

*L3 Mention Informatique  
Parcours Informatique et MIAGE*

# Génie Logiciel Avancé - Advanced Software Engineering

## An Introduction

Burkhart Wolff  
wolff@lri.fr

# What is Software Engineering ?

---

- ❑ Methods, techniques and tools for
  - design: requirement analysis, models, specifications
  - development: programming, integration
  - validation: prototypes, testing
  - verification: formal proof of required properties
  - maintenance: reusability, improvements
  
- ❑ A slightly longer answer:

# What is Software Engineering ?

---

- ... slightly longer answer:

The discipline of software engineering was created to address poor quality of software, get projects exceeding time and budget under control, and ensure that software is built systematically, rigorously, measurably, on time, on budget, and within specification. [Wikipedia [en]]

- Or much shorter:

SE addresses the problems of

« **Development in the Large** »

... so for teams with 100 or 1000 of developers, and budgets of sometimes billions of dollars.

# What is Software Engineering ?

---

- ... lets consider this more closely:

The discipline of software engineering was created to address poor quality of software, get projects exceeding time and budget under control, and ensure that software is built systematically, rigorously, measurably, on time, on budget, and within specification. [Wikipedia [en]]

# What is Software Engineering ?

---

- ... lets consider this more closely:

The discipline of software engineering was created to address **poor quality** of software, get projects **exceeding time and budget** under control, and ensure that **software is built systematically**, rigorously, measurably, on time, on budget, and **within specification**. [Wikipedia [en]]

# What is Software Engineering ?

---

- ❑ poor quality
- ❑ projects exceeding time and budget
- ❑ software is built systematically,
- ❑ ... within specification.

# What do we learn about SE in this Project?

---

Covered in this  
course:

- ❑ poor quality ✓
- ❑ projects exceeding time and budget -
- ❑ software is built systematically, ✓✓
- ❑ ... within specification. ✓

# What are the Sub-disciplines of SE




---

- *Well - again common knowledge [thanks wikipedia!]*
  - [1] **Requirements engineering**: *The elicitation, analysis, specification, and validation of requirements for software.*
  - [2] **Software design**: *The process of defining the architecture, components, interfaces, and other characteristics of a system or component. It is also defined as the result of that process.*
  - [3] **Software construction**: *The detailed creation of working, meaningful software through a combination of coding, verification, testing and debugging.*
  - ...



# What are the Sub-disciplines of SE

---

- *Well - again common knowledge [thanks wikipedia!]*
  - [1] **Requirements engineering**: The elicitation, analysis, specification, and validation of requirements for software. 
  - [2] **Software design**: The process of defining the architecture, components, interfaces, and other characteristics of a system or component. It is also defined as the result of that process. 
  - [3] **Software construction**: The detailed creation of working, meaningful software through a combination of coding, verification, testing and debugging. 
  - ...



# What are the Sub-disciplines of SE

---

- *Well - again common knowledge [thanks wikipedia!]*
  - ...
  - [4] **Software verification**: The verification of the behavior of a program on a finite set of test cases, suitably selected from the usually infinite executions domain, against the expected behavior.  
*[I add here: This may include test-generation, formal proof and model-checking activities ... ]*
  - [5] **Software maintenance**: The totality of activities required to provide cost-effective support to software.
  - ...



# What are the Sub-disciplines of SE

---

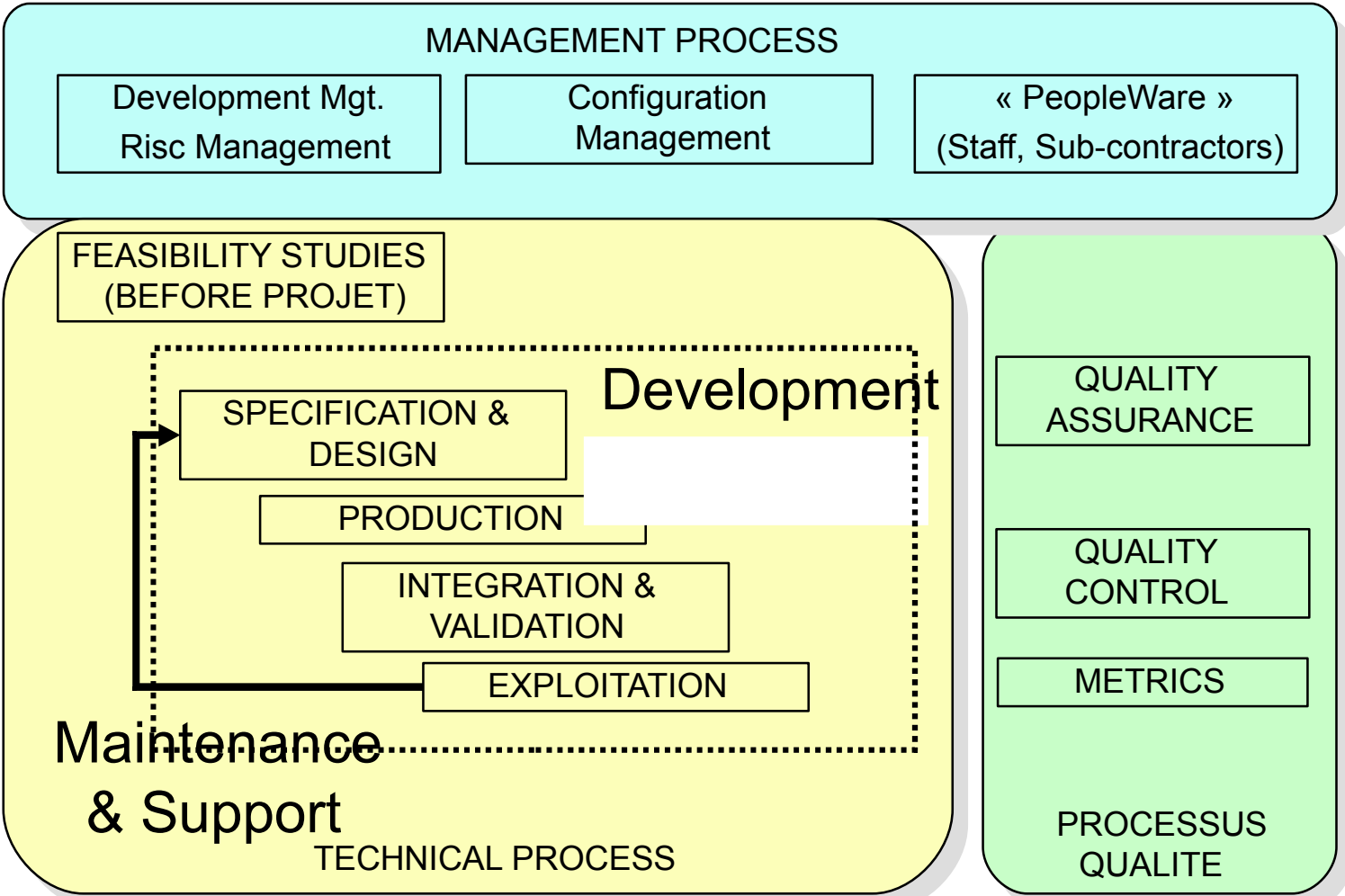
- *Well - again common knowledge [thanks wikipedia!]*
  - . . .
  - [6] **Software configuration management**: The identification of the configuration of a system at distinct points in time for the purpose of systematically controlling changes to the configuration, and maintaining the integrity and traceability of the configuration throughout the system life cycle. 
  - [7] **Software engineering management**: The application of management activities—planning, coordinating, measuring, monitoring, controlling, and reporting—to ensure that the development and maintenance of software is systematic, disciplined, and quantified.   
*[Again: this is not what we do in this course: it requires more experience and a concrete process to do this ...]*

# What are the Sub-disciplines of SE

---

- *Well - again common knowledge [thanks wikipedia!]*
  - ...
  - [8] **Software engineering process**: The definition, implementation, assessment, measurement, management, change, and improvement of the software life cycle process itself. 
  - ~~• [9] **Software engineering tools and methods**: The computer-based tools that are intended to assist the software life cycle processes (see Computer-aided software engineering) and the methods which impose structure on the software engineering activity with the goal of making the activity systematic and ultimately more likely to be successful.~~
  - [10] **Software quality management**: The degree to which a set of inherent characteristics fulfills requirements. 

# One way to view Software Engineering Project



# How can software be «built systematically»?

---

- ❑ Organize a development into formally described process !
  - ... with identified **phases**, (which correspond partly to the aforementioned "SE disciplines" )
  - ... staff (and organization and cost-plans)
  - ... defined **deliverables** (i.e. documents, codes, ... )
  - ... procedures (and tools !) to **validate** the quality of the **deliverables**(reviews, static checks)
  - ... procedures to version and configurate deliverables (in particular code)
  - Compare: **THE SWE-BOOK**

---

IEEE Computer Society an international standard ISO/IEC TR 19759:2005

# How can software be «built systematically»?

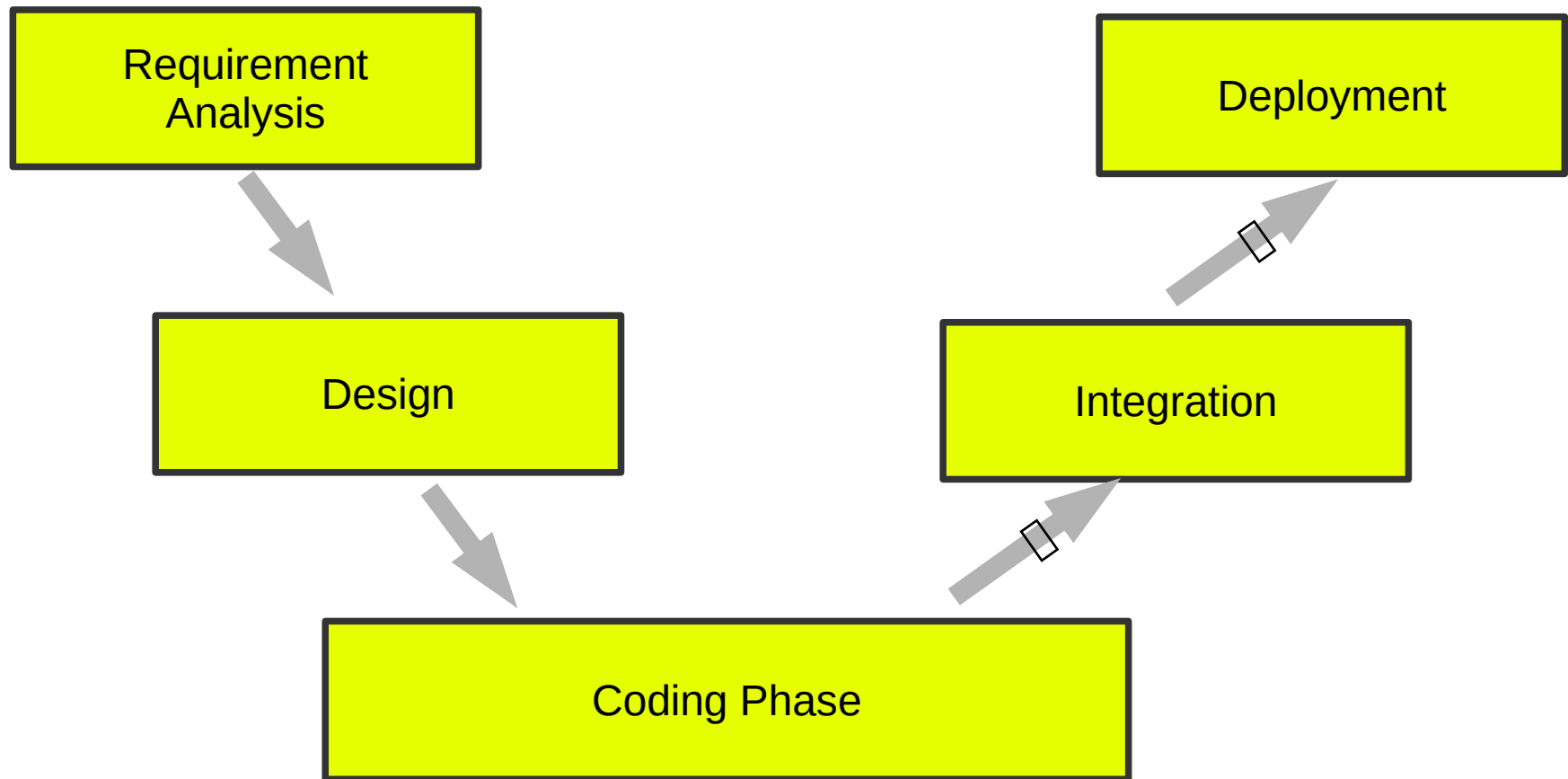
---

## □ An Example Process-Model: The V-Model.

- V-Modell XT [German Administration 2005]  
[http://de.wikipedia.org/wiki/V-Modell\\_\(Entwicklungsstandard\)#cite\\_note-6](http://de.wikipedia.org/wiki/V-Modell_(Entwicklungsstandard)#cite_note-6)
- ... phases : Requirement, Architectural Design, Design, Code, Tests, passed deployments, ...
- ... defined deliverables (i.e. documents, codes, ... )  
templates that have to be "taylorized"
- ... procedures: (and tools !): an xml editor, a version management and an access control management for the 35 (!) different roles in the process

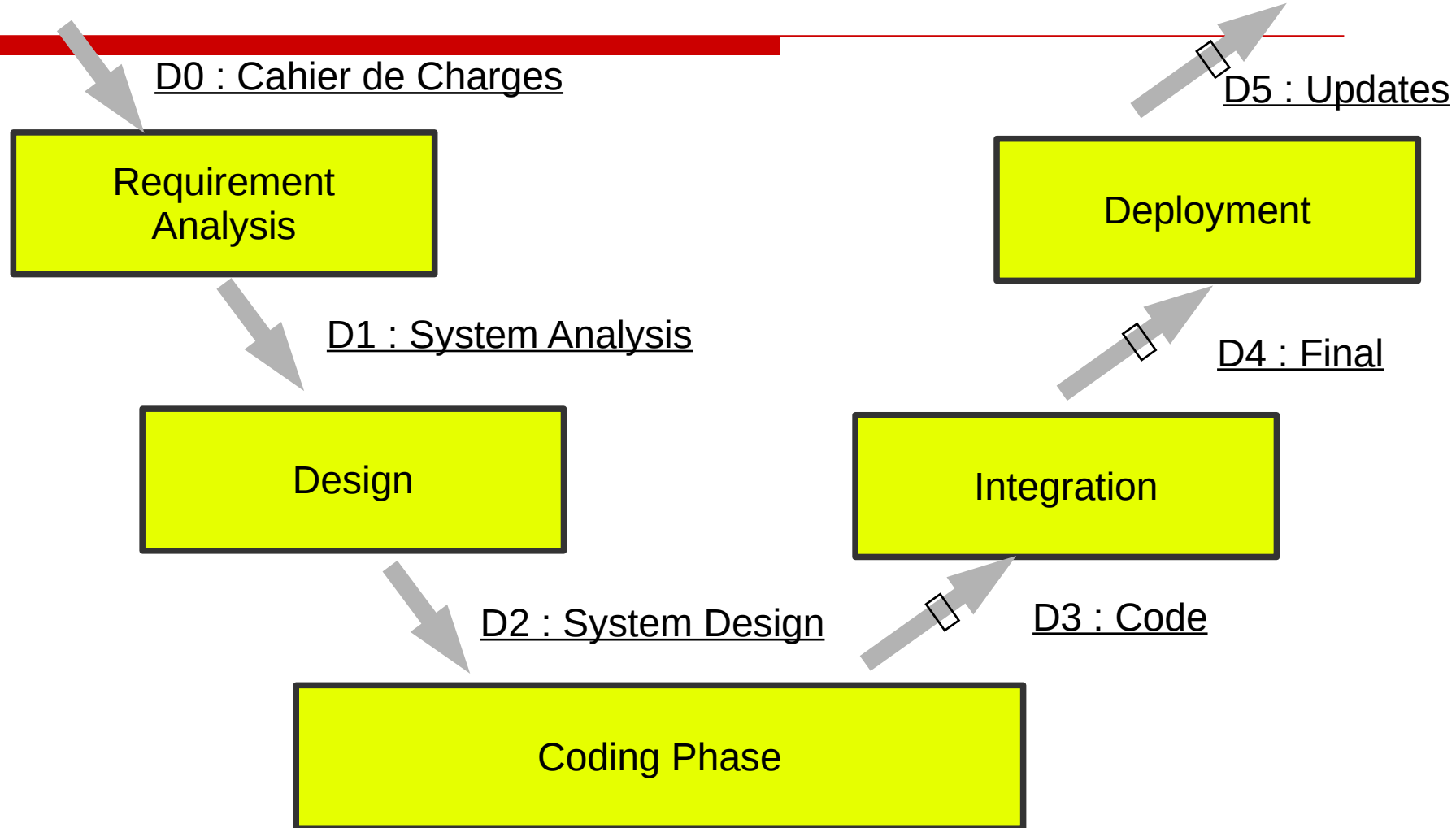
# How can software be «built systematically»?

---



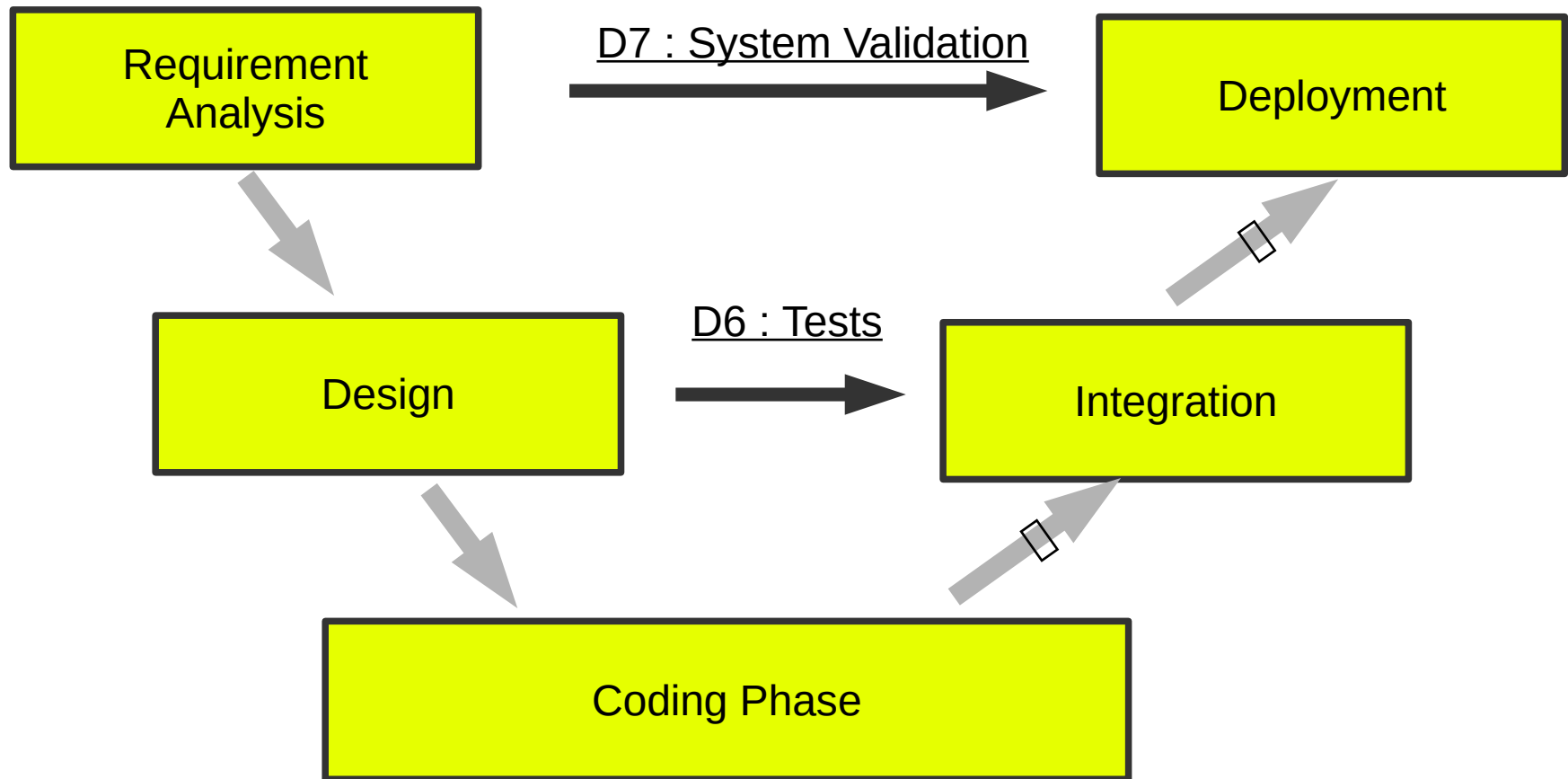


# How can software be «built systematically»?



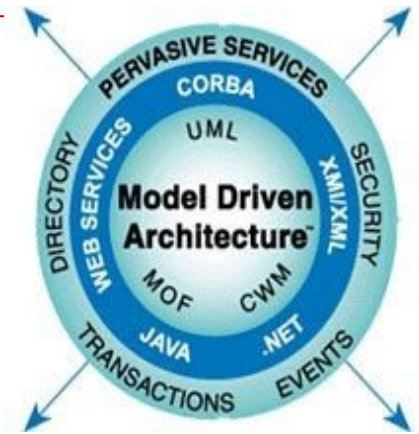
# How can software be «built systematically»?

---



# How can software be «built systematically»?

## □ Another Variant / Alternative IBM Rational Unified Process (RUP)



- Idea : Using UML and OCL integrated into the Deliverables (documents)
- Idea : Allows for semi-formal editing, more precise notation and therefore better communication
- Analysis, Design and Code Documents CONTAIN standardized diagrammatic specification elements (the "model") which can be automatically validated
- Code and Tests can partially be generated from design models (Model-Driven Engineering (MDA))

# Why UML in a SE Course ?

---

- ❑ It is clearly not perfect
  - very syntactic, very diagrammatic flavor
  - diagrams do not necessarily scale up
  - not everyone in industry uses it (large companies typically have their own development process, reflecting their own « culture »)
  
- ❑ **BUT:** Many in industry use it,
  - or use similar things (SysML), and most practitioners in industry would understand UML
  - we use it in requirements analysis, design, and for test generation techniques.