SPECIAL ISSUE

Teaching Modeling in Computer Science as an Ecosystem: a provocative analogy

Martin Gogolla[a] and Perdita Stevens[b]

[a]University of Bremen, Germany; [b]University of Edinburgh, United Kingdom.

ARTICLE HISTORY
Compiled April 4, 2018

ABSTRACT
Teaching modeling in computer science is complicated. Many factors contribute, and are related in diverse ways. We regard some combinations as more successful than others, but we also value diversity, and we struggle to elucidate the relationships and our value structure. Similar remarks apply to the study of biological ecosystems. This contribution views teaching modeling as an ecosystem: we push the analogy as far as we can, with the intention of provoking readers' thought in unaccustomed directions, which may perhaps lead to interesting advances. We identify the factors that contribute to the ecosystem and establish some relationships between the constituent factors. We discuss some of the factors that need further work and improvement. Finally we discuss what it should mean for the ecosystem of the teaching of modeling to be healthy.

KEYWORDS
Computer science education; teaching modeling; ecosystems

1. Introduction

According to the current consensus of Wikipedia editors (WikiTeam (2018)) an ecosystem is characterized as follows:

> An ecosystem is a community of living organisms (e.g. animals, plants) in conjunction with nonliving components of the environment (e.g. water, air, soil) interacting as a system. In an ecosystem there are biotic factors as well as abiotic factors. An ecosystem is self supporting, and the components are linked through nutrient cycles and energy flows. It is a network of interactions among organisms, and between organisms and their environment that can be of any size, but usually encompasses limited space.

Our view of teaching modeling is that it, too, can be seen as an ecosystem. Of course, this is intended as a provocative analogy, not as a claim to be taken literally: yet we find it a useful one for organizing our thoughts about progress that has been made in recent years in understanding how to teach modeling, and about what needs to be done in future.

Let us take up the above definition and connect the notions to the context of teaching modeling. The teaching modeling ecosystem is a community of living and developing organisms [like Teachers, Students, Courses, Curricula] in conjunction with the

CONTACT Martin Gogolla. Email: gogolla@informatik.uni-bremen.de

nonliving components of their environment [like Buildings, Computers, Laws, Society requirements], interacting as a system. (Already we see the role of humans in classifying and abstracting: just as biological entities may not fall neatly into living and nonliving categories – is soil biotic or abiotic? – neither do entities relevant to teaching; neverthless, thinking about the distinction can be instructive.) These components are regarded as linked together through cycles and flows between them [like Know-how flow, Cash and Material flow, exchange between educating institutions and consuming companies or authorities]. Ecosystems are influenced both by external and internal factors. External factors [like the Environment with legal, organizational, or financial regulations] influence the overall structure of an ecosystem and the way things work within it, but are not themselves influenced by the ecosystem. Internal factors [like Teaching content, Teaching medium, Teaching demonstrations, Teaching actors, Teaching timing, Teaching form, Teaching style] not only influence ecosystem processes but are also influenced by them and are often subject to feedback loops. Our summary of an abstract ecosystem, of a biological ecosystem, and of the teaching modeling ecosystem that we will discuss is shown in Fig. 1.

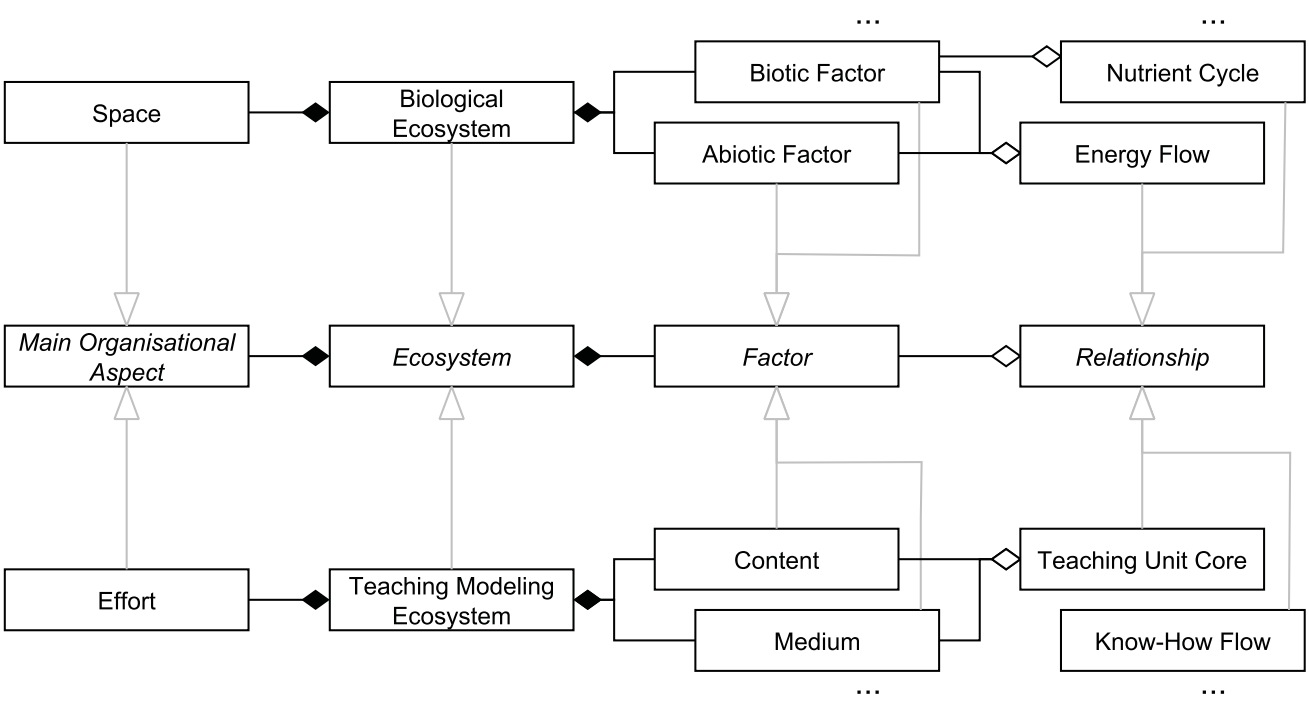

Figure 1. Biological ecosystem - Ecosystem - Teaching modeling ecosystem.

Before we go on we should remark that much of what we say in this paper applies more broadly than the teaching of modeling: many aspects of it probably apply to teaching, especially university teaching, more generally. Modeling education is what we know, however, so we will leave it to readers to determine where there is wider applicability.

This paper originated in the invited talk of the first author at EduSymp 2017, expanded through subsequent discussion between the authors. The paper is structured as follows. In Section 2 we set the scene for our analogical exercise and discuss the relevance of modeling. In Section 3 we set out a candidate abstraction of the teaching modeling ecosystem, by means of a set of nine factors; we discuss the factors in turn, and point at some selected papers from the literature. In Section 4 we turn to the

interactions between these factors. Section 5 is perhaps the most opinionated section of the paper, and the one most open to disagreement from readers; in it we pick out a few of the factors we have identified, as being in particular need of greater attention from the community of those interested in educating modelers. Since this brings into the open the value system capturing what makes a "good" ecosystem, in Section 6 we make this explicit by discussing what it means for an ecosystem to be healthy. Finally, Section 7 concludes.

2. Why Model?

We are writing for a community of people who educate students about modeling in computer science. Yet, as some of these readers will be aware, modeling is also a hot topic in the study of biological ecosystems. Indeed, it is arguable that modeling is the point of studying ecosystems as such: when one thinks about an ecosystem as a whole, rather than about some particular organism within it, it is usually because one is interested in how the ecosystem will, might or might have develop(ed) at some other point in time, and such things are only discussable via models, whether hypothetical or built from gathered data. The models provide the vocabulary for the discussion. However, these communities mean very different things by the term "modeling".

In computer science, or at least in the EduSymp community, we are most often concerned with prescriptive modeling. We model, and teach students to model, aspects of software systems, and when we place a fact – such as the existence of a class – in a model, we normally mean that placement to represent a decision that such a class should exist. Even when we reverse engineer a model from an existing codebase, there is still an expectation that the model will be used to control the codebase in future, not (merely) to predict how it will change.

Modeling of ecosystems is most often descriptive. The ecosystem is as it is; in order to understand its past, and to predict its future, the ecologist may build a model. The model may be executable, and may be run in order to produce an informed guess about what may happen next. The results of running the model may serve to focus minds on what should be done to ensure an outcome that is considered desirable. Moreover, comparing that prediction with what actually happens may lead to important insight concerning the strengths and weaknesses of the scientist's current understanding of the ecosystem; this may help to tease out poorly understood factors or relationships.

This teasing out of poorly understood factors or relationships is our aim in thinking about the teaching of modeling as an ecosystem. For the act of modeling – even, the act of building a descriptive model – is a creative one. It embodies a set of decisions about what is important, and about what things are similar. The descriptive modeler of an ecosystem cannot model everything she knows about the ecosystem: abstraction is required. The choice of abstraction matters in how the ecosystem is thought about, and ultimately in how decisions are made about it. For example, whether bacteria are neglected as being insignificant, or are made important examples of a class of organisms classified as "decomposers" whose effect on the rest of the ecosystem is then considered, may materially affect the accuracy of certain predictions.

In the next section we will introduce a candidate for such an abstraction in the ecosystem of teaching modeling: we will introduce a set of factors that we consider to be important, including analogues of the living organisms of an ecosystem but also of many aspects and influencers of them. The reader's preferred abstraction might be different: the point is that thinking about what the factors should be is itself important,

as it frames the subsequent discussion.

3. Factors in the Teaching Modeling Ecosystem

Let us first turn to the factors, and then to the interplay of these factors. As we suggested above, this list is not intended to be definitive, and different lists of factors are possible: this list of nine is simply one, we hope useful, way of imposing structure on the messy reality of all the teaching we do. (We considered, and still use on occasion, the term *dimension* for what we here term factor: the connotation that a real life teaching session might be projected onto many of the dimensions or factors is interestingly suggestive, but since that does not make sense in every instance, we generally prefer the term *factor*.) We list the factors and give single examples, before explaining each in more detail and connecting to the literature.

(1) Teaching *content* which has three subfactors.
   (a) The modeling *technique* being taught. Example: use case modeling.
   (b) The model *qualities* being taught. Example: abstraction.
   (c) The modeling *style* being used. Example: a domain-specific modeling style.
(2) Teaching *medium* used. Example: Excel.
(3) The *demonstrations* employed for teaching. Example: a particular industrial case study.
(4) The *actors* involved in teaching. Example: a tutor.
(5) Teaching *timing*. Example: teaching undergraduates in the third year.
(6) Teaching *form*. Example: a student project with 10 ECTS points.
(7) Teaching *style*. Example: a style involving gamification.
(8) Related areas of Computer Science *areas* that are involved. Example: artificial intelligence.
(9) Teaching *environment*. Example: legal requirements.

In modeling ecosystems, an important thing distinguishing some models from others is that some models take account of space while others do not. Taking account of space is important because it allows ecologists to notice, for example, that the reproduction of a species may be limited, even in the presence of plenty of energy, nourishment etc., by the necessity for there to be physical space for the new members of the species to be. Moreover, important relationships may be affected by the spatial placement of things in the ecosystem; for example, there may be plenty of sunlight available, but an organism's spatial location under others of its kind may prevent it from benefiting from the sunlight.

What is the analogue, in our teaching modeling ecosystem, of space? We propose that it is *effort*. As with space for biology, this is a fundamentally important factor which is limited for all members of the ecosystem, although it affects different members in different ways; as with space for biology, we are often tempted to abstract it away, and behave as though it did not matter, but fundamental inaccuracies result from doing so. For example, a student who has prerequisites for many courses, and has many courses available to them, might in a naive model take all the courses of interest, with their learning from each one evaluated independently. However, in practice, the student's available effort is limited. Too many courses will result in too little understanding of each; ultimately this will dwarf other factors such as the teaching style, examples and tutor input in its effect on the student's learning. Similarly, although a successful course may expand year on year as it attracts more and more

students, the leading staff member's effort is limited. As the course attracts more students, each student will necessarily receive less of the leading staff member's attention: however successfully the factors we have identified were initially combined, changes will ultimately be necessary.

3.1. The Factors in More Detail

Let us go through the nine factors one by one and explain them in some more detail.

3.1.1. Content

Teaching *content* is key, of course; we have suggested that it be structured into techniques, qualities and style, though other factorings are possible.

There are many modeling *techniques* like structural or behavioral modeling, whose results may be embodied in class diagrams or state charts; techniques taught may involve model organisation such as handling of profiles, or managing relationships between model relationships, perhaps using model transformations. We attempt here to make a distinction between the techniques and the specific languages and tools that support them – but the distinction is, admittedly, a nice one. This is the *what* of modeling: what is it that needs to be recorded?

Model *qualities* are captured by general, admittedly potentially vague, notions like classification, abstraction, structuredness, appropriateness, clearness, and understandability (of a model) as well as by terms like verifiability, testability and executability. The key point is that it matters *how* some information about a system is captured, not merely *that* it is captured: some models are more useful than others for some purposes, and it is important that students appreciate this.

Modeling *styles* can be caught by contrastive pairs as *textual* ... *graphical*, *universal* ... *domain-specific*, or *informal* ... *formal*. Some styles support some qualities in some techniques better than others, but often the choice is driven as much by the software engineering context as by anything specific to the modeling itself.

3.1.2. Medium

The teaching *medium* is the term we use for the specific material we give students to model with. That is, it covers both the language we expect them to use, and the way in which we expect them to express models in that language. It may be a combination of a modeling language used in a particular tool like UML (Rumbaugh, Jacobson, and Booch (2005)) or OCL (Warmer and Kleppe (2003)) in MagicDraw, USE (Gogolla, Hilken, and Doan (2017)), or Umple (Lethbridge, Abdelzad, Orabi, Orabi, and Adesina (2016)). It may be something specially adapted to some purpose such as verification, e.g. EMFtoCSP (González, Büttner, Clarisó, and Cabot (2012)). It may be a transformation language such as ATL (Jouault, Allilaire, Bézivin, and Kurtev (2008)); there are countless other possibilities. The teaching medium may also be some non-mainstream modeling tool like Excel, PowerPoint, or yEd[1], or the medium could be a programming or database language like Java or SQL. Indeed a case has been made for avoiding the use of modeling-specific tools, in favour of general purpose tools (Batory and Azanza (2017)). The medium may even be a blackboard, a sheet of paper or a video (Brandsteidl, Mayerhofer, Seidl, and Huemer (2012)). Within computer

[1] `https://www.yworks.com/products/yed`

science modeling, UML is currently quite a dominant approach for development and teaching. From the ecosystem point of view, one has to be careful that UML does not become an invasive species that takes over a delicately balanced system and crowds out other species.

This aspect of modeling education has had a great deal of attention in the literature; people who develop languages or tools are naturally concerned to evaluate their teachability, while teachers often have to defend their choice of medium rather than their choice of content or other factors. We do not pretend to give a literature survey here; a high proportion of the modeling education papers concern the medium of modeling education, one way or another. Akayama et al. (2013) was an attempt by the second author and others to summarise the use of tools in modeling education.

3.1.3. Demonstrations

Demonstrations in form of examples or case studies are generally considered essential for teaching. What kind of example or case study is useful depends on the other factors. Most obviously, an example of a model transformation will be inadequate for teaching state charts, and vice versa; but equally, the demonstration must be suitable for use by the available actors, at the appropriate time, etc. Aspects of demonstrations may be labeled using points on spectrums such as *tiny* ... *huge*, *vague* ... *detailed*, *concrete* ... *abstract*, *positive* (i.e. always to be emulated) ... *negative* (never to be emulated), *academic* ... *industrial*. A classification of demonstrations and examples along various criteria has proposed by Gogolla and Vallecillo (2012).

3.1.4. Actors

Actors in teaching may be students (beneficiary actors of the teaching process) or teacher, lecturer, tutor etc. (actors making the teaching process happen). Looking further afield, we may sometimes find it helpful to consider project roles such as programmer or designer and also more organisational positions such as examination administrator, technician or curriculum dean.

3.1.5. Timing

Timing in teaching classically decides on whether a teaching activity concerns undergraduate, graduate, or PhD students, but one could also consider education for professionals in order to improve skills needed in the job (Ratiu, Pech, and Dummann (2017)); teaching modeling to professionals is currently under-represented in the teaching modeling literature, including the proceedings of the MODELS Educators Symposium. The authors, between them, have experience of teaching modeling at a variety of stages from second-year undergraduate to professional developers, and the second author has recently experienced teaching some aspects of modeling to ten-year-olds. Naturally the timing of the teaching affects many other factors, because of the prior knowledge and the needs of the students concerned, besides having indirect effects via the amount of time available, the form of assessment expected, etc.

3.1.6. Form

As the teaching *form* one can distinguish between lectures, exercises, courses, seminars, homeworks, projects, and various forms of theses in Bachelor, Master, and PhD curricula. Regarding PhD students one may even consider a research paper as a teach-

ing form trying to transport methods for developing ideas and presentations from a teacher to a student.

3.1.7. Style

The teaching *style* may vary from a classical teacher-in-front option to group-work-oriented styles. Contrastive pairs as *traditional* versus *gamified* or *research-oriented* versus *trainee-oriented* styles may also be considered. A research-oriented style views teaching in the first place as representing research results. A trainee-oriented style looks at teaching from the perspective of the trainee and tries to satisfy the trainee's needs respecting and taking into account the trainee's abilities in the first place.

3.1.8. Area

Teaching modeling does not only involve the software engineering realm, but many other Computer Science *areas* teach modeling in some form, e.g. models are used in teaching programming languages, information systems, networks, theoretical computer science, or in artificial intelligence. Sometimes modeling is taught as a small part of some other teaching objective; sometimes the objective is to teach modeling, but the cohort of students is known to have an intention for their modeling which should be taken into account for maximum engagement.

3.1.9. Environment

Teaching always takes place in a certain *environment* in which a curriculum-performing institution is located. There are always related curricula and related institutions at the teaching institution and in the geographical neighborhood. The present teaching colleagues and laws and norms expressing curricula frame requirements also have a serious influence. For example, some departments' curricula are influenced by the ACM recommendations such as SE2014.[2]

4. Relationships between Teaching Modeling Factors and Feedback in the Ecosystem

4.1. Relationships

In the process of introducing the factors we have begun to point out that they interact. In this section we draw attention to two ways in which this happens. One, which we have already begun to see, is value-based: a "good" factor is not good independently in all circumstances, but rather, works well in a given assembly of factors. The other is that the factors may influence one another over time. Both have their biological counterparts: an organism is not "well-adapted" in isolation, but rather, in its place in a given ecosystem; and the elements of an ecosystem influence one another and the balance of the ecosystem over time.

Figure 2 graphically places the discussed factors and dimensions on a circle that allows us to establish relationship connections in the inner part of the circle, to link a single relationship with many factors.

The figure illustrates two relationship connections. There are, of course, many others, and we suggest future work to explore what the most important relationships

[2] `https://www.acm.org/education/curricula-recommendations`

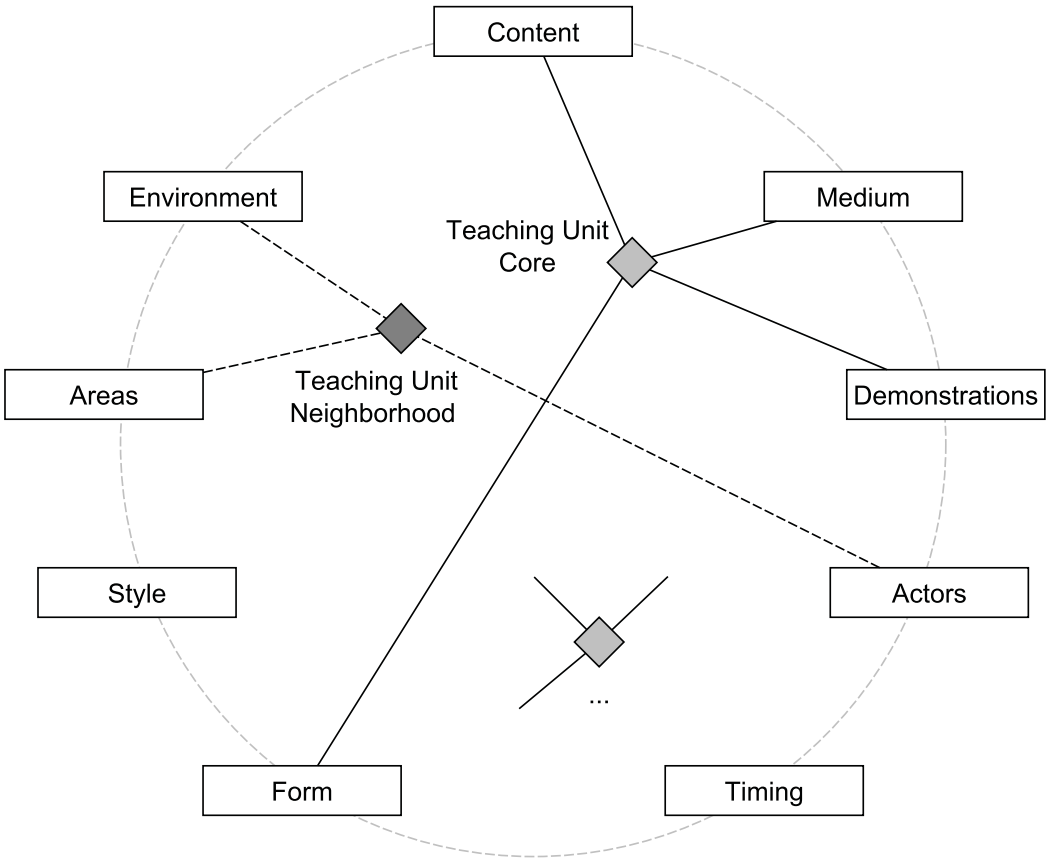

Figure 2. Factors and dimensions of the teaching modeling ecosystem.

are.

The two relationships shown are Teaching-Unit-Core and Teaching-Unit-Neighborhood. Teaching-Unit-Core connects content, medium, demonstrations, and form. It establishes essentials of what makes up a traditional teaching unit. Teaching-Unit-Neighborhood connects actors, areas, and environment. It brings together the neighborhood of a teaching unit, in particular it puts emphasis on the fact that teaching actors are connected to related Computer Science areas.

A possible 'know-how flow' relationship (similar to the energy flow in a natural, biotic ecosystem) that could be added to the figure is a reflexive relationship on actors because a student can become a PhD student who can become a researcher who can become a lecturer teaching again to a student.

The relationships between factors have been discussed in the teaching modeling ecosystem with various focus points. For example, Balaban (2014) pointed to an important connection between actors (students) and content (model nature). The claim was that students would not use models unless enforced to do so, and that models were considered kind of some annoying documents that accompany software and could be written as an afterthought. France (2011) targeted in particular students having a strong background in programming. The work distinguishes between so called basic modeling-in-the-small (MITS) skills, and more advanced modeling-in-the-large (MITL) skills. MITS focuses on program design, whereas MITL turns to modeling larger systems with problematic requirements having architectures consisting of non-trivial subsystems. To support effective development of MITS and MITL skills, particular modeling content, modeling examples and modeling tools would help: (1) modeling patterns and anti-patterns that distill expert modeling experience, (2) a repository of models that illustrate good and bad modeling practices, (3) text books that focus on developing modeling skills rather than on covering syntactic and semantic language

concepts, and (4) lightweight modeling tools that tolerate incompleteness and support exploratory design.

Next, we will look at change over time involving several factors and potential feedback effects.

4.2. Feedback in the Teaching Modeling Ecosystem

In simple cases, we may look at an individual course or other teaching engagement and regard it as a static configuration, an occurrence of certain choices on each of the dimensions we have identified: it is taught by some professor and tutors, taken by some students, covers some material, uses some teaching style, etc.

However, just as the populations of species in an ecosystem are not fixed for all time, neither are these configurations. The characteristics of a course influence one another and vary over time. For example, a course may be well received, and hence attract more students to take its next instantiation. This may, in turn, influence how the course can be delivered, which in turn affects its popularity. In a simplified example, we might imagine that the course has an optimal number of students, deviations from which will cause the course to be less well perceived by those students who take it. The current year's students report on their experience via course questionnaires, and their scores influence, to some degree, the students who consider taking the course the following year. We may model this as follows:

$$\begin{aligned} \text{numStudentsOnCourse}_n &= \text{totalNumStudents}_n \times \text{coursePopularity}_{n-1} \\ \text{coursePopularity}_n &= \text{basePopularity} - \\ &\quad |\text{optimalSize} - \text{numStudentsOnCourse}_n| \times \text{effectOfWrongSize} \end{aligned}$$

Suppose we start with a situation where `totalNumStudents` is 100, and a course with an optimal size of 33 students has `basePopularity` of 33%, meaning that each student chooses this course with that probability, so that the expected size of the course is optimal, and its popularity remains constant over time. Everything is stable. However, one year, for whatever reason, `totalNumStudents` increases to 150 (and stays at that level thereafter). For the sake of argument take `effectOfWrongSize` to be 1%; that is, suppose that each student by which the course size is above or below the optimal detrimentally affects the course so as to make it slightly less popular among the year's students, subsequently reducing the probability of choosing the course by 1% for each student in the following year. We then get a wild oscillation of course size, as shown in Figure 3. Imagine the lecturer realises in the first year that it is a problem that the course has too many students, and plans to adjust the course and its delivery to account for its increased size. But in year 2, just as she is planning to deliver a version of the course that is suitable for 50 students, she realises she only has 25 this year – now what should she do?

This over-simplified model was bad enough. In practice, of course, things are not so simple, and the jobs of people trying to maintain healthy universities may not seem much easier than the jobs of those trying to keep biological ecosystems healthy. Some readers may have experienced course histories similar to one that the second author experienced over some years in Edinburgh. She took over an existing undergraduate design and modeling course, also available to MSc students, which was in need of modernisation and revitalisation. In the first year of the new version of the course,

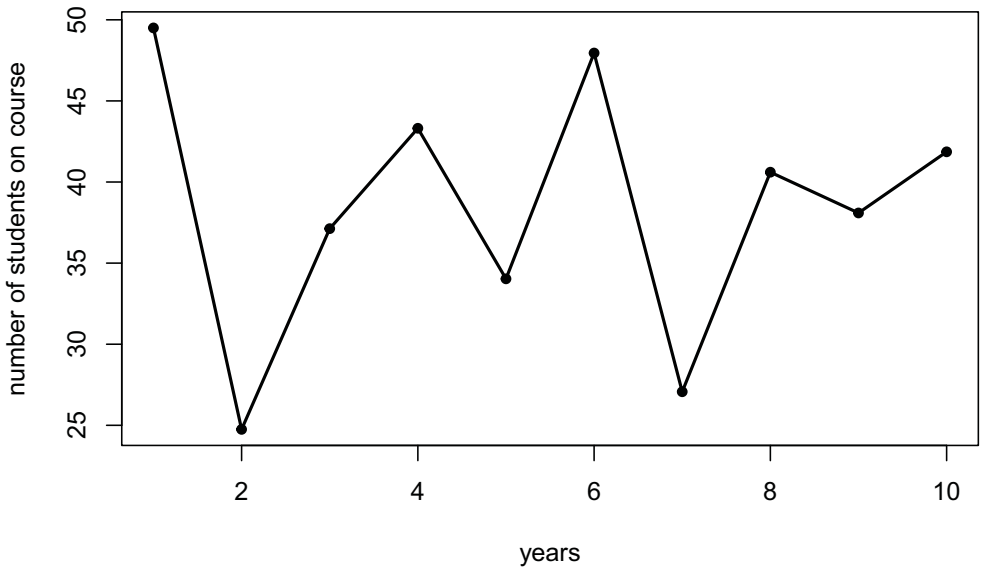

Figure 3. Simplified example showing ongoing effect of change of student numbers.

it attracted 56 students, consistent with the steady state of the previous few years. The following year enrollment was slightly lower at 43; this may have been random variation, or may have been because the total number of MSc students was lower that year; we might hypothesise that the course had temporarily acquired a poor reputation because of rough edges in the rewriting, but in fact the feedback from the students who took the rewritten course was largely positive. This illustrates the difficulty inherent in studying university teaching (or indeed software development) – there are too many factors that vary to be sure about what the causal relations are, and building a descriptive model (whether explicitly, or merely implicitly by what one assumes) risks over-fitting. The following year, the numbers enrolled on the course exploded to 90, causing unplanned changes to several aspects of the course. For example, its delivery style had been partially "flipped": students were expected to do reading and watch videos before the sessions, which were devoted to discussion and exercises, taking advantage of the course's placement in a room with movable furniture to allow students to work in groups. This style was threatened when the numbers exploded and the sessions had to be moved to a raked lecture theatre; group work was more difficult in this setting, and whole-class discussion was less successful, while extra plenary sessions were needed to ensure that every group had access to key findings from the exercises. Another example is that the course had been delivered with weekly small group tutorials; this aspect had to be abandoned when there were too few tutors available for the number of enrolled students. At the same time other elements of delivery, such as automated self-assessment questions based on the reading and video material, were in the ascendancy. All these changes, in their turn, probably affected what students with different preferences, interests and learning styles gained what from the course, both in terms of enjoyment and learning; we might expect the following year's enrollment to be lower, but in fact it rose again to 104 (giving, at least, the advantage of relative stability). At each stage, staff involved had to guess what was driving the changes in numbers and what might affect them in future and try to plan for them in the face of uncertainty. (In the longer term, this course was itself replaced, and the design of the

replacement drew on all this experience, while also being influenced by policy decisions at university level, such as a change in the timing of examinations, and a preference for fewer longer courses.)

Given the difficulty of isolating relevant factors so as to study sufficiently large collections of comparable situations, we are unlikely ever to be able to develop and validate mathematical models of teaching modeling. Even in ecology, with vast amounts of data to draw on, such models are disputed and disputes are difficult to settle because it is hard for experts to agree on an appropriate evidence base and methodology for testing the models Arditi and Ginzburg (2012). Nevertheless, paying conscious attention to the web of factors that affect teaching may benefit us as academics and other stakeholders.

5. Factors Needing Attention and Improvement

We here postulate a working thesis that there are currently six factors that need special attention and improvement: Content (in particular model qualities), demonstrations, actors, timing, areas, and environment.

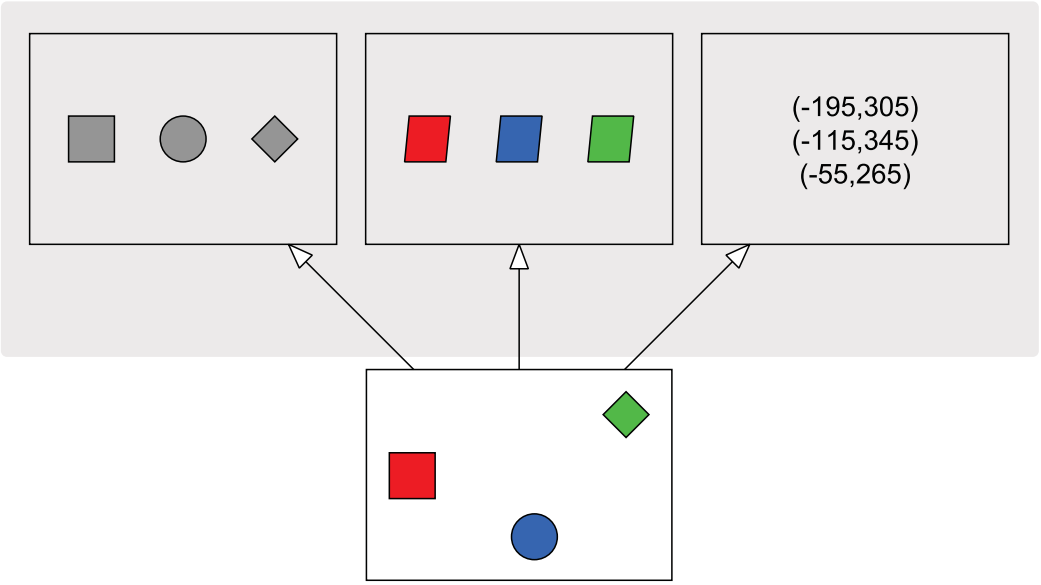

Figure 4. Attempting to exemplify the process for finding abstractions.

The first factor needing improvement is 'Content'. Content is about 'Model qualities and characteristics' concentrating on notions like abstraction, classification, executability, verifiability, and appropriateness. We think that modeling currently underestimates the role of 'traditional' modeling languages like flowcharts, automata, ER modeling, first order predicate calculus, or graphs for teaching model qualities such as abstraction. The modeling community puts the emphasis on 'modern' modeling languages (e.g. UML, OCL, SysML, EMF, ATL, QVT). And the focus is on teaching techniques, not on teaching model qualities like abstraction or classification. It is much easier to teach techniques than to teach qualities. A quality is something that you cannot take into your hands. Figure 4 is an attempt to explain what teaching 'abstraction' basically looks like. The lower part shows the system that needs to be handled and of which an abstraction is needed. The upper part shows different ways this abstraction can be done: by shape, by color, by coordinate. However in the moment a teacher presents these abstracting solution options, the teacher has done already the creative

part of the abstraction process, and the students cannot look into the teacher's head to see how this process of finding the abstractions was done. Finding abstractions is hard to teach! (Engels, Hausmann, Lohmann, and Sauer (2005); Simonot, Homps, and Bonnot (2012))

The second factor needing more explicit attention is 'Demonstrations'. For example, good demonstrations for abstract classes (in the technical sense) covering structure (attributes and operations) as well as behavior (e.g. operation contracts and protocol state machines) could help with teaching the principles of abstraction, which, as we pointed out above, is difficult. Furthermore, demonstration cases at different development stages and refinement levels should not serve only for presentation of perfect, non-developing models. The discusssion of development steps and their rationales, by presenting models at various levels of detail, could benefit teaching modeling. Also negative examples that show how students should *not* model have their place (Paige et al. (2014)).

The third factor needing improvement is 'Actors'. The typical leader of a university modeling course is someone involved in both teaching and research in modeling; such a person's teaching is naturally influenced by their research perspective, which may lead to a preference for presenting the newest results and tools, with a focus on their own research interests. This is not a bad thing; students who attend research-led universities are expecting to get such perspectives, and the cross-influence of research on teaching and vice versa is part of the raison d'être of such universities. Nevertheless, this pattern is not without issues. There is a place for more teaching-focused professionals who present material from (neutral, not self-authored) textbooks in a straightforward way. For example, while exposing students to the controversies and advances in modeling is valuable, it can sometimes be confusing or inappropriate; sometimes it is better to use trainee-oriented teaching methods that help students to apply well-explored techniques in a way they understand and that are in line with their present abilities. It is not that such teaching does not happen – of course, a lot of it does, both inside universities and commercially – but it is underrepresented in the literature on teaching modeling, because it is seldom done by people who have an interest in adding to this literature.

The fourth factor needing improvement or at least discussion is 'Timing'. To bring up a very simple question: could modeling be taught before programming is taught? Modeling comes in curricula quite late, and thus students often use modeling techniques as a way of 'documenting the code'. So what would happen if we ban programming from the first year of Computer Science education and teach modeling instead?

The fifth factor needing improvement is 'Areas'. As mentioned above, modeling is done in many, if not all areas of Computer Science, and not only in software engineering. The identification of confederate Computer Science areas who have interest in modeling and their integration and establishing a connection to our field is worthwhile. The modeling community may sometimes appear, to those outside it, to re-invent the wheel. Identification of partners and confederates and relating 'their' modeling to our view on modeling would be a fruitful task.

The sixth factor needing improvement is 'Environment'. This covers the frameworks of laws, expectations and norms in which the teaching of modeling takes place. What is expected to be delivered in a teaching engagement, and what is each actor expected to bring to it? As part of the environment one can regard the MODELS conference and its internal structuring. The MODELS Educators Symposium and the MODELS Tutorials are separated, non-communicating events at MODELS. There is no common strategy or plan for these events. Tutorials could be a place for MODEL education and teaching,

with concepts being developed hand in hand from both MODELS events. For example, 'bridge' tutorials lying thematically between other Computer Science areas and core modeling techniques could establish connections. One could also discuss the possibility of systematically offering basic modeling tutorials, not only specialized tutorials on newest research directions and tools. (These would need to be appropriately marketed, of course, as their primary audience would not be the researchers who attend the MODELS conference, even though some of these might also be attracted.)

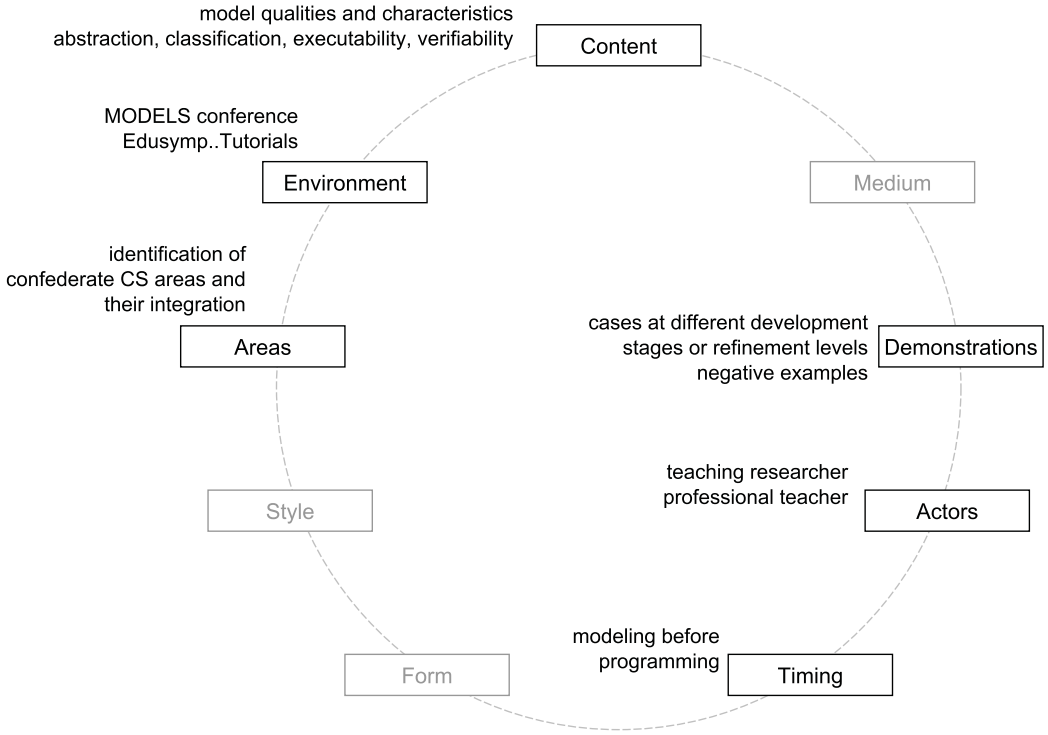

Figure 5. Teaching factors to be improved.

Figure 5 displays the six factors that in our view need improvement and surveys the items we have suggested working on. Readers may well have their own opinions on what else deserves attention.

6. Health of the Ecosystem

When considering biological ecosystems, we have a notion of an ecosystem being healthy. For example, in the context of aquatic ecology, Costanza and Mageau (1999) proposes that: "a healthy ecosystem is one that is sustainable – that is, it has the ability to maintain its structure (organization) and function (vigor) over time in the face of external stress (resilience)".

Let us unpack this and consider the teaching modeling analogues.

In reaching the above definition of a healthy ecosystem, Costanza and Mageau discuss earlier definitions of health in ecosystems, which they describe as being "pieces of the puzzle" but not sufficient in themselves. They explain that health has been regarded as:

- "homeostasis": the idea that a healthy ecosystem is in steady state, and will return to that state after a perturbation. We might say, for example, that if one modeling tool stops being available, another will take its place. In Section 4.2 we described an example of a failure of homeostasis in the teaching modeling

ecosystem.

- "the absence of disease": this one is somewhat tricky to interpret in our setting! What is a disease, in the teaching modeling ecosystem? We might find it thought-provoking to regard dissatisfaction among any of the major stakeholders, such as students, teachers and employers, as a disease, for example. The nature of the disease might involve any of the factors we considered earlier; such dissatisfaction might be caused by problems with the available modeling tools, the regulatory requirements on a degree course, the availability of tutors, the prior knowledge of the students, etc. Indeed, it might arise out of *relationships* in the ecosystem, even where no one factor is problematic. For example, we might say that there is a disease if a course which aims to teach students to produce models having a certain good quality does not effectively do so because the choice of teaching style and material is not appropriate.

  Additionally, we might look at economic factors, and see an organization that spends more money than it brings in as diseased. Here, as in the biological setting discussed by Costanza and Mageau (1999), we see an overlap between this definition and the previous one: certain kinds of instability are incompatible with health.
- "diversity or complexity": we will discuss this below under organization.
- "stability or resilience": see below.
- "vigor or scope for growth": see below.
- "balance between system components": this can be seen as a kind of organization; it is interesting in our setting as suggesting the idea that it is not healthy for any one language, course or book to be universally taught.

The vigor of a system is, according to Costanza and Mageau, "simply a measure of its activity, metabolism or primary productivity". Expanding the definition beyond their primary field of concern, they give Gross National Product in an economic system as an example. The metabolism of the teaching modeling ecosystem is the throughput that changes the "living" components of the ecosystem: teachers, students, courses and curricula. Thus an ecosystem will be more vigorous if it educates more students, or changes the students it does educate more profoundly; if it involves more teachers, or involves them more deeply; if its courses and curricula change over time. This seems a reasonable match for what we might intuitively want to mean by "vigor" in this context.

The organization of a system is "the number and diversity of interactions between the components of the system". A system is considered more highly organized if it contains a great diversity of different organisms, and specialized interactions between them. Why is this considered a good thing? Partly it is axiomatic – diversity is considered a good thing in itself – partly it is because it generally gives the ecosystem a better chance of adjusting to changes in the environment such as climate change. In our setting, it corresponds to a widespread teaching of modeling in many different contexts: we not only teach UML in a single course to computer science students, but teach mathematical modeling to engineers, use domain-specific languages in different settings, have specialized courses on model transformations, etc.

Resilience of a system means "its ability to maintain its structure and pattern of behavior in the presence of stress". A highly resilient ecosystem can maintain its level of organization and its vigor in the presence of small perturbations in the environment. For example, our teaching modeling ecosystem can survive the demise of any one modeling tool or even language.

7. Conclusions

This contribution has explored the potential analogy between the teaching of modeling and a biological ecosystem, as a way to structure discussion and provoke thought about the former. Suggesting an organisation of the features that we need to consider in teaching modeling, we identified the factors that may be said to make up a 'teaching modeling' ecosystem. This enables the discussion of relationships between these constituent factors, so that we can, for example, discuss the way in which aspects of a course may support or interfere with one another; we also pointed out that changes over time may be caused by such interactions in ways which are important but hard to predict. Much more work on identifying the relationships is needed. We have discussed some of the factors that, we consider, would repay further work and improvement. Finally, we looked at what ecologists mean by health in an ecosystem, and discussed the analogues of this notion for the teaching of modeling.

References

Akayama, S., Demuth, B., Lethbridge, T. C., Scholz, M., Stevens, P., & Stikkolorum, D. R. (2013). Tool use in software modelling education. In T. C. Lethbridge & P. Stevens (Eds.), *Proc. Educators' Symposium at 16th MODELS.* (Vol. 1134). CEUR-WS.org.

Arditi, R., & Ginzburg, L. (2012). *How species interact: Altering the standard view on trophic ecology.* Oxford University Press.

Balaban, M. (2014). Symbolic representation of models improves model understanding and tendency to use models. In B. Demuth & D. R. Stikkolorum (Eds.), *Proc. Educators' Symposium at MODELS.* (Vol. 1346, pp. 72–75). CEUR-WS.org.

Batory, D. S., & Azanza, M. (2017). Teaching model-driven engineering from a relational database perspective. *Software and System Modeling*, *16*(2), 443–467.

Brandsteidl, M., Mayerhofer, T., Seidl, M., & Huemer, C. (2012). Replacing traditional classroom lectures with lecture videos: An experience report. In *Proc. 8th Educators' at MODELS* (pp. 21–27). ACM.

Costanza, R., & Mageau, M. (1999, Mar 01). What is a healthy ecosystem? *Aquatic Ecology*, *33*(1), 105–115.

Demuth, B., & Stikkolorum, D. R. (Eds.). (2014). *Proc. Educators' Symposium at MODELS.* (Vol. 1346). CEUR-WS.org.

Engels, G., Hausmann, J. H., Lohmann, M., & Sauer, S. (2005). Teaching UML is teaching software engineering is teaching abstraction. In J. Bruel (Ed.), *Satellite Events at MODELS* (Vol. 3844, pp. 306–319). Springer.

France, R. B. (2011). Teaching programming students how to model: Challenges & opportunities. *ECEASST, 52, Proc. Educators' Symposium at MODELS*.

Gogolla, M., Hilken, F., & Doan, K.-H. (2017). Achieving Model Quality through Model Validation, Verification and Exploration. *Journal on Computer Languages, Systems and Structures, Elsevier, NL.* (Online 2017-12-02)

Gogolla, M., & Vallecillo, A. (2012). On Explaining Modeling Principles with Modeling Examples: A Classification Catalog. In D. Chiorean & B. Combemale (Eds.), *Proc. 8th MODELS Educators' Symposium (EduSymp 2012)* (p. 28-31). ACM Digital Library.

González, C. A., Büttner, F., Clarisó, R., & Cabot, J. (2012). EMFtoCSP: A Tool for the Lightweight Verification of EMF Models. In *Proc. 1st Int. Workshop Formal Methods in Software Engineering* (pp. 44–50).

Jouault, F., Allilaire, F., Bézivin, J., & Kurtev, I. (2008). ATL: A model transformation tool. *Sci. Comput. Program.*, *72*(1-2), 31–39.

Lethbridge, T. C., Abdelzad, V., Orabi, M. H., Orabi, A. H., & Adesina, O. (2016). Merging Modeling and Programming Using Umple. In T. Margaria & B. Steffen (Eds.), *Leveraging*

Applications of Formal Methods, Verification and Validation: Proc. 7th Int. Symposium ISoLA 2016, LNCS 9953 (pp. 187–197).

Paige, R. F., Polack, F. A. C., Kolovos, D. S., Rose, L. M., Matragkas, N. D., & Williams, J. R. (2014). Bad Modelling Teaching Practices. In B. Demuth & D. R. Stikkolorum (Eds.), *Proc. Educators' Symposium at MODELS.* (Vol. 1346, pp. 1–12). CEUR-WS.org.

Ratiu, D., Pech, V., & Dummann, K. (2017). Experiences with Teaching MPS in Industry - Towards Bringing Domain Specific Languages Closer to Practice. In *Proc. 20th MoDELS* (p. 83-92). IEEE.

Rumbaugh, J. E., Jacobson, I., & Booch, G. (2005). *The Unified Modeling Language Reference Manuel - Covers UML 2.0, Second Edition.* Addison-Wesley.

Simonot, M., Homps, M., & Bonnot, P. (2012). Teaching Abstraction in Mathematics and Computer Science - A Computer-supported Approach with Alloy. In M. Helfert, M. J. Martins, & J. Cordeiro (Eds.), *Proc. 4th Int. Conf. Computer Supported Education* (pp. 239–245). SciTePress.

Warmer, J., & Kleppe, A. (2003). *The Object Constraint Language: Getting Your Models Ready for MDA*. Reading/MA: Addison-Wesley.

WikiTeam. (2018). Ecosystem. *Wikipedia*. Retrieved from `https://en.wikipedia.org/wiki/Ecosystem`