Software Engineering Course

Chapter 7
Introduction to development methods

2022-2023

Dr. Kouah S.

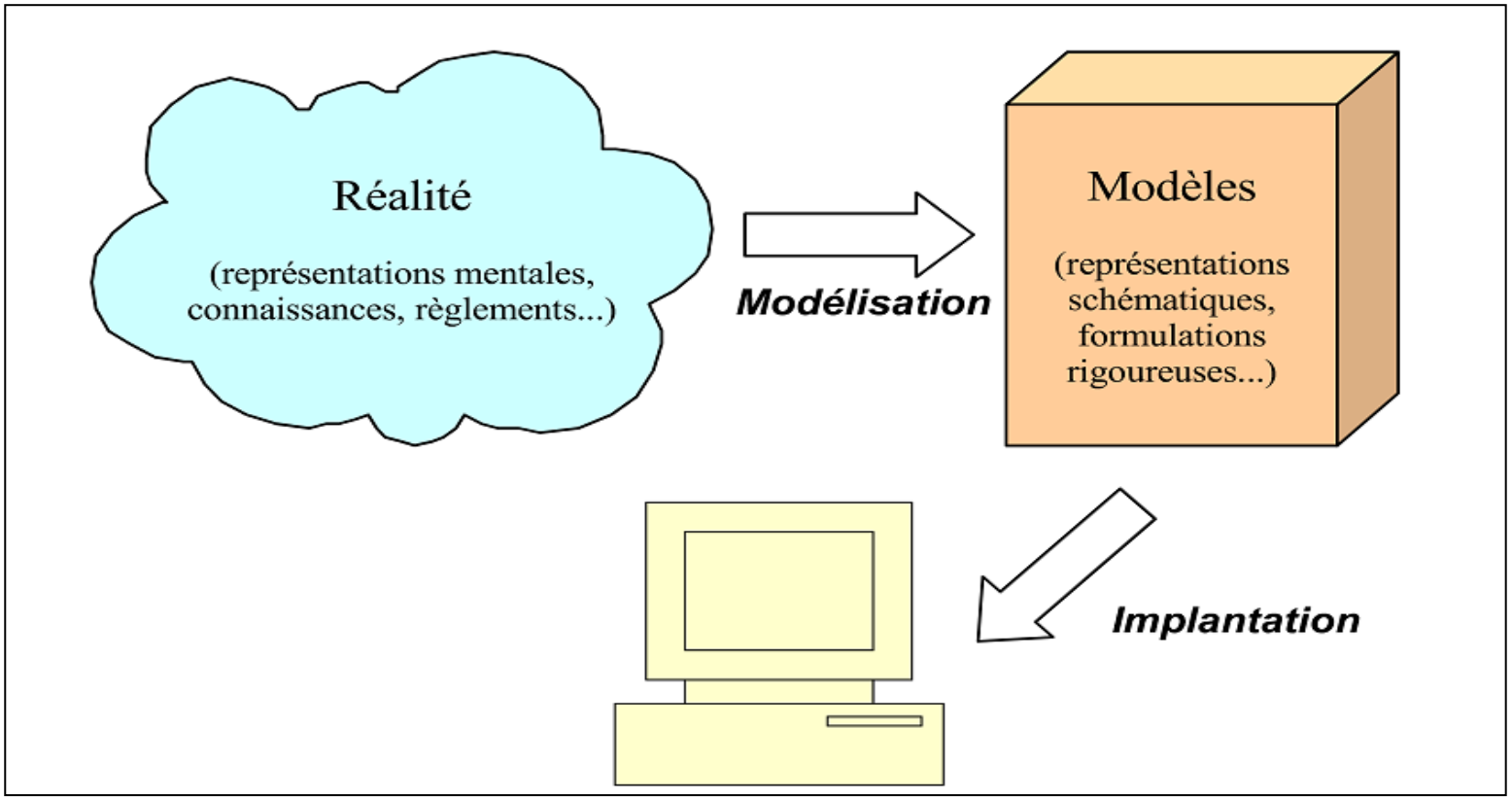
Plan

- **Introduction**
- **Importance of modeling**
- **Development methods**
- **Unified Process (UP)**
- **Rational Unified Process (RUP)**

Introduction

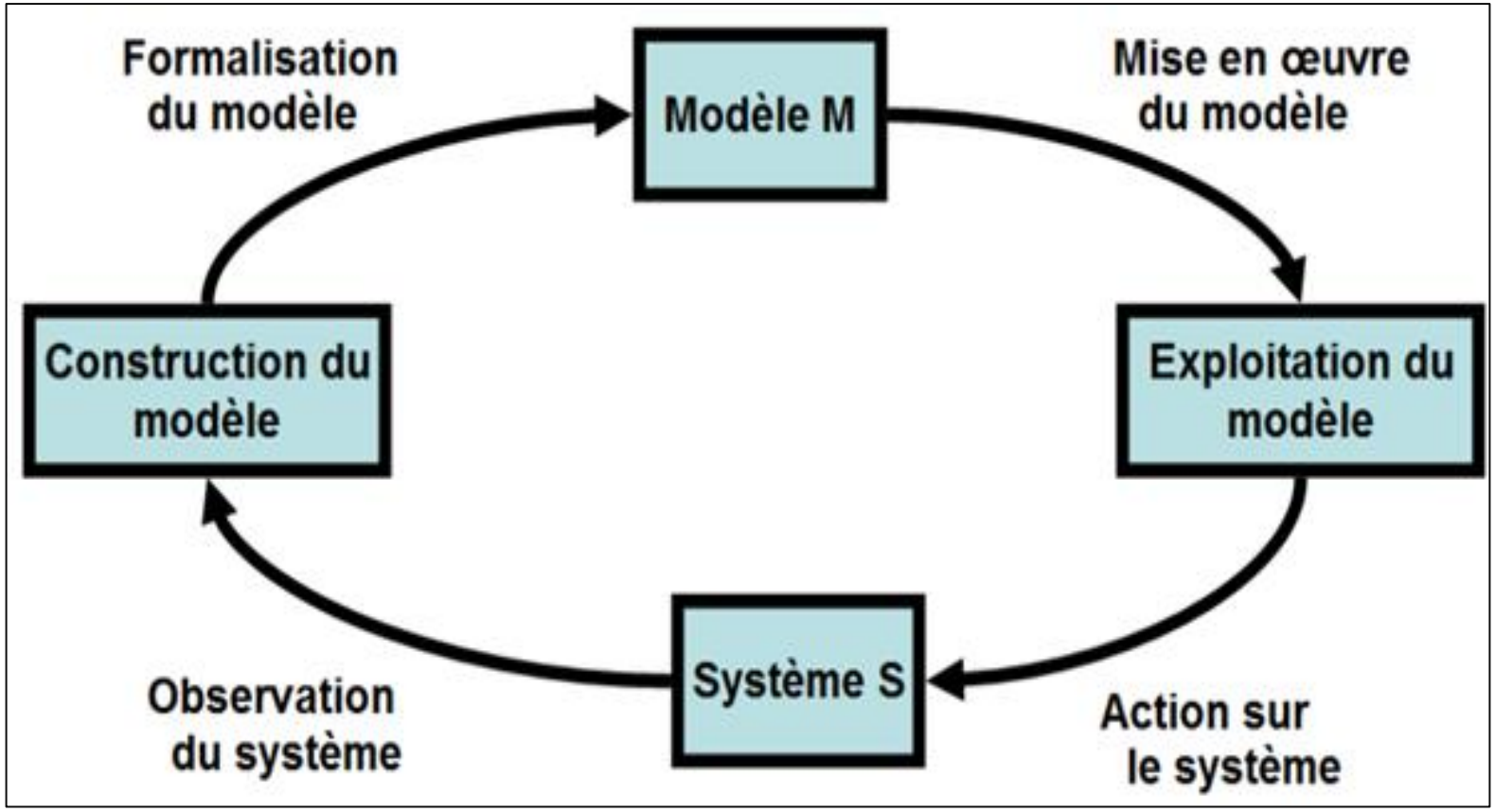
- Modeling a computer system is the process by which **a model is constructed** for that system.

Importance of modeling



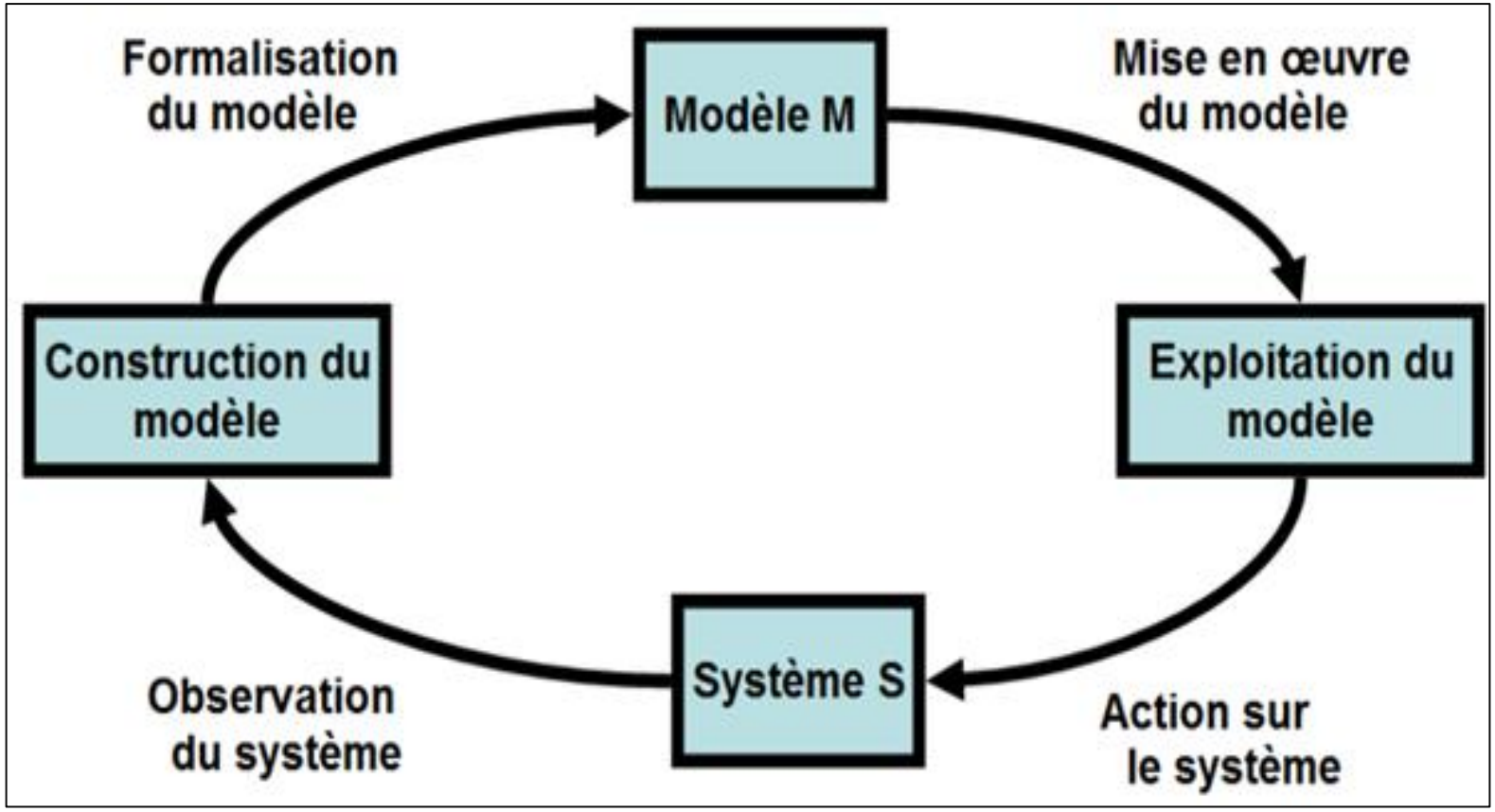
Introduction

- Practically, modeling goes through the following stages:



Introduction

- Practically, modeling goes through the following stages:



Introduction

Why modeling?

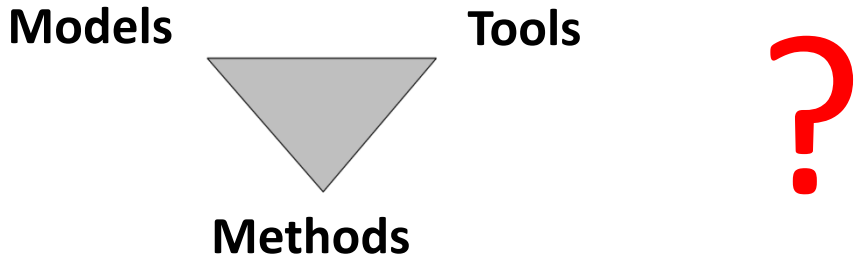
- ✓ Understand the system to be automated.

“A model is a simplification of reality. »Grady BOOCH.

- ✓ Communicate with team members.
- ✓ Mastering complexity
- ✓ Automate software production.
- ✓ Documentation.
- ✓ Code.

Introduction

Modeling = understanding and representation



- **Model**: abstract representation of all or part of reality. A model does not represent an absolute reality but reflects important aspects of reality, so it gives an accurate and relevant view of it (subjective view).
- **Tool**: automated or semi-automated support for supporting methods.
- **Method**: way in which we will carry out an activity.
- **Method** = {model, tools} + implementation approach.

A method is used to channel and order the modeling stages.

Classification of development methods

Several classifications, according to certain criteria:

1. Top-down/Bottom-up decomposition

✓ Top Down Methods.

✓ Bottom-up methods.

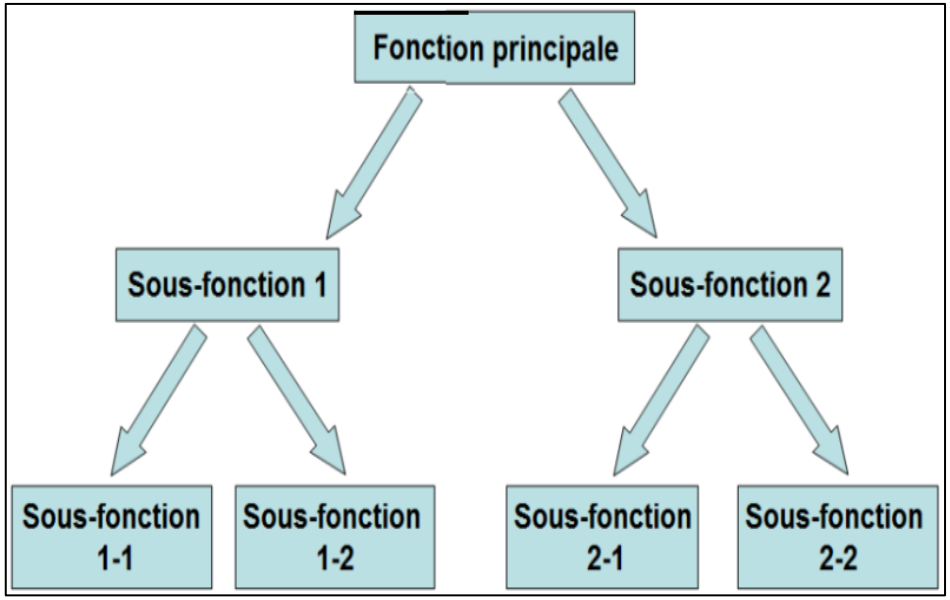
2. Functional/Object Decomposition

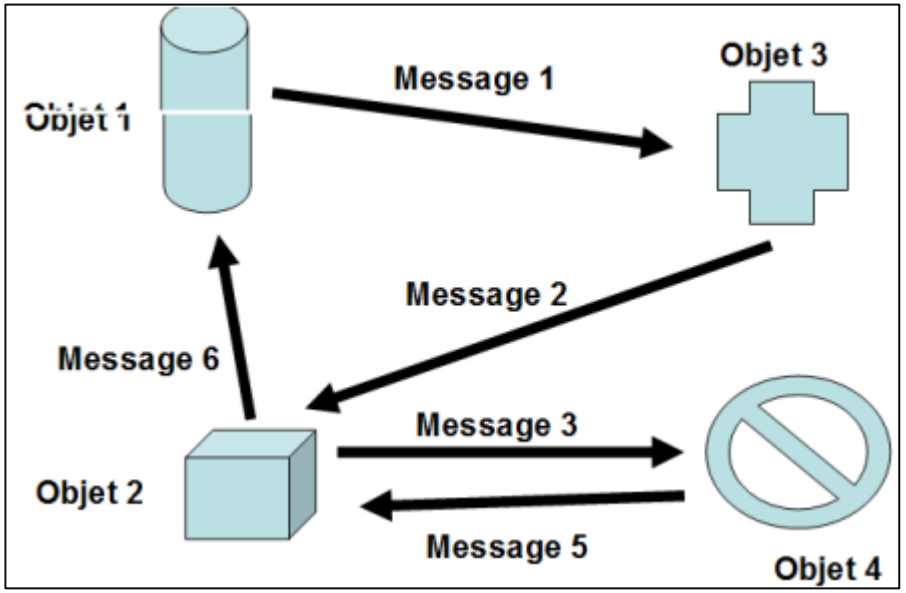
Functional methods

SADT, SA, SD.

Object-oriented methods

OOA, OOD, HOOD, OMT, OOSE





Object Oriented Modeling

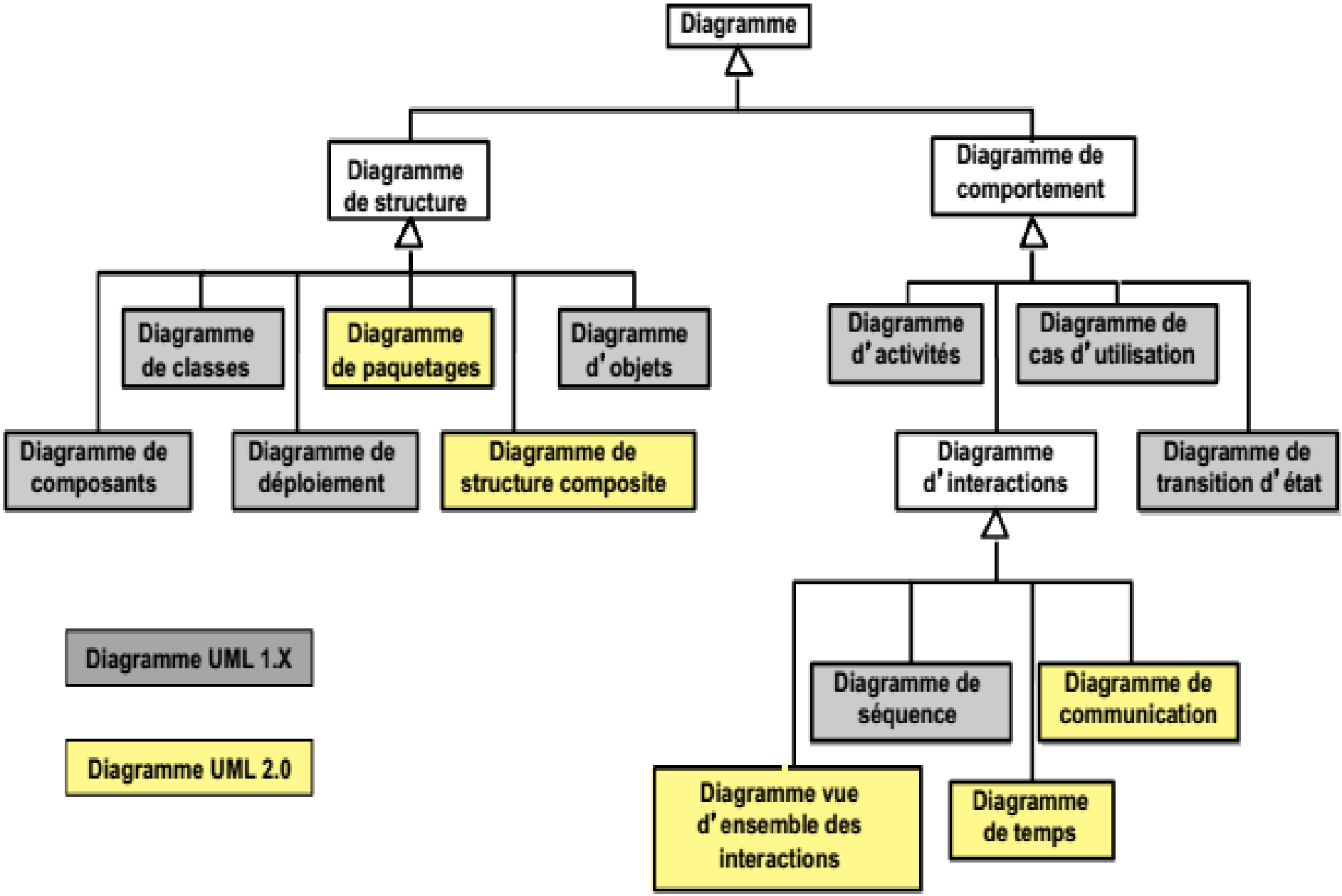
- Object-oriented design is the method that leads to software architectures **based on system objects**, rather than on functional decomposition.
- It is the structure of the system which gives it its form.
- We can start from the objects of the domain (basic building blocks) and go back to the global system: **bottom-up approach**.
- Attention: the object approach is **not only bottom-up.**

Typology of analysis and design methods

Another classification distinguishes:

- Data-driven methods.
- Functional methods (directed by treatments)
- Systemic methods such as Merise
- Object-oriented methods.

UML Diagrams



The Unified Process (UP)

- ✓ The **Unified Process** is a **generic** software development method developed by **UML** designers.
- ✓ **Generic** means that it is **necessary to adapt UP** to the context of the project, team, domain and/or organization.
- ✓ There are therefore a certain number of methods from UP such as: **RUP (Rational Unified Process)**, **2TUP (Two Track Unified Process)**.
- ✓ There are other approaches (which are generally not completely contradictory), e.g. “agile” methods, much less model-oriented, such as XP (eXtreme Programming).

The Unified Process (UP)

- ✓ Unified process **helps assign tasks** and **responsibilities** within a software development organization
- ✓ **Disciplined approach** for large projects (project manager, analysts, integrator, stakeholders, etc.)
- ✓ **Approach to adapt** for small projects.
- ✓ **Not particularly** designed for the development of **embedded systems.**

The Unified Process (UP)

- ✓ The unified process **uses UML**.
- ✓ The unified process is **driven by use cases**
- ✓ **Centered on architecture**
- ✓ **Iterative** and **incremental**
- ✓ **Risk oriented**

UP is driven by use cases:

- ✓ System analyzed, designed and developed for users.
- ✓ Everything must therefore be done from the user point of view.

UP is focused on architecture:

- ✓ The system architecture is described using **different views**.
- ✓ The architect proceeds in an **incremental fashion:**
  - ✓ it begins by defining a simplified architecture that meets the needs classified as priorities;
  - ✓ Then defines from the simplified architecture the subsystems in a much more precise way.

UP is iterative and incremental:

- By proceeding **iteratively**, it is possible to discover errors and misunderstandings **earlier**.
- User **feedback** is also encouraged and the **tests** carried out for each use provide a more objective view of the progress of the project.
- **Iterative** work allows the team to **capitalize on lessons** from the previous cycle in each cycle.

UP is risk-oriented:

- Identify the risks.
- Maintain a list of risks throughout the project.

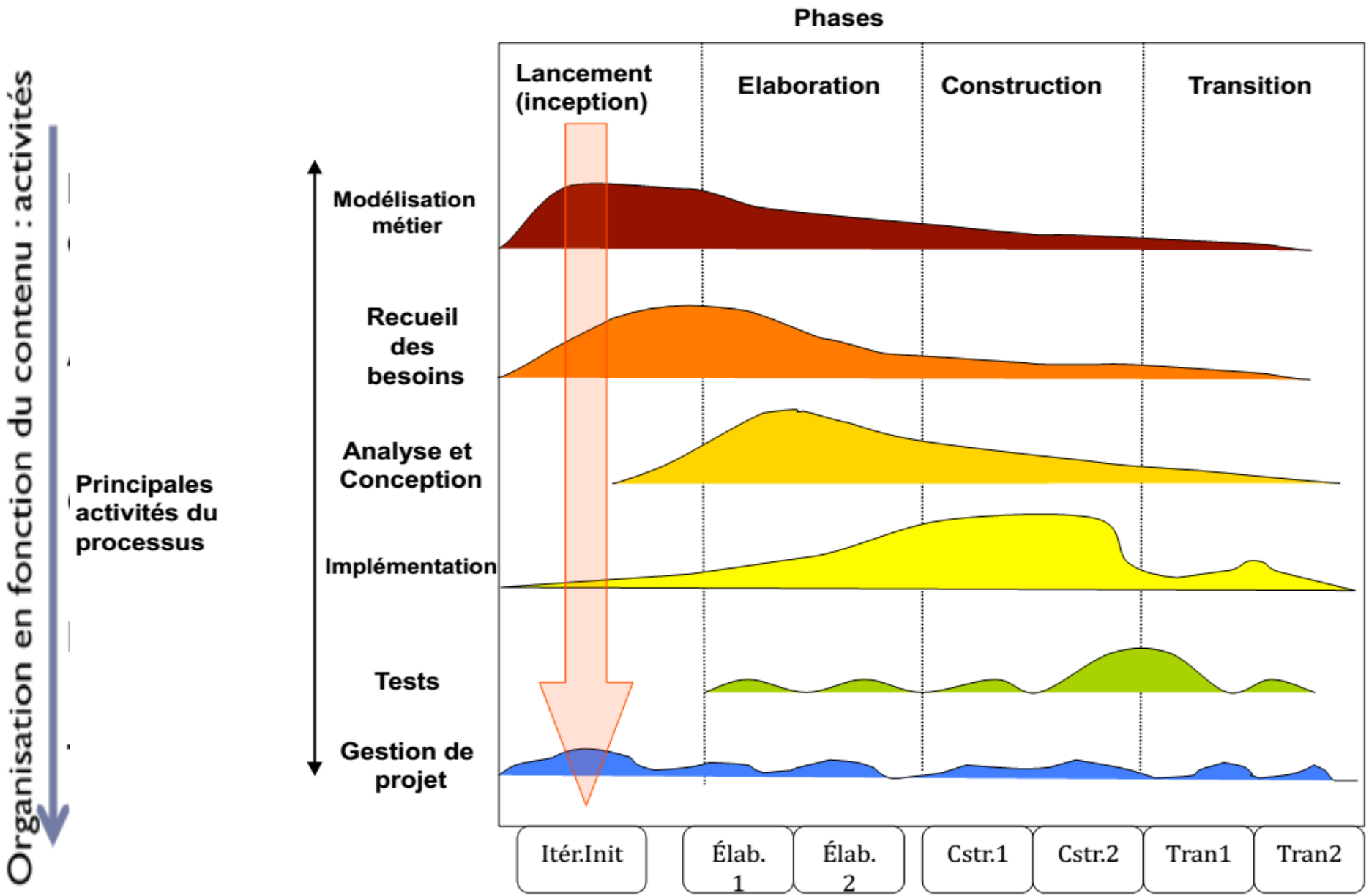
Example of UP implementation: RUP (Rational Unified Process)

- **RUP** is one of the **most famous implementations of the UP method** for providing a framework for software development.

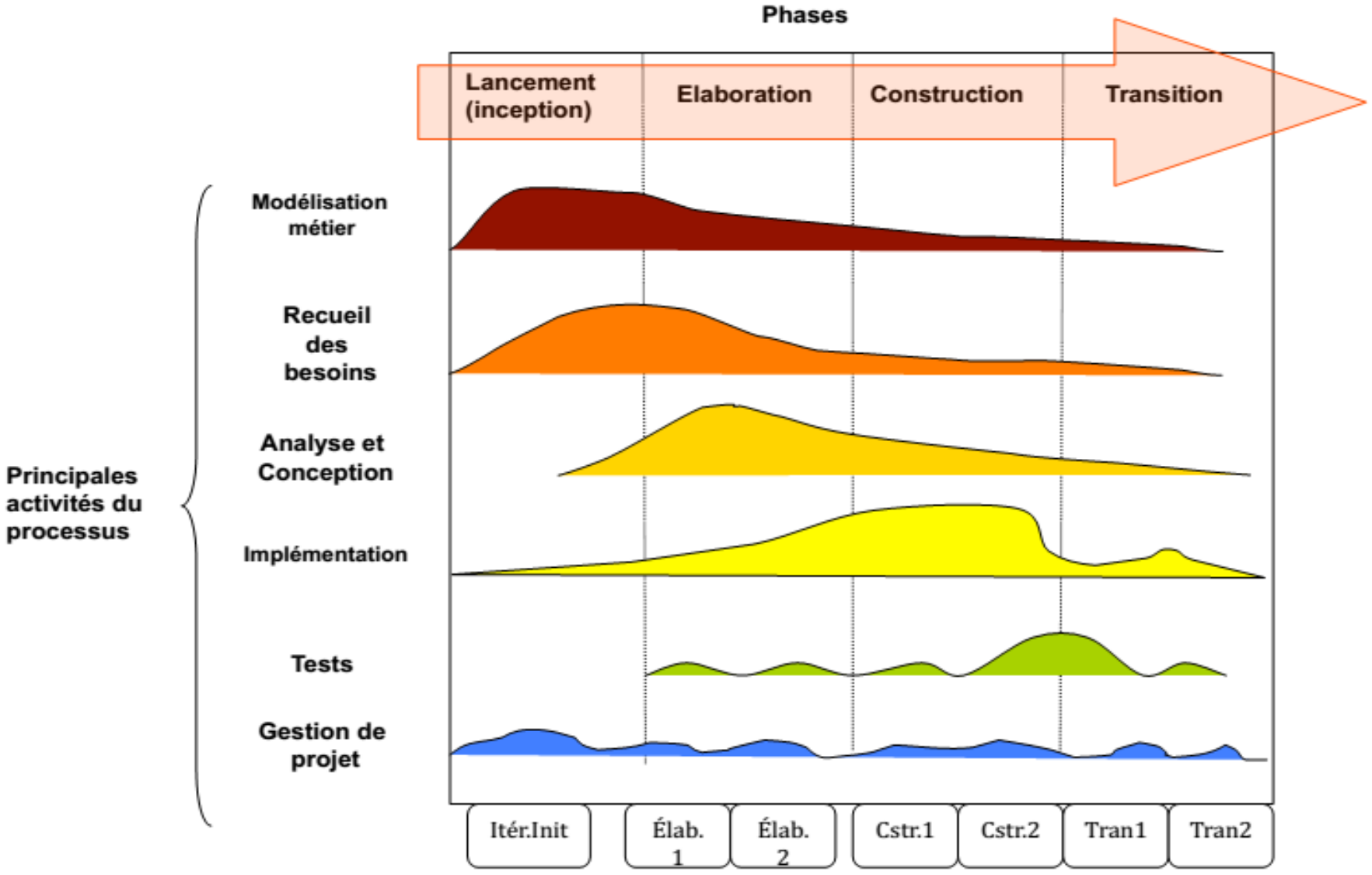
Components of RUP

- ✓ **4 phases:**
  - ✓ Opportunity study.
  - ✓ Elaboration.
  - ✓ Construction.
  - ✓ Transition.
- ✓ **5 activities:**
  - ✓ Expression of requirements.
  - ✓ Analysis.
  - ✓ Design.
  - ✓ Implementation.
  - ✓ Testing.
- ✓ **Set of iterations**: development circuit leading to a deliverable.

RUP: Logical structure of the process



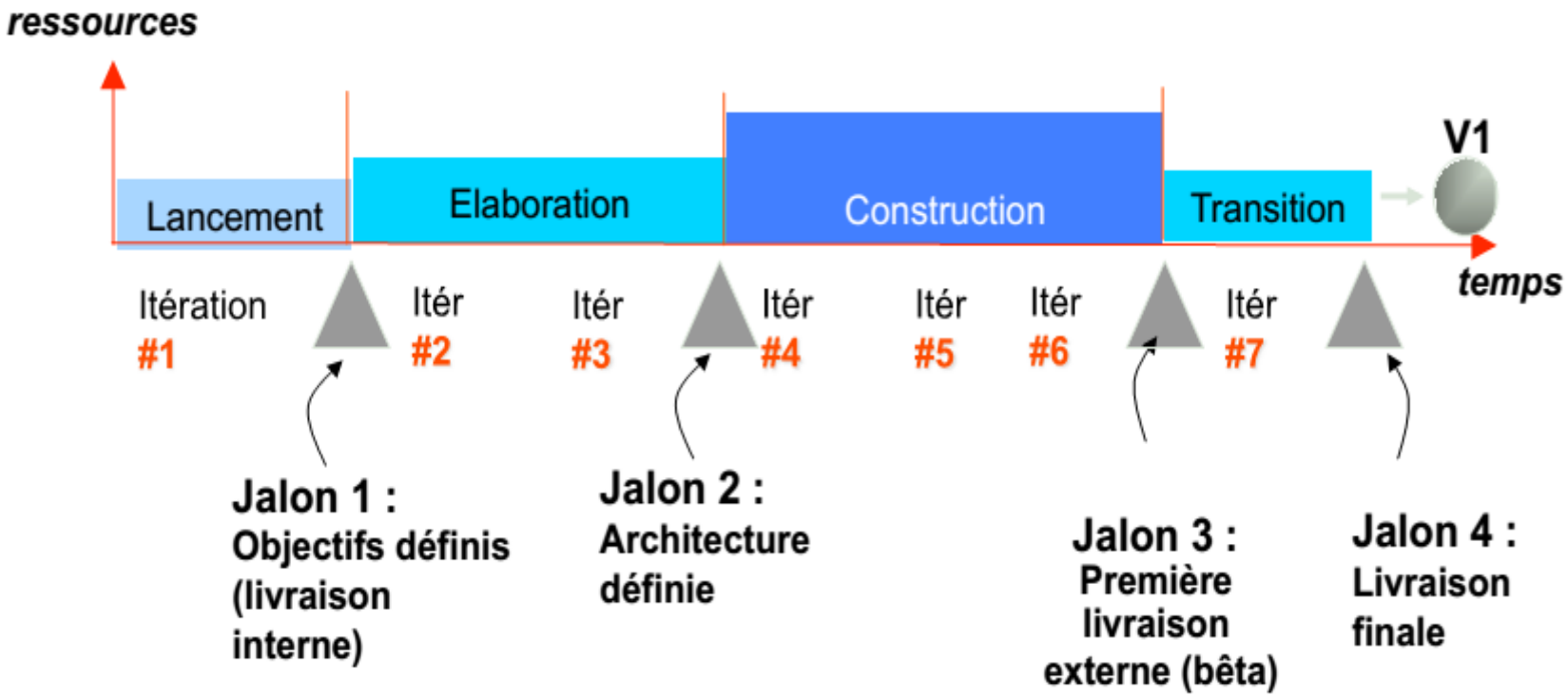
RUP: the four phases



RUP: the four phases

- ✓ **Initialization**:
  - ✓ Problem definition.
- ✓ **Development**:
  - ✓ Business planning, resource allocation, analysis.
- ✓ **Construction** :
  - ✓ Software development in successive increments.
- ✓ **Transition**:
  - ✓ Acceptance and deployment.
- ✓ **The development phases are the main stages of software development**
  - ✓ The project begins in the initialization phase and ends in the transition phase

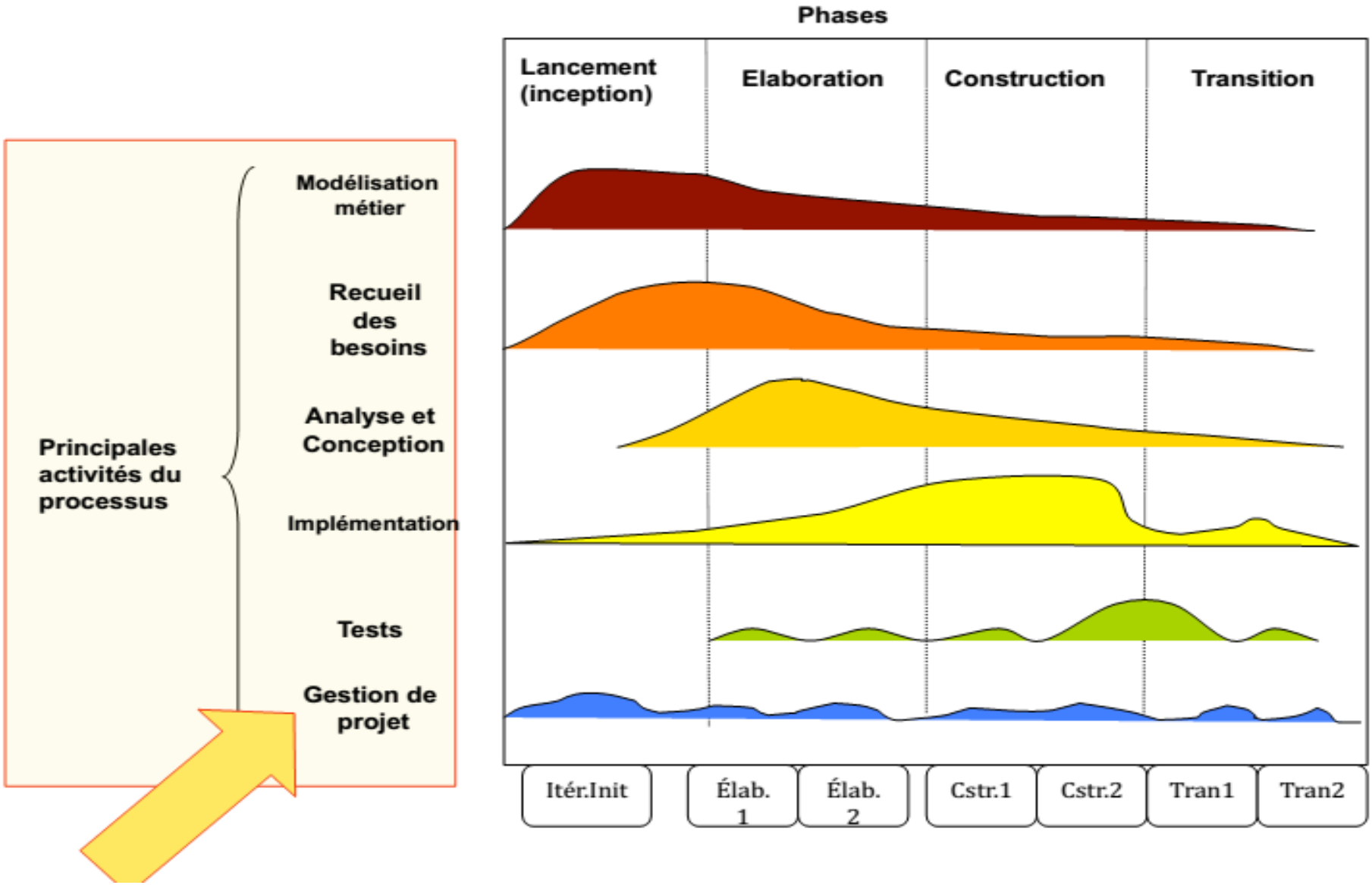
RUP: Progress of the phases



- Phases
- Iterations
- Milestones

Milestones correspond to stages of evaluating the completed phase, and launching the next phase

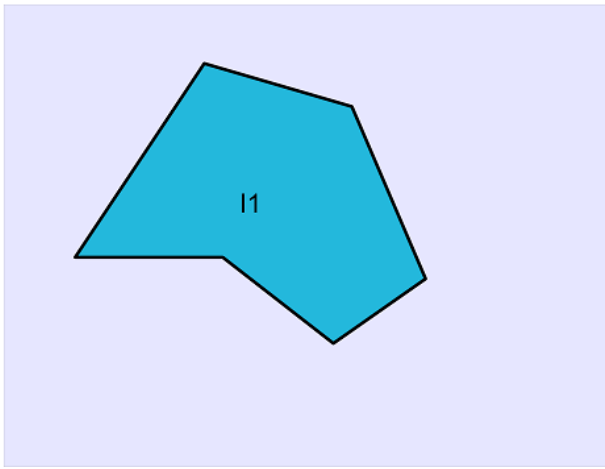
RUP: Progress of the activities



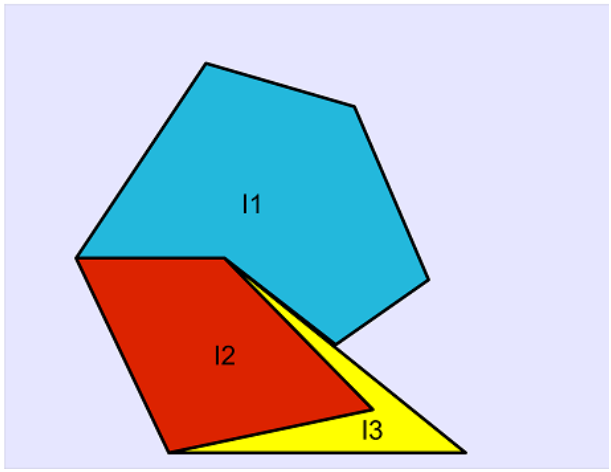
Iterations: example

Itérations 0

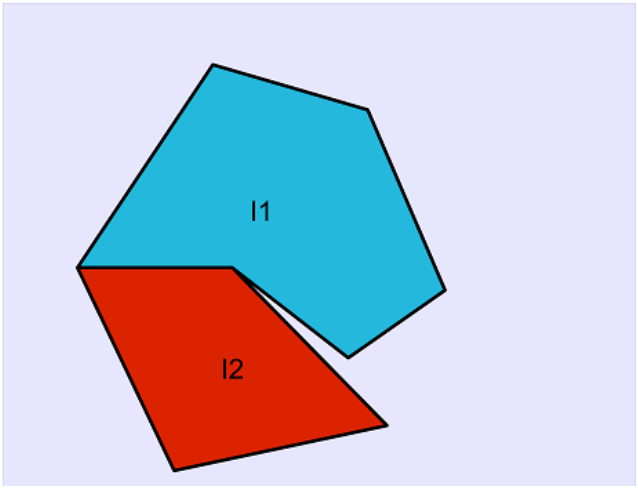
Itérations 1



Itérations 3

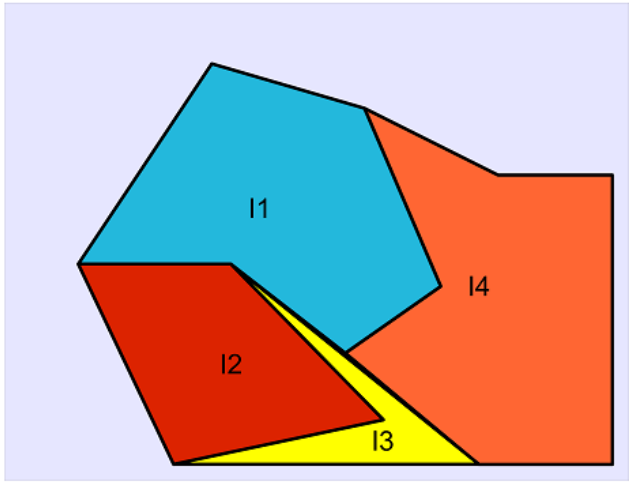


Itérations 2

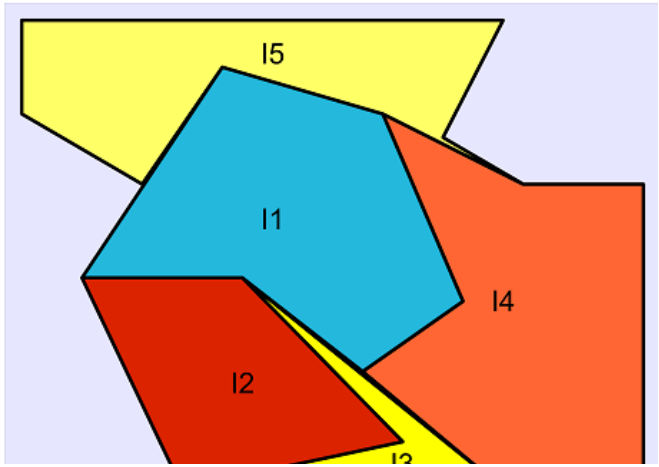


Iterations: example

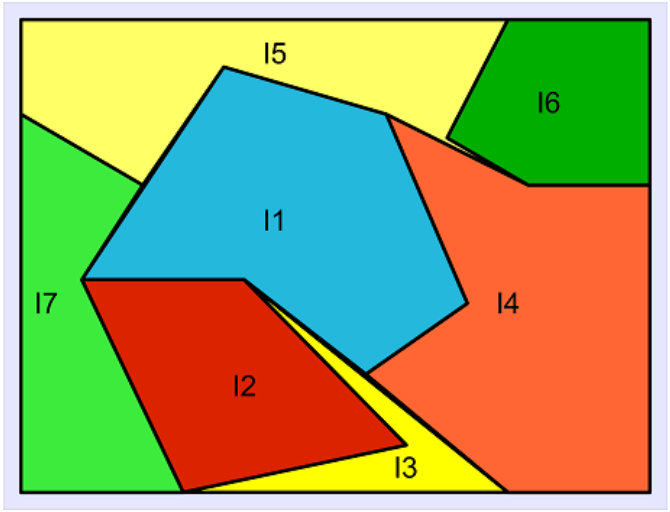
Itérations 4



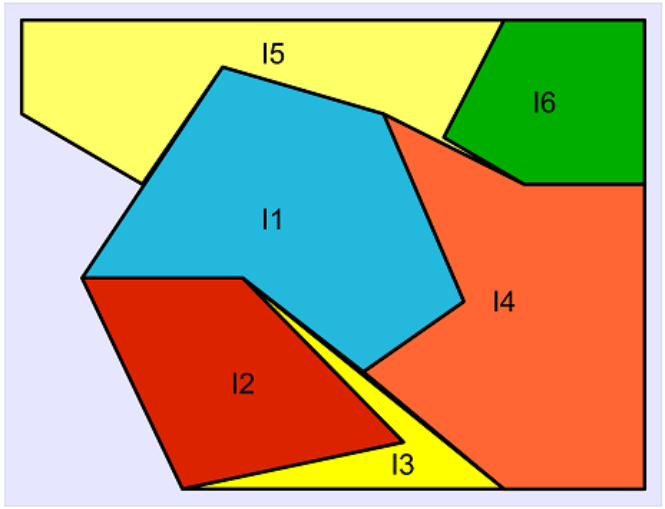
Itérations 5



Itérations 7



Itérations 6



Iterations

At each iteration, we repeat:

1. Specification;

2. Design;

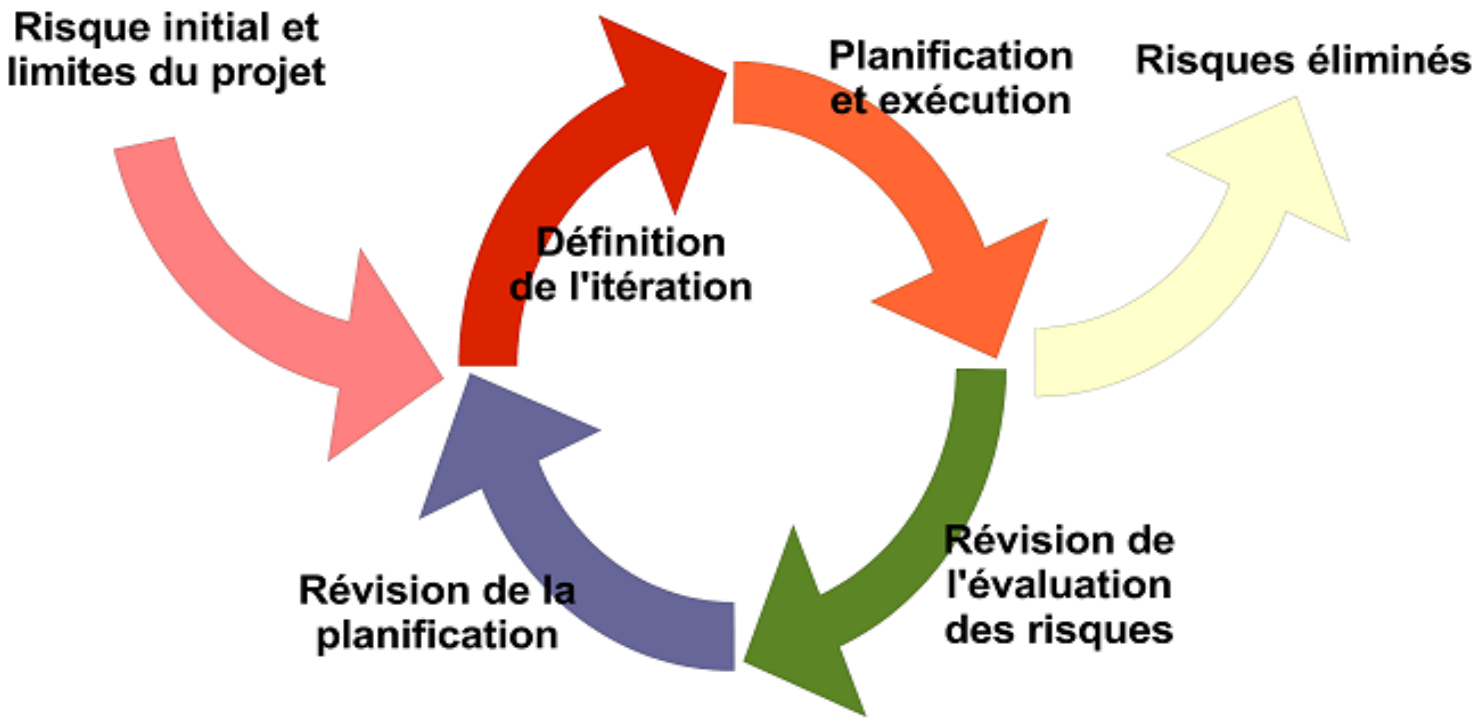
3. Implementation;

4. Tests.

RUP: elimination of risks at each iteration

- ✓ Software development can be seen as a **gradual process of <u>eliminating risk</u>**.
- ✓ It is during "**Planning and Execution**" that we repeat:

  Specification → Design → Implementation → Testing.

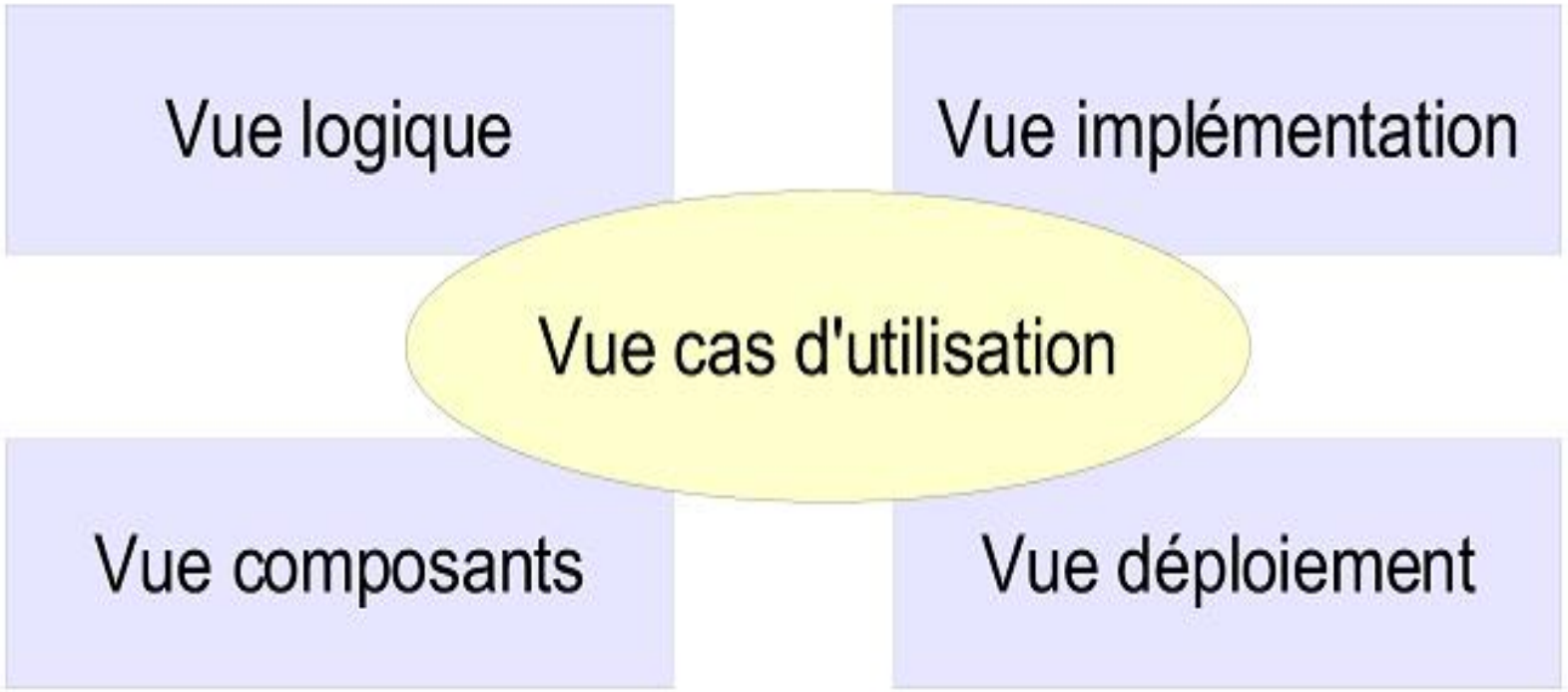


RUP: elimination of risks at each iteration

- ✓ Each iteration takes into account a certain number of use cases.
- ✓ Major risks are prioritized.
- ✓ Each iteration gives rise to an increment and produces a new executable version.

RUP: Centered on architecture

- ✓ Modeling of different independent and complementary perspectives.
- ✓ Layered architecture and views of Krutchen.



RUP: System Views

- ✓ **Use case view:**
  - ✓ Description of the system as a set of transactions from the user's point of view.
- ✓ **Logical view:**
  - ✓ Created during the development phase and refined during the construction phase;
  - ✓ Use of class diagrams, sequences...
- ✓ **Component view:**
  - ✓ Description of the software architecture.
- ✓ **Deployment view:**
  - ✓ Description of the hardware architecture of the system.
- ✓ **Implementation view:**
  - ✓ Description of algorithms, source code.

Phase1: Initialization (Objective)

- ✓ Definition of the project framework, its concept, and inventory of the content; Elaboration of the critical use cases having the most influence on the architecture and the design;
- ✓ Realization of one or more prototypes demonstrating the functionalities described by the main use cases;
- ✓ Detailed estimation of the workload, cost and general planning as well as the next phase of development Risk estimation.

Phase1: Initialization (Activities)

- ✓ Formulation of the project framework, needs, constraints and acceptance criteria;
- ✓ Planning and preparation of the economic justification of the project and evaluation of alternatives in terms of risk management, resources, planning;
- ✓ Synthesis of candidate architectures, cost assessment.

Phase1: Initialization (Deliverable)

- ✓ A **vision document** presenting the basic needs, constraints and main functionalities;
- ✓ A **first version of the use case model**;
- ✓ A **project glossary**;
- ✓ An **economic justification document** including the general context of achievement, success factors and financial forecast;
- ✓ A **risk evaluation**;
- ✓ A **project plan** presenting phases and iterations;
- ✓ One or more **prototypes**.

Phase1: Initialization (Evaluation criteria)

- ✓ Consensus on:
  - ✓ Planning;
  - ✓ Cost estimation;
  - ✓ The definition of all projects and the parties involved.
- ✓ Common understanding of requirements.

Phase 2: Elaboration (objective)

- ✓ Define, validate and finalize the architecture;
- ✓ Demonstrate the effectiveness of this architecture in responding to our needs;
- ✓ Plan the construction phase.

Phase 2: Elaboration (Activities)

- ✓ Development of the general vision of the system, the main use cases are understood and validated;
- ✓ The project process, infrastructure, tools and development environment are established and implemented;
- ✓ Development of the architecture and selection of components.

Phase 2: Elaboration (Deliverable)

- ✓ The use case model is at least 80% produced;
- ✓ The list of non-functional requirements and constraints identified;

  A description of the architecture;
- ✓ An executable allowing the software architecture to be validated through certain complex functionalities;
- ✓ The list of risks reviewed and the update of the economic justification of the project;
- ✓ The implementation plan, including a development plan presenting the phases, iterations and evaluation criteria;

Phase 2: Elaboration (Evaluation criteria)

- ✓ The stability of the vision of the final product;
- ✓ The stability of the architecture;
- ✓ Management of the main risks is addressed by the prototype(s);
- ✓ The definition and details of the project plan for the construction phase;
- ✓ A consensus, by all stakeholders, on updating planning, costs and project definition.

Phase 3 : Construction (Objectif)

- ✓ Minimization of development costs by:
  - ✓ Resource optimization;
  - ✓ Minimization of unnecessary work.
- ✓ Maintaining quality;
- ✓ Creation of executable versions.

Phase 3 : Construction (Activities)

- ✓ Management and control of resources and optimization of the project process;
- ✓ Evaluation of the versions produced against the defined acceptance criteria.

Phase 3 : Construction (Deliverable)

- ✓The executable versions of the software corresponding to the iteration by iteration enrichment of the functionalities;
- ✓User manuals produced in parallel with the incremental delivery of executables;
- ✓A description of the produced versions.

Phase 3 : Construction (Evaluation criteria)

- ✓ The stability and quality of the executables;
- ✓ Stakeholder preparation;
- ✓ The financial situation of the project compared to the initial budget.

Phase 4: Transition (Objectifs)

- ✓ Deployment of the software in the users' operating environment;
- ✓ Handling transition-related issues;
- ✓ Achieve a level of stability such that the user is independent;
- ✓ Achieve a level of stability and quality such that stakeholders consider the project complete.

Phase 4: Transition (Activities)

- ✓ “Packaging” activities for the software to make it available to users and the operations team;
- ✓ Correction of residual errors and improvement of performance and scope of use;
- ✓ Evaluation of the final product with regard to the defined acceptance criteria.

Phase 4: Transition (Deliverable)

- ✓The final version of the software;
- ✓User manuals.

Phase 4: Transition (Evaluation criteria)

- ✓ User satisfaction;
- ✓ The financial situation of the project compared to the initial budget.

Activities

- ✓ Each phase includes several iterations
- ✓ For each iteration, we engage in several activities:
  - ✓ Business modeling;
  - ✓ Expression of requirements;
  - ✓ Analysis;
  - ✓ . Design;
  - ✓ Implementation;
  - ✓ Test;
  - ✓ Deployment.

- ✓ Activities are steps in software development, but at a much finer level of granularity than phases.
- ✓ Each activity is repeated as many times as there are iterations.

Business Modeling Activities

- ✓ Objective: Better understand the structure and dynamics of the organization:
  - ✓ Propose the best solution in the context of the client organization;
  - ✓ Realization of a glossary of business terms;
  - ✓ Mapping of business processes of the client organization.
- ✓ Costly activity but which speeds up understanding of a problem.

Requirements expression activity

- ✓ Objective: Target the requirements of users and customers through a series of interviews.
  - ✓ All of the project's stakeholders, project management and contracting authority, are involved in this activity.
- ✓ The activity of collecting and expressing requirements leads to what the system must do (question “WHAT?”)

Needs expression activity

- ✓ Using use cases to:
  - ✓ Diagram (draw) the requirements;
  - ✓ Structure functional specification documents.
- ✓ The use cases are broken down into system usage scenarios, in which the user “tells” what he does using the system and his interactions with the system.
- ✓ A mock-up can be carried out to better “immerse” the user in the future system.
- ✓ Once the functional limits have been established, the project is planned and a cost forecast is made.

Analysis activity

- ✓ Objective Transform user requirements into UML models.
  - ✓ Object analysis serving as a basis for reflection on the internal mechanisms of the system.
- ✓ Main deliverables:
  - ✓ Analysis models, neutral with regard to technology;
  - ✓ Delivers a more precise specification of needs.
- ✓ Can be considered as a first draft of the design model.

Design Activity

- ✓ **Objective**: Model how the system will work:
  - ✓ Non-functional requirements;
  - ✓ Technological choices.
- ✓ The system is analyzed and we produce:
  - ✓ A proposal of architecture;
  - ✓ A division into components.

Implementation Activity

- ✓ Objective: Implement the system by components.
  - ✓ The system is developed in pieces that are dependent on each other.
  - ✓ Optimization of the use of resources according to their expertise.
- ✓ Functional and layered cuts are essential for this activity.
- ✓ It is entirely possible to rework the analysis and design models at this stage.

Test Activity

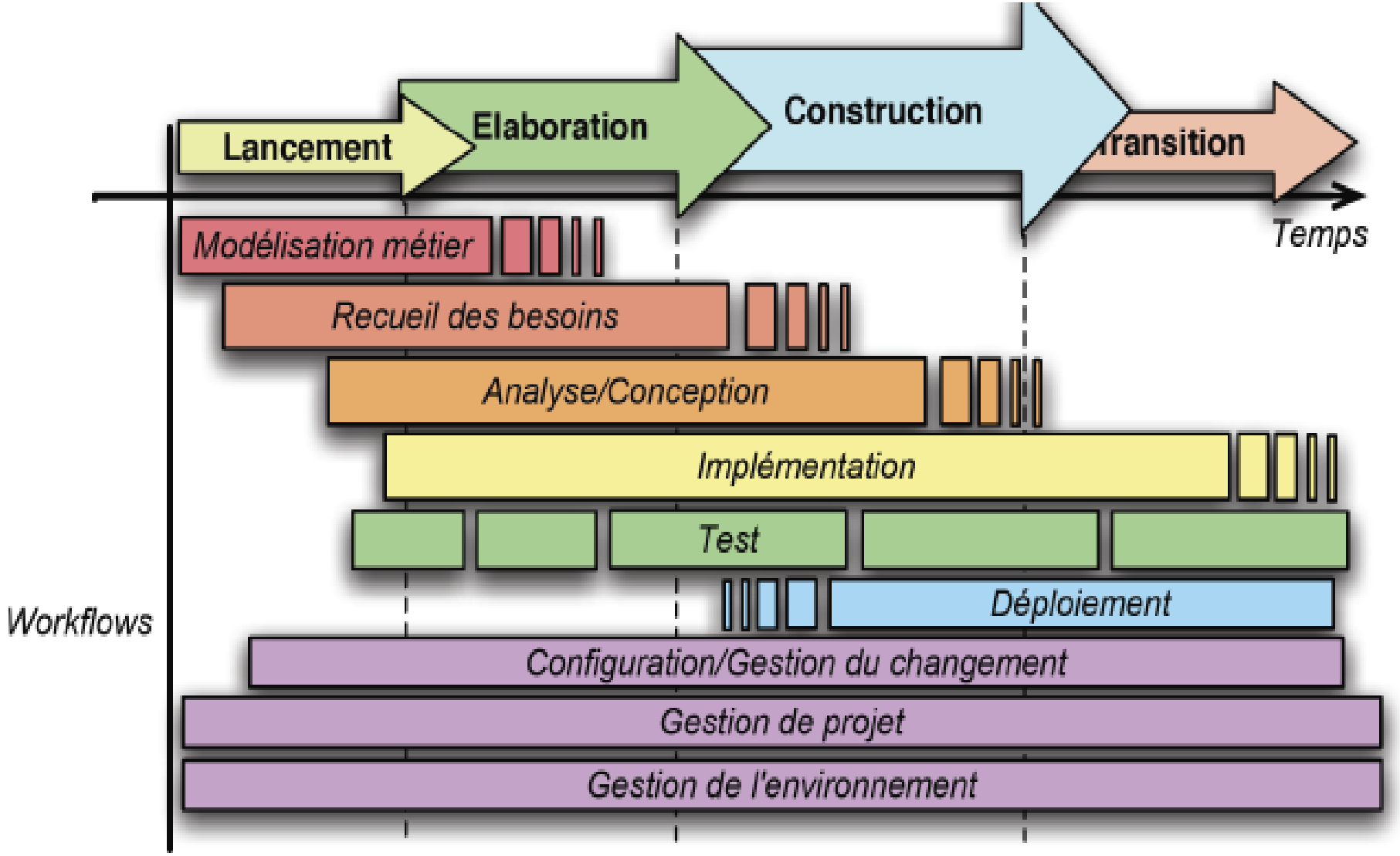
- ✓ **Objective**: Verify the results of the implementation by testing the construction:
  - ✓ Unit testing: component-by-component testing;
  - ✓ Integration tests: tests of the interaction of components previously tested individually.
- ✓ Method :
  - ✓ Planning for each iteration;
  - ✓ Implementation of tests by creating test cases;
  - ✓ Run the tests;
  - ✓ Take into account everyone's results.

Deployment activity

Objective: Deploy developments once completed.

- ✓ Can be done very early in the process in a prototyping sub-activity whose objective is to validate:
  - ✓ The physical architecture;
  - ✓ Technological choices.

Distribution and importance of activities in each phase



The main diagrams by activity

- ✓ Expression of requirements and business modelling:
  - ✓ Business models, domain, use cases;
  - ✓ Sequence diagram;
  - ✓ Activity diagram.
- ✓ Analysis
  - ✓ Business models, use cases;
  - ✓ Class, sequence and deployment diagram.
- ✓ Design
  - ✓ Class, sequence diagram;
  - ✓ State/transition diagram;
  - ✓ Activity diagram;
  - ✓ Deployment and component diagram.

Conclusion

- ✓ The RUP is both a methodology and a ready-to-use tool (web repository)
- ✓ Target projects of **more than 10 people**
- ✓ **Strong points**
  - ✓ Specifies the dialogue between the different stakeholders in the project: deliverables, schedules, prototypes, etc.
  - ✓ Offers document templates and templates for typical projects
- ✓ **Weak points**
  - ✓ Expensive to Customize: Battery of Consultants
  - ✓ Very process-oriented, to the detriment of development