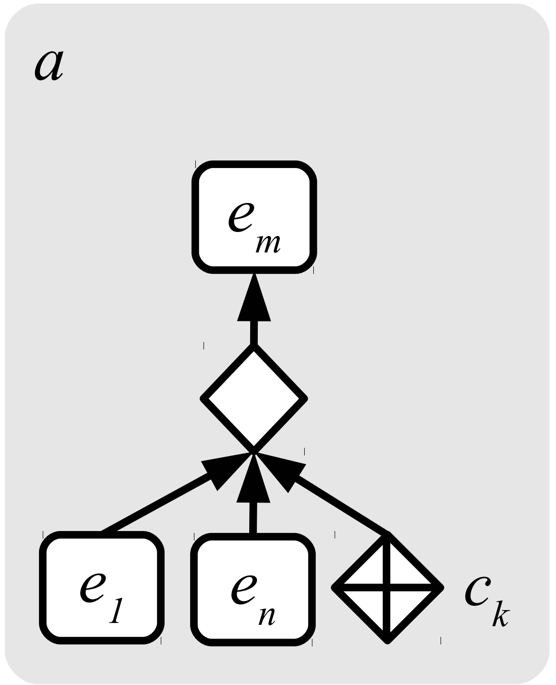
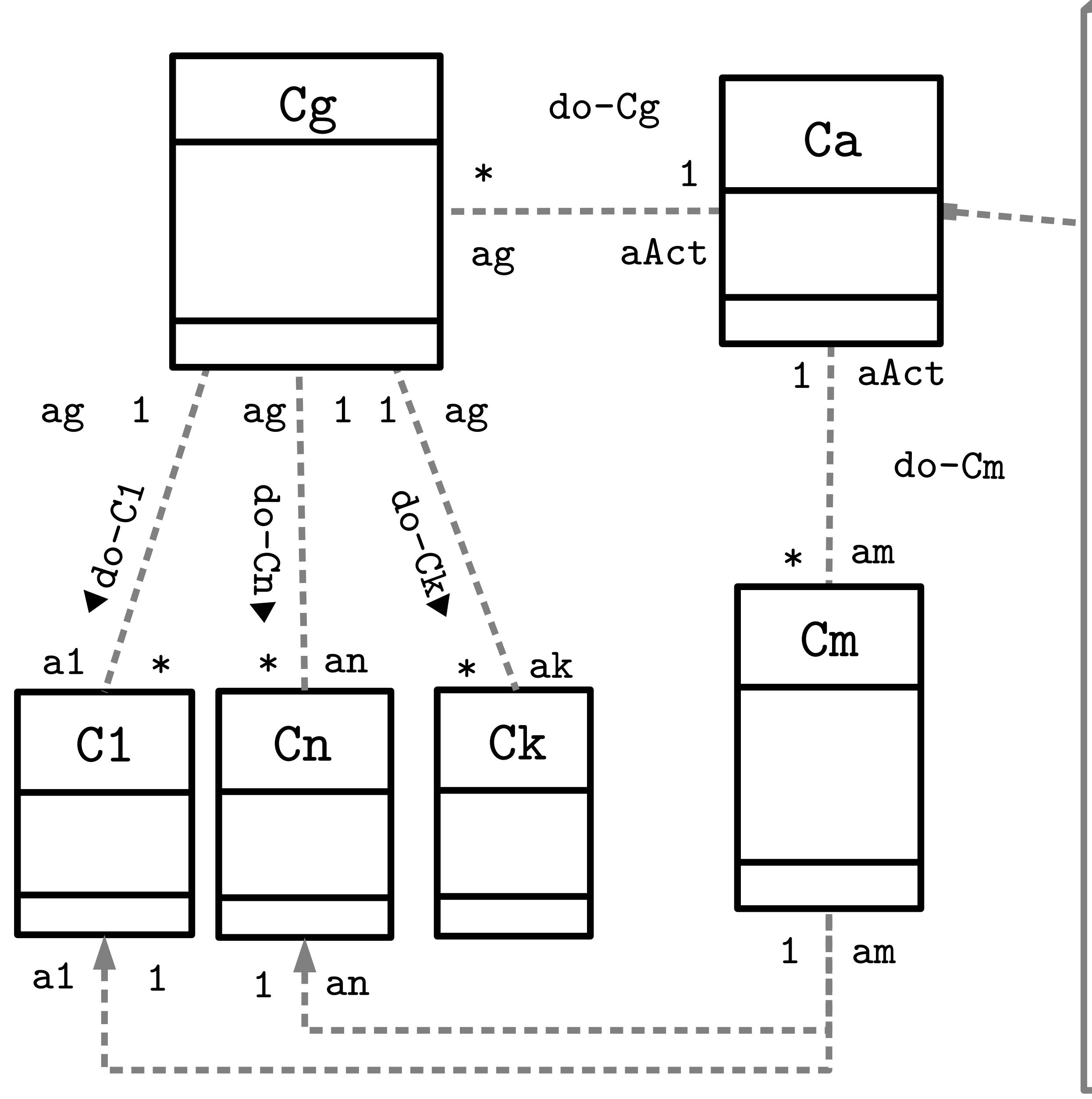
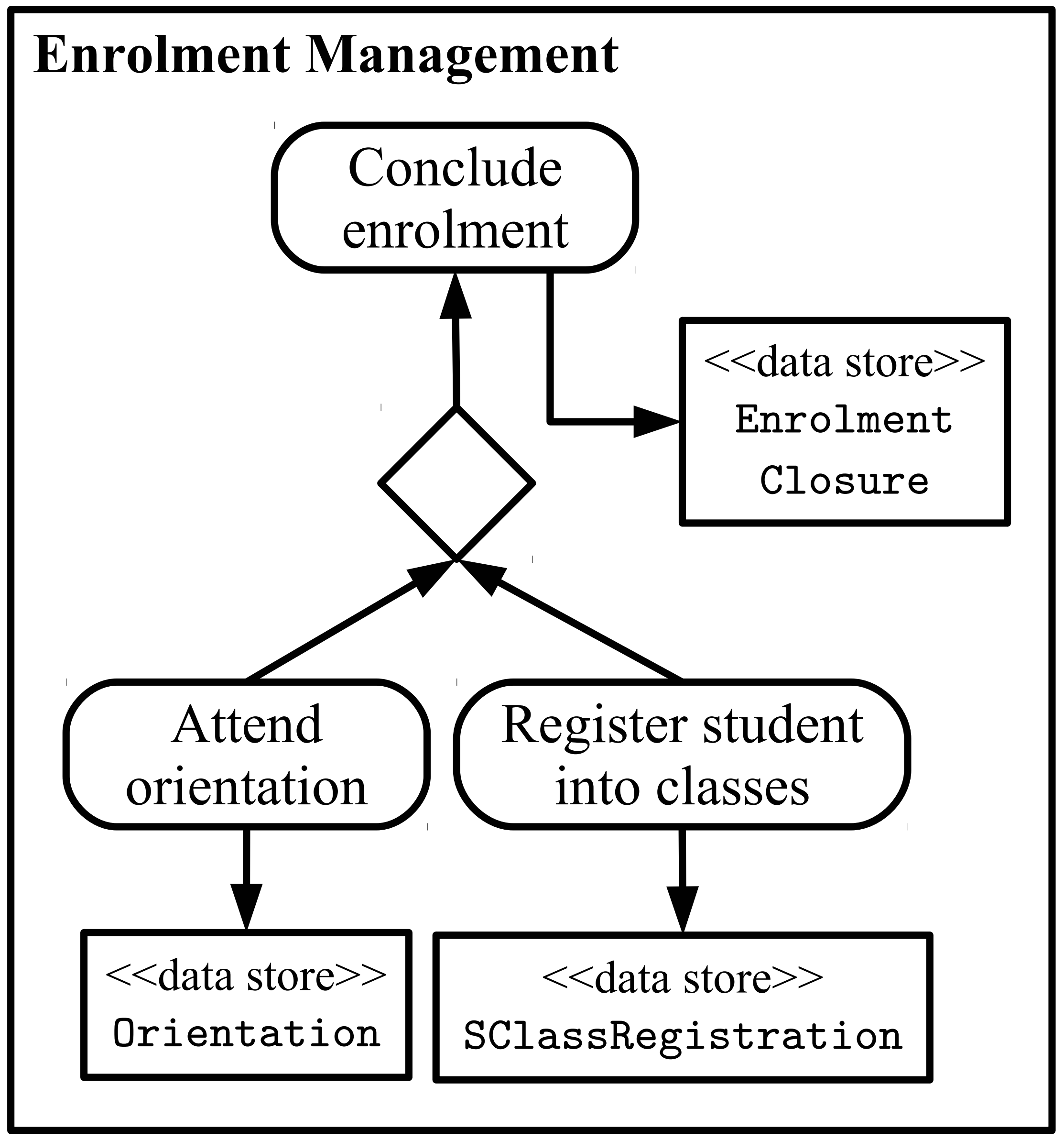
merged pattern
(if c_k is present then $n \geq 1$
else $n \geq 2$)

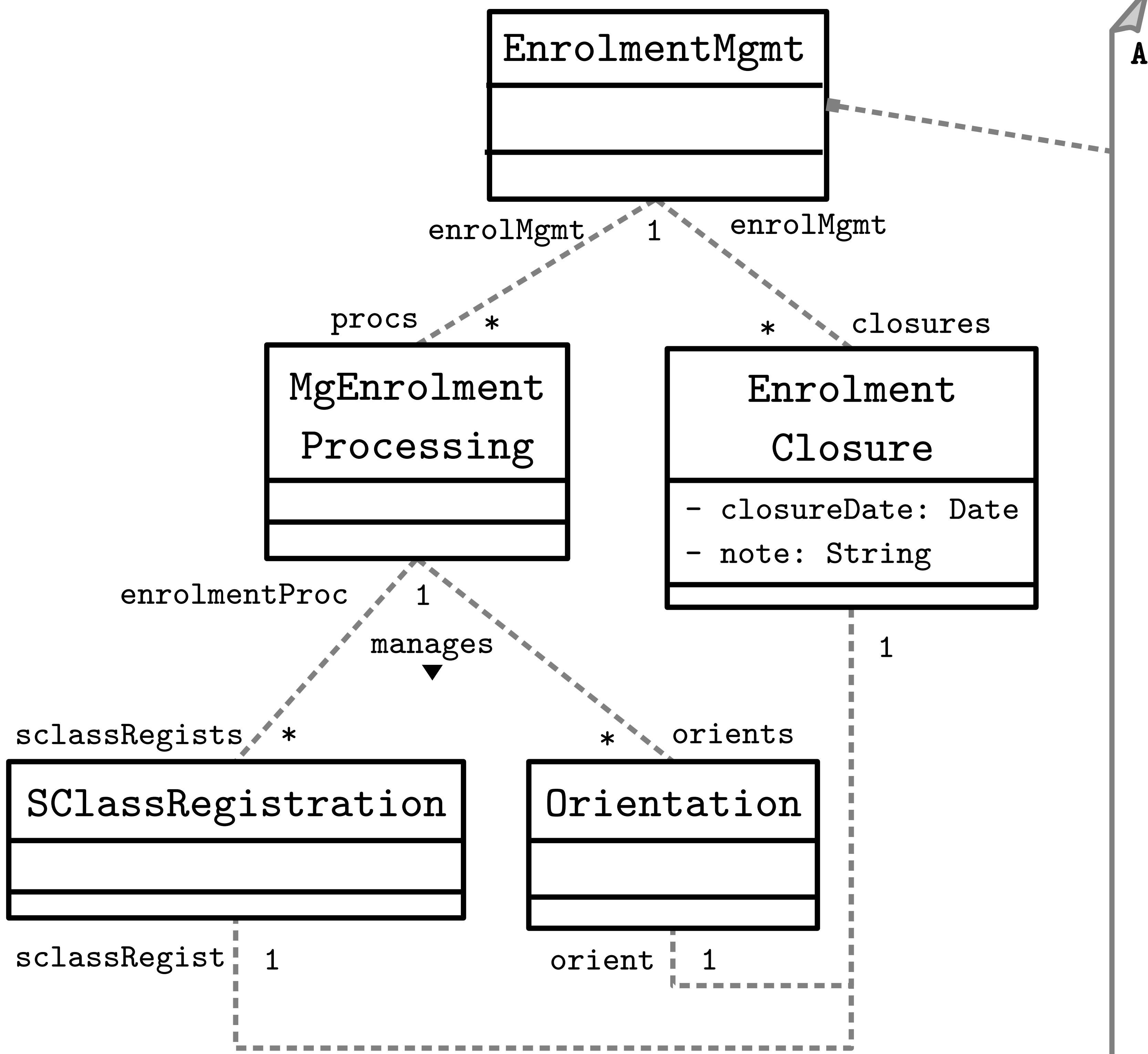




```
AGraph { nodes=[
   ANode { refCls=C1,serviceCls=DataController
   outClses=[Cg]
   actSeq=[
     MAct{actName=newObject,pstStates=[Created]} ]
   init=true}
   ANode { refCls=Cn,serviceCls=DataController
   outClses=[Cg]
   actSeq=[
     MAct{actName=newObject,pstStates=[Created]} ]
   init=true}
   ANode { refCls=Ck,nodeType=k,outClses=[Cg],init=true}
   ANode { refCls=Cg,nodeType=Merge,outClses=[Cm]}
   ANode { refCls=Cm,serviceCls=DataController
   actSeq=[
     MAct{actName=newObject,pstStates=[NewObject]},
     MAct{actName=setDataFieldValues
         fieldNames=["a1","an"], pstStates=[Created]} ]}
 ]}
```



Enrolment Management
Conclude enrolment
<<data store>> Enrolment Closure
Attend orientation
Register student into classes
<<data store>> Orientation
<<data store>> SClassRegistration



```
AGraph { nodes=[
   ANode { refCls=SClassRegistration
    serviceCls=DataController
    outClses=[MgEnrolmentProcessing]
    actSeq=[
     MAct{actName=newObject,pstStates=[Created]} ]
    init=true}
   ANode { refCls=Orientation
    serviceCls=DataController
    outClses=[MgEnrolmentProcessing]
    actSeq=[
     MAct{actName=newObject,pstStates=[Created]} ]
    init=true}
   ANode { refCls=MgEnrolmentProcessing
    nodeType=Merge,outClses=[EnrolmentClosure]}
   ANode { refCls=EnrolmentClosure,
    serviceCls=DataController,
    actSeq=[
     MAct{actName=newObject,pstStates=[NewObject]},
     MAct{actName=setDataFieldValues
              fieldNames=["sclassRegist","orient"]
              pstStates=[Created]} ]}
 ]}
```