D-Case 仕様書

27/NOV/2013

改訂履歴

更新日	版	内容
27/NOV/2013	0.9.2	d-script のスキーマの参照を修正(2.1節)
19/NOV/2013	0.9.1	・script および interval 属性を削除(2章全般) ・2.3 節を追加
7/NOV/2013	0.9.0	新規作成

目次

1. はじめに

1.1. 概要

本仕様書は、D-Case Editor における参照実装を元にした D-Case の仕様書である。

1.2. 用語の定義

名 称	意 味
ダイアグラム	(D-Case Editor 上で D-Case を)図表化したもの。
D-Case 文書	D-Case およびモジュールを表すファイル。
GMF ダイアグラム情報ファイル	D-Case もしくはモジュールの各要素(ノードやリンク)の位置や大きさ、色などの情報を表すファイル。 「D-Case 名もしくはモジュール名.dcase_diagram」のファイル名で記録される。
GMF モデル情報ファイル	D-Case もしくはモジュールの各要素の論理的な構造を表すファイル。 「D-Case 名もしくはモジュール名.dcase_model」のファイル名で記録される。
属性	ノードやリンクがそれぞれ固有に持つ性質。 Name, Desc, Attachment, Userdef001～016 などがある。
パレットビューア	Eclipse 上で動作するグラフィカルエディタ(D-Case Editor)内で、ノードやリンクを選択するツールを提供する部分。
ビュー	Eclipse 内で何らかの情報を提供する部分。通常はタブ形式でいずれか 1 つの情報が表示されている。
プリファレンスストア	Eclipse でプラグインの設定を保存するための領域。ワークスペース(作業領域)内に設けられ、プラグイン毎に分けて記録される。

1.3. 関連文書

- D-Case Editor 機能仕様書
- D-CASE エディタ データ仕様書
- 松野裕, 山本修一郎, 高井利憲, D-Case 入門, ダイテックホールディング, ISBN978-4-86293-079-8
- 松野裕, 山本修一郎, 実践 D-Case, ダイテックホールディング, ISBN978-4-86293-091-0
- DEOS プロジェクト White Paper Version 3.0
  http://www.dependable-os.net/osddeos/data/DEOS-FY2011-WP-03J.pdf
- Yutaka Matsuno, A Design and Implementation of GSN Patterns, 2013.
- GSN COMMUNITY STANDARD VERSION1
  http://www.goalstructuringnotation.info/documents/GSN_Standard.pdf
- Document Associated With Structured Assurance Case Metamodel (SACM) Version 1.0

http://www.omg.org/spec/SACM/1.0/

- Document Associated With Argumentation Metamodel (ARM) Version 1.0 – Beta1
  http://www.omg.org/spec/ARM/1.0/Beta1/
- XML Schema
  http://www.w3.org/2001/XMLSchema
- Eric van der Vlist, XML Schema, オライリー・ジャパン, ISBN4-87311-120-X
- XSL Transformations (XSLT) Version 1.0
  http://www.w3.org/TR/xslt
- 竹添直樹, 志田隆弘, 奥畑裕樹, 里見知宏, 野沢智也, Eclipse3.4 プラグイン開発 徹底攻略, 毎日コミュニケーションズ, ISBN978-4-8399-2972-5
- Frank Budinsky, David Steinberg, Eclipse モデリングフレームワーク, 翔泳社, ISBN978-4-79810-663-2
- Eclipse Documentation
  http://www.eclipse.org/documentation/
- Graphical Modeling Framework - Eclipsepedia
  http://wiki.eclipse.org/Graphical_Modeling_Framework
- Eclipse Modeling Framework - Eclipsepedia
  http://wiki.eclipse.org/Eclipse_Modeling_Framework
- Graphical Editing Framework - Eclipsepedia
  http://wiki.eclipse.org/GEF
- Java Platform Standard Edition 7 Documentation
  http://docs.oracle.com/javase/7/docs/index.html

2. データ仕様

2.1. D-Case ファイル

D-Case ファイルのデータ仕様を定める。

2.1.1. スキーマの概要

D-Case ファイルは XML(Extensible Markup Language)文書であり、後述するスキーマで定義される。

その XML 文書の各要素とその概要を、以下に示す。

なお、d-script 要素については、「D-Case Weaver 仕様書」に含まれる、dre.xsd を参照されたい。

要素名	概要	下位要素	属性
dcase	D-Case もしくは モジュール	nodes links properties description parameters responsibility	id : ID name : 名前 status : 状況
nodes	ノード一覧	node	
node	ノード	properties description parameters responsibility d-script	id : ID name : 名前 status : 状況 type : 種類(下記から選択) Goal Strategy Evidence Undeveloped Context Monitor Justification Assumption Module Contract Action External Pattern Userdef002～Userdef003 subtype : サブタイプ(下記から選択) ※1 Parameter Loop Choice

			Multiplicity ※1 Pattern のみ
properties	プロパティー覧	property	
property	プロパティ		value : 値 name : 名前(下記から選択) Attachment : 添付 Message : メッセージ Requirement : 要件 ParameterizedDesc : {パラメータ名}を含む文章 Parent : 親モジュール RefSource : 参照元ノード Flag : パブリック/プライベートフラグ Userdef001～Userdef016 : ユーザ定義 Score : スコア ※1 Weight : 重み ※1 IsNormal : 正常かどうか ※2 StakeHolder : ステークホルダー ※3 RiskAnalysis : リスク解析 ※3 LeafNode : リーフノード ※4 I,J : 範囲 ※5 SiblingOrder : 兄弟の順序 ※6 ValidUntil : 有効期限 ※1 Goal のみ ※2 Monitor のみ ※3 Justification のみ ※4 Pattern の Loop のみ ※5 Pattern の Choice か Multiplicity のみ ※6 link のみ
description	文章		
links	リンクー覧	link	
link	リンク	description properties	id : ID source : 接続元ノードの ID target : 接続先ノードの ID name : 名前 status : 状況 type : 種類(以下から選択) SupportedBy InContextOf Responsibility Link004

parameters	パラメーター一覧	parameter	
parameter	パラメータ		name : 名前 type : 種類(下記から選択) 　string : 文字列 　int : 整数 　double : 浮動小数点数 　enum : 列挙 　raw : raw データ min : 下限値 ※1 max : 上限値 ※1 digit : 小数点以下の桁数 ※2 inc : 増減幅 ※2 items : 項目一覧 ※3 value : 値 ※1 enum 以外のみ ※2 double のみ ※3 enum のみ
responsibility	責任属性		name : 名前 address : E メールアドレス icon : アイコンのパス

2.1.2. スキーマ定義

2.1.1 に従ったスキーマ定義を以下に示す。

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema targetNamespace="http://www.dependable-os.net/2013/11/dcase"
           elementFormDefault="qualified"
           xmlns="http://www.dependable-os.net/2013/11/dcase"
           xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:import namespace="http://www.dependable-os.net/dre"
             schemaLocation="dre.xsd"/>
  <xs:element name="dcase">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="description" type="xs:string" maxOccurs="1" minOccurs="1" />
        <xs:element name="properties" type="PropertiesType" />
        <xs:element name="parameters" type="ParametersType"
                    maxOccurs="1" minOccurs="0"/>
        <xs:element name="responsibility" type="ResponsibilityType"
                    maxOccurs="1" minOccurs="0"/>
        <xs:element name="nodes" maxOccurs="1" minOccurs="1">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="node" minOccurs="0" maxOccurs="unbounded">
                <xs:complexType>
                  <xs:sequence>
                    <xs:element name="description" type="xs:string"
                                maxOccurs="1" minOccurs="1" />
                    <xs:element name="properties" minOccurs="1" maxOccurs="1"
                                type="PropertiesType"/>
```

```
                <xs:element name="parameters" type="ParametersType"
                            maxOccurs="1" minOccurs="0"/>
                <xs:element name="responsibility" type="ResponsibilityType"
                            maxOccurs="1" minOccurs="0"/>
                <xs:element name="d-script" minOccurs="0" maxOccurs="1"/>
              </xs:sequence>
              <xs:attribute name="name" type="xs:string" use="optional" />
              <xs:attribute name="type" use="required">
                <xs:simpleType>
                  <xs:restriction base="xs:string">
                    <xs:enumeration value="Goal" />
                    <xs:enumeration value="Strategy" />
                    <xs:enumeration value="Evidence" />
                    <xs:enumeration value="Undeveloped" />
                    <xs:enumeration value="Context" />
                    <xs:enumeration value="Monitor" />
                    <xs:enumeration value="Justification" />
                    <xs:enumeration value="Assumption" />
                    <xs:enumeration value="Module" />
                    <xs:enumeration value="Contract" />
                    <xs:enumeration value="Action" />
                    <xs:enumeration value="External" />
                    <xs:enumeration value="Pattern" />
                    <xs:enumeration value="Userdef002" />
                    <xs:enumeration value="Userdef003" />
                  </xs:restriction>
                </xs:simpleType>
              </xs:attribute>
              <xs:attribute name="subtype" use="optional">
                <xs:simpleType>
                  <xs:restriction base="xs:string">
                    <xs:enumeration value="Parameter" />
                    <xs:enumeration value="Loop" />
                    <xs:enumeration value="Choice" />
                    <xs:enumeration value="Multiplicity" />
                  </xs:restriction>
                </xs:simpleType>
              </xs:attribute>
              <xs:attribute name="id" use="required" type="UUID">
              </xs:attribute>
              <xs:attribute name="status" type="xs:string" use="optional" />
            </xs:complexType>
          </xs:element>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="links" maxOccurs="1" minOccurs="1">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="link" maxOccurs="unbounded" minOccurs="0">
            <xs:complexType>
              <xs:sequence>
                <xs:element name="description" type="xs:string" />
                <xs:element name="properties" type="PropertiesType" />
              </xs:sequence>
              <xs:attribute name="source" type="xs:string" use="required" />
              <xs:attribute name="target" type="xs:string" use="required" />
              <xs:attribute name="id" type="UUID" use="required" />
              <xs:attribute name="name" type="xs:string" />
              <xs:attribute name="status" type="xs:string" use="optional" />
              <xs:attribute name="type" use="optional" default="SupportedBy">
                <xs:simpleType>
                  <xs:restriction base="xs:string">
                    <xs:enumeration value="SupportedBy" />
                    <xs:enumeration value="InContextOf" />
                    <xs:enumeration value="Responsibility" />
                    <xs:enumeration value="Link004" />
                  </xs:restriction>
```

```
                    </xs:simpleType>
                  </xs:attribute>
                </xs:complexType>
              </xs:element>
            </xs:sequence>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
      <xs:attribute name="id" type="UUID" use="required" />
      <xs:attribute name="name" type="xs:string" use="optional" />
      <xs:attribute name="status" type="xs:string" use="optional" />
    </xs:complexType>
  </xs:element>
  <xs:simpleType name="UUID">
    <xs:restriction base="xs:string">
      <xs:minLength value="1" />
    </xs:restriction>
  </xs:simpleType>
  <xs:complexType name="PropertiesType">
    <xs:sequence>
      <xs:element name="property" minOccurs="0" maxOccurs="unbounded">
        <xs:complexType>
          <xs:sequence>
          </xs:sequence>
          <xs:attribute name="name" use="required">
            <xs:simpleType>
              <xs:restriction base="xs:string">
                <xs:enumeration value="Userdef001" />
                <xs:enumeration value="Userdef002" />
                <xs:enumeration value="Userdef003" />
                <xs:enumeration value="Userdef004" />
                <xs:enumeration value="Userdef005" />
                <xs:enumeration value="Userdef006" />
                <xs:enumeration value="Userdef007" />
                <xs:enumeration value="Userdef008" />
                <xs:enumeration value="Userdef009" />
                <xs:enumeration value="Userdef010" />
                <xs:enumeration value="Userdef011" />
                <xs:enumeration value="Userdef012" />
                <xs:enumeration value="Userdef013" />
                <xs:enumeration value="Userdef014" />
                <xs:enumeration value="Userdef015" />
                <xs:enumeration value="Userdef016" />
                <xs:enumeration value="Attachment" />
                <xs:enumeration value="Message" />
                <xs:enumeration value="Requirement" />
                <xs:enumeration value="ParameterizedDesc" />
                <xs:enumeration value="Parent" />
                <xs:enumeration value="RefSource" />
                <xs:enumeration value="Flag" />
                <xs:enumeration value="Score" />
                <xs:enumeration value="Weight" />
                <xs:enumeration value="Stakeholder" />
                <xs:enumeration value="RiskAnalysis" />
                <xs:enumeration value="IsNormal" />
                <xs:enumeration value="LeafNode" />
                <xs:enumeration value="I" />
                <xs:enumeration value="J" />
                <xs:enumeration value="SiblingOrder" />
                <xs:enumeration value="ValidUntil" />
              </xs:restriction>
            </xs:simpleType>
          </xs:attribute>
          <xs:attribute name="value" type="xs:string" use="required" />
        </xs:complexType>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
```

```
  <xs:complexType name="ParametersType">
    <xs:sequence>
      <xs:element name="parameter" maxOccurs="unbounded" minOccurs="1">
        <xs:complexType>
          <xs:attribute name="name" type="xs:string" />
          <xs:attribute name="type" use="required">
            <xs:simpleType>
              <xs:restriction base="xs:string">
                <xs:enumeration value="string" />
                <xs:enumeration value="int" />
                <xs:enumeration value="double" />
                <xs:enumeration value="enum" />
                <xs:enumeration value="raw" />
              </xs:restriction>
            </xs:simpleType>
          </xs:attribute>
          <xs:attribute name="min" type="xs:string" />
          <xs:attribute name="max" type="xs:string" />
          <xs:attribute name="digit" type="xs:string" />
          <xs:attribute name="inc" type="xs:string" />
          <xs:attribute name="items" type="xs:string" />
          <xs:attribute name="value" type="xs:string" />
        </xs:complexType>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
  <xs:complexType name="ResponsibilityType">
    <xs:attribute name="name" type="xs:string" />
    <xs:attribute name="address" type="xs:string" />
    <xs:attribute name="icon" type="xs:string" />
  </xs:complexType>
</xs:schema>
```

2.2. GMF モデル情報ファイル

D-Case Editor は GMF(Graphical Modeling Framework)をベースに作成されているため、GMF で扱うフォーマットに依存する。GMF では、下記の 2 種類のファイルを使用する。

名前	概要
GMF モデル情報ファイル	D-Case およびモジュールの各要素の論理的な構造を記録するためのファイル。拡張子は「dcase_model」。
GMF ダイアグラム情報ファイル	D-Case およびモジュールの各要素の大きさや位置、色などの情報を記録するためのファイル。拡張子は「dcase_diagram」。

GMF ダイアグラム情報ファイルには、形状に関する情報だけが格納されている。D-Case ファイルと同様、D-Case の構造を格納しているのは GMF モデル情報ファイルであるため、以降では GMF モデル情報ファイルについて記述する。

2.2.1. ノード

GMF モデル情報ファイルで使用可能なノードとその属性(プロパティ)を、以下に示す。

ノード名	属性名	概要
BasicNode	基本(属性は、以降のすべてのノードに共通)	
	name	ノード名
	desc	文章
	parameterizedDesc	{パラメータ名}を含む文章
	attachment	添付
	status	状況
	message	メッセージ
	requirement	要件
	parameterDefs	パラメータ定義
	parameterVals	パラメータ値
	parent	親モジュール
	refSource	参照元ノード
	flag	パブリック/プライベートフラグ
	respName	責任属性名
	respAddress	責任属性アドレス
	respIcon	責任属性アイコン

	validUntil	有効期限
	userdef001～userdef016	ユーザ定義
Argument	D-Case もしくはモジュール全体	
Goal	ゴール、主張	
	score	スコア
	weight	重み
Strategy	戦略	
Evidence	証拠	
Monitor	モニタ	
	isNormal	正常な状態かどうか
Undeveloped	未達成	
Context	前提	
Justification	正当化	
	stakeholder	ステークホルダー
	riskAnalysis	リスク解析
Assumption	仮定	
Module	モジュール	
Contract	契約	
Pattern	パターン	
	subType	Loop, Choice, Multiplicity または Parameter
	leafNode	(Loop の場合)ループ元のリーフノード
	i	(Choice, Multiplicity の場合)i
	j	(Choice, Multiplicity の場合)j
Action	アクション	
External	外部接続	
Userdef002～Userdef003	ユーザ定義ノード	

2.2.2. リンク

GMF モデル情報ファイルで使用可能なリンクとその属性を、以下に示す。

ノード名	属性名	概要
BasicLink	基本(属性は、以降のすべてのリンクに共通)	
	name	ノード名

	desc	文章
	attachment	添付
	status	状況
	message	メッセージ
	siblingOrder	兄弟の順序
	userdef001～userdef016	ユーザ定義
SupportedBy	支援リンク	
InContextOf	前提リンク	
Responsibility	責任属性リンク	
DcaseLink004	ユーザ定義リンク	

2.2.3. スキーマの概要

2.2.1 および 2.2.2 の内容を、2.1.1 の形式で記述すると、下記のようになる。

要素名	概要	下位要素	属性
Argument	D-Case もしくは モジュール	rootBasicNode rootBasicLink	id : ID name : 名前 desc : 文章 parameterizedDesc : {パラメータ名}を含む文章 attachment : 添付 status : 状況 message : メッセージ requirement : 要件 parameterDefs : パラメータ定義 parameterVals : パラメータ値 parent : 親モジュール refSource : 参照元ノード flag : パブリック/プライベートフラグ respName : 責任属性名 respAddress : 責任属性アドレス respIcon : 責任属性アイコン validUntil : 有効期限 userdef001～userdef016 : ユーザ定義
rootBasicNode	ノード		id : ID name : 名前 desc : 文章 parameterizedDesc : {パラメータ名}を含む文章 attachment : 添付 status : 状況

			message : メッセージ requirement : 要件 parameterDefs : パラメータ定義 parameterVals : パラメータ値 parent : 親モジュール refSource : 参照元ノード flag : パブリック/プライベートフラグ respName : 責任属性名 respAddress : 責任属性アドレス respIcon : 責任属性アイコン validUntil : 有効期限 userdef001～userdef016 : ユーザ定義 score : スコア ※1 weight : 重み ※1 isNormal : 正常な状態かどうか ※2 stakeholder : ステークホルダー ※3 riskAnalysis : リスク解析 ※3 type : 種類(下記から選択) Goal Strategy Evidence Monitor Undeveloped Context Justification Assumption Module Contract Pattern Action External Userdef002～Userdef003 subType : サブタイプ(下記から選択) ※4 Parameter Loop Choice Multiplicity leafNode : ループ元のリーフノード ※5 i ※6 j ※6 ※1 Goal のみ ※2 Monitor のみ ※3 Justification のみ ※4 Pattern のみ

			※5 Pattern で Loop のみ ※6 Pattern で、Choice か Multiplicity のみ
rootBasicLink	リンク		id : ID name : 名前 desc : 文章 attachment : 添付 status : 状況 message : メッセージ siblingOrder : 兄弟順序 type : 種類(下記から選択) SupportedBy InContextOf Responsibility DcaseLink004

2.2.4. スキーマ定義

2.2.1 および 2.2.2 に従ったスキーマ定義を以下に示す。

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<xsd:schema xmlns:dcase="http://www.dependable-os.net/2013/11/dcase_model/"
            xmlns:ecore="http://www.eclipse.org/emf/2002/Ecore"
            xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            ecore:nsPrefix="dcase"
            ecore:package="net.dependableos.dcase"
            targetNamespace="http://www.dependable-os.net/2013/11/dcase_model/">
  <xsd:import namespace="http://www.eclipse.org/emf/2002/Ecore"
              schemaLocation="Ecore.xsd"/>
  <xsd:element ecore:ignore="true" name="BasicNode" type="dcase:BasicNode"/>
  <xsd:element ecore:ignore="true" name="Argument" type="dcase:Argument"/>
  <xsd:element ecore:ignore="true" name="Goal" type="dcase:Goal"/>
  <xsd:element ecore:ignore="true" name="Strategy" type="dcase:Strategy"/>
  <xsd:element ecore:ignore="true" name="Evidence" type="dcase:Evidence"/>
  <xsd:element ecore:ignore="true" name="Monitor" type="dcase:Monitor"/>
  <xsd:element ecore:ignore="true" name="Undeveloped" type="dcase:Undeveloped"/>
  <xsd:element ecore:ignore="true" name="Context" type="dcase:Context"/>
  <xsd:element ecore:ignore="true" name="Justification" type="dcase:Justification"/>
  <xsd:element ecore:ignore="true" name="Assumption" type="dcase:Assumption"/>
  <xsd:element ecore:ignore="true" name="Module" type="dcase:Module"/>
  <xsd:element ecore:ignore="true" name="Contract" type="dcase:Contract"/>
  <xsd:element ecore:ignore="true" name="Pattern" type="dcase:Pattern"/>
  <xsd:element ecore:ignore="true" name="Action" type="dcase:Action"/>
  <xsd:element ecore:ignore="true" name="External" type="dcase:External"/>
  <xsd:element ecore:ignore="true" name="Userdef002" type="dcase:Userdef002"/>
  <xsd:element ecore:ignore="true" name="Userdef003" type="dcase:Userdef003"/>
  <xsd:element ecore:ignore="true" name="BasicLink" type="dcase:BasicLink"/>
  <xsd:element ecore:ignore="true" name="SupportedBy" type="dcase:SupportedBy"/>
  <xsd:element ecore:ignore="true" name="InContextOf" type="dcase:InContextOf"/>
  <xsd:element ecore:ignore="true" name="Responsibility" type="dcase:Responsibility"/>
  <xsd:element ecore:ignore="true" name="DcaseLink004" type="dcase:DcaseLink004"/>
  <xsd:complexType abstract="true" name="BasicNode">
    <xsd:attribute name="id" type="ecore:EString"/>
    <xsd:attribute name="name" type="ecore:EString"/>
    <xsd:attribute name="desc" type="ecore:EString"/>
    <xsd:attribute name="parameterizedDesc" type="ecore:EString"/>
    <xsd:attribute name="attachment" type="ecore:EString"/>
```

```
    <xsd:attribute name="status" type="ecore:EString"/>
    <xsd:attribute name="message" type="ecore:EString"/>
    <xsd:attribute name="requirement" type="ecore:EString"/>
    <xsd:attribute name="parameterDefs" type="ecore:EString"/>
    <xsd:attribute name="parameterVals" type="ecore:EString"/>
    <xsd:attribute name="parent" type="ecore:EString"/>
    <xsd:attribute name="refSource" type="ecore:EString"/>
    <xsd:attribute name="flag" type="ecore:EString"/>
    <xsd:attribute name="respName" type="ecore:EString"/>
    <xsd:attribute name="respAddress" type="ecore:EString"/>
    <xsd:attribute name="respIcon" type="ecore:EString"/>
    <xsd:attribute name="validUntil" type="ecore:EString"/>
    <xsd:attribute name="userdef001" type="ecore:EString"/>
    <xsd:attribute name="userdef002" type="ecore:EString"/>
    <xsd:attribute name="userdef003" type="ecore:EString"/>
    <xsd:attribute name="userdef004" type="ecore:EString"/>
    <xsd:attribute name="userdef005" type="ecore:EString"/>
    <xsd:attribute name="userdef006" type="ecore:EString"/>
    <xsd:attribute name="userdef007" type="ecore:EString"/>
    <xsd:attribute name="userdef008" type="ecore:EString"/>
    <xsd:attribute name="userdef009" type="ecore:EString"/>
    <xsd:attribute name="userdef010" type="ecore:EString"/>
    <xsd:attribute name="userdef011" type="ecore:EString"/>
    <xsd:attribute name="userdef012" type="ecore:EString"/>
    <xsd:attribute name="userdef013" type="ecore:EString"/>
    <xsd:attribute name="userdef014" type="ecore:EString"/>
    <xsd:attribute name="userdef015" type="ecore:EString"/>
    <xsd:attribute name="userdef016" type="ecore:EString"/>
  </xsd:complexType>
  <xsd:complexType name="Argument">
    <xsd:complexContent>
      <xsd:extension base="dcase:BasicNode">
        <xsd:sequence>
          <xsd:element maxOccurs="unbounded" minOccurs="0" name="rootBasicNode"
                       type="dcase:BasicNode"/>
          <xsd:element maxOccurs="unbounded" minOccurs="0" name="rootBasicLink"
                       type="dcase:BasicLink"/>
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>
  <xsd:complexType name="Goal">
    <xsd:complexContent>
      <xsd:extension base="dcase:BasicNode">
        <xsd:attribute default="0" ecore:unsettable="false" name="score"
                       type="ecore:EBigDecimal"/>
        <xsd:attribute default="1" ecore:unsettable="false" name="weight"
                       type="ecore:EInt"/>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>
  <xsd:complexType name="Strategy">
    <xsd:complexContent>
      <xsd:extension base="dcase:BasicNode"/>
    </xsd:complexContent>
  </xsd:complexType>
  <xsd:complexType name="Evidence">
    <xsd:complexContent>
      <xsd:extension base="dcase:BasicNode"/>
    </xsd:complexContent>
  </xsd:complexType>
  <xsd:complexType name="Monitor">
    <xsd:complexContent>
      <xsd:extension base="dcase:BasicNode">
        <xsd:attribute default="false" ecore:unsettable="false" name="isNormal"
                       type="ecore:EBoolean"/>
      </xsd:extension>
    </xsd:complexContent>
```

```
  </xsd:complexType>
  <xsd:complexType name="Undeveloped">
    <xsd:complexContent>
      <xsd:extension base="dcase:BasicNode"/>
    </xsd:complexContent>
  </xsd:complexType>
  <xsd:complexType name="Context">
    <xsd:complexContent>
      <xsd:extension base="dcase:BasicNode"/>
    </xsd:complexContent>
  </xsd:complexType>
  <xsd:complexType name="Justification">
    <xsd:complexContent>
      <xsd:extension base="dcase:BasicNode">
        <xsd:attribute default="" ecore:unsettable="false" name="stakeholder"
                       type="ecore:EString"/>
        <xsd:attribute name="riskAnalysis" type="ecore:EString"/>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>
  <xsd:complexType name="Assumption">
    <xsd:complexContent>
      <xsd:extension base="dcase:BasicNode"/>
    </xsd:complexContent>
  </xsd:complexType>
  <xsd:complexType name="Module">
    <xsd:complexContent>
      <xsd:extension base="dcase:BasicNode"/>
    </xsd:complexContent>
  </xsd:complexType>
  <xsd:complexType name="Contract">
    <xsd:complexContent>
      <xsd:extension base="dcase:BasicNode"/>
    </xsd:complexContent>
  </xsd:complexType>
  <xsd:complexType name="Pattern">
    <xsd:complexContent>
      <xsd:extension base="dcase:BasicNode">
        <xsd:attribute name="subType">
          <xsd:simpleType>
            <xsd:restriction base="ecore:EString">
              <xsd:enumeration value="Parameter"/>
              <xsd:enumeration value="Loop"/>
              <xsd:enumeration value="Choice"/>
              <xsd:enumeration value="Multiplicity"/>
            </xsd:restriction>
          </xsd:simpleType>
        </xsd:attribute>
        <xsd:attribute name="leafNode" type="ecore:EString"/>
        <xsd:attribute name="i" type="ecore:EString"/>
        <xsd:attribute name="j" type="ecore:EString"/>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>
  <xsd:complexType name="Action">
    <xsd:complexContent>
      <xsd:extension base="dcase:BasicNode"/>
    </xsd:complexContent>
  </xsd:complexType>
  <xsd:complexType name="External">
    <xsd:complexContent>
      <xsd:extension base="dcase:BasicNode"/>
    </xsd:complexContent>
  </xsd:complexType>
  <xsd:complexType name="Userdef002">
    <xsd:complexContent>
      <xsd:extension base="dcase:BasicNode"/>
    </xsd:complexContent>
```

```
  </xsd:complexType>
  <xsd:complexType name="Userdef003">
    <xsd:complexContent>
      <xsd:extension base="dcase:BasicNode"/>
    </xsd:complexContent>
  </xsd:complexType>
  <xsd:complexType abstract="true" name="BasicLink">
    <xsd:attribute ecore:reference="dcase:BasicNode" name="source" type="xsd:anyURI"/>
    <xsd:attribute ecore:reference="dcase:BasicNode" name="target" type="xsd:anyURI"/>
    <xsd:attribute name="id" type="ecore:EString"/>
    <xsd:attribute name="name" type="ecore:EString"/>
    <xsd:attribute name="desc" type="ecore:EString"/>
    <xsd:attribute name="attachment" type="ecore:EString"/>
    <xsd:attribute name="status" type="ecore:EString"/>
    <xsd:attribute name="message" type="ecore:EString"/>
    <xsd:attribute name="siblingOrder" type="ecore:EString"/>
    <xsd:attribute name="userdef001" type="ecore:EString"/>
    <xsd:attribute name="userdef002" type="ecore:EString"/>
    <xsd:attribute name="userdef003" type="ecore:EString"/>
    <xsd:attribute name="userdef004" type="ecore:EString"/>
    <xsd:attribute name="userdef005" type="ecore:EString"/>
    <xsd:attribute name="userdef006" type="ecore:EString"/>
    <xsd:attribute name="userdef007" type="ecore:EString"/>
    <xsd:attribute name="userdef008" type="ecore:EString"/>
    <xsd:attribute name="userdef009" type="ecore:EString"/>
    <xsd:attribute name="userdef010" type="ecore:EString"/>
    <xsd:attribute name="userdef011" type="ecore:EString"/>
    <xsd:attribute name="userdef012" type="ecore:EString"/>
    <xsd:attribute name="userdef013" type="ecore:EString"/>
    <xsd:attribute name="userdef014" type="ecore:EString"/>
    <xsd:attribute name="userdef015" type="ecore:EString"/>
    <xsd:attribute name="userdef016" type="ecore:EString"/>
  </xsd:complexType>
  <xsd:complexType name="SupportedBy">
    <xsd:complexContent>
      <xsd:extension base="dcase:BasicLink"/>
    </xsd:complexContent>
  </xsd:complexType>
  <xsd:complexType name="InContextOf">
    <xsd:complexContent>
      <xsd:extension base="dcase:BasicLink"/>
    </xsd:complexContent>
  </xsd:complexType>
  <xsd:complexType name="Responsibility">
    <xsd:complexContent>
      <xsd:extension base="dcase:BasicLink"/>
    </xsd:complexContent>
  </xsd:complexType>
  <xsd:complexType name="DcaseLink004">
    <xsd:complexContent>
      <xsd:extension base="dcase:BasicLink"/>
    </xsd:complexContent>
  </xsd:complexType>
</xsd:schema>
```

2.3. SACM のパラメータ拡張

SACM(Structured Assurance Case Metamodel)は、ARM と SAEM(Software Assurance Evidence Metamodel)を統合した仕様である。

ARM と同様、パラメータに関する情報を格納するための要素、属性が定義されていない。そのため、下記の拡張を行うことで、パラメータに対応させる。

- ArgumentElement ノードに、パラメータ定義およびパラメータ値を格納するための属性「parameterDefs」「parameterVals」を、それぞれ追加する
- ModelElement ノードに、パラメータ値を含む文章を格納するための属性「parameterizedContent」を追加する (GMF モデル情報ファイルの「parameterizedDesc」属性と同等)

それぞれの詳細については、GMF モデル情報ファイルの対応する属性と同じである。

これらを反映したスキーマ定義を、以下に示す。

なお、GMF モデル情報ファイルから SACM への変換については、2.4.5 を参照されたい。

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<xsd:schema xmlns:ARM="www.omg.org/spec/SACM/20120501/Argumentation"
            xmlns:EM="www.omg.org/spec/SACM/20120501/Evidence"
            xmlns:SACM="www.omg.org/spec/SACM/20120501"
            xmlns:xmi="http://www.omg.org/XMI"
            xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            targetNamespace="www.omg.org/spec/SACM/20120501/Argumentation">
  <xsd:import namespace="http://www.omg.org/XMI" schemaLocation="XMI.xsd"/>
  <xsd:import namespace="www.omg.org/spec/SACM/20120501" schemaLocation="SACM.xsd"/>
  <xsd:import namespace="www.omg.org/spec/SACM/20120501/Evidence"
              schemaLocation="Evidence.xsd"/>
  <xsd:complexType name="Argumentation">
    <xsd:complexContent>
      <xsd:extension base="ARM:ArgumentationElement">
        <xsd:choice maxOccurs="unbounded" minOccurs="0">
          <xsd:element name="argumentation" type="ARM:Argumentation"/>
          <xsd:element name="argumentElement" type="ARM:ArgumentElement"/>
        </xsd:choice>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>
  <xsd:element name="Argumentation" type="ARM:Argumentation"/>
  <xsd:complexType abstract="true" name="ArgumentElement">
    <xsd:complexContent>
      <xsd:extension base="ARM:ArgumentationElement"/>
    </xsd:complexContent>
  </xsd:complexType>
  <xsd:element name="ArgumentElement" type="ARM:ArgumentElement"/>
  <xsd:complexType name="CitationElement">
    <xsd:complexContent>
      <xsd:extension base="ARM:ArgumentElement">
        <xsd:choice maxOccurs="unbounded" minOccurs="0">
          <xsd:element name="argumentElementReference" type="ARM:ArgumentElement"/>
          <xsd:element name="argumentationReference" type="ARM:Argumentation"/>
        </xsd:choice>
        <xsd:attribute name="argumentElementReference" type="xsd:string"/>
        <xsd:attribute name="argumentationReference" type="xsd:string"/>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>
  <xsd:element name="CitationElement" type="ARM:CitationElement"/>
  <xsd:complexType name="InformationElement">
```

```
    <xsd:complexContent>
      <xsd:extension base="ARM:ArgumentElement">
        <xsd:choice maxOccurs="unbounded" minOccurs="0">
          <xsd:element name="evidence" type="EM:EvidenceItem"/>
        </xsd:choice>
        <xsd:attribute name="url" type="xsd:string" use="required"/>
        <xsd:attribute name="evidence" type="xsd:string"/>
        <xsd:attribute name="parameterDefs" type="xsd:string"/>
        <xsd:attribute name="parameterVals" type="xsd:string"/>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>
  <xsd:element name="InformationElement" type="ARM:InformationElement"/>
  <xsd:complexType abstract="true" name="ReasoningElement">
    <xsd:complexContent>
      <xsd:extension base="ARM:ArgumentElement"/>
    </xsd:complexContent>
  </xsd:complexType>
  <xsd:element name="ReasoningElement" type="ARM:ReasoningElement"/>
  <xsd:complexType abstract="true" name="Assertion">
    <xsd:complexContent>
      <xsd:extension base="ARM:ReasoningElement"/>
    </xsd:complexContent>
  </xsd:complexType>
  <xsd:element name="Assertion" type="ARM:Assertion"/>
  <xsd:complexType name="ArgumentReasoning">
    <xsd:complexContent>
      <xsd:extension base="ARM:ReasoningElement">
        <xsd:choice maxOccurs="unbounded" minOccurs="0">
          <xsd:element name="structure" type="ARM:Argumentation"/>
          <xsd:element name="describedInference" type="ARM:AssertedInference"/>
          <xsd:element name="describedChallenge" type="ARM:AssertedChallenge"/>
        </xsd:choice>
        <xsd:attribute name="structure" type="xsd:string"/>
        <xsd:attribute name="describedInference" type="xsd:string"/>
        <xsd:attribute name="describedChallenge" type="xsd:string"/>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>
  <xsd:element name="ArgumentReasoning" type="ARM:ArgumentReasoning"/>
  <xsd:complexType abstract="true" name="AssertedRelationship">
    <xsd:complexContent>
      <xsd:extension base="ARM:Assertion">
        <xsd:choice maxOccurs="unbounded" minOccurs="0">
          <xsd:element name="source" type="ARM:ArgumentElement"/>
          <xsd:element name="target" type="ARM:ArgumentElement"/>
        </xsd:choice>
        <xsd:attribute name="source" type="xsd:string"/>
        <xsd:attribute name="target" type="xsd:string"/>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>
  <xsd:element name="AssertedRelationship" type="ARM:AssertedRelationship"/>
  <xsd:complexType name="Claim">
    <xsd:complexContent>
      <xsd:extension base="ARM:Assertion">
        <xsd:attribute name="assumed" type="xsd:string" use="required"/>
        <xsd:attribute name="toBeSupported" type="xsd:string" use="required"/>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>
  <xsd:element name="Claim" type="ARM:Claim"/>
  <xsd:complexType name="AssertedInference">
    <xsd:complexContent>
      <xsd:extension base="ARM:AssertedRelationship"/>
    </xsd:complexContent>
  </xsd:complexType>
  <xsd:element name="AssertedInference" type="ARM:AssertedInference"/>
```

```
  <xsd:complexType name="AssertedEvidence">
    <xsd:complexContent>
      <xsd:extension base="ARM:AssertedRelationship"/>
    </xsd:complexContent>
  </xsd:complexType>
  <xsd:element name="AssertedEvidence" type="ARM:AssertedEvidence"/>
  <xsd:complexType name="AssertedContext">
    <xsd:complexContent>
      <xsd:extension base="ARM:AssertedRelationship"/>
    </xsd:complexContent>
  </xsd:complexType>
  <xsd:element name="AssertedContext" type="ARM:AssertedContext"/>
  <xsd:complexType name="AssertedChallenge">
    <xsd:complexContent>
      <xsd:extension base="ARM:AssertedRelationship"/>
    </xsd:complexContent>
  </xsd:complexType>
  <xsd:element name="AssertedChallenge" type="ARM:AssertedChallenge"/>
  <xsd:complexType name="AssertedCounterEvidence">
    <xsd:complexContent>
      <xsd:extension base="ARM:AssertedRelationship"/>
    </xsd:complexContent>
  </xsd:complexType>
  <xsd:element name="AssertedCounterEvidence" type="ARM:AssertedCounterEvidence"/>
  <xsd:complexType abstract="true" name="ArgumentationElement">
    <xsd:complexContent>
      <xsd:extension base="SACM:ModelElement">
        <xsd:attribute name="description" type="xsd:string" use="required"/>
        <xsd:attribute name="content" type="xsd:string" use="required"/>
        <xsd:attribute name="parameterizedContent" type="xsd:string"/>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>
  <xsd:element name="ArgumentationElement" type="ARM:ArgumentationElement"/>
</xsd:schema>
```

2.4. データ変換

D-Case Editor で D-Case ファイルを閲覧・編集するには、GMF モデル情報ファイルに変換する必要がある。

D-Case Editor には、D-Case ファイルを GMF モデル情報ファイルに変換する機能があり、これを用いて変換することで、D-Case の閲覧・編集が可能となる。

また、他の D-Case ツールで同文書を閲覧・編集するため、GMF モデル情報ファイルから D-Case ファイルに変換する機能も有している。

2.4.1. D-Case ファイルから GMF モデル情報ファイルへの変換

従来の D-Case Editor には、「D-Case から GMF への変換」機能があるため、これを、今回定義した XML 仕様に準拠させる。

変換には、XSLT(Extensible Stylesheet Language Transformations)を使用している。

D-Case ファイルを GMF モデル情報ファイルに変換するための XSLT コードを以下に示す。

```
<?xml version="1.0" encoding="UTF-8"?>

<!--/*****************************************************************************
     * Copyright (C) Yutaka Matsuno 2010-2012 All rights reserved.

*****************************************************************************/-->

<!--Transforms D-Case to GMF-->
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
                xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                xmlns:dcase="http://www.dependable-os.net/2013/11/dcase_model/"
                xmlns:dcase_std="http://www.dependable-os.net/2013/11/dcase"

xmlns:dfunc="net.dependableos.dcase.diagram.common.xml.XsltExtFunctionUtil"
                xmlns:dre="http://www.dependable-os.net/dre"
                exclude-result-prefixes="dcase_std"
                extension-element-prefixes="result dfunc"
                version="1.0">

  <xsl:output method="xml" encoding="UTF-8" indent="yes" />

  <!--Root-->
  <xsl:template match="/">
    <dcase:Argument>
      <xsl:apply-templates select="dcase_std:dcase"/>
    </dcase:Argument>
  </xsl:template>

  <xsl:template match="dcase_std:dcase">
    <xsl:call-template name="BasicAttribute"/>
    <xsl:apply-templates select="dcase_std:nodes/dcase_std:node"/>
    <xsl:apply-templates select="dcase_std:links/dcase_std:link">
      <xsl:sort select="./@source" data-type="text" order="ascending"/>
      <xsl:sort
select="number(./dcase_std:properties/dcase_std:property[@name='SiblingOrder']/@value)
" data-type="number" order="ascending"/>
    </xsl:apply-templates>
  </xsl:template>

  <!--Node-->
  <xsl:template match="dcase_std:node">
```

```
    <rootBasicNode>
      <xsl:attribute name="xsi:type">
        <xsl:choose>
          <xsl:when test="@type='Goal'">dcase:Goal</xsl:when>
          <xsl:when test="@type='Strategy'">dcase:Strategy</xsl:when>
          <xsl:when test="@type='Evidence'">dcase:Evidence</xsl:when>
          <xsl:when test="@type='Undeveloped'">dcase:Undeveloped</xsl:when>
          <xsl:when test="@type='Context'">dcase:Context</xsl:when>
          <xsl:when test="@type='Monitor'">dcase:Monitor</xsl:when>
          <xsl:when test="@type='Justification'">dcase:Justification</xsl:when>
          <xsl:when test="@type='Userdef002'">dcase:Userdef002</xsl:when>
          <xsl:when test="@type='Userdef003'">dcase:Userdef003</xsl:when>
          <xsl:when test="@type='Assumption'">dcase:Assumption</xsl:when>
          <xsl:when test="@type='Module'">dcase:Module</xsl:when>
          <xsl:when test="@type='Contract'">dcase:Contract</xsl:when>
          <xsl:when test="@type='Action'">dcase:Action</xsl:when>
          <xsl:when test="@type='External'">dcase:External</xsl:when>
          <xsl:when test="@type='Pattern'">dcase:Pattern</xsl:when>
          <xsl:otherwise>undefined</xsl:otherwise>
        </xsl:choose>
      </xsl:attribute>
      <xsl:call-template name="BasicAttribute"/>
    </rootBasicNode>
  </xsl:template>


  <!--Link-->
  <xsl:template match="dcase_std:link">
    <rootBasicLink>
      <xsl:attribute name="xsi:type">
        <xsl:choose>
          <xsl:when test="@type='SupportedBy'">dcase:SupportedBy</xsl:when>
          <xsl:when test="@type='InContextOf'">dcase:InContextOf</xsl:when>
          <xsl:when test="@type='Responsibility'">dcase:Responsibility</xsl:when>
          <xsl:when test="@type='Link004'">dcase:DcaseLink004</xsl:when>
         <xsl:otherwise>dcase:DcaseLink001</xsl:otherwise>
        </xsl:choose>
      </xsl:attribute>
      <xsl:apply-templates select="@source|@target"/>
      <xsl:call-template name="BasicAttribute"/>
    </rootBasicLink>
  </xsl:template>

  <!--Attribute-->
  <xsl:template name="BasicAttribute">
    <xsl:apply-templates select="@id|@name|@status" mode="basic"/>
    <xsl:call-template name="Description"/>
    <xsl:apply-templates
select="dcase_std:properties/dcase_std:property"></xsl:apply-templates>
    <xsl:if test="dcase_std:responsibility!='NaN'">
      <xsl:apply-templates select="dcase_std:responsibility"/>
    </xsl:if>
    <xsl:if test="dcase_std:parameters!='NaN'">
      <xsl:apply-templates select="dcase_std:parameters"/>
    </xsl:if>
    <xsl:if test="dre:d-script!='NaN'">
      <xsl:apply-templates select="dre:d-script"/>
    </xsl:if>
  </xsl:template>

  <xsl:template name="Description">
    <xsl:attribute name="desc">
      <xsl:value-of select="dcase_std:description"/>
    </xsl:attribute>
  </xsl:template>

  <!--Attribute value-->
  <xsl:template match="@*" mode="basic">
```

```
    <xsl:attribute name="{local-name()}">
      <xsl:value-of select="."/>
    </xsl:attribute>
  </xsl:template>

  <!--Source and Target of the Link-->
  <xsl:template match="@source|@target">
    <xsl:attribute name="{local-name()}">
      <xsl:value-of select="concat('#',.) "/>
    </xsl:attribute>
  </xsl:template>

  <!--Property-->
  <xsl:template match="dcase_std:property">
    <xsl:choose>
      <xsl:when test="@name='Attachment'">
        <xsl:attribute name="attachment">
          <xsl:value-of select="@value"/>
        </xsl:attribute>
      </xsl:when>
      <xsl:when test="@name='Userdef001'">
        <xsl:attribute name="userdef001">
          <xsl:value-of select="@value"/>
        </xsl:attribute>
      </xsl:when>
      <xsl:when test="@name='Userdef002'">
        <xsl:attribute name="userdef002">
          <xsl:value-of select="@value"/>
        </xsl:attribute>
      </xsl:when>
      <xsl:when test="@name='Userdef003'">
        <xsl:attribute name="userdef003">
          <xsl:value-of select="@value"/>
        </xsl:attribute>
      </xsl:when>
      <xsl:when test="@name='Userdef004'">
        <xsl:attribute name="userdef004">
          <xsl:value-of select="@value"/>
        </xsl:attribute>
      </xsl:when>
      <xsl:when test="@name='Userdef005'">
        <xsl:attribute name="userdef005">
          <xsl:value-of select="@value"/>
        </xsl:attribute>
      </xsl:when>
      <xsl:when test="@name='Userdef006'">
        <xsl:attribute name="userdef006">
          <xsl:value-of select="@value"/>
        </xsl:attribute>
      </xsl:when>
      <xsl:when test="@name='Userdef007'">
        <xsl:attribute name="userdef007">
          <xsl:value-of select="@value"/>
        </xsl:attribute>
      </xsl:when>
      <xsl:when test="@name='Userdef008'">
        <xsl:attribute name="userdef008">
          <xsl:value-of select="@value"/>
        </xsl:attribute>
      </xsl:when>
      <xsl:when test="@name='Userdef009'">
        <xsl:attribute name="userdef009">
          <xsl:value-of select="@value"/>
        </xsl:attribute>
      </xsl:when>
      <xsl:when test="@name='Userdef010'">
        <xsl:attribute name="userdef010">
          <xsl:value-of select="@value"/>
```

```
        </xsl:attribute>
      </xsl:when>
      <xsl:when test="@name='Userdef011'">
        <xsl:attribute name="userdef011">
          <xsl:value-of select="@value"/>
        </xsl:attribute>
      </xsl:when>
      <xsl:when test="@name='Userdef012'">
        <xsl:attribute name="userdef012">
          <xsl:value-of select="@value"/>
        </xsl:attribute>
      </xsl:when>
      <xsl:when test="@name='Userdef013'">
        <xsl:attribute name="userdef013">
          <xsl:value-of select="@value"/>
        </xsl:attribute>
      </xsl:when>
      <xsl:when test="@name='Userdef014'">
        <xsl:attribute name="userdef014">
          <xsl:value-of select="@value"/>
        </xsl:attribute>
      </xsl:when>
      <xsl:when test="@name='Userdef015'">
        <xsl:attribute name="userdef015">
          <xsl:value-of select="@value"/>
        </xsl:attribute>
      </xsl:when>
      <xsl:when test="@name='Userdef016'">
        <xsl:attribute name="userdef016">
          <xsl:value-of select="@value"/>
        </xsl:attribute>
      </xsl:when>
      <xsl:when test="@name='Score'">
        <xsl:attribute name="score">
          <xsl:apply-templates select="@value" mode="Decimal"></xsl:apply-templates>
        </xsl:attribute>
      </xsl:when>
      <xsl:when test="@name='Weight'">
        <xsl:attribute name="weight">
          <xsl:apply-templates select="@value"
mode="IntegerlorOver"></xsl:apply-templates>
        </xsl:attribute>
      </xsl:when>
      <xsl:when test="@name='NodeLink'">
        <xsl:attribute name="nodeLink">
          <xsl:value-of select="@value"/>
        </xsl:attribute>
      </xsl:when>
      <xsl:when test="@name='IsNormal'">
        <xsl:attribute name="isNormal">
          <xsl:value-of select="@value"/>
        </xsl:attribute>
      </xsl:when>
      <xsl:when test="@name='Stakeholder'">
        <xsl:attribute name="stakeholder">
          <xsl:value-of select="@value"/>
        </xsl:attribute>
      </xsl:when>
      <xsl:when test="@name='RiskAnalysis'">
        <xsl:attribute name="riskAnalysis">
          <xsl:value-of select="@value"/>
        </xsl:attribute>
      </xsl:when>
      <xsl:when test="@name='Message'">
        <xsl:attribute name="message">
          <xsl:value-of select="@value"/>
        </xsl:attribute>
      </xsl:when>
```

```
      <xsl:when test="@name='Requirement'">
        <xsl:attribute name="requirement">
          <xsl:value-of select="@value"/>
        </xsl:attribute>
      </xsl:when>
      <xsl:when test="@name='ParameterizedDesc'">
        <xsl:attribute name="parameterizedDesc">
          <xsl:value-of select="@value"/>
        </xsl:attribute>
      </xsl:when>
      <xsl:when test="@name='Parent'">
        <xsl:attribute name="parent">
          <xsl:value-of select="@value"/>
        </xsl:attribute>
      </xsl:when>
      <xsl:when test="@name='RefSource'">
        <xsl:attribute name="refSource">
          <xsl:value-of select="@value"/>
        </xsl:attribute>
      </xsl:when>
      <xsl:when test="@name='Flag'">
        <xsl:attribute name="flag">
          <xsl:value-of select="@value"/>
        </xsl:attribute>
      </xsl:when>
      <xsl:when test="@name='LeafNode'">
        <xsl:attribute name="leafNode">
          <xsl:value-of select="@value"/>
        </xsl:attribute>
      </xsl:when>
      <xsl:when test="@name='I'">
        <xsl:attribute name="i">
          <xsl:value-of select="@value"/>
        </xsl:attribute>
      </xsl:when>
      <xsl:when test="@name='J'">
        <xsl:attribute name="j">
          <xsl:value-of select="@value"/>
        </xsl:attribute>
      </xsl:when>
      <xsl:when test="@name='SiblingOrder'">
        <xsl:attribute name="siblingOrder">
          <xsl:value-of select="@value"/>
        </xsl:attribute>
      </xsl:when>
      <xsl:when test="@name='ValidUntil'">
        <xsl:attribute name="validUntil">
          <xsl:value-of select="@value"/>
        </xsl:attribute>
      </xsl:when>
    </xsl:choose>
  </xsl:template>

  <!--Decimal-->
  <xsl:template match="@*" mode="Decimal">
    <xsl:choose>
      <xsl:when test="string(number(.))='NaN'">0</xsl:when>
      <xsl:otherwise>
        <xsl:value-of select="."/>
      </xsl:otherwise>
    </xsl:choose>
  </xsl:template>

  <!--Integer-->
  <xsl:template match="@*" mode="Integer1orOver">
    <xsl:variable name="num" select="floor(.)"></xsl:variable>
    <xsl:choose>
      <xsl:when test="string($num)='NaN' or $num&lt;1">1</xsl:when>
```

```
      <xsl:otherwise>
        <xsl:value-of select="$num"/>
      </xsl:otherwise>
    </xsl:choose>
  </xsl:template>

  <!--Responsibility-->
  <xsl:template match="dcase_std:responsibility">
    <xsl:if test="@name">
      <xsl:attribute name="respName">
        <xsl:value-of select="@name"/>
      </xsl:attribute>
    </xsl:if>
    <xsl:if test="@address">
      <xsl:attribute name="respAddress">
        <xsl:value-of select="@address"/>
      </xsl:attribute>
    </xsl:if>
    <xsl:if test="@icon">
      <xsl:attribute name="respIcon">
        <xsl:value-of select="@icon"/>
      </xsl:attribute>
    </xsl:if>
  </xsl:template>

  <!--Parameter-->
  <xsl:template match="dcase_std:parameters" >
    <xsl:attribute name="parameterVals">
      <xsl:value-of select="dfunc:parameterizeVals(.)"/>
    </xsl:attribute>
    <xsl:attribute name="parameterDefs">
      <xsl:value-of select="dfunc:parameterizeDefs(.)"/>
    </xsl:attribute>
  </xsl:template>

  <!-- d-script -->
  <xsl:template match="dre:d-script" >
    <xsl:attribute name="userdef011">
      <xsl:value-of select="dfunc:serialize(.)"/>
    </xsl:attribute>
  </xsl:template>

</xsl:stylesheet>
```

2.4.2. GMF モデル情報ファイルから D-Case ファイルへの変換

同様に、GMF モデル情報ファイルを D-Case ファイルに変換するための XSLT コードを以下に示す。

```
<?xml version="1.0" encoding="UTF-8"?>

<!--/*****************************************************************************
     * Copyright (C) Yutaka Matsuno 2010-2012 All rights reserved.

*****************************************************************************/-->

<!--Transforms GMF to D-Case-->
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
                xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                xmlns:dcase_gmf="http://www.dependable-os.net/2013/11/dcase_model/"
                xmlns:dcase="http://www.dependable-os.net/2013/11/dcase"
                xmlns:dre="http://www.dependable-os.net/dre"
                xmlns:exsl="http://exslt.org/common"

xmlns:dfunc="net.dependableos.dcase.diagram.common.xml.XsltExtFunctionUtil"
                extension-element-prefixes="exsl dre"
                version="1.0">

  <xsl:output method="xml" encoding="UTF-8" indent="yes"/>

  <!--Root-->
  <xsl:template match="/">
    <xsl:element name="dcase:dcase"
namespace="http://www.dependable-os.net/2013/11/dcase">
      <xsl:apply-templates select="dcase_gmf:Argument"/>
    </xsl:element>
  </xsl:template>

  <xsl:template match="dcase_gmf:Argument">
    <!-- Descendant links of the root node -->
    <xsl:variable name="linkResult">
      <xsl:for-each select="rootBasicNode">
        <xsl:variable name="id" select="concat('#',@id)"/>
        <xsl:variable name="count"
select="count(/dcase_gmf:Argument/rootBasicLink[@target=$id])"></xsl:variable>
        <xsl:if test="$count=0">
          <xsl:call-template name="RootBasicLink">
            <xsl:with-param name="source" select="$id"/>
          </xsl:call-template>
        </xsl:if>
      </xsl:for-each>
    </xsl:variable>

    <xsl:call-template name="ArgumentAttribute"/>
    <xsl:element name="dcase:description"
namespace="http://www.dependable-os.net/2013/11/dcase">
      <xsl:value-of select="@desc"/>
    </xsl:element>
    <xsl:call-template name="Properties"/>
    <xsl:if test="@respName!='NaN' or @respAddress!='NaN' or @respIcon!='NaN'">
      <xsl:call-template name="Responsibilities"/>
    </xsl:if>
    <xsl:if test="@parameterVals!='NaN' and string-length(@parameterVals) &gt; 0 and
@parameterDefs!='NaN' and string-length(@parameterDefs) &gt; 0">
      <xsl:call-template name="Parameters"/>
    </xsl:if>
    <xsl:element name="dcase:nodes"
namespace="http://www.dependable-os.net/2013/11/dcase">
      <xsl:apply-templates select="rootBasicNode"/>
```

```
    </xsl:element>
      <!-- Links-->
    <xsl:element name="dcase:links"
namespace="http://www.dependable-os.net/2013/11/dcase">
      <!-- Outputs descendant links of the root node eliminating duplicate links-->
      <xsl:for-each
select="exsl:node-set($linkResult)/dcase:link[not(@id=preceding::dcase:link/@id)]">
        <xsl:copy-of select="."/>
      </xsl:for-each>

      <!-- Outputs rest of the links -->
      <xsl:apply-templates
select="rootBasicLink[not(@id=exsl:node-set($linkResult)/dcase:link/@id)]"
mode="basic"></xsl:apply-templates>
    </xsl:element>
  </xsl:template>

  <!--Node-->
  <xsl:template match="rootBasicNode">
    <xsl:element name="dcase:node"
namespace="http://www.dependable-os.net/2013/11/dcase">
      <xsl:call-template name="BasicNodeAttribute"/>
      <xsl:element name="dcase:description"
namespace="http://www.dependable-os.net/2013/11/dcase">
        <xsl:value-of select="@desc"/>
      </xsl:element>
      <xsl:call-template name="Properties"/>
      <xsl:if test="@respName!='NaN' or @respAddress!='NaN' or @respIcon!='NaN'">
        <xsl:call-template name="Responsibilities"/>
      </xsl:if>
      <xsl:if test="@parameterVals!='NaN' and string-length(@parameterVals) &gt; 0 and
@parameterDefs!='NaN' and string-length(@parameterDefs) &gt; 0">
        <xsl:call-template name="Parameters"/>
      </xsl:if>
      <xsl:if test="@userdef011!='NaN' and string-length(@userdef011) &gt; 0">
        <xsl:call-template name="D-Script"/>
      </xsl:if>
    </xsl:element>
  </xsl:template>

  <!--Link (nodeset)-->
  <xsl:template name="RootBasicLink">
    <xsl:param name="ancestors"></xsl:param>
    <xsl:param name="source"></xsl:param>

    <!-- Ancestors -->
    <xsl:variable name="tmpAncestors">
      <xsl:copy-of select="$ancestors"/>
    </xsl:variable>

    <xsl:variable name="newAncestors">
      <xsl:copy-of select="$ancestors"/>
      <AncestorID>
        <xsl:value-of select="$source"/>
      </AncestorID>
    </xsl:variable>

    <!--Tests whether the link is looped-->
    <xsl:if test="not(exsl:node-set($tmpAncestors)/AncestorID[.=$source])">
      <xsl:apply-templates select="/dcase_gmf:Argument/rootBasicLink[(@source=$source)
and (string(number(@siblingOrder)) != 'NaN')]" mode="basic">
        <xsl:sort select="number(@siblingOrder)" data-type="number"
order="ascending"/>
      </xsl:apply-templates>
      <xsl:apply-templates select="/dcase_gmf:Argument/rootBasicLink[(@source=$source)
and (string(number(@siblingOrder)) = 'NaN')]" mode="basic"/>

      <xsl:apply-templates select="/dcase_gmf:Argument/rootBasicLink[(@source=$source)
```

```
and (string(number(@siblingOrder)) != 'NaN')]" mode="procChildren">
        <xsl:sort select="./@siblingOrder" data-type="number" order="ascending"/>
        <xsl:with-param name="ancestors" select="$newAncestors"/>
      </xsl:apply-templates>
      <xsl:apply-templates select="/dcase_gmf:Argument/rootBasicLink[(@source=$source)
and (string(number(@siblingOrder)) = 'NaN')]" mode="procChildren">
        <xsl:with-param name="ancestors" select="$newAncestors"/>
      </xsl:apply-templates>
    </xsl:if>
  </xsl:template>

  <!--Link-->
  <xsl:template match="rootBasicLink" mode="basic">
    <xsl:element name="dcase:link"
namespace="http://www.dependable-os.net/2013/11/dcase">
      <xsl:call-template name="BasicLinkAttribute"/>
      <xsl:element name="dcase:description"
namespace="http://www.dependable-os.net/2013/11/dcase">
        <xsl:value-of select="@desc"/>
      </xsl:element>
      <xsl:call-template name="Properties"/>
    </xsl:element>
  </xsl:template>

  <xsl:template match="rootBasicLink" mode="procChildren">
    <xsl:param name="ancestors"></xsl:param>
    <xsl:call-template name="RootBasicLink">
      <xsl:with-param name="ancestors" select="$ancestors"/>
      <xsl:with-param name="source" select="./@target"/>
    </xsl:call-template>
  </xsl:template>


  <!--Attributes of the argument-->
  <xsl:template name="ArgumentAttribute">
    <xsl:apply-templates select="@id|@name|@status" mode="basic"/>
  </xsl:template>


  <!--Attributes of a node-->
  <xsl:template name="BasicNodeAttribute">
    <xsl:call-template name="NodeType"/>
    <xsl:apply-templates select="@id|@name|@status" mode="basic"/>
  </xsl:template>

  <!--Node type-->
  <xsl:template name="NodeType">
    <xsl:attribute name="type">
      <xsl:choose>
        <xsl:when test="@xsi:type='dcase:Goal'">Goal</xsl:when>
        <xsl:when test="@xsi:type='dcase:Strategy'">Strategy</xsl:when>
        <xsl:when test="@xsi:type='dcase:Evidence'">Evidence</xsl:when>
        <xsl:when test="@xsi:type='dcase:Undeveloped'">Undeveloped</xsl:when>
        <xsl:when test="@xsi:type='dcase:Context'">Context</xsl:when>
        <xsl:when test="@xsi:type='dcase:Monitor'">Monitor</xsl:when>
        <xsl:when test="@xsi:type='dcase:Justification'">Justification</xsl:when>
        <xsl:when test="@xsi:type='dcase:Userdef002'">Userdef002</xsl:when>
        <xsl:when test="@xsi:type='dcase:Userdef003'">Userdef003</xsl:when>
        <xsl:when test="@xsi:type='dcase:Assumption'">Assumption</xsl:when>
        <xsl:when test="@xsi:type='dcase:Module'">Module</xsl:when>
        <xsl:when test="@xsi:type='dcase:Contract'">Contract</xsl:when>
        <xsl:when test="@xsi:type='dcase:Action'">Action</xsl:when>
        <xsl:when test="@xsi:type='dcase:Pattern'">Pattern</xsl:when>
        <xsl:when test="@xsi:type='dcase:External'">External</xsl:when>
        <xsl:otherwise>undefined</xsl:otherwise>
      </xsl:choose>
    </xsl:attribute>
  </xsl:template>
```

```
  <!--Attributes of a link-->
  <xsl:template name="BasicLinkAttribute">
    <xsl:call-template name="LinkType"/>
    <xsl:apply-templates select="@id|@name|@status" mode="basic"/>
    <xsl:apply-templates select="@source|@target" mode="link"/>
  </xsl:template>

  <!--Link type-->
  <xsl:template name="LinkType">
    <xsl:attribute name="type">
      <xsl:choose>
        <xsl:when test="@xsi:type='dcase:SupportedBy'">SupportedBy</xsl:when>
        <xsl:when test="@xsi:type='dcase:InContextOf'">InContextOf</xsl:when>
        <xsl:when test="@xsi:type='dcase:Responsibility'">Responsibility</xsl:when>
        <xsl:when test="@xsi:type='dcase:DcaseLink004'">Link004</xsl:when>
        <xsl:otherwise>SupportedBy</xsl:otherwise>
      </xsl:choose>
    </xsl:attribute>
  </xsl:template>

  <!--Attributes-->
  <xsl:template match="@*" mode="basic">
    <xsl:attribute name="{local-name()}">
      <xsl:value-of select="."/>
    </xsl:attribute>
  </xsl:template>

  <!--Source or target of a link-->
  <xsl:template match="@*" mode="link">
    <xsl:attribute name="{local-name()}">
      <xsl:value-of select="substring-after(.,'#') "/>
    </xsl:attribute>
  </xsl:template>

  <!--Properties-->
  <xsl:template name="Properties">
    <xsl:element name="dcase:properties"
namespace="http://www.dependable-os.net/2013/11/dcase">
      <xsl:apply-templates select="@attachment" mode="property">
        <xsl:with-param name="propertyName">Attachment</xsl:with-param>
      </xsl:apply-templates>
      <xsl:apply-templates select="@userdef001" mode="property">
        <xsl:with-param name="propertyName">Userdef001</xsl:with-param>
      </xsl:apply-templates>
      <xsl:apply-templates select="@userdef002" mode="property">
        <xsl:with-param name="propertyName">Userdef002</xsl:with-param>
      </xsl:apply-templates>
      <xsl:apply-templates select="@userdef003" mode="property">
        <xsl:with-param name="propertyName">Userdef003</xsl:with-param>
      </xsl:apply-templates>
      <xsl:apply-templates select="@userdef004" mode="property">
        <xsl:with-param name="propertyName">Userdef004</xsl:with-param>
      </xsl:apply-templates>
      <xsl:apply-templates select="@userdef005" mode="property">
        <xsl:with-param name="propertyName">Userdef005</xsl:with-param>
      </xsl:apply-templates>
      <xsl:apply-templates select="@userdef006" mode="property">
        <xsl:with-param name="propertyName">Userdef006</xsl:with-param>
      </xsl:apply-templates>
      <xsl:apply-templates select="@userdef007" mode="property">
        <xsl:with-param name="propertyName">Userdef007</xsl:with-param>
      </xsl:apply-templates>
      <xsl:apply-templates select="@userdef008" mode="property">
        <xsl:with-param name="propertyName">Userdef008</xsl:with-param>
      </xsl:apply-templates>
      <xsl:apply-templates select="@userdef009" mode="property">
        <xsl:with-param name="propertyName">Userdef009</xsl:with-param>
```

```
      </xsl:apply-templates>
      <xsl:apply-templates select="@userdef010" mode="property">
        <xsl:with-param name="propertyName">Userdef010</xsl:with-param>
      </xsl:apply-templates>
      <xsl:apply-templates select="@userdef011" mode="property">
        <xsl:with-param name="propertyName">Userdef011</xsl:with-param>
      </xsl:apply-templates>
      <xsl:apply-templates select="@userdef012" mode="property">
        <xsl:with-param name="propertyName">Userdef012</xsl:with-param>
      </xsl:apply-templates>
      <xsl:apply-templates select="@userdef013" mode="property">
        <xsl:with-param name="propertyName">Userdef013</xsl:with-param>
      </xsl:apply-templates>
      <xsl:apply-templates select="@userdef014" mode="property">
        <xsl:with-param name="propertyName">Userdef014</xsl:with-param>
      </xsl:apply-templates>
      <xsl:apply-templates select="@userdef015" mode="property">
        <xsl:with-param name="propertyName">Userdef015</xsl:with-param>
      </xsl:apply-templates>
      <xsl:apply-templates select="@userdef016" mode="property">
        <xsl:with-param name="propertyName">Userdef016</xsl:with-param>
      </xsl:apply-templates>
      <xsl:apply-templates select="@score" mode="property">
        <xsl:with-param name="propertyName">Score</xsl:with-param>
      </xsl:apply-templates>
      <xsl:apply-templates select="@weight" mode="property">
        <xsl:with-param name="propertyName">Weight</xsl:with-param>
      </xsl:apply-templates>
      <xsl:apply-templates select="@nodeLink" mode="property">
        <xsl:with-param name="propertyName">NodeLink</xsl:with-param>
      </xsl:apply-templates>
      <xsl:apply-templates select="@isNormal" mode="property">
        <xsl:with-param name="propertyName">IsNormal</xsl:with-param>
      </xsl:apply-templates>
      <xsl:apply-templates select="@stakeholder" mode="property">
        <xsl:with-param name="propertyName">Stakeholder</xsl:with-param>
      </xsl:apply-templates>
      <xsl:apply-templates select="@riskAnalysis" mode="property">
        <xsl:with-param name="propertyName">RiskAnalysis</xsl:with-param>
      </xsl:apply-templates>
      <xsl:apply-templates select="@message" mode="property">
        <xsl:with-param name="propertyName">Message</xsl:with-param>
      </xsl:apply-templates>
      <xsl:apply-templates select="@requirement" mode="property">
        <xsl:with-param name="propertyName">Requirement</xsl:with-param>
      </xsl:apply-templates>
      <xsl:apply-templates select="@parameterizedDesc" mode="property">
        <xsl:with-param name="propertyName">ParameterizedDesc</xsl:with-param>
      </xsl:apply-templates>
      <xsl:apply-templates select="@parent" mode="property">
        <xsl:with-param name="propertyName">Parent</xsl:with-param>
      </xsl:apply-templates>
      <xsl:apply-templates select="@refSource" mode="property">
        <xsl:with-param name="propertyName">RefSource</xsl:with-param>
      </xsl:apply-templates>
      <xsl:apply-templates select="@flag" mode="property">
        <xsl:with-param name="propertyName">Flag</xsl:with-param>
      </xsl:apply-templates>
      <xsl:apply-templates select="@leafNode" mode="property">
        <xsl:with-param name="propertyName">LeafNode</xsl:with-param>
      </xsl:apply-templates>
      <xsl:apply-templates select="@i" mode="property">
        <xsl:with-param name="propertyName">I</xsl:with-param>
      </xsl:apply-templates>
      <xsl:apply-templates select="@j" mode="property">
        <xsl:with-param name="propertyName">J</xsl:with-param>
      </xsl:apply-templates>
      <xsl:apply-templates select="@siblingOrder" mode="property">
```

```
        <xsl:with-param name="propertyName">SiblingOrder</xsl:with-param>
      </xsl:apply-templates>
      <xsl:apply-templates select="@validUntil" mode="property">
        <xsl:with-param name="propertyName">ValidUntil</xsl:with-param>
      </xsl:apply-templates>
    </xsl:element>
  </xsl:template>


  <!--Property values-->
  <xsl:template match="@*" mode="property">
    <xsl:param name="propertyName">undefined</xsl:param>
    <xsl:element name="dcase:property"
namespace="http://www.dependable-os.net/2013/11/dcase">
      <xsl:attribute name="name">
        <xsl:value-of select="$propertyName"/>
      </xsl:attribute>
      <xsl:attribute name="value">
        <xsl:value-of select="."/>
      </xsl:attribute>
    </xsl:element>
  </xsl:template>

  <!--Responsibilities-->
  <xsl:template name="Responsibilities">
    <xsl:element name="dcase:responsibility"
namespace="http://www.dependable-os.net/2013/11/dcase">
      <xsl:if test="@respName!='NaN'">
        <xsl:attribute name="name">
          <xsl:value-of select="@respName"/>
        </xsl:attribute>
      </xsl:if>
      <xsl:if test="@respAddress!='NaN'">
        <xsl:attribute name="address">
          <xsl:value-of select="@respAddress"/>
        </xsl:attribute>
      </xsl:if>
      <xsl:if test="@respIcon!='NaN'">
        <xsl:attribute name="icon">
          <xsl:value-of select="@respIcon"/>
        </xsl:attribute>
      </xsl:if>
    </xsl:element>
  </xsl:template>

  <!--Parameters-->
  <xsl:template name="Parameters">
    <xsl:copy-of select="dfunc:deparameterize(@parameterVals, @parameterDefs)"/>
  </xsl:template>

  <!-- d-script element -->
  <xsl:template name="D-Script">
    <xsl:copy-of select="dfunc:deserialize(@userdef011)"/>
  </xsl:template>

</xsl:stylesheet>
```

2.4.3. 古い GMF モデル情報ファイルからの変換

従来の D-Case Editor のバージョンで作成した GMF モデル情報ファイルとは、ノード名や属性名が異なる。旧バージョンの GMF モデル情報ファイルを、現バージョンで閲覧・編集できるようにするため、同様の変換機能を D-Case Editor に追加する。違いは以下の通り。

新 GMF	旧 GMF	種別
http://www.dependable-os.net/2013/11/dcase_model/	http://www.dependable-os.net/2010/03/dcase/	ネームスペース
Assumption	Userdef004	ノード
Module	Userdef005	ノード
Contract	Userdef006	ノード
Pattern	System	ノード
Action	Policy	ノード
External	Userdef001	ノード
SupportedBy	DcaseLink001	リンク
InContextOf	DcaseLink002	リンク
Responsibility	DcaseLink003	リンク
parameterizedDesc	userdef005	ノードの属性
message	userdef002	属性
requirement	userdef003	ノードの属性
parameterDefs	userdef009	ノードの属性
parameterVals	userdef007	ノードの属性
parent	userdef013	ノードの属性
refSource	userdef011	ノードの属性
flag	userdef015	ノードの属性
respName	userdef012 (「;」で連結)	ノードの属性
respAddress		
respIcon		
siblingOrder	userdef001	リンクの属性

変換のための XSLT コードを以下に示す。

属性に関しては、対応する属性に値をコピーするが、元の属性の値は削除しない。

```
<?xml version="1.0" encoding="UTF-8"?>

<!--
 * Copyright (C) 2013  The University of Electro-Communications All rights reserved.
 * Copyright (C) 2013  AXE,Inc.
-->

<!-- Transforms Old GMF to New GMF -->

<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
                xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                xmlns:dcase_old="http://www.dependable-os.net/2010/03/dcase/"
                xmlns:dcase="http://www.dependable-os.net/2013/11/dcase_model/"
                version="1.0">

  <xsl:output method="xml" encoding="UTF-8" indent="yes" />

  <!-- Root -->
  <xsl:template match="/">
    <dcase:Argument>
      <xsl:apply-templates select="dcase_old:Argument"/>
    </dcase:Argument>
  </xsl:template>

  <xsl:template match="dcase_old:Argument">
    <xsl:apply-templates select="@*" mode="basicnode"/>
    <xsl:apply-templates select="rootBasicNode"/>
    <xsl:apply-templates select="rootBasicLink"/>
  </xsl:template>

  <!-- Node -->
  <xsl:template match="rootBasicNode">
    <xsl:element name="rootBasicNode">
      <xsl:apply-templates select="@*" mode="node"/>
    </xsl:element>
  </xsl:template>

  <!-- Link -->
  <xsl:template match="rootBasicLink">
    <xsl:element name="rootBasicLink">
      <xsl:apply-templates select="@*" mode="basiclink"/>
    </xsl:element>
  </xsl:template>

  <!-- Attributes -->

  <xsl:template match="@*" mode="basicnode">
    <xsl:choose>
      <xsl:when test="name(.)='userdef005'">
        <xsl:attribute name="parameterizedDesc"><xsl:value-of
select="."/></xsl:attribute>
        <xsl:attribute name="{name(.)}"><xsl:value-of select="."/></xsl:attribute>
      </xsl:when>
      <xsl:when test="name(.)='userdef002'">
        <xsl:attribute name="message"><xsl:value-of select="."/></xsl:attribute>
        <xsl:attribute name="{name(.)}"><xsl:value-of select="."/></xsl:attribute>
      </xsl:when>
      <xsl:when test="name(.)='userdef003'">
        <xsl:attribute name="requirement"><xsl:value-of select="."/></xsl:attribute>
        <xsl:attribute name="{name(.)}"><xsl:value-of select="."/></xsl:attribute>
      </xsl:when>
      <xsl:when test="name(.)='userdef009'">
        <xsl:attribute name="parameterDefs"><xsl:value-of select="."/></xsl:attribute>
        <xsl:attribute name="{name(.)}"><xsl:value-of select="."/></xsl:attribute>
      </xsl:when>
      <xsl:when test="name(.)='userdef007'">
```

```
        <xsl:attribute name="parameterVals"><xsl:value-of select="."/></xsl:attribute>
        <xsl:attribute name="{name(.)}"><xsl:value-of select="."/></xsl:attribute>
      </xsl:when>
      <xsl:when test="name(.)='userdef013'">
        <xsl:attribute name="parent"><xsl:value-of select="."/></xsl:attribute>
        <xsl:attribute name="{name(.)}"><xsl:value-of select="."/></xsl:attribute>
      </xsl:when>
      <xsl:when test="name(.)='userdef011'">
        <xsl:attribute name="refSource"><xsl:value-of select="."/></xsl:attribute>
        <xsl:attribute name="{name(.)}"><xsl:value-of select="."/></xsl:attribute>
      </xsl:when>
      <xsl:when test="name(.)='userdef015'">
        <xsl:attribute name="flag"><xsl:value-of select="."/></xsl:attribute>
        <xsl:attribute name="{name(.)}"><xsl:value-of select="."/></xsl:attribute>
      </xsl:when>
      <xsl:when test="name(.)='userdef012'">
        <xsl:attribute name="respName"><xsl:value-of
select="substring-before(.,';')"/></xsl:attribute>
        <xsl:variable name="respRes" select="substring-after(.,';')"/>
        <xsl:attribute name="respAddress"><xsl:value-of
select="substring-before($respRes,';')"/></xsl:attribute>
        <xsl:attribute name="respIcon"><xsl:value-of
select="substring-after($respRes,';')"/></xsl:attribute>
        <xsl:attribute name="{name(.)}"><xsl:value-of select="."/></xsl:attribute>
      </xsl:when>
      <xsl:otherwise>
        <xsl:attribute name="{name(.)}"><xsl:value-of select="."/></xsl:attribute>
      </xsl:otherwise>
    </xsl:choose>
  </xsl:template>

  <xsl:template match="@*" mode="node">
    <xsl:if test="name(.)='xsi:type'">
      <xsl:attribute name="{name(.)}">
        <xsl:choose>
          <xsl:when test=".='dcase:Userdef004'">dcase:Assumption</xsl:when>
          <xsl:when test=".='dcase:Userdef005'">dcase:Module</xsl:when>
          <xsl:when test=".='dcase:Userdef006'">dcase:Contract</xsl:when>
          <xsl:when test=".='dcase:System'">dcase:Pattern</xsl:when>
          <xsl:when test=".='dcase:Policy'">dcase:Action</xsl:when>
          <xsl:when test=".='dcase:Userdef001'">dcase:External</xsl:when>
          <xsl:otherwise><xsl:value-of select="."/></xsl:otherwise>
        </xsl:choose>
      </xsl:attribute>
    </xsl:if>
    <xsl:if test="name(.)!='xsi:type'">
      <xsl:apply-templates select="." mode="basicnode"/>
    </xsl:if>
  </xsl:template>

  <xsl:template match="@*" mode="basiclink">
    <xsl:choose>
      <xsl:when test="name(.)='xsi:type'">
        <xsl:attribute name="{name(.)}">
          <xsl:choose>
            <xsl:when test=".='dcase:DcaseLink001'">dcase:SupportedBy</xsl:when>
            <xsl:when test=".='dcase:DcaseLink002'">dcase:InContextOf</xsl:when>
            <xsl:when test=".='dcase:DcaseLink003'">dcase:Responsibility</xsl:when>
            <xsl:otherwise><xsl:value-of select="."/></xsl:otherwise>
          </xsl:choose>
        </xsl:attribute>
      </xsl:when>
      <xsl:when test="name(.)='userdef001'">
        <xsl:attribute name="siblingOrder"><xsl:value-of select="."/></xsl:attribute>
        <xsl:attribute name="{name(.)}"><xsl:value-of select="."/></xsl:attribute>
      </xsl:when>
      <xsl:when test="name(.)='userdef002'">
        <xsl:attribute name="message"><xsl:value-of select="."/></xsl:attribute>
```

```
        <xsl:attribute name="{name(.)}"><xsl:value-of select="."/></xsl:attribute>
      </xsl:when>
      <xsl:otherwise>
        <xsl:attribute name="{name(.)}"><xsl:value-of select="."/></xsl:attribute>
      </xsl:otherwise>
    </xsl:choose>
  </xsl:template>

</xsl:stylesheet>
```

2.4.4. GMF モデル情報ファイルから ARM ファイルへの変換

従来の D-Case Editor には、「GMF から ARM への変換」機能があるため、これを、今回定義した XML 仕様に準拠させる。具体的には、以下の処理を追加する。

- Assumption、Pattern および Action ノードを InformationElement に変換
- Module および External ノードを Claim に変換
- Contract ノードを XXX に変換
- SupportedBy, InContextOf および DcaseLink004 は、source もしくは target が Action の場合、AssertedEvidence に変換

GMF モデル情報ファイルを ARM ファイルに変換するための XSLT コードを以下に示す。

```
<?xml version="1.0" encoding="UTF-8"?>

<!--/*******************************************************************************
     * Copyright (C) Yutaka Matsuno 2010-2012 All rights reserved.

*******************************************************************************/-->

<!-- The XLS file that converts the GMF model file to ARM model file. -->
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
                xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                xmlns:dcase_gmf="http://www.dependable-os.net/2013/11/dcase_model/"
                xmlns:ARM="ARM"
                xmlns:xmi="http://www.omg.org/XMI"
                extension-element-prefixes="dcase_gmf"
                version="1.0">

  <xsl:output method="xml" encoding="UTF-8" indent="yes"/>

  <!--root element-->
  <xsl:template match="/">
    <ARM:Argument>
      <xsl:attribute name="xmi:version">2.0</xsl:attribute>
      <xsl:apply-templates select="dcase_gmf:Argument" />
    </ARM:Argument>
  </xsl:template>

  <xsl:template match="xsi:book">
    <xsl:value-of select="xsi:title" />
  </xsl:template>

  <xsl:template match="dcase_gmf:Argument">
    <xsl:call-template name="NodeElementAttribute" />
    <xsl:apply-templates select="rootBasicNode" mode="basic" />
    <xsl:apply-templates select="rootBasicLink" mode="basic" />
  </xsl:template>

  <!-- outputs the node element -->
  <xsl:template match="rootBasicNode" mode="basic">
    <xsl:choose>
        <!-- outputs to the ARM file -->
        <xsl:when test="@xsi:type='dcase:Goal'">
           <xsl:element name="containsArgumentElement">
             <xsl:call-template name="NodeType" />
```

```
              <xsl:call-template name="NodeElementAttribute" />
              <xsl:call-template name="CheckUndevelopNode">
                <xsl:with-param name="sourceid" select="concat('#',@id)" />
              </xsl:call-template>
            </xsl:element>
        </xsl:when>
        <xsl:when test="@xsi:type='dcase:Strategy'">
            <xsl:element name="containsArgumentElement">
              <xsl:call-template name="NodeType" />
              <xsl:call-template name="NodeElementAttribute" />
              <xsl:call-template name="CheckGoalNode">
                <xsl:with-param name="sourceid" select="concat('#',@id)" />
              </xsl:call-template>
            </xsl:element>
        </xsl:when>
        <xsl:when test="(@xsi:type='dcase:Evidence') or (@xsi:type='dcase:Monitor') or
(@xsi:type='dcase:Justification') or (@xsi:type='dcase:Context') or
(@xsi:type='dcase:Assumption') or (@xsi:type='dcase:Module') or
(@xsi:type='dcase:Contract') or (@xsi:type='dcase:Pattern') or
(@xsi:type='dcase:Action') or (@xsi:type='dcase:External')">
            <xsl:element name="containsArgumentElement">
              <xsl:call-template name="NodeType" />
              <xsl:call-template name="NodeElementAttribute" />
            </xsl:element>
        </xsl:when>
    </xsl:choose>
  </xsl:template>

  <!-- outputs the link element -->
  <xsl:template match="rootBasicLink" mode="basic">
    <xsl:variable name="linkSource"><xsl:value-of
select="substring-after(@source,'#')" /></xsl:variable>
    <xsl:variable name="linkTarget"><xsl:value-of
select="substring-after(@target,'#')" /></xsl:variable>
    <!-- gets the name of node type -->
    <xsl:variable name="sourceNodeType">
        <xsl:value-of
select="/dcase_gmf:Argument/rootBasicNode[@id=$linkSource]/@xsi:type" />
    </xsl:variable>
    <xsl:variable name="targetNodeType">
        <xsl:value-of
select="/dcase_gmf:Argument/rootBasicNode[@id=$linkTarget]/@xsi:type" />
    </xsl:variable>
    <!-- checks the name of node type -->
    <xsl:if test="contains($sourceNodeType, 'dcase:Goal') or contains($sourceNodeType,
'dcase:Strategy') or contains($sourceNodeType, 'dcase:Evidence') or
contains($sourceNodeType, 'dcase:Monitor') or contains($sourceNodeType,
'dcase:Justification') or contains($sourceNodeType, 'dcase:Context') or
contains($sourceNodeType, 'dcase:Assumption') or contains($sourceNodeType,
'dcase:Module') or contains($sourceNodeType, 'dcase:Contract') or
contains($sourceNodeType, 'dcase:Pattern') or contains($sourceNodeType,
'dcase:Action') or contains($sourceNodeType, 'dcase:External')">
      <xsl:if test="contains($targetNodeType, 'dcase:Goal') or
contains($targetNodeType, 'dcase:Strategy') or contains($targetNodeType,
'dcase:Evidence') or contains($targetNodeType, 'dcase:Monitor') or
contains($targetNodeType, 'dcase:Justification') or contains($targetNodeType,
'dcase:Context') or contains($targetNodeType, 'dcase:Assumption') or
contains($targetNodeType, 'dcase:Module') or contains($targetNodeType,
'dcase:Contract') or contains($targetNodeType, 'dcase:Pattern') or
contains($targetNodeType, 'dcase:Action') or contains($targetNodeType,
'dcase:External')">
        <xsl:choose>
            <!-- outputs to the ARM file -->
            <xsl:when test="(@xsi:type='dcase:SupportedBy') or
(@xsi:type='dcase:Responsibility') or (@xsi:type='dcase:DcaseLink004')">
                <xsl:element name="containsArgumentLink">
                  <xsl:call-template name="LinkType" />
                  <xsl:call-template name="LinkElementAttribute" />
```

```
                </xsl:element>
              </xsl:when>
              <xsl:when test="@xsi:type='dcase:InContextOf'">
                <xsl:element name="containsArgumentLink">
                  <xsl:call-template name="LinkType" />
                  <xsl:call-template name="LinkElementAttribute" />
                </xsl:element>
              </xsl:when>
            </xsl:choose>
          </xsl:if>
        </xsl:if>
      </xsl:template>

      <!-- sets the "xsi:type" attribute in the node element -->
      <xsl:template name="NodeType">
        <xsl:attribute name="type" namespace="http://www.w3.org/2001/XMLSchema-instance">
          <!-- checks the name of link type -->
          <xsl:choose>
            <xsl:when test="(@xsi:type='dcase:Goal') or (@xsi:type='dcase:Module') or
(@xsi:type='dcase:External')">ARM:Claim</xsl:when>
            <xsl:when test="@xsi:type='dcase:Strategy'">ARM:ArgumentReasoning</xsl:when>
            <xsl:when test="(@xsi:type='dcase:Context') or
(@xsi:type='dcase:Justification') or (@xsi:type='dcase:Evidence') or
(@xsi:type='dcase:Monitor') or (@xsi:type='dcase:Assumption') or
(@xsi:type='dcase:Pattern') or
(@xsi:type='dcase:Action')">ARM:InformationElement</xsl:when>
          </xsl:choose>
        </xsl:attribute>
      </xsl:template>

      <!-- sets the "xsi:type" attribute in the link element -->
      <xsl:template name="LinkType">
        <xsl:attribute name="type" namespace="http://www.w3.org/2001/XMLSchema-instance">
          <xsl:choose>
            <xsl:when test="@xsi:type='dcase:InContextOf'">ARM:AssertedContext</xsl:when>
            <xsl:when test="(@xsi:type='dcase:SupportedBy') or
(@xsi:type='dcase:Responsibility') or (@xsi:type='dcase:DcaseLink004')">
              <xsl:variable name="linkSource"><xsl:value-of
select="substring-after(@source,'#')" /></xsl:variable>
              <xsl:variable name="linkTarget"><xsl:value-of
select="substring-after(@target,'#')" /></xsl:variable>
              <xsl:choose>
                <xsl:when test="/dcase_gmf:Argument/rootBasicNode[(@id=$linkSource) and
((@xsi:type='dcase:Evidence') or (@xsi:type='dcase:Monitor') or
(@xsi:type='dcase:Action'))]">ARM:AssertedEvidence</xsl:when>
                <xsl:when test="/dcase_gmf:Argument/rootBasicNode[(@id=$linkTarget) and
((@xsi:type='dcase:Evidence') or (@xsi:type='dcase:Monitor') or
(@xsi:type='dcase:Action'))]">ARM:AssertedEvidence</xsl:when>
                <xsl:otherwise>ARM:AssertedInference</xsl:otherwise>
              </xsl:choose>
            </xsl:when>
          </xsl:choose>
        </xsl:attribute>
      </xsl:template>

      <!-- sets the "toBeSupported" attribute when the node type is 'Goal' node -->
      <xsl:template name="CheckUndevelopNode">
        <xsl:param name="sourceid">undefined</xsl:param>
        <xsl:attribute name="toBeSupported">false</xsl:attribute>
        <xsl:for-each select="/dcase_gmf:Argument/rootBasicLink[@source=$sourceid]">
          <xsl:variable name="targetid" select="substring-after(@target,'#')" />
          <xsl:variable name="count"
select="count(/dcase_gmf:Argument/rootBasicNode[@id=$targetid and
@xsi:type='dcase:Undeveloped'])" />
          <xsl:if test="$count &gt; 0">
            <xsl:attribute name="toBeSupported">true</xsl:attribute>
          </xsl:if>
        </xsl:for-each>
```

```
  </xsl:template>

  <!-- sets the "describes" attribute when the node type is 'Strategy' node -->
  <xsl:template name="CheckGoalNode">
    <xsl:param name="sourceid">undefined</xsl:param>
    <xsl:variable name="describesAttributes">
      <xsl:for-each select="/dcase_gmf:Argument/rootBasicLink[@source=$sourceid]">
          <xsl:variable name="targetid" select="substring-after(@target,'#')" />
          <xsl:variable name="describesValue">
            <xsl:apply-templates
select="/dcase_gmf:Argument/rootBasicNode[@id=$targetid and @xsi:type='dcase:Goal']"
mode="describes" />
          </xsl:variable>
          <xsl:value-of select="$describesValue" />
        </xsl:for-each>
    </xsl:variable>
    <xsl:variable name="describesCount">
      <xsl:value-of select="string-length($describesAttributes)" />
    </xsl:variable>
    <xsl:attribute name="describes">
      <xsl:value-of select="substring($describesAttributes, 0, $describesCount)" />
    </xsl:attribute>
  </xsl:template>

  <!-- adds the space character to the "describes" attribute -->
  <xsl:template match="rootBasicNode" mode="describes">
    <xsl:value-of select="@id" /><xsl:text  disable-output-escaping="yes"> </xsl:text>
  </xsl:template>

  <!-- sets the attributes in the element of the node -->
  <xsl:template name="NodeElementAttribute">
    <xsl:attribute name="xmi:id">
      <xsl:value-of select="@id"/>
    </xsl:attribute>
    <xsl:attribute name="identifier">
      <xsl:value-of select="@name"/>
    </xsl:attribute>
    <xsl:attribute name="content">
      <xsl:value-of select="@desc"/>
    </xsl:attribute>
  </xsl:template>

  <!-- sets the attributes in the element of the link -->
  <xsl:template name="LinkElementAttribute">
    <xsl:attribute name="xmi:id">
      <xsl:value-of select="@id"/>
    </xsl:attribute>
    <xsl:attribute name="content">
      <xsl:value-of select="@desc"/>
    </xsl:attribute>
    <xsl:attribute name="source">
      <xsl:value-of select="substring-after(@source,'#')"/>
    </xsl:attribute>
    <xsl:attribute name="target">
      <xsl:value-of select="substring-after(@target,'#')"/>
    </xsl:attribute>
    <xsl:attribute name="identifier">
      <xsl:value-of select="@name"/>
    </xsl:attribute>
  </xsl:template>

</xsl:stylesheet>
```

2.4.5. GMF モデル情報ファイルから SACM ファイルへの変換

SACM の Argumentation と、ARM の XML スキーマを比較すると、以下の違いがある。

(「要素名」もしくは「要素名.属性名」で表記。要素名が「*」の場合は共通の属性。)

SACM	ARM
Argumentation	Argument
argumentElement	containsArgumentElement
	containsArgumentLink
argumentation	containsArgument
*.id (必須)	*.identifier (必須でない)
*.description (必須)	*.description (必須でない)
*.content (必須)	*.content (必須でない)
Claim.assumed (必須)	Claim.assumed (必須でない)
Claim.toBeSupported (必須)	Claim.toBeSupported (必須でない)
InformationElement.url (必須)	(該当する属性がない)
InformationElement.evidence	
CitationElement.argumentationReference	CitationElement.refersToArgument
CitationElement.argumentElementReference	CitationElement.refersToArgumentElement
ArgumentReasoning.structure	ArgumentReasoning.hasStructure
ArgumentReasoning.describedInference ArgumentReasoning.describedChallenge	ArgumentReasoning.describes

ここで、description および url 属性の値は、元になる情報がないため、空文字列(〝〞)にする。

また、assumed 属性の値は、Module もしくは External ノードの場合、必ずリーフノードになるため、true にする。Goal ノードの場合、エビデンスがないときには Uneveloped ノードを子ノードに指定するため、false にする。

ArgumentReasoning ノードの describes 属性名は、子ノードが Goal(Claim)になるため、describedInference に変更する。

さらに、2.3 で拡張した XML 仕様を使用して、GMF モデル情報ファイルの parameterDefs および parameterVals 属性の値は、そのまま同名の属性の値に使用する。

以上をふまえた、2.4.4 の XSLT コードをベースとする、GMF モデル情報ファイルを SACM ファイルに変換するための XSLT コードを使用する。XLST コードの基本的な構造は同等であるため、内容の記載は割愛する。