Software Project Management

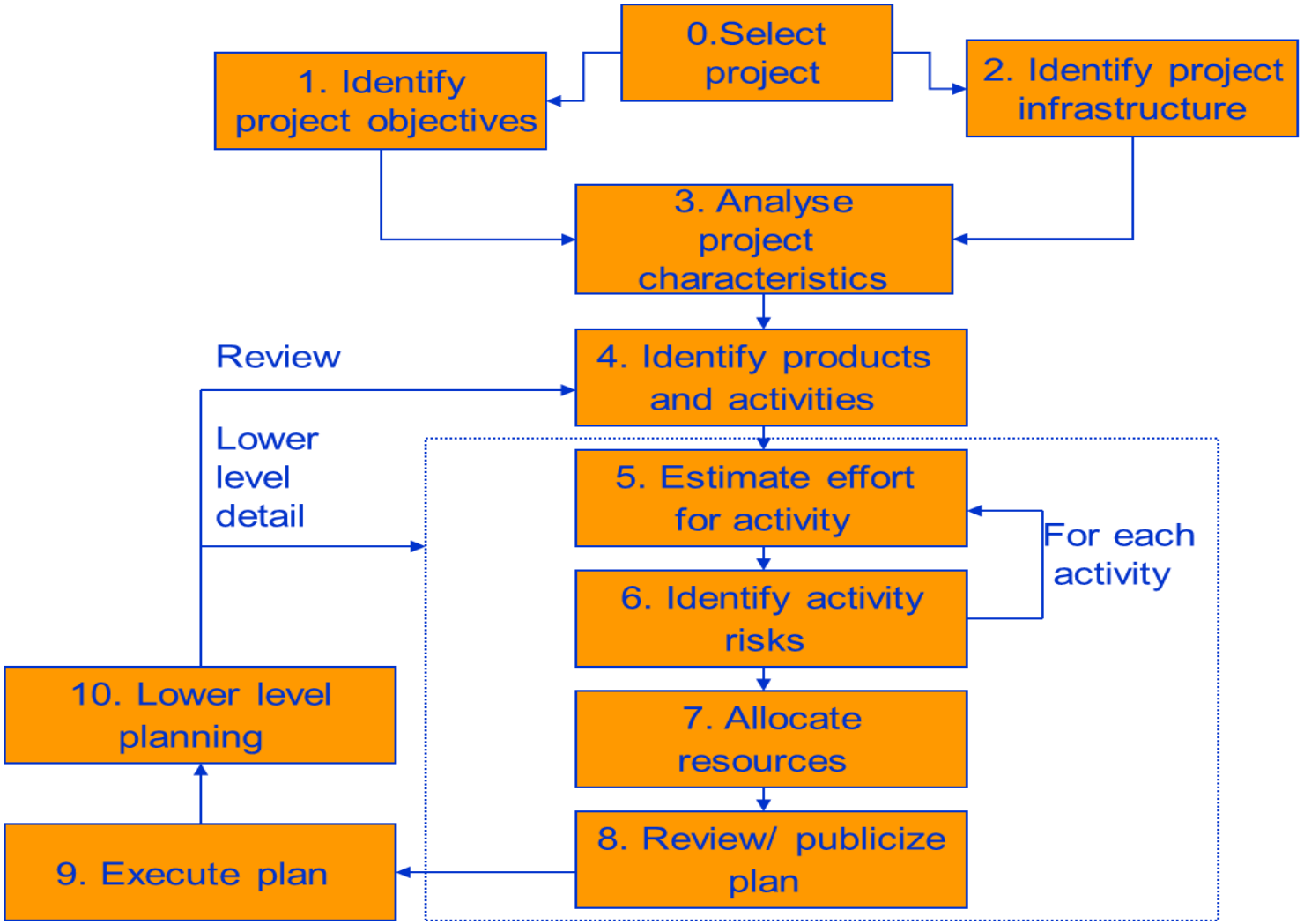
Durga Prasad Mohapatra
Professor
CSE Deptt.
NIT Rourkela

Selection of an appropriate project approach

Outline of lecture

- Building versus buying software
- Taking account of the characteristics of the project
- Process models
  - Waterfall model
  - Prototyping model
  - Iterative model
  - Incremental model
  - Agile models

'StepWise' - an overview



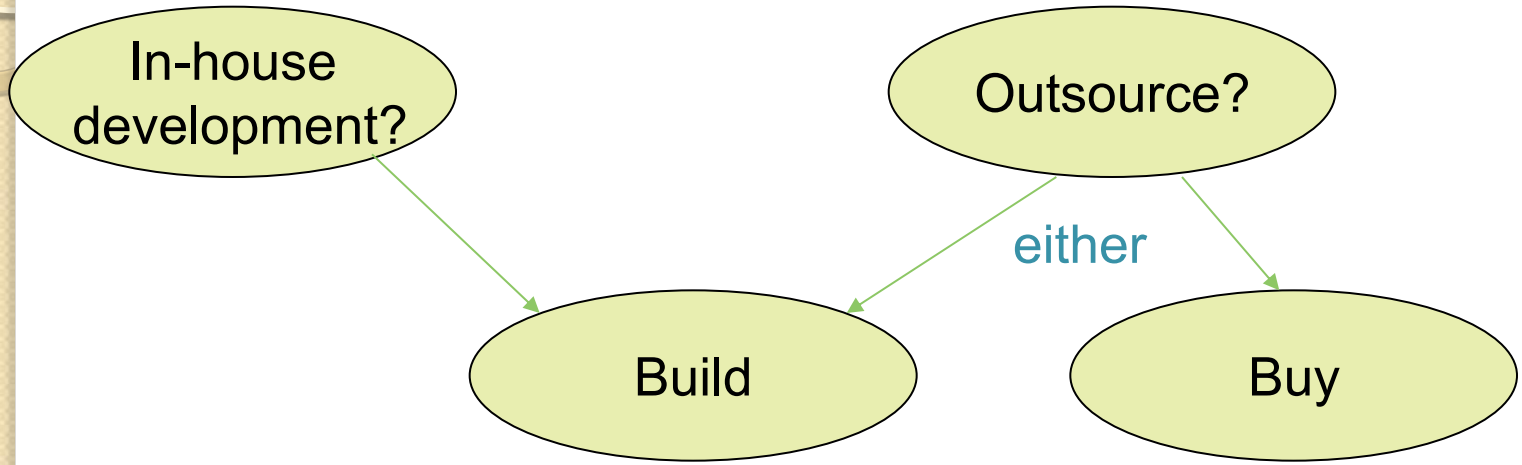
Selection of project approaches

- This lecture concerned with choosing the right approach to build a particular project: variously called technical planning, project analysis, methods engineering and methods tailoring.
- Relevant to Step 3, of the previous figure.
- In-house: often the methods to be used dictated by organizational standards
- Suppliers: need for tailoring as different customers have different needs

Selection of project approaches

- Different types of projects need different types of approach.
- If you are working in one particular environment which specializes in one type of software, then the approach is likely not to change much from one project to another.
- Where you work for different clients in different organizations developing a variety of applications, then the approach for each project may need to be tailored.

Build or buy?



Build or buy?

- In-house development almost always involves developing new code.
- If it is decided to use a specialist organization to implement system, the supplier could either build a 'bespoke' system (created specially for one customer) for you, or could install a pre-existing application.
- There are hybrids of these options, e.g. to build in-house but use temporary contract staff, or Customised Off-the-Shelf (COTS) where a basic existing core system is modified for your particular requirements.

Some advantages of off-the-shelf (OTS) software

- Cheaper as supplier can spread development costs over a large number of customers
- Software already exists
  - Can be trialled by potential customer
  - No delay while software being developed
- Where there have been existing users, bugs are likely to have been found and eradicated

Some possible disadvantages of off-the-shelf

- Customer will have same application as everyone else: no competitive advantage, but competitive advantage may come from the way application is used
- Customer may need to change the way they work in order to fit in with OTS application
- Customer does not own the code and cannot change it
- Danger of over-reliance on a single supplier
- One concern is what happens if the supplier goes out of business. Customer might not then be able to maintain the system: hence the use of 'escrow' services where a 3rd party retains a copy of the code.

General approach

- Look at risks and uncertainties e.g.
  - are requirement well understood?
  - are technologies to be used well understood?
- Look at the type of application being built e.g.
  - information system? embedded system?
  - criticality? differences between target and development environments?
- Clients' own requirements
  - need to use a particular method

General approach

Control systems:

- A real-time system will need to be implemented using an appropriate methodology. Real-time systems that employ concurrent processing may have to use techniques such as Petri nets.

General approach

Information systems:

- An information system will need a methodology, such as Information Engineering, that matches that type of environment.
- Team members would therefore know exactly what is expected.

General approach

Availability of users:

- Where the software is for the general market rather than application and user specific, then a methodology which assumes that identifiable users exist who can be quizzed about their needs would have to be thought about with caution.
- Some business systems development methods assume an existing clerical system which can be analysed to yield the logical features of a new, computer-based, system.
- In these cases a marketing specialist may act as a surrogate user.

General approach

Specialized techniques:

- For example, expert system shells and logic-based programming languages have been invented to expedite the development of *knowledge-based systems*.
- Similarly, a number of specialized techniques and standard components are available to assist in the development of *graphics-based systems*.

General approach

Hardware environment :

- The environment in which the system is to operate could put constraints on the way it is to be implemented.
- The need for a fast response time or restricted computer memory might mean that only low-level programming languages can be used.

General approach

Safety-critical systems:

- Where safety and reliability are essential, this might justify the additional expense of a formal specification using a notation such as OCL.
- Extremely critical systems could justify the cost of having independent teams develop parallel systems with the same functionality.
- The operational systems can then run concurrently with continuous cross-checking. This is known as *n-version programming*.

General approach

Imprecise requirements:

- Uncertainties or a novel hardware/software platform mean that a prototyping approach should be considered.
- If the environment in which the system is to be implemented is a rapidly changing one, then serious consideration would need to be given to *incremental delivery*.
- If the users have *uncertain objectives* in connection with the project, then a *soft systems* approach might be desirable.

Processes versus Process Models

- Starting from the inception stage:
  - A product undergoes a series of transformations through a few identifiable stages
  - Until it is fully developed and released to the customer.
  - This forms its life cycle or development process.
- Life cycle model (also called a process model):
  - A graphical or textual representation of the life cycle.

Structure versus speed of delivery

Structured approach

- Structured methods consists of sets of steps and rules which, when applied, generate system products such as use case diagrams.
- Each of these products is carefully defined.
- Such methods are more time consuming and expensive than more intuitive approaches.
- The pay-off, it is hoped, is a less error prone and more maintainable final system.

Structure versus speed of delivery

Structured approach (cont…)

- This balance of costs and benefits is more likely to be justified on a large project involving many developers and users.
- Because of the additional efforts needed and their greater applicability to large and complex projects, these are often called *heavyweight* methods.

Structure versus speed of delivery

Structured approach

- Also called 'heavyweight' approaches
- Step-by-step methods where each step and intermediate product is carefully defined
- Emphasis on getting quality right first time
- Example: use of UML and USDP
- Future vision: Model-Driven Architecture (MDA). UML supplemented with Object Constraint Language, press the button and application code generated from the UML/OCL model

Structure versus speed of delivery

Agile methods

- Emphasis on speed of delivery rather than documentation
- RAD (Rapid application development): Emphasized use of quickly developed prototypes
- JAD (Joint application development): Requirements are identified and agreed in intensive workshops with users

Choice of process models

- Based on the nature / characteristics of the project, choose an appropriate process model for developing the project. Some of the possible process models are:
  - Waterfall model (also known as 'one-shot', or 'once-through' model)
  - Prototyping model
  - Incremental model
  - RAD model
  - Agile models

What is Agile Software Development?

- Agile: Easily moved, light, nimble, active software processes
- How agility achieved?
  - Fitting the process to the project
  - Avoidance of things that waste time
- Emphasis on speed of delivery rather than documentation

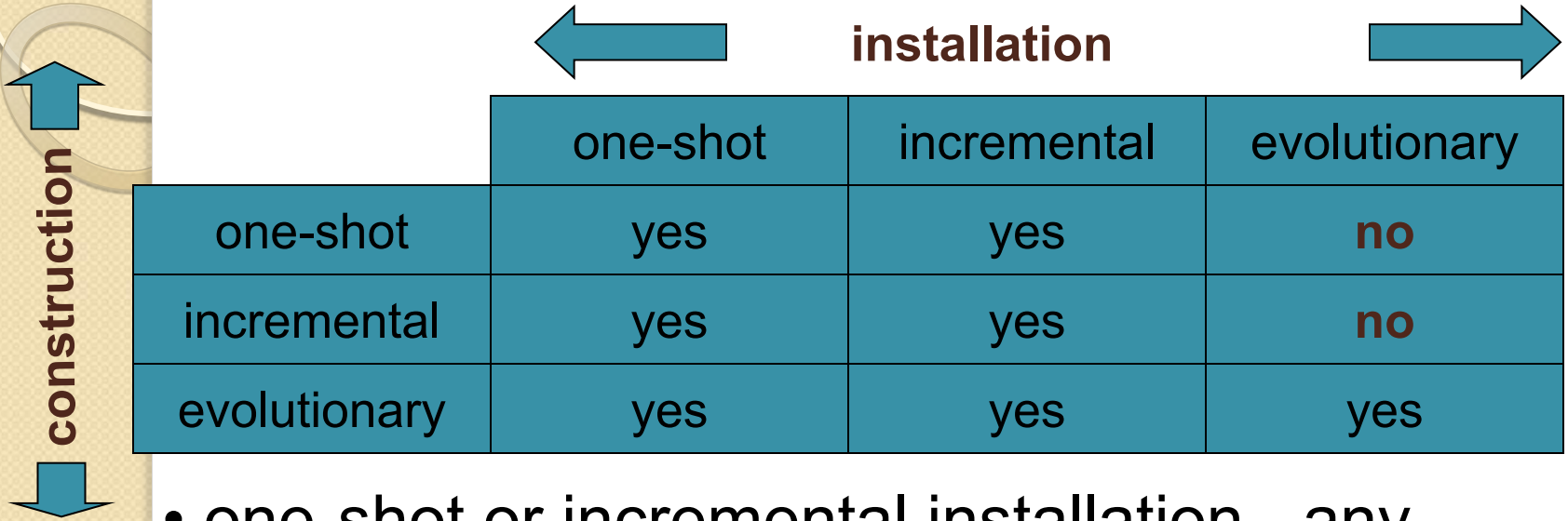
Important Themes of Agile Methods

- Incremental approach:
  - At a time, only one increment is planned, developed, and then deployed at the customer site.
- Emphasizes face-to-face communication over written documents.
- An agile project usually includes a customer representative in the team.
- Agile development projects usually deploy pair programming.(one will write the code, the other one will review the code, then the roles will be swapped).

Agile Models / Methodologies

- XP
- Scrum
- Unified process
- Crystal
- DSDM
- Lean

combinations of approaches



	one-shot	incremental	evolutionary
one-shot	yes	yes	**no**
incremental	yes	yes	**no**
evolutionary	yes	yes	yes

- one-shot or incremental installation - any construction approach possible
- evolutionary installation implies evolutionary construction

‘Rules of thumb’ about approaches to be used

IF uncertainty is high

THEN use evolutionary approach

IF complexity is high but uncertainty is not

THEN use incremental approach

IF uncertainty and complexity both are low

THEN use one-shot model

IF schedule is tight

THEN use evolutionary or incremental model

Comparison of Different Life Cycle Models

- Iterative waterfall model
  - most widely used model.
  - But, suitable only for well-understood problems.
- Prototype model

- is suitable for projects which are not well understood in the following aspects:
  - user requirements
  - technical aspects

Comparison of Different Life Cycle Models cont …

- Evolutionary model is suitable for large problems:
  - can be decomposed into a set of modules that can be incrementally implemented,
  - incremental delivery of the system is acceptable to the customer.
- **Spiral model:**
  - suitable for development of technically challenging software products that are subject to several kinds of risks.

Comparison of Different Life Cycle Models cont …

- Agile model:
  - to help projects to adapt to change requests
  - In the agile model, the requirements are decomposed into many small incremental parts that can be developed over one to four weeks each.

Summary

- We have discussed Building versus buying a software.
- We have discussed some characteristics of the projects such as control systems, information systems, availability of users, specialized techniques, hardware environment, safety-critical systems, imprecise requirements.
- We have mentioned different Process models.

References :

1. B. Hughes, M. Cotterell, R. Mall, *Software Project Management*, Sixth Edition, McGraw Hill Education (India) Pvt. Ltd., 2018.
2. R. Mall, *Fundamentals of Software Engineering*, Fifth Edition, PHI Learning Pvt. Ltd., 2018.

Thank you