Going forward

w41	Rework on the Assignments		
w42	Fall Break	Re-deliver RAD	
w43		[6] SDD	Project
w44		[7] C#	Individual
w45		[8] C#	Individual
w46		[9] Code skeleton	Project
w47		[10] Report on team work	Project

IT University of Copenhagen

Analysis, Design, and Software Architecture (BDSA)
Paolo Tell

RAD - General Feedback

IT University
of Copenhagen

You task!

- all functionality provided to the user in terms of use cases in the sticky man diagram
- 4 non-trivial scenarios
- 4 full non-trivial use case descriptions
- final identified entity, boundary, and control objects
- a complete entity object diagram
- 4 most relevant behavior diagrams, with a minimum of one sequence, one state machine, and one activity diagram

Areas

Description

This should be crystal clear is your mind and also it should become un-ambiguous in chapter 1. You have to put is writing all the application domain knowledge that you need to create the system to verify with the client that you got it right!

- what is study?
- what is a phase?
- what is a task?

Scope

After the first section in chapter 1, you need to start discussing <u>only</u> your sub-system. Use cases or any other analysis related to the other sub-part is irrelevant.

Moreover, you need to describe all the functionalities the sub-system will support.

Consistency

The various parts of the document need to be coherent. For instance:

- you cannot have objects in the sequence diagram that have not been described and classified previously (i.e., entity, control, boundary)
- you cannot have user interface elements supporting functionalities that have not been described in the requirement analysis
- you cannot have detailed use case descriptions of use cases that are not present in the sticky-man diagram.

UML

Part of this course is learning UML. The language must be used correctly. Moreover, I explained in class that some parts of the diagrams might need additional extra text to clarify, for instance, an unusual multiplicity.

Inspirations

- **Yellow team:**
  - state diagram of study;
  - state diagram of phase;
  - use case of study creation including details on phase creation, visibility of data, inclusion and exclusion criteria... (if this becomes too complex, potentially split in several use cases and use <<include>>);
  - wireframe presenting a study/phase creation;
  - sequence diagram detailing how incoming task requests through the API are handled and handed out. Who gets which task of what type for what reason?

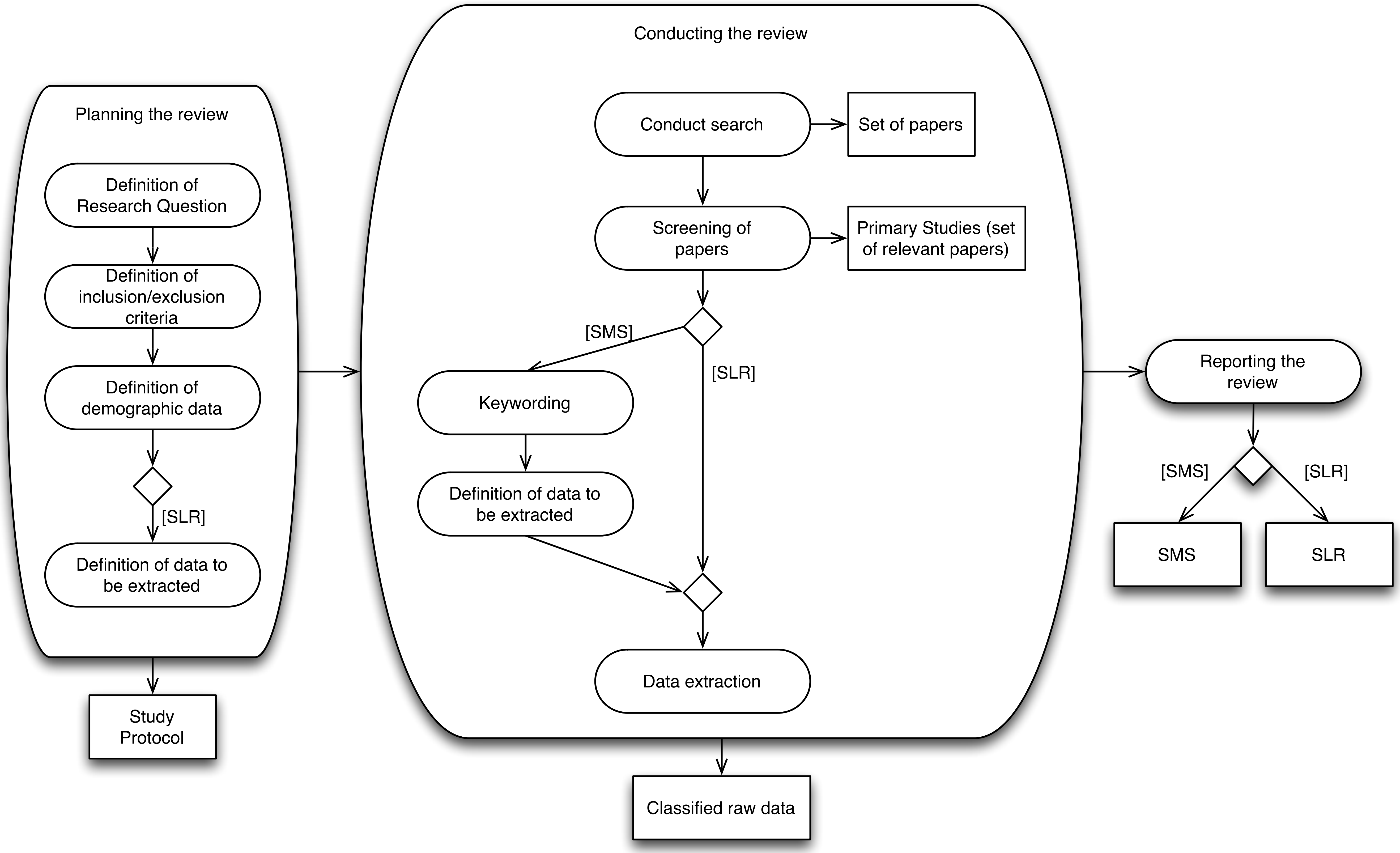
- **Blue team:**
  - sequence diagram detailing how tasks are requested from the APIs and how they are executed (data entry), and afterwards submitted for completion (or cached?);
  - activity diagram showing the workflow used to present tasks to the users;
  - a state diagram of a task. How does its state change as work is done on them?
  - wireframe on how different data types are visualized;
  - wireframe presenting the visible data fields, and the requested fields as part of a task requesting to fill out data;
  - wireframe presenting conflicting data as part of a task to handle conflicting data fields, as filled out by separate users.

From the project description

IT University of Copenhagen

Tasks

- The intricate details of which tasks are handed out to which user, as part of which stage, are left to the server, and does not involve the client. However, a task is defined by:
  - unique ID;
  - a set of <u>visible data fields</u> shown as part of the task, which cannot be
  - modified;
  - a set of <u>requested data fields</u> shown as part of the task, which can be
  - modified;
  - a type either:
    - (1) a request to <u>fill out data</u> field(s)
    - (2) a request to <u>handle conflicting data</u> field(s)


Planning the review
Definition of Research Question
Definition of inclusion/exclusion criteria
Definition of demographic data
[SLR]
Definition of data to be extracted
Study Protocol
Conducting the review
Conduct search
Set of papers
Screening of papers
Primary Studies (set of relevant papers)
[SMS]
[SLR]
Keywording
Definition of data to be extracted
Data extraction
Classified raw data
Reporting the review
[SMS]
[SLR]
SMS
SLR

IT University of Copenhagen

APIs

- CRUD user/team operations. (CreateTeam, AddUserToTeam, . . .): allow retrieving, creating, removing, updating, and deleting users, teams, and their associations. A user can be part of multiple teams. Teams and users have a name.
- GetStudies(userID): retrieves all the studyIDs a user belongs to.
- GetTasks(studyID, userID, count, type=any): retrieves a certain amount of tasks which need to be performed by the specified user for the specified study. Tasks are returned in the order in which they are recommended to be performed.
- GetStudyOverview(studyID): reports the different stages in the study, and per stage, for each user the amount of tasks done, out of all the known tasks to be done.
- DeliverTask(studyID, userID, taskID, modifiedFields): deliver a finished task, including the resulting modified fields. This should return whether or not the task was delivered (e.g., can still be delivered) successfully. This can be called several times for the same task, in which case the latest value is used (if the task is still editable, which is decided by the server).
- GetResource(id): retrieves a specified resource (e.g., PDF file).
- GetReviewableTaskIDs(userID, studyID): retrieves all task IDs of tasks which have already been delivered, and can still be edited. This can be used to clear the cache of tasks which can still be edited within the client.
- GetReviewableTasks(userID, studyID): retrieves all tasks which have already been delivered, and can still be edited. This allows editing already delivered tasks.