Nuclear Engineering and Design 292 (2015) 46–56

Contents lists available at ScienceDirect

Nuclear Engineering and Design

journal homepage: www.elsevier.com/locate/nucengdes

Software development methodology for computer based I&C systems of prototype fast breeder reactor

M. Manimaran*, A. Shanmugam, P. Parimalam, N. Murali, S.A.V. Satya Murty

Real Time Systems Division, Electronics, Instrumentation & Radiological Safety Group, Indira Gandhi Centre for Atomic Research (IGCAR), Kalpakkam 603102, Tamilnadu, India

HIGHLIGHTS

- Software development methodology adopted for computer based I&C systems of PFBR is detailed.
- Constraints imposed as part of software requirements and coding phase are elaborated.
- Compliance to safety and security requirements are described.
- Usage of CASE (Computer Aided Software Engineering) tools during software design, analysis and testing phase are explained.

ARTICLE INFO

Article history:
Received 15 December 2014
Received in revised form 1 May 2015
Accepted 17 May 2015

Keywords:
Real time computer system
Software development methodology
Nuclear reactor
Safety critical software

ABSTRACT

Prototype Fast Breeder Reactor (PFBR) is sodium cooled reactor which is in the advanced stage of construction in Kalpakkam, India. Versa Module Europa bus based Real Time Computer (RTC) systems are deployed for Instrumentation & Control of PFBR. RTC systems have to perform safety functions within the stipulated time which calls for highly dependable software. Hence, well defined software development methodology is adopted for RTC systems starting from the requirement capture phase till the final validation of the software product. V-model is used for software development. IEC 60880 standard and AERB SG D-25 guideline are followed at each phase of software development. Requirements documents and design documents are prepared as per IEEE standards. Defensive programming strategies are followed for software development using C language. Verification and validation (V&V) of documents and software are carried out at each phase by independent V&V committee. Computer aided software engineering tools are used for software modelling, checking for MISRA C compliance and to carry out static and dynamic analysis. Various software metrics such as cyclomatic complexity, nesting depth and comment to code are checked. Test cases are generated using equivalence class partitioning, boundary value analysis and cause and effect graphing techniques. System integration testing is carried out wherein functional and performance requirements of the system are monitored.

1. Introduction

Real-time system is predominantly used in safety critical or time critical applications and it is popularly defined as a system where the correctness of an output relies not only on the correctness of the logical results, but also on the point in time at which these results are delivered (Sanfridson, 2000). If the timing requirements of the system are not met then the results could be catastrophic called hard real-time systems. Even when the timing requirements are violated, the results are not catastrophic called soft real-time systems. Most of these systems are made up of heterogeneous components including sensors, microprocessors and actuators (Wang, 2008). Real-time systems are extensively employed in automotive, avionics, defense and nuclear applications.

Design of real time system is quite challenging and it is significantly different from non-real time system design. These systems need to satisfy not only the functional requirements but also various performance requirements such as timeliness, availability and fault tolerance (Wang, 2008). Since a failure in real-time systems lead to catastrophic event, these systems are required to be highly dependable. Dependability can be built into the system by following rigorous hardware design procedure and software development methodology. This paper describes about software development methodology.

* Corresponding author. Tel.: +91 44 27480500x22337; fax: +91 44 27480228.
E-mail addresses: maran@igcar.gov.in, ma.manimaran@gmail.com (M. Manimaran).

http://dx.doi.org/10.1016/j.nucengdes.2015.05.014
0029-5493/

The documented collection of policies, processes and procedures used to develop the software product is called software development methodology (SDM). It is basically a framework used to structure, plan and control the process of developing a software product. For the development of dependable software for real-time systems, well documented SDM is required starting from the requirement capture phase till the final validation of the software product. Various guidelines are available to develop software for safety critical systems such as MISRA guideline for automotive, DO-178B for avionics and IEC 60880 for nuclear power plant.

Prototype Fast Breeder Reactor (PFBR) is a 500 MWe, sodium cooled, pool type, mixed oxide fuelled nuclear reactor having two secondary loops (Chetal et al., 2006), and it is in the advanced stage of construction at Kalpakkam, India. Based on safety considerations, I&C systems of nuclear reactors are classified into three classes namely IA (I and C safety class A), IB (I and C safety class B) and IC (I and C safety class C) (AERB SG D-1, 2003). In PFBR, the class IA, IB and IC systems are termed as safety class-1 (SC1), safety class-2 (SC2) and non-nuclear safety (NNS) systems, respectively. SC1 systems play a principal role in achievement or maintenance of nuclear power plant safety. SC2 systems play a complementary role to the SC1 systems. NNS systems play auxiliary or indirect role in achievement of nuclear power plant safety (AERB SG D-25, 2010). The proper operation of SC2 systems may avoid the need to initiate safety actions by SC1 system.

Versa Module Europa (VME) bus based Real Time Computer (RTC) systems are deployed for I&C of SC1, SC2 and NNS systems of PFBR. RTC systems used in PFBR are hard real-time systems. This paper describes the software development methodology adopted for computer based I&C systems of PFBR in a comprehensive manner. Efficacy of the adopted methodology is demonstrated with respect to primary sodium circuit as a case study.

The organization of this paper is as follows: Section 2 describes about the related work. Section 3 explains about the selection of software development life cycle, Section 4 describes the software development methodology adopted for PFBR, Section 5 discusses the software development with respect to primary sodium circuits as a case study and Section 6 provides the conclusion.

2. Related work

Literatures related to software development for digital I&C systems of nuclear power plant (NPP) are detailed. Fukumoto et al. (1998), elaborated the verification and validation (V&V) method and its application to digital safety systems in ABWR nuclear power plant of Kashiwazaki–Kariwa unit No. 6, Japan. Li et al. (2002), presented the development process for the first digital reactor protection system (RPS) used in the 10 MW high-temperature gas-cooled reactor designed and operated in China. They described about the architecture of digital RPS, design of safety software, verification and validation of safety software, functional and performance testing of RPS. Lindner and Wach (2003) detailed about the qualification methodology for software based I&C systems important to safety of NPP. They have also explained about functional testing and analysis of software modules. Yoo et al. (2004) proposed PLC (programmable logic controller) based safety critical control software development technique for Korean nuclear power plant. They have prepared software requirements specifications using formal notation called NuSCR and then transformed it into functional block diagram (FBD), one of the PLC programming language. They have manually refined the generated FBD programs, compiled and tested the software in the PLC environment. Yang et al. (2010), explained about software verification and validation approach using V-model for the digital I&C systems of nuclear power plant in China. Cheng et al. (2014), discussed about the quality assurance of safety critical software of NPP simulator by applying IEC 60880 standards. They have briefed about the software requirements, design and implementation process.

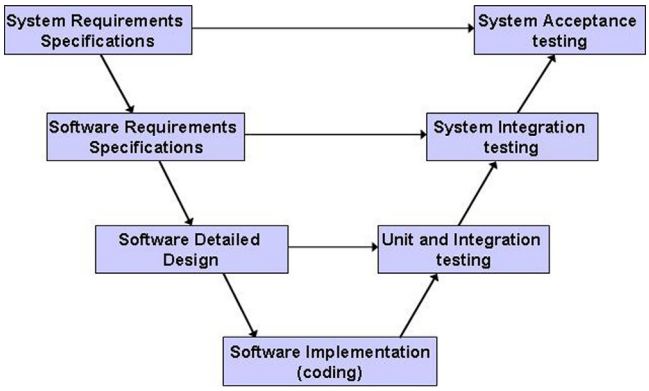

Fig. 1. V-model.

3. Selection of software development life cycle

A software development life cycle (SDLC) is essentially a series of steps, or phases, that provide a model for the development and lifecycle management of software (Davis et al., 1988). The intent of a SDLC process is to produce a software that is cost-effective, efficient and of high quality. There are two different types of SDLC that can be used: waterfall and agile. The major difference between the two is that the waterfall model is more traditional and begins with a well thought out plan and defined set of requirements whereas agile model begins with less stringent guidelines and then makes adjustments as needed throughout the process. Agile development is known for its ability to quickly translate an application that is in development to a full release at nearly any stage, making it well suited for applications that are updated frequently. For nuclear power plant, well thought out model is required.

The strengths of waterfall model are easy to understand, easy to use and it works well when quality is more important than cost. Deliverables created at each phase are frozen. Waterfall model can be used when requirements are very well known and product definition is stable and technology is understood.

A variant of waterfall model that emphasises on verification and validation (V&V) of the product at each stage is called as V-model. This model is used when

- All the requirements are available prior to starting the project.
- Systems require highly reliable software.
- Solution and technology are known.

IEC standard (IEC 60880, 2006) and AERB guideline (AERB SG D-25, 2010) specifies that V&V at each stage of software development is mandatory for nuclear power plant. Hence the obvious choice for software development for computer based I&C systems of nuclear power plant is V-model.

3.1. V-model

Fig. 1 shows the V-model (Pressman, 2010). It describes the sequence of steps to be performed and the results that have to be produced during software development. Instead of moving down in a linear way, the process steps are bent upwards after the coding phase to form the typical V-shape. The V-model demonstrates the relationships between each phase of the development life cycle and its associated phase of testing. The left side of the 'V' represents the decomposition of requirements and creation of system