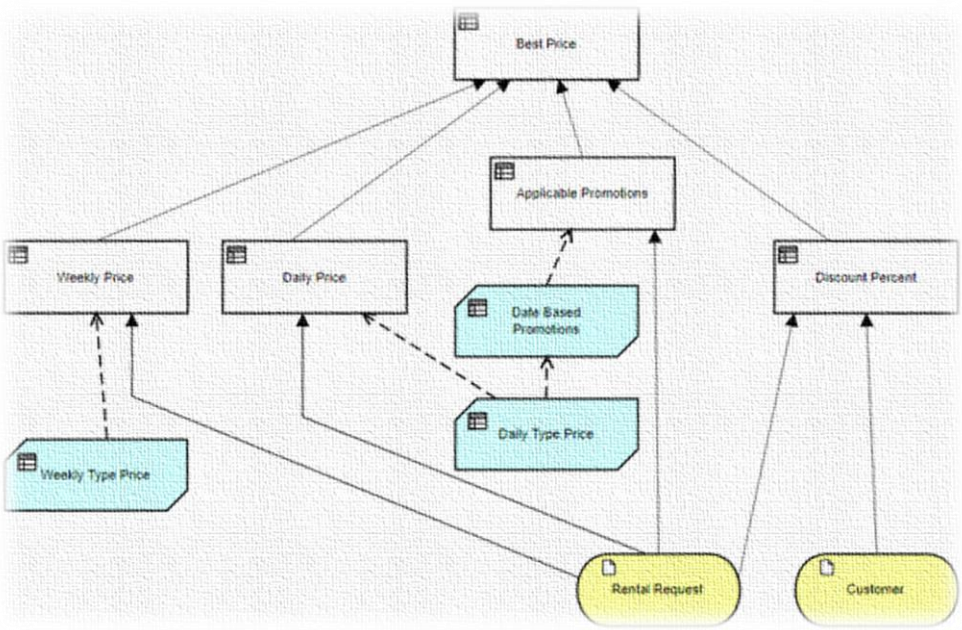
طراحی مدل تصمیم‌گیری یا Decision Diagram به واسطه استاندارد DMN



تهیه و تنظیم: پیمان مالکی

فهرست مطالب

1- نمودارمدل‌تصمیم‌گیری با نمادگذاری

هدف اصلی مدل‌سازی‌تصمیم‌گیری با نمادگذاری (DMN) ارائه مجموعه نمادهای مشترک است که برای همه کاربران کسب‌وکار قابل درک باشد، از تحلیلگران کسب‌وکار که نیاز به معرفی الزامات تصمیم‌گیری اولیه و سپس مدل‌های تصمیم‌گیری دقیق‌تر دارند، تا توسعه‌دهندگان فنی مسئول خودکارسازی تصمیم‌گیری‌ها در فرآیندها و در نهایت، به پرسنل کسب‌وکاری که آن تصمیمات را مدیریت و نظارت خواهند کرد. DMN یک پل استاندارد برای پرکردن شکاف بین طراحی تصمیم‌گیری‌های کسب‌وکاری و پیاده‌سازی آنها ایجاد می‌کند. نمادهای DMN طوری طراحی شده‌اند که در کنار نمادهای BPMN قابل استفاده باشند. هدف دیگر این است که انتقال مدل‌های تصمیم‌گیری در بین سازمان‌ها از طریق یک نمای XML امکان‌پذیر و قابل اطمینان باشد.

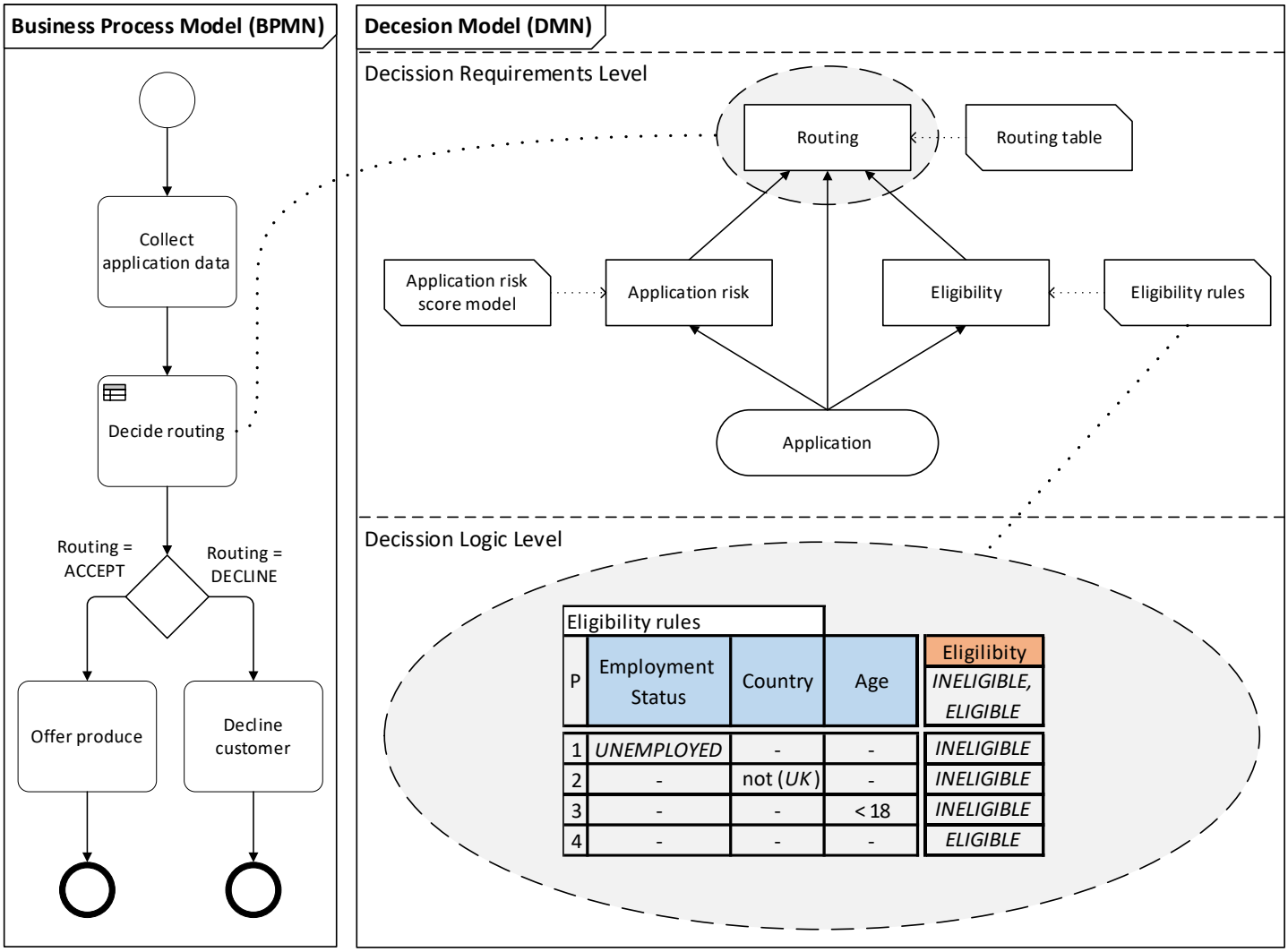
از طرف دیگر، هدف ارائه استاندارد DMN ارائه سازه‌هایی است که برای مدل‌سازی تصمیم‌گیری‌ها لازم است تا بتوان تصمیم‌گیری سازمانی را در نمودارها، به آسانی و دقیق توسط تحلیلگران کسب‌وکار، تعریف کرد و (به صورت اختیاری) خودکار نمود. تلاش برای مدل‌سازی تصمیم‌گیری‌ها توسط دو استاندارد مدل‌سازی موجود از دو منظر متفاوت مورد توجه قرار گرفته است:

- مدل‌سازی فرآیند کسب‌وکار (استاندارد BPMN) سعی کرده است که تصمیم‌گیری در فرآیندهای کسب‌وکار را با تعریف وظایف یا فعالیت‌های خاصی که در آن/آنها الگوریتم تصمیم‌گیری مورد پردازش قرار می‌گیرد، توصیف نماید.
- مدل‌سازی منطق تصمیم‌گیری (مانند مدل‌سازی PRR، PMML) سعی کرده است که منطق خاص مورد استفاده برای تصمیم‌گیری‌های فردی را تعریف کند، از این تلاش‌ها می‌توان به قواعدکسب‌وکار (business rules)، جداول تصمیم (decision tables)، یا مدل‌های تحلیلی قابل اجرا (executable analytic models) اشاره کرد.

تعدادی از صاحب نظران بر این باورند که تصمیم‌گیری، ساختار درونی دارد که در هیچ یک از این دیدگاه‌های مدل‌سازی قابل معرفی نیست. هدف ما این است که در DMN یک دیدگاه سوم (مدل‌سازی الزامات تصمیم‌گیری) را ارائه دهیم که حد واسط مدل‌های فرآیند کسب‌وکار و مدل‌های منطق تصمیم‌گیری است:

- مدل‌های فرآیند کسب‌وکار، وظایفی (وظیفه‌بررسی‌قواعدکسب‌وکار) را در فرآیندهای کسب‌وکار تعریف می‌کنند که در آنها، تصمیم‌گیری لازم است.
- مدل‌های الزامات تصمیم‌گیری، تصمیماتی را که باید در وظیفه‌بررسی‌قواعدکسب‌وکار، گرفته شود، روابط متقابل آنها و الزامات آنها برای منطق تصمیم‌گیری را تعریف می‌کنند.
- مدل‌های منطق تصمیم‌گیری، تصمیمات مورد نیاز را با جزئیات کافی تعریف می‌کنند تا اعتبارسنجی و/یا اتوماسیون آنها، امکان پذیر باشد.

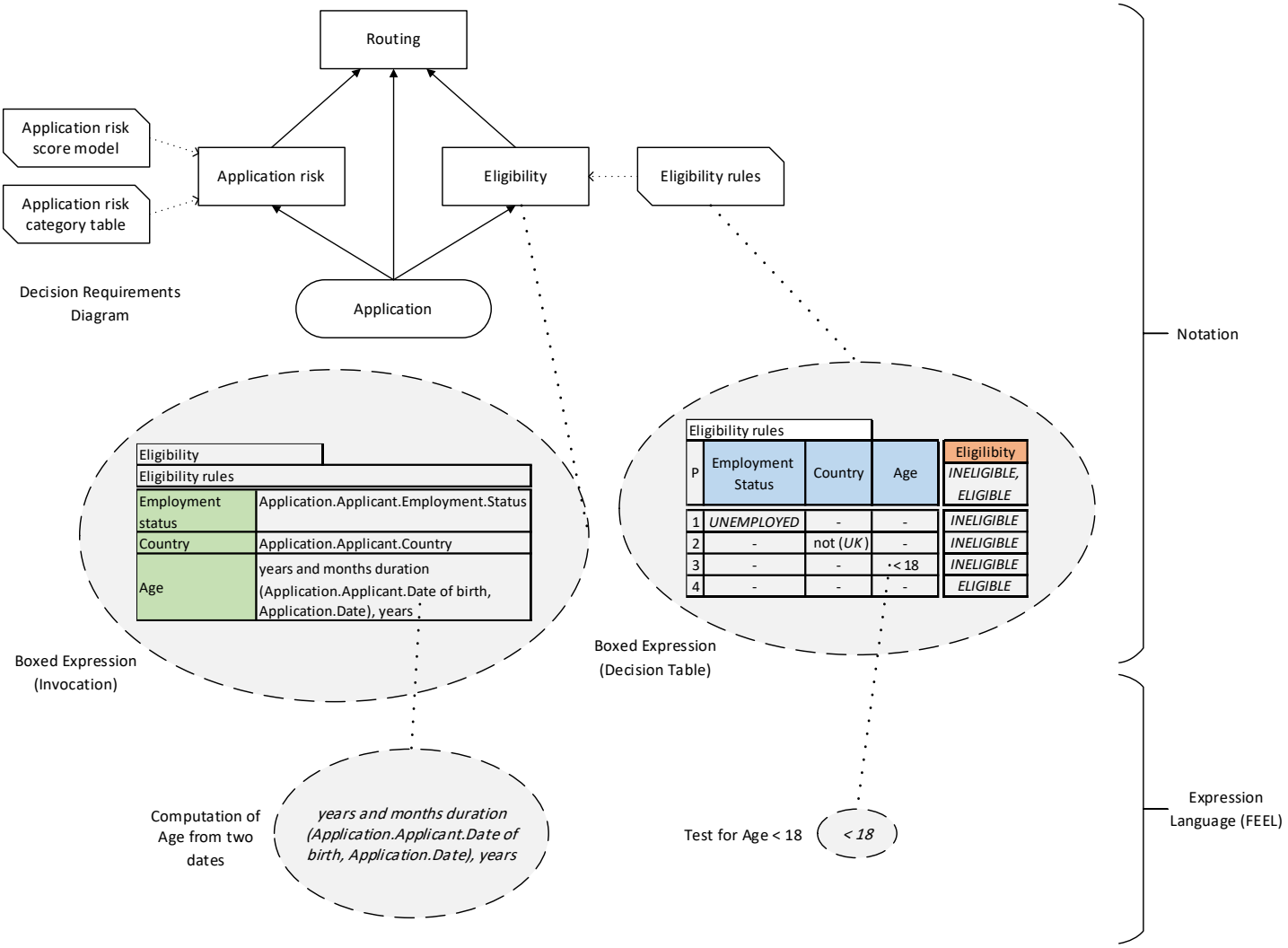
روی هم رفته، مدل‌های الزامات تصمیم‌گیری و منطق تصمیم‌گیری می‌توانند یک مدل تصمیم‌گیری کامل را ارائه دهند که با مشخص کردن جزئیات تصمیم‌گیری انجام شده در وظایف (وظیفه‌بررسی‌قواعدکسب‌وکار) فرآیند، مدل فرآیند کسب‌وکار مورد نظرشان را تکمیل می‌کنند. روابط بین این سه نوع مدل‌سازی در شکل زیر نشان داده شده است.



وجود ارتباط مدل‌های شکل بالا، مدل‌سازی دقیق نقش قواعد کسب‌وکار و مدل‌سازی تحلیلی فرآیندهای کسب‌وکار، اعتبارسنجی مدل‌ها، طراحی و اتوماسیون فرآیندها از بالا به پایین، و اجرای خودکار تصمیم‌گیری (به عنوان مثال، سیستم مدیریت فرآیند که یک سرویس تصمیم‌گیری مستقر شده از یک سیستم مدیریت قواعدکسب‌وکار را فراخوانی می‌کند) را امکان‌پذیر می‌سازد.

اگرچه شکل بالا ارتباط بین یک مدل فرآیند کسب‌وکار و یک مدل تصمیم‌گیری را به منظور توضیح رابطه بین DMN و سایر استانداردها نشان می‌دهد، باید تاکید کرد که DMN به BPMN و دو سطح مربوط به آن (الزامات تصمیم‌گیری و منطق تصمیم‌گیری) وابسته نیست. به عبارت دیگر مدل‌های مبتنی بر DMN ممکن است به طور مستقل یا به صورت با واسطه برای مدل‌سازی یک حوزه تصمیم‌گیری بدون هیچ ارجاعی به فرآیندهای کسب‌وکار استفاده شوند (برای درک بهتر به شکل بعدی توجه کنید).

DMN ساختارهایی را ارائه می‌دهد که هم مدل‌سازی الزامات تصمیم‌گیری و هم مدل‌سازی منطق تصمیم‌گیری را شامل می‌شود. برای مدل‌سازی الزامات تصمیم‌گیری، مفهومی به نام نمای گرافیکی الزامات تصمیم‌گیری (DRG) را تعریف می‌کند که شامل مجموعه‌ای از عناصر و قواعد متصل به آنها، و یک نماد مکاتبه‌ای است: نمودار الزامات تصمیم‌گیری (DRD). برای مدل‌سازی منطق تصمیم‌گیری، زبانی به نام FEEL برای تعریف و جمع‌آوری جداول تصمیم‌گیری، محاسبات، منطق if/then/else، ساختارهای داده نمونه و منطق تعریف‌شده خارجی از جاوا و PMML برای دستورات اجرایی با معانی تعریف شده گفتاری، ارائه می‌کند. همچنین یک نماد برای منطق تصمیم‌پیری ("کادر عبارات") ارائه می‌دهد که بتوان اجرای منطق تصمیم‌گیری را به صورت گرافیکی ترسیم نمود و با عناصر یک نمودار الزامات تصمیم‌گیری مرتبط کرد. رابطه بین این سازه‌ها در شکل بعد نشان داده شده است.



2- الزامات (DRG و DRD)

سطح الزامات تصمیم‌گیری یک مدل تصمیم‌گیری در DMN شامل یک نمودار الزامات تصمیم‌گیری (DRG) است که در یک یا چند نمودار الزامات تصمیم‌گیری (DRD) به تصویر کشیده شده است. یک DRG یک حوزه تصمیم‌گیری را مدل می‌کند و مهمترین عناصر درگیر در آن و وابستگی‌های بین آنها را نشان می‌دهد. عناصری که در مدل‌سازی وجود دارند شامل تصمیم‌گیری‌ها، حوزه‌های دانش کسب‌وکار، منابع دانش کسب‌وکار، داده‌های ورودی و سرویس‌تصمیم‌گیری می‌باشند که در زیر توضیح داده شده است:

- یک عنصر تصمیم‌گیری، بیانگر عمل تعیین خروجی از تعدادی ورودی، با استفاده از منطق تصمیم‌گیری است که ممکن است به یک یا چند مدل دانش کسب‌وکار ارجاع شده باشد.
- یک عنصر مدل دانش کسب‌وکار نشان دهنده تابعی است که دانش کسب‌وکار را محصور می‌کند، به عنوان مثال، می‌توان به قواعد کسب‌وکار، یک جدول‌تصمیم‌گیری، یا یک مدل تحلیلی اشاره کرد.
- یک عنصر داده ورودی، اطلاعاتی را نشان می‌دهد که توسط یک یا چند تصمیم‌گیری به عنوان ورودی استفاده می‌شود.
- یک عنصر منبع دانش کسب‌وکار، یک امکان برای یک مدل دانش کسب‌وکار یا تصمیم‌گیری را نشان می‌دهد.
- یک عنصر سرویس‌تصمیم‌گیری، مجموعه‌ای از تصمیم‌گیری‌های قابل استفاده مجدد را نشان می‌دهد که می‌توانند به صورت داخلی یا خارجی فراخوانی شوند.

وابستگی بین این عناصر به سه الزام، اطلاعات، دانش و امکان وابسته است که در زیر توضیح داده شده است:

- یک نیاز اطلاعاتی به معنای داده ورودی یا خروجی تصمیم‌گیری است که مانند ورودی یک تصمیم‌گیری استفاده می‌شود.
- یک نیاز دانش به فراخوانی یک مدل دانش کسب‌وکار یا سرویس‌تصمیم‌گیری توسط منطق تصمیم‌گیری یک تصمیم‌گیری، اشاره می‌کند.
- یک امکان، نشان دهنده وابستگی یک عنصر DRG به عنصر DRG دیگر است که به عنوان منبع راهنمایی یا دانش عمل می‌کند.

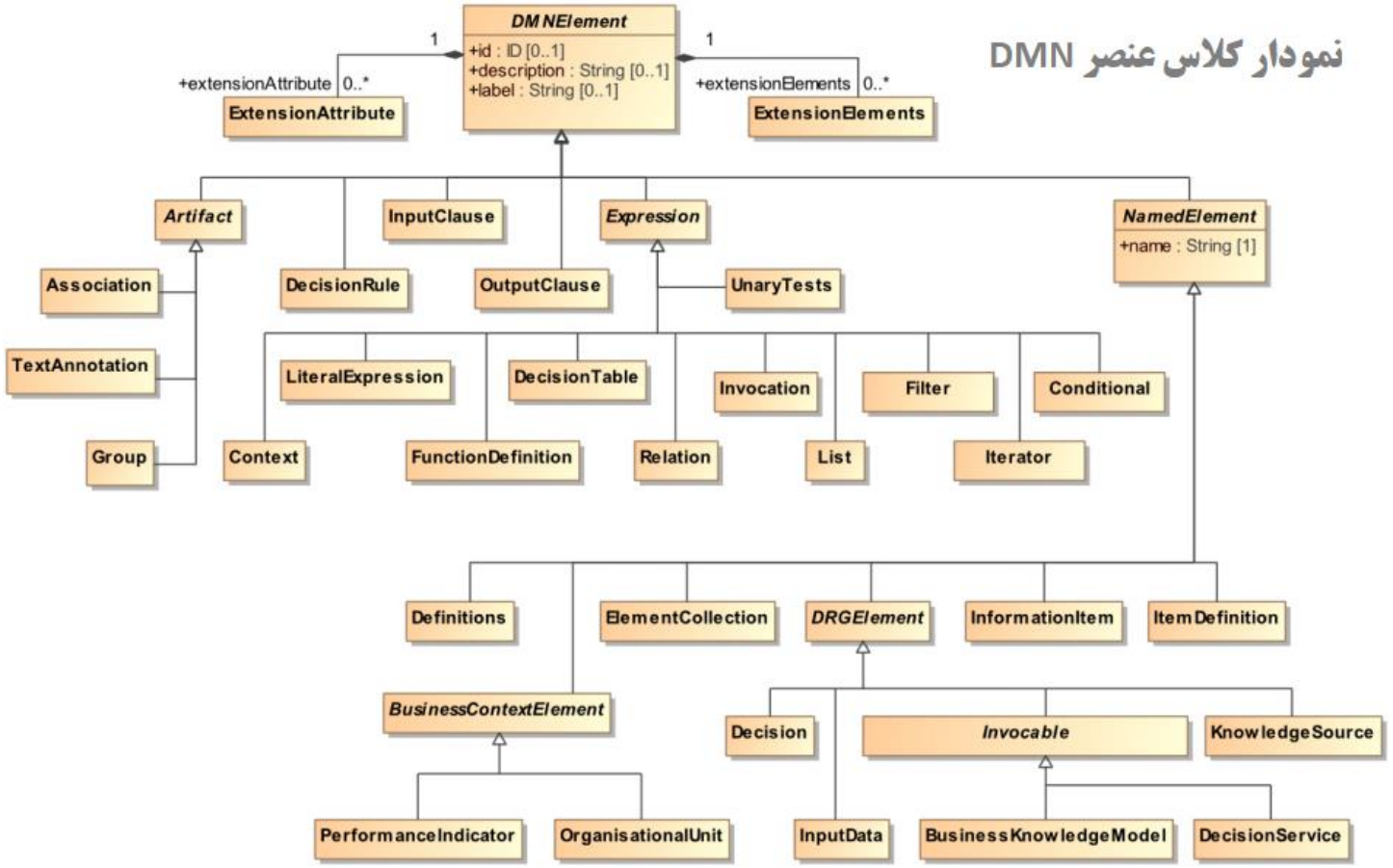
DRD ها همچنین ممکن است شامل تعدادی مصنوع زیر باشند که برای حاشیه‌نویسی نمودار استفاده می‌شود:

- یک حاشیه‌نویسی‌متنی، متنی است که توسط طراحی کننده مدل، وارد شده است و برای توضیح هر چیزی استفاده می‌شود.
- یک وابستگی، یک اتصال‌دهنده نقطه‌دار است که برای پیوند یک حاشیه‌نویسی‌متنی به یک عنصر DRG استفاده می‌شود.
- یک گروه، مکانیزم بصری برای گروه‌بندی غیررسمی عناصر یک نمودار است.

موارد ذکر شده در جدول زیر خلاصه شده است و در بخش‌های بعدی با جزئیات بیشتر توضیح داده شده است.

DRG نموداری است متشکل از عناصری که توسط نیازمندی‌ها به هم متصل شده‌اند، و به این معنا که تمام الزامات مدل‌سازی‌شده برای هر تصمیم‌گیری در DRG (منابع اطلاعات، دانش و امکانات) مستقل بوده و در همان DRG وجود دارد. مهم است که این تعریف کامل DRG را از یک DRD که هر نمای خاصی از آن را ارائه می‌دهد (که ممکن است یک نمایش جزئی یا فیلتر شده باشد)، متمایز کنیم.

2- 1- 1- متامدل عنصر DMN



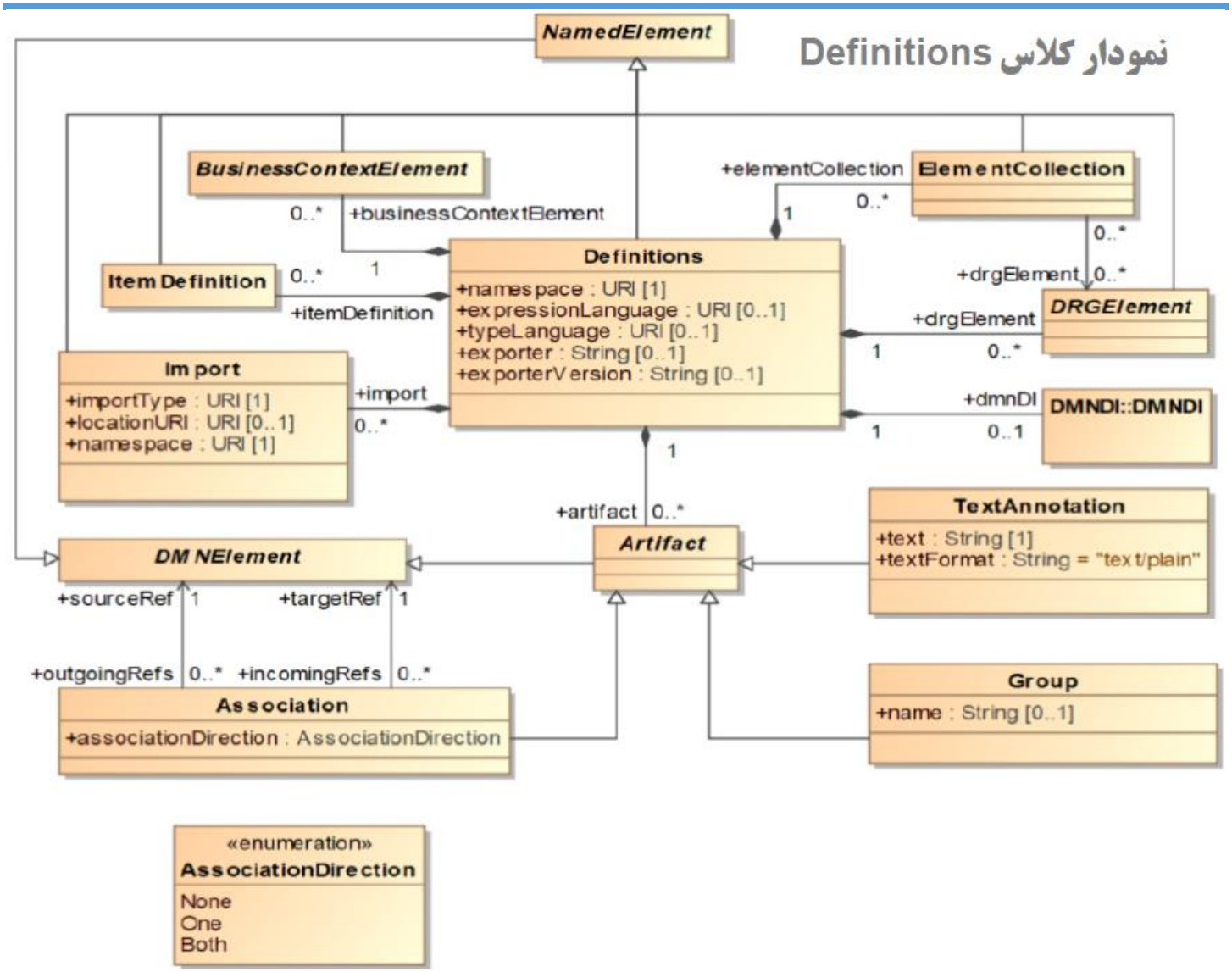
کلاس DMNElement یک ابرکلاس مفهومی برای عناصر مدل تصمیم‌گیری است. این کلاس، مشخصه‌های اختیاری id، description و label را ارائه می‌کند که رشته‌هایی هستند که عناصر دیگر به ارث می‌برند. مشخصه id یک DMNElement بیشتر به نگارش یک شناسه XML (http://www.w3.org/TR/2004/REC-xmlschema-2-20041028/datatypes.html#ID) محدود می‌شود و باید منحصر به فرد باشد. مدل تصمیم‌گیری DMNElement دارای مشخصه‌های انتزاعی NamedElement است و عبارت NamedElement ویژگی name مورد نیاز را اضافه می‌کند و شامل مشخصه‌های انتزاعی BusinessContextElement و DRGElement، و همچنین مشخصه‌های مشخص Definitions، ItemDefinition، InformationItem، ElementCollection و DecisionService است.

جدول زیر مشخصات و ارتباط مدل DMNElement را نشان می‌دهد.

مشخصه‌های DMNElement و ارتباط مدل مشخصه	شرح
id: ID[1..0]	شناسه یک مشخصه اختیاری برای عنصر است. باید در تعاریف عنصر، یک مقدار منحصر به فرد باشد.
description: String [0.. 1]	شرحی برای عنصر مورد نظر است.
label: String [0.. 1]	یک توضیح کوتاه جایگزین برای عنصر مورد نظر است. در درجه اول باید برای عناصری استفاده شود که ویژگی name ندارند، به عنوان مثال، یک عبارت ورودی. مشابه مشخصه description، هیچ نماد تعریف شده‌ای ندارد و نه به

مشخصه‌های DMNElement و ارتباط مدل مشخصه	شرح
	عنصر DMNLabel که در نمودار Interchange استفاده می‌شود و نه به مشخصه outputLabel یک جدول تصمیم‌گیری مربوط است.
extensionElements: ExtensionElement [0..1]	این مشخصه به عنوان یک ظرف برای اتصال عناصر بیشتر به هر عنصر DMN استفاده می‌شود. برای اطلاعات بیشتر در مورد توسعه پذیری به بخش‌های بعدی مراجعه کنید.
extensionAttributes: ExtensionAttribute [0..*]	این مشخصه برای پیوست کردن مشخصه‌های توسعه‌یافته نام‌گذاری شده و تداعی‌های مدل استفاده می‌شود. این ارتباط زمانی که از تبادل شمای XML استفاده می‌شود قابل اجرا نیست. برای اطلاعات بیشتر در مورد توسعه پذیری به بخش‌های بعدی مراجعه کنید.

مشخصه‌های NamedElement و ارتباطات مدل مشخصه	شرح
Name: string	مشخصه نام عنصر است.



کلاس Definitions بیرونی‌ترین شیء حاوی تمام عناصر یک مدل تصمیم‌گیری DMN است. این نمودار کلاس، محدوده قابلیت‌نمایش و فضای‌نام را برای همه عناصر موجود تعریف می‌کند. عناصری که در یک نمونه از Definitions موجود است، چرخه حیات تعریف شده خود را دارند و با حذف عناصر دیگر حذف نمی‌شوند. تبادل فایل‌های DMN همیشه از طریق یک یا چند Definitions انجام می‌شود.

کلاس Definitions نوعی NamedElement است که مشخصه name و مشخصه‌های اختیاری، id، description و label را که از نوع String هستند به ارث می‌برد. یک نمونه از Definitions یک فضای‌نام دارد که یک String است. فضای‌نام مورد نظر، فضای نام هدف پیش‌فرض را برای عناصر موجود در Definitions مشخص می‌کند و از قراردادی که توسط XML Schema ایجاد شده است، پیروی می‌کند.

یک نمونه از Definitions ممکن است یک expressLanguage را مشخص کند، که یک URI است که زبان عبارات پیش‌فرض مورد استفاده در عناصر را در محدوده این Definitions شناسایی می‌کند. این مقدار ممکن است در هر LiteralExpression منفرد بازنویسی شود. زبان باید در قالب URI مشخص شود. زبان عبارات پیش‌فرض، FEEL است که با URI زیر نشان داده شده است:

https://www.omg.org/spec/DMN/20191111/FEEL/

زبان عبارات ساده S-FEEL، که زیرمجموعه FEEL است، با همان URI نشان داده می‌شود. DMN برای زبان‌های بیانی که قرار نیست به طور خودکار تفسیر شوند URI زیر را ارائه می‌کند:

http://www.omg.org/spec/DMN/uninterpreted/20140801

یک نمونه از Definitions ممکن است یک typeLanguage را مشخص کند، که یک URI است که نوع زبان پیش‌فرض مورد استفاده در عناصر را در محدوده این Definitions شناسایی می‌کند. به عنوان مثال، typeLanguage با مقدار "http://www.w3.org/2001/XMLSchema" نشان می‌دهد که ساختارهای داده تعریف شده در آن Definitions، به طور پیش‌فرض، به شکل انواع شمای XML هستند. اگر مقدار typeLanguage مشخص نشده باشد، نوع پیش‌فرض زبان FEEL است. این مقدار ممکن است در هر ItemDefinition منحصر به فرد بازنویسی شود. typeLanguage باید در قالب URI مشخص شود. URI برای FEEL مقدار زیر است:

https://www.omg.org/spec/DMN/20191 111/FEEL/

و URI زیر را می‌توان برای نشان دادن این که نوع تعریف‌شده یک مفسر یا interpreter نیست استفاده کرد.

http://www.omg.org/spec/DMN/uninterpreted/20140801

نمونه‌ای از Definitions ممکن است exporter و exporterVersion را مشخص کند که رشته‌هایی هستند که ابزار و نسخه مورد استفاده برای ایجاد سریال‌سازی XML را نام‌گذاری می‌کنند. در استانداردهایی مانند BPMN، این امر به تبادل نمودارهای مدل بین ابزارها کمک می‌کند.

یک نمونه از Definitions از صفر یا چند drgElement تشکیل شده است، از طرفی نمونه‌های DRGElement، صفر یا چند elementCollections و نمونه‌های ElementCollection، صفر یا چند itemDefinitions و نمونه‌های ItemDefinition، صفر یا چند businessContextElements از BusinessContextElement هستند.

آن ممکن است شامل چندین import مرتبط باشد که نمونه‌هایی از Import هستند. Import ها برای وارد کردن عناصر تعریف شده خارج از این Definitions استفاده می‌شود، به عنوان مثال. در سایر عناصر Definitions و آنها را برای استفاده توسط عناصر در این Definitions در دسترس قرار می‌دهد.

Definitions تمام مشخصات و مدل‌های ارتباطی را از NamedElement به ارث می‌برد. جدول زیر مشخصات افزوده شده و ارتباط مدل عناصر Definitions را نشان می‌دهد.

مشخصه	شرح
namespace: anyURI [1]	این مشخصه، فضای‌نام مرتبط با این Definitions را مشخص می‌کند و از قرارداد ایجاد شده توسط شمای XML پیروی می‌کند.
expressionLanguage: anyURI [0.. 1]	این مشخصه، زبان عبارت مورد استفاده در LiteralExpressions را در محدوده این Definitions مشخص می‌کند. پیش‌فرض آن، FEEL است. این مقدار ممکن است در هر LiteralExpression منحصر به فرد، بازنویسی شود. زبان مورد نظر باید در قالب URI مشخص شود.
typeLanguage: anyURI [0.. 1]	این مشخصه، نوع زبان مورد استفاده در LiteralExpressions را در محدوده این Definitions مشخص می‌کند. پیش‌فرض آن، FEEL است. این مقدار ممکن است در هر ItemDefinition منحصر به منفرد، بازنویسی شود. زبان مورد نظر باید در قالب URI مشخص شود.
exporter: string [0..1]	این مشخصه، ابزار مورد استفاده برای ارسال سریال‌سازی XML را مشخص می‌کند.
exporterVersion: string [0.. 1]	این مشخصه، نسخه ابزار مورد استفاده برای ارسال سریال سازی XML را مشخص می‌کند.
itemDefinition: ItemDefinition [*]	این مشخصه، نمونه‌هایی از ItemDefinition را که در این Definitions موجود است، فهرست می‌کند.

مشخصه	شرح
drgElement: DRGElement [*]	این مشخصه، نمونه‌هایی از DRGElement را که در این Definitions موجود است، فهرست می‌کند.
businessContextElement: BusinessContextElement [*]	این مشخصه، نمونه‌هایی از BusinessContextElement را که در این Definitions موجود است، فهرست می‌کند.
elementCollection: ElementCollection [*]	این مشخصه، نمونه‌هایی از ElementCollection را که در این Definitions موجود است، فهرست می‌کند.
import: Import [*]	این مشخصه، برای دریافت عناصر تعریف‌شده خارجی و در دسترس قراردادن آنها برای استفاده توسط عناصر در این Definitions استفاده می‌شود.
artifact: Artifact [0..*]	مصنوعات، شامل حاشیه‌نویسی‌های متنی، گروه‌ها و وابستگی‌ها در میان عناصر DMN است.
dmnDI: DMNDI [0..1]	این مشخصه، حاوی اطلاعات تبادل نمودار موجود در این Definitions است.

2- 1- 3- متامدل Import

کلاس Import هنگام ارجاع به عناصر خارجی استفاده می‌شود، چه نمونه‌های DMN، از DRGElement یا ItemDefinition موجود در سایر عناصر Definitions یا عناصر غیر DMN، مانند یک شمای XML یا یک فایل PMML. Import ها باید به صراحت تعریف شود.

یک نمونه از Import یک importType دارد که رشته‌ای است که نوع دریافت مرتبط با عنصر را مشخص می‌کند. به عنوان مثال، مقدار زیر نشان می‌دهد که عنصر دریافت شده یک شمای XML است:

http://www.w3.org/2001/XMLSchema

فضای‌نام DMN نشان می‌دهد که عنصر دریافت شده یک عنصر DMN از Definitions است.

مکان عنصر دریافت شده ممکن است با مرتبط کردن یک locationURI اختیاری با یک نمونه از Import مشخص شود. LocationURI یک URI است.

هر نمونه از Import یک namespace دارد، که یک URI است که فضای‌نام عنصر وارد شده را مشخص می‌کند، و همچنین یک name به ارث رسیده از NamedElement، که رشته‌ای است که به عنوان پیشوند در نام‌های واجد شرایط فضای‌نام عمل می‌کند، مانند typeRefs که ItemDefinitions و عباراتی که به InformationItems وارد شده، ارجاع می‌دهند. مقدار namespace باید به صورت سراسری، منحصربه‌فرد باشد، اما name دریافت مورد نظر، که معمولاً یک نام کوتاه مناسب برای کسب‌وکار است، باید از نام‌های دیگر دریافت‌ها، تصمیم‌گیری‌ها، داده‌های‌ورودی، مدل‌های دانش کسب‌وکار، سرویس‌تصمیم‌گیری، و اقلام تعاریف در فقط مدل دریافت، متمایز باشد. جدول زیر مشخصات و ارتباط مدل عنصر Import را نشان می‌دهد.

مشخصه	شرح
importType: anyURI	نوع دریافت مرتبط با این Import را مشخص می‌کند.
locationURI: anyURI [0.. 1]	مکان عنصر دریافت شده را مشخص می‌کند.
namespace: anyURI	فضای‌نام عنصر دریافت شده را مشخص می‌کند.

2- 1- 4- متامدل Element Collection

کلاس ElementCollection برای تعریف گروه‌های نامگذاری شده از نمونه‌های DRGElement استفاده می‌شود. ElementCollection ممکن است برای هر هدف مرتبط با یک پیاده‌سازی استفاده شود، به عنوان مثال:

- برای شناسایی طرح فرعی مورد نیاز یک مجموعه از یک یا چند تصمیم‌گیری.
- برای شناسایی عناصری که باید روی یک DRD به تصویر کشیده شوند.

ElementCollection نوعی NamedElement است به گونه‌ای که هر ElementCollection مشخصه name و مشخصه اختیاری، id، description و label را که رشته هستند، به ارث می‌برد. عنصر id ElementCollection باید در نمونه حاوی Definitions منحصر به فرد باشد. یک عنصر

ElementCollection دارای چند drgElement مرتبط است، که نمونه‌هایی از DRGElement هستند که در این ElementCollection با هم به عنوان یک گروه تعریف می‌شوند. توجه داشته باشید که یک عنصر ElementCollection باید به نمونه‌های DRGElement که جمع‌آوری می‌کند ارجاع دهد، نه اینکه حاوی آنها باشد. نمونه‌های DRGElement فقط می‌توانند در عناصر Definitions موجود باشند. ElementCollection تمام مشخصات و مدل‌های ارتباطی را از NamedElement به ارث می‌برد. جدول زیر مشخصات افزوده شده و ارتباط مدل عنصر ElementCollection را نشان می‌دهد.

مشخصه	شرح
drgElement: DRGElement [*]	این مشخصه، نمونه‌هایی از DRGElement را که این ElementCollection گروه‌بندی می‌کند، فهرست می‌کند.

2- 1- 5- متامدل DRG Element

ابرکلاس DRGElement یک ابرکلاس انتزاعی برای تمام عناصر DMN است که در Definitions موجود است و دارای یک نمایش گرافیکی در یک DRD است. تمام عناصر یک مدل تصمیم‌گیری DMN که مستقیماً در یک عنصر Definitions موجود نیستند (به طور خاص: هر سه نوع الزامات یا requirement، بند یا bindings و قواعدتصمیم‌گیری یا decision rules، دریافت یا import و هدف یا objective) باید در یک نمونه از DRGElement یا در یک عنصر مدل که به نمونه‌ای از DRGElement ارجاع می‌نماید، موجود باشند. کلاس‌های خاص شده ابرکلاس DRGElement عبارتند از Decision، InputData، Invocable و KnowledgeSource می‌باشند.

کلاس Invocable بیشتر در BusinessKnowledgeModel و DecisionService تخصصی شده است. خود ابرکلاس DRGElement یک کلاس تخصص شده از NamedElement است که از آن مشخصه name و مشخصات اختیاری، id، description و label را به ارث می‌برد. id یک عنصر DRGElement باید در نمونه حاوی Definitions منحصر به فرد باشد.

نمودار الزامات تصمیم‌گیری (DRD) نمایش نموداری یک یا چند نمونه از DRGElement و اطلاعات، دانش و روابط مورد نیاز آنها است. نمونه‌های DRGElement به صورت رئوس در نمودار مورد نظر نشان داده می‌شوند. این رئوس، نمونه‌هایی از الزام اطلاعاتی، الزام دانش یا الزام امکان را نشان می‌دهند.

DRGElement تمام مشخصات و مدل‌های ارتباطی NamedElement را به ارث می‌برد. از طرف دیگر، مشخصات افزوده شده و ارتباطات عناصر مدل DRGElement را تعریف نمی‌کند.

2- 1- 6- متامدل Artifact

کلاس Artifact برای ارائه اطلاعات افزوده شده در یک مدل تصمیم‌گیری، استفاده می‌شود. DMN سه Artifact استاندارد را ارائه می‌دهد:

- **وابستگی یا Association**: از وابستگی‌ها می‌توان برای پیوند Artifact ها به هر DMNElement استفاده کرد. یک وابستگی، برای پیوند دادن اطلاعات و مصنوعات با عناصر گرافیکی DMN استفاده می‌شود. حاشیه‌نویسی‌متنی و سایر مصنوعات را می‌توان به عناصر گرافیکی دیگر وابسته کرد. یک سمت وابستگی در صورت لزوم می‌تواند جهت جریان (به عنوان مثال، جریان داده) را نشان دهد.
عنصر وابستگی، مشخصات و مدل‌های ارتباطی DMNElement را به ارث می‌برد. جدول زیر مشخصات افزوده شده و ارتباط مدل را برای یک Association ارائه می‌دهد.

مشخصه	شرح
associationDirection: AssociationDirection = None {None | One | Both}	مشخصه AssociationDirection مشخصه‌ای است که مشخص می‌کند آیا Association با یک نوک پیکان فلش گونه نشان داده می‌شود یا خیر. پیش‌فرض آن، None (بدون جهت گیری) است. مقدار One به این معنی است که نوک فلش گونه باید در سمت هدف این وابستگی، قرار گیرد. مقدار Both به این معنی است که در هر دو انتهای خط وابستگی، نوک فلش گونه وجود خواهد داشت.
sourceRef: DMNElement [1]	DMNElement که Association از آن وصل می‌شود.
targetRef: DMNElement [1]	DMNElement که Association به آن وصل می‌شود.

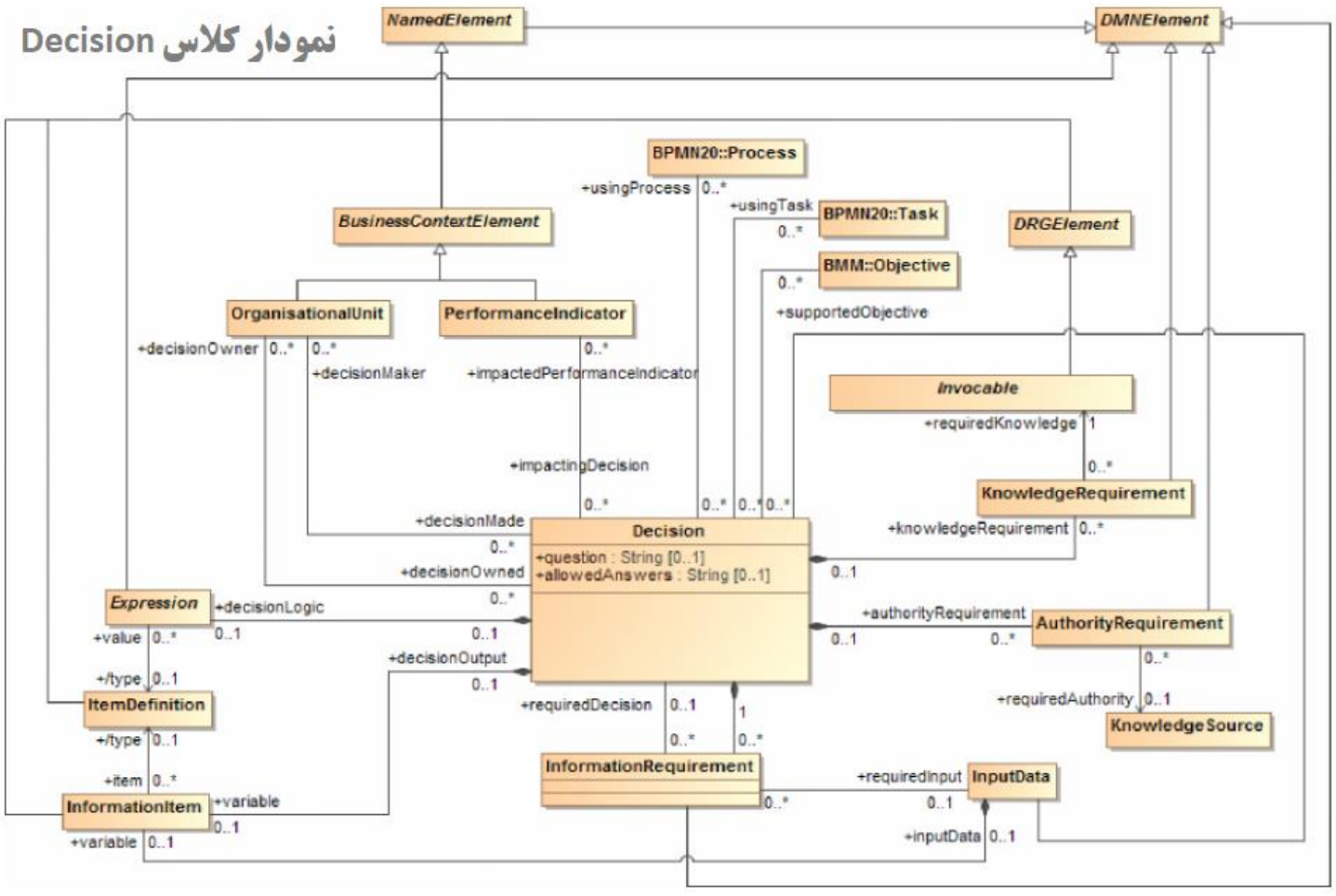
- **گروه یا Group:** شیء گروه یک مصنوع است که مکانیزم بصری را برای گروه‌بندی غیررسمی عناصر یک نمودار فراهم می‌کند. گروه‌ها اغلب برای برجسته کردن بخش‌های خاصی از نمودار بدون اضافه کردن محدودیت‌های اضافی عملکردی، استفاده می‌شوند. بخش‌های هایلایت‌شده (گروه‌بندی‌شده) نمودار اصولا برای اهداف گزارش و تجزیه‌وتحلیل تشکیل می‌شوند. گروه‌ها بر اجرای تصمیم‌گیری‌ها، تأثیری ندارند. به عنوان یک مصنوع، یک گروه یک DRGElement نیست، و بنابراین، نمی‌تواند به یک الزام اطلاعاتی، الزام دانش، یا الزام امکان، متصل شود. فقط می‌تواند به یک وابستگی، متصل شود. عنصر Group مشخصات و مدل‌های ارتباطی Artifact را به ارث می‌برد. جدول زیر مشخصات افزوده شده و ارتباط مدل را برای یک Group ارائه می‌دهد.

مشخصه	شرح
Name: String[0.. 1]	این مشخصه، نام توصیفی عنصر را مشخص می‌کند.

- **حاشیه‌نویسی‌متنی یا Text Annotation:** حاشیه‌نویسی‌متنی، مکانیزمی برای مدل‌ساز است تا اطلاعات متنی اضافی را برای خواننده نمودار DMN فراهم کند. عنصر TextAnnotation مشخصات و مدل‌های ارتباطی DMNElement را به ارث می‌برد. جدول زیر مشخصات افزوده شده را برای یک TextAnnotation ارائه می‌دهد.

مشخصه	شرح
text: string	مشخصه text مشخصه‌ای متنی است که مدل‌ساز می‌خواهد به واسطه آن با خواننده نمودار ارتباط برقرار کند.
textFormat: string = "text/plain"	این مشخصه، فرمت متن را مشخص می‌کند. باید از فرمت mime-type پیروی کند. مقدار پیش‌فرض این مشخصه، "text/plain" است.

2- 1- 7- متامدل Decision



کلاس Decision برای مدل‌سازی یک تصمیم‌گیری استفاده می‌شود. کلاس Decision یک کلاس تخصصی DRGElement است و مشخصه name و مشخصات اختیاری، id، description و label را از NamedElement به ارث می‌برد. name یک Invocable باید با نام هر Invocable، داده ورودی، تصمیم‌گیری، یا دریافت در مدل تصمیم‌گیری متفاوت باشد. علاوه بر این، ممکن است یک question و allowedAnswers داشته باشد که همگی از نوع String هستند.

منظور از مشخصه اختیاری description، شرح مختصری از تصمیم‌گیری است که در Decision گنجانده شده است. منظور از مشخصه اختیاری question، یک سؤال به زبان طبیعی است که Decision را مشخص می‌کند به طوری که خروجی Decision پاسخی به این سؤال است. منظور از مشخصه اختیاری allowAnswers، توصیفی به زبان طبیعی از پاسخ‌های مجاز برای سؤال مورد نظر است. این پاسخ‌ها مانند بله/خیر، لیستی از مقادیر مجاز، طیفی از مقادیر عددی و غیره است.

در یک DRD، یک نمونه از Decision با یک عنصر نمودار تصمیم‌گیری نشان داده می‌شود. یک عنصر Decision از یک decisionLogic، که نمونه‌ای از Expression است و از صفر یا چند informationRequirement، knowledgeRequirement و authorityRequirement تشکیل شده است که به ترتیب نمونه‌هایی از InformationRequirement، KnowledgeRequirement و AuthorityRequirement هستند. بعلاوه، یک Decision یک InformationItem را تعریف می‌کند که خروجی آن را نشان می‌دهد. این InformationItem ممکن است شامل یک typeRef اختیاری باشد که به یک ItemDefinition یا تعریف نوع دیگری اشاره می‌کند که نوع داده نتایج احتمالی Decision را مشخص می‌کند.

زیرگراف مورد نیاز یک عنصر Decision، گراف جهت‌دار متشکل از خود عنصر Decision، informationRequirements، knowledgeRequirements آن، و اتحاد زیرگراف‌های الزامی هر عنصر requiredDecision یا requiredKnowledge است. یعنی، زیرگراف الزامی یک عنصر Decision، بسته شدن informationRequirement، requiredInput، requiredDecision، knowledgeRequirement و requiredKnowledge است که از آن عنصر Decision شروع می‌شود.

به یک نمونه از Decision، مدل یک تصمیم‌گیری، گفته می‌شود که اگر و تنها در صورتی که همه عناصر informationRequirement و knowledgeRequirement به خوبی شکل گرفته باشند، به خوبی شکل گرفته است. این شرط به مستلزم آن است که عنصر Decision زیرگراف الزامی، غیر چرخه‌ای باشد، یعنی یک عنصر Decision، مستقیم یا غیرمستقیم به خود نیاز نداشته باشد. علاوه بر اجزای منطقی آن یعنی الزامات اطلاعاتی، منطق تصمیم‌گیری و غیره، مدل یک تصمیم‌گیری ممکن است محتوای کسب‌وکار تصمیم‌گیری را نیز مستند کند. محتوای کسب‌وکار برای نمونه‌ای از Decision با ارتباط آن با هر تعداد supportedObjectives که نمونه‌هایی از Objective هستند که در OMG BMM تعریف شده است و هر تعداد impactedPerformance Indicators، که نمونه‌هایی از Performance Indicator هستند و هر تعداد decisionMaker هستند و هر تعداد decisionOwner، که نمونه‌هایی از OrganisationalUnit هستند.

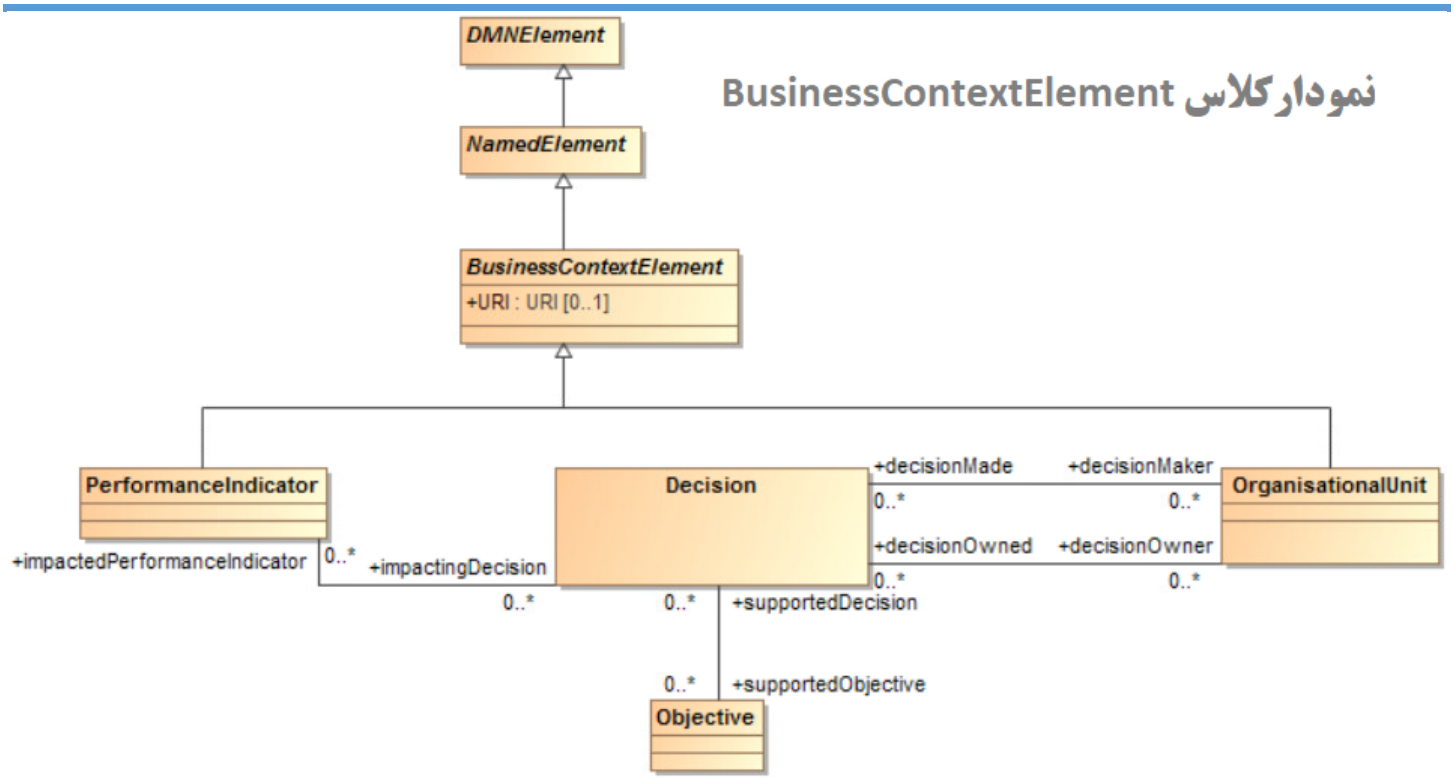
علاوه بر این، یک نمونه از Decision ممکن است به هر تعداد usingProcess اشاره کند، که نمونه‌هایی از فرآیند هستند که در OMG BPMN 2.0 تعریف شده‌اند، و هر تعداد usingTask، که نمونه‌هایی از فعالیت هستند که در OMG BPMN 2.0 تعریف شده‌اند، و که عبارتند از فرآیند و وظایفی که از عنصر Decision استفاده می‌کند.

کلاس Decision تمام مشخصات و مدل‌های ارتباطی را از DRGElement به ارث می‌برد. جدول زیر مشخصات افزوده شده و ارتباط مدل کلاس Decision را نشان می‌دهد.

مشخصه	شرح
question: string [0..1]	یک سؤال به زبان طبیعی که Decision را به گونه‌ای تعریف می‌کند که خروجی Decision پاسخی به سؤال است.
allowedAnswers: string [0..1]	شرحی به زبان طبیعی از پاسخ‌های مجاز مانند بله/خیر، فهرستی از مقادیر مجاز، طیفی از مقادیر عددی و غیره را برای سؤال مشخص می‌کند.
variable: InformationItem	نمونه‌ای از InformationItem که نتیجه این Decision را ذخیره می‌کند.
decisionLogic: Expression [0..1]	نمونه ای از Expression که منطق تصمیم‌گیری را برای این Decision نشان می‌دهد.

مشخصه	شرح
informationRequirement: InformationRequirement [*]	این مشخصه، مواردی از InformationRequirement را که این Decision را تشکیل می‌دهند، فهرست می‌کند.
knowledgeRequirement: KnowledgeRequirement [*]	این مشخصه، مواردی از KnowledgeRequirement را که این Decision را تشکیل می‌دهند، فهرست می‌کند.
authorityRequirement: AuthorityRequirement [*]	این مشخصه، مواردی از AuthorityRequirement را که این Decision را تشکیل می‌دهند، فهرست می‌کند.
supportedObjective: BMM::Objective[*]	این مشخصه، مواردی از BMM::Objective را که توسط این Decision پشتیبانی می‌شود، فهرست می‌کند.
impactedPerformanceIndicator: PerformanceIndicator [*]	این مشخصه، مواردی از PerformanceIndicator را که تحت تأثیر این Decision قرار گرفته‌اند، فهرست می‌کند.
decisionMaker: OrganisationalUnit [*]	این مشخصه، مواردی از OrganisationalUnit که این Decision را می‌گیرند، فهرست می‌کند.
decisionOwner: OrganisationalUnit [*]	این مشخصه، مواردی از OrganisationalUnit که مالک این Decision هستند، فهرست می‌کند.
usingProcesses: BPMN::process [*]	این مشخصه، مواردی از BPMN::process را فهرست می‌کند که نیاز به این Decision دارند.
usingTasks: BPMN::task[*]	این مشخصه، مواردی از BPMN::task که این Decision را می‌گیرند، فهرست می‌کند.

2- 1- 8- متامدل Business Context Element



کلاس انتزاعی BusinessContextElement و کلاس‌های تخصصی‌شده آن یعنی PerformanceIndicator و OrganizationUnit مکان‌هایی هستند که تعریفی از سایر متامدل‌های OMG مانند OMG OSM در خود دارند که پیش‌بینی می‌شود در صورت توسعه بیشتر این استاندارد، به آنها نیاز باشد. کلاس BusinessContextElement یک کلاس خاص شده از کلاس NamedElement است که از آن مشخصه name و مشخصات اختیاری، id، description و

label را به ارث می‌برد. علاوه بر این، نمونه‌هایی از BusinessContextElements ممکن است دارای یک مشخصه به نام URI باشند که یک URI در خود ذخیره خواهد کرد، و همچنین ممکن است موارد زیر را نیز شامل شوند:

- یک نمونه از PerformanceIndicator به تعدادی از ImpactingDecision ارجاع می‌دهد، که عناصر Decision بر آن تأثیر می‌گذارند.
- یک نمونه از OrganisationalUnit به تعدادی decisionMade و decisionOwned ارجاع می‌دهد، که عناصر Decision هستند که تصمیم‌گیری‌هایی را که واحد سازمان می‌گیرد یا مالک آن است مدل می‌کند.

کلاس BusinessContextElement تمام مشخصات و مدل‌های ارتباطی را از NamedElement به ارث می‌برد. جدول زیر مشخصات افزوده شده و ارتباط مدل BusinessContextElementclass را نشان می‌دهد.

مشخصه	شرح
URI: anyURI [0..1]	URI این BusinessContextElement را معرفی می‌کند.

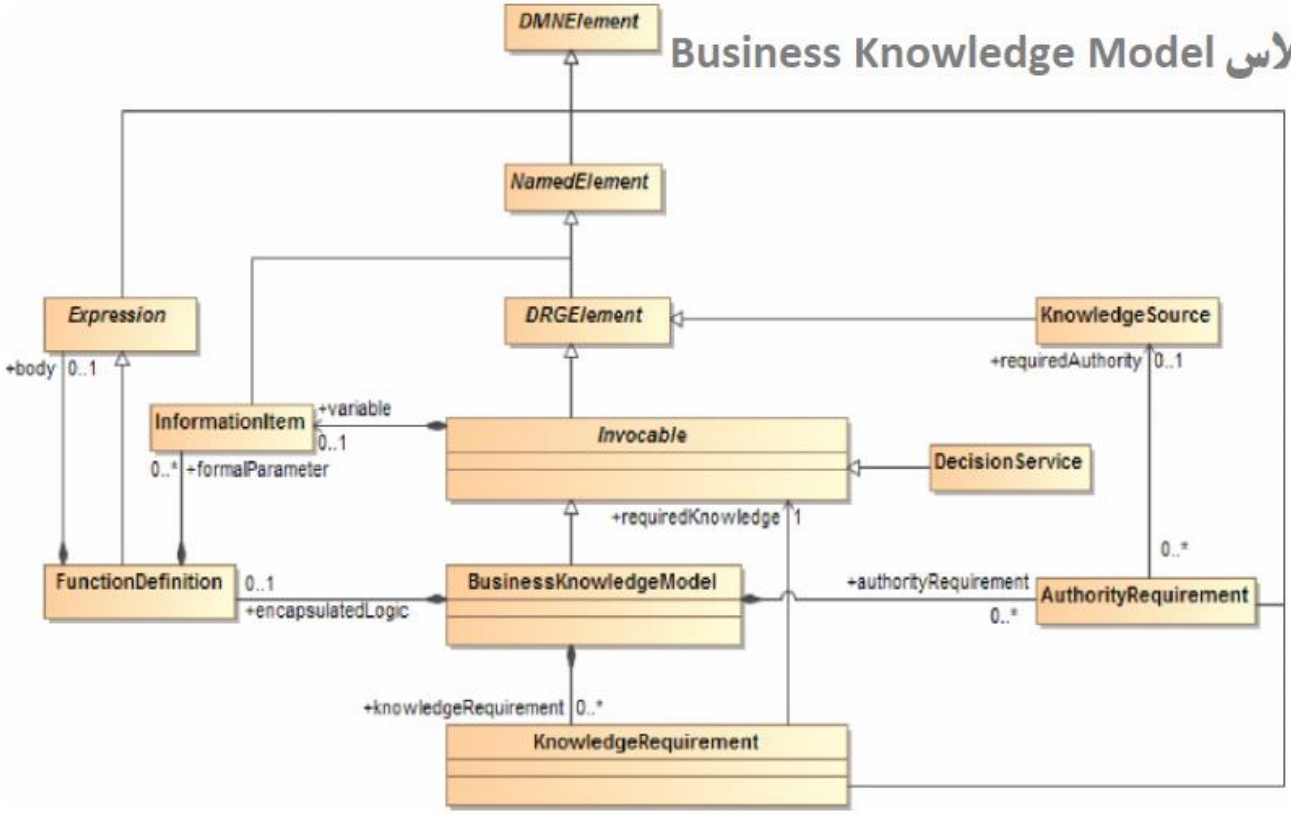
کلاس PerformanceIndicator تمام مشخصات و مدل‌های ارتباطی را از BusinessContextElement به ارث می‌برد. جدول زیر مشخصات افزوده شده و ارتباط مدل کلاس PerformanceIndicator را نشان می‌دهد.

مشخصه	شرح
impactingDecision: Decision [*]	این مشخصه، مواردی از Decision را فهرست می‌کند که بر PerformanceIndicator تأثیر می‌گذارد.

کلاس OrganisationalUnit تمام مشخصات و مدل‌های ارتباطی را از BusinessContextElement به ارث می‌برد. جدول زیر مشخصات افزوده شده و ارتباط مدل کلاس OrganisationalUnit را نشان می‌دهد.

مشخصه	شرح
decisionMade: Decision [*]	این مشخصه، مواردی از Decision را که توسط این OrganisationalUnit گرفته می‌شود فهرست می‌کند.
decisionOwned: Decision [*]	این مشخصه، مواردی از Decision را که متعلق به این OrganisationalUnit است فهرست می‌کند.

نمودار کلاس Business Knowledge Model



یک مدل دانش کسب‌وکار دارای یک بخش انتزاعی است که منطق تصمیم‌گیری قابل استفاده مجدد را نشان می‌دهد، و یک بخش مشخص، که الزام می‌کند که منطق تصمیم‌گیری باید یک کادر تعریف تابع FEEL باشد. یک سرویس‌تصمیم‌گیری نیز یک عنصر قابل استناد است، و بنابراین می‌تواند به عنوان دانش مورد نیاز در سایر تصمیم‌گیری‌ها و مدل‌های دانش کسب‌وکار مورد استفاده قرار گیرد.

کلاس Invocable برای مدل‌سازی یک عنصر invocable و کلاس BusinessKnowledgeModel برای مدل سازی یک مدل دانش کسب‌وکار استفاده می‌شود. کلاس Invocable کلاس خاص شده DRGElement است و name و مشخصات اختیاری id، description و label را از NamedElement به ارث می‌برد. مشخصه name یک Invocable باید با مشخصه name هر Invocable، داده‌ورودی، تصمیم‌گیری یا دریافت در مدل تصمیم‌گیری متفاوت باشد. کلاس BusinessKnowledgeModel کلاس خاص شده Invocable است که علاوه بر این مشخصه variable را به ارث می‌برد. یک عنصر BusinessKnowledgeModel ممکن است صفر یا چند knowledgeRequirement داشته باشد، که نمونه‌ای از KnowledgeRequirement است، و صفر یا چند authorityRequirement، که نمونه‌هایی از AuthorityRequirement هستند. این عناصر مدل در زیر توضیح داده شده است.

زیرگراف مورد نیاز یک عنصر BusinessKnowledgeModel، گراف جهت‌داری متشکل از خود عنصر BusinessKnowledgeModel، عناصر knowledgeRequirement آن، و تجمیع زیرگراف‌های الزامی همه عناصر requiredKnowledge است که توسط knowledgeRequirements آن ارجاع داده می‌شود.

به نمونه‌ای از BusinessKnowledgeModel، نمونه مناسب گفته می‌شود که اگر و تنها در صورتی که هیچ knowledgeRequirement نداشته باشد، یا تمام عناصر knowledgeRequirement آن به خوبی شکل گرفته باشند. این شرط به ویژه مستلزم آن است که زیرگراف الزامات یک عنصر BusinessKnowledgeModel غیر چرخه‌ای باشد، یعنی یک عنصر BusinessKnowledgeModel به طور مستقیم یا غیرمستقیم به خود نیاز نداشته باشد.

در سطح منطق تصمیم‌گیری، یک عنصر BusinessKnowledgeModel حاوی یک FunctionDefinition است که نمونه‌ای از Expression شامل صفر یا چند پارامتر است که نمونه‌هایی از Information Item هستند. FunctionDefinition که در یک عنصر BusinessKnowledgeModel موجود است، ماژول

منطق تصمیم‌گیری قابل استفاده مجدد است که توسط این عنصر BusinessKnowledgeModel نشان داده می‌شود. یک عنصر Invocable شامل یک InformationItem است که دارای یک مرجع غیرقابل استناد به دانش کسب‌وکاری انتزاعی است که به یک Decision اجازه می‌دهد آن را با نام فراخوانی کند. نام آن InformationItem باید با نام عنصر Invocable یکی باشد. Invocable تمام مشخصات و ارتباط‌های مدل را از DRGElement به ارث می‌برد.

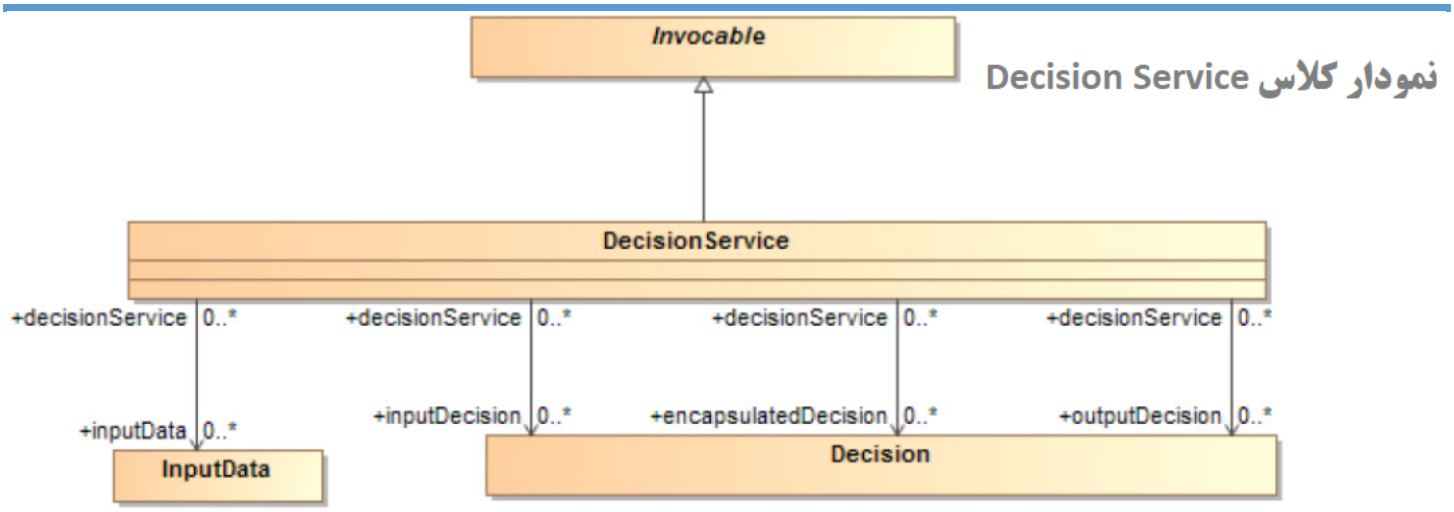
جدول زیر مشخصات افزوده شده و ارتباط مدل کلاس Invocable را نشان می‌دهد. جدول بعدی نیز مشخصات افزوده شده و ارتباط مدل کلاس BusinessKnowledgeModel را نشان می‌دهد.

مشخصه	شرح
variable: InformationItem	این مشخصه، متغیری را تعریف می‌کند که به تابع تعریف شده توسط FunctionDefinition مرتبط شده است و به منطق تصمیم‌گیری اجازه می‌دهد تا بتواند تابع را با نام فراخوانی کند.

مشخصه	شرح
encapsulatedLogic: FunctionDefinition [0.. 1]	تابعی که منطق محصور شده توسط این BusinessKnowledgeModel را کپسوله می‌کند.
knowledgeRequirement: KnowledgeRequirement [*]	این مشخصه، مواردی از KnowledgeRequirement را فهرست می‌کند که این BusinessKnowledgeModel را تشکیل می‌دهند.
authorityRequirement: AuthorityRequirement [*]	این مشخصه، مواردی از AuthorityRequirement را فهرست می‌کند که این BusinessKnowledgeModel را تشکیل می‌دهند.

2- 1- 10- متامدل Decision service

نمودار کلاس Decision Service



کلاس DecisionService برای تعریف سرویس‌های تصمیم‌گیری نامگذاری شده در مدل تصمیم‌گیری موجود در نمونه‌ای از Definitions استفاده می‌شود. DecisionService نوعی عنصر Invocable است به گونه‌ای که نمونه‌های آن DecisionService از مشخصه name و مشخصات اختیاری id، description و label که همگی از نوع Strings هستند و یک مشخصه variable که یک InformationItem است، را به ارث می‌برد. مشخصه id عنصر DecisionService باید در نمونه حاوی Definitions منحصر به فرد باشد. نام مشخصه variable و نام DecisionService باید یکسان باشند. این نام ممکن است برای فراخوانی یک DecisionService از منطق تصمیم‌گیری یک تصمیم‌گیری دیگر یا مدل دانش کسب‌وکار، استفاده شود.

یک عنصر DecisionService دارای یک یا چند outputDecision مرتبط است که نمونه‌هایی از Decision هستند که باید توسط این DecisionService به صورت خروجی، ارائه شوند، یعنی تصمیماتی که سرویس‌تصمیم‌گیری باید آنها را هنگام فراخوانی به شکل نتایج، بازگرداند.

یک عنصر DecisionService دارای صفر یا چند encapsulatedDecisions است، که نمونه‌هایی از Decision هستند که باید توسط این DecisionService کپسوله شوند، یعنی تصمیماتی که باید توسط سرویس‌تصمیم‌گیری در هنگام فراخوانی، ارزیابی شوند.

یک عنصر DecisionService دارای صفر یا چند InputDecisions است، که نمونه‌هایی از Decision هستند که به عنوان ورودی این DecisionService مورد نیاز است، یعنی تصمیم‌گیری‌هایی که نتایج آن هنگام فراخوانی به سرویس‌تصمیم‌گیری، ارائه می‌شود.

یک عنصر DecisionService دارای صفر یا چند InputData است، که نمونه‌هایی از InputData هستند که به عنوان ورودی توسط این DecisionService مورد نیاز است، یعنی داده‌های ورودی که هنگام فراخوانی به سرویس‌تصمیم‌گیری، ارائه می‌شوند.

مشخصات encapsulatedDecisions، inputDecisions و inputData اختیاری هستند. البته حداقل یکی از مشخصات encapsulatedDecisions و inputDecisions باید مشخص شود.

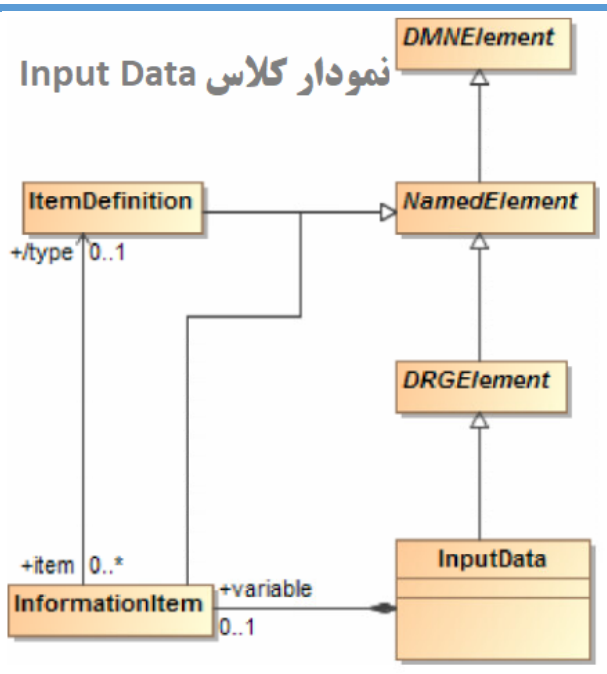
زیرگراف مورد نیاز یک عنصر DecisionService، گراف جهت‌دار متشکل از خود عنصر DecisionService و تجمیع زیرگراف‌های الزامی همه عناصر Decision است که توسط encapsulatedDecisions و outputDecisions آن ارجاع داده می‌شود.

به یک نمونه از DecisionService، یک نم.نه مناسب گفته می‌شود که اگر و تنها در صورتی که زیرگراف مورد نیاز آن acyclic باشد و به خوبی شکل گرفته باشد، یعنی عنصر DecisionService به طور مستقیم یا غیرمستقیم به خود نیاز نداشته باشد.

کلاس DecisionService تمام مشخصات و مدل‌های ارتباطی را از Invocable به ارث می‌برد. جدول زیر مشخصات افزوده شده و ارتباط مدل عنصر DecisionService را نشان می‌دهد.

مشخصه	شرح
outputDecisions: Decision [1..*]	این مشخصه، موارد Decision مورد نیاز برای خروجی توسط این DecisionService را فهرست می‌کند.
encapsulatedDecisions: Decision [0..*]	در صورت وجود این مشخصه، مواردی از Decision را که باید در این DecisionService کپسوله شوند، فهرست می‌کند.
inputDecisions: Decision [0..*]	در صورت وجود این مشخصه، موارد Decision مورد نیاز به عنوان ورودی این DecisionService را فهرست می‌کند.
inputData: InputData [0..*]	در صورت وجود این مشخصه، موارد InputData مورد نیاز به عنوان ورودی توسط این DecisionService را فهرست می‌کند.

2- 1- 11- متامدل Input Data



استاندارد DMN از کلاس InputData برای مدل‌سازی ورودی‌های تصمیم‌گیری، استفاده می‌کند که مقادیر آن خارج از مدل تصمیم‌گیری تعریف شده‌اند.

کلاس InputData یک کلاس خاص شده از DRGElement است و مشخصه name و مشخصات اختیاری id، description و label را از NamedElement به ارث می‌برد. مقدار مشخصه name یک InputData باید با نام هر تصمیم‌گیری، داده ورودی، مدل دانش کسب‌وکار، سرویس‌تصمیم‌گیری یا دریافت، در مدل تصمیم‌گیری متفاوت باشد.

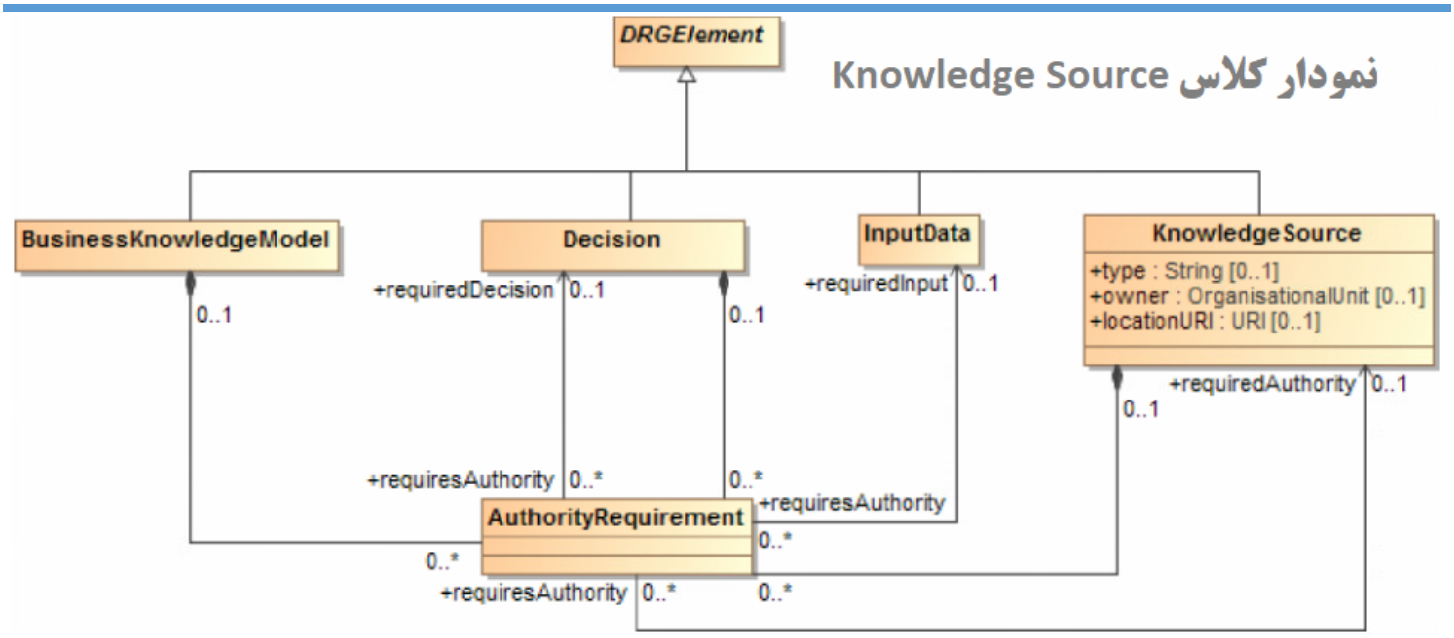
یک نمونه از InputData یک InformationItem را تعریف می‌کند که مقدار آن را ذخیره می‌کند. این InformationItem ممکن است برای مشخص کردن نوع داده ذخیره شده در InputData، شامل typeRef باشد، یا یک ItemDefinition، نوع پایه در expressLanguage را مشخص نماید، یا نوع مورد نظر از خارج، دریافت شود.

در یک DRD، یک نمونه از InputData با یک عنصر داده ورودی نشان داده می‌شود. یک عنصر InputData زیرگراف مورد نیاز ندارد و همیشه به شکل مناسب، شکل گرفته است.

کلاس InputData تمام مشخصات و مدل‌های ارتباطی DRGElement را به ارث می‌برد. جدول زیر مشخصات افزوده شده و ارتباط مدل کلاس InputData را نشان می‌دهد.

مشخصه	شرح
variable: InformationItem	این مشخصه، نمونه‌ای از InformationItem را که نتیجه این InputData را ذخیره می‌کند، نشان می‌دهد.

2- 1- 12- متامدل Knowledge Source



کلاس KnowledgeSource برای مدل‌سازی منابع دانش معتبر در یک مدل تصمیم‌گیری استفاده می‌شود. در یک DRD، یک نمونه از KnowledgeSource با یک عنصر نمودار منبع دانش نشان داده می‌شود.

کلاس KnowledgeSource یک کلاس خاص شده‌ای از کلاس DRGElement است و به تبع آن کلاس خاص شده‌ای از کلاس NamedElement نیز هست که از آن، مشخصه name و مشخصات اختیاری id، description و label را به ارث می‌برد. علاوه بر این، یک KnowledgeSource دارای یک مشخصه locationURI است که یک URI را ذخیره می‌کند. یک مشخصه type دارد که یک مقدار String است و یک owner که نمونه ای از OrganisationalUnit

است. مشخصه type مورد نظر برای شناسایی نوع منبع معتبر طراحی شده است، به عنوان مثال می‌تواند یکی از موارد، سند استراتژی (Policy Document)، مقررات (Regulation)، بینش تحلیلی (Analytic Insight) باشد.

یک عنصر KnowledgeSource نیز از صفر یا چند عنصر authorityRequirement تشکیل شده است که نمونه‌هایی از AuthorityRequirement هستند. کلاس KnowledgeSource تمام مشخصات و مدل‌های ارتباطی DRGElement را به ارث می‌برد. جدول زیر مشخصات و مدل‌های ارتباطی کلاس KnowledgeSource را نشان می‌دهد.

مشخصه	شرح
locationURI: anyURI [0.. 1]	URI که این KnowledgeSource در آن قرار دارد. locationURI باید در قالب URI مشخص شود.
type: string [0..1]	نوع این KnowledgeSource را مشخص می‌کند.
owner: OrganisationalUnit [0..1]	مالک این KnowledgeSource را مشخص می‌کند.
authorityRequirement: AuthorityRequirement [*]	این مشخصه، مواردی از AuthorityRequirement را فهرست می‌کند که به این KnowledgeSource کمک می‌کنند.

2- 1- 13- متامدل Information Requirement

کلاس InformationRequirement برای مدل‌سازی الزام اطلاعات، استفاده می‌شود، این کلاس با یک فلش ساده در یک DRD نشان داده می‌شود. کلاس InformationRequirement یک کلاس خاص شده کلاس DMNElement است که مشخصات اختیاری id، description و label را به ارث می‌برد.

یک عنصر InformationRequirement مؤلفه‌ای از یک عنصر Decision است و آن عنصر Decision را با یک عنصر requiredDecision که نمونه‌ای از Decision است یا یک عنصر requiredInput که نمونه‌ای از InputData است، مرتبط می‌کند.

یک عنصر InformationRequirement به یک نمونه از یک Decision یا InputData اشاره می‌کند که یک مشخصه variable را تعریف می‌کند. آن variable، که نمونه‌ای از InformationItem است، عنصر InformationRequirement را در سطح منطق تصمیم‌گیری نشان می‌دهد.

توجه داشته باشید که یک عنصر InformationRequirement باید به نمونه Decision یا InputData اشاره کند که با عنصر Decision مورد نیاز مرتبط است، نه اینکه حاوی آن باشد. نمونه‌های Decision یا InputData فقط می‌توانند در عناصر Definitions موجود باشند.

به یک نمونه از InformationRequirement، زمانی نمونه مناسب گفته می‌شود که همه موارد زیر به خوبی شکل گرفته باشد:

- به یک عنصر requiredDecision یا requiredInput ارجاع داده شود، اما به هر دو ارجاع داده نشود.
- ارجاع به عنصر requireDecision یا requireInput به خوبی شکل گرفته باشد.
- عنصر Decision که حاوی نمونه InformationRequirement است در زیرگراف InformationRequirement به requiredknowledge ارجاع نشده باشد، اگر این عنصر InformationRequirement به یکی ارجاع دهد.
- عنصر requiredDecision یا requiredInput ارجاع شده در همان مدل تصمیم‌گیری یا در یک مدل تصمیم‌گیری دریافتی، تعریف شده است.

جدول زیر مشخصات و ارتباط مدل عنصر InformationRequirement را نشان می‌دهد.

مشخصه	شرح
requiredDecision: Decision [0..1]	نمونه‌ای از Decision که این InformationRequirement با عنصر Decision حاوی آن مرتبط است.
requiredInput: InputData [0..1]	نمونه ای از InputData که این InformationRequirement با عنصر Decision حاوی آن مرتبط است.

2- 1- 14- متامدل Knowledge Requirement

کلاس KnowledgeRequirement برای مدل‌سازی الزام دانش، استفاده می‌شود، این کلاس با یک فلش خط چین‌دار در یک DRD نشان داده می‌شود. کلاس KnowledgeRequirement یک کلاس خاص شده از کلاس DMNElement است که از آن مشخصات اختیاری id، description و label را به ارث می‌برد.

عنصر KnowledgeRequirement مؤلفه یک عنصر Decision یا یک عنصر BusinessKnowledgeModel است، و آن Decision یا BusinessKnowledgeModel را با عنصر requiredKnowledge که نمونه‌ای از Invocable است، مرتبط می‌کند.

توجه داشته باشید که یک عنصر KnowledgeRequirement باید به نمونه Invocable اشاره کند که با عنصر Decision یا BusinessKnowledgeModel مورد نیاز مرتبط است، نه اینکه حاوی آن باشد: نمونه‌های BusinessKnowledgeModel فقط می‌توانند در عناصر Definitions موجود باشند.

به یک نمونه از KnowledgeRequirement، نمونه مناسب گفته می‌شود، اگر همه موارد زیر به خوبی شکل گرفته باشند:

- به عنصر requiredKnowledge ارجاع داده شده باشد.
- عنصر requiredKnowledge ارجاع شده به خوبی شکل گرفته باشد.
- اگر عنصر KnowledgeRequirement در نمونه‌ای از BusinessKnowledgeModel موجود باشد، آن عنصر BusinessKnowledgeModel در زیرگراف مورد نیاز عنصر requiredKnowledge ارجاع نشده باشد.
- عنصر requiredKnowledge ارجاع شده در همان مدل تصمیم‌گیری یا در یک مدل تصمیم‌گیری دریافتی، تعریف شده است.

جدول زیر مشخصات و ارتباط مدل عنصر KnowledgeRequirement را نشان می‌دهد.

مشخصه	شرح
requiredKnowledge: Invocable	نمونه Invocable که این KnowledgeRequirement با عنصر Decision یا BusinessKnowledgeModel آن مرتبط است.

2- 1- 15- متامدل Authority Requirement

کلاس AuthorityRequirement برای مدل‌سازی یک الزام امکان، استفاده می‌شود، این کلاس با یک خط منقطع که یک طرف آن یک سر دایره‌ای توپر دارد در یک DRD نشان داده می‌شود. کلاس AuthorityRequirement یک کلاس خاص شده از کلاس DMNElement است که از آن مشخصات اختیاری id، description و label را به ارث می‌برد.

عنصر AuthorityRequirement مؤلفه یک عنصر Decision، BusinessKnowledgeModel یا KnowledgeSource است، و این عنصر را به Decision، BusinessKnowledgeModel یا KnowledgeSource مورد نیاز یک عنصر requiredAuthority، که نمونه‌ای از عنصر KnowledgeSource، Decision مورد نیاز است و آن نیز یک عنصر Decision یا requiredInput است، که نمونه ای از InputData است، مرتبط می‌کند.

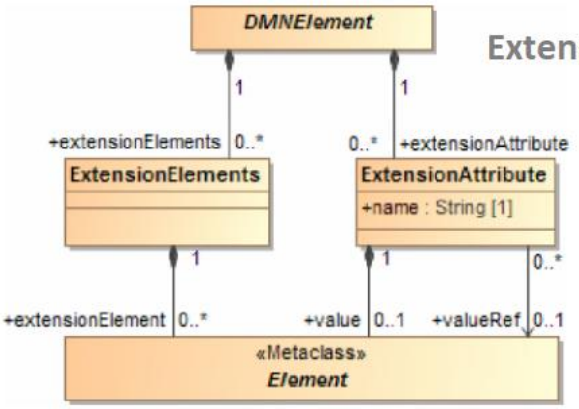
توجه داشته باشید که یک عنصر AuthorityRequirement باید به نمونه KnowledgeSource، Decision یا InputData که با عنصر مورد نیاز مرتبط است ارجاع دهد، نه اینکه حاوی آن باشد. نمونه‌هایی از KnowledgeSource، Decision یا InputData فقط می‌توانند در عناصر Definitions موجود باشند.

جدول زیر مشخصات و ارتباط مدل عنصر AuthorityRequirement را نشان می‌دهد.

مشخصه	شرح
requiredAuthority: KnowledgeSource [0.. 1]	نمونه‌ای از KnowledgeSource که این AuthorityRequirement با عنصر KnowledgeSource، Decision یا BusinessKnowledgeModel آن مرتبط است.
requiredDecision: Decision [0..1]	نمونه‌ای از Decision که این AuthorityRequirement با عنصر KnowledgeSource حاوی آن مرتبط است.

مشخصه	شرح
requiredInput: InputData [0.. 1]	نمونه‌ای از InputData که این AuthorityRequirement با عنصر KnowledgeSource حاوی آن مرتبط است.

2- 1- 16- متامدل Extensibility



نمودار کلاس Extensibility

متامدل DMN توسعه پذیر است. این به پذیرندگان DMN اجازه می‌دهد تا متامدل مشخص شده را به گونه‌ای گسترش دهند که به آنها اجازه دهد همچنان با DMN سازگار باشند. این متامدل، مجموعه‌ای از عناصر افزودنی را فراهم می‌کند که به پذیرندگان DMN اجازه می‌دهد تا مشخصات و عناصر افزودنی خود را به عناصر استاندارد و موجود DMN متصل کنند. این رویکرد منجر به مدل‌های قابل تعویض بیشتری می‌شود، زیرا بعد از هر بار توسعه، عناصر استاندارد هنوز دست نخورده هستند و هنوز می‌توانند توسط سایر پذیرندگان DMN درک شوند. در این حالت، فقط مشخصات و عناصر افزوده شده، هستند که ممکن است در حین مبادله از بین بروند. یک توسعه DMN را می‌توان با استفاده از دو عنصر زیر انجام داد:

1. ExtensionElements
2. ExtensionAttribute

کلاس ExtensionElements محفظه‌ای برای اتصال عناصر دلخواه از متامدل‌های دیگر به هر عنصر DMN است. کلاس ExtensionAttribute به هر توسعه‌ای اجازه می‌دهد نام نیز داشته باشند. این به پذیرندگان DMN اجازه می‌دهد تا هر متامدلی را در متامدل DMN ادغام کنند و از عناصر مدل موجود استفاده مجدد کنند. عنصر ExtensionElements محفظه‌ای برای اتصال عناصر از متامدل‌های دیگر در داخل هر DMNElement می‌باشد.

جدول زیر مشخصات و ارتباط مدل را برای عنصر ExtensionElements نشان می‌دهد.

مشخصه	شرح
extensionElement: Element [0..*]	عنصر موجود این ارتباط، زمانی که از تبادل شمای XML استفاده می‌شود، قابل اجرا نیست، زیرا مکانیسم XSD برای پشتیبانی از "هر" عنصر از فضای‌نام دیگر، از قبل، این نیاز را برآورده می‌کند.

عنصر ExtensionAttribute حاوی یک عنصر یا ارجاع به یک عنصر از متامدل دیگری است. یک ExtensionAttribute همچنین دارای نامی برای تعریف نقش یا هدف عنصر مرتبط است. این نوع در هنگام استفاده از تبادل شمای XML قابل استفاده نیست، زیرا مکانیسم XSD برای پشتیبانی از "هر مشخصه" از فضاهای‌نام دیگر این نیاز را برآورده می‌کند. جدول زیر ارتباط مدل را برای عنصر ExtensionAttribute نشان می‌دهد.

مشخصه	شرح
name: string	نام مشخصه توسعه می‌باشد.
value: Element [0..1]	معرف Element مشمول شده، می‌باشد. این مشخصه نباید همراه با مشخصه valueRef استفاده شود.
valueRef: Element [0..1]	معرف ارجاع به Element مرتبط شده، می‌باشد. این مشخصه نباید همراه با مشخصه value استفاده شود.

3- ارتباط منطق تصمیم‌گیری با الزامات تصمیم‌گیری

در بخش‌های قبل توضیح داده شد که چگونه سطح الزامات تصمیم‌گیری یک مدل تصمیم‌گیری (یک DRG که در یک یا چند DRD نشان داده شده است) ممکن است برای مدل‌سازی ساختار یک حوزه تصمیم‌گیری استفاده شود. با این حال، جزئیات نحوه استخراج نتیجه هر تصمیم‌گیری از ورودی‌های آن باید در سطح منطق تصمیم‌گیری مدل شود. در این بخش، اصولی معرفی می‌شود که از طریق آنها می‌توان منطق تصمیم‌گیری را با عناصر موجود در DRG مرتبط نمود. سپس نماهای خاص منطق تصمیم‌گیری (جدول تصمیم‌گیری و عبارات FEEL) در بخش‌های بعدی، تعریف می‌شوند.

سطح منطق تصمیم‌گیری یک مدل تصمیم‌گیری در DMN از یک یا چند عبارت تشکیل شده است. عناصر منطق تصمیم‌گیری که به عنوان عبارات مدل‌سازی می‌شوند شامل عبارات جدول‌بندی شده مانند جداول تصمیم‌گیری و فراخوان‌ها و عبارات تحت اللفظی (متن) مانند Age > 30 می‌شوند.

- **literal expression**: یک عبارت تحت اللفظی، منطق تصمیم‌گیری را به شکل متنی که توضیح می‌دهد چگونه یک مقدار خروجی از مقادیر ورودی آن مشتق می‌شود، نشان می‌دهد. زبان عبارت مورد نظر ممکن است حالت رسمی (formal) یا قابل اجرا (executable) داشته باشد. نمونه‌هایی از عبارات تحت اللفظی، می‌تواند شامل توصیف انگلیسی ساده از منطق یک تصمیم‌گیری، یک پیشنهاد منطقی سطح اول، یک برنامه کامپیوتری جاوا و یا یک سند PMML باشد. در همین مقاله در ادامه یک زبان عبارت قابل اجرا به نام FEEL معرفی شده است. در بخش دیگری زیر مجموعه‌ای از زبان FEEL یا همان S-FEEL معرفی می‌شود که زبان پیش‌فرض برای عبارات تحت اللفظی در جداول تصمیم‌گیری DMN است.
- **decision table**: یک جدول تصمیم‌گیری، یک نمای جدولی از منطق تصمیم‌گیری است که بر پایه گسسته‌سازی مقادیر احتمالی ورودی‌های یک تصمیم‌گیری بنا شده و قواعدی را سازماندهی می‌نماید که مقادیر ورودی گسسته را به مقادیر خروجی گسسته نگاشت می‌کند.
- **invocation**: یک فراخوانی، یک نمای جدولی از نحوه استفاده از منطق تصمیم‌گیری است که توسط یک مدل دانش کسب‌وکار یا یک سرویس‌تصمیم‌گیری، توسط یک تصمیم‌گیری یا توسط یک مدل دانش کسب‌وکار دیگر، ارائه می‌شود. یک فراخوان ممکن است به عنوان یک عبارت تحت اللفظی نیز نمایش داده شود، اما معمولاً نمایش جدولی قابل درک‌تر است.

نماهای جدولی منطق تصمیم‌گیری در بقیه این مشخصات کادرهای عبارت، نامیده می‌شوند. هر سه سطح انطباق DMN شامل تمام عبارات فوق است. در سطح 1 انطباق DMN، عبارات تحت اللفظی، تفسیر نمی‌شوند و بنابراین استفاده از آنها آزاد است. در سطح 2 انطباق DMN، عبارات تحت اللفظی به زبان S-FEEL محدود می‌شوند.

منطق تصمیم‌گیری با گنجاندن یک مؤلفه عبارت ارزش در برخی از عناصر مدل تصمیم‌گیری در DRG به یک مدل تصمیم‌گیری اضافه می‌شود:

- از دیدگاه منطق تصمیم‌گیری، یک تصمیم‌گیری بخشی از منطق است که بر اساس داده‌های ورودی، چگونگی پاسخ به یک سوال معین را تعریف می‌کند. در نتیجه، هر عنصر تصمیم‌گیری در یک مدل تصمیم‌گیری ممکن است شامل یک عبارت ارزشی باشد که توضیح می‌دهد چگونه یک نتیجه تصمیم‌گیری از ورودی مورد نیاز آن مشتق می‌شود و احتمالاً یک مدل دانش کسب‌وکار را فرا می‌خواند.
- از دیدگاه منطق تصمیم‌گیری، مدل دانش کسب‌وکار بخشی از منطق تصمیم‌گیری است که به عنوان تابعی تعریف می‌شود که امکان استفاده مجدد از آن در تصمیم‌گیری‌های متعدد را فراهم می‌کند. در نتیجه، هر عنصر مدل دانش کسب‌وکار ممکن است شامل یک بیان ارزش باشد که بدنه آن تابع است.

یکی دیگر از اجزای کلیدی سطح منطق تصمیم‌گیری، متغیر است. متغیرها برای ذخیره مقادیر تصمیم‌گیری‌ها و داده‌های ورودی برای استفاده در عبارات ارزش استفاده می‌شوند. InformationRequirements متغیرهایی را در محدوده قابل ارجاع به آن تصمیم‌گیری‌ها و داده‌های ورودی مشخص می‌کنند، به طوری که عبارات ارزش ممکن است به این متغیرها اشاره کنند. متغیرها الزامات اطلاعاتی در DRG را به عبارات ارزش در سطح منطق تصمیم‌گیری پیوند می‌دهند:

- از دیدگاه منطق تصمیم‌گیری، یک الزام اطلاعاتی یک نیاز برای یک مقدار ارائه شده خارجی است که به یک متغیر آزاد در منطق تصمیم‌گیری تخصیص داده شود، تا بتوان یک تصمیم‌گیری را ارزیابی کرد. در نتیجه، هر الزام اطلاعاتی در یک مدل تصمیم‌گیری به یک تصمیم‌گیری یا داده ورودی اشاره می‌کند، که به نوبه خود متغیری را تعریف می‌کند که ورودی داده مرتبط را در عبارت تصمیم‌گیری نشان می‌دهد.
- متغیرهایی که در بدنه تابع تعریف شده توسط یک عنصر مدل دانش کسب‌وکار در DRG استفاده می‌شوند باید در هر یک از تصمیم‌گیری‌های مورد نیاز، به منابع اطلاعاتی، مرتبط باشند. در نتیجه، هر مدل دانش کسب‌وکار شامل صفر یا چند متغیر است که پارامترهای تابع محسوب می‌شوند.

سومین عنصر کلیدی سطح منطق تصمیم‌گیری، تعاریف آیتم است که انواع و ساختار اقلام داده را در یک مدل تصمیم‌گیری توصیف می‌کند. عناصر داده ورودی در DRG، و متغیرها و عبارات ارزش در سطح منطق تصمیم‌گیری، ممکن است به یک تعریف آیتم مرتبط که نوع و ساختار داده‌های مورد انتظار را به عنوان ورودی، اختصاص داده شده به متغیر یا ناشی از ارزیابی عبارت، توصیف می‌کند، اشاره کنند.

توجه داشته باشید که منابع دانش در سطح منطق تصمیم‌گیری نشان داده نمی‌شوند. منابع دانش بخشی از مستندات منطق تصمیم‌گیری هستند، نه خود منطق تصمیم‌گیری.

وابستگی بین تصمیم‌گیری‌ها، منابع اطلاعات الزامی و مدل‌های دانش کسب‌وکار، همانطور که توسط اطلاعات و نیازهای دانش در یک DRG نشان داده می‌شود، نحوه ارتباط عبارات ارزش مرتبط با این عناصر را با یکدیگر محدود می‌کند.

همانطور که در بالا توضیح داده شد، هر تصمیم‌گیری، داده‌های ورودی، و مدل دانش کسب‌وکاری در سطح DRG با یک متغیر مورد استفاده در سطح منطق تصمیم‌گیری، مرتبط است. هر متغیری که در بیان یک تصمیم‌گیری ارجاع داده می‌شود باید با یک تصمیم‌گیری مورد نیاز، داده‌های ورودی مورد نیاز یا دانش مورد نیاز مرتبط باشد. همچنین هر متغیر مرتبط با تصمیم‌گیری‌های مورد نیاز، داده‌های ورودی مورد نیاز و دانش مورد نیاز باید در عبارت تصمیم‌گیری، ارجاع داده شود.

- اگر تصمیم‌گیری مستلزم تصمیم‌گیری دیگری باشد، عبارت ارزش تصمیم‌گیری مورد نیاز، ارزش مورد نظر را برای استفاده در ارزیابی تصمیم‌گیری مورد نیاز به متغیر مورد نظر، اختصاص می‌دهد. این مکانیزم عمومی در DMN برای تصمیم‌گیری در سطح منطق تصمیم‌گیری است.
- اگر تصمیم‌گیری نیاز به داده ورودی داشته باشد، در زمان اجرا مقدار متغیر به مقدار منبع داده متصل به داده ورودی اختصاص داده می‌شود. این مکانیزم عمومی در DMN برای نمونه سازی نیازهای داده برای یک تصمیم‌گیری است.

متغیرهای ورودی منطق تصمیم‌گیری یک تصمیم‌گیری نباید خارج از آن عبارت ارزش یا عبارات ارزش مؤلفه آن استفاده شوند. عنصر تصمیم‌گیری، محدوده واژگانی متغیرهای ورودی را برای منطق تصمیم‌گیری آن تعریف می‌کند. برای جلوگیری از برخورد نام و ابهام، نام یک متغیر باید در محدوده مورد نظر، منحصر به فرد باشد. هنگامی که عناصر DRG به FEEL نگاشت می‌شوند، نام یک متغیر با نام (احتمالاً واجد شرایط) داده ورودی یا تصمیم‌گیری مرتبط با آن یکسان است که منحصر به فرد بودن آن را تضمین می‌کند.

هنگامی که عناصر DRG به FEEL نگاشت می‌شوند، تمام تصمیم‌گیری‌ها و داده‌های ورودی در یک DRG یک محتوا را تعریف می‌کنند، که عبارت تحت اللفظی است که منطق مرتبط با عنصر تصمیم‌گیری را نشان می‌دهد و آن محدوده را نشان می‌دهد.

عناصر الزام اطلاعاتی در یک تصمیم‌گیری، ورودی‌های محتوا در محتوای مرتبط هستند، که در آن کلید، نام متغیری است که الزام اطلاعاتی تعریف می‌کند، و عبارت، محتوایی است که با تصمیم‌گیری مورد نیاز یا عنصر داده ورودی مرتبط است. عبارت ارزشی که به واسطه یک منطق تصمیم‌گیری به یک تصمیم‌گیری وابسته شده است در محتوای ورودی، یک عبارت یا expression نامیده می‌شود که مشخص کننده نتیجه محتوای مورد نظر است.

به همین ترتیب، یک عنصر مدل دانش کسب‌وکار، دامنه واژگانی پارامترهای خود، یعنی متغیرهای ورودی بدنه خود را تعریف می‌کند. در زبان FEEL، بیان تحت اللفظی و ساختار محدوده‌ای که منطق مرتبط با عنصر مدل دانش کسب‌وکار را نشان می‌دهد، یک تعریف تابع است، که در آن پارامترهای رسمی نام پارامترهای عنصر مدل دانش کسب‌وکار هستند و عبارت، عبارت ارزشی است که در بدنه عنصر مدل دانش کسب‌وکار است.

اگر یک عنصر مدل دانش کسب‌وکار به یک یا چند مدل دانش کسب‌وکار دیگر نیاز دارد، باید یک عبارت ارزش صریح داشته باشد که نحوه استفاده از مدل‌های دانش کسب‌وکار مورد نیاز و ترکیب نتایج آنها یا توضیح دیگری را توضیح دهد.

در سطح منطق تصمیم‌گیری، یک تصمیم‌گیری با ارزیابی عبارت ارزش مدل دانش کسب‌وکار با پارامترهای محدود شده به مقدار ورودی خود، یک مدل دانش کسب‌وکار مورد نیاز را فراخوانی می‌کند. چگونگی دستیابی به این امر بستگی به نحوه تقسیم منطق تصمیم‌گیری بین تصمیم‌گیری و مدل‌های دانش کسب‌وکار دارد:

- اگر یک عنصر تصمیم‌گیری به بیش از یک عنصر دانش کسب‌وکار نیاز دارد، عبارت ارزش آن باید یک عبارت تحت اللفظی باشد که مشخص می‌کند چگونه عناصر مدل دانش کسب‌وکار فراخوانی می‌شوند و چگونه نتایج آنها در نتیجه تصمیم‌گیری ترکیب می‌شوند.

- اگر یک تصمیم‌گیری به هیچ مدل دانش کسب‌وکاری نیاز نداشته باشد، عبارت ارزش آن باید یک عبارت تحت اللفظی یا جدول تصمیم‌گیری باشد که کل منطق تصمیم‌گیری را برای استخراج خروجی از ورودی‌ها مشخص می‌کند.
- به طور مشابه، اگر یک عنصر تصمیم‌گیری فقط به یک عنصر مدل دانش کسب‌وکار نیاز داشته باشد، اما منطق تصمیم‌گیری بر منطق مدل دانش کسب‌وکار مورد نیاز آن توضیح دهد، عنصر تصمیم‌گیری باید یک عبارت تحت اللفظی داشته باشد که نگارش عبارت ارزش مدل دانش کسب‌وکار که بیان می‌کند چگونه نتیجه آن برای ارائه نتیجه تصمیم‌گیری مورد استفاده قرار می‌گیرد را مشخص کند.
- در تمام موارد دیگر (یعنی زمانی که یک تصمیم‌گیری دقیقاً به یک مدل دانش کسب‌وکار نیاز دارد و منطق را توضیح نمی‌دهد)، عبارت ارزش یک عنصر تصمیم‌گیری ممکن است عبارت ارزشی از نوع فراخوانی باشد. در یک عبارت ارزش از نوع فراخوانی، فقط اتصالات پارامترهای مدل دانش کسب‌وکار به داده‌های ورودی تصمیم‌گیری‌ها باید مشخص شود. نتیجه تصمیم‌گیری، نتیجه‌ای است که توسط عبارت ارزش مدل دانش کسب‌وکار برای مقادیر ارسال شده به پارامترهای آن بازگردانده می‌شود.

پارامترهای باند شده به یک مدل دانش کسب‌وکار، یک عبارت ارزشی هستند که مشخص می‌کند چگونه مقدار ارسال شده به آن پارامتر از مقادیر متغیرهای ورودی تصمیم‌گیری فراخوانی شده، مشتق می‌شوند.

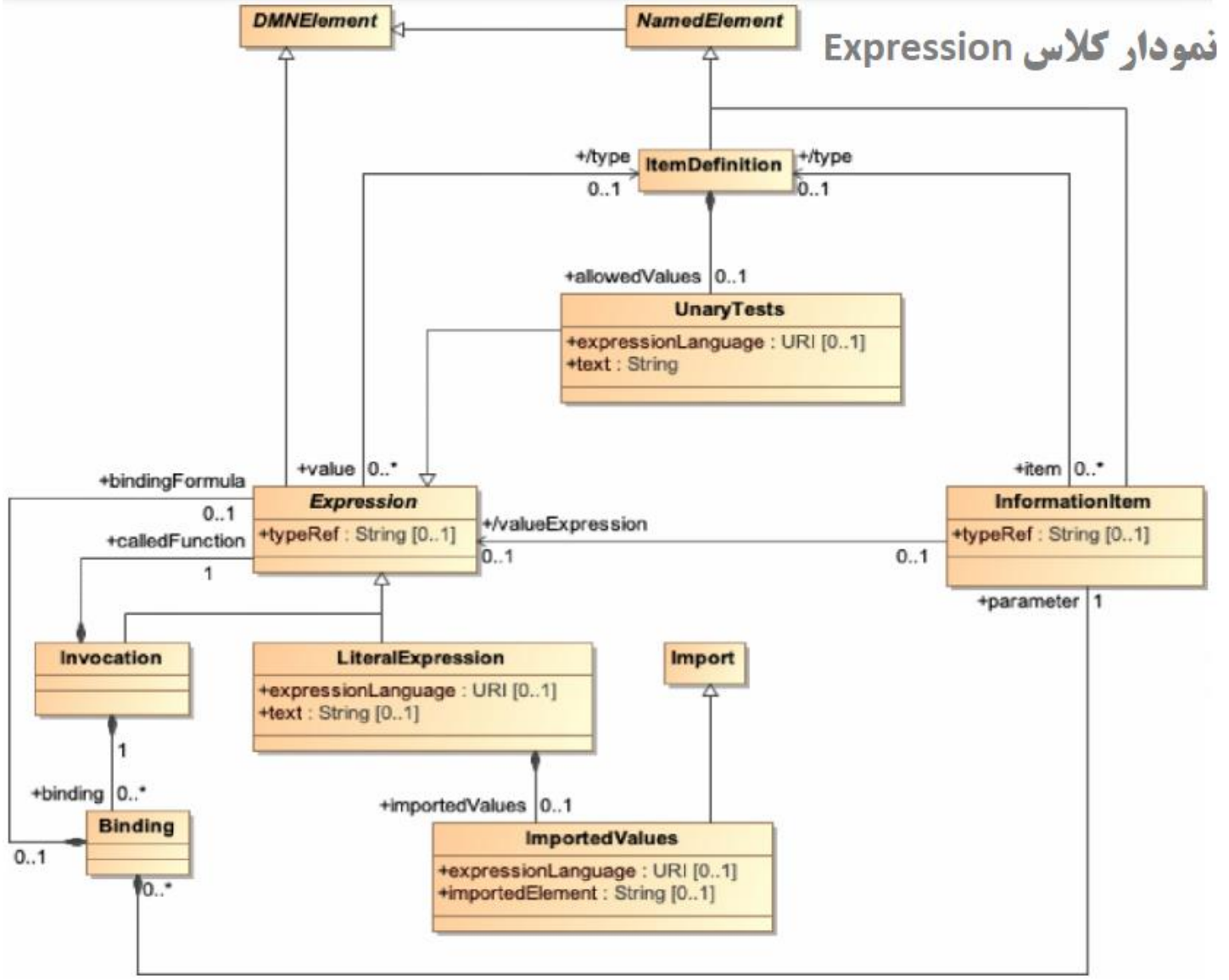
3- 1- متامدل

یکی از مشخصات مهم تصمیم‌گیری‌ها و مدل‌های دانش کسب‌وکار این است که ممکن است حاوی عبارتی باشند که منطقی را که براساس آن یک تصمیم‌گیری مدل‌سازی شده اتخاذ می‌شود، یا قطعاتی از آن منطق را توصیف می‌کند.

کلاس Expression ابرکلاس انتزاعی برای تمام عباراتی است که برای توصیف کامل یا بخش‌هایی از منطق تصمیم‌گیری در مدل‌های DMN استفاده می‌شود و هنگام تفسیر، یک مقدار واحد را برمی‌گرداند. در اینجا "مقدار واحد" احتمالاً شامل داده‌های ساختاریافته است، مانند جدول تصمیم‌گیری با چند بند خروجی. DMN سه نوع مشخص عبارت را تعریف می‌کند:

- LiteralExpression
- DecisionTable
- Invocation

یک عبارت ممکن است به متغیرهایی ارجاع دهد، به طوری که مقدار عبارت، هنگام تفسیر، به مقادیر اختصاص داده شده به متغیرهای ارجاع شده بستگی داشته باشد. کلاس InformationItem برای مدل‌سازی متغیرها در عبارات استفاده می‌شود. مقدار یک عبارت، مانند مقدار اختصاص داده شده به یک متغیر، ممکن است ساختار و محدوده‌ای از مقادیر مجاز داشته باشد. کلاس ItemDefinition برای مدل‌سازی ساختارهای داده و محدوده استفاده می‌شود.



یکی از خصوصیات مهم تصمیم‌گیری‌ها و مدل‌های دانش کسب‌وکار این است که ممکن است حاوی عبارتی باشند که یک منطق تصمیم‌گیری را مدل‌سازی کنند، یا قطعاتی از آن منطق را توصیف کنند.

کلاس Expression یک کلاس خاص شده انتزاعی از کلاس DMNElement است که از آن مشخصات اختیاری id، description و label را به ارث می‌برد. یک نمونه از Expression، مؤلفه یک عنصر Decision، یک عنصر BusinessKnowledgeModel، یا یک عنصر ItemDefinition است، یا مؤلفه‌ای از نمونه دیگری از Expression است، به طور مستقیم یا غیرمستقیم.

یک Expression به طور ضمنی به صفر یا چند متغیر با استفاده از نام آنها در متن عبارت خود ارجاع می‌دهد. این متغیرها، که نمونه‌هایی از InformationItem هستند، بسته به نوع Expression، از نظر واژگانی دارای دامنه هستند. اگر Expression منطق یک Decision باشد، دامنه شامل الزامات آن Decision است. اگر Expression بدنه encapsulatedLogic یک BusinessKnowledgeModel باشد، دامنه شامل پارامترهای FunctionDefinition و الزامات BusinessKnowledgeModel است. اگر Expression مقدار ContextEntry باشد، دامنه شامل ورودی‌های قبلی در Context می‌شود. نمونه‌ای از Expression به typeRef اختیاری ارجاع می‌دهد که به نوع پایه در typeLanguage پیش‌فرض، نوع سفارشی مشخص شده توسط ItemDefinition یا نوع

دریافتی اشاره می‌کند. نوع ارجاع شده محدوده مقادیر ممکن Expression را مشخص می‌کند. اگر نمونه‌ای از Expression که خروجی عنصر Decision را تعریف می‌کند شامل typeRef باشد، نوع ارجاع شده باید با نوع عنصر Decision حاوی آن یکسان باشد.

یک نمونه از Expression را می‌توان به گونه‌ای تفسیر کرد که یک مقدار واحد را از مقادیر اختصاص داده شده به متغیرهای آن به دست آورد. اینکه چگونه مقدار یک عنصر Expression از مقادیر تخصیص داده شده به متغیرهای آن مشتق می‌شود، به نوع Expression بستگی دارد. عنصر ItemDefinition، NamedElement را تخصصی می‌کند و مشخصات و مدل‌های ارتباطی آن را به ارث می‌برد. در ادامه مشخصات افزوده شده و ارتباط مدل عنصر ItemDefinition را مشاهده خواهید نمود.

کلاس Expression مشخصات و مدل‌های ارتباطی DMNElement را به ارث می‌برد.

3- 1- 2- متامدل UnaryTests

کلاس UnaryTests برای مدل‌سازی یک آزمون منطقی (boolean) استفاده می‌شود که در آن آرگومان مورد آزمایش به صورت ضمنی یا با علامت {?} نشان داده می‌شود و مقدار آن توسط متن در برخی از زبان‌های عبارت مشخص شده، مشخص می‌شود.

کلاس UnaryTests یک زیرکلاس مشخص از کلاس Expression است. نمونه‌ای از UnaryTests یک typeRef اختیاری را از Expression به ارث می‌برد که نباید استفاده شود. یک نمونه از UnaryTests همچنین دارای یک text اختیاری است که یک String است و یک expressionLanguage اختیاری که String است و زبان متن عبارت را مشخص می‌کند. اگر هیچ expressionLanguage مشخص نشده باشد، زبان متن عبارت، expressionLanguage است که با نمونه حاوی Definitions مرتبط است. expressionLanguage باید در قالب URI مشخص شود. زبان پیش‌فرض عبارت‌ها، FEEL است. وقتی زبان عبارت، FEEL باشد، متن باید با قواعد 15 گانه گرامری مطابقت داشته باشد. جدول زیر مشخصات افزوده شده و ارتباط مدل عنصر UnaryTests را نشان می‌دهد.

مشخصه	شرح
text: string[0..1]	این مشخصه، متن UnaryTests را معرفی می‌کند. این متن باید یک عبارت معتبر در expressLanguage باشد.
expressionLanguage: anyURI[0..1]	این مشخصه، زبان عبارت مورد استفاده در متن UnaryTests را مشخص می‌کند. این مقدار زبان عبارت مشخص شده برای نمونه حاوی DecisionRequirementDiagram را بازنویسی می‌کند. زبان باید در قالب URI مشخص شود.

3- 1- 3- متامدل ItemDefinition

ورودی و خروجی تصمیم‌گیری‌ها، مدل‌های‌دانش‌کسب‌وکار و سرویس‌تصمیم‌گیری، و خروجی داده ورودی هر DRGElements، اقلام داده‌ای هستند که مقدار آنها، در سطح منطق مورد نظر، به متغیرها تخصیص داده می‌شوند یا با Expressions نشان داده می‌شوند. یکی از خصوصیات مهم اقلام داده در مدل‌های تصمیم‌گیری، ساختار آنهاست. DMN به فرمت خاصی برای این ساختار داده نیاز ندارد، اما زیر مجموعه‌ای از فرمت‌های FEEL را به عنوان پیش‌فرض خود تعیین می‌کند. کلاس ItemDefinition برای مدل‌سازی ساختار و محدوده مقادیر ورودی و نتیجه تصمیم‌گیری‌ها، استفاده می‌شود.

کلاس ItemDefinition کلاس تخصصی از کلاس NamedElement است. یک نمونه از ItemDefinition دارای یک name و یک id و description اختیاری است. نام یک عنصر ItemDefinition باید از نام دیگر ItemDefinition ها و دریافت‌های درون همان مدل متمایز باشد.

زبان نوع پیش‌فرض برای همه عناصر را می‌توان در عنصر Definitions با استفاده از مشخصه typeLanguage، مشخص کرد. به عنوان مثال، مقدار typeLanguage زیر نشان می‌دهد که ساختارهای داده مورد استفاده توسط عناصر در آن Definitions به شکل انواع شمای XML هستند. اگر مشخص نشده باشد، پیش فرض FEEL است.

http://www.w3.org/2001/XMLSchema

توجه داشته باشید که انواع داده‌هایی که در typeLanguage تعبیه شده‌اند که با نمونه‌ای از Definitions مرتبط است، نیازی به تعریف مجدد توسط عناصر ItemDefinition موجود در آن عنصر Definitions ندارند. آنها دریافت شده؛ در نظر گرفته می‌شوند و می‌توانند در عناصر DMN در عنصر Definitions ارجاع داده شوند. زبان نوع را می‌توان به صورت محلی با استفاده از مشخصه typeLanguage در عنصر ItemDefinition بازنویسی کرد.

همچنین توجه داشته باشید که انواع داده‌ها و ساختارهایی که با استفاده از عنصر Import دریافت می‌شوند، نیازی به تعریف مجدد توسط عناصر ItemDefinition موجود در آن عنصر Definitions ندارند. این انواع داده و ساختارهای دریافت شده بدون بازنویسی در نظر گرفته می‌شوند و می‌توانند در عناصر DMN در عنصر Definitions ارجاع داده شوند.

یک عنصر ItemDefinition ممکن است یک typeRef داشته باشد، که رشته‌ای است که به شکل یک نام واجد شرایط به یک ItemDefinition در نمونه فعلی Definitions یا یک نوع داخلی در typeLanguage مشخص شده یا یک نوع تعریف شده در یک DMN، XSD دریافت شده، ارجاع می‌دهد. یا در مورد دوم، سند خارجی باید در عنصر Definitions وارد شود که حاوی نمونه ItemDefinition است، با استفاده از عنصر Import که هم مقدار فضای_نام و هم نام آن را در هنگام استفاده از یک واجد شرایط مشخص می‌کند. به عنوان مثال، در مورد ساختارهای داده ارائه شده توسط یک طرح XML، یک Import برای تعیین محل فایل آن طرح و مشخصه typeRef به تعریف نوع یا عنصر در طرحواره دریافت شده، اشاره می‌کند. اگر typeLanguage، زبان FEEL باشد، انواع داخلی، همان انواع داده‌های داخلی زبان FEEL یعنی number، string، boolean، days and time duration، years و months duration، date، time، date and time و Any هستند. یک typeRef که به یک نوع داخلی ارجاع می‌دهد باید پیشوند را حذف کند.

یک عنصر ItemDefinition ممکن است مقادیر مجاز از typeRef را با استفاده از مشخصه allowValues محدود کند. allowValues نمونه ای از unaryTests است که مقادیر مجاز یا محدوده مقادیر مجاز را در دامنه typeRef مشخص می‌کند. نوع مجاز باید با عنصر ItemDefinition حاوی آن مطابقت داشته باشد. اگر یک عنصر ItemDefinition شامل یک یا چند مقدار مجاز باشد، allowValues محدوده کامل مقادیری را که این ItemDefinition نشان می‌دهد، مشخص می‌کند. اگر عنصر ItemDefinition حاوی allowValues نباشد، محدوده مقادیر مجاز آن، محدوده کامل typeRef مرجع است. در مواردی که یک عنصر ItemDefinition نشان‌دهنده مجموعه‌ای از مقادیر در محدوده مجاز است، تعدد را می‌توان در مشخصه isCollection نمایش داد. مقدار پیش فرض این مشخصه، false است.

یک راه جایگزین برای تعریف یک نمونه از ItemDefinition، ترکیبی از عناصر ItemDefinition است. یک نمونه از ItemDefinition ممکن است حاوی صفر یا چند itemComponent باشند که خود ItemDefinitions هستند. هر itemComponent به نوبه خود ممکن است با یک typeRef و allowValues یا یک itemComponent تودرتو تعریف شود. به این ترتیب، انواع پیچیده ممکن است در DMN تعریف شوند. نام یک itemComponent (تعریف itemComponent تودرتو) باید در داخل ItemDefinition یا itemComponent آن منحصر به فرد باشد.

یک راه جایگزین برای تعریف یک نمونه از ItemDefinition با تعیین یک عنصر FunctionItem است که امضای یک تابع را که شامل پارامترها و خروجی تابع است، تعریف می‌کند. یک نمونه از ItemDefinition ممکن است حداکثر دارای یک FunctionItem باشد. یک FunctionItem ممکن است حاوی صفر یا چند پارامتر باشد که به عنوان InformationItems و یک نوع خروجی تعریف شده به عنوان typeRef تعریف شده است. نام پارامترهای یک FunctionItem منحصر به فرد است. یک عنصر ItemDefinition باید تنها با استفاده از یکی از راه‌های جایگزین زیر تعریف شود:

- ارجاع به یک typeRef داخلی یا دریافتی که احتمالاً با allowValues محدود شده است.
- ترکیبی از عناصر ItemDefinition
- با عنصر امضای تابع

عنصر ItemDefinition نوعی از NamedElement را تخصصی می‌کند و مشخصات و مدل‌های ارتباطی آن را به ارث می‌برد. جدول زیر مشخصات افزوده شده و ارتباط مدل عنصر ItemDefinition را نشان می‌دهد.

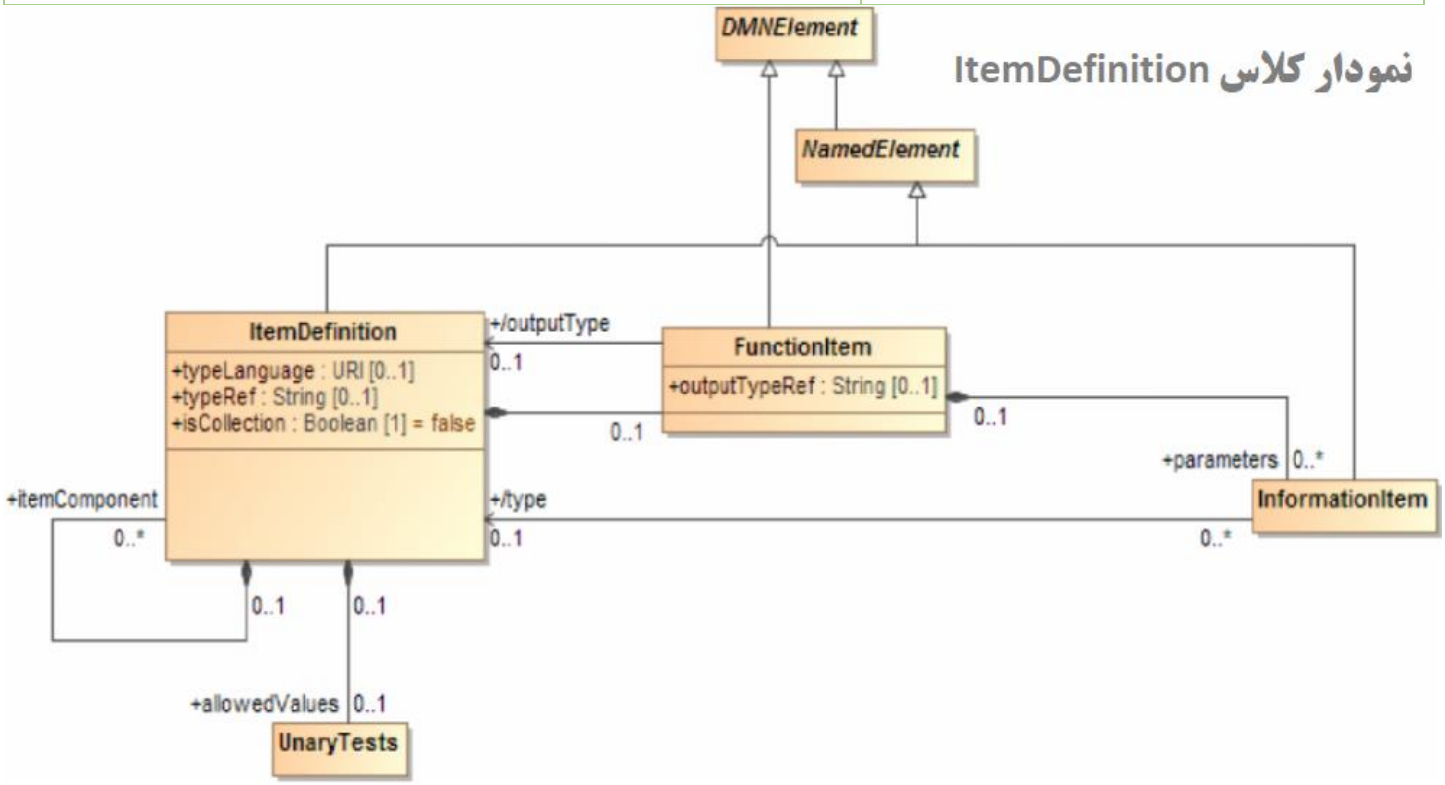
مشخصه	شرح
typeRef: String [1]	این ویژگی با نام پیشوند_فضای_نام نوع پایه این ItemDefinition را مشخص می‌کند.

مشخصه	شرح
typeLanguage: String [0..1]	این مشخصه زبان نوع مورد استفاده برای تعیین نوع پایه این ItemDefinition را مشخص می‌کند. این مقدار، مقدار زبان نوع مشخص شده در عنصر Definitions را بازنویسی می‌کند. زبان باید در قالب URI مشخص شود.
allowedValues: UnaryTests [0..1]	این مشخصه مقادیر ممکن یا محدوده مقادیر در نوع پایه را که در این ItemDefinition مجاز است فهرست می‌کند.
itemComponent: ItemDefinition [*]	این مشخصه صفر یا چند مورد ItemDefinition تودرتو تعریف می‌کند که این ItemDefinition را تشکیل می‌دهد.
IsCollection: Boolean	تنظیم این فلگ یا نشانه روی true نشان می‌دهد که مقادیر واقعی تعریف شده توسط این ItemDefinition مجموعه‌ای از مقادیر مجاز هستند. البته حالت پیش‌فرض آن false است.
functionItem: FunctionItem [0..1]	این مشخصه یک FunctionItem اختیاری را توصیف می‌کند که این ItemDefinition را تشکیل می‌دهد.

جدول زیر مشخصات مدل عنصر FunctionItem را نشان می‌دهد.

مشخصه	شرح
outputTypeRef: String [0..1]	ارجاع به نوع خروجی تابع
parameters: InformationItem [0. .*]	پارامترهای تابع به شکل InformationItem ها

نمودار کلاس ItemDefinition



3- 1- 4- متامدل InformationItem

کلاس InformationItem برای مدل‌سازی متغیرها در سطح منطق‌تصمیم‌گیری در مدل‌های تصمیم‌گیری استفاده می‌شود. InformationItem یک زیر کلاس مشخص از NamedElement است که از آن مشخصات id و name اختیاری، description و label را به ارث می‌برد، با این تفاوت که یک عنصر InformationItem باید یک مشخصه name داشته باشد، که نامی است که برای نشان دادن آن در سایر عناصر Expression استفاده می‌شود. نام یک عنصر InformationItem باید در محدوده خود منحصر به فرد باشد.

متغیرهایی که مقادیر حاصل از یک تصمیم‌گیری را نشان می‌دهند، توسط یک منبع داده خارجی به داده‌های ورودی اختصاص می‌یابند، یا به یک ماژول منطق‌تصمیم‌گیری که به عنوان یک تابع تعریف می‌شود، (و توسط یک عنصر مدل‌دانش‌کسب‌وکار معرفی می‌شود) منتقل می‌شوند. در حالت اول یا دوم، یک متغیر ممکن است توسط تصمیم‌گیری‌های وابسته دیگر با استفاده از الزامات اطلاعاتی آنها ارجاع داده شود. در حالت سوم، متغیر یکی از پارامترهای تابعی است که تحقق عنصر مدل‌دانش‌کسب‌وکار در سطح منطق‌تصمیم‌گیری است.

متغیری که نمونه‌ای از Decision یا InputData ارجاع‌شده توسط InformationRequirement را نشان می‌دهد باید با بیان مقدار منطق‌تصمیم‌گیری در عنصر Decision که حاوی عنصر InformationRequirement است، ارجاع داده شود. یک پارامتر در یک نمونه از BusinessKnowledgeModel باید متغیری در بیان مقدار آن عنصر BusinessKnowledgeModel باشد.

یک عنصر InformationItem موجود در یک Decision، مقدار عبارت Decision را به آن اختصاص می‌دهد.

- یک عنصر InformationItem که یک پارامتر در یک FunctionDefinition است، یک مقدار توسط یک عنصر Binding به عنوان بخشی از یک نمونه از Invocation اختصاص داده می‌شود.
- یک عنصر InformationItem موجود در یک InputData توسط یک منبع داده خارجی که در زمان اجرا متصل شده است، یک مقدار اختصاص داده می‌شود.
- یک عنصر InformationItem موجود در یک ContextEntry یک مقدار توسط عبارت مقدار ContextEntry اختصاص می‌یابد.

در هر صورت، نوع داده‌ای که با typeRef نشان داده شده است که با نمونه‌ای از InformationItem مرتبط است، باید با نوع داده‌ای که با عنصر مدل DMN که مقدار خود را از آن گرفته است، سازگار باشد. InformationItem تمام مشخصات و مدل‌های ارتباطی NamedElement را به ارث می‌برد. جدول زیر مشخصات افزوده شده و ارتباط مدل عنصر InformationItem را نشان می‌دهد.

مشخصه	شرح
/valueExpression: Expression [0..1]	عبارتی که مقدار آن به این InformationItem اختصاص داده شده است. این یک مشخصه مشتق شده است.
typeRef: String [1]	نام واجد شرایط نوع این InformationItem.

3- 1- 5- متامدل Literal expression

کلاس LiteralExpression برای مدل‌سازی یک مقدار عبارت که مقدار آن توسط متن در برخی از زبان‌های عبارت، مشخص می‌شود، استفاده می‌شود. LiteralExpression یک زیر کلاس مشخص از Expression است که از آن مشخصات id و typeRef را به ارث می‌برد. یک نمونه از LiteralExpression دارای یک مشخصه text اختیاری است که یک رشته است، و یک مشخصه expressionLanguage اختیاری، که رشته‌ای است که زبان بیان text مورد نظر را مشخص می‌کند. اگر هیچ expressLanguage مشخص نشده باشد، زبان عبارت text همان expressLanguage است که با نمونه حاوی Definitions مرتبط است. expressLanguage باید در قالب URI مشخص شود. زبان پیش‌فرض عبارت FEEL است.

به عنوان یک زیر کلاس از Expression، هر نمونه از LiteralExpression دارای یک مقدار است. text در نمونه‌ای از LiteralExpression ارزش آن را با توجه به معنای زبان عبارت LiteralExpression تعیین می‌کند. معناشناسی مدل‌های تصمیم‌گیری DMN همانطور که در این مشخصات توضیح داده شد، تنها در صورتی اعمال می‌شود که text تمام نمونه‌های LiteralExpression در مدل، عبارات معتبر در زبان عبارت مرتبط با آنها باشد.

نمونه‌ای از LiteralExpression ممکن است شامل importedValues که نمونه‌ای از زیرکلاس Import است و مشخص می‌کند متن LiteralExpression در کجا قرار دارد، باشد. importedValues عبارتی است که متن را از یک سند وارد شده انتخاب می‌کند. نمونه‌ای از LiteralExpression نباید هم متن و هم importedValues داشته باشد. importType از importedValues نوع سند حاوی متن وارد شده را مشخص می‌کند و باید با expressionLanguage عنصر LiteralExpression سازگار باشد. ExpressLanguage عنصر importedValues نحوه انتخاب متن وارد شده از سند وارد شده را مشخص می‌کند. برای مثال، اگر importType یک سند XML را نشان می‌دهد، expressionLanguage عنصر importedValues می‌تواند XPATH 2.0 باشد.

LiteralExpression تمام مشخصات و مدل‌های ارتباطی Expression را به ارث می‌برد. جدول زیر مشخصات افزوده شده و ارتباط مدل عنصر LiteralExpression را نشان می‌دهد.

مشخصه	شرح
text: string [0..1]	متن این LiteralExpression را معرفی می‌کند. این باید یک عبارت معتبر در Express ionLanguage باشد.
expressionLanguage: anyURI [0.. 1]	این مشخصه زبان عبارت مورد استفاده در این LiteralExpression را مشخص می‌کند. این مقدار زبان عبارت مشخص شده در نمونه حاوی DecisionRequirementDiagram را بازنویسی می‌کند. زبان باید در قالب URI مشخص شود.
importedValues: ImportedValues [0..1]	نمونه ImportedValues که مشخص می‌کند متن این LiteralExpression در کجا قرار دارد.

3- 1- 6- متامدل Invocation

فراخوانی مکانیزمی است که با اتصال محلی متغیرهای ورودی عبارت فراخوانی شده به مقادیر داخل عبارت فراخوانی، امکان ارزیابی یک عبارت مقدار (عبارت فراخوانی شده) را در داخل یک عبارت مقدار دیگر (عبارت فراخوانی کننده) می‌دهد. در یک فراخوانی، معمولاً متغیرهای ورودی عبارت فراخوانی شده، پارامتر (parameters) نامیده می‌شوند. Invocation یا فراخوانی، اجازه می‌دهد که یک عبارت مقدار یکسان در چندین عبارت استفاده شود، بدون اینکه آن را به عنوان یک عبارت فرعی در تمام عبارات استفاده کننده، تکرار کنید.

کلاس Invocation برای مدل‌سازی فراخوان‌ها به شکل نوعی Expression استفاده می‌شود. به عبارت دیگر Invocation یک کلاس تخصصی از Expression است.

یک نمونه از Invocation از صفر یا چند binding ساخته شده است، که نمونه‌هایی از کلاس Binding هستند و نحوه اتصال bindingFormula ها به formalParameter های تابع فراخوانی شده را مدل می‌کنند. formalParameter های یک FunctionDefinition و مقدار InformationItems هستند و پارامترهای Bindings مقدار InformationItems هستند. اتصال مورد نظر با نام InformationItem تطبیق داده می‌شود.

یک Invocation شامل یک calledFunction، یک Expression، که باید با یک تابع ارزیابی شود، می‌شود. معمولاً این یک LiteralExpression است که یک BusinessKnowledgeModel را نامگذاری می‌کند.

مقدار یک نمونه از Invocation مقدار بدنه calledFunction است که به formalParameters آن مقادیری را در زمان اجرا به ازای اتصالات موجود در Invocation اختصاص داده است.

Invocation ممکن است برای مدل‌سازی فراخوان‌ها در مدل‌های تصمیم‌گیری استفاده شود، زمانیکه یک عنصر Decision دقیقاً یک عنصر knowledgeRequirement است، و زمانیکه decisionLogic در عنصر Decision فقط شامل فراخوانی عنصر BusinessKnowledgeModel است که توسط requiredKnowledge ارجاع داده شده است و ضرورتی ندارد که یک عبارت با مقدار پیچیده‌تر باشد.

استفاده از نمونه‌های Invocation به‌عنوان decisionLogic عنصر Decision، امکان استفاده مجدد از encapsulatedLogic یک BusinessKnowledgeModel را به‌عنوان منطق هر نمونه از Decision که به آن BusinessKnowledgeModel نیاز دارد، اجازه می‌دهد، جایی که هر عنصر Decision نیاز به الزام‌آوری‌های خاص خود را برای پارامتر encapsulatedLogic مشخص می‌کند.

calledFunction که با عنصر Invocation مرتبط است، باید encapsultedLogic عنصر BusinessKnowledgeModel باشد که توسط عنصر Decision که حاوی Invocation است مورد نیاز است. عنصر Invocation باید دقیقاً یک اتصال برای هر پارامتر در encapsulatedLogic مربوط به BusinessKnowledgeModel داشته باشد.

Invocation تمام مشخصات و مدل‌های ارتباطی Expression را به ارث می‌برد. جدول زیر مشخصات افزوده شده و ارتباط مدل عنصر Invocation را نشان می‌دهد.

مشخصه	شرح
calledFunction: Expression [1]	عبارتی که مقدار آن تابع است.
binding: Binding [*]	این مشخصه نمونه‌های Binding را فهرست می‌کند که برای اتصال formalParameters مربوط به calledFunction در این Invocation استفاده می‌شود.

3- 1- 7- متامدل Binding

کلاس Binding برای مدل‌سازی یک عنصر Invocation، اتصال مقادیر formalParameters مربوط به formalFunction، استفاده می‌شود. Binding از یک bindingFormula که یک Expression است و از یک parameter که InformationItem است، ساخته می‌شود.

نام پارامترها در عناصر Binding باید زیرمجموعه‌ای از formalParameters مربوط به calledFunction باشند. هنگامی که عنصر Invocation اجرا می‌شود، به هر عنصر InformationItem که به عنوان یک parameter توسط یک binding در عنصر Invocation ارجاع می‌شود، در زمان اجرا، مقدار bindingFormula اختصاص داده می‌شود.

جدول زیر مشخصات و ارتباط مدل عنصر Binding را نشان می‌دهد.

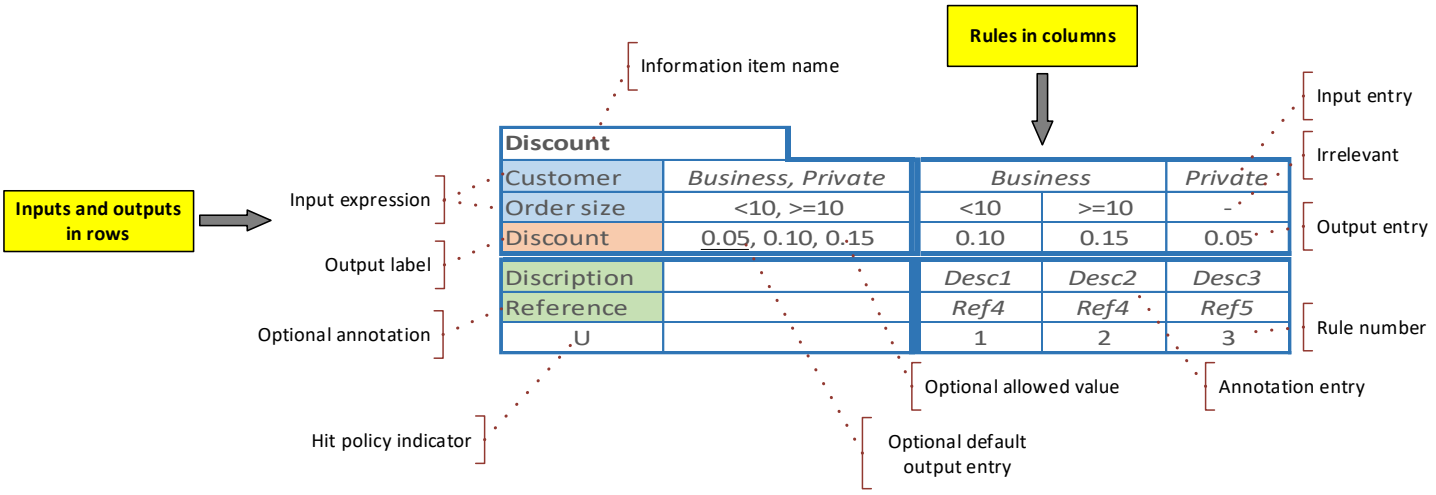
مشخصه	شرح
parameter: InformationItem	InformationItem که calledFunction مالکیت Invocation به آن وابسته است، به این Binding محدود می‌شود.
bindingFormula: Expression [0..1]	نمونه‌ای از Expression که هنگام ارزیابی نمونه مالکیت Invocation، پارامتر در این Binding به آن محدود می شود.

4- جدول‌تصمیم‌گیری

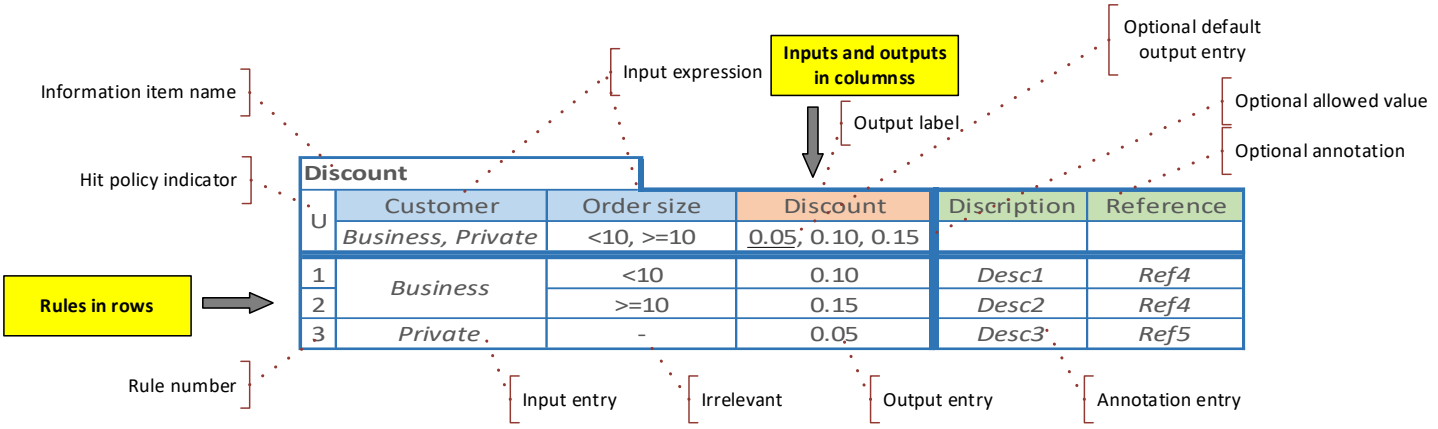
یکی از راه‌های بیان منطق تصمیم‌گیری متناظر با مصنوع یا نماد تصمیم‌گیری DRD، جدول‌تصمیم‌گیری است. جدول‌تصمیم‌گیری یک نمایش جدولی از مجموعه‌ای از عبارات ورودی و خروجی مرتبط است که قواعد را سازماندهی می‌کنند. این ساماندهی به گونه‌ای است که نشان می‌دهد کدام مقدار خروجی برای مجموعه خاصی از ورودی‌ها اعمال می‌شود. جدول‌تصمیم‌گیری شامل تمام (و فقط) ورودی‌های مورد نیاز برای تعیین خروجی است. علاوه بر این، یک جدول کامل شامل تمام ترکیبات ممکن از مقادیر ورودی (همه قواعد) است.

جداول‌تصمیم‌گیری و سلسله مراتب جداول‌تصمیم‌گیری سابقه اثبات شده‌ای در نمایش منطق‌تصمیم‌گیری دارند. یکی از اهداف DMN استانداردسازی اشکال و انواع جداول‌تصمیم‌گیری است. یک جدول‌تصمیم‌گیری از موارد زیر تشکیل شده است:

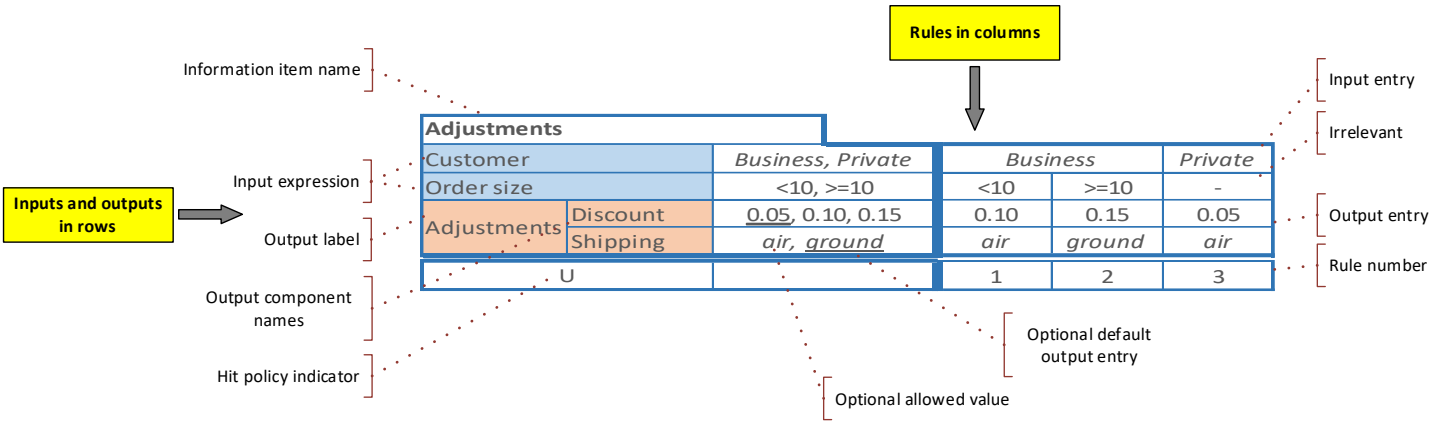
- **نام یک آیتم اطلاعاتی:** نام یک آیتم اطلاعاتی، که جدول‌تصمیم‌گیری بیان مقدار آن است. این نام معمولاً نام تصمیم‌گیری یا نام مدل‌دانش‌کسب‌وکار است که جدول‌تصمیم‌گیری، منطق‌تصمیم‌گیری را برای آن ارائه می‌دهد.
- **فهرستی از بندهای ورودی (صفر یا چند):** هر بند ورودی از یک عبارت ورودی و مقادیر مجاز اختیاری برای ورودی‌های محتمل که با عبارت مطابقت دارند ساخته شده است. ورودی‌های محتمل در قواعد جای می‌گیرند و ورودی i با عبارت ورودی i مطابقت دارد.
- **فهرستی از بندهای خروجی (یک یا چند):** هر بند خروجی از یک نام و مقادیر مجاز اختیاری برای خروجی‌های محتمل مطابق با عبارت، ساخته شده است. خروجی‌های محتمل در قواعد جای می‌گیرند و خروجی i با بند خروجی i مطابقت دارد. یک بند خروجی واحد، نامی ندارد. دو یا چند بند خروجی یک جدول‌تصمیم‌گیری را توصیف می‌کنند که برای هر حالتی یک محتوی را با یک ورودی برای هر عبارت خروجی برمی‌گرداند. هر یک از بندهای خروجی چندگانه باید نامگذاری شوند.
- **مجموعه‌ای از خروجی‌ها (یک یا چند):** یک خروجی واحد نامی ندارد، فقط یک مقدار دارد. دو یا چند خروجی را مؤلفه‌های خروجی می‌نامند. هر مؤلفه خروجی باید نامگذاری شود. هر خروجی (مؤلفه) باید یک خروجی محتمل را برای هر قاعده مشخص کند. مشخصات نام مؤلفه خروجی (اگر چندین خروجی باشد) و همه خروجی‌های محتمل به یک بند خروجی ارجاع می‌شود.
- **فهرستی از بندهای حاشیه‌نویسی (صفر یا چند):** هر بند حاشیه‌نویسی از یک نام ساخته شده است. هر حاشیه‌نویسی باید به عنوان بخشی از یک بند حاشیه‌نویسی قاعده، نامگذاری شود. ورودی‌های حاشیه‌نویسی در قواعد موجود است و ورودی حاشیه‌نویسی i با عبارت حاشیه‌نویسی i مطابقت دارد.
- **فهرستی از قواعد (یک یا چند):** در ردیف‌ها یا ستون‌های جدول (بسته به جهت‌گیری)، که در آن هر قاعده از ورودی‌ها و خروجی‌های خاص و حاشیه‌نویسی قواعد به صورت اختیاری برای هر ردیف (یا ستون) جدول تشکیل شده است. اگر قواعد به صورت ردیف بیان شوند، ستون‌ها عبارت هستند و بالعکس.



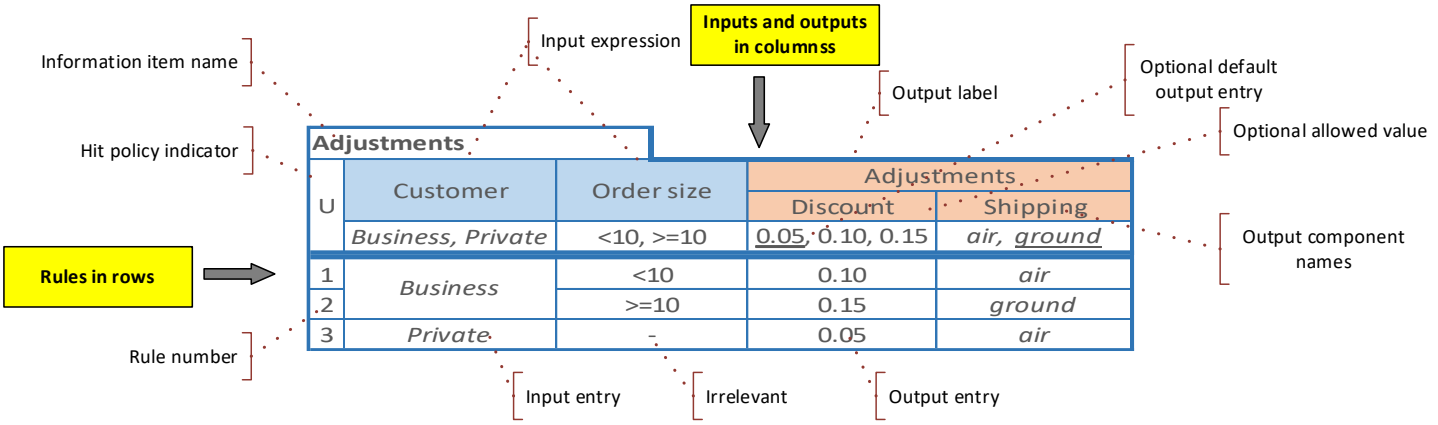
نمایی از یک جدول‌تصمیم‌گیری افقی که قواعد در ستون‌ها قرار می‌گیرند



نمایی از یک جدول‌تصمیم‌گیری عمودی که قواعد در سطرها قرار می‌گیرند



نمایی از یک جدول‌تصمیم‌گیری افقی به همراه خروجی چند مؤلفه‌ای که قواعد در ستون‌ها قرار می‌گیرند



نمایی از یک جدول‌تصمیم‌گیری عمودی به همراه خروجی چند مؤلفه‌ای که قواعد در ستون‌ها قرار می‌گیرند

جدول‌تصمیم‌گیری، قواعد را به صورت اختصار با مرتب کردن ورودی‌ها در سلول‌های جدول نشان می‌دهد. این شکل اختصاری همه ورودی‌ها را در هر قاعده به یک ترتیب نشان می‌دهد و بنابراین دارای مزیت خوانایی و امکان بررسی بهتر است. مثلا:

Customer	Order size	Discount
Business	<10	0.10

چنین خوانده می‌شود:

If Customer = “Business” **and** OrderSize < 10 **then** Discount = 0.10

به طور کلی، این به صورت زیر بیان می‌شود:

Input expression 1	Input expression 2	Output label
Input entry a	Input entry b	Output entry c

سه سلول زرد رنگ شده در بالا که قطعه‌ای از یک جدول تصمیم‌گیری است نشان دهنده قاعده عمومی زیر است:

If the value of input expression 1 satisfies input entry a

and the value of input expression 2 satisfies input entry b

then the rule *matches* and the result of the decision table is output entry c.

یک مقدار input expression یک input entry را برآورده می‌کند اگر مقدار آن برابر با input entry باشد، یا متعلق به لیست مقادیر نشان داده شده توسط input entry باشد (به عنوان مثال، یک لیست یا یک محدوده)، یا یکی از expression در input entry به مقدار true ارزیابی می‌شود. برای مشخصات کامل شرایط تطابق input entry، لطفاً به بخش 8.3.3 مراجعه کنید. اگر input entry "-" باشد (به معنای نامربوط)، هر مقدار input expression حتما input entry را برآورده می‌کند و آن ورودی خاص در قاعده مشخص شده نامربوط است.

اگر مقدار هر input expression، input entry مربوطه را برآورده کند، یک قاعده مطابقت دارد. اگر هیچ input entry وجود نداشته باشد، هر قاعده‌ای مطابقت دارد.

فهرستی از قواعد، منطق‌تصمیم‌گیری را بیان می‌کند. برای یک مجموعه معین از مقادیر ورودی، قاعده تطبیق (یا قواعد) مقدار حاصل را برای نام خروجی نشان می‌دهد. اگر قواعد با هم تداخل داشته باشند، چندین قاعده می‌توانند مطابقت داشته باشند و یک استراتژی‌برخورد نشان می‌دهد که چگونه می‌توان چندین تطابق را مدیریت کرد.

اگر دو input entry یک input expression هیچ مقداری به اشتراک نگذارند، ورودی‌ها (سلول‌ها) ناپیوسته نامیده می‌شوند. اگر تقاطع وجود داشته باشد، ورودی ها همپوشان (یا حتی مساوی) نامیده می‌شوند. عبارت Irrelevant (یا علامت {-}) با هر input expression از input entry همپوشانی دارد.

اگر همه input entry مربوطه همپوشانی داشته باشند، دو قاعده با هم همپوشانی دارند. ممکن است پیکربندی خاصی از داده‌های‌ورودی با این دو قاعده مطابقت داشته باشد.

اگر حداقل یک جفت input expression متناظر با هم تطابق داشته باشد، دو قاعده ناهمگون (غیر همپوشان) هستند. هیچ پیکربندی خاصی از داده‌های‌ورودی با دو قاعده مطابقت ندارد.

اگر در جداول اجازه داده شود که قواعد همپوشانی داشته باشند، استراتژی‌برخورد جدول نشان می‌دهد قواعدی که با هم تداخل دارند چگونه باید مدیریت شوند و مقدار(های) حاصل برای نام خروجی کدام است تا از ناهماهنگی جلوگیری شود.

4- 1- متامدل

4- 1- 1- متامدل Decision Table

4- 1- 2- متامدل Decision Table Input و Decision Table Output

5- زبان عبارات ساده S-FEEL

DMN زبان friendly enough expression language (FEEL) را به منظور ارائه استانداردی برای اجرای معانی در مدل تصمیم‌گیری با انواع عبارات، تعریف کرده است.

این بخش یک زیرمجموعه ساده از زبان FEEL، که S-FEEL نام دارد را تعریف می‌کند تا به معانی موجود در مدل‌های تصمیم‌گیری که فقط از عبارات ساده استفاده می‌کنند، به صورت استاندارد قابلیت اجرایی بدهد. این زیرمجموعه به‌ویژه در مدل‌های تصمیم‌گیری که در آن‌ها منطق‌تصمیم‌گیری عمدتاً یا فقط با استفاده از جداول‌تصمیم‌گیری، مدل‌سازی می‌شود، کاربرد دارد.

تجربه با DMN از زمان انتشار نشان داده است که تعداد کمی از مدل‌های تصمیم‌گیری کامل را می‌توان با استفاده از زبان S-FEEL تعریف کرد. جداول‌تصمیم‌گیری منحصر به فرد را می‌توان تنها با استفاده از زبان S-FEEL تعریف کرد، اما در یک مدل تصمیم‌گیری معمول حداقل یک تصمیم‌گیری وجود دارد که به زبان FEEL نیاز دارد. بنابراین توسعه دهندگان و کاربران، تشویق می‌شوند تا از مشخصات کامل FEEL به جای زیرمجموعه S-FEEL استفاده و پیاده‌سازی کنند.

6- زبان عبارات FEEL

در DMN، تمام منطق‌تصمیم‌گیری در قالب کادرعبارات، نمایش داده می‌شود. بخش 3- 1- مفهوم کادرعبارت را معرفی کرد و دو نوع ساده با عناوین کادر عبارات تحت اللفظی و کادر فراخوانی را تعریف کرد. بخش 4 جداول‌تصمیم‌گیری که یک نوع بسیار مهم از کادرعبارت است را تعریف می‌کند. این بخش نماد گرافیکی منطق‌تصمیم‌گیری را با تعریف انواع دیگر کادرعبارات تکمیل می‌کند.

عباراتی که درون کادرها قرار می‌گیرند، عبارت زبان FEEL هستند. FEEL مخفف Friendly Enough Expression Language است و دارای امکانات زیر است:

- بدون عوارض جانبی
- مدل داده ساده با اعداد، تاریخ، رشته، لیست و محتوی
- نگارش ساده طراحی شده برای سطح وسیعی از مخاطبان
- منطق سه ارزشی (true، false، null)

این قسمت نیز به طور کامل نگارش و معنای زبان FEEL را مشخص می‌کند. نگارش یا هملن دستور زبان در ادامه همین گفتار، شرح داده شده است. قسمتی از شرح این نگارش به صورت گرافیکی در قالب کادرعبارت نیز در دنباله همان گفتار آمده است.

زبان FEEL دو نقش در DMN بازی می‌کند:

1. به عنوان یک نماد متنی در کادرعبارات مانند جداول‌تصمیم‌گیری.
2. به عنوان یک زبان کمی وسیع‌تر برای بیان منطق عبارات و روشی ساده و یکنواخت در DRGها به منظور بیان مفاهیم ترکیبی برای رسیدن به هدف اصلی.