**Efficiency**

Scale up or down correctly, depending on what the needed capacity is for users, terminals, transactions etc.

Consider what to do when the system is overloaded. Decide whether to let the system run at a decreased rate of performance, or allow the system to crash.

**Usability**

Find problems in the UI design, make recommendations to fix them and implement the fixes.

Improving UI design throughout the usage of the system.

- Create criteria to evaluate the UI design against

**Safety**

Make sure the safety of classes is of the utmost importance.

Implement methods to decrease the chances of safety failures

- Hazard avoidance
- Hazard detection
- Hazard removal
- Damage control

Run analysis on possible hazards and risks

- Identify risks
- Risk analysis
- Risk reduction/removal

**Security**

Make sure the security is up to date and properly working.

Deny service to interior and exterior threats.

Monitor corruption of programs, data or the system as a whole.

Implement detection and neutralization systems.

Limit damage if the system becomes compromised.