Real Time Software for the Fusion Experiment WENDELSTEIN 7-X

Heike Laqua*, Helmut Niedermeyer, Jörg Schacht, Anett Spring

Max-Planck-Institut für Plasmaphysik, Teilinstitut Greifswald, Wendelsteinstr. 1, 17491 Greifswald, Germany

Abstract

The super conducting stellarator WENDELSTEIN 7-X will be capable of steady state operation as well as of pulsed operation. All discharge scenarios compatible with these capabilities will be supported by the control system. Each technical component and each diagnostic system will have its own control system, based on a real time computer with the dedicated software described here, permitting autonomous operation for commissioning and testing and coordinated operation during experimental sessions. The system behaviour as far as it is relevant for the experiment, like parameters and algorithms, will be exclusively controlled by complex software objects. By changing references to these objects synchronously in all computers the whole system behaviour can be changed from one cycle to the next. All data required for the construction of the software objects will be stored in one central database and constructed in the control computers well before they are required.

Keywords: Digital control system; WENDELSTEIN 7-X; Real time system; Real time operating system

1. Introduction

The super conducting stellarator WENDELSTEIN 7-X (W7-X) will run pulses of up to 30 minutes duration. All discharge scenarios compatible with these capabilities will be supported by the control system: short pulses with arbitrary intervals, steady state discharges and arbitrary sequences of phases with different characteristics in one discharge. The latter scenario includes short and long pulses. It can substantially reduce the time required for parameter and diagnostic scans and permits short tests of new settings without interfering with the main program. Long discharges with phases of different characteristics are understood as series of short discharge sections called “segments”.

2. The concept of segment control

Segment control is software control by computers.

* Corresponding author. Tel.: +49-3834-88-2756;
Fax: +49-3834-88-2509.
E-mail address: heike.laqua@ipp.mpg.de (H. Laqua).

The behaviour of components is determined by software objects. They contain parameters and algorithms for defining a dynamic behaviour. Corresponding to the hierarchical structure of the control hardware the software objects are structured hierarchically. The description of each discharge segment is realised by one complex distributed software object called “segment description”. Breaks are described by means of idle segments. Segment descriptions are generated and stored in the database by means of a special editor [1]. On request of the session leader all computers involved in the segment control create the partial objects of segment descriptions pertaining to them presumably required in the near future. On command of a central sequence control all partial objects of each segment description are activated synchronously in all computers.

Segment descriptions can be constructed in the database in nearly unlimited number. Control objects can be created out of these at any time, also during a discharge. Thus a running discharge can be continued in an arbitrary manner for an arbitrary duration. Otherwise a short discharge can be controlled as a segment of short duration between two open-ended idle segments.

3. Logical structure of the segment control system

The structure of the control system [2] reflects the hierarchical structure of W7-X which is divided into the basic machine with its technical components, i.e. the electron cyclotron resonance heating (ECRH), and diagnostics. They are called function groups because they generally can consist of several sub-components. Figure 1 shows the parts of the control system essential for segment control. The subsystems have their own control systems and can be operated separately for commissioning and tests. Operational management systems based on PLC technique provide a basic, not segment oriented control. Segment control for each function group is realised by one or several real time computers exchanging information with the operational management and retrieving segment descriptions from a database via Ethernet. Segment control of a passive diagnostic can also be accomplished by a set of data acquisition units (CODA-Stations) [3].

During experimental sessions the local segment controllers are co-ordinated by a central “segment sequence control”. The function groups are then parts of a co-ordinated “project”, e.g. the project W7-X. Prerequisite for co-ordination is a synchronisation of the overall time base provided by the TTE system [4]. The sequence control system is not operated directly but has an interface to a “session leader program” running on another computer [5]. The “session leader program” can also be configured to serve as a “Local operator program” for stand alone operation of the components.

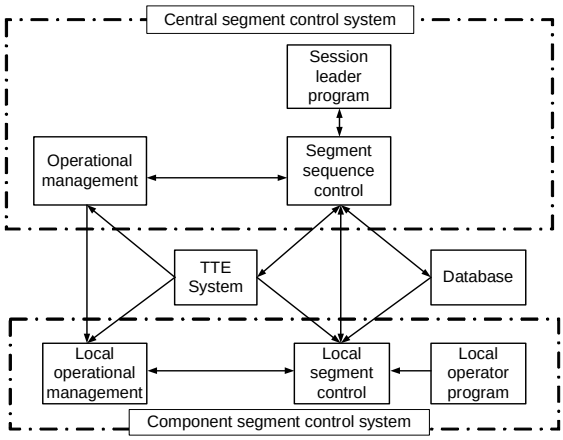

Fig. 1. The structure of the W7-X segment control system as part of the total control system.

4. Real time software structure

Table 1 shows schematically the different types of software loaded into the segment control units. The software is divided into a universal basic software and segment control software. The basic software is partly commercial, partly developed in house in C or C++.

Table 1
Software loaded into a segment control unit. Permanent software is marked with a grey background.

Executable code	Data	
RTOS VxWorks: Variants for different CPU	**System structures**	Basic software
Hardware drivers Same for all units	**Driver initialisation**	
Universal Software: e.g. containers, notification system	**Miscellaneous objects:** e.g. Logger instance	Specific software
W7X Segment Control Software	**Configuration**	
	Segment descriptions (on demand)	

Preprint: Final Version available at http://dx.doi.org/10.1016/j.fusengdes.2006.04.005

The total segment management software is an object oriented, in house development in C++.
All software can be categorised as executable code in the software layer or as data. The segment controllers of W7-X load all executable code including the operating system and drivers when booting and never change or update code while executing. Permanent data structures are integrated in this code.

Drivers and the W7-X specific basic software are integrated into the operating system and provided on a file server. Booting of a segment controller starts with loading of a simple boot system from diskette or flash memory setting a few boot parameters. The running boot system loads the actual system with its extensions using the File Transfer Protocol (FTP). The unit is switched to the new system, creates and initialises the permanent system data structures. The upper three blocks of table 1 are then allocated.

At the end of system initialisation a simple script is carried out specialising the unit for its segment control tasks by setting parameters and choosing the appropriate software modules to be loaded via FTP. The download uses features provided by the real time operating system VxWorks®. The code and the structures allocated so far are permanent.

The predominant part of data structures dedicated to segment control is created from database information via a special mechanism. Only a small part of these data is permanent, the numerous segment descriptions are allocated and deleted on demand.

5 Software modules for segment control

The segment controllers implement modules of the software package W7XSegmentControl. This package contains the software for all tasks of segment control. Among these are provision of a one millisecond time pattern, checking the integrity and technical feasibility of segment descriptions, message distribution and reception as well as distribution of set point waveforms and digital control of plasma processes. Each task is carried out by a software module called "control device". The behaviour of a device is determined by "device segment descriptions", which are structured objects defining the dynamics and parameters. Usually many segment descriptions are available for one device.

5.1. Configuration

The configuration is an aggregation of all control devices implemented in a unit. It contains the name of the function group and the project. All devices are allocated and initialised with a segment program putting the unit in a state as passive as possible according to the unit's segment control tasks. Each device is labelled with a unique role name.
The so called "*basic devices*" with examples listed below are the same for all units and are only configured individually.

- The *real time network system* is an aggregation of Ethernet and TTE based networks with different protocols [6] used for real time data and message transmission mainly by multicast.
- The *message system* is used to send event messages via the networks of the real time network system and to define a reaction on receipt of a message. The reaction can be varied by different segment descriptions.
- *Real time data fields* provide temporary data storage. They are updated via the *real time network system* and provide data exchange between segment controllers similar to a shared memory. Data fields simulate a data bus for time dependent measured data and control values.
- The *timing system* [7] generates the primary cycle of anticipated 1 kHz triggering part of the periodic activities in a unit, especially the synchronised segment switch. Thanks to the TTE system the primary cycles of all units are synchronous. Further cycles with other frequencies or delays are derived from the primary cycle.
- The *object manager* offers a method to update segment descriptions from the database. This method is called by the object monitor and after reception of a message from the session leader program.
- The *segment manager* checks the loaded descriptions for integrity, i.e. all needed objects are defined, and for technical feasibility e.g. the continuity of setpoint curves across segment boundaries. Whereas the formal integrity check is the same for all units the feasibility check must be programmed individually for each unit. The list of segments to be checked is provided by the central sequence controller as a real time data field. The checking results of the function groups are also distributed via data fields.

Additionally every component can implement an arbitrary number of so called “extra devices”. These devices form a toolbox for all tasks of digital control, e.g. waveform generation, digital feedback control.

The segment sequence control unit is a special segment controller with extra devices dedicated to the co-ordination and supervision of the other function groups of the project. One segment sequence control is needed for each project. The segment sequence controller is the only segment control component knowing the planned segment sequence, i.e. the “valid discharge program” defined by the session leader. All others only obey the segment switch message of the sequence controller. The extra devices for sequence control are:

- A *segment scheduler* checks with every primary cycle whether one of the termination conditions defined in the segment description is fulfilled. If so a command to switch to the next segment in the valid discharge program is issued. A termination condition can be defined by expiration of the segment duration, waiting for a session leader interaction, waiting until all function groups are ready for the next segment, or determined by the result of a comparison of plasma parameters. The list of conditions can be exchanged on a per segment base.
- A *segment supervisor* combines the checking results of the segment managers of all function groups to a feasibility status of the project wide segment description. The decision depends on whether the operability of a function group is mandatory, obstructive, or optional for the success of the segment. If the running segment becomes unfeasible another segment according to an error recovery strategy is activated in all units. The error recovery strategy is not defined in the segment description but set by the session leader according to his experience.

5.2 Creation and handling of segment descriptions

5.2.1 Generating segment descriptions in the segment control units

The database contains segment descriptions for several projects (e.g. W7X) and function groups (e.g. ECRH). Segment descriptions are stored in tables in a relational database or as objects of classes in an object oriented one [1]. The tables/classes are named *<projectname>SegDesc* or *<groupname>SegDesc*, respectively, where *<projectname>* and *<groupname>* are replaced by unique names for project and function group e.g. *W7XSegDesc* or *ECRHSegDesc*. A *<projectname>SegDesc* contains an aggregation of *<groupname>SegDesc* objects, one for each function group belonging to the project. Segment descriptors as every database item have an identifier, also called segment number.

The database objects are never completely created in the segment controllers. The corresponding instances of the classes *ProjectSegDesc* and *GroupSegDesc* stored in a symbol table in the real time computers refer to a “unit segment program” containing only selected information relevant for the units. Each group segment description is an aggregation of device segment descriptions for all devices of all units belonging to the group. The name of the attribute of the device segment description is identical to the role name of the corresponding device. Since a unit knows the role names of its devices from the configuration only segment descriptions for the required devices are selected by a database query, created in the units, and bundled as a unit segment program.

Active segments are segment descriptions to be loaded at run-time by the units. A project segment description becomes active when its identifying segment number is listed in a dedicated database container called *<project name>Pool*. The pool can be filled any time using the session leader program. New entries are consecutively numbered. The units remember the consecutive number as a mark to which segment descriptions have been updated. On demand and at regular intervals the pool is checked for new entries. The entries are interpreted as identifiers of project segment descriptors which are generated in the real time computers.

A segment description can be deleted from the units by putting a new entry at the end of the pool containing the identifier and a flag marking the segment for deletion.

The same mechanism is also used to load group segment descriptors from a <group name>Pool in the database for stand alone commissioning and tests of a function group.

5.2.2. Switching between segments

A segment controller can switch at any time to a project segment when its segment description is

allocated and checked; and the function group does not operate stand alone.

The segment switch request is distributed as a multicast via the message system of the real time computers. All units of a project must be configured in such a way that they receive the message. The segment sequence control is set up to distribute the request. The message contains the segment number i.e. the identifier of the project segment descriptor as a positive integer accompanied by a time stamp in future when the segment is to be switched.

The units look for the specified project descriptor in their symbol tables and find the referenced unit segment program. When the time specified in the message will be hit with the next trigger of the primary cycle all units set a “switch” flag. At the next primary cycle trigger the reference to the current unit segment program is replaced by the new one containing references to the new device segment programs. A virtual method of the devices is called to switch the programs of each device and redefine parameters and device states if necessary. A segment switch takes five microseconds on a Pentium 4 processor. Since the timing systems of all units are synchronized with TTE precision, exactly synchronised action of all units can be triggered in the cycles following a segment switch.

When units run in stand alone mode segment switch requests from the central segment controller must not be obeyed. Since the message system is not altered by switching the operational mode the requests are received but ignored in stand alone mode. Segment switch requests in stand alone mode are sent via Unicast and the segment number is negated. In stand alone mode a unit accepts only messages with negative segment numbers, interprets the absolute value of the number as an identifier of a group segment descriptor to look for in its symbol table. After finding the referenced unit segment descriptor the switching mechanism is the same as for coordinated operation.

6. Conclusions

The software infrastructure for a segment control system for W7-X has been set up. The software package W7XSegmentControl provides numerous devices for general control tasks and segment related control, which are configurable and will be used with many units.

The implementation is validated in a prototype installation and during commissioning of the first installed technical components of W7-X.

Acknowledgements

The authors would like to thank the XDV data acquisition group for their support and the helpful discussions. We also wish to thank the participants of the design review meeting held at Greifswald in March 2005 for their fruitful discussions and proposals.

References

[1] Kühntopf H and the XDV Team. Specialized editor for processing objects in a database to prepare discharges for WENDELSTEIN 7-X, Fusion Engineering and Design – this issue

[2] Schacht J, Laqua H, and Niedermeyer H. Tasks and structure of the Wendelstein 7-X control system, Fusion Engineering and Design – this issue

[3] Heimann P and the XDV Team. Status report on the development of the data acquisition system of Wendelstein 7-X, Fusion Engineering and Design 71(2004) 219-224.

[4] Schacht J, Niedermeyer H, Wiencke Ch, Hildebrandt J, and Wassatsch A. A trigger-time-event system for the W7-X experiment, Fusion Engineering and Design 60 (2002) 373-379.

[5] Spring A, Laqua H, Niedermeyer H. User interaction concept for plasma discharge control on WENDELSTEIN 7-X, Fusion Engineering and Design – this issue

[6] Laqua H, Niedermeyer H,Willmann I. Ethernet based Real Time Control Data Bus, IEEE Trans. Nucl. Science. 49 (2002) 478-482.

[7] Schacht J, Laqua H, and Niedermeyer H. Synchronization of processes in a distributed real time system exemplified by the control system of the fusion experiment WENDELSTEIN 7-X, Proc. 14th IEEE-NPSS Real Time Conference, Stockholm, Sweden, 2005.