What is modelling?

Uri Abraham *

March 15, 2013

Abstract

It is not so easy to explain in the first lecture and on one leg what this course is about, but I shall try. It is about the modeling activity.

The course is about modeling. So I'll have to explain, even in rough approximations, what is modeling. Later on we will give mathematical definitions, but today we remain at the pre-mathematical level: at the elementary stage in which concepts have not yet taken any formal appearance. At this stage we employ a metaphor and we say that a model is a map. It is a simplified and scale reduced representation of reality–or rather of some of its relevant aspects. We can never directly comprehend the area that the map represents with its mountains and valleys, wadis and rivers, we can never touch this vast area and see it at once. Unlike that which it represents a map can be comprehended, it is a concrete object and so it allows easy answers to relevant questions: what is the distance between this village and the next? and how far is the sea? A map is never a complete representation and it should never be confused with reality itself. We shouldn't conclude that the earth is flat because the map is... A map is used to learn about the area represented and to understand it. A map is used to communicate, to plan, to record information, and to build (a road for example).

Yet this course is not about drawing geographical maps, or building airplane models—it is rather about formal mathematical models. The modeling activity in this course is that which seeks formal and mathematical representations of complex distributed and concurrent situations. There are three distinct aspects of any formalization activity.

*Departments of Mathematics and Computer Science, Ben-Gurion University, Beer-Sheva, Israel. Email: abraham@cs.bgu.ac.il

1. First there is the reality which is modeled. A complex array of hardware and software systems, computers which can be broken or overheated, communication machines and lines, transmitters and receivers with their multitude electronic circuits, human operators, mechanical instruments that computer systems manipulate and so on.

2. Then there is a certain informal, intuitive description of the system. One which the experts use in order to explain the system and to argue about its properties.

3. The third aspect is the formal part. It is a mathematical object that reflects the reality it is supposed to model. It must be construed in harmony with the intuitive description, because the adequacy of the mathematical object to serve as a model is always judged by reference to the intuitive pre-formal understanding. Yet, and this is one of the greatest virtues of an adequate formalization, the model deepens the intuitive understanding and sometimes even changes it.

   We discern two ingredients in the formal part: semantics and syntax (language and structure).

   (a) The formal language is a collection of well-formed statements that can be used to convey information about the reality being modeled.

   (b) The structure (or model) is a mathematical object that reflects that reality. Any formal statement can hold in the structure or be negated by that structure.

Any model must have those two ingredients, language and structure, and it is not a model if one of them is missing. If we only have a language we may get the wrong impression that the language speaks directly about reality. Indeed this is the case with natural languages such as Hebrew or Chinese spoken by the engineering experts, but a formal mathematical language speaks about mathematical objects, and a proof establishes a mathematical situation–it does not build a channel or a bridge. On the other hand if we only have a mathematical structure with no language, then we shall not be able to say meaningful statements about the structure.

In the literature we often find this term, *model*, used in different ways, but for us a model must have both the structure and the language present. When you read one of the textbooks or papers recommended you will usually find this term used in a somewhat different way and it is important to identify

those cases. For example, we find the following in the text-book of Peleg (Distributed Computing)

> There are numerous models for distributed systems, differing from one another on a large number of important factors and parameters. One major distinction is between the *message passing* and *shared memory* models.

The term *model* is used here in a non-technical way (which is perfectly legitimate of course). We could say, instead, "message passing *mode* of communication" etc. but there is no reason to restrict the term "model" to its technical meaning once we are aware of the different usage that the term may have. In our course, however, the term model has a precise mathematical neaning.

So one of the prime aims of our course is the mathematical definition of models. We will describe in detail two brands of models: Tarskian models and temporal logic models. Tarskian models are simpler and they are the structures mathematicians usually use, but computer scientists concerned with formal correctness proofs mostly employ temporal logic models to study concurrency. The tension between these two brands of models is a main concern of our course: we will study the merit of those two different approaches and their suitability for modeling purposes.

Why is (formal) language such an important ingredient for the modeling activity? Because there is no understanding without a language. Language is essential in order to give precise meaning to the ways that we organize and model the external reality. And, since we want to be able to *prove* properties of concurrent algorithms and distributed systems, we must have a well defined mathematical language in which these properties and the needed proof assumptions are clearly expressed.

1 Distributed and Concurrent systems

In this section we define some of the vocabulary that will be used in this course. Many of these terms will acquire a more formal shape later on, but now we are interested in plain English. A *system* is any complex object consisting of sub-parts that are in constant flux (they are changing). A complex picture is not a system although it may have many parts, but a watch is a system. Of course, we refer here to computer systems, and the term system can refer to a single computer (*operating system*) or to a complex

array consisting of many processors. A *processor* is an entity that operates independently of other processes. Usually, a processor executes a piece of a program (a *protocol*) and the resulting execution is called a *process*. A *distributed system* is one whose different processors are located at different places. A *concurrent* system is one whose different processors operate independently of each other. If the processors operate in unison, under some global clock, the system is said to be *synchronous*. It is *asynchronous* when the different processors and communication devices have no global clock on which they can rely and the local clocks of the processors may operate in different and variable rates. We usually assume that each of the processors of the system is *serial*. That is, it executes its operation one after the other in a serial (linear) order. The code that each processor executes is a serial code, sometimes called a *protocol*, and the totality of the protocols is the *distributed algorithm* that the system executes. The same system can have many different executions. For example, if an airplane is a system and a flight is its execution, then it is inconceivable to have two executions that are exactly the same. The execution of a computer system depends not only on the algorithm, but also on the timing of the different operations by the processors. Some processors may be faster than others, and some may have a variable speed of operation. So the *behavior* of an asynchronous system is not determined: the system is *non-deterministic*. There may be other reasons for non-determinism, for example *failure* of some processors or communication devices.

There are two main modes of *inter-process* communication: *shared memory* and *message passing*. Usually, the expression used is "shared memory model", and "message passing model", but we shall use the term "model" in a very technical way, and so the term "mode" is preferred here.

Concurrent situations can be extremely complex and hard to comprehend. Distributed protocols and algorithms are notoriously difficult, and their correctness proofs are often partial and flawed. There are critical systems whose failure can have dire consequences: a braking system of a train, and a distributed control system of an atomic plant are examples of such communication and monitoring systems. Hence the need for methods that can be used to ensure that the system behaves properly. In fact, even publishing an erroneous algorithm can be an unpleasant experience... The best guarantee is of course a mathematical proof. Yet one must understand the limits of such proofs and what exactly is proven. The most beautiful proof in the world can only give what she has to offer, namely that the proven statements hold in every model that satisfies the assumptions. A proof is

never directly related to the real world. The question of *adequacy* that is, the question whether the models truly and faithfully represent reality is not resolved by mathematics. Only experience, common sense, and expertise can be brought as evidence for or against a model's adequacy.

It has been argued that our thinking and ways of arguing are serial and thence the inherent difficulty in analyzing concurrent systems. There is however very little evidence that this is the case, and, on the contrary, we routinely perform different actions concurrently while splitting our attention and thinking between diverse missions. One may even argue that in the animal kingdom concurrent thinking is necessary for survival, and linear processing is a death sentence.

Why is it the case that modeling, formal specifications, and correctness proofs are so important in computer science in general and in concurrency studies in particular? In other engineering endeavors such as building bridges or audio systems mathematics plays a crucial role, but there are no formal models or correctness proofs (maquettes are extensively used of course, but these are not models in our sense). One reason is that although computers are built with electronics and chemistry, they are essentially symbolic and computing machines. It is the very essence of computer systems that calls for logical analysis.

An interesting painting by Matisse, The Painter and His Model (1917, Musee d'art moderne de la ville de Paris), is a deep reflection by one of the most important artists of the twentieth century about the relation between reality and its modeling–and about the frustration of modeling. There is no motion in this painting, it is silent and we are silent too as if we do not want to disturb the two persons in this tiny room that we entered uninvited. As our eyes wander about the bigger sections of the composition and we try to understand their relationship, we discover that there are two or three or even four persons that inhabit the painting. First we see the painter and his model. The painter seems frozen in his work, painting his model who sits in the background on a pink armchair. She seems quite far and removed from the foreground, back in the other corner of Matisse's small studio. Yet the colors of the model are very vivid; she is in green and her sofa in pink, but the painter is in some rather dull ocre, and if colors tell (and for Matisse colors are essential) the artist is of lesser importance. Matisse is present of course since without him the painting would not come into existence, but the painter is less important: it is the model who is the main protagonist of the painting if we may say so. Then we see the painting in the painting, that is the unfinished canvas that is being painted. This inner canvas is identified

as a painting called "Laurette in a Green Robe" that was painted about a year before (that is in 1916). At first, we may think that this inner canvas is but an object, not a person, just a canvas with some incomplete painting. But then we think that although everything is but a canvas, we *do* take the model and his painter as real persons sitting motionless in some imagined universe; so perhaps the person in the canvas has right to be also part of this reality. For surely Matisse himself would think so. And thus it seem that there are three persons in our painting: the artist, the model, and Laurette in Green Robe. Our tableau is called The Painter and His Model, and we now realize that it is not so obvious to determine who is the artist and who is the model. Is the model actually that unfinished canvas in the painting? It is after all a model (a picture) of the "real model" called Laurette in a Green Robe now in the Metropolitan Museum. Or is the tableau itself a model of the artist's room at a very particular intimate moment in Matisse's life?

But then, finally, we realize that we (the spectators) are also in it. Included and excluded, as if witness to some intimate moment that was almost lost forever. A model is just a rendition of reality, maybe of some essential features of that reality. Then we see the big window. Surely Matisse rented this studio because of that beautiful window. Reality is out there, in the street, houses, open air and light that fills the window and partially enters the studio. The artist is in the dark, Laurette and her painting in the illuminated part, and the mirror-this enigmatic object which is the key to the painting-floats above all and is the reflection of reality. It is directed towards us, and so we should see ourselves in it. Do we?