Editor: **Grady Booch**
IBM, grady@
computingthehumanexperience.com

The Stories We Tell Ourselves

Grady Booch

THINK BACK TO that tragic scene near the end of James Cameron's *Titanic* when, after the ship has sunk, Rose clings to some flotsam. Her lover, Jack, tries to hang on, only to slip away as hypothermia overcomes him.

From a cinematic perspective, Cameron's storytelling skills were particularly powerful: you could almost feel the frigid water enfold you, and you could easily project yourself into the fierce and intimate solitude of the moment. From an astrophysics perspective, however, that scene was all wrong.

> Jack no longer dies under an alien sky.

Titanic Oversights

When he first saw the movie, Neil deGrasse Tyson noticed that the stars in the background were out of place. Even worse, as he went on to observe, they were sloppily rendered, with the right side of the star field projected as a mirror image of the left. For Tyson, this was a fundamental error of astrophysical fact, made further egregious because this error was in such stark contrast to Cameron's usually obsessive attention to detail. Calling Cameron to task for his imprecision, Tyson pointed out that we knew the exact date and time as well as the latitude and longitude of RMS Titanic at its demise, and therefore could, to a high degree of accuracy, specify what the night sky would have looked like.

For almost a decade, Tyson chided Cameron for his inattention. When the 3D remake of *Titanic* was being developed, he finally got through to the director. Now, when you see the remake, you can be content in knowing that Jack no longer dies under an alien sky.

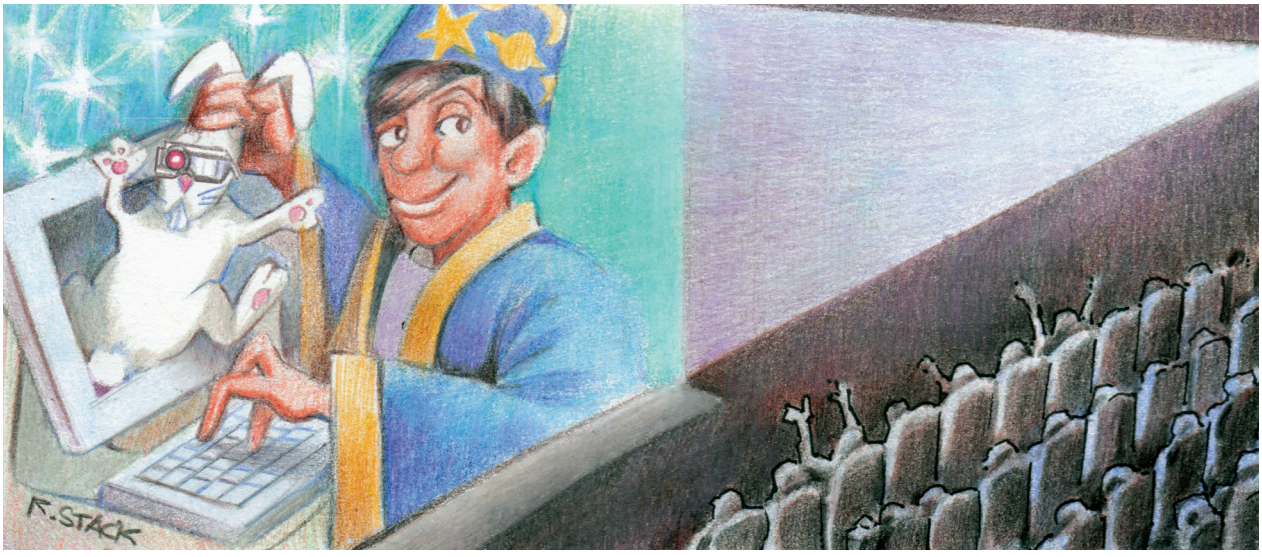
Consider next Steven Spielberg's *Jurassic Park*. Here, we have a techno thriller that teases our imagination with the implications of cloning when carried out by an evil, amoral, and somewhat incompetent genius (to wit, the antagonist, John Hammond). Of course, being a thriller, events quickly spiral downward once the dinosaurs get loose, and it's a fun popcorn escape to get pulled into the journey back to stability. As a computer scientist, however, I found myself laughing at the most inappropriate times.

For a moment, project yourself into the role of Lex, a twelve-year-old girl in the middle of some serious life-threatening chaos. The velociraptors are after you, people are being munched on in the most unsavory fashion all around you, and you have no weapons with which to defend yourself. Stumbling into the park's control room, what do you do? Well, being an expert hacker at such a tender age, you glance at a computer screen and observe, "It's a Unix system! I know this—it tells you everything." Lex quickly hacks into the Jurassic Park Online System (JPOS, as it's called in the movie; you can play with a simulation at www.jpos.jurassicpark.sk), thus setting into motion the events that bring the park back under control.

To quote actor and Twitter superstar George Takei, "Oh, my."

Storytelling involves weaving abstractions about fundamental truths regarding the world and the human experience to entertain and educate. It's the very nature of storytelling to take artistic liberties with reality, as long as doing so advances the story.

0740-7459/14/$31.00 © 2014 IEEE



However, when it comes to movies and television, producers and directors often make computing technology look like it's so far divorced from reality that it's all magic. It doesn't have to be that way.

I respect Tyson's campaign to get Cameron to render the night sky in *Titanic* properly. His rant was a little bit picky but nonetheless, the change that he brought about added a satisfying flourish to an already interesting story. However, when it comes to the profoundly strange ways in which computing is rendered in the media, it would take a legion of Tyson equivalents to bring about some degree of sanity to the entertainment world's portrayal of hardware and software.

Consider another Spielberg movie, *Minority Report* starring Tom Cruise. Here we have a great example of user interface porn: this is what a future human/computer looks like to a great filmmaker. Now, to his credit, Spielberg tried really hard to get it right and he was doing so in a context that far predated our contemporary hyper-connected world of gesture-driven tablets. He assembled a stellar team of futurists, including Jaron Lanier and John Underkoffler (www.wired.com/underwire/2012/06/minority-report-idea-summit) to propose the interface that Cruise, in the role of Capt. John Anderton, would use to investigate precrimes.

In the movie, Anderton uses gestures that any iPad user would find familiar today—for example, pinching and stretching—but the overall effect is unnatural, thanks to the user interface phenomenon known as *gorilla arm*, a term that refers to the uncomfortable feeling you get in your arms when trying to manipulate a large touch screen for a prolonged time. Further stretching the imagination is the fact that this advanced user interface is curiously stupid (the desktop image falls apart when Anderton goes to shake a colleague's hand) and strangely inconsistent (why does Anderton have to pick up a ball that lazily rolls down a tube, a physical object representing the precrime in an otherwise fully virtual environment?).

Additional Techno-Gaffes

Others have collected the more entertaining ways in which movies and television portray computing. The "25 Most Interesting Things That You Learn about Computers in the Movies" has been circulating on the Intertubes for years (here's one reference to the list: www.ariel.com.au/jokes/25_interesting_things_that_you_learn_about_computers_in_the_movies....html). More recently, the good folks at the Den of Geek have generated another list of "The Things That Movies Think Computers Do" (www.denofgeek.com/games/24877/the-things-the-movies-think-computers-do).

Two items on that second list particularly resonate with me: the ability to zoom and enhance any digital image (many of the television crime dramas regularly engage in this magic as a significant plot point) and the fact that software and hardware are portrayed as infinitely compatible (in what universe can a developer working for a cable company develop a virus that can infect

the computers of an alien species as in *Independence Day*?).

James Carter has done a wonderful job of cataloging the use of computer hardware in movies and television on his site Starring the Computer (www.starringthecomputer.com). If you investigate the history of computers in the movies, you'll find that the IBM AN/FSQ-7 looms large in the early history of sci-fi movie making. This computer was actually part of the Semi-Automatic Ground Environment (SAGE) system, and the massive scale of its front panels, together with its propensity for lots of blinking lights, made it perfect for moving making. For you budding filmmakers out there who want to direct your own techno-thriller, you too can rent such ancient computers from any number of places (Woody's Electrical Props in Hollywood is among my favorite in this space).

Software Issues

When it comes to software in the movies, we run into some fundamental problems. Unlike hardware—which blinks, buzzes, and sometimes conveniently blows up in the most spectacular fashion—software is not particularly photogenic. In fact, to the moviemaker, software is positively boring.

John Graham-Cumming has cataloged many examples of source code in the media (http://moviecode.tumblr.com). What you'll find as you peruse his archives is that storytellers tend to grab random code that at least feels like real software to a nontechnical audience. That's understandable: movie making is mostly about feelings, not facts, so if it looks good, that's good enough (which explains a lot about CGI-fueled movies that are heavy on effects but staggeringly thin on plot or character development).

I can accept that illusion, but I'm especially disappointed in a Neil deGrasse Tyson sort of way regarding the way that the process of programming is portrayed: it either looks like the frenzied activity of pizza-fueled post-teens (as shown in Aaron Sorkin's *The Social Network*) or some magical feat that really smart people can do all the time, every time, if and only if they know the proper magic incantations.

The magic of software is nowhere more evident than in movies such as *S1m0ne* or the more recent *Her*, where everything computerish—especially sentient computerish things—is just a simple matter of programming. I accept that late-night pizza-fueled development happens all the time, but as a developer, I also recognize that it's only a small percentage of the reality of sustainable, professional development. Innovative ideas indeed often spring from the solitary work of a small group of programmers, but turning those ideas into industrial strength systems involves a lot of boring yet essential blocking and tackling. Furthermore, as insiders, we know that software-intensive systems aren't magic and that everyone can learn those incantations.

And that's the key to realistic storytelling involving computing. We must certainly celebrate what computing does to advance the human experience, but at the same time, we must demystify the inner workings of the technology itself. To that end, I'm delighted by the Hour of Code project from Code.org because this is part of the journey of making computing an essential life skill. Still, there's much more work to be done. We must continue to tell the story of computing in a way that invites the public behind the curtain and give them the tools to contribute to their own unique life stories.

GRADY BOOCH is an IBM Fellow and one of the UML's original authors. He's currently developing *Computing: The Human Experience*, a major transmedia project for public broadcast. Contact him at grady@computingthehumanexperience.com or on Twitter @Grady_Booch.

Call for Articles

IEEE Software seeks practical, readable articles that will appeal to experts and nonexperts alike. The magazine aims to deliver reliable information to software developers and managers to help them stay on top of rapid technology change. Submissions must be original and no more than 4,700 words, including 200 words for each table and figure.

Author guidelines:
www.computer.org/software/author.htm
Further details: software@computer.org

www.computer.org/software

IEEE Software

See www.computer.org/software-multimedia for multimedia content related to this article.