Grand Valley State University
www.gvsu.edu

Software Engineering
OOP, OOD, OOA (2)
➔ Decomposition

Erik Fredericks // frederer@gvsu.edu

Adapted from materials provided by Byron DeVries, Jagadeesh Nandigam

Types of decomposition

Some design problems have no existing solutions

- Developers must decompose to isolate key problems

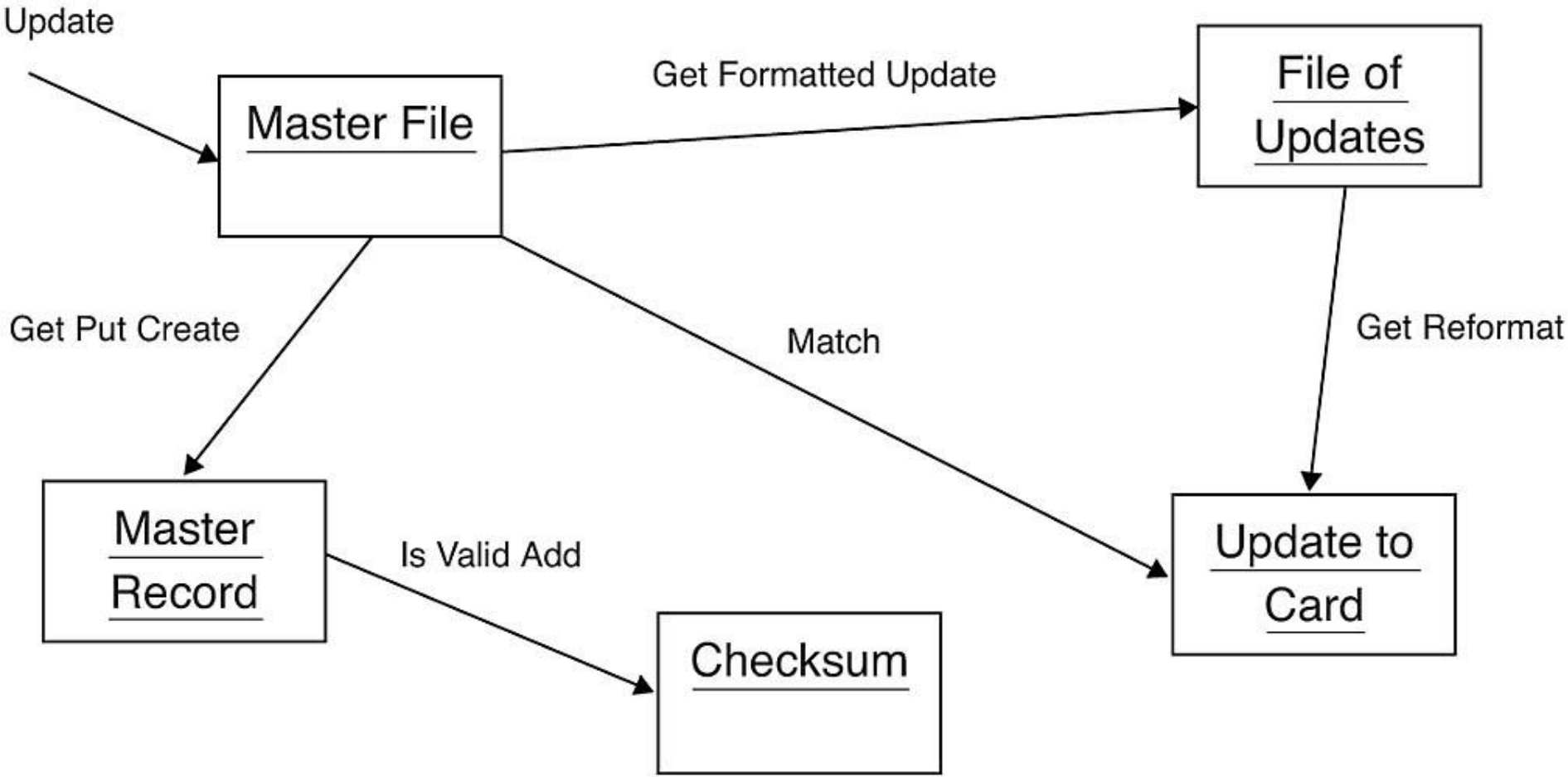
Some popular design methods:

- Functional decomposition
- Feature-oriented decomposition
- Data-oriented decomposition
- Process-oriented decomposition
- Event-oriented decomposition
- **Object-oriented** decomposition

Types of decomposition:

Object-Oriented decomposition:

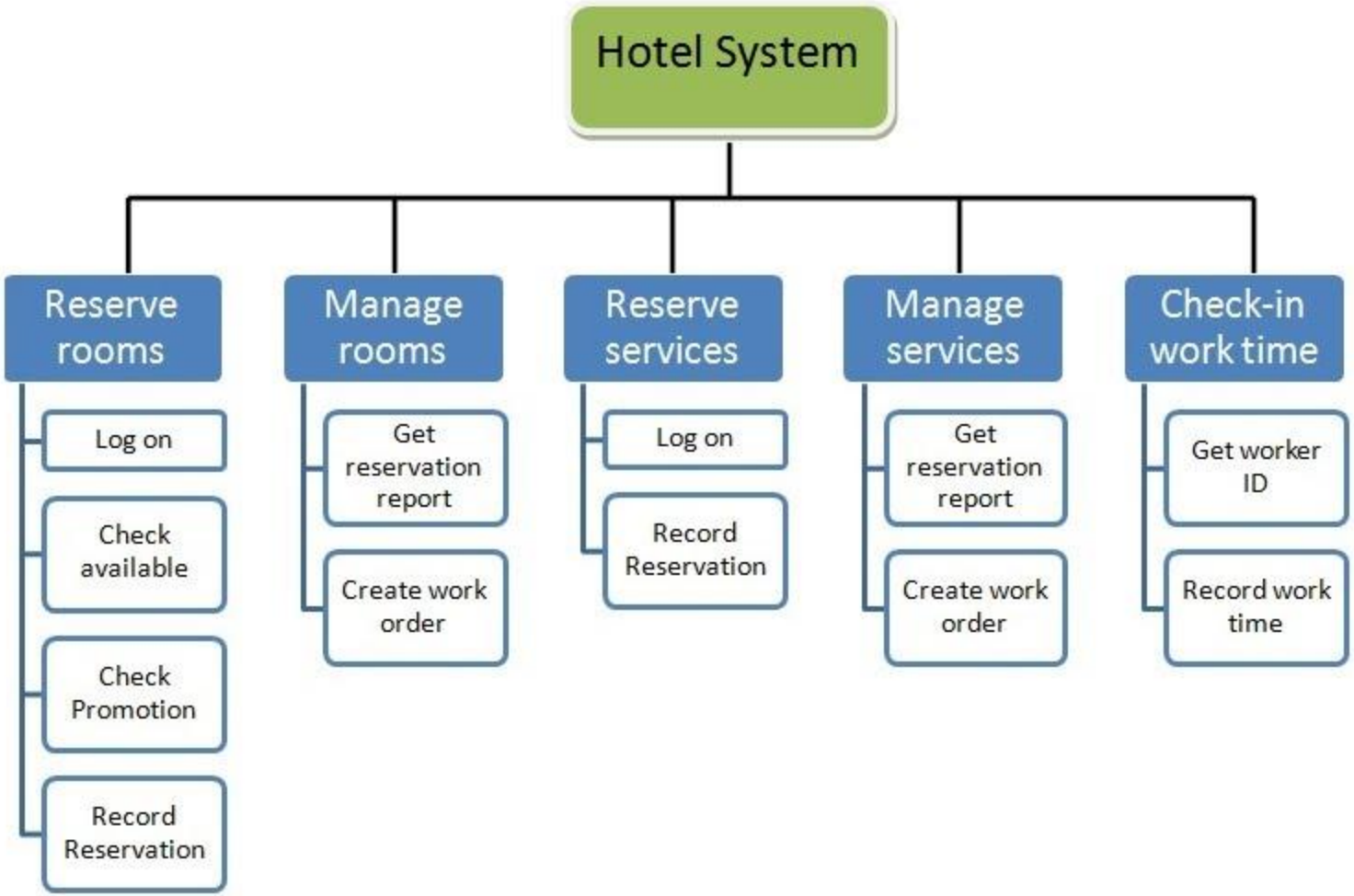
- Assigns objects to module
- **High-level** design:
  - Identifies the system's object types and explains how objects are related to one another
- **Lower-level** design:
  - Detail the objects' attributes and operations


Update
Master File
Get Formatted Update
File of Updates
Get Put Create
Match
Get Reformat
Master Record
Is Valid Add
Checksum
Update to Card

Types of decomposition:

Functional decomposition:

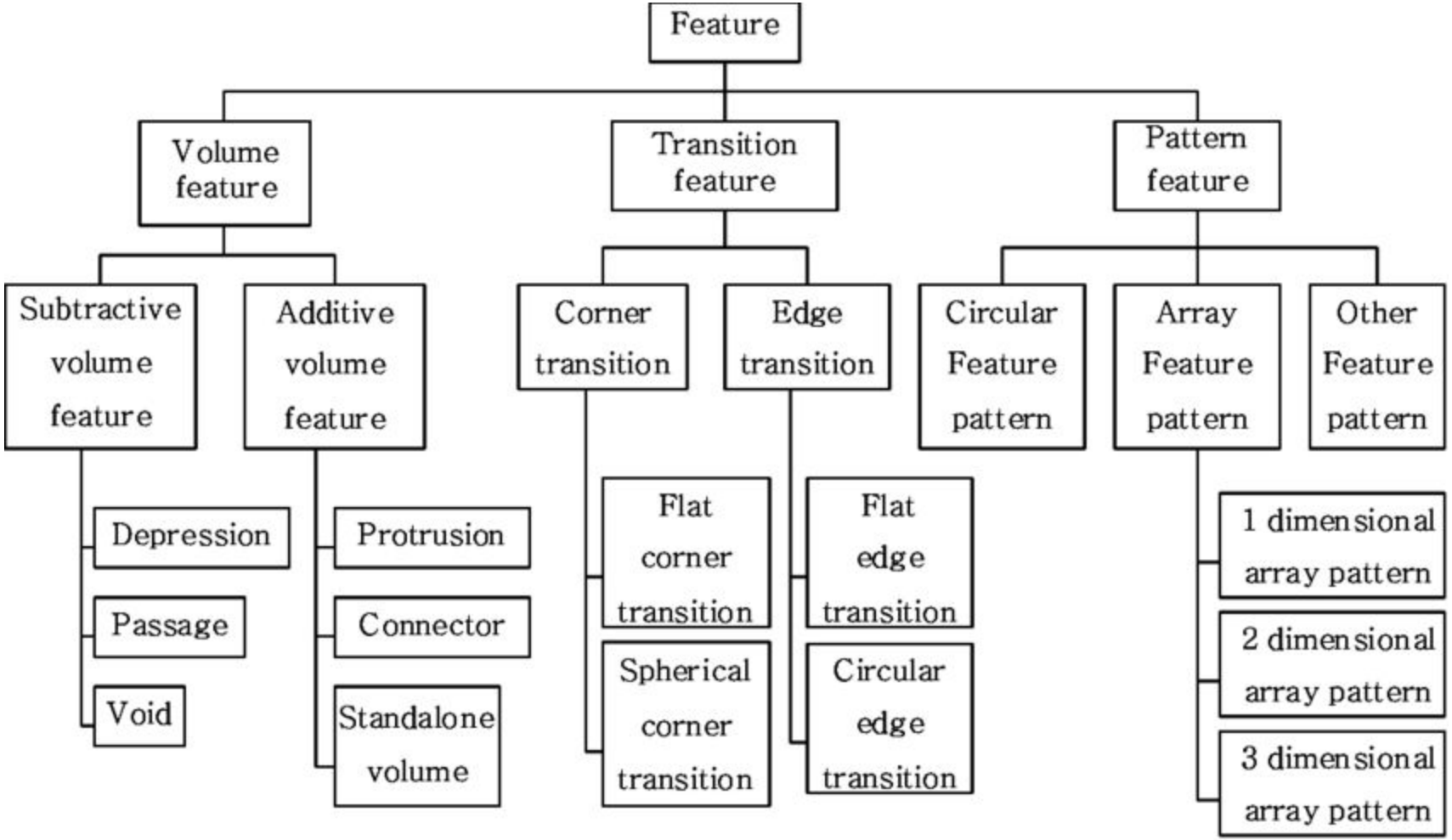
- Partitions functions or requirements into modules
- Begins with the functions that are listed in the requirements specification
- **Lower-level designs** divide these functions into sub-functions, which are then assigned to smaller modules
- Describes which modules (sub-functions) call each other


Hotel System
Reserve rooms
Log on
Check available
Check Promotion
Record Reservation
Manage rooms
Get reservation report
Create work order
Reserve services
Log on
Record Reservation
Manage services
Get reservation report
Create work order
Check-in work time
Get worker ID
Record work time

Types of decomposition

Feature-Oriented decomposition:

- Assigns features to modules
- **High-level design:**
  - Describes the system in terms of a service and a collection of features
- **Lower-level design**
  - Describe how each feature augments the service and identifies interactions among features

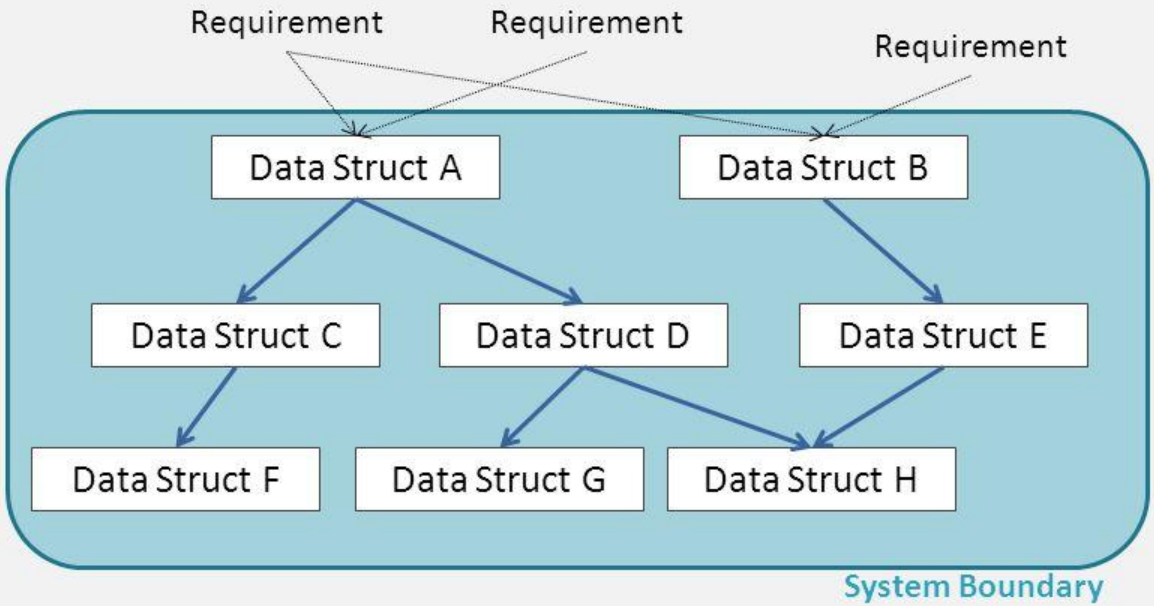

Feature
Volume feature
Transition feature
Pattern feature
Subtractive volume feature
Additive volume feature
Corner transition
Edge transition
Circular Feature pattern
Array Feature pattern
Other Feature pattern
Depression
Passage
Void
Protrusion
Connector
Standalone volume
Flat corner transition
Spherical corner transition
Flat edge transition
Circular edge transition
1 dimensional array pattern
2 dimensional array pattern
3 dimensional array pattern

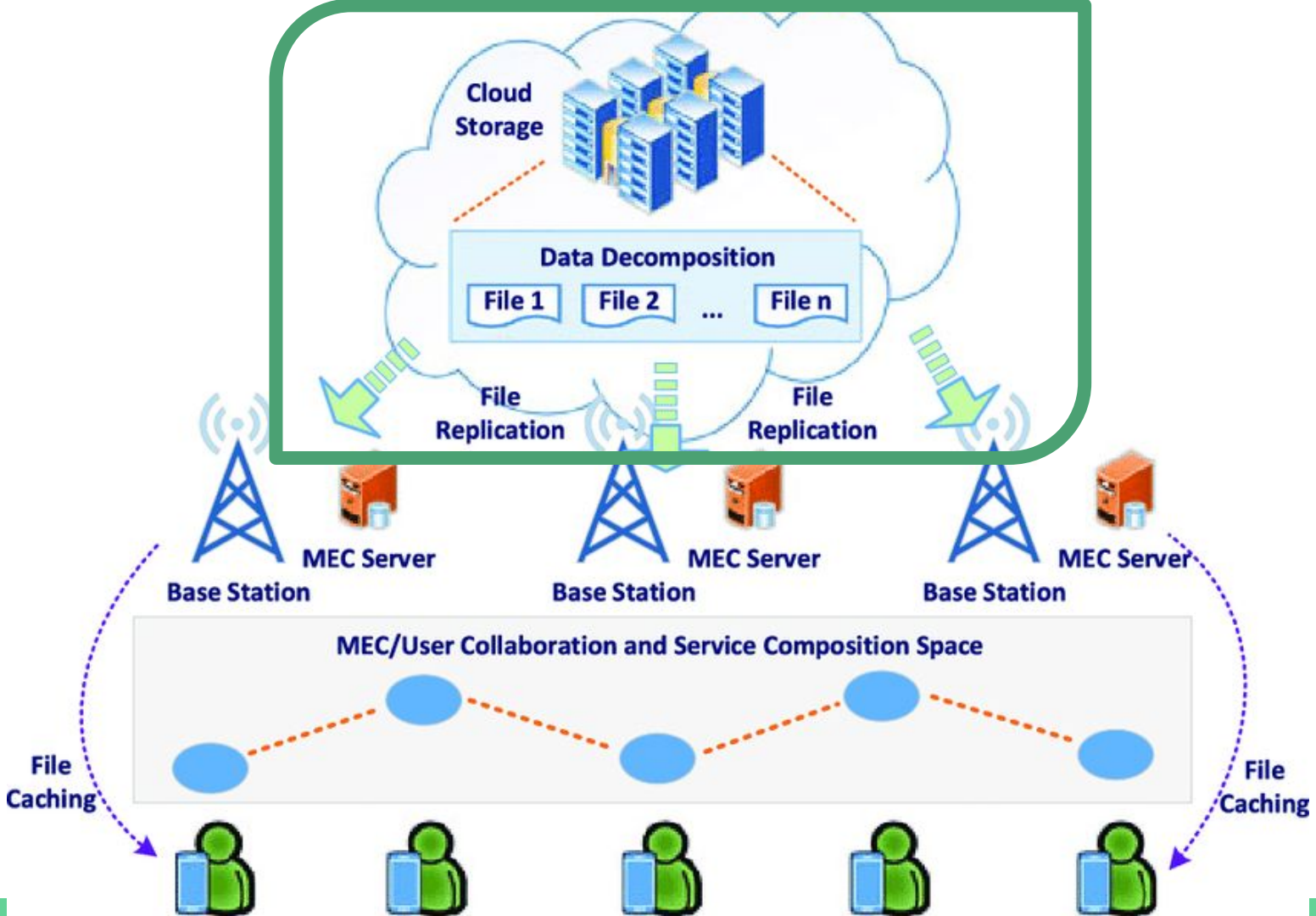
Types of decomposition

Data-Oriented decomposition:

- Focuses on how data will be partitioned into modules
- High-level design
  - Describes conceptual data structures
- Lower-level design
  - Provide detail as to how
    - Data are distributed among modules
    - Distributed data realize the conceptual models

Data-oriented decomposition

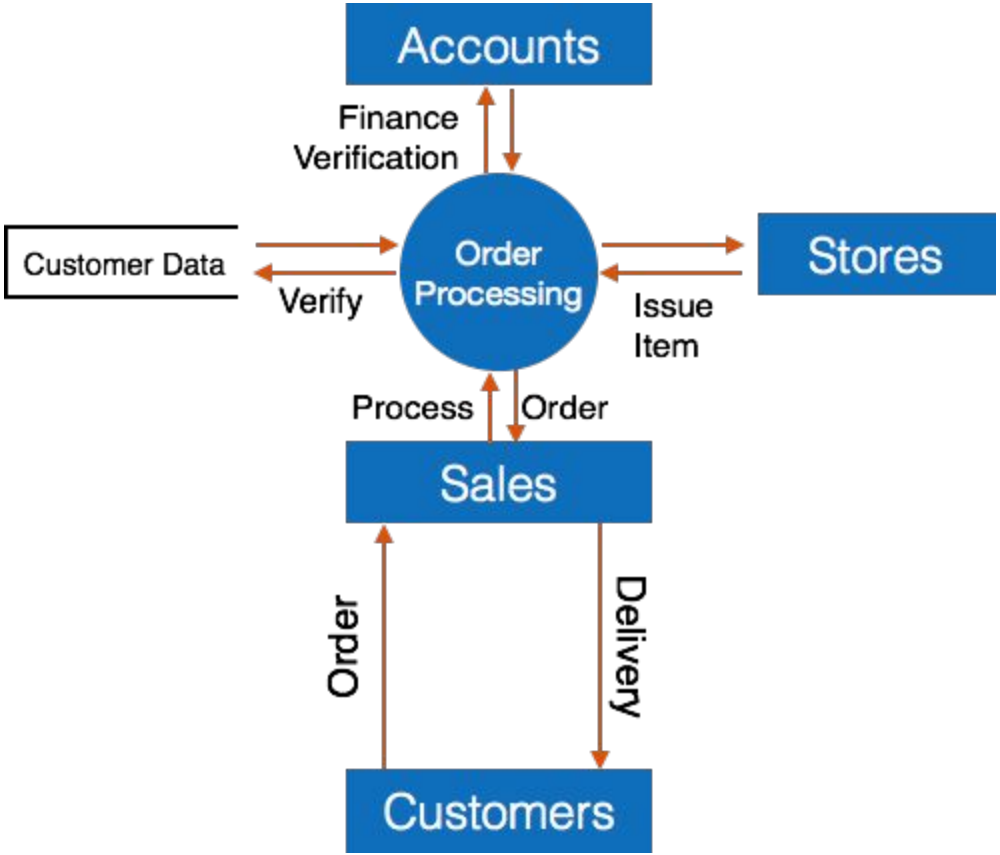



Cloud Storage
Data Decomposition
File 1
File 2
...
File n
File Replication
File Replication
MEC Server
MEC Server
MEC Server
Base Station
Base Station
Base Station
MEC/User Collaboration and Service Composition Space
File Caching
File Caching

Types of decomposition

Process-oriented decomposition:

- Partitions the system into concurrent processes
- **High-level** design:
  - identifies the system's main tasks
  - assigns tasks to runtime processes
  - explains how the tasks coordinate with each other
- **Lower-level** designs describe the processes in more detail


Accounts
Finance
Verification
Customer Data
Verify
Order
Processing
Issue
Item
Stores
Process
Order
Sales
Order
Delivery
Customers

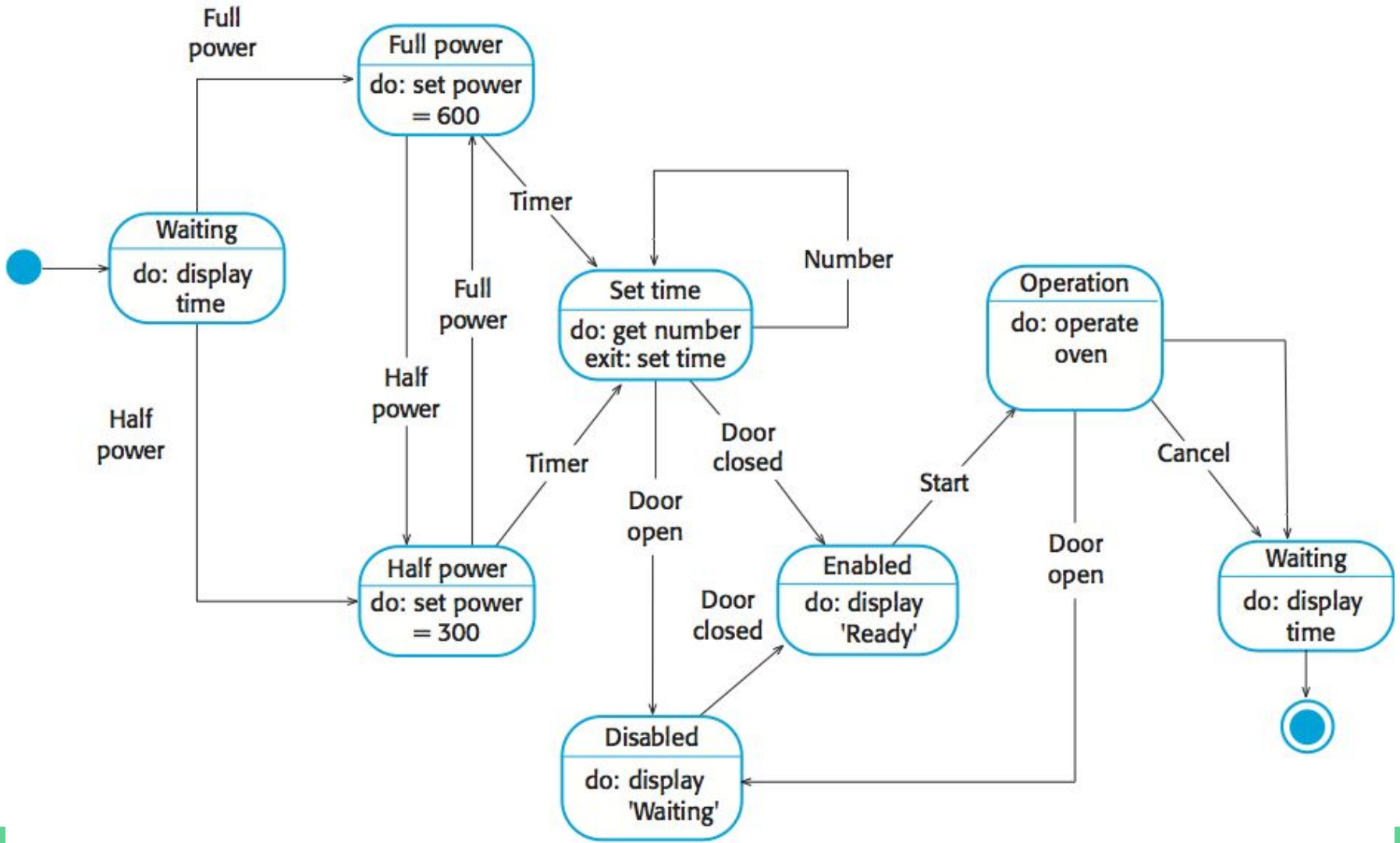
Types of decomposition

Event-oriented decomposition:

- Focuses on the events that the system must handle and assigns responsibility for events to different modules

- High-level design
  - Catalogues the system’s expected input events

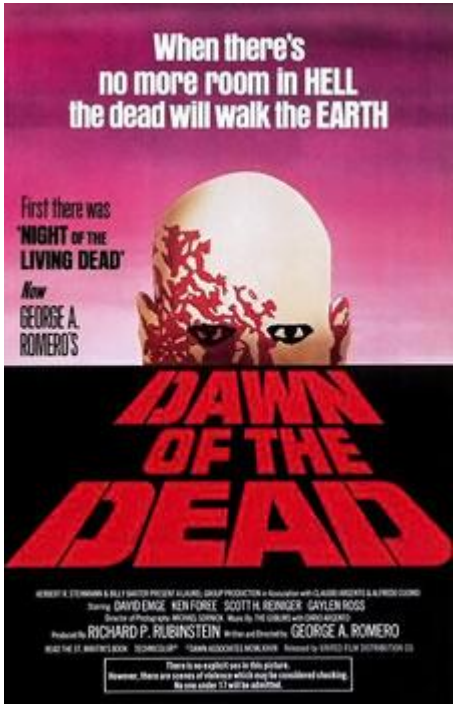
- Lower-level design
  - Decomposes the system into states and describe how events trigger state transformations


Full power
Full power
do: set power = 600
Waiting
do: display time
Timer
Number
Set time
do: get number
exit: set time
Full power
Half power
Half power
Operation
do: operate oven
Timer
Door closed
Door open
Start
Cancel
Half power
do: set power = 300
Enabled
do: display 'Ready'
Door closed
Door open
Waiting
do: display time
Disabled
do: display 'Waiting'

Building an Expression: What kind of Decomposition?

```
ExpressionInterface expression =
    new Multiplication(new Literal(-1.0),
        new Addition(
            new Subtraction(
                new Literal(4.0),
                new Literal(2.0)),
            new Literal(6.0)));
```

Decomposition example in media



Types of decomposition

Why do we typically use object-oriented decomposition?

Could we use other types of decomposition within an object-oriented system?

How could you apply decomposition to your term project?

If time allows

Get ready for **test-driven development**

In your **project teams**, select **two** functions that are *non-trivial* (meaning they do more than simply printing something out) and:

1) Identify the **method signature** (return type, function name, parameters)
2) Identify all acceptable **input values** for that function
3) Identify all acceptable **output values** for that function