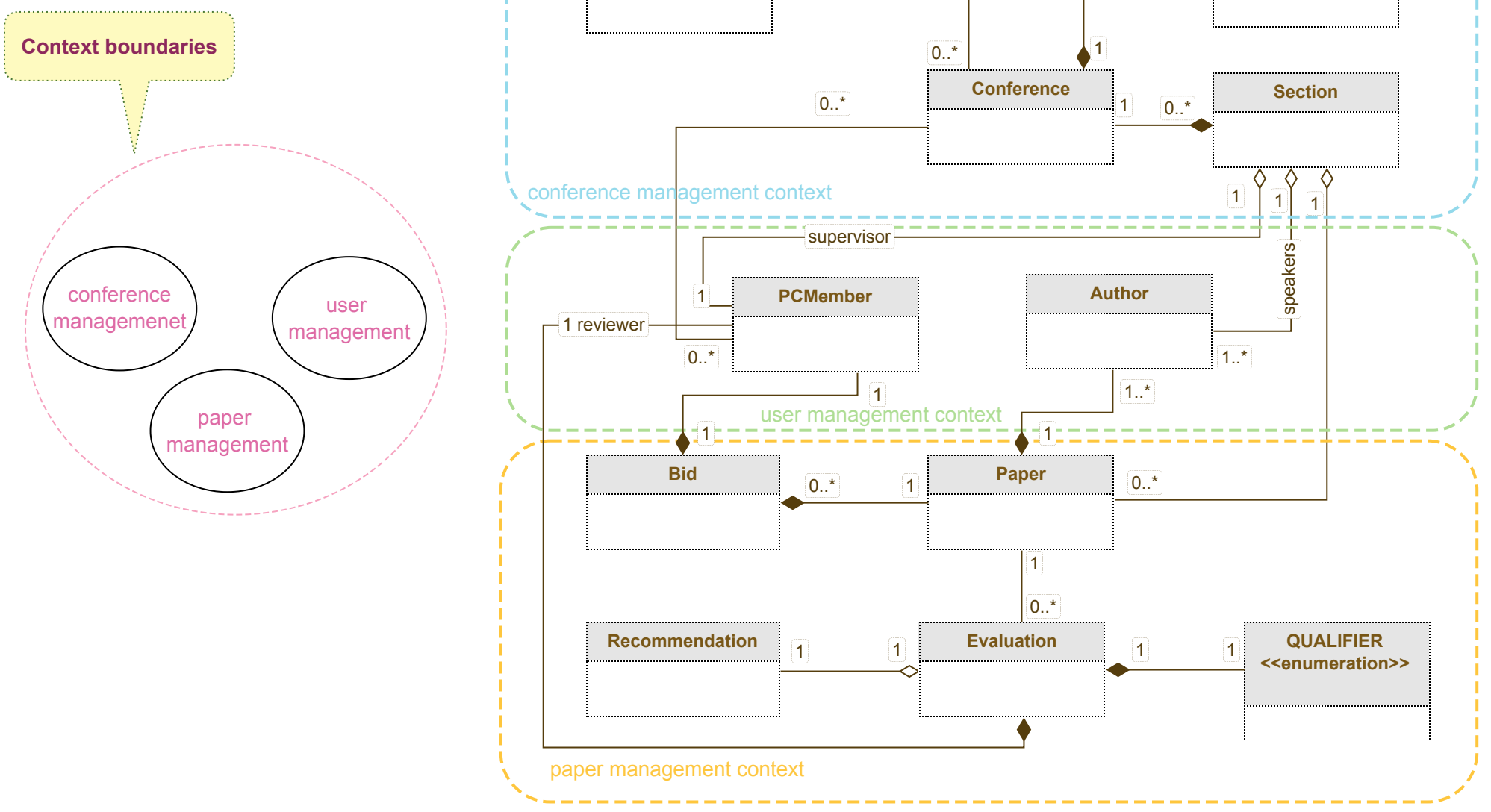


```
context Paper :: bidPaper(Event conference. Paper paper, PCMember reviewer, Bid.Result) {
    Pre: conference.program.bidDeadline().after(date.now()) and  not  paper.bids.stream(bid ->bid.reviewer equals reviewer)
    Post: paper.bids.add(bid.result)
}

context Paper ::  assignPaperToReview(Paper paper, PCMember reviewer) {
   Pre: paper.reviewers.size() < 4 and  not paper.reviewers.contains(reviewer)
          and paper.bids.filter(reviewer).result != "REFUSED"
   Post: paper.reviewers.add(reviewer)
          paper.reviewers() = paper.reviewers()@pre() + 1

}

context Paper :: reviewPaper(Paper paper, PCMember reviewer, Evaluation evaluation) {
   Pre:  paper.reviewers.contains(reviewer)
   Post: paper.add(evaluation)
          paper.evaluations() = paper.evaluations()@pre() + 1
}

context Conference:: acceptedPapers(Event conference) {
  conference.submittedPapers.filter(paper - > paper.evaluations.filter( eval -> eval in [strong accept, accept, weak accept,
borderline paper]))
}

context Conference :: rejectedPapers(Event conference) {
  conference.submittedPapers.filter(paper - > paper.evaluations.filter( eval -> eval in [weak reject, reject and strong reject]))
}

context Conference :: assignSectionChair(Event conference, Section section, PCMember member) {
   Pre: section.chair.isNull() == true and member.isSectionChair() == false
   Post: section.addChair(member)
}

context Conference :: changeBidDeadline(Event conference, Date newDate){
   Pre:   newDate .after(date.now())
   Post: conference.setBidDeadline(newDate)
}

context Conference :: addSection(Event conference, Section section){
   Pre: conference.sections.contains(section) == false
   Post: conference.sections.add(section)
          conference.sections() = conference.sections()@pre() + 1
}

context User :: registerPCMember(Event conference, PCMember newMember){
   Pre: newMember.validate() == "VALID" and conference.pcmembers.contains(newMember) == false
   Post: conference.pcmembers.add(newMember)
          conference.pcmembers() = conference.pcmembers()@pre() + 1
}

context User :: registerAuthor(Event conference, Author newAuthor){
   Pre: newAuthor.validate() == "VALID" and conference.authors.contains(newAuthor) == false
   Post: conference.authors.add(newAuthor)
          conference.authors() = conference.authors()@pre() + 1
}
```