1. If the company has no previous experience with XP, then it is not recommended to adopt XP straight for a large, complicated project without prior exposure or a clear understanding of the approach. Implementing a new methodology requires careful consideration, planning, and execution. A lack of preparation could lead to significant consequences such as increased costs, missed deadlines, and quality issues.

The safer and more stable approach is to experiment XP through smaller, less mission-critical projects. By doing so, the team will better learn and get to know the aspects and practices of XP before applying it on a larger scale. Starting small can help them to understand how to develop user stories, test frequently, and work collaboratively with each other and the clients. Furthermore, through this process, the company could identify different obstacles and create strategies to deal with them. These strategies could also be reused for future projects. Additionally, key stakeholders, including senior management and IT staff, could also be involved in this process to ensure support and alignment with the new methodology.

2. Traditional project management tools, WBS in this case, work to a limited extent in XP. However, they could not reach their full potential like in traditional projects, since XP emphasizes a more collaborative approach to software development. This means a close interaction between developers and customers, facilitating the development and prioritization of user stories, which could be used as the basis for estimation, planning, and testing.

For the XP methodology, agile project management methods, which focus more on adapting to changes as they occur, are preferred. For example, XP relies on a backlog of regularly updated user stories based on customer feedback, instead of a comprehensive WBS. Thus, the team is more flexible and can quickly adapt to changes in customer requirements or priorities. For traditional methods like WBS to fully work in XP, they need to be modified with flexibility and collaboration in mind.

3. Here are methods could be used for estimation in XP:

- Planning poker: This is a collaborative estimation technique that involves both the developers and the customers. They all estimate the effort required to complete a user story. Such estimates are discussed and refined through a series of iterations until a consensus is reached.
- Expert opinion: This technique is especially useful when it comes to estimating tasks that are more complex or unfamiliar to the development team. Expert opinion involves seeking the inputs of experts who have expertise with the relevant topic, such as senior developers, solution architects, business analysts, or other stakeholders. They could provide deeper insights into the complexity and effort required for said tasks.
- Relative estimation: This technique involves comparing the complexity of each task against each other. For example, if task A is twice as complex as task B, we can estimate that task A takes twice the time to finish compared to task B.

It is important to note that we should not rely solely on one technique for effort estimation, but we must adopt a combination of them for the most correct results.

4. If the company’s developer team has no previous experience with XP, introducing this methodology will have significant impacts:

- Changes to development process: Developers who used to work individually might find it hard to adjust to pair programming and collective code ownership. On top of that, they must embrace more frequent testing and early, continuous feedback.
- New skills are needed for XP: This methodology requires developers to learn new skills to effectively work in this environment. For example, a shift in mindset from long-term projects to smaller, more changing features should be adopted. On top of that, XP-specific techniques such as test-driven development or continuous integration are necessary.
- Changes to roles and responsibilities: In traditional software development, developers usually only deal with coding and testing. For XP, they need to be more flexible and take on a multi-faceted role, participating in all aspects of the process, from design to delivery.
- Changes to corporate culture: XP emphasizes transparency, collaboration, and continuous improvement, which might require a complete overhaul of the way the organization approaches development.

Nevertheless, adopting XP is particularly beneficial in the long term: collaboration and teamwork improve morale and job satisfaction; continuous learning and growth help developers to acquire new skills and advance their careers; XP’s techniques ensure better code, fewer bugs, and faster deployment. This will generate more value and profits for the organization.