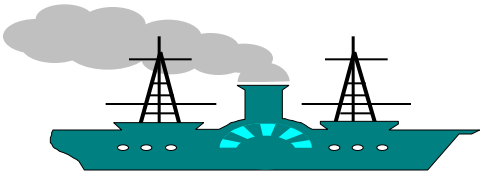
XII. Describing Business Rules

Business Rules
Structured English
Decision Tables
Decision Trees
The Object Constraint Language (OCL)



Business Rules

- Business rules are used to describe the properties of an application, e.g., the fact that an employee cannot earn more than his or her manager, or that every employee has a unique employee number..
- A business rule can be:
  - ✓ An ***integrity constraint*** on the data of the application, e.g., "each employee earns less than her manager";
  - ✓ a ***derivation rule***, whereby information can be derived from other information, e.g., "the price of a train ticket, in Canadian dollars, is given by the distance to be travelled in kilometers, multiplied by 0.2, multiplied by 1.5 for a first class ticket" .

Examples of Business Rules

Constraints
(BR1) The manager of a department must belong to that department. (BR2) An employee cannot earn more than her manager. (BR3) A department of the Toronto office can only be managed by an employee who has ≥ 10yrs experience. (BR4) An employee can only participate in projects associated with her department.
Derivations
(BR5) The budget of a project is the sum of all salaries of participating employees, multiplied by 3.

Specifying Business Rules

- How do we specify business rules? We'll be looking at several alternative notations.
  - ✓ **Natural Language** -- use unrestricted natural language...but such descriptions can be highly ambiguous
  - ✓ **Structured English** -- use a subset of a natural language (both syntactically and vocabulary-wise) to minimize ambiguities...this has been used with some success
  - ✓ **Decision Tables** -- use a table representation of alternative outcomes (similar to truth tables)
  - ✓ **Decision Trees** -- use a tree representation of alternative outcomes

We need representations that are understandable by end user

Structured English

Looks a lot like pseudocode:

For each LOAN ACCOUNT NUMBER in the LOAN ACCOUNT FILE
do the following steps:
    If the AMOUNT PAST DUE is greater than $0.00 then
    while there are LOAN ACCOUNT NUMBERS for the CUSTOMER NAME do the following:
        sum the OUTSTANDING LOAN BALANCES
        sum the MINIMAL PAYMENTS
        sum the PAST DUE AMOUNTS
    report the CUSTOMER NAME, LOAN ACCOUNT on OVERDUE CUSTOMER, LOAN ANALYSIS

Takes some effort to specify, not very readable, too close to an implementation

Another Example

```
do while there are more staff in the list
      calculate staff bonus
      store bonus amount
      begin case
      case bonus > £250
              add name to StarOfTheMonth list
      case bonus < £25
               print warning letter
       end case
end do
```

Some Rules for Structured English

- Use only nouns and terms defined in the project dictionary
- Avoid compound sentences because they can be highly ambiguous
- Avoid undefined adjectives and adverbs (such as "good", "nice" etc.) unless if clearly defined in the dictionary in terms of value ranges (e.g., "good" ↔ 65-75%)
- Avoid language that destroys the natural flow of control within the process (i.e., goto's)
- Use a limited set of flow constructs, such as sequencing, if-then-else, while do etc.

Decision Tables

- If there are n parameters (or, ***conditions***) to a decision, each of which can take k_1, k_2,...k_n values, then make up a table with k_1 * k_2 *... * k_n columns and as many rows as there are possible actions (or, ***outcomes***)
- For example:
- "If the plane is more than half full and the flight costs more than $350 per seat, serve free cocktails, unless it is a domestic flight. Charge for cocktails in all domestic flights where cocktails are served, i.e., those that are more than half full"

conditions	**Domestic**	**Y**	**Y**	**Y**	**Y**	**N**	**N**	**N**	**N**
	≥ half full	**Y**	**Y**	**N**	**N**	**Y**	**Y**	**N**	**N**
	≥ $350/seat	**Y**	**N**	**Y**	**N**	**Y**	**N**	**Y**	**N**
outcomes	**Serve cocktails**	**X**	**X**			**X**	**?**	**?**	**?**
	Free cocktails					**X**			

How to Construct Decision Tables

1. Identify all conditions and all outcomes
2. Create the decision table, with one column for each possible combination of condition values and one row for every possible outcome
3. Fill in the table
4. Eliminate ambiguities, uncover cases, contradictions, redundancy

Completion and Simplification of a Decision Table

Completion

Domestic	Y	Y	N	N	N
≥ half full	Y	N	*	Y	N
≥ $350/seat	*	*	Y	N	N
Free cocktails			X		
Charge cocktails	X			X	
No cocktails		X			X

Simplification

Domestic	Y	Y	Y	Y	N	N	N	N
≥ half full	Y	Y	N	N	Y	Y	N	N
≥ $350/seat	Y	N	Y	N	Y	N	Y	N
Serve cocktails	X	X			X	X	X	
Free cocktails					X		X	
Charge cocktails	X	X				X		
No cocktails			X	X				X

Going to a Place

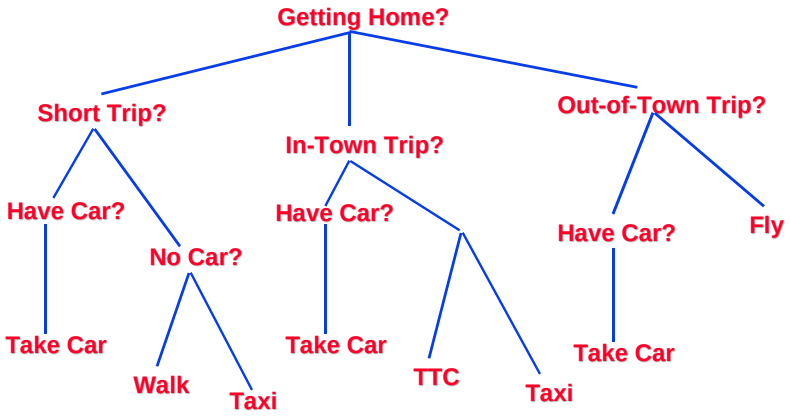
In town?	Y	Y	Y	Y	Y	N	N	N
Short distance?	Y	Y	Y	N	N	Y	N	N
Good weather?	Y	N	N	*	*	*	*	*
Can afford?	*	Y	N	Y	N	*	Y	N
Walk!	X							
Take TTC!			X		X			
Take taxi!		X		X				
Take train!						X		X
Fly!							X	

Another Example

Is budget likely to be overspent?	N	Y	Y
Is overspent likely to be over 2%?	*	N	Y
No action!	X		
Write letter!		X	X
Set up meeting!			X

Decision Trees

- Nodes of a decision tree represent partial outcomes, successors of a node represent mutually exclusive alternatives, leaves of a decision tree represent outcomes.



Decision Trees: An Example

Note: This is a real example (...):

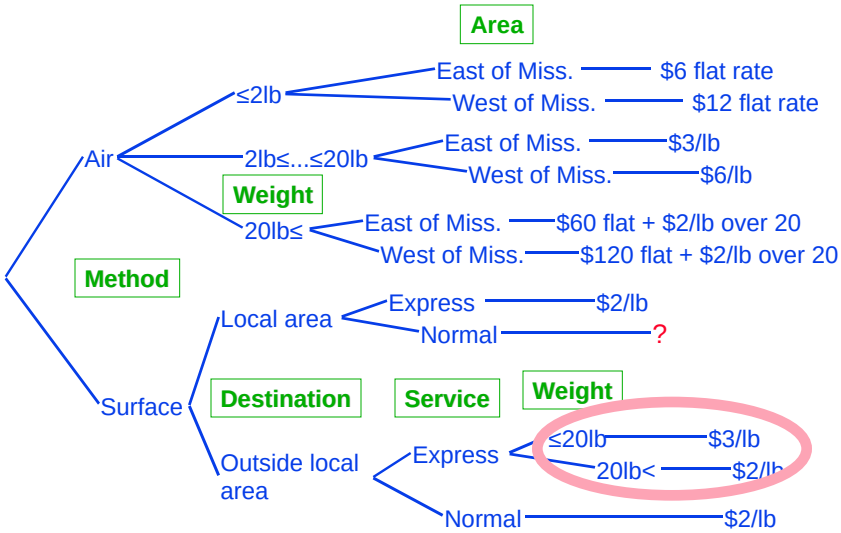
"Air shipping charges are set depending on the weight of a parcel. The basic rate is $3/lb, reducing to $2/lb for excess over 20 lb, with a minimum of $6. Surface freight is $2/lb for express delivery. However, this rate only applies in the local delivery area. If the shipping address is outside the local area and the parcel weighs more than 20lb, or express delivery is not required, the surface rate is the same as for local delivery (express). Normal delivery of packages is $2/lb up to 20lb is $2/lb, with $1 express surcharge (per pound).

Notwithstanding the provisions of the previous paragraph, air freight to destinations west of the Mississippi is charged at double rate"

Clarifications

- **Question:** Is there a difference between freight shipping and handling
- **Answer:** No, all rates include freight and handling.
- **Question:** The description mentions "up to 20lbs" and "over 20lbs". Which rate applies for exactly 20lbs?
- **Answer:** It's generally understood that "up to 20lbs" means "up to and including 20lbs". We can't spell out every little thing, you know?
- **Question:** The fourth sentence could be read in two ways: "both outside the local area and also over 20lbs, or, alternatively, express not required" or "outside the local area and, in addition, either over 20lbs or express not required". Which is correct?
- **Answer:** The second one. The first meaning couldn't be right because you would end up charging the local express rate when express delivery was not required. I see your point though, it is a bit confusing...

The Freight Decision Tree



It costs $57 to send 19lbs outside local area, express, but only $42 to do the same for 21lbs...

Summary

- **Decision trees** are best used with applications involving up to 15-20 outcomes
- **Decision tables** are more appropriate for problems involving complex combinations of up to 5-6 conditions (but can handle much larger number of outcomes)
- **Structured English** (and state-oriented models) are most appropriate for problems involving sequential considerations of alternative steps.

Pre- and Post-Conditions

- What conditions must be true before an operation can take place?
- What are the conditions that will be true after an operation is completed?
  - For example, consider

    Campaign.assignStaff(creativeStaff.id)
    - Pre-condition: *creativeStaffObject*.id is not null;
    - Post-condition: a link is created between *campaignObject* and *creativeStaffObject.*

The Object Constraint Language

- Some constraints can be adequately expressed in the graphical language (e.g., multiplicity of an association).
- Some can not. For example, constraints within operation specifications (pre- and post-conditions)
- The Object Constraint Language (OCL) provides a formal language for specifying constraints which can supplement the models created in terms of UML diagrams.
- The language has a precise syntax that enables the construction of unambiguous statements.
- Each expression has an associated ***context***, which is usually the class to which the expression is attached.

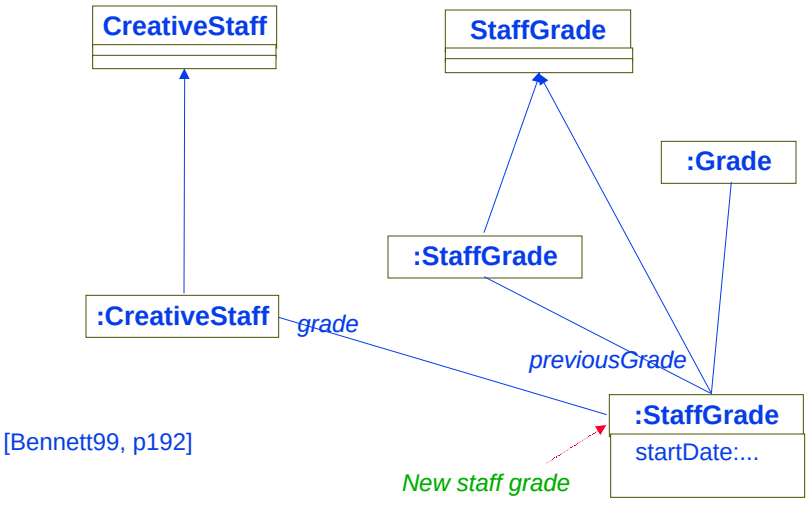
OCL Examples

OCL expression	Interpretation
Person self.age	In the context of a specific person, the value of the property 'age' of that person—i.e. a person's age.
Person self.income >= 5,000	The property 'income' of the person under consideration must be greater than or equal to 5,000.
Person self.wife->notEmpty **implies** self.wife.sex = female	If the set 'wife' associated with a person is not empty, then the value of the property 'sex' of the wife must be female. The boldface denotes an OCL keyword, but has no semantic import in itself.
Company self.employee->size <= 50	The size of the set of the property 'employee' of a company must be less than or equal to 50. That is, a company cannot have more than 50 employees.
Company self.employee->select (age > 50)	This specifies the set of employees of a company whose age is greater than 50.

Pre- and Post-Conditions in OCL

```
CreativeStaff::changeGrade(id:String, grade.id:String,
  gradeChangeDate:Date)
pre:        self.id->notEmpty
            self.grade.id->notEmpty
            self.gradeChangeDate >= today  (assumes no
                                            retroactive changes)
post: staffGrade[grade.id]->exists
            self.staffGrade->notEmpty
      self.staffGrade[grade.id].previousGrade->notEmpty
      self.staffGrade.gradeFinishDate = gradeChangeDate
```


What Does the Post-Condition Mean?



[Bennett99, p192]

Additional Readings

- [Warmer99] Warmer, J. Kleppe, A. ***The Object Constraint Language: Precise Modeling with UML,*** Addison-Wesley 1999.
- [Yourdon89] Yourdon, E. *Modern Structured Analysis*. Prentice Hall, 1989.
- [Meyer97] Meyer, B. *Object Oriented Construction*. Prentice Hall, 1997.
- [Senn89] Senn, J. A. *Analysis & Design of Information Systems*. McGraw-Hill, 1989.