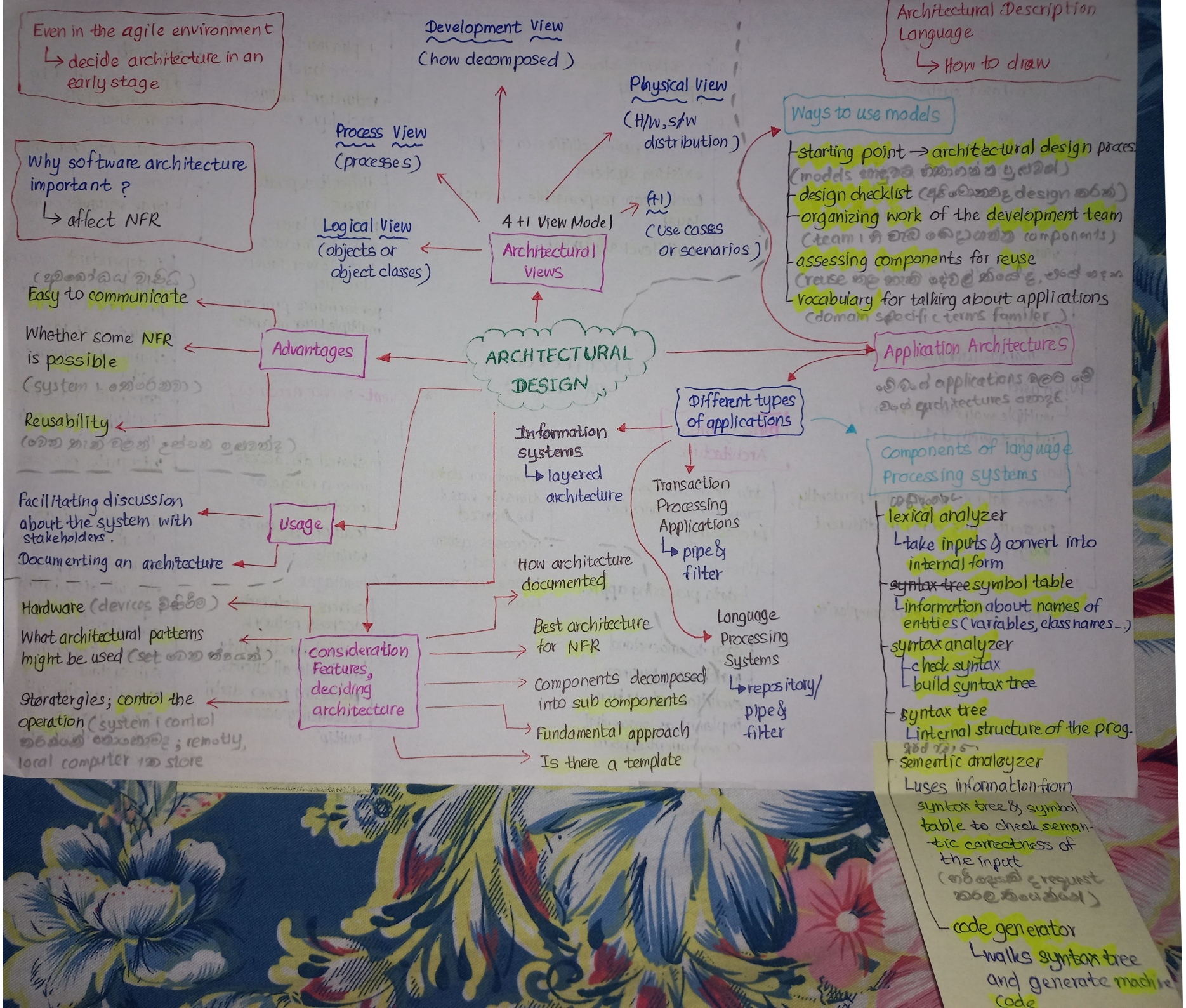
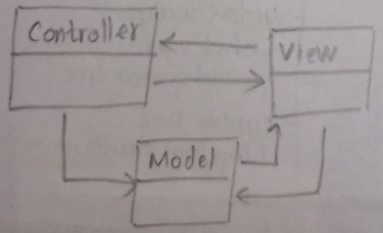

Even in the agile environment
↳ decide architecture in an early stage
Development View
(how decomposed)
Physical View
(H/W, S/W distribution)
Process View
(processes)
Why software architecture important ?
↳ affect NFR
Logical View
(objects or object classes)
4+1 View Model
Architectural Views
(+1)
(Use cases or scenarios)
Architectural Description Language
↳ How to draw
Ways to use models
starting point → architectural design proces
design checklist
organizing work of the development team
assessing components for reuse
vocabulary for talking about applications
Easy to communicate
Whether some NFR is possible
Reusability
Advantages
ARCHTECTURAL DESIGN
Application Architectures
Different types of applications
Information systems
↳ layered architecture
Transaction Processing Applications
↳ pipe & filter
Language Processing Systems
↳ repository/ pipe & filter
Components of language Processing systems
lexical analyzer
↳ take inputs & convert into internal form
syntax tree symbol table
↳ information about names of entities (variables, class names..)
syntax analyzer
check syntax
build syntax tree
syntax tree
↳ internal structure of the prog.
Semantic analayzer
↳ uses information from syntax tree & symbol table to check seman-tic correctness of the input
code generator
↳ walks syntax tree and generate machine code
Facilitating discussion about the system with stakeholders.
Documenting an architecture
Usage
Hardware (devices)
What architectural patterns might be used
Storategies; control the operation (system control ; remoty, local computer store
consideration Features, deciding architecture
How architecture documented
Best architecture for NFR
Components decomposed into sub components
Fundamental approach
Is there a template

ARCHITECTURAL PATTERNS

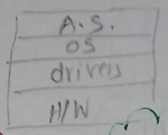
MVC Architecture

- most web based systems
- 3 components
  - model
    - manage system operations data and associated operations on that data
  - view
    - how data is presented to users
  - controller
    - manage user interactions
- When used
  - multiple ways to view and interact with data
- Advantages
  - allows data to change independently
  - present same data in different ways.
- Disadvantages
  - additional code & code complexity



Layered Architecture

- allow seperate elements to change independantly
- When used
  - building new facilities on top of existing systems
  - each team responsible for each layer
  - multi level security



- Advantages
  - replacement of entire layer
  - redundant facilities each layer
- Disadvantages
  - difficult seperate layers
  - high level layers have to interact with lower layers directly
  - performance problem multiple layer interprit.

Pipe & filter Architecture

- data flows from one component to another processing
- When used
  - data processing applic.
- Advantages
  - easy to understand
  - match many business
  - evolution - stwise wflow.
  - implement as sequential or concurrent system
- Disadvantages
  - format for data transfer has to be agreed
  - increases system overhead; impossible to reuse.

Client-server Acrchite.

- When used
  - shared db, access from a range of locations
  - load on system is variable
- Advantages
  - servers - distributed accross network
  - general functionality available all clients.
  - processing power distri
  - concurrency access
  - multiple remote users
  - load balancing
- Disadvantages
  - dos attacks
  - performance - unpredict-able (network, system)
  - management problems
  - network collision
  - high security risk
  - unavailability of service due to network and

Repository Architecture

- shared db
- data is generated by one sub module used by another
- eg: IS, CAD, CASE, OS
- When used
  - large volumes of info
- Advantages
  - independant compon.
  - change to one comp. properated to all.
  - easy to manage.
  - share large amount of data
  - centralized activities (backup, access contro, recovery)
  - intergrate new tool
- Disadvantages
  - single point failure → all system
  - communicating prob.
  - difficult distributing repository several computers.
  - same policy on all sub systems
  - evolution difficult
  - all sub systems agree on repository model