Lecture 9: Matlab APP

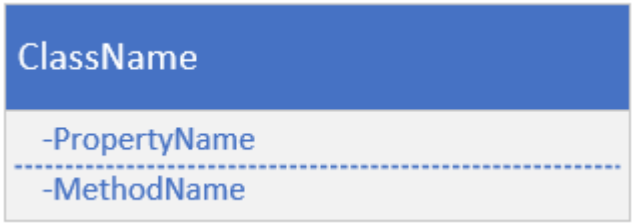
Announcement

- First customer consultation
  - Mon Mar 21st and Wed Mar 23rd
  - Each team has 15min (5min/project)
  - Online via Tencent Meeting (5min for each team)
  - "Executive Summary" due on Fri Mar 19th (<=3 pg of ppt)

Date	Teams
Mon Mar 21st (1pm-2:40pm)	Team 1-5
Wed Mar 23rd (1pm-2:40pm)	Team 6-11

Class definition in Matlab

```
classdef (ClassAttributes) ClassName < SuperClass1 & SuperClass2
    properties (PropertyAttributes)
        ...
    end
    methods (MethodAttributes)
        ...
    end
    events (EventAttributes)
        EventName
    end
end
```



Class Attributes

- Abstract
  - If specified as true, this class is an abstract class (cannot be instantiated).
  - classdef (Abstract = true) ClassName

- Sealed
  - If true, this class cannot be subclassed.

Value Class vs. Handle Class

- Value Class
- Each assignment creates a new copy of the object

```
classdef NumValue
  properties
    Number = 1
  end
end
```

- a = NumValue;
- b=a;
- a.Number = 7;
- b.Number
  - ans=1

- Handle Class
- Upon construction a reference to the object is created

```
classdef NumHandle < handle
  properties
    Number = 1
  end
end
```

- a = NumHandle;
- b=a;
- a.Number = 7;
- b.Number
  - ans=7

Value Class vs. Handle Class (cont.)

- When object passed into a function
  - Value object: a new copy of the object is created inside function workspace
  - Handle object: a copy of the handle (reference) is created instead of the object

- Deleting a handle object
  - Delete(NumHandle)

Object equality

Value object

- Can only evaluate whether value of the objects are the same

- a = NumValue;
- b = NumValue;
- isequal(a,b)
- ans=1

Handle object

- Can check whether they are the same object as well as their value equality

- a = NumHandle;
- b = a;
- a == b (same object?)
  - ans=1;
- isequal(a,b) (same value?)
  - ans=1;

a = NumHandle;

b = NumHandle;

a == b

ans=0;

isequal(a,b)

ans=1;

Class Members Access

- public — Unrestricted access
- protected — Access from methods in class or subclasses
- private — Access by class methods only (not from subclasses)
- List classes (and their subclasses) have access to this member
  - (Access = {?ClassName1,?ClassName2,...})

Property Attributes

- Read and write access
  - GetAccess
  - SetAccess
    - properties(GetAccess = 'public', SetAccess = 'private')
    - % public read access, but private write access.
    - end
    - SetAccess = immutable: set during construction, cannot be changed afterwards
- Constant

```
properties(Constant = true)
    DAYS_PER_YEAR =  365;
end
```

- Dependent
  - depend on other values
  - calculated only when needed.
  - i.e. area of a square depends on the width property

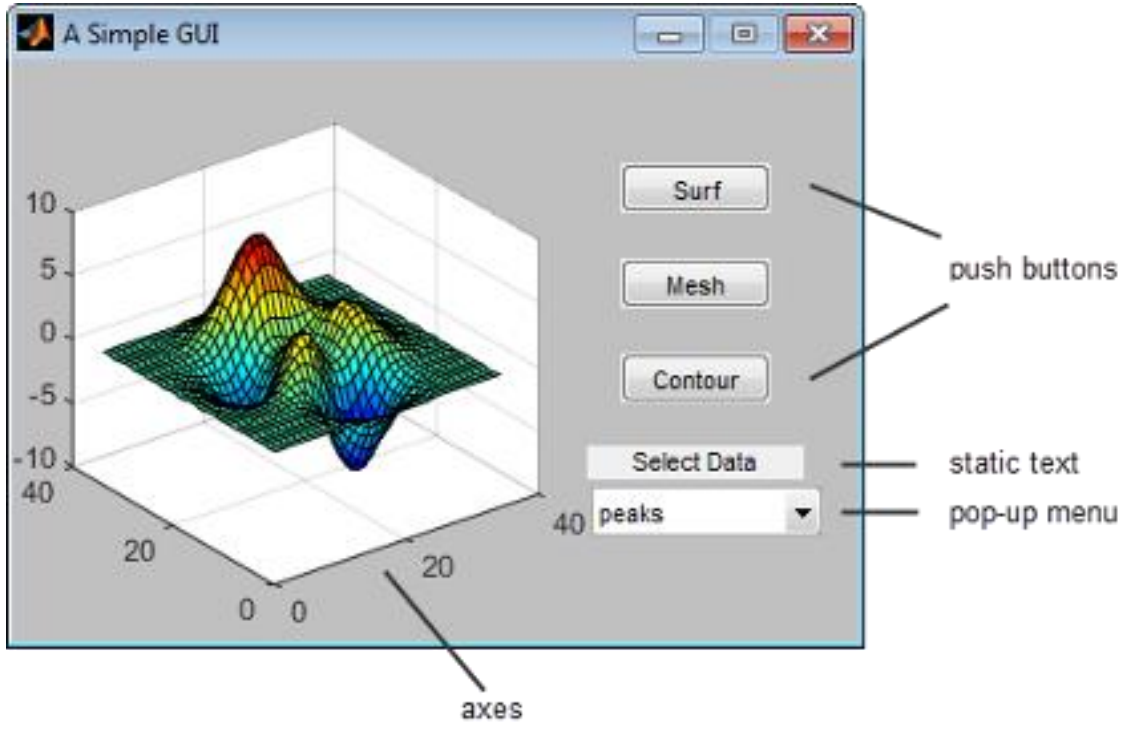
Class Constructor Method

- There is a default class constructor without input arguments
- We can define class constructor that overrides the default one
- Method with the same name as the class name

```
classdef ConstructorDesign < BaseClass1
methods
    function obj = ConstructorDesign(a,b,c)
    end
end
```

Graphic User Interface

- Consists of handle objects
  - figure
  - axes
  - uitable
  - uicontrol
- Each of them have unique properties and methods
- You can view and change these by calling
  - inspect(h)



figure

- h=figure(prop1,’prop1value’,…);
- Make one of the figures active
  - figure(h1)
- Get the handle of the current figure
  - h=gcf;

figure properties

- Units
  - ‘pixels’
  - ‘normalized’: from (0,0) lower left to (1,1) upper right
- Position
  - [left bottom width height]
- Name
- Parent
  - root is the top level
- Children
  - n*1 graphic handle
  - Ordered according to level first, and then stacking order

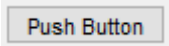
Figure callback events

- ButtonDownFcn
  - When clicking the mouse button while the pointer is located over or near the object.
- KeypressFcn
  - When a key is pressed
- CreatFcn
  - When creating an object
- DeleteFcn
  - When deleting an object.

Callback functions

- h=figure(‘ButtonDownFcn’,@testButtonDown)
- function testButtonDown(src,event)
  - src: the UI component that triggered the callback
  - event: event data. i.e. key pressed
- Provide additional input arguments to the callback function
  - h=figure(‘ButtonDownFcn’,{@testButtonDown,arg1,arg2…})
  - function testButtonDown(src,event, arg1,arg2…)

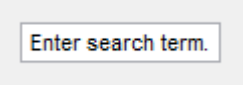
uicontrol

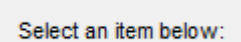
- ctrlhdl=uicontrol(fighdl,’style’,ctrlstyle,’prop1’,prop1value…)
- pushbutton 

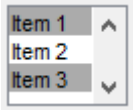
- togglebutton 

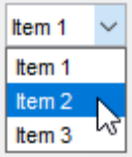
- checkbox 

- radiobutton 

- edit 

- text 

- slider 
- listbox 

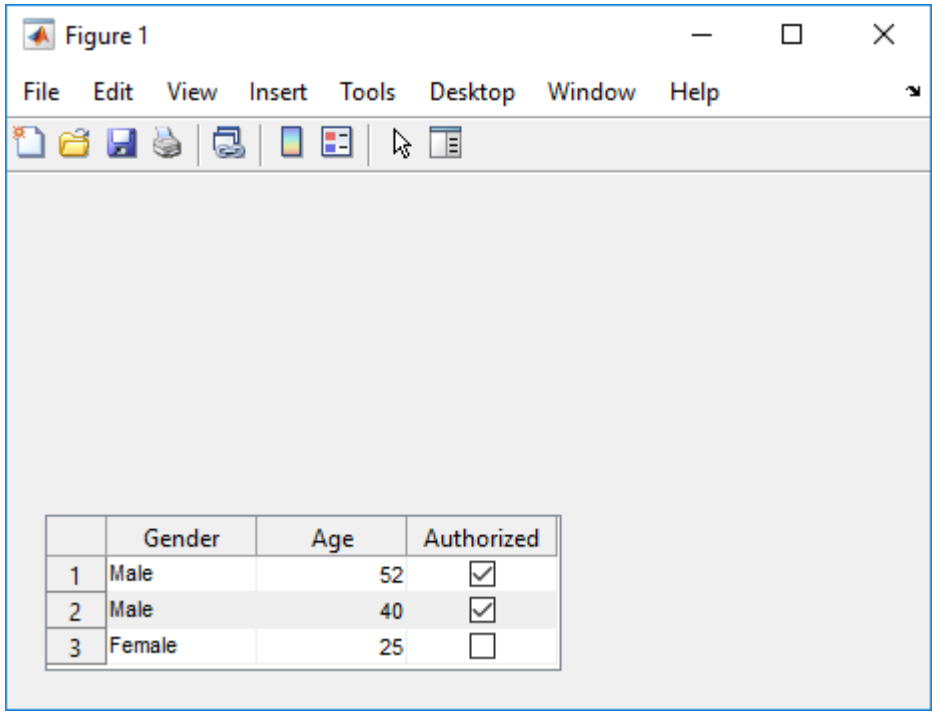
- popupmenu 

Common uicontrol properties

- Value
  - Checked/unchecked, slider position, listbox active index, etc
- String
  - Displayed string
  - Cell array of strings for listbox and popupmenu
  - i.e. {‘item1’;’item2’;’item3’}

uitable

- h=uitable(parent,’prop1’,prop1value…)
- data
  - A cell matrix
  - {'Male',52,true;'Male',40,true;'Female',25,false};
- ColumnName
  - 1*n Cell array
  - {'Gender','Age','Authorized’};

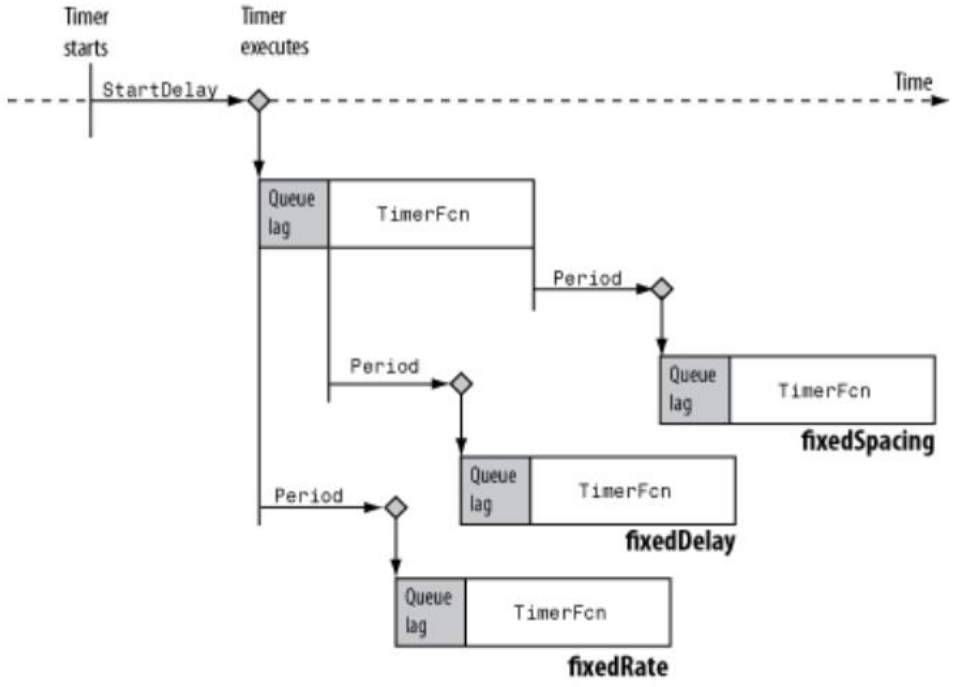


uitable callback events

- CellSelectionCallback
  - CellSelectionChangeData as input argument
    - Indices: row and column indices of the cell the user edited
- CellEditCallback
  - CellEditData as input argument
    - Indices:
    - PreviousData
    - NewData

Timer Class

- t = timer;
- Properties
  - ‘ExecutionMode’
  - ‘Period’: Time between timer functions
  - ‘TimerFcn’: Function handle
- t.TimerFcn=@callback;
- Function callback(hObj,src,event)



Demo: Traffic Light

Example: Information system for restaurants

- The owner of restaurant A would like to improve service efficiency

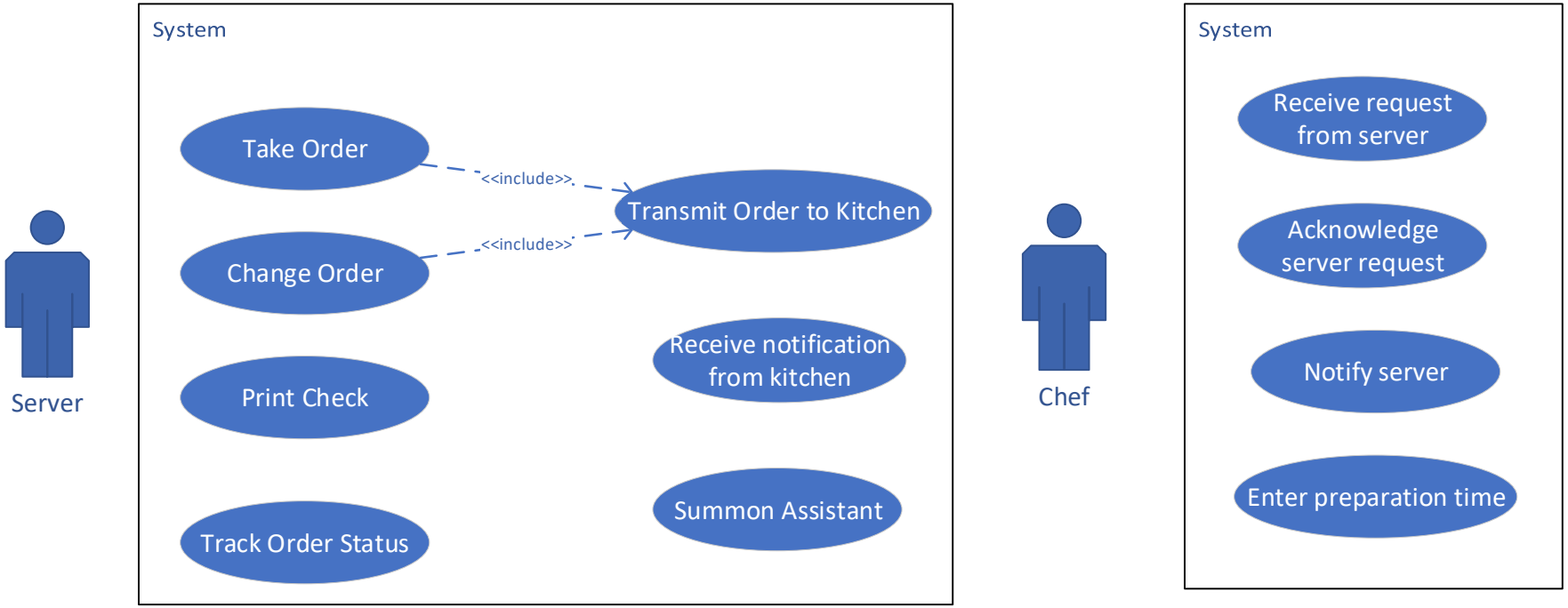
Customer

Server

Chef

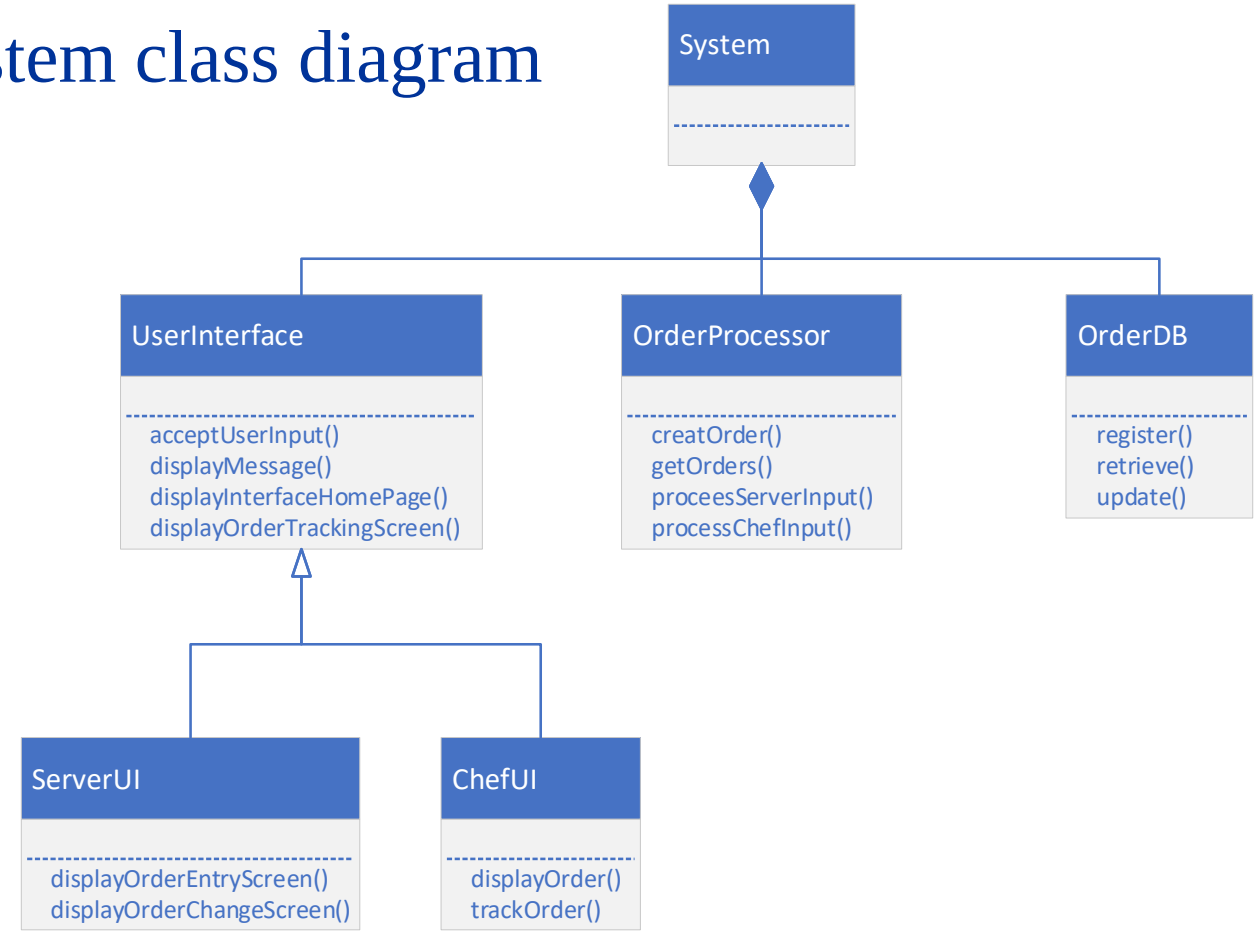
Discover system requirements (cont.)

- System requirements as use cases



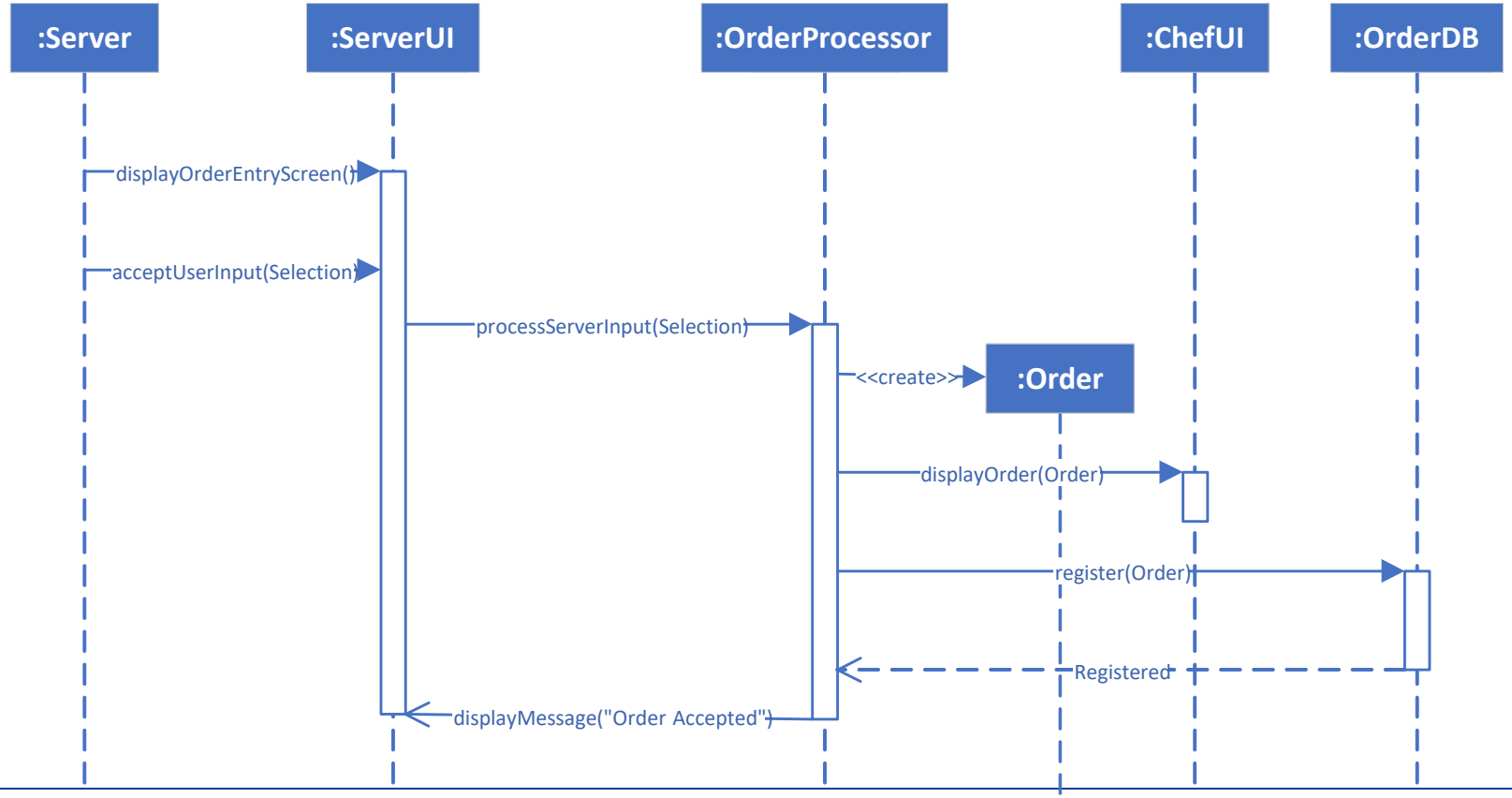
Discover system requirements (cont.)

- Enriched system class diagram



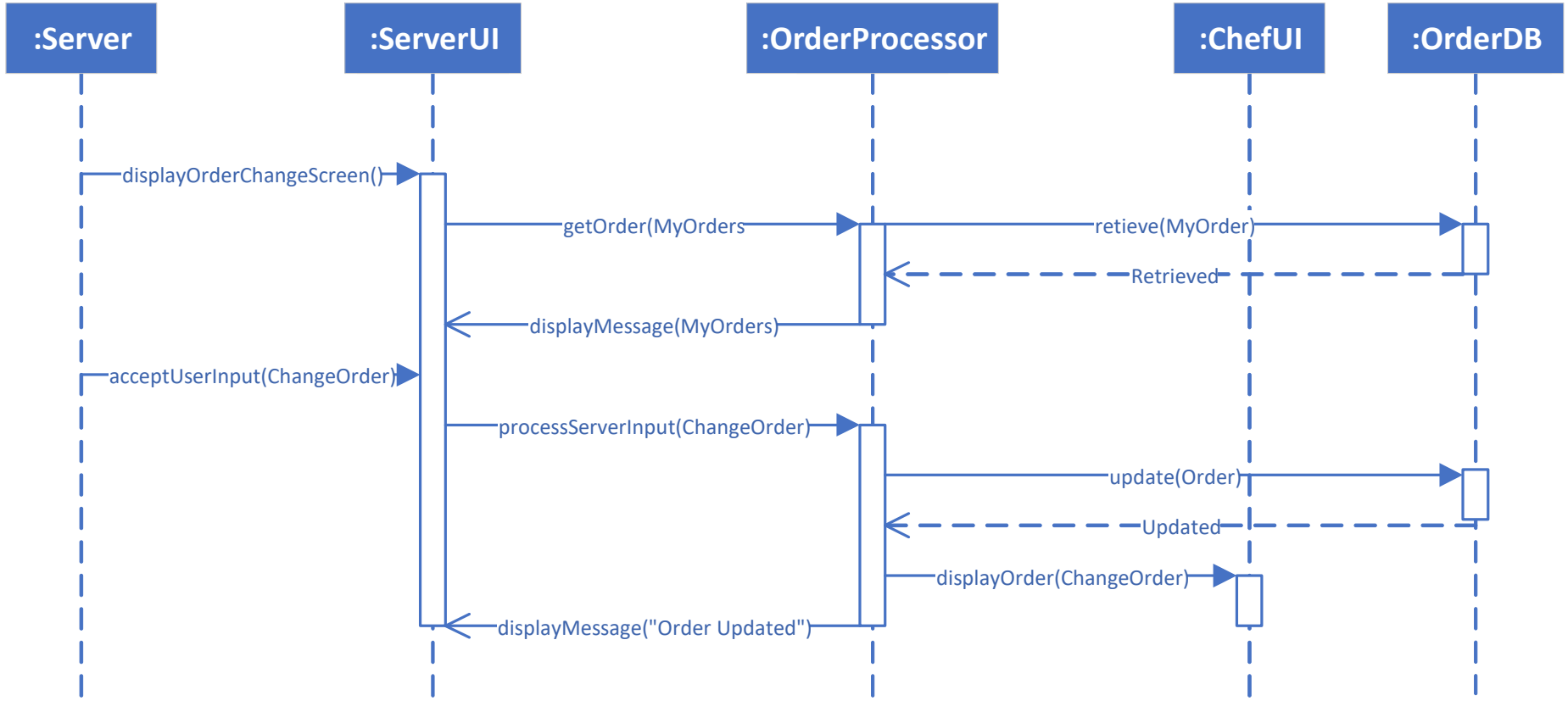
Identify interactions

- Use case “Take an order”



Identify interactions (cont.)

- Use case "Change an order"



Identify interactions (cont.)

- Use case "Track an order"