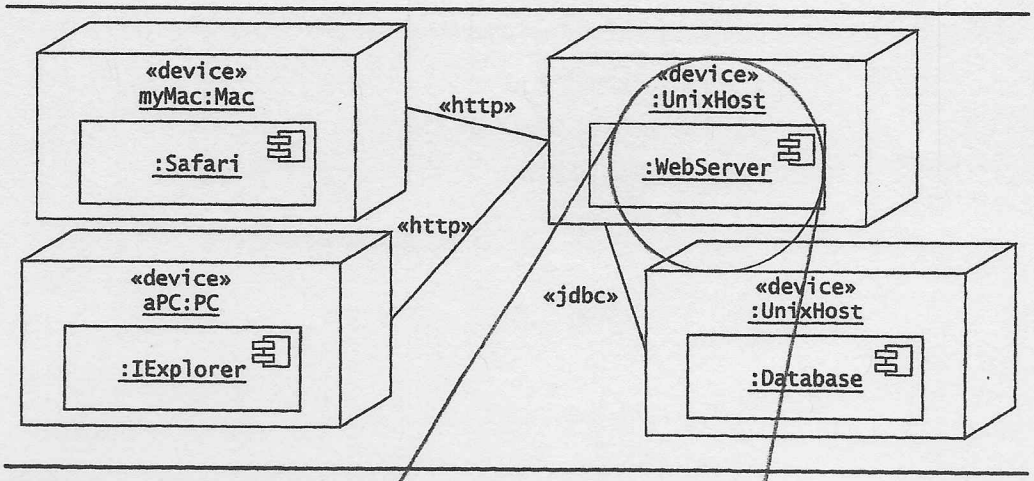


Figure 7-2 A UML deployment diagram representing the allocation of components to different nodes. Web browsers on PCs and Macs can access a WebServer that provides information from a Database.

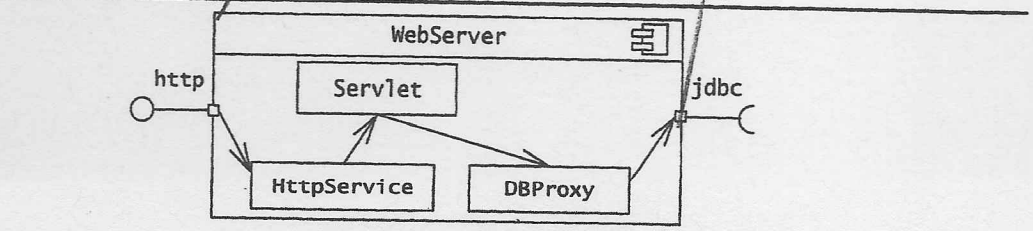


Figure 7-3 Refined view of the WebServer component (UML component diagram). WebServer provides an http interface and requires a jdbc interface. The http interface is realized by the HttpService class.

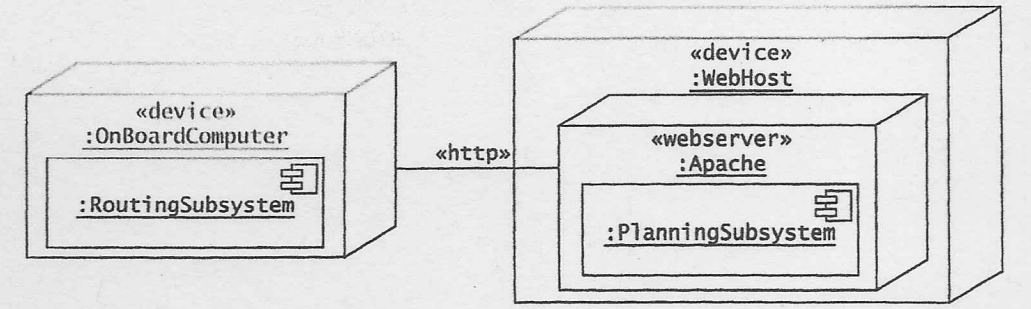
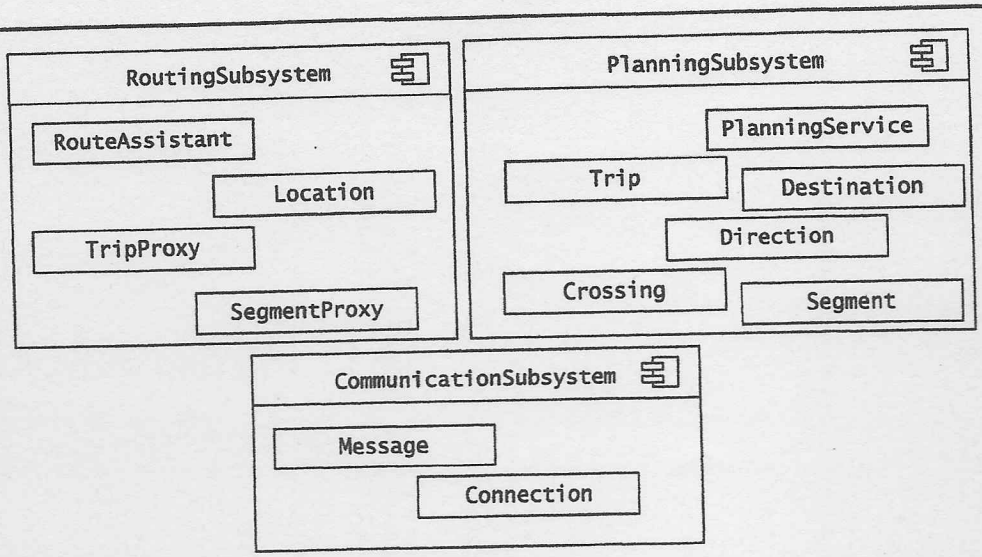
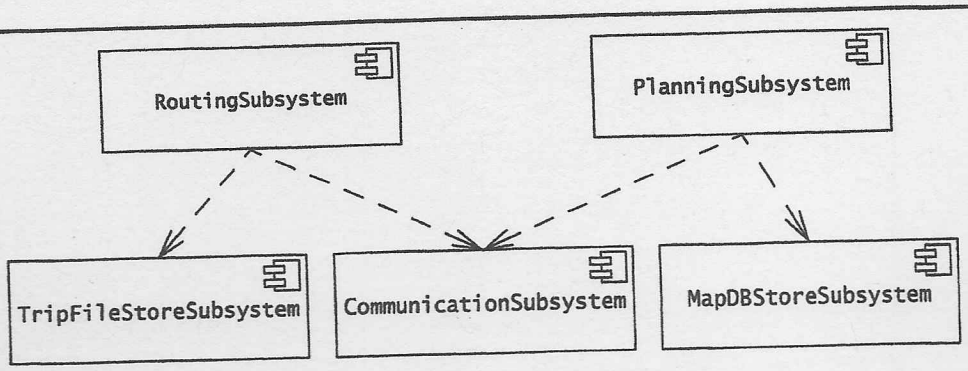


Figure 7-4 Allocation of `MyTrip` subsystems to devices and execution environments (UML deployment diagram). `RoutingSubsystem` runs on the `OnBoardComputer`; `PlanningSubsystem` runs on an `Apache` server.



CommunicationSubsystem	The CommunicationSubsystem is responsible for transporting objects from the PlanningSubsystem to the RoutingSubsystem.
Connection	A Connection represents an active link between the PlanningSubsystem and the RoutingSubsystem. A Connection object handles exceptional cases associated with loss of network services.
Message	A Message represents a Trip and its related Destinations, Segments, Crossings, and Directions, encoded for transport.

Figure 7-5 Revised design model for MyTrip (UML component diagram).



TripFileStoreSubsystem	The TripFileStoreSubsystem is responsible for storing trips in files on the onboard computer. Because this functionality is only used for storing trips when the car shuts down, this subsystem only supports the fast storage and loading of whole trips.
MapDBStoreSubsystem	The MapDBStoreSubsystem is responsible for storing maps and trips in a database for the PlanningSubsystem. This subsystem supports multiple concurrent Drivers and planning agents.

Figure 7-6 Subsystem decomposition of MyTrip after deciding on the issue of data stores (UML component diagram).

Trade-off between flat files, relational databases, and object-oriented databases

When should you choose flat files?

- Voluminous data (e.g., images)
- Temporary data (e.g., core file)
- Low information density (e.g., archival files, history logs)

When should you choose a relational or an object-oriented database?

- Concurrent accesses
- Access at finer levels of detail
- Multiple platforms or applications for the same data

When should you choose a relational database?

- Complex queries over attributes
- Large data set

When should you choose an object-oriented database?

- Extensive use of associations to retrieve data
- Medium-sized data set
- Irregular associations among objects

Figure 7-7 Trade-off between files and databases for storage management.

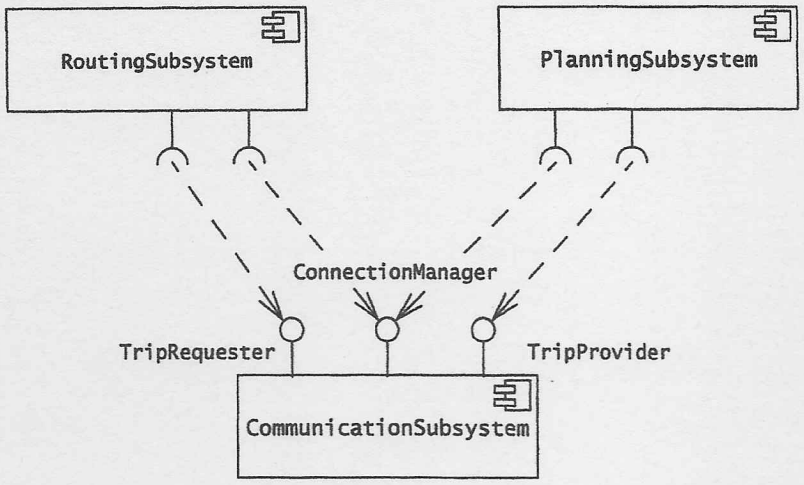


Figure 7-14 Refining the subsystem decompositions by identifying subsystem services (UML component diagram). The `CommunicationSubsystem` provides three services for managing connections, uploading trips, and downloading trips.

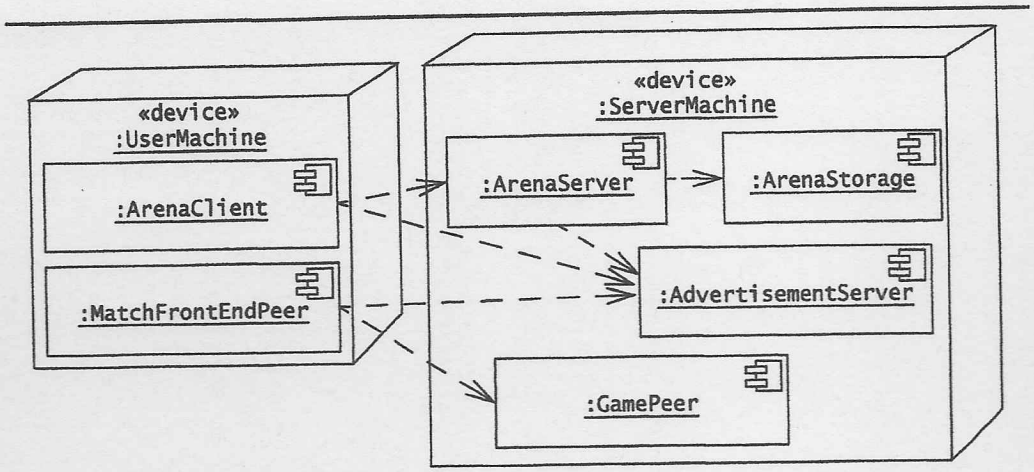


Figure 7-21 ARENA hardware/software mapping (UML deployment diagram). Note that each run-time component may support several subsystems.

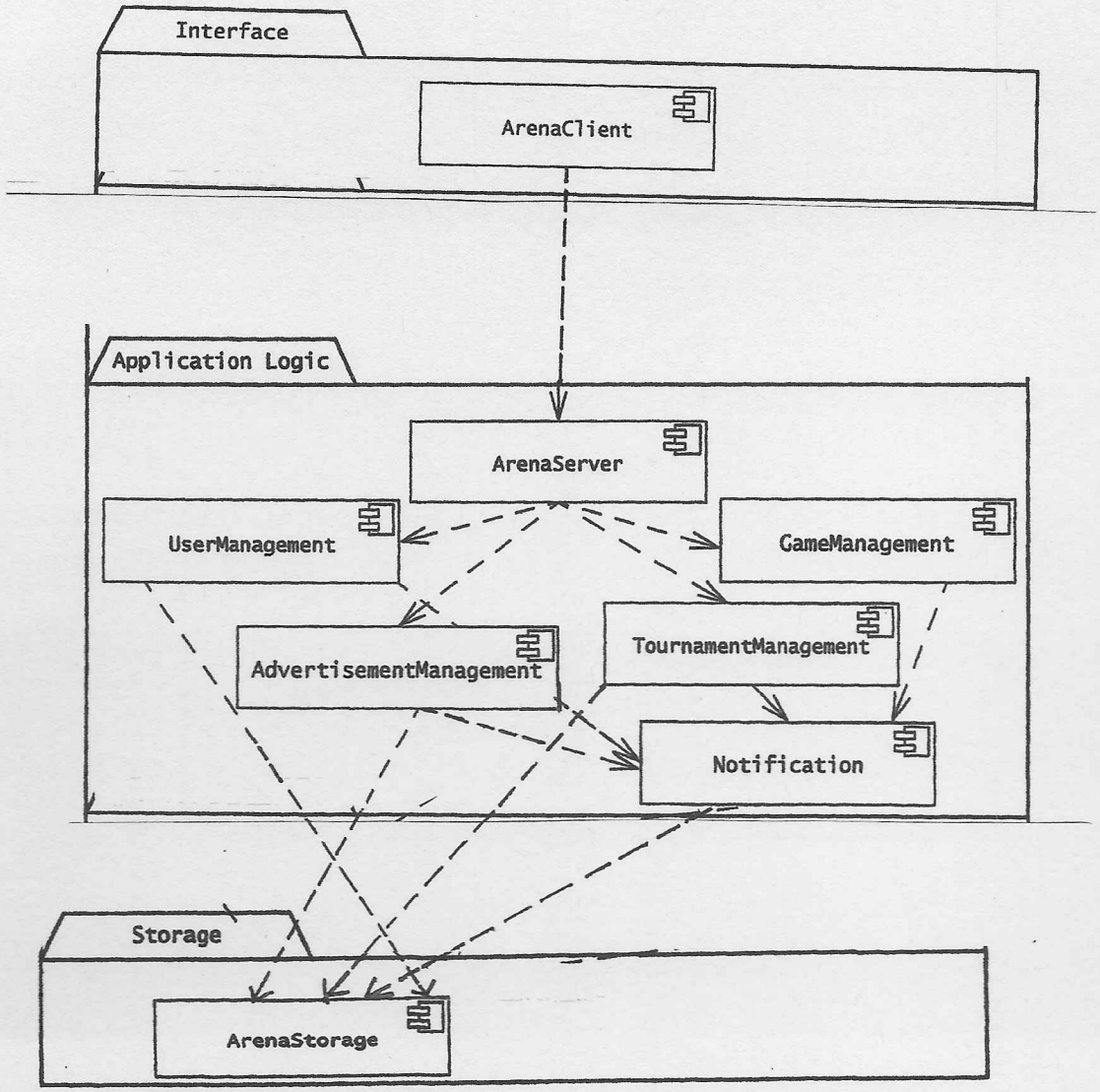


Figure 7-19 ARENA subsystem decomposition, game organization part (UML component diagram, layers shown as UML packages).

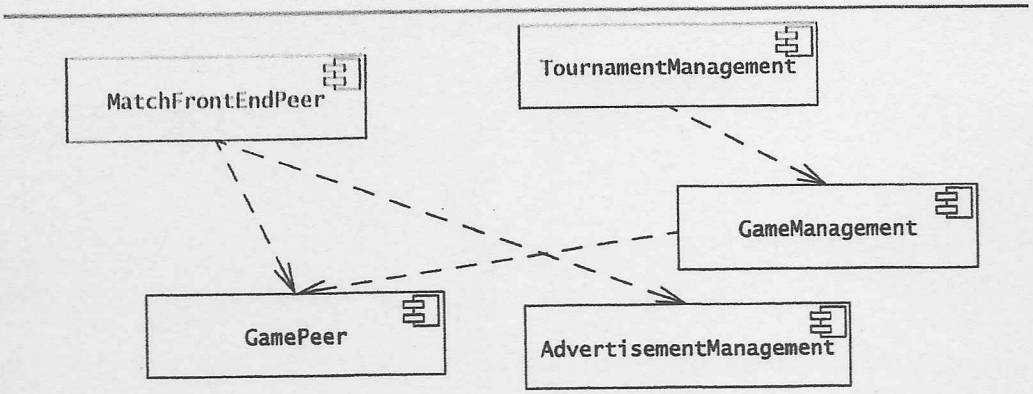


Figure 7-20 ARENA subsystem decomposition, game playing part (UML component diagram).