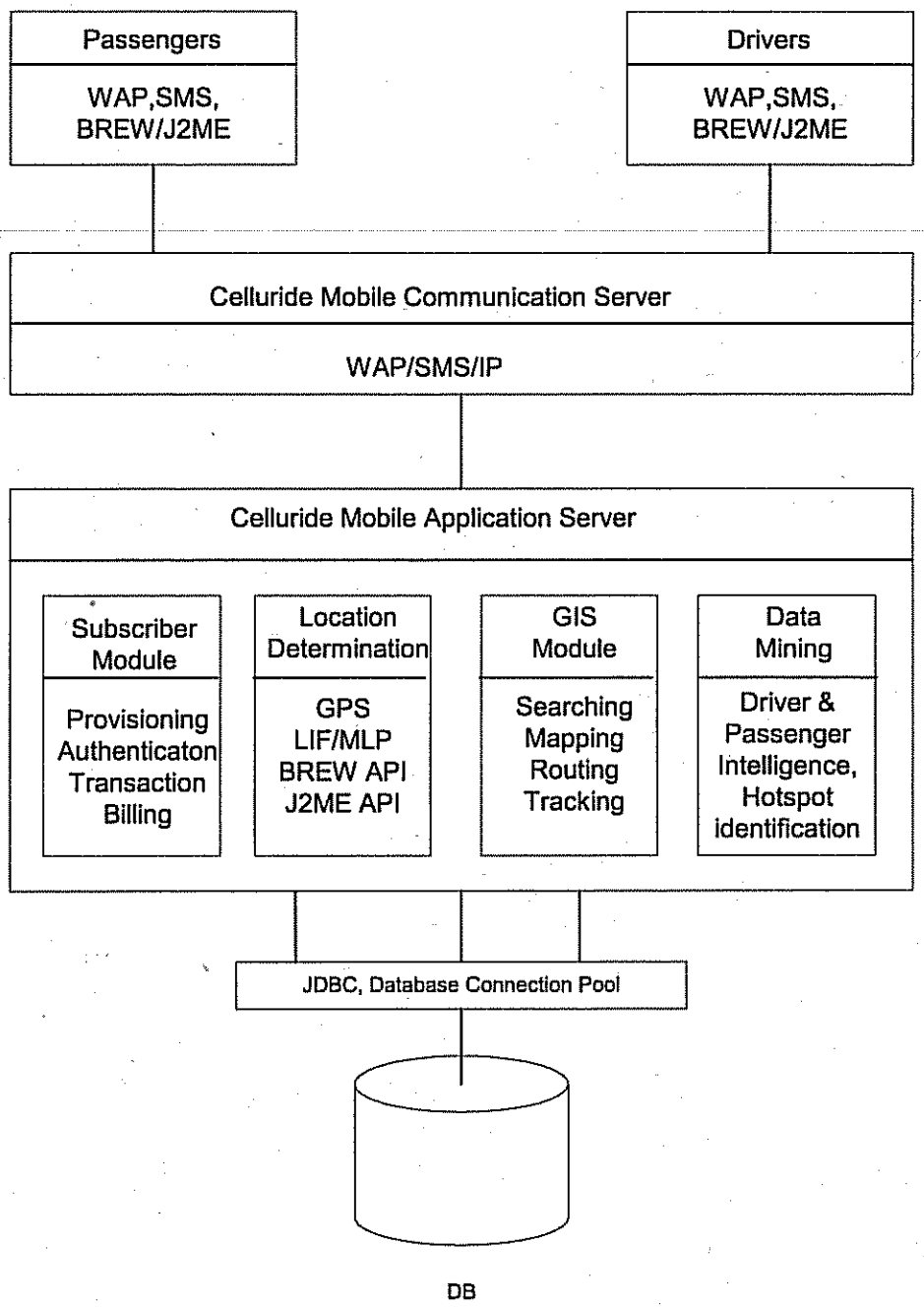

Figure 1. Celluride High-level Architecture

Subscriber Module:

Responsible for all subscriber related operation, including:

(1) Provisioning: all subscribers need to be registered in the system to use the Celluride's service.
(2) Authentication: authenticate an incoming subscriber's identity
(3) Transaction: each time a Celluride transaction occurs, there will be multiple transaction states involved. All transaction details will be recorded thru this module.
(4) Billing: handles all billing related issues. Billing can be per subscriber, per transaction, or tightly integrated with the carriers or service providers.

Location Determination Module:

Responsible for determining the x, y, z of the subscriber based on a variety of methods:

(1) GPS: applied to a GPS-enabled device. Depending on the device's GPS output interface, location information can be fetched via standard API or reading raw GPS sentences from the communication port.
(2) LIF/MLP: using the mobile location protocol defined by the location interoperability forum, the subscriber's location can be queried from a service provider. This is a network-centric solution.
(3) BREW: Qualcomm's IPosDet API supports the fetching of location in a BREW-enabled handset.
(4) J2ME: based in JSR 179 specification, the device's location can be fetched using industry-standard Java Micro Edition.

GIS Module

Responsible for all spatial data oriented operations, including:

(1) Searching: supports the "Find Nearest" spatial searching. Identifies all drivers that are within the passenger's search radius.
(2) Mapping: provides maps to drivers, passengers, and backend system operators.
(3) Routing: finds the shortest path or driving direction for passengers or drivers
(4) Tracking: the movement of passengers and drivers can be continuously tracked to ensure they are moving at the right direction or if they suddenly change course.

Data Mining Module:

An optional add-on to the base Celluride system, aiming to provide intelligence information and data mining capabilities based on the subscriber's database Celluride possess, and the patterns and statistics of subscriber's past transaction behavior and credibility.

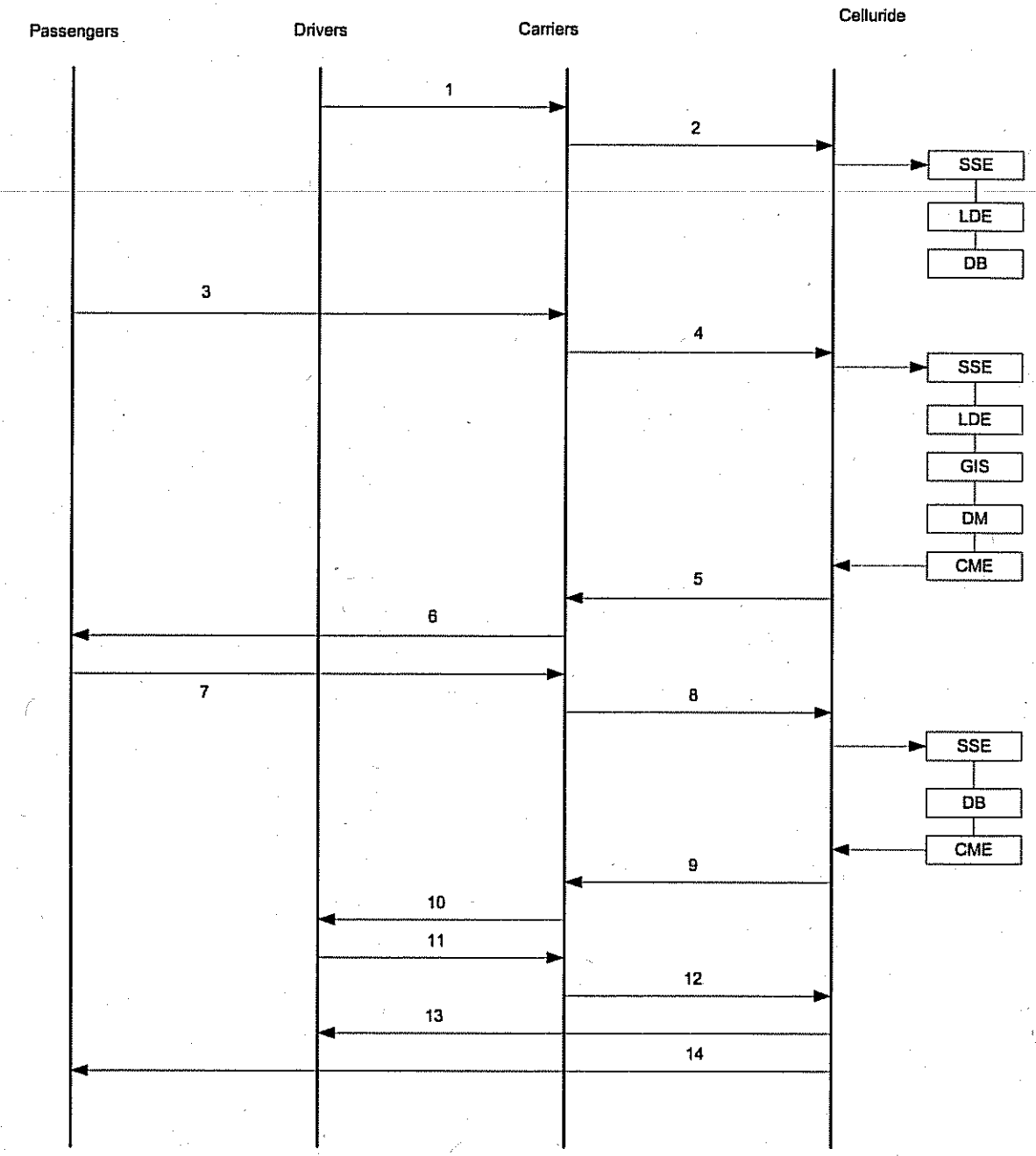

Figure 2. Celluride End-to-End Message Flow

1. A driver who needs to find new passengers pushes a button on his cell phone. A message indicating the driver's availability and location information are sent to the driver's carrier proxy server or gateway.
2. Carrier proxy determines that this packet belongs to Celluride and forwards it to Celluride's server. This packet will first be processed by Celluride's Subscriber & Service Engine (SSE), which authenticates the driver. Then Celluride's Location Determination Engine is called to retrieve the driver's location. The location determination method can be handset based or network based, depending on the cell phones location technology. Finally, the driver's latest location and availability information are saved in the database thru the DB module.
3. A passenger looking for a ride pushes a button on his mobile handset hosting Celluride's application front-end GUI, indicating the passenger's need for a ride. This information with the passenger's location is sent to passenger's carrier proxy server.
4. The carrier proxy determines that this message belongs to Celluride and forwards it to Celluride's server. The server determines this message is from a passenger. It then calls the SSE to authenticate the passenger, retrieves the passenger's location, and then calls the GIS module to find out all the drivers that are close to this passenger's location. The selected drivers are further filtered by the Data Mining (DM) module. The final list of drivers are those that meet the user's preset criteria for vehicle options.
5. The Celluride's Communication Engine (CME) is invoked and the selected lists of drivers are pushed back to the passenger, thru the carrier proxy.
6. The carrier forwards the above message to the passenger.
7. The passenger looks at this list and picks the final choice of the driver. A message is generated.
8. The carrier proxy determines this message belongs to Celluride and forwards it. The Celluride server invokes SSE to log user's transaction information and saves any necessary billing information in the DB
9. Celluride invokes the CME to send a notification message to the selected driver.
10. The carrier pushes this message to the driver's handset.
11. The driver either accepts or denies the request. An indication message was generated.
12. Carrier forwards driver's decision over to Celluride.
13. Celluride logs the transaction information for billing purpose and sends the routing information and maps showing the driver how to get to the passenger's location.
14. Celluride sends a confirmation message to the passenger.