Software Project for SPring-8 Linac Control

H. Yoshikawa, Y. Itoh, H. Sakaki, A. Kuba, M. Kodera, T. Taniuchi,
K. Tamezane, T. Hori, S. Suzuki, A. Mizuno, K. Yanagida and H. Yokomizo.
JAERI-RIKEN SPring-8 Project Team,
Kamigori-cho, Ako-gun, Hyogo, 678-12 JAPAN

The injector Linac control system for SPring-8 is under development and the first trial will be held soon for testing total performance. We designed this system using object oriented methodology to integrate the operator environment and the database architecture and for easy maintainability. A disadvantage of the object oriented approach is a long lead time for abstraction, modeling and definitions of entities and relations. We represent the status of our software project management and our know-how of rapid application development.

I. Introduction

The control system of the SPring-8 injector Linac has been based on the concept of object oriented design from the beginning of construction. The purpose of this Linac is not only injection to the storage ring but also injection to a new VUV ring SUBARU , the SASE source, a parametric x-ray source and possibly other purposes as they arise.

As a matter of course this software system will be modified continuously as the hardware changes. Thus we must consider not only the functionality of this system, but its maintainability, flexibility and presumed longevity.

II. Applicability of OOP

At the beginning of this software project we discussed which was better - conventional structured or object oriented programming. The former is very mature and many companies have huge resources. If we could write down a complete specification of the software (before the design of hardware devices), some big companies could create the entire software system and it would not be necessary for us to hire programmers. But requirements of our Linac control system are high flexibility and adaptability to future modifications and upgrades without large costs in manpower. That is to say, it would be inevitable that we would have to understand the inner structures of the whole program. The modern belief is that object oriented techniques must be applied as the methodology for management of the complexity of such a large scale program, and thus we chose the OOP paradigm.

One of the disadvantages of OOP is a long lead time. It takes a long time before a schematic of the system for users (operators) appears. We faced the additional problem that our staff was not familiar with the concept. It was predicted that a long time would be required for study of OOP and preparation of a satisfactory class library. However, at the beginning the SPring-8 project schedule was stretched by budget conditions. There was not enough staff and money, but there was time.

The next question asked concerning our use of OOP was whether the scale of the software system was or was not appropriate. When you require the system for control of a small experiment, OOP is too heavy. However, you can still use ready-made class libraries for the GUI and windows.

The number of channels required for control of the Linac are as follows.

The number of signals						
DI	DO	AI	AO	RS232c	GPIB	step motor
1605	749	440	139	3	25	108

The scale is large enough, but the number of kinds of devices is small. For instance, several kinds of magnet power supply are similar to each other. Spatial replication of device structures is the rule, e.g. Q-magnets, ion pumps and beam monitors. The configuration of the Linac is shown in figure 1.

The characteristics fit well with the class hierarchies and inheritance of OOP. Thus we felt that it would be possible to

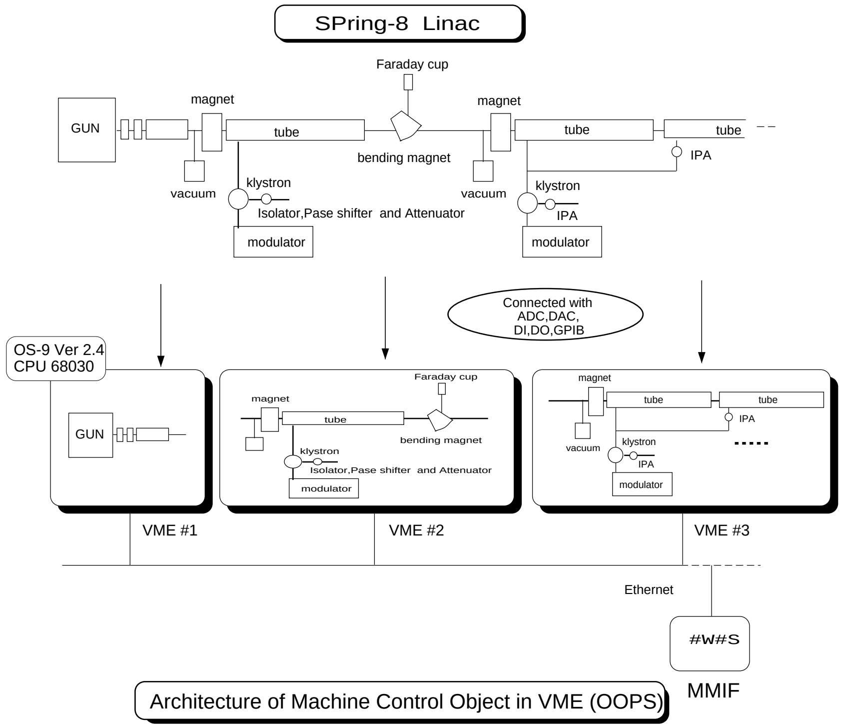

Figure. 1. Configuration of the Linac.

create the required libraries on the appropriate timescale.

III. Progress of the project

Initially, we made an abstraction of the functionality of each atomic device, macro devices and the whole Linac at a software engineering level. The behavior of individual devices is fairly simple. Most of them require only CW operation; and complex feedback (feed-forward for the electron beam) is not needed.

The modulator of the pulsed klystron is an unstable device because it includes discharge elements, but this instability can be hidden from upper level processes, which can consider it as a high voltage power supply.

A. Modeling of the system

We made a common model (MACHINE MODEL) of the status conditions of all devices, as shown in figure 2. We proposed to match the specification of device behavior to this model. The model is event-driven, and a message from some other object is the trigger for a status change. Messages are standardized for all devices inside SCCs (SPring-8 Linac Control Commands). When a device in STANDBY status receives a “GO RUN" SCC, the device acts to change to RUN status. Individual differences in this behavior are encapsulated in the elemental control processes. A macro device, which consists of several atomic devices, operates similarly and forwards SCCs to its internal element devices.

B. Bottom up approach

Code writing began at the level of device drivers. These were written in C due to its potential portability to advanced CPU boards in the future. VME was chosen as the hardware platform as it is popular world-wide and high performance new boards appear daily. Device drivers in C have advantages not only for new CPU boards, but also for new I/O boards, as shown in figure 3.

The process configuration of this system is shown in figure 4. We worked from the lower levels of device drivers, machine control processes and communication processes to the upper level operation process. The behavior of lower level control

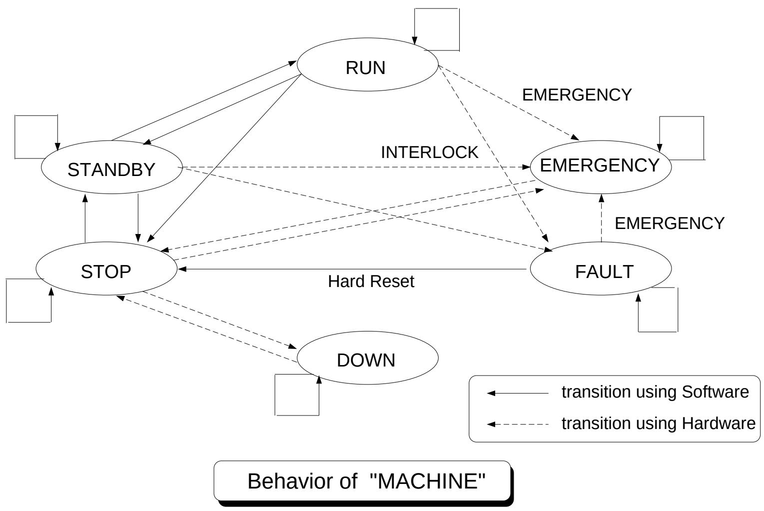

Figure. 2. Status transfer of MACHINE MODEL.

processes depends on hardware design. We required the specification of signal handling to hardware before the software design was fixed.

The communication process is implemented via the original protocol SCD (SPring-8 Control Datagram) [1]. The network attaches directly to the objects, and network transparency is very important. Moreover the communication process is the most important in actual total performance. We isolate the Linac control system segment from other protocols, and only SCC packets are transmitted on it.

The Operation process on EWS (Engineering WorkStations) includes a man-machine interface based on X11 and behavior definitions of device operations. The development environment on EWS is now very sophisticated and many kinds of CASE tools and GUI builders exist. We consider that processes at this level must be modifiable by accelerator specialists and operators to mature.

IV. Framework of control process

The fundamental structure of MACHINE MODEL is shown in figure 5. Receiving of SCCs and interrupts from hardware are recognized by the process as signals. The process checks status, searches the lookup table and jumps to the selected procedure. All input and output except hardware interrupts are dealt with as SCCs. This framework is common to all

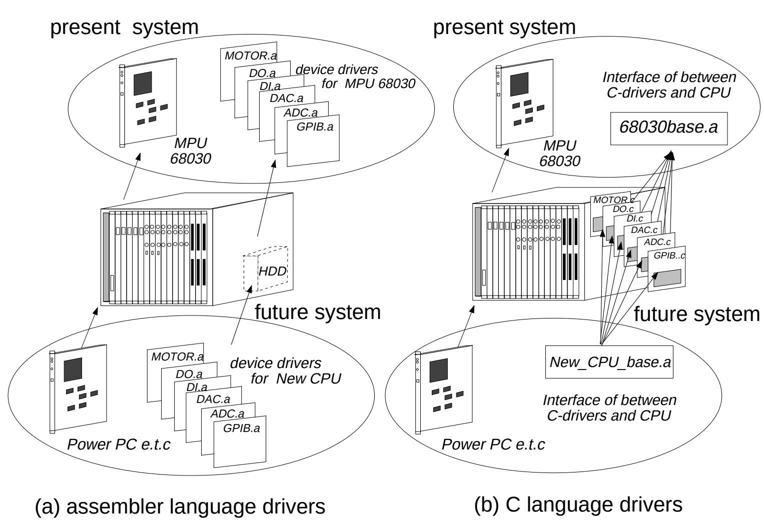

Figure. 3. Differences between assembler drivers and C drivers.

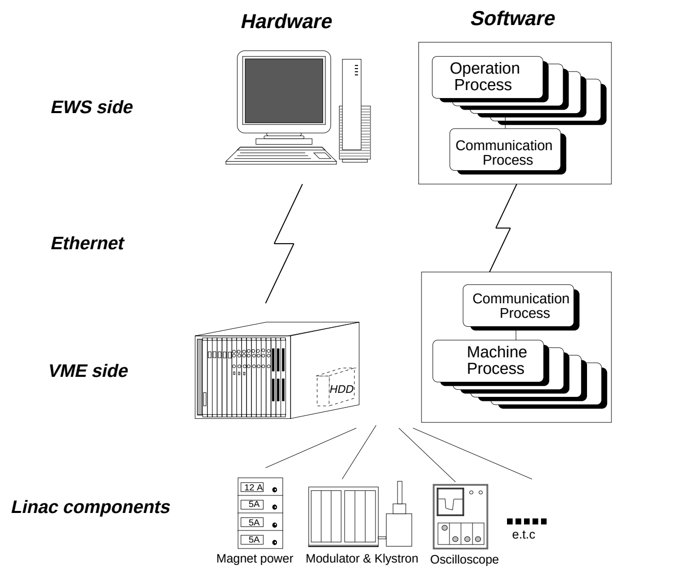

Figure. 4. Hierarchy of processes

processes and is very simple. We represent the framework and the definition of behaviors to programmers, and they write the local procedures. The difficulty in defining behavior is to hide the time sequence of hardware action.

In a most important aspect, we were forced to change the ways of thinking of operators. For example, the modulator of a pulsed klystron has two phases of fault, only High Voltage Down and both High Voltage and Low Voltage Down, in which latter it takes half an hour to warm up the heater of the thyratron and klystron. This is not matched with MACHINE MODEL. We suppressed heat up time in “STOP" status, and the “GO STOP" action includes “heater on". “Warm Up" is defined as a necessary condition of the action to “GO STANDBY" and is independent from identifying of status. Every process on VME and EWS has this same framework. Input to the processes on EWS includes X-window events.

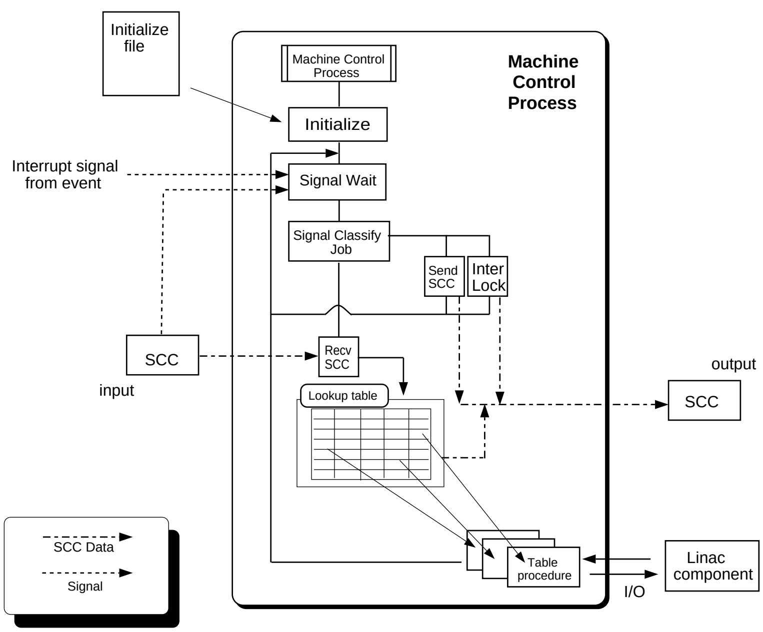

Figure. 5. Framework of control process.

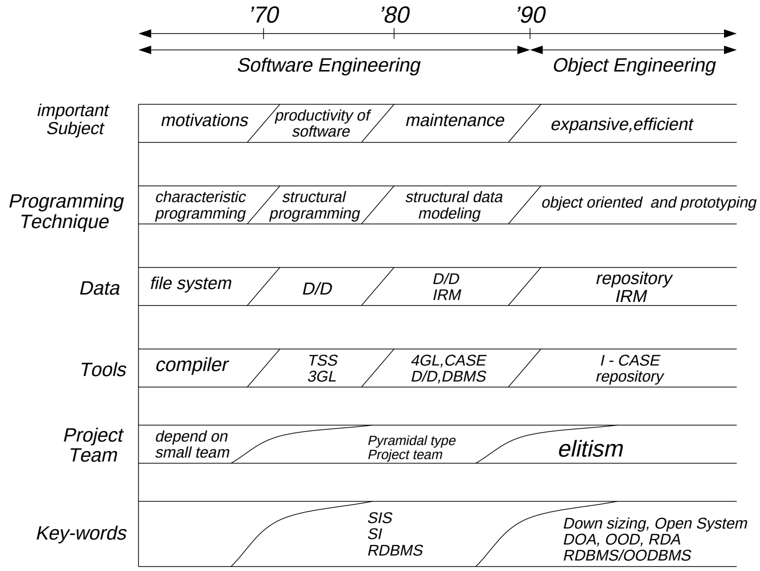

Figure. 6. Trend of software technology.

V. Operation process

Operation processes are located in EWS. This process includes the X-window procedures and operation sequences in the call back functions. We use SoftBench as a CASE tool, and UIM/X as a GUI builder, since it is suitable for combining with SoftBench. A few widgets which we need are written by us, but primarily Motif-based widgets are used.

We prepared three types of man-machine interface program as operator processes. One is a character-based SCC handler for local debugging on VME. It is able to be used through a telnet command from anywhere. The second is a 'primitive' window, which is prepared by the programmer and supports individual device action.

The last is the graphical window, the one to which actual operators interface. On this level, the naked action of MACHINE MODEL is hidden under the X11 environment, and virtual devices appear in front of operators. The virtual devices are objects which have several devices in them, for instance, an energy analyzer (bending magnet, slit and Faraday cup), emittance monitor (Q-magnet, slit and wire grid monitor), or vacuum system (many ion pumps and cold cathode gauges) etc.

The parent process of all operation processes is also one of the graphical operation processes, called LinacMain. LinacMain is the default receiver of SCCs from any VME control process which does not have an operative parent. Finally, this Linac can be operated by only pushing the buttons “STANDBY" and “RUN" in LinacMain.

The control of RF phase is beam-oriented. Previously it had been controlled by a maximum output current condition. However we now detect the phase of the beam wakefield at each accelerating section by a directional coupler near the dummy load, and set the phase of the input power to reverse it. This sequence is written into the operational process “RF system", and when we get fundamental preset values auto-phasing becomes possible.

In the near future many virtual device processes will be written to make the environment of operators more comfortable. Moreover, the modifications for injection into SUBARU, and for beam transport to SASE will be done in the next few years. In this situation the development environment of the software is very important. Use of CASE tools is inevitable, and the advancement of operating system version will be discussed at the level of revision management and efficiency of resource use.

VI. Strategy for construction

The number of persons in the Linac group is eleven, of whom six are working on software. The software staff consists of an overall coordinator, VME programmers, communication process designers, database builders and a handler of visualization tools. By September 1996 all programmers will have become GUI writers.

The first phase of the conceptual design was discussed by only a few members, whose initiative led to understanding of the OOP concept and design policy for others. Each member designs his part, and the team has a discussion to review it. In this phase, it is important to decide not only the functionality but also the structure of the program. Finally we present these prototype codes to programmers, and we require them not to change the structure, except for local functions of individual procedures.

We do not have enough manpower to write all the code of this system, but the job sharing which results can be highly cost-effective; it also helps us to keep a consistent design policy from beginning to end. It took two years to establish the prototype code of MACHINE MODEL. The fundamental structure of MACHINE MODEL has required little change, though several options for adaptation to the hardware design of devices have been added. As a result of OOP such modification is very easy after the plan of modification is chosen.

VII. Conclusion

Our project has had a software engineering history as shown in figure 6. We had to learn the concept of the new methodology, and had to change our way of structured programming. For a software project such as this, to keep a consistent policy and sense of design is very important for purposes of continuity. If you do not have usable class libraries, the project manager must endure a long lead time. But in cases where the scale of the system fits OOP, the final total load is smaller than in conventional structured programming (figure 7). If you can find good class libraries, which your system is able to use, the necessary manpower is drastically reduced.

References

[1] H. Yoshikawa, Y. Itoh, K. Tamezane, Y. Sakaki, M. Kodera, H. Yokomizo. *"Class structure of the Injector Linac control system of SPring-8"* Nucl Inst & Method in Phys. A352, 216-217 (1994)

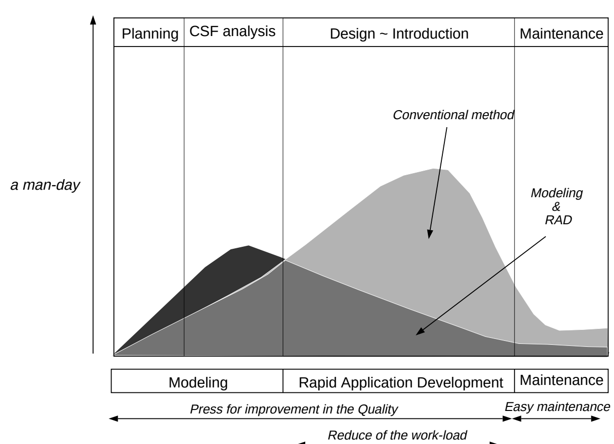

Figure. 7. Manpower at each phase of a project.