

Документ подписан простой электронной подписью

Информация о владельце:

ФИО: Емельянов Сергей Геннадьевич

Должность: ректор

Дата подписания: 18.02.2023 15:10:22

Уникальный программный ключ:

9ba7d3e34c012eba476ffd2d064cf2781957ba730df3374d16ff3c0ce536f0fc6

МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное

образовательное учреждение высшего образования

«Юго-Западный государственный университет»

(ЮЗГУ)

Кафедра программной инженерии

УТВЕРЖДАЮ

Проректор по учебной работе

О.Г. Локтионова

« 25 » 10

2022 г.



## СИСТЕМЫ И ПРИЛОЖЕНИЯ РЕАЛЬНОГО ВРЕМЕНИ

Методические указания для выполнения лабораторных работ  
для студентов направления подготовки 09.04.04 Программная  
инженерия

Курск 2022

УДК 004  
Составитель А.В. Малышев

Рецензент  
к.т.н., Киселев А.В.

Системы и приложения реального времени: методические указания для выполнения лабораторных работ для студентов направления подготовки 09.04.04 Программная инженерия/ Юго-Зап. гос. ун-т; сост.: А.В. Малышев, Курск, 2022. 55 с.

Содержат методические указания для выполнения лабораторных работ.

Методические указания по структуре, содержанию и стилю изложения материала соответствуют методическим и научным требованиям, предъявляемым к учебным и методическим пособиям.

Предназначены для студентов направления подготовки 09.04.04 Программная инженерия.

Текст печатается в авторской редакции

Подписано в печать                                  Формат 60 x 84 1/16.  
Усл. печ. л.      Уч.- изд. л.      Тираж 50 экз. Заказ 2143 Бесплатно.  
Юго-Западный государственный университет.  
305040, Курск, ул. 50 лет Октября, 94.

1		4
2		12
3		22
4		29
5	Ethernet	41
6		50
7		62
8	SCADA	68
9	QNX	75

. , . ( )

. ( , )

.

:

;

**1.**

. , , , , , , . , , , , , , .

:

1.

;

2.

;

3.

.

«

»,

(

)

«

».

,

.

«

»,

(

)

«

».

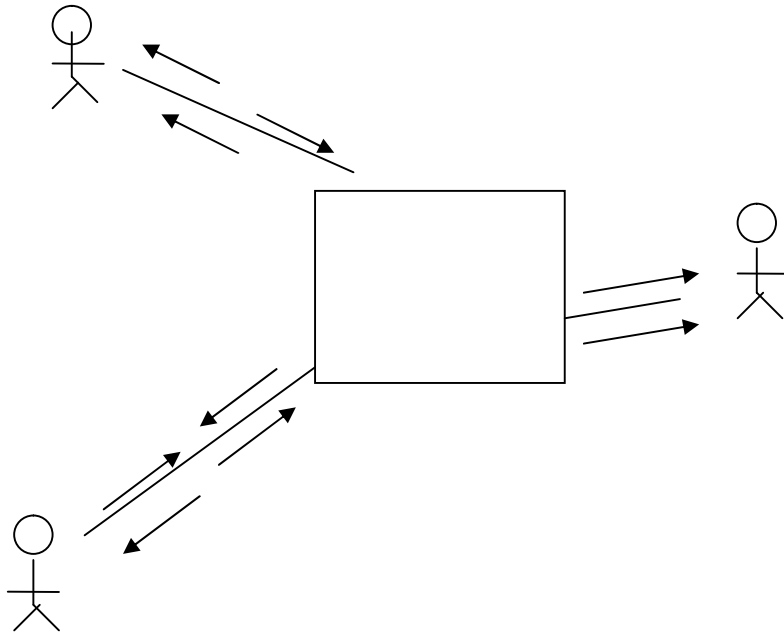
,

«

».

.

.1.



. 1.

2.

. 1

.3.1.

1						1
2						1
3	« / » « »					1
4	« / » « »					1
5						1
6						1
7	« » « »					0.5
8						1
9						1
10						0.25
11						1

3.

,

.

.

,

.

:

- 1.
- 2.
- 3.
- 4.

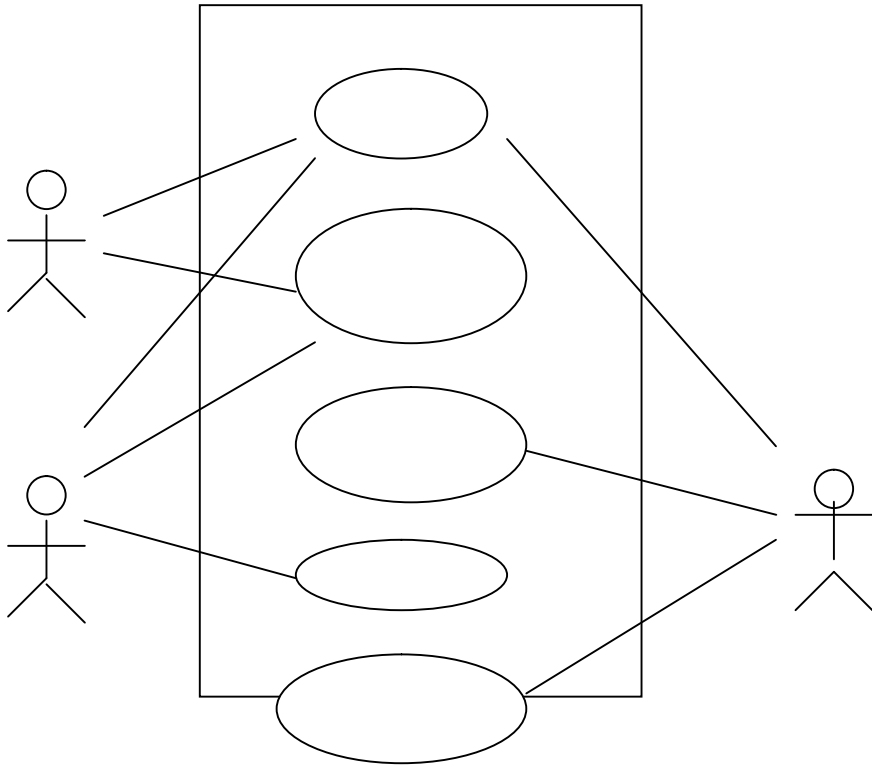
?

?  
?

?

:

- 1.
- 2.



. 2.

4.

1.

2.

3.

. 2

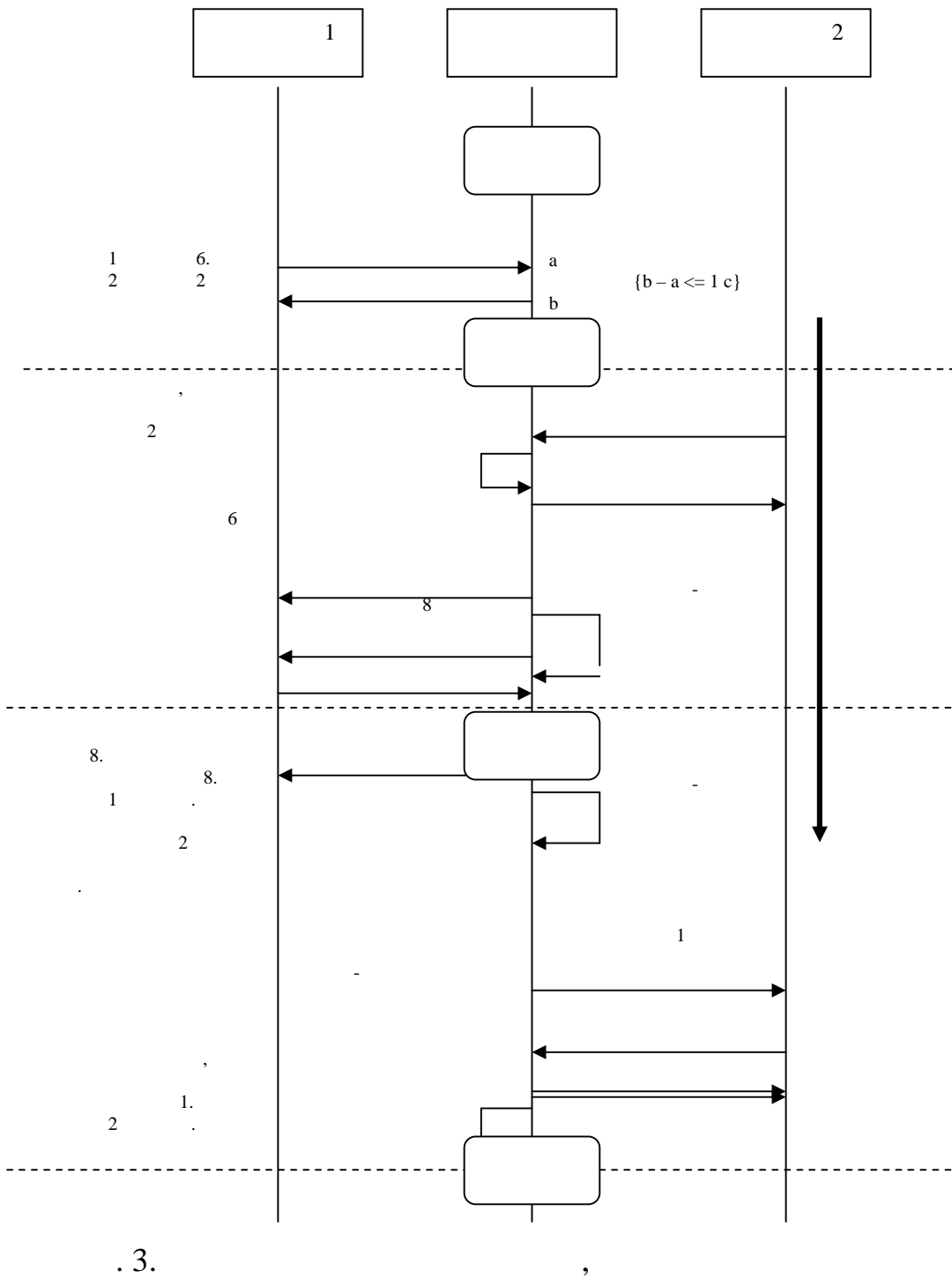
. 2.

	[ 1- ]	
1	6	6
2	[ 2]	
3	2	
4	6	
5	1 8	8
6	8	
7	1	
8		2
9	2	
10	2 1	1
11	1	2
12		

5.

. 3. , {b - a <= 1 c} , b a  
1 . « » « »  
. 3.

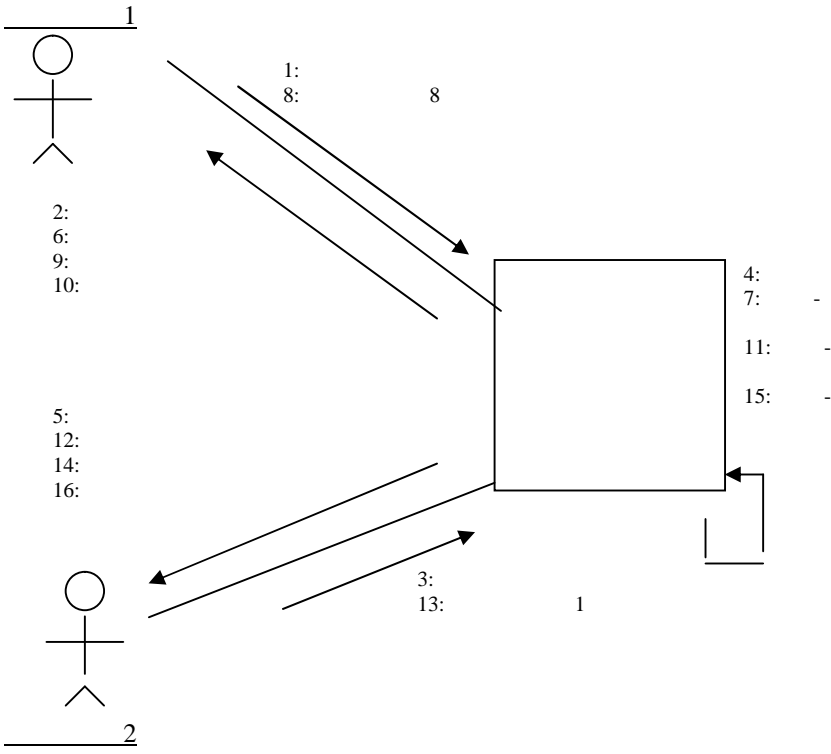




6.

. 3.

. 4.



. 4.

. 3

1	
2	
3	

4	
5	
6	
7	
8	
9	
10	
11	
12	
13	
14	
15	
16	
17	
18	
19	
20	
21	

1. .
2. .
3. , :
- a. ;
- b. ;
- c. ;
- d. .
4. .
5. .

- 1.
- 2.
- 3.
- 4.
- 5.
- 6.

**1.**

.3.

1		
2		
3	( )	
4		
5		
6		
7		
8		

2.

\_\_\_\_\_ , \_\_\_\_\_ .  
\_\_\_\_\_ .  
\_\_\_\_\_ 10 \_\_\_\_\_ .  
\_\_\_\_\_  
\_\_\_\_\_ , \_\_\_\_\_  
\_\_\_\_\_ . \_\_\_\_\_ / \_\_\_\_\_ .  
\_\_\_\_\_ , \_\_\_\_\_ , \_\_\_\_\_  
\_\_\_\_\_ , \_\_\_\_\_ , \_\_\_\_\_  
\_\_\_\_\_ , \_\_\_\_\_ .  
\_\_\_\_\_  
\_\_\_\_\_ , \_\_\_\_\_ .  
\_\_\_\_\_ , \_\_\_\_\_ .  
\_\_\_\_\_ - 5 \_\_\_\_\_ .  
\_\_\_\_\_ . \_\_\_\_\_  
\_\_\_\_\_ - \_\_\_\_\_ .  
\_\_\_\_\_ , \_\_\_\_\_ .  
\_\_\_\_\_ , \_\_\_\_\_  
\_\_\_\_\_ .  
\_\_\_\_\_ , \_\_\_\_\_ , \_\_\_\_\_  
\_\_\_\_\_ , \_\_\_\_\_ .  
\_\_\_\_\_ , \_\_\_\_\_  
\_\_\_\_\_ , \_\_\_\_\_ .  
\_\_\_\_\_ , \_\_\_\_\_  
\_\_\_\_\_ , \_\_\_\_\_  
\_\_\_\_\_ , \_\_\_\_\_  
\_\_\_\_\_ .  
\_\_\_\_\_ , \_\_\_\_\_ , \_\_\_\_\_  
\_\_\_\_\_ , \_\_\_\_\_ .  
\_\_\_\_\_ , \_\_\_\_\_  
\_\_\_\_\_ .

. 4

.4.

	-
	( )

	( )
/	

- :
1. ;
  2. ;
  3. ;
  4. ;
  5. / ;
  6. ;
  7. ;
  8. ;
  9. ;
  10. ;
  11. ;
  12. ;
  13. ;
  14. ;
  15. ;
  16. .

3.

,

.

,

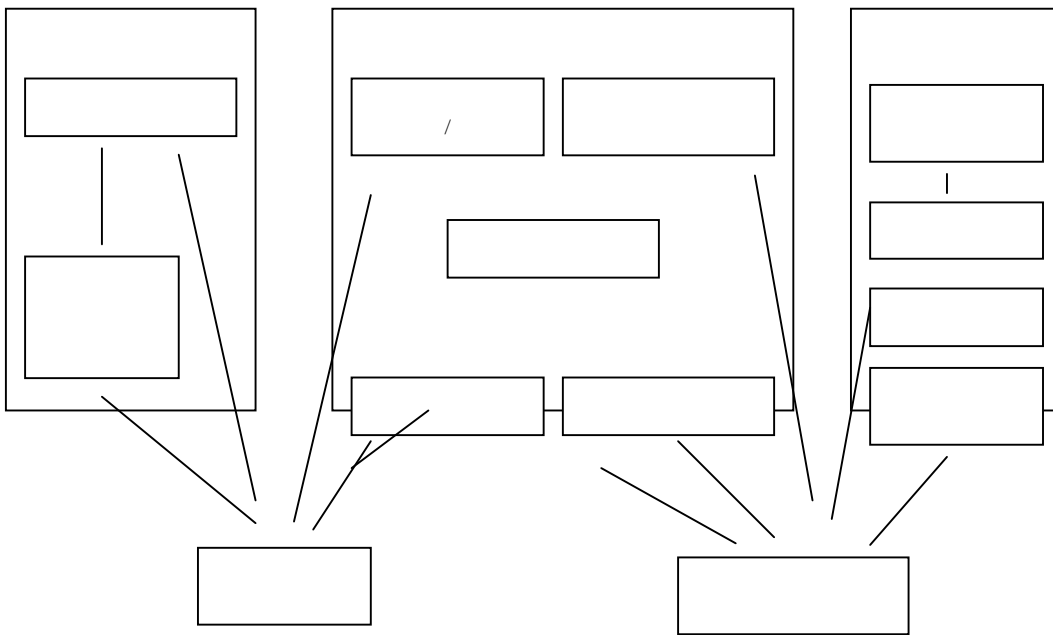
,

.5.

.5.

/		/
/		/

.5.



.5.

4.

1.

2.

3.

?

?

:

?

4. ?  
5. ?

1. ?

2. ?

( , , , ), ( ' ),

3. ?

2.

4. ?

, ), . ( , , ,

5. ?

, ( , , , ,

« ».

1. , ( , , ); :

2. ( , );

3. ;

4. ( — );

5. .

5.

, ,

.

, ,

. ,

.

.

, ,

.

:

1. ;

2. ;

3. ;

4. ;

5. ;

6. ;

7. .

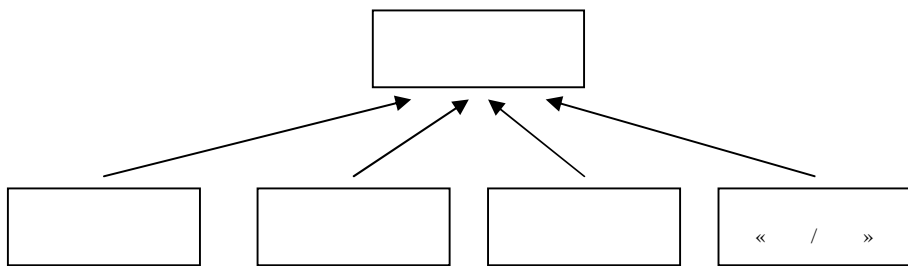


- 8. ;
- 9. ;
- 10. .

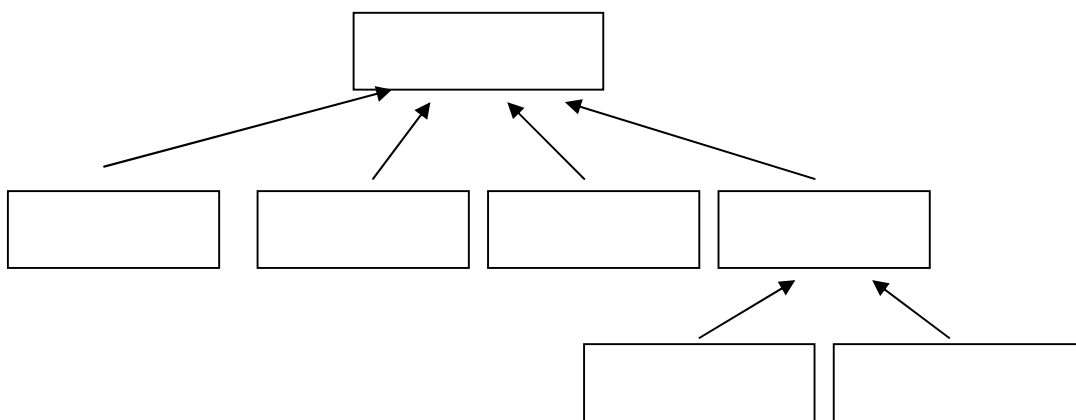
- :
  - 1. ;
  - 2. ;
  - 3. ;
  - 4. .

- :
- 1. ;
- 2. ;
- 3. .
- 4. .

« » « » .6 .7.

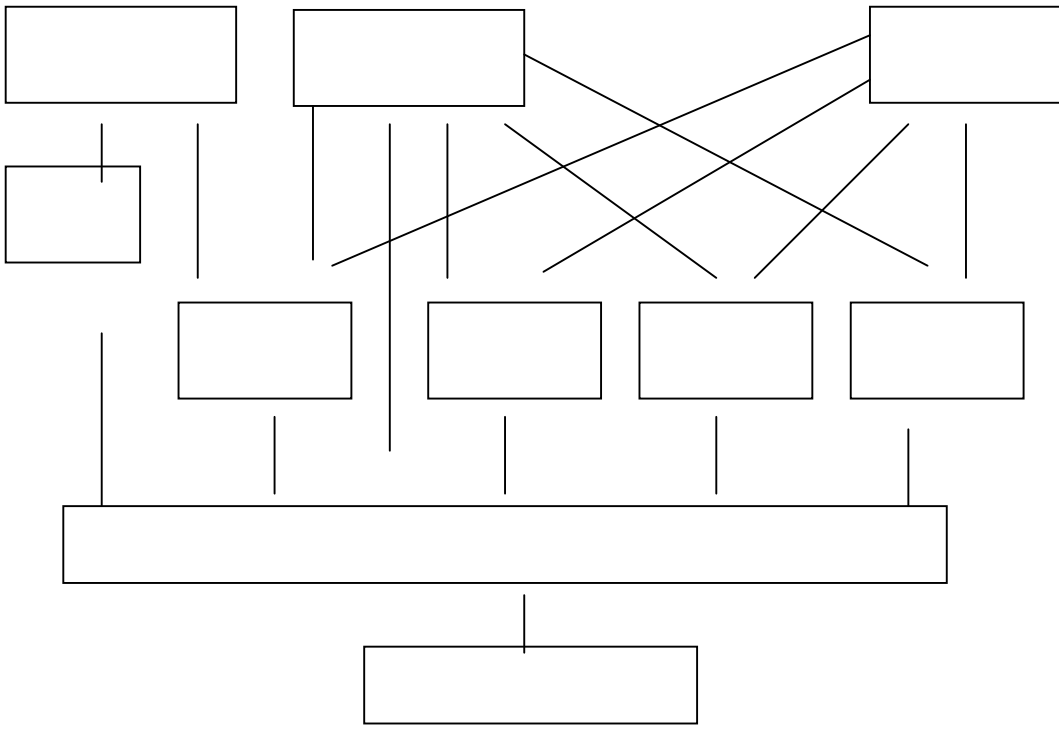


.6 « »



.7 « »

1. « - » - ( , 1 - 1);
2. « - » - « » -  
( , 1 - 1);
3. « - » -  
( , 1 - 1);
4. « - » -  
,  
( , 1 - 1);
5. « - » - « »  
( , - - );
6. « - » - « »  
( , - - );
7. « - » - ( ,  
1 - 1);
8. « - » - « »  
( , 1 - 1);
9. « - » - « »  
( , - - );
10. « - » - « »  
( , - - );
11. « - » - « »  
( , 1 - 1);
12. « - » - « »  
( , - - );
13. « - » - « »  
( , - - );
14. « - » -  
( , 1 - 1);
15. « - » -  
( , 1 - 1);
16. « - » - « »  
, « »  
( , - - );
17. « - » -  
( , - - ).



. 8.

6.

», . «  
 ».  
 . . . . .  
 . , , , , .

1.

2.

3.

a.

b.

c.

d.

e.

4.

5.

- 1.
- 2.
- 3.
- 4.

**1.**

( ) ,

- 1.
- 2.

;

:

,

« » ,

:

- 1.
- 2.
- 3.

;

;

- :
- 1.
- 2.
- 3.

— ;

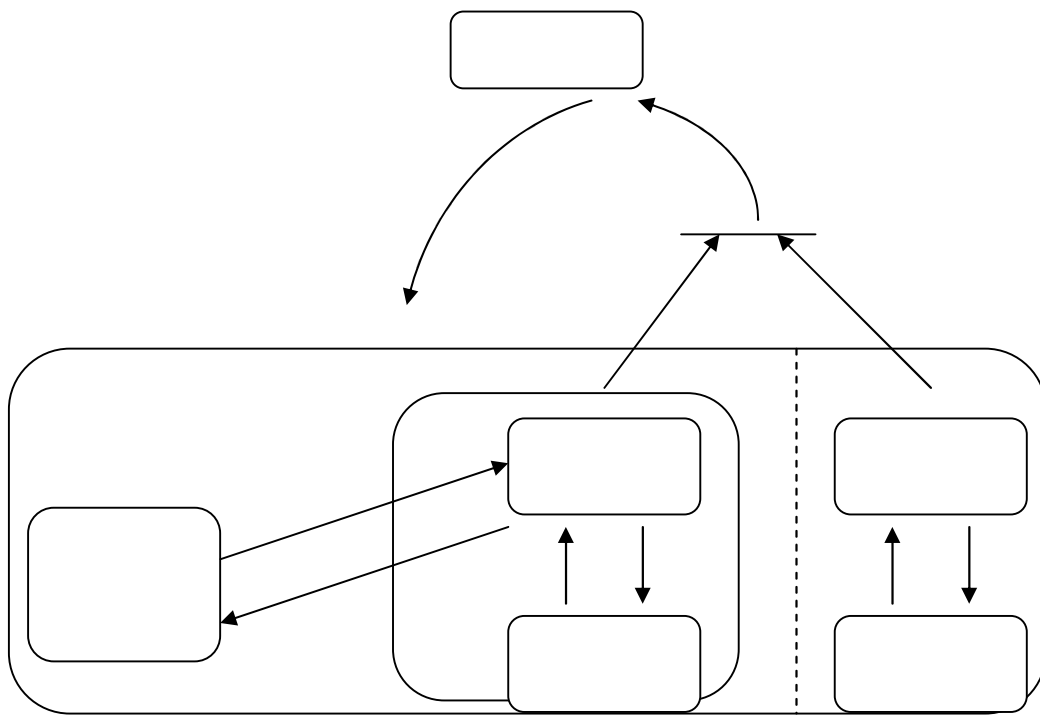
—

;

1 2 ,

2 3

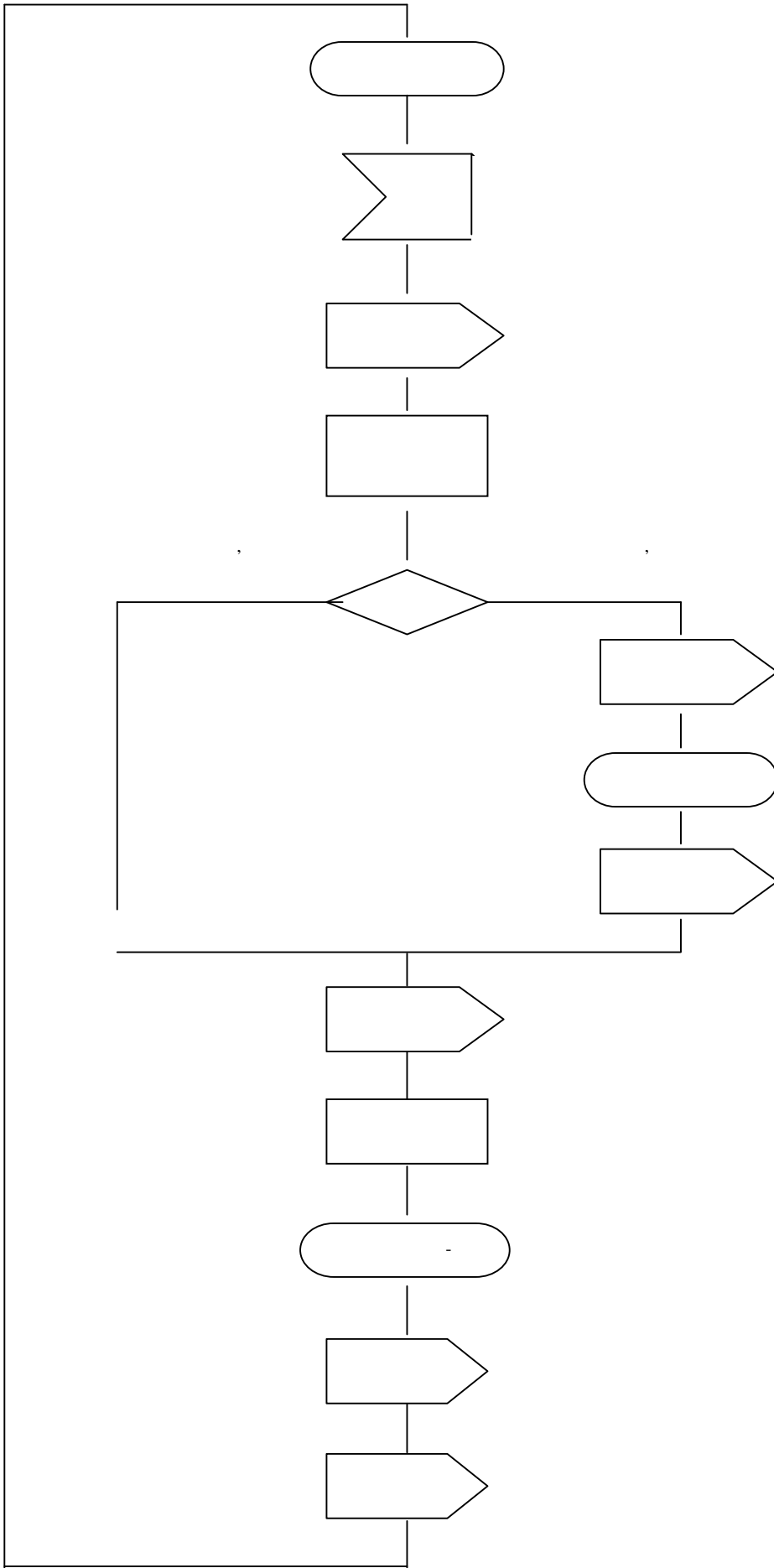
,



.9.

2.

, . , ,  
 , ,  
 .  
 ,  
 « », . 10.  
 , , , « » ,  
 ,  
 , . 10,  
 , , .



3.

1.

2.

3.

4.

1.

2.

3.

4.

5.

6.

7.

8.

8.

9.

«

» (“Loaded”).

(

Loaded

Loaded

Integer.

Boolean,

),

Get() Set()

```
TDirection = (up, down);
TLiftState = (moving, stop);
TPosition = (1..10);
```

```
TLift = object
  State : TLiftState;
  Direction : TDirection;
  Position : TPosition;
  Constructor Create;
  Destructor Done;
  Procedure Start(D : TDirection);
  Procedure Stop;
  Procedure Move;
  Function GetPosition : TPosition;
  Function GetState : TLiftState;
  Function GetDirection : TDirection;
End;
```

```
      Start  Stop      ,
      .
      GetPosition, GetState, GetDirection      ,
      .
      Move -      .
```

```
TDoorState = (opened, closed, opening, closing);
```

```
TDoor = object
  State : TDoorState;
  Constructor Create;
  Destructor Done;
  Procedure Open;
  Procedure Close;
  Procedure Move;
  Function GetState : TDoorState;
End;
```

```
      Open  Close      ,
      .
      GetState      ,
      .
      Move      «      »      .
```

```
TEventType = (Alarm, ButtonPressed);
TButtonType = (LiftButton, FloorButton);
```



```

TEvent = record
    EventType : TEventType;
    ButtonType : TButtonType;
    Message : Integer;
End;

```

```

TController = object
    Constructor Create;
    Destructor Done;
    Function GetEvent : TEvent;
    Procedure Dispatcher;
End;

```

```

    Dispatcher          «          »,
    GetEvent            ,          ,

```

4.

1.

2.

3.

a.

b.

4.

5.

- 1.
- 2.
- 3.
- 4.
- 5.
- 6.
- 7.

:

**1.**

- 1.
- 2.
- 3.
- 4.
- 5.
- 6.

**2.**

,  
: .

.

, .6.

.6.,

	-	.
	« »	.

( ) « - ».  
, ( ).

.



4.

.7.

.7.

		« ».

« »

5.

- 1.
- 2.
- 3.
- 4.
- 5.
- 6.

[1..N], N – integer,

( )

- 1.
- 2.
- 3.

- 1.
- 2.
- 3.
- 4.
- 5.
- 6.
- 7.

**6.**

“1” – “9”  
 “q” – “o”

1.

```

2.         ;
3.         ;
4.         ;
5.         ;
6.         ;
7.         ;
8.         ;
9.         ;
10.        ;
11.        ()      ;
12.        V()     ;

```

“ ” “ ” “ ”

```

#include <setjmp.h>
#include <dos.h>

```

```

class TProcess : {          //
    int stack[1000];       //
    jmp_buf ts;           //
public:
TProcess ( void *p ) {     //
    struct SREGS segs;
    segread ( &segs );
    ts[0].j_sp = FP_OFF(stack) + sizeof stack;
    ts[0].j_ss = FP_SEG(stack);
    ts[0].j_flag = 0x200;
    ts[0].j_cs = FP_SEG(p);
    ts[0].j_ip = FP_OFF(p);
    ts[0].j_bp = ts[0].j_sp;
    ts[0].j_di = 0;
    ts[0].j_es = segs.es;
    ts[0].j_si = 0;
    ts[0].j_ds = segs.ds;
} //TProcess
}; //TProcess

```

```

class TSemaphore {
    int count;           //
    T ollection *List;  //
public:
    TSemaphore();      //
    ~TSemaphore();     //
    void P();          //      P
    void V();          //      V
}; //TSemaphore

```



```

#include <tv.h>
#include "proc.h"
#include "sema.h"

class tkernel {
    TCollection *ReadyList;    //
    TCollection *KillList;    //
    TCollection *DelayList;   //
    TProcess *cur;           //
    unsigned long count;      //      “      ”
    void swt(jmp_buf from,jmp_buf to); //
public :
    tkernel();               //
    ~tkernel();              //
    void start();            //
    void stop();             //
    void delay(unsigned long tik); //
    void resume();           //
};

```

- 1.
- 2.
3.
  - a.
  - b.
- 4.
- 5.

# ETHERNET

:

WEB  
Ethernet

/

«  
Ethernet»

- 1.
- 2.
- 3.
- 4.

IPC ,  
- IPC

:

,  
IPC GGI,

:

1. (telnet)
2. GGI-
- 3.

4. - . - .  
(telnet)

GGI-

- 6.

1 -

(telnet); 2 -

( )

- 7.

( I2C)

- 8.

WEB- (GGI)

- 9.

WEB-

- 10.

- WEB- GGI

- 11.

- 12.

- (GGI),

13. telnet  
 14.  
 15. GGI  
 16. /  
 GGI  
 17.  
 GGI  
 18. GGI  
 19. GGI WEB-browsing (WEB-  
 RTOS)  
 20. (  
 21. )  
 ( )  
 22. RTOS 20  
 23. RTOS 21  
 24. IPC@Chip (telnet)  
 25. IPC@Chip (web- )  
 26. -  
 ( IP ) ( )  
 27. 2 , 1  
 . ( ) (1-8, 2-7 3-6 5-4)  
 28.  
 . (web)  
 29. IPC@Chip - ( )  
 , ,  
 ) (web)  
 30. IPC@Chip  
 (web) ( IP )  
 :  
 1. .  
 2. .  
 3. .  
 4. .



( )

( , , )  
.

:

- 1.
- 2.
- 3.
- 4.
- 5.
- 6.
- 7.
- 8.

« »

( ):

- 1. ( , )
- a. - RM
- b. -
- 2.

( , )

- a. - LSTF
- b. -

( ) ( , , 3

2- , 3- )

- a. - EDF
- b. -

4. ( , )  
a. - RM  
b. -

5. ( 1, 2, )  
a. - LSTF  
b. -

6. « »,  
a. )  
b. - EDF

( 7. )  
a. - RM  
b. -

( 8. ) ( , )  
a. - EDF  
b. -

9. ( / , )  
a. - LSTF  
b. -

10. ( / , )  
a. - RM  
b. -

11. ( , )  
a. - EDF  
b. —

( 12. ) ( , )  
a. - LSTF  
b. -

13. ( , )  
a. - RM  
b. -

14.  $\frac{1}{2}$ ,  $\frac{1}{2}$ ,  $\frac{1}{2}$  ( ) 2
- a. - EDF
- b. -
15.  $\frac{1}{2}$ ,  $\frac{1}{2}$  ( )
- a. - LSTF
- b. -
- ( )
16.  $\frac{1}{2}$ ,  $\frac{1}{2}$  ( )
- a. - RM
- b. -
- ( )
17.  $\frac{1}{2}$ ,  $\frac{1}{2}$  ( )
- a. - LSTF
- b. -
18.  $\frac{1}{2}$ ,  $\frac{1}{2}$  ( )
- a. - EDF
- b. -
19.  $\frac{1}{2}$ ,  $\frac{1}{2}$  ( )
- a. - RM
- b. -
20.  $\frac{1}{2}$ ,  $\frac{1}{2}$  ( )
- a. - EDF
- b. -
- 1..12, ( )
- a. - LSTF
- b. -
- ( )
22.  $\frac{1}{2}$ ,  $\frac{1}{2}$  ( )
- a. - RM
- b. -
- ( )
23.  $\frac{1}{2}$ ,  $\frac{1}{2}$  ( )
- a. - LSTF

b.  
24.

-

(  
)

a.  
b.

- EDF  
-

( )

25.

/

( , , )

a.  
b.

- RM  
-

26.

(

)

a.  
b.

- LSTF  
-

27.

(

, )

a.  
b.

- EDF  
-

( )

28.

«

»

(

a.  
b.

- RM  
-

29.

(

,

,  
)

a.  
b.

- LSTF  
-

30.

(

, )

a.  
b.

- EDF  
-

:

1.  
2.  
3.  
4.

.

.

.

.



( 5 7).

« » ,

,

:

( )  
 ( )  
 ( )  
 ( , )  
 ( , )  
 ( jitter , D )  
 ( jitter , J )  
 « »  
 ( )

(R).

( - 2, )

- a.
- b. ( )
- c. ( )
- d. ( )
- e. 1 ( , R1)
- f. 2 ( , R2)
- g. 3 ( , R3)
- h. 4 ( , R4)

2

:

- i. Message Utilization – (%)
  - j. Bus Utilization – (%)
  - k. Breakdown Utilization – (%)
- ( )

Breakdown Utilization<1,

1.

2.

R(J,D,T),

3.

Breakdown Utilization. ( )

4.

a.

( ).

b.

(

).

c.

d.

e.

1,

2,

f.

( ).

g.

(

,

)

			+	
EIB	9,6 / 100 / 500 / 1200 /	70	+ 6	Siemens
LonWorks	78,6 / 125 / 250 / 1200 /	84	+ 10	Echelon
CAN	125 / 250 / 500 / 1000 /	35	+ 5	Bosh
P-NET	56,6 /	24	+ 5	Process

	78,6 / 120 / 240 /		DATA
TCP-IP	56,6 / 1000 / 2000 / 10000 /	48 + 5	
Profibus	9,6 / 12,4 / 56,6 / 78,6 /	32 + 5	

( ,

.)

:

1. (P-NET)
2. (LonWorks)
3. (P-NET)
4. (CAN)
5. (LonWorks)
6. (EIB)
7. (TCP-IP)
8. (P-NET)
9. (EIB)
10. (LonWorks)
11. (EIB)
12. (TCP-IP)
13. (CAN)
14. (LonWorks)
15. (TCP-IP)
16. (EIB)
17. (P-NET)
18. / (P-NET)
19. (TCP-IP)
20. (LonWorks)
21. (CAN)
22. (CAN)
23. (EIB)
24. (TCP-IP)
25. (LonWorks)
26. (EIB)
27. (TCP-IP)
28. (CAN)

29.  
30.

(TCP-IP)

(LonWorks)

1.  
2.  
3.  
4.

:

.

.

.

.

# SCADA

## SCADA

Simple-Scada

1.3.3.

Explorer – ( , , ( ) , , t : ).

Builder – ,

Runtime – , SCADA

1. Explorer

2.

3.

), (Generic), (MEMORY). ( 1

4.

) ( ) Variable Tag

, – INT, DECEMAL

DECEMAL

D1 D100, INT – I1 I100

5.

6.

.( 1 – 3 ).

Explorer.

7. ,

8. « ».

9. ( )

Access Input

( ).

10.

(Appearance, Movement, Scaling, Fill, Slider).

11.

1.

2.

1.

4-8

2.

« »

3.

4.

1.

2.

2

3.

4.

5.

(2-3

6.

7.

8.

(2

5

)

- 2-3 9. ) (2-3 ,  
10.  
11. ( , 2-3 )  
12.  
13.  
14.  
15. (5-7  
16. ) ( - 4-5 )  
17.  
18.  
19.  
20.  
21.  
22.  
23.  
24.  
25.  
26.  
27.  
28.  
29.  
30.

1. .  
2. .  
3. .  
4. .  
5. .

# QNX

1.

( )

MS-DOS.

MS

Windows

Unix, Linux, QNX

( )

QNX,

QNX.

2.

QNX

QNX,

login:

password:

(  
login)

root,

#,

-\$.

(

( )

(

logout.

shutdown.



# QNX

:

< > < > <Home> <End> <Backspace> <Del> <Ins>.  
- <Enter>.

<Ctrl>+<U> -  
: < > < >.

(/dev/con1, ...):

<Ctrl>+<Alt>+<Enter> <Ctrl>+<Alt>+<+> -  
;

<Ctrl>+<Alt>+<-> - ;  
<Ctrl>+<Alt>+<n> - n.

cfont.

:

<Ctrl>+<Alt>+<>> - ( n);  
<Ctrl>+<Alt>+<<< - ( 0).

- <Ctrl>+<D>

( ): <Ctrl>+<C> <Ctrl>+<Break>

: <Ctrl>+<S>

: <Ctrl>+<D>

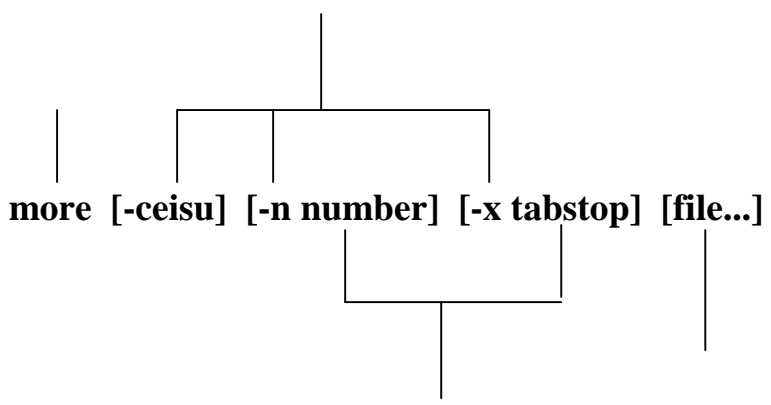
( ):)

<Ctrl>+<Alt>+<Shift>+<Del>

QNX

QNX

( more):



[...] -  
-a | -f -  
... -

(stdin),

```

    (stdout),
    :
    ;
    <(
)
)
)
    stdout , >>(
)
    stdout | ( , )
,
ps > /tmp/pr1
    (node) (tty)
    ( , stdin, stdout, stderr
,
    .
    on node –
    on tty – stdin, stdout tty.
    on node.
    1. sin 4 - :

```

onnode 4 sin

```

    : //4 sin
    2. sin /bin 4
( - sin : ):

```

onnode 4 //2/bin/sin

```

    3. sin 4 1
    2 ( - ) :

```

onnode 4 sin >//2/dev/con1

```

    ontty.
    1. ls ,
    3:

```

ontty /dev/con3 ls

```

    on node on tty
    1. nameloc 4

```

- /dev/con1 4:

onnode 4 ontty //4/dev/con1 nameloc &

shell.

shell

<< >>

nohup:

onnode 4 ontty //4/dev/con1 nohup nameloc &

use:

use sin

\* - (>= 0)

? -

[] -

[1-3] [a-c].

! -

[!a].

cp f\* /tmp

cp \*d /tmp

cp freg? /tmp

cp freg[123] /tmp cp freg[1-3] /tmp

cp \*.\*[ch] /tmp

cp \*.\*[!o] /tmp {  
o}

cp freg /tmp; ls /tmp

QNX

48

0x00 ... 0x1F

0x2f (/)

0x7F (rubout)

0xFF

**POSIX:**

**a...z A...Z 0 1 ... 9 \_ . -**

( «-» ).

( , .txt .c .h .o . . ).

( )

/ - ( ).

//2/ - 2 .

- ...  
(/home/slava).  
cd.

( ) QNX

QNX

( ):

**/bin**

**/boot**

Makefile

**/boot/build**

make-

[/boot/images](#)

**boot/sys**

**/etc**

**/etc/config**

sysinit

**/etc/readme**

**/etc/readme/technote**

**/tmp**

**/usr/bin**

**usr/include**

C

**/usr/lib**

C

**/usr/lib/terminfo**

**/usr/lib/APPLICATION**

QNX

**/usr/spool/lp**

**/home/USERID**

# QNX

**cd** –  
**mkdir** –  
**rmdir** –  
**pwd** –  
**ls [-l]** –

**diff** –  
**cat** –  
**cp** –  
**wc** –  
**more** –  
**less** – ( )  
**lp** –  
**mv** –  
**rm** –  
**grep** –  
**sort** –

**who** –  
**ps** –  
**sin** –  
**find** –  
**write** –  
**wall** –

5.

**Photon**

QNX

Photon.

Motif

Photon

**ph**

**shutdown**

Photon.

1.

( 2).

2.

( 5).  
4,

3.

6.

4.

5.

1.

2.

3.

4.

5.

QNX.

1.  
QNX

2.

QNX

3.

QNX

«

QNX».

4.

QNX»

5.

Photon  
Photon.

1. [ ]: / . . . - :  
| , 2015. - 115 .
  2. . . , [ ]:  
/ . . , .- 4-
  3. . .- .: , 2012. - 560 . [ ] :  
/ . . , . . . - 6- .  
. - : , 2015. - 263 .
  4. , . . [ ]:  
/ . . , . . . . :  
: . . . , 2011-280 .
  5. , . . [ ]:  
/ . . , . . . - : : -  
, 2014. - 223 .
  6. , . . [ ] : / . .  
, . . , . . . - 2- . , . - .: , 2012.-
- 304 .