Design Interfaces Assignment

COM S 309

THERE ARE THREE PARTS TO THIS DOCUMENT.
PART1: GOAL
PART2: WHAT's AN INTERFACE?
PART3: WHAT DOES EACH TEAM HAVE TO DO?

PART-1: GOAL

Goal

- To describe the interfaces between subsystems.

- **This allows teams to develop code separately, test separately, and integrate easily.**

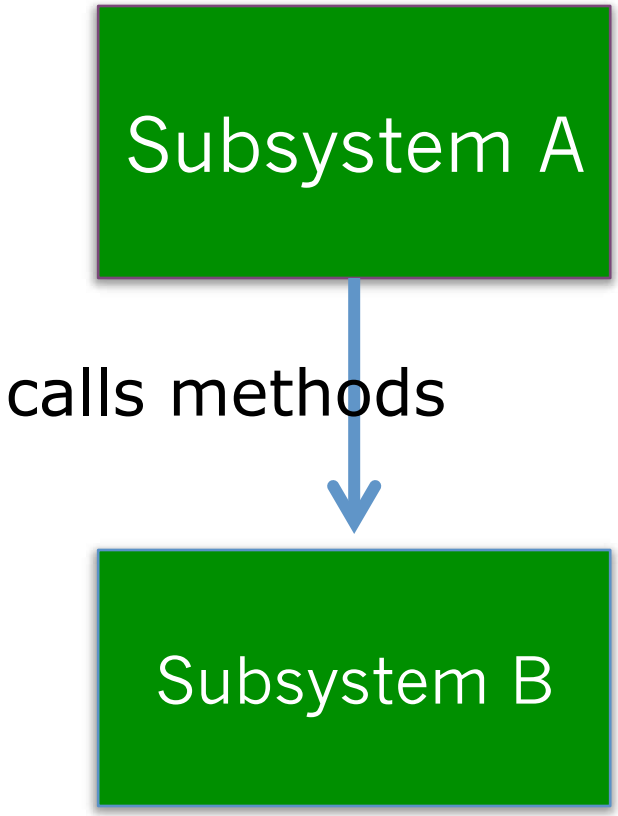
Here, we focus on the **connections or interfaces** between subsystems and components.

PART-2: WHAT IS AN INTERFACE?

What do we mean?

- Here we focus on how different parts of your software connects to each other.
- In the next few slides, we identify at least five different ways in which different parts connect
  1. method calls
  2. event handler registration
  3. writing and reading from same file(s)
  4. database
  5. internet connection and communication
- Also, one needs to focus on how threads and processes work together in the project.

method calls

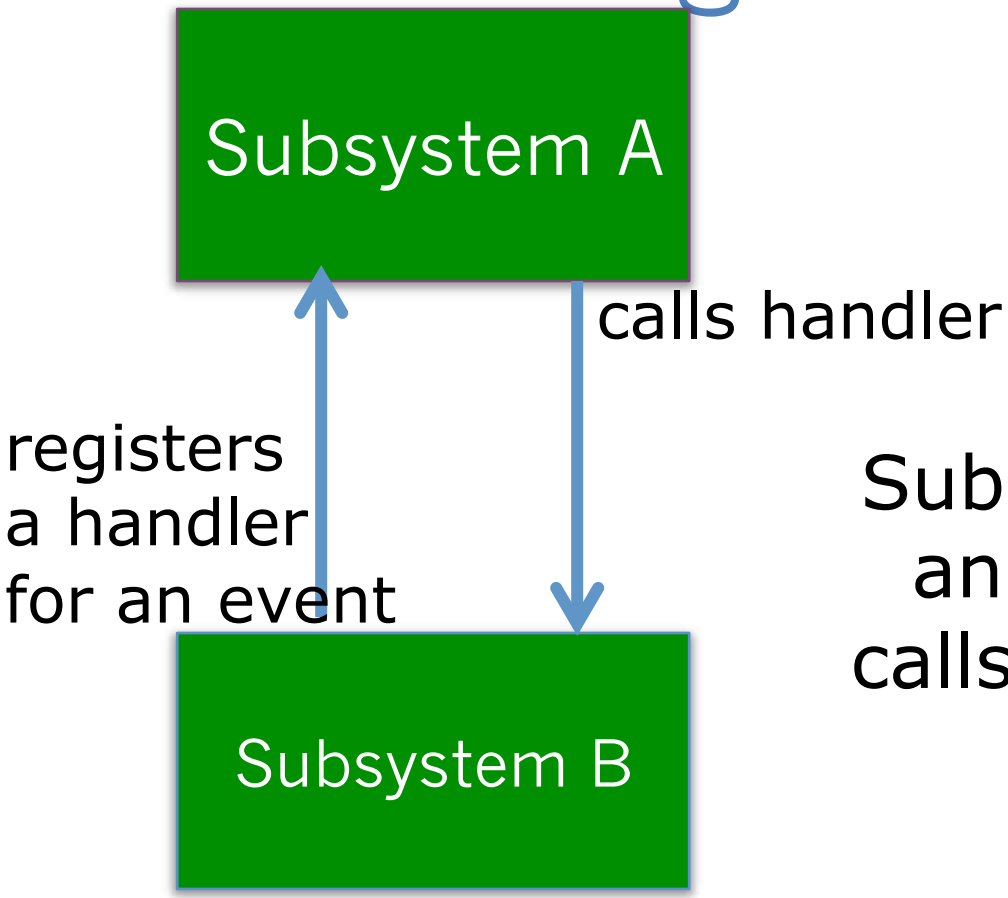


Subsystem A calls services provided by Subsystem B. The "**method signature**" must be clearly specified.

Example of method signature:
String subString(int startIdx, int endIdx)

Note that the service could be a web-service, a remote procedure call, or other server provided services

Event register and handle

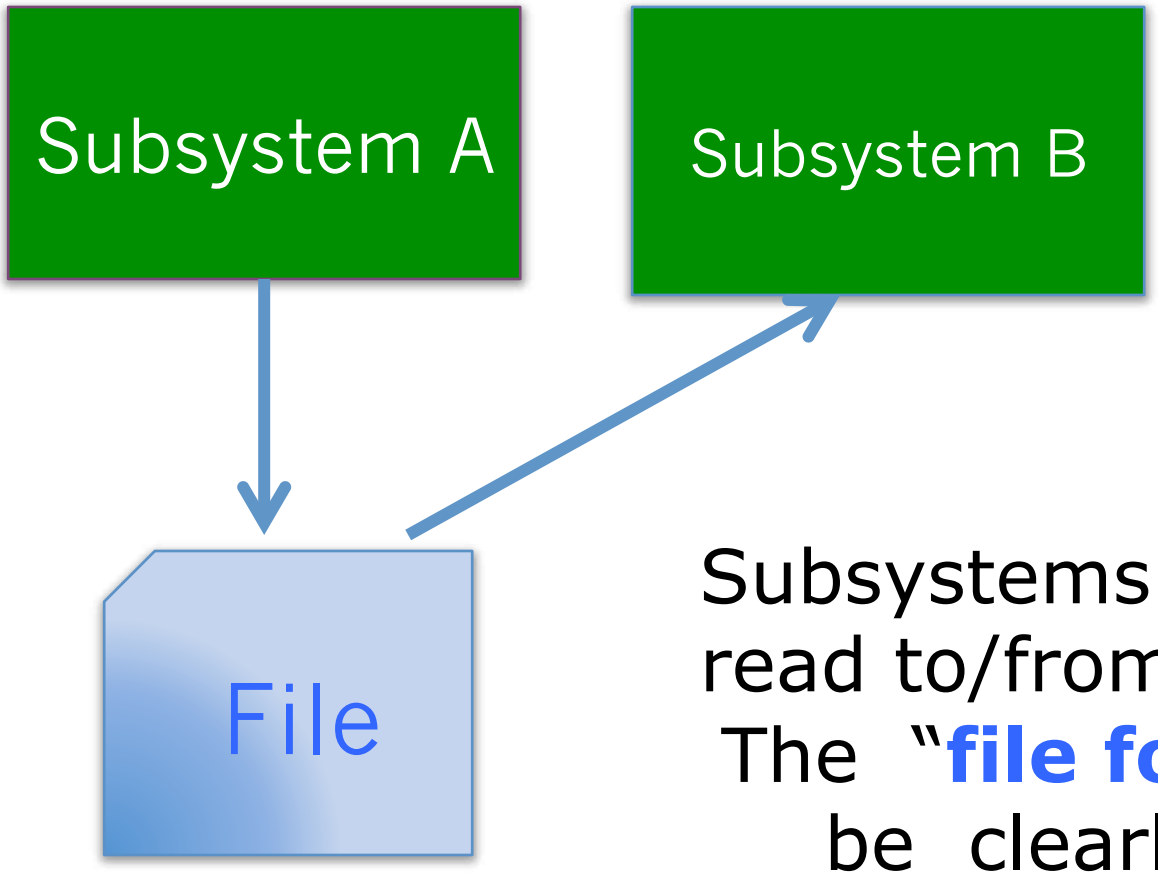


Subsystem B registers for an event. Subsystem A calls the handler when the event occurs.

The events that A offers must be listed.
The events that B subscribes and their actions – must be described.

Example of event registration: registering a Button's action handler.

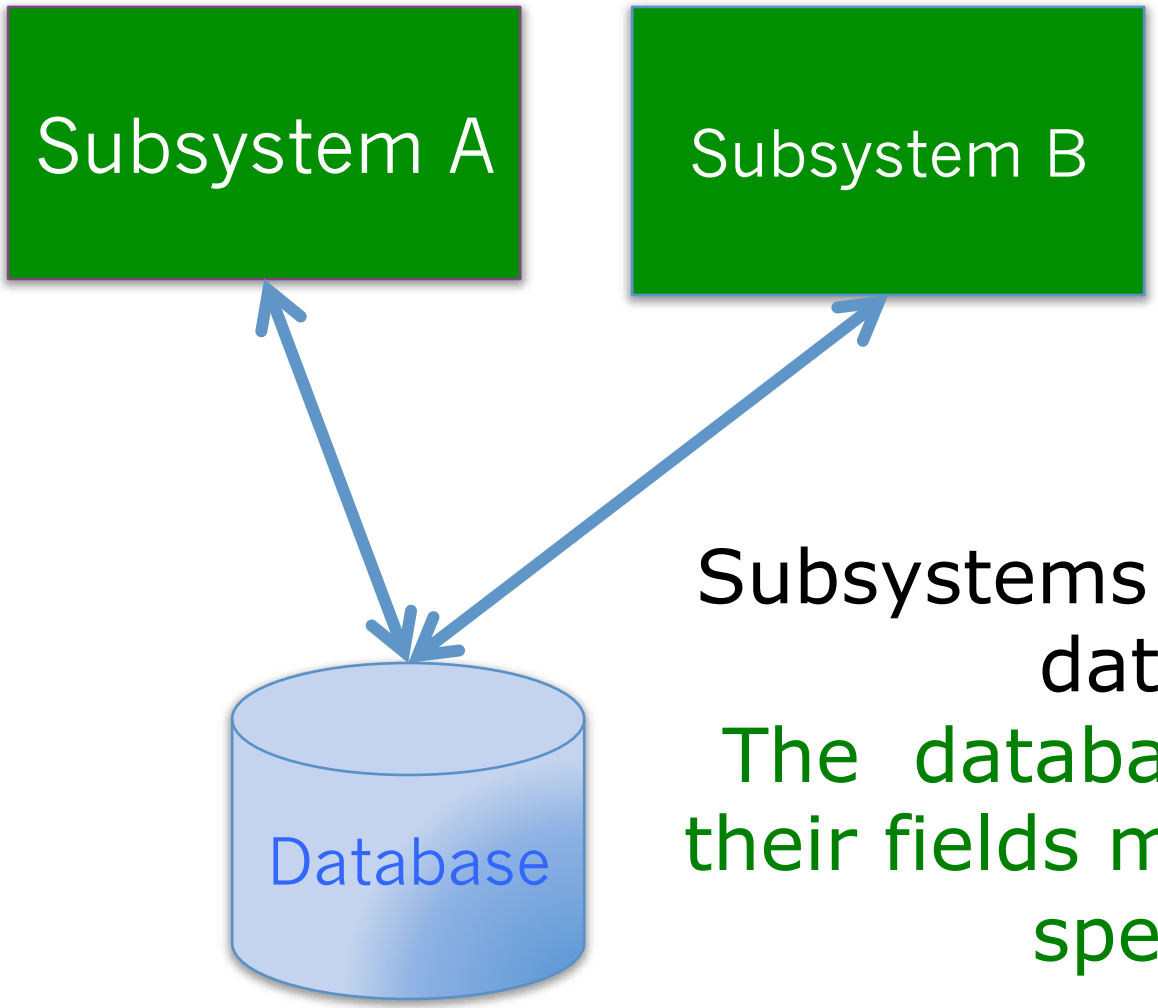
Access the same file



Subsystems A and B write/ read to/from the same file. The "**file format**" must be clearly specified.

Example: if both files use a csv file, then the headers of each column should be specified.

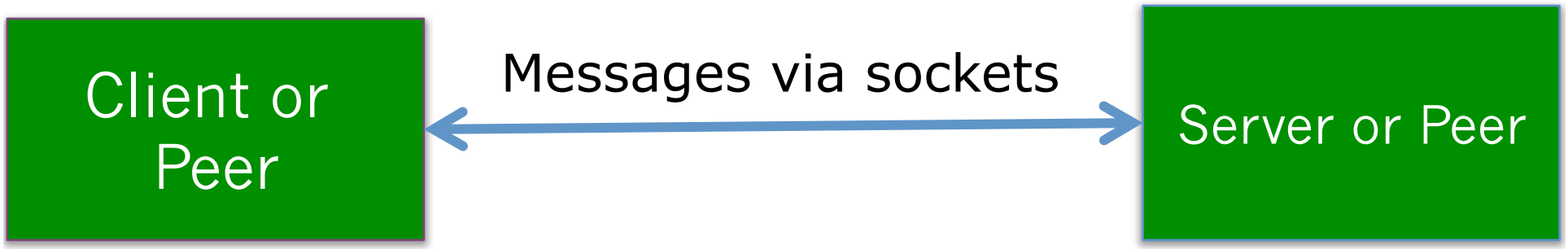
Database



Subsystems A and B use a database

The database tables and their fields must be clearly specified.

Communication



Subsystems A and B communicate via some message protocol

The messages format and protocol must be described.

For example, maybe they use HTTP protocol. Or maybe some custom protocol (like specially formatted strings).

Importance

Developers will develop their parts separately. The separate pieces will need to be combined together to form the system.

Being clear on how the different pieces interact is very important – it allows developers to work independently. It allows for a smooth integration.

PART-3 WHAT YOUR TEAM HAS TO DO

WHAT YOUR TEAM HAS TO DO

1. create a docs folder on your server (proj-309-xx.yy.cs.iastate.edu)
   i.e. in /var/www/html for most teams.

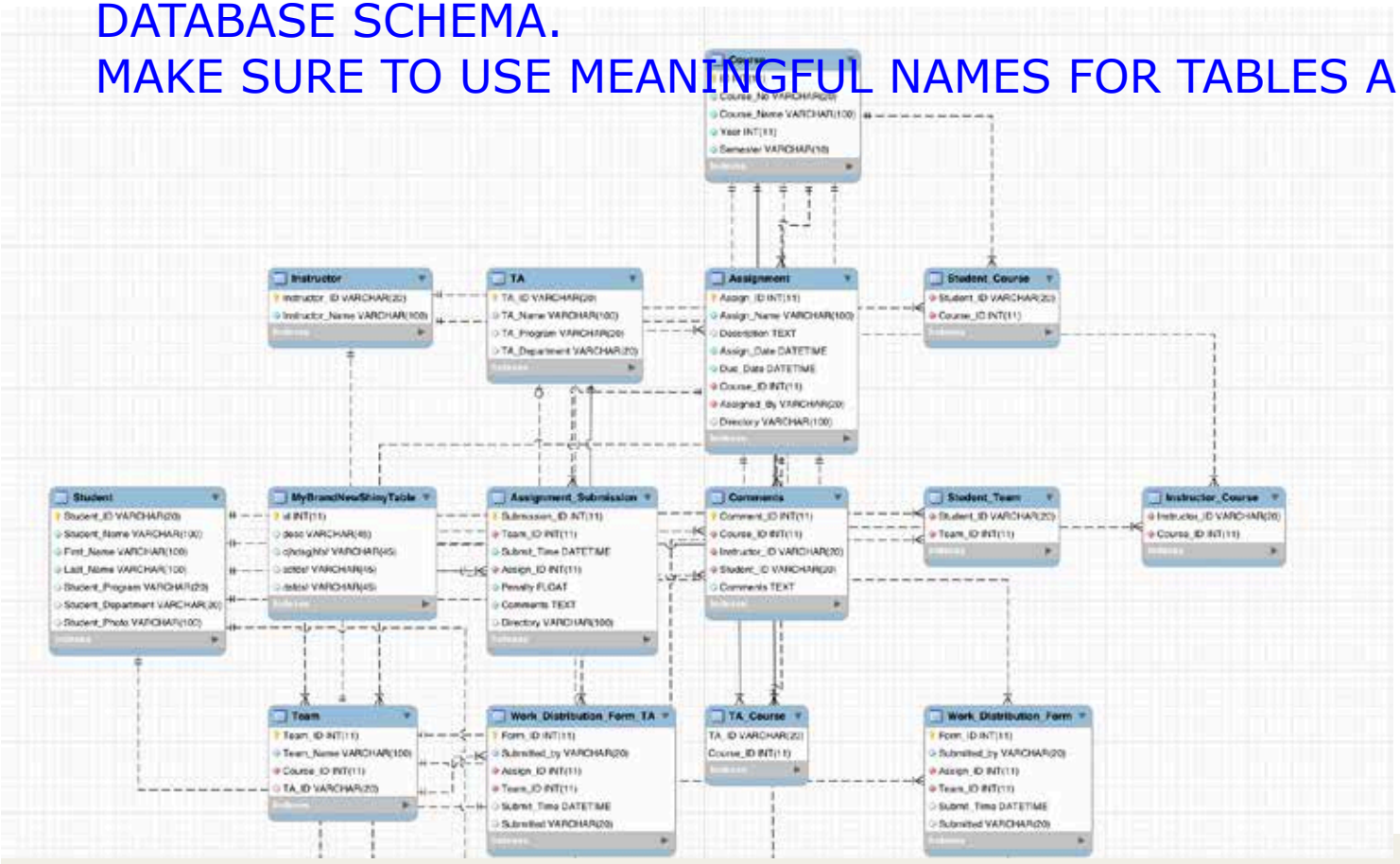
2. create an index.html that looks like

3. You will need to have each team-member create API docs for a different part of the project. Check next TWO pages for examples of database schema and different types of api, and protocol documentation. Make sure to create schema for your database and provide a link to that as well.

4. When your team has finished, submit a BLANK file on BB to indicate that your work is ready to be graded.

EXAMPLE DB SCHEMA

THIS WAS GENERATED BY MYSQLWORKBENCH. CLICK DATABASE AND THEN "REVERSE-ENGINEER". AFTER SEVERAL STEPS IT CREATES A DIAGRAM LIKE THIS. POST AN IMAGE LIKE THIS TO SHOW YOUR DATABASE SCHEMA.
MAKE SURE TO USE MEANINGFUL NAMES FOR TABLES AND FIELDS.



API documentation

- java docs example - sqlrest
- web api example – trello
- web api example - geogig
- php docs example – laravel
- protocol example – http
- js docs example – mozile
- node example – socket.io