1. **Criteria should be used in choosing an appropriate requirements engineering tool.**

* Requirements traceability mechanism
* Requirements analysis mechanism
* Security and accessibility mechanism
* Portability and backend compatibility
* Configuration management approach
* Communication and collaboration mechanism
* Change management support
* Online publishing support
* Usability features such as word processor compatibility
* SRS documentation format

These evaluation dimensions can be used by engineers to compare various commercial and open-source requirements management tools prior to adoption.

1. **Are there any drawbacks to using certain tools in requirements engineering activities?**

Several studies of commercial tools have been conducted using the ISO/IEC TR 24766 framework (e.g., Carrillo de Gea et al. 2011, 2015; Daud et al. 2014). These studies have generally found that the tool market is rapidly changing and that tools are becoming increasingly complex and difficult to use. The complexity of the expensive commercial tools then creates opportunities for inexpensive tools to emerge, but don’t offer sophisticated features. Furthermore, these studies have indicated that validation functionalities such as consistency, correctness, and completeness are still lacking in most of the tools.

There are several potential drawbacks to using certain tools in requirements engineering activities. It's important to be aware of these drawbacks when selecting and using requirements engineering tools.

* **Limitations in Features and Functionality:** Some tools may not support all the necessary features or functionalities for your requirements engineering process. For example, they may not provide adequate support for certain types of requirements, certain levels of detail, certain formats and standards, certain languages and cultures, or certain stakeholder roles and perspectives.
* **Problems in Usability and User Experience:** Some tools may not be user-friendly or intuitive. This can lead to confusion, frustration, errors, inefficiencies, or resistance among the users. This can also require more time, effort, or resources for training, learning, or support.
* **Challenges in Cost and Value:** Some tools may be expensive to purchase, license, maintain, support, train, integrate, or migrate. They may also not provide sufficient value or return on investment for your project. This can be due to their limitations, problems, risks, or inefficiencies.
* **Issues in Adaptability and Scalability:** Some tools may not be flexible or scalable enough to cope with the changing and growing requirements of your project. This can limit their usefulness or longevity, and can cause disruptions, delays, or conflicts in your requirements engineering process.
* **Risks in Reviews and Ratings:** Some tools may have negative or mixed reviews and ratings from their users. This can be due to their drawbacks, flaws, bugs, shortcomings, or failures. This can also be due to their design, implementation, deployment, or usage issues.
* **Difficulties in Demo and Trial:** Some tools may not offer a demo or trial, or their demo or trial may not be representative or realistic. This can make it hard for you to evaluate or compare the tools before making a decision.

1. **When selecting an open-source tool, what characteristics should you look for?**

* **Licensing:** Open source doesn't mean free. While you have free access to the source code, you may need to comply with certain conditions depending on the license. Therefore, you should always check the license of the open-source tool and ensure it aligns with your intended use. If you plan to modify or distribute the software, make sure the license permits these actions.
* **Community Support:** Open-source tools are often supported by communities of developers. A strong, active community can be a good sign of a healthy project. Look for frequent updates, a large number of contributors, and active discussions. These signs indicate that the tool is being actively maintained and that help will likely be available if you encounter problems.
* **Documentation:** Good documentation is crucial for understanding how to use and adapt the tool. It can also provide insight into the tool's reliability and the community's commitment to supporting new users. Check if the tool has clear, comprehensive, and up-to-date documentation.
* **Stability and Maintenance:** Check the tool's update history. Frequent updates can indicate active maintenance, which can lead to better stability and security. However, if a tool is updated too frequently with breaking changes, it could cause instability in your own project.
* **Security:** Open-source tools can be more secure because their code is publicly available for scrutiny, which can lead to the early detection and fixing of security vulnerabilities. However, not all open-source projects actively manage security issues, so it's important to check the project's history of dealing with security issues.
* **Features and Functionality:** Ensure the tool provides the features and functionality you need. Consider how well it meets your requirements now and whether it can be adapted to meet your future needs.

1. **How can tools enable distributed, global requirements engineering activities? What are the drawbacks in this regard?**

Tools can enable distributed, global requirements engineering activities in several ways:

* **Collaboration:** Tools can provide a shared workspace where team members can collaborate in real time or asynchronously, regardless of their geographical locations. This can facilitate communication, coordination, and cooperation among the team members. It can also support the elicitation, modeling, analysis, traceability, verification, and documentation of requirements.
* **Standardization:** Tools can help standardize the vocabulary, formats, and procedures used in the requirements engineering process. This can reduce misunderstandings, inconsistencies, and ambiguities among the team members. It can also improve the quality, consistency, and comparability of the requirements.
* **Automation:** Tools can automate certain tasks in the requirements engineering process, such as data collection, data analysis, data visualization, report generation, and change tracking. This can save time, effort, and cost for the team members. It can also increase their productivity, efficiency, and accuracy.

1. **If an environment does not currently engage in solid requirements engineering practices, should tools be introduced?**

On one hand, tools can provide structure and support for requirements engineering activities, helping to establish good practices. They can facilitate collaboration, automate repetitive tasks, and help manage and track requirements. This can lead to improved efficiency, consistency, and quality in the requirements engineering process.

On the other hand, the introduction of tools can also present challenges. There can be a learning curve associated with using new tools, which requires time and training. If the tools are not intuitive or user-friendly, this can cause confusion and frustration among the team members. There can also be resistance to change, especially if the team members are used to their current ways of working. Furthermore, the tools may not fully meet the needs or expectations of the team members, or they may not integrate well with the other tools or systems used in the organization.

1. **What sort of problems might you find through a traceability matrix that you might not see without one?**

* **Missing Requirements:** A traceability matrix can help you identify any requirements that have not been addressed or implemented in the development or testing phase. Without a traceability matrix, these missing requirements might go unnoticed.
* **Extra Features:** Sometimes, features that were not initially specified in the requirements get implemented. While this might seem like a bonus, these extra features can consume unnecessary resources and could potentially introduce unwanted complexity or bugs. A traceability matrix can help identify these situations.
* **Impact of Changes:** If a requirement changes, it can affect multiple areas of a project, including design, code, and testing. A traceability matrix can make it easier to see the potential impact of changing a requirement, which can be difficult to assess without this overview.
* **Unnecessary Testing:** Without a traceability matrix, there might be tests that are being conducted that don't correspond to any requirements. These unnecessary tests consume resources and time that could be better spent elsewhere.
* **Insufficient Test Coverage:** A traceability matrix can highlight requirements that have insufficient test coverage. Without one, it's harder to ensure that all requirements have been adequately tested.

1. **How is AI being proposed for knowledge acquisition and representation in requirements specifications?**

* **Commonsense Knowledge Acquisition and Representation:** AI can be used to automatically extract commonsense knowledge from text and contextualize it quantitatively and qualitatively. Large-scale models, such as BERT, GPT-2, and T5, can learn to implicitly represent an abundance of commonsense knowledge from reading the web. This knowledge can then be extracted through carefully-designed language prompting, or through fine-tuning on knowledge graph tuples. This can be particularly useful in requirements engineering to understand the implicit assumptions and expectations of the stakeholders, and to ensure the alignment and coherence of the requirements.
* **Knowledge-Enhanced Machine Learning:** AI can be used to imbue systems with the ability to reason about moral value, blame, and intentionality. This can be crucial in requirements engineering to understand the ethical implications of the requirements, and to guide the decision-making and trade-off analysis in the requirements engineering process.
* **Knowledge Acquisition for Second Generation Expert Systems:** AI can be used to elicit, model, and represent human problem-solving knowledge. This can be particularly useful in requirements engineering to understand the problem domain, the problem constraints, and the problem solutions. This can also be useful to guide the elicitation, modeling, and representation of the requirements.
* **Knowledge Acquisition and Design Using Semantics and Perception for Autonomous Robots:** AI can be used to acquire and design knowledge based on semantics and perception. This can be particularly useful in requirements engineering for autonomous robots to understand their sensory inputs, their operational contexts, and their behavioral outputs. This can also be useful to guide the specification, validation, and verification of the requirements.