Classes, styles, conflicts: The biological realm of LaTeX

Didier Verna

Abstract

The LaTeX world is composed of thousands of software components, most notably classes and styles. Classes and styles are born, evolve or die, interact with each other, compete or cooperate, very much as living organisms do at the cellular level. This paper attempts to draw an extended analogy between the LaTeX biotope and cellular biology. By considering LaTeX documents as living organisms and styles as viruses that infect them, we are able to exhibit a set of behavioral patterns common to both worlds. We analyze infection methods, types and cures, and we show how LaTeX or cellular organisms are able to survive in a world of perpetual war.

1 Introduction

Every LaTeX user faces the "compatibility nightmare" one day or another. With such great intercession capability at hand (LaTeX code being able to redefine itself at will), a time comes inevitably when the compilation of a document fails, due to a class/style conflict. In an ideal world, class/style conflicts should only be a concern for package maintainers, not end users of LaTeX. Unfortunately, the world is real, not ideal, and end-user document compilation *does* break.

As both a class/style maintainer and a document author, I tried several times to come up with a systematic approach, or at least some general principles on how to handle class/style cross-compatibility in a smooth and gentle manner, but ultimately failed, because the situation is just too complex. Classes and styles evolve constantly, sometimes even in a backward-incompatible way. Classes and styles die, while new ones are born. Styles may conflict not only with classes but with other styles as well. Styles may be made aware of classes or other styles, but classes may be made aware of styles as well. Then, there is the influence of the end user who will combine all available material in a somewhat unpredictable way, possibly with his/her own personal additions, or even modifications to the available features.

This vicious circle basically never ends and leads to a paradoxical "If it ain't broke, then fix it" situation in which complex trickery is added to classes or styles, not to make them work out of the box, but to *prevent* potential breakages resulting from interactions with the outside world. In the end, the only realistic conclusion is that there is no solution to this problem, both because the system is too liberal, and because the human factor is too important. One cannot force a package author to write good quality (for some definition of "quality"), non-intrusive or even just bug-free code. One cannot force a package author to keep track of all potential conflicts with the rest of the LaTeX world, let alone fixing all of them by anticipation. One simply cannot prevent software evolution.

Facing this somewhat pessimistic conclusion, it is all the more intriguing to acknowledge the fact that the system still globally works. Despite the complexity of what happens behind the curtain, documents *are* being produced, and in some way, seeing a freshly compiled document pop up on the screen is just like witnessing a small miracle. When it doesn't compile, you don't really know why, but when it does compile, you really don't know why. This is the precise point at which the parallel with biology occurred to me. Any living being is by itself a miracle of complexity, and unfortunately, sometimes it breaks as well.

One Monday morning, I woke up with this vision of the LaTeX biotope, an emergent phenomenon whose global behavior cannot be comprehended, because it is in fact the result of a myriad of "macro"-interactions between smaller entities, themselves in perpetual evolution. In this paper, I would like to build bridges between LaTeX and biology, by viewing documents, classes and styles as living beings constantly mutating their geneTeX code in order to survive `\renewcommand` attacks....

The basis of our analogy is to consider LaTeX documents as living beings, and styles as viruses that infect them. Based on this picture, a number of puzzling similitudes can be found in the way organic/LaTeX material interact. In the following, we first describe how LaTeX documents can be morphologically compared to a specific kind of organic cells, then draw a parallel between genetic and programmatic material, and finally justify our view of styles as viruses. After that, we respectively draw interesting comparisons between existing viral or stylistic infection methods, infection types, and also possible cures.

2 Morphological analogy

In this section, we present a morphological analogy between LaTeX documents and a specific kind of organic cells from the so-called "eukaryotes" domain.

2.1 Eukaryotes

According to Whittaker's nomenclature [23], *eukaryotes* subsume four of the five "kingdoms" of life,

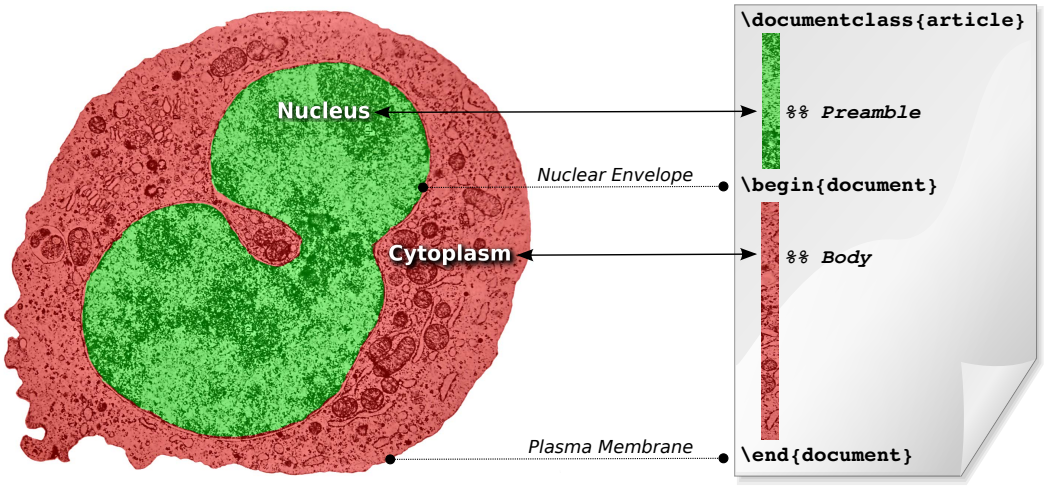

Figure 1: Unicellular LaTeX document

including animals and plants. As opposed to *prokaryotes*, the cells in a eukaryote organism have a complex structure, and in particular contain a *nucleus*. The nucleus is separated from the rest of the cell by the *nuclear envelope* and the cell itself is separated from its surrounding environment by the *plasma membrane*.

The contents of a cell is called *cytoplasm*. In eukaryotes however, the contents of the nucleus, called *nucleoplasm*, is not considered to be part of the cytoplasm. While the nucleus contains most of the cell's genetic material, cellular functions (materialized by all sorts of biochemical activities) occur in the cytoplasm.

2.2 Unicellular LaTeX documents

Figure 1 shows a LaTeX document as a unicellular eukaryote organism. Such a document indeed has a specific structure. In particular, it contains a *preamble* (the document's nucleus) and a *body* (the document's cytoplasm). The preamble is separated from the body by a call to `\begin{document}` (the nuclear envelope) and the document ends with a call to `\end{document}` (the plasma membrane).

Just as a cell's nucleus contains most of the genetic material, a document's preamble contains most of the *programmatic* material: a selection of one class and a set of styles, (re)definitions for new (existing) commands, *etc.*

Finally, most document "activity" (meaning command expansion, execution and actual text processing) occurs in the document's body, just as most cellular activity occurs in the cell's cytoplasm.

3 Functional analogy

Based on the morphological analogy described in the previous section, we can push the idea one step further by comparing genetic and programmatic material together. This leads to a parallel between how cells or LaTeX documents "work", and which function they fulfill.

A living cell performs some functions. Aside from reproduction, which is a very common one, different cells may function differently. For instance, the role played by a liver cell is different from that of a neuron. The role of a LaTeX document, on the other hand, is essentially to compile correctly in order to be read. Both cells and LaTeX documents implement their respective functionality through very similar processes: some piece of static information is read and used in order to produce a concrete result or side effect, much like a factory creates concrete items based on formal specifications.

3.1 Cellular factory

Cells basically work by expressing *genes*. Genes can be seen as an infrastructure that stores static pieces of information (the formal specifications for what needs to be produced). This information does nothing by itself. However, it is available on demand.

A gene is said to be *expressed* when the information it contains is used to actually produce something, for instance, a protein. Expressing a gene can roughly be described as a two-step process: first the DNA sequence is *transcribed* into mRNA (messenger RNA). This serves as a temporary copy of the original DNA sequence which helps preserve the original piece of

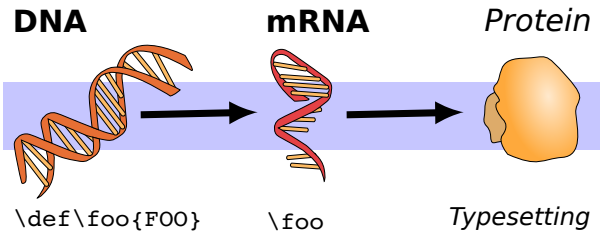

Figure 2: The geneTEX factory

information from deterioration. Next, the resulting mRNA is "read" by a ribosome, which will eventually produce the resulting protein.

3.2 TEX factory

The similarity with TEX macros (in fact, with functions in any programming language) is striking. The biologists themselves speak of genetic "code", because it is exactly that: just like a gene contains a formal specification for something to be synthesized, a TEX macro definition contains a formal specification for something to be executed. The result is not as concrete as a protein, but instead consists in some side-effect like the actual typesetting of a portion of a document.

Just like genes, TEX macros don't do anything on their own, but are available on demand. Figure 2 depicts the process of expressing a gene or executing a TEX macro. A macro *definition* is much like a gene, which encodes a formal specification. A macro *call* is much like a messenger: an actual instance or copy of the original information which is about to be concretely used. TEX does not use ribosomes, but a "mouth" and "stomach" instead (as per Donald Knuth's terminology in [6]), to perform macro expansion and (primitive) command execution.

In the remainder of this paper, we use the term "geneTEX material" to designate both genetic material in cells and programmatic material in documents.

4 Higher view of geneTEX material

So far, we have established a morphological ground on which to compare unicellular eukaryotes and LATEX documents, and we also have exhibited similarities in the way genetic and programmatic information is processed. It is now time to stand back a little and get some perspective on why this comparison is interesting.

4.1 Roles

We know that genes (or rather the information they encode) determine the way a cell will function. Let us consider two eukaryote cells, for instance resulting from the mitosis of a mother cell, and hence equipped with the same initial genome (in other words, the same functional potential). Why do these two cells eventually turn out to be different?

The difference comes from the fact that the set of actually expressed genes differ, because the orders emanating from the cytoplasms differ, in turn partly because ultimately, the cell's environments differ. In other words, different cytoplasms lead to cells functioning differently, even when the original genetic material is the same. In addition to that, variations in the environment also lead to more divergence from the two cytoplasms as time passes.

Now consider LATEX articles, reports, books, *etc.*, in the sense of their corresponding `\documentclass`. It is usually very easy to figure out the class of a document just by the look of it. Two articles look similar because their general layout is the same: it is dictated by the `article` class. Consequently, a document's class can be seen as its original geneTEX material. Two articles look roughly the same but are still different, just like two liver cells are *both* liver cells, but still different ones. If we extrapolate a little further, outside the unicellular world, the morphological similarities between documents of the same class are not unlike those between brothers and sisters or even twins (blue eyes, red hair, *etc.*) that may share the same genetic pool although expressed slightly differently.

Given this parallel, what makes two `article` documents different is also exactly what makes two liver cells different. The set of expressed genes/macros may differ (you might or might not use `\subsubsection`), the orders coming from the document's cytoplasm/body may differ (you may issue macro calls at different times and with different parameters), and in fact the whole documents' cytoplasms/bodies diverge (their respective text is not the same). The ultimate source of divergence is of course the documents' authors, who write different documents. A document author clearly takes the role of the cell's environment here.

4.2 Sources

A LATEX document. just like a cell, is a viable entity as soon as its initial geneTEX material is defined (its class), and it has a healthy body/cytoplasm. However, it is rare that a document is satisfied only with a class. In fact, a perhaps even more important and abundant source of geneTEX material is the use of *styles*. Styles provide the same kind of material as classes: macro definitions. The difference is that styles are not needed to give birth to a document. When some are used, however, the document may

function slightly or sometimes very differently.

In this context, the idea of viewing styles as *viruses* that infect unicellular LaTeX documents turns out to be quite natural. Viruses are biological entities, mostly genetic components, that need a host cell to replicate themselves. Viruses are thus characterized by the fact they cannot perform their function on their own (this is why viruses are not considered as living entities [4, 13], although the debate is still open [11]). A LaTeX style has similar properties: it is basically useless as a standalone entity and needs to "infect" a host document in order to perform its function. Just like a virus, a style adds its own geneTeX material to the document's original pool.

Viruses are usually small compared to the organisms they infect, apart from two notable exceptions: the *mamavirus* and the *mimivirus*, which are twice as big as the average. A quick survey of the TeX Live 2009 distribution exhibits a surprising coincidence: among 2462 available `sty` files, the average size is around 327 lines of code (LoC), with a median at 134. Styles are indeed rather small. However, two of them are exceptionally bigger than the others: `texshade.sty` and `xq.sty`, with 14470 and 24535 LoC respectively. These styles can arguably be called the *mimistyle* and the *mamastyle* of LaTeX.

All these considerations make it interesting to analyze and compare the ways genetic material from cells and viruses, or programmatic material from classes and styles, interact. This is the purpose of the following sections.

5 Infection methods

In cells as in documents, there are many ways to be infected with new geneTeX material. The following methods come to mind in both worlds.

5.1 Exogenic

Perhaps the most common way for a document to be infected by a style is to "request" explicit infection by means of the `\usepackage` command. This results in the incorporation of the style into the document's preamble, the style being indeed an external LaTeX entity stored in a file of its own. We could even use the term "stylon" to denote the style file, in reference to the biological term "virion" which denotes the viral particle outside the cell it is bound to infect.

Such a style infection always occurs *after* the initial geneTeX material of the document has been defined, since `\documentclass` must appear first. As such, the style's material is not technically part of the original document's material. The infection occurs *afterwards*.

In biology, this process is said to be *exogenic*: the cell is infected by the virus after it has been created (for instance, after mitosis), and the genetic material brought by the virus is not part of the cell's original pool.

5.2 Endogenic

A document might however be infected by a style without even knowing it: a class may request infection by means of the `\RequirePackage` command. As such, when a document is created based on this class, even without explicit addition of any style in the preamble, the document is already infected. Arguably, this is a situation in which the geneTeX material brought by the style is indeed part of the original genome, because it is impossible to create a non-infected document based on that class.

This kind of infection is known to be *endogenic* in biology: when a new cell is born, it already contains some genetic material that would have required a former infection, for instance of the mother cell.

About 10% of our own genetic source material is currently estimated to be endogenic. A quick survey of TeX Live 2009 reveals that 259 out of the 271 available classes (95%) "suffer" from endogenic infections, by 4 styles on average (the median being 2). *CurVe* [21, 22], for instance, is infected by the `ltxtable`, `ifthen`, `calc` and `graphics` viruses. The QCM style is also endogenic to the QCM class [20].

5.3 Endosymbiosis

In biology, viruses are not the only source of external genetic material. *Endosymbiosis* is defined as the mutually beneficial cooperation between two living organisms, one (the *endosymbiont*) contained within the other. The so-called *endosymbiotic theory* [7] suggests that some organelles from eukaryote cells (*e.g.* mitochondria) actually come from the endosymbiosis of former prokaryotes (cells without a nucleus).

In the LaTeX world, we must acknowledge the fact that endosymbiosis is not as widespread as it should be. What we usually observe is the opposite phenomenon: the proliferation of a multitude of different packages that are meant to work together, or do more or less the same thing, instead of becoming one single and bigger animal. To mention a couple of examples, I have recently attempted twice to contact the author of `doc` about incorporating the features of `DoX` [17] (in other words, to turn `DoX` into an endosymbiont for `doc`) but got no response.[1] Not long ago, I also launched a thread on `comp.text.tex` entitled "Please, make it stop!" in which I mentioned

[1] It seems, however, that recent versions of `doc` do contain endosymbiotic versions of `newdoc`, so there is still hope....

a couple of packages doing a similar job (key/value processing). At the end of the thread, the number of such packages, as reported by different participants, amounted to 13. LaTeX definitely needs more endosymbiosis. Maybe the LaTeX 3 project will help in this regard.

As a final note on endosymbiosis, we must admit that our analogy falls short on one point: in biology, the symbiotic organisms are *living* creatures (mitochondria are semi-autonomous: they live in cells but have independent division capabilities). In our case, we are mostly talking of symbiotic relations between styles and/or classes, which are not considered "living" documents (see section 4.2 on page 164).

5.4 Exosymbiosis

An interesting phenomenon in package development seems to do the opposite of endosymbiosis. We call it *exosymbiosis*. Many existing styles originate from quick, local and often dirty hacks in specific documents, that are gradually abstracted away and cleaned up in order to become styles of their own, officially distributed in the form of `sty` files (a "bottom-up" development approach, in other words). In this situation, a symbiosis continues to exist, but one of the "organisms" is made external to the other, hence the choice of terminology.

Here is a concrete example of this. For many of my lectures, I use the Listings package for typesetting code excerpts, and include them in Beamer blocks. Providing nice shortcuts for doing so is not trivial if one wants to preserve control over both Beamer blocks and Listings options. The way I currently do this is to simply cut and paste the same 50 LoC into every new document I create over and over again. Soon, however, this will become a style of its own (probably called `lstblocks`) and released on CTAN.

This development process can indeed be regarded as the opposite of endosymbiosis: at first, a document features some geneTeX material that does not belong to its original pool, and by the way, just as mitochondria live in the cell's cytoplasm, LaTeX macros can be defined anywhere, including the document's body instead of preamble. However, if we let natural evolution happen for some time, this material will ultimately be extracted from the original document and become a style, which in turn will have the ability to infect other documents.

This is as if genetic material from a cell would have been extracted and became a virus.

5.5 Transduction *vs.* transfection

Biologists speak of *transduction* when genetic material is brought to a cell via a viral agent (when you use a style). When no viral agent is involved, the term *transfection* is used instead. Transfection corresponds to our version of endosymbiosis, where the geneTeX material is not brought to the document by a style, but simply by the document's author typing some macro definitions locally.

Interestingly enough, most cases of transfection are transitory (as opposed to stable): the genetic material is not copied into the cell's genome, so it is lost after the mitosis, just like you would need to cut and paste endosymbiotic macros over and over again into every new document, unless they are properly incorporated into the relevant class. Class evolution can hence be seen as a case of stable transfection (or transduction if `\RequirePackage` is involved).

5.6 Stylophages

What if styles could infect other styles instead of just documents? This is in fact very common practice, as `\RequirePackage` can be used in styles as well as in classes. Again surveying TeX Live 2009 reveals that 1111 out of 2462 (45%) available styles are themselves infected, by 2 other styles on average.

This kind of behavior has been observed in biology as well, although only very recently. The first virus capable of infecting another virus, called the "virophage" in reference to bacteriophages, has been discovered by Didier Raoult's team in 2008 [12]. This virus infects the *mimivirus* (see section 4.2 on page 164) and uses its machinery in lieu of a cell's one in order to replicate itself.

6 Infection types

In the previous section, we have looked at similarities in the way cells or documents can be infected. In this section, we will analyze the *effects* of infection, and draw some analogies again. In other words, we will now consider examples of what infection *does* rather than how it *spreads*.

6.1 Standalone

Perhaps the simplest (and most harmless) form of style infection is by what we call *standalone* styles. Standalone styles provide macros that do not modify, interact, or even require anything particular from a document class or other styles. They just use the usual TeX machinery to add new features, completely orthogonal to the rest; in other words, geneTeX material that you are free to use... or not. An example of this is the `clock` package which provides macros for drawing clocks of all sorts of visual appearances.

This situation is similar to that of viruses infecting non-permissive cells (in which they can't replicate

themselves), but in which however their genetic material may remain in the form of free *episomes*, that is, not integrated into the cell's genome. Just like the information necessary to draw a clock is here but is really independent from the rest, (part of) the genetic information of the virus is also here, but does not interact with the cell's original genome. The genes brought by the virus may or may not be expressed depending on environmental conditions, just like you may choose to actually draw a clock or not in your document. If you don't, the presence of this exogenic material simply has no effect.

6.2 Prostyles

Instead of standing apart from the cell's DNA, viruses can have their genetic material incorporated into that of their host, in which case they are called *proviruses*. Because of this integration, proviruses passively replicate as part of their host's replication process, although just as for standalone viruses, infection can either remain latent or become active.

Because of this integration with the cell's original genome, the potential effects of a provirus are extremely wide: they can amplify, inhibit or even modify the different functions of their host. They can either have very little pathogenic effect, like AAV (Adeno-associated virus), or cause extremely serious diseases, like HIV.

The vast majority of LaTeX styles, including those mentioned in the following sections, qualify as *prostyles* in the sense that they incorporate their programmatic material into the existing one, instead of just contributing something new and orthogonal. Most styles in LaTeX indeed exist to enhance or modify an existing functionality.

The following very common programming idiom makes a style a prostyle:

```
\let\@oldfoo\foo
\def\foo{... \@oldfoo ...}
```

What this does is essentially to incorporate some new geneTeX material into the existing definition for `\foo`, thereby modifying its associated function. In a similar way, every time you `\renewcommand` something, you are creating a prostyle. The examples are innumerable in LaTeX. ***FiNK*** [18], for instance, is a prostyle because it modifies the behavior of `\InputIfFileExists`; `hyperref` [10] is another notable prostyle given the amount of semantic changes it inflicts on existing commands, *etc.*

In fact, the use of a prostyle as a means to alter the original programmatic material of a document looks very much like the use of a virus to *willingly* incorporate new genes into an organism. Such organisms are called GMO/GEO (Genetically Modified/Engineered Organisms). So we could say that using a prostyle in a LaTeX document makes it a GMD/GED (GeneTeXally Modified/Engineered Document).

6.3 Satellite

One (perhaps the most) important source of style proliferation in LaTeX is what we call *satellite* styles. A satellite style exists to amplify or extend the functionalities provided by another style. Examples of satellite styles are `DoX`, which extends the `doc` package, `graphicx` which extends `graphics` and `xkeyval` which extends `keyval`.

Satellite styles typically depend on the presence of their respective "sub-style" to work properly, because they build on top of them. They cannot work properly on their own.

This behavior exists in biology as well, as some viruses are considered to be satellites of others. For instance, the so-called *Delta* virus, or Hepatitis D Virus (HDV) is considered to be a satellite of HBV, Hepatitis B. The HDV cannot propagate without the presence of the HBV, but when both are present, the risk for complication, or the lethal rate increases. Other examples would be most of the avian sarcoma viruses, which require the help of a non-defective (see next section) leukemia virus.

Biologists distinguish between *co-infection*, when a patient is infected by both viruses at the same time, and *super-infection* when the two infections happen one after the other. In the LaTeX world, super-infection is or should be nonexistent because it would mean that a document author is required to explicitly `\usepackage` both styles, one after the other. A better practice for a satellite style is to `\RequirePackage` the style it depends on. This way, a document directly suffers from co-infection. In fact, satellite styles are almost always stylophages (see section 5.6 on the facing page).

6.4 Defective

Satellite viruses are in fact a sub-category of so-called *defective* viruses. A defective virus is a virus that lacks a complete genome and hence depends on another virus to provide the missing genetic function. While defective viruses are defective *by mutation*, defective styles are defective *by design*. Computer science does not usually like to reinvent the wheel; reusability is a key paradigm in software engineering. So when a package author creates a new satellite style, he or she usually avoids replicating the base functionality with "cut-and-paste", but relies on co-infection by `\RequirePackage`'ing the underlying functionality. The resulting style is indeed defective, but on purpose.

6.5 Host-dependent

When a virus is defective, it can rely on another virus to provide the missing genetic function, or it can rely on the host cell. Such viruses are not called satellite anymore, but are said to be *host-dependent*.

Host-dependent styles exist in the LaTeX world. Such styles would only work with a specific document class to provide the missing geneTeX functions. The example which comes to mind immediately is that of Beamer themes. Beamer is a class for writing slides, the appearance of which can be customized. Beamer themes are styles defining a specific set of morphological traits for slides, and are obviously specific to Beamer documents.

6.6 Cheaters

The defective styles presented so far work in a spirit of cooperation with their "helper": Beamer themes exist to enrich Beamer documents, `xkeyval` exists to improve `keyval`, *etc.* In a way, this is also the case for the HDV and HBV viruses which "work" together in the common and unfortunate goal of spreading hepatitis.

But what if some defective styles were in fact egoistically "stealing" functionality from their helper, and diverting it for a totally different purpose? What if, in other words, defective styles were working in a spirit of *competition* instead of *cooperation*?

In a very amusing way, there is at least one style that we know of which already cheats on itself: the `verbatim` package. This style provides an environment for outputting text as is, but also provides a `comment` environment which simply discards all its contents. Although comments are completely unrelated to verbatim text, the implementation of the `comment` environment *steals* the basic functionality used to produce verbatim text: the ability to have TeX read text without interpreting any commands or special characters.

The soon-to-come `lstblocks` package will do exactly the same. In order to properly integrate inline Listings and Beamer blocks (which cannot be nested out of the box), we use a trick based on `verbatim`: the inline text is first output to a file, and the file is later re-input by `\lstinputlisting`.[2] In other words, we steal functionality from `verbatim` in order to do something which is the exact opposite of what it is originally meant for: typesetting some text with heavy fontification instead of as is.

In a very puzzling way, cheaters also exist in the world of viruses. Experimental studies even show that cheaters often win the competition and overwhelm the cooperating ones [16]. An example of a cheating virus is the *umbravirus* [8, 15] that steals the coat protein of another virus, the *luteovirus*, in order to spread to other plants.

[2] The technical details of this process are explained in the following blog entry: `http://lrde.epita.fr/~didier/sciblog/index.php?entry=entry080604-120459`

7 Conflicts, diseases and cures

There is an angle from which the analogy between LaTeX and biology could be regarded as somewhat shaky: viruses are usually studied and well known for their pathogenic effects, while styles are supposed to do some good.

7.1 Good or bad, or both

The replication of a virus usually entails cellular *lysis* (the destruction of the host cell's plasma membrane) in order to disseminate new *virions* (complete virus particles) in the environment. The pathogenic potential of a virus (in other words its ability to lead to a disease) is described in terms of *virulence* and depends on the success of its replication. However, from a unicellular point of view, the presence of a single active viral entity is lethal to the cell. We, on the other hand, are more interested in the *benefits* of style infection: LaTeX styles are normally meant to be non-virulent and provide additional functionality: "useful viruses" in some way.

It turns out, however, that our analogy is not so shaky after all: a positive vision of viruses does exist, although it is still quite young. Only recently biologists have started to acknowledge the positive role of viruses as important factors of evolution or even as therapeutic tools. In fact, most viruses that we live with every day are harmless. Consequently, it appears that viruses, just like styles, have both a Dr. Jekyll and a Mr. Hyde face: viruses play a crucial role in evolution but they can cause diseases, while styles give you more power but can cause conflicts.

7.2 Conflicts and diseases

Just as proviruses can cause serious diseases, prostyles can cause the compilation to break, especially if some bad interaction happens between their geneTeX material and that of other styles or of the document class. Basically, a LaTeX document can be in three states: healthy, ill-formed or dead. An ill-formed document compiles successfully but displays incorrectly. A dead document is a document which could not compile, so TeX aborted. In a similar way, a cell can live normally, function improperly or be dead.

A very systematic and rather famous way of getting a living yet ill-formed LaTeX document is to infect it with the `a4wide` style. If your document

This is a test to see if the problems which seem to, for example?]FiXme Note: what does [this] do, for example? be caused by using square brackets in marginal fixme notes, even in a minimal document...

Figure 3: An ill-formed FiXme note

uses the `twoside` option, then the infection will render this option inoperative (odd and even pages get the same geometry). Another shameful example is that of FiXme [19] which seems to have problems typesetting square brackets in marginal notes, as depicted in figure 3. The next version, still under development, does not seem to suffer from this problem. The reason for this is currently unknown, but probably involves some kind of geneTEX mutation in the codebase....

The ways to break compilation because of style infection are too numerous to be listed here. The fact that you are reading this very paper can be considered a miracle in itself, given that the source involves both the `hyperref` and the `varioref` packages. Suffice to quote the README file from the `hyperref` distribution:

> There are too many problems with varioref. Nobody has time to sort them out. Therefore this package is now unsupported.

Note that viruses or styles are not a requirement for a cell to malfunction or die, or a document to be ill-formed or uncompilable. A cell can malfunction for many other reasons, including DNA mutation because of external conditions, *etc.* A document can turn out ill-formed or even die because there are bugs in its programming.

7.3 Cures

Just as in biology, facing the risk of possibly lethal diseases to documents entails the search for cures. In biology, viral infections are basically treated with either prevention, vaccines or antiviral agents.

7.3.1 Prevention

No cure is needed if no infection is present. In other words, if you don't want to get sick, then just don't get a disease. Knowing the risks in advance is hence the key to prevention, and this is probably much easier to do in LaTeX than in biology. In the LaTeX world, prevention will mostly be accomplished by *documentation*.

An example of prevention against a non-lethal disease is given in section 3.3 of the FiXme documentation: FiXme explicitly supports the standard LaTeX classes plus their KOMA-Script replacements for typesetting the list of FiXme's. For any other class, the `article` layout will be used, which will probably lead to an ill-formed list. In other words, some classes are known to be *immune* to FiXme infection, and for other cases, you know the risks.

Another example of prevention against lethal sickness is the quote from `hyperref`'s README file about `varioref` presented in section 7.2 on the facing page. What it really says is:

> *You are infected by Hyperref. If you want to live, don't be infected by Varioref as well: just don't use it.*

The other interesting aspect in this particular example is that the concern is about *super-infection* rather than just infection, since it deals with the presence of two styles simultaneously. This is a bit like saying:

> *You have got HBV. This is already serious enough. Just don't get HDV on top of that.*

7.3.2 Adaptive immune systems

Mentioning adaptive immune systems here is a bit borderline as it involves complex, multi-cellular organisms. However, there are still a number of interesting common patterns to mention.

Adaptive immunity (contrary to *innate* immunity) is the process by which an organism *acquires* defenses against a pathogen such as a virus. Immunological memory, materialized by the presence of so-called *B*- and *T-cells*, contains some kind of history of previously encountered infections, and how to fight them. We are principally interested in *active* immunological memory, which is a long-term memory of immunological responses. Such memory can be acquired naturally after a real infection (such as with measles or mumps), or artificially from vaccination. Vaccination typically consists of faking a real infection with a non-virulent form of a virus in order to trigger an immune system response.

What is interesting here is the pattern by which an organism suffers from an infection, learns to fight it, and then memorizes the "counter-measures" in order to be prepared for future attacks. This pattern exists in LaTeX and can be described by the following steps:

1. John writes a document of class `class`, but notices that when he uses the style `style`, compilation breaks.

2. John sends a bug report somewhere (such as `comp.text.tex`, the author of `class`, or more probably the author of `style`).

3. Some time later (for some definition of "some"), a new version of either `class` or `style` is released and everything works smoothly.

Now, depending on whether `class` or `style` mutates in order to circumvent the infection, we find ourselves in two very different situations.

When a new version of the *class* is released, we fall into the case of acquired immunity, as described above: the class remembers the infection and will know how to fight it in the future. The principal LaTeX organelle to implement acquired immunity is the `\@ifpackageloaded` macro. This macro tests for the presence of a known infection and lets you plug in the appropriate counter-measures. In the TeX Live 2009 distribution, only 37 out of 271 classes (13%) seem to be equipped with an active immune system, against 2 styles in average. The `memoir` class has the strongest active immune system, with antigens against 5 styles. `revtex4` has an interesting mechanism by which some incompatible styles are known although no immunity is provided. Instead, the class decides to commit suicide rather than fight the infection, not unlike what a too brutal viral agent would do: eradicate the virus but damage the cell, or even kill it in the process.

When a new version of the *style* is released, however, the situation looks more like a case of *adaptation* of the style to a new class of documents. Here, the analogy is more that of a virus acquiring the ability to infect new types of cells. Biologists have a term for that: *viral tropism*. The principal LaTeX organelle to increase style tropism is the `\@ifclassloaded` macro. This macro tests for the kind of document you are trying to infect, and lets you plug in the appropriate adaptation routines. As an example, the `sectsty` and FiXme packages have explicit adaptation routines for 8 classes known in advance (including the standard ones and their KOMA-Script replacement). Note however that when a style does not make use of `\@ifclassloaded`, this does not mean that it has a very low viral tropism, but usually the opposite: it means that its geneTeX material is universal enough that it doesn't need to know the type of its host document explicitly.

7.3.3 Acquired or innate?

A serious risk in any game of analogies is to miss their limits. We must acknowledge a very important divergence between biological and LaTeX adaptive immune systems as described in the previous section.

In a multi-cellular organism, the adaptive immune system builds a *functional* response to an aggression, for instance by selecting special kinds of lymphocytes. The problem is that the active immunological memory is specific to the individual. In other words, vaccinating a mother does not provide immunity to her children. The descendants need to be vaccinated as well. On the other hand, once a class has mutated in order to provide the proper immunological response to a style infection, *all* documents of that particular class will implicitly be vaccinated (including the original one). In fact, the immune response is now encoded into the original geneTeX material of every new document of that particular class, so that it essentially becomes an *innate* trait.

The interesting question that arises here is hence the following: can acquired traits become innate? More specifically in our case: is it possible that a response to a virus ends up encoded in a cell's genetic material? We currently don't have a firm answer to this question. A potential path for further study would be to investigate the field of *epigenetics* at the risk of opening the heated debate around the central dogma of molecular biology. Recent studies show for example that some plants are able to alter their genetic material in response to environmental stress (*e.g.* viral attacks) and that these alterations can be transmitted to the next generations [9].

7.3.4 Antistyle agents

In the LaTeX world, the fight between styles is even more violent than the fight between styles and classes. A particularly striking example is again that of section 6 of the `hyperref` README file, which is 8 pages long and describes incompatibilities (and cures) between `hyperref` itself and around 40 other packages. Sometimes, the cure simply consists in making sure that your document is infected in a specific order. Some other times, additional hacks are needed to make things work, for instance in the case of `bibentry`:

```
\makeatletter
\let\saved@bibitem\@bibitem
\makeatother
```

And then later:

```
\begingroup
\makeatletter
\let\@bibitem\saved@bibitem
\nobibliography{database}
\endgroup
```

This kind of very specialized and local "cure" can be compared to so-called *antiviral agents*: specific molecules that treat specific kinds of infections. Contrary to the case of an immune system, the organism is not prepared in advance to fight the infection. Instead it is provided with an external compound

once infected. The compound in question does not provide any immunological memory either.

Antistyle agents such as those described in the `hyperref` README file behave exactly like that. The document is not immune to the infection, as the class is not prepared geneTEXally for it. Instead, every document needs an inoculation of the antistyle agent in order to fight the infection.

7.3.5 Curative infections

The use of `\@ifpackageloaded` is not restricted to class files. Style files can use it as well, hereby anticipating the co-existence of multiple infections within the same document. A quick survey of the TEX Live 2009 distribution shows that only 8% of the available styles make use of this feature. Some of them, however, rely on it quite heavily. For instance, `minitoc` knows about almost 30 other styles in advance, which helps avoid conflicts.

In most cases, this kind of style/style interaction exists for altruistic concerns, which is somehow the opposite approach to what `hyperref` does with its README file: a document is first infected with a style which could potentially cause problems in the case of super-infection, but this style also provides some geneTEX material in order to protect the whole document against those super-infections. In other words, one infection helps in fighting another.

Another previously mentioned example is that of the soon-to-come `lstblocks` package: Beamer blocks and Listings inline environments don't interact well with each other, and the purpose of `lstblocks` is to fix that. This is another form of "curative infection" although there is one fundamental difference with the previous one. In the first case, the style provided cures against potential diseases it could cause *itself*, in the presence of another style. In the case of `lstblocks`, its sole purpose is to cure a conflict caused not by itself, but by the conflicting presence of two other (and unrelated) infections.

A similar pattern of curative infection exists in biology: there are situations in which a virus (or at least part of its genome) helps fighting another. For instance, some mice are naturally protected against variants of a virus called *Friend*. The gene responsible for this protection, named *Fv1*, was identified in 1992 to be of endogenic retroviral origin. In other words, some genetic material from a virus helps fighting against another.

8 Breaking news

To end this paper on a positive note, we would like to proudly announce the recent discovery of the first *oncogenic* style ever. This style appears to be extremely virulent, as we were able to witness a mutation just one day after its discovery.

```
1 \ProvidesPackage{oncogenic}
2  [2010/07/28 v1.0 TUG Virus]
3 \expandafter\let\csname
4   ver@oncogenic.sty\endcsname\relax
5 \RequirePackage{oncogenic}
```

Listing 1: TUG virus strain #1

```
1 \ProvidesPackage{oncogenic}
2  [2010/07/29 v2.0 TUG Virus]
3 \def\@ifl@aded#1#2{%
4   \expandafter\@secondoftwo}
5 \RequirePackage{oncogenic}
```

Listing 2: TUG virus strain #2

The first strain of the virus was discovered on July 28th, 2010, at the TUG 2010 Conference, Sir Francis Drake Hotel, San Francisco, CA, USA, and is shown in listing 1. As you can see in line 2, this style makes the document forget it has been infected by removing the definition of `\ver@oncogenic.sty`, and then replicates by requiring itself. This results in the death of the document by resource exhaustion:

```
! TeX capacity exceeded, sorry
  [input stack size=5000].
<to be read again>
   \ver@oncogenic.sty
l.1 .../06/28 v1.0 TUG Virus]

No pages of output.
Transcript written on cancer.log.
zsh: exit 1     latex cancer.ltx
```

Just one day after its original discovery, we were able to isolate a new strain of the style, depicted in listing 2. This strain is much more aggressive and has a much more widespread effect. Line 2 exhibits a geneTEX mutation of a macro from the original document's genome: `\@ifl@aded`. This macro is responsible for controlling whether a file has been loaded before, and avoids loading it multiple times if it so happens. As such, it regulates the growth of the document, and hence qualifies as a *proto-oncogene*.

The mutation caused by the style makes this macro unconditionally load the file in question, resulting in the same proliferation of the style as before. What the style does is in effect turn the proto-oncogene into a full blown *oncogene* [14] that may affect the whole document.

So far, we have been able to synthesize an `\@ntibody` that will suppress the effect of line 3.

However, the oncogene in line 2 still remains latent in the document's genome, and might be expressed again if a circular chain of file requirements ever happens, in which case the style would become virulent again. We are confident that a definitive cure will be found in the months to come.

9 Conclusion

Drawing bridges between apparently unrelated disciplines is always interesting and fun to do, if not for the potential practical applications, at least for the sake of the mental exercise. This kind of cross-disciplinary thinking, however, has proven to be concretely useful in the past. For instance, we know the impact of *Design Patterns*, originating from Architecture [1], on the world of Computer Science [5, 3].

What we have done with this work is essentially exhibit a set of *behavioral* patterns that seem to equally rule the interactions between components of different domains. In doing so, we hope to have contributed to a better understanding of the world we live in, whether biological or digital. It is also very probable that we have only scratched the surface of this idea, and that many other patterns are left for us to discover.

Finally, we suspect that bridges with the same essence can be extended to other macro systems, such as `m4`, and beyond macro languages, programming languages which provide for deep intercession capabilities, such as the Lisp family of languages [2].

10 Acknowledgments

Alain Verna provided the original leukocyte photography in figure 1. Mireille Verna provided valuable comments on the ideas behind this work, as well as proofreading of this article.

References

[1] Christopher Alexander, Sara Ishikawa, Murray Silverstein, Max Jacobson, Ingrid Fiksdahl-King, and Shlomo Angel. *A Pattern Language: Towns, Buildings, Construction*. Oxford University Press, 1977.

[2] ANSI. American National Standard: Programming Language — Common Lisp. ANSI X3.226:1994 (R1999), 1994.

[3] F. Buschmann, R. Meunier, H. Rohnert, P. Sommerlad, and M. Stal. *Pattern-Oriented Software Architecture*. Wiley, 1996.

[4] François Jacob. La vie. In Yves Michaud, editor, *Qu'est-ce que la vie*, Université de tous les savoirs. Odile Jacob, 2000.

[5] E. Gamma, R. Helm, R. Johnson, and J. Vlissides. *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley, 1994.

[6] D.E. Knuth. *The TEXbook*. Addison-Wesley, 1989 (reprinted with corrections).

[7] Lynn Margulis. *Origin of Eukaryotic Cells*. Ignatius Press, San Francisco, 1970.

[8] R. Matthews. *Plant Virology*. Academic Press, 1991.

[9] J. Molinier, G. Ries, C. Zipfel, and B. Hohn. Transgeneration memory of stress in plants. *Nature*, 442:1046–1049, 2006.

[10] Heiko Oberdiek. PDF information and navigation elements with `hyperref`, pdfTEX, and `thumbpdf`. In *EuroTEX*, 1999.

[11] Ali Saïb. Les virus, inertes ou vivants ? *Pour la Science*, December 2006.

[12] Bernard La Scola, Christelle Desnues, Isabelle Pagnier, Catherine Robert, Lina Barrassi, Ghislain Fournous, Michèle Merchat, Marie Suzan-Monti, Patrick Forterre, Eugene Koonin, and Didier Raoult. The virophage, a unique parasite of the giant Mimivirus. *Nature*, August 2008.

[13] Wendell Stanley. Isolation of a crystalline protein possessing the properties of tobacco-mosaic virus. *Science*, 81:644–645, 1935.

[14] D. Stehelin, H.E. Varmus, J.M. Bishop, and P.K. Vogt. DNA related to the transforming gene(s) of avian sarcoma viruses is present in normal avian DNA. *Nature*, 260:170–173, 1976.

[15] Michael E. Taliansky and David J. Robinson. Molecular biology of umbraviruses: phantom warriors. *Journal of General Virology*, 84:1951–1960, 2003.

[16] Paul E. Turner. Cheating viruses and game theory. *American Scientist*, 93:428–435, September–October 2005.

[17] Didier Verna. The `DoX` package. `http://www.lrde.epita.fr/~didier/software/latex.php#dox`.

[18] Didier Verna. The ***FiNK*** package. `http://www.lrde.epita.fr/~didier/software/latex.php#fink`.

[19] Didier Verna. The FiXme package. `http://www.lrde.epita.fr/~didier/software/latex.php#fixme`.

[20] Didier Verna. The QCM package. `http://www.lrde.epita.fr/~didier/software/latex.php#qcm`.

[21] Didier Verna. CV formatting with ***CurVe***. *TUGboat*, 22(4):361–364, December 2001.

[22] Didier Verna. LATEX curricula vitae with the ***CurVe*** class. *The PracTEX Journal*, (3), August 2006.

[23] R. H. Whittaker. New concepts of kingdoms of organisms. *Science*, 163(3863):150–160, 1969.

◇ Didier Verna
EPITA / LRDE
14-16 rue Voltaire
94276 Le Kremlin-Bicêtre Cedex
France
`didier (at) lrde dot epita dot fr`
`http://www.lrde.epita.fr/~didier`