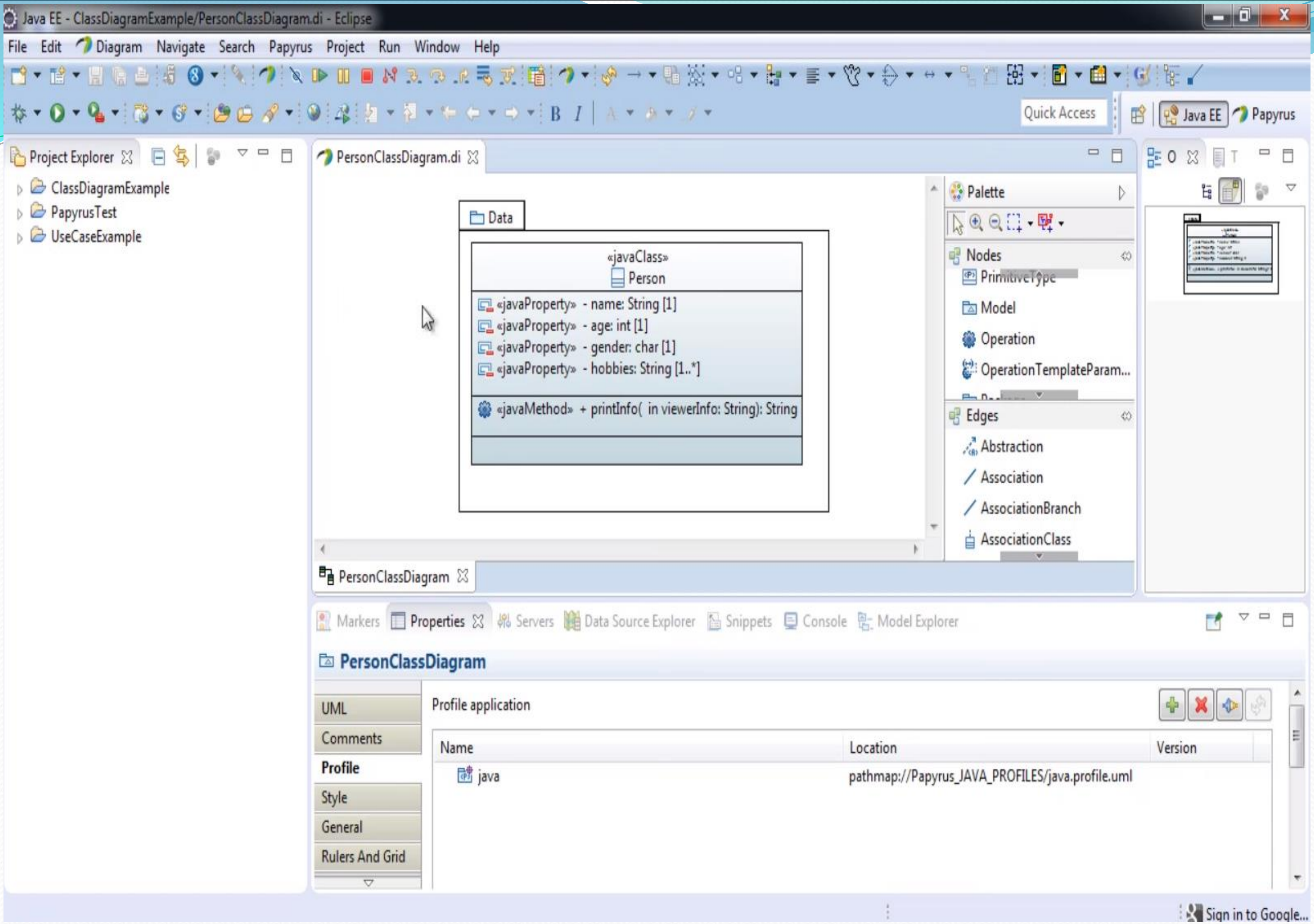
Lecture 10 - Implementation and maintenance of information systems

Content

- ✓ **Implementation of information systems**
  - ✓ Building the system
  - ✓ Putting the new system into operation
- ✓ **Computer system maintenance**
  - ✓ The concept of computer system maintenance
  - ✓ Types of maintenance
  - ✓ Maintenance process
  - ✓ Controlling maintenance requirements
  - ✓ Documentation needed in the maintenance process
  - ✓ Managing maintenance staff
  - ✓ Maintenance costs

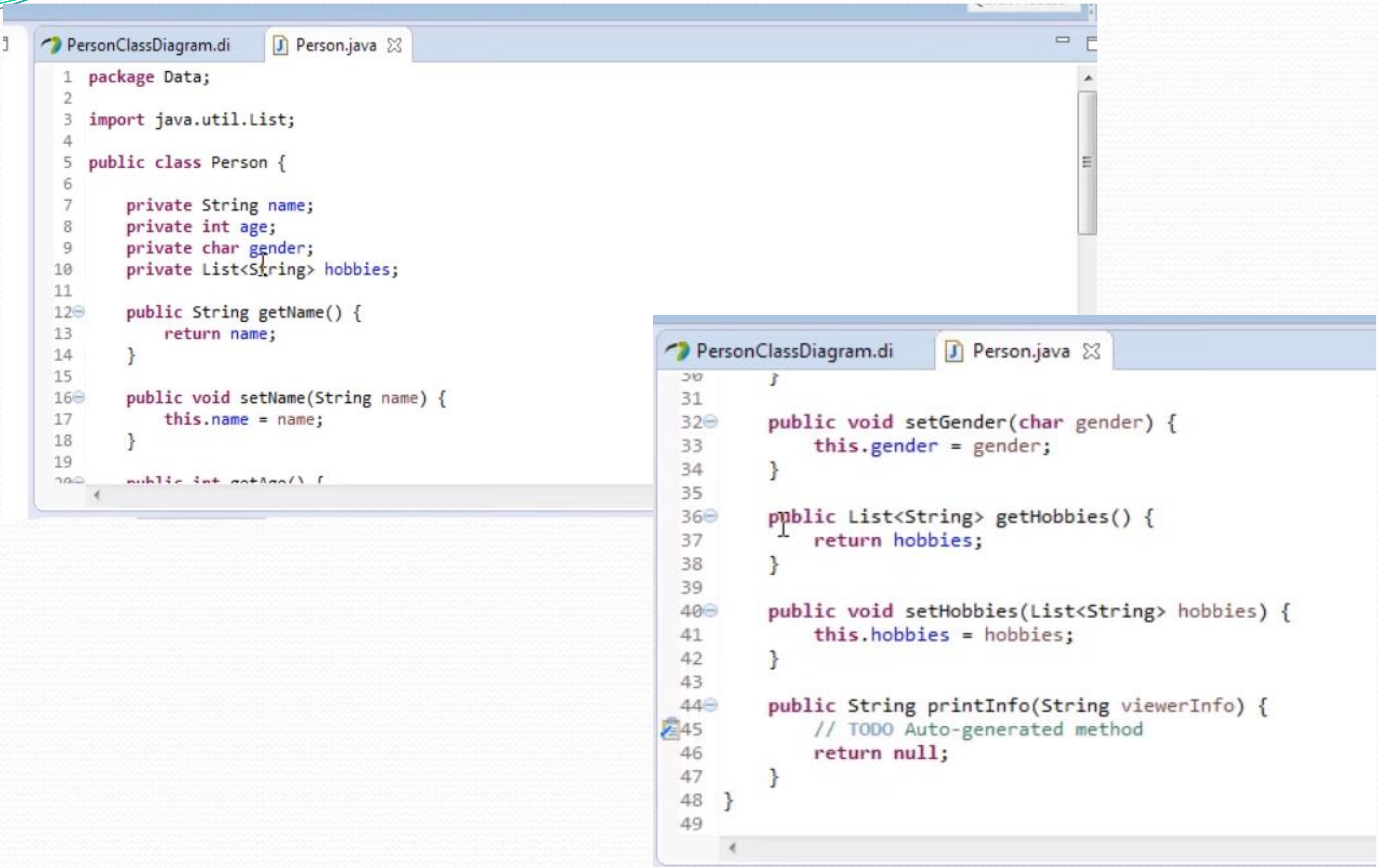
Generate code from UML diagrams using CASE tools

- CASE tools often allow **direct engineering** (direct code generation from UML charts) and **reverse engineering** (obtaining / updating UML charts in the code)
- These features ensure the continuity between the analysis-design-implementation stages and provide developers with a starting point for implementation
- The **Reverse Engineering** facility ensures the consistency between the changes taking place in the code and the models developed at the design stage



Name	Location	Version
java	pathmap://Papyrus_JAVA_PROFILES/java.profile.uml	

Using Papyrus for Eclipse for code generation in Java



```java
package Data;

import java.util.List;

public class Person {

    private String name;
    private int age;
    private char gender;
    private List<String> hobbies;

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public int getAge() {
```

```java
    }

    public void setGender(char gender) {
        this.gender = gender;
    }

    public List<String> getHobbies() {
        return hobbies;
    }

    public void setHobbies(List<String> hobbies) {
        this.hobbies = hobbies;
    }

    public String printInfo(String viewerInfo) {
        // TODO Auto-generated method
        return null;
    }
}
```

System building

Testing the programs

- The **test data se**t, identified in the programming stage, is used to test the modules written in the previous step.
- If the data you selected does not fully satisfy (not all branches and all the situations in the module) can be changed accordingly.
- Program modules are tested **functionally** and **semantically** (logically) by comparing the results obtained at execution with those proposed to be obtained. Testing ends when the two categories of results are identical.
- The **testing approach** can be done on the **top-down** or **bottom-up** module tree.

System building

Testing the programs

- **The top-down approach** starts from the **management module** to the functional and operational ones. The directory module is tested first, and then one at a time; the testing of **operational modules** on procedures, and then **links** between modules by testing on **different functional variants**. Independent testing of the operational modules can be performed in parallel with the functional test. Testing ends when the last operational procedure is tested and integrated into the module tree.
- **The bottom-up approach** starts from the **operational modules** to the **functional ones** in order to finally get to the **management module**. Each module at the lower levels of the tree is first tested separately and the bindings with other modules are tested and the integration is performed. After all the modules have been tested, the management module is tested and then the entire tree is assembled. Assembling has to be done following step by step the main functions of the program.
- The testing will always produce **multiple versions** of modules and of software programs, the latter being the one accepted. For each version, an evaluation is made and a correction is applied.

System building

Documentation preparation

- This documentation is written in the form of manuals dedicated to different categories of computer system users. Developing:
  - ✓ the development documentation,
  - ✓ the presentation manual,
  - ✓ the user and exploitation manual.

A. Development documentation

- It includes documents from the technical project and from the stages of the program product realization.
- In particular, the **technical aspects** (how the program product was achieved) are presented, and therefore the documentation is addressed primarily to IT specialists and, possibly, to intensive users.
- By going through this documentation, the specialist can find the software **solutions that have been adopted** to implement the program product. This documentation is also a model and source of ideas for other programs.

System building – Development documentation

The content of the program documentation:

1. Overview of the software product

Contains:

- the name of the software and the computer system it is part of;
- presentation of the purpose of the information system. It also presents the purpose of the software product within the IT system it is part of;
- the starting point for the software product. Usually this basis is given by the technical project of the computer system, and if there are changes to this project they must be signaled.

2. The general structure of the software product

Contains:

- presentation of the product tree structure. By decomposing the software product, a series of modules and links are obtained. They are graphically represented as a tree. The result is a functional scheme that is widely described;
- the modules in the tree are actually the functional components of the program product. All of these components are described by highlighting their content and functions.

System building - Development documentation

3. Description of data collections - contains:

- the solution chosen for **data collections**: **files** or **database**. It justifies the choice made and presents the advantages and disadvantages;
- the **list of data collections** and their significance. It is based on the study and analysis of the problem solved;
- description of **links** between data collections. It specifies which collections are linked, their type and a brief description of them. If the database solution is chosen, the (conceptual, external and internal) schemes will be presented from the design of the database (including their design).

4. Developing program modules – contains:

- the **system schema for each module** will highlight the peripherals used and how to access them;
- **function descriptions** for each module;
- presentation of **inputs, processing, outputs and data collections** used by each module;
- the **logical / pseudo-code schema** for each module. There are also some technical data on how the code was written;
- the **links** of each module with other program components. Specify the role of each link and how to achieve them.

5. Anexe – contains:

- **menu layouts, formatted screens, windows** of program modules;
- **report layouts** and **sample test data;**
- **the source code** of the program product.

System building – Presentation manual

B. Presentation manual

- It includes **general presentation** of the program: purpose, facilities, restrictions, necessary resources and conditions of use. Through its content, the manual addresses **all categories of users.**
- The manual will not contain technical data on how to write the program, but only the overall aspects, accessible to all.
- The size of the manual must be reduced and the presentation must be clear and concise.

1. General aspects of the program – contains:

- **the objectives** pursued by the program product to highlight its usefulness;
- **program performance and limits:** Specifies what it can do and what it can not do for a particular industry;
- **the field of use** addressed by the program product. It is built to solve a problem or a category of problems. The field of use must be delimited as clearly as possible.

2. The functional structure of the program – contains:

- **the overall schema** comprising **the functional components of the program** and **the links** between them;
- **the presentation of each component of the program**: it contains a brief description and its purpose.

System building – Presentation manual

3. Presentation of elements made for the program product

- presentation of the **inputs** on existing types. Input layouts, video formats, menus, and windows used;
- presentation of the **processing** of the program product. We present the **algorithms** used without detailing the actual algorithms. The name of the algorithm, the function and the result obtained are indicated;
- show **outputs** from the program product. The layout of the reports is presented with test examples;
- presentation of the **data collections** created and manipulated by the program. If the **file** solution is chosen, the following are displayed: the name and content of the file solution, the description of the fields. If the **database** solution is chosen, the data structure is presented: the data model used and the conceptual schema

4. Program terms of use – contains:

- **the resources** needed to run the program product. This includes the material resources (computing equipment, auxiliary materials, data storage space, etc.) and human resources by user category;
- **the restrictions and conditions** necessary for the implementation of the program product and its exploitation;
- **the estimated efficiency** of using the program

System building– User and exploitation manual

C. User and exploitation manual This manual contains a description of the instructions on the conditions of use (preliminary and actual) and exploitation (effective implementation) of the program. These instructions are mainly addressed to end-users, that is to say, the proper users of the IT system.

1. Instrucţiuni de utilizare – contine:

- **data coding procedures** providing instructions on how the user codes are produced. This explains the coding system used and the code structure (meaning on characters). If applicable, specify the validation criteria for each code and any automatic encodings made by the system;
- **data loading / validation procedures**, giving instructions on how to collect data collections. Here are the primary documents from which the data are taken and how they are filled in. This compares with input templates and video formats for primary documents. Instructions are given on validation criteria for uploaded data and how to correct errors;
- **procedures for obtaining reports**, giving instructions on how to display and interpret reports, lists, etc. For each report the layout, the content, the periodicity of production and examples are given. The instructions concern not only how to obtain, but also interpret them;
- other **special procedures** giving instructions on any conversions, interfaces, and communications required by the program.

System building

2. Instructions for use

- **data coding procedures** providing instructions on how the user codes are produced. This explains the coding system used and the code structure (meaning on characters). If applicable, specify the validation criteria for each code and any automatic encodings made by the system;
- **data loading / validation procedures,** giving instructions on how to collect data collections. Here are the primary documents from which the data are taken and how they are filled in. This compares with input templates and video formats for primary documents. Instructions are given on validation criteria for uploaded data and how to correct errors;
- **procedures for obtaining reports**, giving instructions on how to display and interpret reports, lists, etc. For each report the layout, the content, the periodicity of production and examples are given. The instructions concern not only how to obtain, but also interpret them;
- other **special procedures** giving instructions on any conversions, interfaces, and communications required by the program.

II. Putting the new system into operation

Objectives:

- experimenting with the projected system;
- finishing the new system;
- launching it in current operation at the beneficiary;
- reception of the projected computer system.

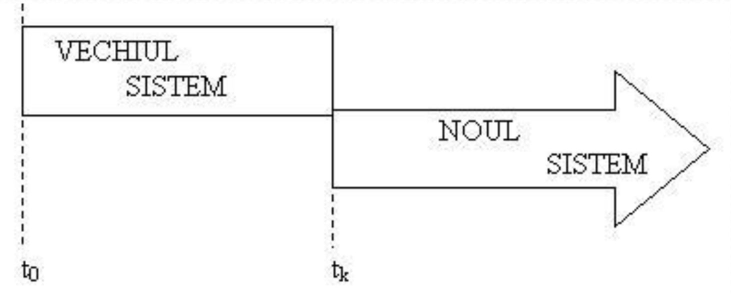
The main activities to be carried out in the implementation phase to achieve these objectives are:

- ensuring the conditions for system operation;
- performing conversion procedures
- putting the system into operation;
- verification of system performance;
- finalizing the documentation;
- system approval.

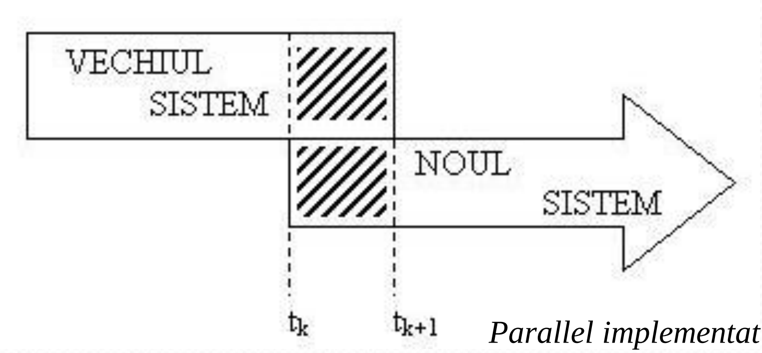
Implementation strategies and tactics

- Since IT systems differ in their nature and complexity and each beneficiary has different concrete conditions, the implementation of the system can be done through different strategies and tactics from one unit to another.
- The choice of one or other of the deployment strategies depends on a number of factors, such as: the degree of professional, material and psychological training of the beneficiary, the nature, complexity and originality of the information system, the volume of data and the diversity of information sources, the degree of satisfaction the information requirements of the beneficiary and its participation in the implementation of the new system.
- The strategies for implementing an IT system as described in the following are:
  - ❑ direct implementation;
  - ❑ implementation of the IT system in parallel with the maintenance of the old system until the new system is put into operation;
  - ❑ pilot implementation.

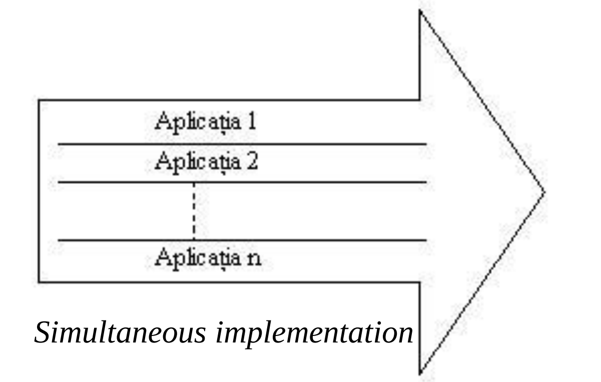
Putting the new system into operation



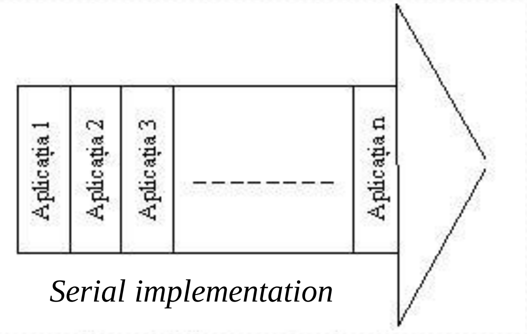
Direct Implementation



Parallel implementation



Simultaneous implementation



Serial implementation

Selecting the implementation strategy and tactic

Criteria:

i. the degree of professional training of computer system users;
ii. the degree of material and psychological training of the beneficiary;
iii. the nature and complexity of the designed computer system;
iv. the originality of the computer system (entirely original system, original system using prefabricated elements, adaptation of an existing computer system);
v. the degree of participation of the beneficiary in the implementation of the information system during the previous stages;
vi. the volume of data and the diversity of their sources;
vii. the degree of satisfaction of the information requirements by the old and the new system.

Implementation tactics

Choosing implementation tactics depends on:

- the number of specialists of the beneficiary available to implement the system implementation,
- the beneficiary's readiness for implementation
- the amount of data available at the time of implementation.

Regarding the data on which the IT system is implemented, these can be:

- **data from the previous period** - it is advisable to practice when the implementation coincides with the peak activity period in the functional sectors. During this time, the use of data not trained in current processing does not disturb the process of recording, processing, and transmitting data used in the old system. After comparing and verifying the results obtained, it is necessary to **switch to the use of current data.**
- **current data.**

Verification of the performance of the IT system

- Verification of the performance of the IT system is achieved by evaluating the **results achieved** by the testing and implementation of all IT products **compared to the requirements and constraints made at the design stage**.
- At the same time, the **verification** will also be taken into consideration:
    - ensuring the necessary conditions for the current operation of the system,
    - ensuring data processing capability,
    - how the designed operating rate (exploitation graph) is achieved both at user level and in-service performance (such as response time, safety, etc.).

- Verification also involves evaluating and validating the results obtained by calculating the **economic efficiency indicators**.
- If **deficiencies** are found during verification, they must be removed. These can be corrected immediately or, if there are deeper causes from design, in a longer time.

Finalizing the documentation for the designed system

- The implementation phase of the IT system is completed when **the new system is in line with the project requirements**;

- Therefore, the designer must draw up the "**Implementation / Experimental Report**", which includes, on the one hand, a brief presentation of the system and the conditions in which the implementation took place (resources, scope etc.) and, on the other hand, the evaluation of the results of implementation / experimentation (performance, costs, efficiency) with a series of indications regarding the transition to the current exploitation and possibly the generalization of the use of the new system.

- The implementation stage can not be considered complete unless **the entire system documentation is finalized.**

Approval / acceptance of the IT system

- **Approval / acceptance involve**s the beneficiary's acceptance of the computer system for use or generalization following verification by the approval / reception committee of how the computer system complies with the rules established by the project.
- When the user receives the first results from the computer it is said that the system has entered into the **current operation**. This implies that the system will have to be constantly modified, adapted and refined according to the requirements of the system of activity it serves.

- The **finalization of the documentation** is carried out in parallel with the other activities within the implementation phase, through a close collaboration between the beneficiaries and the designer,

- The current trend is to develop information systems so that they can **partially resume the previous activities** without any major problems. Thus, if certain operating parameters of the system are not met at the implementation, they may return to the design or even to analysis.

The concept of computer system maintenance

- Maintenance does not only concern hardware, software or business procedures, but maintenance also involves correcting or removing errors, errors, or omissions from the design and system design phase.

- System maintenance and development is a vital issue for most systems. Whatever the reasons, any system will surely require change.

- From the above it can be deduced that **the maintenance of information systems represents the activity of monitoring, evaluation and modification of the system in order to make the desired or necessary improvements.**

- The maintenance activity includes a post-implementation review process to ensure that the newly implemented IT systems meet predefined objectives, requirements and performance. Any errors in the creation or operation of the system must be corrected within the maintenance process.

Types of maintenance

Depending on the purpose pursued, the nature of the changes and the urgency of making changes to the system, there are several types of maintenance, such as: corrective, adaptive, perfection and preventive maintenance.

1. Corrective maintenance refers to making changes within the system in order **to repair and remove defects or errors** in design, programming, or implementation. Programming errors are usually easier to make and less costly. Design errors are difficult to achieve and more costly because they can involve rewriting multiple program components. User information errors are the most difficult and expensive to correct because they require more extensive redesign of the system.

By analogy, for example, if you recently bought a new home, corrective maintenance will involve repairing things that have never worked under the project, such as an electrical failure, a poorly aligned door, poorly leakproof windows, etc. Practical activity estimates that corrective maintenance amounts to about 75% of the costs of all types of maintenance.

2. Adaptive maintenance involves making changes to the information system **to enhance its functionality or to adapt it to a new environment** such as a different hardware platform or to use another operating system or other database management system and so on For example, for some synthetic information / reporting situations required by the company's management, shown in tabular form, they are required to be rendered in graphic form as well. Regarding the previous example with the home, adaptive maintenance would mean adding thermopan windows and air conditioning.

Adaptive maintenance is less urgent than corrective, and represents a small part of system maintenance, adding value.

Types of maintenance

3. Perfective maintenance involves implementing **new functional or non-functional system requirements**, such as improving process performance, easy user interfaces, reducing customer waiting times for booking and hotel accommodation, etc.

So through perfect maintenance it is intended to add to the system the desired traits, but not necessarily necessary. In the example of this house it could mean adding a new room. It is precisely for these reasons that perfect maintenance is seen as a new development and not maintenance (maintenance).

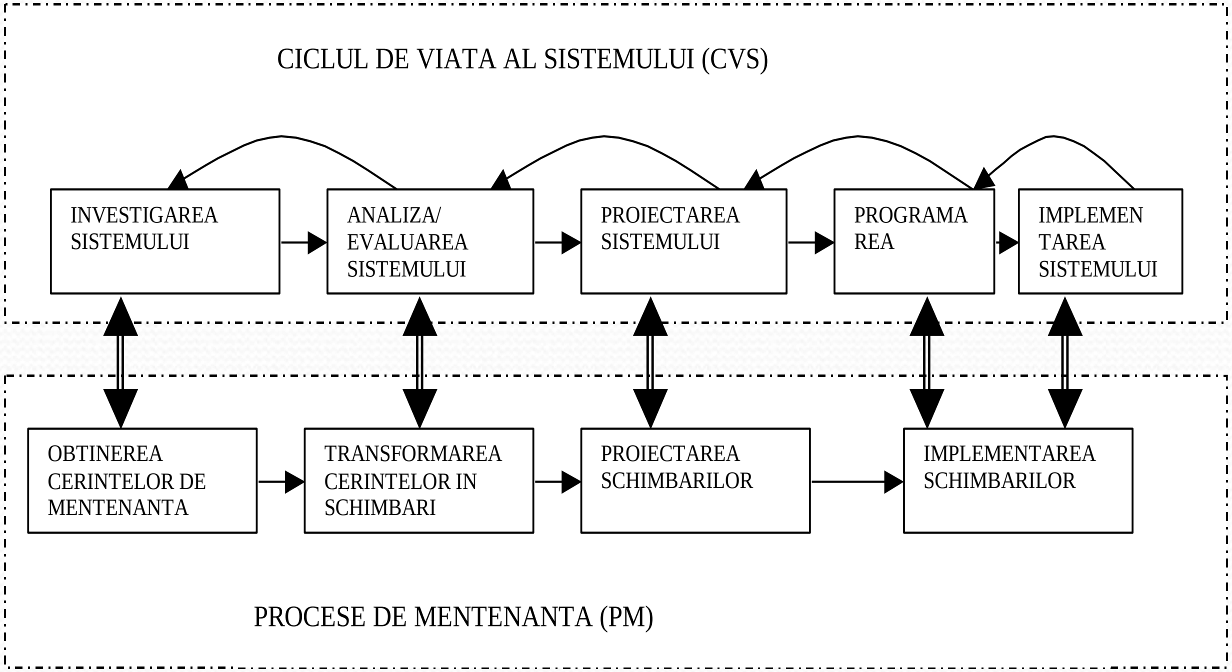
Example: For the raw materials supply subsystem, a new application is added to prevent the immobilization of circulating media or process ruptures.

4. Preventive maintenance involves changes made to the system to **reduce or eliminate the risk of system failure** due to causes such as increasing the number of records well above those currently processed. Another case could be to provide backups and updates tracking logs to restore the database in case of an incident. In the example of housing, preventative maintenance would consist of exterior painting or dyeing of carpentry for weather protection. Like perfection, preventative maintenance is of lower priority. Over the life cycle of the system, corrective maintenance is most likely to occur after the initial installation or after major system changes. In such a final situation, adaptive, perfectional and preventive maintenance, if not carefully designed and implemented, can lead to corrective maintenance

The maintenance process

- The maintenance process begins immediately after the implementation and putting into operation of the informatic system. Includes the following activities:
    - **obtaining maintenance requirements;**
    - **transforming requirements into changes;**
    - **designing changes;**
    - **implementing changes.**
- Maintenance requirements can be developed during the current implementation or operation of the information system by end-users, project managers or customers, constituting a set of change requirements.
- For collecting and requesting change requests, a type of document called "System Service Request" is required, which requires new developments, reports of problems that have arisen or new system features are required for an existing system. All system service requests are collected by a pre-set person within the maintenance team.
- From the way the maintenance process takes place it can be deduced that there are similarities and differences between the life cycle of a computer system (LCS) and maintenance process(MA).

The maintenance process


CICLUL DE VIATA AL SISTEMULUI (CVS)
INVESTIGAREA SISTEMULUI
ANALIZA/ EVALUAREA SISTEMULUI
PROIECTAREA SISTEMULUI
PROGRAMA REA
IMPLEMEN TAREA SISTEMULUI
OBTINEREA CERINTELOR DE MENTENANTA
TRANSFORMAREA CERINTELOR IN SCHIMBARI
PROIECTAREA SCHIMBARILOR
IMPLEMENTAREA SCHIMBARILOR
PROCESE DE MENTENANTA (PM)

It can be noted that "System Investigation" is similar to "Obtaining Maintenance Requirements" in MP, "System Analysis / Assessment" corresponds to "Transforming Requirements into Changes", "Designing the System" corresponds to "Designing Changes“, "Implementation of the system" corresponds to Implementation of changes ".

Controlul cerinţelor de mentenanţă

Pe parcursul implementării și exploatării curente a sistemului pot apărea o serie de **cerințe de schimbare**, unele referitoare la corectarea defectelor minore sau severe din sistem, în timp ce altele se referă la îmbunătățirea și extinderea funcţionalităţii sistemului.

Într-un astfel de context apare problema modului de abordare sub aspectul care dintre ele să fie ignorate şi care să fie executate şi cu ce urgenţă. Întâi trebuie determinat **tipul cerinței**. Dacă, de exemplu, cerinţa e o eroare, adică o cerinţă de mentenanţă corectivă, atunci trebuie pusă o întrebare legată de gravitatea erorii.

Dacă eroarea este foarte gravă, atunci cerinţa are **prioritate** maximă şi e plasată în fruntea unui şir de sarcini ce aşteaptă să fie executate la sistem. Exemplu, o greşeală în sistemul de afaceri al clientului/beneficiarului trebuie rapid corectată pentru a permite continuarea normală a afacerii.

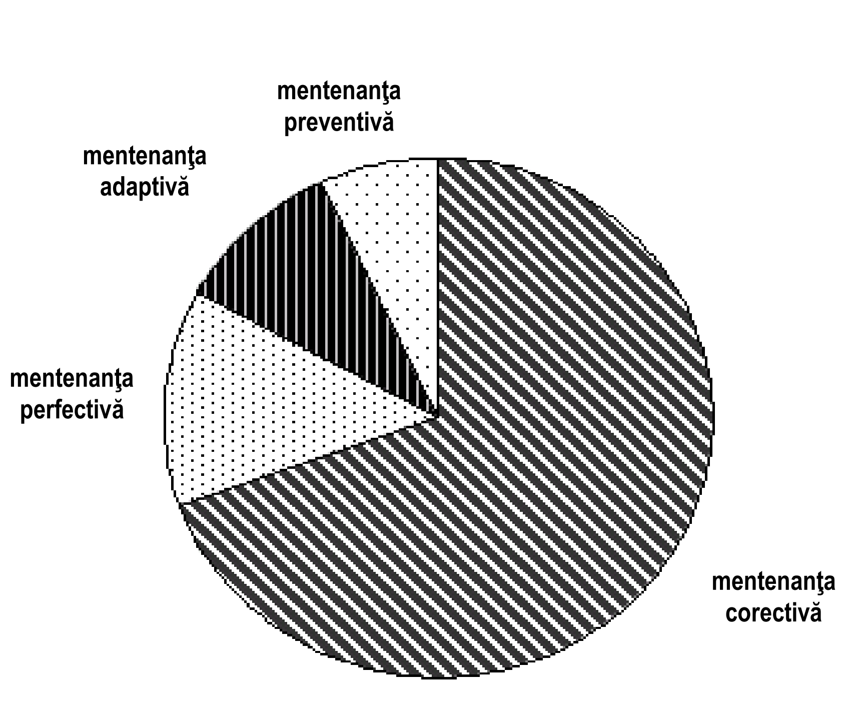
Acest procedeu de reparare de urgenţă este necesar dacă problemele raportate afectează disponibilitatea sistemului. Proiectul şi programele treptat ies din cadenţă. E dificil să se evite o astfel de situaţie ţinând seama de faptul că inginerii de mentenanţă pot fi presaţi să se ocupe de alte reparaţii de urgenţă la software.

Maintenance costs

- Maintenance requires the ease with which the information system can be understood, corrected, adapted and strengthened. Low maintenance systems result in uncontrollable maintenance costs.

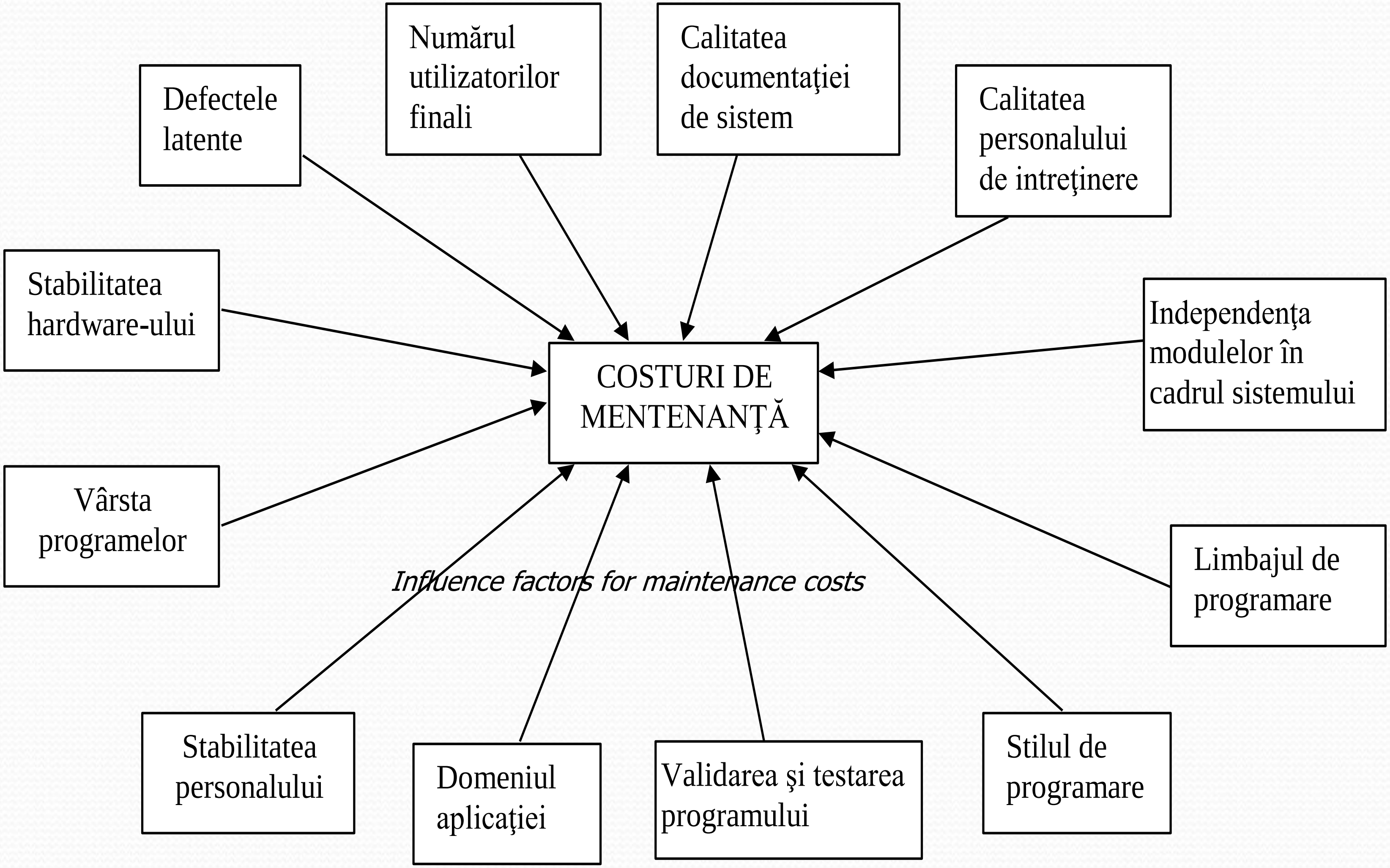
- Regarding the maintenance costs we can make a number of clarifications, among which:
  - **Over time, the maintenance costs have increased**, given that in the first instance, the focus was naturally on the system creation process. Then it has come to the fact that many beneficiary units have old systems that require more and more maintenance costs.
  - Maintenance costs **vary greatly from one scope to another**. For IT systems or business applications, it is estimated that maintenance costs are largely comparable to system realization. For other types of systems, such as real-time systems, maintenance costs are up to 3-4 times higher than those occasioned by system creation. High viability, operational safety and system performance requirements may require modules to be closely interconnected and therefore difficult to change.
  - Maintenance efforts involve different weights of resources, varying for different types of maintenance

Maintenance costs



It can be deduced from the figure that the weight of the **corrective maintenance costs is the highest**, despite the fact that it does not add a new value to the IT system, but only corrects in order to overcome some errors. From the point of view of economic efficiency, it is desirable to invest more effort in analyzing, designing and implementing the IT system to further reduce maintenance costs. The cost of adding functionality to the system after it has been put into operation is much greater than providing a similar functionality at the beginning when the system was created.

Maintenance costs – influence factors



Defectele latente

Numărul utilizatorilor finali

Calitatea documentaţiei de sistem

Calitatea personalului de intreţinere

Stabilitatea hardware-ului

COSTURI DE MENTENANŢĂ

Independenţa modulelor în cadrul sistemului

Vârsta programelor

Limbajul de programare

Stabilitatea personalului

Domeniul aplicaţiei

Validarea şi testarea programului

Stilul de programare

Influence factors for maintenance costs