AN ARCHIVE SYSTEM FOR PETRA

Winfried Schütte, DESY, GERMANY

Abstract

A new, PC based control system is being introduced for the high energy particle accelerator PETRA. All relevant data for the operation and control of the accelerator is send with standardized IPX calls. Any subset of those data can be stored with an archive system at a rate of up to one Hertz. Archivation can be done at certain time intervals and/or on a specified relative/absolute value change and/or for values within a specific range. Archivation can be made dependent on the state of the accelerator like injection and of the accelerated particle type. An example would be to store the values of the proton injection orbit, which differed from the last stored injection value by at least 0.5 mm, or to store the vacuum values which changed by at least 50%. The definition of the archivation is completely given by entries in a database. Archiving can be done in quasi infinite tables or on a first in first out basis in tables of a defined depth. Both the definition of archiving and the archiving itself is done with Microsoft Access 2.0® as a database. We are starting to touch the limits and contemplate on a client server database on Windows NT. We also started working on a WWW access to the PETRA archive (intranet archive access).

1 INTRODUCTION

PETRA is a large lepton and proton accelerator at DESY in Hamburg, Germany. It was commissioned 1978 with a NORD computer based control system. These computers are no longer built. After getting the large HERA project at DESY working a new control system for PETRA was started and operated in parallel to the NORD system. It runs successfully as the only control system since May [1]. The control system should be based on powerful and modular industry standard products with a fast design cycle. It was decided to use a LAN of Windows 3.1 PCs with a Novell 3.1x file server and IPX as the only communication protocol[1]. All the applications are written in Visual Basic 3.0. Even the device servers are true Windows applications. For the IPX communication a visual basic extension (VBX) was written [2]. It allows transmission of integer and single arrays, of strings, of two special formats for hardware access and of a cycle message format. Information that is considered useful by many users of the PETRA subnet is broadcasted by device server PCs. This means that the machine status is available on and only on the subnet to anyone. No special request or registration on the sending PC is necessary. A special message is the cycle message. It contains the most important information of PETRA, and the time and a unique number, the cycle number. It is sent at a rate of one to two Hertz. Most broadcasts are sent as a direct response to the cycle message. Changing the rate of the cycle message changes the network traffic and the reaction time of the entire system.

2 DEFINING THE ARCHIVATION

As part of this control system an archiving system was built[2]. Any defined subset of the broadcasted data can be stored under defined conditions to Access databases by as many as necessary servers. The definitions are kept in an Access database. Following examples should be realisable by the archivation system just by editing database tables: Storage of the DC beam current every minute. Storage of the last 1000 vacuum pressures that changed by at least 50%. Storage of the values of the proton injection orbit, which differed from the last stored corresponding value by at least half a millimeter. Each of those examples we call an **archivation**. The set of all archivations we call the PETRA archivation. Each archivation is represented as a record in an archivation table that has enforced relations to other tables to precisely define the details. Every archivation has the following information: a unique archivation name, a reference to the archivation server, the data definition, the particle type (optional), the machine status (optional), a history filter reference (optional), a data taking interval (optional), the maximum number of stored values of a particular type (optional), a reference to the data output location and auxiliary information (partly optional). Let us first look how the data definition of the archivation is realised with the help of the detailing tables.

2.1 What to archive and how to get it?

Any information on the PETRA subnet is uniquely defined by an id, a format type and an index. The id is a character string sent with all IPX messages. An example

[1] This means there is no IP on the Net. This might change if we introduce NT. It contains following important features: Flat 32 Bit address space. Which should releave us from the noorious „Out of Memory“ problems of our large VB 3 programs. Then it allows for powerful databases and WWW servers. And finally there are some selected programs like ramp coordination or archivation, which could benefit enormously from multithreading. Also we started working with Novell 4.1 for HERA.

[2] A slightly out of date but more detailed description can be found in [3].

could be „PX_All“ for the position in horizontal direction for all beam position monitors. The format could be a string of floats; one for each measured position. The index could be the i'th element. For example the 26'th element contains the position “SL 141“[3]. The information can be most naturally administered in a master table of IPX ids and a detail table with the interpretation of all indexes for each format.

Additions, removal and significant change in the equipment lead to a different way of using the IPX format. The consequences are different table entries. These tables only reflect the format at the time they are used. They are not history aware. The archiving system should be able to deal with last years' data too. In other words: One wants a backward compatible model for the archive data representation.

The ids are not only related to data belonging together, but also to hardware implementation. For example there are 277 getter pumps in PETRA, which are treated by two different device servers, both having different ids for the same type of values. One gets 93 vacuum values for the north half and 164 for the South half of the ring. The archiving system should be able to treat all those values simply as PETRA vacuum getter pump pressures. In other words: One wants the ability to have a logical archive data representation.

Both arguments lead us to use a separate internal and external model for the data. All the getter pump pressures have the internal data name „GP“. They are all of data type single and have the unit mBar. In a detail table there is a list of all components belonging to the data name „GP“ and an external key into the list representing the data sent by the IPX. If a pump is removed one only has to remove the foreign key entry and the corresponding record in the IPX id detail table.[4] A disadvantage is that this has to be done now consistently at two different places. The archivation process involves selection and regrouping of data.

In the internal data model there are two types of data scalar and vector data. The DC beam current is scalar. At any time there is only one measurement of it. The 277 getter vacuum pressures are vector data. For data marked as scalar data in the master table, the cor responding index of the component will not be stored.

2.2 When to Store the Information?

It is not desirable to archive or even look at all the specified information each time it is sent into the net. For example the getter pump pressures are only interesting to look at, at a rate of every ten seconds. This is accomplished by an entry in the master table of the IPX ids. Such a strategy saves data processing power at an early stage. Next some information is only interesting, if there is the right kind of particle in the machine. For example a fine grained archivation of the lifetime is necessary only, when there are leptons in the ring[5]. For the particle type one has the choice of: any, protons, leptons, electrons or positrons in the machine. Next some information is only interesting, when the machine is in the right state. For example a proton injection orbit requires the machine to be in injection mode with current. For the machine state one has the choice between any or only injection, ejection, ramp up, ramp down. Each choice can be supplemented by „with current“. It is simple to implement more choices, but it requires to change code in the archivation server program.

Now lets assume the particle in the machine is right and the machine is in the correct state. Still one does not want to store all values at a one Hertz rate. The simplest way of reduction is to introduce fixed intervals of storage. For example archive the machine current every sixty seconds. Allowed is everything between one and 32767s (a little more than nine hours) and no fixed interval at all. The data are tried to be taken at the same time for every fixed interval of the same length. Data taking with a one minute interval is always due at the exact minute. Data taking with a twenty minute interval is due at the exact hour, the exact hour plus twenty minutes and the exact hour plus forty minutes. If data taking is late relative to the wanted time, then the value is stored immediately. The disadvantage is an extra load imbalance on the server and net. But standard target times make it easy, efficient and reasonable to correlate different channels like current, lifetime and energy, since the data taking times should be exactly equal for equal time intervals.

The fixed interval data taking has two potential problems. First one is blind between the intervals. Second some channels, as the particle type, stay constant for long times. Choosing the data taking interval leads to a trade-off between redundant data and out of date data. Therefore an alternative means of choice is provided: the **immediate history filter**: Any data can be specified for archivation when the value changes. This is ideally suited to the particle type channel. Numeric data can be specified for storage, when the value changes more than a chosen amount (good for „linear channels“) or the value changes more than a chosen percentage relative to the current value (good for „logarithmic channels“). A typical linear channel is the beam position and a typical logarithmic one the vacuum pressure [4]. Also the range of tracked values can be restricted with optional mini-

[3] South Left at 141m

[4] In this system it is also possible to use the same IPX data in multiple ways. Like „GP“ and maybe „GPhighSynch“ for the data of the pumps with high synchrotron radiation.This gives flexibility but it costs explicit effort to keep the mappings consistent for each usage. Should this prove to be extensively needed, we would add another indirection in our data model.

[5] Precisely: PETRA is used as a storage ring for HASYLAB synchrotron radiation experiments.

mum and optional maximum values. The first time a value exceeds the limits, it is stored.[6] No more values will then be stored until the first one that is again within its limits. **Range restriction** allows to ignore the data of a DC current archivation with 10% resolution, when there are no particles in the machine. The zero current noise of the DC transformer is usually not interesting. Range restriction is also a low level implemention of limiting data to their physical range. Finally the filter is implemented in such a way, that one can define a default and add for some special channels different values. This allows for example to cope with noisy vacuum pumps, but still be able to refer to an „UHV50%“[7] filter as a single entity in the definition of the archivation

Another problem to solve for history filtered data is signalling that a particular archivation stopped. This might be due to the breakdown of the corresponding device server or to a pause of the archivation server itself. Here the convention is to fill a Null[8] value right after the last known valid time. This is done in the following situations. First an IPX message of a special id does not occur within the specified time-out. Second the archivation server is shut down in an orderly fashion. At the start up of the archivation server, it is checked that all entries end with a Null. If it is necessary they are added with a timestamp immediately following the last valid one. Such important operations as shut down of an archive server and such important incidents as a lack of IPX messages are logged. But it is sufficient to look at the data to get a valid picture.

2.3 Where and how to store the information?

The way this decision is dealt with is heavily influenced by using Access and not using direct files or a full grown client server database. Here we use a table for each archivation. The table name is by definition equal to the archivation name. The only choice left is how to divide the tables over different databases. This allows to control the size of different database files and to keep some logic in the organisation. The table format itself is simply timestamp, data index (only for vector data), value. The timestamp is the time at data taking as given by the cycle message. It is coded in seconds since a fixed time. As opposed to the famous UNIX timestamp it does not take account of daylight savings times. This means loss of one hour of archivation data each year, but makes it for everybody a lot easier to use. The table per archivation approach is viable, because all data are of the same atomic data type (Integer, Single or Float). In Access very long tables do not perform well.

Any database that is corrupted[9] gets on restart automatically repaired. Any non existent database files or tables will automatically be created including the indices. The counting of the archivation channel entries is done with a special table at archivation start-up. The table is computed using the data themselves. Any first reference of an element will be looked up in the table. The following references use counter arrays in main memory. Since all archivations use exclusive write privileges on a table this leads to efficiency and consistency. The system dynamically adapts to an increase in the specified table depth but not to a corresponding decrease. The archivation server appends and edits records but never deletes any.

3 PERFORMANCE AND OUTLOOK

In 1995 we collected roughly 94 MByte of data in 41 archivations distributed over nine database files. So far only one archivation server PC is sufficient.[10] 92% of all messages are handled within one second, 99% within five seconds and 99.9% within 9 seconds. The archivation server runs reliably at a many weeks' basis. It is shut down for doing a manual backup roughly once a week. As long as the archivation server is running the database files are open and cannot be copied under Novell.

The problems are scalability, net traffic and accessibility. Scalability is an issue for HERA with its much larger size. The archivation leads to a substantial portion of the net traffic, since a fair fraction of the net data is written to the PETRA file server. Both issues could be solved with a client server database. Right now we explore the Microsoft SQL Server for that purpose. Accessibility is the next problem. On our nets this is solved by a Visual Basic tool. Outside we are starting to offer an on site WWW interface (intranet archive access). The major advantage: there is no work for us on the client side.

REFERENCES

[1] F. Peters et al. „The PETRA Control System“, to be published

[2] M. Peters, to be published

[3] W. Schütte, „The PETRA Archive System“, in Proc. of „The 1995 International Conference on Accelerator and Large Experimental Physics Control Systems“, Chicago, Ill., 1995.

[4] The necessity for a relative filter for vacuum data was pointed out to me by Kay Rehlich of DESY.

[6] Though one is by definition not interested in the value itself, it is necessary to store it (or any other non valid value) to flag that the previously stored value is no longer valid.

[7] An Ultra High Vacuum filter that requires a relative change of values of 50% (default) and allows a reasonable pressure range (10**-15 to 10**-3mBar).

[8] Special database value with the meaning of undefined, which is by design different from any possible valid entry.

[9] Reasons are hard archive device server failures (like reboot, taking the fileserver down, loss of wall plug and ups power or major ethernet problems) in the middle of a transaction.

[10] Intel 486, 66 MHz, 8MB of RAM