Les UML

Diagramme de classes de la partie client

Les formes

```
classDiagram
    class Forme{
        #_couleur : int

        virtual calculerAire() double
        virtual getCentreDeSymetrie() Vecteur2D
        virtual getMaxXMaxY() Vecteur2D
        virtual getMinXMinY() Vecteur2D
        virtual homothetie(Vecteur2D centre, double rapport)
        virtual rotation(Vecteur2D centre, double angle)
        virtual translation(Vecteur2D vecteur)
        virtual const sauvegarde(const VisiteurDeSauvegarde* visiteur)
        virtual dessine(const VisiteurDeLibrairie* visiteur, SOCKET
socket)
        virtual operator string()
    }
    <> Forme
    Forme <|-- Cercle
    Forme <|-- Polygone
    Forme <|-- Segment
    Forme <|-- FormeComplexe

    FormeComplexe "1" --* "*" Forme : _listeFormes
    Cercle --* Vecteur2D
    Polygone "1" --* "*" Vecteur2D : _points
    Segment --* Vecteur2D

    class Cercle{
        -_centre : Vecteur2D
        -_rayon : double
    }
    class Segment{
        -Vecteur1 : Vecteur2D
        -Vecteur2 : Vecteur2D
    }
    class Vecteur2D{
        -double _x
        -double _y

        +norme() double
        +produitScalaire(Vecteur2D vecteur) double
        +operator-() Vecteur2D
        +operator-(Vecteur2D vecteur) Vecteur2D
    }
```

Les Experts

```
classDiagram

    ExpertChargement <|-- ExpertChargementCOR
    ExpertChargementCOR <|-- ExpertChargementCercle
    ExpertChargementCOR <|-- ExpertChargementPolygone
    ExpertChargementCOR <|-- ExpertChargementSegment

    ExpertChargementCOR "1" --o "*" ExpertChargementCOR : _suivant

    class ExpertChargement{
        +virtual resoudre(const string & ligne) Forme*
    }
    class ExpertChargementCOR{
        +virtual resoudre1(const string & ligne) Forme*
    }
```

Les Visiteurs

```
classDiagram

    VisiteurDeSauvegarde <|-- VisiteurDeSauvegardeTxt

    class VisiteurDeSauvegarde{
        +virtual visite(const Forme* forme)
    }

    VisiteurDeLibrairie <|-- VisiteurDeLibrairieJava

    class VisiteurDeLibrairie{
        +virtual visite(const Forme* forme, SOCKET socket)
    }
```

Autres classes

```


    class Erreur{
        +operator string()
        +testeEgalite (int m, int n, const char *message)
        +testePlusPetitOuEgal (const int a, const int b, const char
*message)
        +testeAppartient (int a, int x, int b, const char *message)
        +testeNonVide (const void *d, const char *message)
    }
```

```
class ChargeurListeForme{
    +charge(ifstream & fichier) vector
}
```