**ПРИМЕР №7**

Задача: реализация 3D модели изогнутый прямоугольный треугольник

**Поэтапное построение эскиза.**

1. Создаем локальную СК:  
   MbPlacement3Dpl;
2. Затем создаем массив точек, вершин ломаной, и объявляем их.

Создаем массив точек:

SArray<MbCartPoint>arrPnts(19);(возможно, тут надо описать, какое слово что значит)

Добавляем в массив точки:

arrPnts.Add(MbCartPoint(30, 40));

arrPnts.Add(MbCartPoint(30, 20));

arrPnts.Add(MbCartPoint(0, 20));

arrPnts.Add(MbCartPoint(0, 0));

arrPnts.Add(MbCartPoint(100, 0));

arrPnts.Add(MbCartPoint(100, 20));

arrPnts.Add(MbCartPoint(70, 20));

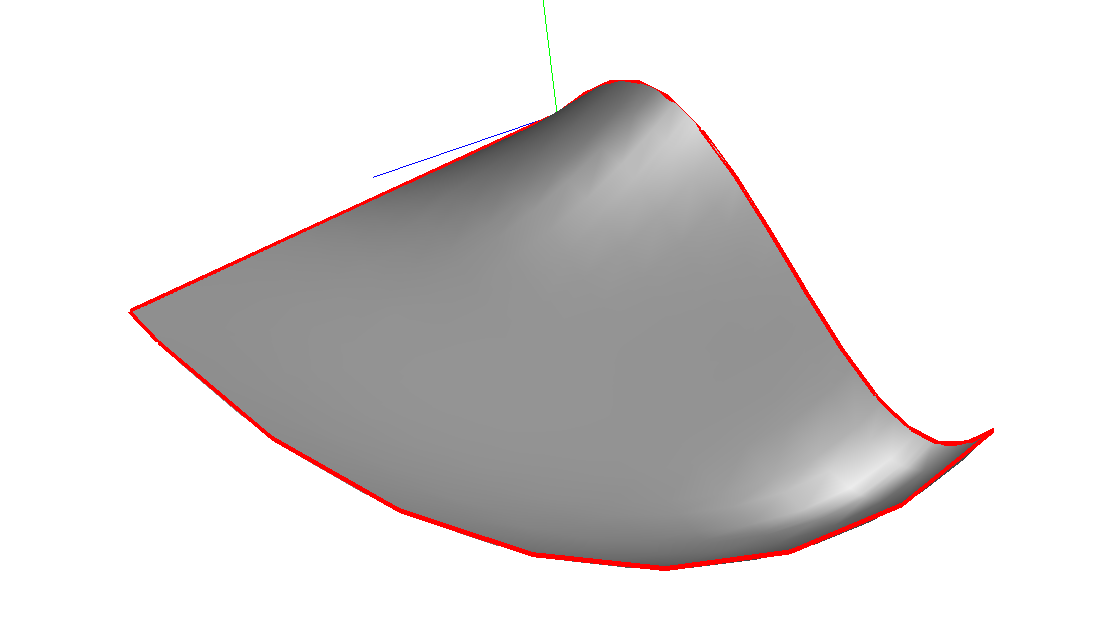
arrPnts.Add(MbCartPoint(70, 40));

1. Создаем ломаную:

MbPolyline\* pPolyline = newMbPolyline(arrPnts, false/\* Флаг незамкнутой линии \*/);

1. Создаем фаску:  
   ChamferPolyContour(pPolyline, 5,5, false, false, arrPnts[4]);
2. Создаем контур:  
   MbContour\* pContour = newMbContour(\*pPolyline, true);
3. Создаем скругление:  
   FilletPolyContour(pPolyline, 5, false, arrPnts[2], pContour);
4. Создаем дугу (Арку):  
   MbCartPointarcCenter(50, 40);   
   const double RADIUS = 20;  
   MbArc\* pArc = new MbArc(arcCenter, RADIUS, arrPnts[7], arrPnts[0], 1 /\*initSense\*/);
5. Добавляем арку в контур:  
   pContour->AddSegment(pArc);
6. Добавляем контур в сцену:  
   viewManager->AddObject(Style(1, RGB(0, 0, 255)), pContour, &pl);
7. Уменьшаем счетчик ссылок динамически созданных объектов ядра:  
   ::DeleteItem(pPolyline);  
   ::DeleteItem(pArc);  
   ::DeleteItem(pContour);

**Результат построения.**



**Код программы.**

void MakeUserCommand6()

{

// Создание пересекающихся кривых pCurve0, pCurve1, pCurve2 - границ поверхности.

// Незамкнутая кривая Безье для приближенного представления одного периода

// синусоиды в плоскости XY (по SIN\_PNT\_CNT точкам).

SArray<MbCartPoint3D> arrPntsBezier;

const double SIN\_WIDTH = 7;

const int SIN\_PNT\_CNT = 15;

for (int i = 0; i < SIN\_PNT\_CNT; i++)

{

double x = i \* 2 \* M\_PI / SIN\_PNT\_CNT;

double y = 5 \* sin(x);

arrPntsBezier.push\_back(MbCartPoint3D(x \* SIN\_WIDTH, y, 0));

}

MbBezier3D\* pCurve0 = new MbBezier3D(arrPntsBezier, false);

// Отрезок в плоскости XZ (начальная точка совпадает с начальной точкой pCurve0)

MbLineSegment3D\* pCurve1 = new MbLineSegment3D(arrPntsBezier[0],

MbCartPoint3D(3, -1, 25));

// Дуга окружности, соединяющая pCurve0 и pCurve1 по конечным точкам

// Дуга строится по трем точкам. Начальная точка pc1 и конечная pc2 совпадают

// с конечными точками pCurve0 и pCurve1.

// Промежуточная точка дуги в этом примере вычисляется произвольным образом

// (точка - центр отрезка pc1-pc3 смещается вниз по оси y).

MbCartPoint3D pc1 = arrPntsBezier[SIN\_PNT\_CNT - 1];

MbCartPoint3D pc3 = pCurve1->GetPoint2();

MbCartPoint3D pc2;

pc2.x = (pc1.x + pc3.x) / 2;

pc2.y = (pc1.y + pc3.y) / 2 - 10;

pc2.z = (pc1.z + pc3.z) / 2;

MbArc3D\* pCurve2 = new MbArc3D(pc1, pc2, pc3, 1, false);

// Построение поверхности

MbCornerSurface\* pSurf = new MbCornerSurface(\*pCurve0, \*pCurve1, \*pCurve2);

// Отображение поверхности

viewManager->AddObject(Style(1, LIGHTGRAY), pSurf);

// Отображение граничных кривых

viewManager->AddObject(Style(5, LIGHTRED), pCurve0);

viewManager->AddObject(Style(5, LIGHTRED), pCurve1);

viewManager->AddObject(Style(5, LIGHTRED), pCurve2);

// Уменьшение счетчика ссылок динамически созданных объектов ядра

::DeleteItem(pSurf);

::DeleteItem(pCurve0);

::DeleteItem(pCurve1);

::DeleteItem(pCurve2);

}