CS846 Week 9 Summary Review

Amaan Ahmed

Paper: With Great Humor Comes Great Developer Engagement [1]

Problem Being Solved

Modern software development is a massive act of coordination; teams distributed across time zones, collaborating asynchronously through commits, pull requests, and issue trackers. While this setup is efficient, it often results in mental fatigue and emotional disengagement. Developers spend long hours managing bugs, tests, and reviews in isolation, which can lead to engagement fatigue, a gradual loss of motivation and connection to the work. Companies have experimented with different ways to maintain engagement, such as gamification, perks, and hackathons. However, these are surface-level solutions that improve the environment around work rather than within the work. They provide short-term boosts in morale but do not change how developers experience their daily tasks. The authors identify this gap and explore whether humor can be used as an integral part of the software development process to sustain engagement from within.

New Idea

The paper introduces the idea of treating humor as a design choice in software engineering. Rather than seeing humor as an external or informal element, the authors argue that it can be part of engineering practice. Their hypothesis is that *responsible humor*, humor that is inclusive, clear, and contextually appropriate, can enhance engagement, creativity, and team collaboration without reducing professionalism or code quality.

To explore this idea, the authors adopt a mixed-method approach combining qualitative and quantitative research:

1. Three case studies of humorous open-source projects:

- **Faker:** A Ruby library that generates fake test data such as names, addresses, and phone numbers. It includes generators that work upon playful datasets and references, turning repetitive testing into a creative and enjoyable activity. With over 10,000 GitHub stars and 845 contributors, it demonstrates that humor can help sustain community interest.
- **LOLcommits:** A Ruby Gem that takes a webcam photo every time a developer makes a Git commit. It links humor with accountability and team bonding by humanizing version control. Originally a joke, it is now used in hackathons, onboarding, and company demos.
- **Volkswagen:** A Node.js package created as satire of the 2015 Volkswagen emissions scandal. It makes all tests pass automatically by overriding test functions. Although an attempt in satire, it led to the creation of a practical tool, ***is-ci***, which detects CI environments. This example shows how humor can inspire real innovation.

2. A developer survey with 125 participants:

The survey was an online questionnaire that was distributed over multiple online platforms. Participants included professional developers, open-source contributors, and students across various domains. The survey examined how humor appears in software development and how it affects engagement and collaboration. The results were strongly positive. 117 respondents (93%) reacted positively to humor in software, and developers were most comfortable with humor in comments, documentation, and test data. Many emphasized the importance of responsible humor, noting that it should never compromise readability, clarity, or inclusivity. Some teams even mentioned using code reviews to ensure that humorous content remained appropriate.

Positive Points

- I liked that the paper bridges human and technical aspects, as it connects the emotional and social sides of software development with its technical dimensions. It reminds us that programming is not just about code but also about people.
- I thought the paper had a strong methodology. Combining case studies and a developer survey provides both depth and breadth. The case studies give detailed quantitative and qualitative insights, while the survey adds into the evaluation of the human element. The blend of both made the findings feel more grounded and credible.
- The notion of *responsible humor* is a genuinely thoughtful and practical contribution. The authors don't just say "humor is good", they give concrete practical guidelines for how to use it in ways that are clear, inclusive, and appropriate. That makes the paper not just descriptive but useful for teams who want to keep their culture light but professional.
- The paper's writing style reflects its theme and the nature of the content. It is accessible, engaging, and easy to follow, which makes it a fun read.

Negative Points

- There's a bit of a gap around participant diversity. The authors mention that they collected demographic information from survey participants, but they never actually present it; we don't know where the respondents were from, what backgrounds they had, or what kinds of teams they worked in. Since humor is cultural, knowing the participants' backgrounds would help evaluate how general the results are, or at the very least it would have been interesting to see responses from participants from different countries.
- There's also a transparency issue with the survey design. The paper doesn't include any of the survey questions or examples of how they were phrased. That makes it hard to evaluate how well the questions captured developers' real experiences or whether they may have unintentionally guided certain responses. Since the conclusions depend so heavily on that survey, a bit more detail there would really help establish trust in the findings.
- The paper also doesn't really explore in depth, when introducing humor causes problems. It focuses on positive examples, but it only briefly mentions cases where humor might reduce understandability. For instance, when developers use funny

variable or function names. In code where comments are missing, naming is crucial to knowing what a method or variable actually does. If a name is too playful, it can inhibit another developer trying to make sense of the code. There is also the case where another developer may not understand the referenced humor, which would also cause issues. The paper also doesn't talk about cases where humor might backfire socially, like jokes that alienate someone on the team or create discomfort. Including examples of such 'failure cases' could have helped balance the argument and make the notion of responsible humor even stronger.

Future Work

- It would be useful to look at humor across different cultures and team contexts. The paper doesn't break down where participants were from, but that's important since humor is incredibly cultural. So, a cross-cultural study could really deepen our understanding of what 'responsible humor' looks like globally.
- There's also space to study failed or problematic uses of humor, when it makes code harder to read, causes misunderstandings, or unintentionally excludes people. The paper mostly focuses on positives, but seeing where humor backfires would help teams create clearer guidelines for when and how to use it.
- I think it would also be interesting to look at measurable effects, whether humor actually improves collaboration, retention, or even code quality. Something like a control experiment where we have one team that has defined practices or activities, and one team that does not. That kind of data would take the argument from anecdotal to evidence-based, helping teams justify humor not just as a nice idea, but as something that truly supports productivity and engagement.
- I think there's also potential in exploring how AI could assist in generating or curating humor in development tools to increase engagement. For example, imagine something like lolcommits, but instead of just snapping your photo, an AI assistant could automatically write a funny one-line summary or story for your commit message based on the diff or context. Or make a small relevant joke or play a short game like trivia just to keep a developer engaged. It could make routine work more engaging for developers, especially for those that work on their own and may feel isolated.

Rating

4, I like humor.

Discussion Points

- **Do you think humor should be a part of code or software engineering, in general?** One student said humor should definitely be part of software engineering and, in fact, should be everywhere. He recalled that while grading undergraduates' assignments, he and other graders enjoyed reading funny messages left by students, like comments saying, "It's 3 am in the morning, this is the best I can do." This sense of humor, he noted, aligns with the paper's discussion of humor in software engineering. Another student emphasized that humor helps with the work itself, based on her industry experience. During stressful times and tight deadlines, humor improves the atmosphere and productivity. She believes humor is an essential part of software work, especially in small teams. Other students mentioned that humor humanizes the software development process. It strengthens relationships among colleagues, helps break the ice, and lets team members understand each other's dynamics, making people "feel human again." One student expressed concern about humor in the era of AI-assisted software development, noting that AI struggles to create or detect sarcasm. He was unsure how effective AI would be at generating humor. Another student referenced an empirical study about distraction, suggesting that humor is ultimately a form of communication. He restated the importance of communication in software teams, arguing that humans' ability to communicate distinguishes them from other animals. He said humor should remain part of SE as long as its communicative benefits outweigh its potential distractions. He also speculated about developing an AI companion that could tell humorous jokes while developers code.
- **Where is the line between humor that's engaging and helpful and humor that's distracting and unhelpful?** One student said that the ordering is very important, especially when he was debugging and had to go through a bunch of documentation. If the first thing he reads in the documentation is a joke, he would be pretty pissed off. But if the documentations give the solutions first, and then gives him a joke, he can live with that. Especially in the system security domain, all kinds of cryptography and system security terminologies could be boring. And to insert some humor inside the documentation could definitely be helpful. Other students talked about the significance of humor during icebreaking. But one should properly balance them. Otherwise, it will create a kind of impression on others that this person is here for fun, and not serious about the technical work.
- **How could AI tools help in this scope? Not only with humor, but also in finding new ways to keep developers engaged?** Students indicated that humor would be okay if what is acceptable is predefined. One student shared his in-depth insights that humor is not the cause of why developers are more engaged, but more of the other way around – it's the outcome of an already engaging and psychologically safe environment. Developers shouldn't be forced to have humor, it's more of a spontaneous and natural thing. So software teams should keep it as that, natural. One student said that this reminds him of a show: The Office; the boss constantly tries to be funny and gets himself into embarassing situations but that also creates a healthy work environment that performs amazingly. The show also shows instances where the boss gets really serious when it comes to work.
- **Could encouraging humor in software engineering change how new developers are taught? for example, making learning how to code more approachable?** One student indicated that if a teacher teaching a class has humor, this sense of humor could propagate to the students in the class. Other students referred to the earlier discussion and said that enforcing humor is not necessary. If you could create an environment where people feel more humanized, the humor will

come out naturally. One good indicator is that if the developers are working seriously all the time, it should alarm the management team that they are under a certain level of stress. If developers are making jokes sometimes, it may indicate that they are in a healthy work environment where people are not taking their jobs too seriously. The professor concluded that, when he sometimes says something is not funny, what he really means is that it's not to his taste. The best practice of making humor part of software engineering is that we should make it a decoration apart from the technical details. If you give enough technical details already, and then you tell a joke, that's okay. But if people should get your joke first in order to get your technical details, that's probably not good. And you can make fun of yourself, make fun of inoffensive stuff, but never make fun of other people, that's the golden rule.

References

[1] Deepika Tiwari, Tim Toady, Martin Monperrus, and Benoit Baudry. 2024. With Great Humor Comes Great Developer Engagement. In *Proceedings of the 46th International Conference on Software Engineering: Software Engineering in Society (ICSE-SEIS'24)*. ACM, 1–11. doi:10.1145/3639475.3640099