Software lifecycle models (Contd.)

Major Difficulties of the Waterfall model

1. The most important difficulty is there is no way change requests can be handled in the waterfall model. In all waterfall based models that is the iterative waterfall model, the classical waterfall model, the prototyping model, the V model and so on. The requirements are gathered up front and these are documented, it is assumed that this will not change and then plan is made based on this documented requirements, design is done on this requirements and then coding and testing is done with respect to these requirements.

- But then in reality, the present projects the requirements keep on changing as development proceeds.
- Typically about 40 to 50 percent of the requirements change after the initial requirement specification

- The second problem is that the waterfall model is called as heavy weight process, it is called a heavy weight process because lot of documentation is produced as part of the process, everything is supposed to be documented., requirement specification, the reviews, review results, the design various types of plans everything is to be documented and that is the reason these are called as heavy weight processes.

- Typically 50 percent of the effort by a development team goes towards creating documents and naturally the project costs increase and there is project delays.
- However, nowadays the project durations are very short such heavy weight processes are not finding favor and we need some processes which do away with these problems.

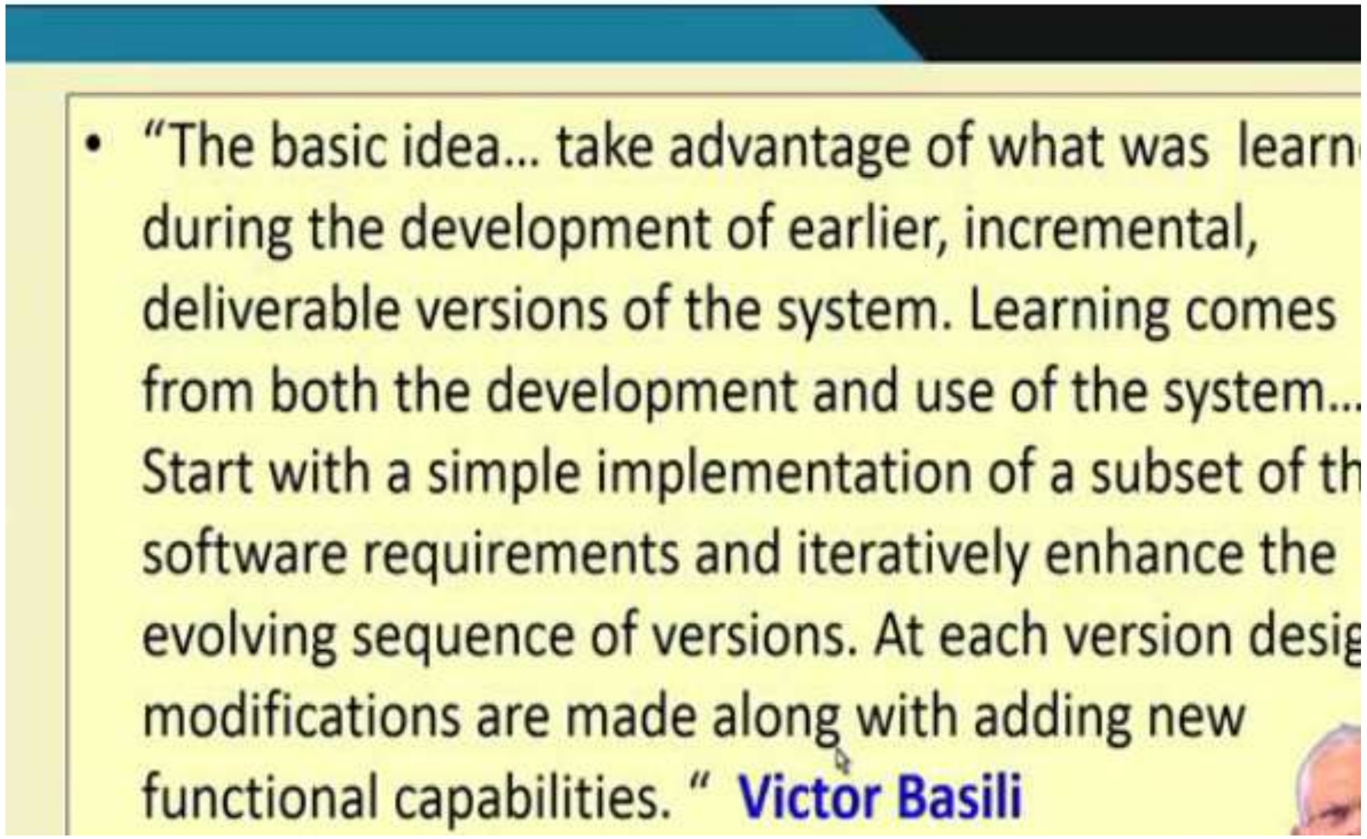
- And therefore, any change to the requirements will require changes to all documents and therefore, those who use the waterfall model heavily discourage the customer to make even the small changes.

- Many of the project failures, project delays etcetera are the result of having the requirements worked out at the beginning of the development.

- And, then all the development proceeds based on the documented requirements is fundamentally wrong and many of the problems arise from this.

- In response to this short coming of the waterfall model one model that was proposed is the incremental model.

The Iterative incremental model



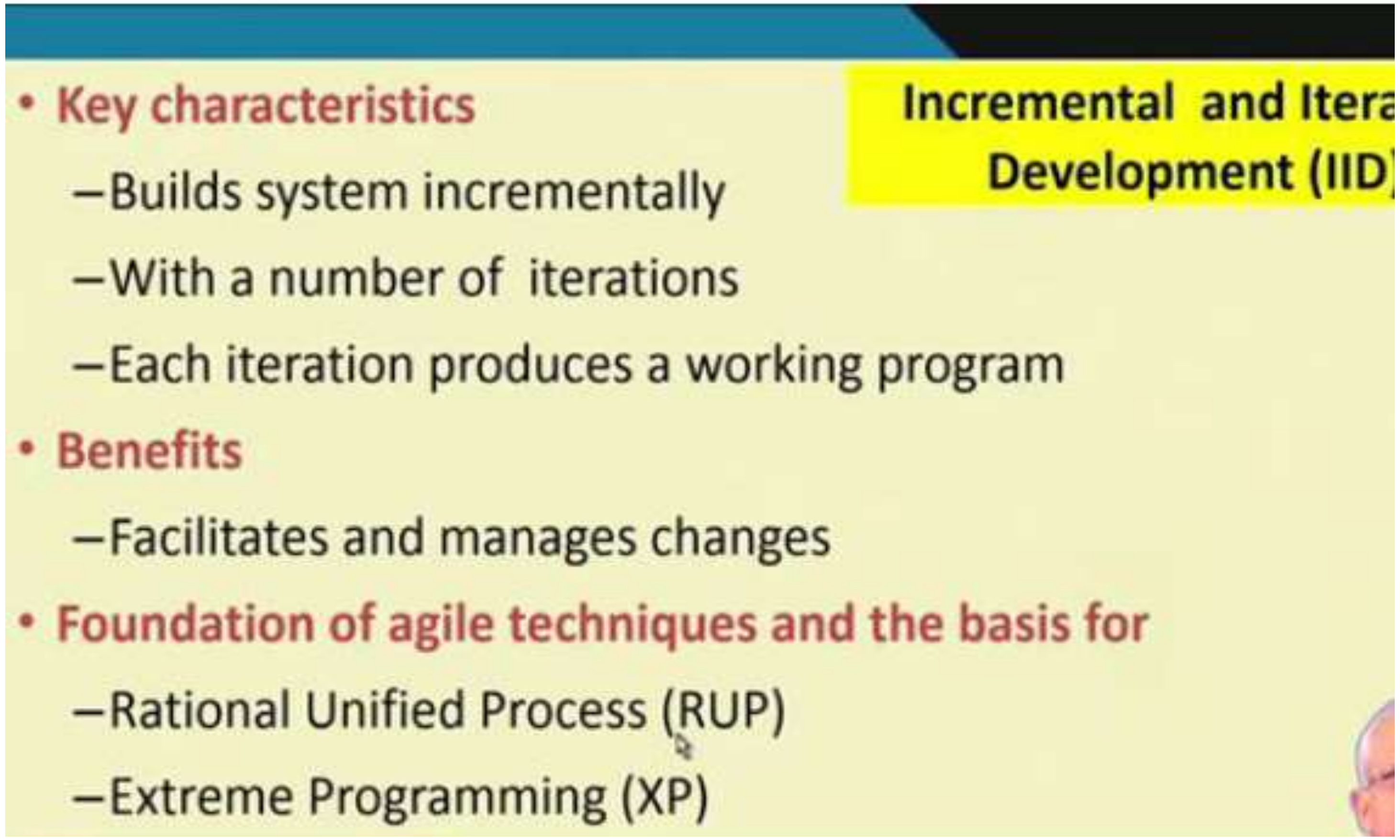
- "The basic idea... take advantage of what was learn
during the development of earlier, incremental,
deliverable versions of the system. Learning comes
from both the development and use of the system...
Start with a simple implementation of a subset of th
software requirements and iteratively enhance the
evolving sequence of versions. At each version desig
modifications are made along with adding new
functional capabilities. " **Victor Basili**

In incremental model the software is developed in increments.

In the first increment some of the core functionalities are implemented and then these are given to the customer who gives feedback and this forms a learning from the use of the system. And based on the feedback the system the software is refined and this is called as the iteration.

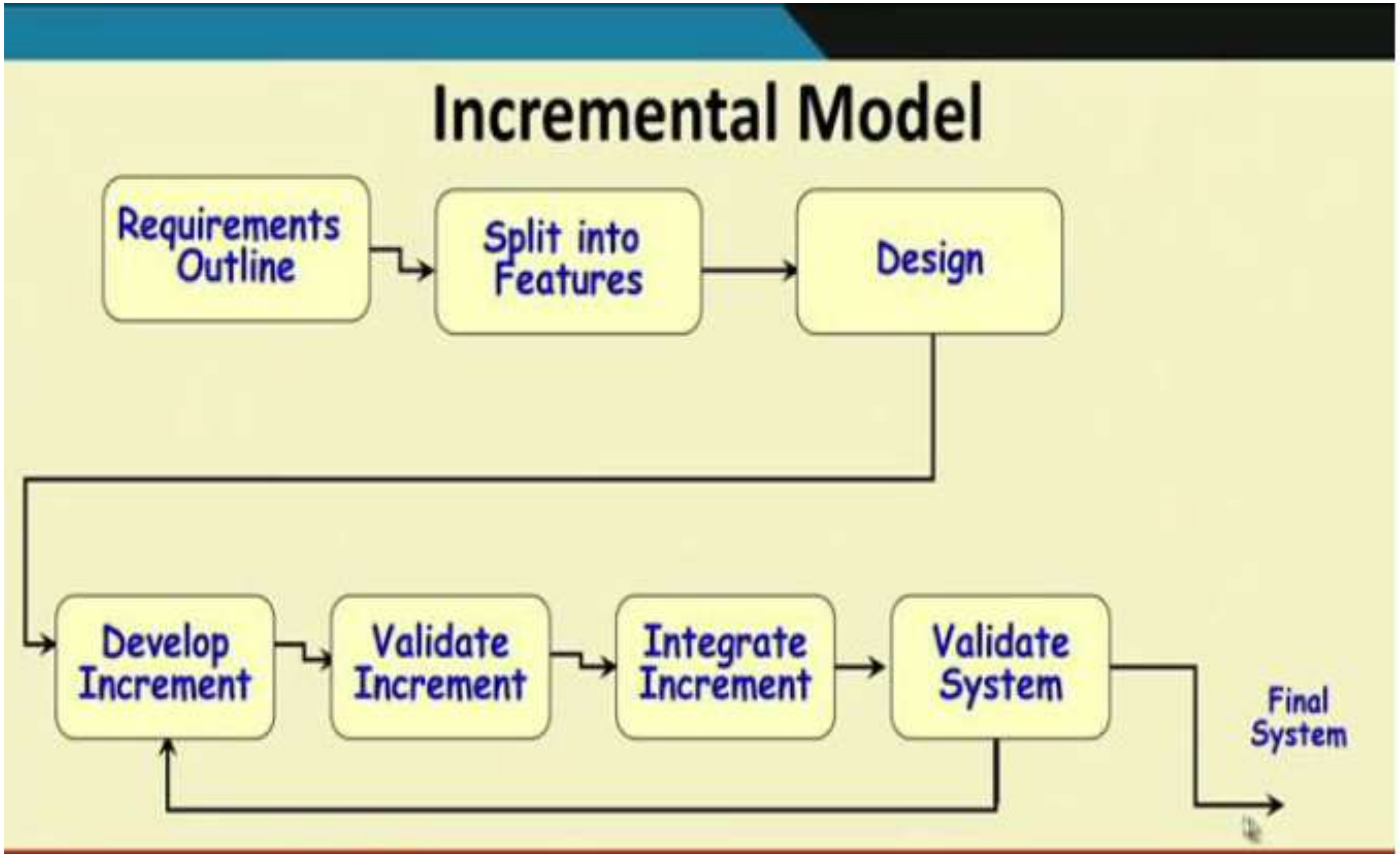
In iteration the same functionality is refined each time the customer gives new feedback, the same functionality gets refined iteratively and also at the same time new functionalities are also implemented these are called as increments.

So, this model is called as incremental model with iteration.

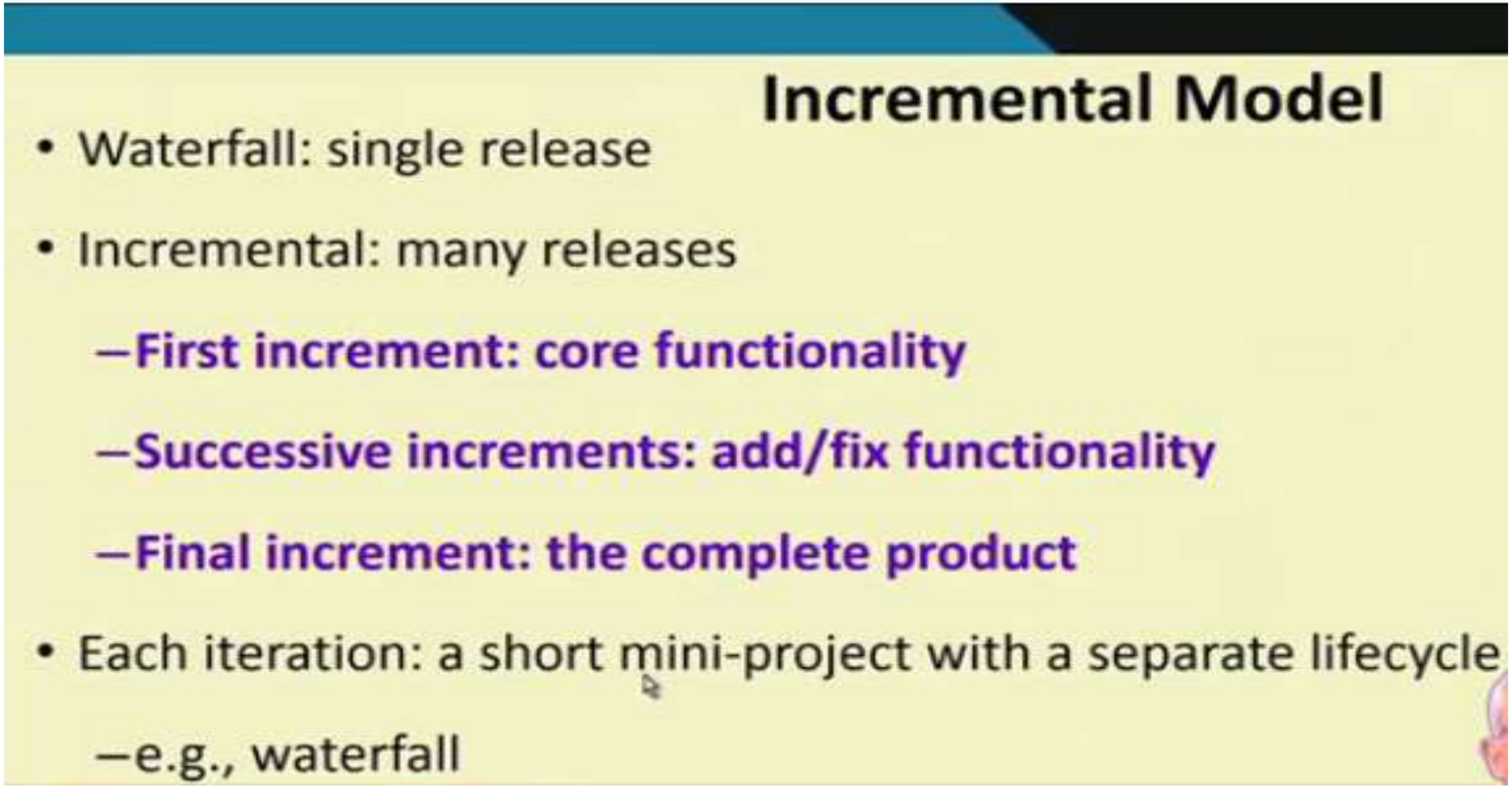


Incremental and Itera
Development (IID

- **Key characteristics**
  - –Builds system incrementally
  - –With a number of iterations
  - –Each iteration produces a working program
- **Benefits**
  - –Facilitates and manages changes
- **Foundation of agile techniques and the basis for**
  - –Rational Unified Process (RUP)
  - –Extreme Programming (XP)



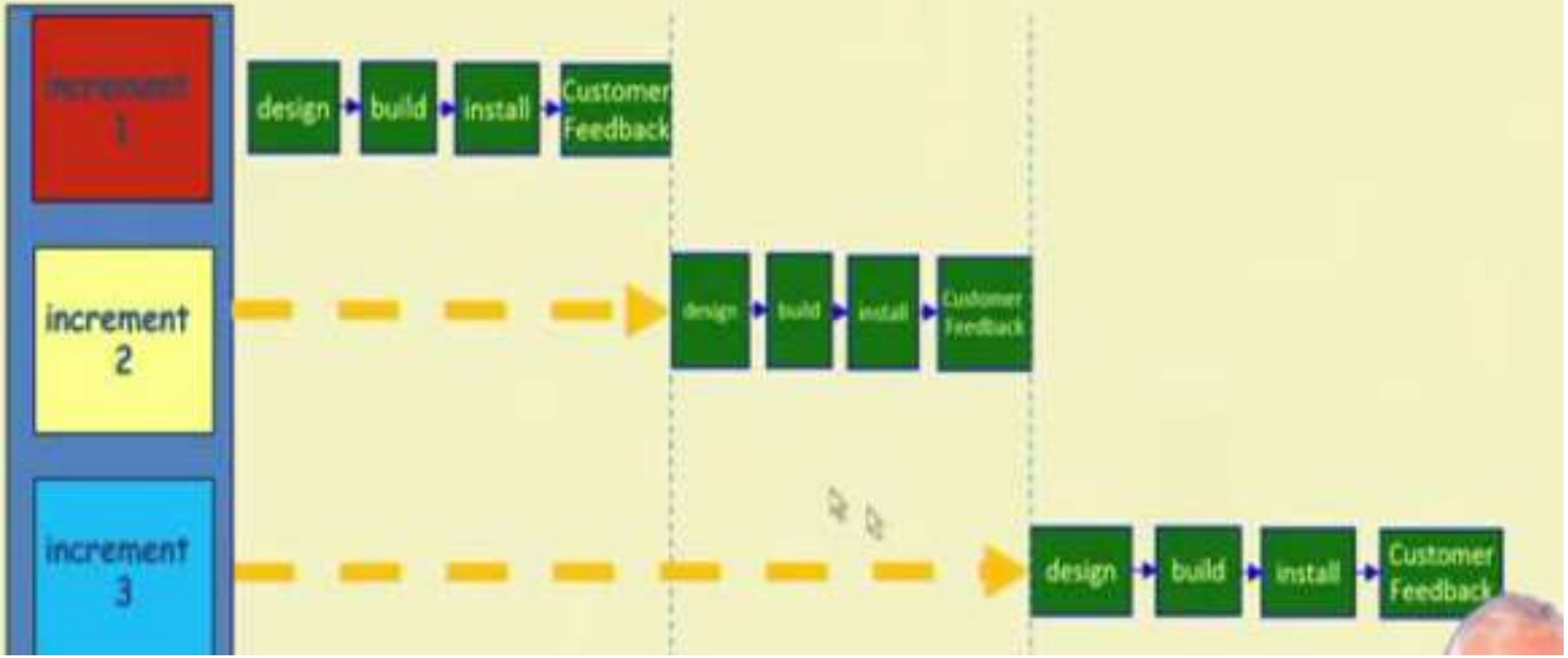
Incremental Model

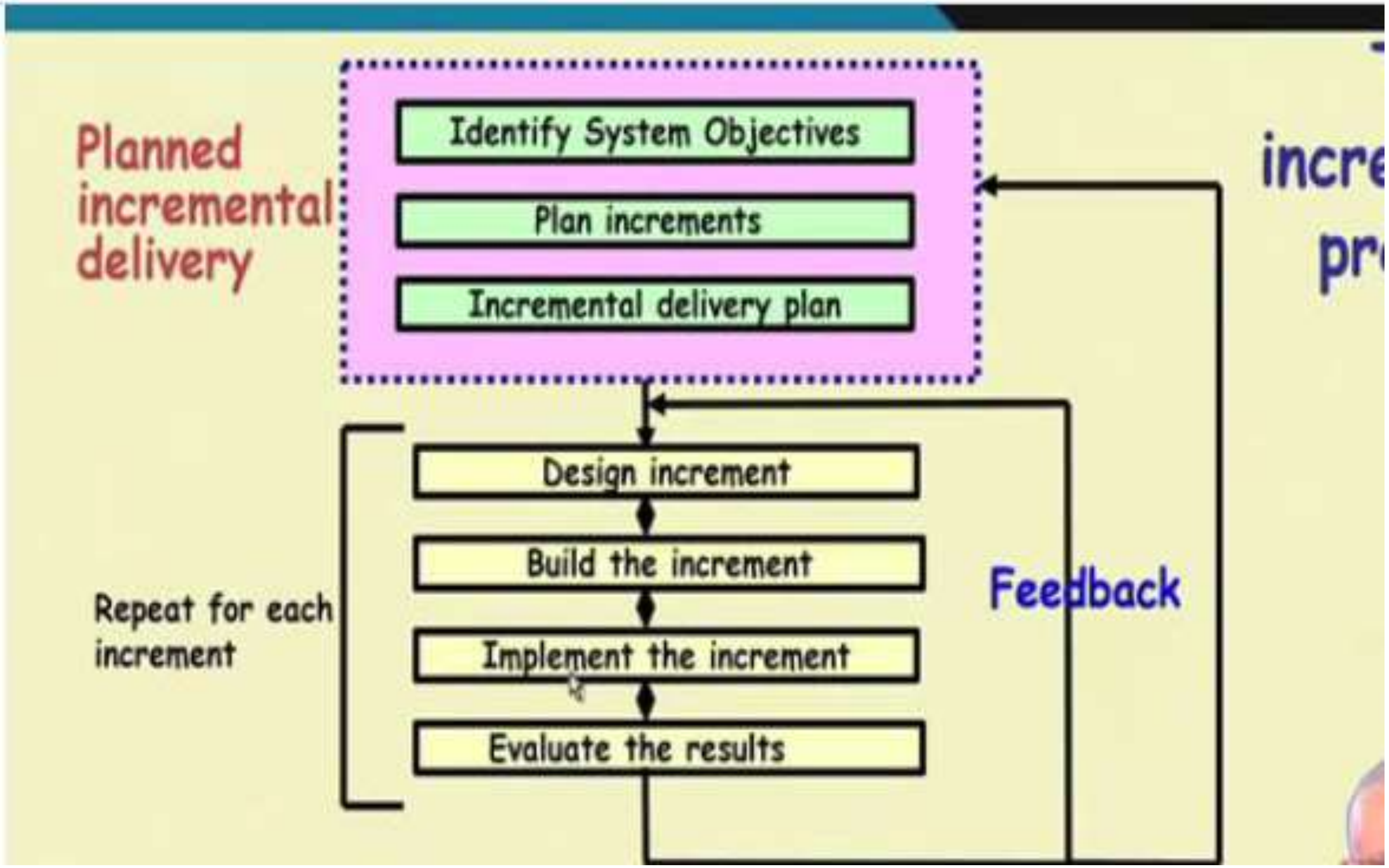


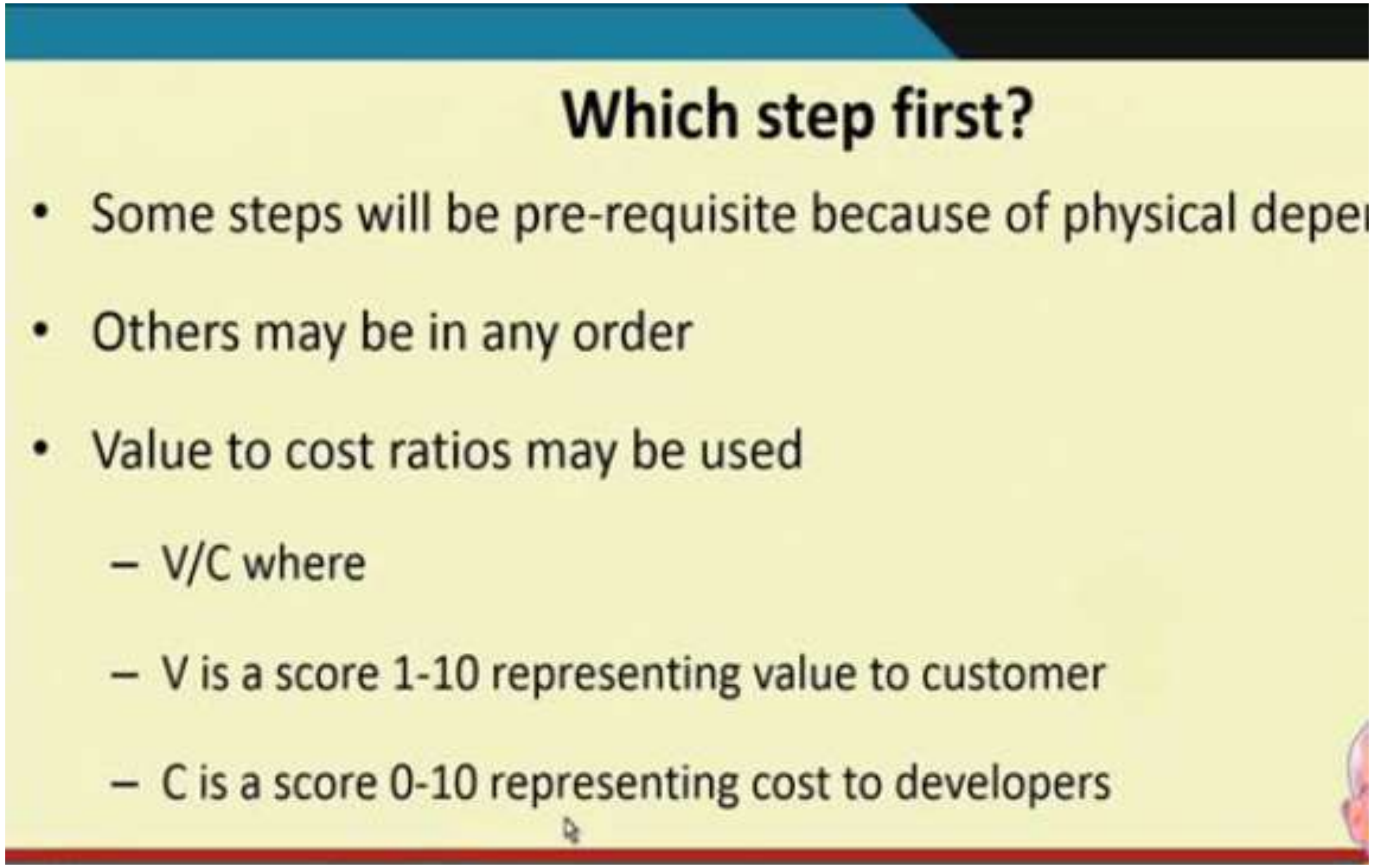
Incremental Model

- Waterfall: single release
- Incremental: many releases
  - **First increment: core functionality**
  - **Successive increments: add/fix functionality**
  - **Final increment: the complete product**
- Each iteration: a short mini-project with a separate lifecycle
  - e.g., waterfall

Incremental delivery




Planned incremental delivery
Identify System Objectives
Plan increments
Incremental delivery plan
Design increment
Build the increment
Implement the increment
Evaluate the results
Repeat for each increment
Feedback
incre
pr

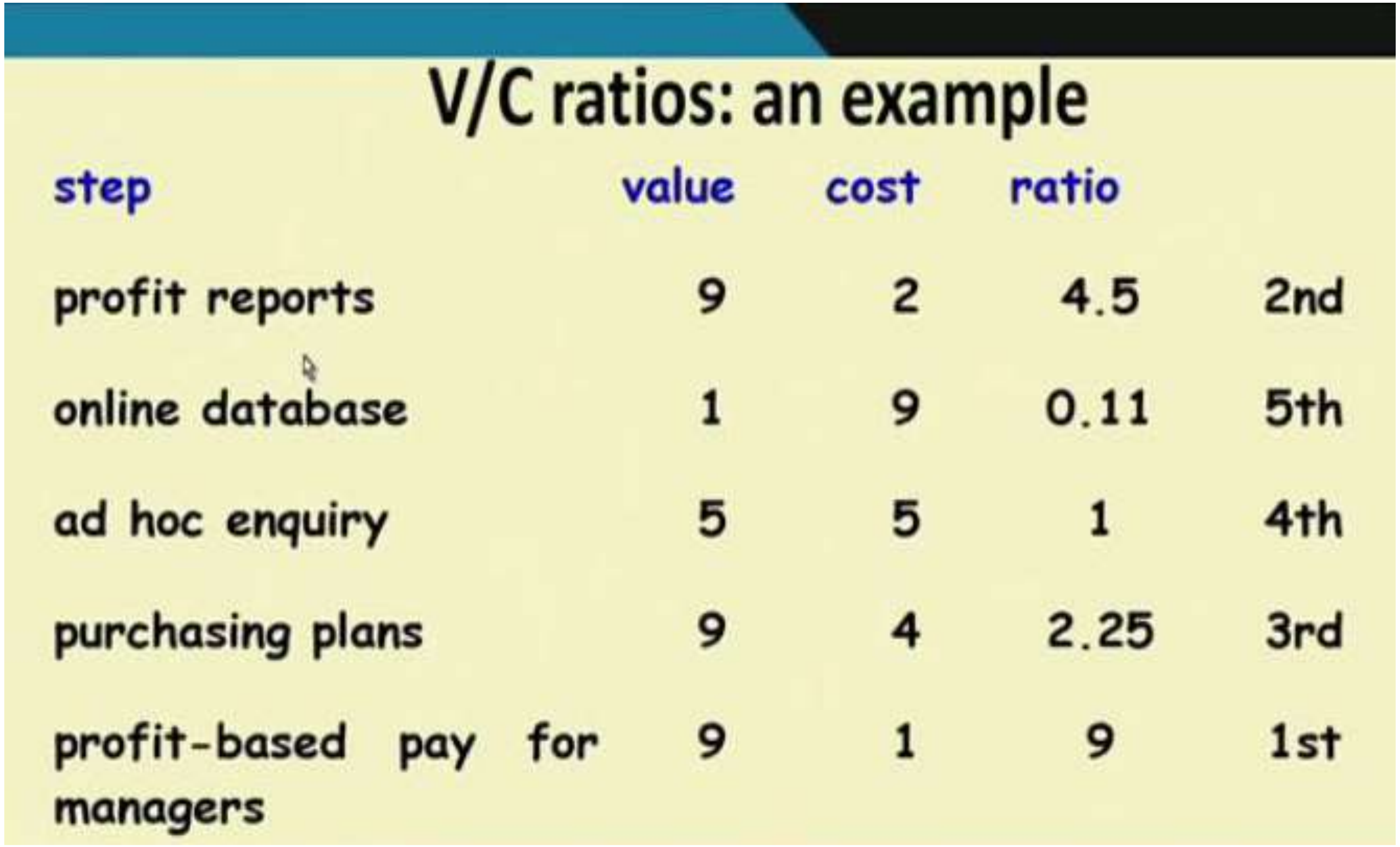


Which step first?

- Some steps will be pre-requisite because of physical depe
- Others may be in any order
- Value to cost ratios may be used
  - V/C where
  - V is a score 1-10 representing value to customer
  - C is a score 0-10 representing cost to developers

- Because there are several increments out of which the designer need to choose one of the increment for delivery in the next iteration, which increment should be selected first? This is an important problem.
- Some parts will be pre requisite of the other parts without that the system will not work.

- For example, until the customer registration is done customer billing cannot be done first we must have the customer registration software developed and then only the customer billing software can be implemented.



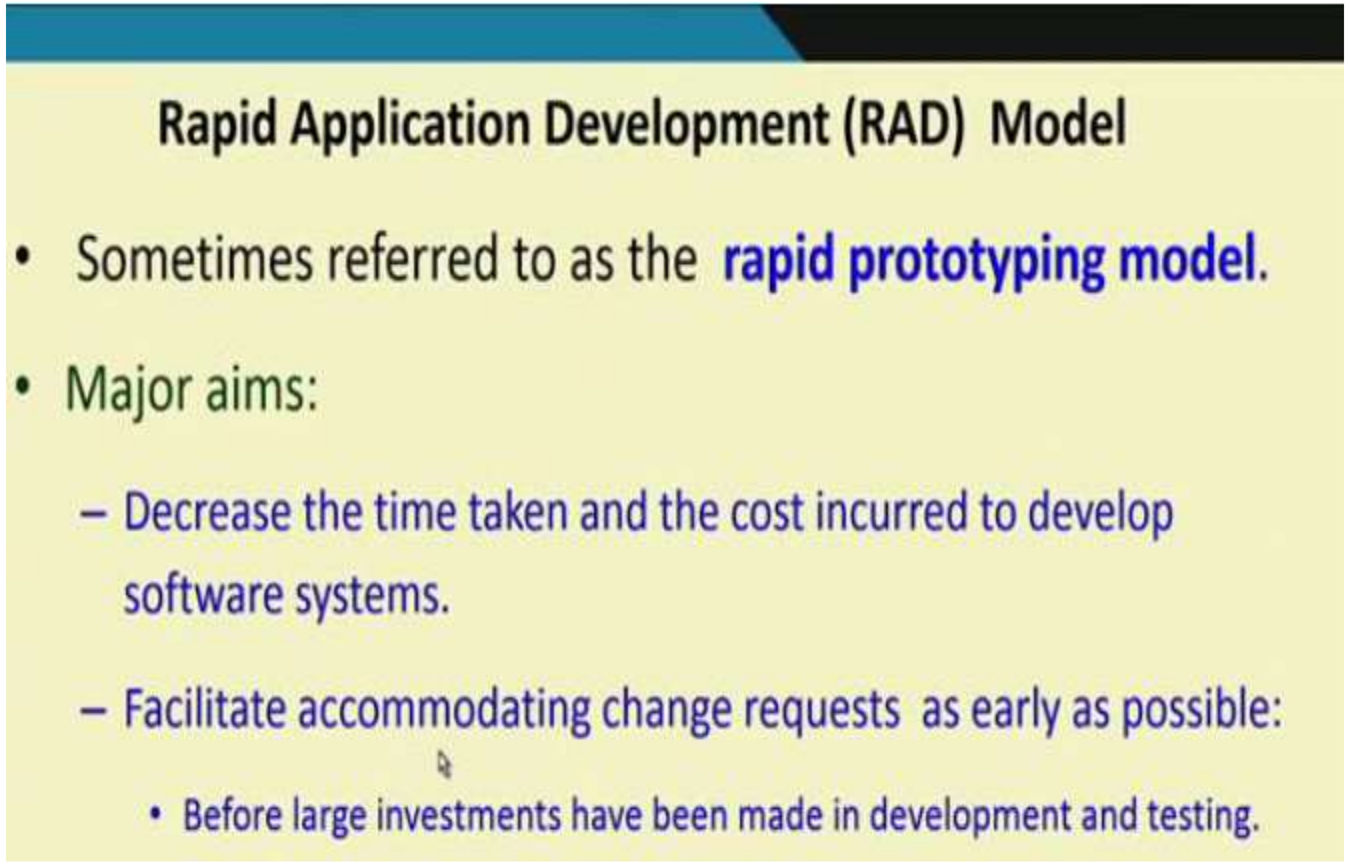
V/C ratios: an example

step	value	cost	ratio	
profit reports	9	2	4.5	2nd
online database	1	9	0.11	5th
ad hoc enquiry	5	5	1	4th
purchasing plans	9	4	2.25	3rd
profit-based pay for managers	9	1	9	1st

- The incremental model as become extremely popular, it is almost part of every model that is being used those are derivatives of the incremental model

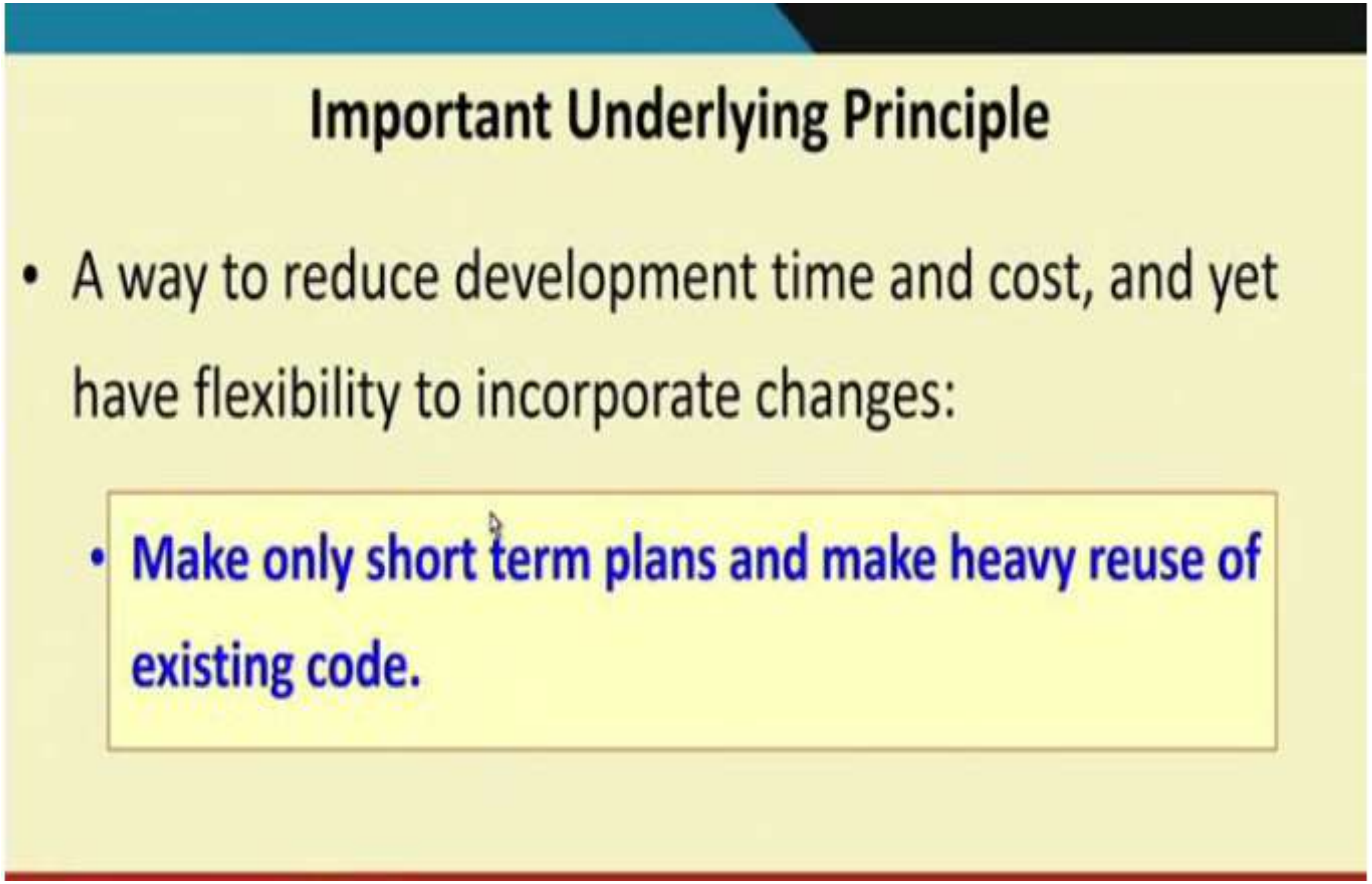
- one important derivative is the RAD model, RAD stands for Rapid Application Development.

Rapid application Development



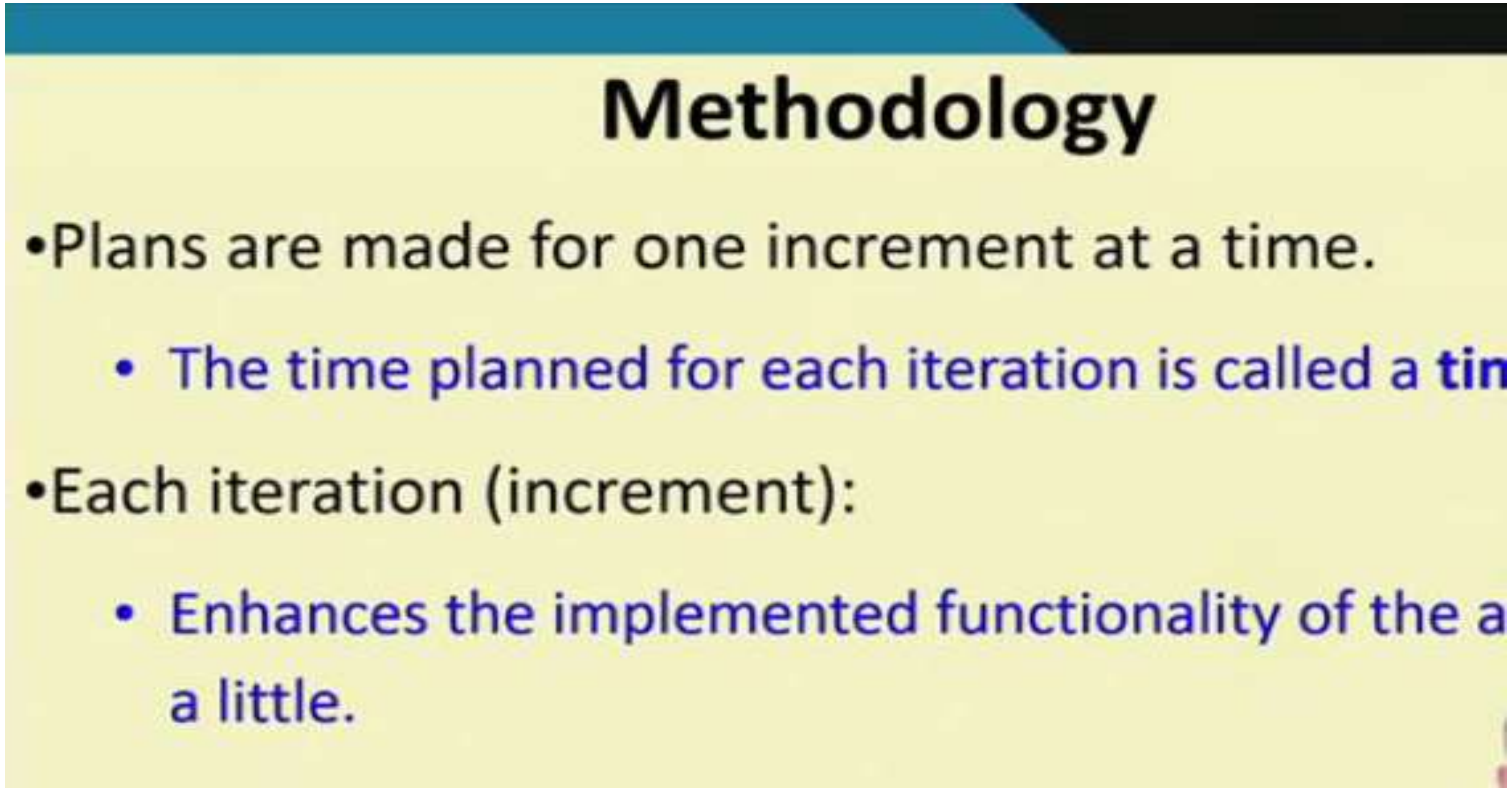
Rapid Application Development (RAD) Model

- Sometimes referred to as the **rapid prototyping model.**
- Major aims:
  - Decrease the time taken and the cost incurred to develop software systems.
  - Facilitate accommodating change requests as early as possible:
    - Before large investments have been made in development and testing.



Important Underlying Principle

- A way to reduce development time and cost, and yet have flexibility to incorporate changes:
  - **Make only short term plans and make heavy reuse of existing code.**



Methodology

- Plans are made for one increment at a time.
  - The time planned for each iteration is called a **tin**
- Each iteration (increment):
  - Enhances the implemented functionality of the a
    a little.

- During each iteration a prototype of the software is first developed is given to the customer for evolution and then this prototype is refined based on the customer feedback,

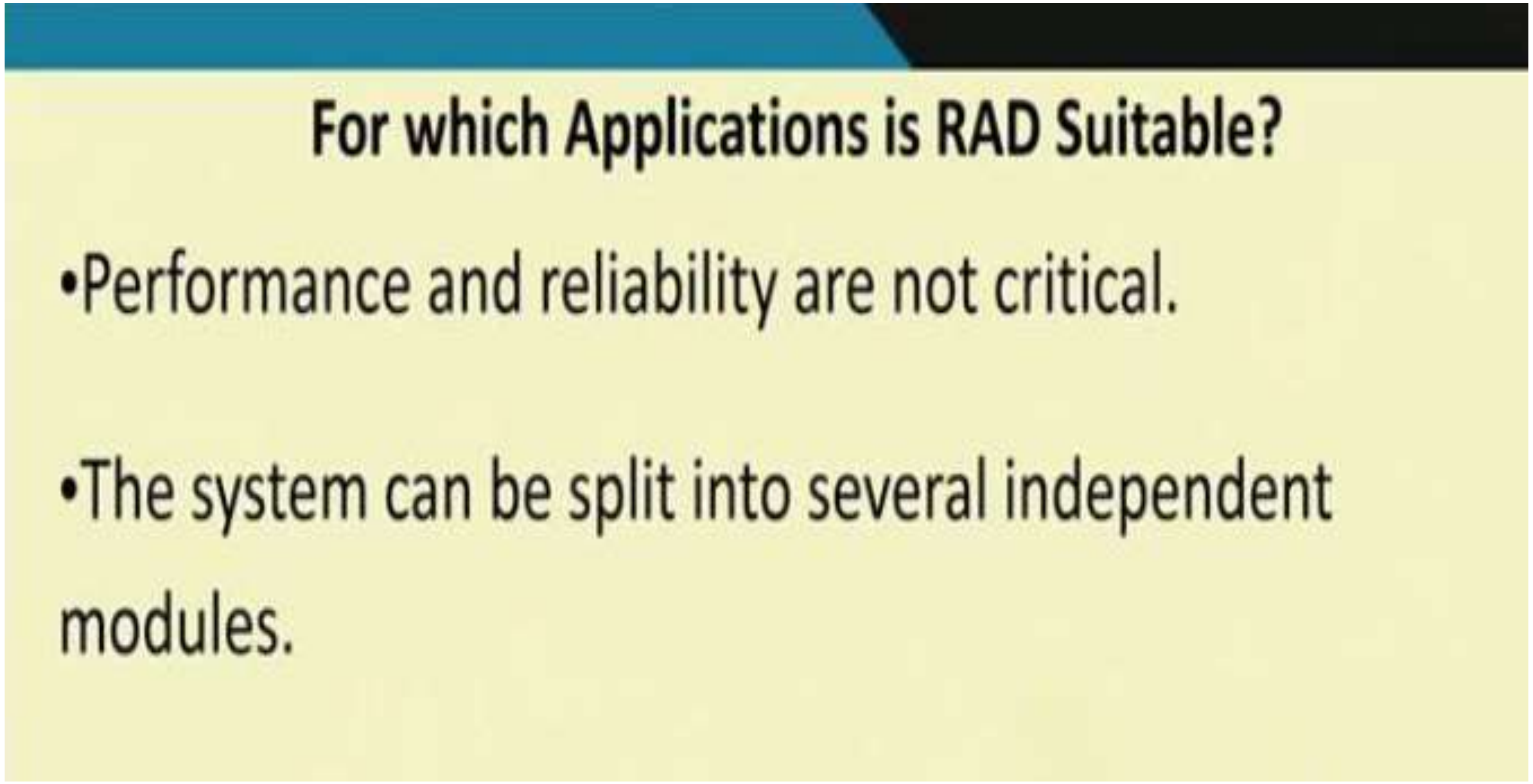
- one point to note here is that in the RAD model the prototype itself is refined to be the actual software, where as in the prototyping model which is a derivative of the waterfall model, The prototype is used to get customer feedback the start of the project and then the prototype is thrown away and the software is developed fresh. But in the RAD model the prototype is refined based on the customer feedback.

- The RAD model achieves faster development by

1. use of specialized tools, the tools should be supporting visual style of development like drag and drop.

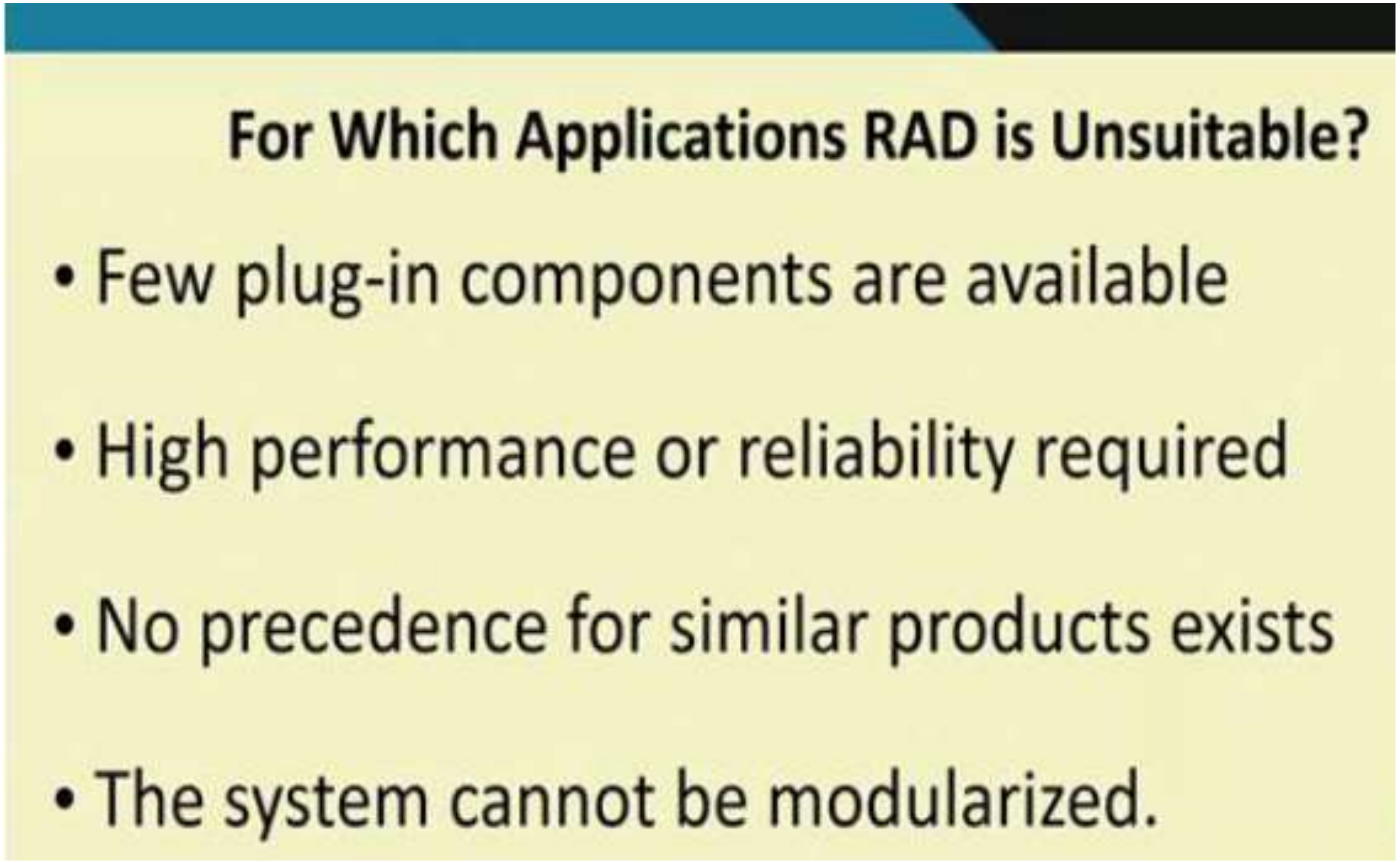
2. Use of reusable components and

3. use of standard APIs these are something that are suppose to help the software implemented faster.



For which Applications is RAD Suitable?

- Performance and reliability are not critical.
- The system can be split into several independent modules.



For Which Applications RAD is Unsuitable?

- Few plug-in components are available
- High performance or reliability required
- No precedence for similar products exists
- The system cannot be modularized.

- The RAD model is unsuitable when we have few login components are available, high performance or reliability is required.

- Reason: in RAD model the prototype itself is refined into the actual software whereas in waterfall In RAD model model a fresh software is designed coded and tested.
- In case of a new software being built , if no plugins are available even then RAD is unsuitable .

Prototyping versus RAD

- The RAD model each time constructs a prototype and gives to the customer for evaluation and based on the customer evaluation the prototype is changed and as the customer gets satisfied the refined prototype is becomes the part of the software.
- In the prototyping model which is a derivative of the waterfall model, the developed prototype is primarily used to gain customer feedback and also to get insight into the solution that is the technical issues and so on.
- And therefore, using the prototyping model you can chose between different design alternatives, you can elicit customer feedback and the developed prototype is usually thrown away, but not in the RAD model.
- In RAD model the prototype itself after enhancement refinement becomes the final software.