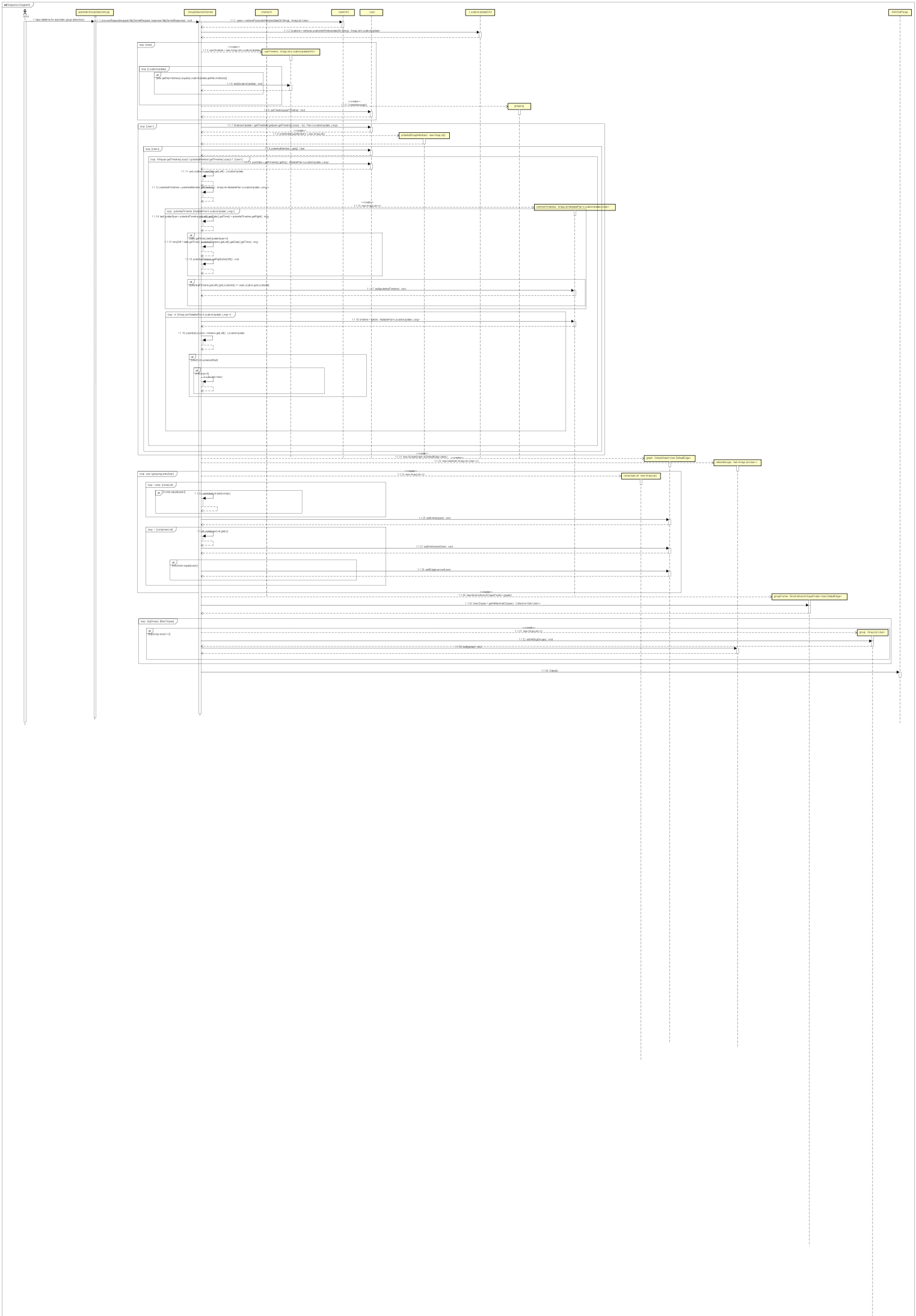
sd Sequence Diagram1
1: Input dateline for automatic group detection()
automaticGroupDetection.jsp
1.1.1. processRequest(request:HttpServletRequest, response:HttpServletResponse) : void
GroupDetectionServlet
LifelineDAO
UserDAO
User
LocationUpdateDAO
HGOOutPut.jsp
1.1.1. users = retrieveTimewiseWhereDateIsOn(String) : Array<dot.User>
1.1.2. locations = retrieve_locationWhereDateIsOn(String) : Array<dot.LocationUpdate>
loop: [User]
<<create>>
1.1.3. userTimeline = new Array<dot.LocationUpdateDAO>
userTimeline : Array<dot.LocationUpdateDAO>
loop: [Location.Updates]
alt
Math.getHaDistance(Location.Update.getMaxAddress())
1.1.4. addLocation(Location.Update) : void
<<create>>
1.1.5. grouping = new Grouping()
grouping
1.1.6. setTimeline(userTimeline) : void
loop: [User]
1.1.7. finalUser.Update = getTimewise(Location.getUser.getTimewise(Location) - 1) : Pair<Location.Update, Long>
<<create>>
1.1.8. potentialMembers = new Array()
potentialSupplierMembers : new Array()
loop: [User.j]
1.1.9. potentialMember = get(j) : User
loop: [Minimum getTimewise().size() - 1 potentialMember.getTimewise().size() > 1 [User.k]]
1.1.10. userData = getTimewise().get(k) : MobilePair<Location.Update, Long>
1.1.11. usrLocationUpdate = getLeft() : Location.Update
1.1.12. potentialTimewise = potentialMember.getTimewise() : Array<dot.MobilePair<Location.Update, Long>>
<<create>>
1.1.13. new Array()
commonTimewise : Array<dot.MobilePair<Location.Update>>
loop: potentialTimewise [MobilePair<Location.Update, Long>]
1.1.14. setLocationOpen = potentialTimewise.getLeft().getOpen() : potentialTimewise.getRight() : long
alt
[usrLocationUpdate.getOpen() < potentialTimewise.getLeft().getOpen()]
1.1.15. timeDiff = date.getTime() - usrLocationUpdate.getLeft().getDate() : getTime() : long
1.1.16. potentialTimewise.setRight(timeDiff) : void
alt
[usrLocationUpdate.getLeft().getLocationId() == user_location.getLocationId()]
1.1.17. addPotentialTimewise() : void
loop: m [Array<dot.MobilePair<Location.Update, Long>>]
1.1.18. timeline = get(m) : MobilePair<Location.Update, Long>
1.1.19. potentialLocation = timeline.getLeft() : Location.Update
alt
[!(potentialLocation.isNull())]
alt
[timeline.isNull()]
1.1.20. add(m)
<<create>>
1.1.21. new Array(graph.getDefaultEdges())
graph : GroupGraph<User,DefaultEdge>
<<create>>
1.1.22. new HashSet<Array(graph.getUserVertices())>
vertexSetUsers : Set<Array(dot.User)>
loop: pair [grouping.getNextSet()]
<<create>>
1.1.23. new Array()
comp(user_id : new Array()
loop: comp : [comp(user)]
alt
[!(comp.equals(user))]
1.1.24. comp(user_id).add(comp)
1.1.25. addVertex(user) : void
loop: j : [comp(user).size]
1.1.26. comp(user).get(j)
1.1.27. addVertex(nextUser) : void
alt
[!(nextUser.equals(user))]
1.1.28. setEdge(user,nextUser) : void
<<create>>
1.1.29. new BronKerbschCliqueFinder<User>(graph)
groupFinder : BronKerbschCliqueFinder<User,DefaultEdge>
1.1.30. maxCliques = getAllMaximalCliques() : Collection<Set<User>>
loop: [cpGroups : [maxCliques]]
alt
[cpGroup.size() >=2]
<<create>>
1.1.31. new Group()
group : Array<dot.User>
1.1.32. add(cpGroup) : void
1.1.33. fillGroupMap() : void
1.1.34. Output()