4+1 VIEW MODEL

UML Views

- Structural classification
- Dynamic behaviour
- Physical layout

UML views and diagrams

Major View	Diagram	Concepts
structural	class diagram	association, class, dependency, generalization, interface, realization
	internal structure component diagram	connector, interface, part, port, provided interface, role, required interface
		connector, collaboration, collaboration use, role
		component, dependency, port, provided interface, realization, required interface, subsystem
	use case diagram	actor, association, extend, include, use case, generalization

UML views and diagrams cont.

Major View	Diagram	Concepts
dynamic	state machine diagram	completion transition, do activity, effect, event, region, state, transition, trigger
	activity diagram	action, activity, control flow, control node, data flow, exception, expansion region, fork, join, object node, pin
	sequence diagram communication diagram	occurrence specification, execution specification, interaction, lifeline, message, signal
		collaboration, guard condition, message, role, sequence number

UML views and diagrams cont.

Major View	Diagram	Concepts
physical	deployment diagram	artifact, dependency, manifestation, node

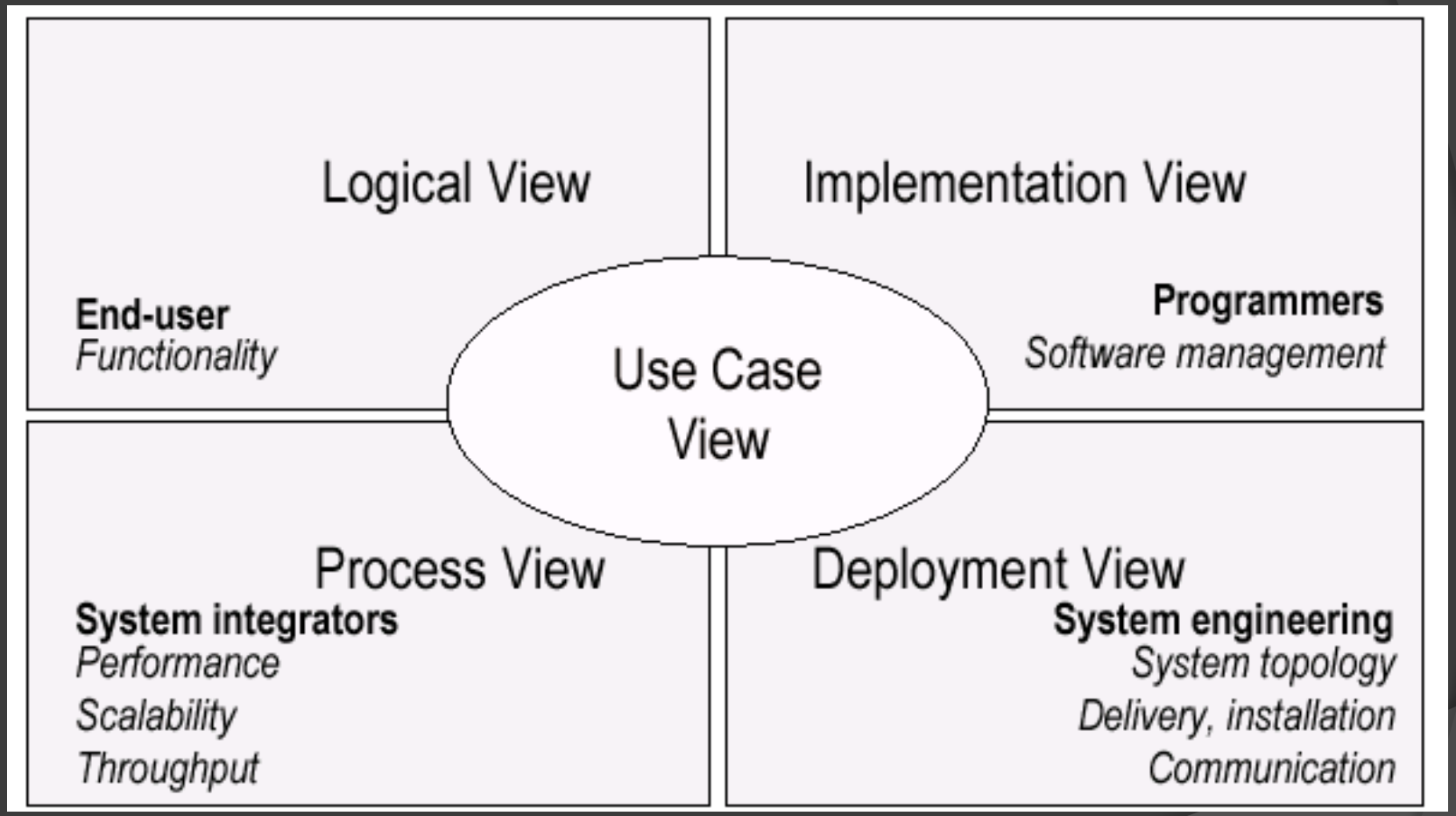
What is View Model ?

- 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.

Intent of 4+1 view model

- To come up with a mechanism to separate the different aspects of a software system into different views of the system.
- But why???? -> Different stakeholders always have different interest in a software system.

4+1 View Model of Architecture



Use Case View

- The use-case view describes the functionality the system should deliver, as perceived by external actors.
- An actor interacts with the system; the actor can be a user or another system.
- The use-case view is used by customers, designers, developers, and testers; it is described in use-case diagrams, sometimes with support from activity diagrams.
- The desired usage of the system is described as a number of use cases in the use-case view, where a use case is a generic description of a function requested.

Central View

- The use-case view is central, because its contents drive the development of the other views.
- The final goal of the system is to provide the functionality described in this view—along with some nonfunctional properties.
- Hence, this view affects all the others.

Logical View

- The logical view describes how the system's functionality is provided.
- Mainly for designers and developers.
- In contrast to the use-case view, the logical view looks inside the system.
- It describes both the static structure (classes, objects, and relationships) and the dynamic collaborations that occur when the objects send messages to each other to provide a given function.
- Properties such as persistence and concurrency are also defined, as well as the interfaces and the internal structure of classes.

Implementation view

- The implementation view describes the main modules and their dependencies.
- It is mainly for developers
- Consists of the main software artifacts. The artifacts include different types of code modules shown with their structure and dependencies.

Process View

- The process view deals with the division of the system into processes and processors. This aspect allows for efficient resource usage, parallel execution, and the handling of asynchronous events from the environment.
- This view must also deal with the communication and synchronization of these threads.

Deployment view

- Finally, the deployment view shows the physical deployment of the system, such as the computers and devices (nodes) and how they connect to each other.
- The various execution environments within the processors can be specified as well.

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.**

Is it important?

- **It makes modeling easier.**
- **Better organization with better separation of concern.**
- **The 4 + 1 approach provides a way for architects to be able to prioritize modeling concerns.**
- **The 4 + 1 approach makes it possible for stakeholders to get the parts of the model that are relevant to them.**

Use Case View

Use Case Analysis is a technique to capture business process from user's perspective.

Static aspects in use case diagrams; Dynamic aspects in interaction (state chart and activity) diagrams.

Design View

Encompasses classes, interfaces, and collaborations that define the vocabulary of a system., Supports functional requirements of the system.

Static aspects in class and object diagrams; Dynamic aspects in interaction diagrams.

Process View

Encompasses the threads and processes defining concurrency and synchronization, Addresses performance, scalability, and throughput.

Static and dynamic aspects captured as in design view; emphasis on active classes.

Implementation View

Encompasses components and files used to assemble and release a physical system.

Addresses configuration management.

Static aspects in component diagrams; Dynamic aspects in interaction diagrams.

Deployment View

Encompasses the nodes that form the system hardware topology, Addresses distribution, delivery, and installation.

- Static aspects in deployment diagrams; Dynamic aspects in interaction diagrams..