A Course on C++

whoami

- Dr. Vishwa Kiran S
  - PhD
  - M.Tech in Computer Science and Engineering
  - B.E in Electronics and Communication Engineering
  - 19+ Years of Experience
  - Worked for
    - Mistral Solutions Pvt Ltd
    - Waveaxis Technologies Pvt Ltd
  - Handled Corporate training for
    - Nokia, Samsung, Cisco, Texas Instruments, Siemens, L&T infotech, Sasken, Honeywell, Western Digital (Japan)
    - C, C++,Java, Python QT, Linux, Device Drivers, Symbian, Meego, Android, Beagleboard, Rasperry Pi
  - Publications
    - 3 IEEE conference paper
    - 1 Springer Journal , 1 InderScience Publisher
  - Currently associated with
    - BMS Institute of Technology & Management
    - Pushkala Technologies Pvt Ltd
    - Pytriot Solutions LLP

Reusability

Reusable Mechanism

- Containership
  - To object code level
- Inheritance
  - To object code level
- Templates
  - To source code level

Containership

- Class string{
  private:
  char * str[100];
  public:
  str(char *s){
  str = s;
  }
  void get(){

  }
  };

Class strlist{
private:
string s[100];
int c;
public:
strlist(){
c = 0;
}
};

Inheritance

- Class plot{
  int wid,len;
  public:
  plot()
  {
  }
  void get_plot()
  {
  }
  };

Class house :public
plot{
int no_floors;
int rooms;
public:
house()
{
}
void get_home()
{
}
};

Inheritance contd..

- object size of derived object
- Which function gets called?
  - When both the class having the same function name

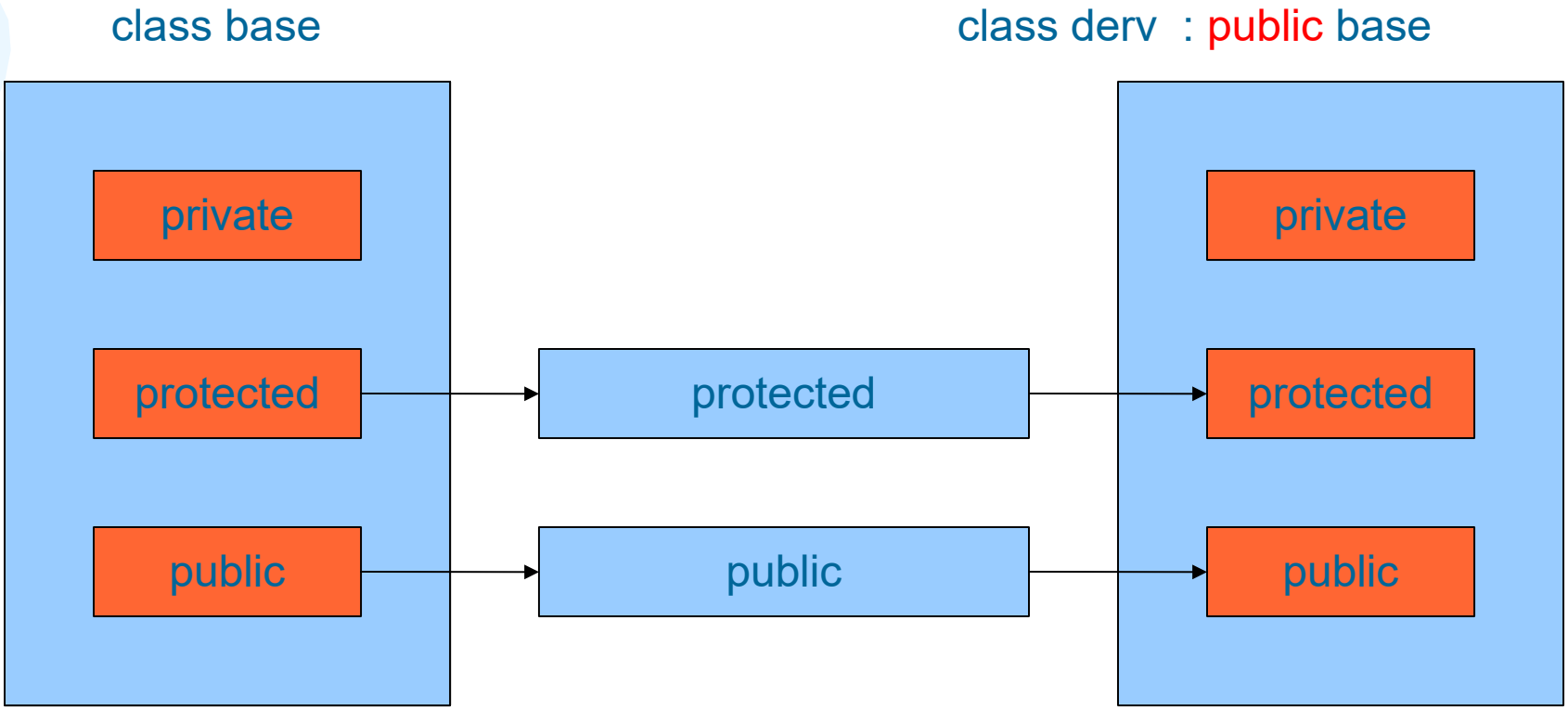
Purpose of inheritance

- Using the existing feature of the class
- To add the new feature
- To override the old feature
- To combine old and new feature

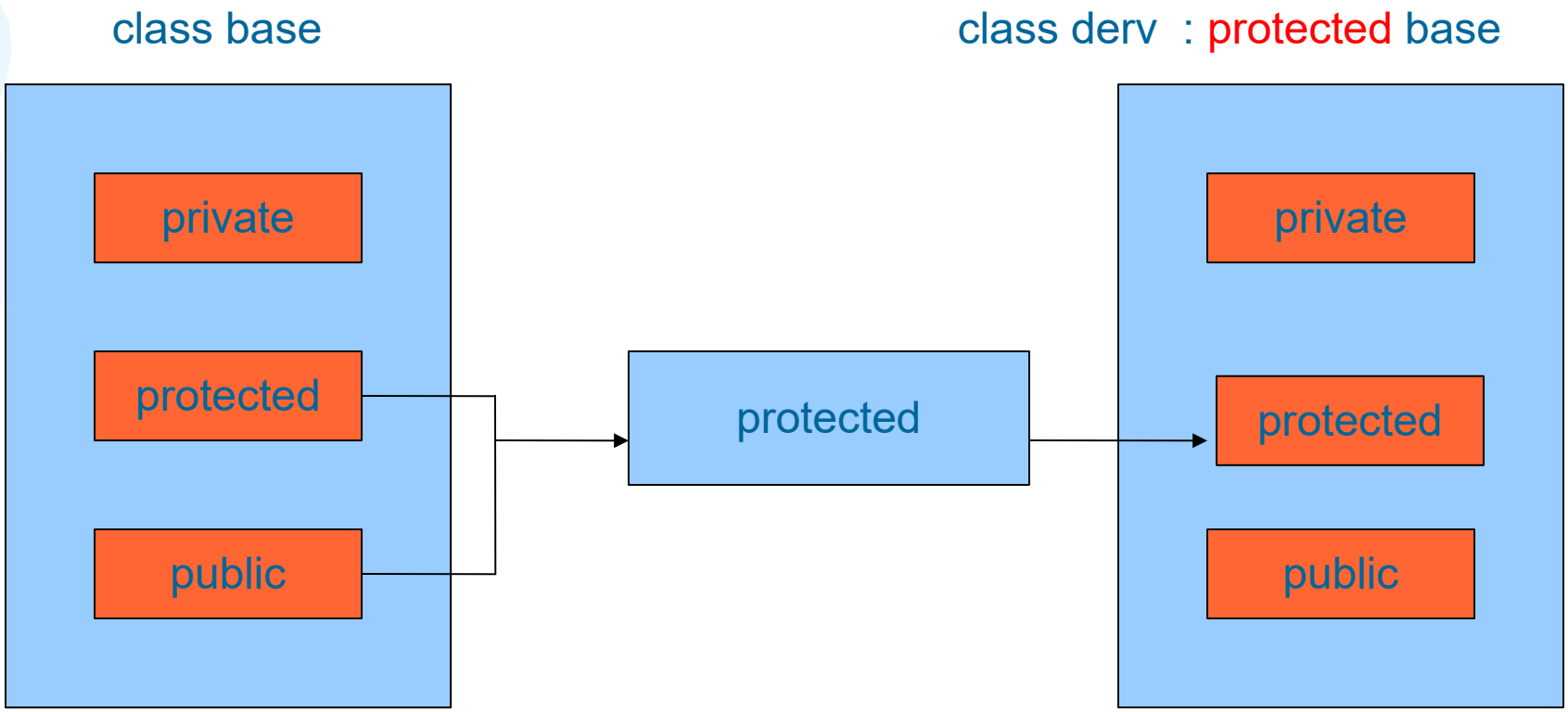
Types of Inheritance I

- Public inheritance
- Protected inheritance
- Private inheritance

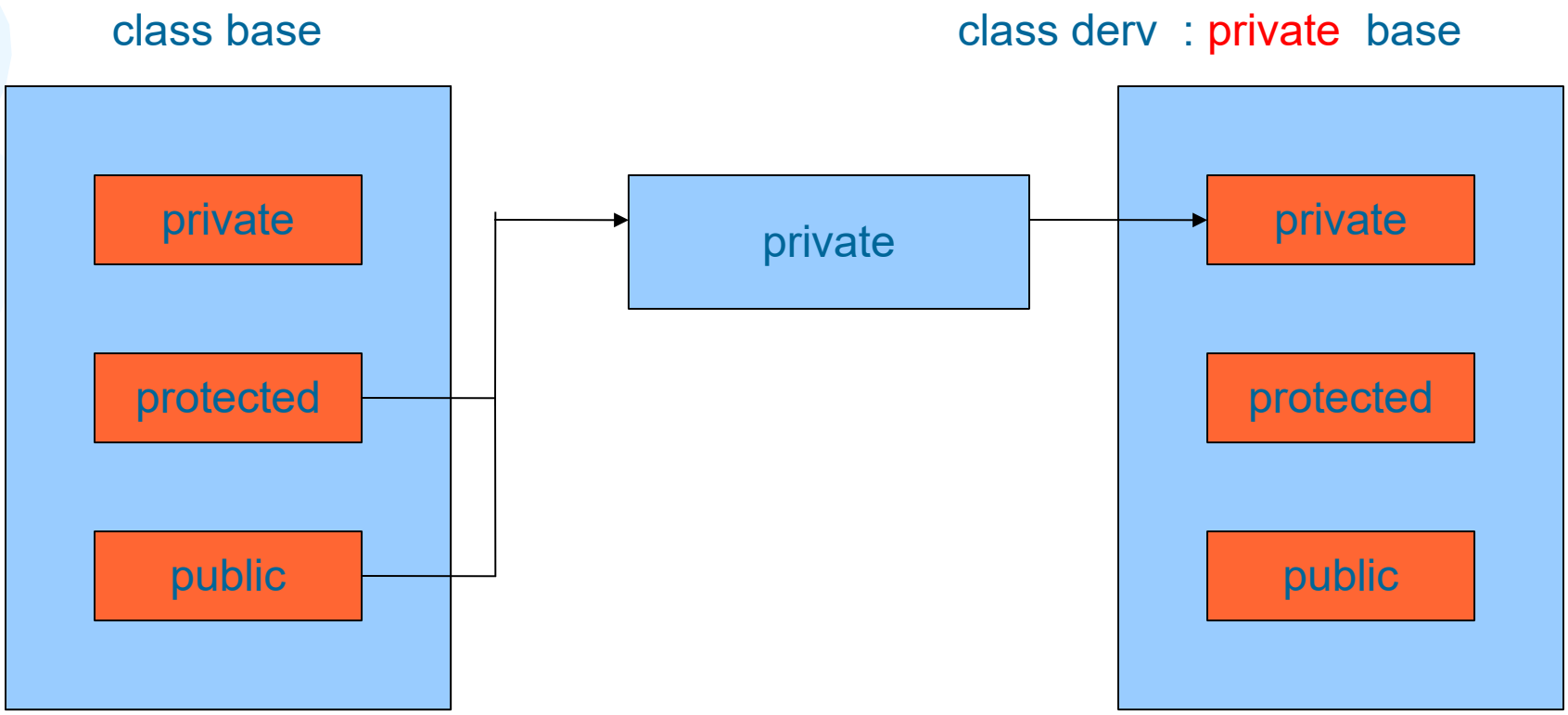
public inheritance



protected inheritance



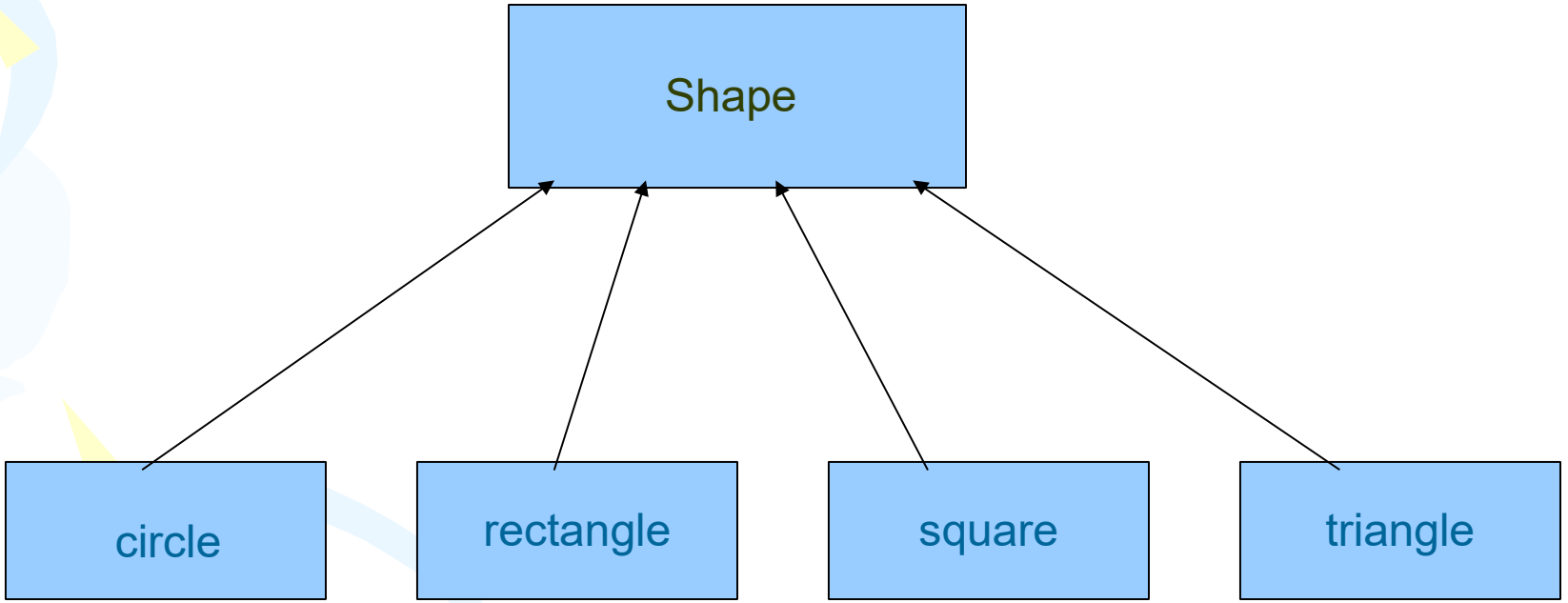
private inheritance



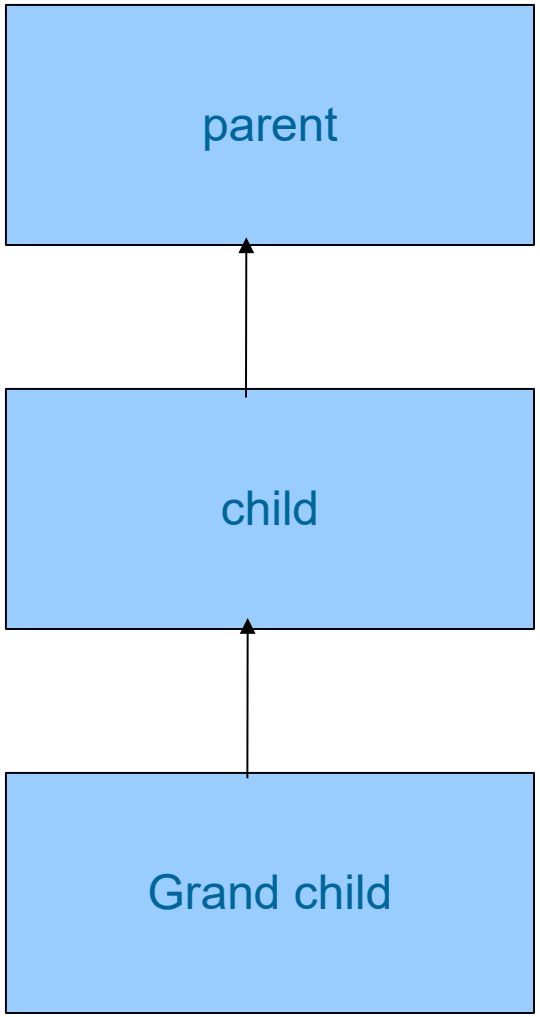
Types of Inheritance II

- Single Level Inheritance
- Multi Level Inheritance
- Multiple Level Inheritance
- Hybrid Inheritance

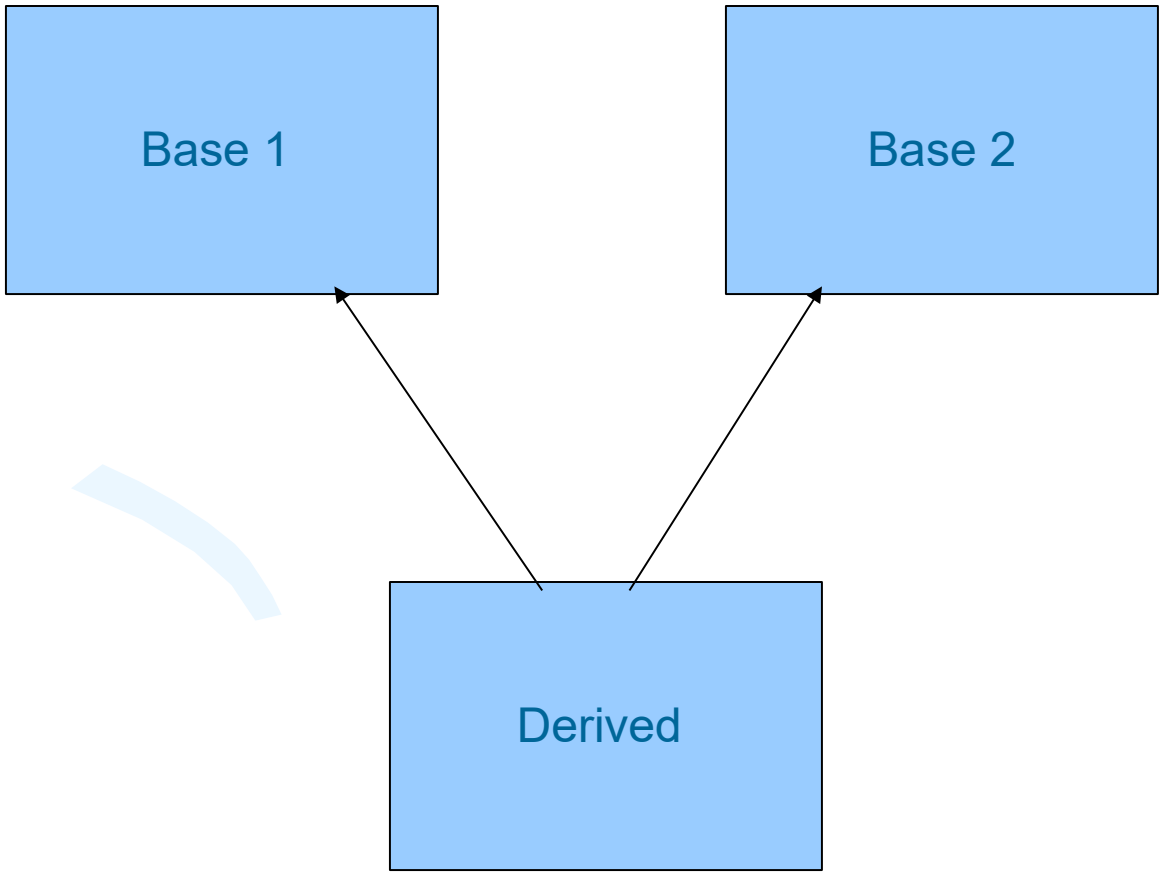
Single Level Inheritance



Multi Level



Multiple Inheritance



Hybrid Inheritance