Review of Miller & Scott's Book Complex Adaptive Systems An Introduction to Computational Models of Social Life Examples-Chapter 7, No.1d

A Review from the Point of View of the DAAI Paradigm
*

Gerd Doeben-Henisch
doeben@fb2.fra-uas.de
Frankfurt University of Applied Sciences
Nibelungenplatz 1
D-60318 Frankfurt am Main

February 5, 2020

Abstract

After a first reviewing of the methodological ideas of the book of Miller[1] and Page[2] (2007) [MP07] the reviewer will analyze and discuss examples from chapter 7 as they would be formalized using the DAAI paradigm[3]. In this text example no.1 is analyzed and discussed. This is the forest fire model with a special extension of the text of the book by the reviewer.

[1] `https://www.cmu.edu/dietrich/sds/people/faculty/john-miller.html`

[2] `https://sites.lsa.umich.edu/scottepage/`

[3] Here version 15.06.07 of the DAAI paradigm is used. See for the text uffmm.org

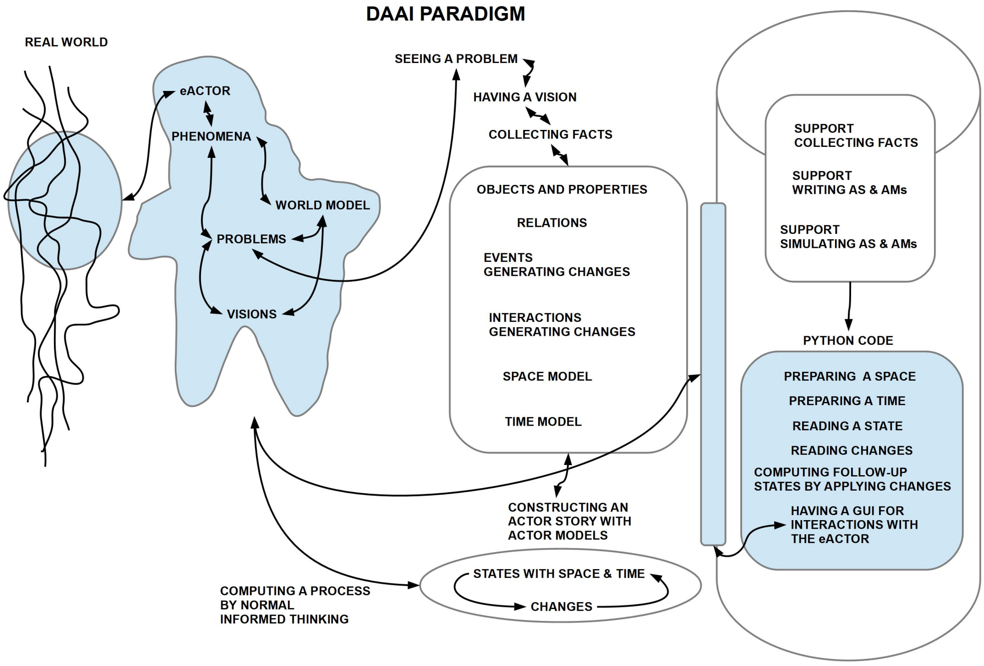

Figure 1: Real world - expert -DAAI descriptions - Simulator

1 Code and the DAAI Theory Format

The goal of this text is to do an exercise in realizing a real programming code – here with python – as part of a DAAI theory example. As it will become clear immediately it is not sufficient to write *some* code. One has to have an explicit formal framework and within this framework there exists a text written in python which is explicitly embedded in this framework.

Figure 1 outlines the main idea for the relationship between the DAAI paradigm and some code; in this case the code is written in the python programming language embedded in some server related coding.

The main idea is given in the two perspectives of the *symbolic space* of a DAAI analysis at one hand and the *coding space* on a server.

Symbolic Space: The symbolic space describes the main points of a problem description, a vision statement, a collection of facts characterizing the details of the vision, then the construction of an actor story with actor models. The symbolic space is the outcome of a process where some experts communicate with each other to elaborate the symbolic space as their *symbolic representation of their cognitive processes* underlying all these activities. The *construction of an actor story* requires that the experts are able to apply certain change rules to certain states to compute those changes which produce a new follow-up state.

Practical Limits: One can stop with thinking at this point of the story. The symbolic space as such is exciting and gives us a lot of new options. But if you start working with the new DAAI paradigm then you will soon experience a practical problem: the complexity of the symbolically encoded structures becomes quickly to be to difficult to handle them manually. This calls for a computational support. But this support should have the format, that the clarity of the formal structures will be conserved: no fuzzy code for clear formal structures! How can this be achieved?

Formal Assistance: There exist examples in the realm of mathematics. Today you can use *symbolic math* programs where you enter the well known formulas and the computer computes these formulas as you know it from pure mathematics. In this sense we want to provide a kind of *symbolic DAAI*. This means that the DAAI experts work *as usual* when they construct the DAAI symbolic space and work with it. This means a DAAI expert enters those formulas, which he would write down for his facts and for an actor story with possible actor models. This can be done either by a simple text based system interface or – later – with a GUI or a speech based interface. How this will be processed inside of the system does not matter. Important is only whether the input is answered by those constructs which are known from the DAAI paradigm: *states and change-rules*. Furthermore it is expected that the system is able to *apply the change rules* by itself to the states and *computes automatically the follow-up states*.

The following text is an experiment to get some ideas how such a *symbolic DAAI* could be realized. For this the first example of the book of Miller & Page is used as a case study. In this *Forest fire Model* not all aspects of a DAAI analysis are needed. The missing ones will be analyzed in later examples.

2 Problem and Vision

The *Forest Fire Model* with *Fixed Rules* is described in chapter 7 on the pages 102-104. The textual description of the example in the book will not be repeated here. This text starts directly with the re-analysis using the terminology and the style of the DAAI paradigm.

Problem, Vision: A DAAI analysis assumes usually a *problem* and a *vision statement* to start the process. The *problem* characterizes the first triggering factor for the existence of the process describing some *deficit*, and the *vision* describes a possible state in a possible future, which one will enter, because there exists at least one criterion why this possible future state *appears to be more promising* than the actual situation classified as having a problem. Given both statements one has a kind of a *norm* by which one can evaluate later the result of the process as being *sufficient similar* to the vision or not.

Example without a Problem and a Vision: If one looks to the *Forest Fire Model [FFM]* – here called example no.1 – then one can not detect a problem statement nor a vision statement. That trees can grow and that lightning can start some fire which burns trees down is a natural process which is known to happen regularly. As such this is not bad and not good; it happens.

Reformatting Example No.1: As known from countries with many forest fires the burning of the forest isn't understood as being *neutral*. Because the human population is spreading in areas which are occupied by a forest the occurrence of a fire is judged as an *unwanted event*. A forest fire can kill people and it usually destroys houses, cars and more goods belonging to some people. Whether the real problem is perhaps the behavior of humans to settle in areas with forests or not will not be discussed here. For this example we take the hypothetical position, that the settling of humans in areas with forests is OK. Then we can define a problem and a vision:

Problem Statement: The number of occurrences of forest fires in area X is too high. This causes damages to human settlements there. This is bad.

Vision Statement: One can arrange the forest in area X in a way, that the probability of the occurrence of a forest fire will be zero. Then no human settlement will be damaged. This is good.

Non-Functional Requirements [NFRs]: The vision "that the probability of the occurrence of a forest fire will be zero" can be understood as a *non-functional requirement* [NFR] which should be valid for the whole actor story. This NFR would be *falsified* as soon as at least one human settlement would be damaged by forest fire. Depending from different goals there can be different NFRs which can induce conflicts.

3 Actor Story

Having a problem and a vision statement one has in a next step to clarify under which conditions the vision can work. Here two main cases are possible: (i) *all* elements needed for a solution *are* already *known*; (ii) *not all* elements needed for a solution *are known*. In the actual example no.1 it is not from the beginning known, what is the ideal setting for an area X with a forest being save for human settlements with a probability of 1.

In such a case of not knowing yet all necessary elements one has to organize some *research* to answer the open questions.

3.1 Research: Defining an Experiment

Doing research means that one tries to find *answers* to identified *questions*. Here we have the question under which conditions the probability of the occurrence of fires which can damage human settlements is zero.

Collecting Facts: Asking for such conditions means asking for known parameters which influence the behavior of fires in a forest. Taking the example from the book[4] the following parameters are assumed:

[4] Extended with the human factor

1. The assumed *space model* – here called area X – is a 1-dimensional line where the positions are equally distributed on this line. A line with trees is called a *forest*.

2. The assumed *time model* assumes the following phases: during a year: (i) spring (sp), (ii) summer (su), (iii) autumn (au). and winter (wi). Outside the year: the *preparation time (pt)*.

3. With probability hs one position for a human settlement will be selected in the forest and immediately a human settlement h will be installed. This is done in the preparation time.

4. During spring the *probability g of growing* of trees at a certain empty position which immediately is followed by the growth of a tree t.

5. During summer the *probability f of lightning* which causes burning of a tree t making the tree a burned tree T.

6. The *neighborhood N* of a tree. Neighborhood is fulfilled if the position besides a tree is occupied by another tree.

7. During autumn the *production $S(t)$* of a forest is the number of trees t at the end of the summer after removing all burned trees T and all burned human settlements H.

3.2 A Textual AS

There are at least three modes for an actor story to become realized: (i) textual (TAS), (ii) pictorial (PAS), and (iii) mathematically (MAS). Here only the textual version is written down.

1. **S_0:** It is preparation time (pt). There is one line L for possible trees t. This line is called the *forest*. In the beginning the positions are empty.

2. Change rule X_{pt}: Select one position in the forest with probability hs for a human settlement h and realize it.

3. **$S_1 = X_{pt}(S_0)$:** Spring time (sp). The forest contains one human settlement h.

4. Change rule X_{sp}: If there is a free position then with probability g let a tree t grow at each free position.

5. **$S_2 = X_{sp}(S_1)$:** Summer time (su). There is a forest with trees.

6. Change rule X_{su}: If there is a position with a tree t then let the lightning make this tree a burning tree T with probability f. If this tree T has a neighbor (left or right) then transform the neighbor as tree t in a burning tree T or as a house h in a burning house H.

7. $S_3 = X_{su}(S_2)$**:**Autumn time (au). There is a forest with trees t, burned trees T, or houses h or burned houses H.

8. Change rule X_{au}: If there is a position with a *burned tree T* or a *burned house H* then remove these from this forest. Then *count* the number of remaining trees t as the *production* $S(t)$ of the forest for that year.

9. $S_4 = X_{au}(S_3)$**:** Winter time: There is a forest with no burned trees T nor burned houses H.

10. Logical decision: if there is still a human settlement h in the forest then continue with $S_1 = S_4$, otherwise stop. In this case the human settlement has been destroyed. A damage happened.

In this version of the actor story there exists no explicit actor. Therefore it is not necessary to define explicitly an actor. Thus in this version it makes no sense to speak of an agent-based approach.

3.3 Simulation

Simulation is here understood as the generation of a sequence of states based on the change rules. Without a computer this has to be done by human experts using *pencil and paper* to compute possible changes, or by teams of experts by playing a *simulation game*. This kind of a *computer-free simulation* is with real cases really limited. Therefore one needs an appropriate automaton which can *read* the symbolic descriptions of an actor story and which can *compute by itself* the possible *follow-up states*. Such an automaton is called a *simulator*. If roles are defined based on explicit actors then the simulator can *read additionally input* from these actors in real time to compute the next state. This is called *interactive* simulation compared to a *passive* simulation.[5] An interactive simulation makes sense where an actor in a state has more than one option to behave. Instead of computing a possible action with the aid of an artificial model or even only by some statistics it is more realistic to ask real persons what they are inclined to do in a certain situation (state). For the participating persons this can trigger some emotions, some ideas, and calls for decisions. This can be used for *training*

[5]The full DAAI paradigm has some more options beyond simulation.

as well as for *learning* or even only for *fun*, or all these factors together.

3.4 Results

Doing the simulations it becomes clear, that this experiment can produce the case where a human settlement can become damaged. Putting occurrences and probabilities aside it is clear, that the only strong protection against burning a human settlement is a sufficient distance to trees.

If one is looking to those countries with frequent forest fires one can observe the fact, that humans still are building there houses in direct neighborhood with trees.

Although the risk of burning can be *known* there seem to be *other factors* which influence the behavior.

3.5 Testing NFRs

As it becomes clear from this example, the NFR of requiring a safety with probability 1 can be checked (and proved) by checking under which conditions (forest, hs, g,f) an actor story can be generated where a the human settlement will not be damaged.

4 Coding in Python

In the next text this case study extended by using the python language is described. The selection of python instead of another programming language is unimportant. But to give some impression how this whole concept works a concrete case study with python will be done (later also embedded in a python based web framework on a real server). To read this go to the next document, which will be placed in the python-co-learning section on the uffmm.org site with the URL `https://www.uffmm.org/2019/04/01/co-learning-with-python-3/`.

References

[MP07] John H. Miller and Scott E. Page. *Complex Adaptive Systems. Introduction to Computational Models of Social Life*. Princeton University Press, Princeton - Oxford, 1 edition, 2007.