USN

Internal Assessment Test 2 – Feb. 2021

Sub:	Software Engineering					Sub Code:	18CS35	Branch:	CSE	
Date:	24 -02 -21	Duration:	90 min's	Max Marks:	50	Sem / Sec:	III (Diploma Sections)		OBE	

	Answer any FIVE FULL Questions	MARKS	CO	RBT
1.	With the help of a sample class model explain the following I. Association and Association end name II. Qualified association III. Multiplicity IV. Generalization	[10]	CO1	L1
2. (a)	Write a class model of windowing system?	[04]	CO1	L3
(b)	With the help of a class model for credit card accounts explain the Object Constraint Language (OCL) constructs for traversing the class model.	[06]	CO3	L3
3.	Explain the state diagram of a micro oven with the working process.	[10]	CO3	L3
4.	Draw a context model for the MHC-patient management system. How are the interactions modeled?	[10]	CO3	L3

CI CCI

USN

Internal Assessment Test 2 – Feb. 2021

Sub:	Software Engineering					Sub Code:	18CS35	Branch:	
Date:	24 -02 -21	Duration:	90 min's	Max Marks:	50	Sem / Sec:	III (Diploma Sections)		

	Answer any FIVE FULL Questions	MARKS	CO	
1.	With the help of a sample class model explain the following I. Association and Association end name II. Qualified association III. Multiplicity IV. Generalization	[10]	CO1	
2. (a)	Write a class model of windowing system?	[04]	CO1	
(b)	With the help of a class model for credit card accounts explain the Object Constraint Language (OCL) constructs for traversing the class model.	[06]	CO3	
3.	Explain the state diagram of a micro oven with the working process.	[10]	CO3	
4.	Draw a context model for the MHC-patient management system. How are the interactions modeled?	[10]	CO3	

CI CCI

		MARKS	CO	RBT
5.	Explain phases of Rational unified process model.	[10]	CO3	L2

6.	Explain open source development in detail.	[10]	CO3	L1
7.	Explain Model driven engineering. Mention its pros and cons.(10M) .	[10]	CO3	L2

CI CCI

		MARKS	CO	RBT
5.	Explain phases of Rational unified process model.	[10]	CO3	L2
6.	Explain open source development in detail.	[10]	CO3	L1
7.	Explain Model driven engineering. Mention its pros and cons.(10M) .	[10]	CO3	L2

CI CCI

Scheme of evaluation

Qno	Questions	Marks distribution
1	Explain following terms Association, Links, Multiplicity, Sequence, and Bag.	4*2.5M
2a	Model of Windowing system	4M
2b	Credit card accounts explain the Object Constraint Language (OCL) constructs for traversing the class model.	6M
3	State diagram of a microoven with the working process. Diagram	7M 3M
4	The organization of MHC-PMS (Mental Health Care Patient Management System) goals and features of the system Diagram	5M 3M 2M
5	Rational unified process model	10M
6	Open source development	10M
7	Model driven engineering Mention its pros and cons	5M 5M

Solution:

1. **I. Association and Association end name**

- Associations are the means for establishing relationships among classes.An association is a description of a group of links with common structure and common semantics.E.g. a person WorksFor a company. If two classes in a model need to communicate with each other, there must be link between them, and that can be represented by an association (connector).
- Associations are inherently bi-directional. The association name is usually read in a particular direction but the binary association may be traversed in either direction. Association can be represented by a line between these classes **with an arrow indicating the navigation direction**. In case arrow is on the both sides, association has bidirectional

Figure 3.7 Many-to-many association. An association describes a set of potential links in the same way that a class describes a set of potential objects.

association.

Association connects related classes and is also denoted by a line.Show association names in italics.

- **Association end name** Associations have ends. They are called ‘Association Ends’. They may have names (which often appear in problem descriptions). Use of association end names is optional. But association end names are useful for traversing associations.

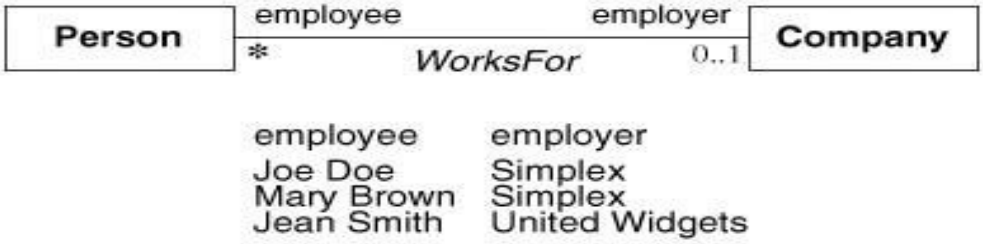


Figure 3.12 Association end names. Each end of an association can have a name.

II.Qualified association

- A qualified association is an association in which an attribute called Qualifier the objects for a 'many' association' end. A qualifier selects among the target objects, reducing the effective multiplicity from 'many' to 'one'.Both below models are acceptable but the qualified model adds information.

Figure 3.22 Qualified association. Qualification increases the precision of a model.

Adding a qualifier clarifies the class diagram and increases the conveyed information. In this

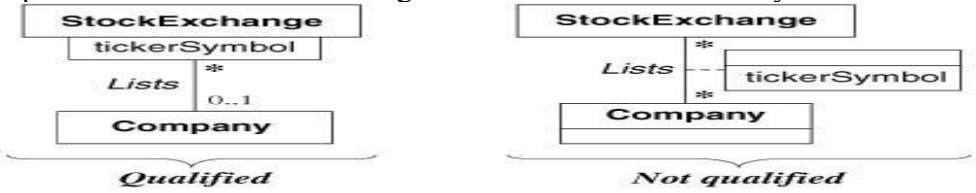
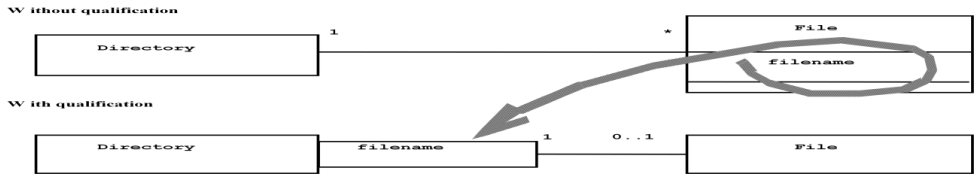


Figure 3.23 Qualified association. Qualification also facilitates traversal of class models.

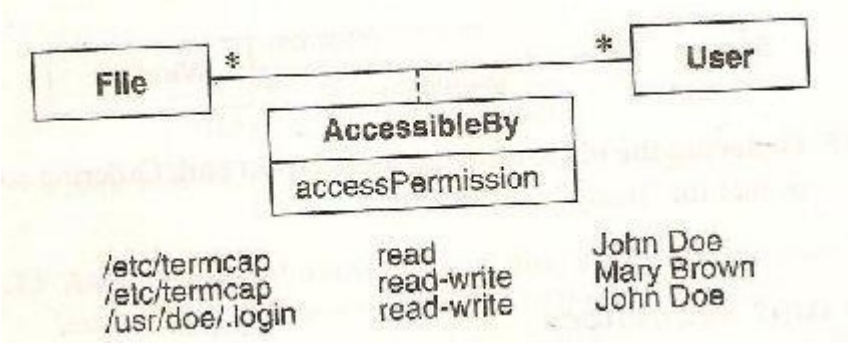
case, the model including the qualification denotes that the name of a file is unique

within a directory. Example of how a qualified association reduces multiplicity (UML class diagram).



III.Association classes

An **association class** is an association that is also a class.Like the links of an association, the instances of an association class derive identity from instances of the constituent classes. Like a class, an association class can have attributes and operations and participate in associations.

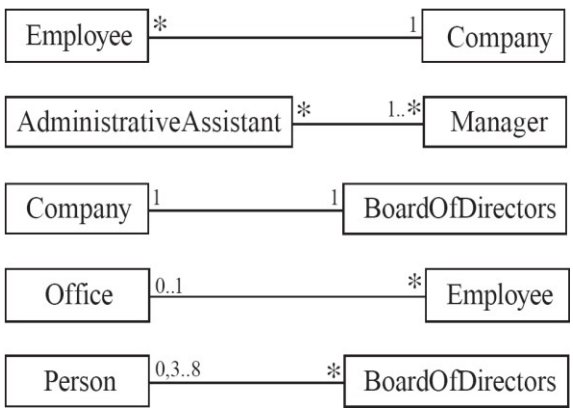


IV. Multiplicity

Multiplicity defines the number of objects associated with an instance of the association.

UML diagrams explicitly list multiplicity at the end of association lines.Intervals are used to express multiplicity:

Indicator	Meaning
0..1	Zero or one
1	One only
0..*	Zero or more
1..*	One or more
n	Only *n* (where *n* > 1)
0..n	Zero to *n* (where *n* > 1)
1..n	One to *n* (where *n* > 1)



Generalization

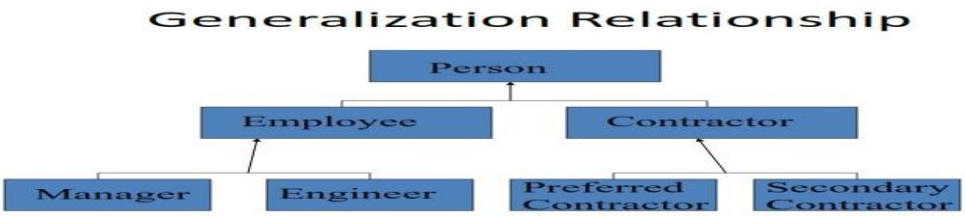
Deriving a class out of a parent class having some inherited property(from the parent class) and some new property of the derived class.

The term generalization is for the inheritance in the bottom to the up direction i.e. from derived class to the parent class. Generalization is the relationship between a class (superclass) and one or more variations of the class (subclasses).

A superclass holds common attributes, attributes and associations.The subclasses adds specific attributes, operations, and associations. They inherit the features of their superclass.

Generalization is called a “IS A” relationship

A *generalization* connects a subclass to its superclass. It denotes an inheritance of attributes and behavior from the superclass to the subclass and indicates a specialization in the subclass of the more general superclass.A solid line with a hollow arrowhead that point from the child to the parent class.



2)

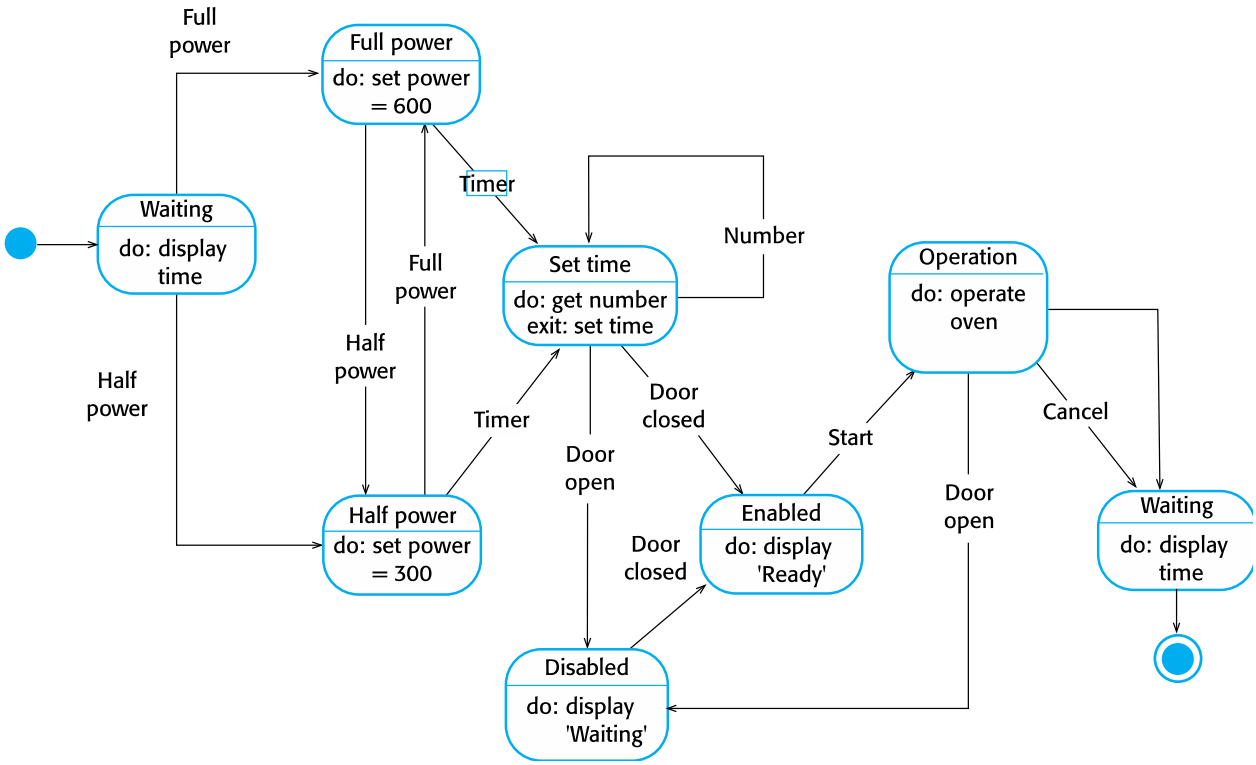
A windowing system is a system for sharing a computer's graphical display presentation resources among multiple applications at the same time. In a computer that has a graphical user interface (GUI), you may want to use a number of applications at the same time (this is called task).

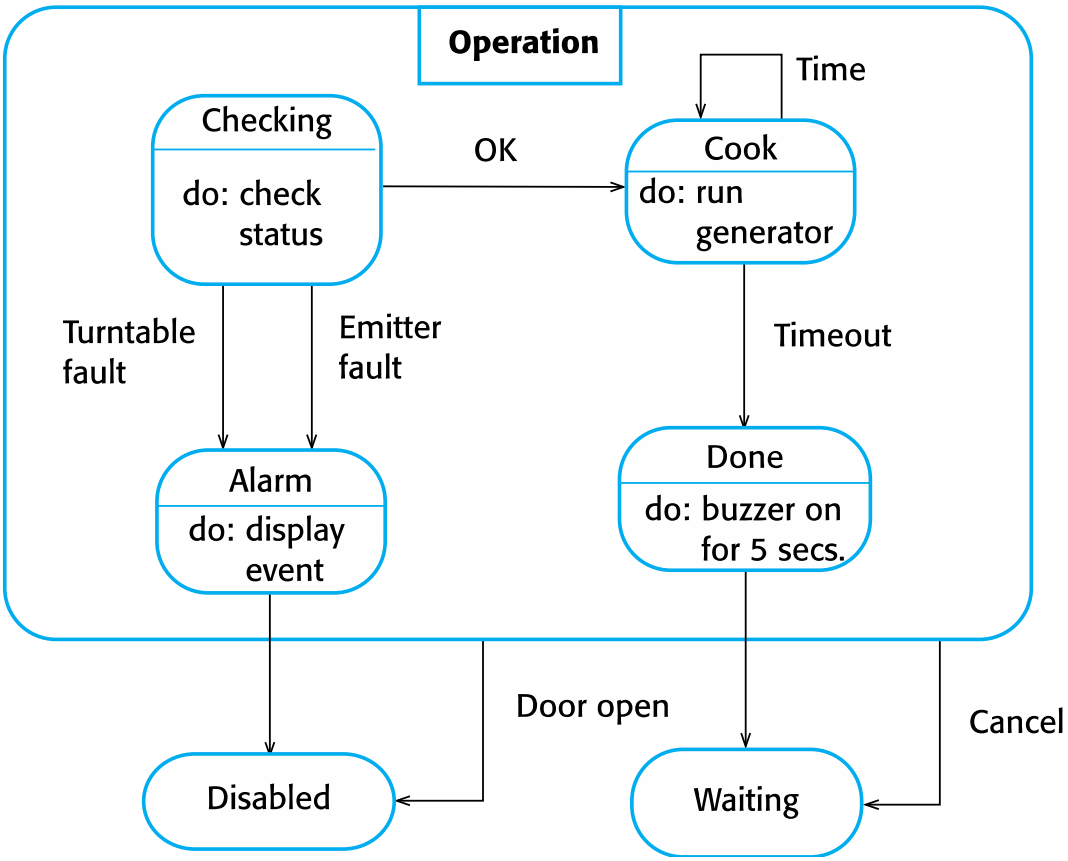
Figure 3.15 Ordering the objects for an association end. Ordering sometimes occurs for "many" multiplicity.

Using a separate window for each application, you can interact with each application and go from one application to another without having to reinitiate it. Having different information or activities in multiple windows may also make it easier for you to do your work. A windowing system uses a *window manager* to keep track of where each window is located on the display screen and its size and status. A windowing system doesn't just manage the windows but also other forms of graphical user interface entities.

3)

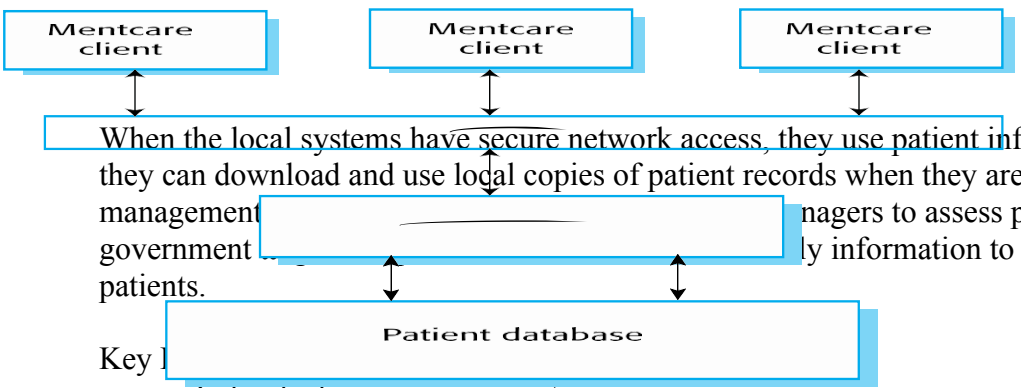
The control software for a very simple microwave oven to illustrate event-driven modeling. Real microwave ovens Are actually much more complex than this system but the simplified system is easier to understand. This simple microwave has a switch to select full or half power, a numeric keypad to input the cooking time, a start/stop button, and an alphanumeric display. I have assumed that the sequence of actions in using the microwave is: 1. Select the power level (either half power or full power). 2. Input the cooking time using a numeric keypad.3. Press Start and the food is cooked for the given time. For safety reasons, the oven should not operate when the door is open and, on completion of cooking, a buzzer is sounded. The oven has a very simple alphanumeric display that is used to display various alerts and warning messages. In UML state diagrams, rounded rectangles represent system states. They may include a brief description (following 'do') of the actions taken in that state. The labeled arrows represent stimuli that force a transition from one state to another. One way to do this is by using the notion of a superstate that encapsulates a number of separate states. This superstate looks like a single state on a high-level model but is then expanded to show more detail on a separate diagram. To illustrate this concept, consider the Operation state in Figure. The Operation state includes a number of sub-states. It shows that operation starts with a status check and that if any problems are discovered an alarm is indicated and operation is disabled. Cooking involves running the microwave generator for the specified time; on completion, a buzzer is sounded. If the door is opened during operation, the system moves to the disabled state.





4) MHC-PMS

• A patient information system to support mental health care is a medical information system that maintains information about patients suffering from mental health problems and the treatments that they have received. Most mental health patients do not require dedicated hospital treatment but need to attend specialist clinics regularly where they can meet a doctor who has detailed knowledge of their problems. To make it easier for patients to attend, these clinics are not just run in hospitals. They may also be held in local medical practices or community centres. Mentcare is an information system that is intended for use in clinics. It makes use of a centralized database of patient information but has also been designed to run on a PC, so that it may be accessed and used from sites that do not have secure network connectivity.



When the local systems have secure network access, they use patient information in the database but they can download and use local copies of patient records when they are disconnected. To generate management nagers to assess performance against local and government ly information to support the treatment of patients.

Key

• Individual care management

– Clinicians can create records for patients, edit the information in the system, view patient history, etc. The system supports data summaries so that doctors can quickly learn about the key problems and treatments that have been prescribed.

- Patient monitoring
  - The system monitors the records of patients that are involved in treatment and issues warnings if possible problems are detected.
- Administrative reporting

The system generates monthly management reports showing the number of patients treated at each clinic, the number of patients who have entered and left the care system, number of patients sectioned, the drugs prescribed and their costs, etc

5) RUP

The Rational Unified Process (RUP) (Krutchen, 2003) is an example of a modern process model that has been derived from work on the UML and the associated Unified Software Development Process (Rumbaugh, et al., 1999; Arlow and Neustadt, 2005).
The RUP recognizes that conventional process models present a single view of the process. In contrast, the RUP is normally described from three perspectives:
1. A dynamic perspective, which shows the phases of the model over time.
2. A static perspective, which shows the process activities that are enacted.
3. A practice perspective, which suggests good practices to be used during the process.
The phases in the RUP. These are:
1. *Inception* The goal of the inception phase is to establish a business case for the system. You should identify all external entities (people and systems) that will interact with the system and define these interactions. You then use this information to assess the contribution that the system makes to the business. If this contribution is minor, then the project may be cancelled after this phase.
2. *Elaboration* The goals of the elaboration phase are to develop an understanding of the problem domain, establish an architectural framework for the system, develop the project plan, and identify key project risks. On completion of this phase you should have a requirements model for the system, which may be a set of UML use-cases, an architectural description, and a development plan for the software.
3. *Construction* The construction phase involves system design, programming, and testing. Parts of the system are developed in parallel and integrated during this phase. On completion of this phase, you should have a working software system and associated documentation that is ready for delivery to users.
4. *Transition* The final phase of the RUP is concerned with moving the system from the development community to the user community and making it work in a real environment. This is something that is ignored in most software process models but is, in fact, an expensive and sometimes problematic activity. On completion of this phase, you should have a documented software system that is working correctly in its operational environment.

6. Explain open source development in detail.

- Open source development is an approach to software development in which the source code of a software system is published and volunteers are invited to participate in the development process
- Its roots are in the Free Software Foundation (www.fsf.org), which advocates that source code should not be proprietary but rather should always be available for users to examine and modify as they wish.
- Open source software extended this idea by using the Internet to recruit a much larger population of volunteer developers. Many of them are also users of the code.
- The best-known open source product is, of course, the Linux operating system which is widely used as a server system and, increasingly, as a desktop environment.

- Other important open source products are Java, the Apache web server and the mySQL database management system.
- Issues:
- Should the product that is being developed make use of open source components?
- Should an open source approach be used for the software's development?

Open source business

- More and more product companies are using an open source approach to development.
- Their business model is not reliant on selling a software product but on selling support for that product.
- They believe that involving the open source community will allow software to be developed more cheaply, more quickly and will create a community of users for the software.

Open source licensing

- A fundamental principle of open-source development is that source code should be freely available, this does not mean that anyone can do as they wish with that code.
    - Legally, the developer of the code (either a company or an individual) still owns the code. They can place restrictions on how it is used by including legally binding conditions in an open source software license.
    - Some open source developers believe that if an open source component is used to develop a new system, then that system should also be open source.
    - Others are willing to allow their code to be used without this restriction. The developed systems may be proprietary and sold as closed source systems.

License models

- The GNU General Public License (GPL). This is a so-called 'reciprocal' license that means that if you use open source software that is licensed under the GPL license, then you must make that software open source.
- The GNU Lesser General Public License (LGPL) is a variant of the GPL license where you can write components that link to open source code without having to publish the source of these components.
- The Berkley Standard Distribution (BSD) License. This is a non-reciprocal license, which means you are not obliged to re-publish any changes or modifications made to open source code. You can include the code in proprietary systems that are sold.

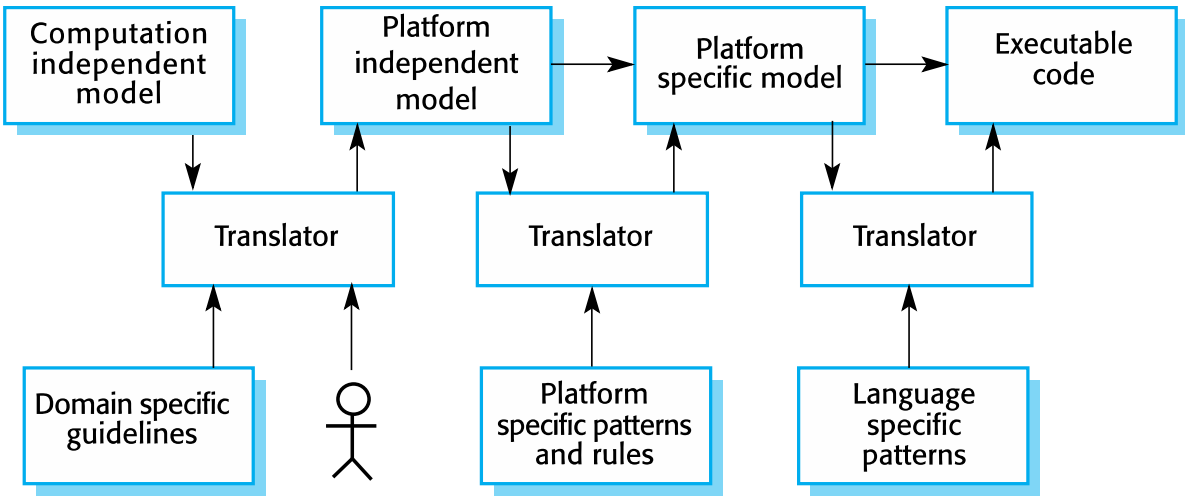
7. Model-driven engineering (MDE) is an approach to software development where models rather than programs are the principal outputs of the development process.

The programs that execute on a hardware/software platform are then generated automatically from the models.

Proponents of MDE argue that this raises the level of abstraction in software engineering so that engineers no longer have to be concerned with programming language details or the specifics of execution platforms

Usage of model-driven engineering

- Model-driven engineering is still at an early stage of development, and it is unclear whether or not it will have a significant effect on software engineering practice.
- Pros
    - Allows systems to be considered at higher levels of abstraction
    - Generating code automatically means that it is cheaper to adapt systems to new platforms.
- Cons
    - Models for abstraction and not necessarily right for implementation.
    - Savings from generating code may be outweighed by the costs of developing translators for new platforms.
- Model-driven architecture (MDA) was the precursor of more general model-driven engineering
- MDA is a model-focused approach to software design and implementation that uses a subset of UML models to describe a system.
- Models at different levels of abstraction are created. From a high-level, platform independent model, it is possible, in principle, to generate a working program without manual intervention.


Computation independent model
Platform independent model
Platform specific model
Executable code
Translator
Translator
Translator
Domain specific guidelines
Platform specific patterns and rules
Language specific patterns