SOLID:

- **S** - The **S**ingle Responsibility Principle (SRP)
  - Cada módulo do software só deve ter uma e uma só razão para mudar.
- **O** - The **O**pen-Closed Principle (OCP)
  - Um módulo deve ser aberto para extensão, mas fechado para modificação
- **L** - The **L**iskov Substitution Principle (LSP)
  - Subclasses devem ser subtituíveis pelas suas classes-base
- **I** - The **I**nterface Segregation Principle (ISP)
  - Muitas interfaces específicas de clientes são melhores que uma interface de propósito geral
- **D** - The **D**ependency Inversion Principle (DIP)
  - Módulos de alto nível não devem depender de módulos de baixo nível. Ambos devem depender de abstrações.
  - Abstrações não devem depender de detalhes. Detalhes devem depender das abstrações.

Princípios da Arquitetura em Pacotes (packages):

- The Release Reuse Equivalency Principle
  - Código não deve ser reutilizável através da cópia de uma classe para outra
  - Só componentes lançadas através de um sistema de tracking podem ser reutilizáveis
- The Common Closure Principle (CCP)
  - Classes que mudam juntas, devem estar juntas
- The Common Reused Principle (CRP)
  - Classes que não são reutilizáveis em conjunto não devem estar no mesmo grupo (package)

Princípios do Agrupamento de Pacotes (packages):

- The Acyclic Dependencies Principle (ADP)
  - As dependências entre pacotes não podem formar ciclos
- The Stable Dependencies Principle (SDP)
  - Depender da direção da estabilidade. Precisa-se de packages "fáceis de mudar"(instáveis) para alterar facilmente o software
- The Stable Abstractions Principle (SAP)
  - Pacotes estáveis devem ser abstratos