F01 PaPdres õ ojPesoPdt 5ort

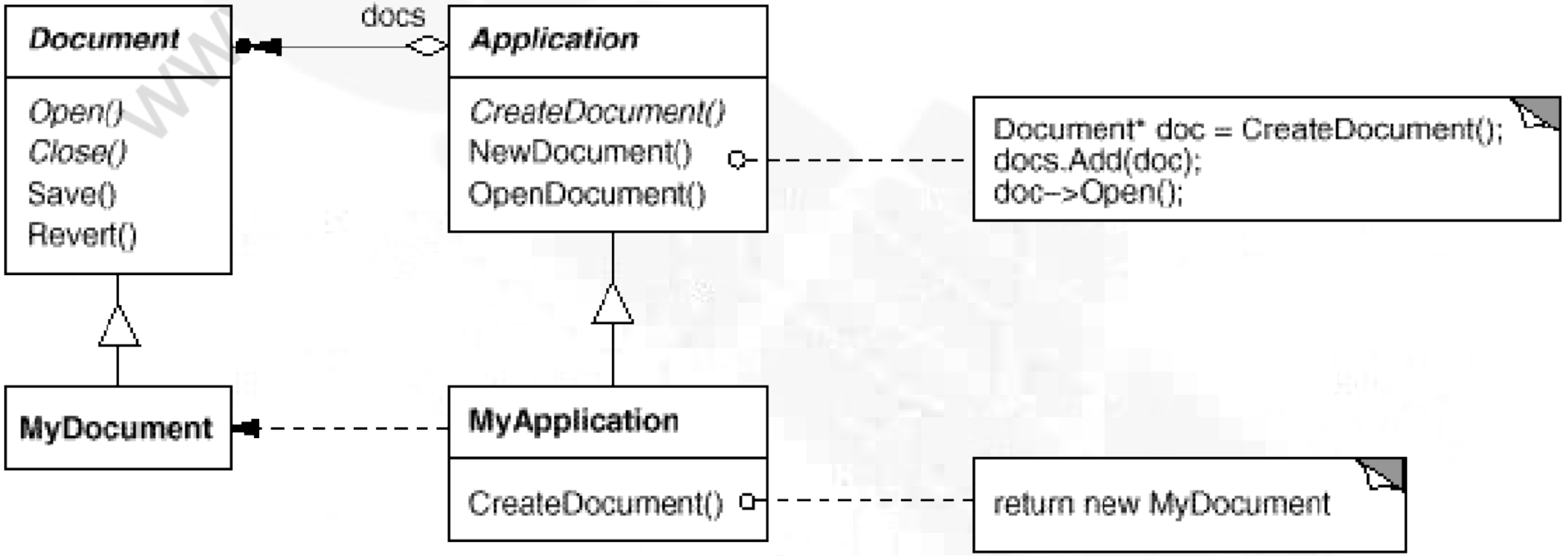
@aP *Factory Method*

jriestrieseof bur posequo

INSTITUTO FEDERAL DE
EDUCAÇÃO, CIÊNCIA E TECNOLOGIA

- d d t ´ : j b t m
  - c o . b i b E D r P i m o r ¸ o P ´ r o r P ¸ d r s x t P e s o P E ¸ D P u 5 o r t P ´ t o q ¸ D P e s o b - r i e s t P e D o P j e D u r ¸ D P T j j o j P e s o ¸ b e s r ¸ D P e D r P e s o P ¸ T j j o P b i j m r i ¸ b n D
- M r ¸ D u q ¸ D P ¸ t i v o ¸ b e s t P ¸ t ¸ D t m P *Virtual Constructor*
- U t m b e D r s x t m
  - g ¸ D P *Application Framework* P ´ r o r P ¸ t i j m o e D s x t P e s o P j b j m o ¸ D r j P l e D o P ´ o j o i m r ¸ D P ¸ D ¸ T ó T j P e s t ¸ e ¸ D D o i n t j P t j P e j e D n d t P e s o . b i o P j P ¸ T j j o j P u j m o r m j P *Application* P o P *Document*
  - g ¸ D P j b j m o ¸ D r P e s o P ¸ t i j m o e D s x t P e s o P e b r C o r ¸ D r j S ´ t o P o - o ¸ D ´ T S P m o d o r P j P ¸ T j j o j P *DrawingApplication* P o P *DrawingDocument*
  - A P *Framework* P e s o e D o P b i j m r i ¸ b r d P u 5 o r t j P ¸ D r j P j : P ¸ t i v o ¸ o P ¸ T j j o j P u j m o r m j

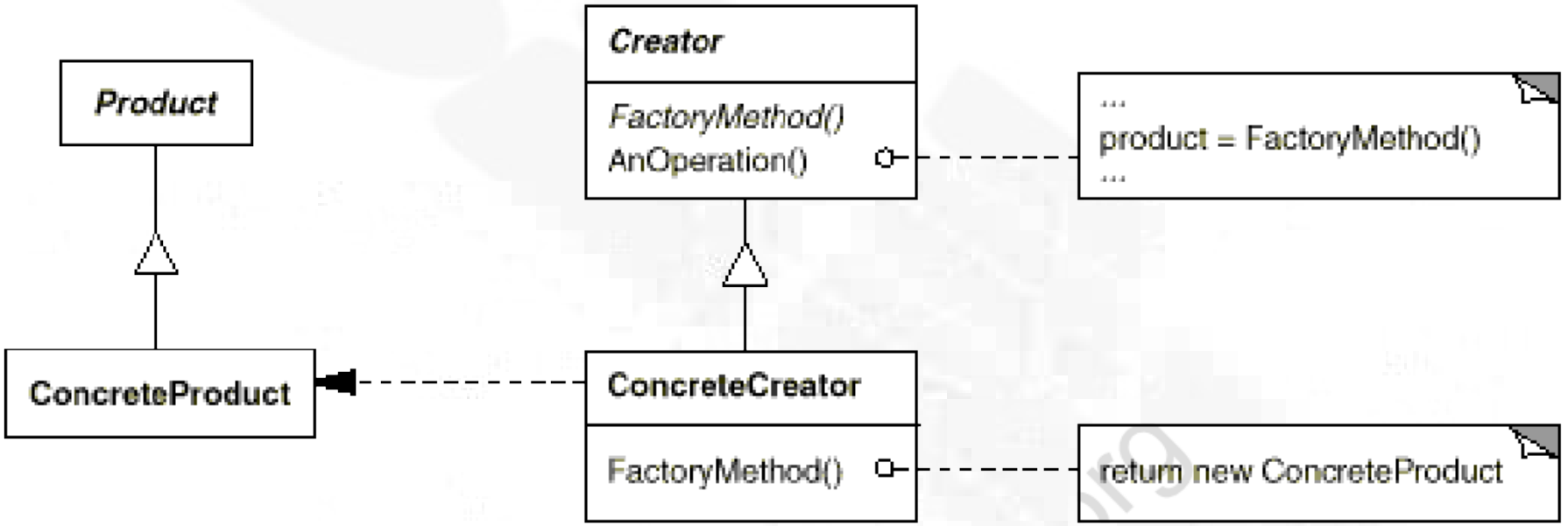
- U t m d D s x t m



  - C e u r T j j o j e s o P *Application* P D T o D o i m D P D q t e s t P r u j m o r t P *CreateDocument* P r o P o o t c i r d P j e u r T j j o e s o P *Document* P ´ c t ´ d o r e s r

- z´ Tb̧r ubTer som
  - g¸oDr P¸ Tr j j oP xt P¸ t i vo¸ oṔ oeDr ¸oDoi moP j P¸ Tr j j ojPst jP t u5ont jP eDoPoTṔ o¸ bjr Pi jmi ¸bro
  - g¸oDr P¸ Tr j j oP eDoP eDoPj eDr jPj eDuŗDTr j j ojPoj´ o¸ bbl eDo¸oDPt jP t u5ont jP Pj oo o¸oDP¸ bres t j
  - E Tr j j ojPesoTbOr ¸oDPoj´ t i j r ubTer soṔ r or PeḐoDr Poi moPeDoor jP jeDuŗDTr j j ojP*helper* PoPesoj o5 r ŀDoPt ¸ r Tb(r oPt P¸ t i vo¸ bçDoi nt P esoṔ r or P eDr Pj eDuŗDTr j j oP*helper* P.t bPesoTbOr esr P P´ oor sxt

-) j r o e D e D o r m

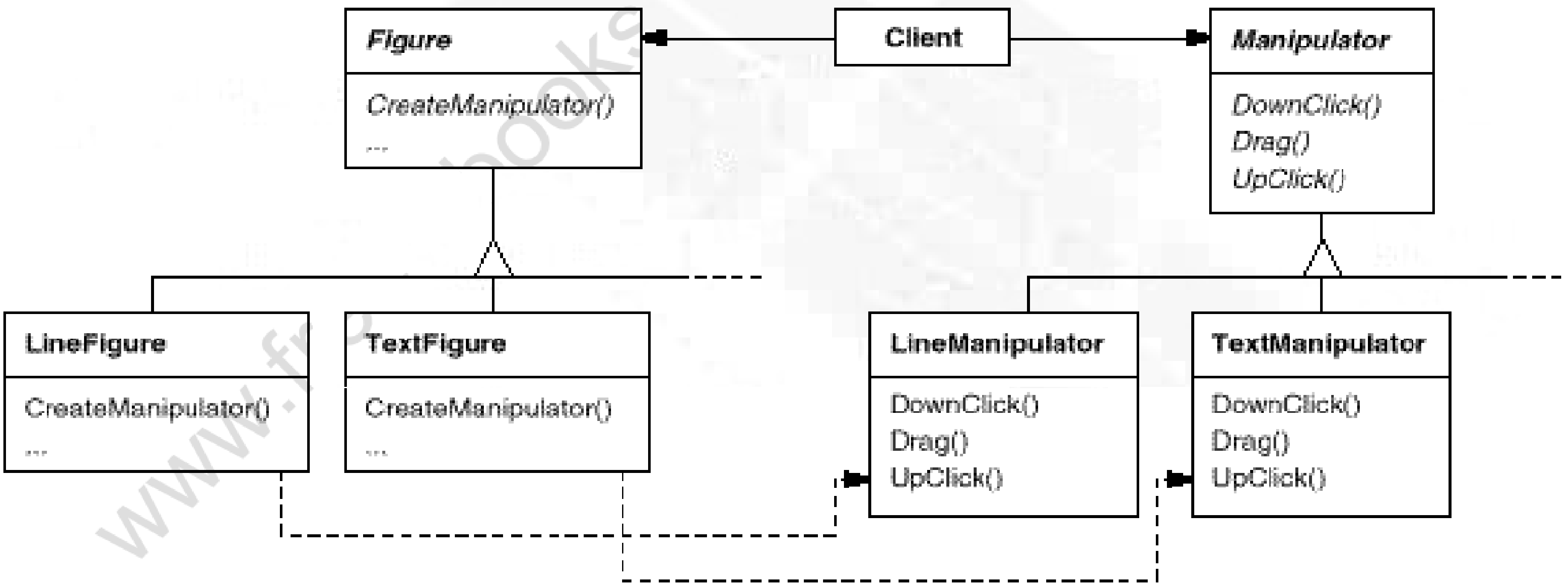

Creator
FactoryMethod()
AnOperation()
...
product = FactoryMethod()
...
Product
ConcreteProduct
ConcreteCreator
FactoryMethod()
return new ConcreteProduct

- dr ob̧ ó r i noj m
  - *Product* P c t ¸ eḐo i n meso.b o P P i n oa r ¸ oPe t j P u 5o n t j P eo P P *factory method* P ¸ dr
  - *ConcreteProduct* PU c t ¸ eḐo i n m P ȩD ́ T b̧o i n P P i n oa r ¸ o P *Product*
  - *Creator* P z ´ ´ T ȩ r n b i m
    - c o ¸ T o r P P *factory method* P o n t ó r i e t P ȩD P u 5o n t P e t P b ́ t P *Product.* P d t e o P j o o P u j m o r n t P e P x t
    - oa r ȩD o i n P b eD ¸ r P P *factory method*
  - *ConcreteCreator* PU z ´ ´ T ȩ r n b i m P ȩD ́ T b̧o i n P t eD P j t u o o ´ ˜ o P P *factory method* P r o P o n t ó r o P ȩD r P ȩD ́ T b̧o i n r s x t P o j ´ o ¸ . ȩ r P e o P *Product* P r T O ȩD P *ConcreteProduct*

- Et Tutorsõojm
  - AP *Creator* P¸ti .brP jPj erjPj eur¸DT jjojP Peso.bibsxt Pst P *factory method* Sont cirist Pe¸DDr Pbjmsi ¸brP´ct´dbrsr PesoP *ConcreteProduct*

- Etijolë Đi ¸brjm
  -) TĐbi r PP o¸ ojjbsr eoReoPĐbT(r oSt P oeĐP¸ : ebOt S¸ T jjojP oj´ o¸ .b r jPeoP´ T r sxt P´ t bjPĐbT(r P t ¸Đoi noPPi oor ¸ oP *Product*
  -) -bOoP PeodĐ sxt PeoP*Creator*P´ r or P PĐ´ T¸Đoi n sxt PeoPe¸ĐP , i b t P¸Đqnt st P´ dt eĐ eĐ TĐoi noP¸ t ¸ĐPĐ¸Đr PT vr
  - dd eĐ P*hooks*P jPeĐurĐT jjojPt Pjbjno¸Đr PeoPet ¸ e¸Đoi nt jS ´ t oP- o¸Đ´ T S´ t eorĐoPeo.bi boPe¸Đ*factory method*P¸ vr ¸Đr et P *createFileDialog*P¸ t ¸ĐPe¸Đr P¸Đ´ T¸Đoi n sxt P*default*PoP ´ oo¸Đbo P eĐoPjeĐurĐT jjojPi ebl eĐo¸ĐP PĐbT(r sxt PeoPe¸ĐP t u5ont P Toci r neĐ

- Et i jol eĐ i ¸br j m
  - Et i o¸ m rPvbo or d eĐr jŔ r or ŏT j Pes P¸T j j oj m



- [illegible]
  - [illegible]
    - *Creator* [illegible]
      - [illegible] *factory method* [illegible]
    - *Creator* [illegible] *default* [illegible] *factory method* [illegible]
      - [illegible] *ConcreteCreator* [illegible] *factory method* [illegible] *factory method*
  - *Factory methods* [illegible]
    - A *factory method* [illegible]

- F D T D i n s x t m
  - *Factory methods* P r c r D n d (r e s t j m

```
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;
}
```

```
Product* MyCreator::Create (ProductId id) {
    if (id == YOURS)  return new MyProduct;
    if (id == MINE)   return new YourProduct;
        // N.B.: switched YOURS and MINE

    if (id == THEIRS) return new TheirProduct;

    return Creator::Create(id); // called if id is unknown
}
```

- Ғ̧οD΄ Tο̧οDoi mr ısxt m
  - r dοr ıs˜ ojΡoΡ΄ ατ uTο̧οDr jΡoj ΄ o̧ .b̧ t jΡesοΡbi ΘeDr Οo̧οDm
    - A Ρ*factory method* Ρ΄ t esοΡont α ι r οΡ Ρ Ρest Ρ u5οnt Ρ Ρj οο Ρ ̧ dοr est ΡoS΄ t οrι i nt SeΡ̧οD Ρ*ConcreteCreator* Ρ΄ t esοΡesο.b i bοΡojnr Ρ ̧ T j j oΡο̧ D Ρ*run-time*
    - zΡ̧ T j j oΡest Ρ u5οnt Ρ Ρj οο Ρ̧ dοr est Ρ΄ t esοΡj οοΡr ̧ Duq̧ DΡeΡ̧ DΡ r mοbuεDt Ρojmlnb̧ t Ρ t Ρ*Creator*
    - 0 t ΡE SΡΡ *factory method* Ρest Ρ*ConcreteCreator* Ρ xt Ρ΄ t esοΡ j οοΡ̧ vr ̧ οDr est Ρ t Ρ̧ t i j mοeDt οesοΡ*Creator* S΄ t lb̧ Ρ b i esr Ρ xt Ρ ojmΡesb̧ ΄ t i eDοTpΡz t Ρb i eDq j SΡ:r ıs Ρ*lazy initialization*
    - 0 t ΡE SΡοmο̧ D΄ Tmοj Ρ΄ t esο̧ DΡj οοΡeDbΤ(r est j Ρ΄ r α ΡoeDr οΡ Ρ ̧ dοr ısxt ΡesοΡj εDuŗ DT j j ojΡ̧ t ̧ οDj t ̧ οDoi mοΡȩ DΡ̧ οDqnt est
    - d r esοα i b(r ısxt Ρ΄ r α Ρ t ̧ οDojΡest jΡ*factory methods* m
      - *Class *doMakeClass()* SΡ i esοΡ*Class* ΡqΡ Ρ̧ T j j oΡest Ρ΄ α esοDt

- E: esbQ Ro-o,dÓ t m

```
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;
}
```

```
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); }
};
```

- E: esbQ RPo-o,Ø Tm

```
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;
};
```

- g j t j P¸t i vo¸ beest j m
  - z Tnr ¸oDoi moP´ oo j oi moPo¸oDP*toolkits* PoP*frameworks*
  - U r ¸ z´ ´ ´ Po-o¸oD´ T tPest Pest ¸ eD¸oDoi nt PoP) M Po-o¸oD´ T tPest j P ¸oDr i b´ eDTr est oo j
  - *Framework* PU E Pest PC¸oDr Tnr T
  - A oub-PA m *factory method* P´ r or POoor oP Pb´ t P´ ´ ot´ oor est PesoP *proxy* P eDr i est PeD¸oDr Pbi eD ¸ r rsxt Po¸oDt mr PqP o¸ oj j h oor

- dresõ˜ojPoor¸bt i r est j m
  - *Abstract Factory* qP.ol eDoi no¸Doi nPo¸D́ To¸Doi n est P¸t ¸DP *factory methods*
  - *Factory methods* Pj xt POor ToDoi nP¸vr ¸Dr est jPesoi ntPesoP *Template methods*
  - gnbT(r i est P*Prototypes* P P¸dr sxt Per jPj eDurDTr j j ojP xt PqP i o¸oj j hdr Soi nonr i nt Pj ochP o¸oj j hdt PeD¸DPDqnt est PesoP bi b¸br T(r sxt Pest Ṕ c ee Dn St Pj ooP¸vr ¸Dr est Ṕ oT P*Creator* pEt ¸DP *factory methods* P¸t i eDoi ¸bt i r bjPqj t P xt PqP o¸oj j hdt

Padrões de Projeto

Factory Method

Prof. Dr. Bruno Gomes Coelho
bruno@ifsp.edu.br