onto functional
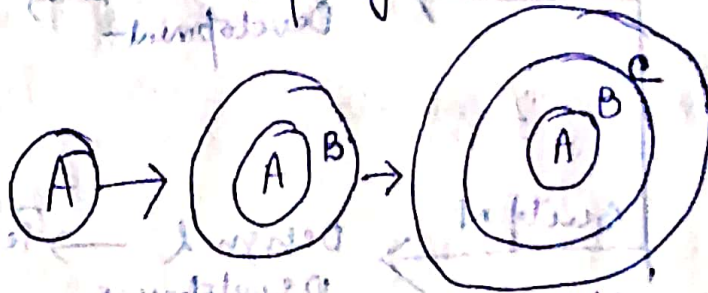
Evolutionary model

*) It is the combination of Iterative & Incremental model of SDLC.

*) Incremental model first implements a few basic feature & deliver to customer. Then build the next part and deliver it again. & repeat this step until the desired system is fully realized. No-longterm plans are made.

*) Iterative model main advantages is its feedback process in every phase.

*) Also known as "Design a little, build a little, deploy a little model"

*) Here after completion of each module, feedback is taken from customer & again design the module, if some changes are there in feedback & go to the next module. Same process will repeat for all module.

:) Here the Requirement that are fixed
at the beguning time, may changes time
to time for better preparation of model.

Advantages

*) customer Requirements are clearly
   specified
*) Risk analysis is better.
*) It Supports changing environment
*) Initial operating time is less.
*) Better suited for large projects mission-
   critical projects.

Disadvantages.

*) Not Suitable for Smaller projects.
*) Cost
*) Highly Skilled Resources are required.

```
┌─────────────────────────────┐
│   Requirement Specification.  │
└─────────────────────────────┘
              ↓
┌─────────────────────────────┐
│ identify the Core parts      │
│ to be developed incrementally │
└─────────────────────────────┘
              ↓
┌─────────────────────────────┐
│ Develop the core part        │
│ using iterative waterfall model│
└─────────────────────────────┘
              ↓
┌─────────────────────────────┐        ┌──────────────┐
│ Collect customer feedback    │ ←───── │ Delevery of  │
│ and modify requirements      │        │ the next     │
└─────────────────────────────┘        │ version to   │
              ↓                          │ the customer.│
┌─────────────────────────────┐        └──────────────┘
│ Develop the next identified  │ ──────────┘
│ features using iterative waterfall│
│           model              │
└─────────────────────────────┘
              ↓
┌──────────────┐
│ maintenance  │
└──────────────┘
```