Structured System and Software Design of Distributed Control Computer of Mechatronic Facility for Space Research

Jiri Zdenek [1)], Jiri Lettl [2)] and Jan Bauer [3)]

[1)] Czech Technical University in Prague/FEE, Prague, Czech Republic, e-mail: *zdenek@fel.cvut.cz*
[2)] Czech Technical University in Prague/FEE, Prague, Czech Republic, e-mail: *lettl@ fel.cvut.cz*
[3)] Czech Technical University in Prague/FEE, Prague, Czech Republic, e-mail: *bauerja2 @ fel.cvut.cz*

Abstract — **The structured system and software design method of the distributed control computer network of the mechatronic scientific facility for automatic high temperature material processing in the space in micro-gravitation environment is presented in the paper. The facility distributed computer controls special extra low speed vibrationless electric drives, multi-zone high temperature furnace heating system and several measurement devices. The computer is equipped with special technological language to be possible to specify experiment time flow. Computer network nodes have many user processes (threads) so efficient, reliable, self documented, and low overhead scheduler dispatcher is required. The table driven coroutines have been designed.**

Keywords — Mechatronic, real time processing, software, cooperating FSM.

I. Introduction

The high temperature scientific facility consists of several different functional parts. They are the special electric drive unit, high temperature heating protection unit, temperature precise measurement and microgravitation units, technology computer, crew interface computer and telemetry computer.

Design task is to define optimal (suboptimal) control computer structure and to design suitable easy modifiable system software architecture to be possible to schedule and to dispatch all user tasks by in time, reliably, safely and well documented fashion.

II. Facility Requirement Statement

Requirement statement defines following main operational and functional features of the facility as follows (in brief):

Three electric drives are required where one of them is very special. The drive has to be vibrationless with a speed control where maximum to minimum speed ratio is to be 3.000.000. The position control has to have 0.1 mm resolution and minimum speed is 0.25 mm/day (one quarter of mm per day)! There is six segment high temperature heating system up to the temperature of 1200 °C. Temperature has to be precisely controlled with 0.01 °C resolution in the range from 40 °C to 1200 °C. Insample temperature measurement with programmable moving probe and three axis micro-gravitation measurement with high accuracy are there. Special technology programming language is required to be possible to prepare and program individual experiments time and functional flow by specialists in physics. The preprogrammed experiments have to be carried out in totally automatic mode. All measured data and process variables have to be recorded for later analysis. Safe and reliable operation is required. In the case of non standard operation behavior it is required fault tolerance against one individual fault anywhere in the facility. Other required features are not listed here. Well documented software data and control flow is required to be easily detected the point of malfunction if non standard behavior occurs. Smooth cooperation of four contractors with long distance place of business has to be available.

III. Requirement Statement Analyze

The carefully undertaken analysis of the facility requirement statement gives first approximation of the extent of the user task number and intertask communication capacity. The individual groups of functions have so different features and temporal requirements that the distributed computer system will be probably the suitable solution. The estimation of the number of user tasks leads to the figure of one thousand. The question is what will be a good system software option, the pre-emptive real time system (RTOS) or another choice with lower overhead. The relatively high number of user tasks (thousand) may result in unacceptable overhead if RTOS scheduler and dispatcher will be used.

IV. Structured Design Method

The design keystone is a correct distributed computer system functional partition to set up no node processor throughput or communication bottleneck [1], [2]. First of all it is necessary to collect all system functional requirements. It is often very difficult and iterative process. The result of the process can be requested to modify size of system. So, another important design point is to make possible to modify (make larger or smaller) the designed system easily. To define control computer system partition we have to select from pool of contractor system requirements group of all user tasks that are to be spread in suitable nodes of the distributed network control computer (DNCC). Further we have to gather user tasks by application of criterion (threshold) functions that can run together in one node of the DNCC, i.e. to make groups of tasks for computer modules with CPU and memory.

Finally for defined groups of tasks we have to design or buy suitable hardware and decide which system software is to be used for user tasks scheduling. Selection of groups is iterative process which we have to continue until no user task is unassigned.

Let us define following symbols:

$pjrq_i$ – project function requirements,
utg_i – user task group,
ut_i – user task,
$ThFR_i$ – task function requirement threshold function,
$ThCM_i$ – inter-task communication threshold function,
$ThTP_i$ – CPU throughput threshold function,
$ThFT_i$ – system fault tolerant threshold function,
$ThUT_i$ – user task selection threshold function,
$utgXX_i$ – user task group selected by XX criterion,
$pjSW$ – project software,
$pjSW_k$ – unassigned software after step k,
ss_i – system software support, RTOS etc.,
$hwNode_i$ – hardware network node with CPU.

Now it has to be defined a pool of user tasks ut_j with help of a set of application selection threshold functions $ThUT_j$ from set of project functions $pjrq_i$ (1) and we get unassigned software tasks $pjSW$ (2).

$$ut_j = ThUT_j\left(\sum_i pjrq_i\right) \tag{1}$$

$$pjSW = \sum_i ut_i + \sum_j ss_j \tag{2}$$

Further we define application function requirements (3), inter-task data flow rate (4), each CPU required (or estimated) throughput (5) and fault tolerance design (6).

$$utgFR_j = \sum_i ThFR_j(ut_i) \tag{3}$$

$$utgCM_j = \sum_i ThCM_j(ut_i) \tag{4}$$

$$utgTP_j = \sum_i ThTP_j(ut_i) \tag{5}$$

$$utgFT_j = \sum_i ThFT_j(ut_i) \tag{6}$$

Finally we get group of tasks utg_j (7) to which proper hardware $hwNode_j$ (8) will be assigned.

$$utg_j = utgFR_j \cap utgCM_j \cap utgTP_j \tag{7}$$

$$hwNode_j \leftarrow utg_j \tag{8}$$

Unassigned software $pjSW_k$ (9) have to be solved in the next iterations until $pjSW_k$ set is not empty.

$$pjSW_k = \sum_i ut_i - \sum_k utg_k + \sum_m ss_m \tag{9}$$

When all groups of the user tasks utg_j are assigned to hardware, type of system software has to be selected. Many important system parameters are to be considered carefully to make correct decision. That means it has to be considered above all number of tasks inside group, suitable method of user task scheduling (preemptive or non-preemptive RTOS) [3], feasible overhead of system software, interrupt latency, data flow rate, communication channels capacity and latency, data and program memory size requirements, maximum stack depth, estimation of required spare resources, real time debugging tools, real time monitor features, etc.

V. System and Software Structure Solution

Partition of functional requirements into group of similar functions and tasks is in Fig. 1. The basic software architecture is the client-server model. There is crew technological client which is user communication center and its function is: to store prepared experiments program, to download it to the technological server and to upload and to store the measured experiment run-time data. The technological server or experiment program client is main operational center of the scientific facility. It controls whole process and defines global timing of the current experiment. Its operation is based on a prepared experiment program which is executed from the technological server memory. There are two global modes of the technological server operation. It is automatic operation with crew supervision or automatic operation with no crew supervision when measured data are stored in the technological server and are not transported to the crew client. The technological server is client of heating server, electric drive server and in sample measurement server. Other parts of facility have been designed by different contractors and are not described here.

Group of function and task have been partitioned so that the inter group communication flow is minimized, so it is possible to use relatively low speed communication channel which impose small burden to the involved processors.

Final facility system architecture has topology of double star with full duplex point to point serial channels (Fig. 3) where the first star core is CIC node and the second star core is CC node.

Resulting from large figure of tasks in particular servers it has been decided to not use preemptive RTOS [4], [5] where higher overhead is possibile. It has been designed a non preemptive scheduler-dispatcher where coroutines in the form of cooperating finite state machines (FSM, Moore type) are used (Fig. 2), [6]. The FSM are table driven to be directly available program control flow documentation which may be important if non standard operation occurs (fault). Time sensitive tasks are executed under standard vectored interrupt. The FSM clock is run-time controlled by the FSM table driven program (Fig. 4). There are three FSM clock options, standard synchronous,

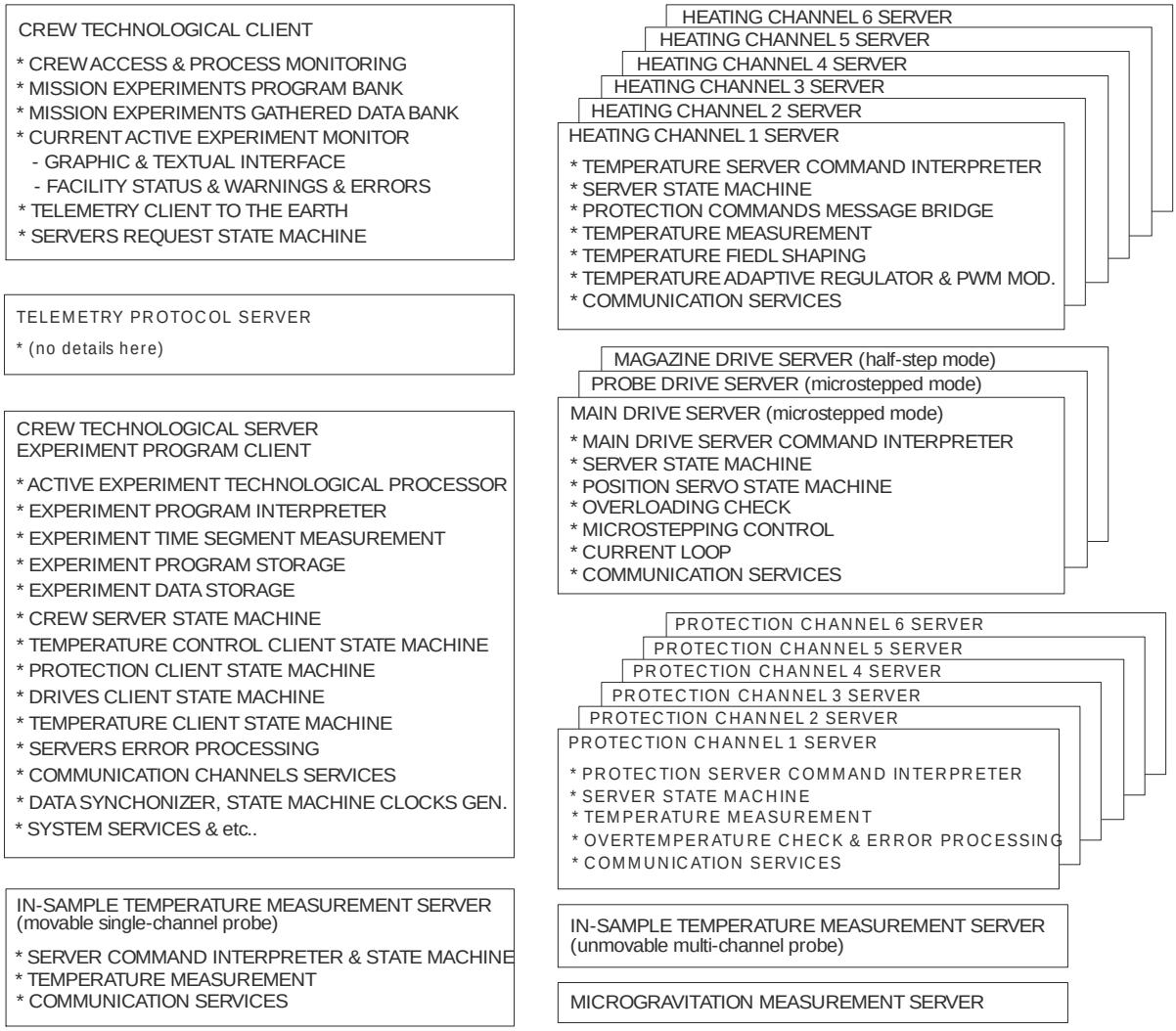

Fig. 1. Task groups (Raw global summary)

next step immediately and programmed time-out. Cooperating FSMs has to be synchronized by synchronizer because the operation speed of different servers is considerably different. Overall facility system structure is shown in Fig. 5. The M1 is main vibrationless extremely low speed ampoule motion drive (deeply micro-stepped steeper motor). The M2 is rotating ampoule magazine motor and the M3 is motor which moves insample temperature probe.

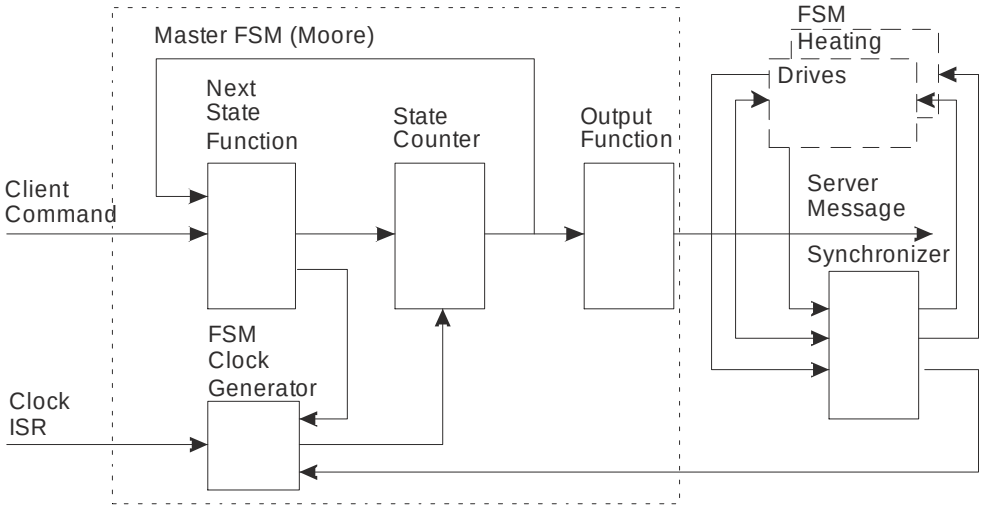

Fig. 2. Scheduler - Dispatcher

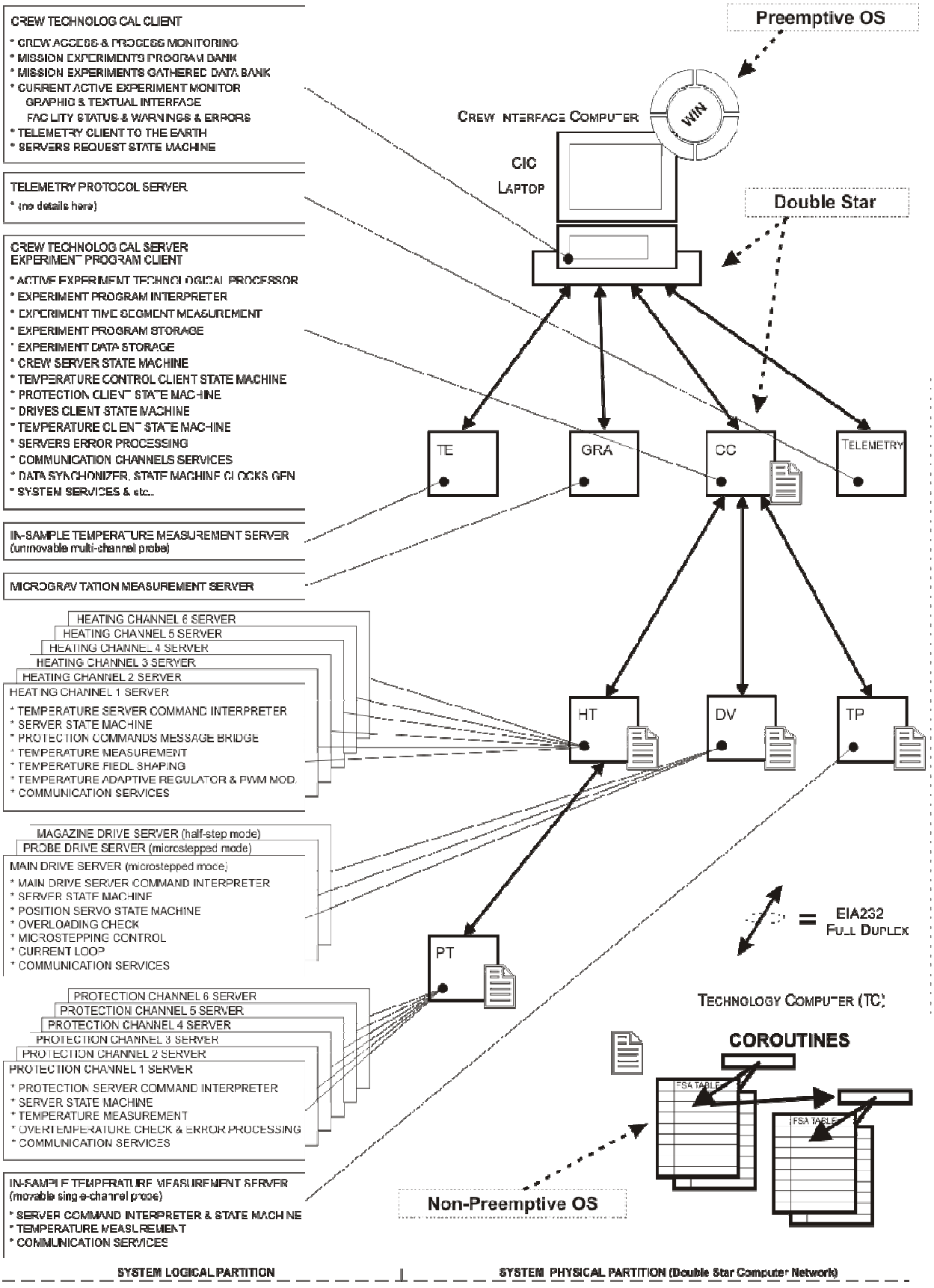

Fig. 3. DNCC system and software partition (Double star computer network)

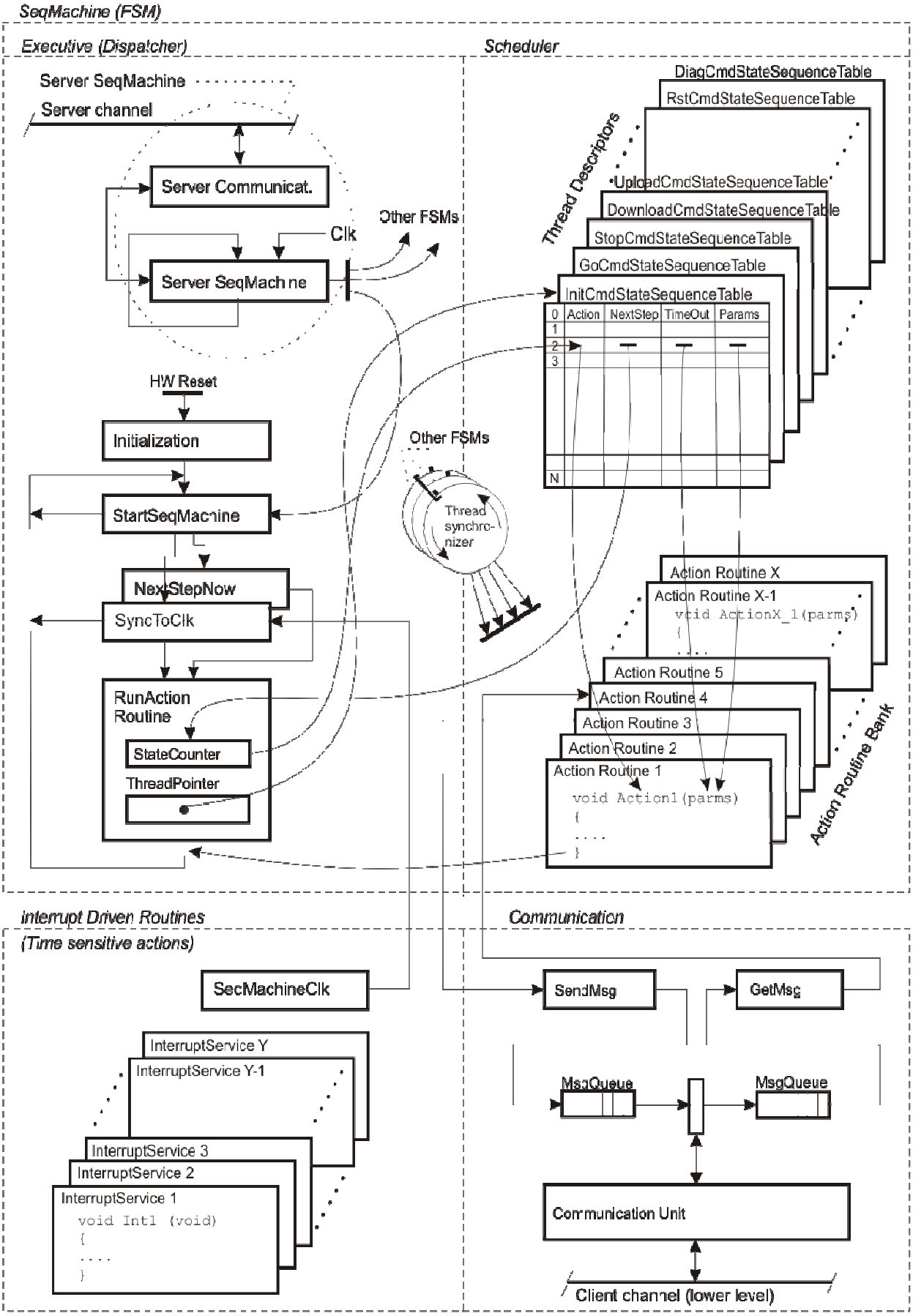

Fig. 4. Server slave FSM – Table driven normalized software architecture

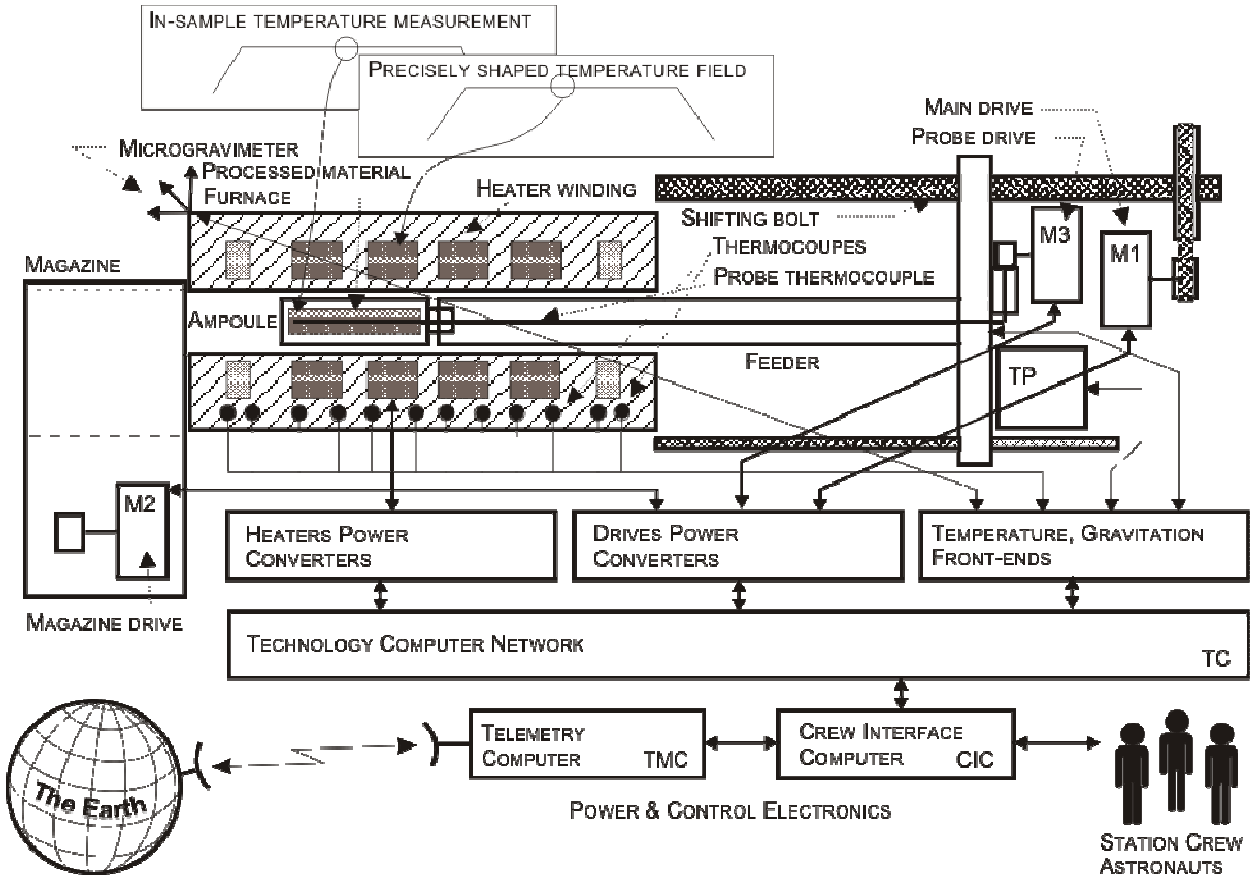

Fig. 5. Space technology facility system structure

VI. CONCLUSION

The scientific facility with presented system structure and software architecture was implemented. The table driven non preemptive cooperating finite state machine system software was proved to be suitable solution in program environment where a large number of user tasks have to be scheduled and dispatched and where the hardware resources are limited. The system small overhead is required in the application and the implemented solution offers it. The facility tests prove the designed scheduler-dispatcher architecture gives stable, efficient and well documented (self-documented processes/threads tables) solution with limited system resources requirements and especially reduced requirements for the processor speed and system stack space. The system software solution has very low overhead. Especially useful option have proved itself to be the possibility to slow down finite state machine clock during debugging stage when the operation path of the complicated FSM threads can be tracked down easily by the programmer in slowed down semi-real time environment. The overall software solution is easily scalable and modifiable as system requirements develop during time of design and operation tests.

The process flow tables (thread descriptors) in the table driven finite state machine are directly used as correct program flow final documentation with no need to transform it to another form. The facility has been in long time operation in the Space and no nonstandard behavior has been detected.

REFERENCES

[1] Buede, D. M.: The Engineering Design of Systems, Models and Methods. John Wiley, 2009, ISBN-9780470164020

[2] Wieringa, R. J.: Design Methods for Reactive Systems, Elsevier Science, 2003, ISBN-1558607552.

[3] Kopetz, H.: Real Time Systems: Scheduling, Design Principles for Distributed Embedded Applications, Kluwer, 2003, ISBN-07923989947.

[4] Zdenek, J.: System Design and Software Architecture of Traction Vehicle Control Computer, Proc. of 12th Int. Conf. EPE-PEMC2006, Aug. 2006, Portoroz, pp. 1205-1210.

[5] Zdenek, J.: "Distibuted Control Computer Backbone Communication Channel of Electric Locomotive with DMA support, Proc.of 14th Int..Conf. EPE-PEMC2010, Ohrid, September 2010, pp.185-192

[6] Zdenek, J.: Efficient Scheduler-Dispatcher Software Architecture of the Space Power Facility Distributed Control Computer, Proc. of 12th Int. Conf. EPE2007, Sept. 2007, Aalborg, pp. 1-10, CD-ROM.

.

The contribution was presented on the conference EDPE 2011, THE HIGH TATRAS, SLOVAKIA