Costina Horia
An 4 IS

S.O.L.I.D Principiile programarii orientate pe obiecte

S.O.L.I.D este un acronim de la primele cinci principii de design software ale programarii orientate pe obiecte create de Robert C. Martin cunoscut si sub numele de Uncle Bob (Brandul pe care il conduce).

Aceste principii au fost create pentru a defini un mod de lucru in a obtine un cod usor de mentinut si usor de extins. Deasemenea ajuta developerii sa evite buguri si in cazul in care este necesar usureaza refactorizarea codului.

- S – Single-responsibility principle
- O – Open-closed principle
- L – Liskov substitution principle
- I – Interface segregation principle
- D – Dependency inversion principle

SRP: O clasa trebuie sa aiba o singura responsabilitate. Orice clasa complexa ar trebui divizata in mai multe clase ce au scop unic. Acest lucru face codul mai usor de inteles.

OCP: Orice modul, clasa, metoda, aplicatie trebuie sa fie deschisa pentru extindere dar inchisa pentru modificari. Daca fiecare modul, clasa sau metoda sunt create intr-un mod modular poti schimba comportamentul sistemului fara a schimba codul sursa.

LSP: Se refera la faptul ca o subclasa trebuie sa extinda clasa de baza ci nu sa o inlocuiasca. Prin alte cuvinte o clasa de baza poate fi inlocuita oricand de o clasa derivata din aceasta fara a se schimba comportamentul sistemului.

ISP: Un client nu ar trebui sa depinda de metodele pe care nu le foloseste. De cele mai multe ori interfetele specifice sunt mai bune ca cele generale. Prin aceasta procedura se evita crearea metodelor de dimensiuni mari despre care se stie ca sunt greu de inteles.

DIP: Modulele high-level nu trebuie sa depinda de modulele low-level. Cu alte cuvinte, dependinta dintre obiecte trebuie evitata folosindu-se abstractizarea.

Adesea pe internet se pot gasi si articole care condamna acest mod de lucru spunand ca este unul ineficient si ca ingreuneaza intelegerea programarii orientate pe obiecte pentru programatorii incepatori. Probabil ca acest lucru este alimentat de dorinta de nou a programatorilor actuali avand in vedere ca principiile S.O.L.I.D dateaza de peste 30 de ani.

In concluzie, principiile S.O.L.I.D nu garanteaza un cel mai bun design al unui cod intr-un limbaj orientat pe obiecte, dar daca este aplicat corespunzator poate ajuta in rezolvarea unor probleme. Daca este aplicat corect te poate ajuta sa implementezi o arhitectura software ce este usor de extins si de mentinut.