# 軟體工程 Homework #1

## Part 1. Software Requirements

### Requirements Elicitation

Requirements elicitation is concerned with the origin of software requirements and how the software engineer can collect them. It is the first stage in building an understanding of the problem that the software is required to solve.

引出需求與軟體需求的來源以及軟體工程師如何汲取需求有關，是建構要讓軟體解決的明確問體的第一步。

#### Requirements Sources

Requirements have many sources in typical software, and it’s essential that all potential sources be identified and evaluated.

典型的軟體有許多需求來源，確認及評估潛在的需求來源是必要的

#### Elicitation Techniques

Once the requirements sources have been identified, the software engineer can start eliciting requirements information from them. It is a very difficult task and the software engineer needs to be sensitized to the fact that (for example) users may have difficulty describing their tasks, may leave important information unstated, or may be unwilling or unable to cooperate.

一旦確認需求來源，軟體工程師便可以開始引出需求。有時候軟體工程師會遇到一些比較棘手的情況，例如：使用者在表達上有困難，或者不願意或沒有辦配合及合作

### Requirements Analysis

While conceptual modeling is important, we include the classification of requirements to help inform tradeoffs between requirements (requirements classification) and the process of establishing these tradeoffs (requirements negotiation).

由於概念上的建模式重要的，我們對需求進行分類來協助了解需求之間的權衡，以及實踐這些權衡的流程

#### Requirements Classification

Requirements can be classified on a number of dimensions. Examples include the following:

需求可被分類為多個維度，例子如下：

1. Whether the requirement is functional or nonfunctional.

需求是功能性需求或非功能性需求

1. Whether the requirement is derived from one or more high-level requirements or an emergent property.

需求是從一個或多個高優先權的需求衍生的或是突發

1. Whether the requirement is on the product or the process.

需求屬產品需還或是流程需求

1. The requirement priority.

需求的優先權

1. The scope of the requirement.

需求涉及範圍

1. Volatility/stability.

異變性/穩定性

#### Conceptual Modeling

The factors that influence the choice of modeling notation include:

影響選擇模型的元素如：

1. The nature of the problem.

問題的本質

1. The expertise of the software engineer.

工程師的專業知識程度

1. The process requirements of the customer.

客戶要求的流程

#### Architectural Design and Requirements Allocation

Architectural design is the point at which the requirements process overlaps with software or systems design and illustrates how impossible it is to cleanly decouple the two tasks.

架構設計在需求流程與軟體或系統設計的點上說明了想要清楚地解偶這兩樣任務上是不可能的

Allocation is important to permit detailed analysis of requirements. Hence, for example, once a set of requirements has been allocated to a component, the individual requirements can be further analyzed to discover further requirements on how the component needs to interact with other components in order to satisfy the allocated requirements.

配置，對於詳細的分析需求是重要的，例如，一旦需求被分配至組件中，個別的需求可以再進一步的分析來發現兩個組件是如何為了滿足被分配到的需求而互相影響的進一步需求

#### Requirements Negotiation

Requirements prioritization is necessary, not only as a means to filter important requirements, but also in order to resolve conflicts and plan for staged deliveries, which means making complex decisions that require detailed domain knowledge and good estimation skills.

需求的優先權化是必須的，不僅為了篩選重要的需求，也為了解決衝突和規劃交付階段，這代表了，做複雜的決定需要有詳細的背景知識和評估技巧

### Requirements Specification

For most engineering professions, the term “specification” refers to the assignment of numerical values or limits to a product’s design goals. In software engineering, “software requirements specification” typically refers to the production of a document that can be systematically reviewed, evaluated, and approved.

對於多數工程職業來說，「specification」與數個產品所訂定目標的價值與限制所成的任務有關。在軟體工程中「software requirements specification」通常代表一個可以系統地審查、評估及認可的文件

#### System Definition Document

The document lists the system requirements along with background information about the overall objectives for the system, its target environment, and a statement of the constraints, assumptions, and nonfunctional requirements.

此文件列出系統需求便伴隨著關於系統總體目標的背景資訊，著重在環境、約束、假設以及非功能性需求

#### System Requirements Specification

Developers of systems with substantial software and non-software components—a modern airliner, for example—often separate the description of system requirements from the description of software requirements. In this view, system requirements are specified, the software requirements are derived from the system requirements, and then the requirements for the software components are specified.

重大系統的軟體和非軟體之工程師，經常將系統需求和軟體需求分開來看，以此來制定系統需求，軟體需求再從系統需求衍生而來，如此軟體組件的需求也被制定

#### Software Requirements Specification

Software requirements specification provides a realistic basis for estimating product costs, risks, and schedules. Organizations can also use a software requirements specification document as the basis for developing verification and validation plans.

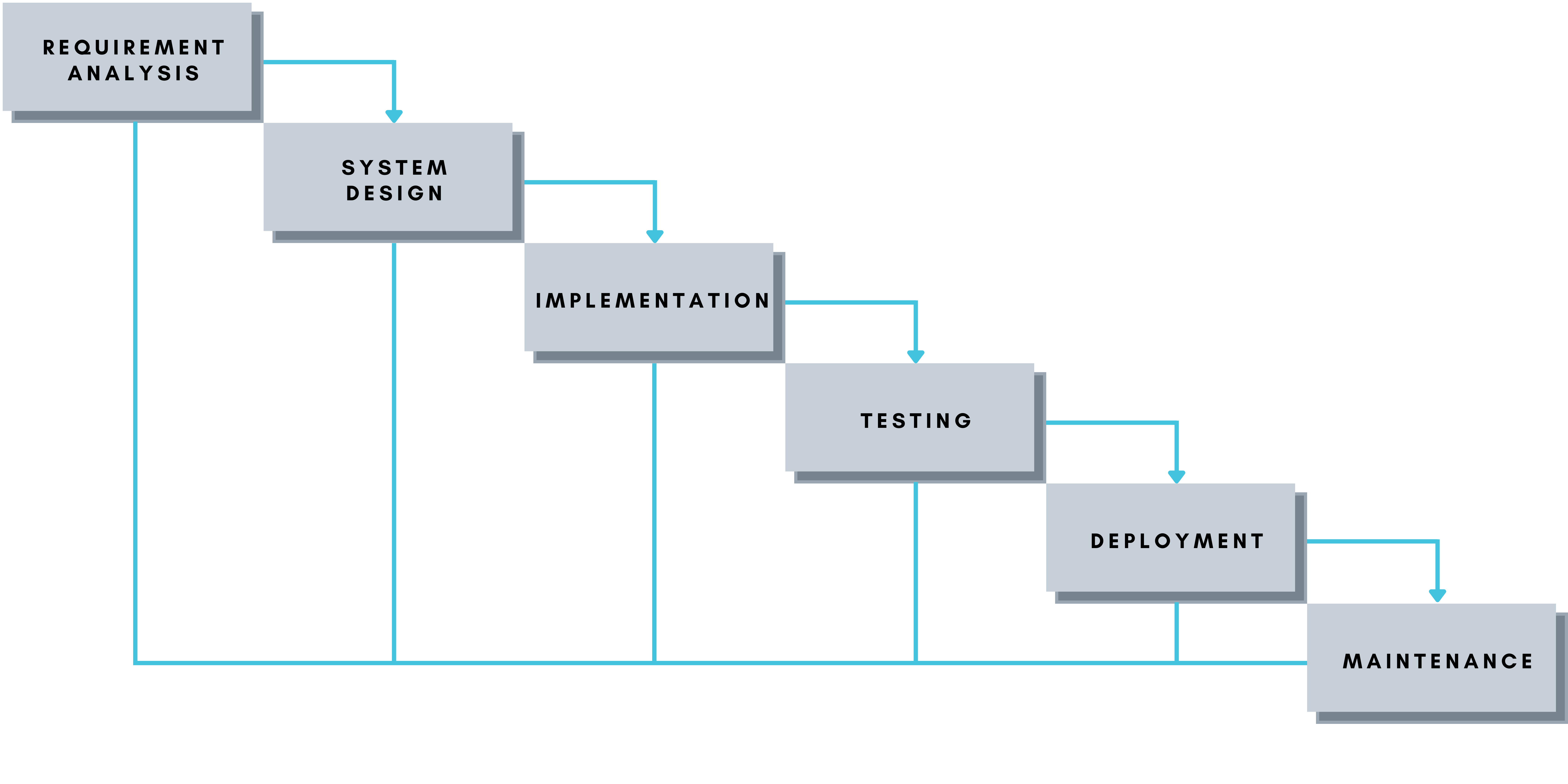
軟體需求規格提供實際的基準來估計產品花費、風險、和時程。組織也可以使用軟體需求規格書當作基礎來開發確認及驗證的計畫。

## Part 2. Process Models

### a. Describe the characteristics of the plan-driven and agile processes.

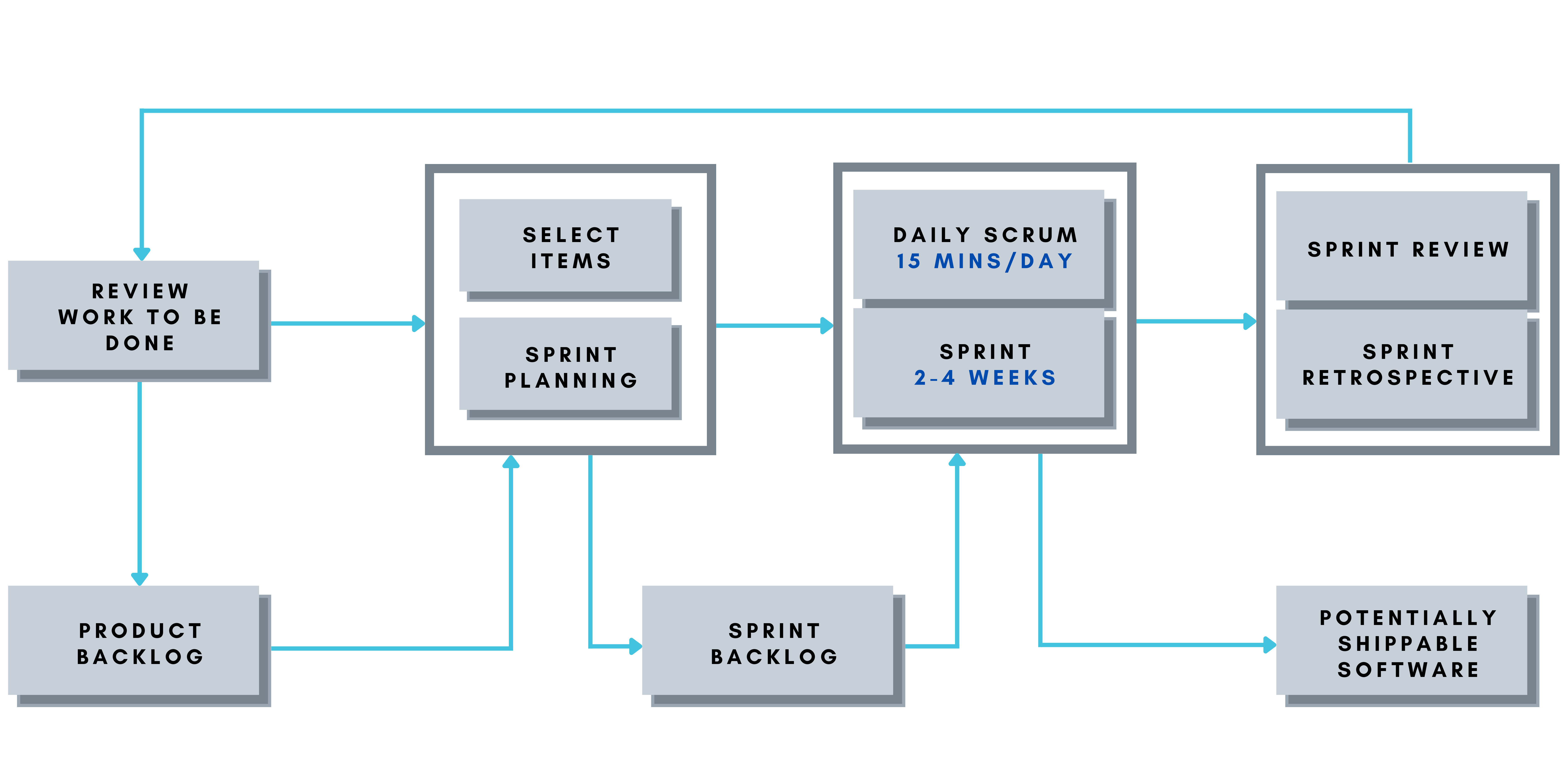
* **Plan-driven processes** are processes where all of the process activities are planned in advance and progress is measured against this plan, it also produces artifacts when finished one of the phases and before taking another. Plan driven processes will take more time and energy to the adjustments of the requirements that was changed.
* **Agile processes** build their plan incrementally; it is easier to change the process than plan driven processes, to reflect changing customer requirements, thus the cost of accommodating changing customer requirements is reduced and is easier to get customer feedback.

### b. Draw and describe the major activities of the waterfall process.



* **Requirement Gathering and analysis** − All possible requirements of the system to be developed are captured in this phase and documented in a requirement specification document.
* **System Design** − The requirement specifications from first phase are studied in this phase and the system design is prepared. This system design helps in specifying hardware and system requirements and helps in defining the overall system architecture.
* **Implementation** − With inputs from the system design, the system is first developed in small programs called units, which are integrated in the next phase. Each unit is developed and tested for its functionality, which is referred to as Unit Testing.
* I**ntegration and Testing** − All the units developed in the implementation phase are integrated into a system after testing of each unit. Post integration the entire system is tested for any faults and failures.
* **Deployment of system** − Once the functional and non-functional testing is done; the product is deployed in the customer environment or released into the market.
* **Maintenance** − There are some issues which come up in the client environment. To fix those issues, patches are released. Also to enhance the product some better versions are released. Maintenance is done to deliver these changes in the customer environment.

### c. Draw and describe the major activities of the Scrum process.



* **Sprint Planning**
  + What Happens in Sprint Planning?

During sprint planning, the entire Scrum team collaborates and discusses the desired high-priority work for the sprint and defines the sprint goal. The Scrum Master’s role is primarily to facilitate the meeting. The Product Owner describes the objective of the sprint and also answers questions from the development team about execution and acceptance criteria/criteria of satisfaction. The development team has the final say in how much of the high-priority work it can accomplish during the sprint.

* + Who attends Sprint Planning?

Sprint planning involves the entire Scrum team: the development team, Product Owner, and Scrum Master.

* + How long should Sprint Planning Last?

Sprint planning is limited to a maximum of eight hours. The general rule of thumb is to allow two hours of sprint planning for every one week of sprint length. That means teams should timebox sprint planning to four hours for a two-week sprint and eight hours for a one-month sprint.

* **Daily Scrum**
  + What Happens in a Daily Scrum?

The development team meets for 15 minutes (or less) every day of the sprint to inspect progress toward the sprint goal. They describe for each other how their own work is going, ask for help when needed, and consider whether they are still on track to meet the sprint goal. This is not a status meeting but is instead an opportunity for the development team to inspect and adapt the product and process on a daily basis.

* + Who Attends the Daily Scrum?

The mandatory participants at the daily scrum are the development team. The Scrum Master typically attends but is optional. The Product Owner is invited but doesn’t have to attend.

* **Sprint Review**
  + What Happens in a Sprint Review?

Sprint reviews focus on the product being developed, specifically on the potentially shippable product increment created during the sprint. During a sprint review, the Scrum team invites stakeholders to discuss what was completed during the sprint. They adapt the product backlog as needed based on this feedback. The Product Owner has the option to release any of the completed functionality.

* + Who Attends a Sprint Review?

The entire Scrum team attends the sprint review. Any stakeholders, senior managers, and other affected departments (e.g., marketing, customer support) are invited to attend and give feedback. Scrum teams should invite as many people as the room can hold--diverse feedback is essential for creating excellent products.

* + How Long Should Sprint Reviews Last?

Sprint reviews are limited to a maximum of four hours. The general rule of thumb is to allow one hour for sprint review every one week of sprint length. That means teams should timebox sprint review to two hours for a two-week sprint and four hours for a one-month sprint.

* **Sprint Retrospective**
  + What Happens in a Sprint Retrospective?

Sprint retrospectives focus on the process. During a sprint retrospective, the Scrum team discusses what went right and areas for improvement in the sprint. They make tangible plans for how to improve their own process, tools and relationships.

* + Difference between Sprint Reviews & Sprint Retrospectives

Sprint reviews focus on the product. Sprint retrospectives focus on the process.

* + Who Should Attend a Sprint Retrospective?

Sprint retrospectives are for the Scrum team, which would include the development team, Scrum Master, and Product Owner. In practice, product owners are recommended but not mandatory attendees.

* + How Long Should Sprint Retrospectives Last?

Sprint retrospectives are limited to a maximum of three hours. The general guidance is to allow 45 minutes for each week of sprint length. So a two-week sprint would cap the sprint retrospective at an hour and a half; a four-week sprint at three hours.

### d. Describe the differences between the plan-driven and agile.

* **Plan-Driven Model - Pros and Cons and** 
  + Advantages
    - Simple and easy to understand and use
    - Easy to manage due to the rigidity of the model. Each phase has specific deliverables and a review process.
    - Phases are processed and completed one at a time.
    - Works well for smaller projects where requirements are very well understood.
    - Clearly defined stages.
    - Well understood milestones.
    - Easy to arrange tasks.
    - Process and results are well documented.
  + Disadvantages
    - No working software is produced until late during the life cycle.
    - High amounts of risk and uncertainty.
    - Not a good model for complex and object-oriented projects.
    - Poor model for long and ongoing projects.
    - Not suitable for the projects where requirements are at a moderate to high risk of changing. So, risk and uncertainty is high with this process model.
    - It is difficult to measure progress within stages.
    - Cannot accommodate changing requirements.
    - Adjusting scope during the life cycle can end a project.
    - Integration is done as a "big-bang. at the very end, which doesn't allow identifying any technological or business bottleneck or challenges early.
* **Agile Model - Pros and Cons**
  + Advantages
    - No working software is produced until late during the life cycle.
    - High amounts of risk and uncertainty.
    - Not a good model for complex and object-oriented projects.
    - Poor model for long and ongoing projects.
    - Not suitable for the projects where requirements are at a moderate to high risk of changing. So, risk and uncertainty is high with this process model.
    - It is difficult to measure progress within stages.
    - Cannot accommodate changing requirements.
    - Adjusting scope during the life cycle can end a project.
    - Integration is done as a "big-bang. at the very end, which doesn't allow identifying any technological or business bottleneck or challenges early.
  + Disadvantages
    - No working software is produced until late during the life cycle.
    - High amounts of risk and uncertainty.
    - Not a good model for complex and object-oriented projects.
    - Poor model for long and ongoing projects.
    - Not suitable for the projects where requirements are at a moderate to high risk of changing. So, risk and uncertainty is high with this process model.
    - It is difficult to measure progress within stages.
    - Cannot accommodate changing requirements.
    - Adjusting scope during the life cycle can end a project.
    - Integration is done as a "big-bang. at the very end, which doesn't allow identifying any technological or business bottleneck or challenges early.
* **Agile Model VS Plan-Driven Model**

Agile is based on the adaptive software development methods, whereas the traditional plan-driven models like the waterfall model is based on a predictive approach. Predictive teams in the traditional plan-driven models usually work with detailed planning and have a complete forecast of the exact tasks and features to be delivered in the next few months or during the product life cycle.

Predictive methods entirely depend on the requirement analysis and planning done in the beginning of cycle. Any changes to be incorporated go through a strict change control management and prioritization.

Agile uses an adaptive approach where there is no detailed planning and there is clarity on future tasks only in respect of what features need to be developed. There is feature driven development and the team adapts to the changing product requirements dynamically. The product is tested very frequently, through the release iterations, minimizing the risk of any major failures in future.

Customer Interaction is the backbone of this Agile methodology, and open communication with minimum documentation are the typical features of Agile development environment. The agile teams work in close collaboration with each other and are most often located in the same geographical location.

## Part 3. CMMI

### What is CMMI? What is it used for?

The Capability Maturity Model Integration (CMMI) is a process and behavioral model that helps organizations streamline process improvement and encourage productive, efficient behaviors that decrease risks in software, product, and service development.

The CMMI was developed by the Software Engineering Institute at Carnegie Mellon University as a process improvement tool for projects, divisions, or organizations. The DoD and U.S. Government helped develop the CMMI, which is a common requirement for DoD and U.S. Government software development contracts. The CMMI is currently administered by the CMMI Institute, which was purchased by the ISACA in 2016.

The CMMI is designed to help improve performance by providing businesses with everything they need to consistently develop better products and services. But the CMMI is more than a process model; it’s also a behavioral model. Businesses can use the CMMI to tackle the logistics of improving performance by developing measurable benchmarks, but CMMI can also help create a structure for encouraging productive, efficient behavior throughout the organization.

### What is CMMI staged representation?

The CMMI model breaks down organizational maturity into five levels. For businesses that embrace CMMI, the goal is to raise the organization up to Level 5, the “optimizing” maturity level. Once businesses reach this level, they aren’t done with the CMMI. Instead, they focus on maintenance and regular improvements.

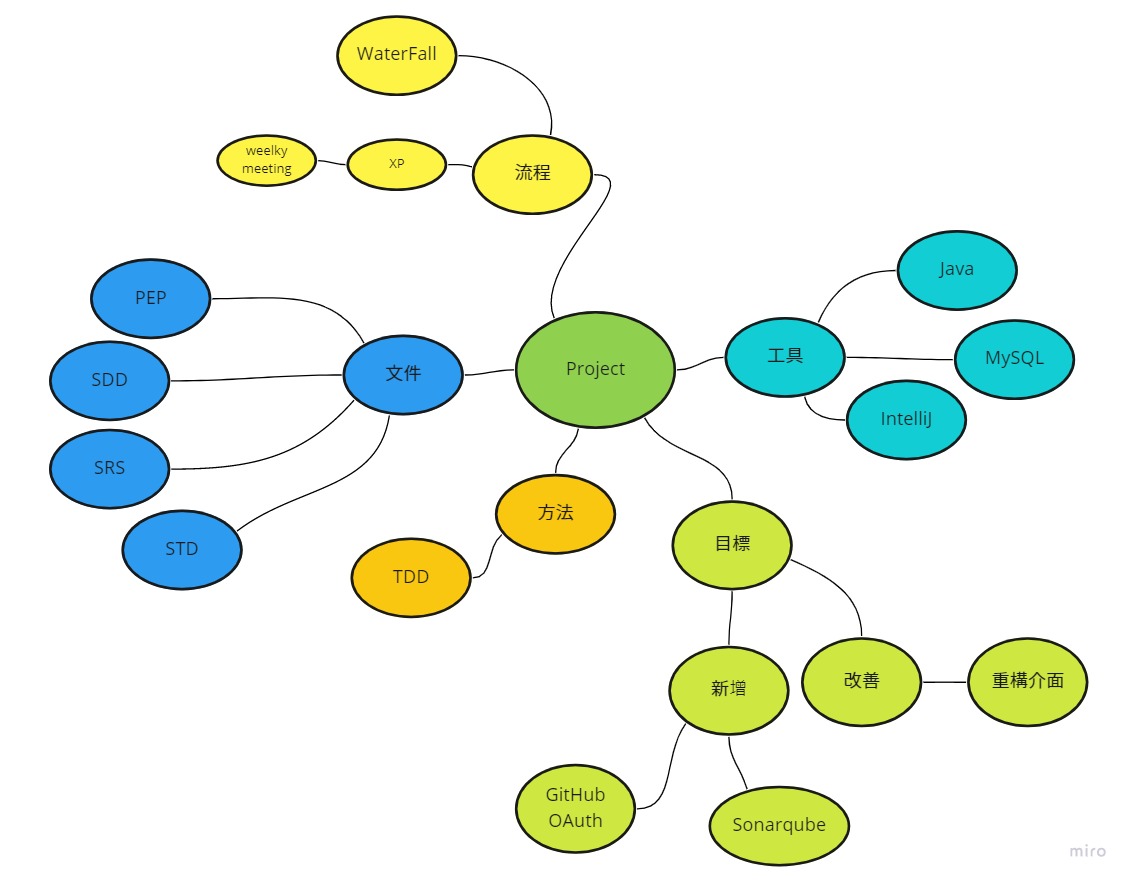
* **Maturity Level 0** – Incomplete: At this stage work “may or may not get completed.” Goals have not been established at this point and processes are only partly formed or do not meet the organizational needs.
* **Maturity Level 1** – Initial: Processes are viewed as unpredictable and reactive. At this stage, “work gets completed but it’s often delayed and over budget.” This is the worst stage a business can find itself in — an unpredictable environment that increases risk and inefficiency.
* **Maturity Level 2** – Managed: There’s a level of project management achieved. Projects are “planned, performed, measured and controlled” at this level, but there are still a lot of issues to address.
* **Maturity Level 3** – Defined: At this stage, organizations are more proactive than reactive. There’s a set of “organization-wide standards” to “provide guidance across projects, programs and portfolios.” Businesses understand their shortcomings, how to address them and what the goal is for improvement.
* **Maturity Level 4** – Quantitatively managed: This stage is more measured and controlled. The organization is working off quantitative data to determine predictable processes that align with stakeholder needs. The business is ahead of risks, with more data-driven insight into process deficiencies.
* **Maturity Level 5** – Optimizing: Here, an organization’s processes are stable and flexible. At this final stage, an organization will be in constant state of improving and responding to changes or other opportunities. The organization is stable, which allows for more “agility and innovation,” in a predictable environment.

### Explain the advantages of implementing CMMI.

**Advantages of implementing CMMI**

* **Detailed coverage of the product life cycle than other process-improvement products used alone.**
* **CMMI provides an opportunity to eliminate the stovepipes and barriers that typically exist in different parts of an organization and that typically are not addressed otherwise**
* **CMMI products incorporate lessons learnt that were learnt as a result of many other standard processes. Therefore, they inherently, handle some of the common problems faced by other process improvement approaches.**
* **Promotes collaboration between systems and software engineering.**
* **e. Allows users to choose model representation that best suits them and their objectives.**

## Part 4. Theory and Practice



Our project using both waterfall and agile process model. Our requirements are stable, and we have a weekly meeting to keep tracking weather our project is still progressing so that we can know if we are making more increments to the project.

We using tools like database, visualize application, and coding environment or software to help us do our project.

Our goal is to make an existing system better by improve its UI and tracking or comparing more information from the user repositories.