CORSO IS1 - 2002/03
MSMILE Requirement Specification
Version 1.0

Gianna Reggio
DISI-Università di Genova, Italy
reggio@disi.unige.it

The Smiling Meeting System "M:-)"

We have to develop a system to support the happening of the meeting of the various committees inside some organization.

Each person belonging to the considered organization is classified by a unique qualification, which may change along the time.

Each committee is characterized by a chairman (whose possible qualification is given) and by a list of members (whose possible qualifications are given).

New committee may be introduced, and existing one may be discontinued.

The chairman and the members of the various committee may change.

The chairman of a committee may schedule a meeting of it, by selecting the time, preparing an agenda, and inviting the members to participate.

An agenda consists of a, possibly empty, list of communications, a non-empty list of items to be discussed, a perhaps some minutes of previous meetings to be approved.

The proceeding of a meeting is as follow. First the chairman declares the meeting open, and determines the actual participants. Checks if the meeting is valid (each committee has its own rule to determine valid meetings); if not the meeting is declared null and rescheduled to another date and time.

The communications are presented by the chairman. Participants to the meeting can briefly discuss them; the most important reactions will be reported in the minutes.

Also the various items will be presented by the chairman, participants may discuss them; after discussion, the chairman proposes a resolution to be reported in the minutes and to be voted. Each item is characterized by the kind of majority need for approving it.

A special discussion item is the so called "other", where participants may propose (usually not very relevant) issues to be discussed, then the result of the discussion will be added to the minutes and voted.

The minutes of previous meeting will be proposed by the chairman, participant may suggest modifications, then the possibly revised version will be voted (each committee defines the kind of majority needed to approve the minutes of the meetings).

When all issues on the agenda have been settled, the chairman declare the meeting closed, and the minutes will be made available to all the members of the committee.

The supporting system should allow to keep the meetings of the various committees electronically (i.e., where the partici[pants are connected by a computer network), to preserve the information about the various committees (chairman, members) and meetings (agendas and minutes) and to allow to the persons belonging to the organization to examine such stuff (visibility of the material related to some committee may be restricted to a given list of qualifications).

M:-) Requirement Specification

Use Case View

Use Case Diagram

P: Person — Browse

Summary level use cases

MAN:Manager — Handle Committee, Handle Person — E:Email

CHAIR: Person — Run Meeting

1..* PTC: Person

User level use cases: Run Meeting

Run Meeting — Start Meeting, Schedule Meeting, Handle Minutes Approval, Handle Discussion Item, Handle Communication, Close Meeting — E:Email

CHAIR: Person

1..* PTC: Person
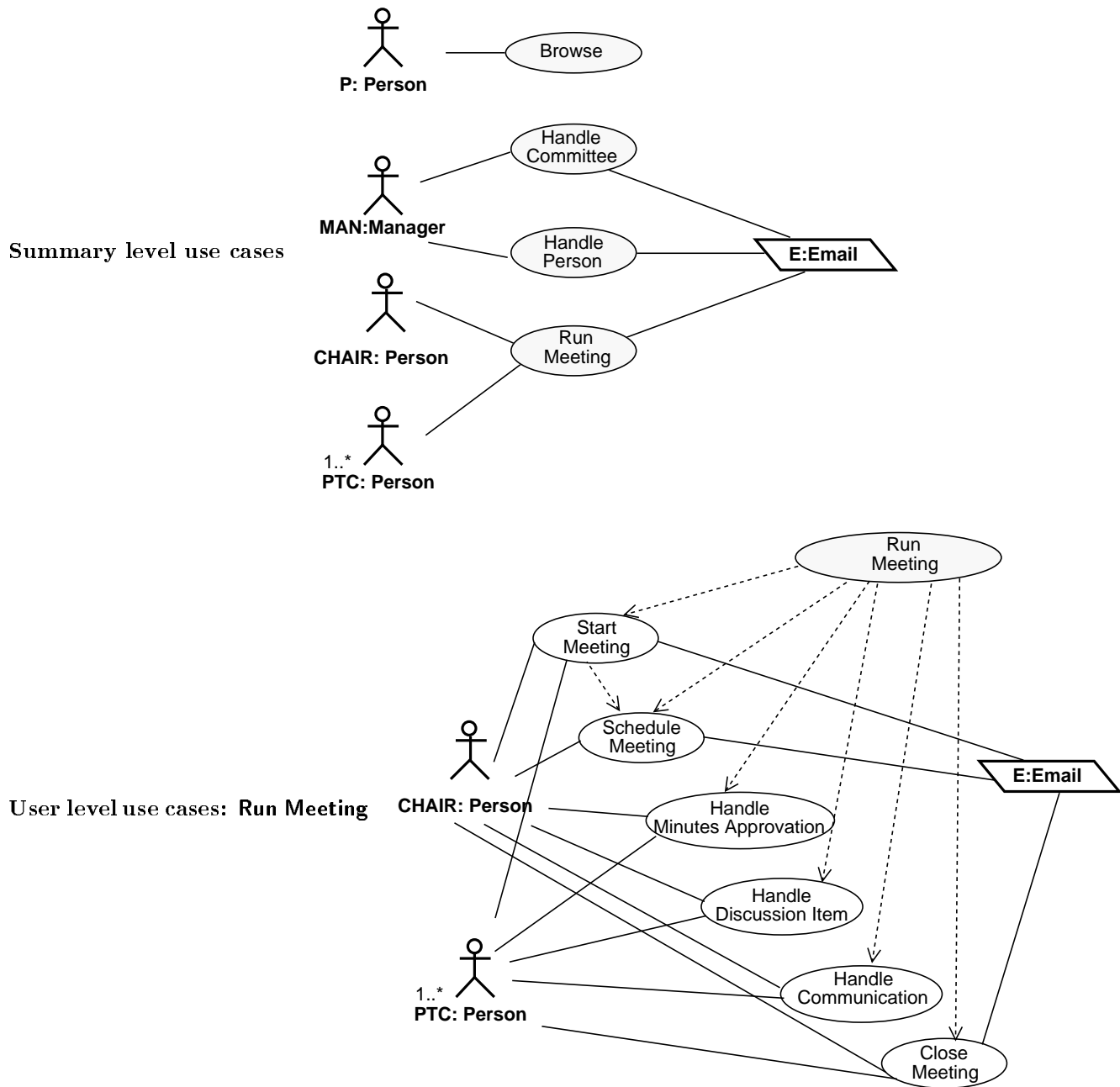
User level use cases: Handle Committee

Handle Committee

Add Committee

Update Committee

Delete Committee

E:Email

MAN:Manager
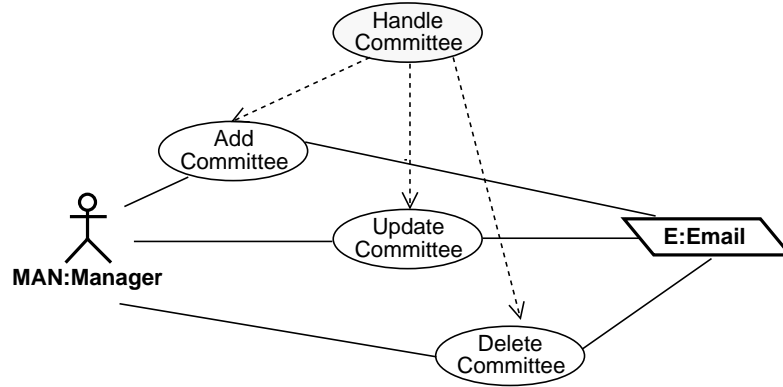
User level use cases: Handle Person

Handle Person

Add Person

Update Person

Delete Person

E:Email

MAN:Manager
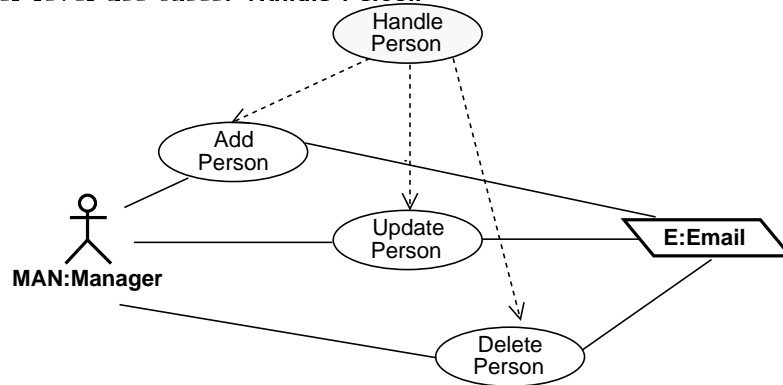
use case Schedule Meeting

textual The chairman of an existing committee may schedule a meeting of the same by fixing the time and the agenda.

If the proposed meeting is conflicting with other meetings, then the system asks for a new time, otherwise the system informs all the members of the committee about the forthcoming meeting.
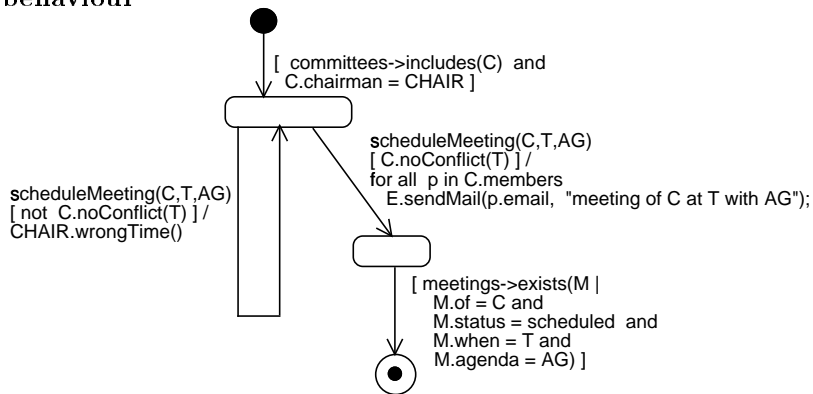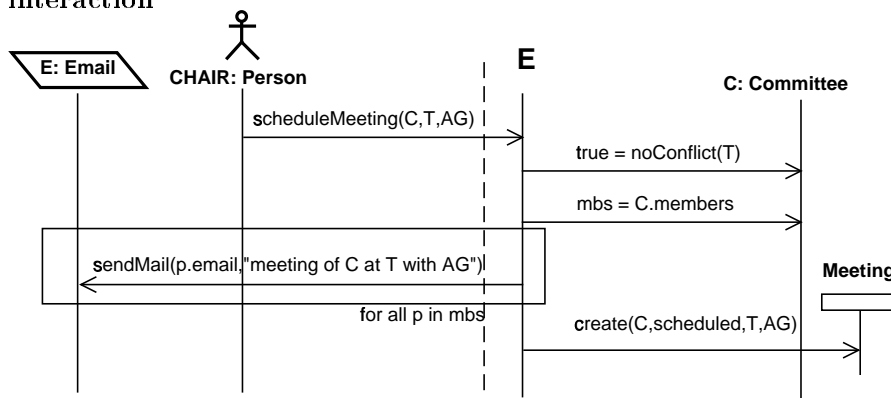
Then, the meeting is scheduled.

behaviour

```
                              ●
                              │  [ committees->includes(C)  and
                              ↓      C.chairman = CHAIR ]
                    ┌──────────────────┐
                    └──────────────────┘
                              scheduleMeeting(C,T,AG)
                              [ C.noConflict(T) ] /
                              for all  p in C.members
                                E.sendMail(p.email,  "meeting of C at T with AG");
scheduleMeeting(C,T,AG)
[ not  C.noConflict(T) ] /        ┌──────────┐
CHAIR.wrongTime()                 └──────────┘
                                        │  [ meetings->exists(M |
                                        │      M.of = C and
                                        │      M.status = scheduled  and
                                        ↓      M.when = T and
                                      ⊙        M.agenda = AG) ]
```

interaction

```
 ╱────────╲      ⚇
╱ E: Email  ╲   CHAIR: Person          E
╲          ╱        │                  ┊              C: Committee
 ╲────────╱         │                  ┊                   │
    │     scheduleMeeting(C,T,AG) │    ┊                   │
    │          │───────────────────────►                  │
    │          │                  ┊   │  true = noConflict(T)  │
    │          │                  ┊   │──────────────────────►│
    │          │                  ┊   │   mbs = C.members      │
    │          │                  ┊   │──────────────────────►│
  ┌─┴──────────┴──────────────────────┤                   │
  │ sendMail(p.email,"meeting of C at T with AG")│      Meeting
  │◄─┤                               │ │                ┌────┐
  │  │         for all p in mbs      │ │                └────┘
  │  │                               │ │  create(C,scheduled,T,AG) │
  │  │                               │ │──────────────────────►│
  │  │                  ┊            │ │                   │
```
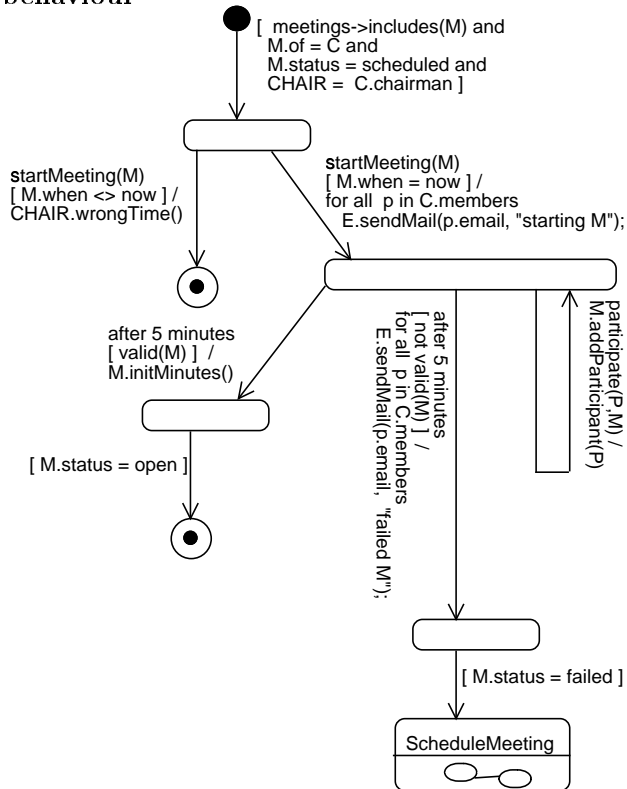
use case StartMeeting

textual When a meeting M of the committee C is scheduled and its time is the current time, its chairman may start it.
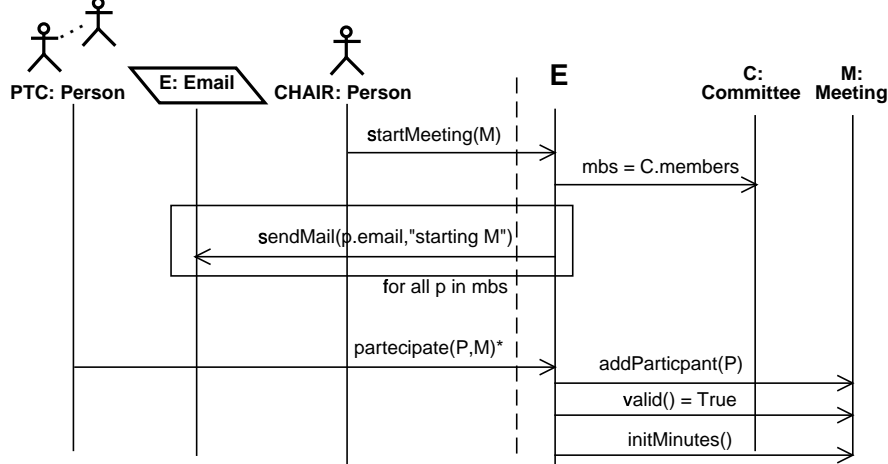
The system asks to all the members of C to confirm their participation to M.

The system checks the validity of M, if it is not valid, then the meeting is failed and the chairman schedules it for another time, otherwise the meeting is open.

behaviour

[meetings->includes(M) and
M.of = C and
M.status = scheduled and
CHAIR = C.chairman]

startMeeting(M)
[M.when <> now] /
CHAIR.wrongTime()

startMeeting(M)
[M.when = now] /
for all p in C.members
    E.sendMail(p.email, "starting M");

after 5 minutes
[valid(M)] /
M.initMinutes()

after 5 minutes
[not valid(M)] /
for all p in C.members
    E.sendMail(p.email, "failed M");

participate(P,M) /
M.addParticipant(P)

[M.status = open]

[M.status = failed]

ScheduleMeeting

interaction

PTC: Person   E: Email   CHAIR: Person   E   C: Committee   M: Meeting

startMeeting(M)

mbs = C.members

sendMail(p.email,"starting M")

for all p in mbs

partecipate(P,M)*

addParticpant(P)

valid() = True

initMinutes()

use case Handle a communication

textual When a meeting M is open, and COM is a communication appearing in its agenda, the chair of M may decide to put forward COM.
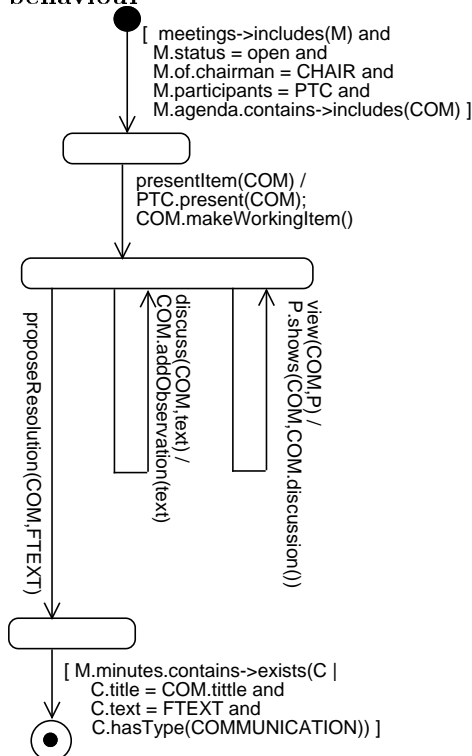The system presents the communication COM to the participants to M.
Each participant may discuss COM, by signalling to the system its observation in the form of a short text, or may examine the status of COM by asking to the system to show the discussion (just all observations made).
The chairman proposes a final resolution.
Then, the comminication COM is settled and the final resolution is added to the minutes of M.

behaviour

[meetings->includes(M) and
M.status = open and
M.of.chairman = CHAIR and
M.participants = PTC and
M.agenda.contains->includes(COM)]

presentItem(COM) /
PTC.present(COM);
COM.makeWorkingItem()

proposeResolution(COM,FTEXT)

discuss(COM,text) /
COM.addObservation(text)

view(COM,P) /
P.shows(COM,COM.discussion())

[M.minutes.contains->exists(C |
C.title = COM.tittle and
C.text = FTEXT and
C.hasType(COMMUNICATION))]

interaction
A scenario of the use case where the participants never look at the status of the discussion.
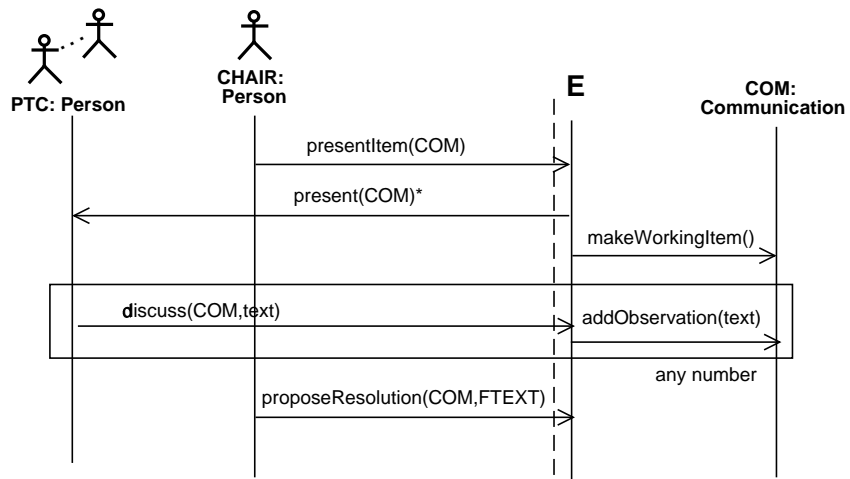
use case Handle a discussion item

textual When a meeting M is open, and DI is a discussion item appearing in its agenda, the chairman of M may decide to put forward DI.
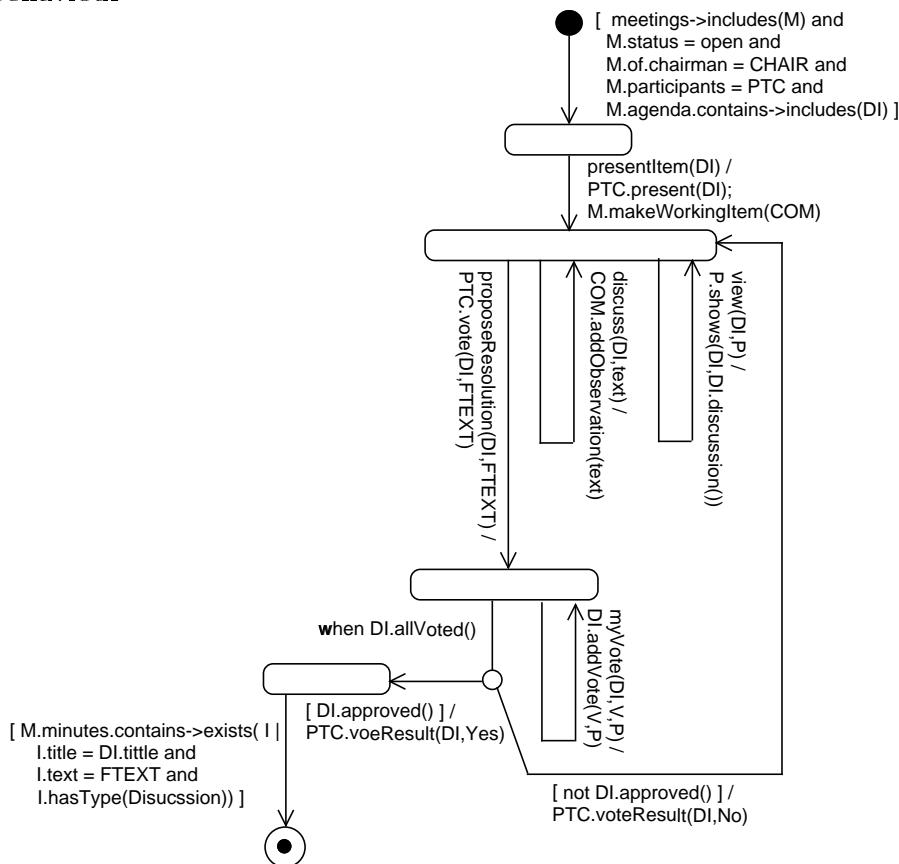
The system presents the discussion item DI to the participants to M.

Each participant may discuss DI, by signalling to the system its observation in the form of a short text, or may examine the status of DI by asking to the system top show the discussion of DI (just all observations made).

The chairman proposes a final resolution and the system invites the participants to vote it.

When all participants have voted, if the result of the votation is positive, then the item is settled and the final resolution is added to the minutes of the meeting, otherwise the chairman opens again the discussion.

behaviour

interaction

A scenario where the particpants never look at the status of the discussion and where the result of the votation is positive.

PTC: Person   CHAIR: E   DI:
                     Person   Discussion

presenttem(DI)

present(DI)*

makeWorkingItem()

discuss(DI,text)   addObservation(text)

any number

proposeResolution(DI,FTEXT)

vote(DI,FTEXT)*

myVote(P,DI,V)   addVote(P,V)

for all PTC

voteResult(DI,Yes)*

use case Handle a minutes approvation

textual When a meeting M is open, and MA are the minutes of a previous meeting to be approved appearing in its agenda, the chairman of M may decide to put forward MA.

The system presents the minutes to be approved MA to the participants to M.

Each participant may discuss MA, by signalling to the system its observation in the form of a short text, or may examine the status of MA by asking to the system to show the discussion of MA (just all observations made).

If there are observations, then the chairman proposes a new version of the minutes.
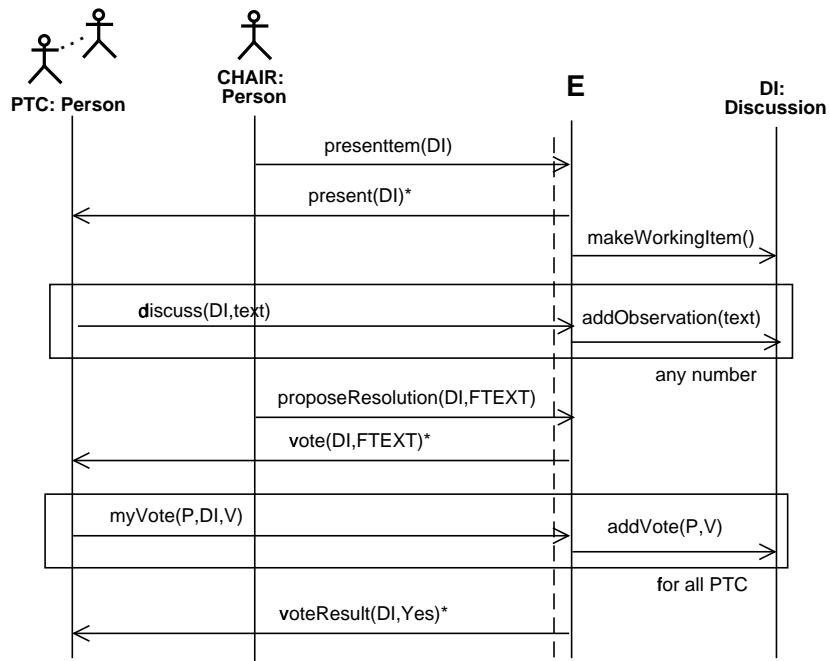
The system invites the participants to vote the minutes.

When all participants have voted, if the result of the votation is positive the minutes are approved and the final version is added to the minutes of the meeting, otherwise the system opens again the discussion.
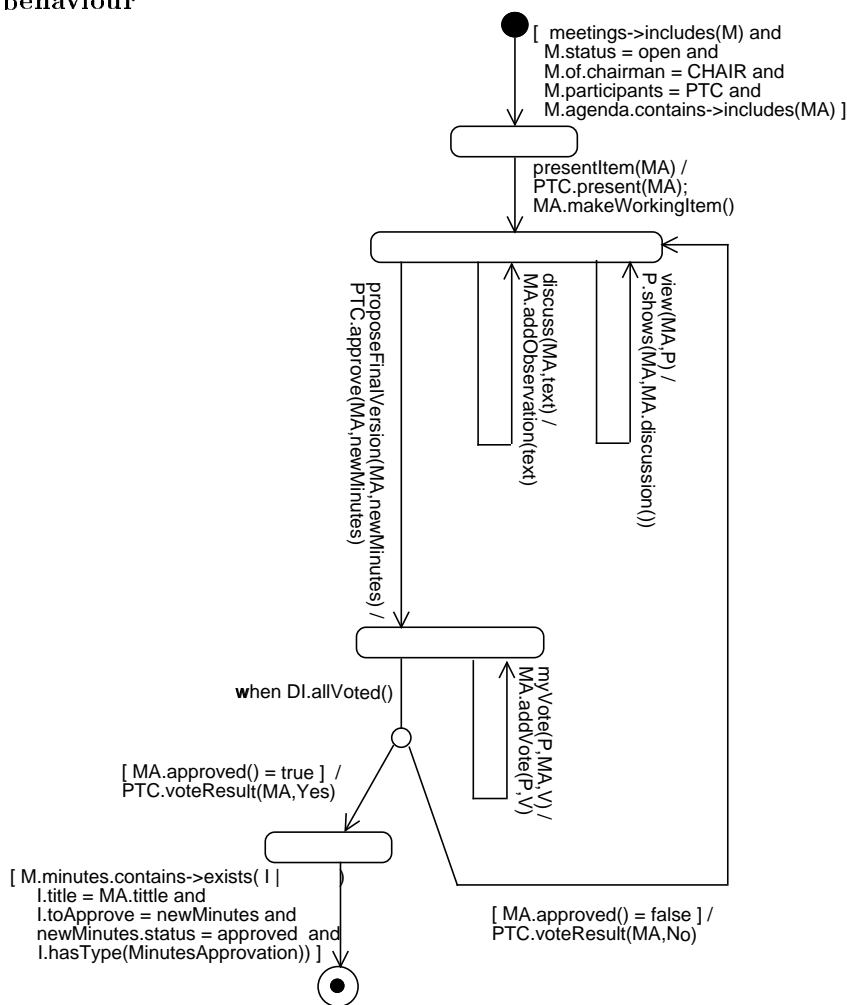
behaviour

interaction

A scenario where the particpants never look at the status of the discussion and where the result of the votation is positive.

use case Close Meeting

textual When all items in the agenda of an open meeting M of the committee C have been settled, the chairman of M may close it.

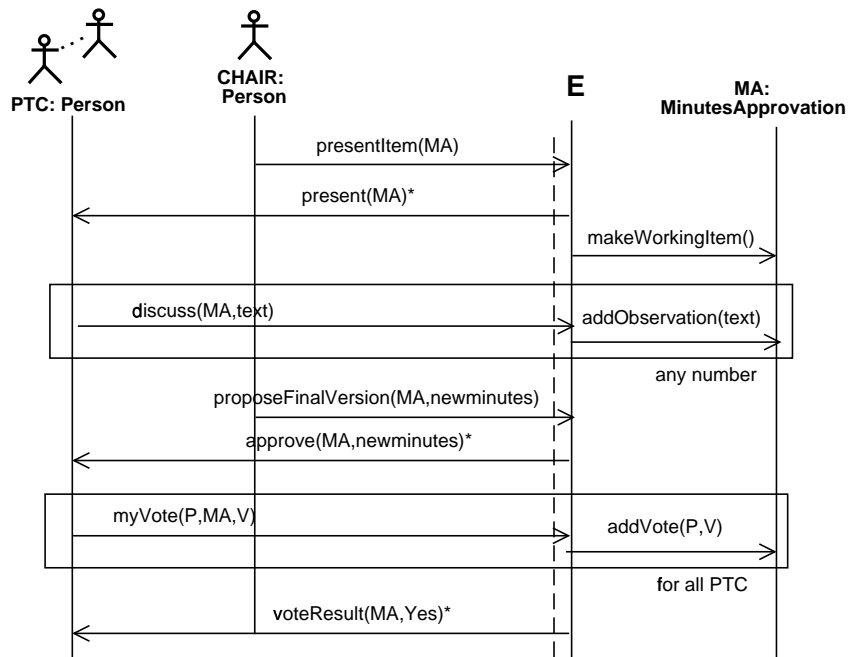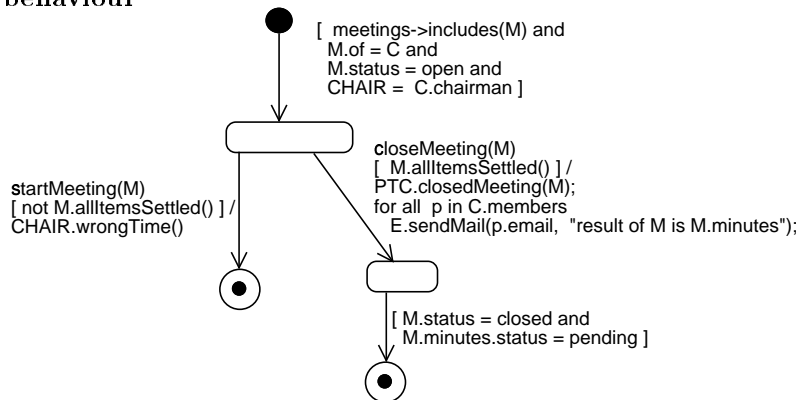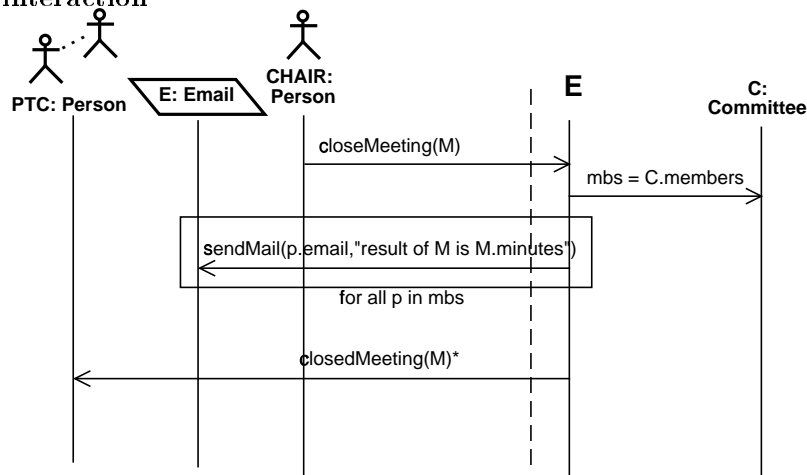The system informs all the participants that the meeting is closed, and sends a copy of the minutes to all the members of the committee C.

Then, the meeting M is closed.

behaviour

```
                          ●
                                  [ meetings->includes(M) and
                                    M.of = C and
                                    M.status = open and
                                    CHAIR =  C.chairman ]

                      ┌──────────┐
                      └──────────┘        closeMeeting(M)
                          │    ╲           [ M.allItemsSettled() ] /
                          │     ╲          PTC.closedMeeting(M);
  startMeeting(M)         │      ╲         for all  p in C.members
  [ not M.allItemsSettled() ] /  ╲            E.sendMail(p.email,  "result of M is M.minutes");
  CHAIR.wrongTime()       │       ╲
                          │     ┌──────────┐
                        ⊙       └──────────┘
                                     │   [ M.status = closed and
                                     │     M.minutes.status = pending ]
                                    ⊙
```

interaction

```
     𝗑      𝗑          𝗑
     ┊     ╱             CHAIR:
  PTC: Person  ╱E: Email╲  Person              E           C:
                                                        Committee
     │         │          │              │              │
     │         │          │ closeMeeting(M) │              │
     │         │          │─────────────────▶│              │
     │         │          │              │ mbs = C.members │
     │         │          │              │──────────────▶│
     │    ┌────────────────────────────────────┐         │
     │    │sendMail(p.email,"result of M is M.minutes")│  │
     │    │◀──────────────────────────────│    │         │
     │    └────────────────────────────────────┘         │
     │         │    for all p in mbs │              │
     │         │          │              │              │
     │         │          │ closedMeeting(M)*│              │
     │◀───────────────────────────────────────│         │
     │         │          │              │              │
```

use case Browse

textual A person belonging to the organization may ask the system to browse the information about the various committees.

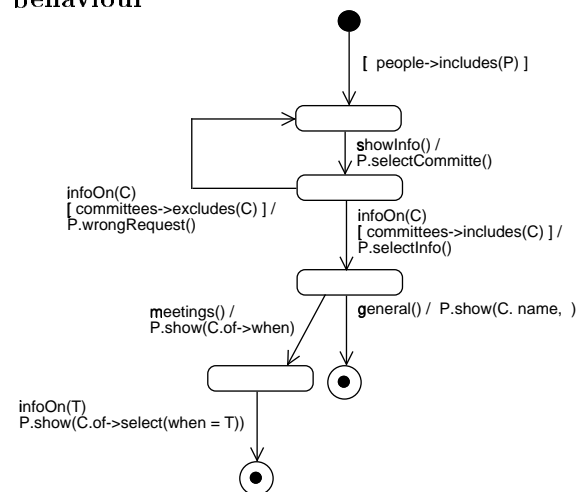The system asks to select a committee.

The person indicates a committe.

The system asks to choose between to show the general info on the committe or about a meeting.

In the first case, the system shows such information.

In the second case, the system shows the list of the meetings for such committee, and the person selects one of them.

The system shows the information on such meeting (agenda, and if available the minutes with their status).
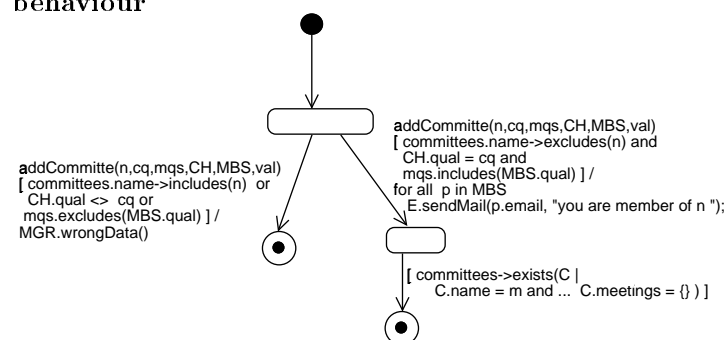
behaviour

[people->includes(P)]

showInfo() /
P.selectCommitte()

infoOn(C)
[committees->excludes(C)] /
P.wrongRequest()

infoOn(C)
[committees->includes(C)] /
P.selectInfo()

general() / P.show(C. name,)

meetings() /
P.show(C.of->when)

infoOn(T)
P.show(C.of->select(when = T))

use case Add committee

textual The manager may add a new committee by giving all its relevant information (name, qualification for chairman, qualifications for the members, rules for the validity of the meetings, chairman, and members). If there is no other committee with the same name and the qualification of the chairman and of the members are ok, then the committee is established and its members are informed of that, otherwise the request is rejected.

behaviour

addCommitte(n,cq,mqs,CH,MBS,val)
[committees.name->excludes(n) and
    CH.qual = cq and
    mqs.includes(MBS.qual)] /
for all p in MBS
    E.sendMail(p.email, "you are member of n ");

addCommitte(n,cq,mqs,CH,MBS,val)
[committees.name->includes(n) or
    CH.qual <> cq or
    mqs.excludes(MBS.qual)] /
MGR.wrongData()

[committees->exists(C |
    C.name = m and ... C.meetings = {})]

use case Delete committee

textual The manager may delete an existing committee.

The members are informed of that, and then the committee is no more existing.

behaviour

. . . .

use case Update committee

textual The manager may update the information about an existing committee.
The system presents one after the other the various information and asks either to confirm or to update
it by providing the new one.
If the new information is ok, then the system updates it, otherwise the system requires another one.

behaviour

. . . .

use case Add person

textual . . . .

behaviour

. . . . . .

use case Delete person

textual . . . .

behaviour

. . . . . .

use case Update person

textual . . . .

behaviour

. . . . . .

0.0.1   Context View

```
              Email
                │ 1
         ┌──────────────┐
         │    MSMILE    │
         └──────────────┘
          │ *          │ 1
          �      �
        Person       Manager
```

Person
wrongTime()
present(Item)
vote(Item,Text)
approve(MinutesApprovation,Minutes)
voteResult(Item,Boolean)
show(Item,Seq(Text))
closedMeeting(Meeting)
selectCommittee(Committee)
wrongRequest
selectInfo
show(AnyType)

Manager
wrongName(String)

Email
send(String,String)

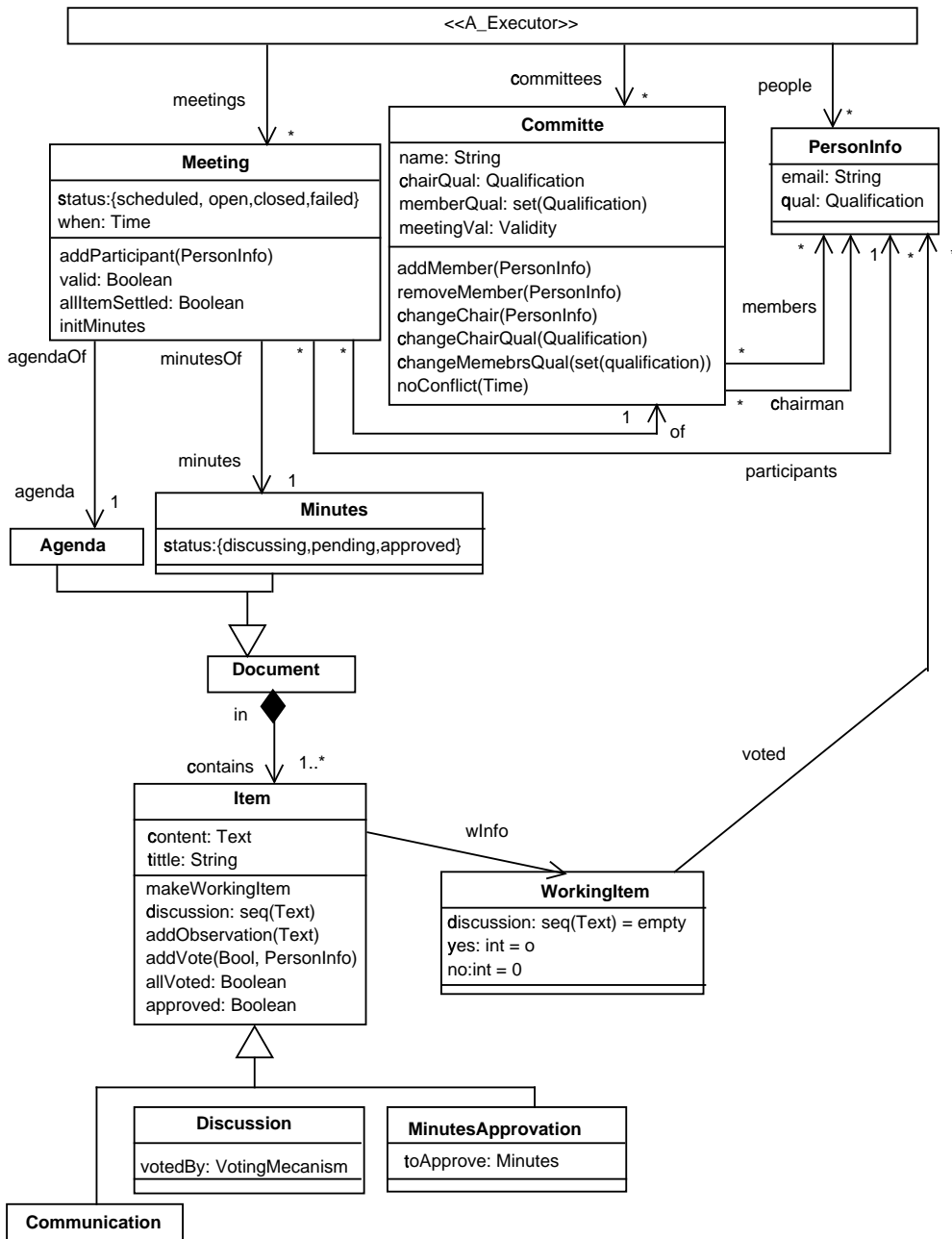| <<System>>
MSMILE |
| --- |
| scheduleMeeting(Committee,Time,Agenda) |
| startMeeting(Meeting) |
| partecipate(PersonInfo,Meeting) |
| presentItem(Item) |
| proposeResolution(Item,text) |
| proposeFinalVersion(MinutesApprovation,Minutes) |
| myVote(Item,Boolean,PersonInfo) |
| view(Item,PersonInfo) |
| discuss(Item,Text) |
| closeMeeting(Meeting) |
| showInfo |
| infoOn(Committee) |
| addCommittee(..........) |

0.0.2 Internal View

Constraints on the Internal View

The qualifications of chairman and of the
members of a committee are ok.

```
context C: Committee  inv:
  C.chairmam.qual = C.chairQual and
  C.memberQual->includes(C.members.qual)
```

All the committes have different names.

```
context Committee inv:
  Committee.allInstances->forall(C1, C2 |
            C1.name <> C2.name <=> C1 <> C2)
```

All the persons belonging to the organization have different emails.

```
context personInfo inv:
  personInfo.allInstances->forall(P1, P2 |
            P1.email <> P2.email <=> P1 <> P2)
```

Only scheduled and failed meetings do not have minutes.

```
context M. Meeting inv:
M.minutes <> {} <=>
    (M.status <> scheduled and M.status <> failed)
```

Definition of the operations of the **Internal View**

A meeting is valid when the participants of any category
are at least in the required rate.

```
context Meeting::valid()
post: result = M.of.memberQual->forall( Q |
            (M.of.members->select(qual=Q) / M.participants->select(qual=Q)) <=
          M.of.meetingVal.howMany(Q) )

context Meeting::addParticipant(P)
post: M.participants->includes(P)

context Meeting::initMinutes()
post: M.minutes->exists(M | M.contains = {} and M.status = discussing)
```

A meeting is settled when all items have been considered, and thus
reported in the minutes.

```
context Meeting::allItemSettled()
post: result = M.agenda->size = M.minutes->size

context Committe::addMember(P)
post: C.members->includes(P)

context Committe::removeMember(P)
post: C.members->excludes(P)

context Committe::changeChair(P)
post: C.chairman = P

context Committe::changeChairQual(Q)
post: C.chairQual = Q

context Committe::changeMemberQual(QS)
post: C.chairQualQual = QS
```

A data for a meeting of a committee C is no conflicting whenever
there is another meeting scheduled for a committee with a common
member with C.

```
context Committe::noConflict(T)
post: meetings->select(M.status = scheduled and
                        M.of.members.meet(self.members)->notEmpty)->
        forall(M | M.when.diff(T) > 5 hour)

context Item::makeWorkingItem()
post: I.winfo = create(WorkingItem)

context Item::discussion()
post: result = self.winfo.discussion

context Item::addDiscussion(TXT)
post: self.winfo.discussion = [ TXT, self.winfo.discussion@pre]

context Item::addVote(B,P)
post: self.winfo.voted->includes(P) and
      if B then self.winfo.yes = self.winfo.yes@pre + 1
       else self.winfo.no = self.winfo.no@pre + 1

context Item::allVoted()
post: result = I.voted->size = I.in.agendaOf.participants->size

context Discussion::approved()
pre: allVoted()
post: result =
D.votedBy.check(winfo.yes,
                winfo.no,
D.in.agendaOf.participants->size,
D.in.agendaOf.of.memebrs->size)
```

0.0.3 Dictionary

```
+-----------------------------------+        +-------------------------------+
|           <<datatype>>            |       /|A voting mechanism determines |
|        VotingMechanism            |      /_|the result of a votation given the|
+-----------------------------------+        |number of yes, of no and of the|
| check(Yes:Int, No: Int, voters: Int): Boolean|- - - -|person having the right to vote.|
+-----------------------------------+        +-------------------------------+
```

```
    +-----------------------------------+      +-------------------------------+
    |           <<datatype>>            |     /|A validity gives the rule for  |
    |             Validity              |    /_|checking if a meeting is valid, by|
    +-----------------------------------+      |saying wich is the required rate of|
    | howMany(Qualification): Int       |- - - |participants for any category of|
    +-----------------------------------+      |them, determined by the        |
                                               |qualificataions.               |
                                               +-------------------------------+
```

```
+---------------------+  +---------------------+  +---------------------+
|    <<datatype>>     |  |    <<datatype>>     |  |    <<datatype>>     |
|   Qualification     |  |        Time         |  |        Text         |
+---------------------+  +---------------------+  +---------------------+
```