Can a Systems Biologist Fix a Tamagotchi?

Luca Cardelli

Microsoft Research

Preface

Gilles Kahn was a serious scientist, but part of his style and effectiveness was in the great sense of curiosity and fun that he injected in the most technical topics. Some of his later projects involved connecting computing and the traditional sciences. I offer a perspective on culture shock between biology and computing, in the style in which I would have explained it to him.

1 The Nature of Nature

In a now classic peer-reviewed commentary, *"Can a Biologist Fix a Radio?"* [5], Yuri Lazebnik describes the serious difficulties that scientists have in understanding biological systems. As an analogy, he describes the approach biologists would take if they had to study radios, instead of biological organisms, without having prior knowledge of electronics:

> *We would eventually find how to open the radios and will find objects of various shape, color, and size [...]. We would describe and classify them into families according to their appearance. We would describe a family of square metal objects, a family of round brightly colored objects with two legs, round-shaped objects with three legs and so on. Because the objects would vary in color, we will investigate whether changing the colors affects the radio's performance. Although changing the colors would have only attenuating effects (the music is still playing but a trained ear of some people can discern some distortion), this approach will produce many publications and result in a lively debate.*

In such a generally humorous style, Lazebnik makes several deep points about the nature of complex systems, and about the concepts and languages that one must develop to even begin to describe them appropriately. One of his main points is that it is not enough to classify or even understand individual components, but one must understand the circuits that arise from them. That point of view has become one of the founding principles of Systems Biology [6], and is inspiring huge experimental efforts aimed at mapping biological "circuits" in order to eventually learn how to "fix" them rationally rather than empirically. (This is related to earlier rational approaches that include, for example, "fixing radios by thinking" [7].)

Lazebnik's analogy between biological and hardware circuits is illuminating, especially coming from a professional biologist who (unlike me) has full moral authority on such a subject. But the analogy is not particularly accurate when considering whole biological systems. Biologists cannot understand radios without any knowledge of electrical engineering but, similarly, electrical engineers can-

not understand mp3 players without any knowledge of software engineering, even though mp3 players play music just like radios. As it turns out, even the simplest biological organisms have a minimum of hundreds of kilobytes of software, in the form of digitally stored genetic information, and can have as much as several megabytes of it (considerably more than mp3 players). There is no life form that is just hardware circuits; not even viruses, which on the contrary are almost purely software. The role of such genetic software is commonly understood as that of "running the organism". All forms of reproduction consist in the software making a copy of itself, by a process that is pretty much standardized across all organisms, and in orchestrating the replication of the required hardware.

Therefore, biologists need to do more than learn how to fix radios. There may come a time when all the "circuits" of a cell will be completely understood, but we will still have little insight on how to go about fixing one. That is because software is not circuits: a full knowledge of microprocessors and radio transmitters is completely irrelevant when trying to understand, for example, why a mobile phone is not fetching email; it has nothing to do with the circuits, usually. Lazebnik's basic analogy might be considered as extending to "software circuits" as well, but it is just too naïve to think of software in term of circuits. Software is much more plastic: it could perhaps be defined as dynamically reconfigurable circuits. As such, it is not easily captured by static diagrams, witness the fact that almost no software is written in any notation resembling circuits. Such a fundamental plasticity implies also variability. All radios of a given brand and model have the same circuit, and behave in exactly the same way, assuming the hardware is intact. Instead, genetically identical cells of the same organism behave differently within a population; just like my phone may fetch email, and your absolutely identical phone may not. Therefore, biological organisms do not really behave much like radios, but, to carry forward Lazebnik's work, we can perhaps find another technological analogy where software has a more prominent role.

2 A Technological Organism

The goal of biology is to reverse-engineer biological organisms. We pick a somewhat less ambitious task, by focusing on a "naturally occurring" technological organism instead of a biological one. By choosing technology over biology, we are sure that someone has engineered the organism in the first place, and therefore there can be no doubt that in principle it *can* be reverse-engineered. Still, even with this simplified task, we will find many practical difficulties that are typical of reverse-engineering biological organisms, including several that are not typical of reverse-engineering radios.

Tamagotchi [8] will be our model organism. In the scientific tradition we use a binomial nomenclature: since it was first observed "in the wild" in Japan, it will thereafter be known as *Tamagotchi nipponensis* (several species and subspecies exist).

T. nipponensis [3]

T. nipponensis is a handheld cyberpet. Morphologically, it consists of an egg-shaped case with a strap, a small bitmap screen, three buttons, sound output, and a highly variable skin pattern. Little else can be easily observed (unless one has the clear-plastic model). An antenna-like appendage appeared in a recent evolutionary turn, but its function is poorly understood. The buttons can be used to

manipulate a small character on the screen (a *cyberpet*) that evolves along a wide range of possible stages, lineages, and shapes. The mode and frequency of interaction with the buttons influences the development of the cyberpet, in highly non-trivial ways.

T. nipponensis makes an interesting case study because it blurs the boundaries between "devices" and "organisms". Like all life forms, it is primarily an information-processing device: it has inputs, outputs, and internal state. It is governed by software, and is certainly not a full computer, but is not a simple automaton either. Most interestingly, it has a fundamentally *stochastic* behavior, as helpfully explained by Bandai [8]:

> *Q: How often do I have to exercise my Tamagotchi?*
> *A: Every Tamagotchi is different. However we do recommend exercising at least three times a day.*

Hence, *T. nipponensis* is nondeterministic ("every one is different"), and is stochastic because the *rate* of interaction matters ("at least three times a day"). Without proper interaction, the cyberpet soon passes away, after which the device needs to be reset. Unlike a radio, which can keep playing music without any interaction, the only way to understand *T. nipponensis* is to understand its full dynamic behavior. Any of its steady states, either at any particular instant, or after a long period of constant input, is in fact of no interest to anybody.

How can we go about unraveling such a complex dynamic phenomenon, in order, for example, to grow healthy cyberpets consistently? That is what science is really all about; we cannot rely on rumor, superstition, and blogs: we need to apply a reliable, field-tested, scientific methodology.

3 The Scientific Method

Our task is now simply stated: reverse-engineer *T. nipponensis* legally, that is, by using the *scientific method* (no industrial espionage). The scientific method consists of:

- Running reproducible experiments that elucidate the phenomenon at hand.
- Building models that explain all past experiments and predict the results of new experiments.

If *T. nipponensis* can be reverse-engineered, then very important applications open up, including:

- Providing scientifically-based cyberpet consulting services.
- Engineering and selling our own species of *T. europaea*.
- Fixing broken *T. nipponensis*, to the joy of kids everywhere.

How can the scientific method be applied to such a problem? Many different approaches, both experimental and theoretical, have been devised over the centuries; let's examine them.

3.1 Understanding the Principles

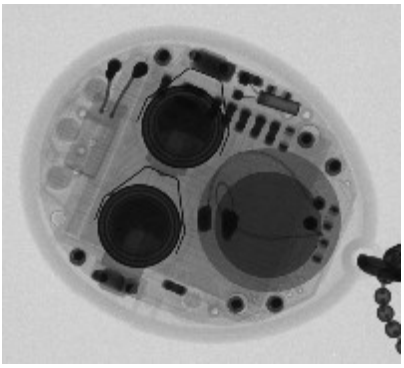
This approach assumes that the organism underwent some kind of design and optimization process, either by an intelligent designer, or as the result of competition and evolution, or both. The basic assumption is that there are some *principles*, either deliberate (design principles) or emerging (empirical principles), that somehow determine the organization of the organism, and whose discovery can be helpful in modeling it. According to Charles Darwin, we should be *"grouping facts so that general laws or conclusions may be drawn from them"*. The opposite point of view, in the words of Norbert Wiener, is that there are no organization principles whatsoever: *"The best material model of a cat is another, or preferably the same, cat"*, but we can safely ignore this opinion in the present case.

Here are some typical questions that arise from attempts at understanding principles of organization. We begin with a couple of questions that are not normally asked by biologists, but that we need to consider because they would be first in the mind of technologists:

The Creator [1]

Q1: Who created it? We actually know the answer to this question (Aki Maita), but it does not help: hiring the creator as a consultant is not considered part of the scientific method. Moreover, how could something so unique and sophisticated as a Tamagotchi have suddenly appeared seemingly out of nowhere? It is such an unlikely and unparalleled phenomenon that we have to question whether we could ever actually understand the mind of the creator, and whether the creator herself truly understands her own design (*"Aki's own Tamagotchi seldom lives longer than its baby stage." - Apple Daily*).

Q2: Where is the documentation? Well, there is no documentation, at least no design manual that explains what its principles are or how it works. Even if we could acquire the design manual from the creator (by industrial espionage), it would be written in the language of the creator, i.e., Japanese, and would be of little use to us. Now, turning to more scientific questions:



T. nipponensis X-Ray [2]

Q3: What is its function? What does a Tamagotchi compute? We have here a relatively primitive information processing device, but there is no easy way to explain what its processing function actually is. In fact, its function is not quantifiable; it does not appear to compute anything in particular. And how can we hope to understand its design principles if we cannot say what it *does*?

Q4: Why does it have 3 buttons? There surely must be a deep reason for this: 3-button devices are comparatively rare in technology. Did it evolve from archaic 2-button devices of which we have no record? Is 3 just the closest integer to e? Is there some general scaling law that relates the size of a device to the number of its buttons? It seems that none of these questions can be answered from abstract principles.

Conclusion: Principle-driven understanding fails.

3.2 Understanding the Mechanism

The mechanistic approach assumes that the organism is a *mechanism*, and hence can be understood by understanding its parts, and how they are joined together. And never mind who designed it or why. Laplace wanted us to understand *"all of the forces that animate nature and the mutual positions of the beings that compose it"*. Here are some typical questions for mechanistic understanding:

Q1: What are the parts? As Lazebnik points out, we need to make a list of parts. Favorite are *moving parts*, because they can be readily seen to be doing something mechanistic. Unfortunately, *T. nipponensis* has no moving parts, except for some minor button displacement. It has parts that never move, and it has a cyberpet moving on the screen, but it cannot really be called a part because it cannot be separated from the rest.

T. nipponensis Surgery [2]

Q2: How are the parts connected? If we open it up we can see how some parts are connected. But what about the cyberpet on the screen; how is that connected to the rest? It certainly seems to be connected to the buttons, and to the battery, somehow, and obviously to the screen. But unfortunately we cannot find any physical connection between it and the rest. As we said, it is probably not a part, but if it is not a part, then there can be no mechanistic understanding of it. And if we cannot understand how the cyberpet is connected, can we ever claim to understand *T. nipponensis*?

Q3: How does it react to perturbations? This means removing or changing parts to discover their contribution to the system, or interfering with their connections (the classic wrench in the clockwork experiment). Since there are very few discrete parts, removing almost any part just makes it stop working. And since there are almost no moving parts, and it is very small, there are very few places where we can usefully stick a wrench, with current technology.

Q4: How is it put together? One way to understand a mechanism is to understand how it is fabricated and assembled. We can call this the *Tamagotchi folding problem*: how do you fold the parts into the whole? Unfortunately, this turns out to be a much harder problem than the original problem. *T. nipponensis* is assembled in top-secret factories in Japan, Taiwan, or China, by robots. We do not have access to those assembly lines, and we do not even know where they are. And even if we did, those robots would surely be more difficult to understand than *T. nipponensis* itself, either mechanistically or otherwise.

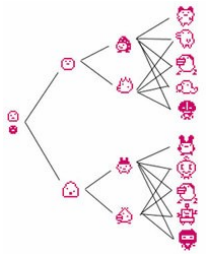
Conclusion: Mechanistic understanding fails.

3.3 Understanding the Behavior

The behavioral approach assumes that it is too difficult or premature to try to understand something in mechanistic detail, but we can still try to understand how it behaves. If we can characterize and predict its behavior in future experiments, which is all that the scientific method really requires, then any mechanism that implements it is accidental, and practically irrelevant. However, we first need to

identify some suitable behavioral quantities to measure. As Galileo put it, *"measure what is measurable, and make measurable what is not so"*, which is easier said than done: what is there to measure about a Tamagotchi?

Q1: How does it react to stimuli? Well, it has no consistent reaction to stimuli: it is nondeterministic and stochastic, remember? It turns out that its behavior is unpredictable *by design*. We have now discovered a design principle, but not a very useful one. Each individual has a huge range of possible behaviors, and it would take the age of the universe to supply all possible sequences of stimuli. Hence, experiments on individual *T. nipponensis* are not reproducible. Oh, yes, that was a requirement of the scientific method.



Cyberpet Growth Chart (draft) [3]

Q2: How does it behave in a population? Still, we can get statistics. For example, we can put 1024 *T. nipponensis* in a basket (such baskets are incidentally readily available in Japanese stores), and shake them violently for 3 hours. This will cause buttons to be pressed in a random but very large number of different combinations. Every 10 minutes, we scan the screen of each unit (this requires very expensive equipment or a large number of postdocs), and we plot the number of configurations and how they change over time. The result of such an experiment is publishable; it is after all an indisputable and fairly reproducible Fact of Nature, but it does not really help us make wide-ranging behavioral predictions.

Q3: How does it communicate? *T. nipponensis* is known to communicate with other *T. nipponensis* and with mobile phones through its antenna. (This can be argued, mechanistically, by clipping off the antenna, although strictly speaking one could say only that the antenna is *implicated in communication*.) Unfortunately, the communication protocol it uses is not an open standard. The question of whether it has a symbolic language, or whether it simply has innate reactions to specific data packets, is still subject to considerable debate.

Q4: How does it react to shock? This is the behavioral version of removing random parts: failures under extreme conditions can provide valuable insights on the structure of a system. Unfortunately, most extreme conditions here result simply in a blank or broken screen, and may void your warranty. Certain temperatures and pressures produce interesting effects on the screen, but again these are completely unrelated to what the cyberpet on the screen is doing.

Conclusion: Behavioral understanding fails.

3.4 Understanding the Environment

Sometimes it is just not possible to understand an organism in isolation or even as a population. Its properties can only really be explained in the context of its environment and how the organism interacts with it, both cooperatively and competitively. *T. nipponensis* could not exist without the Japanese culture and its electronics industry, and hence we must ask how it arose from that marketing envi-

ronment. To paraphrase Theodosius Dobzhansky: *nothing in consumer electronics makes sense except in the light of competition*.

Q1: How did it evolve? *T. nipponensis* evolved and prospered within a modern economic system; unfortunately such systems are just as poorly understood as biological systems. Furthermore, it evolved within the Japanese economic / technological / cultural environment, for which we have no consensus model. It is not entirely known, at present, which other technological organism it evolved from, which it adapted against, and which it displaced from the market: much of that information is proprietary. The archeological record will eventually offer insights through the excavation of Japanese landfills. Provided that adequate funding can be obtained, overcoming those critics who believe that the entire electronics industry was created last thursday.

T. nipponensis in its Natural Environment [3]

Q2: How does it behave in its natural environment? The natural environment of *T. nipponensis* is kids' hands and backpacks, both of which are practically impossible to reproduce in a laboratory. *T. nipponensis* has been painstakingly observed in its natural environment by sociology postdocs, but those studies were focused mostly on kid behavior. In any case, it is very difficult to reliably observe a Tamagotchi screen under natural kid-Tamagotchi conditions, and attempts to attach bulky tracking and telemetry devices result in atypical behavior.

Conclusion: Environmental/evolutionary understanding fails.

3.5 Understanding the Math

Since Pythagoras, who famously proclaimed that "*Number Rules the Universe*", scientists have wondered at the inexplicable effectiveness of Mathematics at explicating Nature. We now seem to have a counterexample:

Q1: What differential equations does T. nipponensis obey? Hmm...

Conclusion: Mathematical understanding fails.

4 Standing Aghast on the Shoulders of Giants

Of course, we know very well why the scientific method seems to fail, or at least to be unusually difficult to apply, in such a case. That is because we are trying to reverse-engineer what is a fundamentally a complicated piece of *software* (with some cheap hardware) instead of some kind of circuit. And that turns out to be quite a special kind of activity. Not much progress can be made until we start dumping and disassembling the software itself, provided of course we have some idea of what kind of hardware it is running on.

We also know very well how to reverse-engineer software [9]; it can be quite difficult, but historically it has never been impossible, even when extraordinary steps have been taken to prevent it. Soft-

ware reverse-engineering techniques include: *tracing* (selectively following the flow of control and the value of variables over time), *breakpointing* (stopping program execution when a certain program point is crossed, or when a certain data structure is modified), *core-dumping* (taking a raw snapshot of the active memory content), *stack-dumping* (taking a snapshot of the active execution context), *packet-sniffing* (examining the traffic on a network wire), *reverse compilation* (partially reconstructing the structured source code from the unstructured binary code, provided that knowledge is available about the source language and the compilation process), *power analysis* (gathering information about a running program from the instantaneous power consumption of the processor), and so on.

Corresponding techniques are often not available in biology (particularly, reverse compilation!), but some are. Tracing is achieved by setting a biological system in a known initial state and then measuring some quantity at timed intervals; the problem is that it is often difficult to measure the quantities of interest. Breakpointing, by various forms of genetic perturbation, is used in biology to stop a pathway at a certain stage, so that one can determine the sequence of events in the pathway. Packet sniffing is commonly used to inspect the nervous system, but it has led so far to limited knowledge of what the packets mean, and what information is actually being transmitted. It is also used to analyze intracellular chemical communication.

How to Debug a Tamagotchi

Debugging will not hurt your Tamagotchi. Rather, it lets you have any character you'd like and can also reveal secret characters. Here is how to debug your Tamagotchi. This does *not* work for Version 3 Tamagotchis, however.

T. nipponensis Debugging [4]

In practice, biologists measure all they *can* measure, and sometimes what they do measure would be of extremely little help in reverse-engineering a software system. Core-dumping is partially feasible in biology (e.g., by detecting the phosphorylation state of a large number of proteins), but because of technical difficulties this is usually applied to whole populations of cells. Certainly, averaging millions of software core dumps would produce very low-quality information about the behavior of a program. Stack-dumping is currently extremely popular in biology: microarray technology can determine what parts of the genetic program are currently running by detecting mRNA molecules. But again, this has to be done on whole populations of cells, and only by detecting the difference between the altered and the standard behavior of the genetic network (which is normally not understood). By comparison, averaging the execution stacks of many program runs could lead to some knowledge about the subroutines that are most used, and about their average parameters, but little else. Even when considering a single concurrent program, summing the stack frames from different threads of control would lead to very little insight. Another common inspection tool in biology is gene perturbation experiments. As already mentioned, this technique can provide useful information, but it is also used more blindly, e.g., by deleting in turn *every* single gene in an organism to see what happens. In software engineering, no one has ever seriously proposed to remove each instruction in a program in turn to see what breaks. One might have a slightly better chance of acquiring useful knowledge by removing all pairs or triplets of instructions, but this immediately becomes unfeasible.

In summary, the most popular high-throughput experimental techniques in biology do not provide the right conceptual or technical tools needed to reverse-engineer software, be it digital or biological. In one case, part of the regulatory region of a single gene has been decoded in great detail, but that has required a heroic effort [11]. More promising techniques are just becoming available that can inspect the state of individual cells, and these will certainly lead to significant progress. But even that is not sufficient in itself: Andreas Wagner has given an example of a situation where no set of classical

experiments can recover the structure of a simple biological network, even from single-cell observations [12]. In general, to succeed one must be able to probe the system in sufficient depth, because just measuring a large quantity of superficial properties may not be enough.

What are the chances, eventually, of reverse-engineering the software of life, assuming that the experimental difficulties can be resolved? We have to believe, with Einstein, that nature is subtle but not malicious: it does not encrypt its software just to make it harder for us to copy it or modify it. Still, if the complexity of ordinary software systems is any hint, the task is going to be much more difficult than people generally realize, even after systematically recovering the raw code (*genomics*), taking stack traces (*transcriptomics*), taking core dumps (*proteomics*), monitoring the power supply and the heap size (*metabolomics*), and intercepting the network traffic (*systems biology*). Any additional amount of understanding is going to require extensive measurements and experiments, just like the ones that are being carried out today, but the focus must be on the peculiarities of software systems, not only of hardware systems.

The reverse-engineering task confronting biologists surely looks hopelessly complex, but we should not despair: progress so far has been incredible. Standing on the shoulders of giants is not particularly helpful when confronted with much taller mountains, but we can rely on a steady tradition of fundamental discoveries. Sydney Brenner captured both the scope and the spirit of the endeavor when he said that: *"The problem of biology is not to stand aghast at the complexity but to conquer it"* [10]. And so we must, because we got things to fix.

References

"Tamagotchi" is a registered trademark of Bandai Co., Ltd. of Tokyo, Japan.

[1] Aki Maita's picture is from <http://www.mimitchi.com/html/q10.htm> (1998).

[2] X-ray and surgery pictures are from Ed T. Toton III, <http://necrobones.com/tamasurg>.

[3] Tamagotchi pictures are from Bandai's Tamagotchi Connection, <http://www.tamagotchi.com>.

[4] Debugging advice can be found at <http://www.wikihow.com/Debug-a-Tamagotchi>.

[5] Y. Lazebnik. **Can a biologist fix a radio? Or, what I learned while studying apoptosis.** Cancer Cell 2 179-182. 2002.

[6] E. Klipp, R. Herwig, A. Kowald, C. Wierling, H. Lehrach. **Systems Biology in Practice**. Wiley 2005.

[7] R. Feynman, R. Leighton, E. Hutchings. **He fixes radios by thinking.** A chapter in: Surely You're Joking, Mr. Feynman!: Adventures of a Curious Character. W W Norton, 1985.

[8] Bandai. **Tamagotchi Connection**. http://www.tamagotchi.com

[9] P. Tonella, A. Potrich. **Reverse Engineering of Object-Oriented Code**. Springer 2005.

[10] S. Brenner. **Interview**, Discover Magazine, Vol. 25 No. 04, April 2004.

[11] C.-H. Yuh, H. Bolouri, E. H. Davidson. **Genomic Cis-Regulatory Logic: Experimental and Computational Analysis of a Sea Urchin Gene.** Science, 279:1896-1902, 1998.

[12] A. Wagner. **How to reconstruct a large genetic network from n gene perturbations in fewer than n^2 easy steps.** Bioinformatics, Vol. 17 No. 12, pp. 1183-1197, 2001.