

DITA 特殊化を ADOBE® FRAMEMAKER® 9 に統合

© 2009 Adobe Systems Incorporated. All rights reserved.

Adobe, the Adobe logo, and FrameMaker are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States and/or other countries. All other trademarks are the property of their respective owners.

This Work is licensed under the Creative Commons Attribution Non-Commercial 3.0 License. To view a copy of this license, visit <http://creativecommons.org/licenses/by-nc-sa/3.0/>

目次

FrameMaker における DITA 特殊化

DITA 特殊化	1
FrameMaker における DITA エLEMENTの特別な取り扱ひ	3
FrameMaker における DITA エLEMENTの特殊化	4
特殊化の DITA DTD を作成する	6
DTD から特殊化 EDD を作成する	11
構造化テンプレートを作成する	11
読み取り / 書き込み規則ファイルを作成する	11
構造化アプリケーション定義を更新する	11
特殊化トピックの発行	12

FrameMaker における DITA 特殊化

DITA 特殊化

特殊化について

特殊化は、既存のデザインに基づいて新しいデザインを作成するプロセスです。特殊化は上位のデザインからのエレメントを再利用できます。DITA を特殊化して、既存の DITA アーキテクチャの利点を活かしつつ、ビジネス要件を満たすようなカスタマイズされた情報モデルを作成できます。

DITA アーキテクチャは、一般ベーストピックとその3つの特殊化バリエーション（概念、タスク、参照トピックタイプ）を提供します。各特殊化トピックタイプ DTD は、そのトピックタイプにのみ固有かつ制限を付けるエレメントを宣言します。たとえば、タスクトピック DTD では、<step> と <choice> エレメントが可能ですが、これらのエレメントは他の DTD にはありません。

新しい構造タイプまたは新しいドメインが必要なときは、タスク、概念、参照 DITA トピックタイプを特殊化できます。たとえば、一貫性や記述性を上げたり特定の出力ニーズを満たすために、デザインを微調整できます。

特殊化のタイプ

特殊化は、以下の2つのタイプに大別できます。

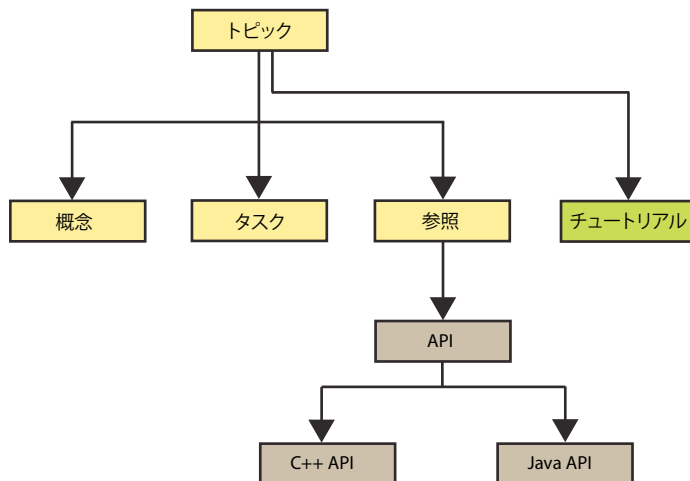
構造の特殊化 概念、タスク、または参照など、ベーストピックとマップから導出した新しいトピックまたはマップ構造を定義します。

ドメインの特殊化 プログラミングやハードウェアなど、特定の情報ドメインまたは主題領域のためにマークアップを定義します。

既存のコンポーネントを特殊化するには、特殊化の親に向かうパスまたは子に進展するパスを明確に特定し、特殊化ステートメントを DITA ファイルに追加します。このようにして、情報の再利用または交換において、少なくとも最小レベルの意味構造を保持できます。

構造の特殊化

構造の特殊化は、新しいトピックタイプや新しいマップタイプのような構造化情報の新しいタイプを定義します。構造の特殊化により、既存のスタイルシート、変換、プロセスとの互換性を保持したまま、新しいトピックタイプを作成できます。



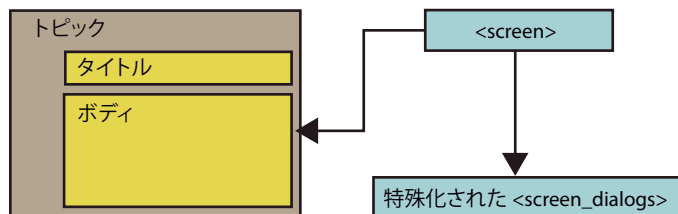
DITA ベーストピック、概念、タスク、参照、および特殊化トピック、チュートリアル、API を表示します。API トピックはさらに、Java API と C++ API に特殊化されます。

構造の特殊化は、新しい最上位のトピック タイプとマップ タイプを宣言します。構造の特殊化を使用して、まったく新しい文書構造を定義します。構造の特殊化では、トピックやマップなどのすべてのベース エレメントを特殊化できるほか、すべてのトピックまたはマップの特殊化内のエレメントも特殊化できます。

構造の特殊化においては、トピックやマップなど、階層の最上位から始めます。次に、そのトピック タイプの特殊化バージョンを作成し、必要に応じてエレメントを特殊化していきます。

ドメインの特殊化

ドメインの特殊化は、`<paragraph>` エレメントの新しいタイプなど、エレメントの新しいタイプを定義します。これらのエレメントの特殊化は、特定の情報ドメインまたは主題領域（たとえばプログラミングまたはハードウェアなど）に固有のものとすることができます。たとえば、ユーザー インターフェイス ドメインからの `<screen>` エレメントを特殊化して、すべてのダイアログ ボックスをカタログ化できます。



ダイアログ ボックスの文書に一貫した特殊化した構造を与えるために、`<screen>` エレメントのドメイン特殊化を表示します。

ドメインの特殊化を実装するときに、すべてのトピックまたはマップ タイプ内で使用するための新しいエレメントを定義します。ドメインの特殊化を使用すると、任意の他のドメイン モジュール内の任意のエレメントの `<topic>` または `<map>` の容認された子であるすべてのベース DITA エレメントを特殊化できます。

たとえば、マップ固有のエレメントである `<topicref>`、`<topichead>`、`<topicgroup>` は、これらをベース `<map>` タイプの一部と考える場合でも、ドメイン エレメントとして定義されます。したがって、`<topicref>` または `<topichead>` の特殊化を定義することはドメインの特殊化であり、マップの特殊化ではありません。

特殊化を使用する理由と時期

特殊化は、新しい構造タイプまたは新しい情報ドメインを定義するために使用します。特殊化により、デザインを改造して一貫性と記述性を向上できます。特殊化はまた、現在のデータ モデルでは扱えない特定の出力要求のためにも使用できます。

すべての特殊化は既存の変換によって処理でき、またさらに一般的なものへと逆変換できます。

特殊化には次のような利点があります。

- ベースの語彙を再利用して特殊化を定義し、それを作成するための時間を節約できます。
- ベース エlement への変更は、その特殊化へも自動的に反映されます。
- 要件に応じてモジュールを簡単にプラグインできます。
- 特殊化をベース タイプに簡単に戻せます。
- 相互運用性または特殊化タイプ文書からベース タイプ文書へのマッピングが保証されます。

特殊化の規則

特殊化エlement タイプのコンテンツ モデルを、ベース タイプのコンテンツ モデルよりも制限を低くできます。

別のエlement からあるエlement を特殊化するとき、この新しいエlement はある規則に従わないと有効になりません。

- 特殊化エlement は、ベース エlement より制限が高いまたは等しいコンテンツ モデルを持つ必要があります。
- 特殊化エlement は、ベース エlement と等しい属性またはそのサブセットを持つ必要があります。
- 特殊化エlement の属性は、ベース エlement と同じ値または値範囲、あるいは値または値範囲のサブセットを持つ必要があります。
- 特殊化エlement は、正しく作成されたクラス属性を持つ必要があります。

FrameMaker における DITA エlement の特別な取り扱い

<table>、<image>、<indexterm>、<footnote> などの特別なエlement の場合は、EDD はタイプを表す特別な情報を含みます。このような特別なエlement の特殊化エlement はすべて、EDD に対応情報を持つことも必要です。こうしないと、特殊化エlement は正しく解釈されません。

Adobe® FrameMaker® 9 で特別な取り扱いを持つエlement のリストを以下に示します。これらのエlement の特殊化したものを作成した場合は、その EDD に対応情報も含めてください。

topicref 特殊化した <topicref> の一般規則に、有効な最初のエlement として <fm-topicreflabel> を含めてください。参照を更新したり、すべての <topicref>、<conref topic ref>、<navtitle> を開くといった機能は、特殊化した <topicref> エlement に対して使用できるようになります。

indexterm 索引用語の読み込みおよび書き出し処理が DITA オプション ダイアログ ボックスで有効になっている場合は、<indexterm> と <Index-see>、<index-see-also>、<index-sort-as> のネスティングも、特殊化された索引用語エlement のために使用できます。

table/simpletable/reltable/choicetable DITA テーブル エlement は <numCols> および <colWidth> プロパティを含みませんが、これらのプロパティは、<reltable> および <simpletable> のためとそれらから特殊化したエlement のための ditafm.ini のなかで明示的に設定する必要があります。<reltable> と <simpletable> を特殊化するとき、EDD ファイル内のベース エlement に類似した構造で、<fm-reltablemeta> および <fm-chtheadrow> にパラレルなエlement を追加します。次に、これらの特殊化エlement のために、読み取り / 書き込み規則ファイルで同様の宣言を行います。特殊化した <table>、<simpletable>、または <reltable> エlement を挿入すると、これらの特殊化エlement の名前が挿入テーブル ダイアログ ボックスに表示されます。

topic/map 正しい特殊化 <topic> または <map> アプリケーションが DITA オプションに設定された場合は、特殊化 <topic> または <map> の名前が DITA / 新規 DITA ファイル サブメニューに表示されます。FrameMaker 文書機能を持つコンボジット FrameMaker 文書またはブック ファイルが特殊化トピックのためにも使用できます。

image/alt FrameMaker グラフィック オブジェクトとして機能するために特殊化イメージ エLEMENTの読み取り / 書き込み規則ファイルに、特殊化イメージ エLEMENTを確実に宣言してください。

注意：FrameMaker 9 は <alt> エLEMENTの特殊化をサポートしていません。

xref/link EDD ファイル内で <xref> エLEMENTを特殊化するときは、この特殊化 <xref> エLEMENTが使用できる場所では <fm-xref> が確実に使用できるようにしてください。DITA 文書内で特殊化 <xref> または <link> エLEMENTを挿入すると、DITA 相互参照ダイアログが開きます。特殊化 <xref> および <link> エLEMENTの名前がこのダイアログ ボックスで使用できます。

linktext 特殊化 <linktext> エLEMENTのすべてのオカレンスにおいて、有効な選択として、EDD 内に確実に <fm-linktext> エLEMENTを追加してください。

prolog/draft-comment DITA オプション ダイアログ ボックスには、ファイルが開いたときに <prolog> または <draft-comment> エLEMENTを条件設定するためのオプションがあります。これらのオプションが選択されると、<prolog> と <draft-comment> エLEMENTの特殊化も条件設定されます。

fn 読み取り / 書き込み規則ファイルで脚注として特殊化 <fn> エLEMENTを宣言することも必要です。

注意：これらすべてのエLEMENTでは、ベース エLEMENTに対して空のクラス属性が許されます。

FrameMaker における DITA エLEMENTの特殊化

特殊化における主な仕事は、必要となる特殊化エLEMENTのリストを計画することと、それらが既存の DITA 階層のなかのどこに納まるかを定義することです。過度に特殊化したエLEMENTや不完全な特殊化をしないように注意してください。既存のエLEMENTで十分なときは、特殊化エLEMENTを作成しないでください。

トピックを特殊化するときは、次のワークフローに従ってください。

- 1 既存のエLEMENT タイプから導出した新しいセットのエLEMENTを定義して新しいセットの DTD を作成します。
- 2 複数の特殊化 DTD を 1 つのベース DTD にまとめます。
- 3 既存の標準トピック / マップ読み取り / 書き込み規則ファイルを使用して、新しい読み取り / 書き込み規則ファイルを作成します。DITA エLEMENTから導出した特殊化エLEMENTのためのエLEMENT マッピングを追加します。これらの DITA エLEMENTはすでにいくつかのマッピングが定義されています。

しばしば特殊化エLEMENTは、読み取り / 書き込み規則ファイル内に宣言ステートメント持つベース エLEMENTから導出されます。この場合は、同じ宣言を、特殊化エLEMENTの読み取り / 書き込み規則ファイルに含めます。

たとえば、DITA <image> エLEMENTを FrameMaker <graphic> エLEMENTにマッピングする場合を考えてみます。この場合、DITA <image> エLEMENTとその特殊化は、それらの読み取り / 書き込み規則ファイルに同じ宣言を含める必要があります。

同様に、ベース エLEMENTに対してラップされていないステートメントがある場合は、そこから導出されたエLEMENTはこのラップされていないステートメントを含むことも必要になります。

- 4 EDD 内にすべての特殊化エLEMENTのクラス属性を設定します。

ベース エLEMENTから特殊化エLEMENTへの正しいエLEMENT階層を作成する必要があります。

- 5 手順 1 ~ 3 で作成した読み取り / 書き込み規則ファイルと DTD を使用して、ベース DTD を FrameMaker 内に EDD として読み込みます。

- 6 EDD ファイル内で FrameMaker 固有の変更を行います。とくに、次のトピック特殊化を行います。
- a FM で始まるすべてのエレメントを、標準トピック EDD から新しい EDD にコピーします。これらは、<table>、<crossref>、<image> などの特別なオブジェクトを扱うよう宣言された FrameMaker 固有のエレメントです。
 - b エレメント プロパティのコンテンツ モデルを、「fm-propheading?, fm-propertybody+」に変更します。
 - c <choicetable> のコンテンツ モデルを「chhead, fm-chbody」に、<simplatable> のそれを「sthead, fm-stbody」に変更します。
 - d <sthead> のコンテンツ モデルを「fm-stheadrow」に、<chhead> のそれを「fm-chheadrow」に変更します。
 - e <xref>、<syntaxdiagram>、<synblk>、<groupseq>、<groupchoice>、<groupcomp>、<fragment>、<fig> のコンテンツ モデル、および <xref> が有効になっている他のすべてのエレメントに、<fm-xref> エレメントを追加します。
 - f コンディショナル タグを使用して、<fm-graphic>、<alt>、<index-base>、<index-see>、<index-see-also>、および <index-see-also> エレメントを非表示にします。
- 7 EDD でこれらのマップ特殊化を次のようにします。
- a FM で始まるすべてのエレメントを、標準トピック EDD から新しい EDD にコピーします。これらは、<table>、<crossref>、<image> などの特別なオブジェクトを扱うよう宣言された FrameMaker 固有のエレメントです。
 - b エレメント プロパティのコンテンツ モデルを、「fm-propheading?, fm-propertybody+」に変更します。
 - c <choicetable> のコンテンツ モデルを「chhead, fm-chbody」に、<simplatable> のそれを「sthead, fm-stbody」に変更します。
 - d <sthead> のコンテンツ モデルを「fm-stheadrow」に、<chhead> のそれを「fm-chheadrow」に変更します。
 - e <xref>、<syntaxdiagram>、<synblk>、<groupseq>、<groupchoice>、<groupcomp>、<fragment>、<fig> のコンテンツ モデル、および <xref> が有効になっている他のすべてのエレメントに、<fm-xref> エレメントを追加します。
 - f コンディショナル タグを使用して、<fm-graphic>、<alt>、<index-base>、<index-see>、<index-see-also>、および <index-see-also> エレメントを非表示にします。
 - g reltable のコンテンツ モデルを、<fm-reltablemeta>?、<relheader>?、<fm-reltablebody> に変更します。
 - h relheader のコンテンツ モデルを (fm-relheaderrow) に変更します。
 - i 次のエレメントを Map EDD に追加します。
 -エレメント (表行) : fm-relheaderrow
 -一般ルール : relcolspec+
 -エレメント (表本文) : fm-reltablebody
 -一般ルール : relrow+
 -
 - j <topicref> のコンテンツ モデルに fm-topicreflabel を追加します。
- 8 標準 DITA エレメントのエレメント フォーマットを、デフォルトの DITA トピックまたはマップ EDD から新しい EDD にコピーします。EDD 内で、新しい特殊化エレメントのためのフォーマット規則を定義します。
- 9 EDD を新しいテンプレート ファイルに読み込みます。次に、段落書式と文字書式を、標準 DITA テンプレートからこの新しいテンプレートに読み込みます。上記の手順で作成したテンプレート、読み取り / 書き込み規則、統合 DTD ファイルを使って、特殊化エレメントのための新しい構造化アプリケーションを作成します。

10 DITA オプション ダイアログ ボックスの「DITA トピック アプリケーション」または「DITA マップ アプリケーション」リスト ボックスから、新しいアプリケーションの名前を選択します。「保存」をクリックします。

これで、特殊化トピック/マップは、DITA /新規 DITA ファイル サブメニューに表示されます。オーサリング作業を開始します。

特殊化の DITA DTD を作成する

DITA DTD はベース エLEMENT階層 (トピックとマップ)、それぞれのドメインと階層特殊化 (たとえば、タスク、概念、ブックマップ、UI ドメイン、プログラミングドメインなど) を反映する小さなモジュールに分割されます。DTD で行う固定セットの変更を以下に説明します。

注意: FrameMaker 9 がマシンにインストールされている場合は、次のところから DITA DTD または .mod ファイルにアクセスできます: <インストールディレクトリ>/Adobe/FrameMaker9/Structure/xml/DITA/app/dtd。また、この手順で使用したすべてのサンプル ファイルを次のところからダウンロードできます:

https://share.adobe.com/DITA_Specialization。

構造の特殊化の DTD を変更する

構造特殊化の実装は、次の 2 つの手順で行います。

- 1 特殊化エレメントの定義を含む .mod ファイルを作成します。
- 2 この .mod ファイルを、database.dtd 内の既存の DITA DTD と統合します。

特殊化エレメント タイプが既存の <topic> 階層で機能するようにするために、特殊化を database.dtd に追加します。あるいは、別個に DTD を作成することもできます。

<map> 特殊化の場合は、map.dtd または bookmap.dtd を変更し使用します。

以下の例では、コンテンツ モデルとして、特殊化した <xref> と <footnote> エレメントのみを使って、新しい特殊化オブジェクト エレメントを定義しています。

<topic> の構造特殊化の .mod ファイルを作成する

- 1 既存の .mod ファイルをコピーし、その名前を変更します。たとえば、reference.mod をコピーし、objectsp.mod として保存します。
- 2 この新しい .mod ファイル objectsp.mod を開きます。Specialization Of Declared Elements のセクションで、info-type 宣言を新しい特殊化構造タイプに変更します。この特殊化構造タイプは、特殊化モジュールを既存のモジュールと統合するために必要です。たとえば、次の行を

```
<!-- ===== -->
<!-- SPECIALIZATION OF DECLARED ELEMENTS -->
<!-- ===== -->
```

```
<!ENTITY % reference-info-types "%info-types;" >
```

この行で置き換えます：

```
<!-- ===== -->
<!--          SPECIALIZATION OF DECLARED ELEMENTS          -->
<!-- ===== -->

<!ENTITY % objectsp-info-types "%info-types;"                >
```

注意：同様に次の3つの手順で、指定されたセクションにある既存の宣言を削除し、それらを新しいエレメントの情報で置き換えます。このようにして、新しいエレメントを宣言するときに、既存の DTD およびマップ パラレル情報のフォーマット化構造を維持できます。

- 3 必要な特殊化エレメントの新しいエントリを、階層の最上部まで宣言します。

```
<!-- ===== -->
<!--          ELEMENT NAME ENTITIES          -->
<!-- ===== -->
<!ENTITY % myobjecttype "myobjecttype" >
<!ENTITY % mybody "mybody" >
<!ENTITY % myp "myp" >
<!ENTITY % myobject "myobject" >
<!ENTITY % myxref "myxref" >
<!ENTITY % myfootnote "myfootnote" >
<!-- ===== -->
```

- 4 他のエレメントについても、新しい特殊化エレメントなどを宣言します。

```
<!-- ===== -->
<!--          ELEMENT DECLARATIONS          -->
<!-- ===== -->
<!--          LONG NAME: myobject          -->
<!ELEMENT myobject      (({%myxref;}*, {%myfootnote;}*) >
<!ATTLIST myobject
    declare      (declare)          #IMPLIED
    classid      CDATA              #IMPLIED
    codebase     CDATA              #IMPLIED
    data         CDATA              #IMPLIED
    type         CDATA              #IMPLIED
    codetype     CDATA              #IMPLIED
    archive     CDATA              #IMPLIED
    standby     CDATA              #IMPLIED
    height      NMTOKEN             #IMPLIED
    width       NMTOKEN             #IMPLIED
    usemap      CDATA              #IMPLIED
    name        CDATA              #IMPLIED
    tabindex    NMTOKEN             #IMPLIED
    longdescref CDATA              #IMPLIED
    %univ-atts;
    outputclass CDATA              #IMPLIED
    longdescre  CDATA              #IMPLIED >
```

- 5 Specialization Attribute Declarations セクションで、特殊化エレメントを導出するエレメントを宣言します。階層をベース <topic> または <map> タイプ（構造特殊化のための「-」で始まるもの）まで宣言する必要があります。たとえば、特殊化エレメントが参照エレメントから導出されたもの場合は、階層全体を含めます。

```
- topic/reference/refbody specialtopic/specialbody
```

次の行を Specialization Attribute Declarations セクションに追加します。

```
<!-- ===== -->
<!-- SPECIALIZATION ATTRIBUTE DECLARATIONS -->
<!-- ===== -->
<!ATTLIST myobjecttype %global-atts; class CDATA "- topic/topic myobjecttype/myobjecttype" >
<!ATTLIST mybody %global-atts; class CDATA "- topic/body myobjecttype/mybody" >
<!ATTLIST myp %global-atts; class CDATA "- topic/p myobjecttype/myyp" >
<!ATTLIST myxref %global-atts; class CDATA "- topic/xref myobjecttype/myxref" >
<!ATTLIST myobject %global-atts; class CDATA "- topic/object myobjecttype/myobject" >
<!ATTLIST myfootnote %global-atts; class CDATA "- topic/fn myobjecttype/myfootnote" >
```

database.dtd を更新する

database.dtd を変更することで新しい .mod ファイルを既存のものと同様に統合します。

注意：オリジナルの database.dtd を上書きしないようにするために、この例では、それを databaseObjectsp.dtd という名前に変更するとよいでしょう。

- 1 Topic Nesting Override セクションで、特殊化トピック タイプのための宣言を追加します。必ず宣言にはベース タイプ情報を含めてください。

```
<!-- ===== -->
<!-- TOPIC NESTING OVERRIDE -->
<!-- ===== -->
<!ENTITY % info-types "topic | concept | task | reference |
glossentry | myobjecttype" >
```

- 2 新しい .mod ファイルを読み込むには、database.dtd の Topic Element Integration にエントリを追加します。

```
<!-- ===== -->
<!-- TOPIC ELEMENT INTEGRATION -->
<!-- ===== -->
<!-- Embed topic to get generic elements -->
<!ENTITY % topic-type PUBLIC "-//OASIS//ELEMENTS DITA Topic//EN"
"topic.mod" >
%topic-type;

<!ENTITY % objectsp-type PUBLIC "-//OASIS//ELEMENTS DITA Topic//EN"
"objectsp.mod" >
%objectsp-type;
```

注意：1つのトピック タイプ内の複数のトピック タイプを制限するには、統合ファイルを作成しますが、この例のようにトピック タイプをいっしょに統合しないでください。

ドメイン特殊化のために DTD を変更する

DITA ドメインは次の2つのファイルで実装されます。

- ドメインの要素を宣言する .mod ファイル。
- ドメインのエンティティを宣言する .ent ファイル。

ドメインの特殊化では、両方のファイルを作成します。.mod ファイルで、特殊化要素を宣言します。.ent ファイルで、統合関係の情報のためのエンティティを宣言します。ベース要素がどこにある場合でもドメイン特殊化要素が使用できなければならないので、.ent ファイルが必要です。

これらのファイルを作成したら、<topic> のドメイン特殊化を実装するために database.dtd を更新します。以下のセクションの手順では、<topic> の <image>、<prolog>、および <link> のための3つの新しいドメイン特殊化要素を定義します。

<map> のドメイン特殊化を実装するために、同じ手順に従いますが、次のファイルを編集します。

- ドメインの要素を宣言するための MapGroup.mod。
- ドメインのエンティティを宣言するための MapGroup.ent。
- .mod および .ent ファイルを統合するための Map.dtd または BookMap.dtd。

.mod ファイルを作成する

- 1 任意の既存の .mod ファイルをコピーし、名前を変更します。たとえば、utilitiesDomain.mod をコピーし、それを domainsp.mod として保存します。

注意：次の3つの手順で、指定されたセクションにある既存の宣言を削除し、それらを新しい要素の情報で置き換えます。このようにして、新しい要素を宣言するときに、既存の DTD およびマップパラレル情報のフォーマット構造を維持できます。

- 2 この新しい mod ファイル domainsp.mod を開きます。Element Name Entities セクションで、特殊化要素のための新しいエンティティを宣言します。

```
<!-- ===== -->
<!--          ELEMENT NAME ENTITIES          -->
<!-- ===== -->
<!ENTITY % Dlink      "Dlink"                >
<!ENTITY % Dprolog    "Dprolog"              >
<!ENTITY % Dimage     "Dimage"               >
<!-- ===== -->
```

- 3 新しい特殊化要素を宣言します。特殊化要素 <Dimage> のための次の行をコピーします。

```
<!-- ===== -->
<!--          ELEMENT DECLARATIONS          -->
<!-- ===== -->
<!--          LONG NAME: Dimage              -->
<!--          ===== -->
<!ELEMENT Dimage     (*alt;)                 >
<!ATTLIST Dimage
  href          CDATA          #REQUIRED
  keyref        NMTOKEN        #IMPLIED
  alt           CDATA          #IMPLIED
  longdescref   CDATA          #IMPLIED
  height        NMTOKEN        #IMPLIED
  width         NMTOKEN        #IMPLIED
  align         CDATA          #IMPLIED
  scale         NMTOKEN        #IMPLIED
  placement     (inline | break | -dita-use-conref-target) "inline"
  %univ-atts;
  outputclass   CDATA          #IMPLIED >
<!-- ===== -->
```

- 4 Specialization Attribute Declarations セクションで、特殊化要素を導出する要素を宣言します。階層をベース <topic> または <map> タイプ（ドメイン特殊化のための「+」で始まるもの）まで宣言する必要があります。たとえば、特殊化要素が別のユーティリティドメイン要素から導出される場合は、特殊化要素からユーティリティドメイン、トピックまでの階層全体を定義します。（ユーティリティドメインは <topic> から特殊化されます。）

```
<!-- ===== -->
<!--          SPECIALIZATION ATTRIBUTE DECLARATIONS          -->
<!-- ===== -->
<!ATTLIST Dprolog %global-atts; class CDATA "+ topic/prolog domainsp-d/Dprolog " >
<!ATTLIST Dlink %global-atts; class CDATA "+ topic/link domainsp-d/Dlink " >
<!ATTLIST Dimage %global-atts; class CDATA "+ topic/image domainsp-d/Dimage " >
```

.ent ファイルを作成する

1 domainsp.ent というファイル名で .ent ファイルを作成します。

このファイルの情報により、エレメントは、集合される代わりに置換できます。すなわち、ベースエレメントが許されるところでは、その特殊化エレメントも許されます。

注意: .mod ファイルに関しては、既存の .ent ファイルの名前を変更し、必要に応じて宣言ステートメントを置き換えることができます。

2 .ent ファイルを開き、(ドメイン拡張子を持つ) 既存のものと新しいエレメントを統合するためのエンティティを宣言します。

```

<!-- ===== -->
<!--          ELEMENT EXTENSION ENTITY DECLARATIONS          -->
<!-- ===== -->
<!ENTITY % domainsp-d-image "Dimage" >
<!ENTITY % domainsp-d-link "Dlink" >
<!ENTITY % domainsp-d-prolog "Dprolog" >

```

3 エレメントが導出されるルートまで祖先を定義するために、ドメイン属性エンティティを宣言します。あるドメイン拡張子からエレメントを特殊化する場合は、最上部まで宣言する必要があります。

```

<!-- ===== -->
<!--          DOMAIN ENTITY DECLARATION          -->
<!-- ===== -->
<!ENTITY domainsp-d-att "(topic ank-d)">

```

ditabase.dtd を更新する

ditabase.dtd を変更することで、特殊化 .mod ファイルを既存のファイルと統合します。ドメインの特殊化の場合は、次のように ditabase.dtd 内で .mod と .ent ファイルの両方を指定します。

1 Domain Entity Declarations セクションの vocabulary セクションで、新しいドメインを定義します。

```

<!-- ===== -->
<!--          DOMAIN ENTITY DECLARATION          -->
<!-- ===== -->
<!ENTITY % domainsp-d-dec PUBLIC "-//domainsp//ENTITIES DITA domainsp Domain//EN" "domainsp.ent" >
%domainsp-d-dec;
<!------- and the other existing ones ----->

```

2 特殊化エレメントの語彙置換を定義します。ドメイン特殊化エレメントが拡張するエレメントを含めます。

```

<!-- ===== -->
<!--          DOMAIN EXTENSIONS          -->
<!-- ===== -->
<!ENTITY % image "image | %domainsp-d-image;" >
<!ENTITY % prolog "prolog | %domainsp-d-prolog;" >
<!ENTITY % link "link | %domainsp-d-link;" >
<!------- and the other existing ones ----->

```

3 語彙属性宣言ステートメントを追加します。

```

<!-- ===== -->
<!--          DOMAINS ATTRIBUTE OVERRIDE          -->
<!-- ===== -->
<!ENTITY included-domains "%ui-d-att; %hi-d-att; %pr-d-att; %sw-d-att; %ut-d-att; %indexing-d-att; %domainsp-d-att;" >

```

- 4 語彙定義を指定し、ドメイン エレメント統合のための .mod ファイルを含めます。このエントリには、.mod ファイルで宣言されたすべての特殊化エレメントが含まれています。

```
<!--===== -->
<!--          DOMAIN ELEMENT INTEGRATION          -->
<!--===== -->
<!ENTITY % domainsp-doctype PUBLIC "-//domainsp//ELEMENTS DITA User Interface Domain//EN" "domainsp.mod" >
%domainsp-doctype;
```

DTD から特殊化 EDD を作成する

これで、特殊化 DTD から特殊化 EDD を作成します。これは、topic.edd.fm をコピーし、その名前を変更することで行えます。たとえば、新しい名前として ObjectSp.edd.fm を付けることができます。

注意：トピック EDD ファイルは <installation_directory>/Adobe/FrameMaker9/Structure/xml/DITA/app/DITA-Topic-FM から見つけられます。これを新しいフォルダ <インストール ディレクトリ >/Adobe/FrameMaker9/Structure/xml/DITA/app/ObjectSp にコピーします。

- 1 topic.edd.fm をコピーし、ObjectSp.edd.fm という名前で保存します。ObjectSp.edd.fm ファイルを開きます。
- 2 構造ツール / DTD の読み込みを選択し、作成した特殊化 DTD を選択します。
- 3 特殊化 EDD ファイルを保存します。

構造化テンプレートを作成する

- 1 dita-topic.template.fm ファイルをコピーし、それに objectsp.template.fm という名前を付けます。
- 2 ファイル / 読み込み / エレメント定義を選択し、特殊化 EDD からエレメント定義をテンプレートに読み込みます。
- 3 テンプレート ファイルを保存し閉じます。

読み取り / 書き込み規則ファイルを作成する

通常、特殊化エレメントが変更された読み込みまたは書き出し動作を持っている場合のみ、読み取り / 書き込み規則ファイルを更新します。

- ❖ topic.rules.txt ファイルをコピーし、ObjectSptopic.rules.txt という名前で保存します。

構造化アプリケーション定義を更新する

新しい特殊化 DTD を使用するには、構造化アプリケーション定義を指定する必要があります。

- 1 構造ツール / アプリケーション定義を編集を選択します。

注意：structapps.fom ファイルは <インストール ディレクトリ >/Adobe/FrameMaker9/Structure フォルダにあります。この例では、structapps を structappsObjectSp.fm という名前に変更できます。

- 2 構造ビューで、<XMLApplication> エレメントを挿入し、それに ObjectSp という名前を付けます。次のエレメントを <XMLApplication> エレメントに追加します。

- 3 <Template> エlementを追加し、特殊化テンプレート objectsp.template.fm へのパスを指定します。
- 4 <DTD> エlementを追加し、databaseObjectsp.dtd DTD へのパスを指定します。
- 5 <ReadWriteRules> エlementを追加し、規則ファイル ObjectSptopic.rules.txt へのパスを指定します。

```

Application name:      ObjectSp
Template:              SSTRUCTDIR\xml\dita\app\DITA-Topic-FM\objectsp.template.fm
DTD:                  SSTRUCTDIR\xml\dita\app\dt\d\databaseObjectsp.dtd
Read/write rules:    SSTRUCTDIR\xml\dita\app\DITA-Topic-FM\ObjectSp-
                    topic.rules.txt
DOCTYPE:              topic
                    task
                    concept
                    reference
                    glossentry
                    dita

myobjecttype
Conditional Text:
  Output Text PI:OutputAllTextWithPIs
Entity locations
Public ID: -//OASIS//DTD DITA Topic//EN
  Filena me: SSTRUCTDIR\xml\dita\app\dt\d\topic.dtd
Public ID: -//OASIS//DTD DITA Task//EN
  Filena me: SSTRUCTDIR\xml\dita\app\dt\d\task.dtd
Public ID: -//OASIS//DTD DITA Concept//EN
  Filena me: SSTRUCTDIR\xml\dita\app\dt\d\concept.dtd
Public ID: -//OASIS//DTD DITA Reference//EN
  Filena me: SSTRUCTDIR\xml\dita\app\dt\d\reference.dtd
Public ID: -//OASIS//DTD DITA Glossary//EN
  Filena me: SSTRUCTDIR\xml\dita\app\dt\d\glossary.dtd
Public ID: -//OASIS//DTD DITA Composite//EN
  Filena me: SSTRUCTDIR\xml\dita\app\dt\d\databaseObjectsp.dtd
Public ID: -//IBM//DTD DITA Topic//EN
  Filena me: SSTRUCTDIR\xml\dita\app\dt\d\topic.dtd
Public ID: -//IBM//DTD DITA Task//EN
  Filena me: SSTRUCTDIR\xml\dita\app\dt\d\task.dtd
Public ID: -//IBM//DTD DITA Concept//EN
  Filena me: SSTRUCTDIR\xml\dita\app\dt\d\concept.dtd
Public ID: -//IBM//DTD DITA Reference//EN
  Filena me: SSTRUCTDIR\xml\dita\app\dt\d\reference.dtd
Public ID: -//IBM//DTD DITA Glossary//EN
  Filena me: SSTRUCTDIR\xml\dita\app\dt\d\glossary.dtd
Public ID: -//IBM//DTD DITA Composite//EN
  Filena me: SSTRUCTDIR\xml\dita\app\dt\d\databaseObjectsp.dtd
Use API client:      ditafm_app
                    
```

ObjectSp の構造化アプリケーション指定

- 6 ファイルを保存して閉じます。
- 7 FrameMaker を閉じて再起動します。
- 8 DITA / DITA オプションを選択し、DITA トピック アプリケーション リストから ObjectSp を選択します。この新しいトピック タイプは、DITA / 新規 DITA ファイル サブメニューに表示されます。
- 9 特殊化ファイルからトピックを作成してオーサリングを開始します。

特殊化トピックの発行

DITA 特殊化では、非特殊化の一般的なツールを使用して特殊化コンテンツを処理できるという利点があります。ただし、これらのツールは、特殊化が導出された一般コンテンツ モデルに従って Element を処理します。たとえば、<paragraph> の特殊化フォームも段落としてフォーマットされます。

ベース フォーマットを微調整または多少変更するには、FrameMaker 環境内で EDD、XSLT スタイルシート、テンプレート、読み取り / 書き込み規則を変更できます。これらの変更を行った後に、新しいフォーマット定義を持つ Adobe PDF を FrameMaker から発行できます。