Event Generation and Simulation of Exception Handling with the ITER PCSSP

G. Raupp[1], M.L. Walker[4], G. Ambrosino[2], G. de Tommasi[2], D.A.Humphreys[4], M. Mattei[3], G. Neu[1], W. Treutterer[1], A. Winter[5]

[1] Max-Planck-Institut fuer Plasmaphysik, EURATOM Association, 85748 Garching,Germany

[2] CREATE/Università Di Napoli Federico II Dip. Ingegneria Elettrica E Delle Tecnologie Dell'Informazione Napoli, Italy

[3] Seconda Università Di Napoli Dip. di Ingegneria Industriale E Dell'Informazione Napoli, Italy

[4] General Atomics, PO Box 85608, San Diego, CA 92186-5608, USA

[5] ITER Organization, Route de Vinon-sur-Verdon, 13115, St. Paul-lez-Durance, France

Abstract:

The Plasma Control System Simulation Platform (PCSSP) for ITER shall support the analysis and development of methods to be used by the ITER Plasma Control System (PCS) for handling exceptions to optimize pulses and assist in machine protection. PCSSP will permit to investigate physical and technical events, such as component failures, control degradation, operation domain excess, plasma state bifurcation or instabilities, and interlock activity. Serving that purpose, the plasma, actuator, diagnostics and PCS simulation modules in PCSSP will be enhanced to compute nominal and off-normal data. Configured by an event schedule, an Event Generator will orchestrate the activation and manipulate the characteristics of such off-normal computation. In the simulated PCS exceptions will be handled in a Pulse Supervision layer operating on top of the Pulse Continuous Control (PCC) feedback loops. It will monitor events, decide on which exceptions to respond, and compute new control references to modify PCC behavior. We discuss basic concepts for the event generation in PCSSP, and a preliminary architecture for exception handling in PCS, and show how these will be configured with event and pulse schedules.

1. Introduction.

The ITER project aims at demonstrating sustained and stable burn of a thermonuclear plasma during long discharges. Such operation domains are not accessible with present machines, and require not only a well-designed plant and technical components, but need also a much more sophisticated active control. The ITER CODAC environment shall provide instrumentation and control functionality to operate the plant at Cadarache [1]. A core application of CODAC is the Plasma Control System (PCS), which drives a dozen of actuator plant systems for heating, fuelling and shaping of the plasma, and reads measured and evaluated data about the plasma and plant state from tens of diagnostic plant systems.

The mandate of PCS not only covers to establish the desired plasma parameters during the nominal evolution of the discharge [2]. A novel requirement for ITER PCS will be to maximize the scientific use of the device, i.e. dynamically optimize control methods in an investigation to improve plasma quality, or in case the results should not be satisfactory then schedule an alternate investigation to make best use of the long pulses. Another novel PCS task with top priority is to assist in investment protection, i.e. actively avoid violation of pulse control allowables which would trigger the Central Interlock System (CIS), or in case of a CIS alarm assist that system in handling complex physics situations like runaways or disruptions and terminate the discharge while minimizing stress and risk [3].

Such novel requirements demand for an ability of PCS to modify control schemes in real-time depending on plasma and plant state. ITER intends to investigate and optimize such dynamic schemes with simulation methods, which have already proven valuable for the development of continuous control in various Tokamaks [4-9].

Requirements, use cases and the preliminary architecture for the ITER Plasma Control System Simulation Platform (PCSSP) were presented in [10]. In this paper we will focus on how PCSSP is enhanced to simulate modification of control schemes in PCS: we will summarize the basic concepts of PCSSP (chapter 2), outline how PCSSP can simulate occurrence of events (chapter 3), propose a preliminary architecture for the exception handling in PCS to be simulated (chapter 4), and show how simulation runs shall be scheduled and logged (chapter 5).

2. Plasma Control System Simulation Platform (PCSSP)

PCSSP [3,10] will allow simulating Tokamak control behavior to help develop and

test ITER PCS architecture, algorithms and code. It will combine

- a Plant Simulator, which includes modules to simulate plasma, plant actuators and diagnostics, and the interlock system,
- a PCS simulator, which includes modules for simulation of pulse continuous control, and exception handling logic to be developed,
- an SDN and CIN simulator, which includes modules to simulate characteristics of ITER networks SDN (Synchronous Data bus Network, for real-time data exchange) and CIN (Central Interlock Network, for communication among interlock systems), if relevant,

and may connect to external codes for more sophisticated plasma simulation, if needed.

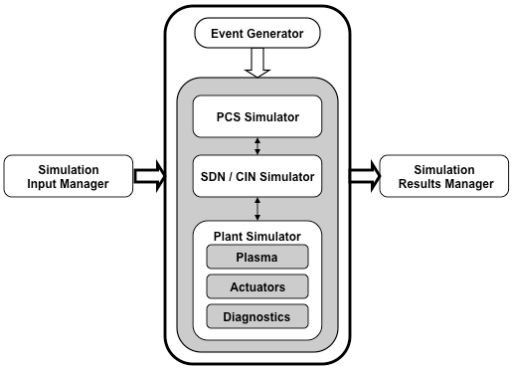

Fig. 1. The PCSSP environment combines various dedicated simulators.

In addition PCSSP will host an Event Generator function to provoke occurrence of events in various modules, to investigate how PCS would handle plasma state changes, component failures, etc.

The basic idea is that the Event Generator will trigger the Plant Simulator or Network Simulator modules to provide a specific set and sequence of off-normal plasma and plant and characteristics. These can be observed by the control modules in the PCS Simulator, where events are detected, and where the required exception handling policies are scheduled. Upon termination of a simulation run the output can be analyzed.

In the MATLAB/Simulink environment chosen PCSSP provides the means to integrate the dedicated simulators, and manage parameter input and results output for the simulation runs.

3. Events and Event Generation (EG)

3.1 Definition of Event and Exception

Different "event" definitions exist in science and philosophy. In the context of the ITER PCS we define an event as any system change that may (!) require to modify a control method. Depending on the system state or, discharge goal the occurrence of an event may then actually trigger a control modification: We define exception handling as the modification of a control method in response to an event.

As a simple example the failure of a magnetic pick-up is an event. Depending on the system state (e.g. redundant pick-up available; plasma state established) various exception handling methods can be specified

- to simply replace the failed pick-up by a redundant sensor (if the plasma is established and a redundant sensor is available)
- or perform a complex soft landing control scheme (if the plasma is established, and no redundant sensor is available).

In this simple example, no exception handing would be performed while no plasma is established, e.g. during technical calibration shots without plasma.

3.2 Classes of Events

The task of PCS "to maximize scientific use and assist in investment protection" does not immediately provide the list of events relevant for ITER PCS. However, based on the event definition, we can analyze existing plasma control systems, and derive situations where control methods must be adapted or modified [11]. Such situations can generally be bifurcation of nominal operation states (which requires to adapt the control method to the actual nominal state), or degradation when off-normal failure states develop (and must be managed with appropriate control action). Bifurcation and degradation relevant to PCS may occur in the domains of the

- plasma
- PCS controllers
- SDN (Synchronous Data bus Network)
- actuator plant systems and actuators
- diagnostics plant systems (including data evaluation tasks)
- CIS (Central Interlock System)
- CIN (Central Interlock Network)
- or plant operation conditions

Plasma events comprise bifurcation of the various plasma regimes with distinct behavior, including onset / termination of instabilities.

Controller events include degradation patterns such as failures of controller hardware, computation time-out, violation of control algorithm operation windows, or observation of excessive control errors.

Actuator events are degradation from hardware failures, trips, and saturation or self-protection.

Diagnostic degradation events comprise hardware failures, noise or impaired accuracy, diagnostic specific measurement artefacts, excess of the accessible observation range, or violation of the evaluation model window.

CIS (including CIN) state bifurcation includes the various alarms relevant to PCS (in particular where PCS shall respond with dedicated actions, to control disruptions or runaways).

SDN network events are degradation due to hardware failures, packet dropouts, or excessive latency.

Operation condition events can be degradation patterns where actual PCAs (pulse control allowables) are violated, or operation state bifurcation from operator intervention, or resource availability.

3.3 Reproduction of Events

To simulate nominal evolution of a pulse we represent the system by dedicated modules to compute the nominal data. E.g. an (ideal) diagnostic model computes the transfer function from a given plasma state into the measured data provided by the (ideal) sensor.

For simulation of events we must compute the nominal data plus the off-normal modifications, which represent the underlying state bifurcation or degradation effects. Such a (non-ideal) diagnostic module must then provide nominal data from the transfer function with off-normal aspects e.g. limit from observation boundaries, measurement artefacts such as fringe jumps, or superimposed noise.

The level of detail to which off-normal effects must be computed depends on the specific simulation purpose.

Various use cases exist for simulation of exception handling, e.g. analysis of dynamic switching and transition among control algorithms, of intervention threshold hierarchies, of local handling of diagnostic artefacts, of alarm propagation through PCS, of the interplay between PCS and CIS to handle disruptions, …

All such cases need specific event patterns, in terms of the choice of events to be produced, the specific event characteristics needed, and the sequence in which event shall occur at specific times or system states. Hence PCSSP must be able to generate on demand the desired sets of events.

3.4 Event Generator

The orchestration of the off-normal computation is the mandate of an Event Generator module in PCSSP [12]. Its preliminary architecture is shown in Fig. 2:

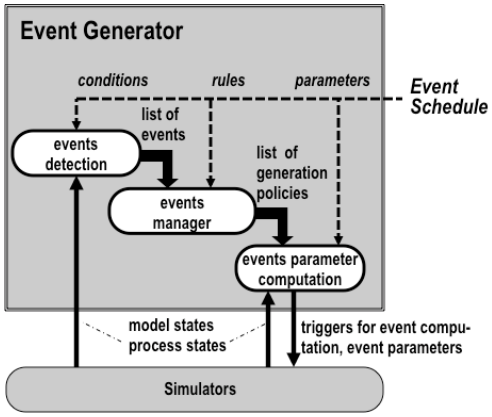

Fig. 2. Event Generator Architecture: Three sequential units detect events, arbitrate these, and compute events triggers and parameters for the underlying simulators

Configured by an event schedule, the Event Generator issues commands and parameters characterizing the events to the dedicated simulators for plant, plasma, SDN and PCS, to compute specific off-normal data patterns. An example is a command to a plasma module to calculate an NTM pattern with a given strength and frequency, or a command to a diagnostic to compute a fringe jump of specified offset.

General requirements to EG are [13]:

- to drive all simulators and asynchronously trigger modules to compute events with given characteristics
- to drive single or concurrent events or event sequences at specified times or states
- to control specific event characteristics
- to be not part of the control loop to be modeled
- to be configurable

The Event Generator will satisfy such requirements with three distinct functional blocks:

An events detection unit will observe if an event is to be triggered as a result of a change in process states (data accessible in an actual PCS control loop), of a change in model states (data accessible only in the simulation), or when a fixed time has elapsed. The result is a list of pending events to be generated.

An events manager unit arbitrates the generation of various events. This includes managing the superposition of one event, which may be alternatively triggered by states or time, and arbitrating different conflicting events with a priority scheme. The result is a conflict free list of event generation policies to be processed. (Today no mandatory use cases are known for this arbitration function, which originates from the requirement for state dependent event generation. However, the place for such functionality is defined, so that dynamic manipulation of event sequences can be supported in the future, if needed.)

An events parameter computation unit generates the event triggers and compiles parameters describing the desired event characteristics to be issued to simulator modules. Such parameters may be specified in the pulse schedule as time-varying waveforms, or be computed with algorithms configured in the event schedule.

The structure of the Event Generator can be replicated: various instances can be assigned to dedicated simulators of PCSSP, if no arbitration is needed throughout instances.

4. Exception Handling (EH)

As defined in Chap. 3.1 EH is the modification of a control method in response to an event.

As an example take a situation where the plasma shape feedback control is done with global parameters such as elongation, triangularity and X point location. In case the plasma shape should come to close to the wall, an alternate method must be activated, where the plasma shape is controlled via a number of plasma-to-wall gaps, to more immediately control the critical parameter and better protect the wall.

4.1 EH Use Cases

R&D for EH use cases and related handling policies of PCS is an ongoing activity. On the base of a preliminary collection of EH applications for ITER and other machines, various methods have been discussed, such as switching among complete segments of the Pulse Schedule [14,15], temporal modification of a sub-set of references for specific controllers (choice of controller operation modes, desired values, control parameters) to partially overlay segment data, or computation of new references in real-time [16,17].

From such preliminary methods we identify a general pattern for EH: state dependent decisions activate a set of handling policies. These are translated into modified references, to govern and modify the continuous control characteristics (e.g. switching from one controller to an alternate one, or changing a gain matrix to better handle some control situation, or activating an alternate data evaluation method in case a measurement channel fails).

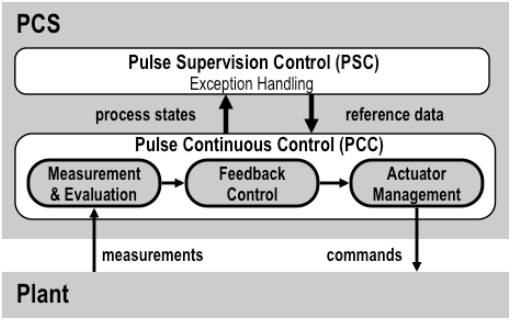

Fig. 3. In PCS the PSC block takes decisions to implement exception handling, and steers the PCC block, which performs the closed loop control

4.2 Preliminary EH Architecture for the PCS Simulator

To support study of EH techniques in PCSSP, a simple initial PCS simulator shall be procured which also includes EH. The PCS simulator (Fig. 3) consists of a Pulse Continuous Control (PCC) layer to operate open and closed loop control, and a Pulse Supervision Control (PSC) layer to take EH decisions and apply these via the PCC.

In more detail, PCC implements all open or closed loop tasks to operate the plant, i.e.

- measurement and evaluation of data into the physical and technical data needed for control and monitoring
- feed forward of actuators and feed back of controlled technical and physical quantities
- management of actuators for sharing, load balancing and redundancy handling

Complementary to PCC, the PSC implements EH functionality and

- continuously reads process states and data from the Pulse Continuous Control (PCC) block
- steers PCC through reference data for the choice of specific controllers, their settings, and the reference data for the desired feed forward or feed back action.

With such task separation in PCS between PCC and PSC [18], the preliminary architecture for PSC is shown in Fig. 4:

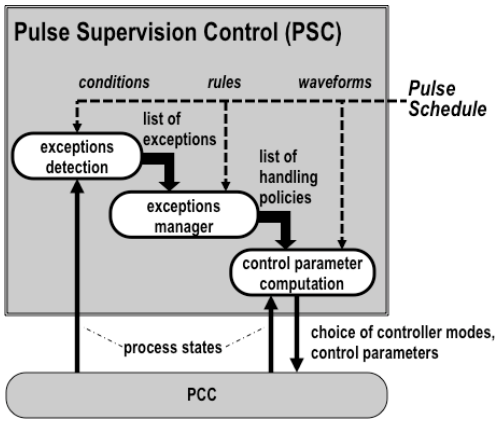

Fig. 4. Pulse Supervision Architecture to implement central exception handling: Three sequential units detect exceptions, arbitrate these, and compute control parameters for the underlying PCC controllers

An exceptions detection unit will monitor occurrence of exceptions, because of a change in process states (data in the actual PCS control loop, provided by PCC), or because a specified time has elapsed. Input to the event detection can be all real-time data provided through SDN; this may include quality tags added to such real-time data, or results from distributed EH (see Chap. 4.3).

An exception manager unit takes the list of active exceptions which might be handled, and executes rules to find out which exceptions have priority, need arbitration to avoid conflicts on the level of controlled quantities or actuators, or can be executed simultaneously. It will also block recursion to avoid infinite loops.

A control parameter computation unit will take the list of exception handling policies to be executed, compute the new choices of controller modes, and the control parameters to be applied (reference waveforms, gain parameters, thresholds, …) from data in the segmented pulse schedule or with real-time algorithms, and issue these to PCC for execution.

That PSC architecture satisfies the requirements identified for EH:

- to access all relevant state data
- to modify control strategies and algorithms in PCS
- to allow termination or interruption of EH, support concurrent handling of multiple exceptions and arbitrate conflicts in EH
- to be configurable and to permit disabling of individual EH functions

4.3 Provisions to Study Distributed EH

Initially only one instance of PSC will be implemented, and provide EH functionality for all PCS, to take decisions of global and local relevance.

However, PSC can be replicated, so that functionally separable instances can be dedicated to various modules in the PCC simulator to perform local EH. Such distributed intelligent control structures improve encapsulation and performance, and would allow investigating e.g.

- how intelligent measurement and evaluation processes could perform local handling of degraded or failed sensors or evaluated data with redundant hardware or with synthetic data
- how intelligent controllers could switch among various sets of controller settings, to optimize control performance
- how intelligent actuator manager circuits could handle degraded or failed actuators with redundant hardware

and to find the optimum allocation of local EH in the PCC layer, and of global EH in the PSC layer.

5. Schedules for Simulation

ITER PCS shall be operated with a pulse schedule, which will include all the information needed to define PCS functionality, drive the technical systems and steer the discharge through the physics phases. Structure and content of the pulse schedule are not yet fully defined, however to serve the purpose it must include

- static configuration data, to choose the set of PCS functionality for a

pulse
- time-varying reference data, to describe the desired pulse evolution in terms of controller choices, waveforms for controlled quantities, and control parameters (including monitor functions)

The (reference data portion of the) Pulse Schedule for ITER shall be segmented. This allows cutting the long pulse into pieces with distinct technical or scientific content, and edit and manage these separately, but also permits to dynamically schedule alternate segments depending on the actual plasma and plant state. This method is currently foreseen for ITER to implement a global EH, which impacts the entire PCS. Other, more finely grained EH methods, which affect only distinct control functions, may be added as result of the ongoing R&D activity on EH. The granularity of the EH methods required for ITER PCS will then define the granularity of the conditional descriptions needed in the Pulse Schedule structure.

The PCS simulator will initially use a segmented ITER pulse schedule look-alike structure, until the final structure will have been defined. That schedule will provide PCC and PSC relevant information, i.e.
- data to configure evaluation, feedback and actuator management controllers
- data to configure the exception monitoring and management
- data to define available exception handling policies (which includes segment transition and the interpretation of segment data).
- and the reference waveforms defined in nominal and off-normal segments

To define the desired sequence of events for a PCSSP simulation run, the Event Generator needs an event schedule. Comparing the preliminary architectures of PSC and Event Generator (Fig. 1 and Fig. 3) shows great functional similarity between these, with blocks to detect events / exceptions, to manage these in case of conflicts, and to compute resulting output data. This suggests re-using the pulse schedule structure for the event schedule, including segmentation. The event schedule will hold:
- data to configure the event monitoring and management
- the sequence of events to be applied when executing the nominal and off-normal segments of a pulse schedule

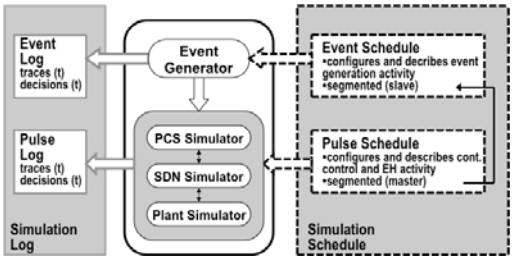

Fig. 5. A PCSSP simulation will be driven by complementary pulse and event schedules, and provide event and pulse logs for analysis. PCS tools shall be re-used to manage these.

The simulation dynamics is thus defined by the combination of an event schedule providing event triggers and a pulse schedule defining the responses to events. If pulse schedule and event schedule use the same segmentation (which eases the editing of a simulation run), the Event Generator must perform event schedule segment synchronously with the pulse schedule segment changes in PCS.

Both PCS and Event Generator will log data, to trace which decisions were taken and which output was computed.

Tools are needed to efficiently store, retrieve, visualize and manipulate schedules and logs. To minimize development effort and cost, and maximize usability to users the PCS data structures and tools for PCS shall be re-used for PCSSP where possible. Fig.5 shows how schedules and logs interact with PCSSP functions during a simulation run.

6. Status of PCSSP Development.

The general concepts shown were the base for the mid-term development of a continuously increasing PCSSP functionality. To demonstrate the concepts, an initial prototype with base features was implemented during 2013. For EH it permits to analyze simple isolated exceptions, such as the occurrence of a machine protection alarm or a component failure, the change of the plasma state, or the violation of an operation boundary, and their handling by modification of controller settings and references as defined in an alternate pulse schedule segment, or computed by an algorithm.

7. Conclusions

The PCSSP environment is designed to support development of ITER PCS and optimize Exception Handling. To create

events, i.e. system changes which may require a change of the control method, an Event Generator is a novel part in that simulation environment: It triggers at specified times or states the plant and plasma simulators to compute off-normal situations with given characteristics. PCSSP allows then to observe how the simulated PCS responds to such events: An initial architecture for such exception handling in PCS was designed, and implemented as a Pulse Supervision Control layer on top of the Pulse Continuous Control layer. The Pulse Supervision accesses the real-time process data (including data quality tags) available on SDN, detects events, takes state-dependent decisions about applicable exception handling methods, and modifies control methods in the underlying Pulse Continuous Control through modified reference data. By end of 2013 the prototype implementation of PCSSP including a simple PCS was demonstrated at ITER to prove the usefulness of the concepts. The event and exception definitions, the method to simulate off-normal characteristics with an Event Generator triggering the dedicated plant simulators, and the initial architecture of the PCS Continuous and Supervision Control layers performing exception handling were found to be clear, intuitive and easy to operate and enhance.

References:

[1] A. Wallander et.al., ITER instrumentation and control-Status and plans, Fus Eng Des 85(3-4):6 (2010)

[2] J. Snipes et al, Physics of the Conceptual Design of the ITER Plasma Control System, this conference

[3] A. Winter et.al., Architectural Conceptual for the ITER Plasma Control System, this conference

[4] M.L. Walker, et al, Advances in Integrated Plasma Control on DIII-D, Fus Eng Des 82 (2007) 1051

[5] D.A. Humphreys, et al, Development of ITER-Relevant Plasma Control Solutions at DIII?D, Nucl. Fusion 47 (2007) 943

[6] W. Suttrop, et.al., Predictive Simulation of Tokamak Discharge Behaviour based on Simple Scalings, Proc 32nd EPS Conf Plasma Phys, Tarragona, v.29C, 2005, P-4.076

[7] G. De Tommasi, et al, XSC Tools: a software suite for tokamak plasma shape control design and validation, IEEE Trans. Plasma Sci., v.35, no. 3, 709-723, Jun 2007

[8] J.B. Lister, et al, Development of the DINA-CH Full Tokamak Simulator, 38th European Phys Soc Conf on Plasma Physics, Strasbourg, France, 27 June - 1 July, 2011, P1.105.

[9] M. Ferrara, et al, Alcasim Simulation Code for Alcator C-MOD, Proc. 45th IEEE Conf. on Decision and Control, San Diego, CA, 13-15 Dec. 2006, 2238

[10] M. Walker et.al., A simulation Environment for the ITER PCS Development, this conference

[11] Exception Handling, v1.1, 12 Nov 2012, CXAFLL

[12] Preliminary Architecture for PCSSP, v1.0, 13 Dec 2012, CAARKY

[13] PCSSP Final Requirements Document, v.1.2, 9 Apr 2012, 4FK397

[14] G. Raupp et al., ASDEX Upgrade Discharge Control and Shot Management", Proc. 17th Symposium on Fusion Technology, Roma (I), 1992, p. 1072

[15] A. Spring et al., A first W7-X experiment program editor, Fus Eng Des 85(2010) 525-528

[16] W. Treutterer et al., Event Detection and Exception Handling strategies in the ASDEX Upgrade discharge control system, Proc. 27th Symposium on Fusion Technology, Liege(B), 2012

[17] P. Moreau et al., Conceptual Desing of the WEST plasma control system with built-in event handling functionalities, Proc. 7th IAEA Technical Meeting on Steady State Operation of Magnetic Fusion Devices, Aix-en-Provence (F), 2013

[18] W. Treutterer et.al., Architectural Concept for the ITER Plasma Control System, this conference