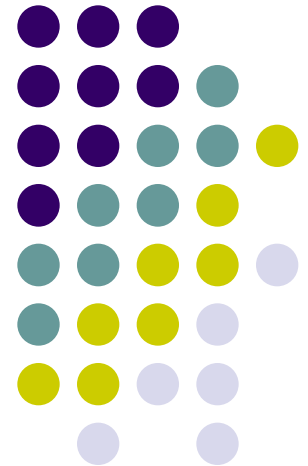
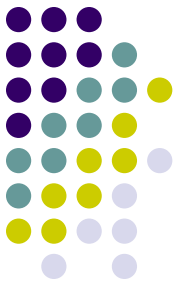


**ПРЕДСТАВЛЕНИЕ
ЗНАНИЙ:**

**РАЗРАБОТКА
ОНТОЛОГИИ**

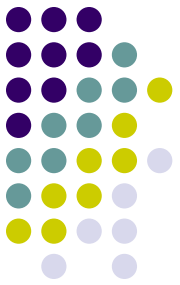


СОДЕРЖАНИЕ



1. Разработка онтологии и
фреймовая модель
2. Базовые принципы и шаги разработки
3. Определение классов и атрибутов
4. Проблемы концептуализации:
 - Подкласс или свойство?
5. Анализ построенной иерархии
6. Соглашения об именовании концептов
7. Домашнее задание

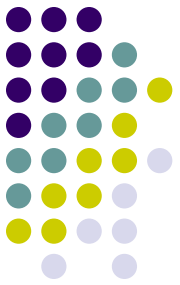
РАЗРАБОТКА ОНТОЛОГИЙ



- Рассматриваем онтологии, основанные на системах представления декларативных знаний
- Фреймовая модель – самая гибкая из существующих моделей ПЗ (декларативно-процедурная модель)
- Фреймовая модель применялась для онтологий:
 - начальная версия *CYC*
 - онтологии на базе *Protégé-Frames* (1999 -2000 гг.)
- *PROTÉGÉ* – свободно распространяемый редактор онтологий и фреймворк для построения БЗ, поддерживает два способа моделирования:
Protégé-Frames и *Protégé-OWL*
- В настоящий момент для разработки онтологий – в основном применяется версия *Protégé-OWL*
- Методология разработки онтологии
для декларативных фреймовых систем

ОНТОЛОГИИ НА ОСНОВЕ ФРЕЙМОВОЙ МОДЕЛИ

Система *PROTÉGÉ* 3.5



- Основные компоненты онтологии:
 - Классы, или понятия/концепты (фреймы-прототипы)
– сущности с набором одинаковых свойств)
 - Экземпляры классов (фреймы-экземпляры)
 - Слоты – атрибуты/свойства классов и экземпляров
 - Отношения – связи классов и экземпляров:
Класс-Подкласс, Класс-Экземпляр,
Слот (значение)-Класс/Экземпляр
 - Ограничения на значения слотов, в том числе
процедурные (процедуры,
присоединенные к классам ~ роль аксиом)
- База знаний: Онтология + экземпляры классов

БАЗОВЫЕ ПРИНЦИПЫ РАЗРАБОТКИ ОНТОЛОГИИ



Не существует единственно правильного способа моделирования предметной области (модели ПО) – возможны альтернативные варианты

- Онтология строится исходя из цели – предполагаемого приложения, а также ожидаемых расширений, иногда: от требований по совместимости с другими моделями
- Разработка онтологии – итеративный процесс, при котором происходит оценка и уточнение компонентов
- Понятия в онтологии должны быть близки к объектам, понятиям и отношениям в ПО, т.е. так или иначе отражать реальный мир

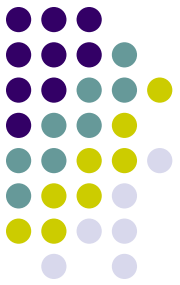
Важно: Наглядность, расширяемость, простота онтологии

ШАГИ РАЗРАБОТКИ ОНТОЛОГИИ



1. Определение области и масштаба онтологии
2. Выявление важных терминов онтологии
3. Определение классов, их связей, иерархии (концептуализация, детализация)
 - ✓ Нисходящая разработка (от самых общих понятий)
 - ✓ Восходящая разработка (от самых конкретных)
 - ✓ Комбинированная разработка: сочетание нисходящего и восходящего подходов
4. Выявление свойств/атрибутов классов
5. Уточнение аспектов значений атрибутов
6. Создание экземпляров классов

Шаги 3 и 4 – самые важные, тесно переплетены



РАЗРАБОТКА: ШАГИ 1 и 2

1. Определение области и масштаба онтологии, т.е. степени охвата ПО, уровня детализации, необходимого для приложения. Основные вопросы:
 - Какую область будет охватывать онтология?
 - Для каких целей будет она использоваться?
 - На какие типы вопросов должна давать ответы БЗ?

Онтология не должна содержать всевозможную инф-ию о ПО: не нужно конкретизировать или обобщать больше, чем необходимо для приложения

2. Выявление важных терминов онтологии
 - Сбор *гlossария* (словаря ПО): термины + определения
 - Очень часто эти термины – существительные (объекты) или глаголы (отношения объектов) в текстах ПО

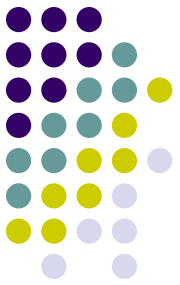
ШАГ 3: ОПРЕДЕЛЕНИЕ КЛАССОВ



Выявление классов, их иерархических связей

- Критерий: Если класс *A* – надкласс класса *B*, то каждый экземпляр *B* также является экземпляром *A*
- Транзитивность иерархических отношений:
Если *B* – это подкласс *A*, а *C* – подкласс *B*, то *C* – подкласс *A*
Сыр – Мягкий_Сыр – Тильзитер
- Возможная ошибка: включение в иерархию варианта одного и того же понятия как в единственном, так и во множественном числе, сделав первое подклассом второго:
Сыры – Сыр (способ избежать: всегда делать имена классов – или в единств., или во множеств. числе)
- Важно различать класс и его имя: Классы представляют понятия ПО, а не слова, которые обозначают эти понятия
Синонимы – всего лишь разные имена понятия:
Твердый_Сыр = Выдержанный_Сыр

ШАГ 4: ВЫЯВЛЕНИЕ АТРИБУТОВ

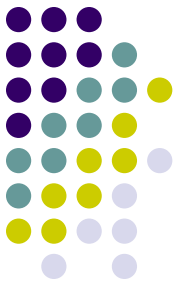


Описание внутренней структуры классов/понятий:

Сыр: Название, Жирность, Состав, Производитель

- Слоты/атрибуты/свойства классов – вероятнее всего возникают из терминов глоссария (из оставшихся)
- Представляют самые значимые свойства класса
- Все подклассы класса наследуют слоты класса
- Возможно **множественное наследование**
- Допускается значение слота **по умолчанию**
(отлично от значения слота)
- Возможны **обратные слоты** (обратные отношения):
значение слота – ссылка на другой класс, и в этом классе есть слот, ссылающийся на класс исходного слота:
Производитель (Завод) – Производит (Сыр)
по сути, дублирование информации

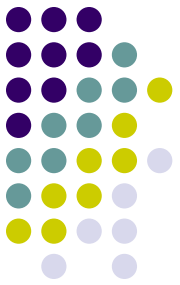
ТИПЫ СЛОТОВ-АТТРИБУТОВ



Несколько типов свойств /атттрибутов объектов:

- «внутренние» свойства, такие как *Вкус Сыра*
- «внешние» свойства, такие как
Название Сыра, Производитель
- части, если экземпляры классов имеют структуру
 - могут быть физическими (части машины)
 - абстрактными (блюда, входящие в обед)
- связи экземпляров (индивидуальных концептов):
отношения между конкретными членами классов:
Сыр – Производитель – Завод

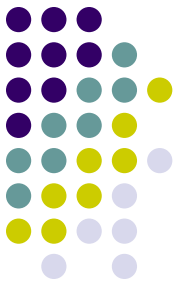
ШАГ 5: УТОЧНЕНИЕ АТТРИБУТОВ



Определение различных аспектов значения каждого слота-атрибута:

- Ограничение на значение:
 - Тип значения: числовой, строковый, логический, нумерованный (=перечислимый, *Symbol*)
 - Ссылка на экземпляр другого класса
- **Мощность** (кардинальность) значения: единичная, множественная (максимальная и минимальная)
- Значение по умолчанию (*Default*)
- Процедуры, присоединённые к слоту
- ❖ **Диапазон (область)** слота-атрибута:
классы, к которым слот присоединен
(свойство которых описывает)

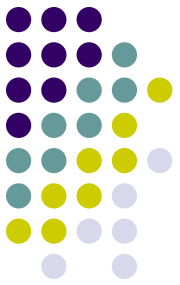
ДИАПАЗОН СЛОТА



Основные правила определения **области** слота:

- Слот должен быть определен у самого общего класса, у которого может быть данное свойство (т.е. делается как можно более общим);
тогда диапазон слота – этот класс и все его подклассы
- Необходимо гарантировать, что каждый класс, к которому слот присоединяется, на самом деле имеет это свойство:
экземпляры всех классов в диапазоне значений слота должны быть потенциальными заполнителями слота
- Если список классов из диапазона слота включает класс и его подкласс, следует удалить подкласс
- Если список классов, определяющих диапазон значений слота, включает все подклассы класса A , но не включает сам класс A , то в диапазон значений должен входить только класс A , а не его подклассы

КАТЕГОРИЗАЦИЯ: ВВЕДЕНИЕ ПОДКЛАССА



Принципы введения новых классов

Обычно подклассы:

- имеют дополнительные свойства, которых нет у надкласса (чаще: внутренние свойства), или же
- имеют ограничения, отличные от тех, которые есть у надкласса, или же
- состоят в других отношениях, нежели надклассы
- ❖ На практике к вводимому подклассу нужно
 - добавить новые слоты
 - и/или определить новые значения слотов
 - и/или переопределить некоторые аспекты значений унаследованных слотов
- ❖ Однако не следует создавать подклассы класса для каждого дополнительного свойства/ограничения

КОНЦЕПТУАЛИЗАЦИЯ: НОВЫЙ КЛАСС Vs ЗНАЧЕНИЕ СВОЙСТВА?



Одно из самых сложных решений: определить, когда ввести новый класс, а когда сформулировать различие с помощью разных значений слотов/свойств:

Студент : бакалавриат, магистратура, специалитет

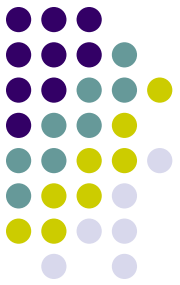
Ответ зависит от цели/масштаба онтологии:

- Введение многих классов ведет к иерархии с очень большой степенью вложенности, с ненужными классами
- Введение многих свойств ведет к «плоской иерархии», где очень мало классов, но с большим количеством слотов

Новый класс создается, если

- Понятия с разными значениями атрибута-слота становятся ограничениями для различных слотов в других классах
- В данной ПО разграничение важно и различаемые объекты считаются разными типами объектов

В ином случае различие представляется в значении слота



ШАГ 6: СОЗДАНИЕ ЭКЗЕМПЛЯРОВ

Экземпляры классов – индивидуальные концепты,
для их определения требуется:

1. Выбрать класс
2. Создать отдельный экземпляр этого класса
3. Ввести значения слотов

Важный вопрос: Экземпляр или Класс? *Сыр_Тильзитер*

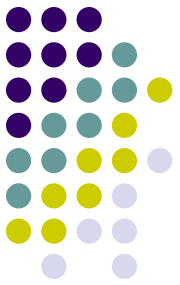
- Это зависит от потенциальных приложений онтологии и нужной глубины детализации
 - Глубина детализации также определяется приложением
 - Отдельные экземпляры – самые конкретные понятия в БЗ
 - Если понятия формируют естественную иерархию, то следует представить их как классы
- ❖ Возможность *абстрактных классов*:
У абстрактного класса не может быть прямых экземпляров
Корневой абстрактный класс *Thing*

АНАЛИЗ ДЕРЕВА ИЕРАРХИИ



- «Эстетический» анализ иерархического дерева: глубина ветвей, соотношение с шириной, симметричность
- Логический и концептуальный анализ – основа *Дочерние вершины* в дереве иерархии – классы, являющиеся непосредственными подклассами одного и того же класса
- Все дочерние вершины должны соответствовать одному принципу категоризации,
неверно: *Твердый_Сыр* и *Козий_Сыр*
- Все дочерние вершины должны располагаться на одном уровне обобщения, т.е. быть однорядковыми понятиями,
неверно: *Твердый_Сыр* и *Тильзитер*

АНАЛИЗ УЗЛОВ ОДНОГО УРОВНЯ ИЕРАРХИЧЕСКОГО ДЕРЕВА



- Количество дочерних вершин?
 - Если класс имеет только один прямой подкласс, то, возможно, онтология неполная
 - Если у класса более 7-9 подклассов, то, возможно, необходимы дополнительные промежуточные категории
- Однако если нет способа естественной группировки классов, то не надо создавать искусственные классы (т.к. онтология – отражение реального мира)
- Нередко дочерние вершины – *дизъюнктивные* (непересекающиеся) подклассы

ПРАВИЛА ПРИСВАИВАНИЯ ИМЕН



- Определение **единых правил именования** понятий в онтологии, их строгое соблюдение:
 - делает онтологию более простой для понимания
 - помогает избежать некоторых ошибок при разработке онтологии
- Правила зависят от ряда факторов:
 - Имеет ли система ПЗ одно и то же пространство имен классов, слотов и экземпляров?
 - Различает ли система регистр букв?
 - Какие разделители разрешаются в именах?
- *PROTÉGÉ 3.5*: единое пространство имен для всех фреймов, различается регистр букв, т.е. возможен класс *Аудитория* и слот *аудитория*

СОГЛАШЕНИЯ ОБ ИМЕНАХ



Общепринятые соглашения:

- Начинать имена классов с большой буквы, а имена слотов – с маленькой (если регистр букв важен)
- Использовать имена классов либо в единственном, либо во множественном числе
(на практике чаще применяется единственное)
Следует придерживаться чего-то одного !
- Обычно лучше не сокращать имена понятий
- Имя надкласса должно входить или во все имена прямых подклассов, или ни в одно из них:
Аудитория_лекционная, Аудитория_простая



СПАСИБО ЗА ВНИМАНИЕ!

ПРАКТИЧЕСКОЕ ЗАДАНИЕ № 2

(обязательно для допуска к экзамену)



- Разработка формальной декларативной онтологии на основе системы *PROTÉGÉ 3.5*
- Работа индивидуально или в группах по 2-3 человека
- Выбор ПО – свободный (м.б. взяты ПО из задания по БД), но ПО обсуждается и подтверждается преподавателем
- При желании за дополн. баллы в разработанную онтологию и БЗ добавляются ограничения в виде функций ЯП
- Контрольный срок сдачи – **14 апреля**, выбор ПО – **31 марта**
- Обязательная сдача отчета по заданию (1-5 стр.) с кратким описанием всех основных этапов (этапы 1-6) разработки онтологии и ее компонентов
- Задание сдается последовательно: (1 этап – до **7 апреля**)
 1. Отсылка по почте Баевой Н.В. файла с онтологией и отчетом
 2. Окончательная сдача на занятии