

Systems Analysis and Design
Prof. V Rajaraman
Department of Super Computer Education and Research
Indian Institute of Science., Bangalore

Lecture - 26

Till now, we have been talking about a number of topics. We looked at data flow diagrams, we looked at structured English, we looked at decision tables. We also learnt about the fact that where, you create data flow diagrams. You can create a data dictionary, which is only an enumeration of all data which is used. And later on we looked at logical database design and how relational databases are organized or logically designed.

And some aspects of, so called database manageable system, which primarily converts a logical database to a physical layout of the database or a physical device such as a hard disk. And all these tools and techniques, which have been there for number of years. Starting possibly from the mid sixties, late sixties till even now. They are all called part of what is known as structured design.

Structured design is something, which where the technical term used for this technique of designing systems. And just like in programming languages, the early programming languages, which looked at structured programming was Pascal. And later on C programming came into picture, which also was somewhat like you know structured program. Except that Pascal was somewhat easier to understand. And C was a little bit more difficult, but C was lot more efficient.

And so the current trend is that we, go towards what is known as object oriented design. Structured design had a lot of as I said lot of use, over the years and even now. I would not say that structured design is not used. It is used depending on the project. If it is a small project you continue to use the structured design ideas. But, there have been newer techniques, which come into the picture.

Just like in programming, the structured programming ideas later on led to the. So, called object oriented programming methods with primarily the use of C plus plus and Java these two are, so called object oriented programming languages. So, the same organisms where expressed. Now, in an object oriented manner that is object oriented programming languages was the vehicle for communication. And the lot of advantages are found in

that. Similarly in a system design for businesses, there has been a movement towards object oriented system design, there reasons are several.

What are the main reasons for movement, towards object oriented design has been recurrent problem in data processing or in information systems. The recurrent problems was at, whenever a new project came. The new project was always started have been issue. And it is not used much of what transpired in the previous projects and so on. So, in other words the lot of work and lots of programs were written, for projects. And one would normally assume that all these, which has been done could be in some way reused in the new projects.

Particularly because, certain aspects of the old systems repeat, particularly if you are looking at say a class of applications like, banking applications. If you are done one bank then doing another bank should not be very difficult. And should actually be able to use whatever is used as a earlier bank with some little modifications. What is that new bank, this is a kind of a feeling people had. So, number 1 is that one should be able to modify and reuse, whatever is done earlier.

And secondly, whenever you design a system, one would like to kind of componentized the design, what is meant by componentizing the design is that. Suppose, you are able to come up with some self contained modules, in those self contained modules can be picked from a library of modules called component library. And use those components, whenever a new problem comes along based on requirement specification. You find out what are the components, from the component library you can pick up and put together to create this new system.

This is having to go through and designing each module from scratch, it will be very nice idea, if you could use this components. And it is necessary some small variations or small modifications are the components can be made, which is of course, possible in a software system. This method of componentization that is used in components, to design large system from earlier experience has been the dream of all system designers. Because, that will have lot of advantages.

Because the component, which have been designed earlier have been used. And whatever bugs are there or errors are there must have been already found out. So, whatever components being used is while tested. So, it will not hopefully bring in new

errors, only errors, which can come is in terms of interconnection of components, but not within the components themselves. So, that is the advantage that component is made efficient hopefully and made error free. So, you can just pick those components and use them together.

So, it will expedite the project completion time and time is money. So, it will also; that means, it will also reduce the cost of designing a new system. This idea of using components is important when the complexity of the systems increase. When the complexity is low one does not really worry too much. You can actually for a low complexity system, where the business rules are simple. And where the requirements are not all that complicated. System you are designing a small then all the structured methods you have been talking about, are the one's which are relevant which can continue to use it.

Even apart from that structured methods, really go towards also the componentization. And even in programming languages this idea of componentization has been, because right from the beginning of programming. In fact, the early programming languages like fortran, cobol and C and so on. People have always talked about sub routines and functions. So, functions effectively were thought of as components. So, you have a main program, which calls all this functions for resistance. And essentially strings together a set of functions.

And the functions can be picked from a library of functions or sub routine library or sub routines library of sub routines. This has been of course; they practice for design of large systems, where sub routines are well tested to some extent generalizable and so on. But, sub routines have a little bit of a problem. In the sense that they are not data independent in other words, there are some errors which can creep in, if you do not pass a right type of data to it.

And also the results of sub routine, if the sub routine has side effects. In other words, if that kind of a changes some aspects of the main program. In some way, that can really lead to difficulties in debugging and so on. So, these are the problems, which have been present with the idea of support, but the idea of componentization has always been with us. Let me talk one more analogy of componentization in the area of hardware, hardware design.

Hardware design again hardware are become continuously more complex. Today's personnel computer, the entire personnel computer particularly mother board is fabricated in one board. And the success of the design of complex hardware has been brought about because of the componentization again. Components there are memory is actually a chip, there was chip called CPU chip, there are some chips, which store are register a group of registers are in a chip.

So, these are all well tested integrated circuits, internally they will have thousands or even tens of thousands of transistors and they would have been laid out, tested producing large numbers and so on. And so they are first of all become inexpensive as the volume increases the cost of the IC is come down. Like, for micro processor is introduced it is expensive. But, soon there are clowns in the market and the volume increases. And the price is come down, that is why in the hardware area price have been coming down.

In fact, the other way it is been happen whereas, the power of the computer has been doubling without the price going up. In other words, you get double the computing power in every 18 months without spending more money that came about, because of the fact that large scale components are used. And they were use to kind of book together on a printed circuit board. A very complex system like a personnel computer like, of fairly complex personnel computer of today.

Now, people have been looking at this software people have been looking at disk. This with some envy saying that. These guys are able to make components. And reduce their cost, had been able to improve their systems without increasing, the price of the customer. Whereas, software cost are continuously going up because software is human intensive. And human salaries are always going up from year to year because of inflation and so many other reasons.

So, why not talk about software ICs, as another way of looking at these people said that componentization is equivalent to looking at a complex piece of software as a component. So, componentization or software ICs can we talk about a software IC. And this is the trend, which has lead to the idea of object oriented programming first. And based on that we expand it, to have object oriented modeling. In other words when you go modeling of the system from the requirements specification, you model using objects.

Just like in a database, we model using entities and attributes and relations. And we have a model of a logical database as a group of relations and physically. Of course, later on it is transferred to a, put in put into hard disk. Similarly, model is something which is a logical model, which uses the ideas of, so called componentization. The idea of reuse and so on. And becomes are actual you might say a part of a larger system.

In other words, the whole idea of object oriented modeling is that what are the components advantages are in this case, the components are going to be called objects. So, what are the objects, which are going to be developed and how can they make this, the object fairly general purpose you might say. And how do I know, what the object can do and how I know that a full system is to be constructed requirements. SRS is given that is system requirement specification is given, what objects to pick and how to kind of put together these are all the part of the modeling.

Once you do an object oriented modeling. Then the next step is that uses a model to ((Refer Time: 17:14)) do an object oriented design, which includes the programming aspects also. So, here you have an object oriented programming language for implementation. So, implementation issues come up an object oriented design because design even in systems analysis and design, analysis is modeling part and design is a part where you just implement.

Similarly, database also people are talking about object oriented database design, where the database or objects which are stored and objects can be retrieved and used and so on. It is just like having a library of objects. So, these are all being the trend. So, in the computer science curriculum, the object oriented design and object oriented programming is nowadays a very, very important part just like DBMS is an important part of the curriculum. A whole course is normally devoted to object oriented programming.

Another course is devoted to object oriented design, partly it is covered in software engineering courses and partly in separate course called o o design, depends on the curriculum or the universities where you have take the course. So, in this course I cannot; obviously, look by great detail the entire gamut of the object oriented analysis and design. That would really take us off to a different direction all together. But, what I would like to do is to be able to give you a flavor of what is object oriented modeling.

So, the questions I am going to be asking in this module are what are objects, first of all we have to define what are objects. And why necessary do we identify objects in application, why is it necessary as essentially what having talking about all along, that is the idea of need for computer componentization and things like that, the motivation is very clear.

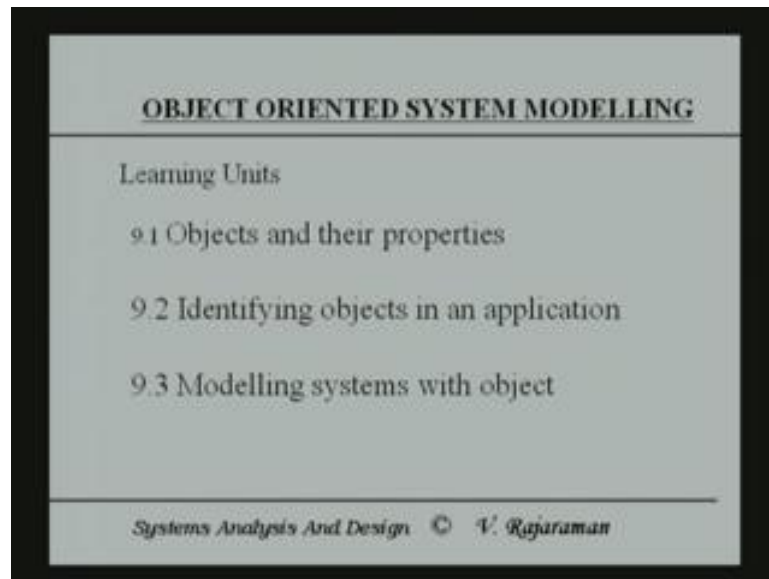
Ultimately, once you going towards object oriented modeling and object oriented design hopefully, the time for completion of projects will come down, the cost of completion of projects will come down and companies can well tested systems well in time. So, these are the primary reasons, why it is necessary and how to identify objects, given the system requirement specification there is a most difficult of interesting part.

So, given the systems requirement specification, how do I identify objects and once I know what are objects, then I can talk about identification the objects. So, then how objects are used to model the application, there is after all identifying objects, how do you use them to model an application. So, these are the primary questions that going to be asking.

And as I said in this probably at the most two or three lectures, I am giving you a birds eye view, an over view of the object oriented modeling. And these only a something, which is kind of make you understand why objects and make you also eager to learn more about objects and how to use them and so on. Todays world when you go to a regular job in software industry, you are expected to be able to design using objects. So, you should be fairly thorough with the subject, when you get out of your engineering college curriculum.

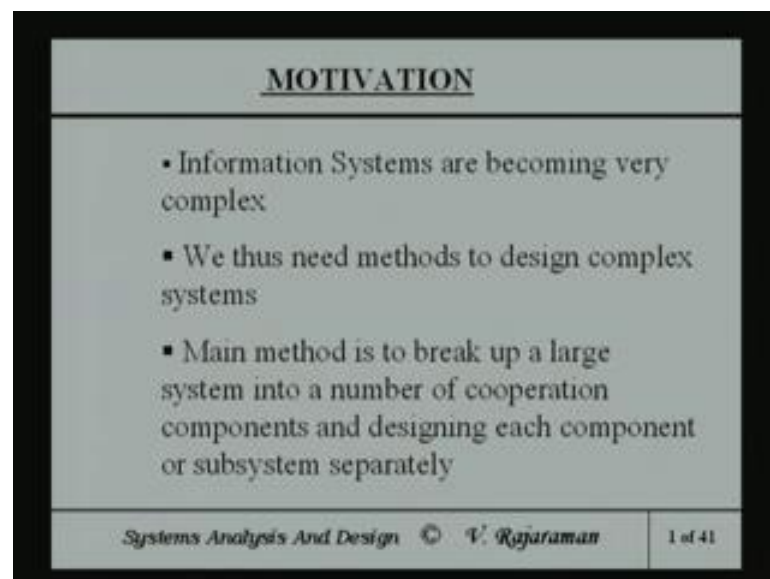
So, the major impact of my talks are essentially, to give you the basic foundation of object orientation and that is what these all about. Now, to summarize what I said is that, the entire object oriented modeling is towards componentization and towards reuse and towards also reducing the cost and using readymade components from a library to develop new projects. So, the new systems and so stare effectively, use earlier while tested components and so on, that is the whole idea.

(Refer Slide Time: 22:39)



And So, the learning units of course, is object in their properties, identify objects in an application, modelling systems with the objects that is what you are going to be looking at.

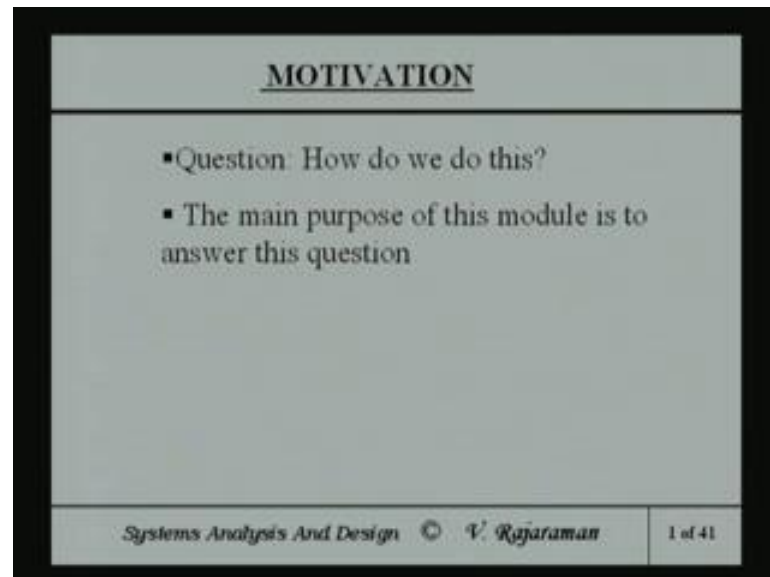
(Refer Slide Time: 22:53)



And I said that information systems are becoming very complex. So, we need methods to design complex systems. The main method is to break up a large system into number of cooperating components and designing each component or subsystem separately. And so the crux of the matter is a cooperating component, each component can be designed independently, but you should be able to cooperate, you have to cooperate then there is some properties, they must have some properties.

So, those properties are the important properties, which will distinguish an object from say sub routine or a function. So, the best what is a best method of designing or you might say breaking up a system into components, which can cooperate there is each one can use the services of the other.

(Refer Slide Time: 23:59)



So, how do we do this. The main purpose of this module is to answer this question will be take an analogy of the general idea of cooperating components and use of libraries and. So, on it is an example, in other words day to day example. Nowadays when people want to have a marriage conductor, the marriage conduction, now it is a very complex affair the organization of marriage.

Because an you have to be hiring of a hall, you have to kind of get the proper food, served at the proper time, you have to print the invitations and you got to have the proper time of a proper methods of arranging for reception and so on. There is a whole lot of small details one has to go through, lighting and gods know what there are so many, many items flowers which are required and so on

Nowadays, people do not do that all by themselves from scratch, they use you might say cooperating, cooperating agents each agent does a type of job, like for instance if you have to, if you want to kind of get proper meals and so on, you get over a caterer. The caterer's job is provides the food at the right time and so on. So, what the caterer does is

to give you, a set of menus for various parts like for instance breakfast menu, lunch menu, dinner menu, tea menu and so on.

And he gave you a choice menu and you pick and the cost associate with each, each menu. So, you decide on which menu you want to use, based on your budget then based on this menu, which has been selected by you, you tie to marriage the caterer how much person expected for each of the meals, breakfast this many, lunch this many, dinner this many and so on the number may vary.

The people will may take breakfast may much smaller, the people who take dinner. So, you make an estimate of the number and you also tell the caterer plus or minus some and whatever, is to be done he has to be able to if more people come, he still has to kind of do able to cater. So, once you give this order to the caterer, you do not interfere with that caterer any more, the job has been assigned to the the caterer and you expect the caterer to deliver as per your requirements. So, it is a sub contract which has been given.

Similarly, you can give sub contract for the marriage hall; you gave sub contract for the flowers to be given, what type of flowers want and so on. So, the question is the number of many contactors, each contactor is got a specific job and the contactors have to some extent cooperate, particularly in terms of timings and so on, certain things have to happen at the right time.

So, the project are the project of conducting a marriage is simplified by giving the complex job of conducting a marriage, has been simplified by giving contracts to several contractors who cooperate to get your work done, at whatever cost you already decide on your budget. So, it can be within the budget, within what you require and you essentially are giving orders and not interfering with those afterward.

Similarly, in the area of objects you might think of objects or components as contractors, the contractors been given some contract to do some job. And they are self contained to do that job in other words, there are they know what methods you use to get the job done, what data do use to get the job done. Just similarly like caterer, a caterer once gives you a menu the method of cooking you do not tell him, the method he knows the best method of cooking, he knows what items are required to be purchased on the market, to have the menu prepared for the number of people who are told.

So, in that case the items are like your data. So, data is kind of he has encapsulated you might say the requirements required data, the encapsulated method of a doing this job. And only thing you do is to give an order and when you give an order, to carry out then uses his own methods. And he uses his own data to deliver some output and that essentially what object oriented is somewhat you know it is analogous to that.

So, one way of looking at object oriented modeling, then how to pick sub contractors and how to kind of make the sub contractors cooperate to get a large job done. So, that is a primary, if you keep that kind of an analogy in mind then modeling becomes somewhat simple. In other words, you should be able to kind of do this, you might say you use of course, technical terms like encapsulation, we talk about polymorphism, inheritance and so on but these are all somewhat have analogies in terms of real world items.

But, we will describe we talk about these aspects like polymorph, which are very specific to the object oriented modeling world is this you might say jargon terms or terminologies or was you because they got meaning they have purpose. So, the main purpose of this module is to answer the question, how do we componentized our object and use them.

(Refer Slide Time: 31:07)

DESIRABLE PROPERTIES OF COMPONENTS

Each subsystem or component must

- Have clearly defined responsibility
- Acts when requested by an "order"
- How the component does its task need not be known to other components
- What the component does should be known

9.1.1 System Analysis And Design © 2012 Rajanman 1 of 41

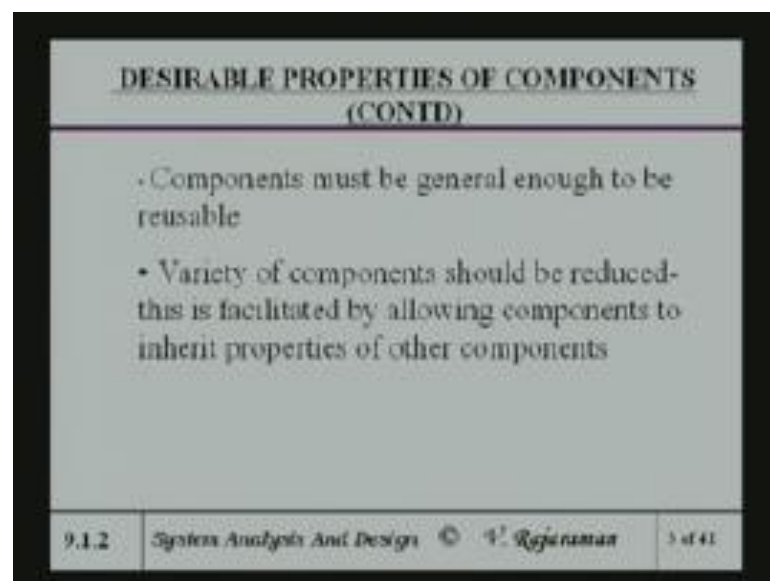
So, each sub system or components have a clearly defined responsibility, just like a caterer have a clearly defined responsibility. Acts when requested by an order. So, you give an order to a caterer he will produce the items you required. How the component does the work or task need not be known to you or to other components that is, it is self

contained in other words, how the caterer is going to cater to the number of people need not be known to the person, who is going to serve you know the flowers and so on, he does it independently, and he did not know.

So, they could be cooperating components, which they have to know like for instance suppose the food has to be served at the particular time, the marriage hall people must be ready to provide you the tables and chairs and arranged on appropriately at the right time to cater. So, caterer that marriage hall person has to cooperate with the caterer, to be help to provide the number of chairs, tables and so on required for each meal.

So, he essentially request, the another contractor namely, the marriage hall person to do it and so these two have to cooperate. Once the component does of course, should be known I mean, in case of caterer you know other caterer is suppose to prepare meals as per your menu, for serving number of people.

(Refer Slide Time: 32:54)



Components must be general enough to be reusable, there is in other words there are no contractor of the name we will be able to survive, unless he has a a method of cooking several menus, methods of serving. So, you know if you can give only one menu, you may not one customer may not like it. So, there are several customers. So, each customer may have specific requirement.

So, the person must have enough menus and must employ cooks and have methods are recipes, which is general enough to be reusable in other words, it should be able to kind of use a same cooks and a same vessels and. So, on which he has to be able to caterer to meet different customers with many different requirements, with many different number of people who may take the dinner and stuff like that see. So, it is general enough to be reusable, variety of component should be reduced.

This facilitated by allowing components to inherit properties of other components this somewhat specific in other words, the variety should be reduced in the sense that one component should be able to do a fairly number of different things. So, versatility of the component is important like you know the caterer, if the caterer says I can only prepare breakfast and I am not good at preparing dinner, you would go to another caterer who is is genuine, who is got a, in other words it is not a very good analogy.

Because, the inherit properties of other components the sense that you should able to inherit the properties of another caterer, who does good north Indian food you might say because the suppose, the person wants north Indian food we given for dinner then the properties of those recipes, which are required should be able to inherit or we taken by this in other words should be able to generalize.

Of course I am taking an example, which is not really directly relevant to data processing, but this brings out the ideas. And of course, I look at specific examples in data processing also.

(Refer Slide Time: 35:21)

The slide is titled "DESIRABLE PROPERTIES OF COMPONENTS (CONTD)". It contains two bullet points: "• Another aid to generalize the function of a component is to allow generic commands which make components do their task" and "• This is called POLYMORPHISM". The footer of the slide includes the text "9.1.2 System Analysis And Design © 2019 Rajan Kumar 5 of 41".

9.1.2	System Analysis And Design © 2019 Rajan Kumar	5 of 41
-------	---	---------

Another aid to generalization a functions because they have a key idea is generalization. And also there is something called specialization, we will come to that another aid to generalize functions of a component is to allow generic commands, which make components do their task When I mean take talk about generic commands there is we look at greater detail later on like you know, when you say admit, the case of a system in a college, you would be admit a student in the case of a hospital when you say admit it may be admit is a patient.

So, if suppose the admission process is a component, which can be which is general enough to care at either of these two, when you give admit in a hospital context you should be able to use appropriate methods. And we give admit in the case of a college you must use the appropriate methods. So, that is what is meant by generalization these called polymorphism say in other words, these able to kind of have different avatars for different purposes.

(Refer Slide Time: 36:46)

OBJECT ORIENTED MODELLING

Use of component oriented design

- Facilitates changes in the system at low cost
- Promotes reuse of components
- Problem of integrating components to configure large system simplified
- Simplifies design of distributed systems

9.1.3 System Analysis And Design © P. Rajanathan 6 of 41

So, object oriented modeling is your component oriented design; you make a changes in the system at your cost mainly, because the components are all. Suppose, if some change is required I mean same right from the beginning of this course that what are the most important aspects of system design, particularly software system design, phase design for change. In other words, you always have to design for change and not design for stability or permanence when you all design for change.

Then, if you use components and a change change requirements comes along, you may be able to change certain components only. And other components can be left untouched. To give an example of what I mean by this is, that suppose in a college you have a component or an object whose purpose is for say collecting fees, another object may be having a purpose of creating the mark sheet.

So, suppose the fee structure is changed. So, the object which does the fee calculation is only one more touch as far as the examination and marks and creating a mark sheet is concerned, that component is left untouched. So, any change and requirements has to be met; that means, when a big components also have to be big components, in such a way that they have some orthogonal purpose.

So, when requirement of change arises, you can just change some of the components and not mingle with the other components. So, the whole idea is the fact that change can be implemented at low cost, promotes reuse of components. So, reuse of course, have been

talking about the importance of reuse that is whatever we did for an earlier project should again be usable in the new project.

Projects of integrating components to configure large system simplified and by the kind of models and by the kind of properties, which objects have it also simplifies the design of distributed systems in other words, for you remember distributed systems. The question what is distributed system, nowadays if you look at a computer, no computer is a self contained box sitting alone, number of computers are connected together by either local area network or the internet or whatever.

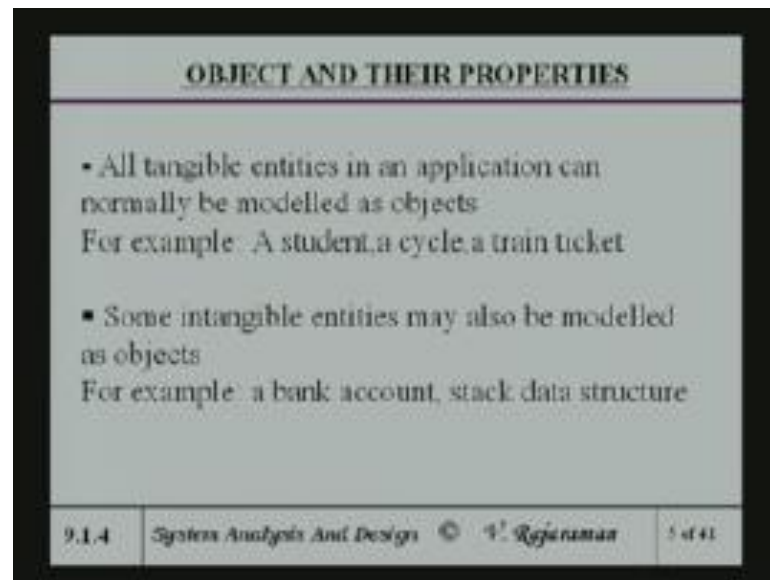
In other words, the isolated computer does not know any more exist, every computer since be connected to every other computer. Sometimes every other computer in the world for instance in your home PC, when you connect to the internet through a internet service provider, you are automatically becoming a part of worldwide network. So, you were connected the whole world, you do not know that, but you are connected.

Virtually, you are connected you might say through the internet service provider because internet is a network of networks and networks are all international networks, otherwise you cannot be going through the web and things like that. So, the point is that no machine is an isolated machine and so distributed computer means that you have a number of different servers, which cooperate to get the job done.

Like you might have in a local area network, in any college then we want computer, which will be a printer server, there be a another computer which will be a storage server, a third server may be a number crunching server and fourth server may be for communications and so on. So, also you distribute jobs to these servers for load balancing, you know what overload a particular server and you would like to balance.

So, this kind of load balancing and distributing task become simple, if you have some kind of a component because different components can be assigned to different different processors or servers and they can cooperates each one of them, to get the job done. What can the modulus objects all tangible entities can be in application can be normally modeled as objects a student these are all tangible means you have to exist has an animate inanimate being.

(Refer Slide Time: 42:01)

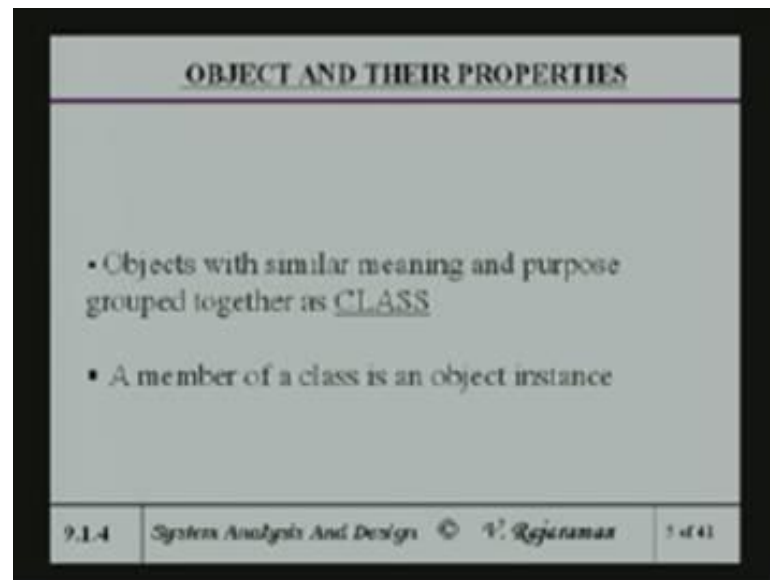


A student, a cycle, a train ticket, a vendor an item an order, so these are all actually entities you might say these are tangible entities. A tangible in the sense that a vendor has got an address and exist or item is something, which is physical and order is something is an number and there is certain kind of just like, the train ticket is an order is something, which you place an order with a document.

And there are also intangible entities like a bank account, stack data structure or insurance policy and stuff like that and to some extent, the order is somewhere in between and the order can be thought of us intangible from that point of view. Because order unlike a vendor, order is not a physical entity, it is a kind of a logical entity like for instance whereas, train ticket is more of a physical entity because you can hold train ticket in your hand, that is the reason why order also is something, which can hold in your hand to that extent it is you might say tangible entity.

And, but data structure is something which is clearly an intangible item, but does not really matter.

(Refer Slide Time: 43:55)



Objects with similar meanings and purpose are grouped as a CLASS, like for instance a student may you know is an object, a student say for instance is a entity, but you have class of students together of similar properties, students of a class. And so those CLASS. So, it is called a CLASS that is object with similar meanings and purpose are grouped together and are called a CLASS.

A member of a CLASS is an object instance in other words, I want to make the distinction between object, it is an object instance. Object instance is a specific one particular kind of a resistance of an object whereas, the object is generalized. It is a general idea, instance is a particular case of an object particular, like particular student in object instance whereas, the class of students is an object class.

(Refer Slide Time: 45:20)

CHARACTERISTICS OF OBJECTS

- All objects have attributes

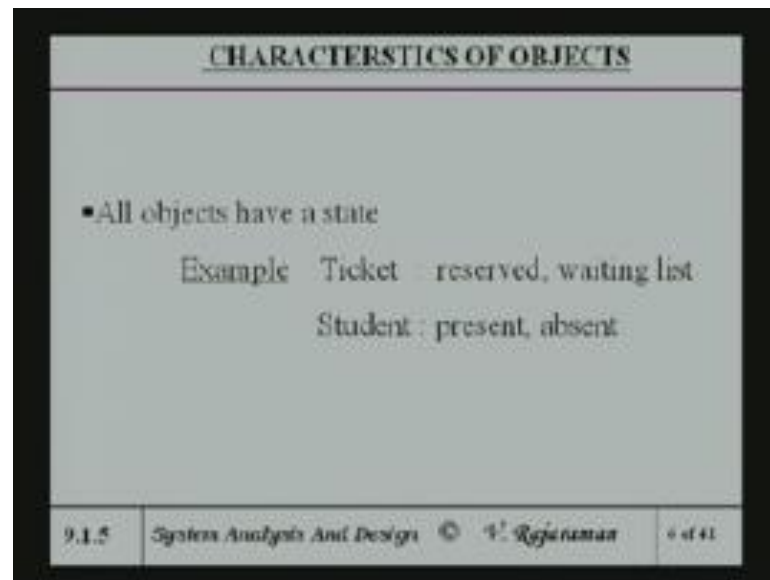
Example student Name
 Roll no
 Address
 Year
 Department

9.1.5 System Analysis And Design © V. Rajaraman 6 of 41

All objects are attributes, attributes just like you know entities, attribute kind of a thing. Attributes, itself is a kind of an entity might say and it is got attributes like student name, roll number, year of admission, department there will be other attributes also like, he is hostel resident, hostel number and local guardian and so on. So, number of attributes of an object depends upon the application to which you want to put.

In other words, you should have a fairly general sort of entities, a large number of entities corresponding to an object class, to be able to do a number of different things with the object to your extent. I think work should be I think it is a question of art, you know to some extent it is not really science, in the sense that you cannot have a strict rule saying that an object cannot have more than seven attributes or in a, but you normally have a certain number of attributes, which are relevant to the application.

(Refer Slide Time: 46:42)



CHARACTERISTICS OF OBJECTS

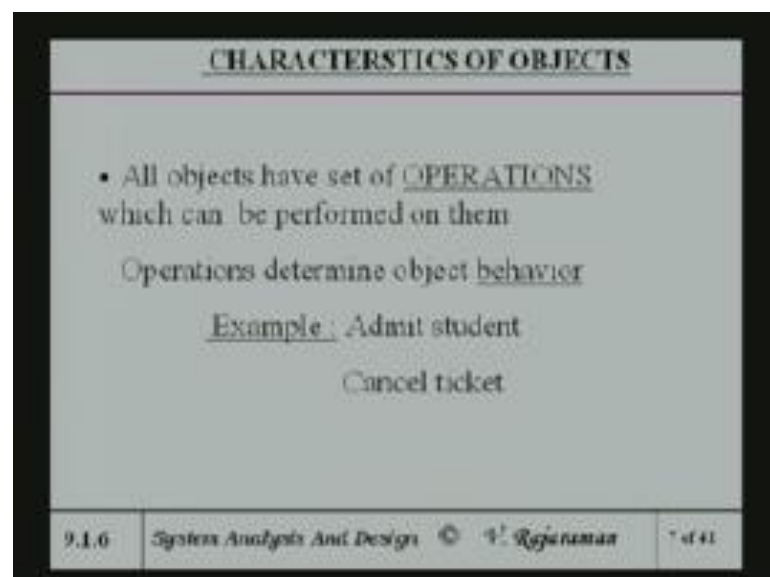
- All objects have a state

Example Ticket : reserved, waiting list
Student : present, absent

9.1.5 System Analysis And Design © V. Rajanathan 6 of 41

All objects have a state that is important, state for instance there is ticket, an object like a ticket has a state like reserved or waiting list a student has present or absent or student. If it is a examination kind of a thing, student state been passed or wait or not passed or deferred an admission process a student will be admitted, wait listed or kept pending. So, we have or either of course or admitted. So, these are all states and so each object will have states, which will change as per the context.

(Refer Slide Time: 47:41)



CHARACTERISTICS OF OBJECTS

- All objects have set of OPERATIONS which can be performed on them

Operations determine object behavior

Example : Admit student
Cancel ticket

9.1.6 System Analysis And Design © V. Rajanathan 7 of 41

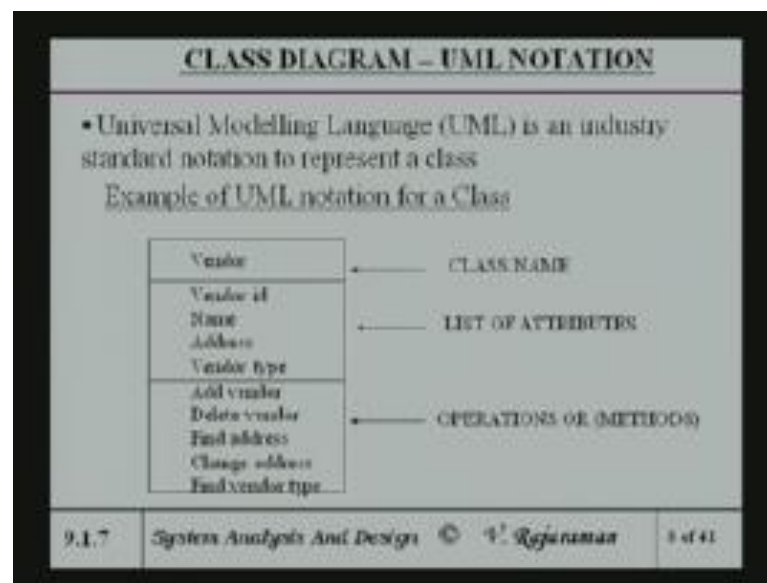
All objects have set of OPERATIONS, which can be performed on them admit a student, cancel a ticket. See, now the OPERATIONS, when you say admit a student cancel a ticket and so on, when you do a implementation of that OPERATION, the implementation is done by some program, this program which does implementation of the OPERATION, is normally called in the object oriented context a method. The method of course, the method implement is an algorithm of a method of admitting or set of rules, business rules for admitting. So, the business rule that being incorporated in that.

So, the methods are essentially part of the object. So, this is called encapsulation the method encapsulates within the object and you invoke the method by sending out a request to object to invoke the method. So, you send a message or an order, the order gets implemented by means of the method, which is in inside the the object. So, he knows the difference between the idea of a sub routine or functions and the idea of an object.

Object has a both general methods, as well as data of which the methods act all encapsulated within, you might say sealed boundaries. And it does not depend on a data from outside, it is self contained. So, he only send a trigger or an order and the object gets activated using whatever object data is required is encapsulated within it. Unlike a sub routine where some data may be global, some may be local and so these where it deals with all kinds of problem because the data is not re encapsulated, this case the data is encapsulated.

The method again OPERATION, which requires a data is also encapsulated, he only give a trigger for this.

(Refer Slide Time: 50:14)



There is something called universal modeling language, which has been designed by three stalwarts in the area, object oriented modeling, which and two other people I do not know other name and the three people who really cooperated, each one have a different method and ultimately they thought that it has been standardized by cooperation And say cooperated to come up with an universal modeling language, which has rotation for classes objects, entities, attributes, operations and so on.

And also as well as many other parts. So, UML is a very complex modeling language like, a data flow diagram modeling, object modeling is normally done nowadays to the UML modeling tool. In this particular set of lectures I am not going to look at UML at great lengths where, simple reason that a whole books have been written on UML. And in order to deeply study UML where, you go through these books and normally it will be taught in object oriented design course, which are follow this course.

And. So, you should really understand those kind of flavor of what UML looks like. So, I am not going to talk about UML at any great length, for a simple reason it is outside the scope of this particular course. This course is a very broad base, you might say a foundation course on which you build the rest of the computer science curriculum, it comes early in the curriculum for you to understand.

So, UML notations of a class is there is a class name, in the case of say for instance, I am taking in a vendor, vendor has got a vendor id, vendor name, vendor address, vendor

type, I am just taking some attributes then I need other attributes, depending upon the application. And the operations or methods, like a method is essentially means that they are algorithms and encapsulating within that class say.

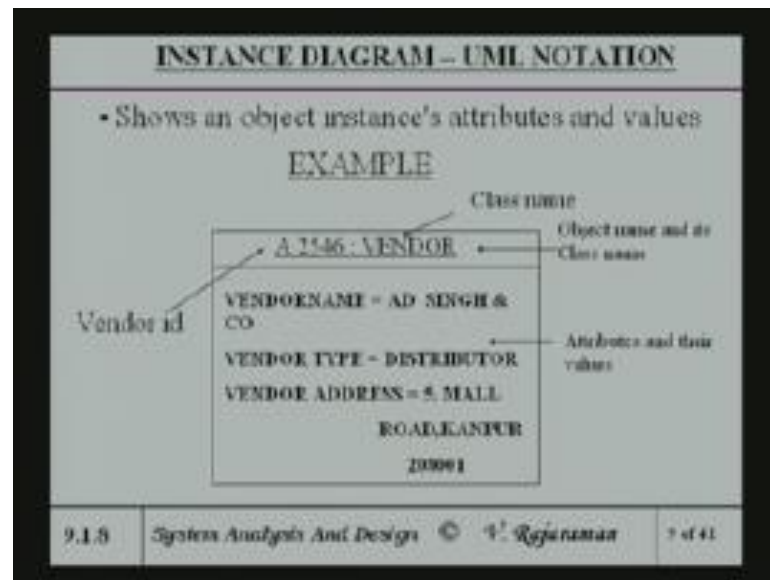
So, add vendor, delete vendor, find address, change address, find vendor type all these things, which are operate on the attributes, which are already stored internally. See the way we looking at database design, earlier in database design when you do a logical entity attribute modeling, you came up with a relation, the relations only talked about the data, they did not talk about any kind of a method which will operate in a data.

In fact, the coming of the object oriented system partly came out of the work, which has done by database people, partly out of the work which has done by the AI people in terms of agents and agents and so called software agents and so on. And also programming people, so it kind of many of the disciplines together cooperated, in terms of coming up with this general structure.

So, it is actually using the ideas from all these and hopefully a better ideas from all these to get to a system, which is hopefully going to last longer, then what others lasted, I mean of course, I can never predict in computer science things change, so rapidly. But I think it is a, the change is what about by the fact that we are very young field and we are always try to improve and improve our productivity, else wherever cost and so on.

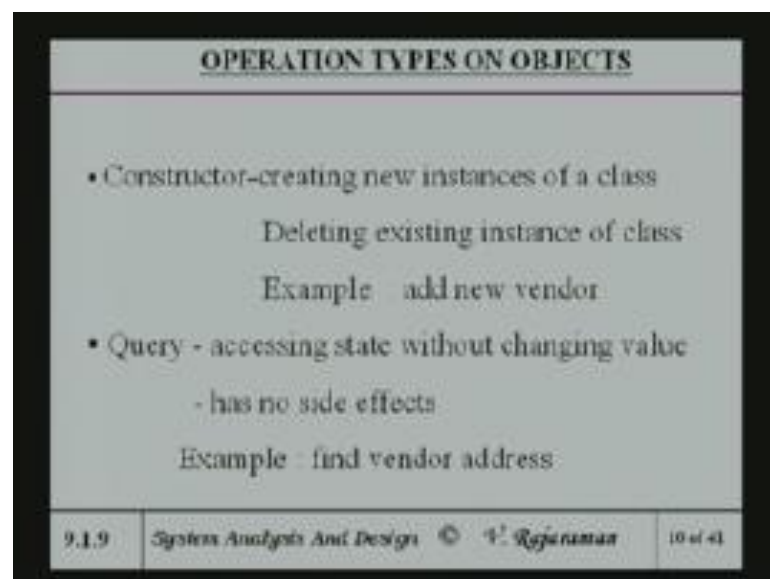
And in that process we are always trying to found out better and better methods, which is only a good thing because that way as a software engineer, working in the field you will always keep hope to that you will always try to find out, what the latest trends, keep in touch and learn as you go along. Because learning is in our area, you have to be continuously learning and things change in ask things change, you have to continuously learn when you changes which happen.

(Refer Slide Time: 55:19)



Actual values and attributes when you give because of object instance. So, object instance of this object is say vendor id see, object name in its class name, see vendor id is a specifically about because you want to kind of talk about a specific vendor, when you do an operation. Vendor id and the class name and vendor see these all actual values of attribute whereas, vendor name, vendor type, vendor address whatever attributes are there, the values may be attributes and. So, the actual values are in object instance.

(Refer Slide Time: 56:04)



Constructor is an operation types on objects, what kind of an operations an objects can perform. Constructor is creating a new instances of a class, deleting existing instance of a class for instance add a new vendor. And in other words, I should really say that I am putting an updated you might say putting a new instance or deleting existing instance, suppose is nothing may be static you know I got a vendor list and I have vendor list today and I would like to kind of particular vendor is not performed very well, I would like to relive that vendor.

When new vendor has come into the market, I would like to add that vendor and try him out. So, deleting an existing instance and adding a new one is gives a constructor and a query is accessing state without changing the value. So, there is no side effects like for instance a query is what is a vendor address. So, vendor address is found out by just looking at what is already stored. So, data is stored and. So, it is data independent.

Data independence is one of the most important parts of any object oriented design. So, I need to have data independence. So, I query accessed access a state without changing the any value has no side effects, like you know some sub routines have side effects.

(Refer Slide Time: 57:52)

OPERATION TYPES ON OBJECTS

- Update - changes value of one or more attributes
 - affect state of object
 - has side effects

example - change address of vendor

Implementation of operations on objects called methods

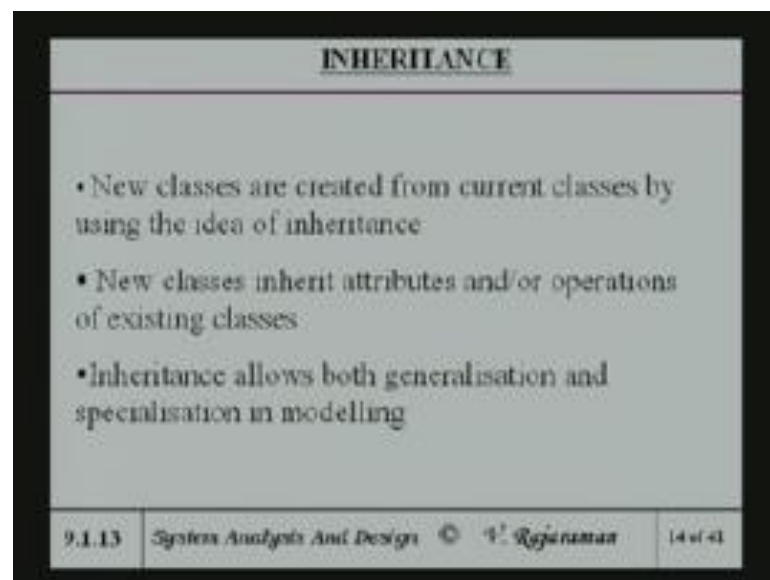
9.1.10 System Analysis And Design © V. Rajanman 11 of 41

Change values of one or more attribute update, affect states of object has side effects. So, change of address of a vendor, there is a any errors it is a requirement you know the vendors address is changed hopefully, outside the object there is no affect. So, only

internally there are change the address and there address change gets reflected in whatever method are used in their object.

Implementation of object operation on objects called methods, because already used the terminology called methods. Implementation there is any algorithm to implement an operation is called a method in the object oriented context.

(Refer Slide Time: 58:37)



New classes are created inheritance is the very interesting and important idea, the reason why inheritance is important idea is that whenever is a new application, which comes along I want to use some of the objects already created in the old application. I like to be able to use that and create a new object, which inherits some of the other properties in the earlier object. So, new classes are created from or current classes using idea of inheritance, new classes inherit attributes and or operations of existing classes.

Inheritance allows both generalization and specialization in modeling, specialization in the sense that I would be able to kind of take out, some of the methods we have been and make a specialized or make it where general by adding more methods, but data is encapsulated and so on. So, inheritance like is an important idea.

(Refer Slide Time: 59:53)

INHERITANCE

- Specialisation - given student class, arts students and science student are two subclasses
 - Subclasses inherit properties of parents and in addition may have their own special attributes and operations

9.1.13 System Analysis And Design © V. Rajanathan 14 of 41

Specialization gives to a, I talk about inheritance and specialization at much greater length, next time because you are important ideas to be discussed. So, I do not want to rush through with this. So, next time we start with this inheritance idea and go further.