CS 499 – Distributed Systems  
Wednesday, September 09, 2009

Class notes:

* This Friday will be a workshop in a socket programming
  + look into the programming codes and review before this Friday
  + this is more of a crash course in getting
* For Monday, we will begin Quizzes
  + we will start a series of quizzes that will test us on book and lecture material

Lecture Notes:

* A model of Distributed Systems
  + Events
    - Events can be internal event, message sent event, message received event
    - exi 🡪 event x on process Pi
    - for message sent(m), receive(m)
    - events change the state of a DS
      * internal events changes state of process
      * send event 🡪 process sending message, channel
      * receive event 🡪 process receiving message, channel
    - Execution of process Pi 🡪 e1i, e2i, e3i, …, eni 
      * something is happening when you send a message
    - History Hi = (hi, 🡪 i)
      * hi is the set of events produced by Pi
      * “🡪” is the binary relationship that orders the events
    - Send/Receive Events
      * flow of information between processes
        + create a causal dependency between sender and receiver

“🡪msg” is the binary relationship

* + - Space-Time Diagram
      * Look to figure in book; three lines representing time. Each line represent a process and events are dots on the graph.
      * relationships between events are not necessarily related
    - Causal Precedence
      * distributed application 🡪 results in a set of distributed events
      * H = Uibi 🡪 “Binary relationship on history”
        + (H)istory, (U)nion
        + for all exi, forall eyj are elements of H, exi 🡪 eyj ⬄ Set{

exi 🡪 ieyj,

* + - * for any two events ei and ej we say ei is unrelated to ej; no causal dependency between the two
      * For any two events ei and ej,
        + ei -/-> ej =/=> ej -/-> ei
        + e.g. e22-/-> e21 =/=> e21 -/-> e22
        + ei 🡪 ej => ej -/-> ei
    - Concurrent Events
      * If ei -/-> ej & ej -/-> ei
      * then => ei || ej