15.3 運転用中央計算機とネットワーク

入射器の制御系は、1993 年 10 月から Mini Computer の保守打ち切りを受けて、また、機能増強を目指して、制御系の上位 ネットワーク と Mini Computer 部分の更新が行なわれた。周辺技術の進歩に合わせ、その方針としては、ハードウェア及びソフトウェアに国際標準と業界標準を採用、を挙げた。[1, 2]

具体的には Unix System、VME System、TCP/IP Network の採用であり、これらの標準の採用によってシステムの将来の継続性と拡張性が高くなった。

KEKB での運転における制御系に対する要求は格段に複雑になっているが、基本的にはこれまでに構築した制御系の方針で進めていくことができると思われる。その上にたって、これまでの運転の経験を元に、さらに次のような基準を設けている。(1) 対象指向のシステム構成とモジュール化。(2) 高信頼化。(3) リングの制御系との接続。

以下のセクションにおいてはこれらの基準について具体的に説明をする。

15.3.1 対象指向の設計

対象指向 (Object Oriented) の設計とは、これまで制御系の “機能” を中心に考えてシステムの設計を行っていたが、各々の機能は対象に付随したものであると捕え直して、設計を行うことである。例えば、同じ機能であっても、対象が異なれば実装が異なるので、より自然な形態であると言える。実装が同じになるのなら、より上位の対象領域 (クラス) の機能であると捕えればよい。

対象領域を明確に捕えることによって、よりわかりやすい制御サービスのモジュール化が可能になり、また、ソフトウェアとハードウェアの対応もつけやすくなる。標準システムの採用によってシステムの拡張性が高まったが、拡張を行う場合に必要になる制御サービスの分散を行う場合にも、対象領域の認識とモジュール化は重要である。

KEKB 制御においては大きく分けて 2 つの対象領域の考え方がある。一つは制御対象の機器群を対象領域の階層に対応させる捕え方と、もう一つは制御サービス群を対象領域の階層に対応させる捕え方である。それぞれ必要なものであり、また、お互いに利用させあうことになる。いずれにしても、公開する必要のない情報を隠蔽し、公開するものはその Interface を明確にすることによって、サービスの確実性と再利用性が高まることになり、ひいては確実で効率的な運転を支援することになる。

制御対象機器による分類

加速器の機器、例えば、クライストロンや電磁石を対象とする、階層化された対象である。クライストロンにはメインブースタ、サブブースタ、高出力クライストロンがあり、それぞれサブシステムやモニタ系も持つ。電磁石は電磁石自体の種類だけでなく現在のところコントローラの種類も多種類に渡る。それらの機器の操作情報を、外部には不要なものを隠しながら、各機器に適した方法でやり取りすることは重要なことである。

KEKB の制御においては各グループの協力により、クライストロンやビームモニタ系、トリガ系の制御が、中央計算機とは別の取りまとめ計算機を置いて、行われることになっており、計算機系としても対象領域の明確な切り分けを押し進めた形になっている。これまでは、機能の追加のために新しい計算機が導入されることが多かったが、制御対象によって計算機を導入することによって、論理的、物理的にわかりやすい構成になってくる。

これまで、機器の制御については様々な情報が最終的なアプリケーションプログラムの中にも含まれ、それらの保守性に影響を与えてきた。しかし、既に対象指向の設計を行って定めた Interface を使っているアプリケーションプログラムについては、その変更はほとんど必要ないはずであり、また、内部プログラムも計算機が分離されることによる変更はあまりない。

制御サービスによる分類

機器データベース、履歴、シミュレーションなどはこれまで対象領域として明確に捕えられてこなかったが、アプリケーションプログラムが増えるに従って、再利用の必要性が高まると思われる。データベースについては既に実装があり、ほとんどのプログラムで分散利用しているが、機能追加とリレーショナルデータベースへの対応のために、再実装の必要があると思われる。シミュレーションなどの他のサービスについても重要度が増して来ている。

サービスの利用

加速器を見る場合、

1. 機器担当者が各加速器機器の内部パラメータを監視するときのような、技術的視点、
2. 運転担当者がビームを調整するときのような運転時の視点、

3. 加速器の改善を行うためにビームの挙動を調査するときのような科学的視点、

の 3 つがあり、それぞれが制御系を通して得られる必要がある。

それらは主にアプリケーションプログラムで、サービスをどのように利用するかによって区別されるが、サービスの実装を行う場合にもできるだけ無駄が起こらないよう注意している。

対象指向のプログラミング

現在のところ、一部を除いて環境が整っていないので、対象指向プログラミングは利用していない。特に対象指向プログラミング言語からは分散環境の利用が有利にならないために、その利用を遅らせてきた。現在はほとんど全ての内部プログラムは C 言語で書かれている。

しかし、分散環境の標準がやっと国際標準化され、実装も手にはいるようになって来たので、少なくとも上位層において C++ 言語を使った実装を行うことを予定している。このことは KEKB の Ring との インターフェイスにおいても重要になると思われる。

15.3.2 高信頼化

1993 年からの新しい制御系の運用によって、いくつかの新しいサービスが始まったが、その経験によって、制御系の信頼性の検討が必要なことがわかってきた。更新時には経済的な問題もあって、以前の制御系と同等の信頼性を考慮した。標準を採用したことと技術の進歩によって個々の機器の信頼性は向上しているので、全体の信頼性も向上していると考えていたのだが、制御系が複雑化しているので、さらに高い信頼性が必要になっていたのである。

すぐに障害箇所の置き換えができるような体制は進めていたが、やはり、自動的に障害復帰ができるようになっていないと運転停止の時間は長くなる場合も多い。KEKB の制御の高信頼化のためにハードウェア及びソフトウェアの両面から 2 重化や冗長性の検討実装を進めており、以下にいくつか重要なものを挙げる。

運転用中央計算機

1993 年の更新時には経済的な理由で運転用中央計算機が 1 台しか導入されなかった。そのためのハードウェアの障害はほとんどなかったものの、ソフトウェアの障害が起こった場合にそのソフトウェアが修正再起動されない限り対応するサービスが利用できなくなっていた。ソフトウェアの担当者がいつもすぐに対応できるとは限らないし、完全にバグのないソフトウェアは有りえないので、複数の運転用計算機の導入を始めている。

現在、機種の異なる 2 台の計算機が並列運転を行っているが、KEKB の運転には同機種の 2 台の計算機とすぐに置き換えが可能な開発用計算機が用意される。機種は、DEC 社の Alpha 計算機で、運転用の 1 台と開発用の 1 台は現在最も高速な CPU を搭載している。(なお古い機種も、直接制御機能にかかわらず、速度を必要としないサービスには当面使い続ける。)

2 台の運転用計算機を密結合にすることも可能であったが、障害の波及を最小限に抑えるために切り離した。将来 1 台の計算機の CPU の数を増やすことは可能である。

アプリケーションプログラムは 2 台の運転用計算機のサービスを、明示的にまたは自動的に切り替えることによって、サービスの中断を最小限にすることができる。開発用計算機ではすぐに置き換えが可能となるように、重要な運転用サービスは常に起動しておくことを考えている。

表 1: 運転用及び開発用中央計算機 (カッコ内は最大値)

	運転用 A	運転用 B	開発用
CPU	Alpha-21161	Alpha-21604	Alpha-21161
CPU 数	1 (4)	1	1 (4)
性能 (SPECint)	337	120	337
メモリ (MB)	512 (2048)	192 (256)	128 (2048)
OS	Digital Unix	Digital Unix	Digital Unix

ファイルサービス

現在の制御系はさまざまな部分で計算機のファイルサービスに依存している。ところが、記憶媒体であるハードディスクは計算機の構成要素の中でも最も信頼性の低いものである。そこで、いくつかの対策を検討しており、有用性の高いと思われるものから以下に挙げる。

まず、RAID Disk の導入で、複数のディスクの内の 1 台に故障が起こっても、他には影響を与えないようになる。ディスク装置で閉じた機能なので、この範囲の障害であれば、最も有効で、ソフトウェアの対応は必要なく、計算機機種依存性もない。

RAID Disk の信頼性が高くても、ファイルサービスをしている計算機が止まってしまえば、他の計算機はファイルサービスを受けることができず、制御系の主要部分が影響を受ける。そこで、1 台の RAID Disk を

2 台の計算機に接続し、どちらか止まっていない計算機からサービスをさせるように設定する。この設定はソフトウェア 的に機種依存性があるが、KEKB では同機種の計算機の運用になるので、問題はない。

RAID Disk 自体が複雑化しているために故障する可能性はあり、実際既に故障を経験した。そこで、制御に直接必要なファイルサービスは、2 台の運転用計算機に独立に持たせることにした。ファイルの更新は一方で行ない、他方は自動的に更新させるように設定する。

さらにディスクのテープバックアップを自動的に行うためのソフトウェアの導入を予定している。ハードディスクシステムの容量が増大するとともに、その利用方法が複雑になり、最終的な復旧に役立つはずのテープバックアップも操作の誤りにより、機能していないこともある。システム全体が複雑化しているので、できるだけ自動化を行う。(しかし、テープバックアップは重要なので定期的な手動のバックアップも行う。)

ネットワーク形態

これまで計算機ネットワークとしては通信プロトコルとして TCP/IP を使用することだけを規定してきたが、媒体としては国際標準になっている 10Base5、10Base2、10BaseT、10BaseFL 及び FDDI と、様々なものを導入している。しかし、今後は障害回復の容易さから、できるだけネットワークのトポロジをスター型とすることとし、FDDI、10BaseT、10BaseFL、100BaseTX、100BaseFX の利用に切り替えを進めている。幸いにしてこれまで運転中の大きなネットワーク障害は起こっていないが、バス型やリング型であると、障害箇所の特定が著しく困難になることを経験している。

また、制御機器グループ別に独立のスター型ネットワークを用意し、一つの系統に障害が発生しても影響が他に及ばないようにすると同時に、トラフィックの分離も行う。一つのグループネットワークの内部でも、リピータの代わりにスイッチングハブやブリッジを使用して、トラフィックの局所的な飽和による効率の低下を防ぐよう、ネットワークを構成する。

FDDI は中央計算機間及び各制御機器グループネットワークのスイッチングハブを繋ぐ。中央制御室と各制御機器の間は全て光ネットワーク 10BaseFL で結び、クライストロンモジュレータのノイズの影響によるエラーを最小限に抑える。さらにこれまで使用していたリング型の現場機器ネットワーク Loop-2、Loop-3 を KEKB の運転開始までに、10BaseFL によって置き換える。

ネットワークの冗長性

計算機ネットワークは障害が発生すると 2 つに分断されてしまう。スター型のトポロジの採用により、障害発見は早くなるが、制御機能の一時停止は避けられない。そこで、経済的に許す限りネットワークに冗長性を持たせる必要がある。

FDDI はその規格の中に 2 重化が含まれており、1 箇所の障害が全体に影響を与えることはない。制御系の主要部分は FDDI で結ばれるので、比較的信頼性は高い。

10BaseFL などの Ethernet を元にしたネットワークでも接続を行う際に冗長性を持たせる技術が開発されてきている。例えば、ブリッジ間のスパニングツリープロトコルを利用する方法、リピータ間の 2 重接続を行う方法、などがある。

スター型のネットワークであることとさらに経済性を考慮すると、中央側にスイッチングハブと、そのハブと論理的に同じネットワークに接続されたハブから、2 重に光ファイバを現場に張り、現場側で冗長性を持った光トランシーバを使って受けるのが最も好ましいことがわかった。

このような接続を各グループネットワークについてそれぞれ 15 箇所の中継点で行うことにした。末端の接続には単純な接続を用いるが、この場所での障害は全体に影響を及ぼすことはない。

VME システムの Diskless 化

先にファイルサービスについて述べたが、同じ理由で現場のコントローラのハードディスクの障害も避けたいものである。そこで、信頼性の高いファイルサーバを利用して、現場の VME コントローラを全て Diskless 化した。

VME システムは、bootp、tftp、NFS といったプロトコルでファイルサーバとネットワークに依存して機能することになる。1 台のファイルサーバが停止していても自動的に他方のファイルサーバを選ぶので、問題は少ない。また、bootp のサービスを利用することによって、複数の VME システムに同じ IP アドレスを与えてしまうといった誤りを防ぐこともできる。マイクロ波グループの VXI システムも同様の機能を選択し Diskless で導入されることになっている。

ソフトウェアの信頼性

これまで、ほとんどの制御プログラムは古い C で書かれてきたが、最近、環境が整ってきたので新しいプ

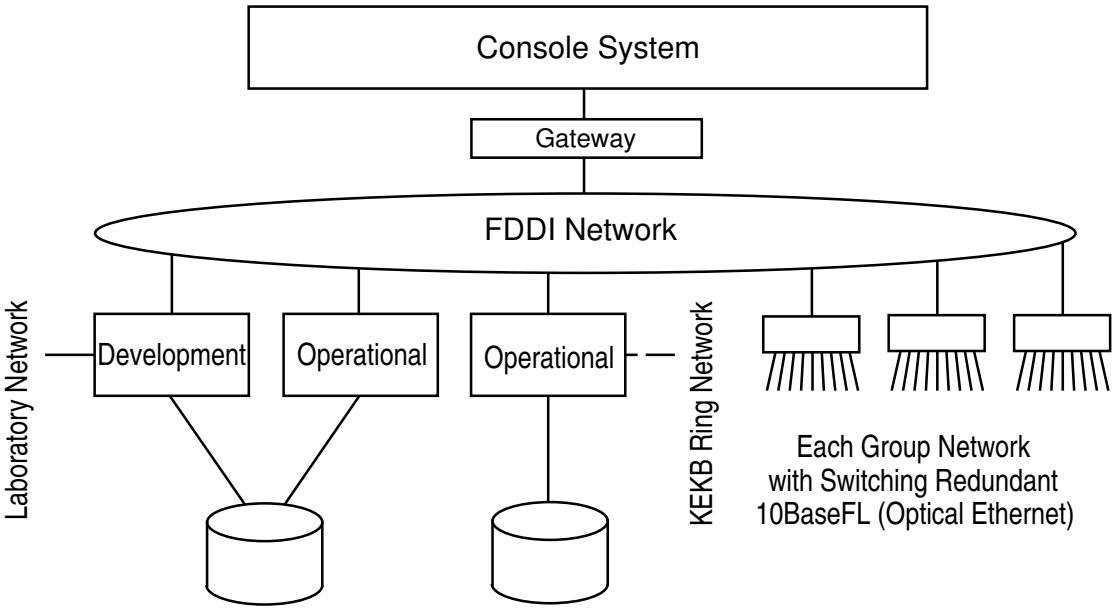

図 1: 運転用中央計算機とネットワーク。2 台の運転用計算機が並列運用され、また、2 重化されたファイルサービスも行なう。負荷を分散させるためにそれぞれの機器グループには独立で冗長性をもった光スイッチングネットワークが用意される。主要機器間は FDDI ネットワークで結ばれる。

ログラムは ANSI C で書き始めている。また、古いプログラムも変換することを検討している。この書き換えによって、少なくとも引き数の誤りや、タイプミスなどアルゴリズムに関係ない問題をかなり減らすことができる。現在、対象指向のソフトウェアデザインによって信頼性が高まっていると思われるが、さらに将来、先に述べた対象指向プログラミングで、誤りを減らせることと思われる。

プログラムから誤りを完全になくすことはおそらく不可能だが、プログラムが止まってしまった際に、記録を残したあとで、少なくとも再起動を行う機構をいくつか準備している。この機能はプログラムの書き方にも制限を与えるので、新しいプログラムはその制限に従うよう書いている。

15.3.3 KEKB のリングとの協調

1995 年に KEKB リングの制御においては EPICS (Experimental Physics and Industrial Control System) [3] を採用することが決まった。最近の加速器制御のソフトウェア共同開発 の一つの成果である EPICS を取り入れることは、自然なことだと思われる。

一方、入射器とリングの制御の協調は効率的な KEKB の物理実験のためには不可欠だと思われる。特にリングへの入射路のビーム通過率がビーム入射時間の短縮に必要である。そのための方策を検討している。[4]

いくつかの方法を検討した結果、当面 EPICS の標準通信プロトコルである Channel Access を使った入射器の制御サービスを作ることが、最も容易で効果も大きいことがわかった。すでに用意されている入射器の制御サービスのインターフェイスを変換することでそのようなサーバを作ることができる。

また、条件が整えば、EPICS に限らない上位インターフェイスとして定義された Cdev を利用することによって、将来の入射器自体の制御にも有用となる。Cdev は上に書いた対象指向プログラミングを基本とするので、比較検討のためにも近い将来に実装を試みる必要があると思われる。

参考文献 (15.3)

[1] K. Furukawa *et al*, Proc. Int'l Conf. on Accelerator and Large Experimental Physics Control Systems, Tsukuba Japan, 1991, p.89.

[2] N. Kamikubota *et al*, to be published in Proc. Int'l Conf. on Accelerator and Large Experimental Physics Control Systems, Chicago, USA., 1995.

[3] L. Dalesio *et al*, Nucl. Instr. and Meth. A **352**(1994)179.

[4] K. Furukawa *et al*, to be published in Proc. Int'l Conf. on Accelerator and Large Experimental Physics Control Systems, Chicago, USA., 1995.