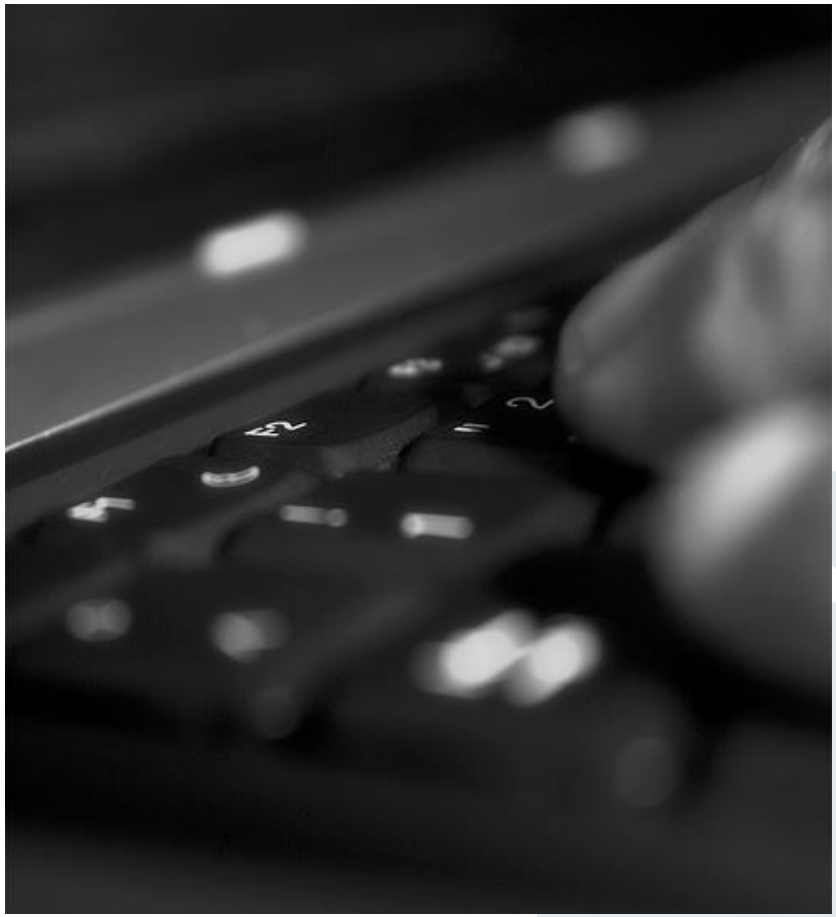
Mentorium 5

Business Informatics 2 (PWIN)

Markup Languages & Unified Modeling Language

Christopher Schmitz, M.Sc.
www.m-chair.de



Jenser (Flickr.com)

Agenda

- XML and DTD
- Unified Modeling Languages

Repetition: XML
Manage Dates via XML

```
<?xml version=„1.0“ encoding=„ISO-8859-1“ ?>
```

Prologue

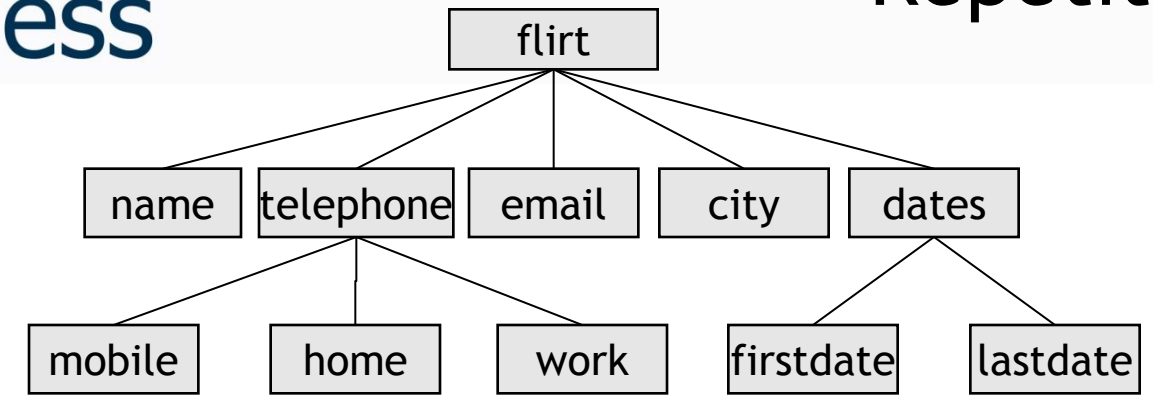
```
<flirt>
    <name>Daisy</name>
    <mobile>+436508469249</mobile>
    <email>daisy@m-chair.de</email>
    <city>Innsbruck</city>
    <first date>2019-01-23</first date>
    <last date>2019-05-01</last date>
    <birthday>1993-11-13</birthday>
    <vegetarian>no</vegetarian>
    <status>single</status>
</flirt>
```

Body

Repetition: DTD

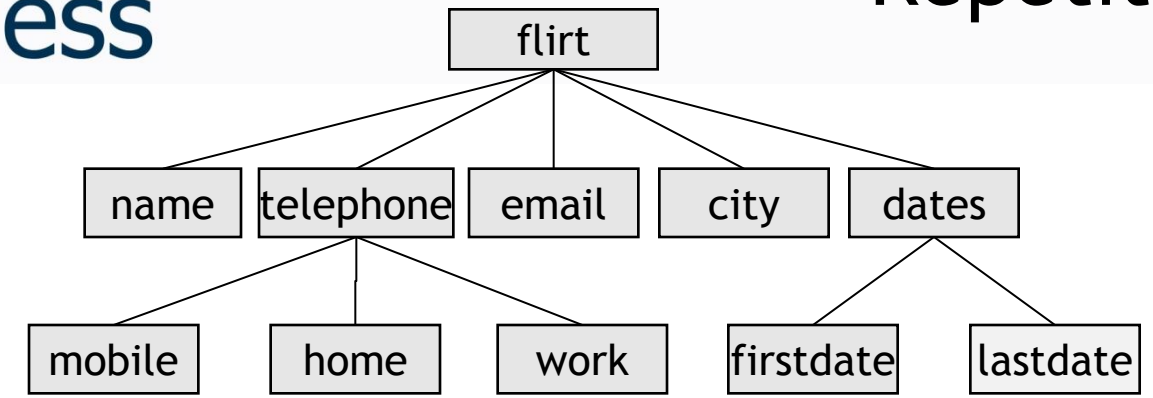
- The Document Type Definition (DTD) describes the structure of a document and defines a *grammar* for the XML document.
- Comparable to a type or variable declaration in a programming language.
- The DTD defines which elements and references may appear in the document based on it.
- The DTD also declares entities that are allowed to be used in the XML document.

Repetition: DTD




```
<flirt>
        <name>Daisy</name>
        <telephone>
                <mobile> 0177 / 1234567 </mobile>
        </telephone>
        <email> daisy@m-chair.de </email>
        <city>  Innsbruck </city>
        <dates>
                <firstdate>2019-01-01 </firstdate>
                <lastdate> 2019-01-10 </lastdate>
        </dates>
</flirt>
```

Repetition: DTD



Rule declaration for the elements in a DTD:

```
<!DOCTYPE flirt [
<!ELEMENT flirt (name, telephone, email, city, dates)>
<!ELEMENT name      (#PCDATA)>
<!ELEMENT telephone (mobile | home| work)>
<!ELEMENT mobile    (#PCDATA)>
<!ELEMENT home      (#PCDATA)>
<!ELEMENT work      (#PCDATA)>
<!ELEMENT email     (#PCDATA)>
<!ELEMENT city      (#PCDATA)>
<!ELEMENT dates     (firstdate, lastdate)>
<!ELEMENT firstdate(#PCDATA)>
<!ELEMENT lastdate  (#PCDATA)>
]>
```

Repetition: DTD – Grouping and Cardinality

Repetition: DTD – Cardinality

- Cardinalities (for elements):

empty	exactly one value is necessary
+	At least one value
?	None or one value
*	None or multiple values

- Content (in elements):

EMPTY	Empty element
ANY	Any content
|	Selection list
,	Sequence
()	Grouping
(#PCDATA)	*Parsed Character Data* (mixed data)

Repetition: DTD – Cardinality

```
<!DOCTYPE flirt [
<!ELEMENT flirt (name, telephone,
  email, city, dates)>
<!ELEMENT name      (#PCDATA)>
<!ELEMENT telephone (mobile | home |
  work)>
<!ELEMENT mobile    (#PCDATA)>
<!ELEMENT home      (#PCDATA)>
<!ELEMENT work      (#PCDATA)>
<!ELEMENT email     (#PCDATA)>
<!ELEMENT city      (#PCDATA)>
<!ELEMENT dates     (firstdate,
  lastdate)>
<!ELEMENT firstdate(#PCDATA)>
<!ELEMENT lastdate  (#PCDATA)>
]>
```

Exemplary XML document

Select one element
→ Exactly three solutions:

```
<telephone>
   <mobile>…</mobile>
</telephone>
```

or

```
<telephone>
   <home>…</home>
</telephone>
```

or

```
<telephone>
   <work>…</work>
</telephone>
```

Repetition: DTD – Cardinality

```
<!DOCTYPE flirt [
<!ELEMENT flirt (name, telephone,
  email, city, dates)>
<!ELEMENT name      (#PCDATA)>
<!ELEMENT telephone (mobile | home |
  work)+>
<!ELEMENT mobile    (#PCDATA)>
<!ELEMENT home      (#PCDATA)>
<!ELEMENT work      (#PCDATA)>
<!ELEMENT email     (#PCDATA)>
<!ELEMENT city      (#PCDATA)>
<!ELEMENT dates     (firstdate,
  lastdate)>
<!ELEMENT firstdate(#PCDATA)>
<!ELEMENT lastdate  (#PCDATA)>
]>
```

Exemplary XML document

At least one (+) repetition of the selection rule:

```
<telephone>
   <mobile>…</mobile>
   <mobile>…</mobile>
   <home>…</home>
</telephone>
```

or

```
<telephone>
   <work>…</work>
</telephone>
```

or

```
...
```

Repetition: DTD – Cardinality

```
<!DOCTYPE flirt [
<!ELEMENT flirt (name, telephone*,
  email, city, dates)>
<!ELEMENT name      (#PCDATA)>
<!ELEMENT telephone (mobile | home |
  work)>
<!ELEMENT mobile    (#PCDATA)>
<!ELEMENT home      (#PCDATA)>
<!ELEMENT work      (#PCDATA)>
<!ELEMENT email     (#PCDATA)>
<!ELEMENT city      (#PCDATA)>
<!ELEMENT dates     (firstdate,
  lastdate)>
<!ELEMENT firstdate(#PCDATA)>
<!ELEMENT lastdate  (#PCDATA)>
]>
```

Exemplary XML document

```
<telephone>
   <mobile>…</mobile>
</telephone>
<telephone>
   <home>…</home>
</telephone>
<telephone>
   <work>…</work>
</telephone>
...
```

XML→DTD

1.1 Create a DTD based on the following XML document. Note that the <Location> element can appear 1 or more times.

XML→DTD

```
<?xml version="1.0"?>
<Locations>
   <Location>
      <Name>DASEIN</Name>
      <Rating>Good</Rating>
   </Location>
   <Location>
      <Name>Sturm und
               Drang</Name>
      <Rating>Good</Rating>
   </Location>
</Locations>
```

```
<!DOCTYPE Locations [
<!ELEMENT Locations (Location+)>
<!ELEMENT Location (Name,
Rating)>
<!ELEMENT Name (#PCDATA)>
<!ELEMENT Rating (#PCDATA)>
]>
```

DTD → XML

1.2 Create an exemplary XML document based on the following DTD file. The <records> element must not be empty.

DTD → XML

```
<?xml version="1.0"?>
<records>
   <record>
      <title>Help!</title>
      <artist>The Beatles</artist>
      <price>9.95</price>
   </record>
   <record>
      <title>The very best
      of</title>
      <artist>Lang Lang</artist>
      <composer>Bach</composer>
      <price>19.90</price>
   </record>
</records>
```

```
<!DOCTYPE records [
<!ELEMENT records (record*)>
<!ELEMENT record (title,
artist, composer?, price)>
<!ELEMENT title (#PCDATA)>
<!ELEMENT artist (#PCDATA)>
<!ELEMENT composer (#PCDATA)>
<!ELEMENT price (#PCDATA)>
]>
```

XML → DTD

1.3 Create a DTD based on the following XML document. Note that the <email> element is mandatory and <student> should appear at least 1 time.

XML → DTD

```
<?xml version="1.0"?>
<lecture>
   <student>
      <name>Max Muster</name>
      <subject>WiWi</subject>
      <semester>4</semester>
      <email>max@abc.de</email>
      <email>max@xyz.de</email>
      <fax>123456789</fax>
   </student>
   <student>
      <name>Erika Muster</name>
      <subject>WiWI</subject>
      <semester>4</semester>
      <email>em@mail.de</email>
   </student>
</lecture>
```

```
<!DOCTYPE lecture [
<!ELEMENT lecture (student+)>
<!ELEMENT student (name,
subject, semester, email+,
fax?)>
<!ELEMENT name (#PCDATA)>
<!ELEMENT subject (#PCDATA)>
<!ELEMENT semester (#PCDATA)>
<!ELEMENT email(#PCDATA)>
<!ELEMENT fax(#PCDATA)>
]>
```

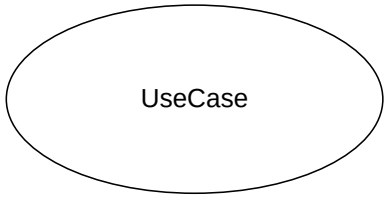
Agenda

- XML and DTD
- Unified Modeling Languages

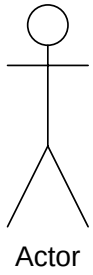
UML – Use Case Diagram

Repetition: Use Case Diagram Notation Elements (1/3)

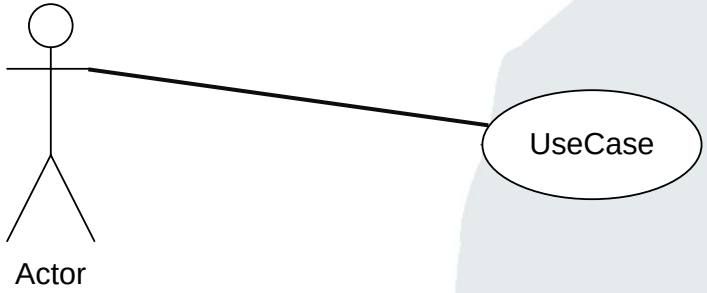
- **Use Case**
  - Representation of a sequence of actions that provides value to an actor.



- **User of the system**

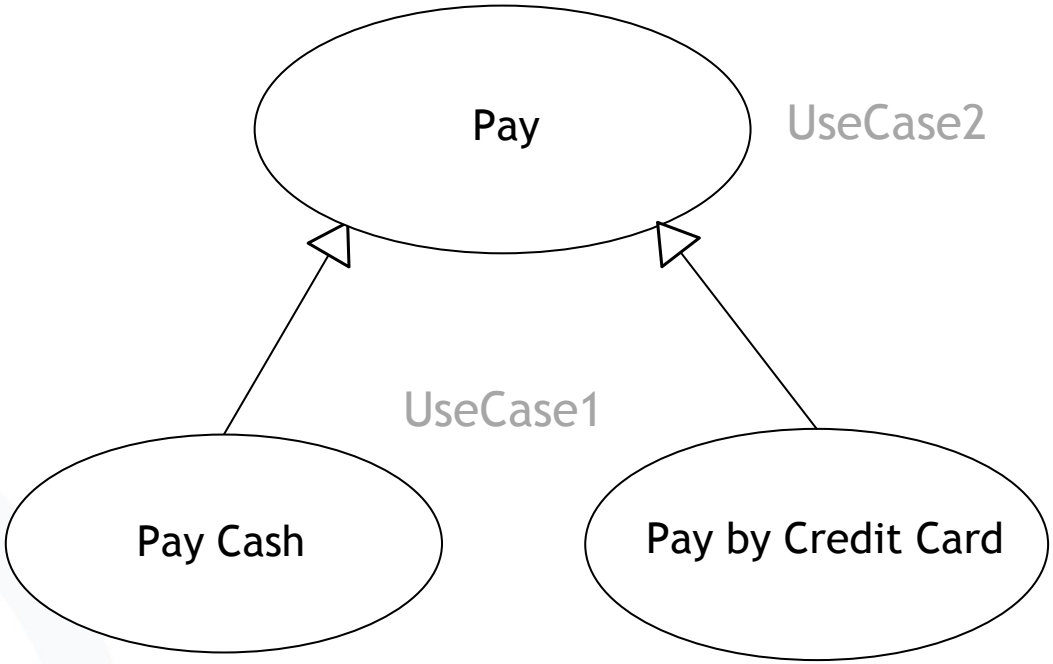


- **Association**
  - Interaction of an actor with a use case



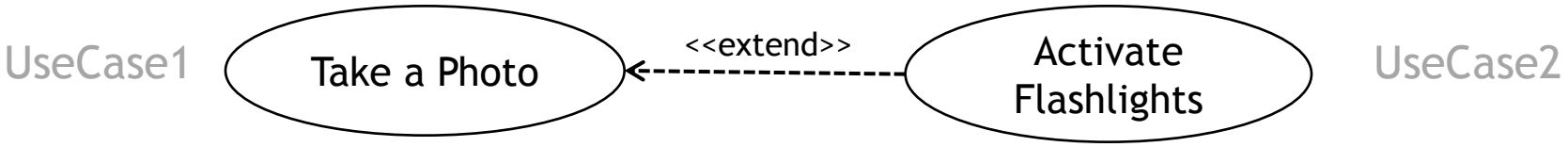
Repetition: Use Case Diagram Notation Elements (2/3)

- **Generalisation**
  - Generalisation of use cases
  - UseCase2 generalises the behaviour of UseCase1

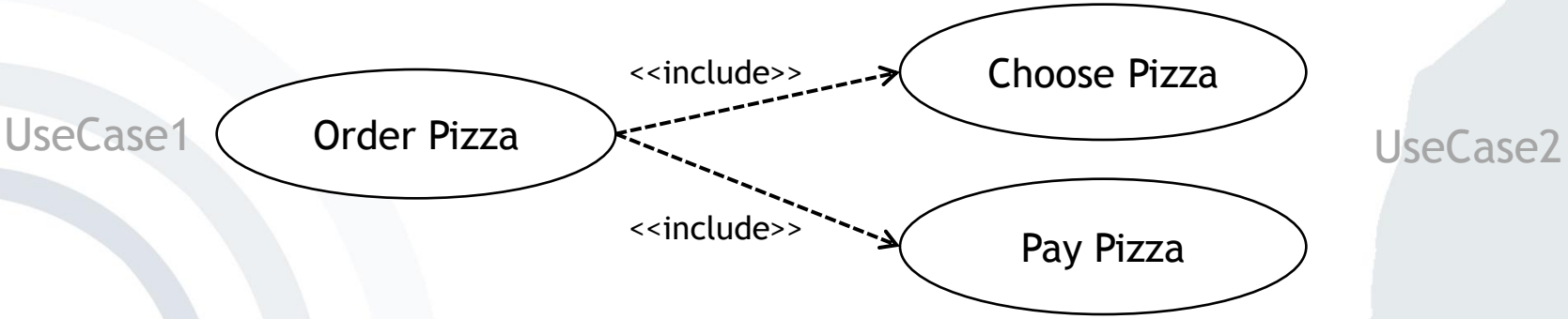


Repetition: Use Case Diagram Notation Elements (3/3)

- **<<Extend>>**
  - Extends a use case
  - UseCase2 extends UseCase1



- **<<Include>>**
  - Inclusion of a use case
  - UseCase1 includes the behaviour of UseCase2

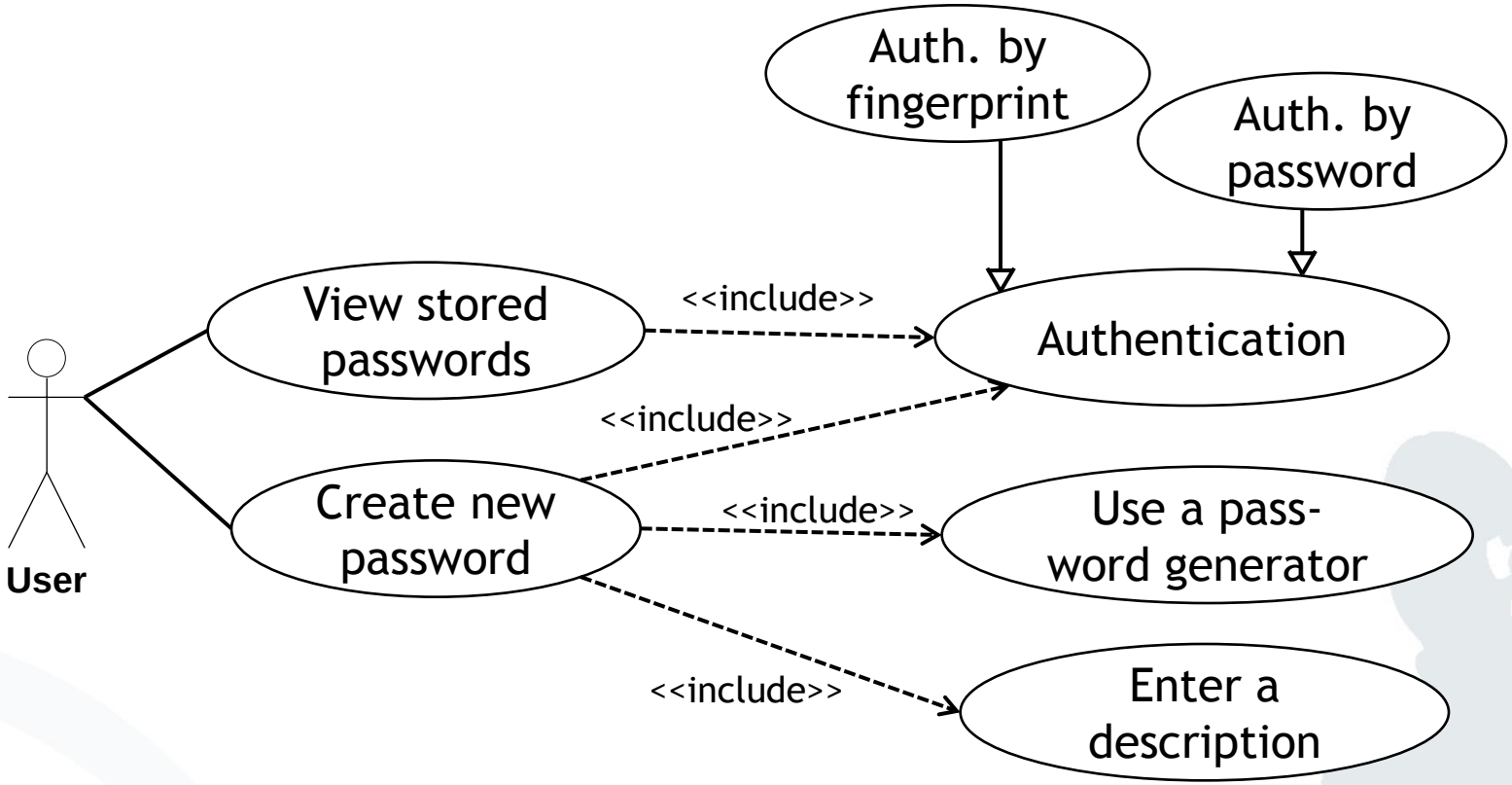


Use Case Diagram

3.1 Create a use case diagram for a password manager app:

A user can either view stored passwords or create a new password. Both use cases require an authentication. The app supports both user authentication by password and authentication by fingerprint. To create a new password, the user has to use an integrated password generator. Furthermore, he has to enter a description for his new password.

Use Case Diagram

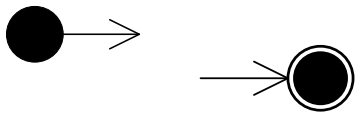


UML – Activity Diagram

Repetition: Activity Diagram Notation Elements

Notation elements

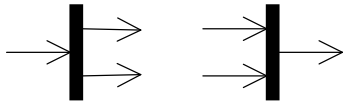
- Initial state/final state



- Activity

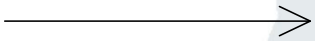
- Decision

- Split/join



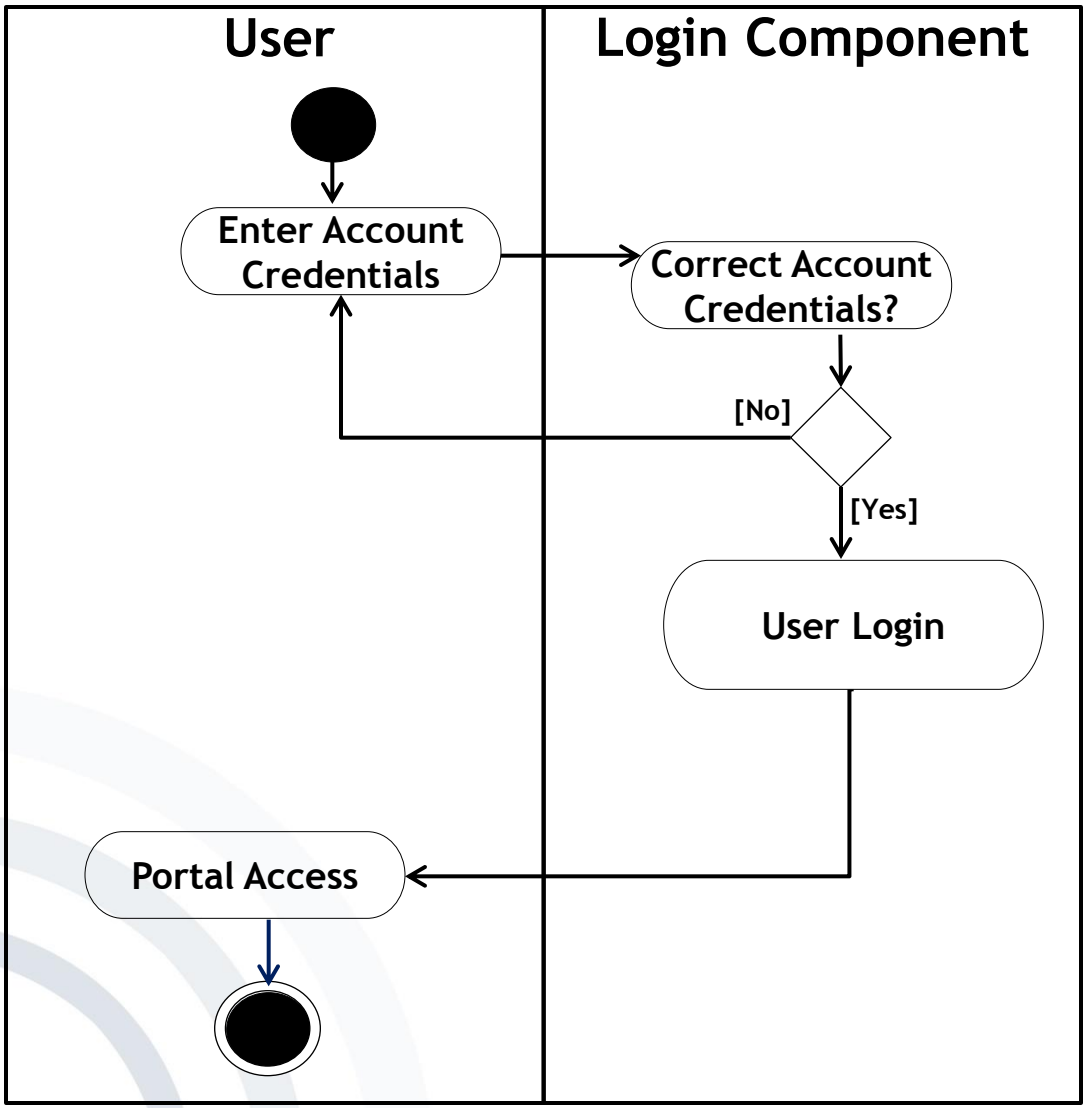
- Responsibility

- Activity flow



3.2 a) You are one of the programmers of the MyPlace Service. Assume your manager wants you to model the login process by using the appropriate UML diagram.

Activity Diagram



3.2 b) Additionally, your manager plans to integrate the Facebook login functionality into the myPlace Service. Model the corresponding extended login process by using the appropriate UML diagram.

Activity Diagram