実用EPICS入門-4
技術部専門講習

飛山真理
帯名崇

複雑な処理をさせたいとき

- データベースでがんばる(下策)
  - 見通し最悪、保守性も極悪
- 上位ソフトウエア(SADとかPythonとか)でがんばる(場合による、中策)
  - 汎用性に難、上位計算機の負荷にも難
- EPICS Sequencerで処理する(今回のオススメ)

EPICS Sequencer

- State Notation Language (状態遷移指向言語)
- EPICSデータベースと通信
- IOC内、あるいは独立に動作可能、もちろんnative codeで実行
- Cに似た(しかし結構違う)言語で複雑なコードを記述可

- State Notation Language and the Sequencer

  by Andrew Johnson

  APS Engineering Support Division

  October 18th, 2006

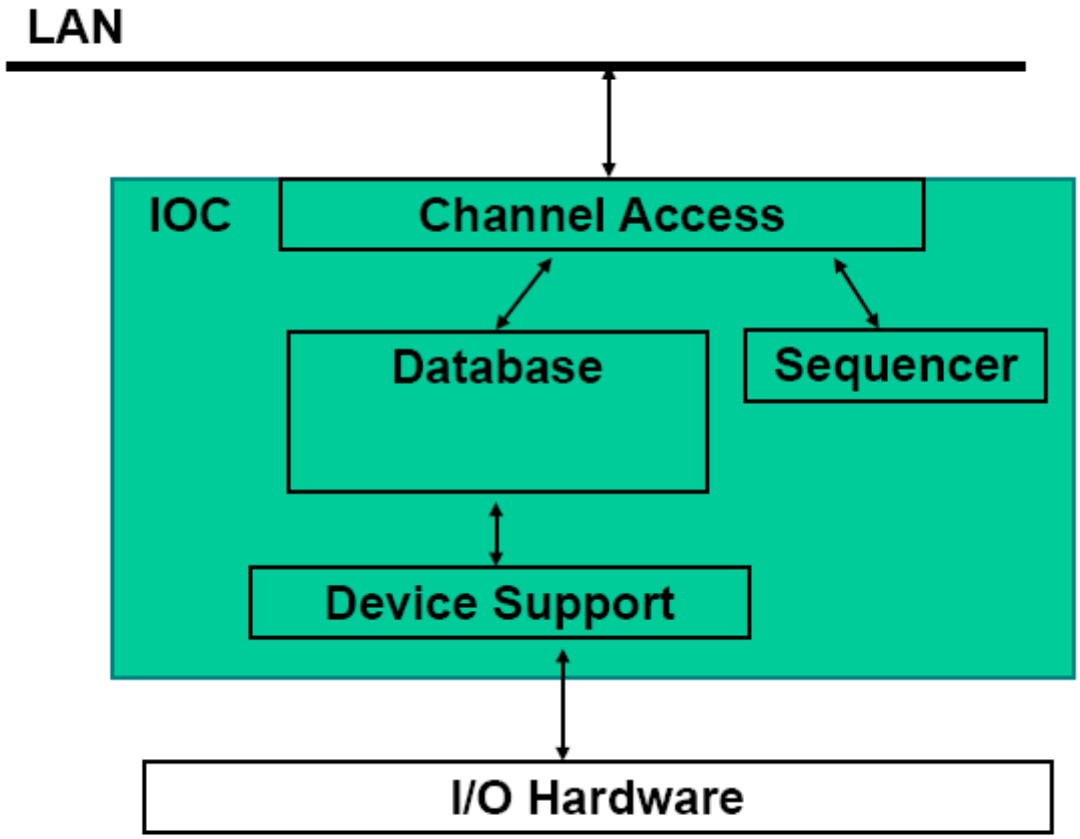
  SNS EPICS Training

  から引用しています

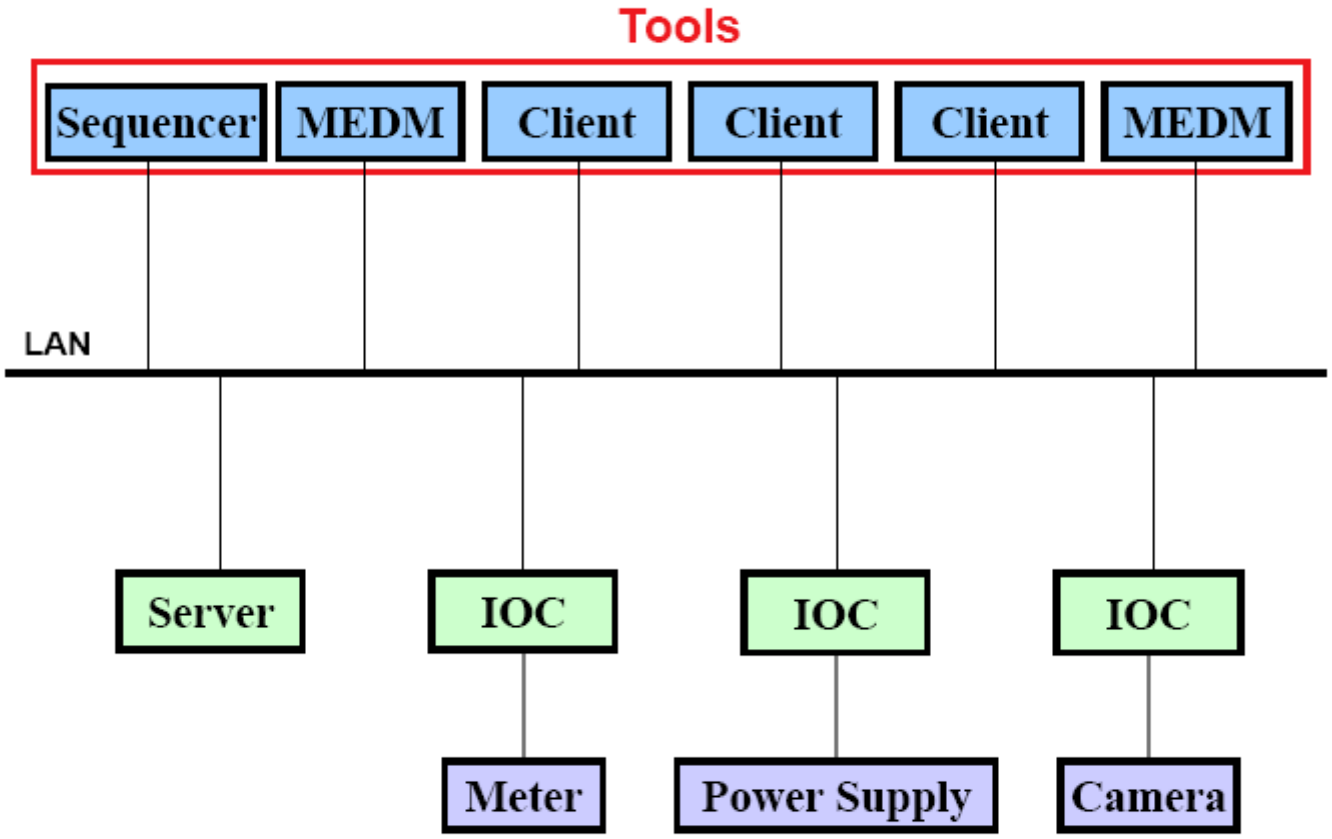
SNLとSequencer

- SequencerはState Notation Language(SNL)で記述されています
- SNLはCに似た言語で、sequential動作を簡単に記述できます
- コンパイルしてあるので、高速動作します
- リアルタイム環境下にあるEPICSを拡張するプログラムインターフェースを提供します

元々のSequencerのイメージ



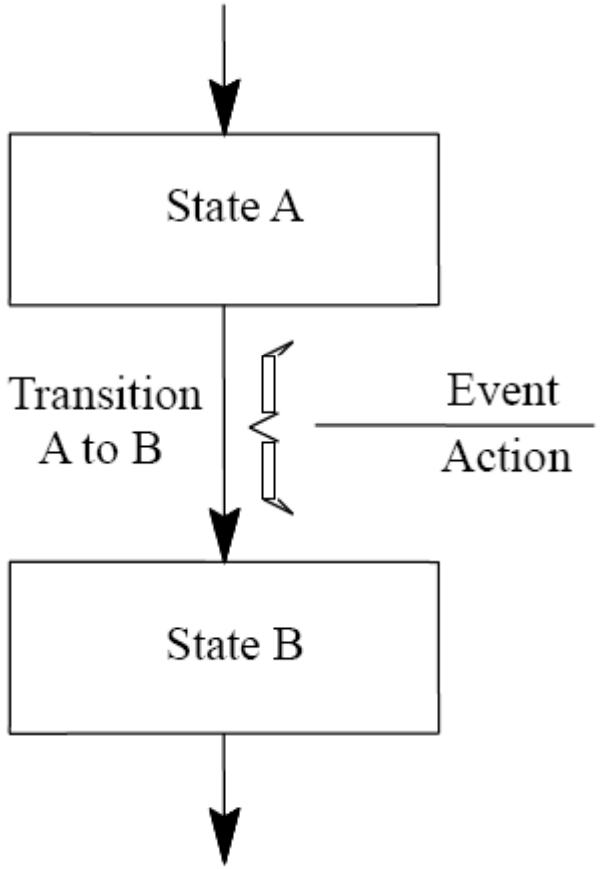
今では


Tools
Sequencer
MEDM
Client
Client
Client
MEDM
LAN
Server
IOC
IOC
IOC
Meter
Power Supply
Camera

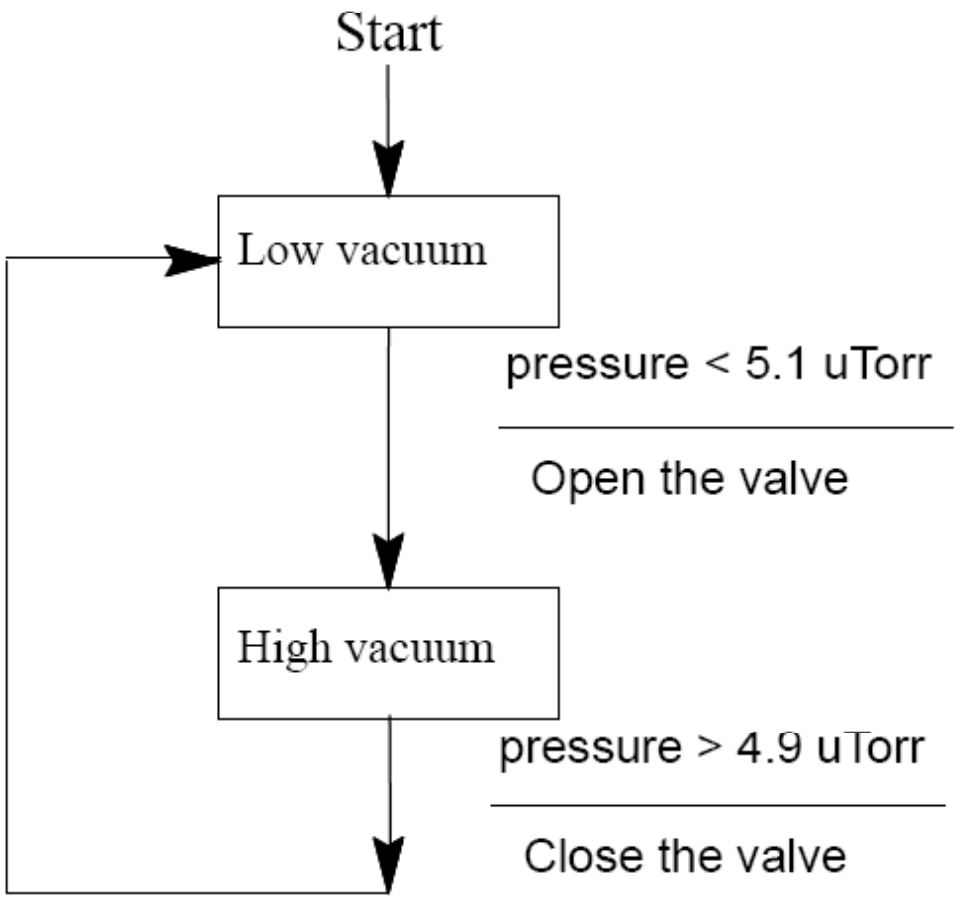
Sequencerは

- 最近のEPICSでは、sequencerはIOC上のプロセスの一部としても、あるいはスタンドアロンプログラムとしても走ります
  - 元々はIOC上で走るプロセスでした
- IOC上のプロセスとして走らせておくと、管理しやすい
  - 通信負荷もちょびっと少ないらしい
- スタンドアロンプログラムとして走らせていれば、テスト、デバッグが容易になる
- workstation上で(簡単な)CAクライアントを書く必要があるとき、SNLを使えば簡単

状態遷移図



状態遷移図の例


Start
Low vacuum
pressure < 5.1 uTorr
Open the valve
High vacuum
pressure > 4.9 uTorr
Close the valve

SNL: General Structure and Syntax

```
program program_name
declarations

ss state_set_name {
    state state_name {
        entry {
            entry action statements
        }
        when (event) {
            action statements
        } state next_state_name
        when (event) {
            ...
        } state next_state_name
        exit{
            exit action statements
        }
    }
    state state_name {
        ...
    }
}
```

SNL: General Structure and Syntax

- program なんとか
  - program中にいくつかのstate setがあることがあります
  - 名前は、process名などに使われます(ので、わかりやすい名前が良いです)
- ss なんとか{
  - 並列動作プロセスです。並列動作が無いときはprogram中に1個だけあることになります
- state なんとか{
  - 状態ブロックです。eventが起きると、上から順に実行されます。
- when (event) { ... } state 次
  - このstateが起動される(起こるまで待つ)event条件を書きます。ブロックが実行され終わると、最後に書かれたstateへ移行し、そのstateのwhen条件が満たされるまで待ちます

変数宣言

- Appear before a state set and have a scope of the entire program.
- Scalar variables

```
int     var_name;
short   var_name;
long    var_name;
char    var_name;
float   var_name;
double  var_name;
string  var_name;    /* 40 characters */
```

- Array variables: 1 or 2 dimensions, no strings

```
int     var_name[num_elements];
short   var_name[num_elements];
long    var_name[num_elements];
char    var_name[num_elements];
float   var_name[num_elements];
double  var_name[num_elements];
```

assignment宣言

- Assignment connects a variable to a channel access PV name

```
float  pressure;
assign pressure to "CouplerPressureRB1";
double pressures[3];
assign pressures to {"CouplerPressureRB1",
  "CouplerPressureRB2", " CouplerPressureRB3"};
```

- To use these channel in *when* clauses, they must be monitored

```
monitor pressure;
monitor pressures;
```

- Use preprocessor macros to aid readability:

```
#define varMon(t,n,c) t n; assign n to c; monitor n;
varMon(float, pressure, "PressureRB1")
```

Event

Event: The condition on which actions associated with a **when** are run and a state transition is made.

Possible events:

- Change in value of a variable that is being monitored:

```
when (achan < 10.0)
```

- A timer event (not a task delay!):

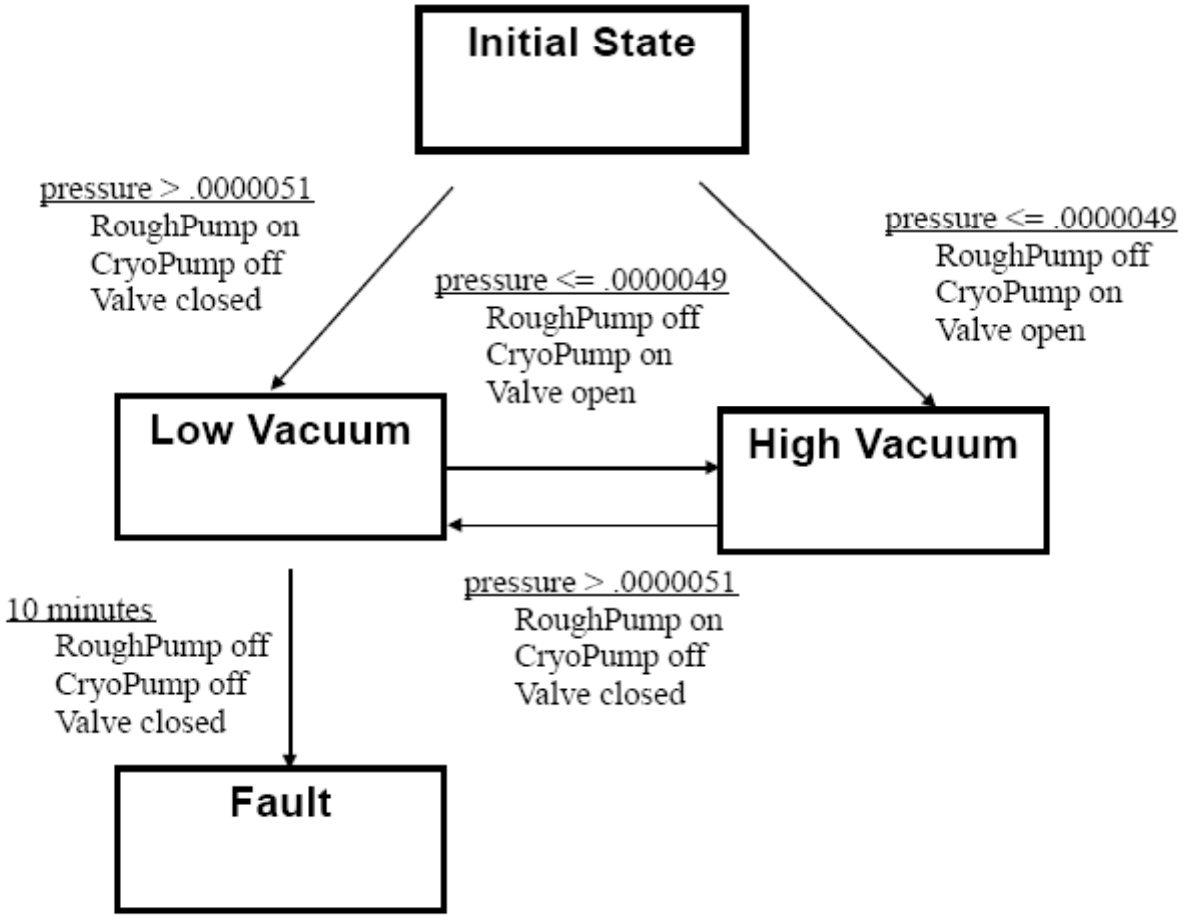
```
when (delay(1.5))
```

  - The delay time is in seconds. It is declared internally as a double; constant arguments to the delay function **must** contain a decimal point.
  - A delay is normally reset whenever the state containing it is exited.
  - Use the state specific `option -t;` to stop it from being reset when transitioning to the same state.

実行文

- CA関連
  - pvPut(var_name);
  - pvGet(var_name);
- もろC
  - プリプロセッサでSNLはCに変換されるので、もろCの記述は一般的にOK、但し、SNLに定義してない部分はescapeする必要がある
    - %%で書いた一行
    - %{　と　}%でくくった部分
  - escapeした中にSNL独自の記述は許されない(例えばpvPut、pvGetなど)

Example – State Definitions and Transitions



宣言の例

```
double  pressure;
assign  pressure to “Tank1Coupler1PressureRB”;
monitor pressure;

short   RoughPump;
assign  RoughPump to “Tank1Coupler1RoughPump”;
short   CryoPump;
assign  CryoPump to “Tank1Coupler1CryoPump”;
short   Valve;
assign  Valve to “Tank1Coupler1IsolationValve”;
String  CurrentState;
assign  CurrentState to “Tank1Coupler1VacuumState”;
```

state transitionの例

```
program vacuum_control

ss coupler_control
{
        state init{
                when (pressure > .0000051){
                } state low_vacuum
                when (pressure <= .0000049){
                } state high_vacuum
        }
        state high_vacuum{
                when (pressure > .0000051){
                } state low_vacuum
        }
        state low_vacuum{
                when (pressure <= .0000049){
                } state high_vacuum
                when (delay(600.0)){
                } state fault
        }
        state fault {
        }
}
```

初期化部の例

```
state init {
        entry {
                strcpy(CurrentState,"Init");
                pvPut(CurrentState);
        }
        when (pressure > .0000051){
                RoughPump = 1;
                pvPut(RoughPump);
                CryoPump = 0;
                pvPut(CryoPump);
                Valve = 0;
                pvPut(Valve);
        } state low_vacuum
        when (pressure <= .0000049){
                RoughPump = 0;
                pvPut(RoughPump);
                CryoPump = 1;
                pvPut(CryoPump);
                Valve = 1;
                pvPut(Valve);
        } state high_vacuum
}
```

Example – State low_vacuum

```
state low_vacuum{
        entry {
                strcpy(CurrentState,"Low Vacuum");
                pvPut(CurrentState);
        }
        when (pressure <= .0000049){
                RoughPump = 0;
                pvPut(RoughPump);
                CryoPump = 1;
                pvPut(CryoPump);
                Valve = 1;
                pvPut(Valve);
        } state high_vacuum
        when (delay(600.0)){
        } state fault
}
```

Example – State high_vacuum

```
state high_vacuum{
        entry {
                strcpy(CurrentState,"High Vacuum");
                pvPut(CurrentState);
        }
        when (pressure > .0000051){
                RoughPump = 1;
                pvPut(RoughPump);
                CryoPump = 0;
                pvPut(CryoPump);
                Valve = 0;
                pvPut(Valve);
        } state low_vacuum
}
```

Example – State fault

```
state fault{
        entry{
                strcpy(CurrentState,"Vacuum Fault");
                pvPut(CurrentState);
        }
}
```

SNLの長所

- 複雑なアルゴリズムをimplementできる
- (スタンドアロンプロセスの時)IOCを止めたりリブートしたりせずにsequencerプログラムを止めたり、リロードしたり、リスタートできるので、デバッグが容易
- Cコードを直接使える(ライブラリパッケージなど)
- channel accessが容易にできる
- fileアクセスも(容易に)可能となる