This diagram is the fourth iteration of the ZooKeeper narrative intended to model the use of classes and behaviors to model the definition of model behavior based on Use Case.

(C) Les Waguespack, Ph.D. 2005

**Class Diagrams**

- class Animal
- class Cage
- class Dietary_Item
- class Feeder
- class Food
- class Serving
- class Serving_List
- class Staff_Member
- class Veterinary_Nutritionist
- class ZooArea
This diagram is the fourth iteration of the ZooKeeper narrative intended to model the use of classes and behaviors to model the definition of model behavior based on Use Case.

(C) Les Waguespack, Ph.D. 2005

Class Nodes
Animal
Class Detail

Class Animal

public class Animal

This is an animal housed in the zoo.

Field Summary

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>aGender</td>
<td>int</td>
<td>The gender of this animal.</td>
</tr>
<tr>
<td>aName</td>
<td>int</td>
<td>The given name of the animal.</td>
</tr>
<tr>
<td>aSpecies</td>
<td>int</td>
<td>The biological species of this animal.</td>
</tr>
<tr>
<td>lnkRation</td>
<td>Dietary_Item</td>
<td>This is a collection of one or more serving lists for a particular animal.</td>
</tr>
<tr>
<td>lnkServingList</td>
<td>Serving_List</td>
<td>This is a collection of one or more serving lists for a particular animal.</td>
</tr>
</tbody>
</table>

Method Summary

<table>
<thead>
<tr>
<th>Method</th>
<th>Signature</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>createServingList</td>
<td>public Serving_List createServingList(Animal theanimal, int howmuch, int howoften)</td>
<td>This service creates a list of food servings based upon the specific dietary items designated for this animal.</td>
</tr>
<tr>
<td>whichIsYourCage</td>
<td>public Cage whichIsYourCage()</td>
<td>This service returns a link to the cage in which this animal is domiciled.</td>
</tr>
</tbody>
</table>

Field Detail

aGender

private int aGender

The gender of this animal.

aName

private int aName

The given name of the animal.

aSpecies

private int aSpecies

The biological species of this animal.

lnkRation

private Dietary_Item lnkRation
InkServingList

private Serving_List InkServingList

This is a collection of one or more serving lists for a particular animal.

**Method Detail**

**createServingList**

public Serving_List createServingList(Animal theanimal, int howmuch, int howoften)

This service creates a list of food servings based upon the specific dietary items designated for this animal.

**whichIsYourCage**

public Cage whichIsYourCage()

This service returns a link to the cage in which this animal is domiciled.

**Class Cage**

public class Cage

This is an enclosure that houses an animal.

**Field Summary**

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cLoc</td>
<td>int</td>
<td>The location of the cage.</td>
</tr>
<tr>
<td>cSize</td>
<td>int</td>
<td>Cage size: small, medium, large.</td>
</tr>
<tr>
<td>cType</td>
<td>int</td>
<td>Type of cage: moat, bars, unbarred.</td>
</tr>
<tr>
<td>linkAnimal</td>
<td>Animal</td>
<td>A cage may be empty.</td>
</tr>
</tbody>
</table>

**Method Summary**

<table>
<thead>
<tr>
<th>Method Name</th>
<th>Signature</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>assignAnimalToCage</td>
<td>void assignAnimalToCage(Animal theAnimal)</td>
<td>This service allows a zookeeper to assign a particular animal to a particular cage.</td>
</tr>
<tr>
<td>enumerateAnimals</td>
<td>Animal enumerateAnimals()</td>
<td>This service successively returns a link to each of the animals housed in it.</td>
</tr>
</tbody>
</table>

**Field Detail**

**cLoc**

private int cLoc

The location of the cage.

**cSize**

private int cSize

Cage size: small, medium, large.
cType
private int cType

Type of cage: moat, bars, unbarred.

lnkAnimal
private Animal lnkAnimal

A cage may be empty. Every animal must be in a cage.

### Method Detail

#### assignAnimalToCage

```java
public void assignAnimalToCage(Animal theAnimal)
```

This service allows a zookeeper to assign a particular animal to a particular cage.

#### enumerateAnimals

```java
public Animal enumerateAnimals()
```

This service successively returns a link to each of the animals housed in it.

### Class Dietary_Item

```java
public class Dietary_Item
```

This is a particular ration definition of food for a specific animal.

#### Field Summary

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>rHowMuch</td>
<td>int</td>
<td>How many units of the designated food are allotted to one ration for the designated animal.</td>
</tr>
<tr>
<td>rHowOften</td>
<td>int</td>
<td>The number of times during the feeding period that this animal is given this ration (e.g. twice a week, everyday, etc.).</td>
</tr>
</tbody>
</table>

#### Field Detail

**rHowMuch**

```java
private int rHowMuch
```

How many units of the designated food are allotted to one ration for the designated animal.

**rHowOften**

```java
private int rHowOften
```

The number of times during the feeding period that this animal is given this ration (e.g. twice a week, everyday, etc.).

### Class Feeder

```java
public class Feeder
```

Extends:
This is a specially trained staff member who is responsible for the care and feeding of the animals.

### Field Summary

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>private</td>
<td>Serving_List</td>
<td>InkServingList A serving list is the sole responsibility of a single feeder staff member.</td>
</tr>
</tbody>
</table>

### Method Summary

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>public void</td>
<td>feedTheAnimals() This service actually brings the servings to each cage to feed the animals.</td>
</tr>
<tr>
<td>public void prepareServingList(ZooArea theArea)</td>
<td>This service prepares a list of food servings derived from the dietary needs of each animal.</td>
</tr>
</tbody>
</table>

### Field Detail

InkServingList

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>private</td>
<td>Serving_List</td>
<td>InkServingList A serving list is the sole responsibility of a single feeder staff member.</td>
</tr>
</tbody>
</table>

A feeder may be in the process of delivering a serving list or have completed same, thus having no current feeding list to work with. The serving lists are ordered according to the adjacency of the cages in the area. This is accomplished by the order that the ZooArea returns each of the cages in its EnumerateCages service.

### Method Detail

**feedTheAnimals**

```java
public void feedTheAnimals()
{
    This service actually brings the servings to each cage to feed the animals.
}
```

**prepareServingList**

```java
public void prepareServingList(ZooArea theArea)
{
    This service prepares a list of food servings derived from the dietary needs of each animal.
}
```

### Class Food

```java
public class Food

This is a category of food which is stored in the zoo warehouse for the feeding of the animals.
```

### Field Summary

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>private</td>
<td>int</td>
<td>fDesc A description of the food type (i.e.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The number of units of this food found in the food storage.</td>
</tr>
<tr>
<td></td>
<td>int</td>
<td>fUnits The type of units with which this food is measured.</td>
</tr>
<tr>
<td>private</td>
<td>Dietary_Item</td>
<td>InkRation A collection of dietaryitem objects created from a specific food type.</td>
</tr>
</tbody>
</table>
**Method Summary**

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>public void createRation(Animal theanimal, int howmuch, int howoften)</td>
<td>This service creates a dietary item for a specific animal designating the amount and frequency of this ration for that animal.</td>
</tr>
<tr>
<td>public void createServing()</td>
<td>This service withdraws food from the food warehouse and prepares a single serving of same for its particular animal.</td>
</tr>
<tr>
<td>public void stockFood()</td>
<td>This service updates the current inventory of this food when supplies are placed in the warehouse.</td>
</tr>
<tr>
<td>public boolean withdrawFood()</td>
<td>This service notes the withdrawal of food of this type form the warehouse.</td>
</tr>
</tbody>
</table>

**Field Detail**

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fDesc</td>
<td>int</td>
<td>A description of the food type (i.e. Meat, Fish, Grain, etc.)</td>
</tr>
<tr>
<td>fInv</td>
<td>int</td>
<td>The number of units of this food found in the food storage.</td>
</tr>
<tr>
<td>fUnits</td>
<td>int</td>
<td>The type of units with which this food is measured.</td>
</tr>
<tr>
<td>InkRation</td>
<td>Dietary_Item</td>
<td>A collection of dietary item objects created from a specific food type.</td>
</tr>
</tbody>
</table>

**Method Detail**

**createRation**

public void createRation(Animal theanimal, int howmuch, int howoften)

This service creates a dietary item for a specific animal designating the amount and frequency of this ration for that animal.

**createServing**

public void createServing()

This service withdraws food from the food warehouse and prepares a single serving of same for its particular animal.

**stockFood**

public void stockFood()

This service updates the current inventory of this food when supplies are placed in the warehouse.

**withdrawFood**

public boolean withdrawFood()
This service notes the withdrawal of food of this type from the warehouse. If insufficient food is on hand the service fails.

**Class Serving**

```java
dietary_item

| public class Serving

Extends:

Dietary_Item

A specialization of DietaryItem indicating a physical instance of food to be given to an animal.

**Field Summary**

| private EasternStandardTime completionTime |
| Time the serving was actually delivered. |
| private EasterStandardTime scheduledTime |
| Time the serving is scheduled to be delivered. |

**Method Summary**

| public EasternStandardTime deliver(EasternStandardTime time) |

**Field Detail**

**completionTime**

private EasternStandardTime completionTime

Time the serving was actually delivered.

**scheduledTime**

private EasterStandardTime scheduledTime

Time the serving is scheduled to be delivered.

**Method Detail**

**deliver**

public EasternStandardTime deliver(EasternStandardTime time)

**Class Serving_List**

```java

public class Serving_List

This is a zoo staff member whose responsibility is to manage the feeding of the animals in the zoo.

**Field Summary**

| private CalendarDay date |
| The calendar date that this serving list is intended to be fed to the animal. |
| private Serving lnkRation |
| A collection of servings to be delivered to a particular animal. |
### Field Summary

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>private</td>
<td>String</td>
</tr>
<tr>
<td>servingName</td>
<td></td>
</tr>
<tr>
<td></td>
<td>A string indicating the name of the serving list.</td>
</tr>
</tbody>
</table>

### Field Detail

**date**

```java
private CalendarDay date
```

The calendar date that this serving list is intended to be fed to the animal.

**InkRation**

```java
private Serving InkRation
```

A collection of servings to be delivered to a particular animal.

**servingName**

```java
private String servingName
```

A string indicating the name of the serving list.

---

### Class Staff_Member

```java
public class Staff_Member
```

This is the general representation of a zoo staff member.

#### Field Summary

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>private</td>
<td>int</td>
</tr>
<tr>
<td>badgeNumber</td>
<td></td>
</tr>
<tr>
<td></td>
<td>A unique identifying code used to verify employee identity.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>private</td>
<td>int</td>
</tr>
<tr>
<td>employeeName</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Legal name of zoo staff member.</td>
</tr>
</tbody>
</table>

#### Field Detail

**badgeNumber**

```java
private int badgeNumber
```

A unique identifying code used to verify employee identity.

**employeeName**

```java
private int employeeName
```

Legal name of zoo staff member.

---

### Class Veterinary_Nutritionist

```java
public class Veterinary_Nutritionist
```

Extends:

```java
Staff_Member
```

```java
|-- Veterinary_Nutritionist
```

```java
public class Veterinary_Nutritionist
```
**Method Summary**

<table>
<thead>
<tr>
<th>Method</th>
<th>Signature</th>
</tr>
</thead>
<tbody>
<tr>
<td>prepareDiets()</td>
<td>public void prepareDiets()</td>
</tr>
</tbody>
</table>

**Method Detail**

**prepareDiets**

```java
public void prepareDiets()
```

**Class ZooArea**

```java
public class ZooArea
{
    This is a collection of cages designated as an area for assigning zoo staff.
}
```

**Field Summary**

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>areaName</td>
<td>String</td>
<td>This is the name of the zoo area which encloses a series of cages.</td>
</tr>
<tr>
<td>lnkCage</td>
<td>Cage</td>
<td>This records the assignment of cages to an area.</td>
</tr>
<tr>
<td>lnkFeeder</td>
<td>Staff_Member</td>
<td>All areas have one or more staff assigned.</td>
</tr>
<tr>
<td>lnkFeeder1</td>
<td>Feeder</td>
<td>There is one feeder employee assigned to each area of the zoo.</td>
</tr>
</tbody>
</table>

**Method Summary**

<table>
<thead>
<tr>
<th>Method</th>
<th>Signature</th>
</tr>
</thead>
<tbody>
<tr>
<td>enumerateCages()</td>
<td>public Cage enumerateCages()</td>
</tr>
</tbody>
</table>

**Field Detail**

**areaName**

```java
private String areaName
{
    This is the name of the zoo area which encloses a series of cages.
}
```

**InkCage**

```java
private Cage InkCage
{
    This records the assignment of cages to an area. Every area has one or more cages. Every cage belongs to an area.
}
```

**InkFeeder**

```java
private Staff_Member InkFeeder
{
    All areas have one or more staff assigned. A staff member may or may not be assigned to a particular area.
}
```

**InkFeeder1**

```java
private Feeder InkFeeder1
{
    There is one feeder employee assigned to each area of the zoo. Although many staff members may actually participate in caring for the animals, one staff
member, the feeder, is responsible for the preparation and delivery of their food.

### Method Detail

**enumerateCages**

```java
public Cage enumerateCages()
```

This service successively returns a link to each of the cages belonging to this area.

### UseCase Diagrams

**UseCase Diagram Feeding the Animals**

This use case describes the interaction of the Feeder with the zoo system in creating a list of animals to be fed.

**Diagram Contents Summary**

- **Actor** Pat Feeder
- **Actor** Randy Nutritionist
- **System Boundary** Zoo
  - **UseCase** Create Serving List
  - **UseCase** Feed the Animals
UseCase Prepare Animal Diets

Actor Detail

Actor Pat Feeder

"Pat" is a typical feeder employee of the zoo.

"Communicates" links

to UseCase Create Serving List

to UseCase Feed the Animals

Actor Randy Nutritionist

"Communicates" links

to UseCase Prepare Animal Diets

System Boundary Detail

System Boundary Zoo

The zoo system boundary represents the information system functions that support the zoo operations.

backgroundColor: 200,200,200

UseCases

UseCase Create Serving List

The feeder is responsible for feeding a group of animals housed in the part of the zoo for which he/she is responsible. This use case describes the "Pat" visible activities that the system exposes to Pat.

preconditions:
The employee is a feeder. All the animals have been assigned to cages. All the cages have been assigned to areas in the zoo. All the dietary items for each animal have been defined.

postconditions:
A serving list has been created that lists all animals in the feeder's area of responsibility. A complete list of serving objects has been created which satisfies the collective needs of the animals in the feeder's charge.

normalFlow:
1. Feeder gets a list of cages in the area he/she is responsible for.
2. Feeder gets list of animals in each of the cages in his/her area.
3. Feeder instructs each animal in his/her list to create an individual serving list using the defined diet.
4. The serving list for each animal is check against available food stores for adequacy.
5. The complete serving list is ready for scheduled delivery.

alternateFlow:
4.a There are insufficient food stores for a particular animal.
5.a Some animals are omitted from the final feeding list for lack of food.

UseCase Feed the Animals

preconditions:
The feeder has prepared a serving list for all animals in his/her area.

postconditions:
Every serving on the feeder's serving list has been delivered and the feeding times have been recorded.
normalFlow:
1. Iterate through the serving items in the serving list (these should be ordered by cages and areas).
2. Deliver the serving to the animal.
3. Record the time the animal is fed.

**UseCase Prepare Animal Diets**

**preconditions:**
All animals have been assigned to cages. All cages have been assigned to areas. All necessary food stores have been defined.

**postconditions:**
Every animal has one or more defined dietary items to direct their feedings.

**normalFlow:**
1. Iterate through the areas.
2. Iterate through the cages.
3. For each animal create a dietary item for that animal based on available food stores.