

EECS 223:
**Introduction to
Real-Time Distributed Programming**
**Lecture : TMOs Containing
both SpMs and SvMs**

05.5.4	1
--------	---



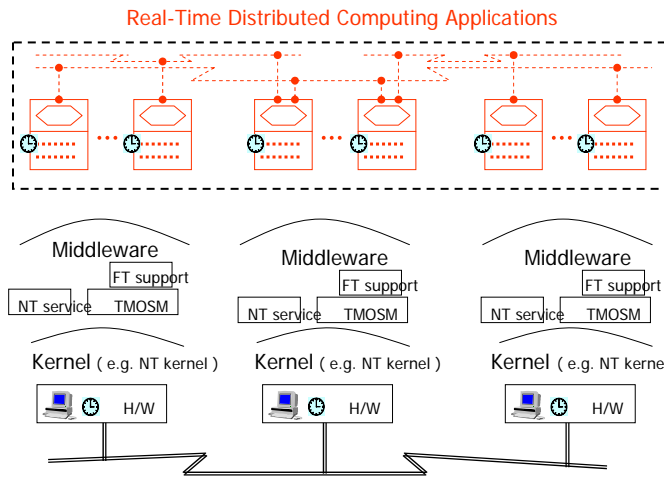
Lecture Outline

- TMO programming scheme : A revisit
- TMO Support Middleware (TMOSM)
- TMO Support Library (TMOSL)
- Basic Concurrency Constraint (BCC)
- Candidate AAC
- One complete example program

05.5.4	2
--------	---



Making Application Programmers' Life Easier: Structure as TMO networks relying on intelligent execution facilities



- No concerns with
 - Processes & Threads
 - Object locations (except in avoiding overloaded nodes)
 - Low-level comm protocols

- No specification of timing requirements in indirect terms (e.g., priorities)
 - Only *start-windows* and *completion deadlines* for object methods and
 - *time-windows* for output actions



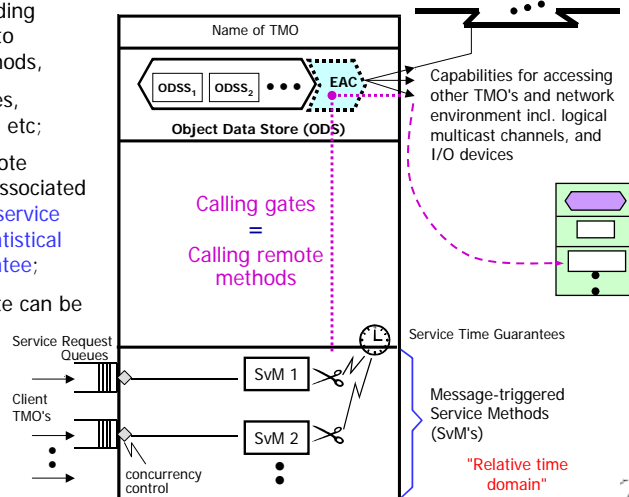
05.5.4 | 3

TMO Programming Scheme

- High-level distributed computing component:

EAC (Environment Access Capability) section (an ODS extension) provides

- *Gate* objects providing efficient call-paths to remote object methods;
- I/O device interfaces, channel interfaces, etc;
- Each *gate* to a remote service method is associated with a *guaranteed service time bound* or a *statistical service time guarantee*;
- Client's call to a gate can be associated with *deadline for result arrival*, nothing more;

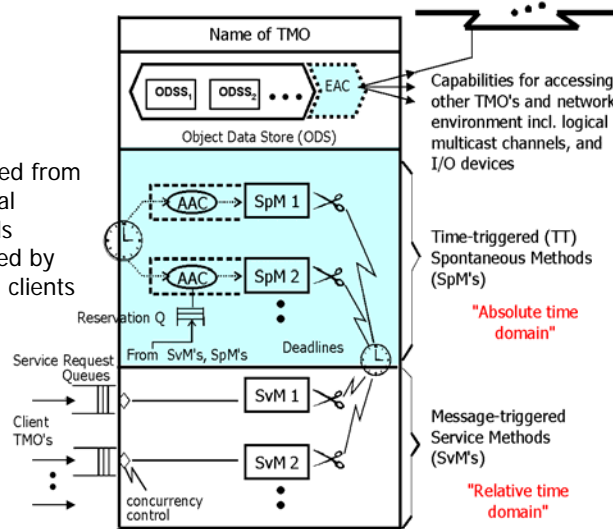


05.5.4 | 4

TMO Programming Scheme (cont)

- Time-triggered (TT-) or spontaneous methods (SpM's):

- Clearly separated from the conventional service methods (SvM's) triggered by messages from clients



05.5.4 | 5

TMO Programming Scheme (cont)

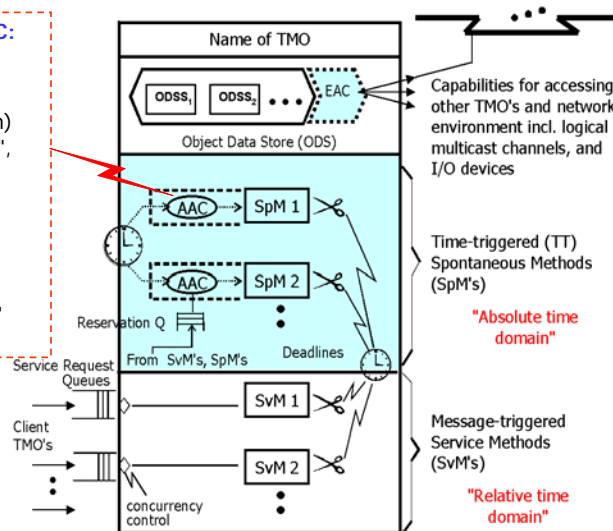
- Time-triggered (TT-) or spontaneous methods (SpM's):
- Clearly separated from the service methods (SvM's) triggered by messages from clients

L0

```

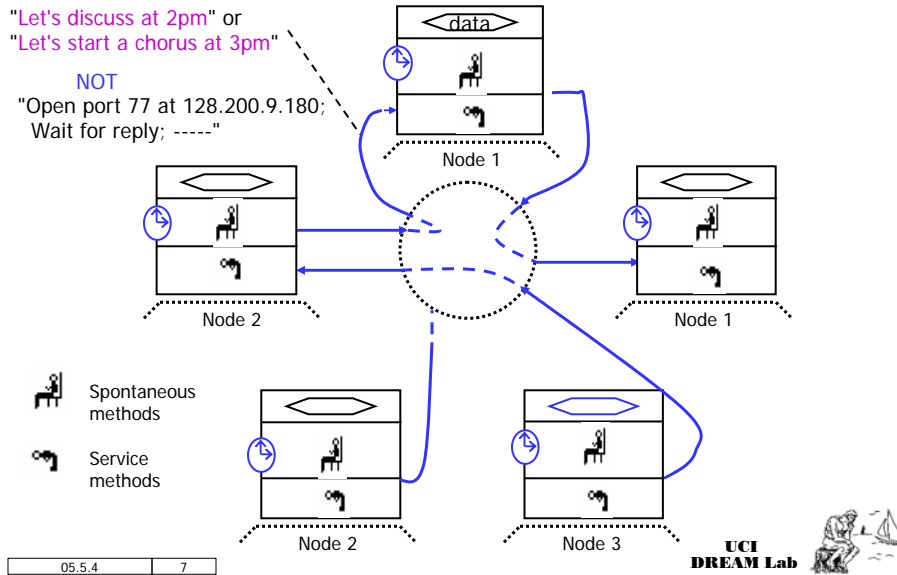
Example of AAC:
{
  "start-during (10am, 10:05am)"
  "finish-by 10:10am",
  "for t = from 10am to 10:50am every 30min"
  "start-during (t, t+5 min)"
  "finish-by t+10min"
}
    
```

Actions to be taken at **real times** determined at the **design time** appear only in SpM's



05.5.4 | 6

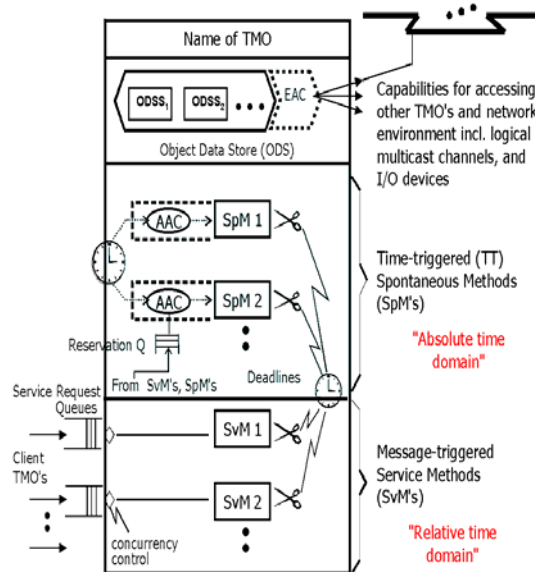
TMO Programming Scheme (cont) Time-Based Coordination of Distributed Actions



05.5.4 | 7

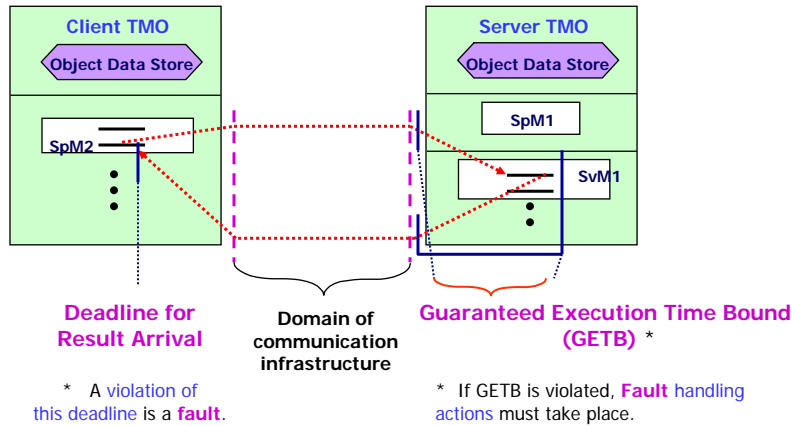
TMO Programming Scheme (cont)

- Time-triggered (TT-) or spontaneous methods (SpM's):
- **Time-window** imposed on each **output action** and **method completion**
 - Guaranteed service time
 - Client's deadline demand
 - Statistical assurances of better service times



05.5.4 | 8

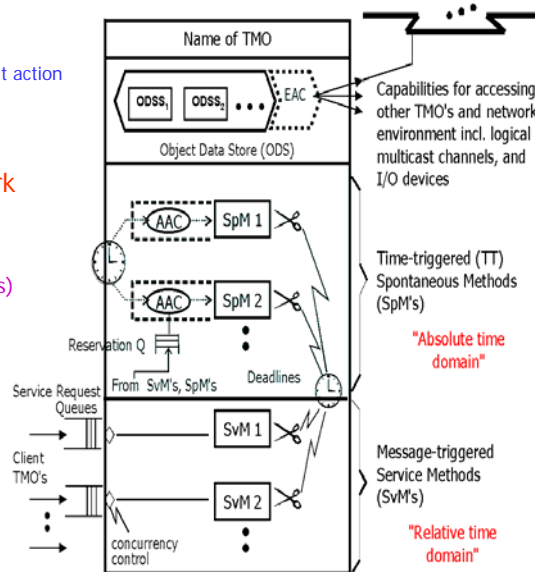
Return Deadline vs. Guaranteed Service Time



- * **Deadline for result arrival** - Call initiation time
 - > Max trans times imposed on comm infrastruc + GETB
 - > Time consumed by communication infrastructure + GETB

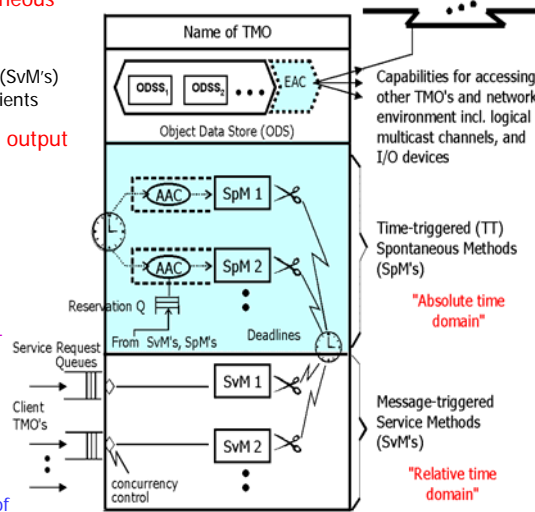
TMO Programming Scheme (cont)

- Time-triggered (TT-) or spontaneous methods (SpM's):
- Time-window imposed on each output action and method completion
- **Connections to the network environment** as possible data members:
 - **TMO gates (= access capabilities)** (opening the services available from other, possibly remote, TMO's)
 - **Real-time Multicast & Memory-replication Channel (RMMC)** (including distributed replicated variables) **gates**



Time-triggered Message-triggered Object (TMO) Structuring Scheme

- **Time-triggered (TT-) or spontaneous methods (SpM's):**
 - Clearly separated from the conventional service methods (SvM's) triggered by messages from clients
- **Time-window imposed on each output action and method completion**
- **Connections to the network environment as possible data members:**
 - **TMO gates** (= access capabilities)
 - **Real-time Multicast & Memory-replication Channels (RMCC)**
- **Basic concurrency constraint (BCC):**
 - SpM executions not disturbed by SvM executions.
 - Eases **design-time guarantee of timely services** of TMO's



UCI DREAM Lab

05.5.4 11

TMO Execution Engine – TMOSM (TMO Support Middleware)

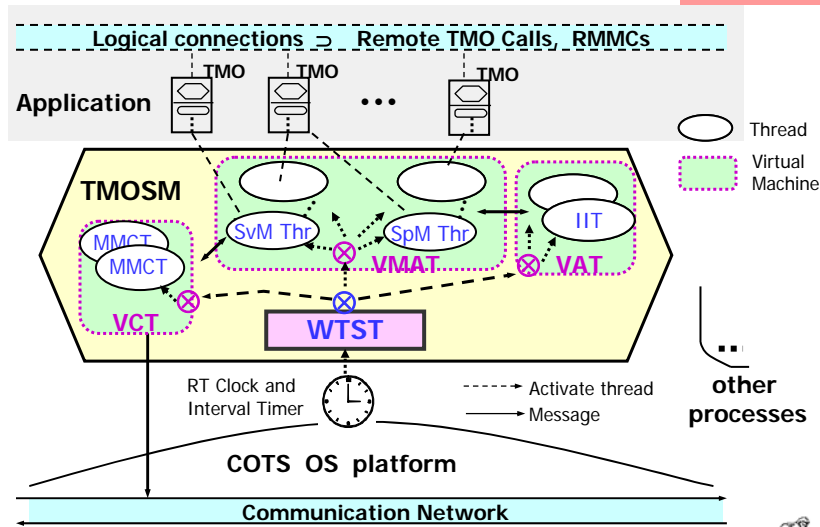
- A **middleware architecture** supporting TMO execution
- Supports **distributed, real-time programming** on COTS platforms
 - Allows programmers to express action timings *flexibly* and *well-structured forms* (at the level of 10 milliseconds with an implementation based on Windows XP)
- User-friendly C++ API, **TMOSL (TMO support library)**
- High **portability** and expandability
 - Can be ported to most modern OS with small effort
- Communication based on UDP.
 - **Can be based on IP multicast, CORBA, DCOM or SOAP.**
- Two application demos, CAMIN and DOFS, are available at <http://dream.eng.uci.edu>

UCI DREAM Lab

05.5.4 12

TMO Support Middleware on Windows XP & CE -- TMO SM / XP or CE / Socket

Currently running



05.5.4 | 13



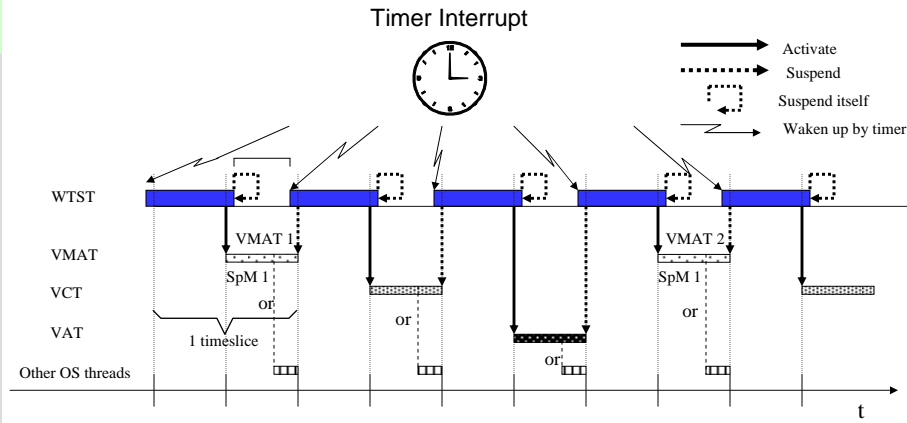
TMO SM -- Thread Structure (cont.)

- WTST (Watchdog Timer & Scheduler Thread): *Super-Micro-Thread*
 - Manages the scheduling / activation of all other threads in TMO SM and checks if there are deadline violations
- VMAT (Virtual Machine for Main Application Threads)
 - A virtual machine representing all application threads including:
 - SpM threads
 - SvM threads
- VCT (Virtual Machine for Communication Threads)
 - Distributes messages coming through the communication network to their destination threads
- VAT (Virtual Machine for Auxiliary Threads)
 - A virtual machine representing all threads managing local I/O activities such as serial character I/O and disk I/O

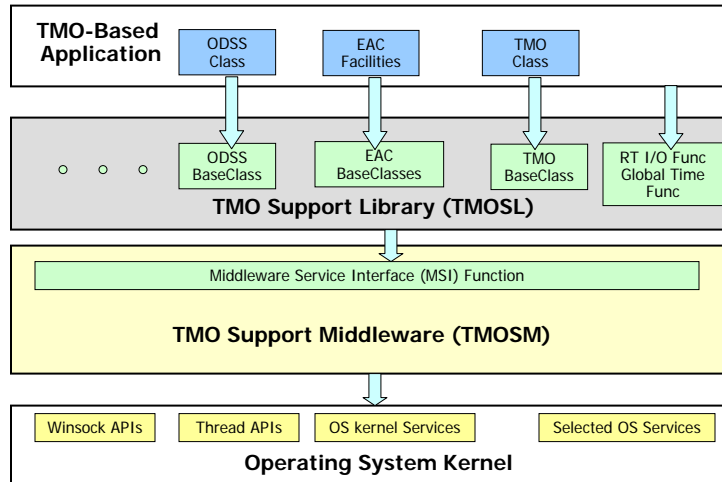
05.5.4 | 14



TMOSM – The Time-slicing Scheme



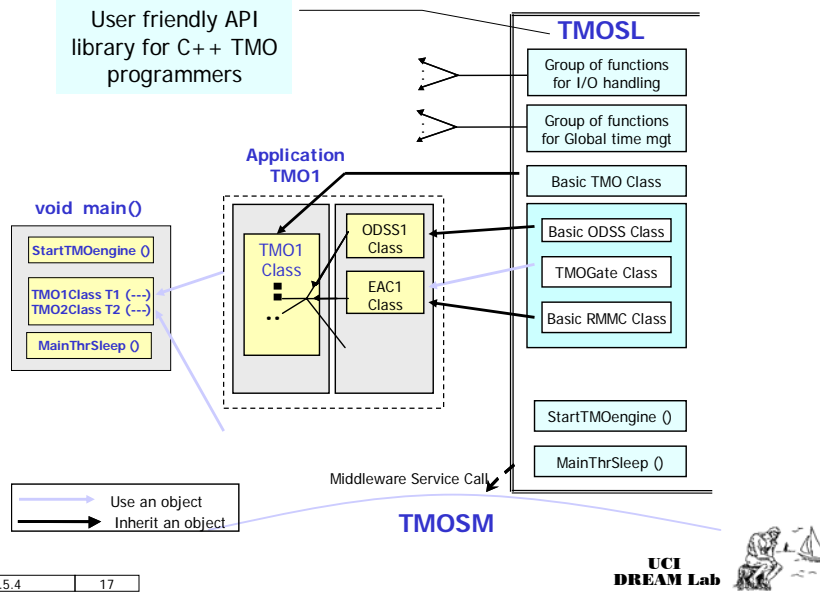
05.5.4 15



05.5.4 16

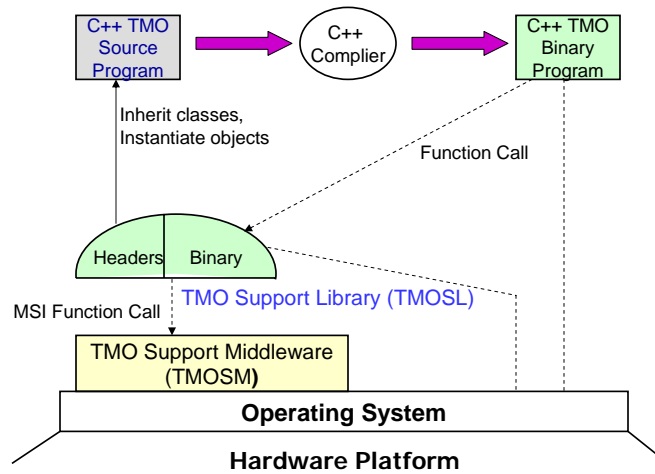


TMOSM Support Library (TMOSL)



05.5.4 | 17

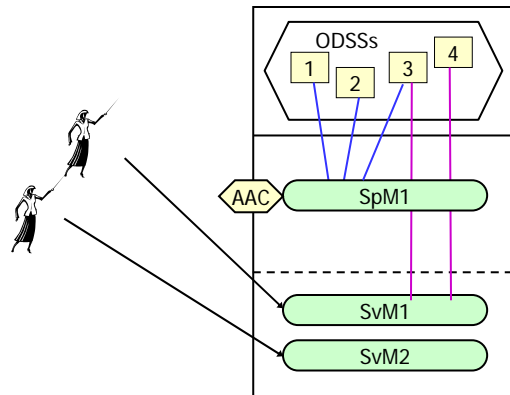
TMO Programming Environment



05.5.4 | 18



Basic Concurrent Constraint (BCC)



- An SvM is allowed to execute only if there is no SpM that requires access to the same object data store segment (ODSS) and will execute in the time window of this SvM.

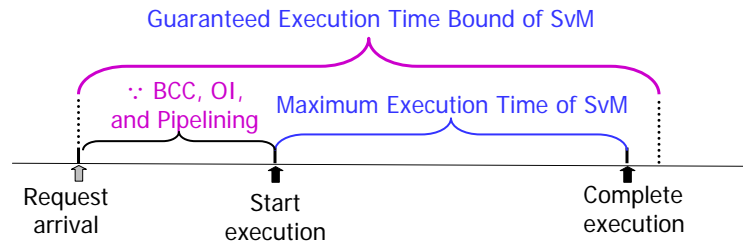
05.5.4	19
--------	----

BCC (cont.)

- An SvM is allowed to execute only if there is no SpM that requires access to the same object data store segment (ODSS) and will execute in the time window of this SvM.
- => SpM executions are given higher priority over SvM executions.
- BCC prevents potential conflicts between SpMs and SvMs and reduces the designer's efforts in [guaranteeing timely service capabilities](#) of TMO.
 - Note that this BCC does [not impose any restriction](#) on concurrent execution of SpMs or concurrent execution of SvMs.

05.5.4	20
--------	----

BCC (cont.)



- In general, the maximum execution time of an SvM depends on how many SvMs and non-conflicting SpMs compete for machine resources

Fig. 1 : Guaranteed Service Time and Maximum Execution Time of SvM

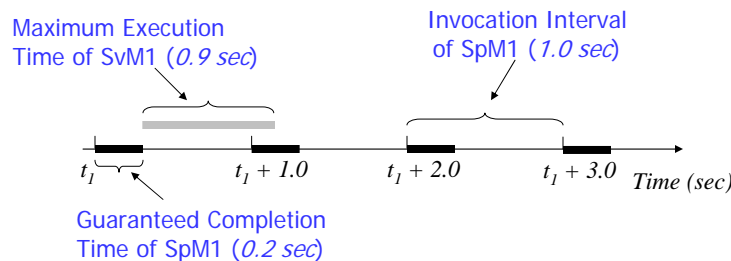
05.5.4 | 21



Potential SvM Starvation Due to BCC

Example: Suppose that SpM1 and SvM1 are accessing the same ODSS. One of the two methods is a writer for the ODSS, the invocation interval of SpM1 is 1 sec., the GCT of SpM1 is 0.2 sec., and the GCT of SvM1 is 0.9 sec.

- Since the Maximum execution time of SvM1 is too long, it cannot be scheduled between the end of one round of SpM1 execution and the beginning of the next round of SpM1 execution.
- => SvM1 is never allowed to execute.
 => SvM1 may be split into several smaller SvMs, and the client-transfer-call is recommended.



05.5.4 | 22



Candidate AAC

No need to use
this in EECS223

- An AAC of SpM can be a **candidate**, which means it is not turned on initially like a permanent AAC.
- A **candidate AAC** can be turned on/off by an SpM or SvM.

Program Example:

```
// An AAC with a name is a candidate AAC, while an AAC without a name is
// a permanent AAC.
// The following three candidate AACs start at 10am, 1pm, and 4pm, respectively,
// and stop after one hour. Depending on the current DCS_age, an SvM decides to
// activate one of them.
```

```
// AAC 1
```

```
from1 = 10 * 60 * 60;    from1 *= 1000 * 1000;
until1 = 11 * 60 * 60;  until1 *= 1000 * 1000;
every1 = (MicroSec)5 * 60 * 1000 * 1000;
est1 = 0;               lst1 = 5 * 100 * 1000;
by1 = 100 * 100 * 1000;
```

```
AAC AAC1 ( _T("Candidate_AAC1"), tm4_DCS_age(from1), tm4_DCS_age(until1),
every1, est1, lst1, by1 );
```

05.5.4

23

UCI
DREAM Lab



05.5.4

24

UCI
DREAM Lab



Program Example using Candidate AACs

```

TMO1::TMO1 (TCHAR* TMO_external_name, tms TMO_start_time,
           AAC & AAC1, AAC & AAC2, AAC & AAC3)
{
    ...
    // register SpM
    spm_spec.build_regist_info_AAC (AAC1);
    spm_spec.build_regist_info_AAC (AAC2);
    spm_spec.build_regist_info_AAC (AAC3);

    // This function call may be repeated if there are
    // than one ODSSs to be associated with this SpM.
    spm_spec.build_regist_info_ODSS (ODSS1.GetId (), RW);
    RegisterSpM ( (PFSpMBody) SpM1, & spm_spec);
    ...
    // register TMO
    RegisterTMO (...);
}

```

05.5.4

25

UCI
DREAM Lab

Program Example using Conditional AAC

```

// SpM1 downloads one big file from a server
void TMO1::SpM1 ( )
{
    ---
    // Retrieve one big file from a server
    Retrieve (...);
    ---
}

// User request will invoke the execution of SvM1, which in turn activates one of
// the AACs of SpM1 according to current DCS age.
Void TMO1::SvM1 ()
{
    __int64 CurrentTime = GetCurrentDCSAge ( );

    if ( CurrentTime < 10 * 3600)
        activate_AAC (_T("Candidate_AAC1"), tm4_DCS_age(CurrentTime));
    else if ( CurrentTime < 13 * 3600)
        activate_AAC (_T("Candidate_AAC2"), tm4_DCS_age(CurrentTime));
    else if ( CurrentTime < 16 * 3600)
        activate_AAC (_T("Candidate_AAC3"), tm4_DCS_age(CurrentTime));
    ---
}

```

05.5.4

26

UCI
DREAM Lab

Ordered Isolation (OI) Rule

- Initiation timestamp (I-timestamp)
 - In the case of an SpM execution, the *I-timestamp* is defined as the record of the time instant at which the SpM execution was initiated according to the AAC specification of the SpM.
 - In the case of an SvM execution, the *I-timestamp* is defined as the record of the time instant at which the execution engine initiated the SvM execution after receiving the client request for the SvM execution and ensuring that the SvM execution can be initiated without violating the BCC and other execution rules.
- Ordered isolation rule
 - A method execution with an older I-timestamp must not be waiting for the release of an ODSS held by a method execution with a younger I-timestamp.
 - In addition, a method execution may never be rolled back due to an ODSS conflict.

05.5.4

27

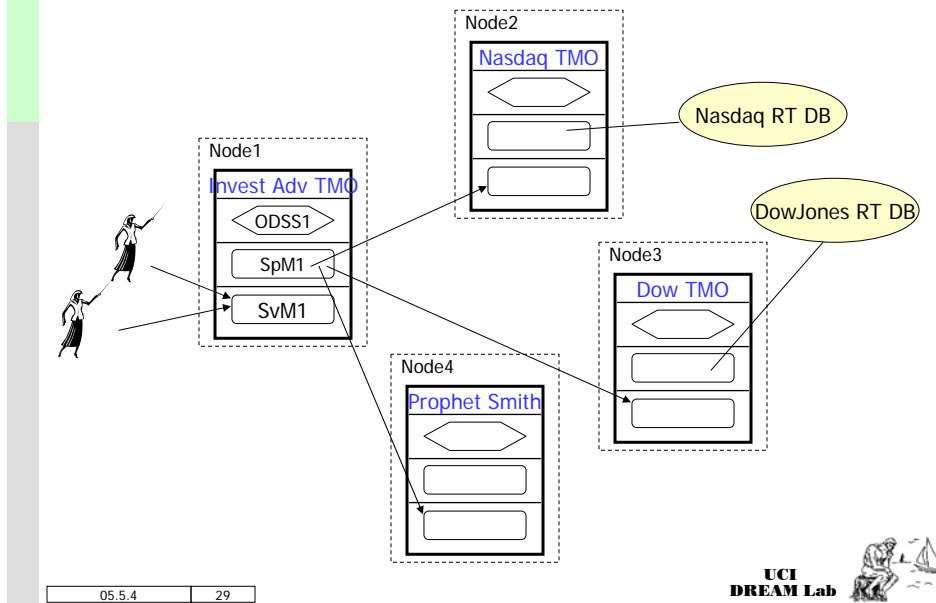
ICT time-stamp

- The term “initiation time-stamp” was already introduced in the previous slide.
- However, we need an additional time-stamp which is associated with each initiation candidate (i.e., method execution), can be used as a priority number, and leads to issuing such proper I-timestamps satisfying the OI rule. This additional time-stamp is called the [init-cand-ticket \(ICT\) time-stamp](#).
- The ICT-timestamp of an SpM is *LST* of the SpM.
- The ICT-timestamp of an SvM is the moment at which the SR is recognized by *VCT* of the server node.

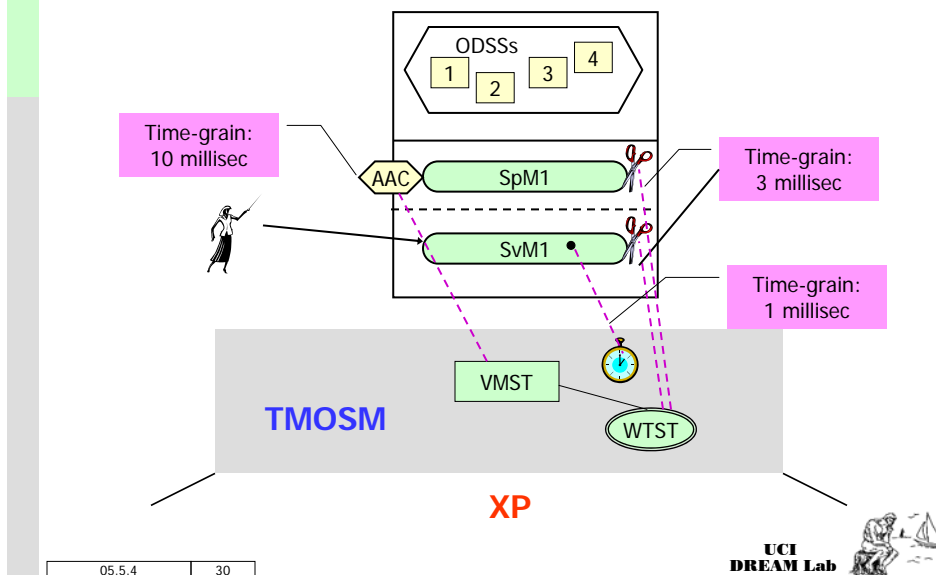
05.5.4

28

An example of a TMO network

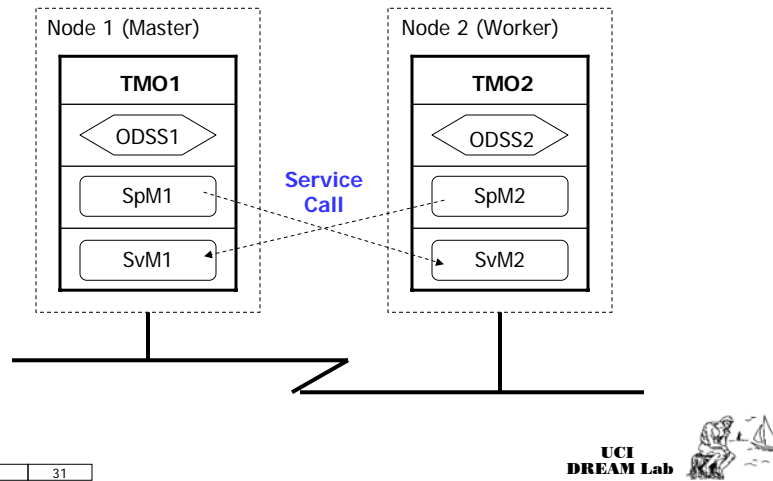


Time-Grains to be used in TMO Programming with TMOSM/XP



A complete example of TMO program

An example of TMO program
(running on two nodes)



05.5.4 31



A complete example of TMO program

Description:

TMO1:

- ODSS1: Maintain a Counter, initial value is 0.
- SpM1: Get ODSS1 Counter's current value and print it.
Get current System Age and print it.
Call SvM2 of TMO2 to get TMO2's System Age and print it out together with local System Age.
Increment the counter.
- SvM1: Return local System Age.

TMO2:

- ODSS2: Maintain a Counter, initial value is 0.
- SpM2: Get ODSS2 Counter's current value and print it.
Get current System Age and print it.
Call SvM1 of TMO1 to get TMO1's System Age and print it out together with local System Age.
Increment the counter.
- SvM2: Return local System Age.

05.5.4 32



A complete example of TMO program

```
// TestTMO.h

#include "TMOSL.h"

using namespace TMO;

typedef struct {
    __int64      system_age;
} ParamStruct_SvM;

class ODSSClass1: public ODSSBaseClass<ODSSClass1>
{
public:
    int  Count;
    ODSSClass1 () {Count = 0; }
};
```

05.5.4

33

UCI
DREAM Lab

A complete example of TMO program (TMO1)

```
////////////////////////////////////
// TMO1 class definition
////////////////////////////////////

class TMO1: public CTMOBase
{
public:
    TMO1 (TCHAR *, tms);
private:
    ODSSClass1      ODSS1;
    SvMGateClass    gate1;
    int             SpM1 ();
    int             SvM1 (ParamStruct_SvM  *);
};
```

05.5.4

34

UCI
DREAM Lab

A complete example of TMO program (TMO1)

```
// TestTMO1.cpp
#include "TestTMO.h"

////////////////////////////////////
// TMO1 class body
////////////////////////////////////
int TMO1::SpM1 ()
{
    int cur_count = ODSS1.Count;
    TMOSLprintf (_T("TMO1::SpM1:  current count is %d\n"), cur_count);
    ParamStruct_SvM param1;
    __int64 issue_tmp = param1.system_age = GetCurrentDCSage ();
    TMOSLprintf (_T("TMO1::SpM1:  current system age is %l64d\n"),
                issue_tmp);

    MicroSec deadline = 50* 1000;
```

05.5.4	35
--------	----



A complete example of TMO program (TMO1)

```
gate1.BlockingSR (&param1, sizeof(param1), deadline);
TMOSLprintf (_T("TMO1::SpM1 issues request: %l64d, gets the server
age: %l64d \n"), issue_tmp, param1.system_age);
ODSS1.Count = ++cur_count;
return 1;
}
int TMO1::SvM1 (ParamStruct_SvM * pReq)
{
    // Get the timestamp when the SR is issued at the client side
    MicroSec Timestamp = GetSRTimestamp();
    pReq->system_age = GetCurrentDCSage ();
    return 1;
}
```

05.5.4	36
--------	----



A complete example of TMO program (TMO1)

```

TMO1::TMO1 (TCHAR * TMO_name, tms TMO_start_time1) :
    gate1 ( _T("TMO2"), _T("SvM2"), tm4_DCS_age (1*1000*1000) )
        // initialization list
{
    // register SvM
    SvM_RegistParam  svm_spec;
    svm_spec.GETB = 300 * 1000;
    _tcscopy (svm_spec.name, _T("SvM1"));
    svm_spec.build_regist_info_ODSS (ODSS1.GetId (), RW);
    RegisterSvM ( (PFSvMBody) SvM1, & svm_spec);

```

05.5.4

37

UCI
DREAM Lab

A complete example of TMO program (TMO1)

```

// register SpM
AAC aac1 ( NULL,
    tm4_DCS_age (1 * 1000 * 1000),
    tm4_DCS_age ( (MicroSec) 60 * 1000 * 1000),
    1 * 1000 * 1000,
    0,
    100 * 1000,
    200 * 1000 );

SpM_RegistParam  spm_spec;
spm_spec.build_regist_info_AAC (aac1);
spm_spec.build_regist_info_ODSS (ODSS1.GetId(), RW);
spm_spec.build_regist_info_ODSS (gate1.GetId(), RW);
RegisterSpM ((PFSpMBody) SpM1, & spm_spec);

```

05.5.4

38

UCI
DREAM Lab

A complete example of TMO program (TMO1)

```

// register TMO
TMO_RegistParam tmo_spec;
_tcscopy (tmo_spec.global_name, TMO_name);
tmo_spec.start_time = TMO_start_time1;
RegisterTMO (& tmo_spec);
}

```

05.5.4

39

UCI
DREAM Lab

A complete example of TMO program (TMO1)

```

// TestTMO1main.cpp
#include "TestTMO.h"

void main()
{
    StartTMOengine();

    tms TMO_start_time1 = tm4_DCS_age (2 * 1000 * 1000);

    TMO1 T1 (_T("TMO1"), TMO_start_time1);

    MainThrSleep();
}

```

05.5.4

40

UCI
DREAM Lab

config.ini for Node1 (Master node)

```
[master_node]
ip_addr = 128.195.164.144
#
[tncm]
num_of_LAN_devices = 1
local_ip = 128.195.164.144
protocol = UDP
port = 4041
num_of_DC_node = 2
#
[clock_sync]
protocol = UDP
port = 4051
#
[VM_exec_order]
order = 1, 2, 3
time_slot_length = 3000
```

05.5.4

41

UCI
DREAM Lab



config.ini for Node1 (cont.)

```
[rmmc]
protocol = PUDP
port = 4061
#
[vmst]
default_tmo_comm_port_num = 3322
protocol = PUDP
#
[rmc]
num_of_link = 2
(128.195.164.144, 128.195.164.144) = 0
(128.195.164.144, 128.195.164.155) = 0
```

05.5.4

42

UCI
DREAM Lab



config.ini for Node1 (cont.)

```
#####  
#  
#                               Mwconfig.ini  
#####  
#  
#  
[mw_config_param]  
#  
##### Thread Pool  
#####  
#  
# [1]  
id = S_VMST_THREAD_POOL_SIZE  
type = integer  
value = 5  
#  
# [2]  
id = S_VMMCT_THREAD_POOL_SIZE  
type = integer  
value = 2  
#  
# [3]  
id = S_VIST_THREAD_POOL_SIZE  
type = integer  
value = 2  
#
```

05.5.4	43
--------	----



config.ini for Node1 (cont.)

```
##### RMMC  
#####  
# [1]  
id = S_MAX_TMO_PER_NODE  
type = integer  
value = 5  
#  
#  
# [2]  
id = S_MAX_STATE_MSG_PER_QUEUE  
type = integer  
value = 10  
#  
#  
# [3]  
id = S_MAX_TMO_PER_RMMC  
type = integer  
value = 5  
#  
#  
# [4]  
id = S_MAX_NODE_PER_RMMC  
type = integer  
value = 5  
#  
#
```

05.5.4	44
--------	----



config.ini for Node1 (cont.)

```
# [5]
id = S_MAX_STATE_MSG_PER_RMMC
type = integer
value = 3
#
#
# [6]
id = S_MAX_EVENT_MSG_PER_RMMC
type = integer
value = 30
#
#
# [7]
id = S_MAX_RMMC_IN_SYSTEM
type = Integer
value = 2
#
##### Gate
#####
# [11]
id = S_MAX_GATE_IN_SYSTEM
type = integer
value = 2
#
##### VMMCT
#####
```

05.5.4	45
--------	----

**UCI
DREAM Lab**



config.ini for Node1 (cont.)

```
# [12]
id = S_MAX_COMM_DEVICE_PER_NODE
type = integer
value = 2
#
#
# [3]
id = S_MAX_CONNECTION_PER_DEVICE
type = integer
value = 10
#
#
# [3]
id = S_MAX_OUTGOING_PACKET_PER_CONNECTION
type = integer
value = 20
#
#
# [3]
id = S_MAX_CALLBACK_PER_CONNECTION
type = integer
value = 5
#
#
# [4]
id = S_MAX_ITEM_IN_RECV_WINDOW
```

05.5.4	46
--------	----

**UCI
DREAM Lab**



config.ini for Node1 (cont.)

```
type = integer
value = 50
#
#
##### VIST
#####
# [12]
id = S_MAX_AAC_PER_IIT
type = integer
value = 2
#
##### Topology
#####
# [3]
id = S_MAX_CHILD_PER_NODE
type = integer
value = 5
#
#
# [3]
id = S_MAX_SIBLING_PER_NODE
type = integer
value = 5
#
##### TNCM
#####
# [3]
```

05.5.4	47
--------	----



config.ini for Node1 (cont.)

```
id = S_MAX_NODE_NUM_IN_SYSTEM
type = integer
value = 5
#
#
# [3]
id = S_MAX_TMO_NUM_IN_SYSTEM
type = integer
value = 10
#
#
##### MCB
#####
# [3]
id = S_MAX_SPM_PER_TMO
type = Integer
value = 5
#
#
# [3]
id = S_MAX_SVM_PER_TMO
type = integer
value = 5
#
#
# [3]
```

05.5.4	48
--------	----



config.ini for Node1 (cont.)

```
id = S_MAX_ODSS_PER_TMO
type = integer
value = 5
#
#
# [3]
id = S_MAX_ODSS_ACCESSED_PER_METHOD
type = integer
value = 5
#
# [3]
id = S_MAX_AAC_PER_SPM
type = integer
value = 2
#
#
# [3]
id = S_MAX_ITEM_IN_TMO_TABLE
type = integer
value = 10
#
#
# [3]
id = S_MAX_ITEM_IN_SPM_TABLE
type = integer
value = 20
```

05.5.4	49
--------	----



config.ini for Node1 (cont.)

```
#
#[3]
id = S_MAX_ITEM_IN_SVM_TABLE
type = integer
value = 20
#
#
# [3]
id = S_MAX_WAITING_WRITER_PER_ODSS
type = integer
value = 10
#
#
# [3]
id = S_MAX_WAITING_READER_PER_ODSS
type = integer
value = 10
#
##### VMST
#####
# [3]
id = S_MAX_ITEM_IN_SPM_RESERVATIONQ
type = integer
value = 25
#
```

05.5.4	50
--------	----



config.ini for Node1 (cont.)

```
#  
# [3]  
id = S_MAX_ITEM_IN_ARRIVED_SVMQ  
type = integer  
value = 10  
#  
# [3]  
id = S_MAX_ITEM_IN_SERVICE_RESULTQ  
type = integer  
value = 10  
#  
# [3]  
id = S_MAX_ITEM_IN_SERVICE_STUBQ  
type = integer  
value = 10  
#  
# [3]  
id = S_MAX_ITEM_IN_READYQ  
type = integer  
value = 20  
#  
# [3]  
id = S_MAX_ITEM_IN_SLEEPQ
```

05.5.4	51
--------	----



config.ini for Node1 (cont.)

```
type = integer  
value = 20  
#  
# [3]  
id = S_MAX_ITEM_IN_BLOCKED_FOR_MSGQ  
type = integer  
value = 20  
#  
# [3]  
id = S_MAX_ITEM_IN_BLOCKED_FOR_IOQ  
type = integer  
value = 20  
#  
##### Misc  
#####  
# [3]  
id = S_MAX_EVENT_IN_EVENTMGR  
type = integer  
value = 20  
#  
# [3]  
id = S_MAX_SUBSCRIBER_PER_EVENT  
type = integer
```

05.5.4	52
--------	----



config.ini for Node1 (cont.)

```
value = 10
#
#
# [3]
id = S_MAX_TIMER_SUBSCRIBER
type = integer
value = 10
#
#
# [3]
id = S_MAX_NUM_MW_THREAD
type = integer
value = 10
#
##### RMP #####
id = S_MAX_RMPSPECLIST
type = integer
value = 10
#
id = S_MAX_RECENT_MSG_LIST
type = integer
value = 20
#
id = S_MAX_INORDER_DELIVERY_LIST
type = integer
value = 20
```

05.5.4	53
--------	----



config.ini for Node1 (cont.)

```
#
id = S_TIMEOUT_LIST
type = integer
value = 20
#
id = S_UNACKD_MSG_LIST
type = integer
value = 20
#
id = S_MAX_PROTOCOL_LIST
type = integer
value = 10
```

05.5.4	54
--------	----



A complete example of TMO program (TMO2)

```
// TestTMO.h (cont.)

////////////////////////////////////
// TMO2 class definition
////////////////////////////////////
class TMO2: public CTMOBase
{
public:
    TMO2 (TCHAR *, tms);
private:
    ODSSClass1          ODSS2;
    SvMGateClass        gate2;
    int                 SpM2 ();
    int                 SvM2 (ParamStruct_SvM *);
};
```

05.5.4

55

UCI
DREAM Lab

A complete example of TMO program (TMO2)

```
// TestTMO2.cpp
#include "TestTMO.h"

// TMO2 class body
int TMO2::SpM2 ()
{
    int cur_count = ODSS2.Count;
    TMOSLprintf (_T("TMO2::SpM2: current count is %d\n"), cur_count);

    ParamStruct_SvM param2;
    __int64 issue_tmp = param2.system_age = GetCurrentDCSage ();
    TMOSLprintf (_T("TMO2::SpM2: current system age is %I64d\n"),
        issue_tmp);

    MicroSec deadline = 50 * 1000;
    gate2.BlockingSR (&param2, sizeof (param2), deadline,
        tm4_DCS_age ( GetCurrentDCSage () + 50 * 1000 ));
}
```

05.5.4

56

UCI
DREAM Lab

A complete example of TMO program (TMO2)

```
TMOSLprintf (_T("TMO2::SpM2 issues request: %l64d, gets
the server age: %l64d \n"), issue_tmp, param2.system_age);
```

```
    ODSS2.Count = ++cur_count;
    return 1;
}
int TMO2::SvM2 (ParamStruct_SvM * pReq)
{
    // Get the timestamp that the SR is issued at the client side
    MicroSec Timestamp = GetSRTimestamp ();
    pReq->system_age = GetCurrentDCSage ();
    return 1;
}
```

05.5.4

57

UCI
DREAM Lab

A complete example of TMO program (TMO2)

```
TMO2::TMO2 (TCHAR * TMO_name, tms TMO_start_time2):
    gate2 (_T("TMO1"), _T("SvM1"), tm4_DCS_age (1*1000*1000))
{
    // register SvM
    SvM_RegistParam svm_spec;
    svm_spec.GETB = 300 * 1000;
    _tcscopy (svm_spec.name, _T("SvM2"));
    svm_spec.build_regist_info_ODSS (ODSS2.GetId(), RW);
    RegisterSvM ( (PFSvMBody)SvM2, &svm_spec);
    // register SpM
    AAC aac2 ( NULL,
        tm4_DCS_age (1 * 1000 * 1000),
        tm4_DCS_age ((MicroSec)60 * 1000 * 1000),
        1 * 1000 * 1000,
        300 * 1000,
        400 * 1000,
        500 * 1000);
```

05.5.4

58

UCI
DREAM Lab

A complete example of TMO program (TMO2)

```

SpM_RegistParam spm_spec;
spm_spec.build_regist_info_AAC (aac2);
spm_spec.build_regist_info_ODSS (ODSS2.GetId(), RW);
spm_spec.build_regist_info_ODSS (gate2.GetId(), RW);
RegisterSpM ( (PFSpMBody)SpM2, &spm_spec);

// register TMO
TMO_RegistParam tmo_spec;
_tcscpy (tmo_spec.global_name, TMO_name);
tmo_spec.start_time = TMO_start_time2;
RegisterTMO (& tmo_spec);
}

```

05.5.4

59

UCI
DREAM Lab

A complete example of TMO program (TMO2)

```

// TestTMO2main.cpp
#include "TestTMO.h"

void main()
{
    StartTMOengine ();

    tms TMO_start_time2 = tm4_DCS_age (2*1000*1000);
    TMO2 T2 (_T("TMO2"), TMO_start_time2);

    MainThrSleep ();
}

```

05.5.4

60

UCI
DREAM Lab

config.ini for Node2 (Worker node)

```
[master_node]
ip_addr = 128.195.164.144
#
[tncm]
num_of_LAN_devices = 1
local_ip = 128.195.164.155
protocol = UDP
port = 4041
num_of_DC_nodes = 2
#
[clock_sync]
protocol = UDP
port = 4051
#
[VM_exec_order]
order = 1, 2, 3
time_slot_length = 3000
```

05.5.4

61

UCI
DREAM Lab



config.ini for Node2 (cont.)

```
[rmmc]
protocol = PUDP
port = 4061
#
[vmst]
default_tmo_comm_port_num = 3322
protocol = PUDP
#
[rmc]
num_of_link = 2
(128.195.164.155, 128.195.164.155) = 0
(128.195.164.155, 128.195.164.144) = 0
```

05.5.4

62

UCI
DREAM Lab



config.ini for Node2 (cont.)

```
#####  
#####  
#                               Mwconfig.ini  
#####  
#####  
#  
[mw_config_param]  
#  
##### Thread Pool  
#####  
#  
# [1]  
id = S_VMST_THREAD_POOL_SIZE  
type = integer  
value = 5  
#  
# [2]  
id = S_VMMCT_THREAD_POOL_SIZE  
type = integer  
value = 2  
#  
# [3]  
id = S_VIST_THREAD_POOL_SIZE  
type = integer  
value = 2  
#
```

05.5.4	63
--------	----



config.ini for Node2 (cont.)

```
##### RMMC  
#####  
# [1]  
id = S_MAX_TMO_PER_NODE  
type = integer  
value = 5  
#  
# [2]  
id = S_MAX_STATE_MSG_PER_QUEUE  
type = integer  
value = 10  
#  
# [3]  
id = S_MAX_TMO_PER_RMMC  
type = integer  
value = 5  
#  
# [4]  
id = S_MAX_NODE_PER_RMMC  
type = integer  
value = 5  
#  
#
```

05.5.4	64
--------	----



config.ini for Node2 (cont.)

```
# [5]
id = S_MAX_STATE_MSG_PER_RMMC
type = integer
value = 3
#
#
# [6]
id = S_MAX_EVENT_MSG_PER_RMMC
type = integer
value = 30
#
#
# [7]
id = S_MAX_RMMC_IN_SYSTEM
type = integer
value = 2
#
##### Gate
#####
# [11]
id = S_MAX_GATE_IN_SYSTEM
type = integer
value = 2
#
##### VMCT
#####
```

05.5.4	65
--------	----

**UCI
DREAM Lab**



config.ini for Node2 (cont.)

```
# [12]
id = S_MAX_COMM_DEVICE_PER_NODE
type = integer
value = 2
#
#
# [3]
id = S_MAX_CONNECTION_PER_DEVICE
type = integer
value = 10
#
#
# [3]
id = S_MAX_OUTGOING_PACKET_PER_CONNECTION
type = integer
value = 20
#
#
# [3]
id = S_MAX_CALLBACK_PER_CONNECTION
type = integer
value = 5
#
#
# [4]
id = S_MAX_ITEM_IN_RECV_WINDOW
```

05.5.4	66
--------	----

**UCI
DREAM Lab**



config.ini for Node2 (cont.)

```
type = integer
value = 50
#
#
##### VIST
#####
# [12]
id = S_MAX_AAC_PER_IIT
type = integer
value = 2
#
##### Topology
#####
# [3]
id = S_MAX_CHILD_PER_NODE
type = integer
value = 5
#
#
# [3]
id = S_MAX_SIBLING_PER_NODE
type = integer
value = 5
#
##### TNCM
#####
# [3]
```

05.5.4	67
--------	----

UCI
DREAM Lab



config.ini for Node2 (cont.)

```
id = S_MAX_NODE_NUM_IN_SYSTEM
type = integer
value = 5
#
#
# [3]
id = S_MAX_TMO_NUM_IN_SYSTEM
type = integer
value = 10
#
#
##### MCB
#####
# [3]
id = S_MAX_SPM_PER_TMO
type = Integer
value = 5
#
#
# [3]
id = S_MAX_SVM_PER_TMO
type = integer
value = 5
#
#
# [3]
```

05.5.4	68
--------	----

UCI
DREAM Lab



config.ini for Node2 (cont.)

```
id = S_MAX_ODSS_PER_TMO
type = integer
value = 5
#
#
# [3]
id = S_MAX_ODSS_ACCESSED_PER_METHOD
type = integer
value = 5
#
# [3]
id = S_MAX_AAC_PER_SPM
type = integer
value = 2
#
#
# [3]
id = S_MAX_ITEM_IN_TMO_TABLE
type = integer
value = 10
#
#
# [3]
id = S_MAX_ITEM_IN_SPM_TABLE
type = integer
value = 20
```

05.5.4	69
--------	----



config.ini for Node2 (cont.)

```
#
#[3]
id = S_MAX_ITEM_IN_SVM_TABLE
type = integer
value = 20
#
#
# [3]
id = S_MAX_WAITING_WRITER_PER_ODSS
type = integer
value = 10
#
#
# [3]
id = S_MAX_WAITING_READER_PER_ODSS
type = integer
value = 10
#
#
##### VMST
#####
# [3]
id = S_MAX_ITEM_IN_SPM_RESERVATIONQ
type = integer
value = 25
#
```

05.5.4	70
--------	----



config.ini for Node2 (cont.)

```
#  
# [3]  
id = S_MAX_ITEM_IN_ARRIVED_SVMQ  
type = integer  
value = 10  
#  
#  
# [3]  
id = S_MAX_ITEM_IN_SERVICE_RESULTQ  
type = integer  
value = 10  
#  
#  
# [3]  
id = S_MAX_ITEM_IN_SERVICE_STUBQ  
type = integer  
value = 10  
#  
# [3]  
id = S_MAX_ITEM_IN_READYQ  
type = integer  
value = 20  
#  
#  
# [3]  
id = S_MAX_ITEM_IN_SLEEPQ
```

05.5.4	71
--------	----



config.ini for Node2 (cont.)

```
type = integer  
value = 20  
#  
#  
# [3]  
id = S_MAX_ITEM_IN_BLOCKED_FOR_MSGQ  
type = integer  
value = 20  
#  
# [3]  
id = S_MAX_ITEM_IN_BLOCKED_FOR_IOQ  
type = integer  
value = 20  
#  
#  
##### Misc  
#####  
# [3]  
id = S_MAX_EVENT_IN_EVENTMGR  
type = integer  
value = 20  
#  
#  
# [3]  
id = S_MAX_SUBSCRIBER_PER_EVENT  
type = integer
```

05.5.4	72
--------	----



config.ini for Node2 (cont.)

```
value = 10
#
#
# [3]
id = S_MAX_TIMER_SUBSCRIBER
type = integer
value = 10
#
#
# [3]
id = S_MAX_NUM_MW_THREAD
type = integer
value = 10
#
##### RMP #####
id = S_MAX_RMPSPECLIST
type = integer
value = 10
#
id = S_MAX_RECENT_MSG_LIST
type = integer
value = 20
#
id = S_MAX_INORDER_DELIVERY_LIST
type = integer
value = 20
```

05.5.4	73
--------	----



config.ini for Node2 (cont.)

```
#
id = S_TIMEOUT_LIST
type = integer
value = 20
#
id = S_UNACKD_MSG_LIST
type = integer
value = 20
#
id = S_MAX_PROTOCOL_LIST
type = integer
value = 10
```

05.5.4	74
--------	----

