

CMU SCS

**Carnegie Mellon Univ.
Dept. of Computer Science
15-415/615 - DB Applications**

Lecture #19 (not in book)
Database Design Methodology handout

CMU SCS

Based on handout:

Adaptable methodology for database design
by N. Roussopoulos and R.T. Yeh, IEEE
Computer Vol. 17, no. 5, pp. 64-80. 1984

Faloutsos CMU SCS 15-415/615 2

CMU SCS

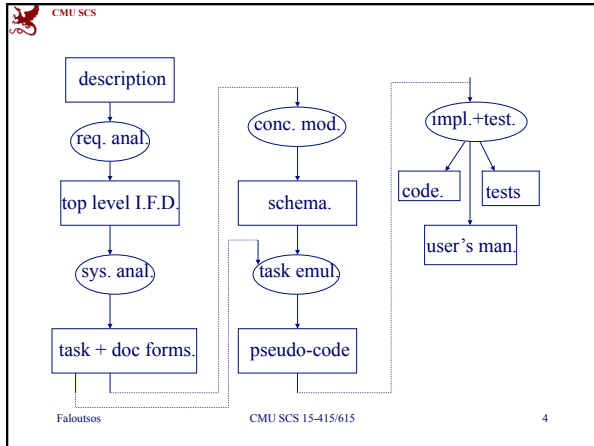
Goal

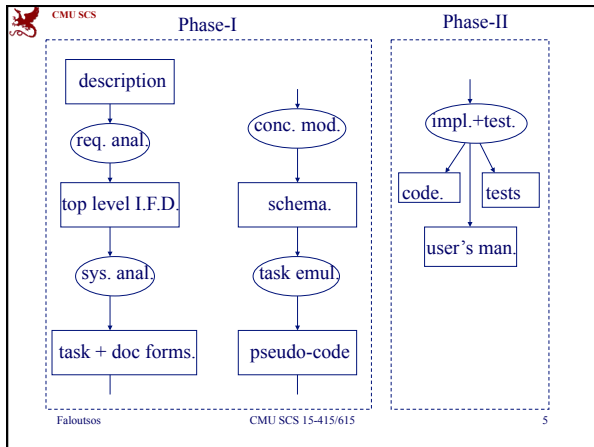
- Given an English description of an enterprise
- build a system to automate it and
- produce the documentation

In diagram form

- tasks ○
- documents □

Faloutsos CMU SCS 15-415/615 3





Running example - 'Mini-U'

- Students register
- Students enroll in courses
- Students ask for transcripts
- Administrator records grades
- Every semester: print class lists

The diagram is labeled 'Faloutsos' and 'CMU SCS 15-415/615' at the bottom, with a page number '6'.

CMU SCS

Requirement analysis

Turn English description in to **top level information flow diagram**, where

- boxes -> documents (~ db tables)
- ovals -> tasks (= db programs)

Important: **system boundary**

Faloutsos CMU SCS 15-415/615 7

CMU SCS

Top level inf. flow diagram

```

graph TD
    A[reg. form] -- input --> B((reg.))
    B -- output --> C[student rec.]
  
```

Faloutsos CMU SCS 15-415/615 8

CMU SCS

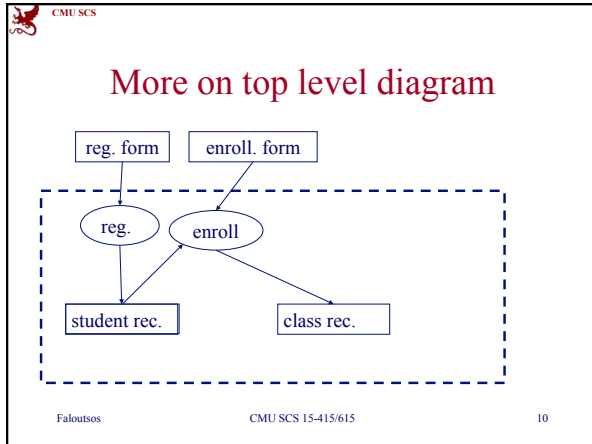
System boundary

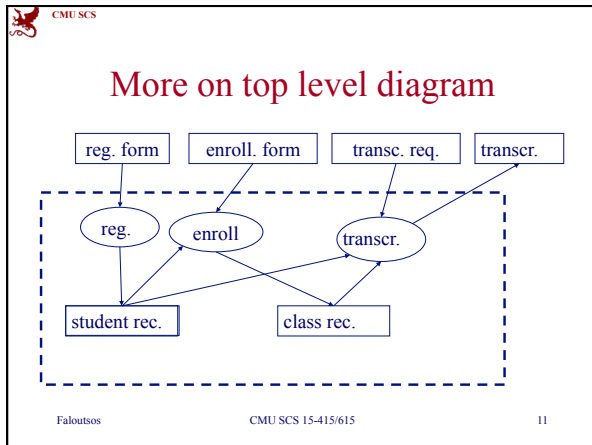
```

graph TD
    A[reg. form] -- input --> B((reg.))
    B -- output --> C[student rec.]
    subgraph SystemBoundary [ ]
        B
        C
    end
  
```

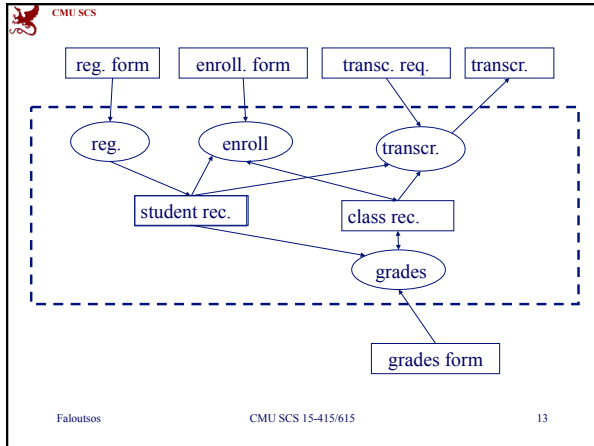
- internal documents -> db tables
- tasks -> db programs
- tasks: internal only

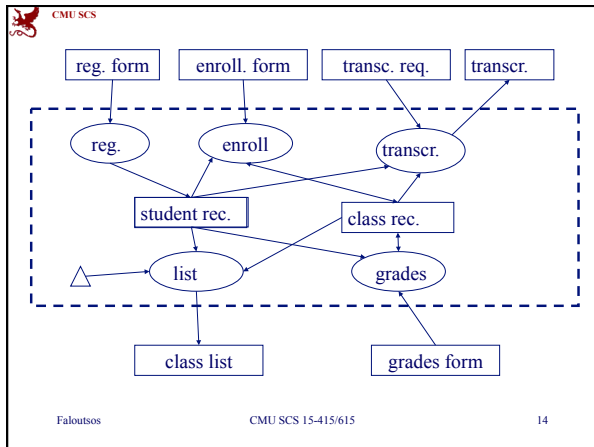
Faloutsos CMU SCS 15-415/615 9

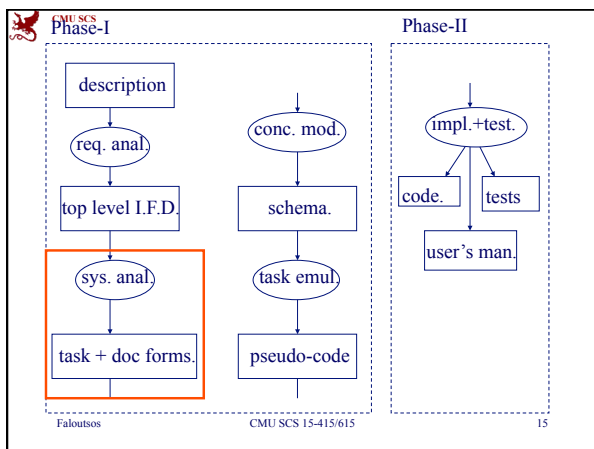




- ### Example - Mini-U
- Students register
 - Students enroll in courses
 - Students ask for transcripts
 - Administrator records grades
 - every semester: print class rosters
- Faloutsos CMU SCS 15-415/615 12







CMU SCS

Document + Task forms

Top level diagram: only half of the info - we also need:

- Document forms and document list
- Task forms and task list

Faloutsos CMU SCS 15-415/615 16

CMU SCS

Document list

- D1: registration form
- D2: enrollment for
- ...
- D7: student record
- D8: class record

} INTERNAL

Faloutsos CMU SCS 15-415/615 17

CMU SCS

Document forms

<ul style="list-style-type: none"> • D1: registration <ul style="list-style-type: none"> - ssn - name - address 	D2: enrollment <ul style="list-style-type: none"> ssn name List-of: <ul style="list-style-type: none"> course id course name
--	---

Faloutsos CMU SCS 15-415/615 18

CMU SCS

Document forms - cont'd

- D3: transcript request form
 - ssn
 - name

D4: transcript

- ssn
- name
- List-of:
 - class-id
 - class name
 - grade

Faloutsos CMU SCS 15-415/615 19

CMU SCS

Document forms - cont'd

(Internal documents - VERY IMPORTANT)

D7: student record

- ssn
- name
- address

Faloutsos CMU SCS 15-415/615 20

CMU SCS

Document forms - cont'd

D8: class record

- class-id
- class-name
- syllabus
- List-of
 - ssn
 - grade

Faloutsos CMU SCS 15-415/615 21

CMU SCS

Document forms - cont'd

- IMPORTANT POINTS
 - avoid redundancy in internal documents: ie., grades should be stored in ONE place only
 - there are many, different, correct solutions

Faloutsos CMU SCS 15-415/615 22

CMU SCS

Task List

- T1: Registration
- T2: Enrollment
- T3: Transcript
- ...

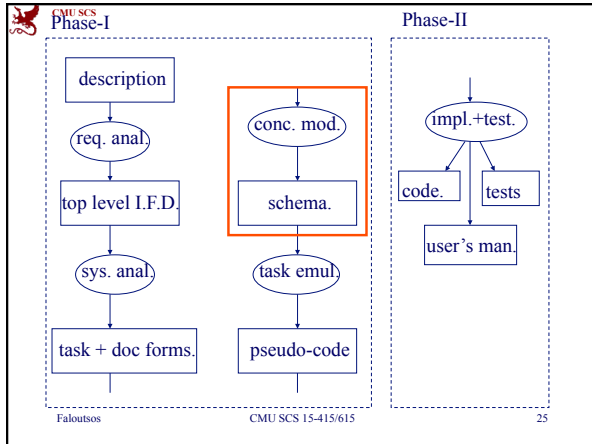
Faloutsos CMU SCS 15-415/615 23

CMU SCS

Task forms

- As in [R+Y]
- not required for this homework
- sub-tasks: probably there won't be any
 - otherwise: ~3-7 sub-tasks per task

Faloutsos CMU SCS 15-415/615 24



Database schema - E-R

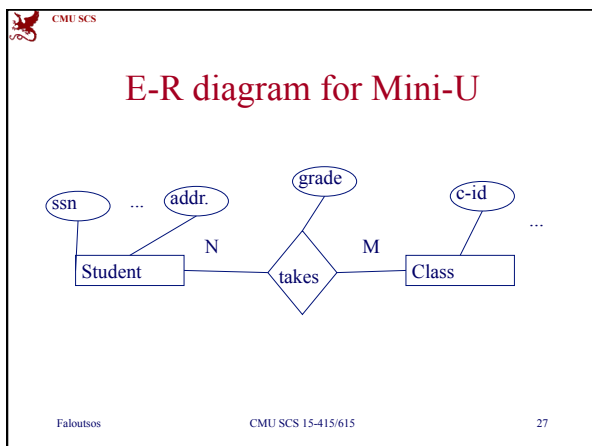
- from the **internal** documents
- use their forms
 - ‘List-of’ constructs -> relationships

Eg., for ‘Mini-U’:

D7: Student record (ssn, name, address)

D8: Class record (c-id, ..., List-of ...)

Faloutsos CMU SCS 15-415/615 26



CMU SCS

Relational schema

student(ssn, name, address)
 class(c-id, c-name, syllabus)
 takes(c-id, ssn, grade)

Make sure that

- Primary keys are underlined;
- tables are in BCNF (or 3NF at worst)

Faloutsos CMU SCS 15-415/615 28

CMU SCS

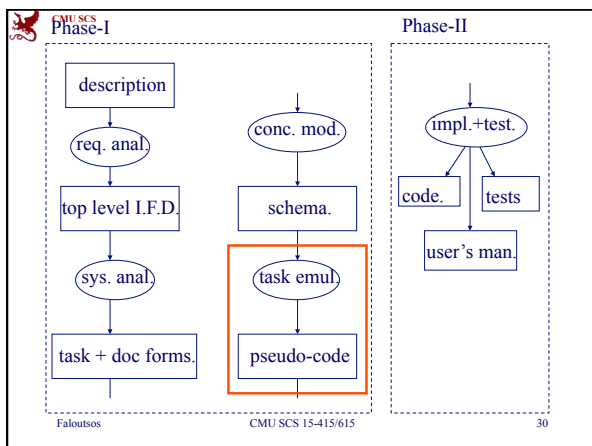
SQL DDL statements

create table student (ssn char(9), ...);

create table class (c-id char(5), ...);

...

Faloutsos CMU SCS 15-415/615 29



CMU SCS

Task emulation

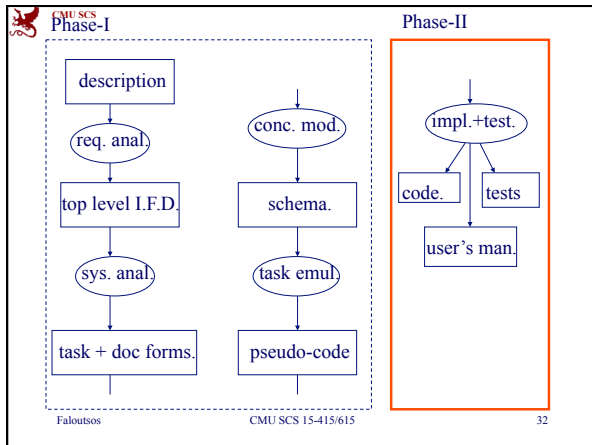
T1: Registration

```

read ssn, name and address
if ( ssn does not exist in 'student'){
  insert into student values ( :ssn, :name, :address);
} else{print "error: duplicate ssn"}

```

Faloutsos CMU SCS 15-415/615 31



CMU SCS

Testing

- For T1 (registration), we check
 - duplicate ssn
 - ssn with 9 digits
- For T2 (enrollment) we check
 - for valid ssn (9 digits)
 - for registered ssn
 - for valid c-id
 - for duplicate (ssn, c-id) entry

Faloutsos CMU SCS 15-415/615 33

CMU SCS

User's manual

Short (~1 page or less) - eg.,:

- copy myproject.tar
- do 'make'
- follow the menu

<anything else the user should know, like OS, space requirements, etc etc>

Faloutsos CMU SCS 15-415/615 34

CMU SCS

Important points for Phase-I

- No redundancy in the fields of internal documents
- don't forget the system boundary
- make sure the top level diagram agrees with the internal document forms
- explain if/when we deviate from BCNF

Faloutsos CMU SCS 15-415/615 35
