

Lecture 19 objectives

- **Teams (from EE155 by Neal Howard)**
- **Design Process**

What is a team?

"A team is a small number of people with complementary skills who are committed to a common purpose, performance goals, and approach for which they are mutually accountable."

(Katzenbach, J.R. & Smith, D.K. (1993). *The Wisdom of Teams: Creating the High-performance Organization*. Boston: Harvard Business School.)

Stages of Team Development

▶ Forming

The stage where team members are just becoming acquainted—the "honeymoon"

▶ Storming

Conflict begins as team members negotiate work assignments, discuss what to do

▶ Norming

Team members learn to work together—pride begins to develop

▶ Performing

Team settles down and most of the work gets done

Decker, Philip, J. (1996) "Characteristics of an Effective Team," (Powerpoint Presentation)
http://www.cl.uh.edu/bpa/hadm/HADM_5731/ppt_presentations/29teams

Team Leader Role

▶ Responsibilities:

Calling meetings including finding a mutually agreeable time and place

Setting a meeting agenda (more on this later)

Facilitating the meeting (more later)

Monitoring progress against the plan

Identifying problem areas that need action

▶ Some rules:

The leader is not "the boss"

The team needs to agree on decisions and directions

Compromise is essential

Gantt chart

ID	Task Name	Start	Finish	Duration	Jan 2004																
					20	21	22	23	24	25	26	27	28	29	30						
1	Organize Team All Team Members	1/21/2004	1/21/2004	1d		■															
2	Identify Alternative Project Topics Who: John, Sue	1/22/2004	1/23/2004	2d			■	■													
3	Call Team Meeting to Discuss Topics Who: Karen	1/22/2004	1/23/2004	2d			■	■													
4	Submit Team Topic Who: Karen	1/23/2004	1/23/2004	1d				■													
5	Team Topic Due	1/28/2004	1/28/2004	0d																	◆

Holding Effective Meetings—Tips for Success

- ▶ Before the meeting,
 - Name someone to be the facilitator
 - Create an agenda and send it to all team members
- ▶ Set a time limit for the meeting
- ▶ During the meeting, if issues emerge that are not on the agenda, the facilitator should:
 - Ask the team if this should be discussed now, or
 - Table the issues for the end of the meeting
- ▶ During the meeting:
 - Keep a list of decisions and actions items
 - Keep to the time commitment
 - Create an agenda for next meeting and agree on time and place
- ▶ After the meeting:
 - send out a brief summary
 - list of action items
 - those responsible for those actions

Brainstorming

- ▶ Select someone to be the recorder
- ▶ Invite everyone to give their ideas and input
- ▶ Write down all ideas without criticism or discussion
- ▶ After complete list is generated, return for discussion/analysis
- ▶ Carefully select the best approach or idea from the list

Brainstorming-Hints for success

- ▶ Avoid being judgmental of others' ideas
- ▶ Try to look at all sides of an idea.
- ▶ Listen attentively and treat your teammates' opinions with respect
- ▶ Try to encourage the widest range of new ideas
- ▶ Everyone should participate
- ▶ Don't stop the idea session too soon
- ▶ Try to remove your ego from the discussions.
- ▶ Don't take the rejection of your ideas or disagreements personally.

Group Communication

- ▶ Listen attentively and respect the opinions and ideas of your teammates
- ▶ Ask questions
- ▶ Give constructive feedback:
 - Present your ideas forcefully, but keep an open mind.
 - Restate the original idea to be sure it's understood
 - Critique the idea, not the person
 - Be courteous
 - Be aware of body language and tone
- ▶ Meetings don't need to be a death march—OK to have some fun
 - Use humor effectively
 - Laugh with someone, do not laugh at someone

Team Problems

- ▶ Frustration over the size of the project
 - Members think of an individual endeavor rather than a group endeavor
 - Break the project up into tasks and engage all group members
 - Set realistic dates for each task

Team Problems: Conflict

- ▶ Internal conflict - An team member is experiencing a personal conflict that is interfering with his or her ability to perform

- ▶ Individual conflict with another team member - One team member is in conflict with another
- ▶ Individual conflict with the entire team - One team member is experiencing conflict with the entire team
- ▶ Conflict between several team members - The entire team is experiencing conflict with several other team members

Conflict Resolution

- ▶ Acknowledge that the conflict exists.
- ▶ Gain common ground.
 - Seek to understand all angles: Let each person state his or her view briefly.
 - Have neutral team members reflect on areas of agreement or disagreement.
 - Explore areas of disagreement for specific issues.
 - Have opponents suggest modifications to their points of view as well as others.
- If consensus is blocked, ask opponents if they can accept the team's decision.
- ▶ Attack the issue, not each other.
- ▶ Develop an action plan.

An Effective Team Checklist

- ▶ Define a common goal for the project.
- ▶ List tasks to be completed.
- ▶ Assign responsibility for all tasks.
- ▶ Develop a timeline and stick to it.
- ▶ Develop and post a Gantt chart for the plan
- ▶ Document key decisions and actions from all team meetings.
- ▶ Send reminders when deadlines approach.
- ▶ Send confirmation when tasks are completed.
- ▶ Collectively review the project output for quality

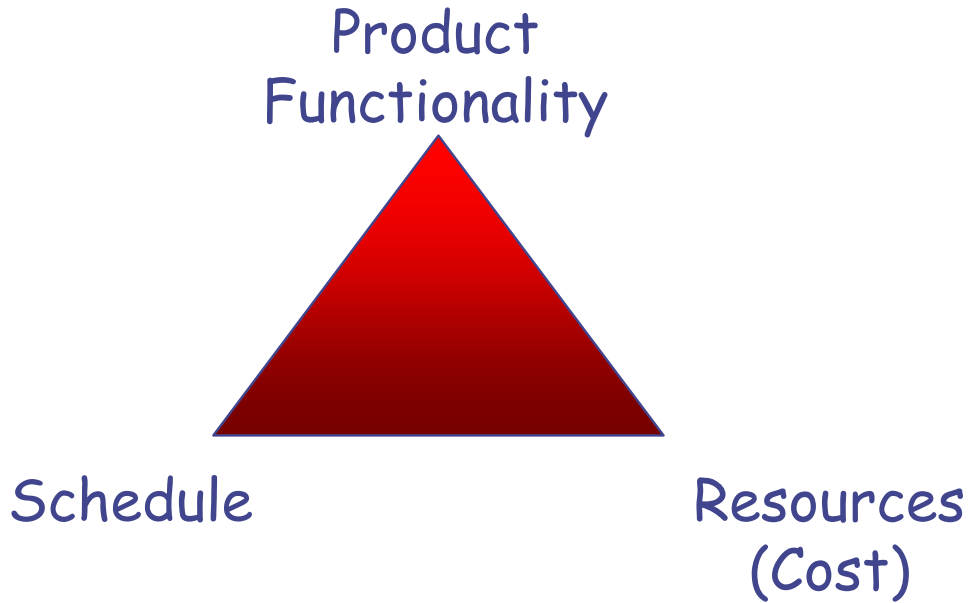
References

Breslow, L. (1998). *Teaching Teamwork Skills, Part 2*. Teach Talk, X, 5.
<http://web.mit.edu/tll/published/teamwork2.htm>. 13 May 2003.
Building Blocks for Teams, (Website). Penn State University,
<http://tlt.its.psu.edu/suggestions/teams/student/index.html>

Design Process

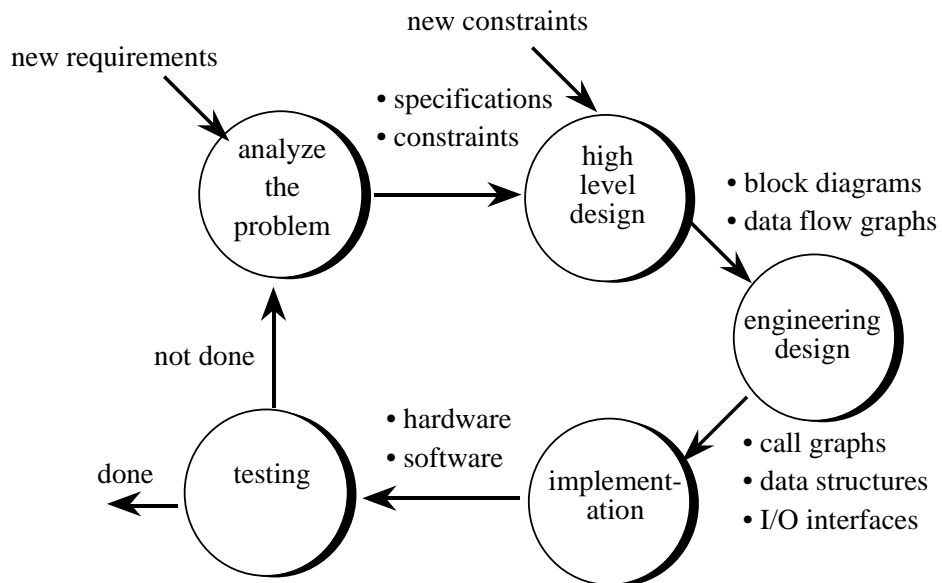
We can only optimize two of the following

- Schedule
- Resources
- Cost



Design Cycle

Example: Lab 30



1) Analysis phase

requirements parameters that the system must satisfy

- Find the ball
- Roll the ball to other side
- Know where you are
- 2 minute time

specifications parameters describing how the system should work

- 9 inch wide
- 12 inch long
- one 8.4V battery
- +2 point for hitting opponent's end wall
- +1 point for 20-sec violation
- 2 minute competition

constraints limitations, within which the system must operate

- motors and sensors from the kit
- play nice with other robots
- interfaces with other instruments and test equipment,
- development schedule.

2) High-level design phase (project proposal)

- build conceptual models
 - *data flow graph*
 - *block diagrams*
 - *fundamental equations*
- exploit abstraction
- search for existing components
- estimate the cost, schedule, expected performance, expected profit

draw a data flow graph

3) Engineering design phase

- hierarchical structure
 - *Call-graphs*
 - *Data structures*
 - *Flow charts*
- basic I/O interfaces
- overall software scheme
- direct correlation between hardware/software systems and conceptual models
- built mock-ups of the mechanical parts (connectors, chassis, cables etc.)
- mock-ups user software interface

draw a call graph

4) Implementation phase

- concurrent implementation
- initially implement using simulation
 - quicker (more loops)
 - cheaper (can carry alternative designs later into the development process)
 - greater visualization (easier to debug)
- evaluation boards

5) Testing phase

- concurrent debugging
- evaluate the performance
 - specifications
 - constraints
- verify the system
- validate basic functions
- convinces new customers to buy it
- provides legal documentation in a lawsuit

System maintenance is extremely important.

- verification of proper operation,
- updates,
- fixing bugs,
- adding features,
- extending to new applications,
- change user configurations,
- optimizing for execution speed or program size
- porting to new hardware/software platforms

Maintenance

- not a separate phase
- additional loops around the development cycle.

Modular Design

Of the following three objectives when dividing a software project into subtasks, it is really only the first one that matters

- **make the software project easier to understand**
- increase the number of modules
- decrease the interdependency (minimize coupling).

Formally, I/O devices are

- considered as global
- reside permanently at fixed addresses
- any module has access to any I/O device

To reduce the complexity of the system in practice we will

- treat I/O devices as local to the module
- restrict the modules that actually do access the I/O device
- clarify which modules have access to I/O devices
- specify rules about when they are allowed to access it
- develop ways to arbitrate (which module goes first)
- develop synchronization (make a second module wait)

