Lecture day 2016-04-07
======================

SLIDES: Introductory Modeling Example Employing UML and OCL
  [UML: Unified Modeling Language, OCL:Object Constarint Language]

- System design in an object-oriented way employing USE with UML and OCL
  [USE: Uml-based Specification Environment]
  Structure and behavior (overview Slide 4):
    Class diagram, Statechart diagram, Object diagram
    Structure: Class dia (classes, attrs, assocs, ...), OCL invariants
    Behavior: Statechart dia, OCL contracts (pre- and postconditions)
              SOIL implementation
              [SOIL: Simple Ocl-like Imperative Language]
  Aim: model a system in an abstract way without going into a
       programming language implementation

- OCL used for giving precision (in example Slide 29-40)
  - in class diagrams for
    (a) class invariants (Slide 36ff)
    (b) operation contracts (pre- and postconditions) (Slide 37ff)
        @pre in postconditions
    (c) attribute and association derivation rules (Slide 33ff)
    (d) attribute initializations (Slide 29ff)
  - in protocol state machines for
    (e) state invariants (Slide 31ff)
    (f) transition pre- and postconditions (not in example)
   - furthermore for
    (g) ad-hoc OCL queries in object diagrams (Slide 4ff)
    (h) expressions within SOIL (Slide 33ff)

Aim of USE: support development by reasoning about model through

(a) validation, i.e., checking informal expectations
    against formally present properties, for example,
    by stating OCL queries against a reached system state

    Slide 4 (Object diagram, OCL query), Slide 11-18 (animation)

    scenarios - states as object diagram
              - operation calls as sequence or collaboration diagrams

(b) verification, i.e., checking formal properties of the model,
    for example by considering model consistency or
    independence of invariants; USE supports making deductions from
    stated model on the basis of finite search space of possible
    system states

    not in the current example

Lecture day 2016-04-14

System behavior determined by OCL contracts (Slide 10)
- pre- and postconditions
- role of @pre in postconditions

Scenarios as Sequence diagrams or Collaboration diagrams
- selecting currently interesting (relevant) parts
- Sequence diagrams: emphasis on 'time'
  object lifeline selection
  manual selection via context menue:
    - message kind, statechart states
- Collaboration diagrams: emphasis on object connection / 'space'
  selction via message kind or interval
  selection with OCL

Slides: Teaching Touchy Transformations

Example: Library
- class diagram for MaxInvsMinPrepos
  classes: User, Copy, Book
  associations: BelongsTo, Borrows
  roles
  multiplicities

OCL collections: Set, Sequence, Bag (three collection kinds)
  User.allInstances:Set(User)
  elNa.authSeq:Sequence(String)
  Copy.allInstances.numReturns:Bag(Integer)

  Fourth OCL collection kind: OrderedSet

Lecture day 2016-04-21
======================

Slides: Teaching Touchy Transformations

Example: Library
- cloze test
- class diagram for MaxInvsMinPrepos / MaxPrepos
  classes: User, Copy, Book
  associations: BelongsTo, Borrows
  roles, multiplicities
  attributes (Sequence-valued attribute authseq)
  operations (assoc Borrows modifiable from both sides;
    different parameter structure for 'return' due to multiplicities)
  role name example: Person with reflexive association
      parenthood(parent[0..2]:Person,child[0..*]:Person)
    role names in object diagram needed:
      ali -parent---child- ben -parent---child- cyd
      ali parent of ben (= ben child of ali), ben parent of cyd
      ali grandparent of cyd

- invariants
  - keys
  - formatOk
  - noDoubleBorrowings (could be in User, Copy OR Book)
  - authSeqExistsAndUnique (building integer sets;
    universal quantification; access to sequence element)

Different versions of the same model in different styles:

- MaxInvsMinPrepos: maximal restrictions in invariants,
    minimal restrictions in pre- and postconditions

- MaxPrepos: maximal restrictions in pre- and postconditions
    turning invariants into preconditions
    guaranteeing that the invariants cannot fail

- Assoc2Attr: transforming each association into 2 attributes with
    additional invariants guaranteeing that the 2 attributes express
    the same facts as the original association

- Invs2Super: abstract superclass Keyed for User,Copy,Book
    advantage: formulation of invariant for keys
    only one time and not three times as in the original model

Lecture day 2016-04-28
==========

Slides: Syntax und semantics of UML class diagrams

Part 1: Examples from UML Notation Guide
- **class**
- **association**
- **association class**
  [class - object, association - link, association class - link object]
- **ternary association**
- **qualified association**
- **association on association class**
- **aggregation, composition**
- **generalization [alias specialization, inheritance]**
- **generalization discriminator [e.g. venue in Vehicle example]**
- **generalization constraints [default: overlapping, incomplete]**
  - **overlapping, disjoint**
  - **incomplete, complete**

Part 2: Explaing UML class diagram features with transformations
- **basic idea:**
  transform complex language element into simple element
  with additional OCL constraint
- **multiplicity to constraint (multiplicity = cardinality)**
  - **binary and ternary case**
- **ternary association to class**
- **association class to ternary association**
- **aggregation, composition**
- **acyclicity on object diagram level**
- **object sharing (or forbidding object sharing)**
- **distinction: association, aggregation, composition**
- **Paper example**
- **Brain example**

Lecture day 2016-05-12
==========

Slides: UML associations (Example for all association kinds)

- **binary association**
- **asociation class**
- **reflexive association**
- **functional association**
- **ternary association**
- **aggregation**
- **composition**
- **qualified association**
- **derived association**

class diagram, object diagrams

variations with possible invariants

Slides: Syntax und semantics of UML class diagrams

- **generalization**
  - **disjoint, overlapping**
  - **complete, incomplete**
  - **OCL constraints**

Slides: Collection operations in OCL

- **Collection kinds: Set, Bag, Sequence, OrderedSet**
- **Collection operations**
  - **operations on all collection kinds**
  - **operations only defined on particular collection kinds**

- **Differences and commonalities between collection kinds**
  - **commutativity law**
  - **absorbtion law**

- **Overview with Col{7,8}, Col{8,7}, Col{7,8,7}**

Lecture day 2016-05-19

Slides: Teaching Touchy Transformations

Transformation of UML class diagram into relational database schema

```
create table User ( name String, address String,
  primary key (name) )

create table Copy ( signature String, numReturns Integer,
  name String, -- assoc Borrows(user:User,copy:Book)
  title String, -- assoc BelongsTo(copy:Copy,book:Book)
  primary key (signature),
  foreign key (name) references User (name),
  foreign key (title) references Book (title) )

create Table Book ( title String, year Integer,
  primary key (title) )

create table authSeq ( -- attr Book::authSeq:Sequence(String)
  titleAS String, pos Integer, author String,
  primary key (titleAS,pos),
  foreign key (titleAS) references Book (title),
  unique (titleAS,author) )
```

Slides: Collection operations in OCL and concepts of and
examples for OCL collections

Collection operations

- collection kinds Set, Bag, Sequence, OrderedSet

- equality =, inequality <>
- including, excluding
- includes, excludes
- isEmpty, notEmpty, size
- select
- collect
- forAll, exists
- iterate

- examples for all operations
- general scheme of collection operations with variables and expressions
- many details for iterate, among them:
  COLEXPR->iterate(ELEMVAR:ELEMTYPE; RESVAR:RESTYPE=INITEXPR | ITEREXPR)
- examples for expressing operations with iterate

Lecture day 2016-06-02

SLIDES Collection operation closure and addendum to iterate

- closure syntax

- examples
  - flight connections (from SLIDE Summary OCL collection operations)
  - parent/child
  - cycles in the underlying object/data graph

- sets versus bags

- Summary of collection operations
  OCL collection operations on one page

SLIDES Employing the tool USE for Model-Based Engineering

- Motivation modeling

- USE

- class, object, sequence, statechart, communication diagram

- role of OCL

- model validator
  - configurations
  - use cases

- classifying terms

[- transformation models]

[- filmstrip models]

Lecture day 2016-06-09
==========

SLIDES Metamodeling of the ER and Relational Model

Example ER 2 RelMod Employee-worksFor-Company
in ER and RelMod

Example Diana-marries-Charles

In the example and in general one can consider

- ER syntax level

- RelMod syntax level

- ER semantics level

- RelMod semantics level

+ syntax level corresponds to the datamodel schemata

+ semantics level corresponds to datamodel states

ER syntax concepts

- ErSchema

- Entity

- Relship

- Relend

- Attribute

- DataType

Construction of an object diagram: Facebook example

- Entity Person

- Relship Friendship

- Attribute(s) userid, name, date

- Relend(s) inviter, invitee

- Datatype String, Date

Lecture day 2016-06-16
==========

SLIDES Metamodeling of the ER and Relational Model

Example Diana-marries-Charles

Constraints for

- ER syntax level

- RelMod syntax level

- ER semantics level

- RelMod semantics level

+ explanation of constraints by

  - OCL formulation

  - object diagrams satisfying / violating the constraints

+ consideration of special scenarios

  - diana moves from wembley to windsor

  - diana moves and marries

+ transformation constraints

  - only syntax level

Lecture day 2016-06-30

USE / OCL experiment

The goal of this case study is to evaluate some of the properties of classifying terms such as their usability, effectiveness and scalability. For that we have prepared an exercise where a set of classifying terms have to be specified to characterize some particular sorts of models.

The experiment consists of two parts. The first one (with a maximum duration of 90 min.) is dedicated to explain the main concepts used in the experiment, and the example that will serve as the case study for the experiment. The second part (with a duration of 60 min.) is a test that the subjects of the experiments should do, by responding the questions provided in a questionnaire.

The questionnaire with the concrete questions will be provided at the beginning of the second part of the experiment. Subjects of this study are supposed to be knowledgeable of UML and OCL. A minimum requirement is that they have written at least 10 OCL expressions in their lives. This would account for a knowledge level of 1. OCL gurus have a knowledge of 10. If you do not know OCL, or have never written an OCL expression, please contact the experiment instructor as soon as possible.

For any question you may have about the documentation, the USE tool or anything related to the experiment, please feel free to consult the instructor.

Lecture day 2016-06-30

Course summary

UML

Diagrams: Class, Object, Sequence, ...

Class diagram: Class, Association, Role, Multiplicity, Generalization, Disjoint/Overlapping, Complete/Incomplete, Aggregation, Composition, Association class, ...

OCL

Concepts based on data types, classes, associations, roles, multiplicity; core concepts: objects, values, navigation, logic, collections, collection operations.

Collection kinds [Set, Bag, Sequence, OrderedSet], collection properties [insertion order, insertion frequency], ...

Collection operations: [many operations on all collection kinds] exists, ..., select, collect, ..., iterate, asSet, ...; [few operations on special collection kinds] at, first, last, subsequence, ...

USE

USE Windows (Project browser, project browser detail, log, class diagram, object diagram, invariant evaluation, ocl expression evaluation, object properties, class extent, command script, sequence diagram, communication diagram, [Advanced: evaluation browser], ...), USE Functionalities though GUI and CLI

Supported constraints (inv, pre/post), operation definitions, [init, derivation]

Metamodeling

ER, RE, Class diagram, Syntax (Schema), Semantics (State), diana-and-charles-example, ...
