

## Introduction - Purpose and Outline

In a project the size of SNS, standardization is essential. Standardization saves engineering cost by reducing the number of different hardware and software modules that require development; and lowers operating cost by lessening training, maintenance and spares inventory. The need to standardize is exacerbated for SNS, where different components and subsystems are being built at different laboratories. Although each laboratory might itself standardize, maximum benefit will be derived for SNS if variety is reduced across the entire project. This will be a difficult exercise, because each laboratory has its own experience and culture. Compromise must be the order of the day if project-wide agreement is to be attained. At the same time, an approach to standardization more rigid than appropriate, necessary or acceptable will be counterproductive. This document tries to strike the right balance.

Standards need to be agreed in four broad areas – system architecture, software tools, interface hardware and development process. In the system area, a standard architecture must be adopted. This includes client-side processor and operating system, and server-side processor, processor bus, and real-time kernel. These choices are tightly coupled. Moreover, rapidly changing technology makes the choice of an appropriate architecture for the year 2005 particularly difficult. The options are discussed in Section I. The recommendation is to “play with” a specific subset, but defer a final decision until shortly before quantity purchases are required.

The choice of the EPICS toolkit as the basis for the SNS control system software goes a long way towards imposing a standard. However EPICS runs on many platforms (that is both the good news and the bad news) and offers a wide variety of mutually exclusive tools for development and operation. This document discusses some software tools that could be adopted for SNS, and recommends a particular suite of these tools. In some cases it is deemed not essential to make a decision at this stage of the project, and the recommendation is simply to follow developments within the EPICS community, and make a decision based upon stated requirements at the appropriate time. The following issues are discussed in Section 2:

- Database development tools
- Display Editor
- Display driver
- Archiver
- Alarm Handler

Standardization of control system architecture and software tools requires the agreement of the controls groups at the participating labs. Standardization of interface hardware requires the agreement of many more players, in particular the diverse and geographically scattered engineering groups who are designing the various systems which will be interfaced to the control system. Consensus here can be expected to be much harder. The following topics are discussed in Section 3:

- PLCs
  - Choice of PLC
  - Interface method
  - Programming methods and tools
  - Local Display philosophy
- Use of Fieldbuses
- Power Supply Interfaces
- Vacuum Equipment Interfaces
- Racks – power, cooling, grounding
- PLDs

Finally, to insure a consistent program development approach and documentation level, it is important to define a process for software, hardware and subsystem development, including appropriate documentation and reviews as well as a consistent development environment to be applied at all the participating laboratories. These issues are discussed in Section 4 under the following headings:

- Application Development Environment
- Software Development Process and Documentation
- Use of a Relational Database

#### Intro to software section

Database  
 Need for an administrator NOW  
 Application Development Environment  
 Development Tools  
 Colour, symbol rules etc

#### Intro to hardware section

Comment on what needs to be standardized and what does not.  
 PLCs  
 Fieldbuses  
 Crates  
 Processors  
 Controllers

#### Intro to development environment

What it is  
 What it includes  
 How the DB fits in  
 Requirements  
 Need to change at a later date (for operations)  
 Distributed CVS  
 File structure (drawings)  
 Procedures  
 Responsibilities

#### Software Development Procedures (per IOC or subsystem as appropriate)

Documentation for each IOC  
 Channel list  
 Modules  
 Engineering screens  
 Cabling  
 Interlocks to and from other systems  
 Sequences  
 Later the database in some visual form  
 Reviews  
 Requirements  
 Preliminary Design  
 Operator manual

#### Control Room Configuration

All this depends upon operational philosophy  
 How many seats?  
 Choice between few seats on tables (BESSY, APS) or traditional consoles  
 Knobs  
 Security issues, firewall  
 Log-in permissions (one operator log in)  
 Printers  
 Analog signals – how many, how displayed  
 Relationship to other activities

Software development  
Hardware lab  
Controls offices  
Computer (server) room