Megan Landau
CSCI 360 Software Architecture
Bowring

Booch - Gotterbarn / Miller Comparison

The articles “Architecture as a Shared Hallucination” by IBM Fellow Grady Booch and “Unmasking Your Software’s Ethical Risks” by professors Don Gotterbarn and Keith Miller were both published in *IEEE Software* in early 2010. While Booch’s article mainly discusses the collaborative generation of software architecture to support the intangible abstractions to model human problems, Gotterbarn and Miller discuss how a software engineer might recognize ethical flaws in their practices.

Both articles highlight the importance of the design phase of software engineering. Booth argues that “The best models have two important characteristics: they codify each design decision in a manner that is predictive, and they serve as the basis against which we can ask questions” (Booch 95). The initial model unites all stakeholders’ visions of the project. Without the model, everyone might have a different idea of what the team is striving for, what the end product is going to look like. Booch declares that a good model will not only provide answers for design problems, but will also serve as a sounding board for more questions to be asked about the project. Gotterbarn and Miller argue that many, many questions must be asked about a possible system, especially during the design phase and especially in relation to ethical issues.

Ethical engineers will look for the human values in the problem, thinking about people who are affected by the solution and the secondary person affected by the users of the system. The discussions about the software will include environmental, cultural, and social effects or consequences of a system, even if it will only affect a small population. Discussions such as these help engineers anticipate all possible circumstances and devise solutions to any issues that may arise. By generating lots of user stories, engineers can get to know who will be using the system, who will be affected by the system and how. They can ensure that the system works for everyone - that it does not leave out groups of people such as the blind or deaf. Gotterbarn and Miller argue that analyzing how the new system will affect, disrupt, or enforce human relationships is important. If engineers can put themselves in the shoes of someone affected by this system, they can ask properly ethical questions and really analyze the problem from every angle.

Gotterbarn and Miller also highlighted the importance of keeping on track with the professional standards identified in the IEEE/ACM Code of Ethics and Professional Practice. The Code points software engineers towards asking the right questions about the ethical aspects of their project. Concern for the health, safety, and welfare of the public is the primary concern for ethical software engineers. They declare that “Software engineering is a service profession. What we develop must be designed for the public good” (Gotterbarn & Miller 13). They also claim that “Even developers with the best intentions have walked into ethical traps,” which, unfortunately, just isn’t acceptable to the public or to fellow engineers (Gotterbarn & Miller 12). If you create software that could possibly endanger a user, worker, passenger, innocent bystander, or anyone at all, then you have not thought it through or tested it to the appropriate levels. Mistakes made by careless engineers could endanger the profession for engineers who do follow ethical guidelines and put the time and effort in to make sure their work is safe. We must recognize and manage ethical issues before they get out of hand or our profession could be called into question which would inhibit advances in science and our civilization as a whole.

Booch’s article differs from Gotterbarn and Millers because he is mostly talking about the significance of a strong, supportive architecture for a project. While the collaborative creation of the software architecture itself is important, it’s significance pales in comparison to the ethical discussions that Gotterbarn and Miller emphasize; this discussion must happen long before the first line of architecture is programmed. However, Booch does raise the point that software architecture models must “present an abstraction with some degrees of freedom yet without ambiguity” (Booch 96). This means that the model must embody the ethical discussion that has already taken place. The users must have some freedom in their choices of using the system, but not one that could endanger others. Just having freedom of choice is an ethical discussion all in itself…

Once the software and procedures of the system have been analyzed from all angles and found to be ethical, then it is time for the abstraction of the system to be turned into a model for all to understand. Booch cleverly states “An architecture therefore serves as a means of anchoring an extended set of stakeholders to a common vision of that world, a vision around which they may rally, to which they are led, and for which they work collectively to make manifest” (Booch 96). I like the idea that we “rally” around a software’s architecture. It is the moment where everyone has agreed to what it will be built upon, and then they can begin to add to its foundation.