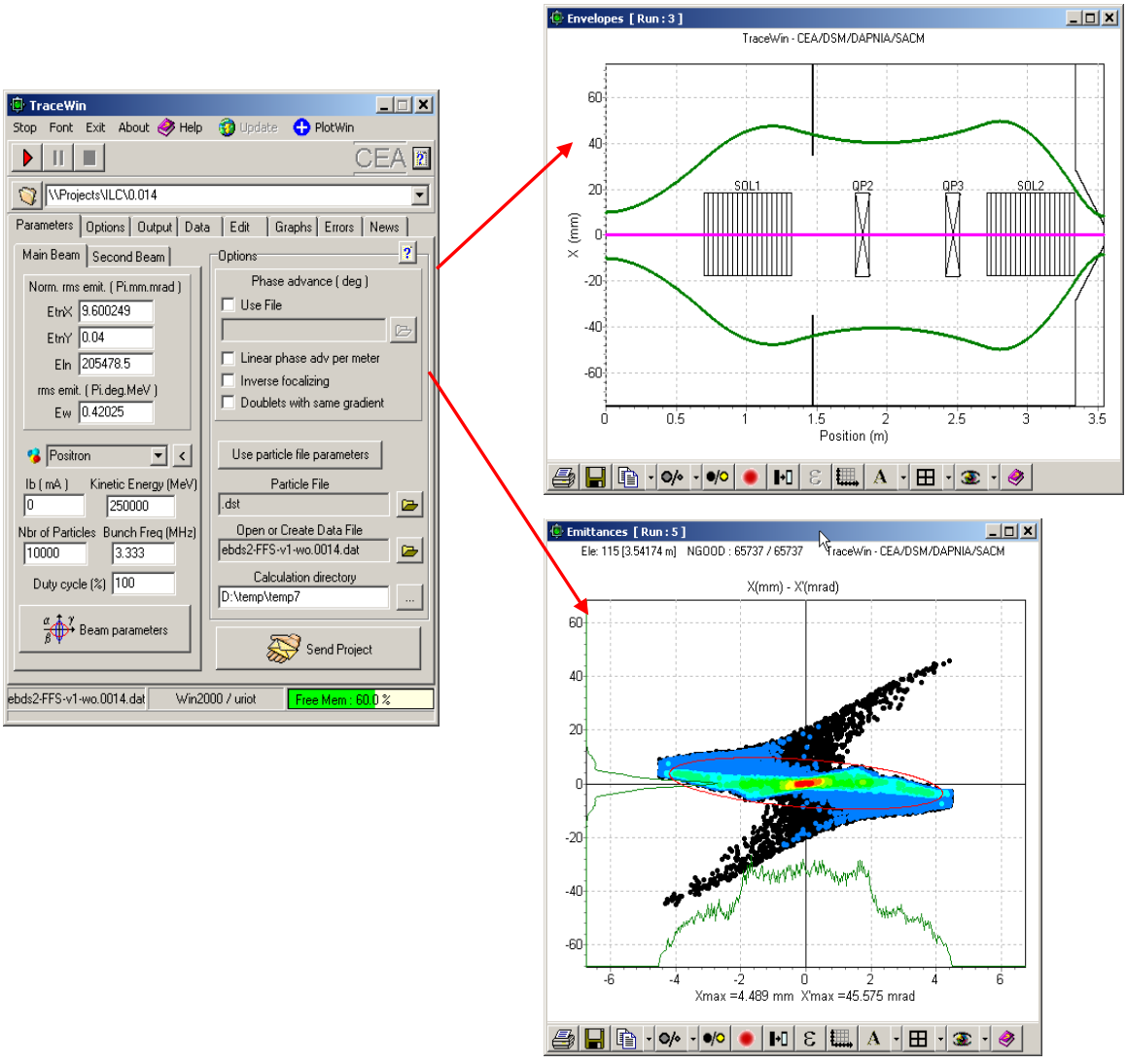

Figure 6 : Exemple de sorties du code TraceWin

4.3 Interface TraceWin ↔ EPICS

Les dialogues entre TraceWin et EPICS sont réalisés via l'utilisation de « EPICS Base ». C'est le cœur d'EPICS, comprenant les codes sources du système de base ou noyau, les

bibliothèques d'interfaçage avec les différents systèmes d'exploitation, les bibliothèques du Channel Access, le RECORD standard et de nombreux autres outils. Il y a différentes versions d'EPICS Base. Celle qui est utilisée ici, est la 3.14.8.2. Cette base permet de construire un Chanel Access Client (CAC) permettant la lecture et l'écriture dans la base de données d'EPICS. Le processus doit utiliser une base de données, à définir, interne au code de simulation affectant les correspondances entre les notations et les grandeurs physiques de TraceWin et d'EPICS tel que décrit Figure 7. Par exemple, TraceWin connaît le champ magnétique (T) d'un solénoïde nommé « SOL1 » alors que EPICS travaille avec le courant (A) de son alimentation et y accède par la variable PV appelée « lbe:sol1_Imes ». De plus, cette base de données doit contenir des coefficients permettant de faire coïncider les valeurs de commande avec les valeurs d'acquisition. En effet, commander par exemple 200 A sur la bobine d'un solénoïde n'aboutit pas nécessairement à l'acquisition de 200 A. Ces coefficients permettent de simuler réellement les grandeurs mesurées.

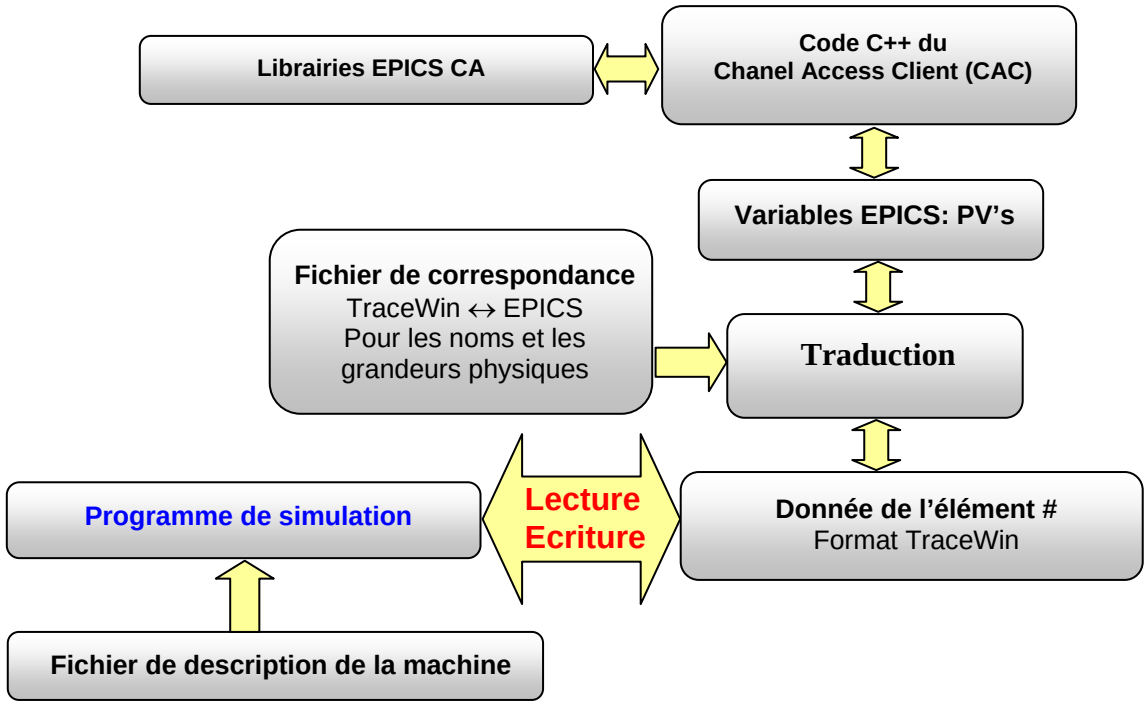

Figure 7 : Structure de base du code de l'accélérateur virtuel

Si le processus de lecture est relativement simple, l'écriture de données doit reposer sur un algorithme plus complexe afin de s'affranchir du non-synchronisme des réponses des équipements interrogés et commandés. D'autre part, il faut tenir compte du fait qu'une valeur de commande est la plupart du temps différente de la valeur d'acquisition retournée par l'équipement, ce qui n'est jamais le cas dans un code de simulation qui ne connaît pas cette nuance.

4.3.1 Synchronisation de l'écriture d'une donnée

L'algorithme permettant la synchronisation de l'envoi d'une grandeur sur un équipement est présenté Figure 8. Il repose sur deux boucles d'attente. La première est destinée à attendre l'arrivée de la commande sur l'équipement tandis que la seconde permet d'attendre la stabilisation de la valeur d'acquisition. La vitesse d'envoi d'une donnée est donc liée à la

vitesse de réponse des équipements. La durée des pauses doit être judicieusement ajustée de manière à ne pas ralentir les équipements rapides tout en évitant de surcharger le réseau d'accès inutiles.

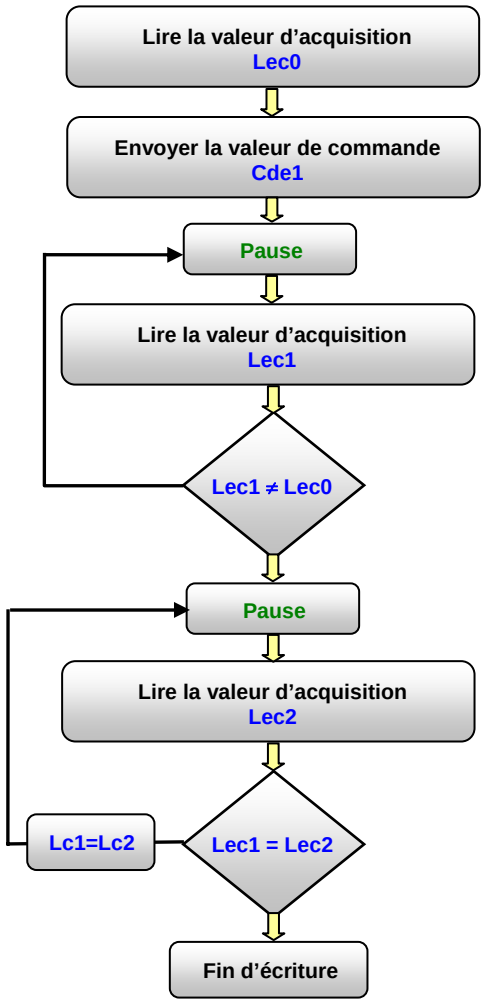

Figure 8 : Algorithme d'écriture

5 Le test sur SILHI

5.1 Problématique

L'injecteur SILHI est actuellement constitué d'une source ECR suivie de 2 solénoïdes (SOL1, SOL2) transportant le faisceau vers un cône métallique destiné à simuler l'entrée du futur RFQ et à collecter les espèces secondaires telles que les H_2^+ et H_3^+. Quatre déviateurs magnétiques (DV1, DH1, DV2, DH2) permettent de faire coïncider l'axe du faisceau avec l'axe du cône. Enfin, deux mesures de courant (D1, D2) indiquent la transmission de l'ensemble. Cette ligne est décrite Figure 9. Régler l'ensemble manuellement pour optimiser la transmission n'est pas aisé compte tenu du nombre de variables à ajuster et des couplages du système. Une étude uniquement orientée sur l'aspect alignement a montré les difficultés et les enjeux de ce réglage [8]. Il faut noter, que transmettre dans de bonne condition le faisceau à travers ce cône est une condition nécessaire mais pas suffisante pour valider la future

injection dans le RFQ de IPHI. En effet, le cône est un dispositif beaucoup plus acceptant que le RFQ.

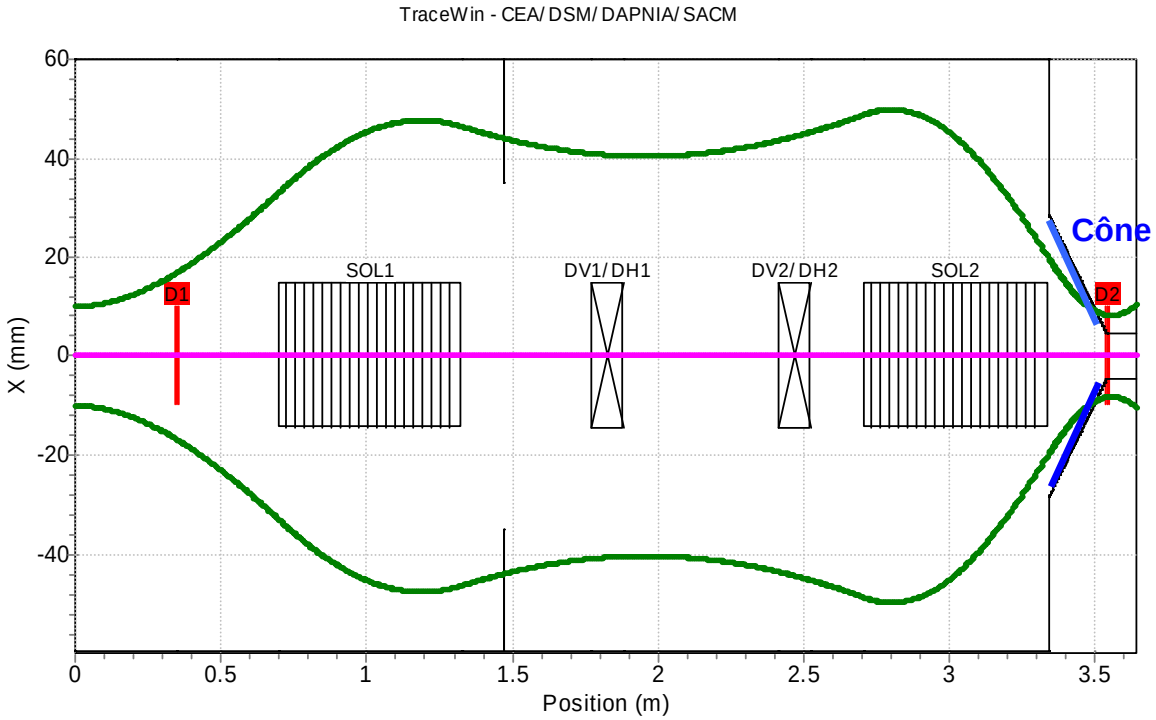

Figure 9 : La ligne SILHI

5.2 Mise en œuvre

Un code spécifique a été développé pour tester la possibilité d'automatiser l'optimisation de la transmission. Les algorithmes d'optimisation sont ceux issues de TraceWin (Gradient et Simplex). Leur rôle est d'ajuster par itérations successives les différents éléments de la ligne en fonction de la transmission. Le code contient évidemment l'interface TraceWin↔EPICS décrite précédemment. Il autorise un accès en lecture et écriture au Channel Access. C'est le noyau de l'accélérateur virtuel en mode « pilotage automatique ».

5.3 Résultats

Le Tableau 1 montre les transmissions et les réglages de la ligne SILHI pour une optimisation manuelle et pour celle obtenue avec le code en partant du précédent réglage manuel. Le gain est d'environ 4 %. Le Tableau 2 représente le résultat d'une optimisation effectuée à partir d'un réglage donnant une transmission très médiocre. Il faut noter, que ces 2 optimisations ne sont pas réalisées pour les mêmes conditions de la source. Elles n'aboutissent donc pas nécessairement aux mêmes paramètres et transmissions optimales. Dans les 2 cas la proportion de proton du faisceau principal est d'environ 80%. On peut donc estimer, pour la première optimisation, que l'on est proche de la transmission optimale de la ligne. Les images de la Figure 10 montrent le faisceau à la sortie du cône avant et après optimisation de la transmission. On voit clairement le décentrement initial du faisceau. Les optimisations prennent environ une demi-heure.

Réglage	Sol1 (A)	Sol2 (A)	Dh1 (A)	Dv1 (A)	Dh2 (A)	Dv2 (A)	Transmission (%)
Manuelle	115.2	165.6	0.3	0	0	0.3	82.7
Par code	130.6	177.6	0.99	-0.7	-0.25	0.67	87.0

Tableau 1 : Différents réglages de la ligne

Réglage	Sol1 (A)	Sol2 (A)	Dh1 (A)	Dv1 (A)	Dh2 (A)	Dv2 (A)	Transmission (%)
Départ	100	100	0	0	0	0	3.4
Optimisation	152.9	182.9	1.08	-0.3	-0.39	-0.21	79.2

Tableau 2 : Point de départ et optimisation de la ligne

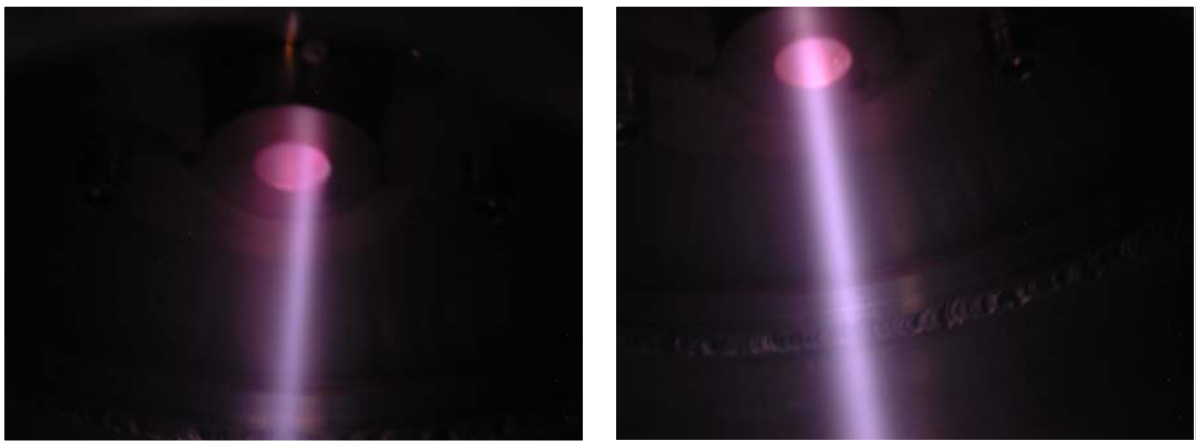

Figure 10 : Faisceau à la sortie du cône avant (gauche) et après optimisation (droite)

5.4 Bilan

Ce premier essai a donc permis de tester avec succès le mode « pilotage automatique ». C'est le mode de fonctionnement le plus évolué. Avoir démontré sa faisabilité, va permettre d'implémenter et de tester rapidement le mode « surveillance ». Ce dernier mettra plus spécifiquement en lumière les éventuels problèmes liés aux interfaces et sorties graphiques. Le mode « simulateur de vol » ne pourra vraisemblablement pas être testé sur l'injecteur SILHI car il imposerait de sérieuses modifications du système de contrôle commande de la machine actuelle.

Plus spécifiquement, ce test a aussi démontré la validité de l'algorithme de réglage sur SILHI et sa supériorité vis-à-vis d'un réglage manuel démontrant l'évidence de l'intérêt d'une machine virtuelle.

6 Conclusion

Une partie de l'accélérateur virtuel a montré son efficacité sur SILHI. Son élaboration n'a été rendue possible que grâce à l'assistance éclairée des spécialistes d'EPICS du CEA (DSM/DAPNIA/SIS) et l'aide de la communauté d'utilisateurs internationale (forum) pour certains problèmes très spécifiques. Ceci met d'ailleurs en lumière l'intérêt d'utiliser un système largement diffusé tel qu'EPICS en termes de délais et de coûts très réduits pour le développement d'un système de commande contrôle.
Le premier test sur l'injecteur SILHI réalisé avec le concours de son équipe a permis de franchir une étape essentielle et a laissé entrevoir plusieurs des avantages que pourrait apporter un accélérateur virtuel totalement opérationnel.

7 Perspectives

- Définition et écriture d'une première base de données du type TraceWin ↔EPICS.
- Implémentation du mode « surveillance ».
- Nouveaux tests sur SILHI.
- Réglage de la transmission du RFQ de IPHI.

Référence :

[1] Integration of the Modeling Program with Accelerator Control Systems in TRISTAN, Proceedings of the 1995 International Conference on Accelerator and Large Experimental Physics Control Systems, Chicago, Illinois, pp. 423-428

[2] USE OF A VIRTUAL ACCELERATOR FOR A DEVELOPMENT OF AN ACCELERATOR CONTROL SYSTEM, Noboru Yamamoto, KEK,High Energy Accelerator Research Organization, Oho,Tsukuba, Ibaraki,JAPAN 305

[3] PAC2003: THE EPICS BASED VIRTUAL ACCELERATOR – CONCEPT AND IMPLEMENTATION, A.Shishlo**, P. Chu, J. Galambos, T. Pelaia, ORNL, Oak Ridge, TN 37830, USA

[4] http://www.aps.anl.gov/epics/

[5] R. Duperrier, N. Pichoff and D. Uriot, CEA Saclay codes review for high intensities linacs computations, in Proceedings of ICCS-2002, Amsterdam, edited by P.M.A. Sloot, Springer Verlag, Berlin, 2002, p. 411-418.

[6] http://www.winehq.com

[7] Phys. Rev. ST Accel. Beam 3, 124201 (200): TOUTATIS: A radio frequency quadrupole code. Romuald Duperrier; CEA-Saclay, 91191 Gif sur Yvette Cedex, France.

[8] Etude et mesure de l'alignement du faisceau de SILHI, D. Uriot, *CEA/DSM/DAPNIA/SACM/LEDA,* 20 février 2005