

```
class System {
                                          34.Statemachine FIFOMachine {
                                               InitialState Idle;
2. public:
     Producer p:
                                          36.
                                               State SignalChecking {
                                         37.
     Consumer c;
                                                 StateEntry entryCheck;
                                          38.
     FIFO fifo;
                                                 StateExit exitCheck;
                                          39.
     void configuration() {
                                              };
                                          40.
                                               State ErrorNotification {
       bindPorts(p.pPush, fifo.pPush);
       bindPorts(c.pPull, fifo.pPull);
                                          41.
                                                 StateEntry entryError;
9.
                                          42.
10.}
                                          43.
                                               State DataOueuing {
11.class IPull {
                                          44.
                                                 StateEntry entryOueue;
12.public:
                                          45.
                                                 PseudoChoice queueChoice;
13. virtual Data* pull() = 0;
                                          46.
                                                 State Queuing;
14.}
                                          47.
                                               };
                                          48.
15.class IPush {
                                               State Discarding:
                                          49.
                                               PseudoChoice dataChoice;
16.public:
17. virtual void push (Data& data) = 0; 50.
                                               CallEvent(DataPushEvent, push(Data&));
18.}
                                               TransitionTable {
                                          51.
                                          52.
                                                 ExT (Idle, SignalChecking,
19.class Producer {
                                          53.
20.public:
                                                     DataPushEvent, NULL, signalCheck);
                                          54.
21. RequiredPort<IPush> pPush;
                                                 ExT (SignalChecking, dataChoice,
                                          55.
22.};
                                                              NULL, NULL, NULL);
23.class Consumer {
                                          56.
                                                 ExT (dataChoice, queueChoice,
                                          57.
24.public:
                                                              NULL, Valid, NULL);
                                          58. }
     RequiredPort<IPull> pPull;
25.
                                          59.1:
126.};
27.class FIFO : public IPush, IPull {
                                          60.void entryCheck(){//fine-grained code}
28.public:
                                          61.void exitCheck(){//fine-grained code}
     ProvidedPort<IPush> pPush;
                                          62.void entryError(){//fine-grained code}
     ProvidedPort<IPull> pPull;
                                          63.void signalCheck(Data& item) {
     Data* pull(){//fine-grained code}
                                          64.//trans effect from Idle to SignalChecking
     void push(Data& data){//..}
                                          65.}
                                                                   //attributes + methods...
                                          66.}
```