TÉCNICO LISBOA

Departament of Computer Science
(MEIC & METI)

2024/2025

Communication Skills I

1º semestre
(P1)

Effective Feedback and Feedforward in Software Engineering

Prof. Cláudia Silva

In this activity, students will have the opportunity to hone their skills in effective feedback and feedforward in a context related to the world of technology. They will be divided into pairs and will have to interpret and be creative in scenarios related to the IT field, in which they will have to give and receive feedback and feedforward. The goal is to apply theoretical knowledge about effective feedback and feedforward practically, develop skills in critical analysis, and clear and concise communication, and demonstrate reflective thinking. They will also be encouraged to maintain a respectful attitude and openness to receiving feedback and feedforward, using it as an opportunity for personal and professional growth. The main objective is to practice using these techniques, develop communication skills, and promote the exchange of constructive ideas.

Learning Objectives

By the end of this task, the student should be able to:

- Apply theoretical knowledge about effective feedback and feedforward in a practical manner.
- Develop skills in critical analysis.
- Provide effective feedback by identifying strengths, positives, and negatives in a concise, clear, and objective manner.
- Utilize feedforward, offering constructive suggestions for improving those points in the future.
- Develop communication skills, expressing ideas and suggestions clearly and concisely.
- Demonstrate reflective thinking by analyzing the feedback and feedforward they receive.
- Participate in constructive discussions and exchange ideas with peers.

***Task Key Objective*:** Understand the difference and importance of both feedback (focusing on past actions) and feedforward (focusing on future improvements) in software engineering practices.

***Instructions*:**

- Find a colleague to work with. One student will play the role of the 'Developer' and the other the 'Reviewer.'
- Choose two scenarios from the list below.
- **The 'Reviewer' should provide feedback and feedforward based on the scenario, applying the techniques learned during the theoretical class.**
- After discussing each scenario, switch roles and move to the next scenario.
- Perform and present your scenario to the class briefly.
- After performing the scenario, students should be able to explain which techniques they used.

Scenarios:

1. **Bug Fixing:**
    - **Problem Situation**: **Sara**, a meticulous senior developer, noticed that the newly hired developer, **João**, successfully tackled a critical bug that had stumped the team. However, she found nothing when she sought details on the fix in their documentation system. She realized the importance of coaching Jake on the value of documentation.

2. **Code Quality:**
    - **Problem Situation:** In the bustling startup 'TechDream,' **Pedro** was known to produce results swiftly. However, **Maria**, his team lead, found it challenging to decipher his code. While it was functional, the lack of comments and unconventional methods made it a puzzle for everyone else.

3. **Late Deliverables:**
    - **Problem Situation:** Everyone in 'NexaSoft' knew **Alex** for his unparalleled creativity. But his manager, **Rachel**, also knew him for another reason: he consistently missed deadlines. His late submissions, though filled with innovation, disrupted the team's rhythm.

4. **Team Collaboration:**
    - **Problem Situation:** At 'BlueWave Technologies', **Joana**, a passionate coder, often dived

deep into her coding world, forgetting the world outside. Her teammates, while respecting her dedication, often felt disconnected, as **Joana** rarely shared her progress or challenges.

5. **Test Coverage:**
   - **Problem Situation:** **Bernardo** was a star developer in the rapid development cycle of 'QuickSoft'. But **Amy**, the QA lead, noticed a recurring issue. **Bernardo's** features, though excellent, came without any tests, making the QA process a guessing game.

6. **Code Reviews:**
   - **Problem Situation:** **Paulo**, the lead at 'FutureTech', felt uneasy. He observed that whenever **Rita**, a junior developer, reviewed code, her feedback was generic. The team needed more than just "Looks good" to ensure the software's robustness.

7. **Continuous Learning:**
   - **Problem Situation:** 'SolarTech' was a company on the cutting edge, always evolving. However, **Mike**, one of the seasoned developers, seemed stuck in his old ways. **Luciana**, a fellow developer, noticed that **Mike**'s methods, though tried and true, were falling behind the times.

8. **Performance Optimization:**
   - **Situation:** At 'Zeta Systems', **Carlos**'s code was always functional. It never failed. But when it was run through a performance audit by the system analyst, **Nuno**, it often showed inefficiencies, demanding more resources than necessary.

9. **Handling Feedback:**
   - **Problem Situation:** In the open-floor plan of 'NeuroTech', discussions were frequent. But when **Samuel**, the project lead, offered feedback to **Miguel**, the room's atmosphere would tense up. **Miguel**, though brilliant, struggled to see feedback as a tool rather than a criticism.

10. **Scalability Concerns:**
    - **Situation:** DataDynasty' was a rising startup with an increasing user base. **Emília**, the CTO, realized that while **Ronaldo**'s solutions were perfect for the present, they lacked the foresight for scalability, raising concerns for future growth.

***Debrief*:**

After you and your colleague have gone through the scenarios, regroup and discuss the importance of feedback and feedforward in the software development process with the class and instructor.