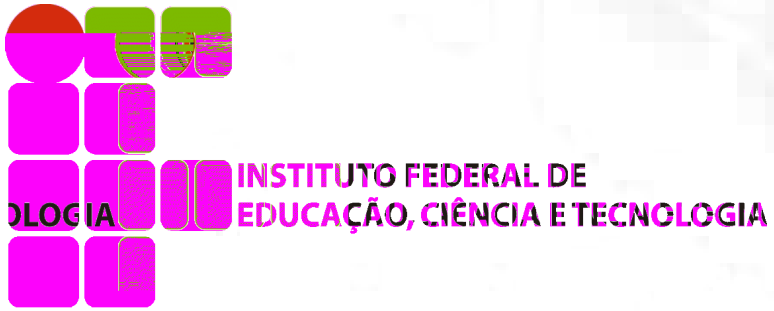
# *Factory Method*

**Sandro Santos Andrade**

**Instituto Federal de Educação, Ciência e Tecnologia da Bahia**
**Departamento de Tecnologia Eletro-Eletrônica**
**Graduação Tecnológica em Análise e Desenvolvimento de Sistemas**



# Factory Method

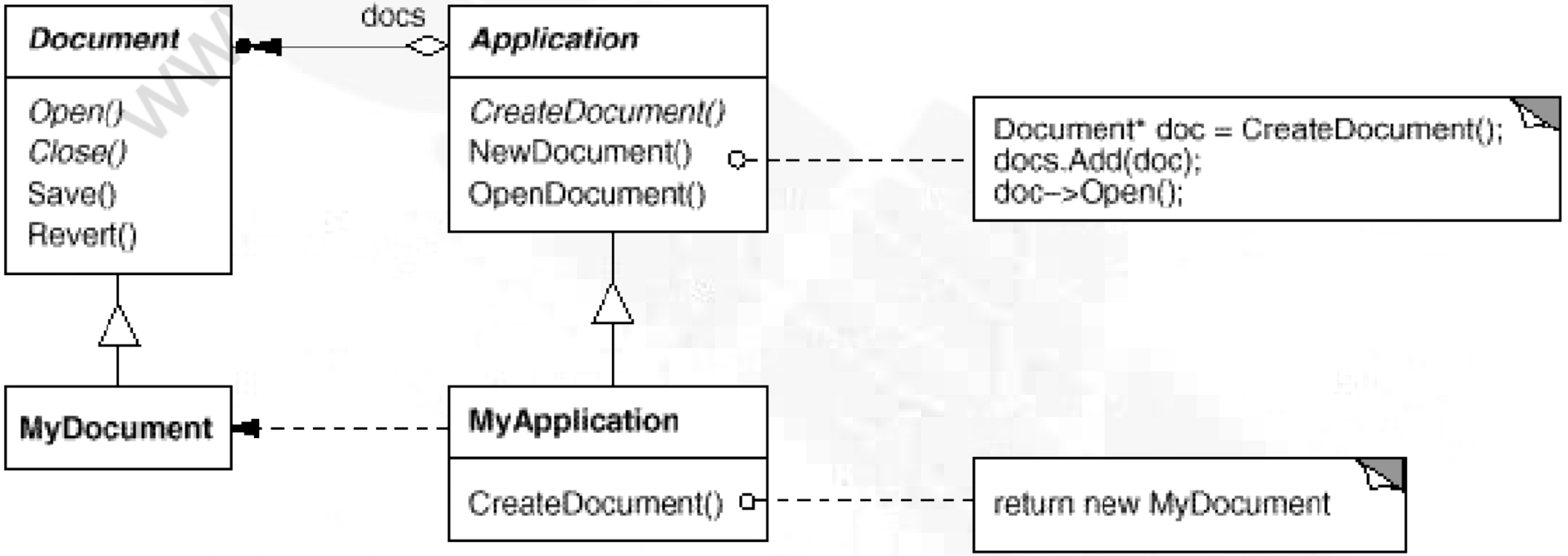
- 
  - 
- 
- 
  - *Application Framework*

    *Application* *Document*
  - 

    *DrawingApplication* *DrawingDocument*
  - *framework*

*Virtual Constructor*

# Factory Method




- *Application*
  *CreateDocument*
  *Document*

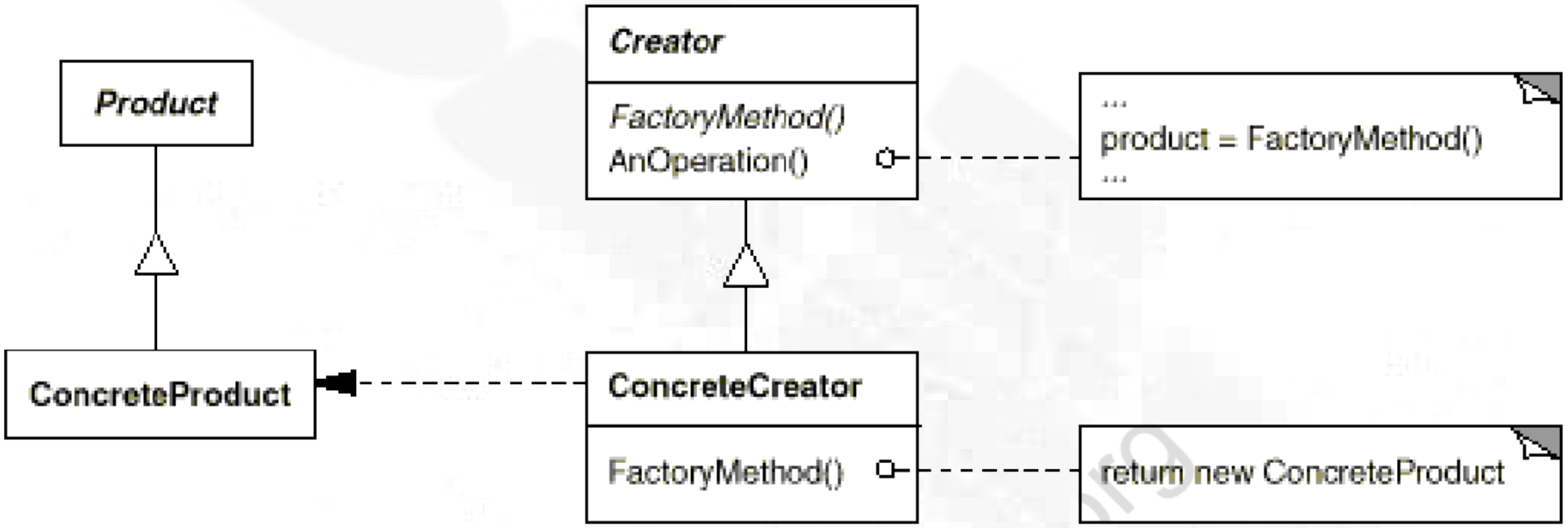
# Factory Method

- 
  - 
  - 
  - 

*helper*

*helper*

# Factory Method



# Factory Method

- 
  - *Product*
    *factory method*
  - *ConcreteProduct*
    *Product*
  - *Creator*
    - *factory method*
      *Product.*
    - *factory method*
  - *ConcreteCreator*
    *factory method*
    *Product*
    *ConcreteProduct*

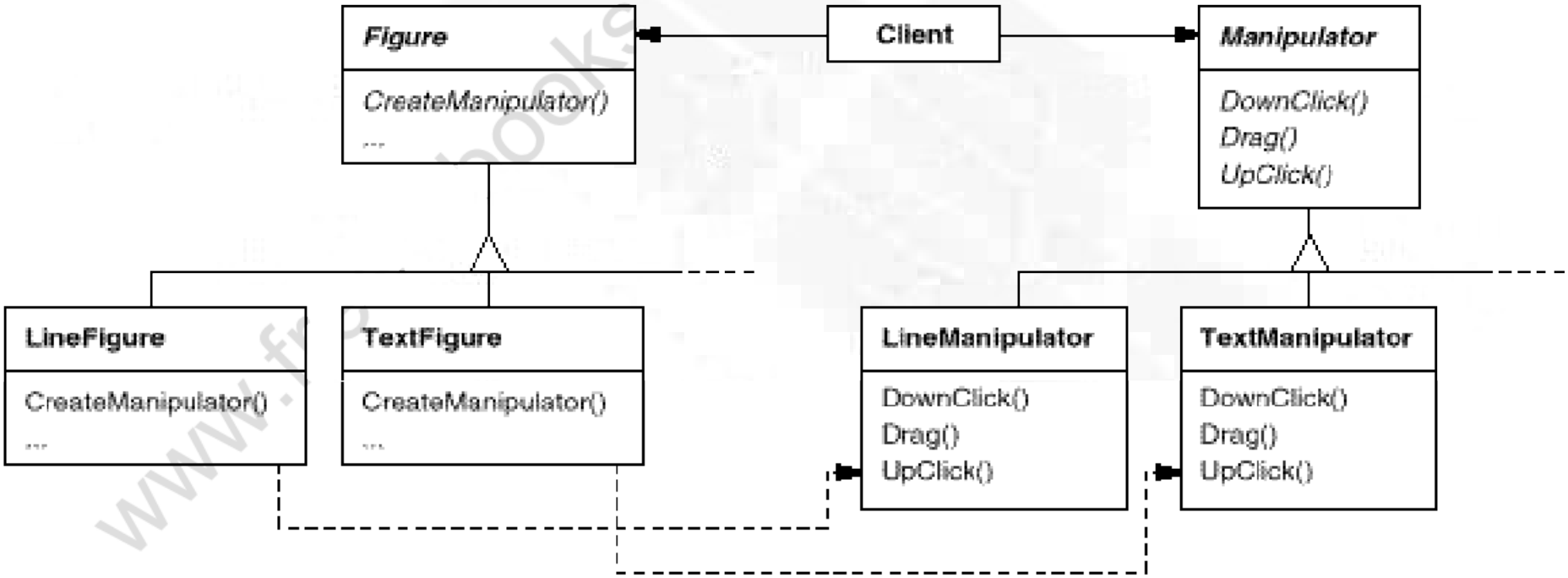
# Factory Method

-
  - *Creator*
    *factory method*
    *ConcreteProduct*

# Factory Method

- 
  - *Product*
  - *Creator*
  - *hooks* *factory method*
    *createFileDialog()* *default*

# Factory Method



# Factory Method

- 
  - 
    - *Creator*
      - 
        *factory method*
    - *Creator* *default* *factory method*
      - *ConcreteCreator* *factory method*

      *factory method*
  - *Factory methods*
    - *factory method*

# Factory Method

- *Factory methods*

```
class Creator {
public:
    virtual Product* Create(ProductId);
};

Product* Creator::Create (ProductId id) {
    if (id == MINE)  return new MyProduct;
    if (id == YOURS) return new YourProduct;
    // repeat for remaining products...

    return 0;
}
```

```
    if (id == YOURS)  return new MyProduct;
    if (id == MINE)   return new YourProduct;
        // N.B.: switched YOURS and MINE

    if (id == THEIRS) return new TheirProduct;

    // call Creator::Create(id) if all else fails
}
```

# Factory Method

- 
  - 
    - *factory method* **classe** *ConcreteCreator* *run-time*
    - *Creator*
    - *factory method* *ConcreteCreator* *Creator* *lazy initialization*
    - *templates*
    - *factory methods*
      - *Class *doMakeClass()* *Class*

# Factory Method

```
class MazeGame {
public:
    Maze* CreateMaze();

// factory methods:

    virtual Maze* MakeMaze() const
        { return new Maze; }
    virtual Room* MakeRoom(int n) const
        { return new Room(n); }
    virtual Wall* MakeWall() const
        { return new Wall; }
    virtual Door* MakeDoor(Room* r1, Room* r2) const
        { return new Door(r1, r2); }
};
```

```
Maze* MazeGame::CreateMaze () {
    Maze* aMaze = MakeMaze();

    Room* r1 = MakeRoom(1);
    Room* r2 = MakeRoom(2);
    Door* theDoor = MakeDoor(r1, r2);

    aMaze->AddRoom(r1);
    aMaze->AddRoom(r2);

    r1->SetSide(North, MakeWall());
    r1->SetSide(East, theDoor);
    r1->SetSide(South, MakeWall());
    r1->SetSide(West, MakeWall());

    r2->SetSide(North, MakeWall());
    r2->SetSide(East, MakeWall());
    r2->SetSide(South, MakeWall());
    r2->SetSide(West, theDoor);

    return aMaze;
}
```

# Factory Method

```
class BombedMazeGame : public MazeGame {
public:
    BombedMazeGame();

    virtual Wall* MakeWall() const
        { return new BombedWall; }

    virtual Room* MakeRoom(int n) const
        { return new RoomWithABomb(n); }
};
```

```
class EnchantedMazeGame : public MazeGame {
public:
    EnchantedMazeGame();

    virtual Room* MakeRoom(int n) const
        { return new EnchantedRoom(n, CastSpell()); }

    virtual Door* MakeDoor(Room* r1, Room* r2) const
        { return new DoorNeedingSpell(r1, r2); }
protected:
    Spell* CastSpell() const;
};
```

# Factory Method

- 
  - *toolkits* *frameworks*
  - 
  - *Framework*
  - *factory method*
    *proxy*

# Factory Method

- 
  - *Abstract Factory*
    *Factory Methods*
  - *Factory methods*
    *Template methods*
  - *Prototypes*

*Creator*

*factory methods*

# *Factory Method*

**Sandro Santos Andrade**

**Instituto Federal de Educação, Ciência e Tecnologia da Bahia**
**Departamento de Tecnologia Eletro-Eletrônica**
**Graduação Tecnológica em Análise e Desenvolvimento de Sistemas**