School of Computer Science and Engineering (CSE)

COMP9900 Information Technology Project
COMP3900 Computer Science Project

2023 Term 3

Week 8

Dr Rachid Hamadi
(r.hamadi@unsw.edu.au)

UNSW
SYDNEY

Outline

- Teamwork
- Managing Project Conflict
- Progressive Demo B
- Retrospective B
- Week 8 Lab Tasks
- Q & A

Teamwork

Introduction

- Most **software development** is a team (group) activity
  - The development schedule for most non-trivial software projects is such that they **cannot** be **completed** by **one person** working alone
- A good team is **cohesive** and has a **team spirit**
  - The people involved are **motivated** by the **success of the team** as well as by their **own personal goals**
- Team **interaction** is a key determinant of team **performance**
- **Flexibility** in team composition is **limited**
  - Managers must do the best they can with available people

Team Cohesiveness

- In a **cohesive** team, members consider the **team** to be more **important** than any **individual** in it

- The advantages of a cohesive team are:
  - Team **quality standards** can be developed by the team members
  - Team members learn from each other and get to know each other's work. Inhibitions caused by ignorance are reduced
  - **Knowledge** is **shared**. Continuity can be maintained if a team member leaves
  - **Refactoring** and **continual improvement** is encouraged. Team members work **collectively** to deliver **high quality results** and **fix problems**, irrespective of the individuals who originally created the design or program

Team Spirit Building Scenarios

- Alice, an experienced project manager, understands the importance of creating a **cohesive** team. As they are **developing a new product**, she takes the opportunity of involving all team members in the product specification and design by getting them to **discuss possible technology with elderly members of their families**. She also encourages them to bring these family members to meet other members of the development team
- Alice also arranges **monthly lunches** for everyone in the team. These lunches are an opportunity for all team members to meet informally, talk around issues of concern, and get to know each other. At the lunch, Alice tells the team what she knows about organizational news, policies, strategies, and so forth. Each team member then briefly summarizes what they have been doing and the team discusses a general topic, such as **new product ideas from elderly relatives**
- **Every few months**, Alice organizes an '**away day**' for the team where the team spends two days on 'technology updating'. Each team member prepares an update on a relevant technology and presents it to the team. This is an off-site meeting in a good hotel and plenty of time is scheduled for discussion and social interaction

Team Effectiveness

- The people in the team
  - You need a **mix** of people in a project team as software development involves diverse activities such as **negotiating with clients**, **programming**, **testing** and **documentation**
- The team organization
  - A team should be **organized** so that individuals can contribute to the best of their abilities and tasks can be completed as expected
- Technical and managerial communications
  - Good **communications** between team members, and between the software engineering team and other project stakeholders, is essential

Selecting Team Members

- A manager or team leader's job is to create a **cohesive** team and organize their team so that they can work together **effectively**

- This involves creating a team with the **right balance** of **technical skills** and **personalities**, and organizing that team so that the members work together effectively

Assembling a Team

- May **not** be **possible** to appoint the **ideal** people to work on a project
  - Project budget may not allow for the use of highly-paid staff
  - Staff with the appropriate experience may not be available
  - An organisation may wish to develop employee skills on a software project
- Managers **must** work within these **constraints** especially when there are **shortages** of trained staff

Team Composition

- Team composed of members who share the same motivation can be problematic
  - **Task**-oriented – everyone wants to do their own thing
  - **Self**-oriented – everyone wants to be the boss
  - **Interaction**-oriented – too much chatting, not enough work
- An effective team has a **balance** of all types
- This can be difficult to achieve as **software engineers** are often **task-oriented**
- **Interaction**-oriented people are very important as they can detect and defuse tensions that arise

Team Composition Scenario

In creating a team for **assistive technology development**, Alice is aware of the importance of selecting members with **complementary** personalities. When interviewing potential team members, she tried to assess whether they were **task**-oriented, **self**-oriented, or **interaction**-oriented. She felt that she was primarily a self-oriented type because she considered the project to be a way of getting noticed by senior management and possibly promoted. She therefore looked for one or perhaps two interaction-oriented personalities, with task-oriented individuals to complete the team

The final assessment that she arrived at was:

Alice – self-oriented
Brian – task-oriented
Bob – task-oriented
Carol – interaction-oriented
Dorothy – self-oriented
Ed – interaction-oriented
Fred – task-oriented

Team Organization

- The way that a team is organized affects the **decisions** that are made by that team, the ways that **information** is **exchanged** and the **interactions** between the **development team** and **external project stakeholders**
  - Key questions include:
    - Should the **project manager** be the **technical leader** of the team?
    - Who will be involved in making **critical technical decisions**, and how will these be made?
    - How will **interactions** with external stakeholders and senior company management be handled?
    - How can teams **integrate** people who are **not co-located**?
    - How can **knowledge** be **shared** across the team?

Team Organization

- Small software engineering teams are usually organised informally without a rigid structure
- For **large projects**, there may be a **hierarchical** structure where different **teams** are responsible for different **sub-projects**
- **Agile** development is always based around an **informal** team on the principle that formal structure inhibits **information exchange**

Informal Teams

- The team acts as a whole and comes to a consensus on decisions affecting the system
- The team leader serves as the external interface of the team but does not allocate specific work items
- Rather, **work** is discussed by the team as a whole and tasks are **allocated** according to **ability** and **experience**
- This approach is successful for teams where all members are experienced and competent

Team Communications

- Good **communications** are essential for **effective** team working
- Information must be exchanged on the **status of work**, **design decisions**, and **changes** to previous decisions
- Good **communications** also strengthens team **cohesion** as it promotes understanding

Team Communications

- Team size
  - The larger the group, the harder it is for people to communicate with other group members
- Team structure
  - Communication is better in informally structured groups than in hierarchically structured groups
- Team composition
  - Communication is better when there are different personality types
- The physical work environment
  - Good workplace organisation can help encourage communications

Team Communications

Communication Channels

2 - 1
3 - 3
4 - 6
5 - 10
6 - 15
...
10 - 45

n is team size

n * (n - 1) / 2 is the **number** of communication channels

Managing Project Conflict

What is Conflict?

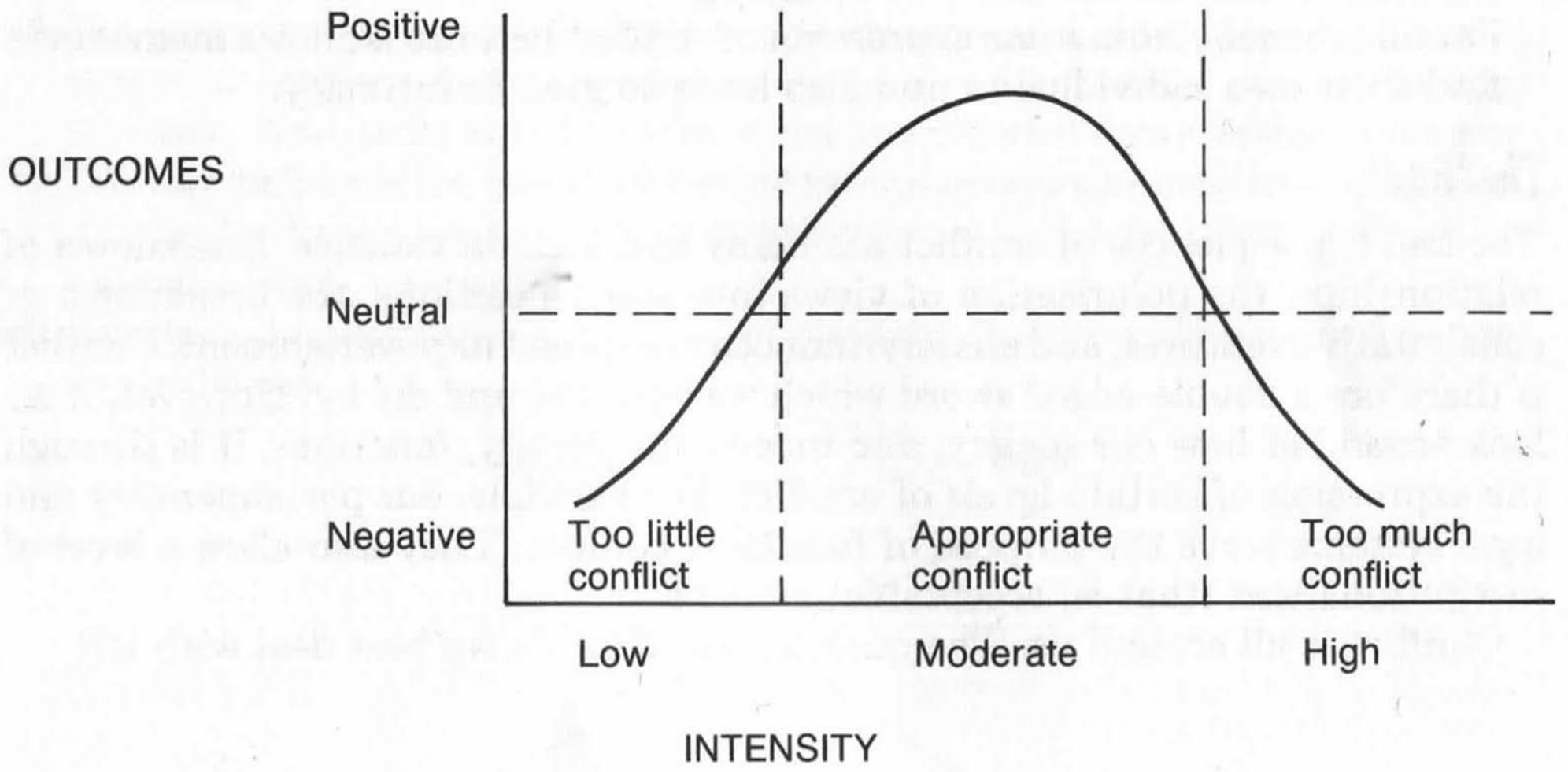
- Conflict is a form of relating or interacting where we find ourselves (either as individual or groups) under some sort of **perceived threat** to our **persona** or **collective goals** (Condliffe, 1991)

- According to Davidson et al. (2006) conflict manifests itself in a variety of ways:
  - people **compete** with one another
  - people **glare** at one another
  - people **shout** or **withdraw**

Conflict Intensity

- **Too much** or **too little** conflict can be **dysfunctional** for an organisation
- An optional level of conflict that sparks motivation, creativity, innovation and initiative can result in higher levels of performance
- If there is no conflict in the team or organisation, its members may become complacent and apathetic
- Too much conflict can produce undesirable results such as hostility and lack of cooperation, which lower performance

Conflict Intensity

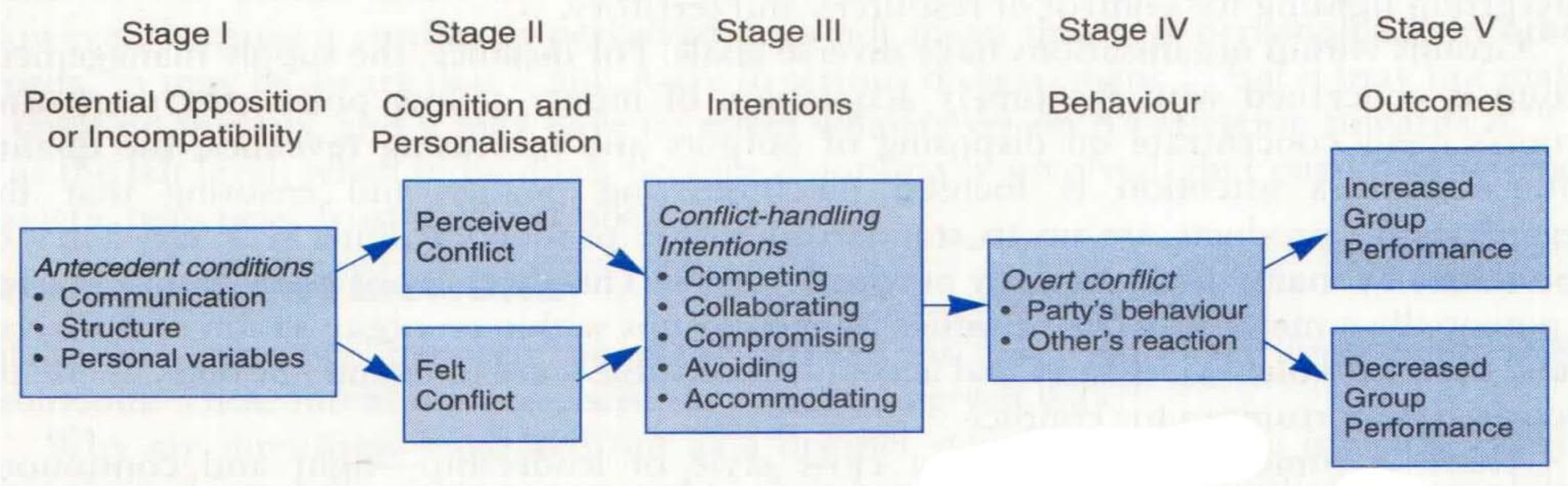


Conflict outcomes and intensity (Condliffe, 1991)

Process of Conflict

- Five Stage Model:
  - Stage I: **Potential Opposition** – Presence of conditions create opportunities for conflict to rise.
  - Stage II: **Cognition and Personalisation** – The potential for opposition becomes actualised.
  - Stage III: **Intentions** – Conflict-handling behaviours are initiated.
  - Stage IV: **Behaviour** – The conflict becomes visible.
  - Stage V: **Outcomes** – The action-reaction interplay between the conflicting parties result in consequences.

Process of Conflict



The Conflict Process

(Robbins, Millett, Cacioppe, & Waters-Marsh, 1998)

Types of Conflict

- Pinto (2010) suggests three categories of conflicts:
  - **Goal-oriented** conflict: this is associated with **disagreements** regarding results, project scope outcomes, performance specifications and criteria, and project priorities and objectives
  - **Administrative** conflict: this arises through **management hierarchy**, organisational **structure**, or company **philosophy**
  - **Interpersonal** conflict: this occurs with **personality differences** between project team members and important project stakeholders

Sources of Conflict

- There are numerous potential sources of conflict in projects.

- Common sources of **organisational** conflict include:
  - reward systems
  - scarce resources
  - uncertainty and differentiation

- Common causes of **interpersonal** conflict include:
  - faulty attributions (i.e., misconceptions around the reasons behind another's behaviour)
  - poor communication
  - holding personal grudges (Pinto, 2010)

Sources of Conflict

Sources of Conflict	Conflict Intensity Ranking	
	Thamhain & Wilemon	Posner
Conflict over project priorities	2	3
Conflict over administrative procedures	5	7
Conflict over technical opinions and performance trade-offs	4	5
Conflict over human resources	3	4
Conflict over cost and budget	7	2
Conflict over schedules	1	1
Personality conflicts	6	6

Sources of Conflict on Projects
(Pinto, 2010)

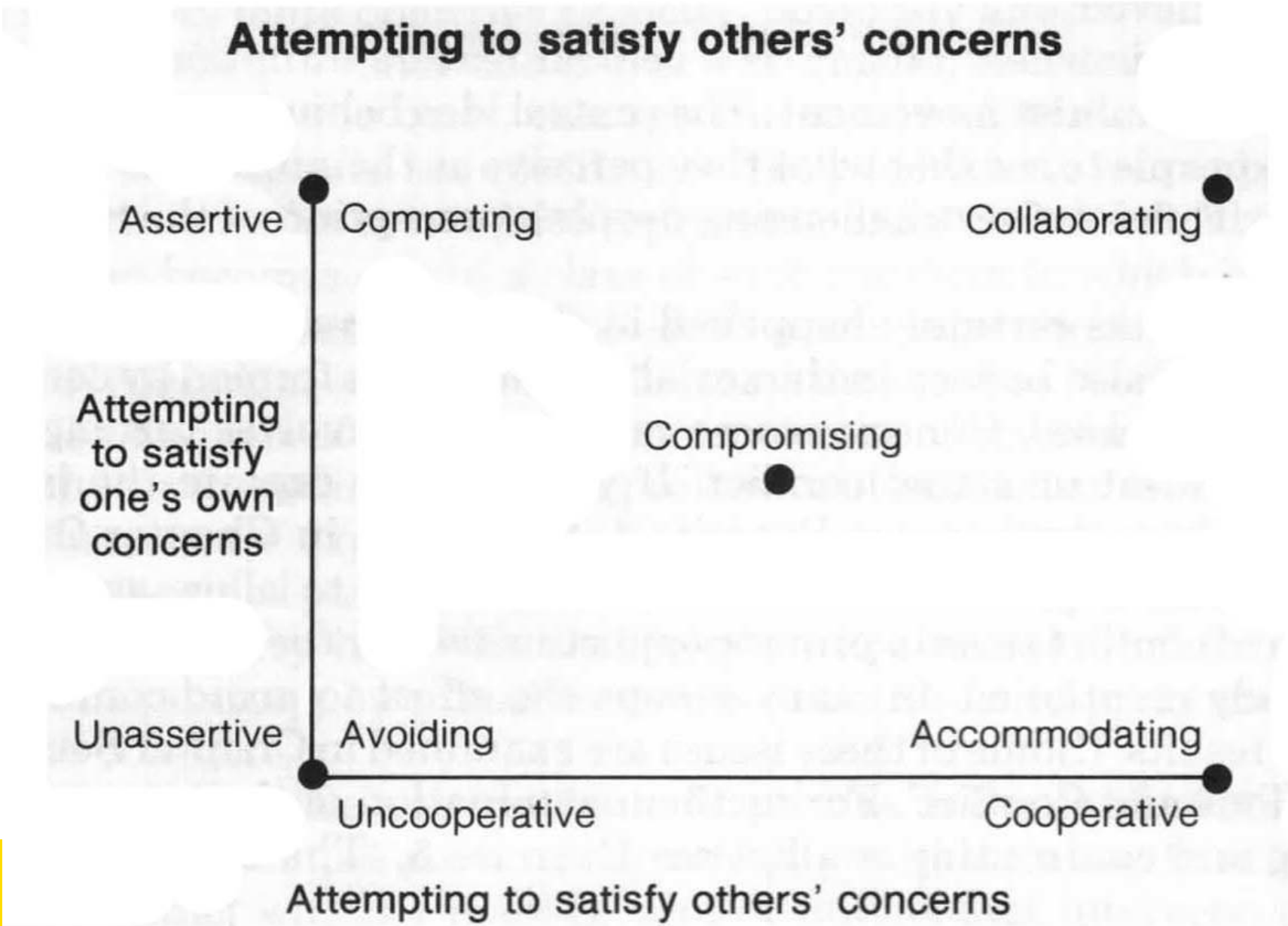
Resolving Conflict

- A number of methods for resolving group conflict are at the project leader's disposal
- It is important to consider a number of issues prior to deciding on an approach, e.g., is the conflict professional or personal in nature?
- Blake and Mouton (1964) conflict handling model:
  - **competing** (forcing): a desire to satisfy one's interests, regardless of the impact on the other parties to the conflict
  - **collaborating** (confrontation or problem-solving): a situation where the parties to a conflict each desire to satisfy fully the concerns of all parties

Resolving Conflict

- Blake and Mouton (1964) continued ...
  - **avoiding** (withdrawal): the desire to withdraw from, or suppress, a conflict
  - **compromising**: a situation in which each party to a conflict is willing to give up something of value
  - **accommodating** (smoothing): the willingness of one party in a conflict to place the opponent's interests above their own

Resolving Conflict



Dimensions of conflict handling intentions
(Condliffe, 1991)

Resolving Conflict

The five styles can be characterised in terms of the following kinds of behaviour:

Avoiding
- Ignoring conflicts and hoping that they'll go away
- Putting problems under consideration or on hold
- Invoking slow procedures to stifle the conflict
- Use of secrecy to avoid confrontation
- Appeal to bureaucratic rules as a source of conflict resolution

Compromise
- Negotiation
- Looking for deals and trade-offs
- Finding satisfactory or acceptable solutions

Competition
- Creation of win-lose situations
- Use of rivalry
- Use of power plays to get one's ends
- Forcing submission

Accommodation
- Giving way
- Submission and compliance

Collaboration
- Problem-solving stance
- Confronting differences and snaring ideas and information
- Search for integrative solutions
- Finding situations where all can win
- Seeing problems and conflicts as challenging

(Condliffe, 1991)

Resolving Conflict

Conflict mode	Situation
Competing	
	1 When quick, decisive action is vital — e g, emergencies
	2 On important issues where unpopular actions need implementing — e g, cost cutting, enforcing unpopular rules, discipline
	3 On issues vital to company welfare when you know you're right
	4 Against people who take advantage of noncompetitive behaviour
Collaborating	
	1 To find an integrative solution when both sets of concerns are too important to be compromised
	2 When your objective is to learn
	3 To merge insights from people with different perspectives
	4 To gain commitment by incorporating concerns into a consensus
	5 To work through feelings which have interfered with a relationship
Compromising	
	1 When goals are important, but not worth the effort or potential disruption of more assertive modes
	2 When opponents with equal power are committed to mutually exclusive goals
	3 To achieve temporary settlements to complex issues
	4 To arrive at expedient solutions under time pressure
	5 As a backup when collaboration or competition is unsuccessful
Avoiding	
	1 When an issue is trivial, or more important issues are pressing
	2 When you perceive no chance of satisfying your concerns
	3 When potential disruption outweighs the benefits of resolution
	4 To let people cool down and regain perspective
	5 When gathering information supersedes immediate decision
	6 When others can resolve the conflict more effectively
	7 When issues seem tangential or symptomatic of other issues
Accommodating	
	1 When you find you are wrong — to allow a better position to be heard, to learn, and to show your reasonableness
	2 When issues are more important to others than to yourself — to satisfy others and maintain cooperation
	3 To build social credits for later issues
	4 To minimise loss when you are outmatched and losing
	5 When harmony and stability are especially important
	6 To allow subordinates to develop by learning from mistakes

When to use conflict handling styles
(Condliffe, 1991)

Progressive Demo B

Progressive Demo B

- The **Progressive Demo B** and **Retrospective B** related to the **second sprint** (or **2nd and 3rd sprints** for those who chose to have **five** sprints in total) are due resp. during your **Week 8 lab time** and **Saturday 4 November 2023 @ 9pm** (**Week 8**)

- They are worth **5%** of the total marks for the course:
  - Progressive Demo B – **2.5%**
  - Retrospective B – **2.5%**

Progressive Demo B

- **Progressive Demo B** provide an opportunity to showcase user stories completed during **Sprint 2** (or **Sprints 2 and 3** for those who chose to have **five** sprints in total) and how well your team has developed functionality to support these

- The **demonstrated user stories** are shown in Jira and described, with these user stories having the correct status "**Done**" (or "**In Progress**" if acceptance criteria not yet satisfied or not yet completed) in Jira

- Your team should **demonstrate** the **functionality** used to support each **completed user story**

Progressive Demo B

One way to conduct the demo is described below:

- Use **Jira** and your **developed software** so far to do the demo
- For each story:
  - **show**, **read**, and **describe** the completed user story from **Jira**, also showing its **ideally 'Done'** status in Jira
  - walk-through and demonstrate the completed functionality described in the user story in your **developed software**

Progressive Demo B

- The progressive demonstration should not go beyond **12 minutes** and no less than **10 minutes**
- Team members **absent** for a progressive demo will receive **zero (0) mark out of 2.5** for that demo
- Not **necessarily** all team members **speak** during the progressive demo
- However, all team members should be involved in **preparing** it and being **present**

Progressive Demo B

Marking Rubric

Category	*Team Mark*	*Max Mark*
Completed user stories to be demonstrated are shown in Jira and described, with these stories having the correct status in Jira (i.e., Done)		**/1**
Demonstrates the functionality used to support each completed story		**/1**
Keep the Demo between 10 and 12 minutes		**/0.5**
Progressive Demo B Mark	**0**	**/2.5**

General Comments

Retrospective B

Retrospective B

- Retrospective B is a **reflective** activity where team members meet to think about their **teamwork process** over **Sprint 2** (or **Sprints 2 and 3** for those who chose to have **five** sprints in total)

- The team will discuss:
  - How **effective** '**things to try**' from **Retrospective A** were at improving the teamwork process
  - What **went well**
  - What **did not go so well**
  - What the team members should **try over the next sprint** to **improve** their **teamwork process**

Retrospective B

- This meeting should follow **soon after the Sprint 2 demo** (usually in the **same day**)

- **At least one team member** should be **assigned** responsibility for attempting to **enforce** or **follow up** on each **action** on the '**to try**' list

- Team members **absent** for the **retrospective meeting**, as per the brief document's members present/absent list, will **receive zero (0) mark out of 2.5** for Retrospective B

Retrospective B

- A brief retrospective report must be submitted
- The retrospective report includes:
  - A **title page**
  - A section giving **meeting details** (date, time, and members present/absent)
  - A section **outlining how effective 'things to try' from Retrospective** A were at **improving the teamwork process**
  - A section describing **what went well**
  - A section describing **what did not go well**
  - A section describing **actions 'to try' next sprint**
    - Actions must be **concrete** and **measurable**
    - Each action in the 'to try' list is **assigned at least one team member** who is responsible for attempting to enforce it or follow it up

Retrospective B

Marking Rubric

New

Category	*Team Mark*	*Max Mark*
Includes column or section describing what went well (empty section/column must have an explanation)		/0.5
Includes column or section describing what didn't go so well (empty section/column must have an explanation)		/0.5
Includes column or section describing items 'to try' next sprint (empty section/column must have an explanation)		/0.5
Includes outline of how effective 'things to try' from the previous retrospective (Retrospective A) were at improving the team work process (if this outline is empty, there should be an explanation for why it is empty)		/0.25
A team member assigned responsibility for attempting to enforce or follow up on each item on the 'to try' list (If an item in the to-try list is not assigned a member this must have an explanation)		/0.5
Title page, date, time, and team members present or absent at the Retrospective B meeting		/0.25
Retrospective B Mark	**0**	**/2.5**

Week 8 Lab Tasks

Week 8 Lab Tasks

- **Progressive Demo B** should only be between the mentor and the group presenting for F2F and Online labs (for Online labs, the break-out group/room should only have the mentor and the group presenting in it). Clients are welcome to join.
- A reminder that **Progressive Demo B** will take place during the lab and must be **live** (**not recorded**)
- Agree with your team members on a time for when you will conduct your team's **Retrospective B meeting** which should be as soon as possible after your progressive demo and preferably on the **same day**

Week 8 Lab Tasks (cont'd)

- Your presentation should **not exceed 12 minutes** and **not be less than 10 minutes** (excluding Q & A time)
- Not all members are required to speak for **Progressive Demo B**. However, all team members need to be **present during the whole demo** otherwise they will get **0/2.5 mark** for Progressive Demo B
- Have your **Retrospective B meeting** and take notes during the meeting, write your **Retrospective B report**, and submit it to Moodle by **Saturday 4 November 2023 @ 9pm** (**Week 8**)

Week 8 Lab Tasks (cont'd)

- You can demo your system using your own computer and not necessarily in the clients' environment
- However, your system needs to be **tested regularly** and **must run** in **the clients' environment** since the marking of your final project will be done in this environment
- Make sure your team schedules a meeting with the clients in Week 8 to **show them your progress** and to **get more feedback** before you start **Sprint 3**
- A reminder to also keep your individual **work diaries** up to date in **GitHub Classroom**

References

Blake, R., & Mouton, J. (1964). The managerial grid. Gulf.

Condliffe, P. (1991). Conflict management: A practical guide. RMIT TAFE.

Davidson, P., Simon, A., Gottschalk, L., Hunt, J., Wood, G., & Griffin, R. (2006). Management - Core concepts and skills. John Wiley.

Pinto, J. (2010). Project management - Achieving competitive advantage. Pearson.

Posner, B. (1986). What's all the fighting about? Conflicts in project management. IEEE Transactions on Engineering Management, 33, 207-211.

Robbins, S. P., Millett, B., Cacioppe, R., & Waters-Marsh, T. (1998). Organisational behaviour - Leading and managing in Australia and New Zealand (2nd ed.). Prentice Hall.

Saeed, T., Almas, S., Anis-ul-Haq, M., & Niazi, G. (2014). Leadership styles: Relationship with conflict. International Journal of Conflict Management, 25(3), 214 - 225.

Thamhain, H., & Wilemon, D. (1975). Conflict management in project life cycles. Sloan Management Review, 16(3), 31-50.

Q & A