Indian Institute of Technology, Kharagpur

Department of Computer Science and Engineering

Software Engineering (CS 20006), Spring 2015-16

Story Management System (PMS)

Req. Spec., and Outline of Analysis, and Design

---

1. A Newspaper House publishes daily newspaper and wants to manage its activities through a **Story Management System (SMS)**. You are to develop the **SMS**. The requirements specification of the system is given below. Read the specifications carefully, analyse the requirements, and design the following aspects of the system using UML and DP.

   (a) Identify the use-cases and design suitable Use-Case Diagrams for **SMS**. Highlight the relationships amongst the actors and the use-cases.

   (b) Design Class Diagrams for *Story*, detailing the attributes and operations with their properties.

   (c) Show the State-chart Diagram for a *Story* as it passes through the **SMS**.

   (d) Show all other classes / objects (in addition to Question 1b) by brief Class Diagrams (with name and key attributes). For the entire collection of classes (that is, including *Story*) show the associations, aggregations/compositions, generalization/specialization, and abstract/concrete etc.

   (e) Design suitable Sequence Diagrams for use-cases arising from *Submit* (of *Reporter*), *Review* (of *Manager*), *Revise* (of *Reporter*), and *Approve* (of *Manager*) actions.

   (f) Choose appropriate Design Patterns for your design. Briefly justify your choice.

Requirements Specification for Story Management System (SMS)

(a) The staff structure of the Newspaper House is as follows:

- *Editor*. The *Editor* is responsible for the overall activities and directly manages the *Editorial Division*.
- *Associate Editor*s. Every *Associate Editor* is responsible for a *Division* and reports to the *Editor*. No *Associate Editor* manages more than one *Division*.
- *Reporter*s. Every *Reporter* works for a *Division* and reports to the corresponding *Associate Editor*. *Reporter*s working for the *Editorial Division* reports directly to the *Editor*.

Every employee is identified by the *Employee Code*, and has *Name*, *Email* and *Mobile Number*.

(b) The Newspaper House has 3 *Divisions*:

- *Editorial Division*: This publishes the *Editorial* expressing the views of the Newspaper House, *Special News Items* and the *Letters from Readers*.
- *News Division*: This publishes stories on national and international news. A story here is political, social or economic in nature.
- *Features Division*: This publishes national and international feature stories in art, culture, cinema, sports, and the like. A story here is an entertainment event report, critique, celebrity interview, match report, team analysis, or statistics.

Every *Division* has a *Manager*. With the exception of the *Editorial Division*, every *Division* is managed by an *Associate Editor*. The *Editorial Division* is managed directly by the *Editor*.

(c) At the Newspaper House a *Reporter* needs to:

- *Collect*: A reporter goes to places or liaison with external agencies to collect news items.
- *Compose*: A collected news item is cast in the form of a story.

- *Submit*: A completed story is submitted to the corresponding *Manager*.
- *Revise*: Up on review, if the *Manager* desires, the story is revised and re-submitted.

(d) At the Newspaper House the responsibilities of an *Associate Editor* include all responsibilities of a *Reporter*. Naturally she / he can report their own stories. In addition, an *Associate Editor* needs to:

- *Review*: A story submitted by a reporter (of the *Division*) needs to reviewed and edited. Up on review, the *Associate Editor* may request the *Reporter* to revise and re-submit.
- *Approve*: A submitted story may and may not be approved – with or without revision.
- *Paginate*: Compose the day's page/s with the approved stories of the *Division*.

(e) At the Newspaper House the responsibilities of the *Editor* include all responsibilities of an *Associate Editor* (and hence those of a *Reporter*). In addition, the *Editor* needs to:

- *Edit*: Manage the *Editorial Division*, write the editorial, and set & comply with the policies for the Newspaper House.

(f) A *Story*:

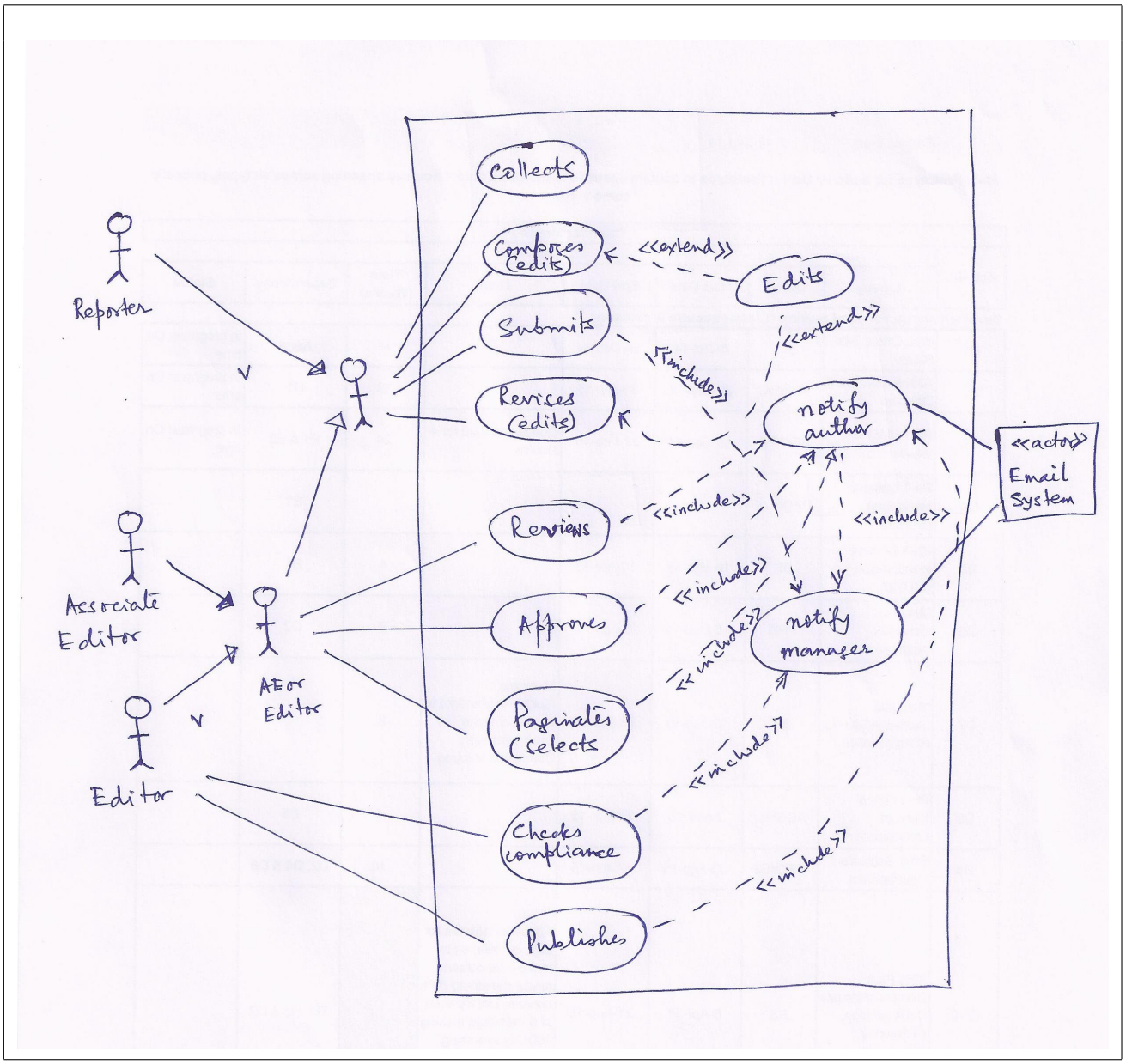
- Is a piece of text for publication in the newspaper.
- Has title, place, date-time, sources (optional), and reporter / associate editor (optional).
- Is of a type that matches the *Division* in which it is published.
- Has a nature as specified above under different *Division*s.

(g) The *Work flow* in the Newspaper House is as follows:

- A *Reporter* collects a news item from primary lead, secondary agency or continuity of events. She / he explores the details and prepares the facts.
- The *Reporter* then composes the facts in terms of a *Story* filling in the necessary and auxiliary parts.
- Once composed, the *Reporter* submits the *Story* for review.
- The *Manager* of the *Division* retrieves the submitted *Story* and takes one of the actions as follows:
  - Review & edit and approve the *Story* for publication.
  - File review comments on the *Story* for the *Reporter* (who wrote the *Story*) to make revisions. The *Reporter* then revises the *Story* and submits again for review.
  - Review and disapprove the *Story*. This *Story* will not be published.
- Once the cut-off time for the day is over, the *Associate Editor* of the *Division* will preview the approved stories and prepare the page/s for the *Division*. Stories selected for a day during pagination are marked published and will not be selected again. Other stories continue to remain in **SMS** for possible publication in future.
- The *Editor* reviews the page/s for compliance to the policies of Newspaper House. If a *Story* is found to be non-compliant, the *Editor* may ask the corresponding *Associate Editor* to revise or replace the *Story*.
- Once all stories become compliant, the *Editor* adds the *Editorial* and orders publication.
- The newspaper goes to press.

Every action in **SMS** generates notification (by email) to all concerned stakeholders.

Part 1a: The Use-cases of **SMS** are:



Note:

- We have grouped the use-cases based on the actors. Hence union of Associate Editor and Editor, and union of Reporter, Associate Editor and Editor are shown as two actors.
- We assume that whenever an author (typically, Reporter) is notified, her / his Manager will also be notified. Hence their <<include>> relationship.
- Compose and Revise use-cases imply editing. Hence Edit use-case has an <<extend>> relationship with them.
- Implied use-cases like management of collection of stories in a story-board etc, have been skipped.
- Use-cases related to Systems and Administration have been ignored.

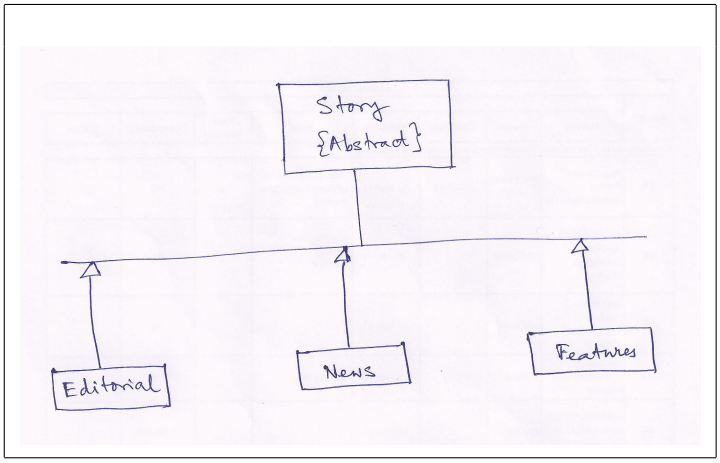
Part 1b: Class diagram for ***Story***:

	Story {*Abstract*}			*Remarks*
–	id	:	String	Auto-generated
–	title	:	String	Set by Create, Change by EditAndSubmit
–	place	:	String	Set by Create, Change by EditAndSubmit
–	date	:	Date	Set by Create
–	time	:	Time	Set by Create
–	source	:	String {optional}	Set by Create, Change by EditAndSubmit
–	author	:	String {optional}	Set by Create
–	body	:	Text	Set to null by Create, Change by EditAndSubmit
–	comments	:	Text	Set to null by Create, Change by ReviewAndApprove
#	division	:	Division	Set by Create
#	/type	:	String	Set by Create – derived from division
#	nature	:	String	Set by Create, Change by ReviewAndApprove
–	status	:	Bool[5]	Set 5 flags to False by Create. Changes by state-chart
–	numberOfApprovedStories	:	Int	
–	numberOfPublishedStories	:	Int	
+	Create(title: String, place: String, date:Date, time: Time, division: Division, nature: String source: String = "", author: Employee = 0): Story *			Create factory for Story
+	Display(employee: Employee): void			Display all fields
+	EditAndSubmit(author: Employee): void			Edit & save-as-draft or Edit & submit
+	ReviewAndApprove(reviewer: Manager): void			Review, Populate comments & Approve or Disapprove
+	Select(manager: Manager): void			Select a story for pagination
+	Publish(editor: Editor): void			Publish or Reject the story

The status values are:

Index	**Status**	**Remarks**
0	draft	If the story is being edited
1	submitted	If the story is ready for review, selection, publication
2	approved	If the story has been approved
3	selected	If the story is selected for pagination
4	published	If the story is published

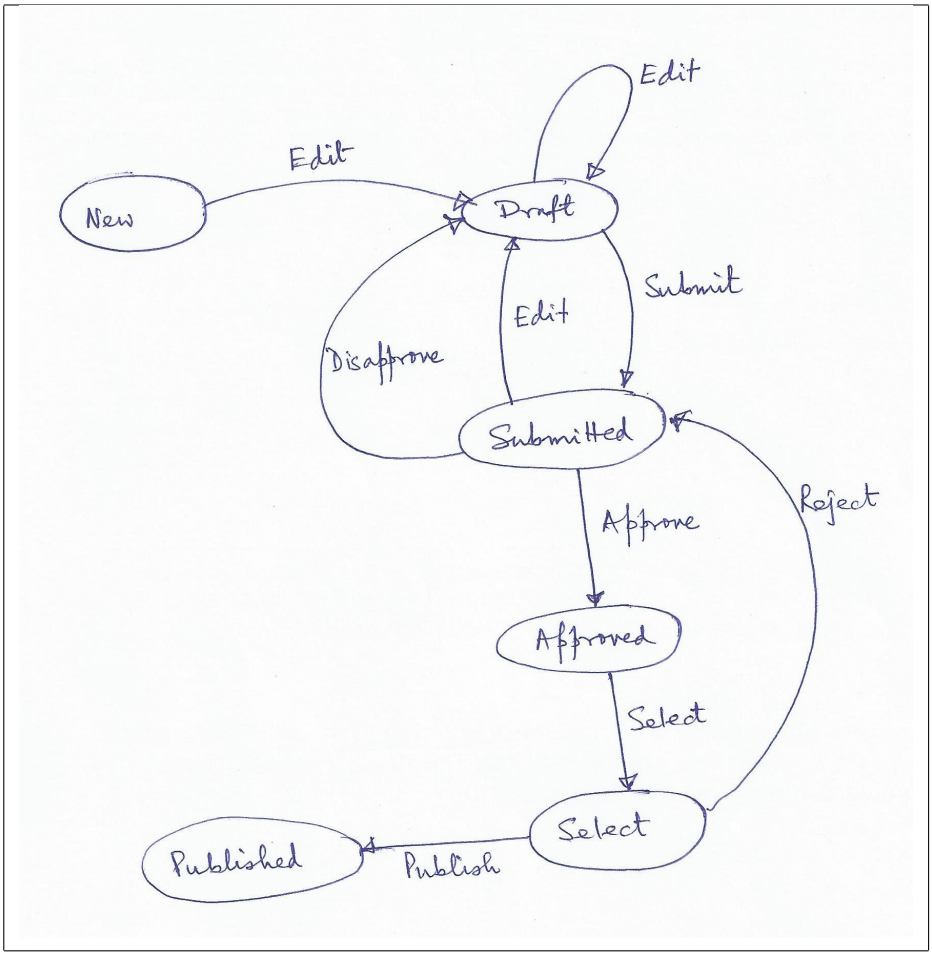
There are three specializations – **Editorial**, **News**, and **Feature** – of ***Story***. These are concrete classes.



The division and nature of a story is set at the time of Create() as:

Specialization	**Division**	**Nature**
Editorial	Editorial	Editorial, SpecialNews, or LettersFromReaders
News	News	Political, Social, or Economic
Feature	Feature	EventReport, Critique, CelebrityInterview, MatchReport, TeamAnalysis, or Statistics

Part 1c: The story has the following states that change according to the state-chart:

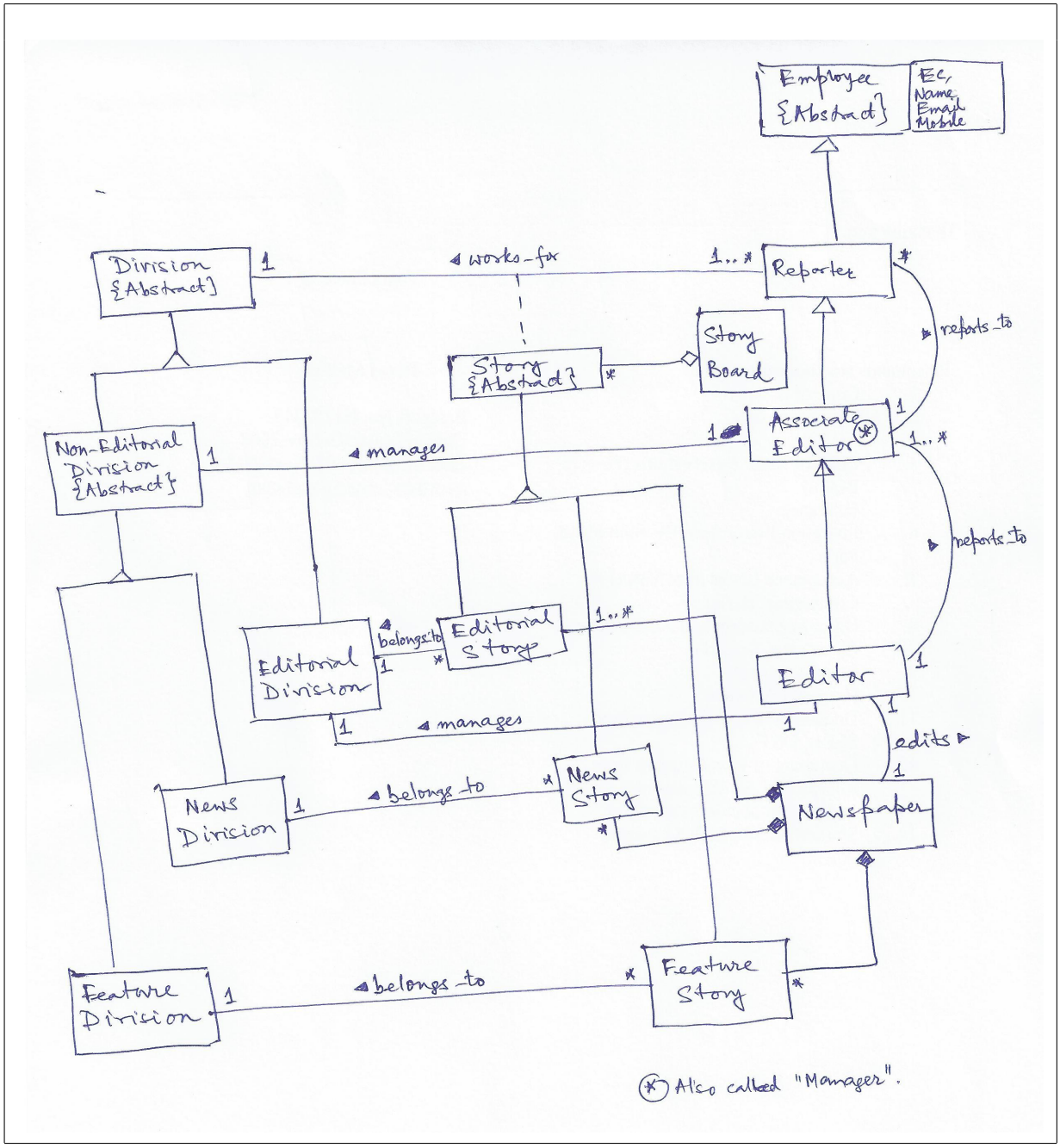


Present State	Action	Next State	Actor	Status					Method
				draft	submitted	approved	selected	published	
null	Create	New	Author	False	False	False	False	False	Create()
New	Edit	Draft	Author	True	False	False	False	False	EditAndSubmit()
Draft	Edit	Draft	Author	True	False	False	False	False	EditAndSubmit()
Draft	Submit	Submitted	Author	False	True	False	False	False	EditAndSubmit()
Submitted	Edit	Draft	Author	True	False	False	False	False	EditAndSubmit()
Submitted	Disapprove	Draft	Manager	True	False	False	False	False	ReviewAndApprove()
Submitted	Approve	Approved	Manager	False	True	True	False	False	ReviewAndApprove()
Approved	Select	Selected	Manager	False	True	True	True	False	Select()
Selected	Publish	Published	The Editor	False	True	True	True	True	Publish()
Selected	Publish	Submitted	The Editor	False	True	False	False	False	Publish()

Note:

- The state of a Story is implicitly managed by the Bool status vector. The above table illustrates the relationship.
- We have not shown the Discarded or Deleted states for a story that does not eventually get published. The same may be included. Of course, this will need one more Bool status flag.

Part 1d: The Class diagram for **SMS**:

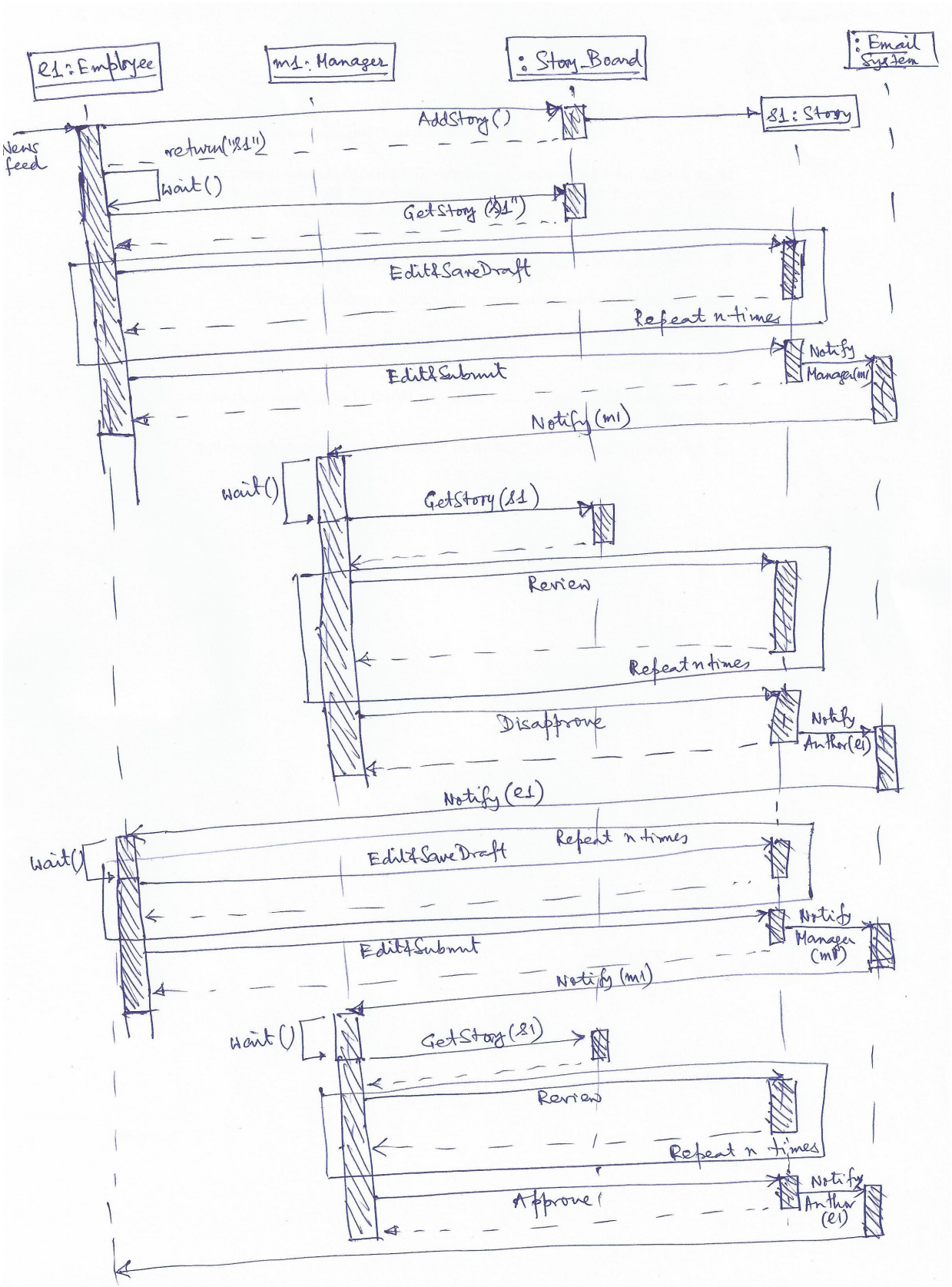


Note:

- An abstract class to describe ***Non-Editorial Division*** has been introduced to easily model the fact that all divisions except the **Editorial Division** is managed by an **Associate Editor** while the **Editorial Division** is managed by the **Editor**.
- The **Story-Board** class is a mere collection of story objects for maintenance. It is a container that has references to the objects, but does not own them. Hence the use of weak aggregation. In contrast, **Newspaper** is a strong aggregation of ***Story*** objects because it *carries* them.
- There could have been a ***Manager*** class between **Reporter** and **Associate Editor** classes on the hierarchy to manage ***Non-Editorial Division***. We have merged it with **Associate Editor**.

- The ***Story*** is shown as an association class between ***Story*** and **Reporter** because only a reporter working for a division can do a story for that division.
- We could also show a subset relation between works_for and manages (of Associate Editor) because to manage a division an Associate Editor needs to work for it. Note that the same holds trivially for Editor and Editorial Division.
- works_for may be shown as an association between ***Division*** and ***Employee***.

Part 1e:


e1: Employee
m1: Manager
: Story_Board
: Email System
s1: Story
News feed
AddStory()
return("s1")
wait()
GetStory("s1")
Edit&SaveDraft
Repeat n-times
Edit&Submit
Notify Manager(m1)
Notify(m1)
wait()
GetStory(s1)
Review
Repeat n times
Disapprove
Notify Author(e1)
Notify(e1)
wait()
Edit&SaveDraft
Repeat n times
Edit&Submit
Notify Manager (m1)
Notify(m1)
wait()
GetStory(s1)
Review
Repeat n times
Approve
Notify Author (e1)

Note:

- We have arbitrarily captured an instance with author / reporter *e1* and her manager *m1*. *e1* writes a story *s1* that *m1* reviews.
- *wait()* is assumed to depict arbitrary delay in synchronization.
- **Story-Board** is a singleton collection of all unpublished story objects. Hence it does not need a name.

Part 1f:

Situation	Pattern	Marks
Generate story objects of appropriate specialization.	Abstract Factory, Factory Method	0 mark – Not covered in class
Browse various lists of story objects, reporters, pages etc., Display collection of story objects.	Iterator	2 marks
There is only one editor and only one story-board for the Newspaper House. Each division is also unique.	Singleton	1 mark
Manager asks Reporter to Revise story, Editor asks Associate Editor to Revise or Replace Story. Asking for the action and the execution of the action are separated in time as is needed in Command Pattern.	Command Pattern	2 marks