Statecharts in the Making: A Personal Account
by David Harel
Article Response

The main idea of this article is, basically, that not everyone is a computer scientist, and as computer scientists it is our obligation to understand this commonly misconception and to do our mightiest to evade participation in it. The article begins with Harel explaining how the whole idea of his language of statecharts came about. He continues to discuss the fundamentals of his language, statecharts, and, more importantly, how the language can be applied to real world practice. Finally, Harel summarizes his article with a brief conclusion. In short, the main idea that this article brings to it’s readers is how computer science can be applied to practice, and how computer scientists need to learn to communicate their ideas to other areas of profession.

Harel opens his article by describing the toils it took to figure out his language of statecharts. He begins by explaining how a military company, IAI, had recruited him into this project, which was an aircraft of some kind. The project was slowly creeping along, and even worse, after an intense questioning session between the engineers developing the aircraft and Harel, it became evident that the entire design was so overly complicated that even the engineers who designed it couldn’t explain it’s behavior under certain conditions. Harel decided to take the whole thing back to the drawing board. After many failed attempts using conventional methods, a language was finally developed that allowed Harel’s computer science mind to communicated clearly with the engineer’s minds. This language was later dubbed statecharts.

Harel then delves into some great detail on the concepts of his language, and how these concepts can be applied to real world projects. The language is basically combination of state diagrams and flow charts, hence the name statecharts; but why not chartdiagrams? It doesn’t matter. What’s important is that Harel founded a brilliant method of mental interaction between spanning professions, and that this method had proven itself to be invaluable to a myriad of practices.

Finally, Harel concludes his article with a short summary of everything he had written about. He points out, rather concisely, the main statement of his article, also, saying that “too much computer science research on languages, methodologies, and semantics never finds it’s way into the real world, even in the long term, because these issues do not get sufficient priority”. Basically, the field of computer science is a bounty of knowledge and incredible ideas, that if applied to real world problems could provide real world solutions; however, the ideas cannot be understood unless they are properly communicated to the real world from computer science, via computer scientists.