

Software Design & Architecture

Engr. Abdul-Rahman Mahmood

DPM, MCP, QMR(ISO9001:2000)

 armahmood786@yahoo.com


 alphapeeler.sf.net/pubkeys/pkey.htm

 pk.linkedin.com/in/armahmood

 www.twitter.com/alphapeeler

 www.facebook.com/alphapeeler

 abdulmahmood-sss  alphasecure

 armahmood786@hotmail.com

 http://alphapeeler.sf.net/me

 alphasecure@gmail.com

 http://alphapeeler.sourceforge.net

 http://alphapeeler.tumblr.com

 armahmood786@jabber.org

 alphapeeler@aim.com

 mahmood_cubix  48660186

 alphapeeler@icloud.com

 http://alphapeeler.sf.net/acms/

Architectural views of the system

Logical and process views

Implementation, deployment and use case views

4+1 View Model

Designed for : Describing the architecture of software-intensive systems, based on the use of multiple, concurrent views

What is View Model ?

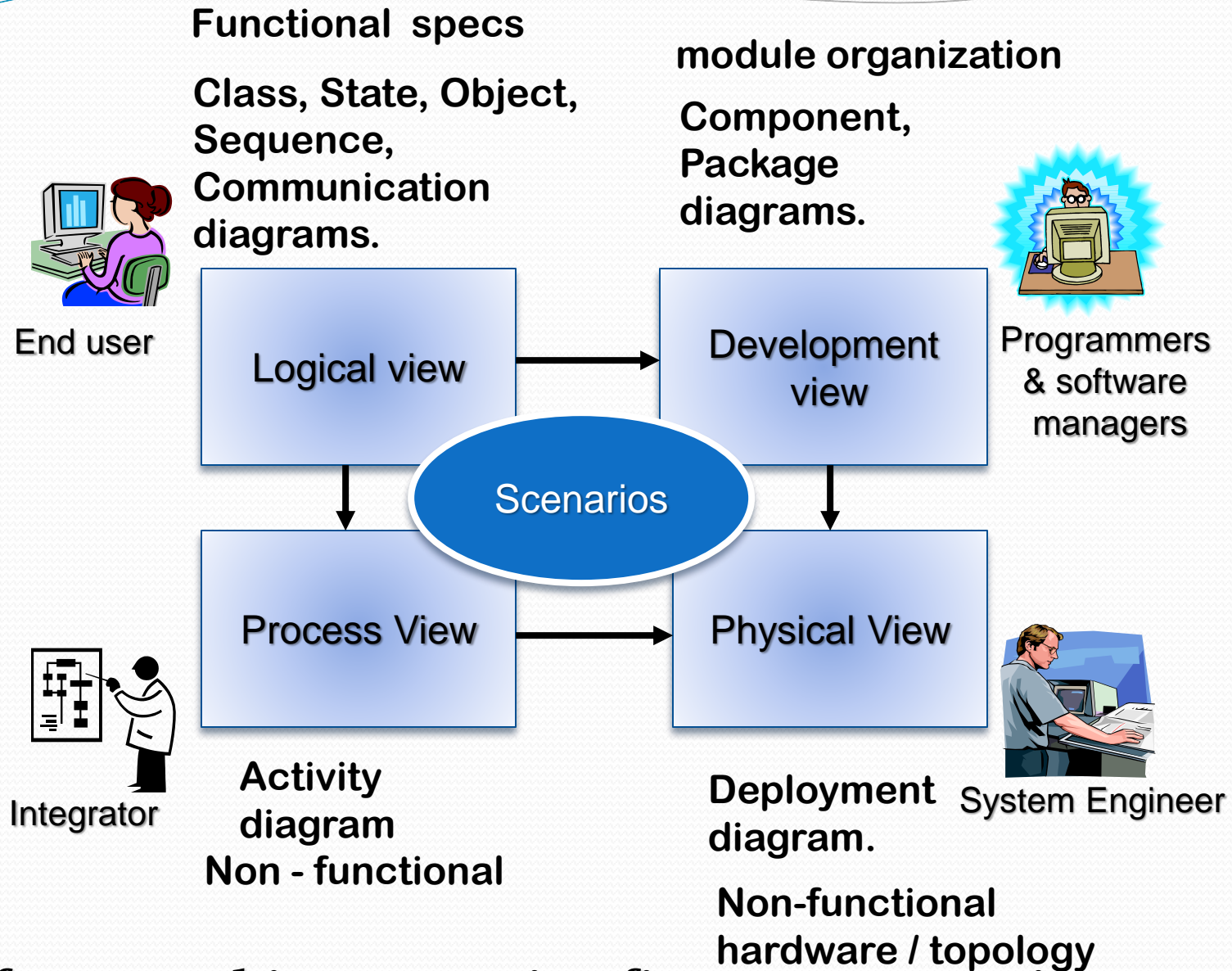
- A view model in systems engineering or software engineering is a framework.
- It defines a coherent set of views to be used in the construction of a system architecture or software architecture.
- A *view* is a representation of a whole system from the perspective of a related set of concerns.
- Viewpoint modeling has become an effective approach for dealing with the inherent complexity of large distributed systems.

4+1 View Model

- The views are used to describe the system from the viewpoint of different stakeholders, such as end-users, developers and project managers.
- The four views of the model are:
 - > Logical View
 - > Development View
 - > Process View
 - > Physical View
- In addition selected use cases or scenarios are utilized to illustrate the architecture serving as the '**plus one**' view.

*“4+1” view model presented to address large
and challenging architectures*

The 4+1 View model



software architecture using five concurrent views.

Logical View

(Object-oriented Decomposition)

Viewer : End-user

Considers : Functional requirements- What the system should provide in terms of services to its users.

- This view shows the components (objects) of the system as well as their interactions / relationships.
- UML diagrams – Class, State, Object, Sequence, Communication diagrams.

Process View

(The process decomposition)

Viewer : Integrators

**Considers : Non - functional requirements
(concurrency, performance, scalability)**

- The process view shows the processes / workflow rules of a system and how those processes communicate with each other.
- UML diagrams – Activity diagram

Development View

(Subsystem decomposition)

Viewer : Programmers and Software Managers

Considers : software module organization
(Hierarchy of layers, software management, reuse, constraints of tools)

- It gives a building block view of the system.

Eg: Packages Used, Execution Environments, Class Libraries and Sub systems utilized.

- UML diagrams – Component, Package diagrams.

Physical Views

(Mapping the software to the Hardware)

Viewer : System Engineers

Considers : Non-functional req. regarding to underlying hardware (Topology, Communication)

- This view shows the systems execution environment
- UML diagram – Deployment diagram.

Scenario / Use case View

(Putting it all together)

Viewer: All users of other views and Evaluators.

Considers : System consistency, validity

- Validation and illustration to show the design is complete
- It is redundant with other views.
- UML diagram – Use case diagrams.

Relationship among the views

- Concurrency is not addressed in the logical view, to achieve the process view, we need to map classes and their objects onto tasks and processes addressing concurrency and synchronization.
- The processes and process groups are mapped onto the processing nodes of a physical computer network to obtain the physical view. For each dependency between components, there must be a corresponding link between nodes.
- The logical view is also the basis for the development view. Normally each class corresponds to a module. Large classes may be decomposed into packages. Collections of classes are grouped into subsystems.

Modeling a system

- Decide which views are needed to best express the architecture of the system and to expose the technical risks to the project.
- For each of these views, decide the artifacts(eg. UML Diagrams) needed to be created to capture the essential details of that view.
- As part of process planning, decide which of the UML diagrams require frequent reviews and which have to be documented.
- Allow room for diagrams that are thrown away. Such transitory diagrams are useful for exploring the implications of decisions taken and for experimenting with changes.

Views – UML Diagrams

- **A simple Monolithic Application that runs on a single machine requires...**

- Use case view

Use case Diagrams

- Logical view

Class Diagrams

Interaction Diagrams

- Process view

None

- Development view

None

- Deployment view

None

Views – UML Diagrams

- **A complex, distributed system requires...**

- | | |
|--------------------|----------------------------------------------------------------|
| • Use case view | Use case Diagrams
Activity Diagrams |
| • Logical view | Class Diagrams
Interaction Diagrams
State chart Diagrams |
| • Process view | Class Diagrams
Interaction Diagrams |
| • Development view | Component Diagrams |
| • Deployment view | Deployment Diagrams |

Why is it called the 4 + 1 instead of just 5?

- The use case view has a special significance.
- When all other views are finished, it's effectively redundant.
- However, all other views would not be possible without it.
- It details the high levels requirements of the system.
- The other views detail how those requirements are realized.

