A software/system-engineering (SE) project review meeting in the era of wise computing: Examining prototypes of a medical patient-assistance robot (on desks) and associated software. A computer (for illustration is shown in the bottom right) will be able to contribute questions and observations in ways commonly expected only from humans.

### Special Human Competencies

1. Notice irregularities, unexpected properties:
   - “The arm movement is not smooth!”
   - “Hear this strange noise!”
2. Detect missing requirements and undocumented assumptions:
   - “Will it understand voice commands from a hoarse patient?”
   - “Can it process the patient's voice commands when the TV is on?”
3. Ask and answer hard “what if” and “why” questions:
   - “Will a loud command from a TV show, like in a gym class, confuse the robot?”
   - “Why is it walking aimlessly around? Is it looking for something?”
4. Communicate concurrently with multiple levels of abstraction:
   - “In <this> line of code we turn on the busy bit; this will make the system status light blink; the user will thus know that he/she should wait before stating the next voice command”
   (from programming detail → system capability → user need)
5. Free Association:
   - “I have heard that someone remotely hacked a pacemaker. Can this happen here?”
6. Creativity: Thinking outside the box
   - “I bet it will be confused if I ask it to fetch a bottle that is glued to the table.”
   - “Etc.”

### CF: A Common Formalism for all Knowledge: Project, SE, Domain

- A single formalism for:
  - All SE artifacts
  - SE knowledge
  - Domain knowledge
- Externalizing internal information from SE tools
- With extensive annotations and relationships
- Enables applying general knowledge to specific cases
- Extensive ability for meta-referencing
- Based on, and substantially enhancing, Statecharts and Scenario-Based Programming (SBP)

### AE: An Analysis Engine that Mimics Unique Human Skills

- Ongoing analysis of all project artifacts (in CF):
  - Simulations and testing
  - Verification
  - Detection of unexpected properties
  - Monitoring
  - Automated Repair
  - Application of external knowledge
  - Source code and documentation analysis
- A single interface to all wise computing tools
- Based on text statements
- Controlled English
- Fully formalized

### IL & IE: An Interaction Facility for Communicating on all SE Matters

**The Interaction Language (IL)**

- Captures everyone engineers may state about a system
- Steps in an algorithm
- Requirements/User needs
- Development tasks
- System components, configuration
- Bug descriptions
- A single interface to all wise computing tools
- Based on text statements
- Controlled English
- Fully formalized

**The Interaction Engine (IE)**

- Controls all interaction with engineers
- In offline analyses and interactive sessions
- Clarifies exact meaning of user inputs
- Disambiguation, context, confirmation
- Enables all tools to produce consistent information with regard to all development stages, tasks and artifacts

### DP: Design Patterns that Facilitate Wise Computing

- Specify knowledge as scenarios that try to apply it
- Components self-integrate into a target system without requiring that the system be changed to accommodate them
- Actively seek out/collect component’s own input
- Components make outputs and constraints available to all
- New components comply with constraints imposed by existing one
- Example: A model of Citric Acid (Krebs) Cycle in the cell, where chemical reactions are modeled separately

---

1. If you want to...