**Java**

1. Distributed systems
   1. What it is (multiple computers or languages and components)
   2. Features (reliable, scalable, complex)
   3. Architecture: client-server (e.x. 3-tier)/ peer-to-peer
   4. Examples of systems (fb, google)
2. 3-tier
   1. Connections between tier (who is client, server)
   2. Responsibilities and examples of functions
   3. Example: sep
3. Remote invocation
   1. Request-Reply protocol (used in UDP???(making it more reliable?), Rest) ->also in SEP
   2. General RMI
   3. RPS (book) -e.x. Soap  
      =>worth reading the book
   4. Remote invocation – issues
   5. Different languages- different types- converting types for the languages- serialization- work both ways in both languages
   6. Maintaining the state- when a change on one computer, should also be on the other (state passed, not copy)
   7. Garbage collector-
   8. Book???
4. Java RMI
   1. Initialize registry across repositories, references, unicast remote object (maintaining the state) -> parameter passing+ interfaces to call methods
5. Peer-to-peer
   1. Differences with client-server, where better (pros and cons)
   2. E.x. torrent
   3. Algorithms? Architecture, routing (ring -pastry algorithm?, address server, groups, book)
   4. Issues
6. Indirect communication
   1. Publish-subscribe, group ->algorithms to distribute
   2. Reasons: not directly to a person, publishing broker
   3. Time/ space uncoupled- not logged in all the time to receive/ not knowing the sender
   4. E.x. message queues, etc
7. Web services, client/server, soap
   1. WSDL file
   2. Tutorials written, step by step
   3. Middleware Soap: RMI across languages
8. Web services, client/server, rest
   1. Rest: not that heavy, commands to communicate, not objects but serialization, no XML files in both book?
   2. On example of Spring?
9. Coordination of distributed systems
   1. Those are ???
   2. Token ring
   3. Multi-cast
   4. Buli algorithm
   5. Coordination
   6. Mutual exclusion?
10. Distributed transactions
    1. E.x. on hibernate- sessions
    2. Also from DBS