□ 기획연재 □

컴퓨터 과학 산책(25)

컴퓨터 과학을 위한 명심보감

부산대학교 조환규*

긴 말보다 짧은 말이 훨씬 효과적이다. 비용과 성과면에서도 효과적이고, 실제 절대적인 면에서도 효과적이기도 하다. 링컨이 행한 그 유명한 "인민에 의한, 인민을 위한…"하는 연설 바로 전에 당대 최고의 연설가라는 한 찬조 연설가가 눌변의 링컨을 보조하기 한시간 동안에 걸친 열정의 연설을 풀어놓았다. 하지만 그 당대 최고의 변사가 행한 연설은 아무도 기억하지 못하고 역사속으로 완전히 사라지고, 짧은 링컨의 경구만이 살아있다. 짧은 경구나 격언이 강력한 이유는 거개의 경구가 역설적인 표현이나 댓구로 구성되어 있어서 구조적으로 기억하기가 쉽다는 것이며, 또한 그 character의 수도 작기 때문에 암기하기에도 간편하기 때문이 아닌가 한다. 이러한 경구중에서 가장 유명한 것은 머피의 법칙(Murphy's law)일 것이다. 필자가 학생시절 얻어들은 몇개의 머피의 법칙은 수십권의 책에서 abstraction시킨 구구한 설명보다도 적확하고, 강렬한 것이었다. 예를 들면 아래의 교수들에 관한 커-마틴의 법칙은 교수들의 특성과 사회적, 행동적 특성을 분석한 어떤 두터운 책보다 간결하면서도 예리하게 잘 꼬집고 있다.

커-마틴의 법칙

(1) 자신들의 문제를 다룰 때에 가장 보수적인 사람들은 교수진이다.

(2) 다른 사람의 문제를 다룰 때에 가장 진보적인 사람들은 교수진이다.

이 글에서는 그간 필자의 눈에 보인 머피의 법칙류와 같은 것들 중에서 컴퓨터에 관련된 것들에 대해서 소개하려고 한다. 어떤 국내 번역서에 소개된 머피의 법칙중에서 컴퓨터에 관한 것에는 상당히 잘못 번역된 것도 있어서 이를 바로 잡는 의미도 있다고 하겠다. 하여간 필자는 수업시간에 자주 이러한 짧은 경구를 소개하여 딱딱한 분위기도 해소하고, 부족한 강의실력을 때우기도 한다. 우리 독자 여러 제위께서는 이 글을 보시고 이 중에서 쓸만한 것이 있으면 그 발전형을 만들어 봐도 좋을 것이다. 하여간 머피의 법칙이 아니라 우리 현실에 좀 더 합당한 새롭고 참신한 우리 식의 법칙을 만들어 나가야 할 것이다.

1. 그린의 법칙 :

컴퓨터는 당신이 하고 싶은 일을 하는 것이 아니라 당신이 시킨 일을 한다.

2. 서틴의 법칙 :

아무짝에도 쓸모없는 컴퓨터 작업이야말로 가장 즐거운 작업이다.

3. 맥크리스티의 공리 :

소프트웨어의 버그는 그 소프트웨어가 완전히 구식이 되면 박멸된다.

4. 스타인백의 법칙 :

대응방법을 모르는 에러는 결코 테스트되어서는 안된다.

5. 레오베리져의 공리 :

메모리에 기억시킨 경우에는 그 메모리한 장소를 반드시 다른 장소에 메모리시켜두어야만 한다.

6. 프로그래밍 법칙 :

6.1 지금 사용하는 프로그램은 항상 시대

* 정회원

에 뒤떨어진 것이다.

6.2 프로그램이 쓸만 해지게 되면, 자연히 그것은 다른 것으로 바뀌게 된다.

6.3 프로그램이 쓸데없어지면, 누군가가 그것을 다큐먼트화하라고 한다.

6.4 사용하는 프로그램의 크기는 메모리가 버틸 수 있는 한도까지 최대한 늘어난다.

6.5 프로그램의 복잡도는 그 프로그래머가 버틸 수 있는 한도까지 최대한 복잡해진다.

7. 트라우만의 법칙 :

가장 치명적인 에러는 프로그램이 상품화된지 6개월이 지나서야 발견된다.

8. 길브의 신뢰성의 법칙 :

8.1 컴퓨터를 믿을 수는 없다. 그러나 인간은 더욱 더 믿을 수 없다.

8.2 인간의 신뢰성에 의존하는 시스템은 결코 신뢰할 수 없다.

9. 브룩의 법칙 :

소프트웨어 개발 프로젝트중에 사람을 더 투입하면 개발은 더 늦어진다.

10. 골럽의 프로젝트의 법칙 :

10.1 프로젝트 목표의 모호함은 경비산출시에 난처한 일이 일어나는 것을 피하기 위해서이다.

10.2 개발 프로젝트가 복잡하면 예상보다 3배의 시간이 걸린다. 그러나 계획이 면밀하면 2배밖에 걸리지 않는다.

10.3 프로젝트의 궤도수정을 위한 노력은 시간이 지남에 따라 기하급수적으로 늘어난다.

11. 미첼의 프로젝트 미팅의 법칙 :

11.1 아무리 단순한 문제라도 자주, 길게 회의를 하면 해결불가능한 문제가 된다.

11.2 일단 프로젝트를 망쳐놓을 방법이 제안되면 그것이야말로 가장 적절한 해결책이라고 모두 다 믿게 된다.

11.3 프로젝트가 망쳐졌을 때 그 계획에 가장 먼저 대찬성한 사람들은 꼭 이런 말을 한다. "내가 염려했던 점을 지적했어야 했는데.."

12. 분과위원 선출에 관한 마틸다의 법칙 :

자리에 뜨면, 선출된다.

13. 자력의 법칙 :

새로운 시스템을 구입하는 것이 더 경제적일 경우에 사람들은 이전의 시스템을 고쳐 쓴다.

[발전형] 간단히 수리해서 쓰는 편이 나을 경우에는 틀림없이 최신기종을 들여 놓는다.

14. 루바르스키의 사이버네틱 곤충학 :

마지막으로 발견된 버그는 아직도 프로그램에 한 마리가 더 있음을 나타내는 증거다.

15. 로이 칼슨의 법칙 :

코딩을 빨리 할수록, 프로그램은 길어진다.

16. 피터 하퍼너의 법칙 :

해야 할 일을 글로 쓸 수 없다면, 결코 프로그램을 할 수 없다.

17. 노옴 쉬라이어의 법칙 :

코드와 컨멘트(comments)가 일치하지 않으면 둘 다 틀린 것이다.

18. 데이빗 스토어의 조언 :

컨멘트 문장으로 디버깅하지 말고, 코드로서 디버깅하라.

19. 커니간의 법칙 :

새로운 사용자는 새로운 버그를 찾아낸다.

20. 프로그램 관리자의 모토 :

고치지 않으면, 고장나지 않는다. 그런데로 돌아가면 고치지 마라.

21. 리차드 페얼리의 법칙 :

모든 소프트웨어/하드웨어 시스템의 구조는 그것을 만든 기관의 구조를 닮게 된다.

22. [무명씨]

보고서가 끝나지 않으면 결코 프로젝트가 끝날 수 없다.

23. 프레드 브룩의 경험 :

좋은 판단은 경험으로부터 오고, 좋은 경험은 나쁜 판단으로부터 온다.

24. 리차드 힐의 조언 :

수작업으로 잘 할 수 있는 컴퓨터에게 시켜서는 안된다.

25. 레리 번스타인의 법칙 :

프로토타입핑(Prototyping)은 전체 공정을 40%까지 앞당긴다.

[변형] 4인치 천체 망원경을 만든 후에 6인치 천체 망원경을 만드는 일은 처음부터 6인치를 만들려고 하는 것보다 빠르다.

26. [송우길의 법칙]

프로그램 데모가 성공할 확률은 그것을 쳐다보는 사람의 지위에 반비례한다.

[발전형 1] 그 확률은 쳐다보는 사람의 수에도 반비례한다.

(아무도 보지 않으면 반드시 돌아간다.)

[발전형 2] 그 확률은 그 프로그램이 개발된 장소와 데모 장소간을 잇는 직선거리의 제곱에 반비례한다.

27. [조환규 : 학부생 프로그램 과제의 법칙]

모든 과제는 주어진 기간에 상관없이 마감 3일 전부터 시작된다.

[발전형] 프로그램 과제의 시간을 넉넉히 주는 것은 하지 말라는 이야기와 같다.

28. [조환규 : 보고서 감별의 법칙]

28.1 copy한 리포터가 항상 먼저 제출된다.

28.2 copy한 리포터는 항상 다채롭다(colourful).

28.3 사용한 폰트의 종류와 copy한 정도는 비례한다.

29. [대학원생 밤샘의 법칙]

29.1 일이 꼬이면 밤샘작업을 하게 된다.

29.2 밤샘을 하면 일은 꼬인다.

29.3 밤샘의 성과는 야식의 양과 식사시간에 반비례한다.