SFWR ENG 3RA3 Summary

Author: Kemal Ahmed

Instructor: Dr. Ryszard Janicki

Date: Fall 2014

*Math objects made using* [*MathType*](http://www.dessci.com/en/products/mathtype/)*; graphs made using* [*Winplot*](http://math.exeter.edu/rparris/winplot.html)*.*

Please join GitHub and contribute to this document. There is a guide on how to do this on my GitHub.

Table of Contents

[Lecture 1 1](#_Toc401483819)

[Lecture 2 – Types of Statements 1](#_Toc401483820)

[Lecture 5 2](#_Toc401483821)

[Defining Requirements 2](#_Toc401483822)

[Knowledge Acquisition 2](#_Toc401483823)

[Lecture 6 3](#_Toc401483824)

[Lecture 6 3](#_Toc401483825)

[e.g. 3](#_Toc401483826)

[Entity Relationship (ER) Diagram 3](#_Toc401483827)

[e.g.) 4](#_Toc401483828)

[Data Flow Diagrams 4](#_Toc401483829)

[State Machine Diagram 4](#_Toc401483830)

[DDP 4](#_Toc401483831)

[Pairwise Comparisons 4](#_Toc401483832)

# Lecture 1

And so it begins…

# Lecture 2 – Types of Statements

**Descriptive Statement**: facts about the system, such as natural laws and physical constraints

[**Prescriptive Statement**](#_Prescriptive_Statements): desired behavioural properties of a system; can be negotiated

Types of prescriptive statements:

* **System Requirement** (SYSREQ): when the software interacts with the other system components, i.e. environment
  + vocabulary understandable by all parties
  + Types of SYSREQ:
    - Domain Property (DOM): affecting environmental phenomena, such as physics
    - Assumptions (ASM):
* **Software Requirement** (SOFREQ): relationship between a set of input variables, *I*, and *O*, the set of output variables
  + vocabulary understandable by software developers

# Lecture 5

## Defining Requirements

Types of projects:

* Rabbit:
  + Agile
  + Short life
* Horse:
  + Fast, strong, dependable
  + Most common in corporate
  + Medium longevity
* Elephant:
  + Solid, strong, long life

**Artifact-driven**: basing the requirements on data collection, questionnaires, etc.

* You can often collect too much data
* Only keep what you need to know
* *prune* the document space, so you only keep the useful data.

**Scenario**: similar to *storyboards*…

**Positive Scenario**: behaviour system should cover

* **Normal Scenario**: everything proceeds as expected
* **Abnormal Scenario**: a desired exception

**Negative Scenario**: behaviour system should exclude

## Knowledge Acquisition

**Stakeholders**: important to identify when determining who to customize the project towards

* Who is responsible for funding/using/managing the project?
* Caution: interactions with them must be done carefully

**Domain expertise**: what does the domain know / qualifications? Domain is who the project is directed towards

# Lecture 6

**Stakeholders-driven Elicitation Techniques**: methods of knowledge acquisition

* Interviews
  + Single interview for multiple stakeholders: faster, but less involving
  + Steps:
    - Select stakeholders
    - Background study
    - Predesign sequence of questions, focused on concerns of present stakeholder(s)
    - Begin by asking easy questions
    - Keep focus during interview
    - Ask ‘why’-questions
    - Record answers and reactions
    - Write report from transcripts
    - Confirm report with stakeholders interviewed
  + Types:
    - **Structured**: predetermined set of questions
    - **Unstructured**: free discussion of current system
    - Optimal: start with *structure*, then do *unstructured*

# Lecture 6

Observation:

* people behave differently when observed
* slow & expensive

Group sessions: more than 4 people

**Inconsistencies**: conflicting views or incorrect

### e.g.

**Boundary Condition**: the sample of instances where conditions conflict

**Divergence**: when two viewpoints have boundary conditions; they must be clarified

## Entity Relationship (ER) Diagram

|  |
| --- |
| **Entity**: class of concept instances |
| Attribute 1  …  Attribute *n*: intrinsic feature of an entity (regardless of other entities) |

|

| relationshipName

|

|  |
| --- |
| Entity 2 |

**arity**: range of entities that contribute to the relationship

### e.g.)

|  |
| --- |
| participant |
| Name Address  e-mail |

arity↓ |

1..\* | invitedTo

| Invitation

0..\* | invites

|

## Data Flow Diagrams

**Rectangles**: actors outside of system who either input to or receive output from the system

**Arrows**: direction of flow of information, the description of the information is usually described along the length of the arrow

**Circles**: actions by system

## State Machine Diagram

Arrow:

* [constraint]: necessary input to get to next state
* flow: what the machine is doing

Circles: description of state

All states must go to a termination state!

# DDP

Defect Detection Prevention (DDP):

AHP Comparison Matrix

# Pairwise Comparisons

This is a way of seeing if your values for your AHP matrix are consistent.

**Weights**: measure of importance from 0 to 1



Although the sum of your weights, should equal 1, don’t worry if it doesn’t. Instead, normalize them by dividing them all by the sum of your weights.



*axy*, where *x* is columns and *y* is rows

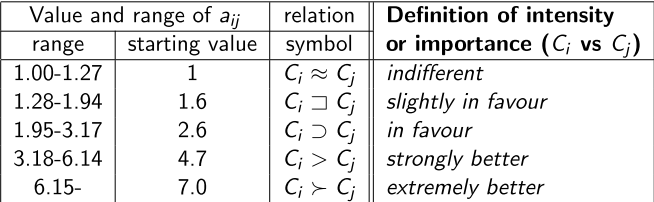
*i*, *j*, and *k* are index variables with a range of the number of elements

a: *aij*

b: *aik*

c: *akj*

**Inconsistency coefficient** [cmA]: 



If the inconsistency coefficient is > 0.3, then you need to tweak your values.