**TransCAD Constraint API Guide Book**

|  |
| --- |
| 목 차  1. Constraint 개요  2. D-Cubed 2D DCM License 설정  3. TransCAD constraint API 구조  4. TransCAD constraint API 사용법  4.1 Constraint 생성  4.2 Constraint 속성 획득  5. 맺음말  6. 기타 참고 자료 |

작성 이력

- 신수철, 2011.07.12, 초안 작성

- 신수철, 2011.01.02, 참고자료 추가

- 신수철, 2012.10.29, 라이선스 옵션 내용 추가

**1. Constraint 개요**

Constraint(구속조건)은 파라메트릭 형상 모델링에 있어서 편리함과 설계 모델의 완벽성을 제공하는 역할을 한다. Constraint는 크게 수치(dimension) 구속조건과 논리적(logical) 구속조건으로 나눌 수 있다. 수치 구속조건은 해당 구성요소의 형상을 수치로서 정량적으로 모델링 할 때 사용되는 것이다. 예를 들자면, 선분의 길이는 10cm, 원의 반지름은 5cm, 두 선분이 이루는 각은 45도 등이 수치 구속조건에 해당된다. 반대로 논리적 구속조건은 어떤 구성요소가 수평 또는 수직인지, 그리고 두 구성요소간의 기하학적 관계가 평행인지 직각인지, 또 두 점이 일치하는지 등을 정의하는 것이다. 이러한 구속조건은 특히 설계 변경 시에 아주 중요한 역할을 한다. 예를 들어 서로 수직인 두 선분이 있고, 한쪽 선분의 방향을 변경한다고 가정해 보자. 만약 수직(perpendicular) 구속조건이 부여되지 않다면, 서로 수직인 관계를 유지하기 위해 반대쪽 선분의 방향도 함께 수정해줘야 한다. 그러나 수직 구속조건이 있다면 한쪽 선분의 방향을 변경하면 반대쪽 선분의 방향도 수직 구속조건에 따라 자동으로 변경이 된다.

**2. D-Cubed 2D DCM License 설정**

TransCAD에서는 constraint solver로서 D-Cubed社의 2D DCM(dimensional constraint manager)을 사용하기 때문에 이에 대한 라이선스 설정을 별도로 해주어야 한다. 라이선스는 네트워크 카드의 Mac address에 종속적이기 때문에 PC 1대에 라이선스가 하나씩 부여된다. 따라서 개발 환경에 따른 라이선스를 별도로 신청해야 하는데, 라이선스 신청을 위해서는 자신의 개발 환경에 따른 reference code가 필요하다. Reference code를 얻는 방법은 다음과 같다.

1) SketchEditor를 열 때, constraint를 사용함(true)으로 한다.

2) 만약 라이선스가 없다면 그림1과 같은 에러 메시지가 출력된다.

3) 이 창에서 Quote reference code, 즉, “G3SVF9TBDR7NWN”를 DCM 관리자에게 알려주고 라이선스 발급을 신청한다.

4) 새로 발급되는 라이선스는 아래와 같은 형식을 갖는다.

G3SVF9TBDR7NWN WFHBPF52FR7DB8

5) dcubed.lic라는 텍스트 파일을 생성하여 위 코드를 삽입하고 dcm 폴더에 저장하면 라이선스 설정이 완료된다.

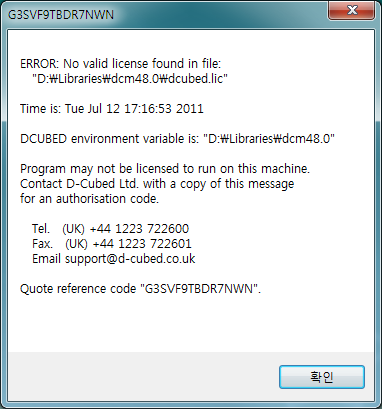


Figure 1. D-Cubed DCM 오류 메시지 창

**2.1 개발시 DCM 라이선스 적용 정책**

TransCAD 내에서 구속조건 solving을 위해서는 반드시 D-Cubed 2D DCM 소프트웨어가 필요하지만, 입력 CAD 모델에 구속조건 관련 정보가 모두 포함되어 있는 경우 변환 목적으로는 DCM이 필요하지 않다. 즉, 입력 받은 구속조건 정보를 목표 시스템에 그대로 전달만 해주면 되기 때문에 TransCAD 내부에서 구속조건 solving을 할 필요는 없다. DCM 라이선스는 평균 한 달을 주기로 하여 정기적으로 갱신을 해줘야 하기 때문에 라이선스 관리가 어렵고 라이선스 갱신 과정 동안 개발 시간을 지연시키는 문제가 있기 때문에, 라이선스가 없어도, 즉 DCM을 사용하지 않고도 구속조건 속성 생성을 가능케 하여 변환 기능을 수행할 수 있도록 할 필요가 있다. 앞으로 MACRO 번역기 프로젝트에서 번역기 개발 시에는 DCM 사용을 강제하지 않으며, test rally에서는 전원이 가급적 DCM 사용을 하는 것으로 정책을 정한다. 이를 위해 TransCAD에서는 DCM 사용 (solver 구동) 여부를 선택할 수 있는 기능을 제공한다. 이 옵션은 SketchEditor를 열 때 지정할 수 있으며, 관련 API는 다음과 같다.

|  |
| --- |
| // Open a sketch  ISketchEditorPtr spEditor1 = spSketch1->OpenEditorEx(VARIANT\_TRUE); // Using DCM  ISketchEditorPtr spEditor1 = spSketch1->OpenEditorEx(VARIANT\_FALSE); // Not using DCM |

**3. TransCAD constraint API 구조**

현재 TransCAD에서는 sketch 구성요소에 대한 2D constraint만을 제공하며, constraint API 구조는 그림2와 같다.

SketchFeature는 Geometry 클래스를 관리하는 Geometries 클래스와 Constraint 클래스를 관리하는 Constraints 클래스로 구성된다. Constraint 클래스는 constraint에 대한 원형 클래스이며 실제로는 이를 상속받은 CoincidentSamePoint 클래스, Perpendicular 클래스, Parallel 클래스, Horizontal 클래스, Vertical 클래스가 실질적인 구속조건 기능을 한다. 각 Constraint 상속 클래스는 해당 구속조건에 따라 하나 또는 두 개의 Geometry segment를 input argument로 가진다. 이 때, Geometry segment는 별도의 Reference 인스턴스를 통해서 input argument로 넘겨진다. Geometry segment에 대한 Reference 인스턴스는 아래와 같은 명명법을 통해 참조할 수 있다.

예) “Sketch1”이라는 이름을 가지는 Sketch 인스턴스에서 “Line1”이라는 이름을 가지는 sketch segment를 참조하는 경우 해당 명명법은 *"Sketch1,Line1,0,0,0,0,Sketch\_Segment"*와 같다.

**SketchFeature**

**Geometries**

**Constraints**

**Geometry**

**Constraint**

**SamePoint**

**Perpendicular**

**Parallel**

**Horizontal**

**Vertical**

**Line**

**Circle**

\*

\*

1

1

Figure 2. TransCAD constraint API 구조

**4. TransCAD constraint API 사용법**

이제 TransCAD constrain API를 사용하는 방법에 대해 알아보자. 번역기의 pre-processor에서는 해당 시스템의 구속조건을 읽어서 TransCAD 상에 동일한 구속조건을 생성해야 하고, post-processor에서는 TransCAD 상의 구속조건을 읽어서 해당 시스템에서 동일한 구속조건을 생성해야 한다. 따라서 pre-processor에서는 TransCAD 구속조건을 생성하는 법을, post-processor에서는 TransCAD 구속조건을 읽어오는 법을 익혀야 한다.

먼저 구속조건을 생성하는 방법에 대해 살펴보자.

**4.1 Constraint 생성**

TransCAD에서 구속조건을 생성하는 절차는 그림3과 같은 흐름에 따라 진행된다.

**Open a SketchEditor**

**Draw geometry segments**

**Create References for the segments**

**Create constraints using the References**

**Close the SketchEditor**

Figure 3. TransCAD constraint 생성 절차

즉, 기존의 sketch geometry segment를 생성하는 과정에서 다음 두 가지 절차만 추가되었다.

1) Geometry segment를 참조하기 위한 Reference 생성

2) 위에서 생성한 Reference를 이용하여 구속조건 생성

위 두 가지 절차는 그림 3에서 붉은색으로 음영 처리한 부분에 해당된다.

실제로 TransCAD의 constraint API를 이용해서 어떻게 구속조건을 생성하는지 다음의 L-Block 예제를 통해서 살펴보자.

|  |
| --- |
| // Open a sketch  ISketchEditorPtr spEditor1 = spSketch1->OpenEditorEx(VARIANT\_TRUE);  // Create sketch lines  spEditor1->Create2DLine2Points("Line1", 0.0, 0.0, 200.0, 0.0);  spEditor1->Create2DLine2Points("Line2", 200.0, 100.0, 100.0, 100.0);  spEditor1->Create2DLine2Points("Line3", 200.0, 0.0, 200.0, 100.0);  spEditor1->Create2DLine2Points("Line4", 100.0, 100.0, 100.0, 180.0);  spEditor1->Create2DLine2Points("Line5", 0.0, 180.0, 0.0, 0.0);  spEditor1->Create2DLine2Points("Line6", 100.0, 180.0, 0.0, 180.0);  // Create References  IReferencePtr spLineReference1 = spPart->SelectSketchGeometryByName("Sketch1,Line1,0,0,0,0,SketchSegment");  IReferencePtr spLineReference2 = spPart->SelectSketchGeometryByName("Sketch1,Line2,0,0,0,0,SketchSegment");  IReferencePtr spLineReference3 = spPart->SelectSketchGeometryByName("Sketch1,Line3,0,0,0,0,SketchSegment");  IReferencePtr spLineReference4 = spPart->SelectSketchGeometryByName("Sketch1,Line4,0,0,0,0,SketchSegment");  // Create constraints using the References  spEditor1->CreatePerpendicular("Perpendicular1", spLineReference1, spLineReference3);  spEditor1->CreateParallel("Parallel1", spLineReference1, spLineReference2);  spEditor1->CreateHorizontal("Horizontal1", spLineReference2);  spEditor1->CreateVertical("Vertical1", spLineReference3);  spEditor1->CreateCoincidentSamePoints("Coincident1", spLineReference2, EndPoint, spLineReference4, StartPoint);  spEditor1->CreateCoincidentSamePoints("Coincident2", spLineReference3, EndPoint, spLineReference2, StartPoint);  // Close the sketch  spEditor1->Close(); |

먼저, ISketchFeature의 OpenEditorEx() 함수를 이용하여 ISketchEditor를 생성한다. 이 때, 인자로 VARIANT\_TRUE 값을 넘겨서 constraint 기능을 사용하도록 한다. VARIANT\_FALSE로 줄 경우 constraint 기능은 꺼지게 된다. 그 다음, 필요한 geometry segment들을 생성한다. 위의 L-Block 예제에서는 2D Line 6개를 생성하였고, 각 segment는 “Line1”, “Line2” 등과 같은 이름을 가진다. Geometry segment 생성이 완료된 후에는, 이를 참조하기 위한 Reference 인스턴스를 생성하여야 한다. 이를 위해 IPart의 SelectSketchGeometryByName() 함수를 이용하였다. 이 때 사용하는 명명법은 “Sketch1,Line1,0,0,0,0,SketchSegment”와 같은 형태를 갖는다. 즉, 먼저 해당 스케치의 이름을 적고, 그 다음 해당 segment의 이름을 적은 후, 마지막에 entity type으로서 SketchSegment를 명기해주면 된다. 2D Line이든 circle이든 geometry segment의 type과 관계없이 해당 스케치와 segment 이름, 그리고 SketchSegment type으로 해당 segment를 참조할 수 있다. 이렇게 생성한 Reference를 이용해서 최종적으로 원하는 구속조건을 생성할 수 있다. 위 예제에서와 같이 각 구속조건을 생성하는 명령어는 다음과 같다.

|  |
| --- |
| * **Perpendicular 구속조건** CreatePerpendicular(STRING result\_object\_name, IReference segment1, IReference segment2) * **Parallel 구속조건** CreateParallel(STRING result\_object\_name, IReference segment1, IReference segment2) * **Horizontal 구속조건** CreateHorizontal(STRING result\_object\_name, IReference segment) * **Vertical 구속조건** CreateVertical(STRING result\_object\_name, IReference segment) * **SamePoint 구속조건** CreateCoincidentSamePoints(STRING result\_object\_name, IReference segment1, SamePointType type1, IReference segment2, SamePointType type2) |

주의할 점은 SamePoint 구속조건의 경우 해당 점을 선택하기 위해 점에 대한 Reference를 바로 생성하지 않고, 해당 점을 포함하는 line이나 arc에 대한 Reference를 생성한 후, 구속조건을 생성할 때, 해당 점이 segment에서 StartPoint인지, EndPoint인지, CenterPoint인지를 SamePointType으로서 지정해준다는 점이다.

이로서 TransCAD constraint API를 이용한 구속조건 생성 방법에 대한 설명을 마친다.

**4.2 Constraint 속성 획득**

TransCAD에서 구속조건의 속성을 가져오는 절차는 그림4와 같은 흐름에 따라 진행된다.

우선 SketchFeature로부터 스케치의 geometry segment 정보를 모두 가져온 후, 다시SketchFeature로부터 SketchFeature로부터 SketchConstraints를 가져온다. SketchConstraints는 다수의 SketchConstraint를 리스트 형태로 가지고 있는데, 각 SketchConstraint를 순회하면서 해당 SketchConstraint의 종류에 따라 그에 맞는 Referencee 정보를 가져온다. 이 정보를 앞서 가져온 geometry segment의 정보와 비교하여 매핑한 후, 필요한 정보를 가져온다.

**Get the geometry information from SketchFeature**

**Get SketchConstraints from SketchFeature**

**Get SketchConstraint from SketchConstraints**

**Check the type of SketchConstraint**

**Get the Referencee information from SketchConstraint**

**Map the Referencee information to the geometry segment**

Figure 4. TransCAD 구속조건 정보 획득 절차

실제로 TransCAD constraint API를 이용하여 구속조건 정보를 어떻게 가져 오는지 다음 예제를 통해서 살펴보자.

|  |
| --- |
| IStdSketchConstraintsPtr spConstraints = spSketchFeature->Constraints;  for(int i = 1; i <= spConstraints->Count; ++ i)  {  IStdSketchConstraintPtr spItem = spConstraints->Item[i];  // IStdSketchCoincidentSamePoints  IStdSketchCoincidentSamePointsPtr spCoincident;  HRESULT hr = spItem.QueryInterface(\_\_uuidof(IStdSketchCoincidentSamePoints), &spCoincident);  if(SUCCEEDED(hr))  {  // The Reference for the first segment  IReferencePtr spReference1 = spCoincident->FirstObject;  CString referenceeName1 = spReference1->ReferenceeName;  StdCoincidentSamePointsType firstType = spCoincident->FirstType;  // The Reference for the second segment  IReferencePtr spReference2 = spCoincident->SecondObject;  CString referenceeName2 = spReference2->ReferenceeName;  StdCoincidentSamePointsType secondType = spCoincident->SecondType;  }  // IStdSketchPerpendicular  IStdSketchPerpendicularPtr spPerpendicular;  hr = spItem.QueryInterface(\_\_uuidof(IStdSketchPerpendicular), &spPerpendicular);  if(SUCCEEDED(hr))  {  // The Reference for the first segment  IReferencePtr spReference1 = spCoincident->FirstObject;  CString referenceeName1 = spReference1->ReferenceeName;  // The Reference for the second segment  IReferencePtr spReference2 = spCoincident->SecondObject;  CString referenceeName2 = spReference2->ReferenceeName;  }  } |

제일 먼저 IStdSketchFeature의 Constraints 속성을 가져와서 IStdSketchConstraintPtr 변수에 저장한다. 이 속성에서 다시 Constraint 개수에 관한 속성인 Count 속성을 가져와서 그 개수만큼 반복문을 수행하면서 해당 Constraint의 정보를 가져온다. 반복문 내에서는 IStdSketchConstraints의 Item 속성을 이용하여 IStdSketchConstraint 인스턴스를 가져온다. 다음으로는 IStdSketchConstraint가 실제로 어떤 종류의 Constraint인지를 알아보기 위해 다음과 같은 명령어를 이용하였다.

|  |
| --- |
| HRESULT hr = spItem.QueryInterface(\_\_uuidof(IStdSketchCoincidentSamePoints), &spCoincident); |

해당 IStdSketchConstraint 인스턴스가 IStdSketchCoincidentSamePoints 형식인 경우 hr은 true값을 가지고, 그렇지 않은 경우에는 hr은 false 값을 가진다. 이렇게 검사한 IStdSketchConstraint가 만약 IStdSketchCoincidentSamePoints 형식이라면, 두 개의 Reference와 두 개의 SamePoint type 속성을 가지기 때문에, 이들 속성을 가져오면 된다. 각 속성을 가져오는 방법은 아래와 같다.

|  |
| --- |
| // The Reference for the first segment  IReferencePtr spReference1 = spCoincident->FirstObject;  CString referenceeName1 = spReference1->ReferenceeName;  StdCoincidentSamePointsType firstType = spCoincident->FirstType;  // The Reference for the second segment  IReferencePtr spReference2 = spCoincident->SecondObject;  CString referenceeName2 = spReference2->ReferenceeName;  StdCoincidentSamePointsType secondType = spCoincident->SecondType; |

이렇게 구한 Referencee 정보와 이전에 가져온 geometry segment의 이름을 비교하여 해당 구속조건이 어떤 geometry segment를 참조하고 있는지, 다시 말해 해당 구속조건이 어떤 geometry segment에 부여되어 있는 지를 알 수 있다. 여기까지의 결과를 바탕으로 해서 각 번역기마다 대상 시스템에 동일한 구속조건을 생성할 수 있다.

이상으로 TransCAD constraint API를 이용한 구속조건 정보 획득 방법에 대한 설명을 마친다.

**5. 맺음말**

지금까지 TransCAD를 이용하여 구속조건을 생성하는 방법과 이미 생성된 구속조건의 정보를 가져오는 방법에 대해 살펴보았다. 그러나 실제 번역기에서 구속조건 번역 기능을 구현하기 위해서는 여기서 설명한 내용 외에도 해당 번역기의 구조와 대상 CAD 시스템의 구조에 대한 충분한 이해가 바탕이 되어야 한다. 이는 구속조건뿐만 아니라 다른 기능을 구현할 때도 동일하다. 중요한 것은 구속조건을 생성하고 정보를 가져오기 위해서는 단지 스케치의 이름과 geometry segment의 이름만 알면 된다는 것이다. 따라서 번역기가 어느 정도 완성되어 있다면 구속조건 번역 기능을 추가하는 것은 약간의 추가적인 작업만 해주면 된다. 물론 대상 CAD 시스템에 따라 그 작업량은 다르겠지만..

**6. 기타 참고 자료**

**1) Implementation Guide (Korean)**   
– ftp://open.kaist.ac.kr/icad\_only/@ICAD\_MACROPGroup/\_DOCU/3\_Documents\_Reference/구현가이드/Implementation Guide.Kor.201108.docx

**2) Macro Handbook (Korean)**   
– ftp://open.kaist.ac.kr/icad\_only/@ICAD\_MACROPGroup/\_DOCU/3\_Documents\_Reference/핸드북/Macro\_Handbook\_Kor\_081124.docx

**3) Macro Handbook (English)**   
– ftp://open.kaist.ac.kr/icad\_only/@ICAD\_MACROPGroup/\_DOCU/3\_Documents\_Reference/핸드북/Macro\_Handbook\_eng\_09.docx