Requirements engineering (**RE**) is a critical aspect of software systems development, particularly **requirements elicitation**, which involves identifying and gathering requirements from various sources like stakeholders, documentation, and existing systems. This stage is crucial yet challenging because it heavily relies on **communication** and understanding between project teams and stakeholders.

Key highlights:

* **Requirements Elicitation** is complex and iterative, influenced by communication-rich activities. Many effective techniques for this process come from disciplines like **social sciences, organizational theory, and group dynamics**, rather than traditional software engineering.
* A **recent study** highlighted effective elicitation as one of the most critical practices for project success.
* The **complexity** of elicitation is due to communication barriers and misunderstandings among stakeholders, especially when participants come from different backgrounds or domains. The chosen techniques often depend on factors like **project type, system purpose, budget, and regulatory constraints**.
* **Context-specific approaches** are needed; for example, methods for a custom embedded system would differ from those for a commercial product or a web-based system.

**Definition and Importance**

Requirements elicitation is the initial stage in the **Requirements Engineering (RE)** process, focused on identifying, understanding, and communicating the needs of users and stakeholders to system developers. The process is aimed at uncovering explicit and implicit requirements and ensuring that all necessary information is gathered. This stage is critical because gathering more information than needed is preferable to missing important requirements.

**Key Activities in Requirements Elicitation**

1. **Understanding the Application Domain**:
   * This involves exploring the current environment where the system will operate, including its **political, organizational, and social aspects**. Analysts must identify existing work processes, constraints, and business goals.
2. **Identifying Sources of Requirements**:
   * Requirements can come from multiple sources, such as **stakeholders, users, subject matter experts, existing systems**, and related documentation (manuals, forms, etc.).
3. **Analyzing Stakeholders**:
   * Stakeholders include anyone affected by the system. It is important to **identify, analyze, and involve all relevant stakeholders**, including users, project sponsors, and other interested parties. Understanding their needs and priorities is essential.
4. **Selecting Techniques and Tools**:
   * Different elicitation techniques (e.g., interviews, workshops, surveys) are used depending on the project's context. **No single technique** fits all projects; the choice is often driven by the analyst’s expertise, project methodologies, or intuition.
5. **Eliciting Requirements**:
   * This involves gathering detailed requirements using selected techniques. Analysts must clarify the **scope, future processes, and user needs** to ensure the system aligns with business goals.

**Challenges and Considerations**

* Requirements elicitation is influenced by factors like **project context, organizational structure, budget, and timelines**. The process often involves multiple iterative sessions to refine requirements, with the final output typically being detailed requirements documents.
* There are **cultural and communication gaps** between stakeholders and developers that analysts must bridge.

**Roles of Requirements Engineers**

Requirements engineers play diverse roles, including:

* **Facilitators** during group sessions to guide discussions and ensure comprehensive input.
* **Mediators** to resolve conflicts and prioritize requirements from different stakeholders.
* **Documenters** responsible for capturing and validating requirements, ensuring alignment with project goals.

They may also need to take on roles like **system architect, designer, or tester** due to early-stage project needs. Validating requirements against stakeholder expectations and system objectives is crucial for project success.

**Outcome of Elicitation**

The elicitation process results in a set of requirements, often documented in **natural language and diagrams**, which serve as the foundation for subsequent development phases. This process may extend over several sessions, being both incremental and iterative to address complexities and refine details over time.

** Interviews:**

* Most traditional method for gathering requirements, dependent on the skill of the interviewer.
* Three types:
  + **Unstructured**: Free-flowing conversations, ideal for exploration but may miss critical topics.
  + **Structured**: Predefined questions, rigorous but limited in generating new ideas.
  + **Semi-structured**: Combines both, providing flexibility with some structure.

 **Questionnaires**:

* Useful in early stages for gathering data from multiple stakeholders quickly.
* Can include open or closed questions but lack depth and opportunities for clarification.
* Best used as checklists to cover fundamental topics and prepare for deeper elicitation.

 **Task Analysis**:

* A top-down approach breaking down high-level tasks into detailed steps.
* Helps understand user-system interactions and required knowledge.
* Can be resource-intensive; requires defining the needed level of detail.

 **Domain Analysis**:

* Involves reviewing existing documentation and systems to capture domain knowledge and identify reusable components.
* Useful for projects involving legacy systems.
* Often combined with other techniques like observation and interviews.

 **Introspection**:

* Analysts derive requirements based on their understanding of user needs.
* Useful as a preliminary approach but relies heavily on the analyst’s domain expertise.

 **Repertory Grids**:

* Stakeholders categorize domain entities, creating a matrix to highlight similarities and differences.
* Typically used with domain experts, but limited in expressing complex requirements.

 **Card Sorting**:

* Stakeholders group entities on cards based on their understanding, explaining the rationale behind their organization.
* Useful for clarifying and categorizing requirements but limited in detail.

 **Laddering**:

* A hierarchical method where stakeholders answer a series of prompts to organize domain knowledge.
* Effective if stakeholders can logically structure their understanding, often visualized in tree diagrams.

 **Group Work**:

* Collaborative meetings involving multiple stakeholders to promote cooperation and gather diverse input.
* Requires skilled facilitation to ensure balanced participation and avoid dominance by certain individuals.

 **Brainstorming**:

* Informal discussions among stakeholders to generate ideas rapidly.
* Focuses on free expression without critiquing ideas in detail, fostering creativity for new solutions.

**Joint Application Development (JAD)**

* Involves **stakeholders collaboratively discussing** problems and solutions.
* Focuses on **business and user needs** rather than technical details.
* Unlike brainstorming, JAD is **structured** with predefined goals and roles, led by a facilitator.
* Efficient for **quick decision-making** and resolving issues.

**2. Requirements Workshops**

* A generic term for **group meetings** aimed at gathering system requirements.
* Types include:
  + **Cross-functional** workshops with diverse stakeholders.
  + **Co-operative Requirements Capture (CRC)**, focusing on development community engagement.
  + **Focus Groups** for market analysis.
* Encourages collaborative discovery and brainstorming.

**3. Ethnography**

* Involves **studying users in their natural environment** to understand how they work.
* Useful for understanding **contextual factors** like usability and collaboration.
* Effective for identifying **social patterns** and improving existing systems but **time-consuming** and **requires skilled interpretation**.

**4. Observation**

* A common ethnographic technique where the analyst **observes users** without interfering.
* Helps analysts understand **real-world processes** but users may change behavior when observed.
* Often combined with interviews and task analysis.

**5. Protocol Analysis**

* Participants **perform a task while talking aloud** to explain their actions and thought process.
* Useful for understanding **decision-making processes** but may miss repetitive or implicit steps.
* May not fully reflect the natural way tasks are performed.

**6. Apprenticing**

* The analyst **learns by doing** under the guidance of an experienced user.
* Involves **active participation** rather than passive observation.
* Useful when the analyst is unfamiliar with the domain or when users struggle to articulate their processes.

**7. Prototyping**

* Involves creating **prototypes to gather feedback** on system functionality.
* Useful for systems with a **user interface** focus or when stakeholders are unfamiliar with potential solutions.
* Prototypes encourage **active stakeholder engagement** but may lead to attachment, making stakeholders resistant to changes.
* Includes techniques like **storyboards, throwaway prototypes, and evolutionary prototypes**.

**8. Goal-Based Approaches**

* Focus on **decomposing high-level goals** into sub-goals to derive requirements.
* Uses models like **KAOS and i**\* frameworks.
* Effective for systems where **only high-level needs are initially understood**, but errors in early goal definition can have cascading effects.

**9. Scenarios**

* Utilize **narrative descriptions** of processes to define interactions between users and the system.
* Similar to use cases but emphasize **incremental development**.
* Useful for **requirements validation** and test case creation, with structured approaches like **CREWS and Scenario Plus**.

**10. Viewpoints**

* Involves describing the system from **multiple perspectives** (e.g., operational, interface, user roles).
* Helps develop a **complete and consistent system model** and prioritize requirements.
* Effective for complex systems with interrelated entities but can be **expensive** and less suited for non-functional requirements.

**Key Takeaways:**

* **Interviews and questionnaires** are best for early-stage requirements gathering.
* **Task and domain analyses** are more detailed techniques for understanding interactions and existing systems.
* Techniques like **card sorting and laddering** help organize information but may lack depth.
* **Group work and brainstorming** promote collaboration and innovative thinking but require careful management to be effective.
* Techniques vary from **collaborative workshops (JAD, CRC)** to more **individual-focused methods (Ethnography, Protocol Analysis)**.
* **Prototyping and goal-based approaches** are ideal for systems with **unclear requirements**.
* **Scenarios and Viewpoints** provide structure and organization, ensuring all perspectives are captured.
* Each method has its **strengths and weaknesses**, often depending on the project’s complexity and stakeholder engagement level.

The **comparison of techniques and approaches** in requirements elicitation focuses on two main questions:

1. **Which techniques should be used for a specific requirements elicitation activity?**
2. **Which techniques can complement or serve as alternatives to each other?**

Each project is unique, so the selection of techniques depends on the project's context. Although exceptions may occur, the authors provide a high-level overview to guide the selection process.

Key highlights include:

* The study identifies a core set of **eight techniques** that cover a broad spectrum of elicitation approaches: **Interviews, Domain Analysis, Group Work, Ethnography, Prototyping, Goals, Scenarios, and Viewpoints**.
* **Table 2.1** outlines which techniques are best suited for specific activities like:
  + **Understanding the domain**
  + **Identifying sources of requirements**
  + **Analyzing stakeholders**
  + **Selecting techniques**
  + **Eliciting requirements**

Notably, **Interviews, Domain Analysis, and Group Work** are versatile, supporting multiple activities. Additionally, **Goal, Scenario, and Viewpoint-based approaches** are useful across various stages.

* **Table 2.2** focuses on which techniques are **complementary ("C")** and which serve as **alternatives ("A")**. For instance:
  + **Interviews** are complementary to **Goals, Scenarios, and Viewpoints**, while also serving as alternatives to **Ethnography** and **Prototyping**.
  + Using a combination of techniques, like **Prototyping with Ethnography**, can yield richer requirements data.
* The complementary techniques enhance the elicitation process by providing flexibility, especially when one technique is ineffective or unsuitable. For example:
  + If observing users directly is impractical (e.g., in hazardous environments), analysts can use **Scenarios** instead.

Ultimately, using **a mix of complementary and alternative techniques** ensures a more flexible and effective requirements elicitation process.