Week 4

CS2212
Introduction to Software Engineering

Domain Analysis

Ask Questions Live
cs1.ca/ask

Domain Analysis

- A **domain** is the targeted subject area of a computer program.
- It is the *"sphere of knowledge and activity around which the application logic revolves." - Andrew Powell-Morse*

- **Example:**
  - A desktop software program was created manage appointments for a hospital.
  - The **domain** would be the specific hospital it was created for or hospitals in general (if it was made for multiple hospitals).
  - The **software domain** would be a desktop application, specifically Appointment Scheduling Software.

Domain Analysis

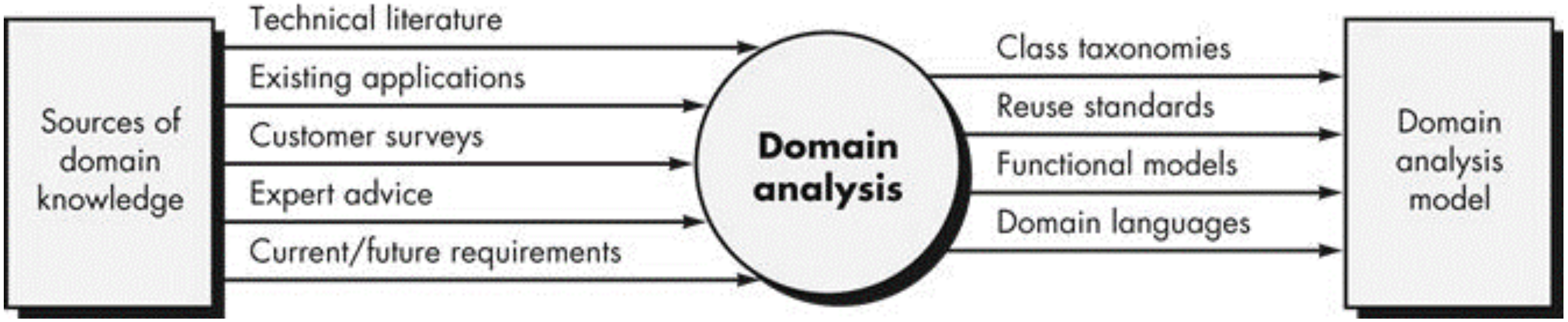
Software domain analysis is the **identification**, **analysis**, and **specification** of **common requirements from a specific application domain**, typically for **reuse** on multiple projects within that application domain.

Object-oriented domain analysis is the **identification**, **analysis**, and **specification** of **common, reusable capabilities within a specific application domain**, in terms of **common objects, classes, subassemblies, and frameworks** . . .

Domain Analysis

- A "**domain expert**" is someone who has a **deep knowledge of the domain** and can provide valuable information to an analysis.
- **Domain analysis** has numerous benefits to **requirements engineering**:
  - Accelerates development.
  - Improves communication between stakeholders.
  - Leads to a higher quality system, better meeting the needs of stakeholders.
  - Helps in anticipating extensions and changes that could be forthcoming.

Domain Analysis



The Domain Analysis Process

1. **Identify** and **define** the **domain** to be investigated.

2. Collect a **representative sample** of applications in the **domain**.

3. **Analyze** each application in the sample.

4. **Develop** a domain analysis model using this information.

For Our Project

Domain Analysis Should Answer:

- **What is the software domain for this project?**
  - Both at high level (e.g. system software, application software, embedded software, etc.) and at a more specific level (e.g. video game, GIS, word processor, etc.).
- **What do we know about the domain?**
  - Both about the software (i.e. a GIS system) and the general domain (Western University).
- **What are the common issues encountered in this domain?**
- **What are the common solutions to the above issues in this domain?**
- **How can we use this domain understanding to improve or accelerate development of this project?**

For Our Project

Minimal Example for Hospital Appointment System:

Software domain: Desktop Application (Health Information Systems)

Subdomain: Appointment Scheduling Systems

What do we know about the domain?

1. Health Information Systems are systems used in healthcare to manage patient data, appointment scheduling, and other clinical processes.
2. Appointment Scheduling Systems are a subcategory within Health Information Systems that specifically deal with scheduling appointments for patients.
3. There are regulations such as HIPAA that govern the handling of patient data and must be taken into consideration.

Common issues encountered in this domain:

1. Integration with existing systems, such as electronic medical records (EMR) systems, can be complex and time-consuming.
2. Ensuring data privacy and security of patient information is critical.
3. Scalability to handle high volume of appointments can be challenging.

Common solutions to the above issues:

1. Use of APIs and standards (such as HL7) for seamless integration with existing systems.
2. Implementation of strong security measures and adherence to privacy regulations.
3. Use of cloud-based solutions for scalability and reliability.

For Our Project

How can we use this domain understanding to improve or accelerate development of this project?

1. **Referencing best practices:** By understanding the common issues and solutions in the appointment scheduling systems subdomain, the development team can reference best practices and avoid common pitfalls. For example, they can implement strong security measures and privacy regulations to avoid data breaches, and utilize cloud-based solutions to ensure scalability.
2. **Prioritizing requirements:** Based on the common issues in the domain, the development team can prioritize the requirements for the system to address the most pressing needs. For example, integration with existing electronic medical records (EMR) systems could be a high priority to ensure a seamless experience for medical staff.
3. **Improved communication:** Understanding the appointment scheduling systems domain can help the development team communicate more effectively with stakeholders, such as hospital administrators and medical staff, as they have a shared understanding of the context and requirements of the project. This can lead to a more streamlined development process and reduced misunderstandings.
4. **Reduced development time:** By leveraging existing solutions and best practices in the domain, the development team can reduce the time and effort required to develop the system. For example, they can use APIs and standards such as HL7 for integration with EMR systems, reducing the time and effort required for integration.
5. **Better user experience:** The domain understanding can also inform the design of the system to provide a more user-friendly and efficient experience for medical staff and patients. For example, the team can design an intuitive interface for scheduling appointments and generating reports, reducing the time and effort required for these tasks.