ĮVADAS Į OBJEKTIŠKAI ORIENTUOTĄ PROGRAMAVIMO KALBĄ JAVA

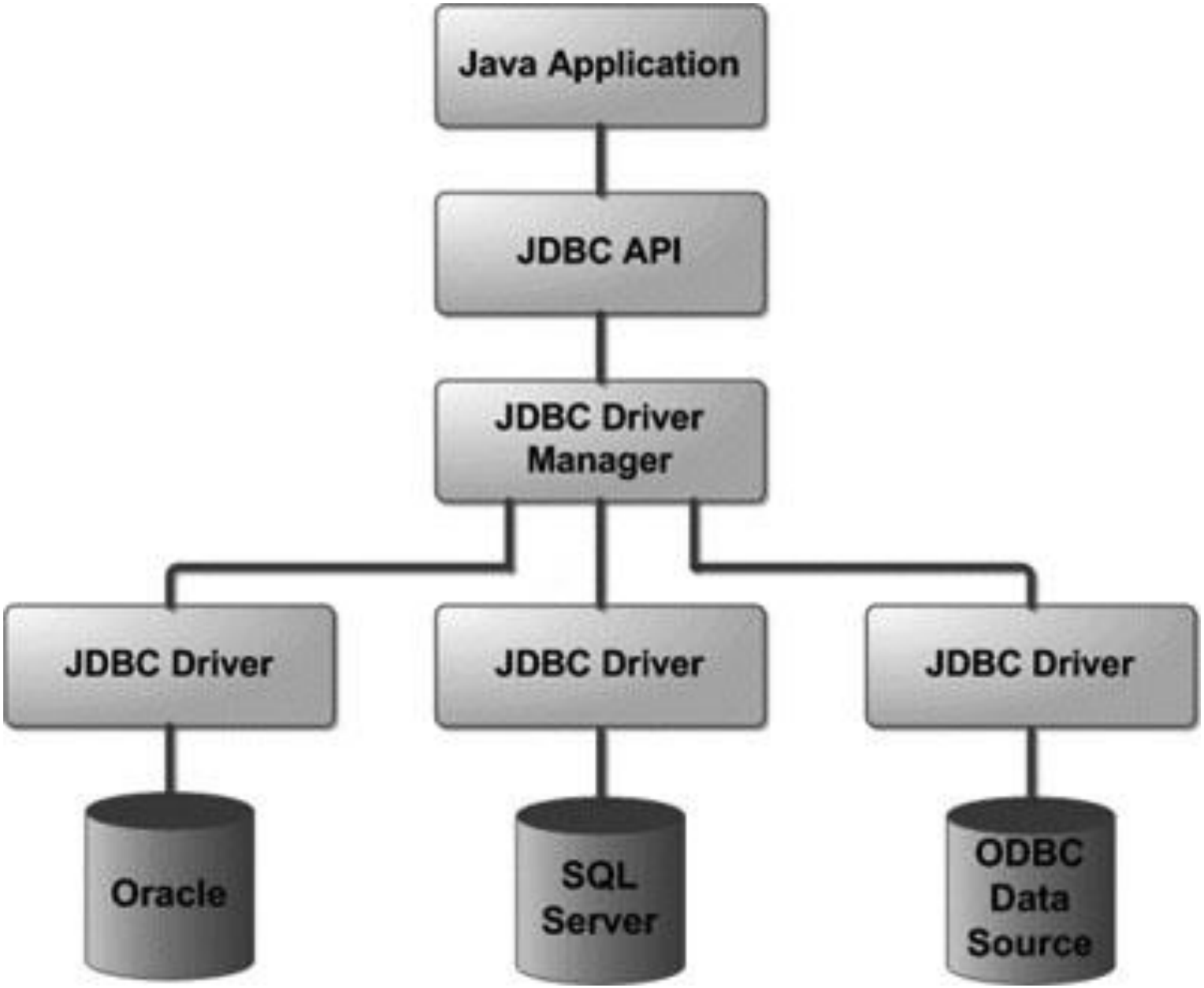
Marius Gžegoževskis

2 paskaita

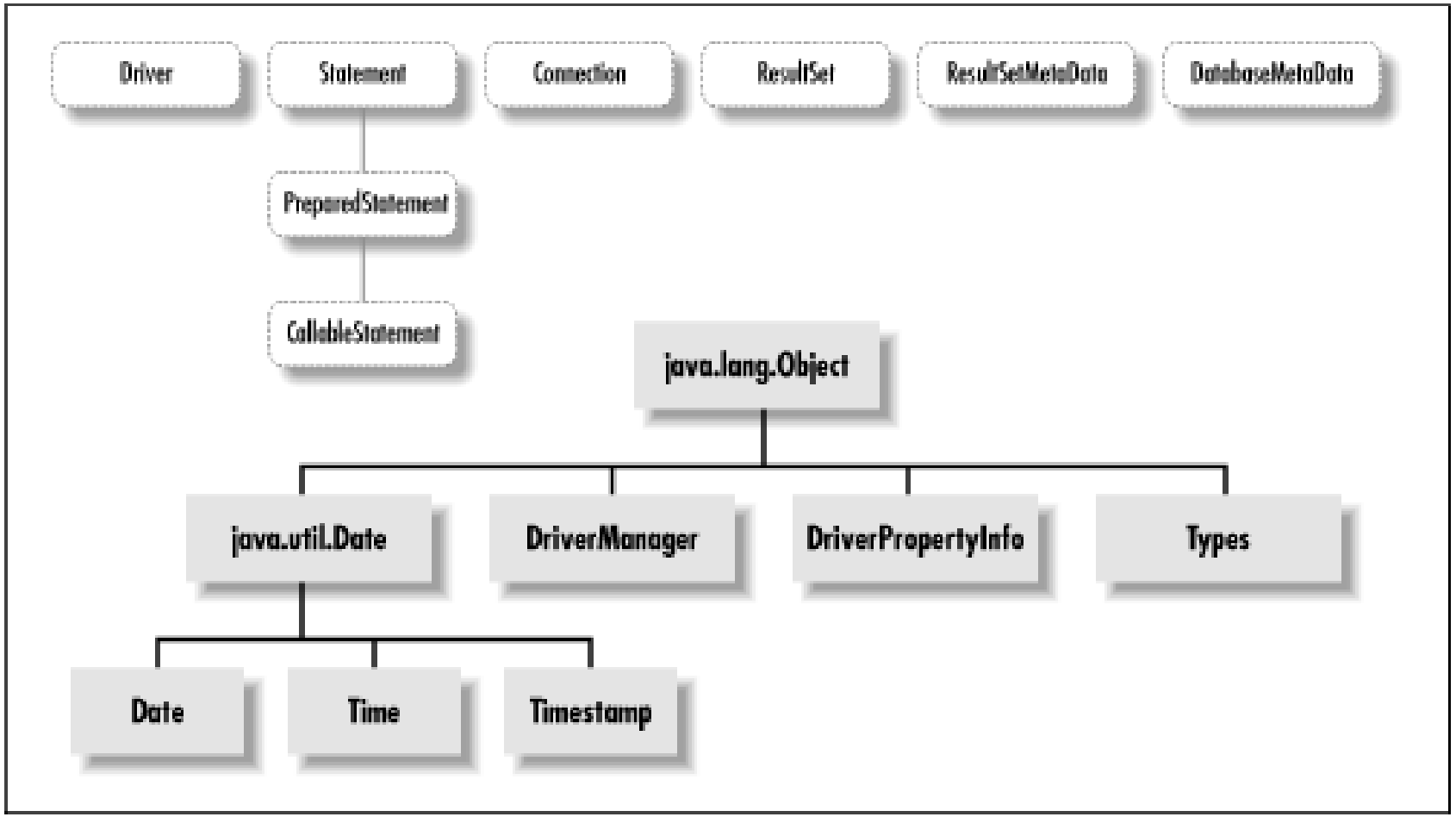
JDBC API (APPLICATION PROGRAMING INTERFACE)

Nuorodos:

- JAVA JDBC (JAVA DATABASE CONNECTION):
- http://www.tutorialspoint.com/jdbc/index.htm
- Kompiuteriniai terminai:
- http://www.tutorialspoint.com/computer_glossary.htm


Java Application
JDBC API
JDBC Driver Manager
JDBC Driver
JDBC Driver
JDBC Driver
Oracle
SQL Server
ODBC Data Source

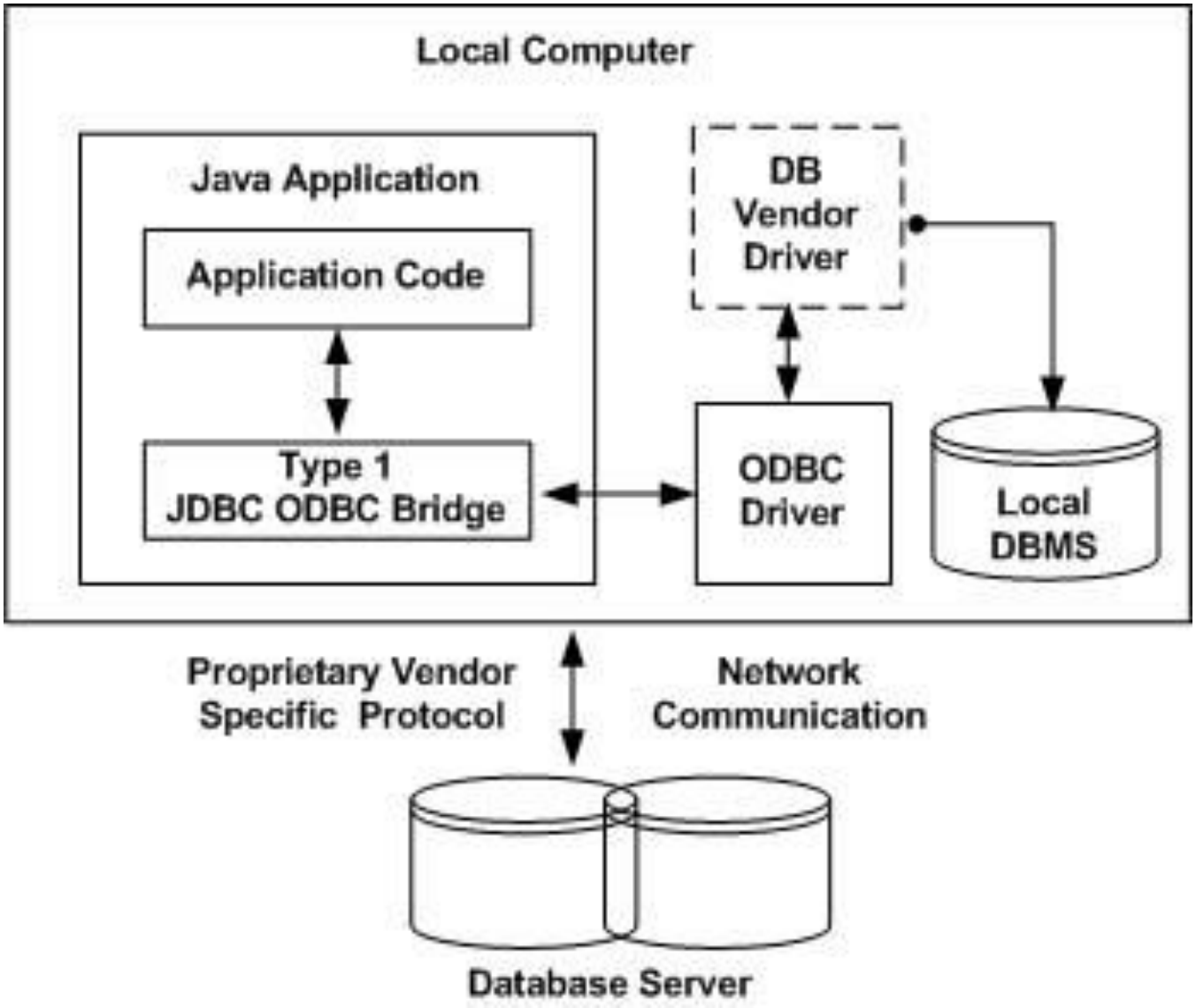
JAVA + DUOMENŲ BAZĖS = JDBC API



TO HELP YOU UNDERSTAND WHAT DIFFERENT DRIVERS REQUIRE, JAVASOFT HAS DEFINED THE FOLLOWING DRIVER CATEGORIZATION SYSTEM

Type 1. These drivers use a bridging technology to access a database. Bridge solutions generally require software to be installed on client systems, meaning that they are not good solutions for applications that do not allow you to install software on the client.

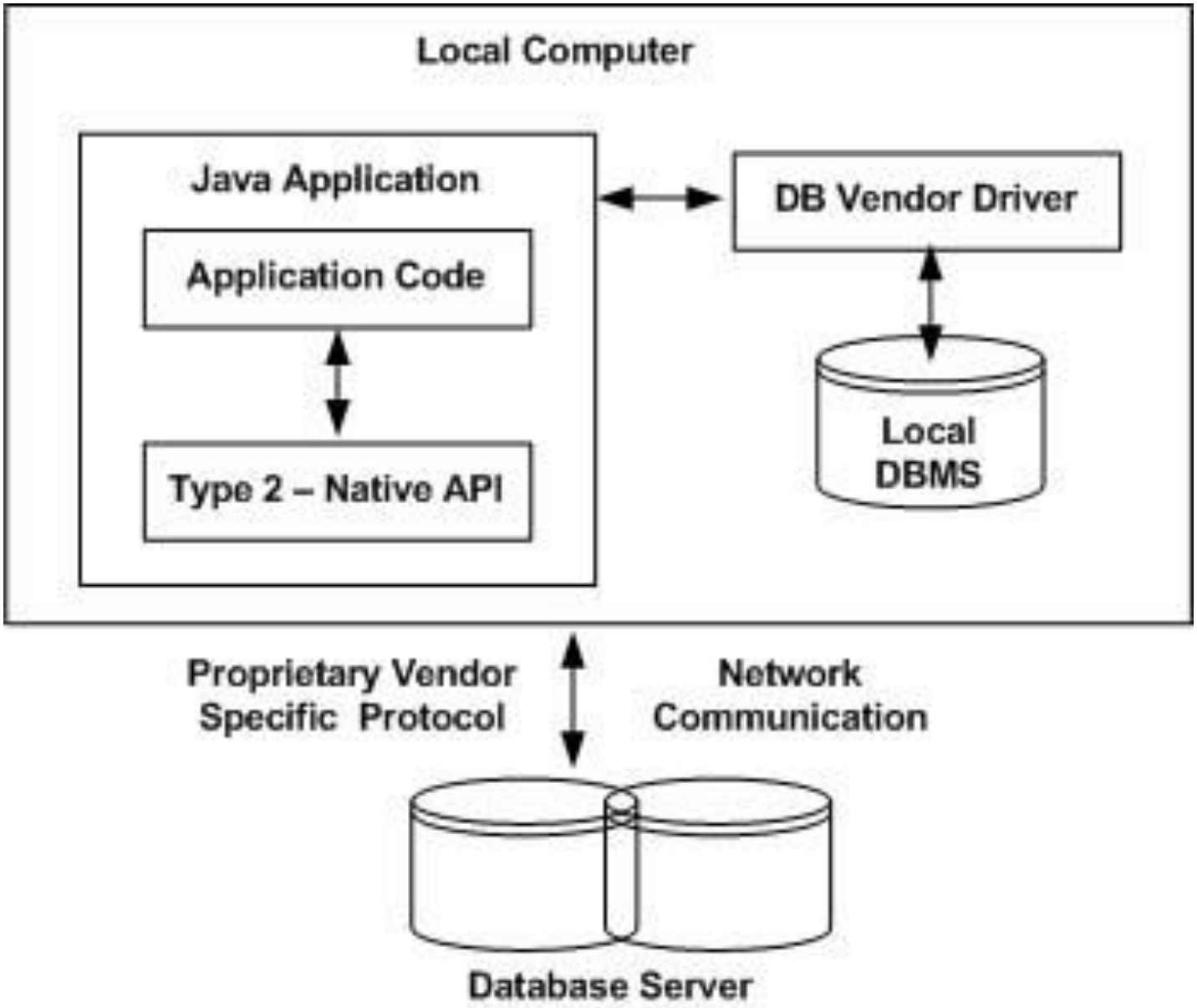
TYPE 1


Local Computer
Java Application
Application Code
Type 1
JDBC ODBC Bridge
DB
Vendor
Driver
ODBC
Driver
Local
DBMS
Proprietary Vendor
Specific Protocol
Network
Communication
Database Server

TYPE 2 (1 OF 2)

- **Type 2.** The type 2 drivers are native API drivers. This means that the driver contains Java code that calls native C or C++ methods provided by the individual database vendors that perform the database access. Again, this solution requires software on the client system.

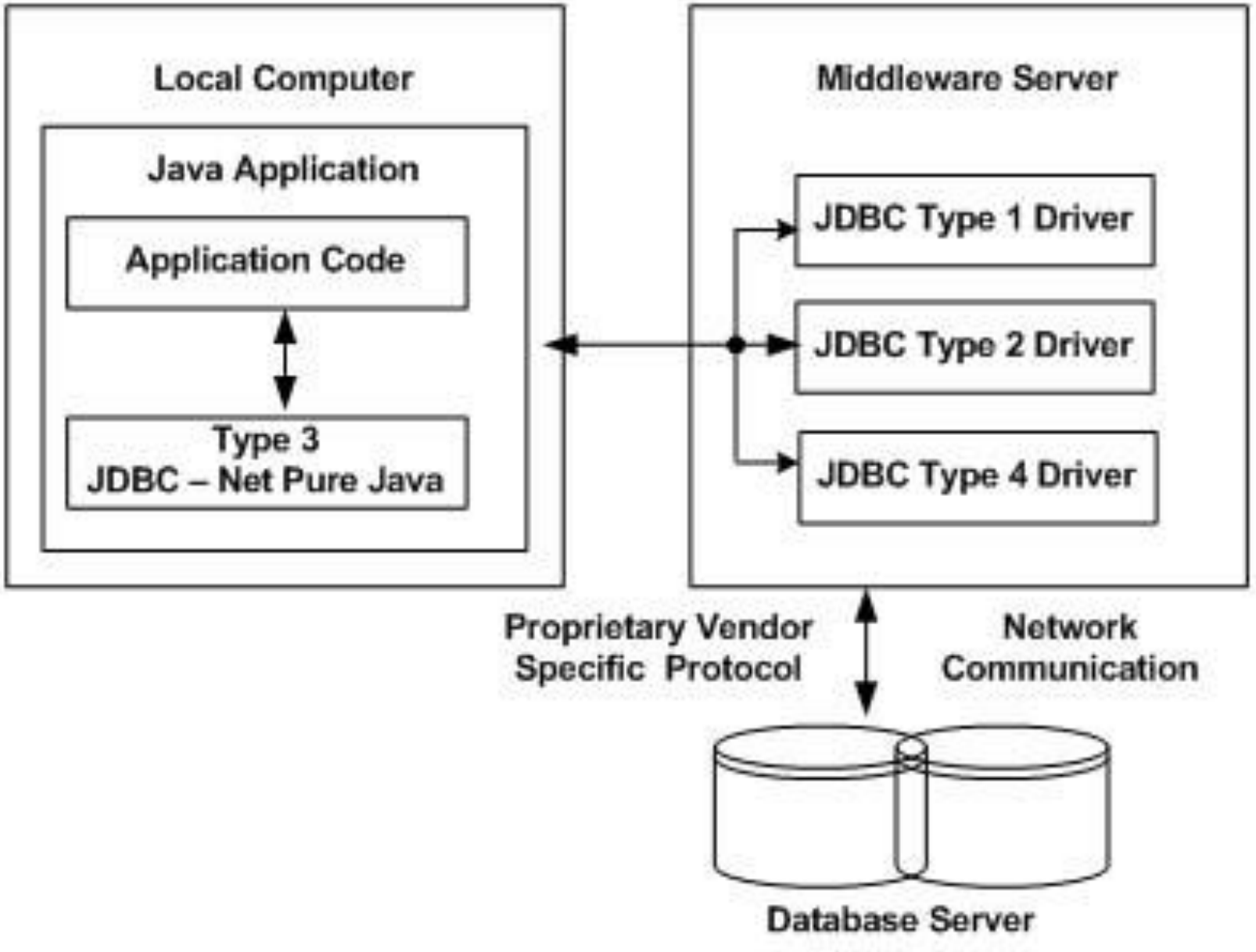
TYPE 2 (2 OF 2)


Local Computer
Java Application
Application Code
Type 2 – Native API
DB Vendor Driver
Local
DBMS
Proprietary Vendor
Specific Protocol
Network
Communication
Database Server

TYPE 3 (1 OF 2)

- **Type 3.** Type 3 drivers provide a client with a generic network API that is then translated into database specific access at the server level. In other words, the JDBC driver on the client uses sockets to call a middleware application on the server that translates the client requests into an API specific to the desired driver. As it turns out, this kind of driver is extremely flexible since it requires no code installed on the client and a single driver can actually provide access to multiple databases.

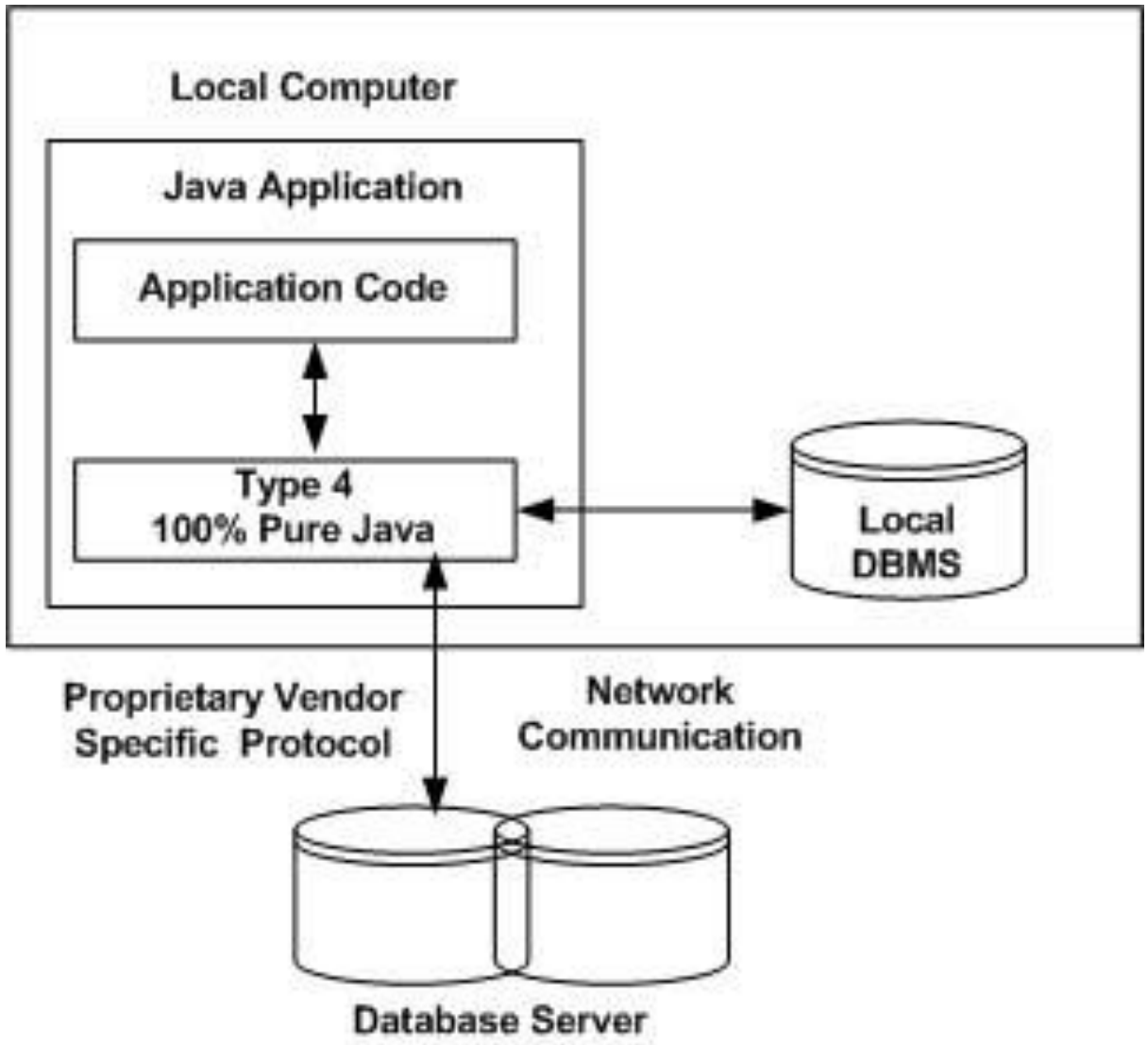
TYPE 3 (2 OF 2)


Local Computer
Java Application
Application Code
Type 3
JDBC – Net Pure Java
Middleware Server
JDBC Type 1 Driver
JDBC Type 2 Driver
JDBC Type 4 Driver
Proprietary Vendor
Specific Protocol
Network
Communication
Database Server

TYPE 4

- **Type 4.** Using network protocols built into the database engine, type 4 drivers talk directly to the database using Java sockets. This is the most direct pure Java solution. In nearly every case, this type of driver will come only from the database vendor.

TYPE 4 (2 OF 2)


Local Computer
Java Application
Application Code
Type 4
100% Pure Java
Local
DBMS
Proprietary Vendor
Specific Protocol
Network
Communication
Database Server

Vendor	Type	Supported Databases
Agave Software Design	3	Oracle, Sybase, Informix, ODBC supported databases
Asgard Software	3	Unisys A series DMSII database
Borland	4	InterBase 4.0
Caribou Lake Software	3	CA-Ingres
Center for Imaginary Environments	4	mSQL
Connect Software	4	Sybase, MS SQL Server
DataRamp	3	ODBC supported databases
IBM	2/3	IBM DB2 Version 2
IDS Software	3	Oracle, Sybase, MS SQL Server, MS Access, Informix, Watcom, ODBC supported databases
InterSoft	3	Essentia
Intersolv	2	Oracle, Sybase
JavaSoft	1	ODBC supported databases
OpenLink	3	CA-Ingres, Postgres95, Progress, Unify
SAS	4	SAS, and via SAS/ACCESS, Oracle, Informix, Ingres, and ADABAS
SCO	3	Informix, Oracle, Ingres, Sybase, Interbase
StormCloud Development	3	ODBC supported databases
Sybase	3/4	Sybase SQL Server, SQL Anywhere, Sybase IQ, Replication Server
Symantec	3	Oracle, Sybase, MS SQL Server, MS Access, Watcom, ODBC supported databases
Visigenic	3	ODBC supported databases
WebLogic	2/3	Oracle, Sybase, MS SQL Server/ODBC supported databases

THE JDBC KLASĖS PRISIJUNGIMUI PRIE DUOMENŲ BAZĖS SUKŪRIMUI

- **java.sql.Driver :**

- **java.sql.DriverManager:**

- **java.sql.Connection:** Ši klasė siunčia seką SQL užklausų į duomenų bazę ir nusprendžia ar įvykdyti (angl. commit) ar atmesti šiuos sakinius (SQL užklausas).

```
public class SelectApp {
public static void main(String args[]) {
String url ="jdbc:msql://athens.imaginary.com:4333/db_web";  // Duomenų bazės prisijungimo adresas.
try {
        Class.forName("imaginary.sql.iMsqlDriver");  // Duomenų bazės priklausomai nuo DB yra nurodomas draiveris.
}
catch( Exception e ) {
        System.out.println("Failed to load mSQL driver."); return;
 }
try{
        Connection con = DriverManager.getConnection(url, "borg", ""); // Bandoma prisijungti su nurodytais parametrais
        Statement select = con.createStatement(); // Sukuriamas SQL užklausom skirtas objektas
        ResultSet result = select.executeQuery ("SELECT key, val FROM t_test"); // Rezultatui išvesti sukuriamas result
objektas, kuriame yra išsaugomas SQL sakinio „select.executeQuery ("SELECT key, val FROM t_test");“ rezultatas
        System.out.println("Got results:");
        while(result.next()) { // skaitoma po viena eilutę šiuo atveju iš result objekto rezultatų aibės
                int key = result.getInt(1);
                String val = result.getString(2);
                System.out.println("key = " + key);
                System.out.println("val = " + val);
}
select.close(); con.close();
} catch( Exception e ) {
e.printStackTrace(); } } }
```