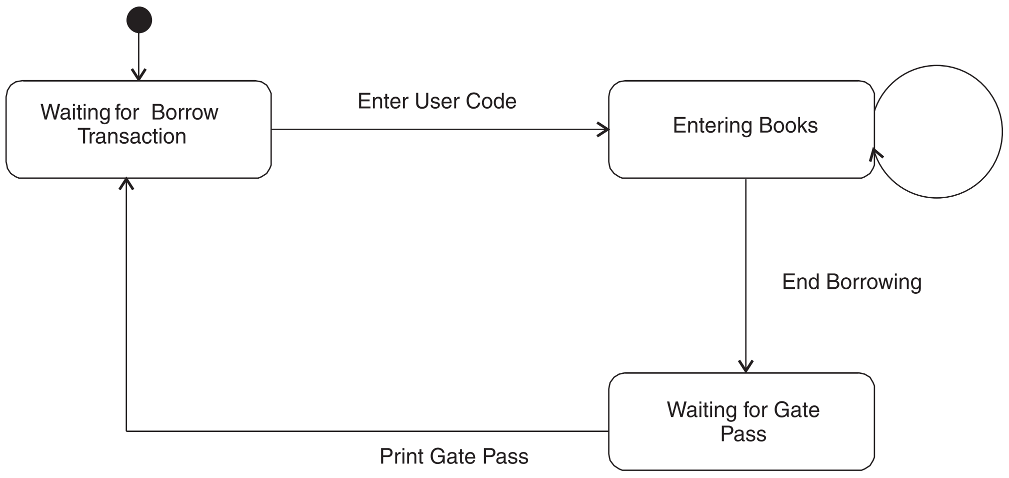

Fig. 9.14. Borrow book use case state (or statechart) diagram

9.13 WORKFLOWS AND ACTIVITY DIAGRAMS

Business processes can be described in the form of high-level flows of work and objects. Activity diagrams best depict these *workflows*. Usually, these diagrams are developed for important workflows, and not for all workflows. A workflow starts with an initial state and ends with an exit state. Although used for workflows, they are flexible enough to depict system operations as well.

Use cases, sequence diagrams, collaboration diagrams (to be described in the chapter dealing with object-oriented design), and statecharts model the dynamics of a system. Whereas use cases are very high-level artifacts for depicting system dynamics, sequence and collaboration diagrams are concerned with flow of control from object to object, and statecharts deal with flow of control from state to state of a system, use case, or of an object. An activity diagram is a special case of statecharts in which flow of control is depicted from activity to activity.

An activity is an ongoing non-atomic execution of a state machine. An activity diagram is a directed graph where nodes represent activity states and action states, and arcs represent transitions from state to state or flows of control. Whereas action states result from executable computations and are atomic in nature not being amenable for further breakdown, activity states are non-atomic that can be decomposed further into a set of activity and action states. Action states are not interrupted and generally take insignificant execution time whereas activity states may be interrupted and take some time to complete.

The common transition (or flow of control) takes place in a sequential manner. However, *activity diagrams* can also depict more realistic transitions involving *branching* and *concurrency*. Modelling concurrency requires *forking* and *joining*. Details of these are given below with the help of an example. Activity diagrams are often extended to include flow of objects showing change in state and attribute values. Further, for easy comprehensibility, one often organizes states in the activity diagram into related groups and physically arranges them in vertical columns that look like *swimlanes*. The notations used in an activity diagram are given in Fig. 9.15.

We give an example of workflow and an activity diagrammatic representation of the issue of general books, reserve books, and reference books in Fig. 9.16. In Fig. 9.16, the action state is *Request Issue of a Book*, whereas all other states are activity states. There are many cases of branching, whereas there is one case of concurrency involving updating the records and printing the gate pass that result in forking and joining. Notice the flow of *Book* object during the execution of *Update Records* state. State of the object is written below the object name. Notice also the use of the vertical lines to give the shape of the swimlanes.

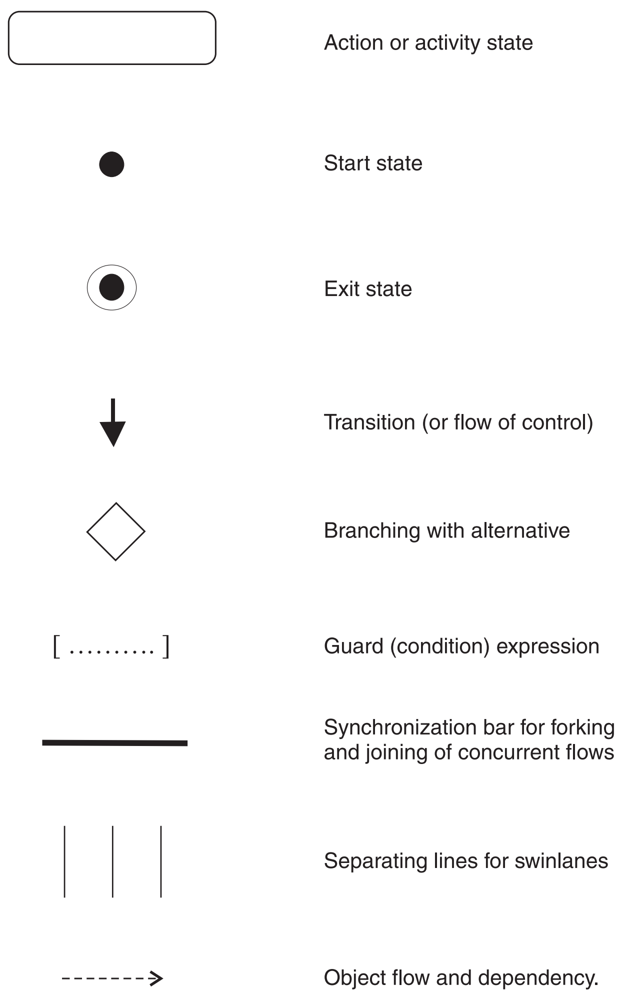
Action or activity state
Start state
Exit state
Transition (or flow of control)
Branching with alternative
[..........]
Guard (condition) expression
Synchronization bar for forking and joining of concurrent flows
Separating lines for swinlanes
Object flow and dependency.

Fig. 9.15. Notations used in activity diagrams

Before ending this chapter we would like to reiterate that Rational Unified Process model emphasizes incremental, iterative development. Thus, in the beginning, only the very basic user requirements are taken up. The inception phase may constitute only up to 10% of the total number of requirements for which use cases are developed and specification are written. In *iteration* 1 of the *elaboration* phase, domain class objects and their most useful parameters and operations are identified, system sequence diagrams are developed, contracts for system operations are written, and only association relationship between classes are established. This phase is followed by design and code and unit test phases. Meanwhile the analysis team firms up some more requirements. Iteration 2 of the elaboration phase begins thereafter. It is in iteration 2 or in subsequent iterations that relationships among classes, statechart, activity diagrams, and grouping models into packages are defined.

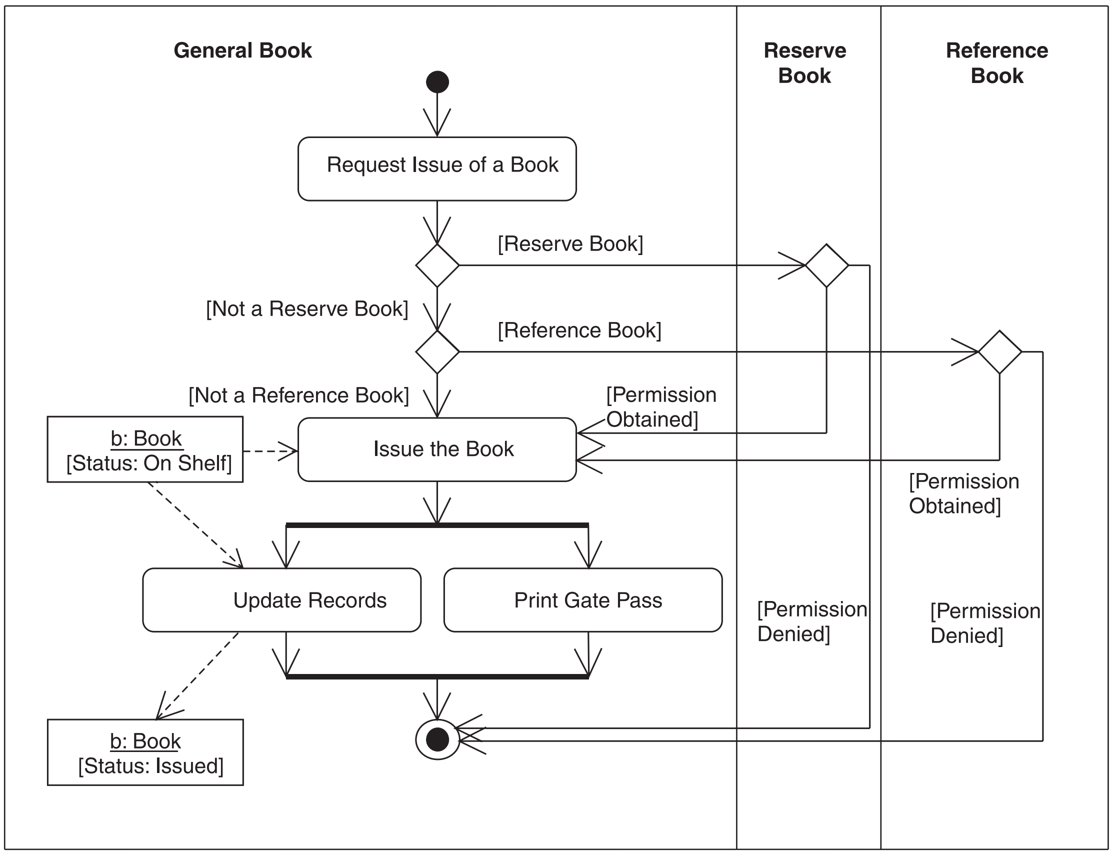

Fig. 9.16. Activity diagram for issue of various categories of books

REFERENCES

Beck, K. and W. Cunningham (1989), A Laboratory for Object-oriented Thinking, Proceedings of OOPSLA 1989, SIGPLAN Notices. Vol. 24, No.10.

Booch, G. (1994), Object-oriented Analysis and Design with Applications, Addison-Wesley, Reading, Mass, 2nd Edition.

Booch, G., J. Rumbaugh and I. Jacobson (2000), The Unified Modeling Language User Guide, Addison-Wesley Longman (Singapore) Pte. Ltd., Low Price Edition.

Coad, P. and E. Yourdon, (1991), Object-oriented Analysis, Second Edition, Englewood Cliffs, Yourdon Press, New Jersey.

Jacobson, I., M. Christerson, P. Jonsson and G. Övergaard (1992), Object-oriented Software Engineering: A Use Case Driven Approach, Addison-Wesley (Singapore) Pte. Ltd., International Student Edition.

Larman, C. (2000), Applying UML and Patterns: An Introduction to Object-oriented Analysis and Design, Addison-Wesley, Pearson Education, Inc., Low Price Edition.

Pressman, R.S. (1997), Software Engineering: A Practitioner's Approach, McGraw-Hill, International Editions.

Rumbaugh, J., M. Blaha, W. Premerlani, F. Eddy and W. Lorensen (1991), Object-oriented Modeling and Design, Englewood Cliffs, Prentice-Hall, New Jersey.

Wirfs-Brock, R., B. Wilkerson and L. Wiener (1990), Designing Object-oriented Software, Englewood Cliffs, Prentice Hall, New Jersey.