Tim Pengajar IF2150

IF2150 – Rekayasa Perangkat Lunak
Pendahuluan

SEMESTER I TAHUN AJARAN 2024/2025

Sixty years ago no one could have predicted…

- software would enable the creation of new technologies (e.g., genetic engineering and nanotechnology),
- the extension of existing technologies (e.g., telecommunications),
- the radical change in older technologies (e.g., the media);
- software would be the driving force behind the PC revolution;
- software applications would be purchased by consumers using their smart phones;
- software would slowly evolve from a product to a service as "on-demand" software companies deliver just-in-time functionality via a Web browser;
- a software company would become larger and more influential than all industrial-era companies;
- software-driven network would evolve (from library research to consumer shopping /political discourse / the dating habits).

Software

- Software is designed and built by software engineers.
- Software is used by virtually everyone in society.
- Software is pervasive in our commerce, our culture, and our everyday lives.
- Software engineers have a moral obligation to build **reliable** software that does no harm to other people.
- Software engineers view computer software, as being made up of the **programs**, **documents**, and **data** required to design and build the system.
- Software users are only concerned with whether or not software products meet their **expectations** and make their tasks **easier** to complete.

When computer software succeeds?

- when it **meets the needs of the people** who use it,
- when it **performs flawlessly** over a long period of time,
- when it is **easy to modify** and even **easier to use**

it can and does change things for the better.

When software fails?

- when its **users are dissatisfied**,
- when it is **error prone**,
- when it is **difficult to change** and even **harder to use**

bad things can and do happen

Important Questions for Software Engineers

- Why does it take **so long** to get software finished?
- Why are development **costs so high**?
- Why can’t we find all **errors** before we give the software to our customers?
- Why do we spend so much time and effort **maintaining** existing programs?
- Why do we continue to have difficulty in **measuring** progress as software is being developed?

What is software ?

- Definitions:

  Computer programs, procedures, and possibly associated documentation and data pertaining to the operation of a computer system
  *(***IEEE Standard Glossary of Software Engineering Terminology, 1990***)*

Software Characteristics

- Software is both a **product** and a **vehicle** for delivering a product (information).
- Software is **engineered** not manufactured.
- Software does **not wear out**, but it does **deteriorate**.
- Industry is moving toward **component-based software construction**, but most software is still **custom-built**.

Software Application Domains

- **System** software
- **Application** software
- **Engineering** or Scientific Software
- **Embedded** software
- **Product-line** software (includes entertainment software)
- **Web**-Applications
- **Mobile Based** Applications
- **Artificial intelligence** software

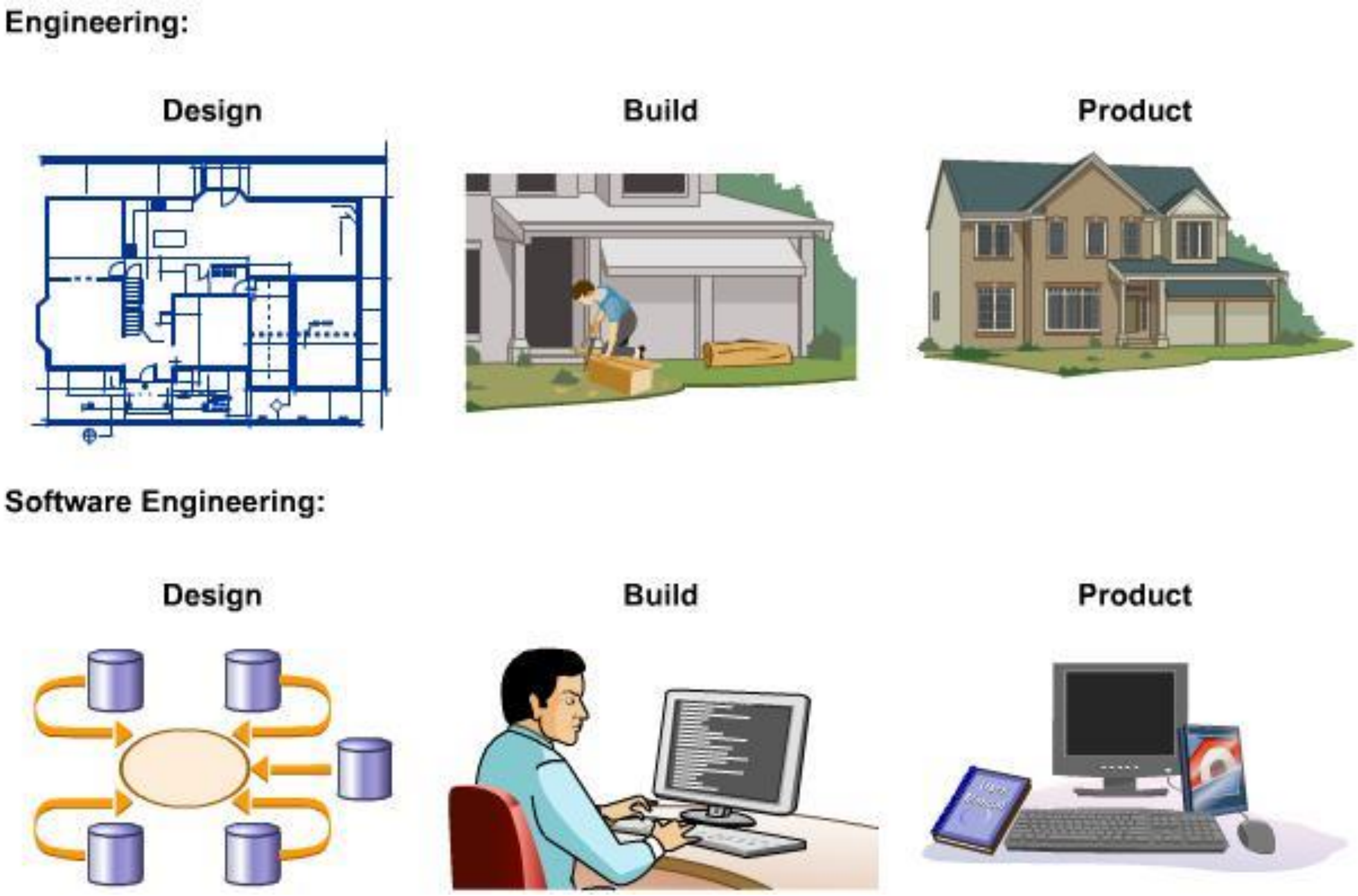
Legacy Software Evolves

- The software must be adapted to meet the needs of new computing environments or technology.
- The software must be enhanced to implement new business requirements.
- The software must be extended to make it interoperable with other more modern systems or databases.
- The software must be re-architected to make it viable within a evolving computing environment.

Software Engineering (I)

- Software engineering is the establishment of sound **engineering principles** in order to obtain **reliable** and **efficient** software in an **economical** manner.
- Software engineering is the application of a **systematic**, **disciplined**, **quantifiable** approach to the **development**, **operation**, and **maintenance** of software.
- Software engineering encompasses a **process**, **management techniques**, **technical methods**, and the **use** of tools.

Software Engineering (2)



Four broad categories of software are evolving to dominate the industry

1. Web-based systems and applications (WebApps)
2. Mobile Applications
3. Cloud computing
4. Product Line Software

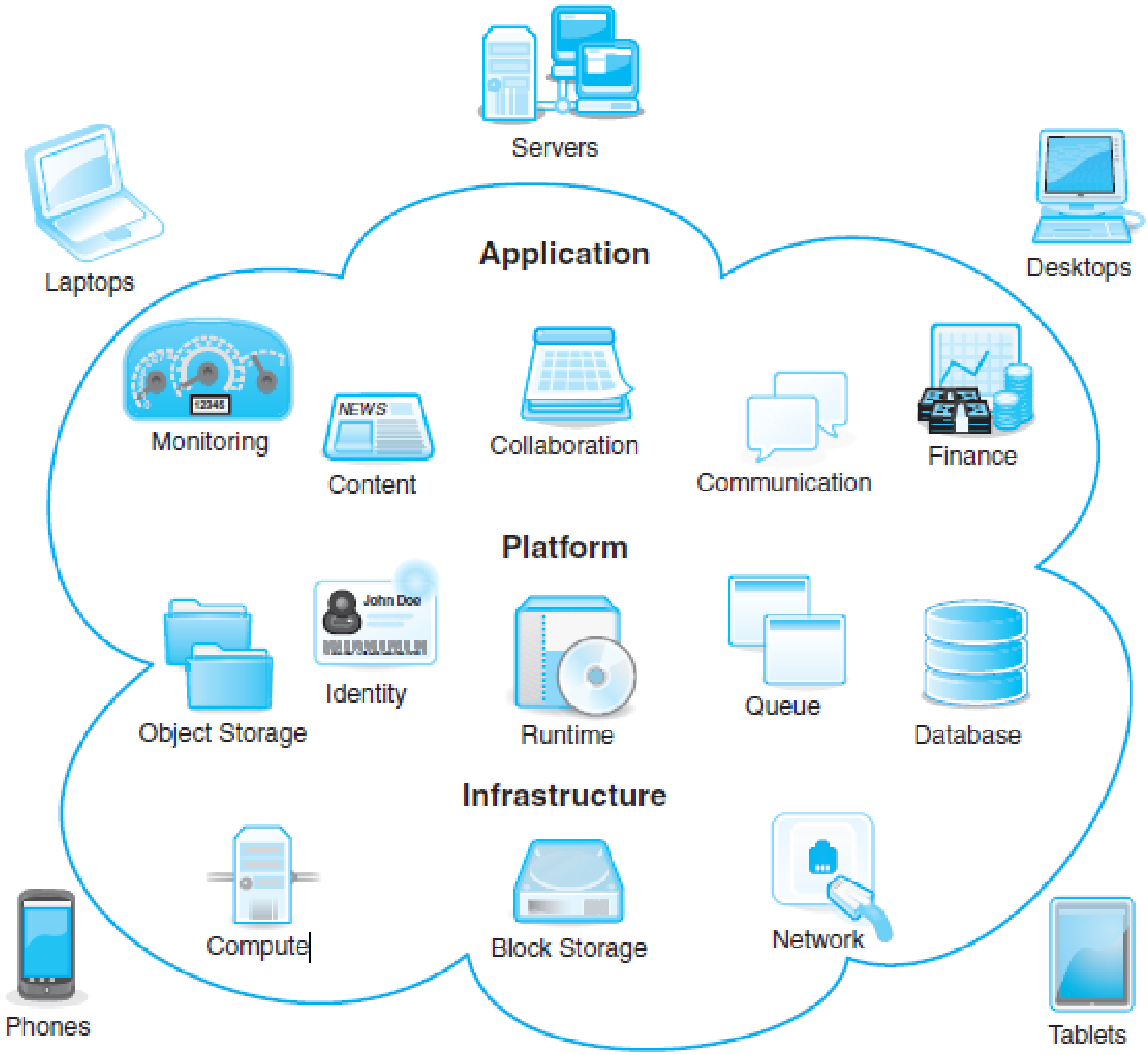
I. Web-based systems and applications

- The augmentation of HTML by development tools (e.g., XML, Java) enabled Web engineers to provide computing capability along with informational content.
- Over the past decade, Semantic Web technologies (Web 3.0) have evolved into sophisticated corporate and consumer applications that encompass "semantic databases [that] provide new functionality that requires Web linking, flexible [data] representation, and external access APIs."
- Sophisticated relational data structures will lead to entirely new WebApps that allow access to disparate information in ways never before possible.

2. Mobile Applications

- The term *app has evolved to connote software that has been specifically designed* to reside on a mobile platform (e.g., iOS, Android, or Windows Mobile).
- encompass a user interface that takes advantage of the unique interaction mechanisms provided by the mobile platform,
- interoperability with Web-based resources

3. Cloud computing

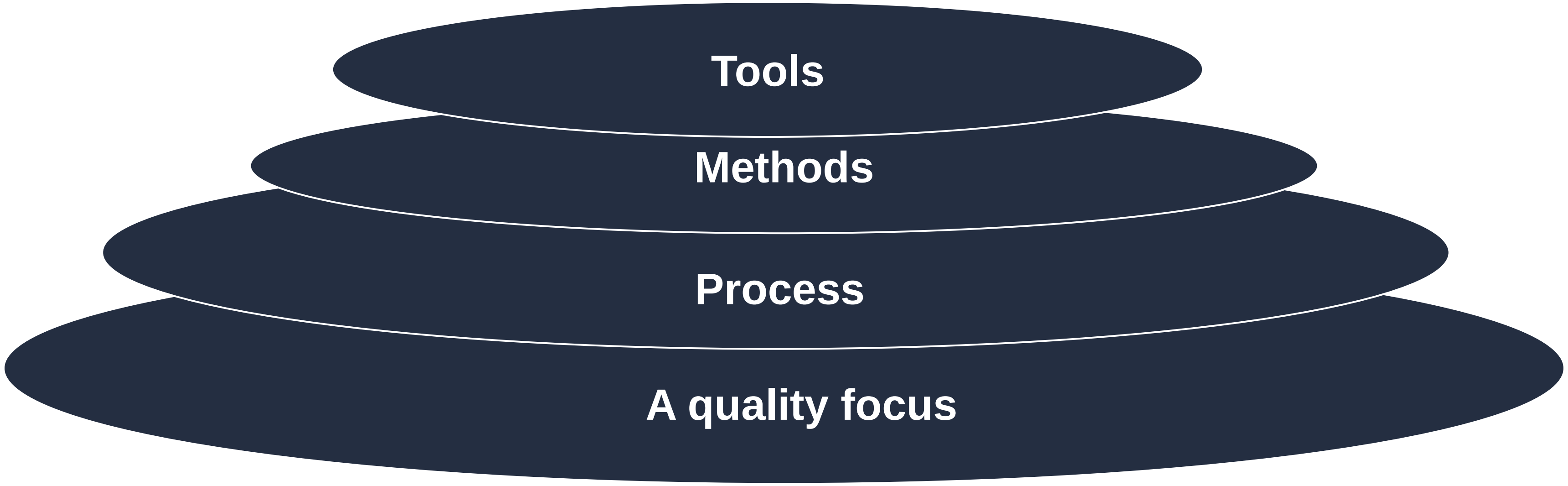


Cloud Computing

4. Product Line Software

- The Software Engineering Institute defines a *software product line as "a set of* software-intensive systems that share a common, managed set of features satisfying the specific needs of a particular market segment or mission and that are developed from a common set of core assets in a prescribed way."
- include requirements, architecture, design patterns, reusable components, test cases, and other software engineering work products

Software Engineering – a layered technology



Tools

Methods

Process

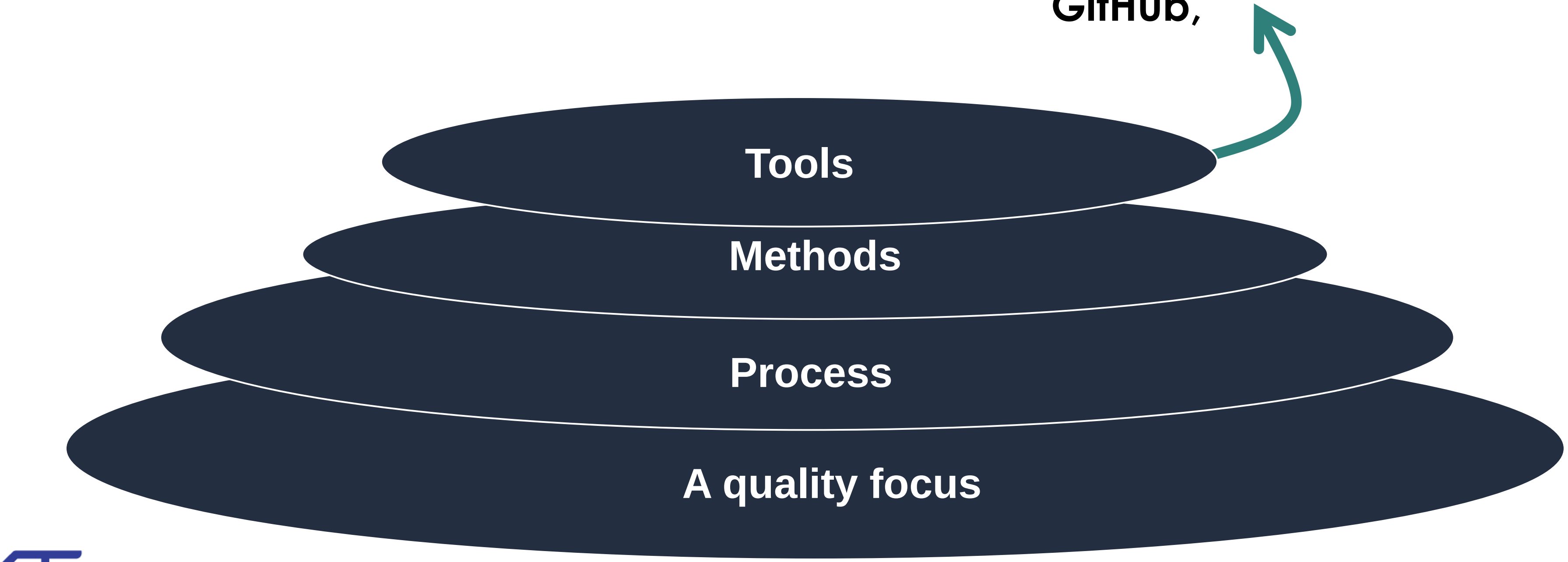
A quality focus

Software Engineering – a layered technology (2)

- The foundation for software engineering is the **process** *layer,* defines a framework that must be established for effective delivery of software engineering technology.
- Software engineering ***methods*** *provide the technical how-to's for building* software.
- Software engineering ***tools*** *provide automated or semi-automated support* for the process and the methods
  - *computer-aided software engineering* : e.g Rational Rose; various IDE (Integrated Development Environment) such as: VisualStudio, Eclipse, NetBeans; Software version, such as: CVS, SVN, and GitHub

Software Engineering – a layered technology (3)

- CASE Tool: Rational Rose,
- IDE (Integrated Development Environment): VisualStudio, Eclipse, NetBeans
- Versioning Software: CVS, SVN, GitHub,



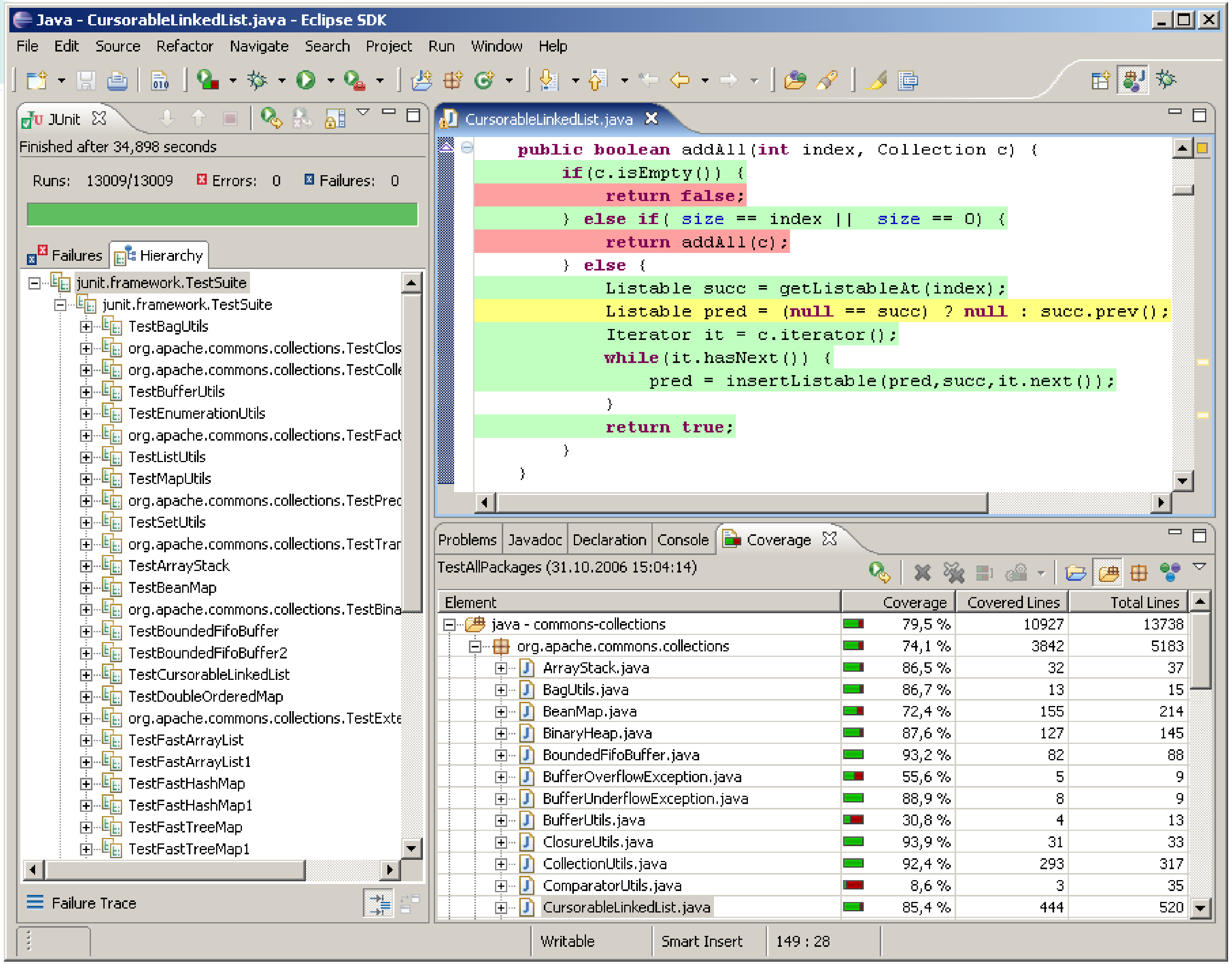
CASE tools (Computer-Aided Software Engineering)

- Software systems that are intended to provide automated support for software process activities
- CASE systems are often used for method support
- Upper-CASE
  - Tools to support the **early process** activities of **requirements** and **design**
- Lower-CASE
  - Tools to support **later activities** such as **programming**, **debugging** and **testing**

***** Software Engineering 7th ed, Ian Sommerville***

KNOWLEDGE & SOFTWARE ENGINEERING

Case Tools – example



Java - CursorableLinkedList.java - Eclipse SDK
File Edit Source Refactor Navigate Search Project Run Window Help
JUnit
Failures
Hierarchy
Failure Trace
CursorableLinkedList.java
Problems
Javadoc
Declaration
Console
Coverage

Finished after 34,898 seconds

Runs: 13009/13009 Errors: 0 Failures: 0

```
junit.framework.TestSuite
  junit.framework.TestSuite
    TestBagUtils
    org.apache.commons.collections.TestClos
    org.apache.commons.collections.TestColle
    TestBufferUtils
    TestEnumerationUtils
    org.apache.commons.collections.TestFact
    TestListUtils
    TestMapUtils
    org.apache.commons.collections.TestPred
    TestSetUtils
    org.apache.commons.collections.TestTrar
    TestArrayStack
    TestBeanMap
    org.apache.commons.collections.TestBina
    TestBoundedFifoBuffer
    TestBoundedFifoBuffer2
    TestCursorableLinkedList
    TestDoubleOrderedMap
    org.apache.commons.collections.TestExte
    TestFastArrayList
    TestFastArrayList1
    TestFastHashMap
    TestFastHashMap1
    TestFastTreeMap
    TestFastTreeMap1
```

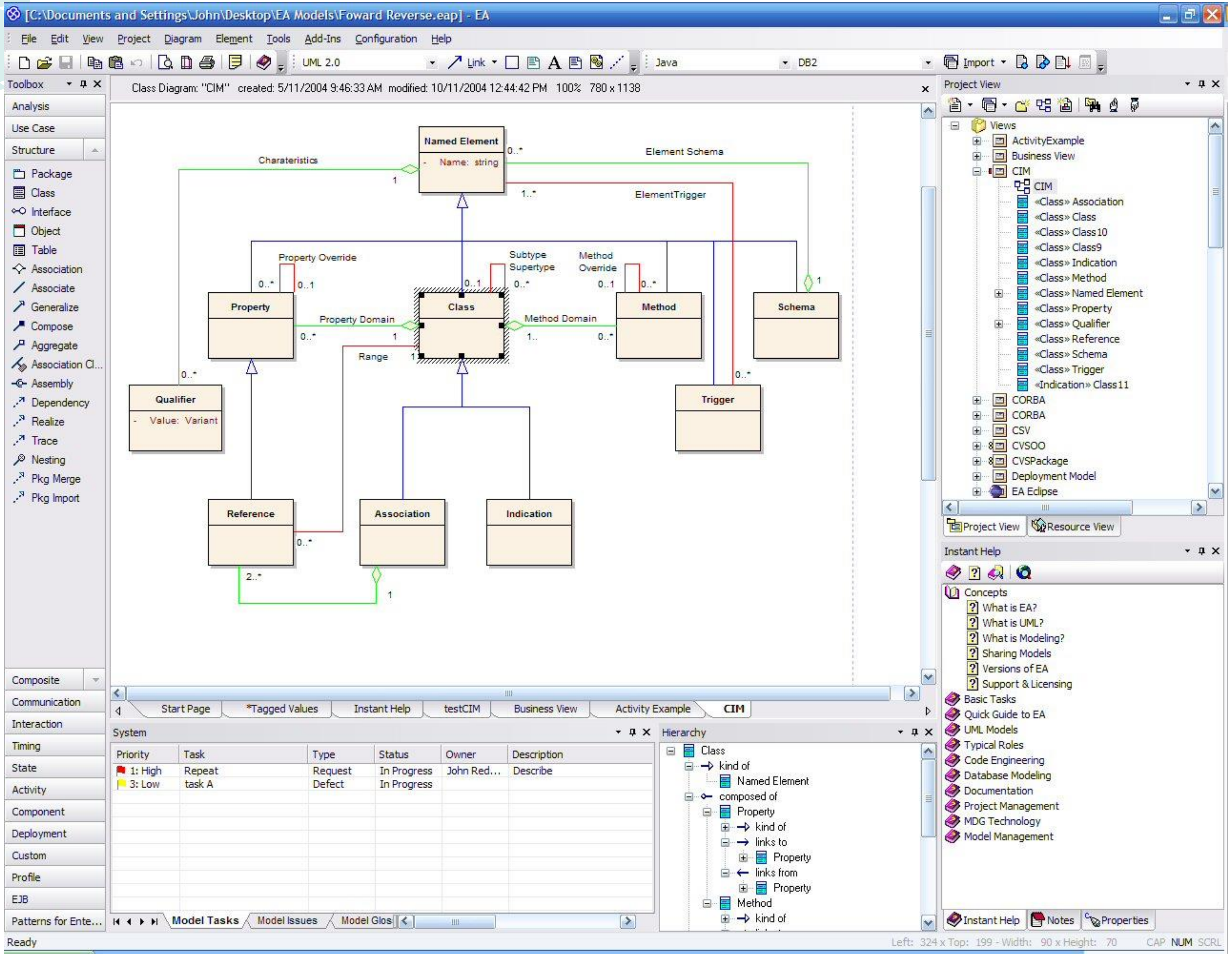
```java
    public boolean addAll(int index, Collection c) {
        if(c.isEmpty()) {
            return false;
        } else if( size == index ||  size == 0) {
            return addAll(c);
        } else {
            Listable succ = getListableAt(index);
            Listable pred = (null == succ) ? null : succ.prev();
            Iterator it = c.iterator();
            while(it.hasNext()) {
                pred = insertListable(pred,succ,it.next());
            }
            return true;
        }
    }
```

TestAllPackages (31.10.2006 15:04:14)

Element	Coverage	Covered Lines	Total Lines
java - commons-collections	79,5 %	10927	13738
org.apache.commons.collections	74,1 %	3842	5183
ArrayStack.java	86,5 %	32	37
BagUtils.java	86,7 %	13	15
BeanMap.java	72,4 %	155	214
BinaryHeap.java	87,6 %	127	145
BoundedFifoBuffer.java	93,2 %	82	88
BufferOverflowException.java	55,6 %	5	9
BufferUnderflowException.java	88,9 %	8	9
BufferUtils.java	30,8 %	4	13
ClosureUtils.java	93,9 %	31	33
CollectionUtils.java	92,4 %	293	317
ComparatorUtils.java	8,6 %	3	35
CursorableLinkedList.java	85,4 %	444	520

Writable | Smart Insert | 149 : 28

Case Tools – example



[C:\Documents and Settings\John\Desktop\EA Models\Foward Reverse.eap] - EA

Class Diagram: "CIM" created: 5/11/2004 9:46:33 AM modified: 10/11/2004 12:44:42 PM 100% 780 x 1138

System

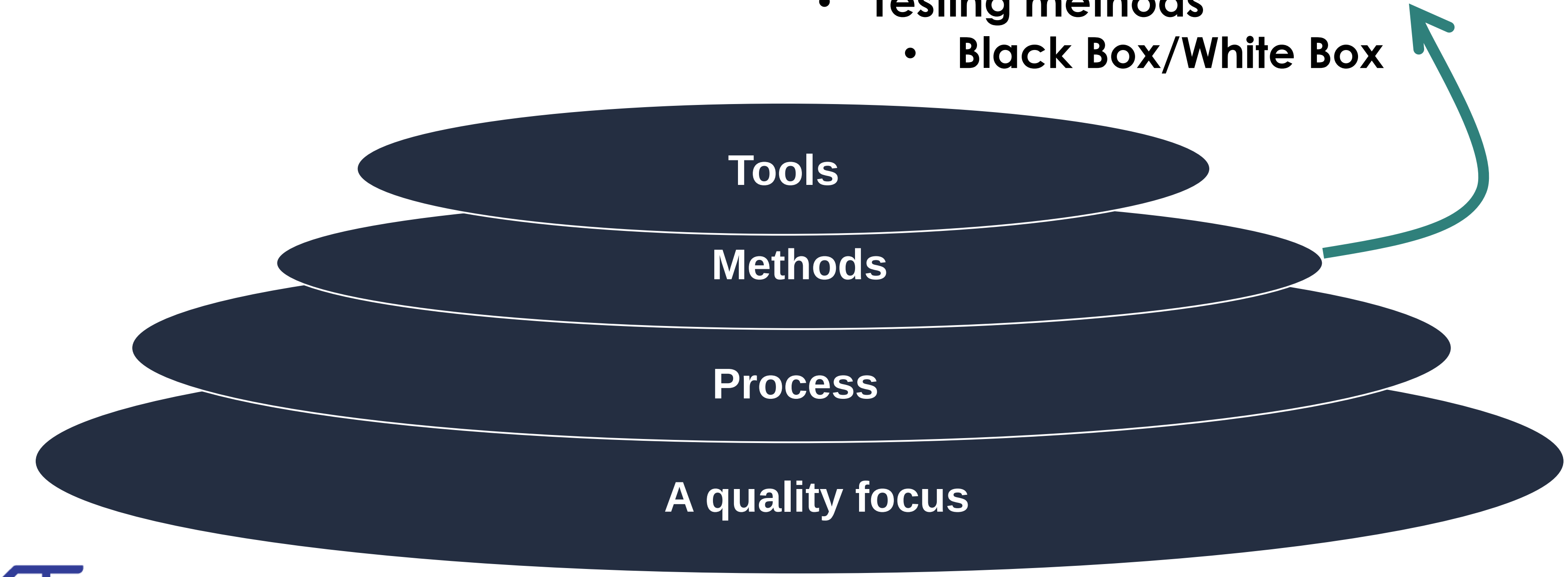
Priority	Task	Type	Status	Owner	Description
1: High	Repeat	Request	In Progress	John Red...	Describe
3: Low	task A	Defect	In Progress		

Ready

Left: 324 x Top: 199 - Width: 90 x Height: 70 CAP NUM SCRL

Software Engineering – a layered technology (4)

- Requirement gathering methods
  - Goal Oriented, Viewpoints, etc
- Analysis methods
  - Structured/OO
- Design methods
  - Structured/OO
- Testing methods
  - Black Box/White Box



What are software engineering methods?

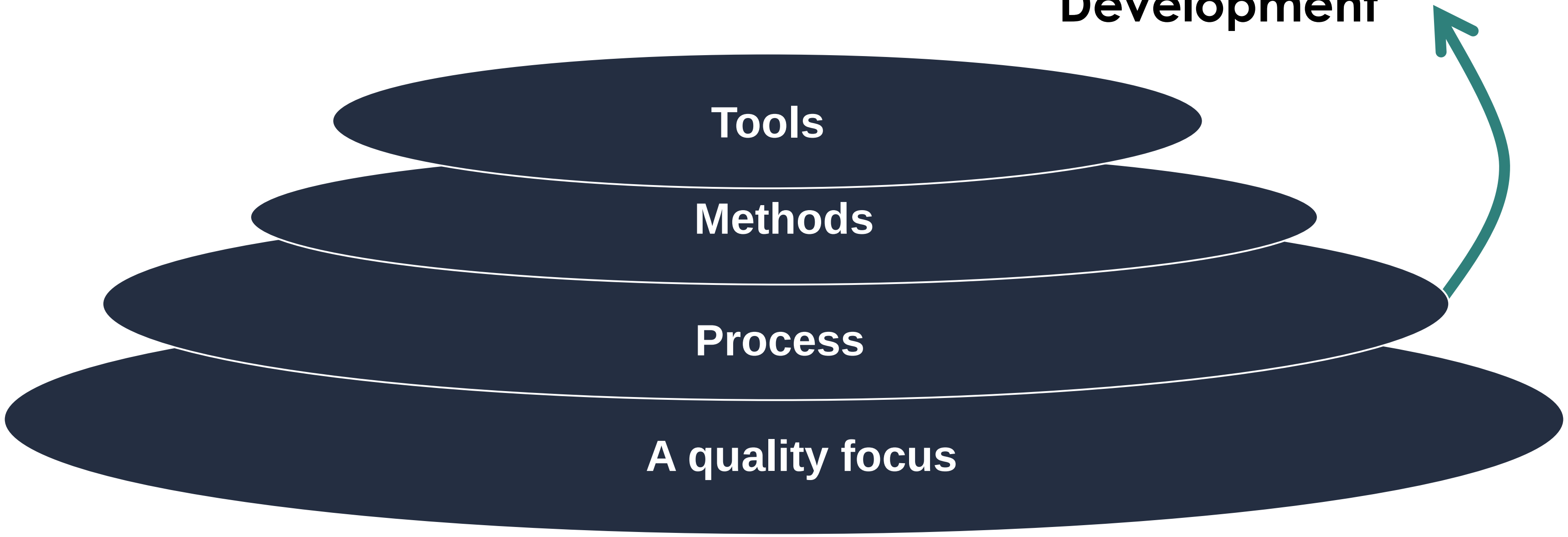
- Structured approaches to software development which include system models, notations, rules, design advice and process guidance.

- Model descriptions
  - Descriptions of graphical models which should be produced
- Rules
  - Constraints applied to system models
- Recommendations
  - Advice on good design practice
- Process guidance
  - What activities to follow

***** Software Engineering 7th ed, Ian Sommerville***

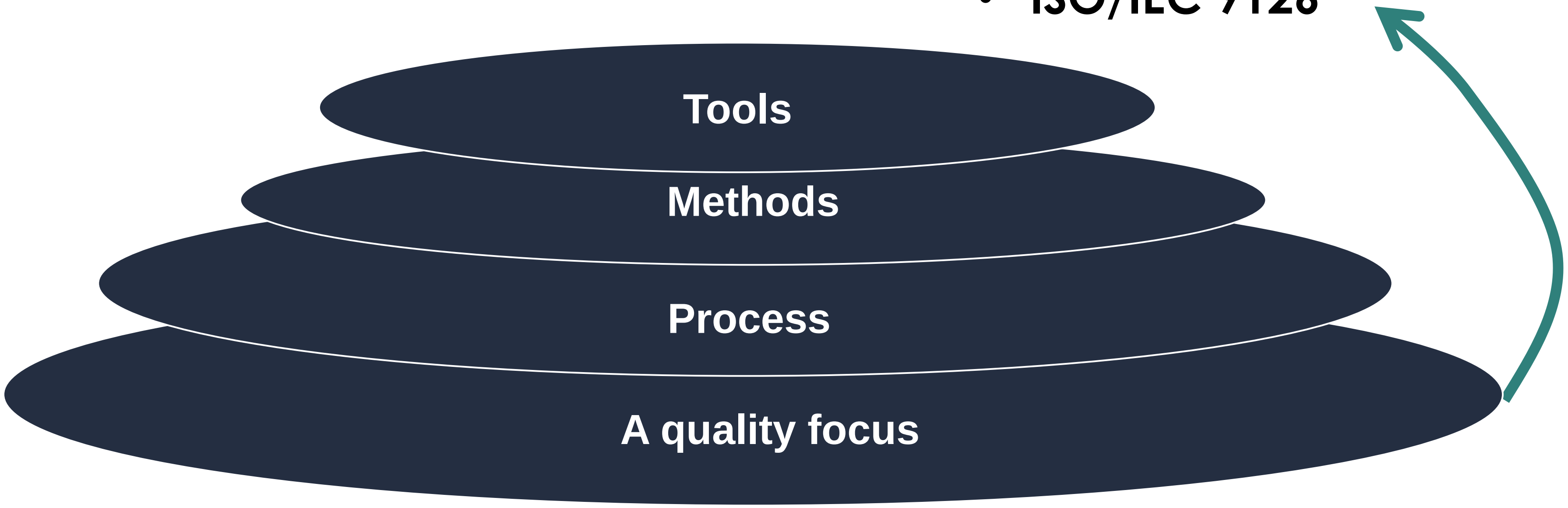
Software Engineering – a layered technology (5)

- **Waterfall Model**
- **Incremental Model/Incremental Process**
- **Spiral model**
- **Agile Development**
- **Rapid Application Development**



Software Engineering – a layered technology (6)

- Six Sigma
- Total Quality Management
- CMM (Capability Maturity Model)
- ISO/IEC 9126



Tools

Methods

Process

A quality focus

System Engineering vs Software Engineering

System – Definition

Webster's Dictionary

- A set or arrangement of things so related as to form a unity or organic whole
- A set of facts, principles, rules, etc., classified and arranged in an orderly form so as to show a logical plan linking the various parts
- A method or plan of classification or arrangement
- An established way of doing something; method; procedure….
- …..
- ….

Computer-Based Systems [PRE2007]

- *A set or arrangement of elements that are organized to accomplish some predefined goal by **processing information***
- The goal:

  To support some business function or to develop a product that can be sold to **generate business revenue**
- To accomplish the goal, a computer-based system makes use of a variety of system elements

KNOWLEDGE & SOFTWARE ENGINEERING

Computer-Based System Elements

- Software
- Hardware
- People
- Data
- Documentation
- Procedures

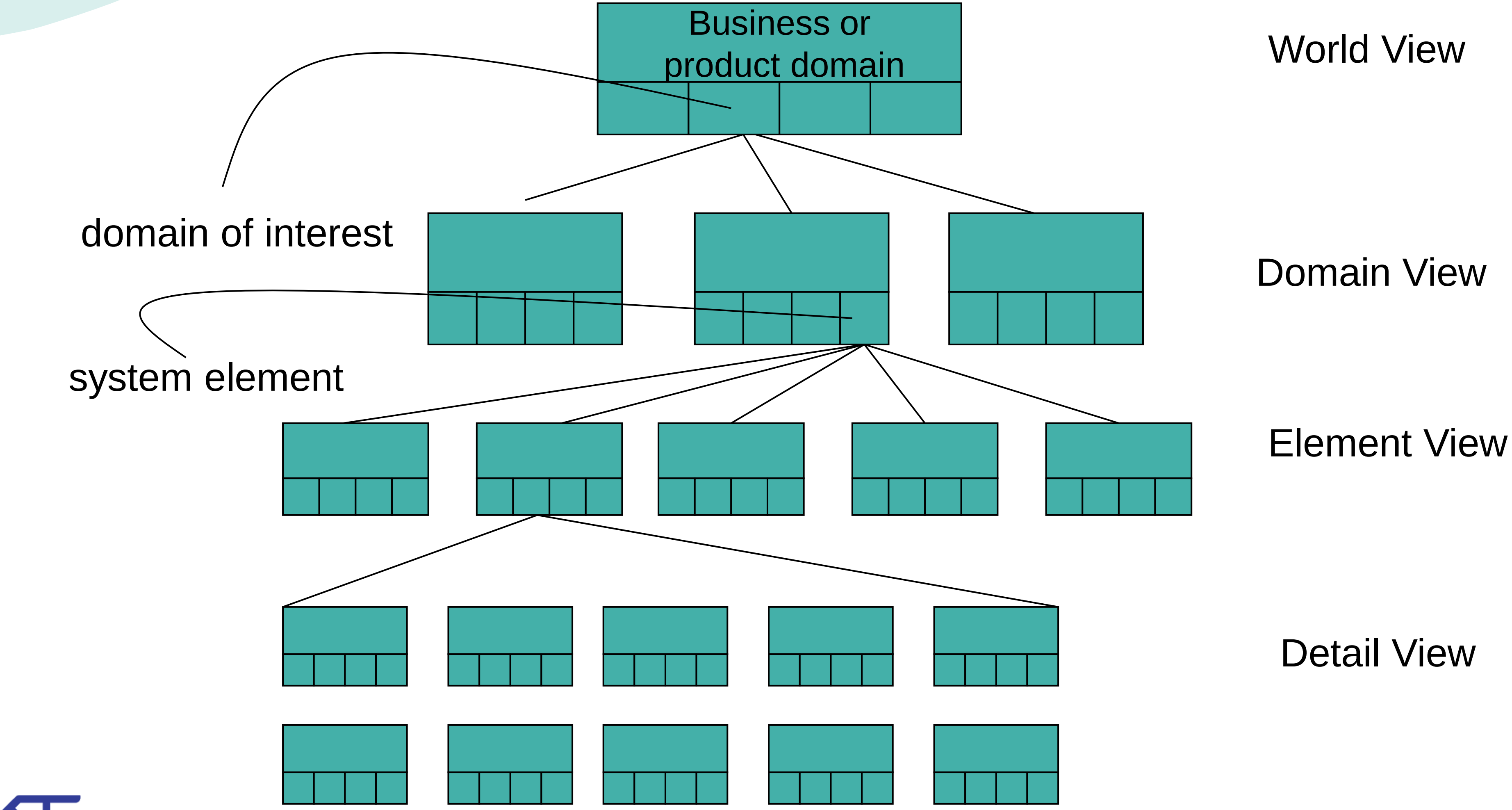
***** SEPA 6th ed, Roger S. Pressman**

System Engineering Hierarchy

- World view → WV = $\{D_1, D_2, D_3, \ldots, D_n\}$
  - Composed of a set of domains (D_i) which can be each be a system or system of systems
- Domain view → DV = $\{E_1, E_2, E_3, \ldots, E_m\}$
  - Composed of specific elements (E_j) each of which serves some role in accomplishing the objective and goals fo the domain or component
- Element view → EV = $\{C_1, C_2, C_3, \ldots, C_k\}$
  - Each element is implemented by specifying the technical component (C_k) that achieve the necessary function for an element
- Detail view

**** SEPA 6th ed, Roger S. Pressman***

System Engineering Hierarchy

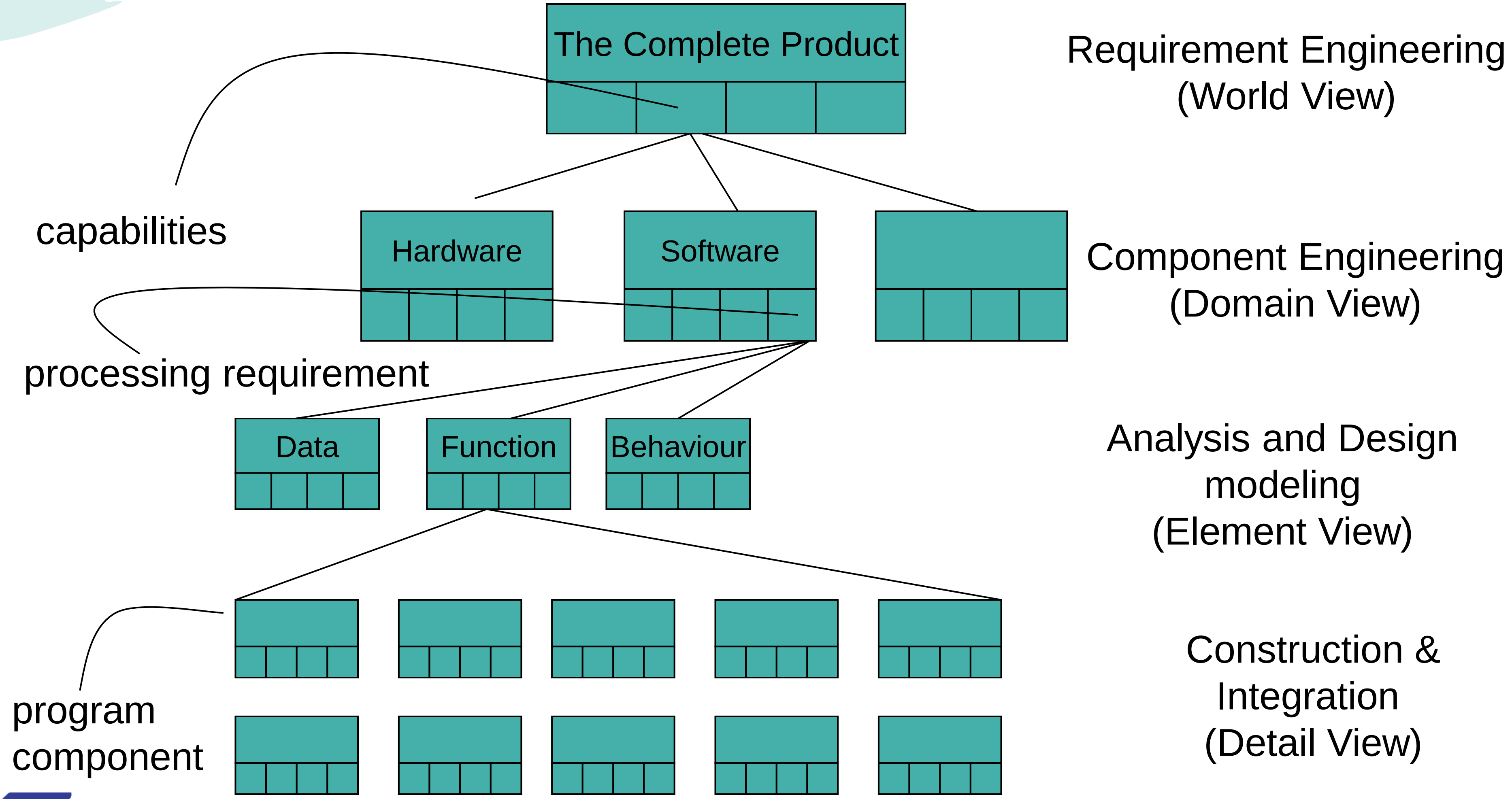


Product Engineering

- Goal
  - to translate the customer's desire for a set of defined capabilities into a working product
- Hirarchy
  - Requirements engineering (world view)
  - Component engineering (domain view)
  - Analysis and Design modeling (element view - software engineers)
  - Construction and Integration (detailed view - software engineers)

***** SEPA 6th ed, Roger S. Pressman**

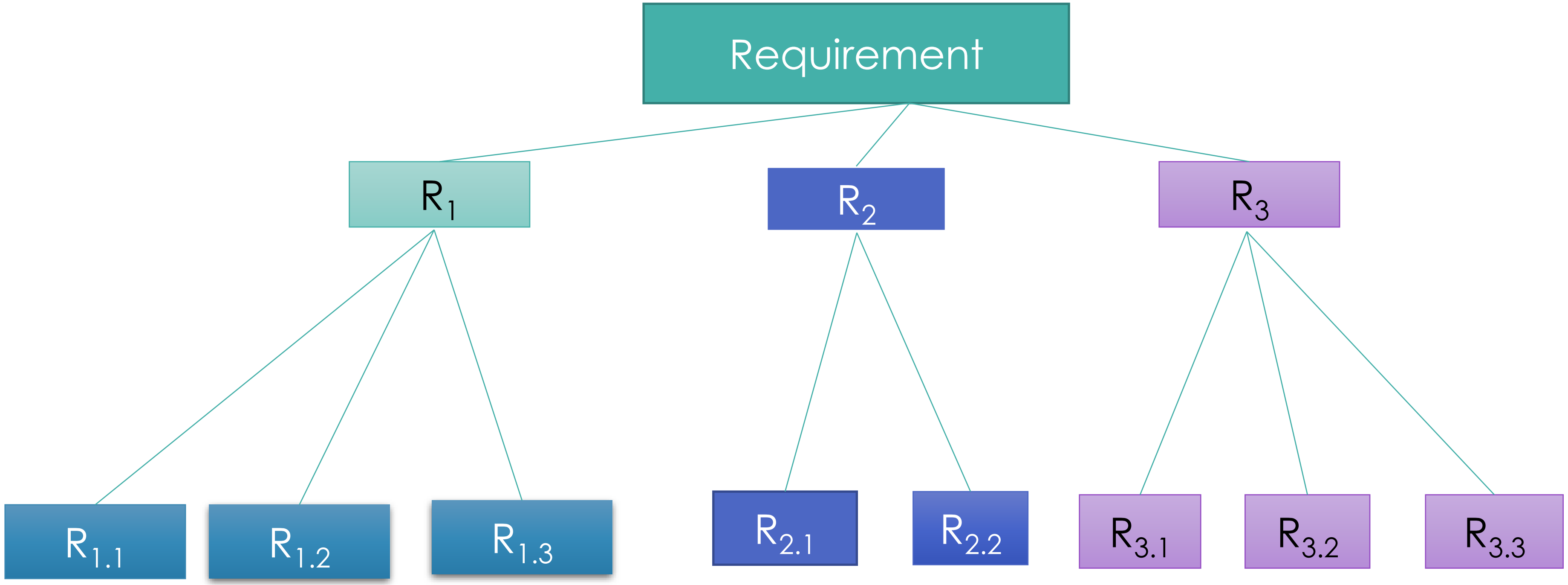
The Product Engineering Hierarchy



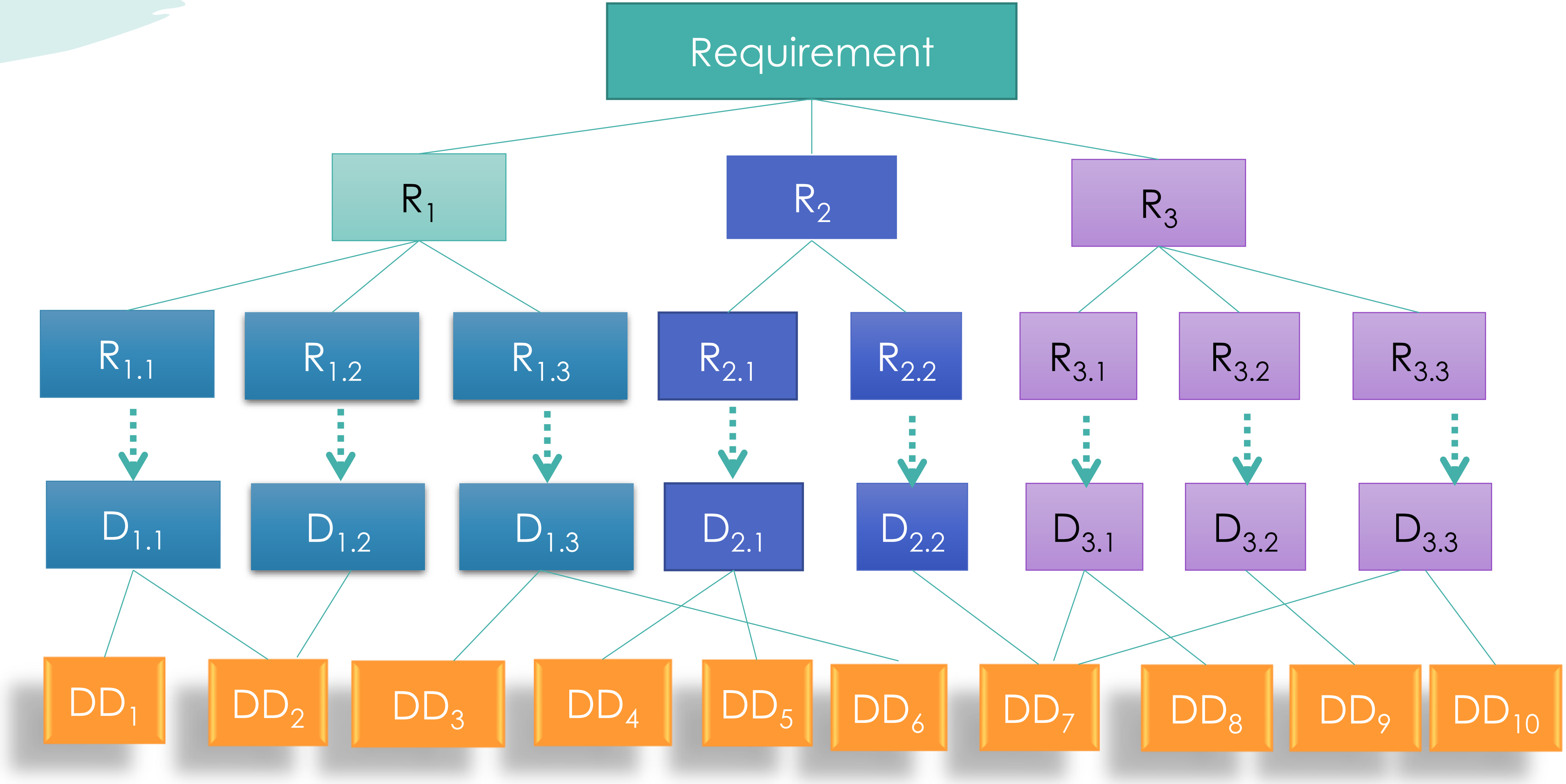
KNOWLEDGE & SOFTWARE ENGINEERING

Software Development Activities

Requirements Gathering

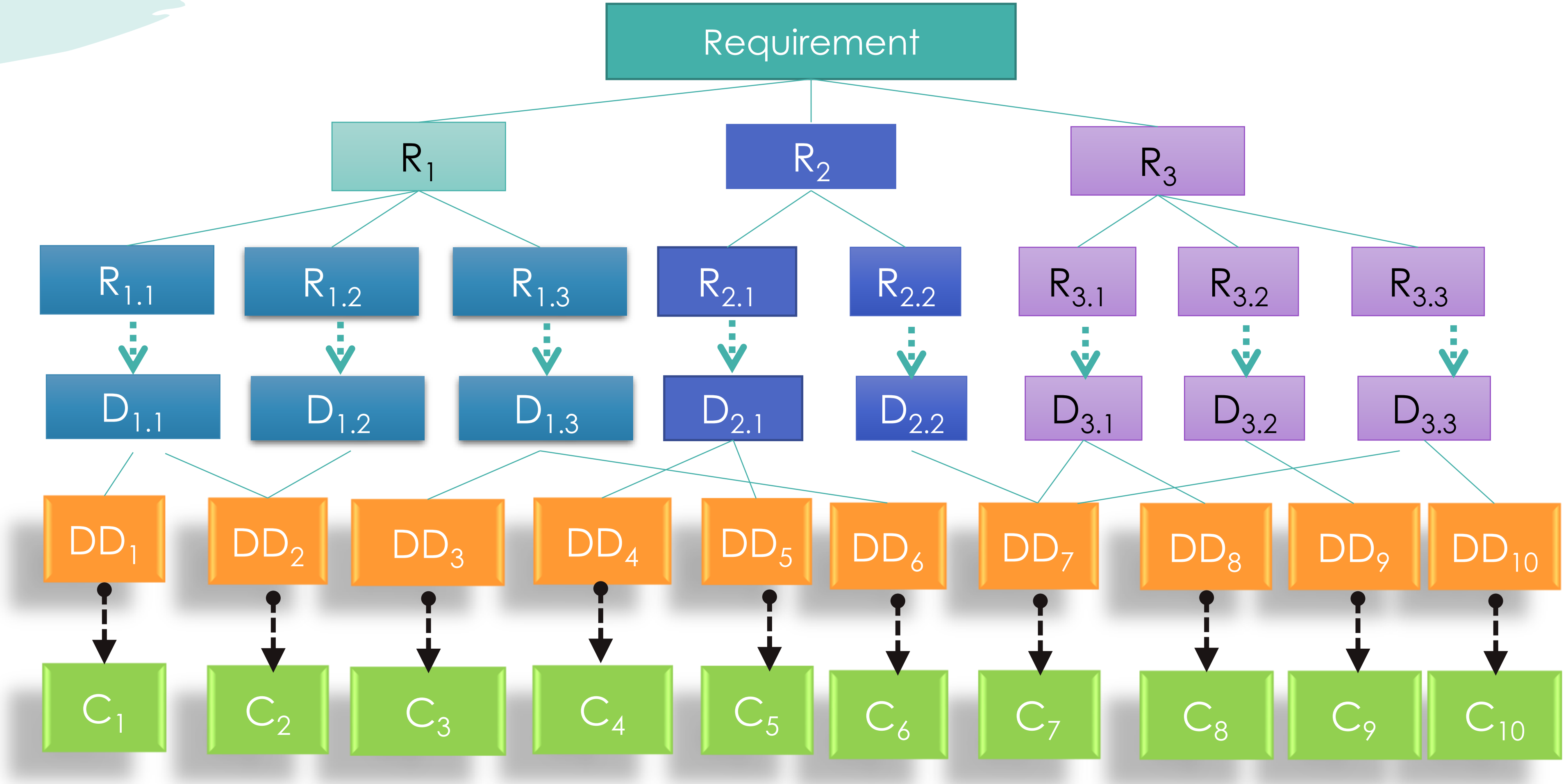


Designing the requirement…

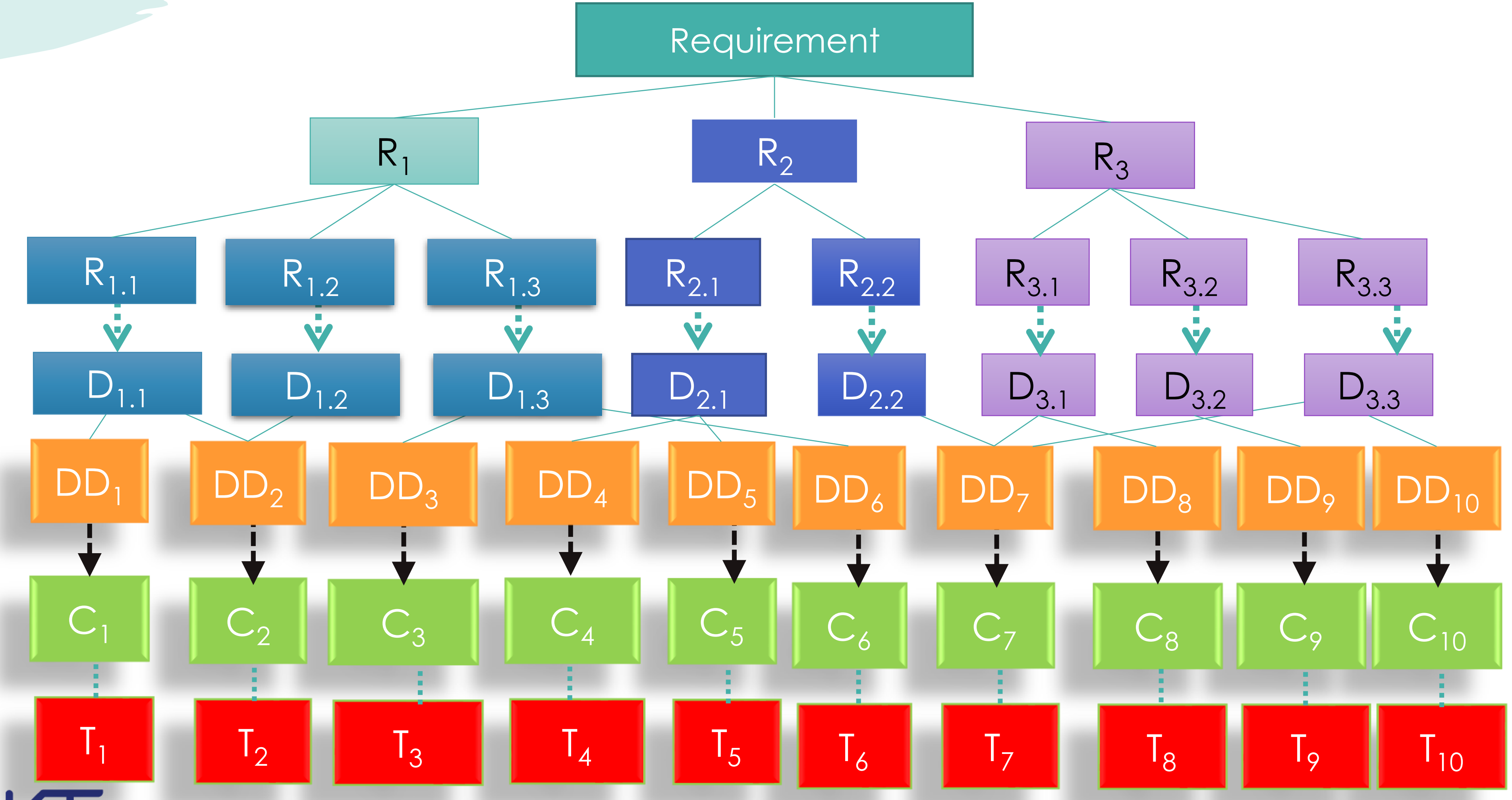


KNOWLEDGE & SOFTWARE ENGINEERING

From Design to Coding...

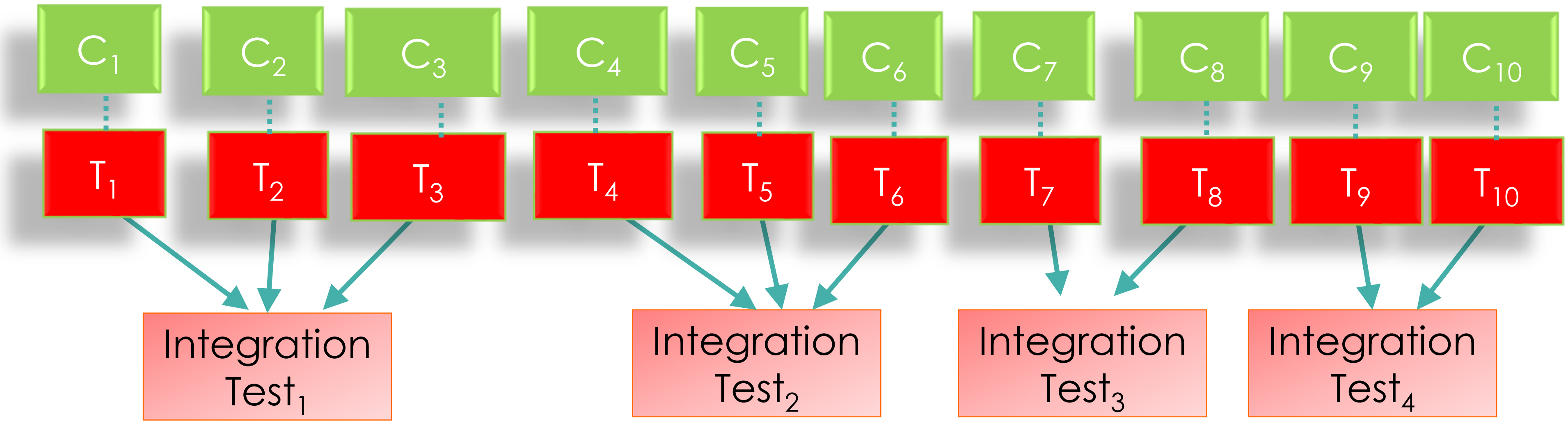


Unit Testing…



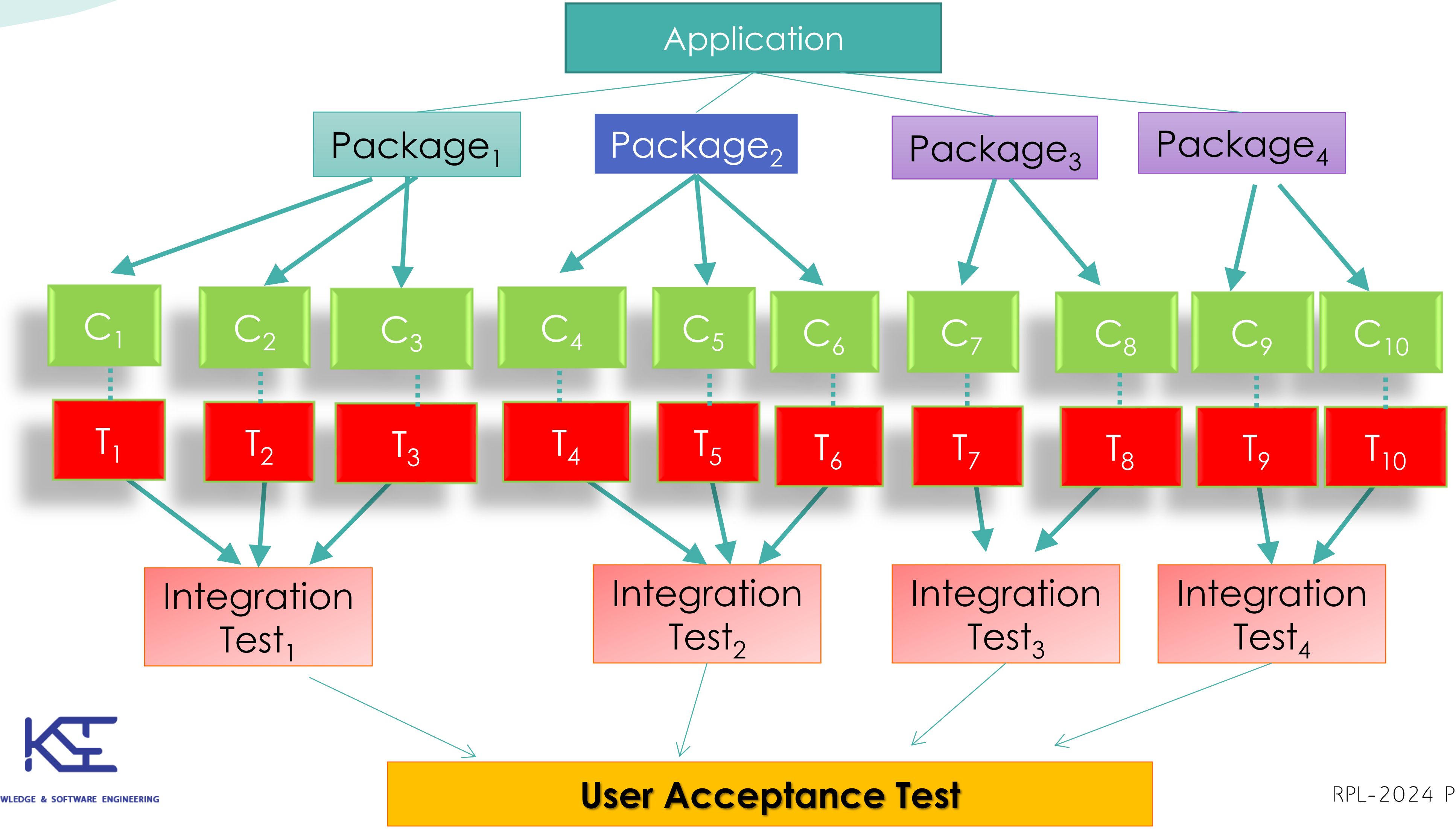
KNOWLEDGE & SOFTWARE ENGINEERING

Integration Testing…



KNOWLEDGE & SOFTWARE ENGINEERING

User Acceptance Testing...



KNOWLEDGE & SOFTWARE ENGINEERING