ARIS メソッド マニュアル

ARIS メソッド

バージョン 9.7 – Service Release 1

2015 年 1 月

Product Documentation

This document applies to ARIS Version 9.7 and to all subsequent releases. Specifications contained herein are subject to change and these changes will be reported in subsequent release notes or new editions.

Copyright © 2010 - 2015 Software AG, Darmstadt, Germany and/or Software AG USA Inc., Reston, VA, USA, and/or its subsidiaries and/or its affiliates and/or their licensors.

The name Software AG and all Software AG product names are either trademarks or registered trademarks of Software AG and/or Software AG USA Inc. and/or its subsidiaries and/or its affiliates and/or their licensors. Other company and product names mentioned herein may be trademarks of their respective owners. Detailed information on trademarks and patents owned by Software AG and/or its subsidiaries is located at http://documentation.softwareag.com/legal/.

Use of this software is subject to adherence to Software AG's licensing conditions and terms. These terms are part of the product documentation, located at http://documentation.softwareag.com/legal/ and/or in the root installation directory of the licensed product(s).

This software may include portions of third-party products. For third-party copyright notices and license terms, please refer to "License Texts, Copyright Notices and Disclaimers of Third Party Products". This document is part of the product documentation, located at http://documentation.softwareag.com/legal/ and/or in the root installation directory of the licensed product(s).

目次

1 はじめに 8

2 ARIS (Architecture of Integrated Information Systems) 10
- 2.1 ARIS アーキテクチャの概念 10
- 2.2 説明ビュー 10
- 2.3 説明レベル 13

3 プロセス連鎖分析 17
- 3.1 ビジネス管理の問題の説明 17
- 3.2 プロセス連鎖図 (PCD) 17

4 ARIS 概念のビューおよびレベル内のモデリング 20
- 4.1 ファンクション ビュー 20
  - 4.1.1 要件定義 20
  - 4.1.2 仕様設計 - アプリケーション システム タイプ図 27
  - 4.1.3 実装 - アプリケーション システム図 31
- 4.2 データ ビュー 33
  - 4.2.1 要件定義 33
  - 4.2.2 仕様設計 62
  - 4.2.3 実装 - テーブル図 65
- 4.3 組織ビュー 67
  - 4.3.1 要件定義 67
  - 4.3.2 仕様設計 - ネットワーク トポロジ 74
  - 4.3.3 実装 76
- 4.4 プロセス ビュー 79
  - 4.4.1 要件定義 79
  - 4.4.2 仕様設計 120
  - 4.4.3 実装 - アクセス図 (物理的) 127
- 4.5 製品/サービスのモデリング 131
  - 4.5.1 製品/サービス交換図 131
  - 4.5.2 製品/サービス ツリー 132
  - 4.5.3 製品割当図 133
  - 4.5.4 製品ツリー 136
  - 4.5.5 製品選択マトリクス 136
  - 4.5.6 競争力モデル 137

5 ARIS の統一モデリング言語 138
- 5.1 はじめに 138
- 5.2 ARIS UML Designer - サポートされる UML 標準 138

6 Object Modeling Technique (OMT) 139
- 6.1 はじめに 139
- 6.2 OMT 法のまとめ 139
- 6.3 ARIS のオブジェクト モデリング手法 139
  - 6.3.1 OMT オブジェクト モデル 140
  - 6.3.2 OMT 動的モデル 146
  - 6.3.3 OMT 機能モデル 149

6.3.4 オブジェクトを階層的に配置する方法 152

7 知識管理の手法 153

7.1 はじめに 153
7.2 知識処理モデリングのためのオブジェクト タイプ 153

7.2.1 知識カテゴリ 153
7.2.2 有形知識 155

7.3 知識処理モデリングのためのモデル タイプ 155

7.3.1 知識構造図 155
7.3.2 知識マップ 156
7.3.3 業務プロセスにおける知識処理の表現 158

8 ユース ケース 159

8.1 一般的な文書化 161
8.2 データベース管理/データ ウェアハウジング 162
8.3 PC ハードウェアとネットワークの管理 163
8.4 プロセス原価管理 163
8.5 品質管理 164
8.6 再編成措置 165
8.7 SAP R/3 導入 165
8.8 ソフトウェアの開発と導入 166
8.9 知識管理 167
8.10 ワークフロー管理 168

9 バランス スコアカード メソッド 169

9.1 はじめに 169
9.2 バランス スコアカード メソッドの概念 169

9.2.1 バランス スコアカード メソッドの主な要素 169
9.2.2 戦略的管理プロセスおよびバランス スコアカード 170
9.2.3 バランス スコアカードの利点 174

9.3 ARIS BSC を使用したバランス スコアカードの作成 175

9.3.1 用語および略語 175
9.3.2 ARIS BSC を使用したバランス スコアカードの作成 176
9.3.3 ほかのモデルとの関係 182

10 e-ビジネス シナリオ図 183

10.1 はじめに 183
10.2 e-ビジネス シナリオ図手法 185

10.2.1 考え方 185
10.2.2 モデルおよびそのオブジェクト 185
10.2.3 [転送タイプ] 属性グループ 186

10.3 レポートを使用した評価 187

10.3.1 データ セキュリティのチェック 187
10.3.2 システム サポート 187
10.3.3 情報フロー 187
10.3.4 協調ビジネス マップ 187

10.4 ほかのメソッドおよびコンポーネントへの接続 188

11 IT 都市計画 192

11.1 エンタープライズ アーキテクチャおよび IT 都市計画 192
11.2 IT 都市計画を有効に使用できる企業 192
11.3 ARIS を使用した IT 都市計画 193
11.4 IS ビュー 194
11.5 サービス タイプとそのデータ 198
11.6 サービス タイプの詳細な説明 198
11.7 IS 要素の時系列の操作順序 199
11.8 IT ビュー 199
11.9 IT 要素および IT 要素のデータ 201
11.10 IT 要素の詳細記述 201
11.11 組織的側面 202
11.12 IT 要素の時系列の操作順序 202
11.13 アーキテクチャ内の時系列の操作順序 203
11.14 使用できる評価 204

12 業務プロセスのモデル化 205

12.1 プロセス クラスと業務プロセス図 205
12.2 ARIS での BPMN の実装 207
12.3 業務プロセス図の要素 208

12.3.1 プールとレーン 208
12.3.2 プールとレーンのモデリング基準 209
12.3.3 シーケンス フロー 209
12.3.4 シーケンス フロー接続線のモデリング基準 210
12.3.5 メッセージ フロー 210
12.3.6 メッセージ フロー接続線のモデリング基準 211
12.3.7 関連 211
12.3.8 イベント 211
12.3.9 イベントのモデリング基準 212
12.3.10 アクティビティ 213
12.3.11 アクティビティのモデリング基準 214
12.3.12 ゲートウェイ 215
12.3.13 ゲートウェイのモデリング基準 215
12.3.14 成果物 216
12.3.15 図の出典 218

13 参考文献 219

13.1 一般的な参考文献リスト 219
13.2 テーマ関連の参考文献 220

13.2.1 ARIS の統一モデリング言語 220
13.2.2 Object Modeling Technique (OMT) 221
13.2.3 知識管理の手法 221
13.2.4 バランス スコアカード メソッド 221
13.2.5 IT 都市計画 221
13.2.6 業務プロセスのモデル化 222

図表一覧

図 1: 業務プロセス モデル 11
図 2: プロセス モデルのビュー 12
図 3: プロセス モデルのビュー 13
図 4: 情報システムの説明 15
図 5: ARIS の概念 16
図 6: プロセス連鎖図の例 18
図 7: [顧客引合いの確認] ファンクションの表現 20
図 8: ファンクション ツリー (抽出) 21
図 9: オブジェクト指向のファンクション ツリー 22
図 10: プロセス指向のファンクション ツリー 23
図 11: 操作指向のファンクション ツリー 24
図 12: Y 図 25
図 13: SAP アプリケーション図 26
図 14: 目標図 27
図 15: アプリケーション システム タイプのグラフィック 28
図 16: アプリケーション システム タイプのモジュール構造 28
図 17: IT ファンクション タイプのグラフィック 29
図 18: アプリケーション システム タイプへのファンクションの割り当て 29
図 19: アプリケーション システム タイプの設定 30
図 20: 画面と一覧の割り当て 30
図 21: アプリケーション システムとモジュールのグラフィック 31
図 22: アプリケーション システム タイプへのアプリケーション システムの割り当て 31
図 23: 同じタイプの 2 つのアプリケーション システムが異なるモジュール構造を持つ例 32
図 24: アプリケーション システム タイプ、プログラム モジュール タイプ、およびプログラム モジュールの割り当て 32
図 25: エンティティ タイプの例 34
図 26: [顧客] エンティティ タイプの属性の例 34
図 27: 関係タイプの例 35
図 28: 2 つのエンティティ タイプ間の関係の多重度 36
図 29: ERM における多重度 36
図 30: 部品一覧の ERM 37
図 31: ERM における属性の割り当て 37
図 32: 「顧客 (CUSTOMER)」の分類 38
図 33: 汎化/特化 39

図 34: 完全な特化 39
図 35: 集約の例 40
図 36: 再統合された関係タイプとの集約 40
図 37: データ クラスター (図形シンボル) 41
図 38: 複数のオブジェクトのデータ クラスタ ビュー 41
図 39: グループ化 41
図 40: 上限/下限 (1) 42
図 41: 上限/下限 (2) 42
図 42: 上限/下限 (3) 42
図 43: 上限/下限 (4) 43
図 44: 存在の依存関係 43
図 45: 用語 (1) 44
図 46: 用語 (2) 44
図 47: エンティティ タイプへの ERM 属性の割り当て 45
図 48: 属性タイプ グループの表現 45
図 49: eERM 表記および SAP ERM 表記 47
図 50: IE 表記のデータ モデル 48
図 51: SeDaM 表記のデータ モデル 48
図 52: eERM: 用語および表記法 49
図 53: 内容がテキストのみの DTD 要素タイプ 50
図 54: 混在した内容を持つ要素タイプと、DTD への変換 51
図 55: 列挙属性タイプを持つ要素タイプ 54
図 56: 材料図の例 57
図 57: スター スキーマのデータ ウェアハウス 58
図 58: 認証階層 59
図 59: 原価作用因図の例 60
図 60: 原価要素図の例 61
図 61: 情報媒体図 61
図 62: リレーションのグラフィック 62
図 63: 要件定義属性およびデータ オブジェクトの割り当て 62
図 64: 属性割当図 63
図 65: ビューの定義 63
図 66: ERM 関係タイプの属性への割り当て 63
図 67: [システム属性] モデルの例 64

図 68: システム属性ドメイン 64
図 69: テーブルとフィールドのグラフィック 65
図 70: フィールドの割り当て 65
図 71: 要件定義および設計仕様オブジェクトの割り当て 66
図 72: テーブルの実例 66
図 73: 製品による組織分割 68
図 74: ハイブリッドな組織形態 68
図 75: 組織図 69
図 76: 役職と要員を割り当てた組織図 70
図 77: 要員タイプ 71
図 78: 場所の割り当て 71
図 79: 場所の階層 72
図 80: シフト カレンダーの例 74
図 81: ネットワーク タイプ図のグラフィック 74
図 82: ネットワーク トポロジ 75
図 83: 場所の割り当てを含むネットワーク図 76
図 84: ハードウェア コンポーネントと場所の割り当てを含むネットワーク図 77
図 85: [技術的リソース] モデルの例 79
図 86: ファンクションへの組織要素の割り当て 80
図 87: イベント (グラフィック表示) 81
図 88: EPC の例 82
図 89: ルールの例 83
図 90: 論理演算子 (ルール) 84
図 91: 起動イベントの AND 演算子 85
図 92: 起動イベントの OR 演算子 85
図 93: 起動イベントの XOR 演算子 86
図 94: 作成されるイベントの AND 演算子 86
図 95: 作成されるイベントの OR 演算子 86
図 96: 作成されるイベントの XOR ルール 87
図 97: 生成されるイベントとファンクションの AND 演算子 87
図 98: 作成されるイベントとファンクションの OR 演算子 87
図 99: 生成されるイベントとファンクションの XOR 演算子 88
図 100: 起動イベントとファンクションの AND 演算子 88
図 101: ファンクション割当図 (I/O) の例 89

図 102: ファンクション割当図の詳細な表現 90
図 103: 入出力データを含む EPC 90
図 104: 入出力データを含む EPC 91
図 105: 情報フロー図と、[アサインメント ウィザード] を開いた画面 92
図 106: イベント図の例 93
図 107: プロセス連鎖の例 (要件定義) 94
図 108: ファンクション、データ、組織ユニット、およびイベントを含む EPC 95
図 109: 入出力図 96
図 110: 付加価値連鎖の例 96
図 111: ルール図における複雑な演算子の表現 97
図 112: プロセス選択マトリクス (SAP AG R/3 リファレンス モデルからの抽出) 99
図 113: EPC (マテリアル フロー付き) からの抽出 100
図 114: EPC (列表示) 102
図 115: ロール割当図 (RAD) 103
図 116: ビジネス コントロール図の例 104
図 117: DW 変換 : データ ウェアハウスのデータ変換 105
図 118: 自動車業界の e-ビジネス シナリオ図の例 107
図 119: 構造モデルの例 (VDA 6.2 標準からの抽出) 108
図 120: EPC、製造プロセス、オフィス プロセスの各モデル タイプでの事実の例 109
図 121: EPC から作成された PPC の例 110
図 122: プロセス インスタンス生成モデル 111
図 123: RAMS 図の例 112
図 124: ロール図 113
図 125: c3 モデルの構造 115
図 126: 登録ダイアログの画面設計と C++ での実装の例 116
図 127: イベントによる画面ナビゲーションの例 117
図 128: 事業区分マトリクスの例 118
図 129: レポート 119
図 130: アプリケーション システム タイプ間での情報フロー 120
図 131: 仕様設計レベルにおける I/O データ 120
図 132: アクセス権 121
図 133: 責任の定義 121
図 134: アクセス図 (抜粋) 122
図 135: プログラム フロー図 (PF) の例 124

図 136: 画面図の例 125
図 137: 前の図の画面図から取得した画面 126
図 138: データ フロー 127
図 139: 入出力の関係 128
図 140: ハードウェア コンポーネントの割り当て 129
図 141: プラットフォームとしてのハードウェア コンポーネント 129
図 142: ユーザーとアプリケーション システム 130
図 143: 場所の割り当て 130
図 144: あるソフトウェア会社における製品/サービス交換の例 132
図 145: 製品/サービス ツリー 133
図 146: 製品割当図の例 134
図 147: 製品割当図 - 当座預金口座 134
図 148: 製品割当図 - 販売品 135
図 149: 製品ツリーを利用した、「市民登録業務」製品グループの分類 136
図 150: 社会福祉事務所の製品選択マトリクス 136
図 151: スポーツ カー市場の競争 137
図 152: インスタンスの表現 140
図 153: クラスの表現 140
図 154: インスタンスとクラス間の接続線 140
図 155: クラスへの属性の割り当て 141
図 156: クラスへの操作の割り当て 141
図 157: インスタンス間の関連 141
図 158: クラス間の関連 142
図 159: 3 個のクラス間の関係 142
図 160: クラスとしての関連のモデリング 143
図 161: 限定子付き関連の表現 143
図 162: 関連の順序の表現 143
図 163: クラス間の集約 144
図 164: 属性の制約の表現 145
図 165: 関連の制約の表現 145
図 166: OMT オブジェクト モデルの例 146
図 167: 初期状態、終了状態、遷移の表現 146
図 168: 状態間の遷移の表現 147
図 169: 遷移の追加情報の表現 147

図 170: OMT 動的モデルの例 148
図 171: データ ストアの表現 149
図 172: プロセスの表現 149
図 173: アクターの表現 149
図 174: データ フローの表現 149
図 175: データ フロー分割の表現 150
図 176: OMT 機能モデルの例 151
図 177: 知識構造図 155
図 178: 知識マップ - 組織ユニットに関連させた場合 156
図 179: 知識マップ - マトリクス表現 157
図 180: EPC における知識処理 158
図 181: 戦略的管理構造としての BSC 170
図 182: BSC の視点 172
図 183: BSC 因果関係図 177
図 184: BSC KPI 割当図 179
図 185: KPI ツリー 181
図 186: e-ビジネスで可能な取引 184
図 187: e-ビジネス シナリオ図のオブジェクト 186
図 188: e-ビジネス シナリオ「オンライン ショップ」からの抜粋 189
図 189: パイプライン図からの抜粋 190
図 190: DTD からの抜粋: 順序 191
図 191: プロセス ビュー、IS ビュー、IT ビュー 194
図 192: 企業の情報システムのゾーン 195
図 193: エリアに分割されたゾーン 195
図 194: 構築クラスタに分割されたエリア 196
図 195: 機能ブロックに分割される「人材サービス」構築クラスタ 196
図 196: 「給与」機能ブロックの機能と IS サービス 197
図 197: IS ビュー、関係、およびエンティティ タイプの間の [所有者である] 接続線 198
図 198: IS 要素およびファンクション間の [サポートする] 接続線 199
図 199: DATEV システムのサブシステム構造 200
図 200: アクセス図の IT 要素の詳細記述 201
図 201: 技術的インフラストラクチャの影響および効果 202
図 202: 時系列な操作順序への IS および IT 要素の統合 203
図 203: シーケンス フローとメッセージ フローを含む 2 つのプール 207

図 204: BPMN による 2 つのレーンを持つプール 208
図 205: シーケンス フロー接続線 209
図 206: メッセージ フロー接続線 210
図 207: 関連接続線 211
図 208: イベント カテゴリ 211
図 209: イベント タイプの例 212
図 210: BPMN のアクティビティ 213
図 211: ARIS でアクティビティとして割り当てられたファンクション 213
図 212: ゲートウェイ タイプ 215
図 213: 電子メール投票プロセス 217

1 はじめに

ハードウェアの標準化が進み、ハードウェア価格が劇的に下がったため、情報システムの開発タスクの焦点は著しく変化しました。

以前は、最適化の可能性は主にシステム設計とシステム統合にありました。 しかし、今日では、個々の部門の特殊な要求に対するソリューションを作成する方向に焦点が移ってきています。 分散情報システムが登場し、これらを統合情報システム インフラストラクチャへ組み込むことが可能になったことにより、企業の組織構成において新しいコスト削減の可能性が生まれました。

以前は、組織構造は制限された機能のみを持つ集中管理ホスト環境を基盤とする場合が多かったため、機能的に分割され、中央で適応されていました。 このため、企業は柔軟性に欠けていました。 当初は、コンピューター サービスの分散化を進めることによって生まれる新たな可能性を認識し、同時に新しい情報システム アーキテクチャ概念 (たとえば、クライアントとサーバー、ワークフロー管理など) を開発することに注目する人はほとんどいませんでした。

しかし今日では、激しい競争の中で、どの企業にとっても、この可能性が最も強い関心を持つテーマになりました。 内部の業務プロセスに焦点を絞った柔軟な構造は、企業にとって重要な競争力になります。 しかし、企業が、最適化された情報システム インフラストラクチャによって、相互接続されたプロセスを認識、合理化、およびサポートできるようになるのは、業務プロセスの全体的なビューが利用できる場合のみです。 従来の中央集中的なビジネス環境に比べて、このような新しい構造の管理はかなり複雑になっています。 このよう問題に対応するには、責任を明確に割り当てること、構造の透過性を最大にすること、すべての企業レベルにおける均質な通信基盤、および定義された業務目標に基づく合理的なプロジェクト管理が必要です。

企業モデリングの手法は、このような複雑な業務を達成できるように企業経営者をサポートします。 業務プロセスを分析し、プロジェクトを業務全体の目標に適合させ、合理的な組織構造を最適にサポートする分散された統合システムという形で情報システム インフラストラクチャを使用するためには、企業モデルが不可欠です。

会社の実際の状況をモデリングすること (そして、その過程で、ビジネス全体を調査すること) が、ますます議論の的になっています。 使用されるモデリング手法が増えたこととその多様性により、複雑さが増し、混乱を招くようになりました。 その結果、開発およびモデリング手法に対して、標準化された概念の枠組み (アーキテクチャ) を定義するための試みが行われました。

ARIS© (Architecture of Integrated Information Systems) はこのようなアーキテクチャの 1 つで、シェアー博士によって開発されました (Scheer 著『Architecture of Integrated Information Systems』(1992 年) 参照)。このアーキテクチャの概念により、手法を評価し、それらの手法をそれぞれの特徴に合わせて組み込むことが可能になります。また、この概念を複雑な開発プロジェクトを方向づけするフレームワークとしても使用できます。これは、統合情報システムの開発に関するプロシージャ モデルが含まれた構成要素によって実現されます。

このタイプのアーキテクチャにより、手法の使用における標準化が自然と導き出されます。 このアーキテクチャに基づいて、既存および新規のモデリング手法が統合され、業務プロセスをモデリングする総体的な手法が誕生しました。

また、ARIS アーキテクチャは ARIS Platform の基盤となります。 ARIS Platform は、Software AG 製品の範囲内で ARIS Architect などの製品を統合します。 これらの製品は、業務プロセス リエンジニアリングの観点から、業務プロセスの作成、分析、評価において、コンサルタントおよび企業を支援します。 事業部門の関連する業務プロセスの記録とモデル化は、ARIS Designer を使用して行うと便利です。

このマニュアルでは、関連するモデリング手法について紹介します。 また、ARIS 製品とこれに提供されるシステム アドオンのすべてを活用するアプローチおよびメソッドについて説明します。 このマニュアルは、ツールの使用に関する疑問や問題の対処方法を目的としていない場合も、モデリング手法に取り組むユーザーにとって大変役立つ内容となっています。

2 ARIS (Architecture of Integrated Information Systems)

2.1 ARIS アーキテクチャの概念

ARIS (**AR**chitecture of integrated **I**nformation **S**ystems) は、業務プロセス全体の視点から導き出された統合概念に基づいています。 アーキテクチャ作成の最初のステップは、業務プロセスを記述するためのすべての基礎要素を含む業務プロセス モデルを開発することです。 この開発結果は複雑なモデルとなるため、モデルを個別のビューに分割することによって複雑さを軽減します。 この分割によって、個々のビューの内容を、そのビューに適した特別な方法で記述できます。この段階では、さまざまなビューの相互関係に注意を払う必要はありません。 ビュー間の関係は最終段階で組み込まれ、プロセス連鎖全体の分析に重複することなく連結されます。

複雑さを減らすためのもう 1 つの方法は、説明によって区別することです。 ライフサイクルの概念に従って、IT とどれだけ深く関わっているかによって、さまざまな方法で情報システムを分類して記述できます。 これにより、ビジネス管理の問題から、技術的な実装まで一貫した記述が保証できます。

このように、ARIS の概念は、統合化された情報システムの開発と最適化、およびその実装を記述するためのフレームワークを構成します。 技術的な説明レベルに重点が置かれるため、ARIS の概念はビジネス管理関連のプロセス連鎖の作成、分析、および評価のモデルとして機能します。 シェアー博士は著書で統合情報システム アーキテクチャについて詳しく説明しています (Scheer 著『Architecture of Integrated Information Systems』(1992 年) および『ARIS - Business Process Frameworks』(1998 年) を参照してください)。

2.2 説明ビュー

この手法の焦点は、次の図に示されるようなビジネスプロセスです。

このプロセスは、「注文が届く」イベントによって発生します。 このイベントによって、「注文の受理」ファンクション (プロシージャ) が起動されます。 このプロシージャを実行するためには、まず関連するプロセス環境の現在の状態を記述する必要があります。 これには、特に顧客や品目に関連するデータが含まれます。 関連するオブジェクトの状態は、ワークフローの処理中に変更される可能性があります。たとえば、新たな予約データによって、在庫データが更新される場合などです。

この処理は、部門に割り当てることのできる販売部員によって実行されます。 部門では、PC やプリンタなど、タスクの実行に必要な特定の IT リソースを使用します。

「注文の受理」プロシージャが完了すると、「注文の確認」イベントが発生し、これによってほかのプロシージャ（注文の追跡、生産計画など）が開始されます。「注文が受け取られた」オブジェクトが「注文が確認された」オブジェクトになるため、「注文」オブジェクトは新しい状態になります。「注文の受理」ファンクションが実行されると、製品/サービスが生成されます。これは、以降のプロシージャを処理するための入力データとして、人員リソースおよび技術的リソースとともに使用されます。

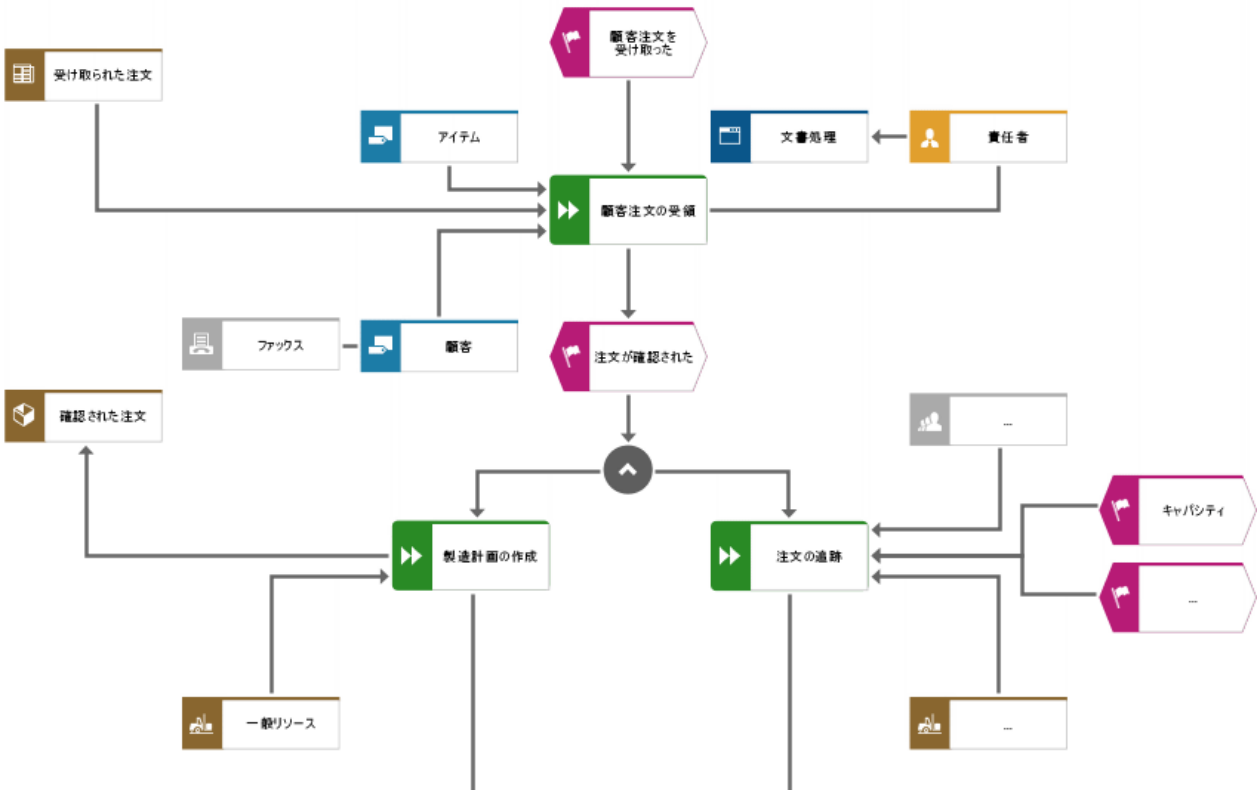


図 1: 業務プロセス モデル

完全な業務プロセスを記述するために必要な要素は、プロシージャ、イベント、製品/サービス（状態）、ユーザー、組織ユニット、および IT リソースなどです。 対象となる全イベントのすべてのプロシージャ要素への影響を全部含めると、モデルがかなり複雑になり、記述の中に重複が生じます。

この複雑さを軽減するためには、全般的なコンテキストを、特定のモデリングおよび設計側面を表す個別のビュー（次の図を参照）に分割します（Scheer 著『Architecture of Integrated Information Systems』(1992 年)、13 ページ以降参照）。 各ビューは、独立して処理することができます。 ビューの分割は、個々のビュー間の関係は極力少なくし、ビュー内の要素同士の関係は多数になるように行います。

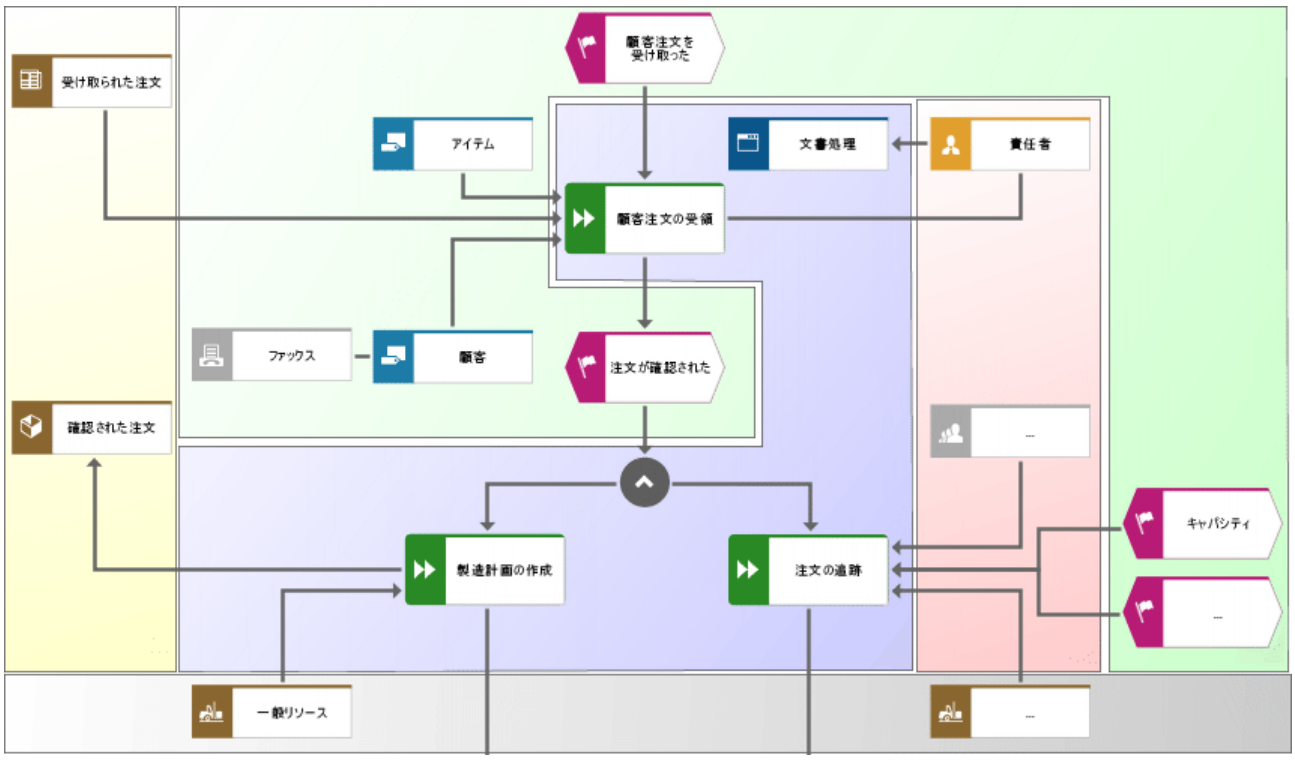


図 2: プロセス モデルのビュー

「顧客注文が受け取られた」や「請求書が作成された」などのイベントは、情報オブジェクト（データ）の状態が変わったことを表します。 イベントは、ARIS アーキテクチャのデータ ビューで記述されます。

オブジェクトの環境内（たとえば、顧客注文のスコープ内）の状態は、製品/サービスによって表されます。 製品/サービスという用語は、商品またはサービスを提供することを意味します。 情報を作成および提供するサービスは、情報サービスです。 製品/サービスには、財務リソースを提供することも含まれます。 製品/サービス間の関係は、ARIS アーキテクチャの [製品/サービス ビュー] に示されます。

実行（プロシージャ）されるファンクションとそれらの相互関係は、もう 1 つのビューである [ファンクション ビュー] を構成します。 このビューには、ファンクションの記述、全体のコンテキストを構成する個々のサブファンクションの一覧、および各ファンクション間の関係が含まれます。

[組織ビュー] は、ユーザーと組織ユニットの組み合わせを含みます。また、それらの関係や構造も含みます。

IT リソースは、4 番目の、対象となる領域である [リソース ビュー] を構成します。 ただし、このビューが業務プロセスの技術面を考慮したビューとしての意味を持つのは、より直接的に業務管理に適応するほかの要素を記述する一般的な条件を提供する場合に限ります。 このため、ほかのビュー（データ、ファンクション、組織）の要素は、それらに関連する IT リソースに関して記述されます。 したがって、リソースは、ほかのビューの仕様設計および実装のレベルで扱われます（「説明レベル『13 ページ 』」の章を参照）。 リソース ビューのかわりに、含める個別のオブジェクトとして、説明レベルのアプローチの結果定義されたライフサイクル モデルを使用します。

プロセスを個々のビューに分割することによって複雑さは軽減されますが、各ビューのプロセス要素間の関係は失われます。 このため、[プロセス ビュー] がビュー間の関係を記述する追加のビューとして提供されています。 これらの関係を別のビューと組み合わせることにより、すべての関係を重複なく体系的に記録することが可能になります。

プロセス ビューは ARIS の重要な要素です。 これは、ARIS の概念とほかのアーキテクチャ アプローチとの間に一線を画す基本的な機能です (ほかのアーキテクチャ アプローチとの比較については、Scheer 著『Architecture of integrated information Systems』、24 ページ以降を参照)。

このため、以下のメソッド説明の基準を形成する 5 つの ARIS ビューがあります。

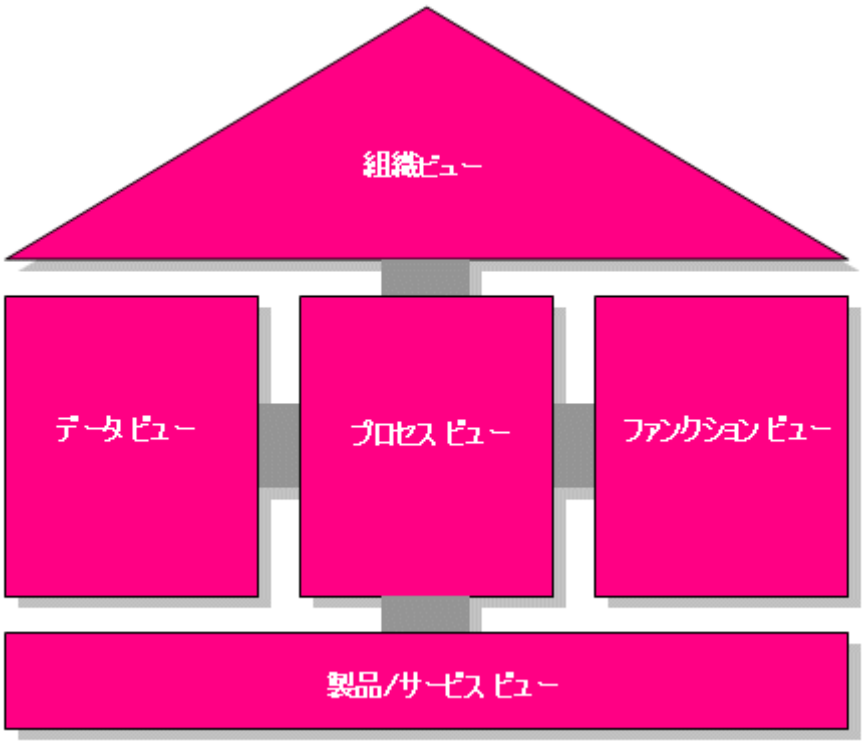


図 3: プロセス モデルのビュー

2.3 説明レベル

先に述べたように、ARIS のリソース ビューは、情報システムの説明のライフサイクル概念に置き換えることができます。

レベル、またはフェーズ概念の形をとるライフサイクル モデルは、情報システムのライフサイクルを記述します。 しかし、ARIS のライフサイクル モデルは、情報システムを開発するためのプロシージャ モデルとして理解すべきではありません。 むしろ、IT との関連性において異なるさまざまな説明レベルを定義するために使用されます。

ARIS では、次の図に示す 3 層分割を使用します (Scheer 著『Architecture of Integrated Information Systems』(1992 年)、16 ページ以降を参照)。

この手法の焦点は、ビジネス管理の問題です。 この記述には、技術的な目標と技術的な言語に焦点を置いたおおまかな事実情報が一覧されます。 ここでは、業務管理プロセスや決定をサポートする IT のオプションも含められます。 したがって、こ

れを記述するためには、略式の記述手法のみが使用されます。 このような記述メソッドは概略的で、あまり専門的な技術用語が使用されていないので、正式な実装の開始点としての役割を果たすことができません。

そのため、[要件定義] は、整合性のある IT への変換の基礎として使用できるように、サポートする業務管理の手法を正式な記述言語で記述します。 このプロセスは、(意味論的) モデリングとも呼ばれます。 要件定義は、次の図で矢印の太さにより表されているように、ビジネス管理の問題と密接な関係にあります。

要件定義の概念を IT システム設計に適用すると、[仕様設計] レベルになります。 ここでは、技術ファンクション自体ではなく、技術ファンクションを実行するモジュールやトランザクションを定義します。 このレベルでは、要件定義は IT で使用される一般概念に適合されます。 ただし、要件定義と仕様設計は、ゆるやかな関係しか持ちません。 これは、要件定義に影響を与えずに仕様設計を変更できることを意味しています。 ただし、要件定義と仕様設計を相互から切り離して開発できるというわけではありません。 実際のところ、要件定義が完了したら、情報システムのパフォーマンスなどの純粋な IT 面の検討事項によって技術的な内容が影響を受けないよう考慮しながら、ビジネス管理のトピックを定義する必要があります。

[実装] レベルでは、仕様設計を具体的なハードウェアおよびソフトウェア コンポーネントに変換します。 これにより、IT への関係が構築されます。

記述には、それぞれの更新サイクルが含まれます。 更新頻度が最も低いのは要件定義レベルで、最も高いのは実装レベルです。

実装レベルは、IT の開発と密接に関わります。実装レベルは、IT の技術革新のサイクルが速い結果、絶えず変更されます。

要件定義レベルは、長期的な観点での業務管理アプローチ、および技術的な導入へ向けての後々のステップの開始点となるため、特に重要です。 要件定義は、ビジネス管理の問題と密接に関係するため、最も長いライフサイクルを持ちます。また、情報システムが持つ技術的な利点を記録します。 このような理由から、要件定義、つまり意味論的モデルの開発を取り扱うビューの優先度が最も高くなります。 意味論的モデルは、ユーザーと、ビジネス管理の問題を IT 関連の言語の記述への最初の移行との間の架け橋を形成します。

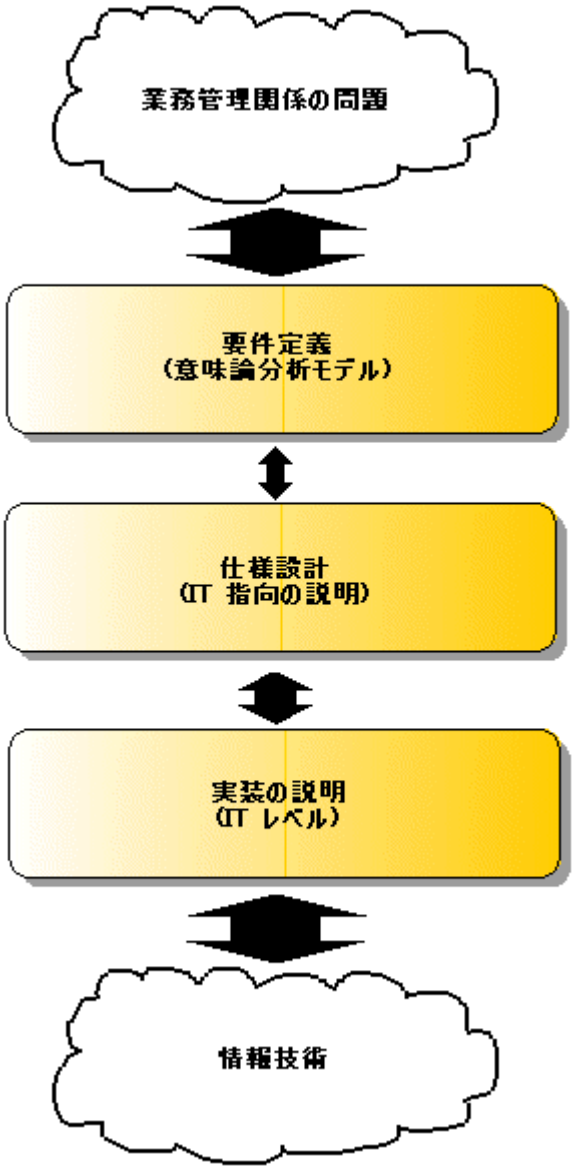


図 4: 情報システムの説明

ARIS 概念の本質は、ビュー、記述、ビジネス管理ソリューションの組み合わせから成ります。 次の図に示すように、すべての説明ビューは、3 つのレベル（[要件定義]、[仕様設計]、[実装]）で記述されます。

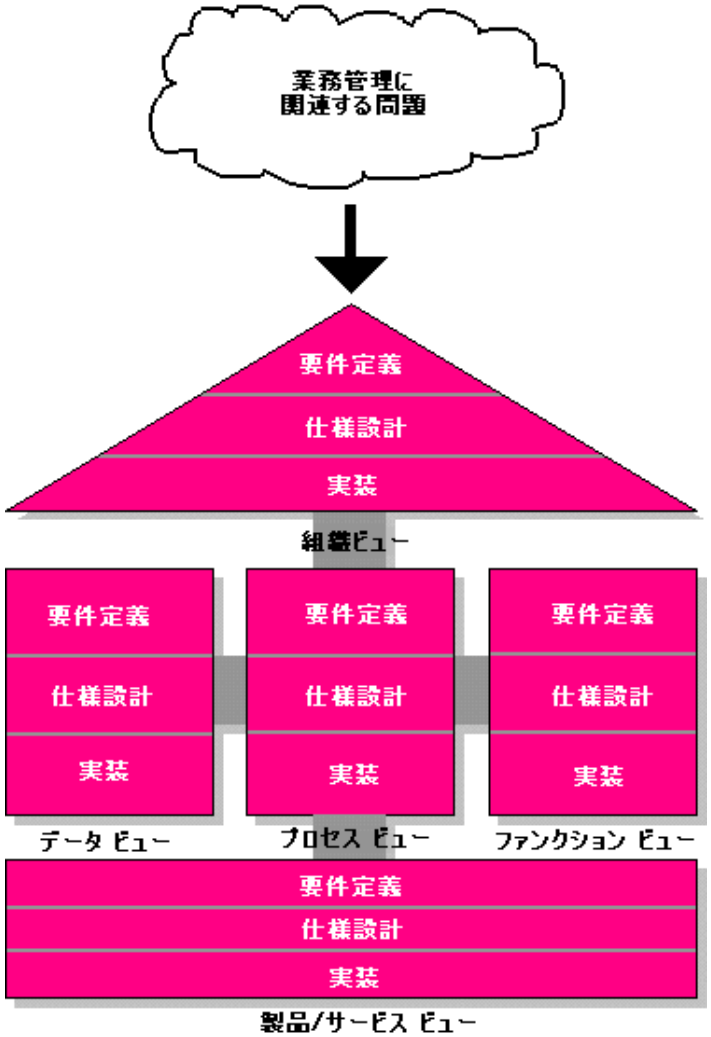


図 5: ARIS の概念

ARIS の概念によって、アーキテクチャの説明ビューと説明レベルで定義されるように、関連するオブジェクトや考慮する領域が総括されます。 この手法の焦点であるビジネス管理の問題を含む、13 の要素から構成されます。 次の手順では、各オブジェクトまたは対象となる領域に適した記述方法を選択して説明することが必要です。

これらの方法を選択する基準は次のとおりです（Scheer 著『Business Process Engineering』(1994 年))。

- 表現方法の単純性と明瞭性
- 表現する内容に対する適合度
- 表現するすべてのアプリケーションに対して、一貫した方法が使えること
- その方法に対する現在、または将来の習熟度
- 通信技術における技術開発からメソッドが独立していること

オブジェクトまたは考慮する領域に適用される各方法については、後の章で説明します。

3 プロセス連鎖分析

3.1 ビジネス管理の問題の説明

各オブジェクトや考慮する領域は、ビジネスの初期状況（ビジネス管理の問題）に基づいて、ARIS アーキテクチャ（ビューおよびレベル）内でモデル化されます。 説明では、既存の業務プロセスに対して提供されるサポートの観点から、現在使用されている情報システムの弱点を挙げ、開発するシステムの目標概念の主な内容も記載します。 一方、目標概念は、新しい情報システムを使用して達成すべき目的を表します。

そのため、ビジネス管理の問題を表現するモデルは、データ ビュー、ファンクション ビュー、組織ビューから、それぞれの相互関係を含め、できるだけ多くの事実情報を記録できる必要があります。 さらにこのモデルを使って、この仕様が、その後のモデリング プロセスの開始点として機能するように目標概念を定義できなければなりません。 ビジネス問題が ARIS の概念で規定されるビューに分割されるのは、要件定義を作成する段階でのみです。

最初のビジネス管理状況は首尾一貫して記述し、また、既存の情報システムの弱点は簡潔に表現する必要があります。このような要件のため、従来のモデリング手法を使用するには限界があります。 表現方法に注目すると、従来のモデリング手法は、個々のビューをモデリングする場合にのみ適しています。

プロセス連鎖図（PCD）を使用すると、ビジネスの問題を簡潔に記述することができると同時に、考慮する情報システムの概要を表すことができます（Scheer 著『EDV-orientierte Betriebswirtschaftslehre』(EDP-oriented business management) 1990 年、39 ページ以降)。

3.2 プロセス連鎖図（PCD）

プロセス連鎖図は、閉じているプロセス連鎖を表します。 業務プロセスのすべてのビュー（組織、データ、ファンクション、リソースの各ビュー）は、相互関係の整合性を維持した形で表現されます。

次の図に、「受注処理」のプロセス連鎖図の例を示します。

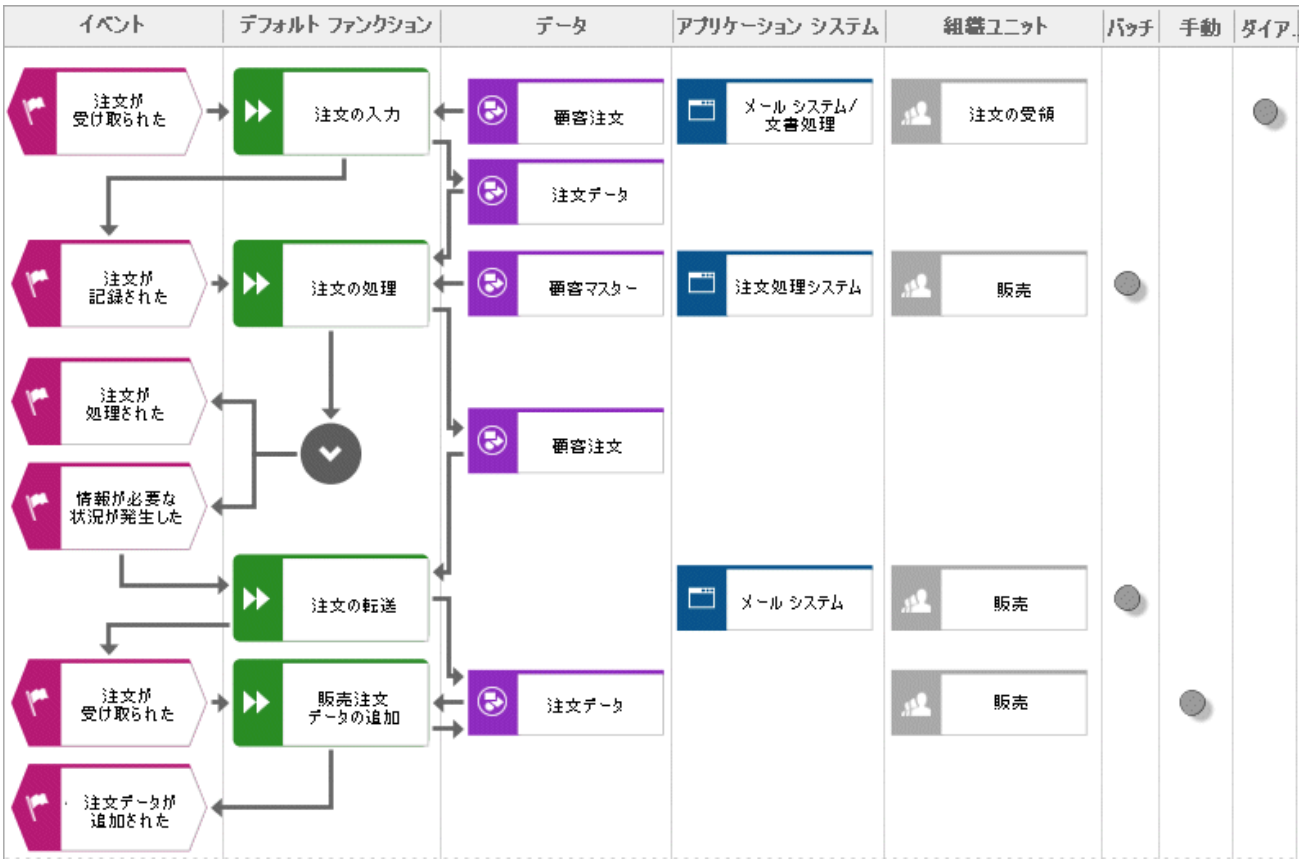


図 6: プロセス連鎖図の例

左側の 2 つの列では、対象の業務プロセスの操作順序が時系列に表されます。 手順の各ファンクションが 2 番目の列に一覧され、それらを起動したり生成したりするイベントに接続されます。 ファンクションとイベントをつなぐ接続線によって、どのイベントがファンクションを開始し、どのファンクションがイベントを作成するのかが明確に定義されます。これによって、ファンクション間の制御フローが調整されます。 上の例では、「注文を受領した」イベントが「注文を入力する」ファンクションを起動します。 このファンクションの結果は、「注文入力完了」終了イベントで定義されます。 このイベントは、「注文を処理する」という次のファンクションを起動します。 イベントとファンクションのこのように接続することで、ファンクションの時系列な操作順序 (プロセス連鎖と呼ぶ) を作成します。 制御フロー内で使用可能な分岐点やループの論理的な依存関係は、ルールによって表現できます。

ファンクションによって要求される入出力データは、次の列にクラスター/データ モデルの形で表されます。 ファンクション「注文を処理する」は入力データとして「注文データ」と「顧客マスタ」を必要とし、出力データとして「受注」を作成します。 情報オブジェクトだけを表示したり、情報が記録された情報媒体 (メディア) も含めて表示したりできます。 情報媒体には、たとえば、文書、リスト、手書きの領収書、あるいはハード ディスクなどのストレージ メディアがあります。

右側の列には、関連するファンクションの実行を担当する組織ユニット (部門) を指定します。

[処理タイプ] (ダイアログ、バッチ、マニュアル) 列と [アプリケーション システム] 列には、ファンクションに対して使用可能な IT サポートの度合いに関する追加情報を入力します。 使用されるアプリケーション システムや、アプリケーション システムのコンポーネントは、[アプリケーション システム] 列に入力します。 [処理タイプ] 列には、ファンクションの実行方法 (ダイアログ、バッチ、またはマニュアル処理) についてより詳細に定義します。

プロセス連鎖図を使用して実際の状況を記述する業務プロセスを分析すると、現在の問題解決の方法の弱点を表示できます。 これらの弱点として、IT ベースの処理と手動処理との間のメディア ブレーク、または組織ブレーク (たとえば、担当部門 (組織) が頻繁に変わるなど) が考えられます。 特に、分析を行うと、プロシージャ内のデータの冗長性、重複、および遅延が明確になります。その結果、ユーザーは定義する目標プロシージャを改善するためのさまざまなアイデアを考え出すことができます。

初期状況を記述する、比較的高度に集約されたプロセス連鎖図が作成されます。 プロセス連鎖図は、主にすべての ARIS 要素間の相互作用を表現するために使われます。また、ARIS のプロセス ビュー内の表現手段でもあります (「プロセス ビュー (コントロール ビュー) 『79 ページ 』」の章を参照)。 プロセス ビューでは、プロセス連鎖図だけでなく、イベント駆動プロセス連鎖 (EPC: Event-driven Process Chain) も使用できます (「イベント制御 - イベント駆動プロセス連鎖 (EPC) 『80 ページ 』」の章を参照)。 イベント駆動プロセス連鎖は、PCD と同じモデリング機能を提供しますが、自由に配置できるため、PDC と異なりオブジェクトはあらかじめ定義された列に配置されません。 プロシージャ モデルが、一貫していずれかのモデル タイプ (PCD または EPC) でサポートされている場合、その目標プロシージャは EPC としても表現できます。

ARIS の概念の説明に続いて、モデリング手法の説明を行います。 まず、ビュー (ファンクション ビュー、データ ビュー、組織ビュー、プロセス ビュー) について説明し、次に、それらのビュー内での説明レベル (要件定義レベル、仕様設計レベル、実装レベル) について説明します。

4 ARIS 概念のビューおよびレベル内のモデリング

4.1 ファンクション ビュー

4.1.1 要件定義

モデリング手法の多くでは、ファンクションをほかの ARIS ビューのオブジェクトのコンテキストにおいて参照することがよくあります。たとえば、データとファンクションの関係によって、入出力データがそのファンクションの変換処理に及ぼす影響が表されます。

一方、ARIS アーキテクチャは、さまざまな考慮する領域を厳密に分離しています (Scheer 著『Architecture of Integrated Information Systems』(1992 年)、62 ページを参照)。 この結果、ファンクション ビュー内では、ファンクション間の相互関係を表す表現手段のみが使われます。 ファンクションとデータの関係は、ARIS プロセス ビューに示されます。

ファンクションとは、1 つ以上のビジネスの目標を支援するために、オブジェクトに対して実行される技術的なタスクまたはアクティビティです (Scheer 著『Architecture of Integrated Information Systems』(1992 年)、63 ページを参照)。

ファンクションは、角の丸い長方形で表します。

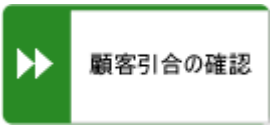


図 7: [顧客引合いの確認] ファンクションの表現

通常、このようなファンクションは、顧客からの引合や生産注文などの情報オブジェクトに基づいて作成されます。 これは、ファンクションの名前で表します。 上の図に、これを示します。「顧客引合」はオブジェクトを定義し、「確認」はこのオブジェクトに対して実行される操作を定義します。 ただし、より上位のレベルでは、ほとんどの場合、名詞がファンクション名に使用されます (たとえば、構内輸送、生産、販売など)。

4.1.1.1 ファンクション ツリー

ファンクションは、さまざまな集約レベルで記述できます。 最上位の集約レベルは、業務プロセスやプロセス連鎖などのファンクションを集めたものにより構成されます。 たとえば、顧客からの引合が来てから出荷までという受注処理を考えます。 このような業務プロセスは複雑なファンクションを表しますが、サブファンクションに分割して複雑さを減少させることができます。「ファンクション」という用語は、すべての階層レベルに対して使用できます。 ただし、階層レベルを示すために、「プロシージャ」、「プロセス」、「下位ファンクション」「基本ファンクション」といった用語もまた使用されます。

複数の階層レベルに渡って、ファンクションを細分化することができます。 基本ファンクションは意味論的ファンクション ツリーの最下位レベルを表します。

基本ファンクションとは、ビジネス管理の観点からこれ以上分割できないファンクションです。

階層構造は、ファンクション ツリーまたは階層モデルを使用して最適に表すことができます。

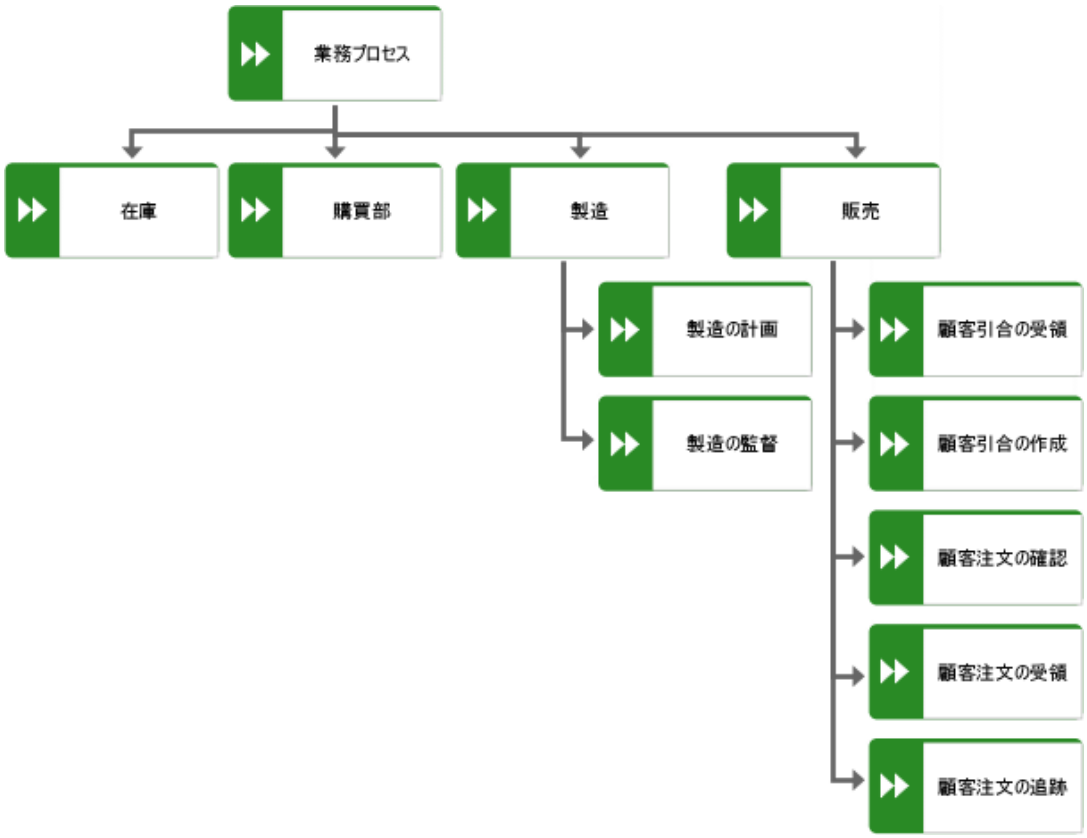


図 8: ファンクション ツリー (抽出)

ファンクション ツリー内では、さまざまな条件に基づいてファンクションをグループ化できます (Brombacher/Bungert 著『Praxis der Unternehmensmodellierung』(Enterprise modeling practise)(1992 年) を参照)。この目的で頻繁に使用される基準には、同じオブジェクトを処理する (オブジェクト指向)、属するプロセスに基づいて分割する (プロセス指向)、同じ操作に基づいてファンクションをグループ化する (処理指向) という基準があります。

次の図は、オブジェクト指向に基づいてファンクションを分割した例を示しています。 上位のファンクション「生産注文を処理する」は、「生産注文を作成する」、「生産注文を更新する」、「生産注文をキャンセルする」、「生産注文をリリースする」、「生産注文を確認する」、「生産注文を監視する」の各ファンクションに分割されます。 これらのファンクションは、「生産注文」という 1 つの同じオブジェクトに対して実行されるさまざまな操作 (作成、更新、キャンセルなど) を記述しています。

図 9: オブジェクト指向のファンクション ツリー

業務プロセスのモデリング結果を表すファンクション ツリーには、プロセス指向のファンクション ツリーによる表現が適しています。 次の図は、プロセス指向に基づいてファンクションを分割した例を示しています。

「顧客注文を受理する」、「顧客注文を検証する」、「顧客データを作成する」、「顧客のクレジットの有効性を確認する」、「製品の在庫を確認する」、および「顧客注文を確認する」の各ファンクションは、業務プロセスの「顧客注文を処理する」の一部です。 オブジェクト指向による分割とは対照的に、異なるオブジェクト (「顧客注文」、「製品の在庫」など) に対して実行される操作が分割の基準になります。

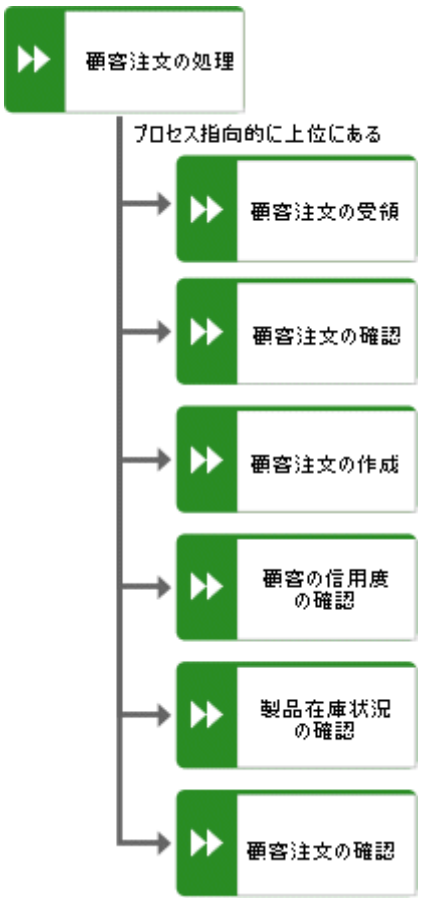


図 10: プロセス指向のファンクション ツリー

処理を基準にしたグループ化とは、個々の情報オブジェクトに対して同じプロセス (たとえば、「確認」、「作成」、「削除」など) を実行するファンクションをグループ化することです。 次の図に、「更新」操作の例を示します。 図中の各ファンクションは、異なるプロセスに含まれていても、異なるオブジェクトの処理に含まれていてもかまいません。 ただし、異なるオブジェクトに対して実行される操作の種類は常に同じです。

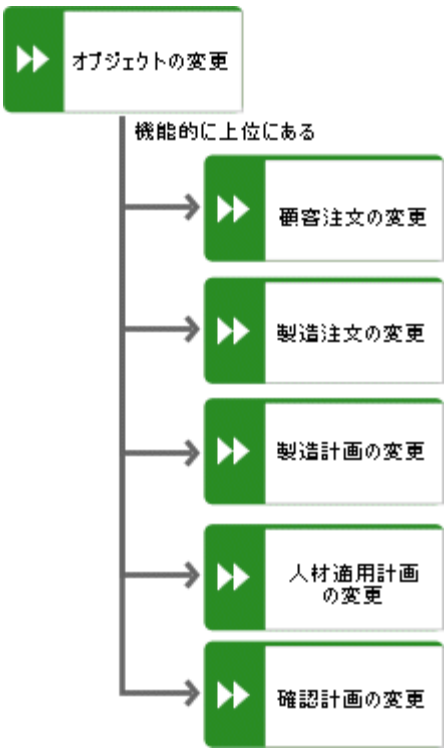


図 11: 操作指向のファンクション ツリー

ファンクションをファンクション ツリーに表示すると、複雑さは軽減されますが、これは静的な記述です。 この静的な記述に加えて、ファンクションがどのような順序で処理されるかにも焦点が当てられます。 時系列的な操作順序は、イベント駆動連鎖図 (EPC) で表します。 これらには、ファンクションだけでなく、ファンクションに接続するイベントも含まれます。 イベントは ARIS のデータ ビューに属します。 ARIS (「要件定義 『20 ページ 』」の章を参照) で規定されるビューの分割原則に従って、イベント駆動プロセス連鎖は ARIS のプロセス ビューに記述されます。

要件定義関連の視点からファンクションを記述する場合には、ファンクションをいくつかのサブファンクションに分割する原則だけでなく、業務プロセスの設計に影響を与えるほかのファンクション プロパティも考慮します。

たとえば、ファンクションには、ファンクションの実行にユーザーが関与するかどうかに関する情報を常に含めることを推奨します。 自動的に実行できる類似した種類のファンクションは、バッチを実行してまとめて処理できます。

たとえば、1 日に処理される引合の数などのファンクションの量構造に関する情報と、ファンクションの実行にかかる時間の合計によって、業務プロセスを再構成するために行う決定の基礎として利用できる追加データが得られます。 合計時間は、さらに個別の時間単位 (学習時間、処理時間、待機時間) に分割できます。 ARIS では、この情報は [ファンクション] オブジェクト タイプの属性に保存できます。 使用可能なすべての属性タイプの一覧は、インストール媒体に含まれている『ARIS メソッド マニュアル - テーブル』マニュアル (ARIS Method tables.pdf) を参照してください。

4.1.1.2 Y 図

Y 図は、企業の業務を高度に集約したレベルで表すために使用します。 Y 図には、製品設計、原材料管理、保守など包括的な業務範囲が含まれています。 Y-CIM モデル (Scheer 著『Business Process Engineering』(1994 年) を参照) 形式の構造化表現は、各ファンクションの分類を表します。 シェアー博士は、生産計画および生産管理のビジネス管理計画ファンクションを Y の左側の枝に配置しました。右の枝は、生産計画および実現の技術指向のファンクションを含んでいます。 計画ファンクションは、Y の上部セクションに配置され、管理および実装ファンクションは下部セクションに配置されています。

このように、Y-CIM モデルは、生産関連企業のすべてのファンクションを分類する枠組みとなります。

ARIS では、Y-CIM モデルは、ファンクション指向の手法から複雑なリファレンス モデルまでを表すことができます。 図に示したオブジェクトは、[ファンクション] オブジェクト タイプです。 階層を構成すると、このオブジェクト タイプを [ファンクション ツリー] や [EPC] のようなモデル タイプに接続させることができます。

次の図に例を示します。

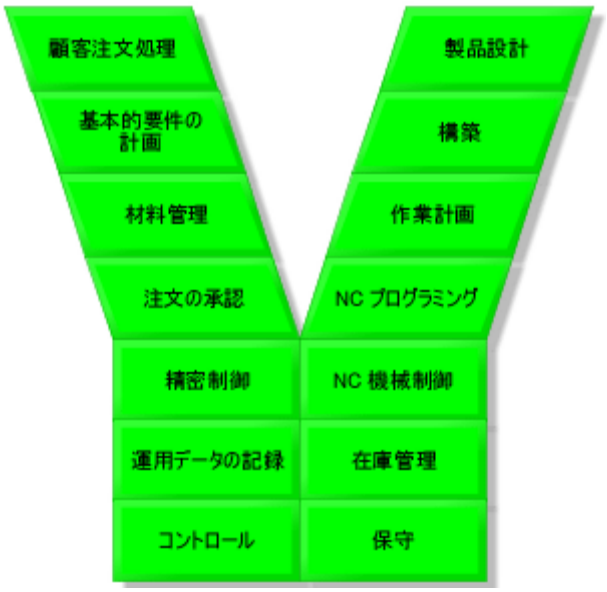


図 12: Y 図

4.1.1.3 SAP アプリケーション図

SAP アプリケーション図によって、 SAP R/3 アプリケーション システム モジュールに適合する SAP R/3 リファレンス モデルへの手法が提供されます。 R/3 リファレンス モデルでは、このモデル タイプの各オブジェクトにプロセス選択マトリクスがアサインされます。 マトリクスは、関連する R/3 モジュールで利用可能なメイン プロセス、および図示できるプロセス シナリオを表します。

次の図は、SAP R/3 システムが提供する SAP アプリケーション図を示しています。

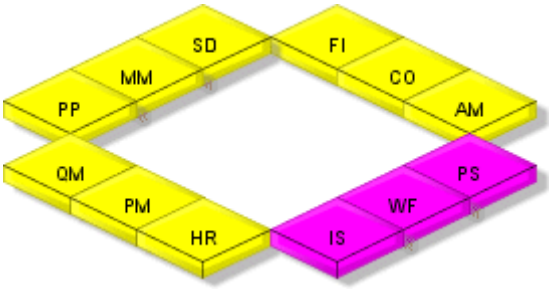


図 13: SAP アプリケーション図

4.1.1.4 目標図

業務プロセスのモデリング、分析、または最適化 (業務プロセス リエンジニアリング) を行う前に、自社の業務プロセスのモデリングの目的を定義しておく必要があります。

目標図では、会社の (業務) 目標を定義して、目標の階層などに配置することができます。

目標とは、成功要因を促進し、新しい業務プロセスを実装することによって達成される将来の業務目標です。

目標を達成するために重要な要素を指定して、階層に配置して、これらによって達成が支援される目標に割り当てることができます。

成功要因とは、特定の業務目標を達成するために考慮する必要がある側面です。 重要成功要因は、目標図では業務目標に割り当てます。

このモデル タイプは、[ファンクション] オブジェクト タイプによって、要件定義のほかのモデル タイプに接続されます。 目標の達成に貢献するファンクション (業務プロセス) は、すべての目標に関して表示することができます。 業務プロセスのモデリングおよび最適化の段階で、プロシージャ モデルを確立する際に目標と割り当てられたファンクションの優先度を考慮することも必要です。

次の図に、目標図の例を示します。

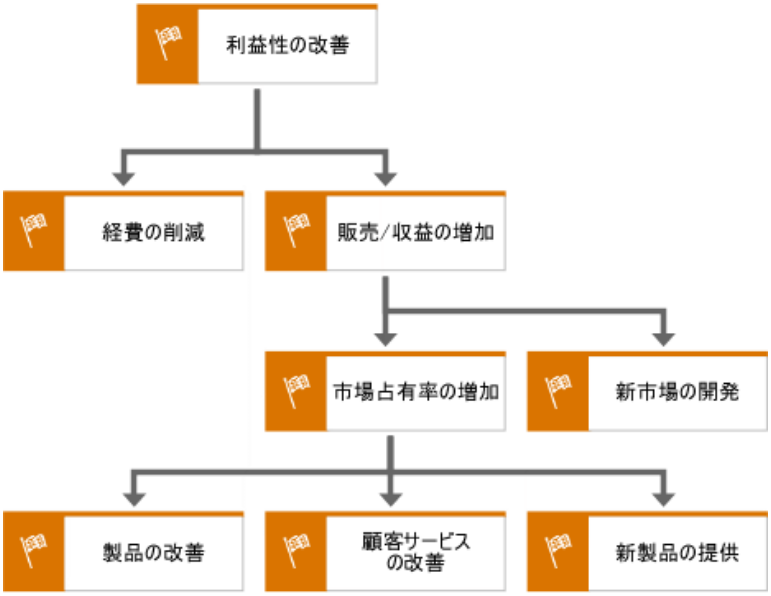


図 14: 目標図

4.1.2 仕様設計 - アプリケーション システム タイプ図

ファンクション ビューの仕様設計には、アプリケーション システムとモジュール タイプの仕様、アプリケーション システム タイプのモジュール構造、各トランザクション手順の概要、および一覧と画面の設計のドラフトを表す入出力の定義が含まれます。

ファンクション ビューの仕様設計が回答となる主要な質問には、次の質問があります:

- アプリケーション システム タイプ、モジュール タイプまたは IT ファンクションが、どのようにして要件定義で定義されるファンクションをサポートするか。
- アプリケーション システム タイプのモジュール構造またはモジュール タイプは何か。
- ファンクションを処理するために、どのような一覧および画面が必要か。
- アプリケーション システム タイプまたはモジュール タイプを使用してどの一覧が作成されるか。また、どの画面がアプリケーション システム タイプおよびモジュール タイプによって使用されるか。
- アプリケーション システム タイプが基礎とする技術 (オペレーティング システム、ユーザー インターフェイス、またはデータベース管理システム) は何か。
- 特定のアプリケーション システム タイプを使用することによって、どのような業務目標達成が支援されるか。

[アプリケーション システム タイプ] は、ファンクション ビューの仕様設計における重要なオブジェクト タイプです。

ファンクション ビューの実装レベルでのみ関係してくる具体的なアプリケーション システムや、企業内で (ライセンス番号などにより) 識別可能な特定のアプリケーション システムを表す具体的なアプリケーション システムとは異なり、アプリケーション システム タイプは、同じ技術に基づくアプリケーション システムをすべて類型することよって作成されます。

アプリケーション システム タイプとは、同一の技術に基づいた個々のアプリケーション システムを類型化したものです。

例: ARIS Architect は、アプリケーション システム タイプです。 このアプリケーション システム タイプのライセンスを複数購入すれば、それぞれのアプリケーション システムを所有することができます。

アプリケーション システム タイプは、次のグラフィック シンボルで示します。

図 15: アプリケーション システム タイプのグラフィック

多くの場合、アプリケーション システム タイプは、モジュール構造を持ちます。 アプリケーション システム タイプ図により、そのモジュール構造を図示することができます。 アプリケーション システム タイプはモジュール タイプに分割できます。 次の図に例を示します。

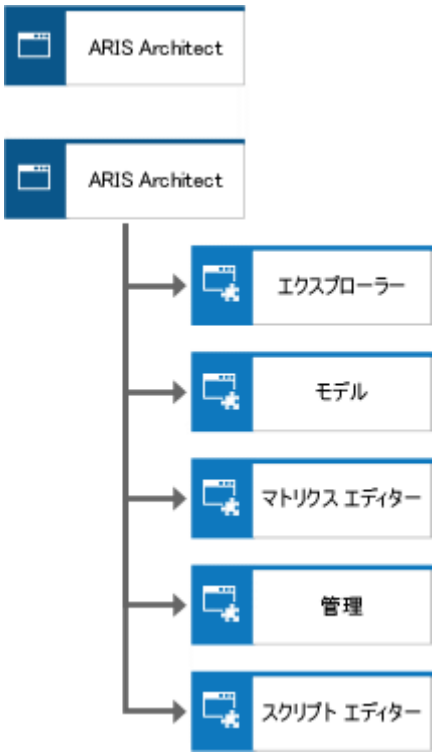


図 16: アプリケーション システム タイプのモジュール構造

上の例では、ARIS Architect は、エクスプローラー、モデル、マトリクス、管理、スクリプト エディターのモジュール タイプから構成されます。 アプリケーション システム タイプの場合と同様、モジュール タイプは同じ技術に基づいた個々のモジュールを類型化したものです。 モジュール タイプは、アプリケーション システム タイプのコンポーネントで、 独立した操作を行うことができます。

モジュール タイプは、独立した操作を実行できるアプリケーション システム タイプのコンポーネントです。 モジュール タイプは同じ技術に基づいた個々のモジュールを類型化したものです。

アプリケーション システム タイプおよびモジュール タイプは、任意の階層に配置できます。 モジュール タイプは、最下位層で IT ファンクション タイプに分割されます。

IT ファンクション タイプは、トランザクションという点で、モジュール タイプの最小単位です。 IT ファンクション タイプは、個別のプログラム モジュールとして実現されます。個々の作業ステップを処理するには、IT ファンクション タイプを常に完全に実行する必要があります。

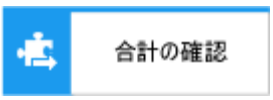


図 17: IT ファンクション タイプのグラフィック

アプリケーション システム タイプ図でも、指定されたアプリケーション システム タイプおよびモジュール タイプによってサポートされる要件定義のファンクションを定義できます。 この割り当てによって、ファンクション ビューの要件定義と仕様設計が関係付けられます。 次の図に例を示します。

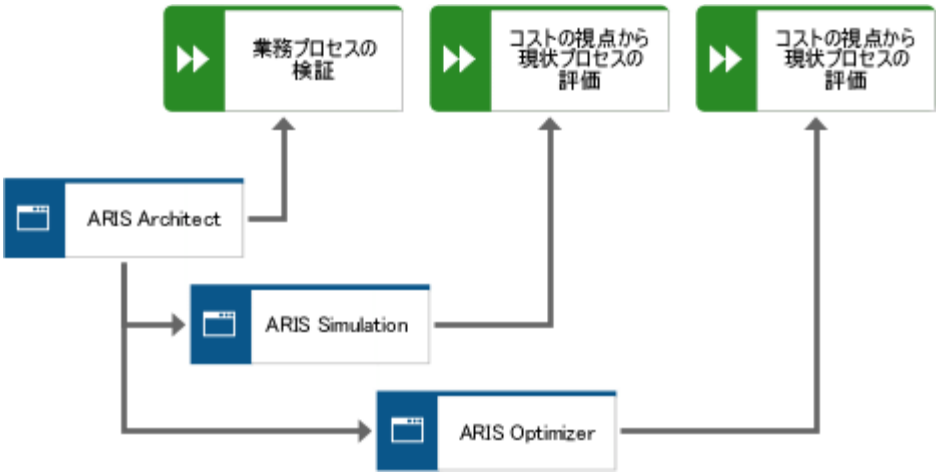


図 18: アプリケーション システム タイプへのファンクションの割り当て

アプリケーション システム タイプおよびモジュール タイプの基盤となる技術のより詳細な仕様を定義するために、ユーザー インターフェイス タイプ、データベース管理システム、使用可能なオペレーティング システム、およびそれらを導入するために用いられるプログラミング言語を割り当てることができます。 ここでは、具体例ではなくタイプが対象なので、複数の関係が可能です。たとえば、アプリケーション システム タイプに Windows 7 および Windows 8 のユーザー インターフェイスの両方を割り当てることができます。これは、このバージョンのアプリケーション システム タイプは、両方のユーザー インターフェイスで動作することを意味します。 グラフィカル ユーザー インターフェイスが、ファンクション ビューの実装レベルの具体例 (つまり、特定のアプリケーション システム) に割り当てられて初めて、一意の関係が必要になります。 この関係は、その企業が購入したアプリケーション システム タイプのライセンスの実際の構成を表します。

次の図に、可能な割り当ての例をアプリケーション システム タイプ図で示します。

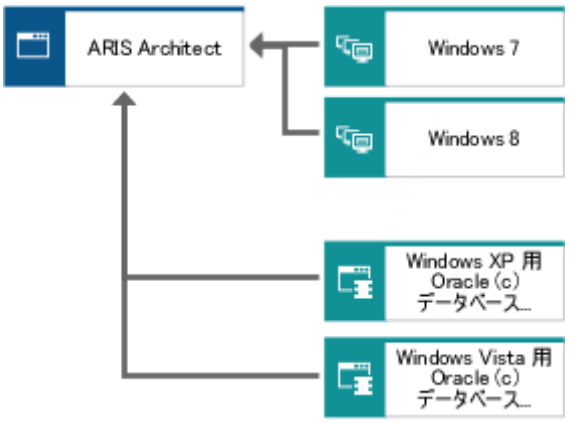


図 19: アプリケーション システム タイプの設定

アプリケーション システムを使用した技術的なファンクションの処理には、さまざまな画面の使用、および該当するアプリケーション システムによって提供される一覧の作成や使用が伴います。 これには、[一覧] と [画面] オブジェクトを使用することができます。これらのオブジェクトは、技術ファンクション、またはアプリケーション システム タイプとモジュール タイプのいずれかに割り当てることができます。

最初の段階で、特定のアプリケーション システム タイプを参照しないで一般的な操作プロシージャを定義する場合、[リスト設計] オブジェクトおよび [画面設計] オブジェクトを使用して、必要な画面と一覧を指定できます。 どちらのオブジェクト タイプも、まず、特定のアプリケーション システム タイプの一覧や画面を参照しないで、どのタイプの一覧または画面 (「顧客データの入力」など) を一般的に使用するかを指定します。 その後、これらの一覧と画面の設計を特定の一覧と画面に割り当てることができます。 既存の割り当てで、使用できる実装方法を指定できます。 次の図に例を示します。

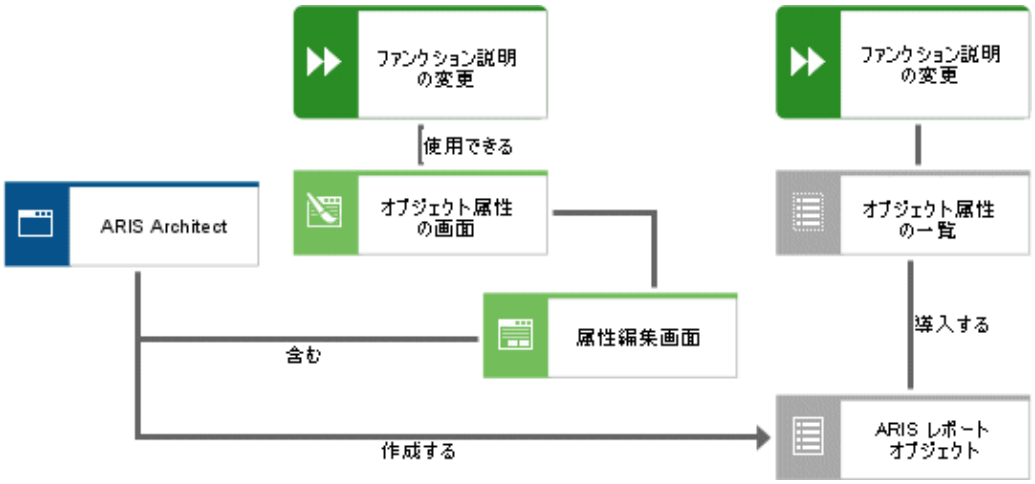


図 20: 画面と一覧の割り当て

アプリケーション システム タイプ図で使用可能なオブジェクトタイプと関係の一覧は、インストール媒体に含まれている『ARIS メソッド マニュアル - テーブル』マニュアル (ARIS Method tables.pdf) を参照してください。

4.1.3 実装 - アプリケーション システム図

アプリケーション システム タイプ図では、具体的なアプリケーション システムおよびモジュールを、仕様設計に記述されているアプリケーション システム タイプおよびモジュール タイプに割り当てることができます。 アプリケーション システムは、企業のアプリケーション システム タイプの具体例で、たとえばライセンス番号などで一意に識別できます。

アプリケーション システム (モジュール) は、アプリケーション システム タイプ (モジュール タイプ) の具体例であり、ライセンス番号などによって一意に識別できます。

アプリケーション システムとモジュールは次のようにグラフィック表示されます。

図 21: アプリケーション システムとモジュールのグラフィック

企業では、1 つのアプリケーション システム タイプ (モジュール タイプ)のライセンスを複数所有していることがあります。したがって、アプリケーション システム図では、1 つのアプリケーション システム タイプ (モジュール タイプ)に複数のアプリケーション システム (モジュール) を割り当てることができます。

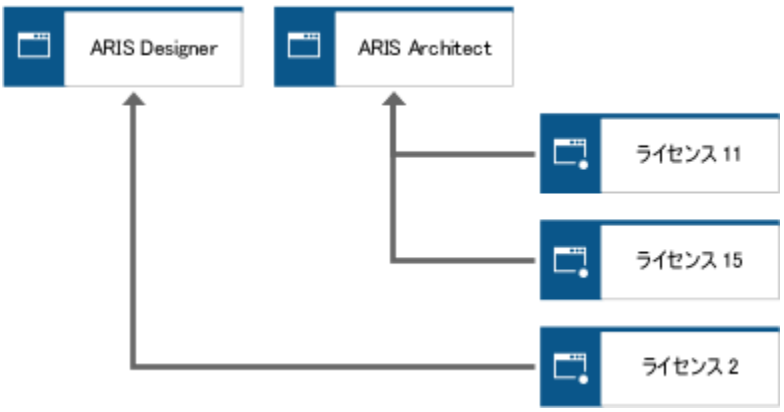


図 22: アプリケーション システム タイプへのアプリケーション システムの割り当て

アプリケーション システム図は、アプリケーション システムの実際のモジュール構造を表します。 仕様設計では 1 つのアプリケーション システム タイプに含めることができるすべてのモジュール コンポーネントが示されますが、ここでは、ライセンスごとにモジュール コンポーネントを一意に定義することができるように、1 つのアプリケーション システムのライセンスについて扱います。 したがって、企業では、アプリケーション システム タイプが同じでモジュール構造が異なる複数のアプリケーション システムを所有することができます。

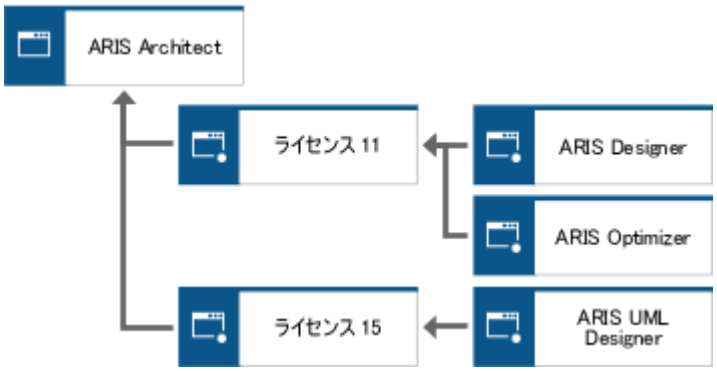


図 23: 同じタイプの 2 つのアプリケーション システムが異なるモジュール構造を持つ例

実装レベルでは、すべての既存のアプリケーション システムとモジュールを示すことができるだけでなく、アプリケーション システムを個別のプログラム ファイルの形式を使用して技術的 (物理的) な実装を定義できます。

アプリケーション システム タイプ図では、アプリケーション システム タイプやモジュール タイプを導入するために必要なプログラム モジュール タイプを図示します。

プログラム モジュールとは、記憶媒体上のプログラム ファイル (たとえば、EXE ファイル、COM ファイルなど) であり、ライセンスの購入によって取得されます。 プログラム モジュール タイプは、同一の技術に基づくプログラム モジュールを類型化することによって作成されます。

次の図は、プログラム モジュール タイプのアプリケーション システム タイプへのアサインメント、および個々のプログラム モジュールのプログラム モジュール タイプへのアサインメントを表しています。

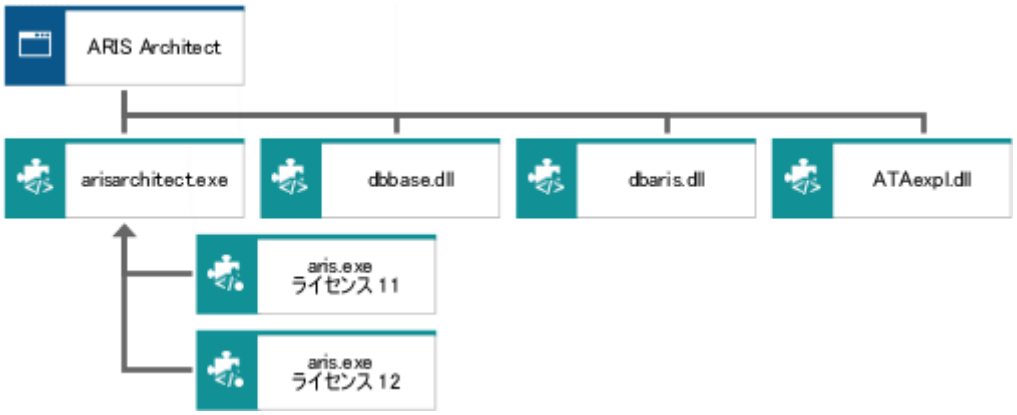


図 24: アプリケーション システム タイプ、プログラム モジュール タイプ、およびプログラム モジュールの割り当て

ARIS Architect アプリケーション システム タイプのプログラム モジュール タイプには、arisarchitect.exe、dbbase.dll、abaris.dll、ATAexpl.dll があります。 複数のライセンスを購入したり、バックアップ コピーを作成したりした場合、企業内に各プログラム モジュール タイプのコピー (プログラム モジュール) が複数存在する場合があります。

プログラム モジュール タイプおよびプログラム モジュールは任意の階層に配置できます。 プログラムの技術的な仕様をより正確に定義するために、アプリケーション システム タイプ図に、プログラム モジュール タイプのプログラム ライブラリへのアクセスを表示することもできます。

アプリケーション システム タイプ図で使用可能なオブジェクトタイプと関係の一覧は、インストール媒体に含まれている『ARIS メソッド マニュアル - テーブル』マニュアル (ARIS Method tables.pdf) を参照してください。

4.2 データ ビュー

4.2.1 要件定義

データ ビューの要件定義には、対象となる領域の意味論的データ モデルの記述が含まれます。 ARIS に規定される分割アプローチに従って、ここには、プロセス連鎖の開始イベントと終了イベントを指定するオブジェクトと、関連するプロセス連鎖環境の現在状態の説明の両方が含まれます。

ファンクションのモデリングと違い、データのモデリングは手法に関するかぎり特に要求が厳しくなります。 ファンクション ビューでは、調査対象のオブジェクトはファンクションだけです。 さらに、ファンクション間の関係は上位・下位関係のみが示されます。

Chen が開発した ERM (実体関連モデル) は、意味論的データ モデルの設計手法として広く利用されています (Chen 著『Entity-Relationship Model』(1976 年) を参照)。 このモデリング手法では、エンティティ タイプ、関係タイプ、属性などさまざまな特殊用語を使います。これらのオブジェクト間には多くの関係が存在し、ファンクション モデリングに比べると分類がかなり難しくなっています。

次に、ERM のモデリング手法を紹介します。 最初に、Chen のベース モデルのオブジェクトとそれらの関係を説明します。 次に、別の章で後からベース モデルに追加された複数のルールについて説明します。

4.2.1.1 ERM ベース モデル

ベース モデルでは、エンティティ、属性、関係を区別します。 基本的に、タイプ レベルとオカレンス レベルが区別されます。

[エンティティ] とは、検証中の業務管理タスクに関連する実際のオブジェクト、または抽象オブジェクトです。

たとえば、検討するオブジェクトには、業務プロセスなどがあります。 ARIS の分割モデルに従うと、対象であるデータ オブジェクトは、環境のオブジェクトとイベントを規定するオブジェクトです。「顧客注文の処理」プロセスには、たとえば次のようなエンティティがあります。

- 顧客 1235
- 品目 471
- 受注 11

エンティティは、特定の属性 (プロパティ) によってさらに正確に定義できます。 たとえば、姓、名、住所によって、顧客をより正確に定義できます。

類似のエンティティをグループ化したものを、エンティティ タイプと呼びます。エンティティ タイプの個々のオカレンスがエンティティです。

類似した種類のエンティティは、同じ属性で定義できます。 たとえば、顧客「田中」と顧客「鈴木」は、共にエンティティ タイプ「顧客」に含まれます。品目「4710」と品目「4712」は、エンティティ タイプ「品目」に含まれます。 ERM では、エンティティ タイプは長方形で表されます (次の図を参照)。 次のグラフィックに例を示します。

図 25: エンティティ タイプの例

属性とは、エンティティ タイプを定義する性質です。

属性オカレンスとは、個々のエンティティが持つ属性の具体的な値のことです。 たとえば、顧客「1235」は、「鈴木」、「健一」、「東京都」などの属性オカレンスによって説明できます。 これに関連する属性は「顧客の姓」、「顧客の名」、「市町村」です。

属性は、通常楕円または円で表します。 以降、属性は楕円形で表します。 次の図は、エンティティ タイプ**「顧客」**に対する属性の例を表しています。

図 26: [顧客] エンティティ タイプの属性の例

エンティティ タイプと属性は区別しにくいことがよくあり、モデリング手順のコンテキストからしか判断できない場合あります。 たとえば顧客の住所は、「顧客」の属性ではなくエンティティとして捉えることもできます。 その場合は、「顧客」との関係を所有する、別のエンティティ タイプ「住所」が作成されます。 エンティティには属性が含まれますが、属性には独自の属性が含まれないことを、エンティティ タイプと属性のどちらであるかを見極める有効な判断基準として利用できます。 ERM で作成され、あとからその他の属性で説明される属性は、エンティティ タイプになります。 オブジェクトがほかのエンティティ タイプと関係を持つかどうかも有効な判断基準です。 オブジェクトがほかのエンティティ タイプと関係を持つ場合もまた、そのオブジェクトはエンティティ タイプです。

関係とは、エンティティ間の論理的な関係です。

したがって、関係が存在するかどうかは、まさに、エンティティが存在するかどうかによります。

類似する関係の集合を、関係タイプと呼びます。

たとえば、「納入業者」と「部品」の間の関係タイプは [提供する] です。 以後、本文中では、関係タイプも太字で表記します。 ERM では、関係タイプは菱形で表され、接続線によってエンティティ タイプに連結されます (次の図を参照)。

図 27: 関係タイプの例

多くの場合、関係タイプ名は一方向のみ読み取りが可能な接続線になります。 上の例では、「納入業者」は「部品」を「提供する」という関係を表しています。 反対方向に読むと、これは「部品」が「納入業者」を「提供する」ことになり、適切ではありません。 正しい方向を一意に決定できない場合は、用語を慎重に選択する必要があります。

関係タイプはいくつかの種類に分類できます。 分類基準は、関係タイプが連結しているエンティティ タイプの数と、関係の複雑度です。

エンティティ タイプの間には、単項、二項、または n 項の関係が存在できます。

複雑度または多重度とは、あるエンティティ タイプのエンティティが、ほかのエンティティ タイプのエンティティに、いくつ関連付けられているかを表します。

次の図に、関係の分類を示します (Scheer 著『Business Process Engineering』(1994 年)、34 ページを参照)。

関係は、多重度の異なる次の 4 つの種類に分類できます。

- 1:1 関係
- 1:n 関係
- n:1 関係
- n:m 関係

1:1 関係では、第 1 集合の各エンティティが、第 2 集合の 1 つのエンティティに割り当てられます。

1:n 関係では、第 1 集合の各エンティティが第 2 集合の 1 つのエンティティに関連付けられますが、第 2 集合の各エンティティは第 1 集合さまざまなエンティティに関連付けられます。

n:1 関係は、ちょうどその逆の状況を表します。

n:m 関係では、第 2 集合の複数のエンティティが、第 1 集合のすべてのエンティティに関連付けられます。または、その逆です。

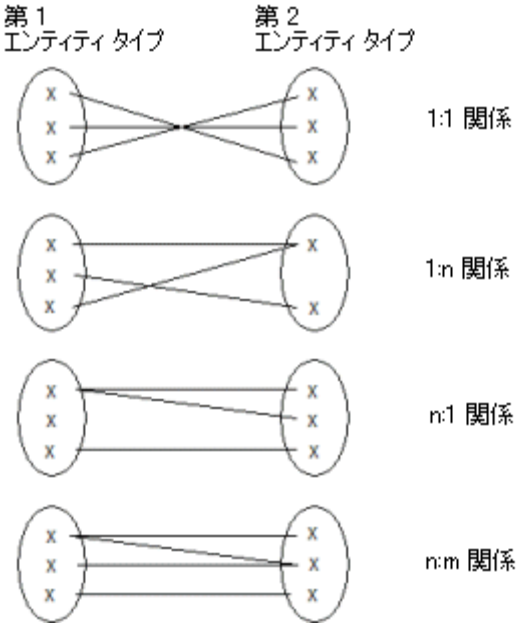


図 28: 2 つのエンティティ タイプ間の関係の多重度

関係タイプの多重度 ([多重度] 属性タイプ) は、実体関連モデルの接続線に表します。

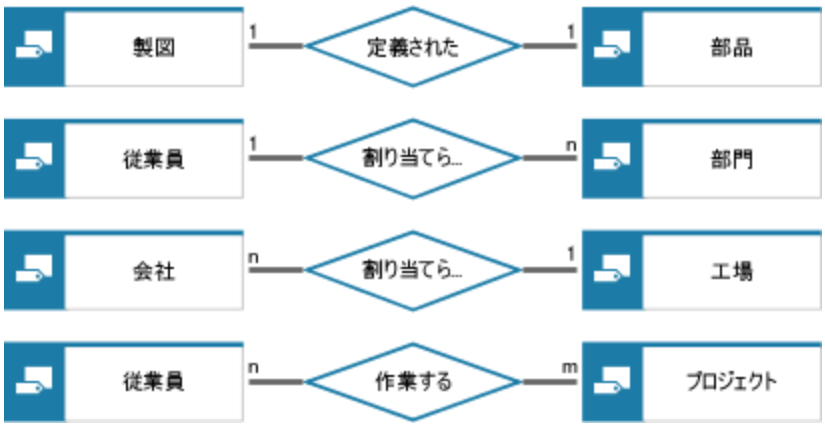


図 29: ERM における多重度

多重度が指定するのは、あるエンティティ タイプのエンティティが持つことができる特定の関係タイプの関係の最大数です。 上の図に示されている n:1 関係は、企業が複数の工場で構成されているため、「企業」エンティティ タイプの企業は複数の [割り当てられる] 関係を持つことが可能であるということを意味します。一方、各工場は最大 1 つの [割り当てられる] 関係を持つことが可能であり、一意に企業に割り当てられます。

Chen のオリジナル モデルは、多重度を別の方法で説明します。 ただし、このマニュアルで使用されている表記を使うと、特に複数のエンティティ タイプ間の関係を表す場合に、より明確な記述ができます。 不要な混乱を避けるために、Chen のオリジナル モデルについては、ここでは詳しく取り扱いません。

1 つのエンティティ タイプのエンティティ間で関係を持つことが許可されているので、1 つのエンティティ タイプと関係タイプの間には 2 本の接続線が並列して存在することがあります。 これらの接続線は役割名を割り当てることにより区別することができます。 再帰的な関係の例を次の図に示します。 上位の部品は、さまざまな下位の部品から構成されます。 下位の部品は、さまざまな上位の部品の構成要素として使われます。

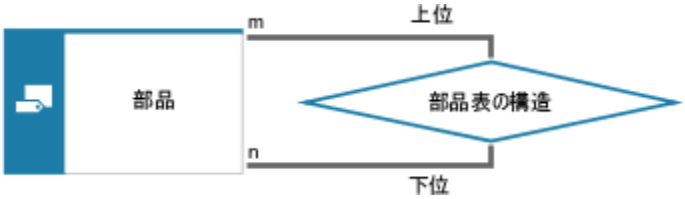


図 30: 部品一覧の ERM

エンティティ タイプと関係タイプの両方を、属性によって説明できます (次の図を参照)。

属性の値の範囲をドメインと呼びます。

エンティティ タイプまたは関係タイプの要素に対するドメイン要素の割り当ても関係であり、適切に付けられた名前の接続線により識別することができます。

エンティティ タイプと、少なくとも 1 つのドメインとの間に 1:1 関係が存在する必要があります。 このドメインの値によって、個々のエンティティを一意に識別できます。 この関係を満たすドメイン名をそのエンティティ タイプのキー属性と呼びます。

次の図の例では (Scheer 著『Business Process Engineering』(1994 年)、33 ページを参照)、「顧客」のエンティティは、[顧客番号] キー属性によって一意に識別できます。

関係は、連結されているエンティティのキー属性をマージして識別します。 したがって、[居住する] 関係タイプのキー属性は、「顧客番号」と「住所番号」です。

関連のあるデータ オブジェクトの記述属性は、エンティティ タイプまたは関係タイプと 1:n 関係を持つドメインから導き出された値によって定義されます。

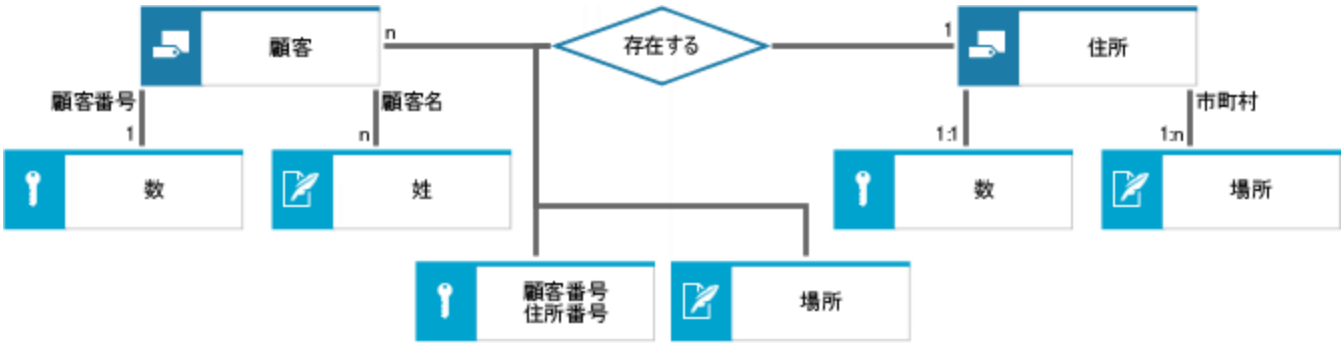


図 31: ERM における属性の割り当て

4.2.1.2 ERM - eERM 拡張

この数年間で、Chen のベース モデルは大幅に拡張されました。 このマニュアルでは、これらの拡張の中で、ARIS 概念のデータ ビューのモデリングに重要なものだけを説明します。

4.2.1.2.1 設計演算子の追加

設計演算子は、データ モデルの作成を正式にサポートします。 設計演算子を使用することによって、データ構造が体系的に作成され、既存のデータ構造を設計プロセスの視点から見ることができます。 設計演算子を活用すると、既存の用語に基づいて新しい用語が開発されます。 このプロセスは、ビジネス管理の知識レベルで実行される知的な手順です。 データ構造の観点からビジネス管理の事実を調査することによって、既知の構造が新しい手法に従って変更されたり、まったく新しい結論が導かれます。

ERM モデリングの手法を拡張するためのさまざまなアプローチの中では、次の 4 つの設計演算子が広く使用されています (Scheer 著『Business Process Engineering』(1994 年)、35 ページ以降を参照)。

- 分類
- 汎化
- 集約
- グループ化

分類

分類によって、同じタイプのオブジェクト (エンティティ) が識別され、1 つの用語 (エンティティ タイプ) に割り当てられます。 2 つのオブジェクトを記述するために同一のプロパティ (属性) が使用されている場合、その 2 つのオブジェクトは同一です。

分類は、以前に記述されたエンティティ タイプを識別します。

図 32: 「顧客 (CUSTOMER)」の分類

汎化 / 特化

汎化は、類似のオブジェクト タイプを 1 つの上位オブジェクト タイプの元にまとめることを意味します。

次の図に示すように、「顧客」エンティティ タイプと、「納入業者」エンティティ タイプは、上位概念「取引先」の下に汎化されます。 両方のソース オブジェクトに共通の性質 (属性によって説明される) は、汎化されたオブジェクト タイプに移行されます。これにより、ソースのオブジェクト タイプで異なっている属性だけが残ります。これを記述します。 新しいエンティティ タイプ「取引先」の構造は、「である」関係とも呼ばれる三角形で図示されます。

特化とは、一般的な概念を下位概念に分割することです (「取引先」は「顧客」と「納入業者」に分割されます)。

特化は汎化の逆です。 特化されたオブジェクトは、汎化されたオブジェクトの性質を継承します。 継承されたこれらの属性のほかに、特化されたオブジェクト タイプは固有の属性を持ちます。 特化も汎化も図では同じ表現をします。

このため、図中の接続線には方向を表す矢印を描きません。

図 33: 汎化/特化

特化は、主にトップダウンの方法でデータ モデルを作成する場合に使用されます。一方、汎化はボトムアップの方法で使われます。

特化の範囲内では、サブセットを作成する際に、その完全性と分離 (代理) を定義できます。

分離されないサブセットは、オブジェクトのオカレンスが複数のサブセットに存在することができるという特徴があります。 上の例では、顧客が同時に納入業者である可能性もあります。 エンティティが 1 つのサブセットにしか割り当てられない場合は、これらの集合は分離されています。

1 つの汎化オブジェクト タイプについて、ある特化基準で特化できるすべての特化オブジェクト タイプが列挙されている場合、それを完全な特化と呼びます。 たとえば、「人間」エンティティ タイプを特化すると、「女性」および「男性」エンティティ タイプを挙げることができます。 これは、性別という基準における完全な特化です。

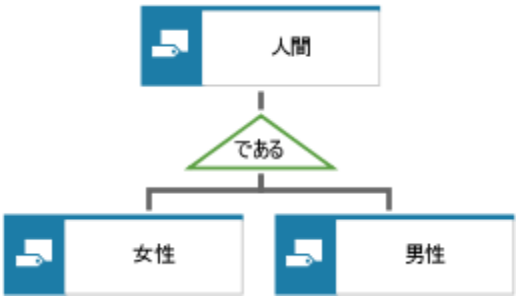


図 34: 完全な特化

これらの基準を組み合わせることによって、汎化/特化は次の 4 種類に分類できます。

- 分離/完全
- 分離/不完全
- 非分離/完全
- 非分離/不完全

集約

集約とは、既存のオブジェクト タイプを組み合わせて新しいオブジェクト タイプを構成することです。 新しいオブジェクト タイプには、新しい性質を追加することができます。

ERM では、集約は関係タイプの構成によって表します (次の図を参照)。 エンティティ タイプ「製造指図」と「作業手順」を集約することによって、新しいオブジェクト タイプ「指図作業手順」が作成されます。

図 35: 集約の例

集約演算子は、関係にも適用できます。 既存の関係タイプを 1 つのエンティティ タイプとして扱うと、それが新しい関係を作成する際の起点になります。 次の図に、これを表す例を示します。

最初の集約によって、エンティティ タイプ「製造指図」と「作業手順」の間の関係タイプ「指図作業手順」が構成されました。 キー属性である「製造指図番号」(PONO) と「作業手順番号」(RNO) は、「指図作業手順」の複合キーになります。 つまり、指図作業手順には、複数の操作が割り当てられることになります。 このため、「注文処理」関係が「指図作業手順」関係タイプと「処理」エンティティ タイプの間に形成されます。 関係はエンティティ タイプ間にしか作成できないので、元の関係タイプ「指図作業手順」を再統合する必要があります。 次の図では、菱形を長方形で囲むことによってこれを表しています。 この再統合された関係タイプは「通常」のエンティティ タイプとして扱います。 関係タイプの構成を図で示すため、構成に含まれるエンティティ タイプから菱形まで接続線を描きます。 新しい関係を形成する、再統合された関係タイプからの接続線は、菱形を囲む長方形の端で止まり、シンボル内部の菱形には達しません。

一般的なルールとして、複合キーを単純なキーで置き換えることは可能ですが、データ モデルの作成プロセスが追跡できるようになるため、複合キーを保持すると便利です。

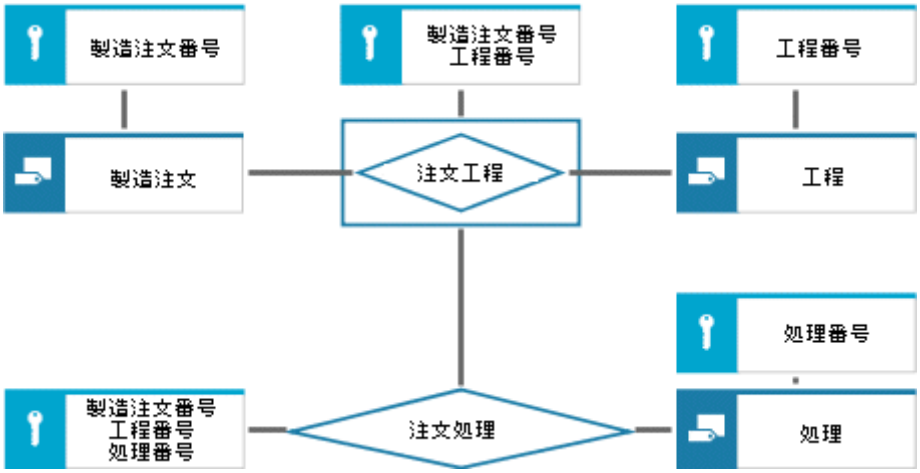


図 36: 再統合された関係タイプとの集約

ERM では、複合的な構造コンテキストをわかりやすい構造に分割します。 構造全体への関係はあいまいになることがあるため、クラスター/データ モデル形式の複合オブジェクトを導入します。

クラスター/データ モデルは、複合オブジェクトの記述に必要な、データ モデル内の複数のエンティティ タイプと関係タイプに関する論理ビューです。

エンティティ タイプや関係タイプのほかに、クラスター/データ モデルは、別のクラスター/データ モデルに含めることができます。 エンティティ タイプおよび関係タイプと異なり、クラスター/データ モデルは階層の中に自由に配置できます。したがって、データ モデルの作成プロセスには、主にトップダウンの方法が採用されます。 しかしながら、ボトムアップの方法で、サブモデルを組み合わせたり統合する場合にも、クラスター/データ モデルを構成することは非常に役立つことがあります。

次の図にクラスター/データ モデルを示します。

図 37: データ クラスター (図形シンボル)

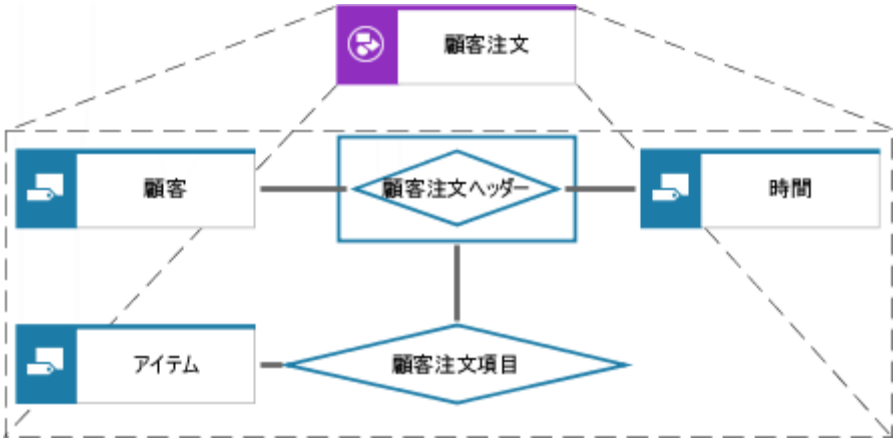


図 38: 複数のオブジェクトのデータ クラスタ ビュー

クラスター/データ モデルは、複数のエンティティ タイプおよび関係タイプを表す論理的なビューです。 複合オブジェクトである「受注」を記述するためには、エンティティ タイプ「顧客」、「時間」、「受注ヘッダー」、「品目」、「受注明細」および関係タイプが必要です。

グループ化

グループ化とは、1 つのエンティティ集合の要素からグループを構成することです。

次の図の例では、すべての「機器」をまとめて「機器グループ」を形成しています。 「機器グループ」は独立オブジェクトで、個々の「機器」には含まれていない属性 (「機器グループ」としての名前、「機器」の構成要素数) を追加するとさらに詳しく記述できます。 その他の例としては、ワークステーションの部門へのグループ化や、注文明細項目の注文へのグループ化などがあります。

図 39: グループ化

4.2.1.2.2 多重度の拡張

多重度を指定する場合、これまでは関係オカレンス数の上限のみを指定しました。 たとえば、次の図の多重度は、1 つのプロジェクトに最大 m 人の従業員を割り当てることができて、1 人の従業員が最大 n 個のプロジェクトに参加できることを表しています。

図 40: 上限/下限 (1)

この上限のほかに、下限を使用して、関係オカレンスの最小数を指定することもできます。 この目的で、多重度を一組の文字 (a, b) で表すことができます (Scheer 著『Business Process Engineering』を参照)。 次の図の (a1, b1) は、すべてのプロジェクトが最低 a1 個、最高 b1 個の [参加する] タイプの関係オカレンスに参加できることを表しています。つまり、各プロジェクトには、最低 a1 人、最高 b1 人の従業員を割り当てることができます。 もう 1 つの組の文字 (a2, b2) は、1 人の従業員が最低 a2 個、最高 b2 個のプロジェクトに参加できることを表しています。

図 41: 上限/下限 (2)

このように、すべての関係は 2 つの複雑度 (最小, 最大) で定義されます。 下限には、通常 0 または 1 の値を割り当てます。上限には、1 <= max <= * (* はワイルドカード) の値範囲を指定します。

min = 0 という下限は、1 つのエンティティが 1 つの関係に参加することができるが、必須ではないことを表します。

min [illegible] という下限は、1 つのエンティティが少なくとも 1 つの関係に参加する必要があることを表します。

次の図では、従業員は関係に参加することができるが必須ではない (min = 0) ことが下限により表されます。一方、プロジェクトは少なくとも 1 つの関係に参加する必要がある (min = 1) ことを表します。 つまり、プロジェクトに割り当てられていない従業員がいてもいいことを表しています。 逆に、各プロジェクトには、少なくとも 1 人の従業員が割り当てられなければなりません。

図 42: 上限/下限 (3)

最小値が 0 または 1、最大値が 1 または * であるとすると、(min,max) 表記には、(1,1)、(1,m)、(0,1)、(0,m) の 4 つのパターンが考えられます。 または、次の省略形を使用することができます (Schlageter/Stucky 著『Database systems』(1983 年)、51 ページを参照)。

- 1 ((1.1) に相当)
- c ((0,1) に相当)
- m ((1,m) に相当)
- cm ((0,m) に相当)

次の図に、省略形を使用して上記のグラフィック例を示します。

図 43: 上限/下限 (4)

4.2.1.2.3 識別と存在の依存

「ERM - eERM 拡張『37 ページ 』」の章で述べたように、下限と上限を指定して多重度を拡張する方法によって、データ オブジェクト間の特定の依存性を定義できます。

定義上、関係タイプおよび再統合された関係タイプは独立して存在することはできません。それらが接続するエンティティ タイプがあることにより、存在できるようになります。 これは、関係タイプおよび再統合された関係タイプが、識別と存在の両方においてほかのエンティティ タイプに依存していることを意味します。

また、固有のキー属性を持っていながら、ほかのエンティティの存在に依存するエンティティ タイプもあります。 この依存性は、たとえばグループ化によって発生します。 次の図に示すように、部門には少なくとも 1 つのワークステーションが割り当てられている必要があります。また、ワークステーションがある部門に割り当てられるように定義します。 次の図に示すように、これらの依存に関する関係は複雑度によって表されます。 (min, max) 表記では (1,1) および (1,*) を使用します。 データ モデル内での存在に関する依存関係の定義は、実装の際に参照データの整合性の条件になります。 つまり、これらの条件を満たすことによって、トランザクションが実行されたあとでも、データベースの内容の一貫性が確実に維持されることを意味します。 下の例では、部門を削除できるのは、その部門に割り当てられているワークステーションがすべて削除されている場合に限られることを意味します。

図 44: 存在の依存関係

4.2.1.2.4 企業で使用する用語のモデリング - 用語モデル

モデリング、特にデータのモデリングの際、規模の大きな企業では、情報オブジェクトを定義する用語が統一されていないという問題がしばしば起こります。 たとえば、購買部門と生産部門では「注文」という語の定義が異なります。 社内および部門内で一貫した用語を使うことによって、情報の理解度が高まります。 このような理由から、ARIS メソッドでは、いわゆる用語モデルが提供されています。これは、同義語の管理という意味で異なる用語の管理ができるだけでなく、データ モデルのオブジェクト (エンティティ タイプ、関係タイプ) 間に存在する関係や、その企業内で使用される用語を指定できます。

このような関係を表すために、[用語] オブジェクトが導入されています。 このようにして、データ モデルのすべての情報オブジェクトに複数の用語を割り当てることができます。 次の図に例を示します。

図 45: 用語 (1)

用語を相互に関連付け、階層構造を作成することができます。 次に、用語間で接続線タイプがどのように使用されるかを図示します。

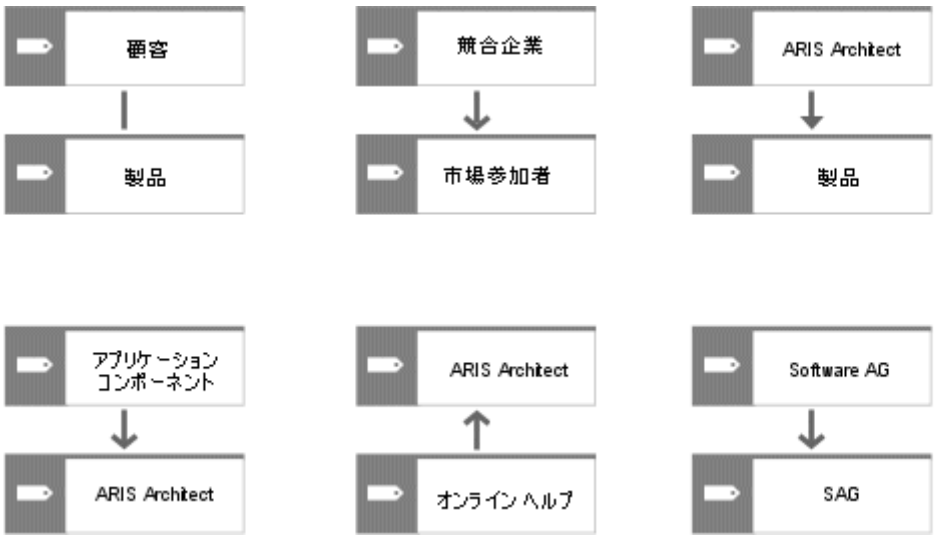


図 46: 用語 (2)

用語モデルで定義された用語は、情報オブジェクトを含むほかのモデル タイプでも利用できます。たとえば、プロセス連鎖図で、ファンクションの入出力データを表すために利用できます。

4.2.1.2.5 eERM 属性割当図

eERM 形式のデータ モデルは、エンティティ タイプと関係タイプを表示するだけであっても、ほとんどの場合複雑な構造になります。 ERM 属性をそのままこれらのモデルに含めると、図が読みにくくなってしまいます。

eERM 属性割当図を利用すると、別のモデルで、各エンティティ タイプおよび関係タイプへの ERM 属性の割り当てを行うことができます。 eERM のオブジェクト タイプ (エンティティ タイプまたは関係タイプ) をオカレンス コピーでこのモデルに含めることで、ERM 属性に対する関係をモデル化できます。 接続された ERM 属性が、キー属性、外部キー、または記述属性かどうかを区別できます。 次の図に例を示します。

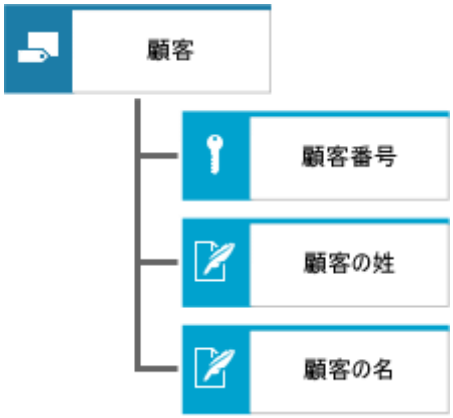


図 47: エンティティ タイプへの ERM 属性の割り当て

このモデル タイプは、各 ERM 属性の表現や割り当て以外にも、属性タイプ グループとそれらの割り当てを表示できます。

属性タイプ グループは、あるエンティティ タイプの ERM 属性のうち、意味的に密接に関連した属性のグループを表します。たとえば、全体としては二次キーを構成する、あるエンティティ タイプの ERM 属性を集めて、1 つの属性タイプ グループを作ることができます。

属性タイプ グループは次のように表します。

図 48: 属性タイプ グループの表現

ERM 属性割当図で使用可能な関係の一覧は、インストール媒体に含まれている『ARIS メソッド マニュアル - テーブル』マニュアル (ARIS Method tables.pdf) を参照してください。

4.2.1.3 その他の表記法

4.2.1.3.1 SAP SERM

これまでに説明した表記法のほかにも、ほかの ERM 表記法もまた使用されています。 これらのモデルでは、通常、関係タイプは独立のオブジェクトとしてではなく、エンティティ間の接続として表現されます。 独自の属性を持つ関係タイプは、(弱い) エンティティ タイプとして表示されます。

このような表記法には、Sinz によって開発された SERM (構造化実体関連モデル : 以下 SER モデル) 手法があります (Sinz 著『Entity Relationship Model』(1990 年) 参照)。 元のデータ オブジェクトと依存データ オブジェクトを有向グラフを使って表示し、モデルのオブジェクトを左 (強いエンティティ タイプ) から右 (弱いエンティティ タイプおよび関係タイプ) へ配置することによって、開発中のデータ構造の方向性を視覚化できます。 接続線のアンカー ポイントは関連するオブジェクトにより制限されるため、方向性がさらに見やすくなります。 SER モデルと、ここまでに説明してきた eERM 手法をベースにしたモデリングとの主な違いは、グラフィック表現にあります。

ドイツのヴァルドルフに本社がある SAP 社で開発されたモデリング手法には、SER 手法の概念とここで紹介した eERM 手法の両方が取り入れられています (Keller/Hechler 著『Information Model』(1991 年) 参照)。

SAP が採用する SAP-ER モデルのコンテキストでは、オブジェクトの構成中に、エンティティ タイプと関係タイプの表記を区別しません。 情報オブジェクト間の依存性は、矢印で示される関係の複雑さにより表されます。 ただし、階層関係、集約関係、参照関係は区別します (図を参照)。

階層関係は、情報オブジェクト間の単方向の依存関係を表します。

集約関係は、eERM 手法に基づいた関係タイプの構成に対応しています。

参照関係は、eERM 手法に基づく、再統合されたエンティティ タイプと元のエンティティ タイプとの間の論理的な依存性を表します。

特化は、ERM 手法と同様に表示されます。

次の図では、eERM 表現を SAP ERM 表現に変換する例を挙げて、この表記法について説明します (Seubert 著『SAP Data Model』(1991 年) 94 ページを参照)。

これは、基本的な技術的内容を、情報が失われることなく、この形式の表現に変換できることを示しています。

SAP ERM は、作成プロセスが完了した後のデータ モデルの表現と捉えることができます。 SAP ERM では、情報オブジェクトがグラフ指向で配置されているため、情報オブジェクトをすばやく見つけることができます。これは、大きいデータ モデルの場合には特に有効です。

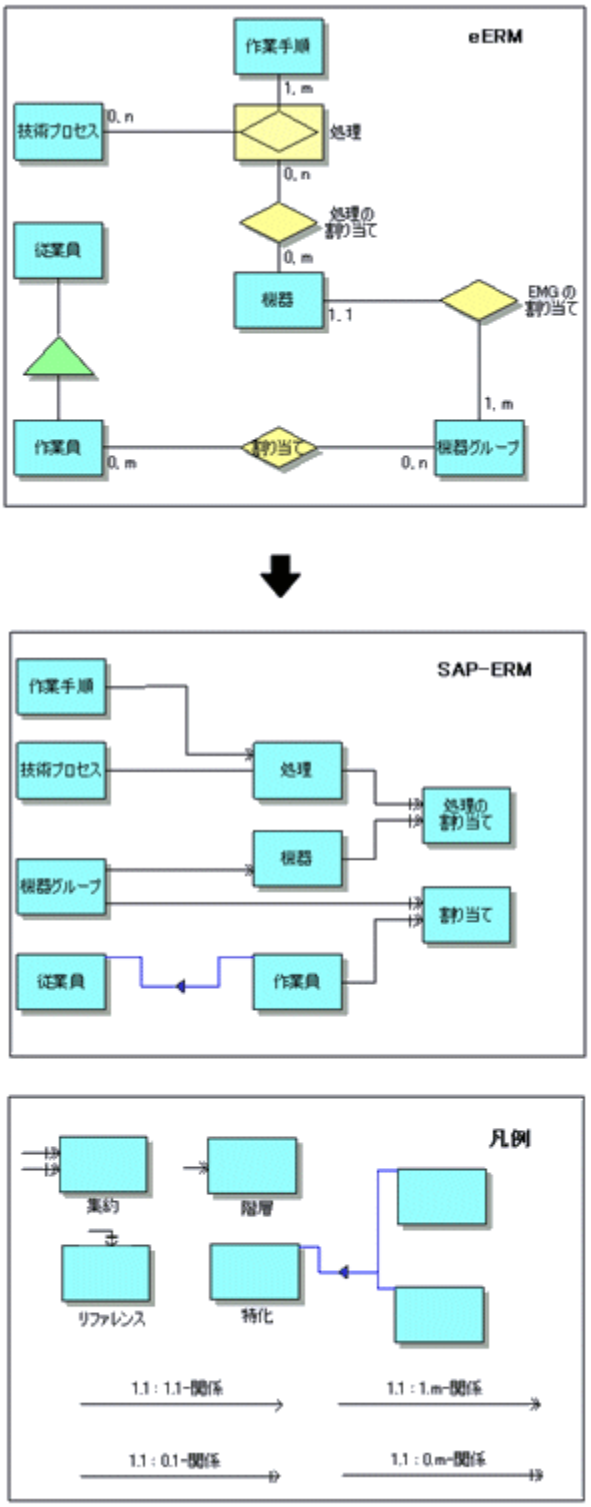


図 49: eERM 表記および SAP ERM 表記

4.2.1.3.2 IE データ モデル

IE データ モデルは、Texas Instruments Inc. の CASE tool、Information Engineering Facility (IEF) のデータ モデリング表記に準拠します。

SAP SERM 表記と同様に、IE 表記にも、エンティティ タイプ間の関係を表すための独自のオブジェクト タイプがありません。

次の図に、IE 表記のデータ モデルの例を示します。

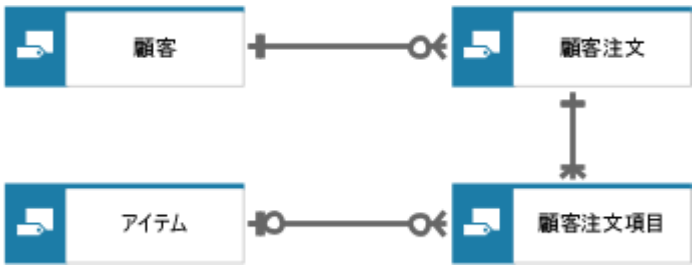


図 50: IE 表記のデータ モデル

4.2.1.3.3 SeDaM モデル

SeDaM (Semantic Data Model) データ モデル表記は、BASF 社の表記です。

この表記には、エンティティ タイプ間の関係を表す独自のオブジェクト タイプはありません。

エンティティ タイプは、左から右へ配置されるとは限りません (SAP SERM 表記を参照)。

[クラスター/データ モデル] および [汎化タイプ] オブジェクト タイプも利用できます。

次の図は、SeDaM 表記法のデータ モデルの例を示します。

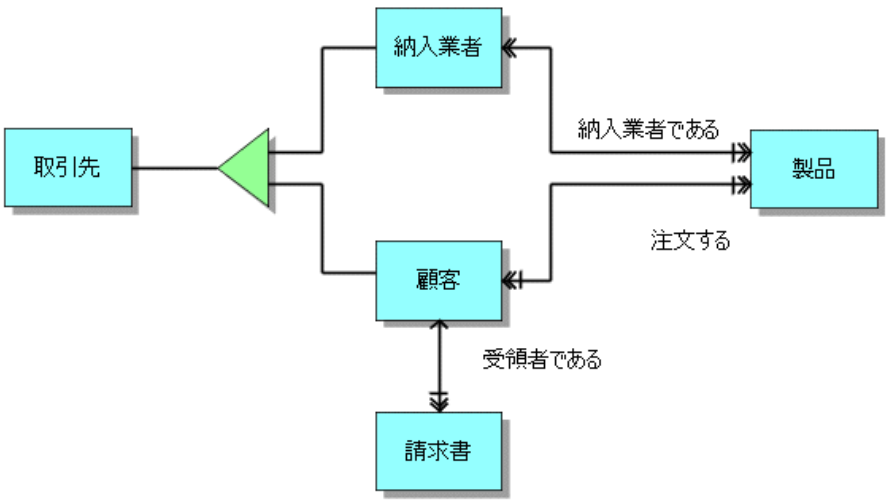


図 51: SeDaM 表記のデータ モデル

SeDaM モデルで使用可能な関係タイプの一覧は、インストール媒体に含まれている『ARIS メソッド マニュアル - テーブル』マニュアル (ARIS Method tables.pdf) を参照してください。

4.2.1.4 eERM の主な用語と表記法のまとめ

次の図は、eERM (拡張実体関連モデル) の構成要素および設計演算子の用語および表記法をまとめたものです (Scheer 著『Business Process Engineering』(1994 年)、45 ページを参照)。

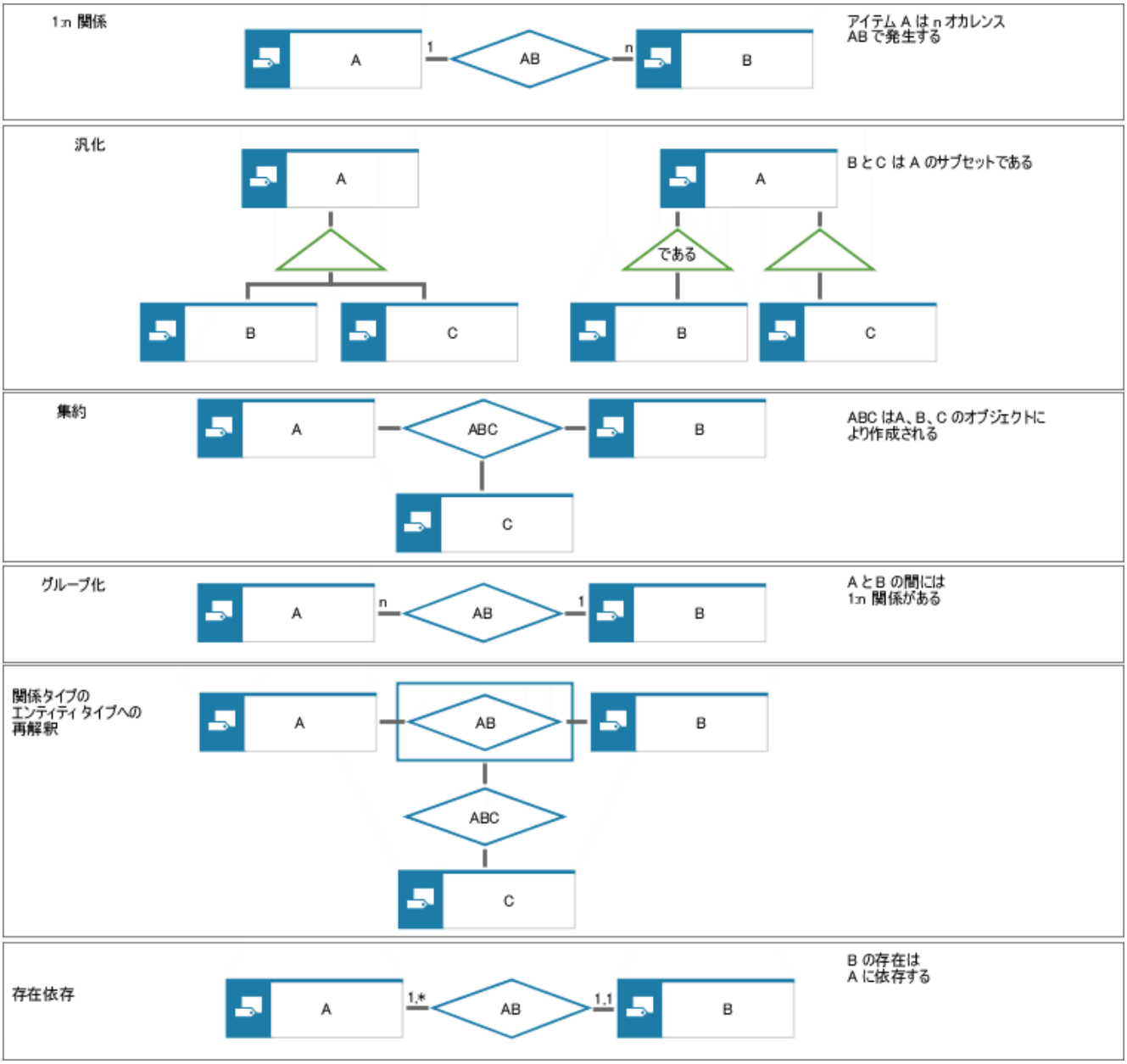


図 52: eERM: 用語および表記法

4.2.1.5 文書型定義

DTD タイプ（文書型定義）のモデルには、特定のタイプの XML 文書を作成する際に従う必要のあるルールが記述されます。 この記述は、要素タイプの宣言による形式で行います。

たとえば、DTD を使用して文書タイプの全般的な構造を定義できます。 DTD で定義される文書タイプの有効な文書は、XML 文書として作成できます。 これには、対応する DTD に基づいてさまざまなプログラムを使用して文書を処理できるという利点があります。

[要素タイプ] オブジェクトから始めて、厳格な階層関係に従ってモデルを設定する必要があります。 ソース オブジェクトには入力接続線を接続できません。 また、接続線は常に、上位オブジェクトから下位オブジェクトへという方向を持つ必要があります。 文書構造を明確にするために DTD モデルにはあらゆる要素タイプを記述できますが、構造が複雑な場合はアサイン DTD モデルに要素タイプを宣言することもできます。

アサインメントで作業する場合、アサイン モデルには要素タイプの完全な記述が含まれている必要があります。

DTDExport.rsm レポートを使用すると、ベース モデルから、DTD モデル構造全体に適用できる DTD ファイルを作成できます。

テキスト ファイルの形式で文書タイプが宣言されている場合は、DTDImport.rsg レポート を使用して [DTD] タイプのモデルを生成できます。

4.2.1.5.1 要素タイプ

DTD において重要なコンポーネントは要素タイプです。 有効な XML 文書には、DTD モデル階層に存在する要素タイプのインスタンスを使用できます。このインスタンスを要素と呼びます。

各要素タイプは、内容と属性タイプを使用して記述します。

要素タイプは、内容によって 3 つのタイプに分類できます。

- 内容としてテキストを持つ要素タイプ
- 内容を持たない要素タイプ
- 内容としてテキストまたはほかの要素タイプ、あるいはその両方を持つ要素タイプ

内容としてテキストだけを持つ要素タイプを記述するには、必要な名前を持つ [要素タイプ] オブジェクトと、[#PCDATA] シンボルを持つ [内容] タイプの要素を配置します。 その後、要素タイプと内容との間に [含む] タイプの接続線を描きます。

図 53: 内容がテキストのみの DTD 要素タイプ

内容を持たない要素タイプを記述するには、必要な名前を持つ [要素タイプ] オブジェクトと、[EMPTY] シンボルを持つ [内容] タイプの要素を配置します。 その後、要素タイプと内容との間に [含む] タイプの接続線を描きます。 内容を持たない要素タイプの典型的な例としては、HTML の <img> タグがあります。 空の要素タイプの重要な利点は、<img> タグにおける「SRC」、「ALIGN」、「ALT」、「ISMAP」属性など、役に立つ属性を持てることです。

最も複雑な要素タイプの形式は、内容が混在している要素タイプです。 この場合、テキストまたは要素タイプを [要素タイプ] タイプのオブジェクトに割り当てることができます。 どのような形式で割り当てが相互に関係するかの記述には、ルールが使用できます。 要素タイプと割り当てとの間の接続線の数により、その要素タイプの内容において割り当てが発生する頻度が説明されます。

割り当ての接続には、順序演算子および XOR ルールが使用できます。

記述する要素タイプの内容で、指定されている数の割り当てのうちいずれか 1 つしか許可されない場合、それらの割り当てを相互に XOR ルールを使用して接続する必要があります。

記述する要素タイプの内容で、特定の順序でのみ複数の割り当てが許可されている場合、それらの割り当てを順序演算子で接続する必要があります。 必要な順序を明確に指定するには、ルールと割り当ての間の接続線に、その割り当てに必要な順序内の位置を指定します。

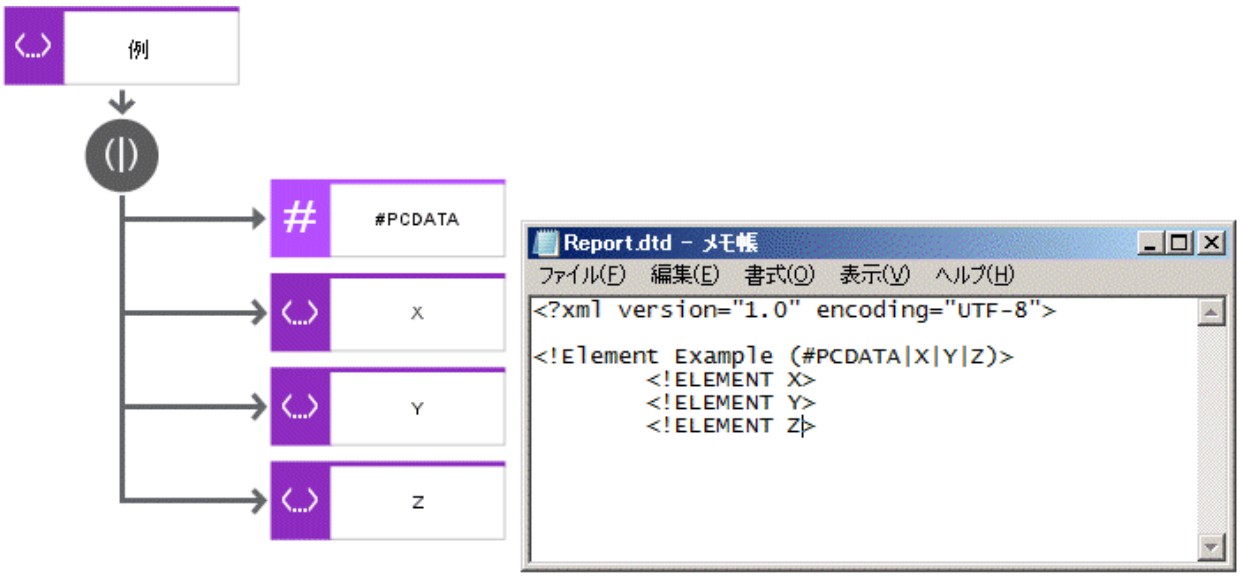


図 54: 混在した内容を持つ要素タイプと、DTD への変換

また、割り当ては、ルールを使用せずに、記述されている要素にも接続できます。 この場合、順序は任意であるとみなされます。

4.2.1.5.2 接続線タイプ

要素タイプの内容に含まれる割り当てが発生する頻度には、ばらつきがあります。 要素タイプの内容に含まれている割り当てのオカレンスのうち、許可されるオカレンスの数は、要素タイプと割り当てとの間の接続線タイプによって決まります。

次の接続線タイプが利用できます。

- 任意の回数を持つ
- 少なくとも 1 回持つ
- 多くて 1 回持つ
- 1 回持つ

次の表に、さまざまな接続線タイプと、オカレンスを表すために、生成された DTD で使用される対応する文字を示します。

接続線タイプ	説明	文字
任意の回数を持つ	この割り当ては、記述する要素タイプの内容で必ずしも発生する必要はありません。 何度でも発生できます (最小値 = 0、最大値 = 任意の正の整数)。	*
少なくとも 1 回持つ	この割り当ては、記述する要素タイプの内容で発生する必要があります。発生する回数は 1 回以上です (最小値 = 1、最大値 = 任意の正の整数)。	+
多くて 1 回持つ	この割り当ては、記述する要素タイプの内容で必ずしも発生する必要はありません。 発生回数は 1 回に限られます (最小値 = 0、最大値 = 1)。	?
1 回持つ	この割り当ては、記述する要素タイプの内容で発生する必要があります。 発生回数は 1 回に限られます (最小値 = 1、最大値 = 1)。	なし

4.2.1.5.3 属性タイプ

DTD には、構造の記述だけでなく、属性タイプの宣言を含めることもできます。 属性タイプにより、要素タイプの性質が記述されます。 属性タイプには常に、要素タイプに割り当てられます。

DTD では、簡単な属性タイプと列挙属性タイプを宣言できます。

簡単な属性タイプを定義するには、次の手順に従います。

1. 必要な名前の [属性タイプ] オブジェクトをモデルに配置します。
2. この属性タイプを使用して性質が記述されている要素タイプから新しい属性タイプに接続線を引きます。
3. 属性編集を開き、その属性タイプの宣言に関する詳細情報を入力します。

次のような情報を入力します。

- 属性値のデータ型
- 属性デフォルト
- デフォルト値

4.2.1.5.3.1 属性値のデータ型

属性値のデータ型を指定するには、[データ型] 属性を指定します。 次の表は、指定するデータ型を示します。

データ型	説明
CDATA	属性値に文字列を使用できます。
ID	属性値に一意の識別子を使用できます。 値が一意でない場合、XML プロセッサからエラー メッセージが通知されます。
IDREF	文書内の別の場所で定義した識別子への参照を、属性値に使用できます。 現在の XML 文書に割り当てられていない識別子を値として使用すると、XML プロセッサからエラー メッセージが通知されます。
IDREFS	属性値として、IDREF タイプの複数の属性値をスペースで区切って列挙できます。 現在の XML 文書に割り当てられていない識別子が属性値のなかで参照されると、XML プロセッサからエラー メッセージが通知されます。
ENTITY	DTD で宣言されているリモートのバイナリ エンティティへの参照を属性値として使用できます。
ENTITIES	属性値として、ENTITY タイプの複数の属性値をスペースで区切って列挙できます。
NMTOKEN	文字、数字、ピリオド、ダッシュ、セミコロン、アンダースコアの任意の組み合わせを属性タイプとして使用できます。
NMTOKENS	属性値として、NMTOKEN タイプの複数の属性値をスペースで区切って列挙できます。
NOTATION	DTD で宣言されている表記への参照を属性値として使用できます。

[NMTOKEN] の値は、その [データ型] 属性のデフォルトとして指定されます。

4.2.1.5.3.2 属性デフォルト

[属性デフォルト] 属性の値は、次の中から選択できます。

- #REQUIRED
- #IMPLIED
- #FIXED

属性タイプの属性デフォルトの値として [#REQUIRED] が設定されており、またこの属性が XML 文書で要素に対して指定されている場合は、その属性に有効な値を指定する必要があります。 値が指定されていない場合、XML プロセッサからエラー メッセージが通知されます。

属性タイプの属性デフォルトの値として [#IMPLIED] が設定されており、またこの属性が XML 文書で要素に対して指定されている場合、その属性に値を指定するかどうかは任意です。

属性タイプの属性デフォルトの値として [#FIXED] が設定されている場合、その属性値には固定値が使用されます。 この値は [デフォルト値] 属性に指定する必要があります。 XML 文書においてその属性が設定値を持っていない場合、XML プロセッサでは、XML 文書においてそれが設定値を持っているかのように動作します。

[#IMPLIED] の値は、[属性デフォルト] 属性のデフォルトとして指定されます。

列挙属性タイプを宣言するには、次の手順に従います。

- 必要な名前の [列挙属性タイプ] オブジェクトをモデルに配置します。
- 列挙属性タイプを使用して性質が記述されている要素タイプから新しい属性タイプに接続線を引きます。

[列挙] タイプのオブジェクトを配置し、列挙属性タイプの名前に使用できる値を入力します。 それぞれの値はカンマで区切る必要があります。 値の一覧に 250 以上の文字が含まれる場合は、値を複数の列挙オブジェクトに分配します。

列挙属性タイプと列挙との間に接続線を作成します。

最後に、デフォルトとして設定できる値を列挙の中から選択して [デフォルト値] 属性に指定します。

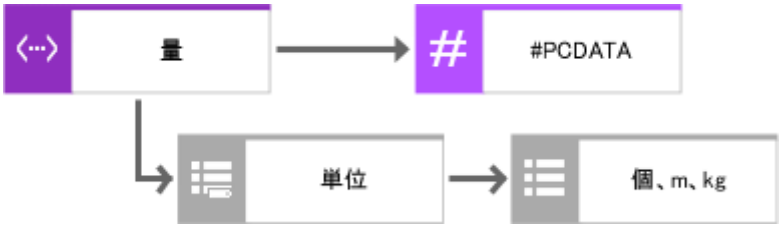


図 55: 列挙属性タイプを持つ要素タイプ

DTD モデルの [説明/定義] 属性に関する情報およびモデルに含まれている要素タイプは、DTDGenerator.rsm レポートによって生成される DTD にコメントとして追加されます。

モデル属性によっては、DTD でパラメータ エンティティ、内部または外部エンティティ、表記を宣言できるものがあります。

パラメーター エンティティ

パラメーター エンティティを宣言するには、次の情報を下記の順序で入力します。

- 象徴的な名前
- キーワード
- 必要なパラメータ エンティティの宣言の統一リソース識別子 (URI)

象徴的な名前は、現在の DTD でパラメーター エンティティ参照として使用できます。

キーワードには、「SYSTEM」および「PUBLIC」を使用できます。 パラメーター エンティティを使用して社内でのみ認識および使用されているファイルを参照するには、キーワード「SYSTEM」を入力します。 ただし、これが標準ライブラリの一部であるファイルを参照する場合は、「PUBLIC」を入力します。

キーワード「PUBLIC」に続く URI の例としては「-//w3c//ENTITIES Latin//JA//HTML"、"http://www.w3.org/DTD/ISOLAT1.ent」があります。

内部エンティティ

内部エンティティを宣言するには、次の情報を下記の順序で入力します。

- 象徴的な名前
- 一般的な内部エンティティの宣言に使用するテキスト。 名前とテキストの間にスペースを入れ、テキストを引用符で囲みます。

指定された象徴的な名前を持つエンティティ参照が現在の DTD の XML 文書にある場合、この名前はここで指定したテキストに置き換えられます。

外部エンティティ

外部エンティティを宣言するには、次の情報をこの順序で入力します。

- 象徴的な名前
- キーワード
- 統一リソース識別子 (URI)
- 一般的な外部エンティティの宣言に使用する表記

現在の DTD の XML 文書において、象徴的な名前をエンティティ参照として使用できます。

キーワードには、「SYSTEM」および「PUBLIC」を使用できます。 外部エンティティを使用して社内でのみ認識および使用されているファイルを参照するには、キーワード「SYSTEM」を入力します。 ただし、これが標準ライブラリの一部であるファイルを参照する場合は、「PUBLIC」を入力します。

URI により、外部エンティティが参照しているオブジェクトの場所が XML プロセッサに通知されます。

キーワード「SYSTEM」に続く URI の例としては、

「c:¥images¥test.gif」があります。

キーワード「PUBLIC」に続く URI の例としては、

「-//w3c//ENTITIES Latin//EN//HTML" "http://www.w3.org/DTD/ISOLAT1.ent」があります。

エントリの末尾にある表記により、XML プロセッサに外部エンティティが参照しているオブジェクトのタイプが指定されます。 この表記は、現在の DTD で宣言されていなければなりません。

たとえば、エンティティ宣言で最初の例の URI を使用する場合、まず DTD で GIF データ形式を表す表記を宣言し、それから NDATA.GIF を挿入してこの属性タイプのエントリを完成させます。

表記

表記を宣言するには、次の情報をこの順序で入力します。

- 象徴的な名前
- キーワード
- 統一リソース識別子 (URI)

この情報により、XML プロセッサに現在の DTD の XML 文書においてこのタイプのオブジェクトが発生した場合の対処方法が指定されます。

表記の象徴的な名前を、現在の DTD における属性およびエンティティの宣言に使用できます。

キーワードには、「SYSTEM」および「PUBLIC」を使用できます。 たとえば、GIF 形式のオブジェクト用の表記を宣言し、ローカルで利用可能な Internet Explorer を使用してこのタイプのオブジェクトを表示するよう XML プロセッサに指示する場合は、キーワード「SYSTEM」を使用します。 ただし、TEX タイプのファイルの表記を宣言して、一般的にアクセス可能なリソースまたはソースを参照する場合は、キーワードに「PUBLIC」を入力します。

URI により、XML プロセッサに、指定したタイプのオブジェクトを扱う命令が格納されているアプリケーションまたは情報の場所が指示されます。 たとえば、キーワード「SYSTEM」での GIF 表記の URI は、「c:\Program Files\Internet Explorer\Iexplore.exe」のように指定し、キーワード「PUBLIC」を使用した GIF 表記の URI は、「ISBN 0-201-13448-9:://NOTATION TeX//JA」のように指定します。

4.2.1.5.4 DTD のテスト

必要な DTD を作成したあと、それを 2 つの手順でテストするには、[条件付きセクション] オブジェクト タイプを使用して、DTD のうち現在のテストに含まれていない部分を有効または無効にします。

条件付きセクションに属している要素タイプを DTD から隠すには、該当する条件付きセクションの [無視] 属性をアクティブにします。

「DTDImport.rsg」レポート スクリプトを使用して条件付きセクションを含むテキスト形式の DTD から [DTD] タイプのモデルを生成する場合、条件付きセクションの内容は含まれません。

[DTD] タイプのモデルを、ARIS メソッドのデータ要素にアサインすることができます。 データ要素には次のようなものがあります。

- クラスター/データ モデル
- パッケージ
- エンティティ タイプ
- 関係タイプ
- 用語

4.2.1.6 マテリアル フロー モデリング - 材料図

プロセス モデル (EPC (マテリアル フロー付き)、PCD (マテリアル フロー付き)) でマテリアル フローを図示できるようにするため、材料タイプは、ファンクションの入力または出力という形式で、業務プロセスの個々のファンクションに割り当てられます。 情報オブジェクトのファンクションへの割り当て (情報の変換はファンクションを使用して表現) と同様に、この割り当ては入力材料タイプの出力材料タイプへの変換を表します。

材料図では、材料タイプを定義し、それらを階層に配置したり、材料クラスに割り当てたりできます。

[材料タイプ] は、同一の材料特性を持つ個々の材料を分類します。

類似の [材料タイプ] を 1 つにまとめて [材料クラス] を形成できます。 異なる分類基準に基づいて、類似性を定義できます。 1 つの [材料タイプ] は複数の [材料クラス] に属することができます。

[材料タイプ] を [包装材タイプ] に割り当てることができます。 これは、特定の [材料タイプ] を特定の [包装材タイプ] の中でのみ運搬できることを意味しています。

また、[包装材タイプ] も階層に配置したり分類したりできます。 これによって、たとえば、複雑な大量包装の構成や制約を表現できます。

包装材タイプは、同一の特性 (材料特性など) を持つ個々の包装材を類型化したものです。

類似の [包装材タイプ] をまとめて、[包装材クラス] を構成できます。 異なる分類基準に基づいて、類似性を定義できます。 したがって、1 つの [包装材タイプ] は複数の [包装材クラス] に割り当てることができます。

次の図に材料図の階層レベルおよび分類の例を示します。

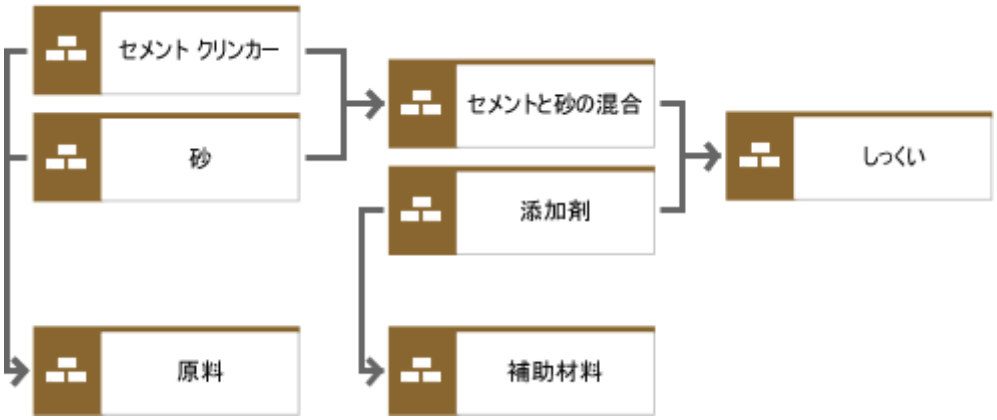


図 56: 材料図の例

4.2.1.7 データ ウェアハウスの構造図のモデリング

データ ウェアハウス構造図は、データ ウェアハウスの構造を記述するために使用します。 基本的に、この構造図には静的な記述、つまりデータ間の関係およびその場所が示されます。 ARIS アーキテクチャでは、このタイプの記述はデータ ビューで実現されます。 焦点は、情報の相互関係と配置に置かれます。 データの次元は、InfoCube を使用して記述します。 次元の連携は、スター スキーマで表されます (下の図を参照)。 次元は、ほかの時間と接続するためのキーとして使用できます。 独自の次元を持つオブジェクトには特定の値を設定できます。この値はファクト テーブルにカタログ化されており、KPI により厳密に指定されます。 依存関係は、主要属性およびその機能を含めて次元テーブルに記述されます。 機能の階層関係の記述には、ツリー構造を使用します。 最後に、次元を構造図を使用してマスター データ テーブルに割り当てます。

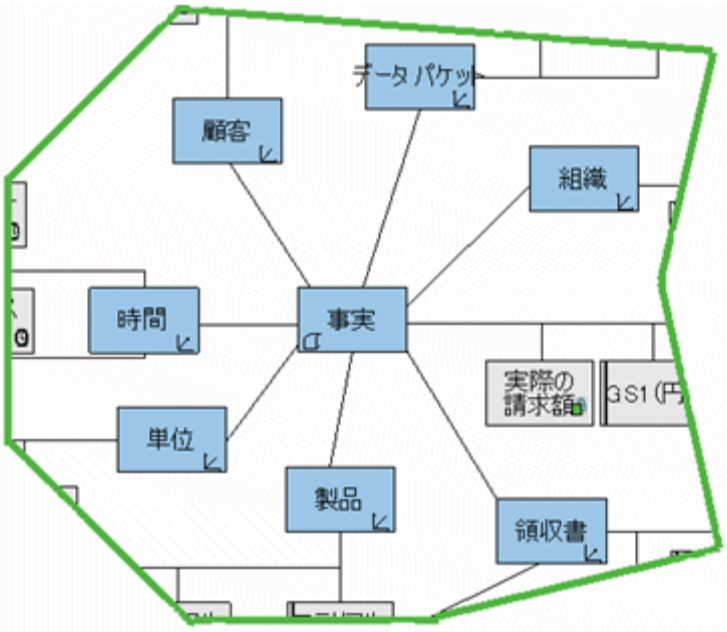


図 57: スター スキーマのデータ ウェアハウス

4.2.1.8 認証階層

認証階層図は、ロールや組織のモデリングに使用します。 この図は、ロール図で定義された認証関係を表します。 上位認証や下位認証が定義されているので、構造は論理的で、認証の衝突を回避できます。

[認証階層図] は、[ロール図] と密接な関係があります。 ロールの記述には認証の一覧を使用し、必要とされるプロファイルを定義します。 構造は、ファンクション ツリーの構造と対応しています。

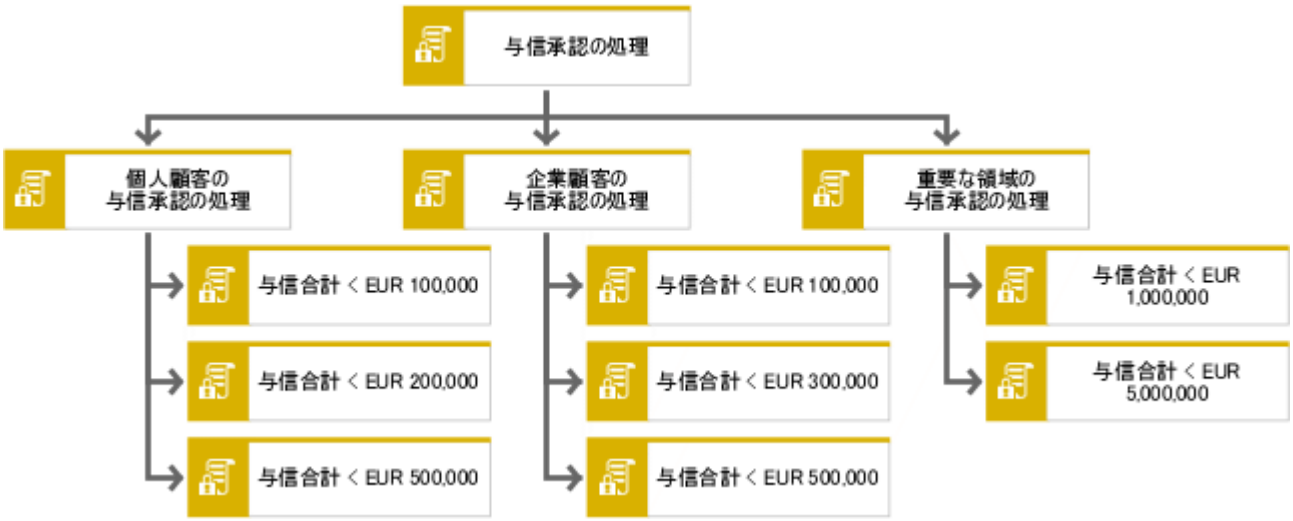


図 58: 認証階層

4.2.1.9 プロセス原価管理のデータ モデル

4.2.1.9.1 原価作用因図

原価作用因図 (CD 図) は、(ARIS Optimizer を使用するなどして) プロセス原価管理で使用されます。 原価作用因図は、原価作用因階層を表します。

原価作用因は、特定プロセスの原価を見積るための情報となる指標/参照値です。 参照値は、利用可能な情報源から容易に取り出せる操作可能な値で、原価見積りに比例している必要があります。

原価作用因は、パフォーマンス量変数型プロセスまたはパフォーマンス量誘発型プロセスに対してのみ定義することができます。 原価作用因は、「部門を管理する」などのパフォーマンス量中立型プロセスに対して定義することはできません。 原価作用因の例として、「通りをアスファルト舗装する」プロセスの「通りの長さ」が挙げられます。

原価作用因図において、原価作用因の階層は [量を決定する] タイプの有向接続線によって表されます。 [原価作用因比例分子] および [原価作用因比例分母] 属性を、これらの接続線に指定する必要があります。 [原価作用因比例分母] が指定されていない場合は、値「1」が使用されます。 これらの 2 つの属性から商を求めることによって、2 つの原価作用因間における量的な比率が決まり、これがプロセス計算に使用されます。

次の図に、「乗用車数 (リムジン)」および「ドア数」という 2 つの原価作用因を使用した、上記の例を示します。 リムジンにはドアが 4 つあることを表すには、「乗用車数 (リムジン)」原価作用因から「ドア数」原価作用因への接続線で [原価作用因比例分子] 属性を「4」に設定します。

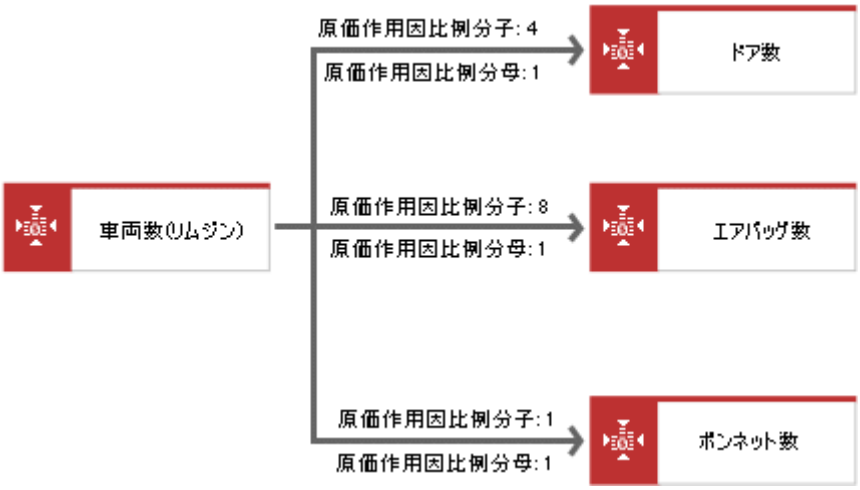


図 59: 原価作用因図の例

原価作用因はプロセス ビュー テーブル内の個々のプロセスに割り当てられます。 プロセス内の各ファンクションの利用度は原価作用因の階層から自動的に決定されます。

4.2.1.9.2 原価要素図

原価作用因図は、(ARIS Optimizer を使用するなどして) プロセス原価管理で使用されます。 これらは、原価要素の階層を表します。

原価要素は、原価オブジェクトの作成と使用から生じるすべての費用を体系的に構成するのに役立ちます。 問題となるのは、計上する原価の選択です。

たとえば、材料費は材料の使用に関する原価要素であり、減価償却は資産額の減少を記録する原価要素です。

総費用は、異なる基準に従って構成することができます。 使用された製造要因のタイプに従って費用を分配すると、労務費 (給与、コミッションなど)、材料費 (原材料費、機械の減価償却費など)、資本的支出、外部業者への支払い (運搬費、電力費など)、税金、手数料、および寄付などになります。 原価要素は、調達費、保管費、製造費、管理費、および販売費といった主要な運用ファンクションに沿って、さらに分割することができます。 いずれの構造も詳細化が可能です。

原価要素の階層は、[上位にある] タイプの有向接続線によって原価要素図で表されます。

原価要素の重要な属性は [パフォーマンス指標] です。 パフォーマンス指標は、原価要素パフォーマンスを測定する単位を表します (賃金時間、設備費の平方メートルなど)。

次の図に、使用される製造要因のタイプに基づく、前述の分類に対応する原価要素図の例を示します。これはまた、労務費の下位構造も表しています。

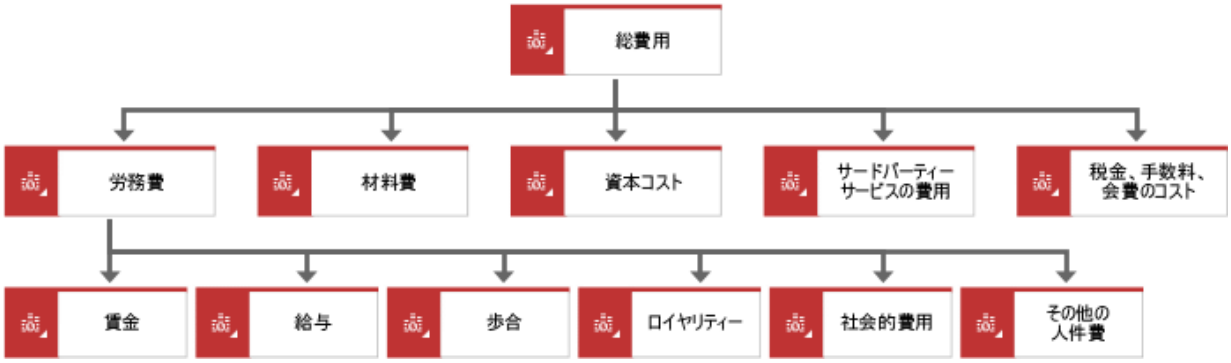


図 60: 原価要素図の例

原価要素図により、原価要素間の依存関係が視覚化されます。 この階層の配置は、ARIS Optimizer では計算の一部になることがあります。 たとえば、各原価要素の算出された費用が原価要素図の階層に従って集約されるよう、ARIS Optimizer を設定できます。

4.2.1.10 プロジェクト管理データ モデル

4.2.1.10.1 情報媒体図

情報媒体図は、ARIS Architect を使用したプロジェクト管理のオプション コンポーネントです。 情報媒体図は、データ ビューの要件定義に属し、文書、ログ、ARIS モデルなどの入出力データの記録に使用されます。

ARIS モデルは、PPC (プロジェクト プロセス連鎖、プロセス ビューの要件定義を参照) でクラスタに割り当てることができます。 このようにして、クラスター データの予備的な仕様が実現できます。 ワープロ文書ファイルなどの必要なファイルは、ARIS Architect 内から [リンク 1] から [リンク 4] 属性を使用して明示的に表示および呼び出すことができます。

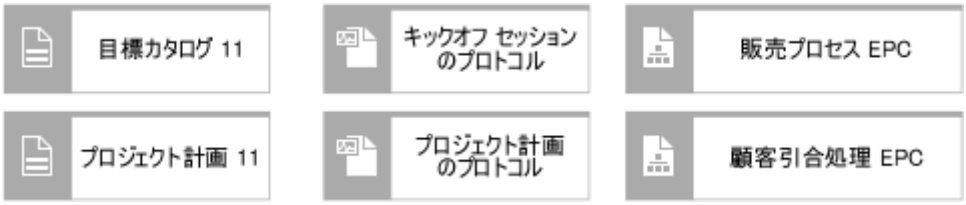


図 61: 情報媒体図

4.2.2 仕様設計

4.2.2.1 リレーション図、属性割当図

仕様設計では、要件定義で設計した論理的なデータ構造を、具体的なデータベース システムを構築できるような形式の記述に変換します。 ARIS では、これにを実現できるリレーション図が使用できます。

リレーション図および属性割当図を使用して、既存の関係および属性、また、それらと要件定義で導入された情報オブジェクトとの関係を定義できます。

最初に、リレーション図で、必要なリレーションを定義します。

リレーションのエンティティ タイプは、その属性によって表されます。 リレーションは、各属性が取り得る値の範囲を組み合わせたサブセットです。

リレーションは、次のようにグラフィックで示されます。

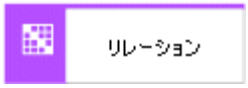


図 62: リレーションのグラフィック

これで、すべての eERM エンティティ タイプはリレーション図で関係を構成します。 eERM の関係タイプを実現する場合、その関係タイプに独立したリレーションを作成するかどうかを判断する際に、多重度がとても重要な要素になります。 1:n 関係と異なり、n:m 関係は、個別のリレーションにそれぞれ表す必要があります。

次に、リレーション図は、リレーションが表す eERM のエンティティ タイプまたは関係タイプを示します。

また、属性を列挙することによって、リレーションをより詳細に指定できます。 対応する属性がキー属性、外部キー属性、記述属性のいずれであるかは、リレーションとその属性を接続する関連する接続線を指定して定義できます。 また、すべての属性に関して、表される要件定義の ERM 属性への関係も構築できます。

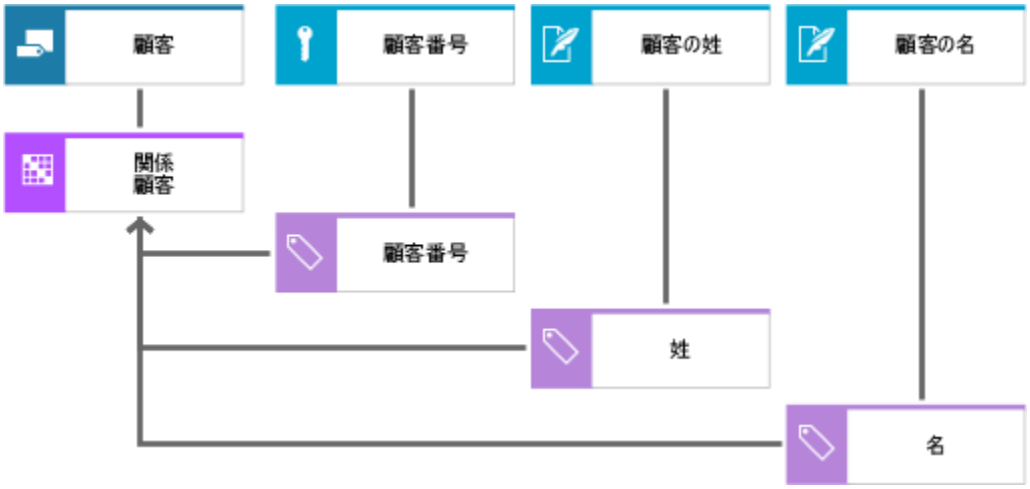


図 63: 要件定義属性およびデータ オブジェクトの割り当て

表現の複雑さを低減するために、各リレーションの属性は、リレーションと接続されている属性割当図に定義できます。 次の図に例を示します。

図 64: 属性割当図

仕様設計では、要件定義のクラスター/データ モデルは、「ビュー」という別個のオブジェクト タイプを使用して実現されます。クラスター/データ モデルの定義に基づいて、「ビュー」は次のように定義されます。

「ビュー」は、複数のリレーションの論理ビューです。

「ビュー」に割り当てられたリレーションも、リレーション図に表すことができます。

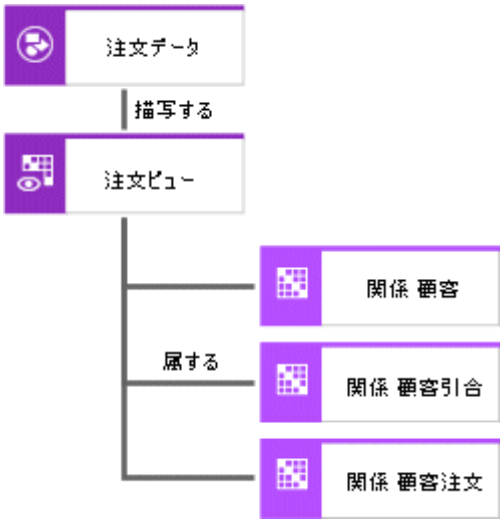


図 65: ビューの定義

ERM の 1:n 関係は、リレーション図の個別のリレーションでは表されません。 この関係は、上位のエンティティ タイプのキー属性を下位のエンティティ タイプのリレーションに統合することによって示されます。 このプロセスで、元々のキー属性がリレーションの外部キーになります。

リレーション図で使用できる接続線は、ERM の関係タイプを示すリレーション図の属性を表すこともできます。

図 66: ERM 関係タイプの属性への割り当て

リレーション図で使用可能なオブジェクト タイプと関係タイプの一覧は、インストール媒体に含まれている『ARIS メソッド マニュアル - テーブル』マニュアル (ARIS Method tables.pdf) を参照してください。

4.2.2.2 システム インターフェイス モデル - システム属性、システム属性ドメイン

[システム属性] モデル タイプは、主に ARIS Architect でデータ エクスポート指向のタスクを実行するために設計されました。 このモデル タイプを使用すると、エンティティ タイプ、イベント、用語、ファンクション、情報媒体、組織ユニット、および要員を階層的に配置できます。また、それらのデータ処理要件に従って一意に包括的に定義できます。 このデータは、プライマリ キーおよび外部キー、記述フィールドおよび必須フィールドなど、通常のデータベース要件に基づいて類型化できます。 これらのデータ オブジェクトのドメイン タイプを決定するために、[システム属性ドメイン] モデル タイプを割り当てることもできます (以下を参照)。

ERM 属性とは対照的に、システム属性の主な機能は、インターフェイス指向のデータの表現と管理です。 エクスポートされる内容の柔軟性を高くするために、システム属性オブジェクトには関連情報を入力する値フィールドが 2 つあります。

次の例は、プロジェクト管理システムに転送する、ARIS Architect を使用して定義された、あるプロジェクトのプロジェクト ヘッダー定義の一部です。

図 67: [システム属性] モデルの例

[システム属性ドメイン] モデル タイプは、データ型に基づいてシステム属性オブジェクトを定義するために使用されます。たとえば、ドメイン タイプ (char、int、date など) およびフィールド長を規定します。 このモデル タイプは、主に、データを外部システムにエクスポートするときの情報を提供するために使われます。

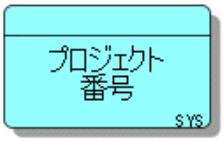


図 68: システム属性ドメイン

4.2.3 実装 - テーブル図

テーブル図を使用して、データベース システムのテーブルおよびフィールドを記述できます。 次の図にテーブルおよびフィールドのグラフィックを示します。

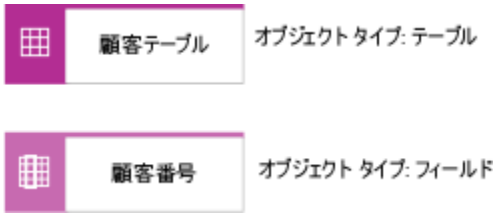


図 69: テーブルとフィールドのグラフィック

このテーブルに割り当てられている個々のフィールドは、テーブルごとに表示できます。 詳細に記述する場合は、並べ替えのためのインデックスやドメインを各フィールドに割り当てることができます。 次の図に例を示します。

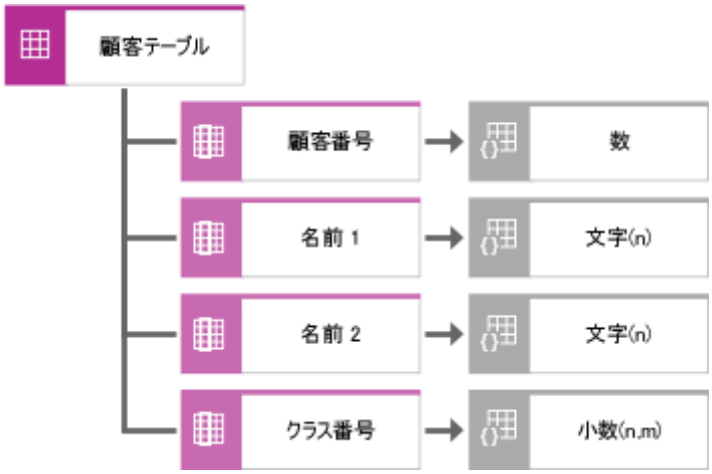


図 70: フィールドの割り当て

リレーション図のリレーションが、(たとえば、データベースのパフォーマンスの理由などから) 必ずしも 1 対 1 にテーブルおよびフィールドに変換されるとは限らないので、テーブルとリレーションまたはエンティティ タイプとの間に、複数方向の関係が生じる場合もあります。 これらの関係は、関連する接続線を選択することによって、テーブル図に表すことができます。 要件定義で定義されたクラスター/データ モデル、またはリレーション図で定義されたビューは、テーブル図では、[ビュー (物理的)] オブジェクトとして表されます。

企業内で使用されるデータベース テーブルとフィールドを変換または記録するために、リレーショナル スキーマの定義は必ずしも必要ではないため、リレーション (または属性) とテーブル (またはフィールド) の間、およびエンティティ タイプ (または ERM 属性) とテーブル (またはフィールド) の間にも実現関係を表すことができます。

表現方法としては、テーブルおよびフィールドによって実現されたリレーションおよび属性に焦点を置く方法と、リレーションの定義を省略し、テーブルおよびフィールドによって示されるエンティティ タイプ、関係タイプ、ERM 属性に焦点を置く方法があります。両方のタイプの方法を次の図に示します。

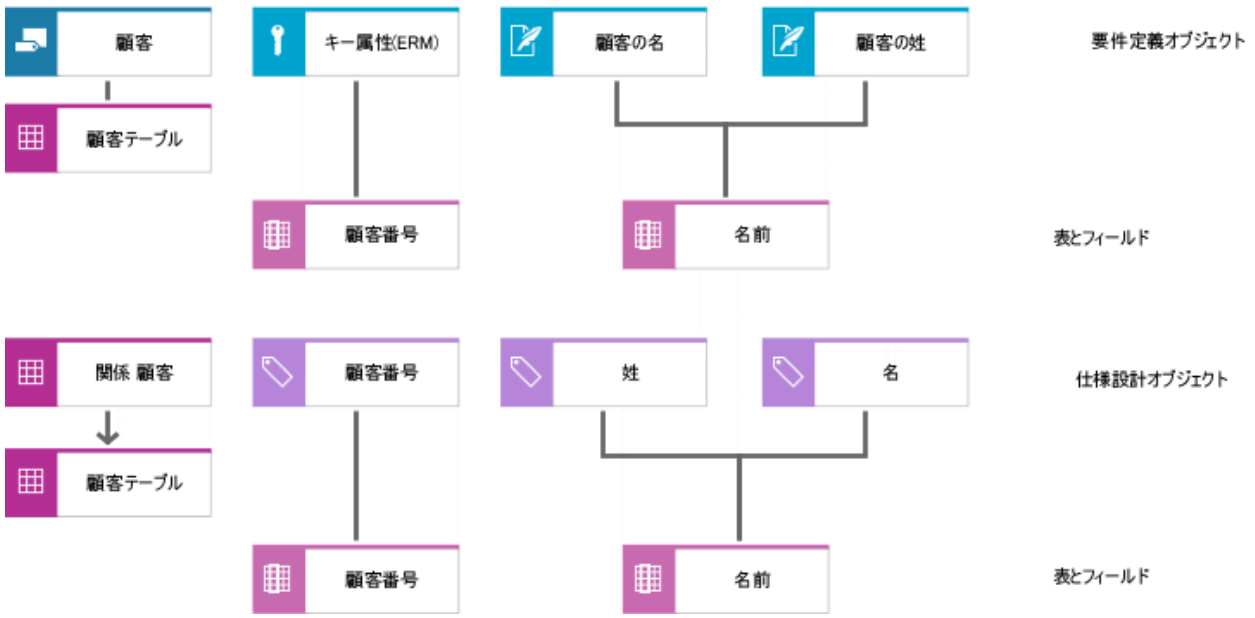


図 71: 要件定義および設計仕様オブジェクトの割り当て

特定のテーブルおよびフィールドの社内での正確な場所を決定するには、テーブルのすべての具体例を定義できなければなりません。 これは、組織ユニットに関してテーブルおよびフィールドへのアクセス権を指定する場合にも同様です。 先に導入された [テーブル] オブジェクト タイプによって、物理的なテーブルおよびフィールドの「タイプ レベル」での論理構造が決定します。 ただし、このようにして定義された各テーブルの複数の具体例は、異なるメディアに格納されているか、企業のさまざまな場所にあります。 このような事実を表す手段として、[テーブル (実例)] および [フィールド (実例)] オブジェクト タイプを使用できます。

これらのオブジェクトを使用すると、テーブルまたはフィールドの実例数を正確にカウントできます。 次の図にこれについて示します。

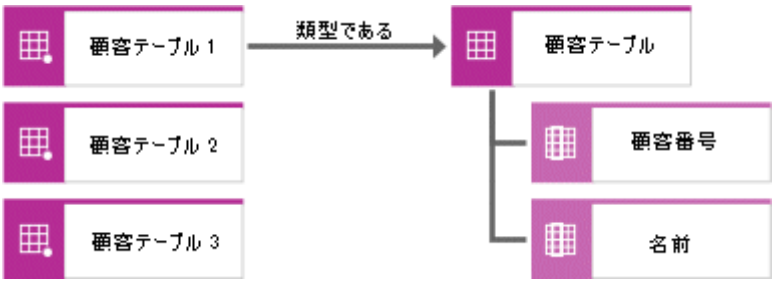


図 72: テーブルの実例

テーブル図で使用可能なオブジェクトと関係の一覧は、インストール媒体に含まれている『ARIS メソッド マニュアル - テーブル』マニュアル (ARIS Method tables.pdf) を参照してください。

4.3 組織ビュー

4.3.1 要件定義

4.3.1.1 企業の組織構造

企業は、管理しやすい単位に分割された複雑な社会構造です。 この複雑さに対処するために、構造パターンを定義して、規則を確立します。 このプロセスの結果を組織と呼びます。 最近まで、情報システム開発の一側面としての組織分析の役割は、ほとんど研究の対象ではありませんでした。 ただし、リーン生産、リーン経営、CIM などの比較的新しいビジネス概念は、判断領域である組織設定と密接に関連しています。 このため、ARIS の概念には、組織を記述するための独立した説明ビューがあります。

企業の組織構成では、組織構造と処理組織を区別できます。

組織構造には、その企業を静的に構成する規則が含まれます。 処理組織には、その企業によって実行されるタスクに関係する規則が含まれます。 このタスク関連の構造は、タスク実行者にファンクションを提供するという意味で、ARIS ハウスのプロセス ビューで扱います。 組織ビューは、基本的に、企業の組織構造を対象とします。

調整に必要な負担を減少させるという目的を持つ、理想的な企業組織の設計は、その企業のビジネス環境と目標に依存します。 そのため、参照構造として使用できる、普遍的に有効で理想的な組織構造を定義することはできません。

組織ユニットの構成は、さまざまな基準により異なります。

非常に一般的な基準は機能的な構造です ここでは、企業の機能 (調達、製造、財務、販売) が、すべての製品と領域に責任を持つとします。 この手法の利点には、スタッフの専門性の高さがあります。しかしながら、機能領域の間の通信と調整に関する負担が多くなります。

情報システムの設計と利用においては、企業の機能分割に長い間焦点が置かれていました。 しかし、類似したデータ オブジェクトのまとまった処理として統合されたプロセス連鎖を考える場合、そのような構造的設計の各機能間の相互関係を確立することは困難です。

このため、統合化されたデータ処理についての議論の結果、異なる機能をサポートするための一貫したデータの基礎が必要であるという結論に至りました。 ただし、機能を統合することにより、機能構造によって複雑性を低減させるという望ましい効果が妨げられます。

したがって、機能統合を達成するという目的を扱う場合は、組織分割のほかの基準が使用されることがよくあります。

たとえば、部門や製品などの基準に焦点を置いて分割されます。 次の図に、製品による分割 (Scheer 著『Business Process Engineering』(1994 年)、26 ページ以降を参照) の図を示します。

部門に基づく組織構造では、組織ユニットは企業または企業の部門の地域分布によって規定されます。 この種の構造は、特に販売部門に適しています。これは、法律の違いなど地域的な要因に適切に対応できるからです。

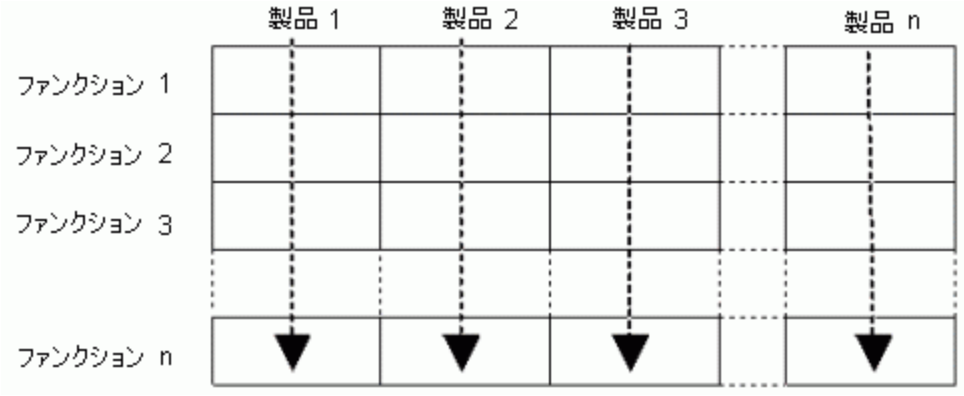


図 73: 製品による組織分割

製品に基づく組織構造は、製品または製品グループの組織ユニットを定義します。 製品グループ内で、特定の製品グループに関連するできるだけ多くの機能を統合します。 この手順の目的は、機能構造に必要な通信の負担を削減することです。しかし、この結果、製品グループに基づくサブシステム間に情報を交換する必要性が出てきます。

これらの影響に対処するため、ハイブリッドな組織形態が作成されることがよくあります。 次の図に、購買を例にしてこれを示します (Scheer 著『Business Process Engineering』(1994 年)、26 ページ以降を参照)。

	製品グループ 1	製品グループ 2	製品グループ 3
中央購買		供給業者の選択	
		契約の合意	
スケジュール作成			
注文			
経理管理			

図 74: ハイブリッドな組織形態

純粋な機能構造を使用することにより、中央の購買部門がすべての製品グループに責任を持つことが表されます。 この場合、製品グループ間で生じる相乗効果を利用できます。ただし、すべてのサブファンクションにおいて 1 つの購買手順が実行されるため、大幅に調整が必要になるという問題が発生します。 購買機能を各製品グループに従って分割する場合は、製品グループごとに、すべての購買機能を実行する購買部門を設けなければなりません。 たとえば、供給業者の選択や契約交渉の際には、非常な調整努力をしないと、相乗効果を得ることができない可能性があります。

上の図に示したように、高い相乗効果が期待できる購買機能は、機能レベルに分割されます。機能レベルに分割されるとは、たとえば、購買機能が中央の購買部門で実行される場合などです。 個々の製品グループの特定の要求や制約に準拠する必要がある機能は、オブジェクト指向的な手法で製品グループに分割されます。 これらの機能は、ただちに各製品グループのプロセス フローに統合されます。 これは、プロセスの組織的な処理は分散された組織ユニットで処理されるが、これらの分散された組織ユニット間の関係は上位および中央の調整レベルで処理されていることを意味します。

これらの柔軟な組織形態は、プロセス指向を重要視したアプローチであるため、ARIS 概念では特に重点が置かれます。 利益中心の概念などの特別に会計指向のアプローチでは、さまざまな分割基準が同時に適用されるような組織形態をとることが必要です。

4.3.1.2 組織図

組織図は、組織構造を表す典型的な形式です。 この図には、選択された構成基準に基づいて、(タスク実行者としての)組織ユニットとそれらの相互関係が表されます。

組織ユニットとは、業務目標を達成するために実行すべきタスクの担い手です。

組織ユニットは、関係を使用して接続されます。 次の図に例を示します。

階層関係をより正確に定義するために、組織ユニット間で使用できるさまざまな接続線タイプが区別されます。 組織ユニット間に次の接続線タイプのいずれかを使用できます。

- 技術的に上位にある
- 専門的に上位にある
- 構成する

機能上の責任が記録されている場合、組織図はビジネス タスクの分布を示します。

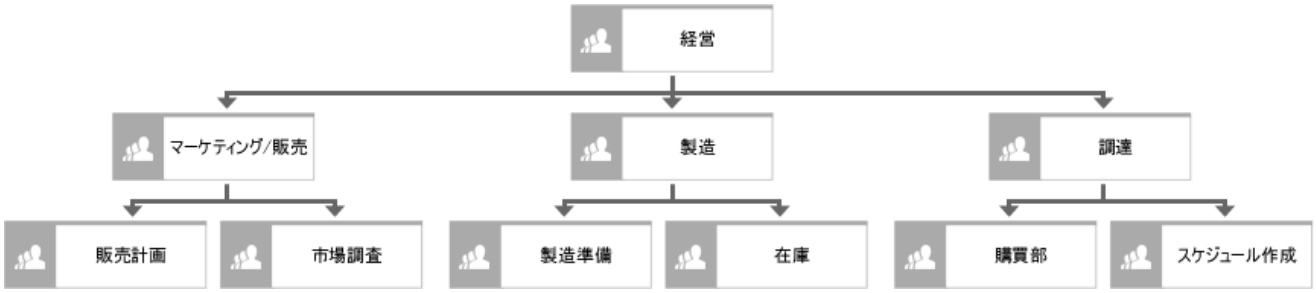


図 75: 組織図

企業内の個々の役職、たとえば記述がある役職を表すために、[役職] オブジェクト タイプを利用できます。 次の図に、このオブジェクト タイプを示します。 複数の役職を 1 つの組織ユニットに割り当てることができます。 接続線の意味は、組織ユニット間の相互作用に対応します。

役職および組織ユニットには、企業内の要員を割り当てることができます。 ARIS では、要員に別々のオブジェクトが提供されています。これを、次の図に示します。 各要員と組織ユニットの関連は、その要員が従業員としてその組織ユニットに割り当てられていることを表します。一方、個々の役職との関連は、企業内での現在の職員配置を定義します。 次の図に例を示します。

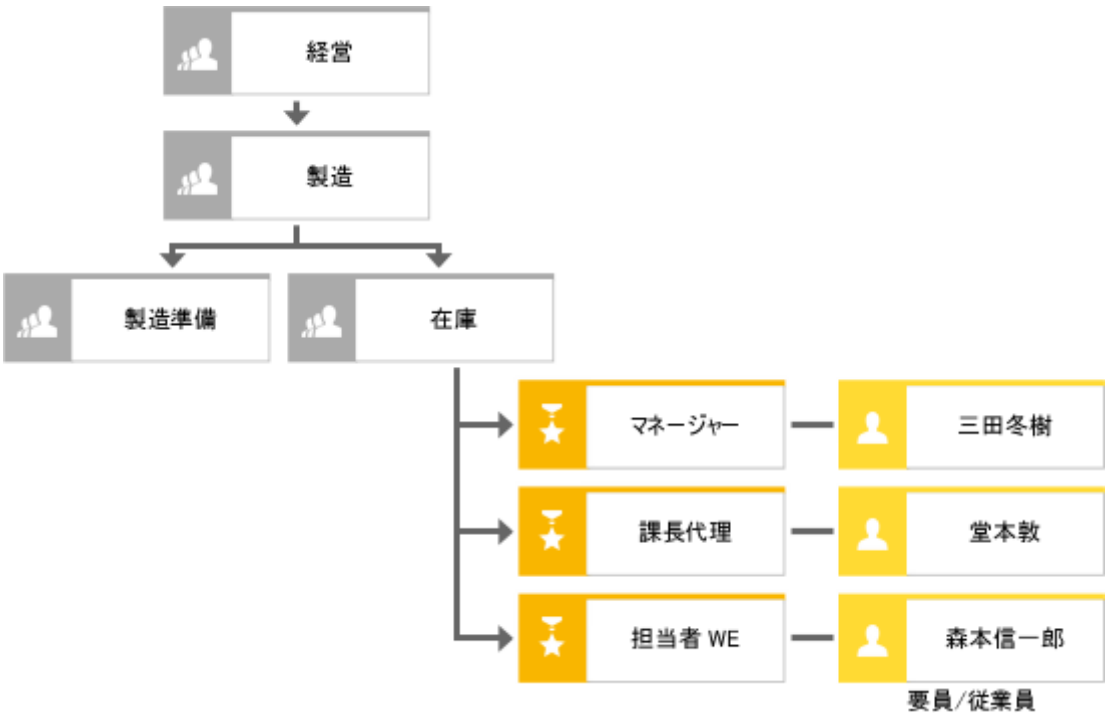


図 76: 役職と要員を割り当てた組織図

組織ユニットと要員は類型化できます。 たとえば、各組織ユニットが、部門、主要部門、またはグループなどであるかを定義できます。 要員は、「部門長」、「グループ リーダー」、または「プロジェクト管理者」などのロールに割り当てることができます。

この目的に使用できる [組織ユニット タイプ] オブジェクトと [ロール] オブジェクトによって、類型化が表されます。 次の図に、組織ユニットおよび要員のタイプをまとめた例を示します。

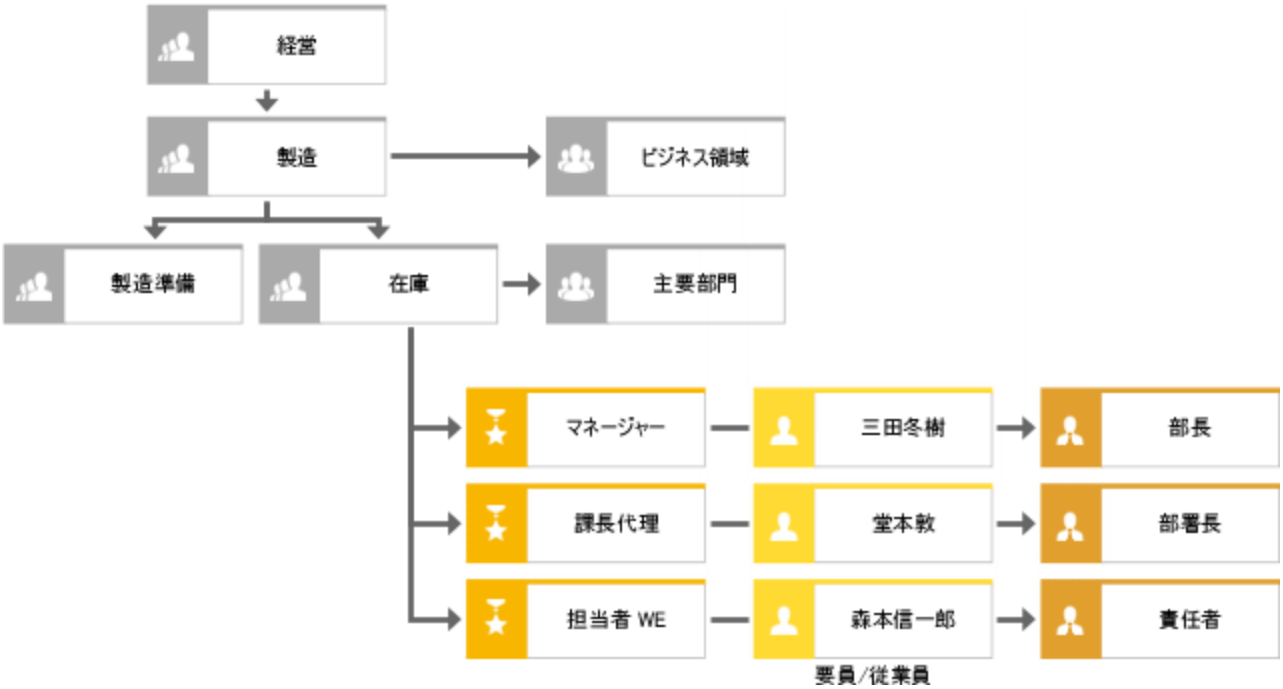


図 77: 要員タイプ

これらのオブジェクト タイプを使用して、企業の実際の組織ユニットや社員から、一般的なビジネス ルールを表すことができます。 このように、プロセス連鎖では、特定のロールのみが機能を実行できる、または情報オブジェクトにアクセスできることを定義できます。

企業の組織構造のモデリングは、仕様設計レベルで定義されるネットワーク トポロジを考える開始点です。これは、組織構造を最善にサポートする目的を持ちます。 ネットワーク トポロジの定義に含まれるものには、ネットワーク接続およびネットワーク ノードがあります。これらは、企業内の特定の場所に配置されることがあります。 その結果、組織ユニットの場所は、組織ビューの要件定義と仕様設計との間の最も重要な関連となります。 このため、各組織ユニットの場所は、要件定義に既に定義されています。 次の図に例を示します。

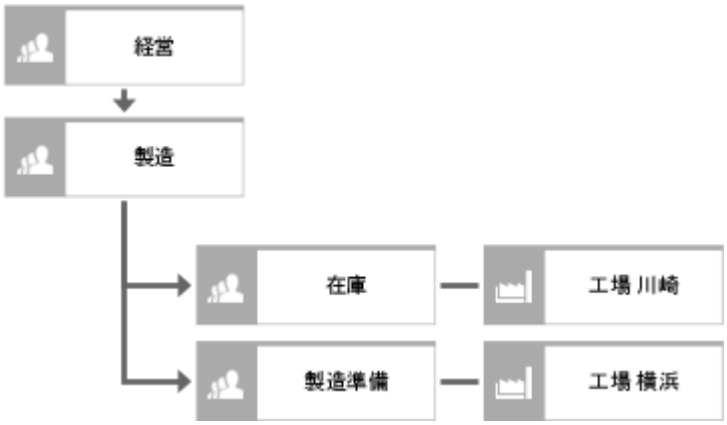


図 78: 場所の割り当て

場所は、任意の階層に配置できます。 場所は、工場全体、建物、または詳細な分析をする場合は、個々のオフィスや部屋の中の 1 台のワークステーションでもかまいません。 仕様設計では、組織ユニットのワークステーションそれぞれにネットワーク ノードを割り当てることができます。 たとえば、特定のオフィス (部屋番号 202) 内で、合計 3 つのネットワーク ノードを利用できることが規定されます。

次の図に場所の階層の例を示します。

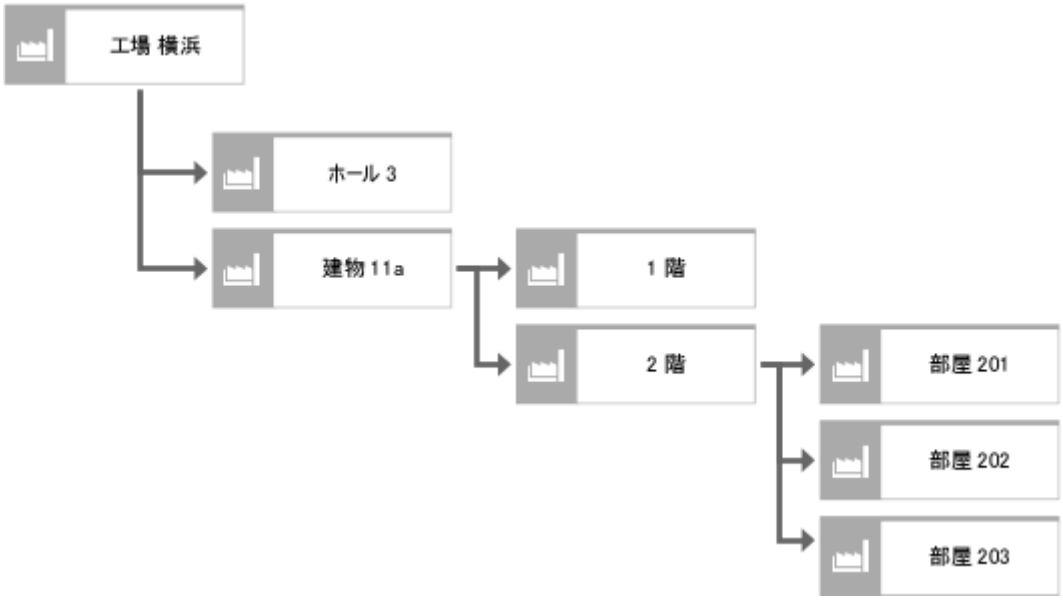


図 79: 場所の階層

4.3.1.3 シフト カレンダー

リソースが使用可能になる時を指定するシフト カレンダは、人材リソースや技術的リソースに割り当てることができます。

組織図や EPC は、シフト カレンダーをリソースに割り当てる際に使用できる適当なモデルです。 どの人材リソースや技術的リソースにも割り当てることができます。 人材リソースの階層では、関連するカレンダは常に最下位階層レベルにあります。

シフト カレンダーは複数レベルのオブジェクト モデルです。 最下位レベルには [休憩時間] タイプがあります。 休憩時間とは、のシフト内で作業を行わない毎日の時間の間隔を指します。 休憩時間は、相対的開始時間およびその期間によって定義されます。 相対的開始時間では、休憩時間が割り当てられるシフトが常に参照されます。 たとえば、シフトの開始が午前 8 時で、休憩の相対的開始が 2 時間である場合、休憩の開始時刻は午前 10 時になります。

次の階層レベルには [シフト] タイプのオブジェクトが含まれます。 シフトとは、作業を行う毎日の時間の間隔を指します。 シフトは、シフト開始 (相対) およびその期間によって定義されます。 1 つのシフトには 1 つ以上の休憩時間が含まれることがあります。 休憩の相対的開始時間はシフト時間内になければなりません。

シフトの例として、早朝シフト、昼間シフト、夜間シフト、および終日シフトが挙げられます。 各シフトは 24 時間ごとに繰り返されます。 シフト サイクルとは、作業が行われる毎週または複数日の時間の間隔を指します。 シフト サイクルにより、あるシフトの稼働日と非稼働日が決まります。 シフト サイクルは、サイクル開始 (相対) およびサイクル期間によって定義されます。 シフト サイクルが継続的に繰り返される場合の条件は、[サイクルで繰り返す] 属性を使って定義することができます。 さらに、[周期] 属性によって、サイクルの繰り返し回数が定義されます。

通常、シフト サイクルは 1 週間または 2 週間をカバーします。 したがって、従業員は 1 週間を早朝シフトで勤務し、次の週を昼間シフトで勤務することができます。 このシフト順序は、シフト サイクルを使用して繰り返すことができます。

例

前述の例の場合、2 つのシフト サイクルが次のように定義されます。

- 早朝シフトのシフト サイクル:

  サイクル開始 (相対) = 0

  サイクル期間 = 5 日

  サイクルで繰り返す = yes

  周期 = 14 日
- 昼間シフトのシフト サイクル:

  サイクル開始 (相対) = 7 日

  サイクル期間 = 5 日

  サイクルで繰り返す = yes

  周期 = 14 日

個々のシフトは 14 日 (2 週間) 周期で繰り返されます。 同じ従業員が 4 週間ごとの周期で土曜日の早朝シフトに勤務する場合、3 番目のシフト サイクルは次のように定義することができます。

- 早朝シフトのシフト サイクル:

  サイクル開始 (相対) = 20 日

  サイクル期間 = 1 日

  サイクルで繰り返す = yes

  周期 = 28 日

前述の例を、下のモデルに示します。 1:n でのシフトおよびシフト サイクルの割り当てを見ることができます。

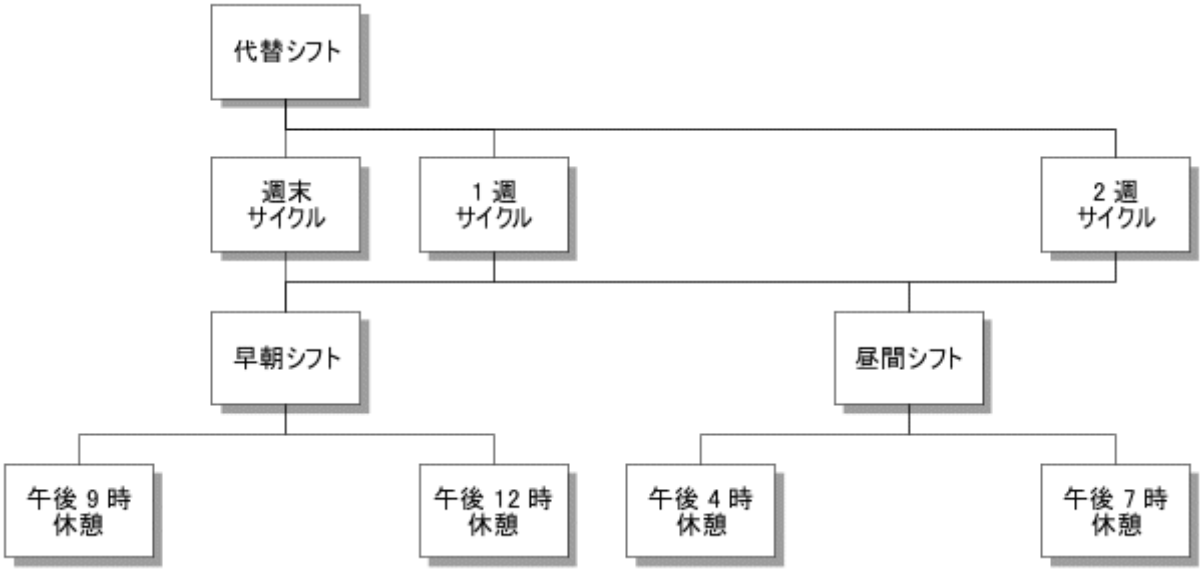


図 80: シフト カレンダーの例

シフト カレンダーとは、リソースの作業時間を記述する、すべてのシフト サイクルおよび関連シフトの合計数を指します。 この記述には、周期的に繰り返される部分のみ含まれます。特別な規則で管理される休暇、病欠、休日、または勤務していないほかの日は、シフト計画には含まれません。

[シフト計画] オブジェクト タイプには、[計画開始] および [計画期間] 属性が含まれます。 これらの属性により、シフト カレンダーが有効である間の期間が定義されます。 また、シフト カレンダには [サイクルで繰り返す] および [周期] 属性もあります。

4.3.2 仕様設計 - ネットワーク トポロジ

組織図に示された企業の組織構造は、通信および情報システム インフラストラクチャによってサポートされます。 これらの情報システムの構造的な要件は、一般に仕様設計において、ネットワーク トポロジの形式で定義できます。

最初に、各種ネットワーク タイプを [ネットワーク トポロジ] モデルに組み込むことができます。

ネットワーク タイプは、まったく同じ技術に基づいた個々のネットワークの具体例を類型化したものです。

次の図にネットワーク タイプの例を示します。

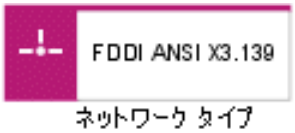


図 81: ネットワーク タイプ図のグラフィック

[ネットワーク タイプ] は相互に連結することができます。また、論理的な構造を持つので、階層的に配置できます。

ネットワーク ノード タイプとネットワーク接続タイプは、各ネットワーク タイプに割り当てることができます。 したがって、企業用に特定の [ネットワーク タイプ] を選択した結果生じる技術的な制約をただちに識別できます。 すべての [ネットワーク接続タイプ] に関して、その末端の [ネットワーク ノード タイプ] を確認できます。

[ハードウェア コンポーネント タイプ] に関しては、あらかじめ定義されているネットワーク構造を実現するための [ネットワーク ハードウェア]、または [ネットワーク ノード タイプ] に接続されている [ハードウェア コンポーネント タイプ] のいずれかを指します。

[ハードウェア コンポーネント タイプ] は、[アプリケーション システム] や [ネットワーク タイプ] と同様に、(たとえば、企業によって割り当てられた資産番号によって) 識別可能な [ハードウェア コンポーネント] の個々の具体例を表すものではありません。 同じ技術に基づいたすべての [ハードウェア コンポーネント] を類型化したものです。 [ハードウェア コンポーネント タイプ] は、必要な任意の階層に配置できます。

[ハードウェア コンポーネント タイプ] は、まったく同じ技術に基づいた個々の [ハードウェア コンポーネント] の具体例を類型化したものです。

ネットワーク ノード タイプやネットワーク接続タイプと一緒に、ネットワーク トポロジの一種のリファレンス モデルを、この段階で作成できます。 このモデルは、特定のネットワーク接続タイプやネットワーク ノード タイプを実現するために、どのハードウェア コンポーネント タイプを使用するかを表します。 ネットワーク接続タイプの例には、ある特定の種類の転送ケーブルなどがあります。 また、どのハードウェア コンポーネント タイプをどのネットワーク ノード タイプに接続するかを表すことができます。 さらに、ネットワーク ノード タイプは、ネットワーク ノード タイプの作成に使用される、ハードウェア コンポーネント タイプと関係を持つことができます。 次の図に例を示します。

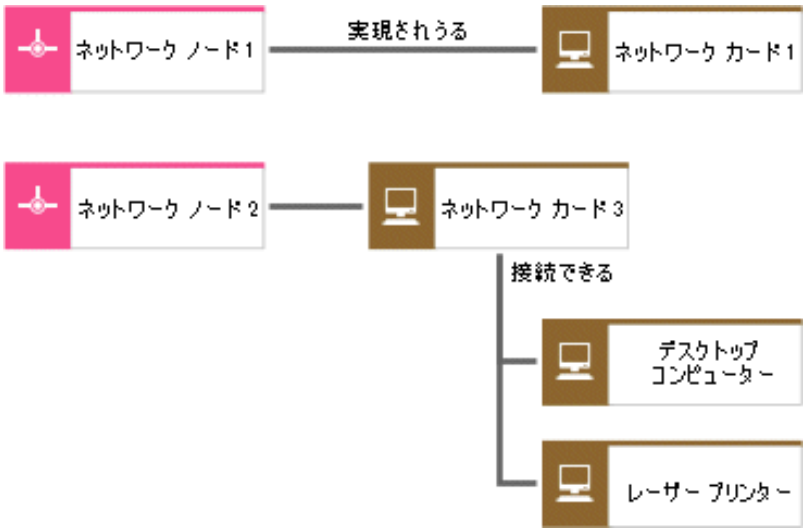


図 82: ネットワーク トポロジ

ネットワーク トポロジと要件定義のオブジェクト間の接続は、次の 2 つのアプローチによって確立されます。

1 つ目は、すべてのハードウェア コンポーネント タイプに関して、それに責任を持つ組織ユニットまたは役職を定義する方法です。

2 つ目は、各ネットワーク タイプ、ネットワーク ノード タイプ、ネットワーク接続タイプ、およびハードウェア コンポーネント タイプを置く企業内の場所を定義する方法です。 このように、場所は、組織ビューの要件定義とその仕様設計との間の中心的な関連になります。

ネットワーク トポロジ モデルで使用可能なオブジェクト タイプと関係タイプの一覧は、インストール媒体に含まれている『ARIS メソッド マニュアル - テーブル』マニュアル (ARIS Method tables.pdf) を参照してください。

4.3.3 実装

4.3.3.1 ネットワーク図

ネットワーク図は、仕様設計で定義したネットワーク トポロジの実現を表します。

[ネットワーク] オブジェクトによって、企業内のネットワークが記録されます。 すべてのネットワークに関して、ネットワーク ノードとそれに含まれるネットワーク接続を指定できます。

企業内のすべてのネットワーク、ネットワーク ノード、およびネットワーク接続の正確な場所を指定できます。 場所には、工場全体、特定の建物、建物コンプレックス、オフィスや各ワークステーションなどが指定できます。

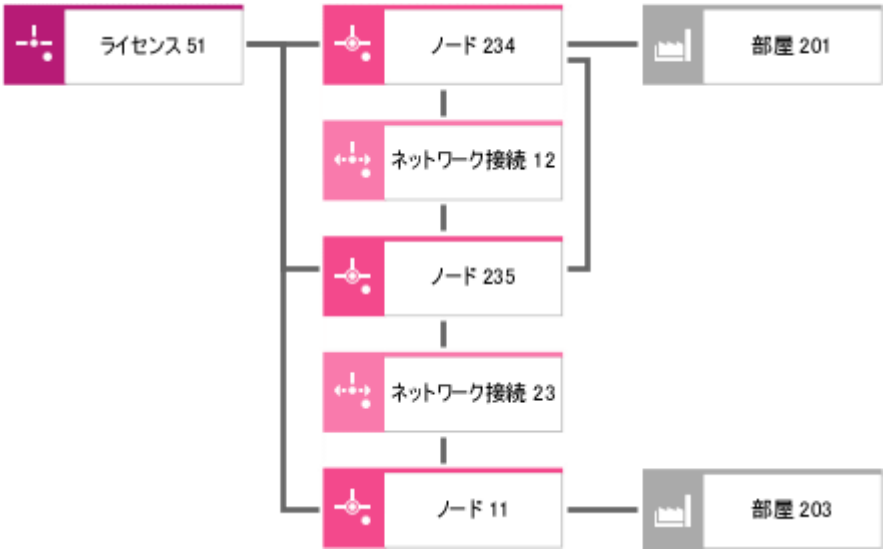


図 83: 場所の割り当てを含むネットワーク図

また、ネットワーク図は各ネットワーク接続とネットワーク ノードの実現に使用されるハードウェア コンポーネントも記録します。さらに、各ハードウェア コンポーネントの構成を示すことができます。 ハードウェア コンポーネントは、ネットワーク接続およびネットワーク ノードを構成するために使用されます。また、ネットワーク ノードと連結することもできます。 ネットワーク図にこの関係を示すこともできます。 実例レベルの各オブジェクトについて、仕様設計レベルの対応するオブジェクトとの関係もモデル化できます。 たとえば、「FDDI ANSI X3.139」タイプのネットワークが「川崎工場」に設置されていることを表すことができます。

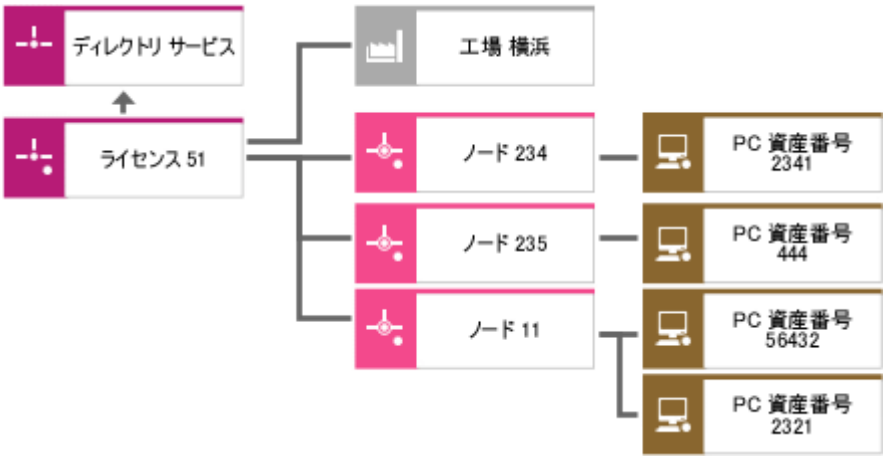


図 84: ハードウェア コンポーネントと場所の割り当てを含むネットワーク図

ネットワーク図では、タイプを割り当てることにより、仕様設計への関係が作成されます。また、ネットワーク コンポーネントを特定の場所に割り当てることにより、要件定義への関係が確立されます。

ネットワーク図で使用可能なオブジェクト タイプと関係タイプの一覧は、インストール媒体に含まれている『ARIS メソッド マニュアル - テーブル』マニュアル (ARIS Method tables.pdf) を参照してください。

4.3.3.2 マテリアル フロー モデリング - 技術的リソース

プロセス モデル (EPC (マテリアル フロー付き)、PCD (マテリアル フロー付き)) でマテリアル フローを図示できるようにするため、材料タイプは、ファンクションの入力または出力という形式で、業務プロセスの個々のファンクションに割り当てられます。 情報オブジェクトのファンクションへの割り当て (情報の変換はファンクションを使用して表現) と同様に、この割り当ては入力材料タイプの出力材料タイプへの変換を表します。 さらに、材料の変換に必要な技術的リソースをプロセス連鎖に記録できます。このコンテキストでは、運用リソース、倉庫設備、運搬システム、および機材を区別できます。

[技術的リソース] モデル タイプでは、技術的リソースを階層的に配置したり、それらにタイプを割り当てたり、分類したりできます。 次のオブジェクト タイプをこの目的に使用できます。

運用リソース

運用リソースとは、業務を実行するために企業が利用できるさまざまな運用リソース タイプの具体例です。 運用リソースは、通常、資産番号 (製造工場の番号など) によって識別します。

運用リソース タイプ

運用リソースとは、まったく同じ技術に基づいた個々の運用リソースを類型化したものです。

運用リソース クラス

類似の運用リソース タイプが集合して運用リソース クラスを形成します。 異なる分類基準に基づいて、類似性を定義できます。 そのため、1 つの運用リソース タイプを複数の運用リソース クラスに割り当てることができます。

倉庫設備

倉庫設備アイテムとは、企業が職務を行うために利用できるさまざまな倉庫設備タイプの具体例です。 倉庫設備アイテムは、資産番号を使用して識別されることがよくあります。

倉庫設備タイプ

倉庫設備タイプは、まったく同じ技術に基づいた個々の倉庫設備アイテムを類型化したものです。

倉庫設備クラス

類似の倉庫設備タイプが集合して倉庫設備クラスを形成します。 異なる分類基準に基づいて、類似性を定義できます。 そのため、1 つの倉庫設備タイプを複数の倉庫設備クラスに割り当てることができます。

機材

機材とは、さまざまな機材タイプの具体例です。 一般に、これは資産番号によって識別できます。

機材タイプ

機材タイプとは、まったく同じ技術に基づいた個々の機材品目を類型化したものです。

機材クラス

類似の機材タイプが集合して、機材クラスを形成します。 異なる分類基準に基づいて、類似性を定義できます。 そのため、1 つの機材タイプを複数の機材クラスに割り当てることができます。

運搬システム

運搬システムは、運搬システム タイプのそれぞれの具体例です。 運搬システムの識別には資産番号や工場番号などを使用できます。

運搬システム タイプ

運搬システム タイプとは、まったく同じ技術に基づいた個々の運搬 システムを類型化したものです。

運搬システム クラス

類似の運搬システム タイプが集合して、運搬システム クラスを形成します。 異なる分類基準に基づいて、類似性を定義できます。 そのため、1 つの運搬システム タイプを複数の運搬システム クラスに割り当てることができます。

[技術的リソース] モデル タイプを階層的に作成するさまざまな機能により、複雑な技術的設備の構成を記述できます。 たとえば、これによって、複雑な製造工場の構成要素や構成要素間の関係を表現できます。

上述したモデリングのオプションのほかに、技術的リソースに対する場所の割り当ておよび組織上の責任を定義できます。 この場合、[組織図] モデル タイプに既出のオブジェクト タイプ [場所]、[組織ユニット]、[グループ]、[役職] および [要員]

が利用できます。 これらのオブジェクト タイプは、[運用リソース]、[倉庫設備]、[機材] および [運搬システム] オブジェクト タイプに接続できます。

次の図に、[技術的リソース] モデル タイプの例を示します。

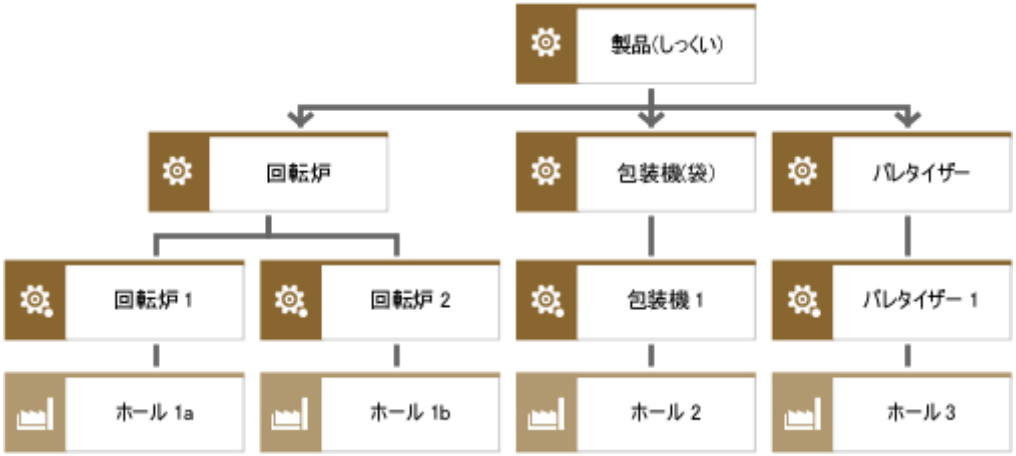


図 85: [技術的リソース] モデルの例

4.4 プロセス ビュー

4.4.1 要件定義

データ ビュー、組織ビュー、ファンクション ビューのオブジェクト間の相互関係は、プロセス ビューで分析されます。 分析する相互関係は、ビュー間の接続線から得られます。

最初に、2 つのビューの間に存在する関係を調査し、その後、図を導入して 3 つのビューの間の関係を描きます。

4.4.1.1 ファンクションと組織の接続 - EPC、ファンクション / 組織レベルの図

ファンクション ビューと組織ビューとのリンクは、ファンクション ツリーで定義されたファンクションを組織図のタスク実行者 (組織ユニット) へ割り当てるために使用されます。 この割り当てによって、割り当てられたファンクションに対して責任を持つ組織ユニットと意思決定権が定義されます。 プロセス連鎖 (業務プロセス) でこうした組織的な割り当てが実現される方法を表すことで、機能統合、すなわち組織ユニットが処理する業務プロセス ファンクションの数を識別できます。

次の図に、組織ユニットのファンクションへの割り当ての例を示します。 この図では、左側に配置されたファンクションには、そのファンクションの実行に責任を持つ組織ユニットが割り当てられています。 ファンクションの階層関係は、ファンクション ビュー（ファンクション ツリー）に表されています。組織ユニット間の関係は、組織ビュー（組織図）に表されています。 したがって、ここでは、それらを定義する必要はありません。

図 86: ファンクションへの組織要素の割り当て

4.4.1.2 ファンクションとデータの接続

4.4.1.2.1 イベント制御 - イベント駆動プロセス連鎖（EPC）

業務プロセスにおけるファンクションの処理手順は、プロセス連鎖で表現されます。 すべてのファンクションの開始イベントと終了イベントを指定できます。 イベントは、ファンクションを開始するだけでなく、ファンクションの結果も表現します。

イベントは、情報オブジェクトが、業務プロセスを制御したりその進行状況に影響を与えたりする、ビジネス管理に関連する状態を持っている事実を表します。 イベントはファンクションを起動するものであり、ファンクションの結果にもなります。 時間の経過を伴うタスクであるファンクションとは異なり、イベントはある時点に結び付けられています。

情報オブジェクトの状態変化は、情報オブジェクトの最初のオカレンス（たとえば、「顧客注文が受け取られた」）、または属性オカレンスに記録された状態変化という意味での状態の変化（たとえば、「見積もりが拒否された」）のいずれかを意味します。情報オブジェクトと属性は ARIS データ ビューに記述されるので、プロセス連鎖のイベント駆動的な表現により、データ ビューとファンクション ビュー間の関係が形成され、ARIS プロセス ビューに割り当てられます。

イベントは、六角形のシンボルで表します。 名前には、情報オブジェクトそのもの（「注文」）だけでなく、その状態の変化（「受領した」）も含めます。 次の図にイベントを示します。

図 87: イベント（グラフィック表示）

イベントはファンクションを起動するものであり、ファンクションの結果にもなります。 このイベントとファンクションを順番に配置することによって、いわゆるイベント駆動プロセス連鎖（EPC）が作成されます。 イベント駆動プロセス連鎖（EPC）は、業務プロセスの時系列の操作順序を表します。

次の図に、EPC の例を示します。 イベントは、ファンクションを起動する状態または条件、およびファンクションの終了を定義する状態を決定します。したがって、EPC の開始ノードと終了ノードは常にイベントとなります。 1 つのイベントから複数のファンクションが同時に起動される場合もあり、また、1 つのファンクションが結果として複数のイベントを持つこともあります。 EPC では、円で表されたルールを使用して、分岐および処理ループを表します。 しかし、これらの接続は、表示されたルールとして機能するだけなく、接続するオブジェクト間の論理的な関連を定義します。

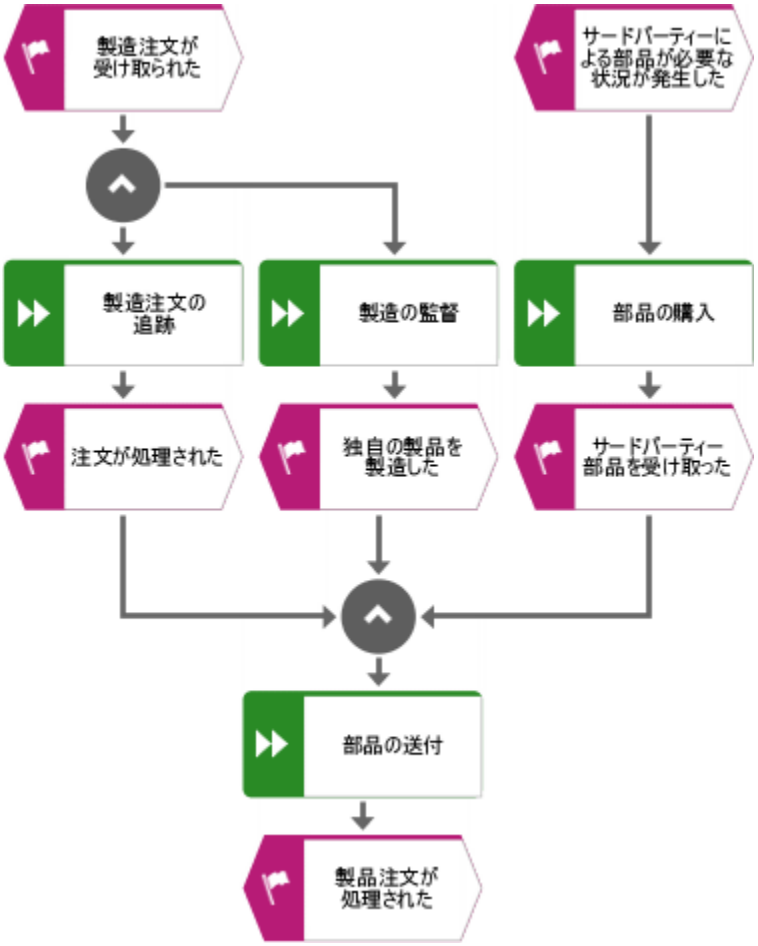


図 88: EPC の例

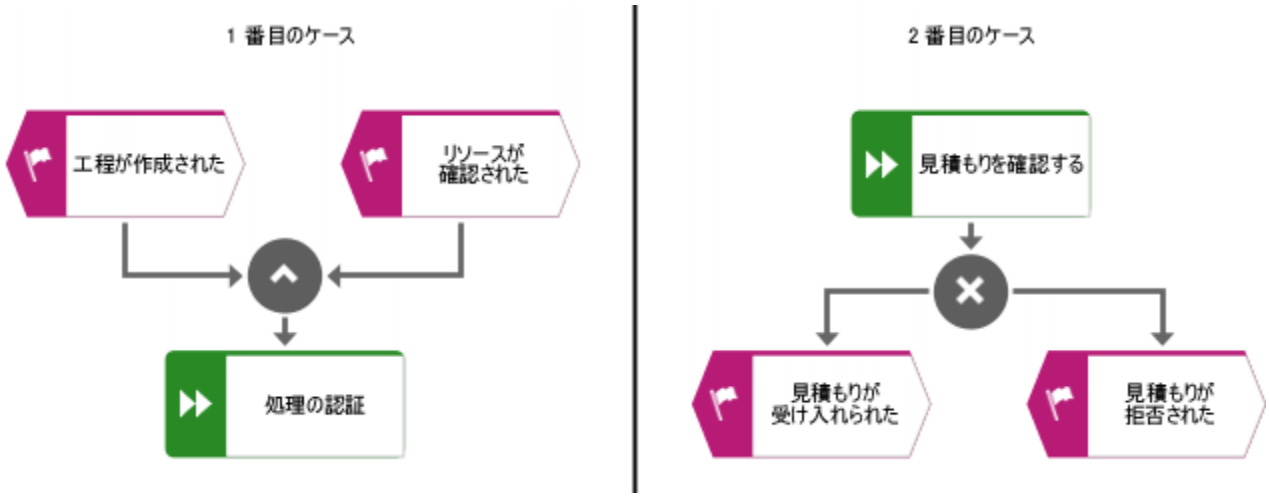


図 89: ルールの例

この図の最初の例では、開始イベントは AND ルールで接続されています。 これは、「作業を開始する」手順は、作業手順書と必要なリソースが使用可能な場合に初めて開始されることを意味します。 つまり、この処理が開始される前に、両方のイベントが発生している必要があります。 2 番目の例では、XOR ルールを使用した排他的 OR 接続が示されています。「業者の見積もりを確認する」ファンクションは、この見積もりを受理するか、拒否するかのいずれかの結果になります。 ただし、両方の結果が同時に発生することはありません。 これらの 2 つのケースのほかにも「包含的 OR」タイプを使って、より複雑な関係を作成することができます。 一般的なルールは EPC で表すことができ、ルール図の形式でより詳細に記述されます。

このように、次の 2 つの異なるタイプの演算子があります。

1. イベント演算子
2. ファンクション演算子

次の図は、可能なすべてのイベント演算子およびファンクション演算子の概要を表しています (Hoffmann、Kirsch、Scheer 著『Modellierung mit Ereignisgesteuerten Prozessketten』(Modeling with event-driven process chains)(1993 年)、13 ページを参照)。

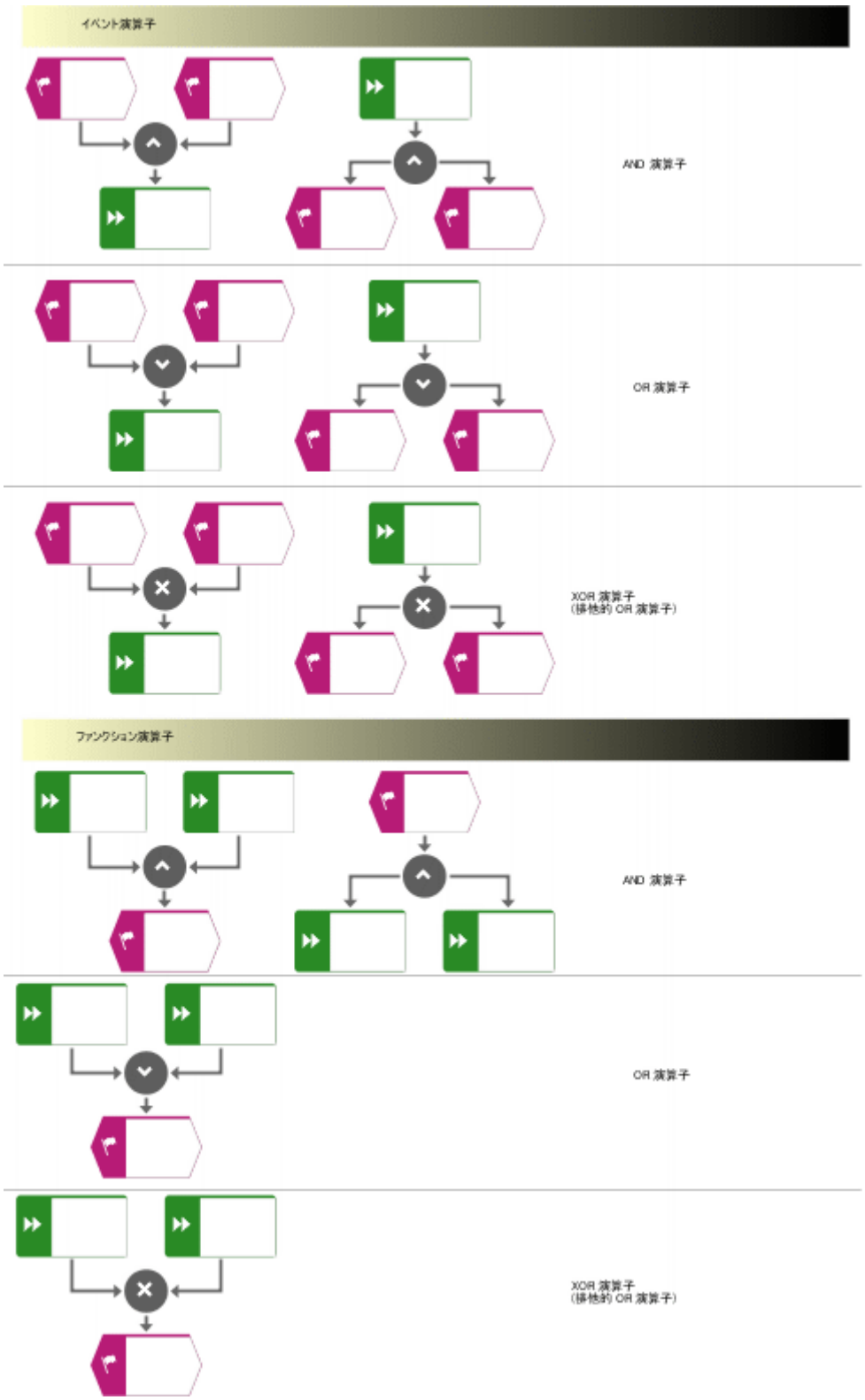


図 90: 論理演算子 (ルール)

このコンテキストでは、ファンクション演算子の持つ制約に特に注意しなければなりません。 イベントは決定を作成できません。決定の作成はファンクションのみが行えます。このため、起動イベントを、OR または XOR ルールを使って接続しないでください。

以下、考えられるルールについて例を示しながら説明します。

起動イベントの接続

AND ルール

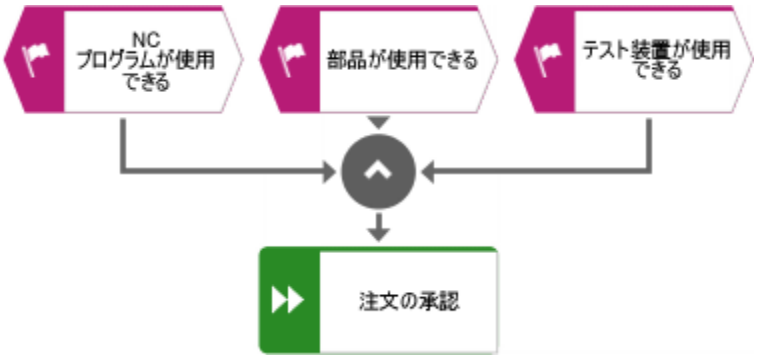


図 91: 起動イベントの AND 演算子

ファンクションは、すべてのイベントが発生したあとに起動されます。

OR ルール

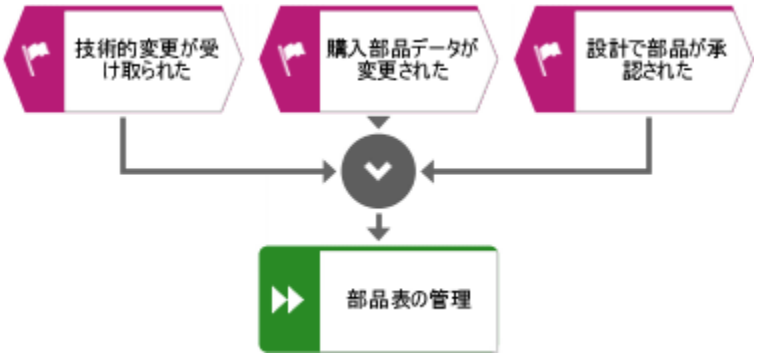


図 92: 起動イベントの OR 演算子

ファンクションは、少なくとも 1 つのイベントが発生したあとに実行されます。

排他的 OR ルール (XOR ルール)

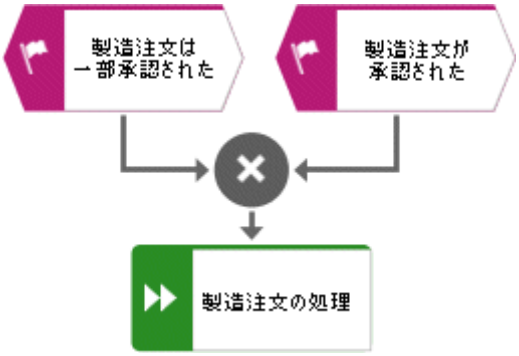


図 93: 起動イベントの XOR 演算子

このファンクションは、いずれか一方のイベントだけが発生したあとにのみ起動されます。

作成されるイベントの接続

AND ルール

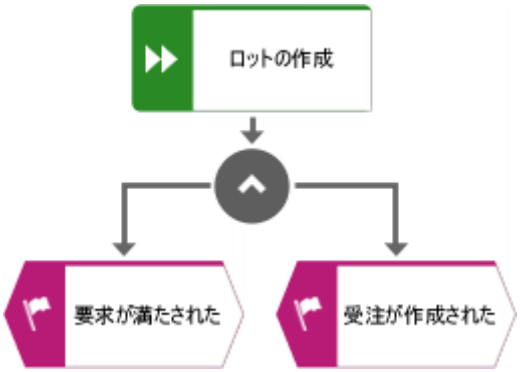


図 94: 作成されるイベントの AND 演算子

すべてのイベントは、ファンクションの実行が完了すると発生します。

OR ルール

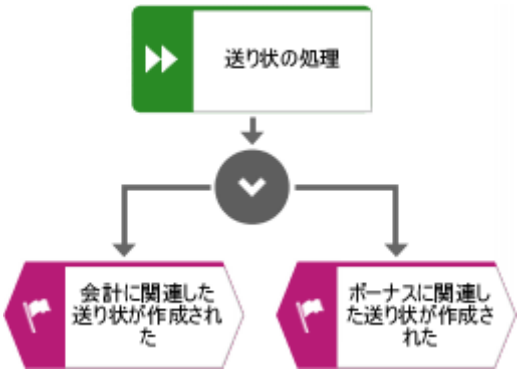


図 95: 作成されるイベントの OR 演算子

ファンクションの実行が完了すると少なくとも 1 つのイベントが発生します。

排他的 OR ルール (XOR ルール)

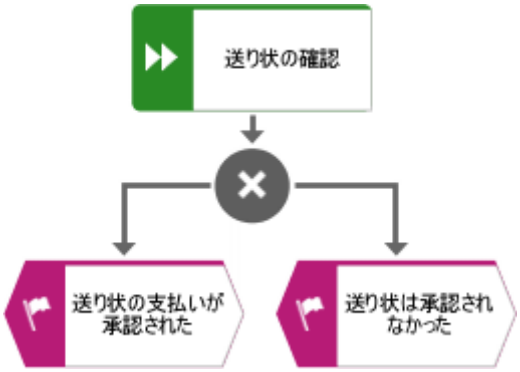


図 96: 作成されるイベントの XOR ルール

ファンクションの実行が完了すると 1 つだけイベントが発生します。

生成されるイベントとファンクションの接続

AND ルール

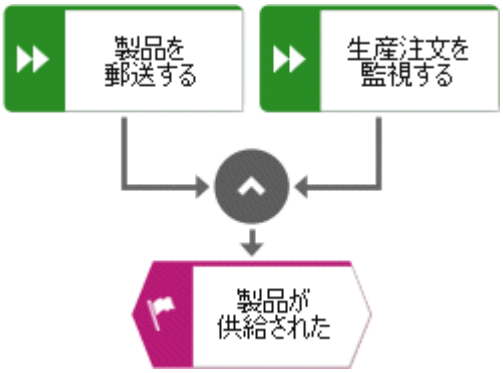


図 97: 生成されるイベントとファンクションの AND 演算子

このイベントは、すべてのファンクションが実行されたあとにのみ発生します。

OR ルール

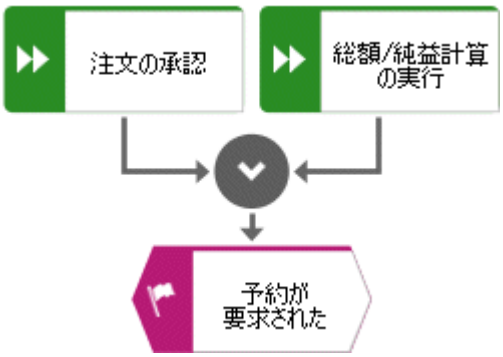


図 98: 作成されるイベントとファンクションの OR 演算子

このイベントは、少なくとも 1 つのファンクションが実行されたあとに発生します。

排他的 OR ルール (XOR ルール)

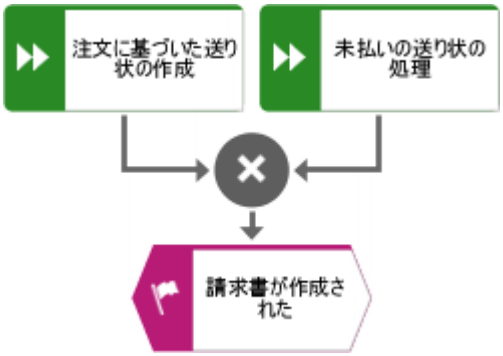


図 99: 生成されるイベントとファンクションの XOR 演算子

1 つのファンクションのみが実行されるとイベントが発生します。

起動イベントとファンクションの接続

AND ルール

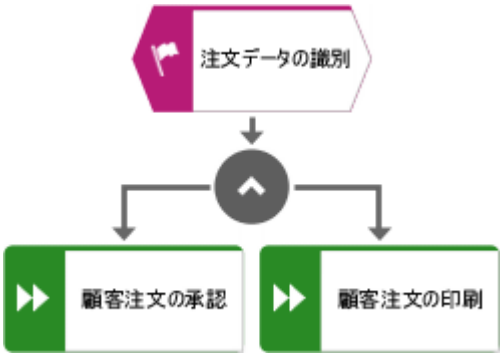


図 100: 起動イベントとファンクションの AND 演算子

このイベントは、すべてのファンクションを起動します。

OR ルール

イベントは決定できないので、 このルールは無効です。

排他的 OR ルール (XOR ルール)

イベントは決定できないので、 このルールは無効です。

イベント駆動プロセス連鎖形式のほかに、これらの分岐は、プロセス連鎖図 (PCD) のイベントおよびファンクション列 (「プロセス連鎖の分析 『17 ページ 』」の章を参照) のテーブル形式でも記述できます。 プロセス連鎖図では、ファンクションが順番に並べられるので、分岐や処理ループを表すと、図が非常に複雑で分かりにくくなる可能性があります。

EPC: BPML エクスポート

ARIS は BPML ファイル エクスポート形式をサポートします。 ただし、BPML の開発はもう行われておらず、BPML を使用できるシステムの数も限定されているため、ARIS による BPML のサポートは間もなく打ち切られます。

4.4.1.2.2 ファンクション割当図 (I/O)

「イベント駆動プロセス連鎖 (EPC)『80 ページ 』」の章で説明したイベント制御の表現に加え、入力データから出力データへの変換とファンクション間のデータ フロー表現が、ARIS 概念におけるデータ ビューとファンクション ビューの間の関係を形成します。 入力データから出力データへの変換は、いわゆる [ファンクション割当図 (I/O)] で表します。これは、基本的には、ほかの手法で使われている入出力図に相当します。 次の図に、ファンクション割当図 (I/O) の例を示します。「出荷日を決定する」ファンクションの入力データは、「部品一覧データ」、「部品データ」、「在庫データ」および「出荷データ」です。「引合データ」は、入力データおよび出力データとしての役割を果たします。 このように、[ファンクション割当図 (I/O)] には、ファンクション ビューのファンクションと、データ ビューの情報オブジェクトが含まれます。 矢印は、情報オブジェクトが、入力データ、出力データ、または入出力データのいずれとしてのみ使用されるかを決定します。 たとえば、ファンクションが情報オブジェクトを作成したり、削除したりすることを示す、詳細な指定もできます。 詳細さの度合いによって、情報オブジェクトは、データ ビューの [クラスター/データ モデル]、エンティティ タイプ、関係タイプ、または属性になります。

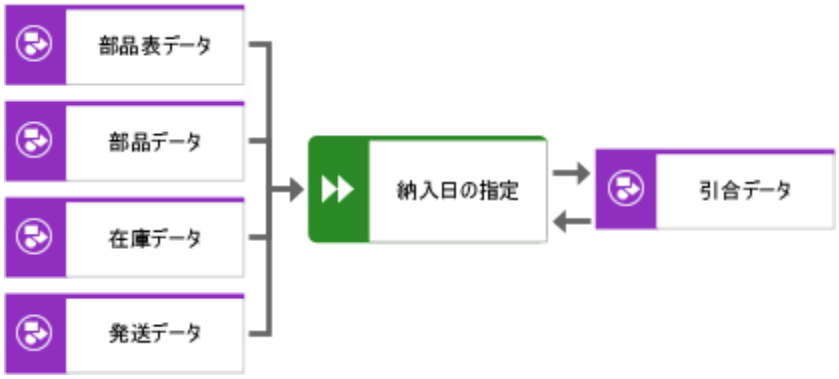


図 101: ファンクション割当図 (I/O) の例

この図は、ファンクションの入出力データを表すという、ファンクション割当図の実際の目的を示しています。

[ファンクション割当図 (I/O)] ではファンクションの入出力データのほかにも、EPC 内のイベントやファンクションに割り当て可能なすべてのオブジェクトが利用できます。 このように、EPC 図では、プロセス連鎖のモデリングをイベントとファンクションの記述に限定しておき、各ファンクションを、ファンクションが持つその他の関係をすべて含む割当図 (I/O) に割り当てることができます。 これにより、業務プロセスの表記法がさらに明確になります。また、このモデル タイプに新しい名前を使用することが説明されます。 次の図に、ファンクション割当図のより詳細な表現の例を示します。

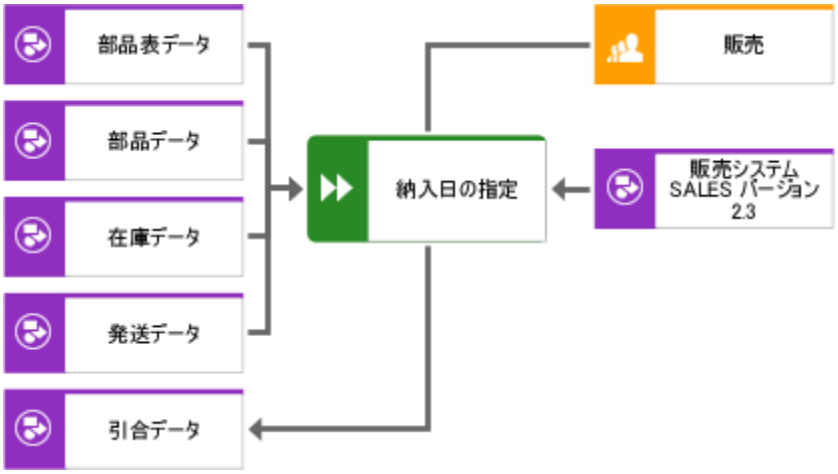


図 102: ファンクション割当図の詳細な表現

[ファンクション割当図 (I/O)] にデータ変換の方法を表すかわりに、この情報を [EPC] に含めて表現することもできます。 次の図に例を示します。 この場合、ファンクションと情報オブジェクト間の関係は、[ファンクション割当図 (I/O)] の中の関係と同じ役割を果たします。 ただし、これらの関係を分岐が多いプロセス連鎖に含めると、表示が非常に複雑になります。

図 103: 入出力データを含む EPC

PCD (プロセス連鎖図) では、オブジェクトは列の説明に従って配置されます。 EPC で表示することにより、自由なオブジェクトの配置が可能です。 ただし、入出力データを追加すると、モデルは複雑でわかりにくくなります。 したがって、操作順序に従う業務プロセスには特に PCD 表示を推奨します。 次の図は、上の図で PCD で示した入出力データを EPC で表したものです (「EPC/PCD 『94 ページ 』」の章も参照)。

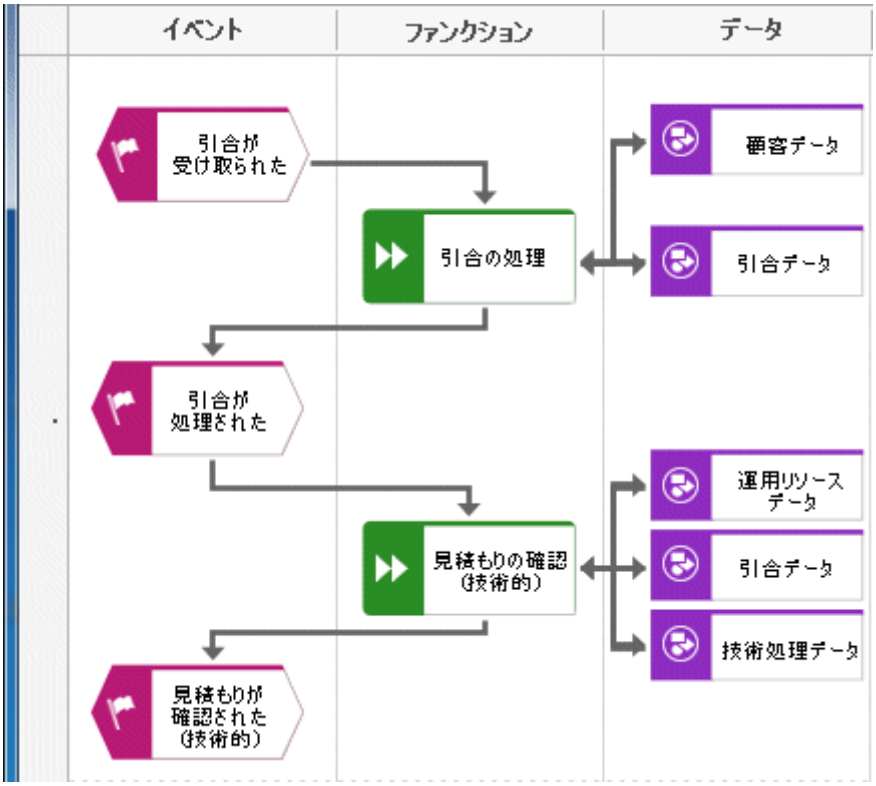


図 104: 入出力データを含む EPC

4.4.1.2.3 情報フロー図

情報フロー図は、ファンクション間のデータ フローを表すのに適しています。 そのため、情報フロー図では 2 つのファンクションをデータ フロー オブジェクトによって相互に接続できます。 データ フロー オブジェクトは、ソース ファンクションからターゲット ファンクションへのデータの流れが存在することを示します。 表示されているファンクションの入力または出力であるデータ オブジェクトをさらに詳しく定義する場合は、データ モデルをこのオブジェクトに割り当てることができます。その結果、このデータ フロー オブジェクトに階層が形成されます。 データ モデルは、ファンクション間で交換される情報オブジェクトを示すために使用されます。対象のファンクションの詳細さの度合に応じて、この情報オブジェクトは、[クラスター/データ モデル]、エンティティ タイプ、または ERM 属性のいずれかになります。 このタイプの表現の例を次の図に示します。

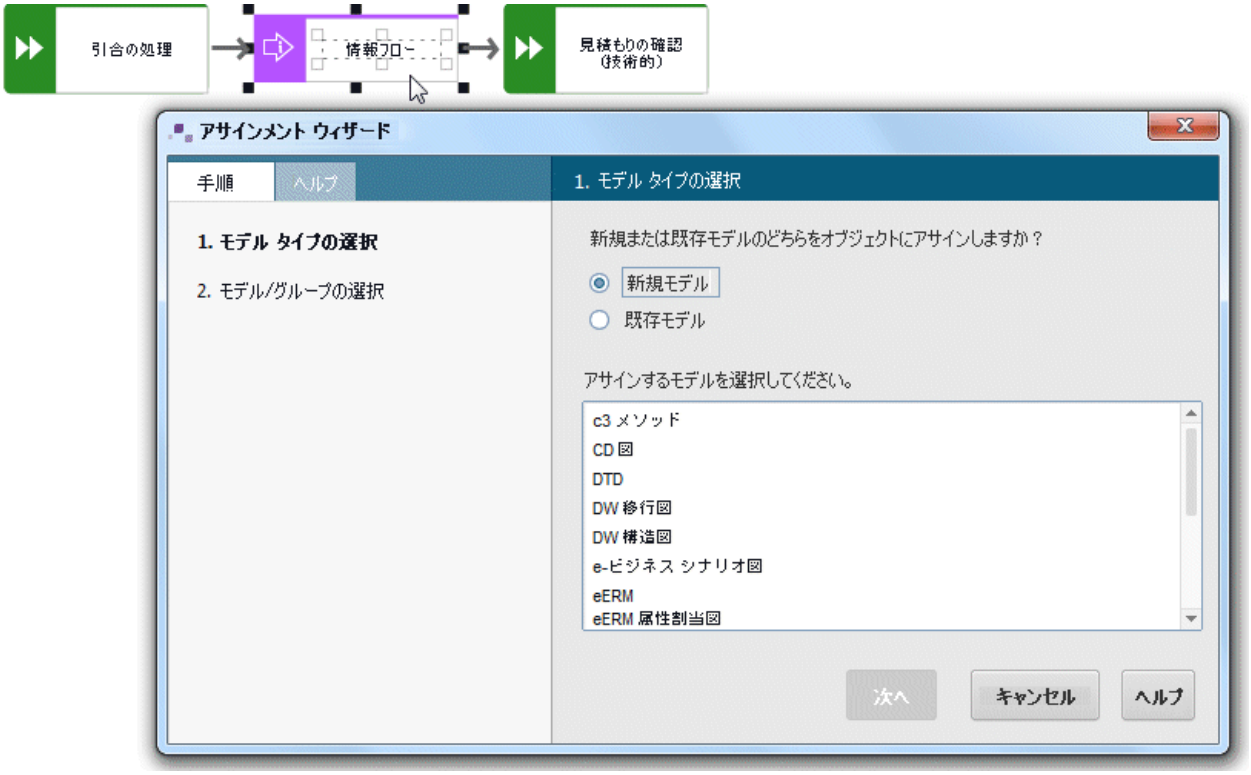


図 105: 情報フロー図と、[アサインメント ウィザード] を開いた画面

4.4.1.2.4 イベント図

イベントは、情報オブジェクトの状態が変更された事実を定義します。 つまり、すべてのイベントは、データ モデルの特定の情報オブジェクトを参照し、この情報オブジェクトのある時点におけるステータスを定義します。

まず最初に、イベントはトップダウン的な処理でおおまかに定義されます (たとえば、「注文が処理された」など)。 プロセス モデリングの次の詳細ステップで、特定の組み合わせによって大まかなレベルでイベントが発生する原因となるイベントを詳細に定義します。 たとえば、イベント「実行可能性の確認が完了した」、「注文ヘッダーの登録が完了した」、「注文項目の登録が完了した」のオカレンスがまとまって「受注処理が完了した」ステータスを定義できます。

イベント図を使用することで、おおまかなモデリング レベルおよび詳細なモデリング レベルで、これらのイベントの相関関係を表すことができます。 このために、大まかなレベルのイベントにもイベント図をアサインできます。このイベント図には、詳細レベルの

イベントと該当するルールを表すことができます（結果として階層構造が形成されます）。 さらに、このモデル タイプに、データモデルの情報オブジェクトを含めて、それらをイベントに結合できます。 このようにして、どのイベントが、どの情報オブジェクトの状態の変化を定義するかを表します。

次の図に例を示します。

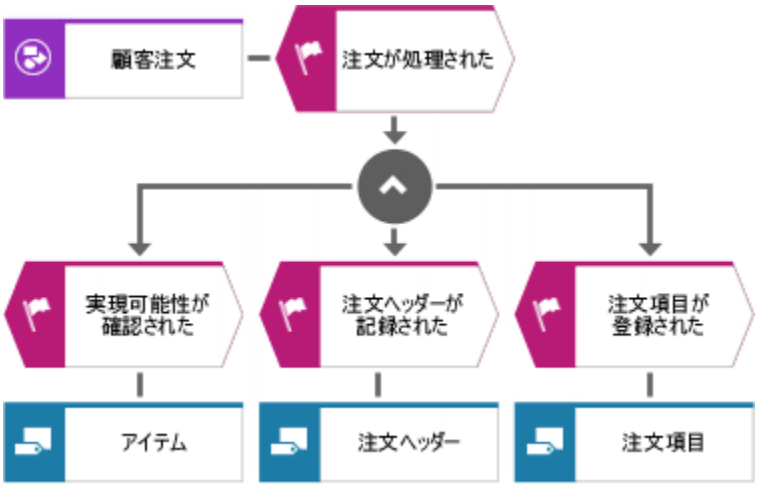


図 106: イベント図の例

4.4.1.3 ファンクション - 組織 - データ

4.4.1.3.1 EPC / PCD

EPC と PCD は同じ事実を表します。

これまでは、2 つのビューだけを扱ってきましたが、ここでは 3 つめのビューを紹介します。 プロセス連鎖の部分ビューを連結して全体ビューを構成して、ARIS 概念のすべてのコンポーネント間のインタラクションを考察することができます。 すると、プロセス連鎖 (当初の対象オブジェクト) がふたたび詳細に示されます。 ただし、ここでの考察は、考察するオブジェクトに関して個々のビューに表示される詳細にではなく、これらのオブジェクト間の接続線に焦点を当てます。

次の図は、すべてのビューを含むプロセス連鎖を表しています。 データ ビュー オブジェクトのイベントは、第 1 列に配置されます。 矢印は、一覧されたプロセス連鎖のファンクションを含むプロセス列に向かいます。 このように、第 1 および 第 2 列は、イベント制御を定義します。 第 3 列にはデータ オブジェクトが一覧され、各ファンクションに対する関係が示されます。 PCD の第 2 列および第 3 列のビューは、プロセス連鎖内のデータ フローを定義します。「プロセス連鎖図『17 ページ 』」の章で述べた PCD と異なり、要件定義の PCD には、プロセスの種類や IT システムを定義するための列はありません。 これらは、会社の実際の状況を記録するためには必要ですが、業務プロセスのテーマに関連した記述には含まれません。 プロセス連鎖の個々のファンクションの実行に責任を持つ組織ビューの組織ユニットは、第 4 列に定義します。

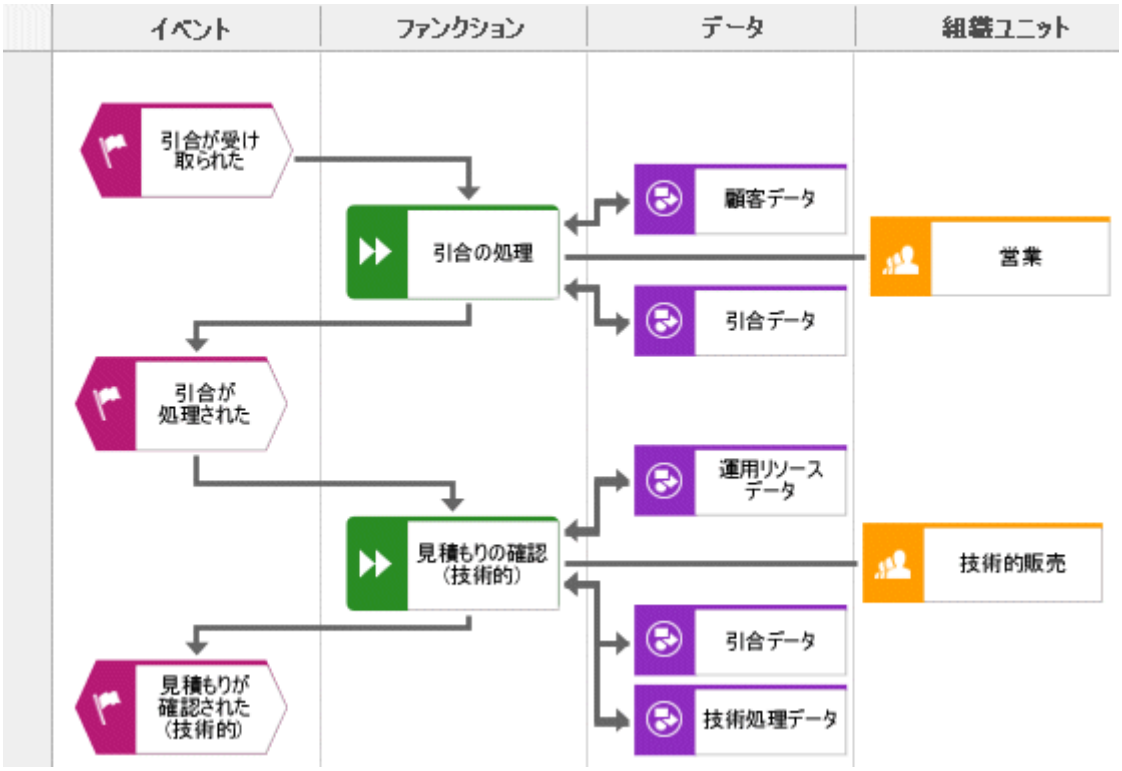


図 107: プロセス連鎖の例 (要件定義)

次の図のプロセス連鎖は、EPC 形式で表すこともできます。

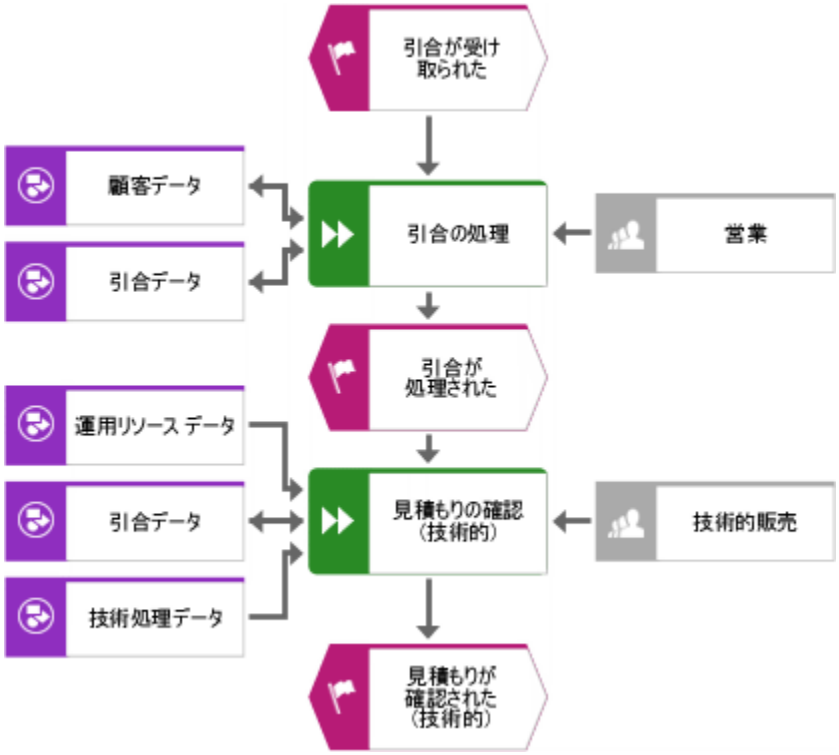


図 108: ファンクション、データ、組織ユニット、およびイベントを含む EPC

4.4.1.3.2 入出力図

入出力図では、入出力データと情報媒体の概要が提供されます。 このモデルの場合、各グリッド（線で区切られたフィールド）にシンボルが 1 つだけ配置されます。 一番上の行には、特定のファンクションによって作成されたデータまたは情報媒体（出力）が含まれます。 左の列には、特定のファンクションへの入力データまたは情報のシンボル（入力）が含まれます。 ファンクションに複数の入力シンボルまたは出力シンボルが必要である場合、それらはオカレンスのコピーとして作成されます。

入出力図内では、ファンクションおよび保存メディア/情報媒体のシンボルが作成されるとき、[入力を提供する] および [出力する] という非表示の関係が自動的に作成されます。

次の図は、入出力図の簡単な例です。

図 109: 入出力図

4.4.1.3.3 付加価値連鎖図

付加価値連鎖図は、主に企業の付加価値生成に直接関連するファンクションの識別に使用されます。 これらのファンクションは、一連のファンクションとして相互に連結され、付加価値連鎖を構成します。 次の図に付加価値連鎖の例を示します。

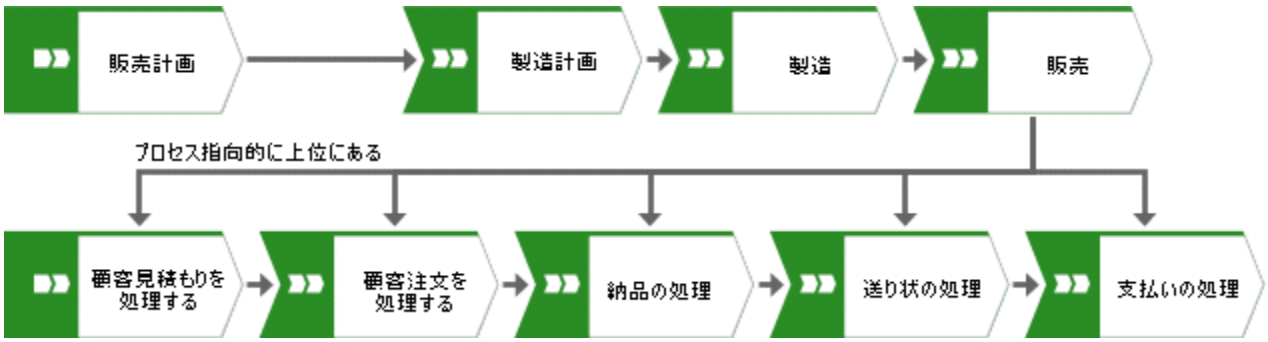


図 110: 付加価値連鎖の例

付加価値連鎖図では、ファンクション ツリーと同様に、ファンクションを階層的に配置することができます。 常にプロセス指向的な上位/下位を表します。

付加価値連鎖図によって、ファンクションの上下関係が表現できるだけでなく、組織ユニットおよび情報オブジェクトへのファンクションの関係を表現できます。 組織ユニットをファンクションに割り当てる場合は、プロセス連鎖と同様に、ファンクションの技術的責任、IT 関連の責任、および実際の実行を区別します。

付加価値連鎖図で使用可能な他の関係の一覧は、インストール媒体に含まれている『ARIS メソッド マニュアル - テーブル』マニュアル (ARIS Method tables.pdf) を参照してください。

4.4.1.3.4 ルール図

プロセス連鎖では、イベントとファンクションの相互関係を定義するための演算子としてルールを使用できます。 論理演算子を表すためのルール表現は、非常に複雑になることがよくあります。特に、ルールが互いに関係付けられている場合は複雑になります。 この種の表現によって、プロセス連鎖が複雑になり過ぎるのを防ぐために、EPC や PCD では一般的なルールを使います。 この一般的なルール演算子をルール図に接続して、複雑なルールの詳細を示すことができます（結果的に階層形式が形成されます)。

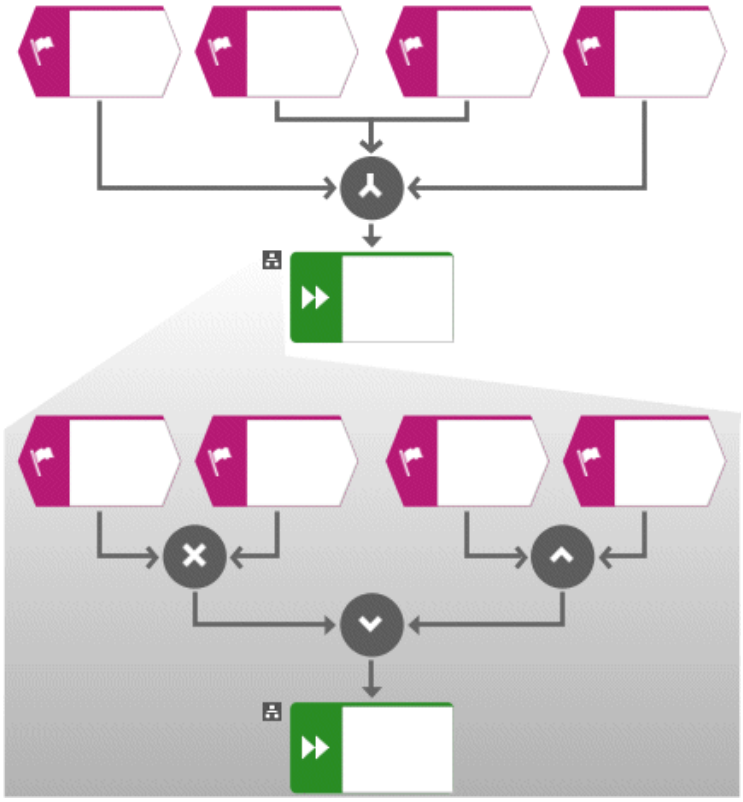


図 111: ルール図における複雑な演算子の表現

4.4.1.3.5 コミュニケーション図

大規模なリファレンス モデルには、さまざまなプロセス モデルが含まれます。 組織ビューのアイテムをこれらのプロセス モデルに含めることによって、プロセス中に、どのコンポーネントとどのコンポーネントがコミュニケーションを取るかが表わされます。 [コミュニケーション図] を利用すると、[]組織ユニット] 間のコミュニケーションに従って、すべてのプロセスをグループ化できます。

したがって、[コミュニケーション図] は、相互に連絡を取り合うすべての [組織ユニット] を表します。 たとえば「販売」組織ユニットは、「通信」タイプのオブジェクトを介して「顧客」組織ユニットに接続されます。「通信」タイプのオブジェクトは階層内に配置することができます。 これは、[プロセス選択マトリクス] モデル タイプに接続できます。 このプロセス選択マトリクスには、営業部門が顧客と連絡を取り合うすべてのプロセスが表されます。

4.4.1.3.6 分類図

分類図を使用すると、オブジェクト タイプ クラスを分類図に割り当てることでファンクションを分類できます。 分類は、さまざまな分類基準に従って定義できます。 分類基準を指定するには、[オブジェクト タイプ クラス] オブジェクト タイプと [分類基準] オブジェクト タイプを関係付けることができます。

4.4.1.4 オブジェクト指向モデリング

UML 環境でのモデリングには、UML を使用できます。 UML 図と UML 要素のメソッド関連のすべての情報には、UML インターフェースを使用して直接アクセスできます。

4.4.1.5 プロセスのバリアント

4.4.1.5.1 プロセス選択マトリクス

プロセス選択マトリクスは、メイン プロセスを個々のシナリオに割り当てることで、さまざまなプロセス シナリオを表します。

ユーザーは、シナリオ プロセスのどのファンクションが企業内で発生するかを決定できます。 このために、アプリケーション システムまたは企業のリファレンス モデルのすべてのメイン ファンクション (シナリオ ファンクション) が、プロセスとして含まれている必要があります。

プロセス選択マトリクスのモデリングでは、次のシンボル タイプを使用できます。

- シナリオ
- プロセス
- メイン プロセス

シナリオは、異なるメイン プロセスをグループにまとめた選択マトリクス内のシナリオ プロセスを表します。

プロセスは、シナリオ プロセスのファンクションを表します。シナリオ プロセスは、プロセス モデルによってリファレンス モデルにおいてより詳しく記述されます。

メイン プロセスは、そのプロセス (シナリオ プロセスのファンクション) が割り当てられたファンクション ツリー内のメイン ファンクションを表します。

次の図は、プロセス選択マトリクスの例を表しています。

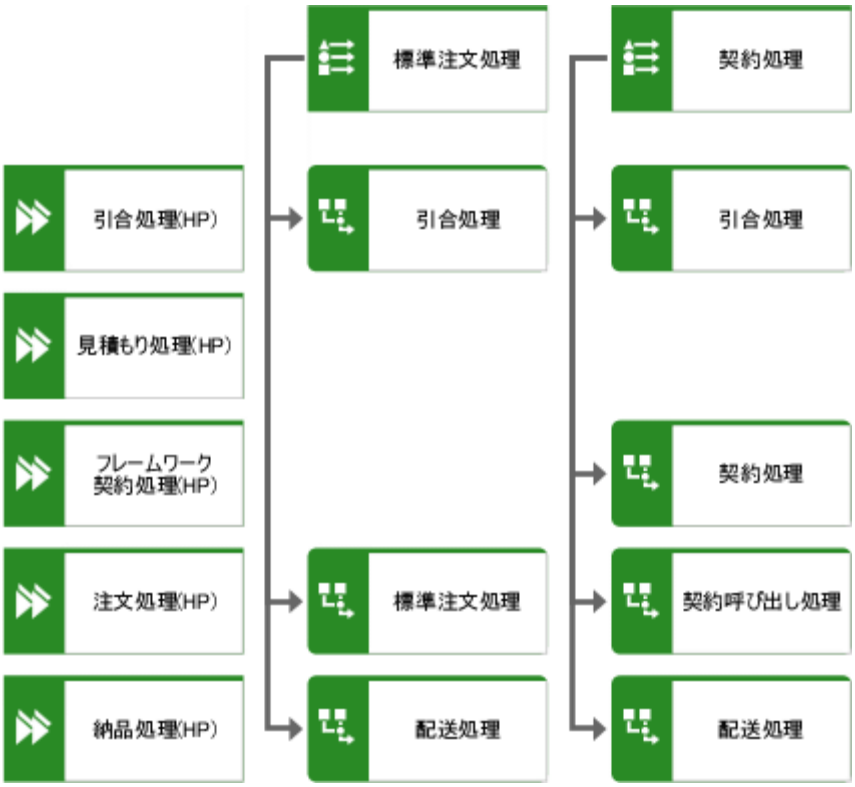


図 112: プロセス選択マトリクス (SAP AG R/3 リファレンス モデルからの抽出)

4.4.1.6 マテリアル フロー モデリング

プロセス モデル (EPC および PCD) を使用して、情報フローだけでなく、材料の変換も表すことができます。 業務プロセス内のマテリアル フローを表すために、ARIS では、[EPC] モデル タイプの拡張である [EPC (マテリアル フロー付き)] モデル タイプを利用できます。

4.4.1.6.1 EPC (マテリアル フロー付き)

EPC のオブジェクト タイプのほかに、[EPC (マテリアル フロー付き)] では、次のオブジェクト タイプも使用できます。

- 材料タイプ
- 包装材タイプ
- 運用リソース タイプ
- 運用リソース
- 機材タイプ
- 機材
- 倉庫設備タイプ
- 倉庫設備

- 運搬システム タイプ
- 運搬システム

[材料タイプ] オブジェクト タイプは、入力または出力接続線によって、[ファンクション] オブジェクト タイプに接続できます。 入力接続線の場合は、ファンクションの入力として要求される材料が定義されます。 このコンテキストでは、対応する接続線タイプを選択することによって、そのファンクションが材料の一部を使うか、全部を使うか、またはまったく使わないかを定義できます。 出力接続線は、そのファンクションによって作成される材料タイプを定義します。

技術的リソースは、材料の変化に必要になります。 プロセス連鎖では、技術的リソースを [ファンクション] オブジェクト タイプに連結することもできます。 利用可能な代替リソースを定義する場合、[必要とする] および [どちらかが必要となる] の 2 つの接続線タイプが利用できます。

ファンクションの実行中に材料が包装される場合は、包装材タイプが必要です。 対応する包装材タイプを指定するために、ファンクションとそれに必要な包装材タイプの関係をモデル化できます。

次の図は、EPC (マテリアル フロー付き) および対応する技術的リソース タイプと包装材タイプを表しています。

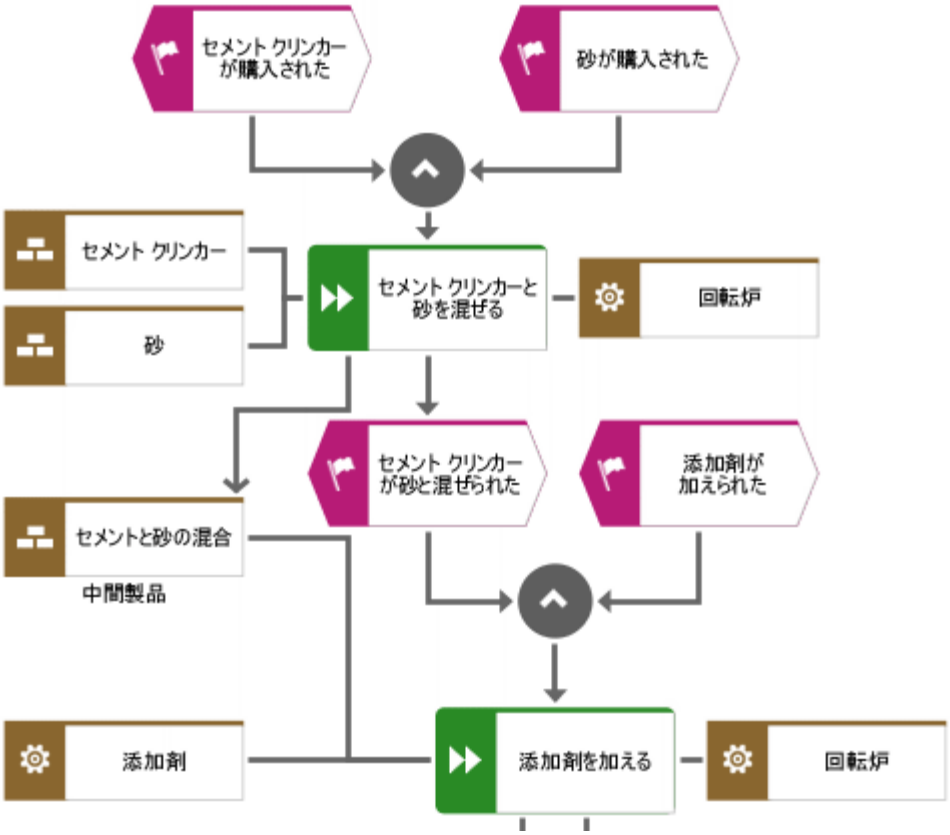


図 113: EPC (マテリアル フロー付き) からの抽出

4.4.1.6.2 マテリアル フロー図

マテリアル フロー図を使用して、ファンクション間のマテリアル フローを表すことができます。 モデリングの際、これらは情報フロー図と同様に扱われます。 マテリアル フロー図では、マテリアル フロー接続線によって 2 つのファンクションを接続します。 この接続は、材料がソース ファンクションからターゲット ファンクションに流れることを意味します。 表示されているファンクションの入力または出力である材料をさらに詳しく定義する場合は、材料図を接続線に割り当てることができます。その結果、マテリアル フロー接続線に階層が形成されます。 材料図では、ファンクション間で交換される材料または材料タイプが示されます。

4.4.1.6.3 EPC (列表示) / EPC (行表示)

次の説明は、EPC (行表示) についても同様です。

EPC に関する説明の大部分は、[EPC (列表示)] のモデル タイプにも当てはまりますが、すべてのシンボルがさまざまな列に分けて表示されるという点が EPC と異なります。 このような表現によって EPC がはるかに理解しやすくなるという利点があります。 図のヘッダーには組織要素およびアプリケーション システムが配置され、 その他のシンボルはすべて各列の 2 番目の行に置かれます。

このようなレーン モデル、すなわち、列や行の中に配置されたモデルの大きな特徴は、非表示の関係が自動的に作成されることにあります。 たとえば、アプリケーション システムとファンクションをモデリングする場合、[EPC (列表示)] のデフォルトの列に [サポートする] という非表示の関係が自動的に設定されます。 組織要素とファンクションは [実行する] という非表示の関係で接続されます。 このほか、次のような列を追加することもできます。これらの列は、非表示の関係に従って名前が付いています。

- 寄与する
- 決定する
- IT 面の責任を持つ
- 技術的責任を持つ
- キャンセル時に通知される必要がある
- 結果を通知する必要がある
- 結果を通知される必要がある
- 承認する
- コンサルティングする

次の図に、EPC (列表示) の例を示します。

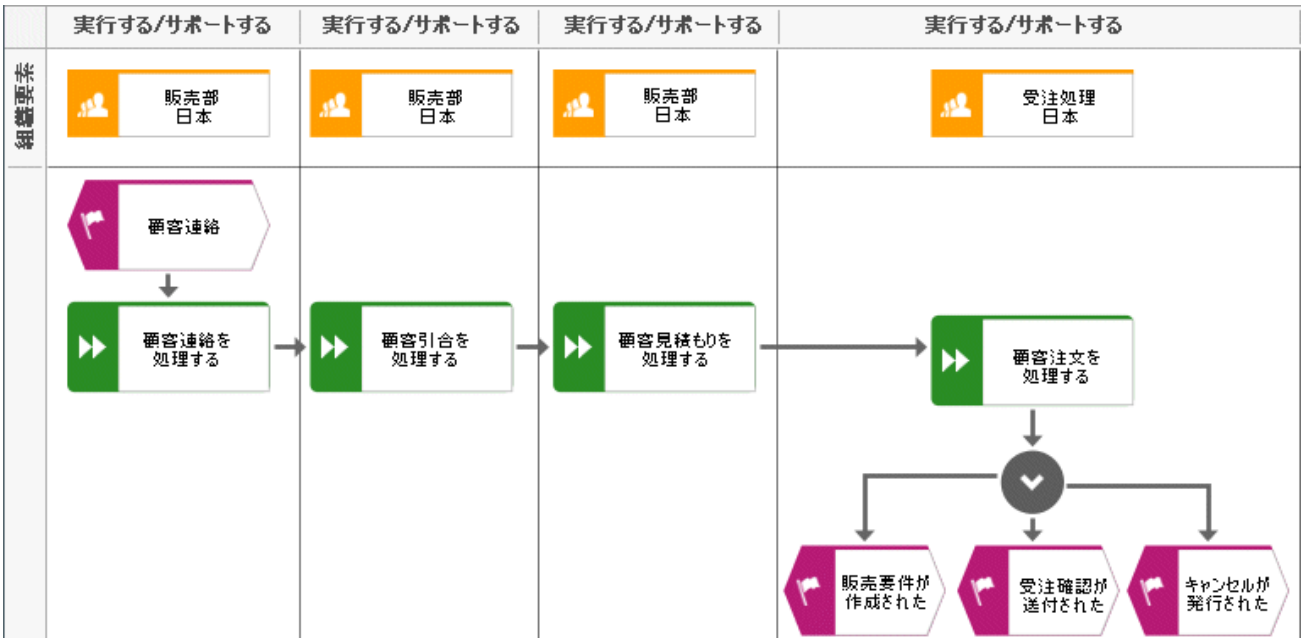


図 114: EPC (列表示)

EPC (列表示) と EPC (行表示) の違いはモデリングの方向です。 EPC (列表示) では上から下にモデリングが行われ、EPC (行表示) では左から右にモデリングが行われます。

4.4.1.7 SAP ALE モデル

SAP ALE (SAP Application Linking Enabling) タイプのモデルは、ファンクションとアプリケーションとの関係のモデリングに使用されます。

4.4.1.7.1 SAP ALE フィルター モデル

SAP ALE フィルター モデルは、1 つのシステム (アプリケーション) の中でファンクション タイプを 2 回以上使用できる基準を表します。 この場合、[フィルター オブジェクト タイプ] オブジェクトは、分布モデルにおける分割基準として機能します。

4.4.1.7.2 SAP ALE メッセージ フロー モデル

SAP ALE メッセージ フロー モデルでは、[メッセージ フロー] オブジェクト タイプを使って、システムとファンクション タイプ間のメッセージ フローを方向を含めて表します。

4.4.1.7.3 SAP ALE メッセージ タイプ モデル

SAP ALE メッセージ タイプ モデルでは、[メッセージ タイプ] オブジェクト タイプを使って、関与するデータおよびトランザクションに従って、分散システム間で交換されるメッセージを分類します。

4.4.1.8 ロール割当図 (RAD)

SAP リファレンス モデルは EPC を使用して表されます。 これらの EPC は、業務プロセスを異なる詳細レベルで表示します。 最も詳細なレベルの EPC (SAP 用語では「プロセス」) では、SAP システムにおける処理プロセスの手続きがモデリングされます。 これらのプロセスにロールとトランザクションの両方を割り当てることができます。

ARIS では、このような状況を EPC のファンクション割当図で表します。ファンクション割当図では、モデリングされたプロセスを含む EPC を該当するファンクション定義にアサインする必要があります。 これにより、トランザクションの実行に必要なロールを [ファンクション割当図] で表現できます。 ただし、ロールとトランザクションには直接的な関係がないため、複数のロールがある場合、どのトランザクションに責任があるかは自動判定できません。 このため、ロール割当図 (RAD) で、ロールをトランザクションに割り当てます。 列ごとに 1 つのロールが表示されます。 トランザクションは、非表示の関係が作成された列に表示されます。

この情報を R/3 の導入時に使用して、SAP システムの運用に使用するユーザーのプロファイルと認証概念を作成できます。

図 115: ロール割当図 (RAD)

4.4.1.9 その他のモデル

4.4.1.9.1 ビジネス コントロール図

ビジネス コントロール図は、プロセスまたはファンクションの潜在的なリスクとリスクのコントロール方法を表示します。

リスクは、設定されたプロセス目標を達成できない潜在的危険性を表します。

リスク コントロールは、リスクを除去または最小化するための一般的な方法です。

リスク ソリューションは、リスクに対してリスク コントロールを導入することを意味します。

ビジネス コントロール図のレイアウトは、マトリクスまたはテーブルに対応しています。 すなわち、横座標には潜在的なプロセスのリスクが配置され、縦座標には使用可能なリスク コントロールの方法が配置されています。 この上に、リスクとリスク コントロールとの間の演算子としてリスク ソリューションが挿入されます。 さらに、リスクに対するリスク コントロールの導入をサポートするものとして、(ユーザー要件の意味での) 組織ユニットおよび文書をモデルに追加することもできます。

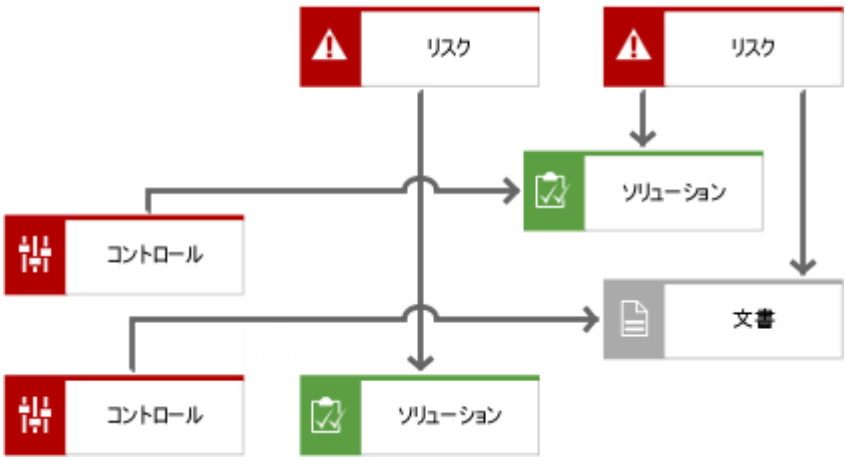


図 116: ビジネス コントロール図の例

このモデル タイプは、一般に SAP 標準プロセスを記述するのに使用され、 考察対象のプロセスに対する SAP ソリューションのリスクおよびリスク コントロールの方法を表します。

4.4.1.9.2 DW 変換

データ ウェアハウス データ変換図は、データ ウェアハウスの記述に使用します。 ARIS アーキテクチャのプロセス ビューに記述される動的な側面に注目します。

モデルでは、一般的な情報オブジェクト データを InfoCube のデータ フォーマットに変換する方法を示します。 データが失われないよう変換を効率的に処理するには、モデリングにメソッド仕様が必要です。 この方法には変換のルールとプロシージャが含まれており、これらはモデルにグラフィック表示されます。

変換は次の 2 つの手順で実現されます。 最初に、「転送構造」アイテムが「通信構造」アイテムに変換されます。 次に、InfoCube に転送されます。

この手順は異なるレベルで実行できます。 高次（要件定義）の抽象レベルでは、InfoCube への変換に使用する変換構造と通信構造を表示できます。 一方、低次（実装）レベルでは、個々のデータ要素が互いにマッピングされる方法を表示することができます。

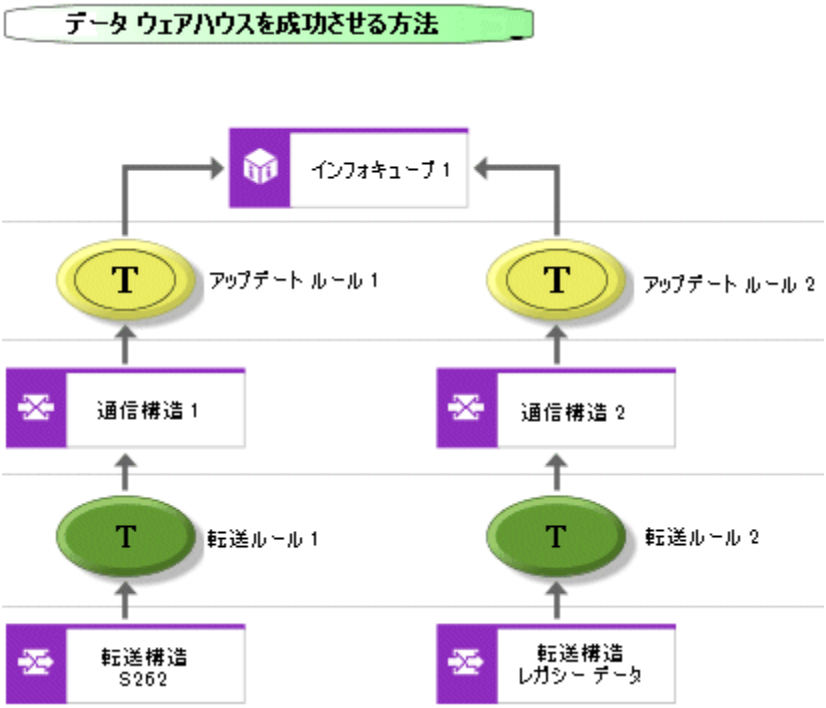


図 117: DW 変換 : データ ウェアハウスのデータ変換

4.4.1.9.3 e-ビジネス シナリオ図

企業内の業務プロセスを順調に遂行する重要性は、着実に高まっています。 一方では企業間の接点における、他方では企業とその顧客の間の接点における特定手順の業務の流れが関心の的になっています。 企業と顧客の接点に必要であるものは、透明性、迅速性、一貫性、そして方法の直接性です。

また、企業の観点からは適切なビジネス パートナーを速やかに見つけること、そして顧客の観点からは販売業者を見つけることがますます重要になってきています。 これらのプロセスを最大限に最適化することが、競争上の優位につながります。 これらの相互関係をサポートする理想的なプラットフォームがインターネットです。 上記の環境におけるプロセスは大変複雑なため、「e-ビジネス」という用語を定義する必要があります。

「e-ビジネス」とは、コンピューターによって支援される 2 つのビジネス主体間のあらゆる処理、および新しい媒体を介した付加価値獲得のための試みを意味します。

このため、e-ビジネスでは、インターネット経由での商品の購入という単純な行為から、2 つの企業が関わる複雑なプロジェクト、または企業の営業活動のための Web ページ作成まで、さまざまな形態が可能です。

企業間の関係は B2B (Business-To-Business)、企業と顧客の間の関係は B2C (Business-To-Consumer) と呼ばれます。

[e-ビジネス シナリオ図] は、e-ビジネスをサポートするために開発されました。

付加価値連鎖の全体を表示できること、つまりエンド ユーザーからプロセスに関与する各企業までを表示できることにより、最適化の可能性を引き出す基礎が提供されます。 目的としては、たとえば、サプライ チェーンの改善、調達コストや配布コストの低減、または情報システムのアーキテクチャの最適化などがあります。 その目的によって表される内容は、この手法でモデリングできます。

経済主体は図の右上に配置され、「取引先」と呼ばれます。 取引先は組織図を使用して割り当てることができます。 ここでは、プロセス全体の一部として経済主体が実行する個々のプロセス、およびサブプロセス間の接点が関心の対象となります。 個別プロセスは企業間の連係に緊密に関わるビジネス プロセスで、プロセス モデルに割り当てることができます。 ビジネス プロセスは、R/3 システムなどのアプリケーション システム (ビジネス コンポーネント) でサポートされます。

プロセスに関係する従業員の役割も定義できます。 これらは、モデルでは [従業員の役割] と呼ばれます。

接点の主な機能はプロセス特有の情報の転送です。 この情報は XML または HTML 形式でビジネス ドキュメントの形にまとめられます。 また、ビジネス ドキュメントはデータ モデルとしても割り当て可能です。 このオブジェクトのかわりに、「金銭取引」オブジェクト (金銭の流れを表す)、「商品発送」オブジェクト (商品の流れを表す)、「電子メール」、「インターネット」、「イントラネット」、「エクストラネット」、および「携帯電話」(データ転送の技術的側面を指定する) の各オブジェクトも使用できます。

企業に関連するすべての業務手順が、取引先の下の行の同じ列にモデリングされます。

このようにして、列の枠線が抽象的な接点を形成します。 これらには、最適化の主たる対象が含まれているため特別な注意を払うこととなり、モデルに組み込むことは常に有益です。

用語解説: 次のサンプル モデルにおいて、OEM は「相手先商標製造会社 (Original Equipment Manufacturer)」を表し、MRP は「資材所要量計画 (Material Resource Planning Controller)」を表します。

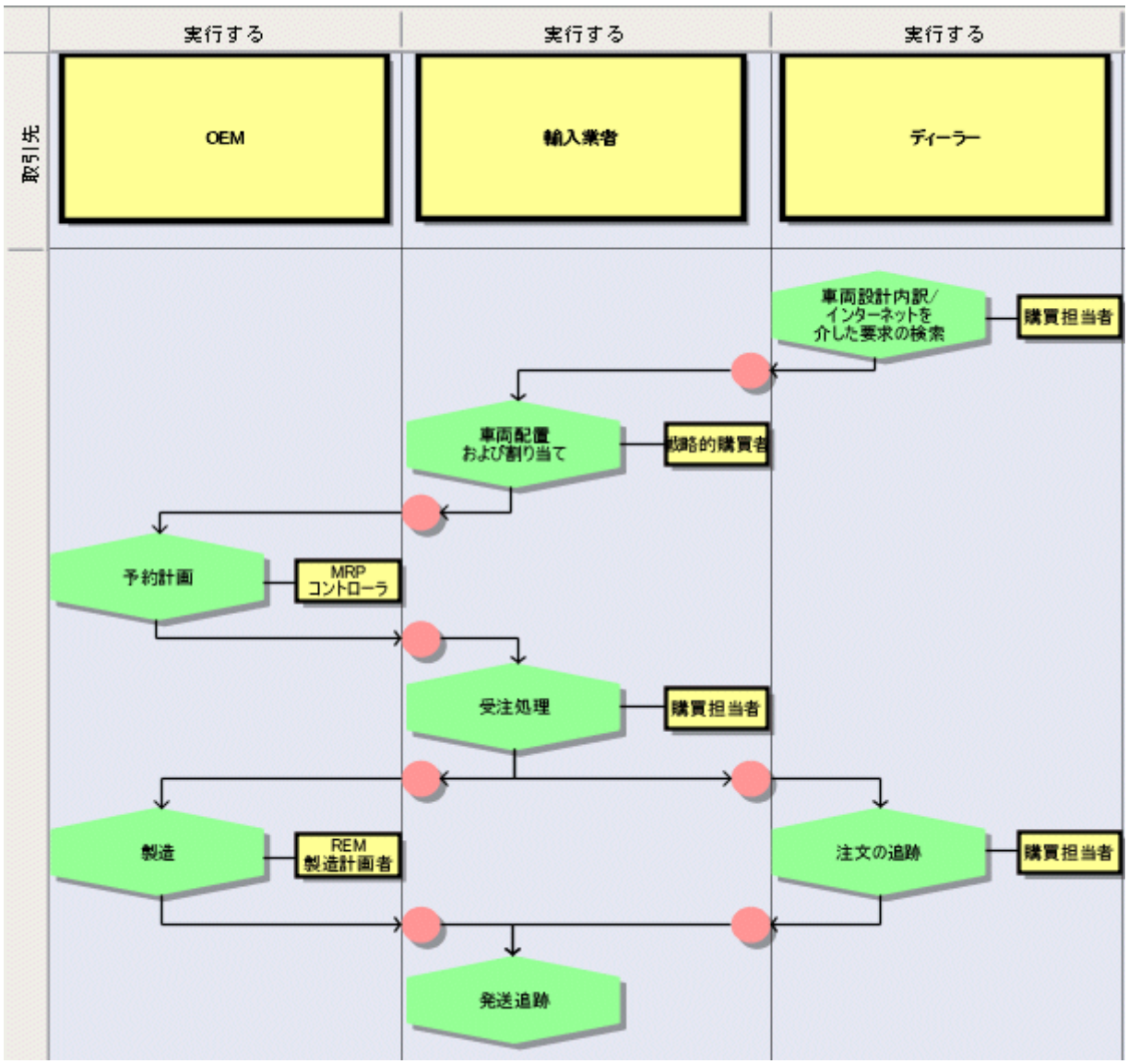


図 118: 自動車業界の e-ビジネス シナリオ図の例

サンプル モデルは、製造業者、輸入業者、ディーラーの協力関係を表しています。 各業者は、全体構造の中で特定のプロセスを担当しています。この構造では、ビジネス ドキュメントを使用して別の事業提携者のプロセスへの接点を介して情報が交換されています。 業務プロセスに関連するスタッフは、各自記録され、役割が割り当てられます。

4.4.1.9.4 構造モデル

構造モデルは、一般に、事実の階層または体系化（事実の特化または汎化）を表すために使用されます。

構成要素は、(意図された体系化の方向における) 1 つの事実を表します。

事実階層の個々の構成要素には、事実に関するモデルをアサインすることができます。

構造モデルは、品質管理、特に認証の目的でもっとも頻繁に使用されます。 この場合、構造モデルでは、規格を個々の構成要素に分割します。個々の構成要素には、品質基準を満たすために役立つモデルがアサインされます。

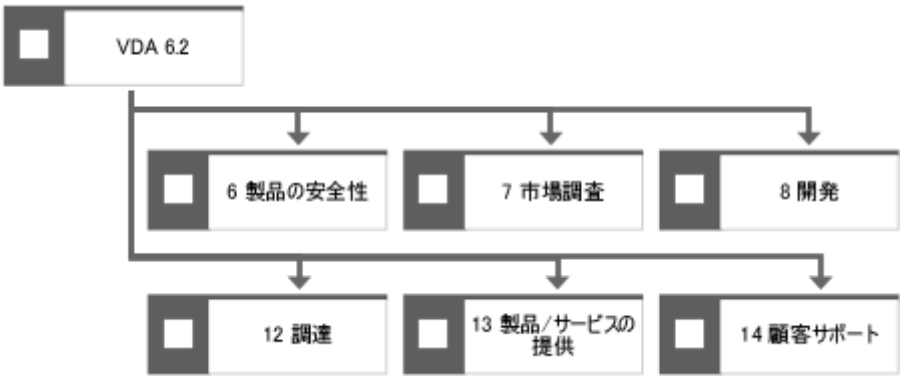


図 119: 構造モデルの例（VDA 6.2 標準からの抽出)

レポートを使用すると、これらの事実を簡単に評価したり、文書化の目的で使用したりできます。

4.4.1.9.5 製造プロセスとオフィス プロセス

[製造プロセス] と [オフィス プロセス] のモデル タイプは、本質的には [EPC] モデル タイプまたは [EPC (マテリアル フロー付き)] と同じ事実を表します。限られたオブジェクトしか使用できない点と、シンボルがグラフ形式で表示される点が異なります。

グラフィック表示には、事業部門の従業員がトレーニングなしでモデルを理解し、自分でモデルを変更したり作成したりできるという利点があります。 たとえば、3 人の人間をかたどったシンボルがグループを表すことは、容易に理解できます。それに対して、EPC のシンボル（2 重の楕円）は抽象的であり、わかりやすくはありません。 すなわち、この 2 つのモデル タイプは、事業部門に対し、プロセスのモデリング、プロセスの最適化、およびプロセスの利用方法を確立することを目指したものです。--

シンボルを見分けやすくするために、生産プロセス（有形の商品や製品の製造）を表す [製造プロセス] と、オフィス プロセス（無形の商品/サービスの製造）を表す [オフィス プロセス] という 2 つのプロセス タイプ（モデル タイプ）を使用できます。

前の図の 3 つのモデル タイプ間では、あるモデル タイプの内容を別のモデル タイプにコピーすることによって、表現タイプを変換することができます (相手のモデル タイプに関連するオブジェクトが存在する場合)。 コピーするときは、ARIS によって自動的にシンボルが変換されます。 次の図は、[EPC]、[製造プロセス]、[オフィス プロセス] の各モデル タイプを使って同じ事実を表現した例です。

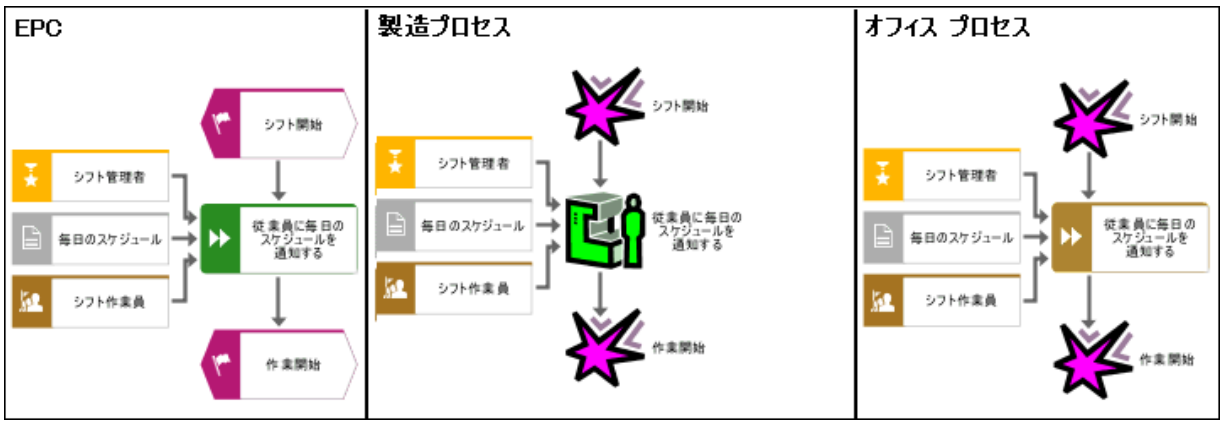


図 120: EPC、製造プロセス、オフィス プロセスの各モデル タイプでの事実の例

4.4.1.9.6 プロジェクト プロセス連鎖 (PPC)

PPC モデル タイプは、ARIS と MS Project のリンクを形成します。 ARIS Architect では、イベント駆動プロセス連鎖 (EPC) を使って、業務プロセスにおけるファンクションの操作順序を表現することができます。 しかしながら、プロジェクトのキャパシティおよび時間を計画するには、このような抽象的なレベルでは不十分です。 かわりに、実際のイベントのインスタンスおよびタスクを考察し、指定する必要があります。 PPC モデルでは、このような要件を満たすため、[イベント] および [ファンクション] オブジェクト タイプのかわりに、オカレンス レベルに独自のオブジェクト タイプが用意されています。

これまでは、PPC モデルは以前のバージョンの ARIS MS プロジェクト インターフェイスを使用して EPC から自動的に生成できました。 ARIS Architect と MS Project 2000 をつなぐ現在のインターフェイスは、PPC を参照しなくなりました。 現在では、このモデル タイプは副次的な役割を果たし、主として以前のモデルを表示する目的にのみ使用されます。

イベント インスタンスは、具体的なプロセス インスタンス中に生じるイベントです。 [イベント インスタンス] は、評価、すなわち真か偽かの判定が可能です。

タスクは、特定のプロセス インスタンスで発生するファンクションです。 ファンクション インスタンスには、一意の開始/終了日時およびその他の必要な属性値を設定できます。

プロジェクトの構造アイテム (タスク、イベント インスタンス、ルール、接続線) は、プロジェクトの時系列的な流れを表すために使用します。 PPC には、[内部要員/外部要員]、[運用リソース]、[一般リソース] リソース オブジェクトも含まれます。 これらは、時間とキャパシティの計画を対象としています。

[一般リソース] とは、詳細な説明がなく、要員または運用リソースに限定されないリソースです。 [一般リソース] を使用して手順を実行できます。

また、[クラスター インスタンス] を使用すると、PPC でより詳細にタスクを指定できます。

クラスタ インスタンスは、[クラスタ/データ モデル] オブジェクトのインスタンスで、 複数のデータ オブジェクトまたはデータ構造の論理ビューを表します。

[PPC] は [クラスター インスタンス] を使用してタスクとデータとの関係を表します。 [クラスタ インスタンス] オブジェクト タイプは [情報媒体図] タイプのモデルにアサインできます (データ ビューの要件定義を参照)。 その結果、データがどの情報媒体に格納されるかを示すことができます。

次の図に、EPC の変換で作成されたプロジェクト プロセス連鎖の例を示します。

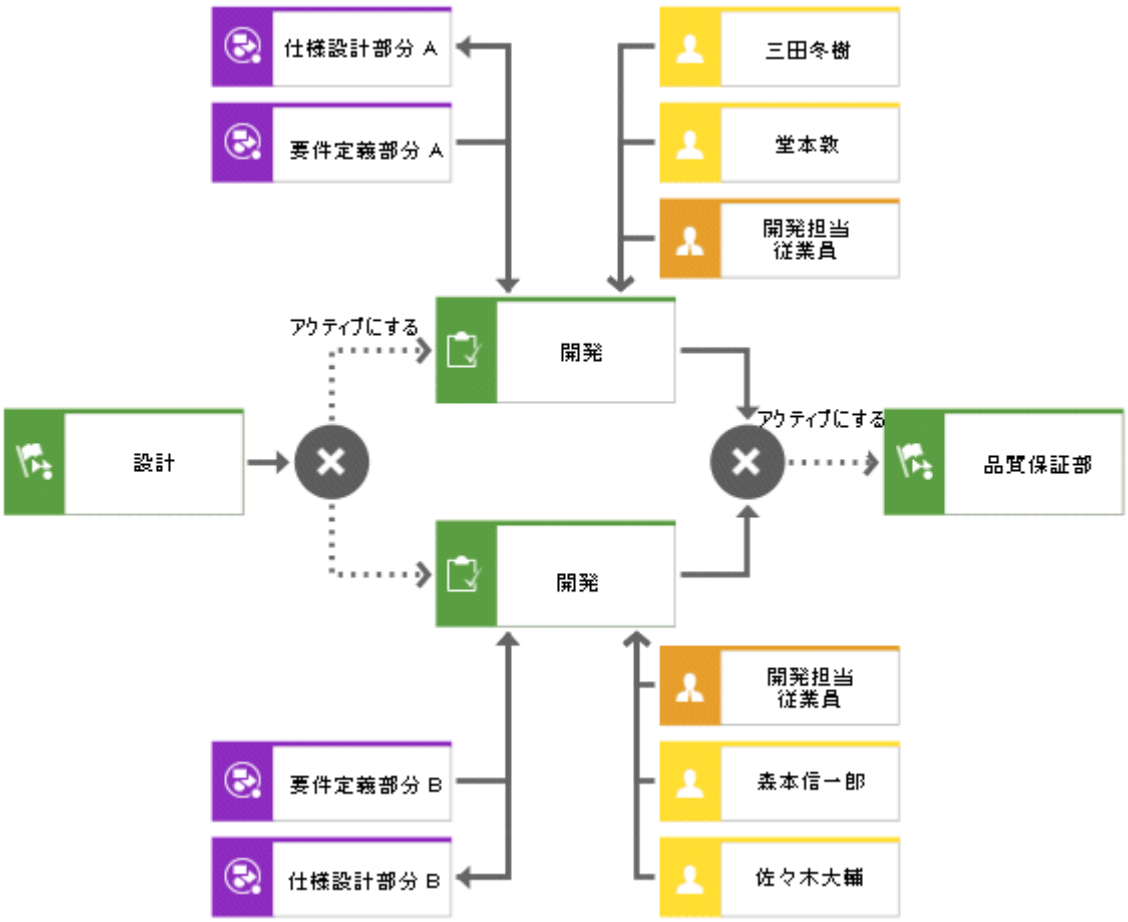


図 121: EPC から作成された PPC の例

例の XOR ルールは、変換された EPC 内のこの位置で分岐が発生したことを示します。 この分岐はプロジェクトの代替経路として解釈され、一意に指定する必要があります。

[PPC] モデル タイプは、ARIS Architect で直接モデル化することもできます。

4.4.1.9.7 プロセス インスタンス生成モデル

シミュレーションの主な側面は、動的な進行のあいだにプロセスの操作順序を分析することです。 開始イベントで、分析されるプロセスのインスタンスが生成 (開始または作成) されます。 それぞれの適用領域に従って、ユーザーは、プロセスのインスタンスを生成する時期と頻度を決定できます。 さらに、たとえばプロセスの優先度を指定して、緊急を要するプロセスについて考慮することができます。

ARIS メソッドでは、開始イベントに対して [優先度] 属性 ([シミュレーション] 属性タイプ グループ) を指定して、タスクに優先順位を付けます。対応する開始イベントでインスタンス生成されるすべてのプロセスは、この優先順位に従います。

記述した要件は、プロセス インスタンス生成モデルによって満たされます。 このモデルは、複数レベルのオブジェクト モデルとして作成されます。 [インスタンス生成インターバル] オブジェクトは最下層レベルになります。 このようなインターバルには、[インターバル開始 (相対)]、[インターバルの長さ]、[プロセス インスタンス数]、[分布]、[サイクルで繰り返す]、および [周期] 属性が設定されます。 [インターバルの長さ] に「0 (ゼロ)」を指定した場合、特定の時点を表します。 インターバルは比較的短い時間を表し、プロセス インスタンス生成サイクルは複数のインターバルによって構成されるシーケンスを繰り返すために使用します。 たとえば、4 つの異なるインターバルで 1 日をモデリングできます。これらの 4 つのインターバルは、シミュレーション期間全体 (1 週間など) においてサイクルとして繰り返されます。 ただし、シミュレーション期間を複数のサイクル (就業日と週末など) に分割することもできます。そして、これらのサイクルそれぞれに、異なるインターバルを指定できます。 プロセス インスタンス生成計画には、1 つまたは複数のサイクルを指定できます。 次の例では、このオブジェクト モデルについてより明確に説明します。

プロセス モデルが開始イベントの EPC として存在します。 次の仮定をこのプロセスに適用します。平日 (月曜日から金曜日) は、50 個のプロセスが就業日の就業開始時間、午前 8 時に開始されます。 午前 8 時から昼の 12 時まで、および、午後 1 時から午後 5 時までは、20 個のプロセスが同じ間隔で開始されます。昼の 12 時から午後 1 時までと就業時間外には、プロセスは開始されません。 土曜日は、60 個のプロセスが午前 9 時から午後 3 時までの間、三角分布で開始されます。 通常、日曜日にはプロセスは開始されません。 この週単位の周期は、7 月から 8 月までの休暇を除いて、1 月から 12 月まで適用されます。 休暇期間中は、土曜日に就業する人はいません。

前述の仮定に基づくと、次のモデルを生成できます。

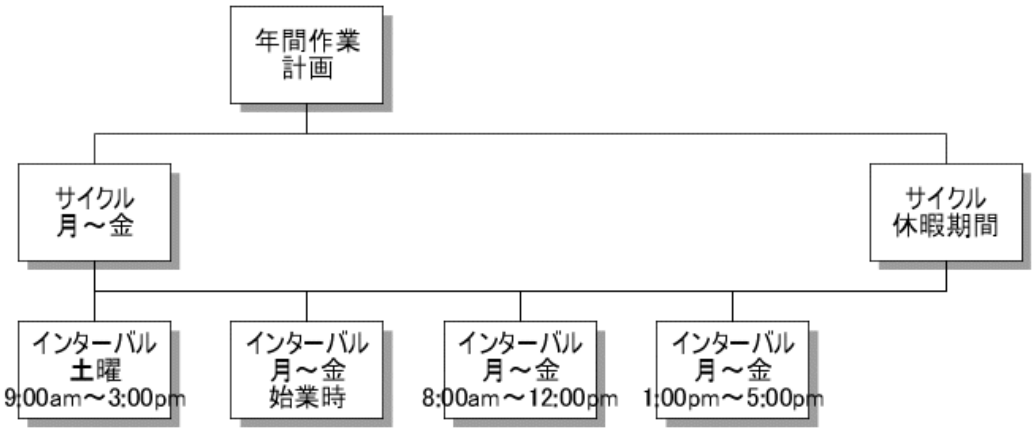


図 122: プロセス インスタンス生成モデル

4.4.1.9.8 管理システム要件分析モデル (RAMS)

RAMS (管理システム要件分析モデル) は、Digital Equipment によって開発された企業分析手法です。

RAMS は、IT 実装の可能性を表示および評価したり、情報システム要件のソリューション シナリオを作成したりするためのモデル化されたプロシージャです。 その結果として要件仕様を作成できます。この仕様に基づいて、業務目標、業務手順、情報フローおよび情報システムの調整を実行します。

このモデルでは、考慮されるすべての部門、アクティビティ、既存のアプリケーションが斜めの方向に表現されます。 この斜めの記述は、個別のファンクション ユニット間の最も重要な情報フローのそばにマトリクス形式で追加されます。 必要に応じて、重要な商品、金銭、材料のフローを追加できます。

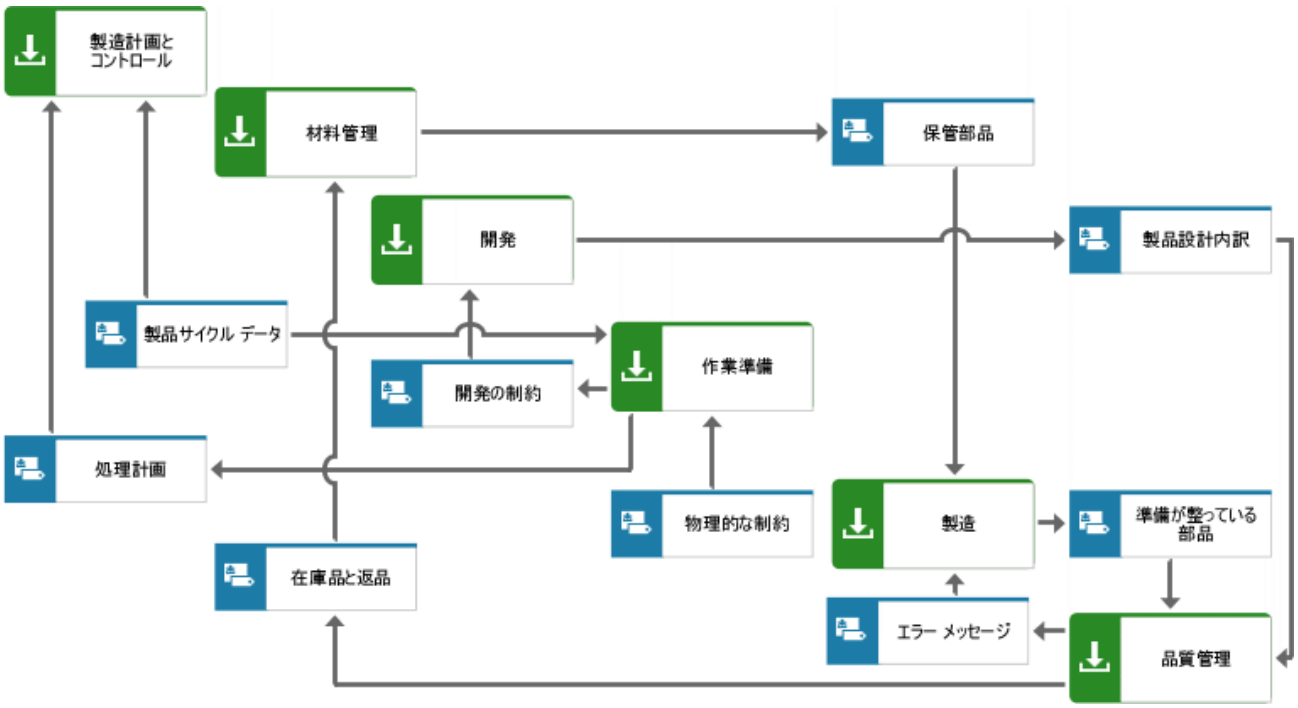


図 123: RAMS 図の例

RAMS 調査のプロシージャ モデルは、次のとおりです。

1. ステップ 1: 最初のステップでは、期待される調査結果、必要な参加者の指名、および、初期のタイムスケジュールについての予想を定義します。
2. ステップ 2: 2 番目のフェーズでは、まず、調査対象となる個別の部門、アクティビティ、既存のアプリケーションを選択します。 これらは、斜めのマトリクスで表され、個別の機能ユニット間の重要な情報フローを識別します。 必要に応じて、重要な商品、金銭、材料のフローを追加して、表示できます。 さらに、詳細な要件分析を実行する部門やファンクションを指定します。
3. ステップ 3: 調査範囲を定義すると、選択した部門やファンクション グループが、業務目標や用務処理に関してだけでなく、タスクや関連した情報要件に関しても詳細に分析されます。 図を上手に使用することにより、複雑なプロセスや手順が理解しやすくなります。 オリジナルの業務書類や報告書、画面などを使用すると、側面を明確にすることができます。 この分析の重要なタスクは、情報フローおよび既存のアクティビティとシステムとの相互関係における業務手順に発生する可能性のある不規則性を発見することです。 既存のアクティビティとシステムの改善の可能性についても調査できます。 分析を実行中に、問題や疑問、提案に対して、使用可能な最良の解決法が示されます。 その後、収集した情報が構造化され、業務アクティビティに関する因果関係が詳細に調査されます。 改善の可能性が明らかになると、その情報が記録され、その有用性が評価されます。
4. ステップ 4: 実際の詳細な分析の結果は、今後の要件仕様の基礎として使用されます。 問題のある領域が明白になり、それに対する新しいアイデアや代替の解決法が開発されます。 推奨された解決法 (複雑なシステムから、プロセスに対する簡単な変更までさまざまです) を、以前に記録した初期状況に常に密接に関連させることが重要です。 関連

する領域におけるすべてのファンクションとアクティビティについて、初期状況との比較を実行する必要があります。 調査結果により、一般的なソリューションが実用的であることが確認されたり、またはユーザー要件が作成および同期されたりします。

5. ステップ 5: 分析の各ステップで収集された結果に基づいて、最終的な要件仕様が作成されます。 審査の過程で収集されたすべての情報、詳細な審査項目、および提案は、最終報告としてまとめられ、今後のシステム要件の基礎となります。 次のソリューションの導入ステップでは、機能的なシステム仕様を作成します。

4.4.1.9.9 ロール図

一般的に、ロール図はプロセスをより正確に記述するために使用します。 プロセスに参加している [組織ユニット] およびその [ロール] が焦点となります。 オブジェクトとその関係には次のプロパティがあります:

ロールは、認証について考慮するプロセスに関係します。 プロセスにおける認証タイプの指定 (ロールは「実行に関与」) は実行可能性と同様に重要です。 特定の認証によるプロセスの実行では、「ロール - 参加」、「参加 - プロセス」という関係の連鎖 (「参加 - 認証条件」、「参加 - 認証値」の両方を含む) が確立されます。

ロールには、要員、役職、または情報システムを割り当てることができます。 ロールは互いに関係するプロセスとリソース間の関連を形成します。 これは、プロセスにかかわるリソースに必要な要件を総合して定義されます。

プロセスの実行には、参加しているロールまたは割り当てられたリソースに関連した、プロセスを処理するためのスキルが必要です。 プロセス指向でロールを定義できるようにするには、プロセスを評価して、関連する要員やシステムについてのプロセス要件を指定する必要があります。 さらに正確には、要員またはシステムの要件は、これらの要員またはシステムが持つ知識や能力 (スキル) を形成します。 スキルの評価は、割り当てられた評価の割合によって標準化されます。

したがって、ロール図を使用して、プロセスと特定の基本システムを表すことができます。また、関連するリソースも表示され、それらのリソースが持つスキルや要求されるスキルが記録され、認証も表示されます。

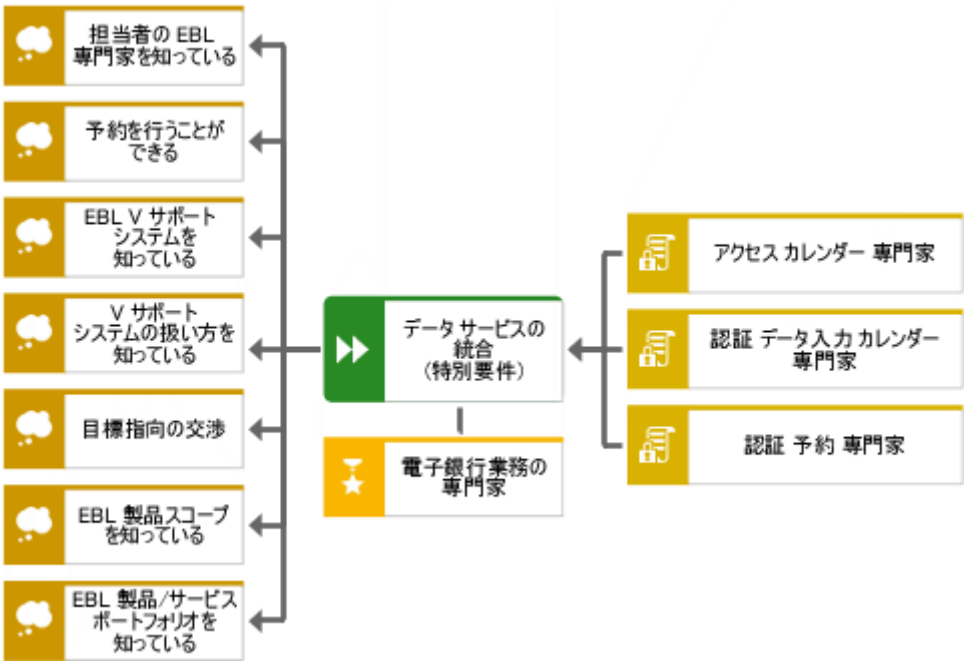


図 124: ロール図

サンプル モデルでは、ロール (能力と認証) に対する基本プロセスの要件、および能力と認証に関するリソースの基本ロールの要件が表示されます。

図は対応する基本プロセスと基本ロールにアサインされます。 図を基本プロセスにアサインすることにより、基礎となるプロセス [EPC] (プロセスのリファレンス モデルに対応) の要件を表示できるようになります。 基本ロールへのアサインを行うと、能力と認証に関するリソースの基本ロールの要件がロール構造図に表示されます。

4.4.1.9.10 クイック モデル

[クイック モデル] モデル タイプを使用すると、メソッドの制約なしにモデリングできます。 クイック モデルには、30 個の異なるシンボルを持つ [クイック オブジェクト] オブジェクト タイプがあります。 [関係がある] タイプの関係は、クイック オブジェクト間で作成されます。 このタイプの接続線を、2 つのオブジェクト間で複数使用できます。

対応するデフォルト属性を、モデル、オブジェクト、接続線に対して指定できます。

複数のクイック モデルを、ARIS メソッドが提供するオブジェクト タイプのあらゆるオブジェクトに割り当てることができます。 さらに、モデル タイプに関係なく、ARIS メソッドが提供する任意の数のモデルを、クイック オブジェクトに割り当てることができます。

ARIS の表記生成を使用すると、[クイック モデル] モデル タイプまたは [クイック オブジェクト] オブジェクト タイプ、あるいはその両方を、メソッドに基づくモデルまたはオブジェクトに変換できます。

4.4.1.9.11 C3 メソッド

[C3 メソッド] モデル タイプでは、変更管理プロジェクトのプロセス レベルの上位にある最初のプロセス アプローチを記述します。

フォーカスは、常に、改善されるプロセスに置かれます。 調査対象のプロセスについて、プロジェクトに関連する情報を表す多数のオブジェクトが一覧形式にモデリングされます。 これには次の情報が含まれます。

- 組織的な側面 (プロセスにおける責任、代替ルールなど)
- プロセスを改善するために実行されるタスク
- プロセスの改善を測定する KPI
- プロセスの改善に使用されるツール
- 近い将来におけるプロセス変更計画のアクティビティ
- 調査されるプロセスの改善の可能性
- プロセスの実行に必要なスキル
- プロジェクトの目標
- 現在使用中のツール (ソフトウェア、メソッド、継続的なトレーニング)
- プロセスの改善およびシステム全体の統合に使用されるツール

[C3 メソッド] タイプの構造を、次の図に示します。

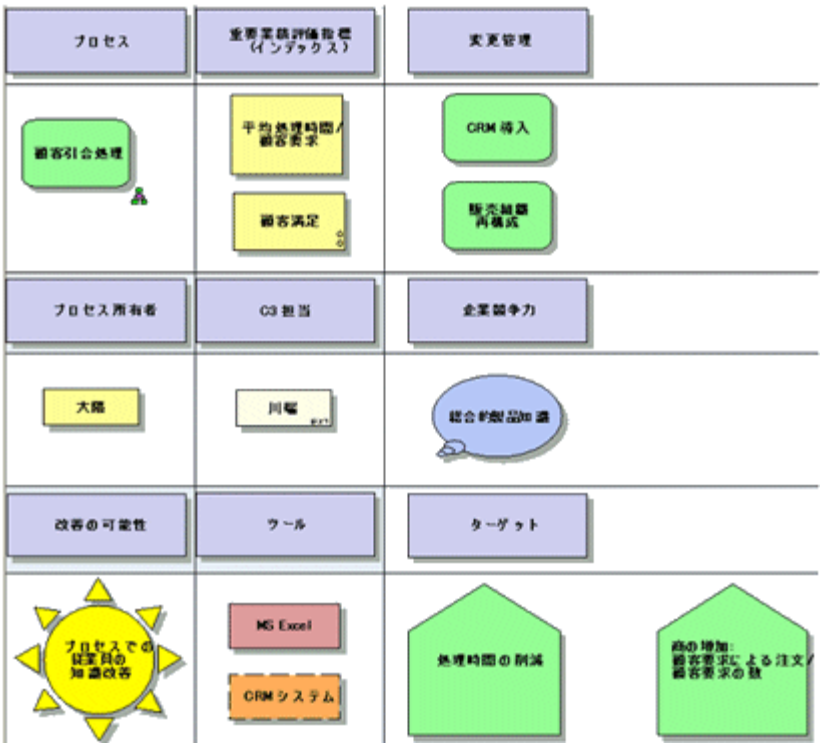


図 125: c3 モデルの構造

接続線は、プロセス/タスクとプロセス所有者、プロセス担当者、またはプロセス管理者の間でのみ引くことができます。

その他の関係は、モデル内のオブジェクトの位置から作成されます。

組織ビューのオブジェクトと同様に、[C3 メソッド] モデルのオブジェクトにあとから記録されるプロセスを割り当てることができます。

[説明/定義] 属性を使用して、オブジェクトの概要を説明できます。

4.4.1.9.12 画面設計

ソフトウェアの設計時に、ARIS で [画面設計] を使用して、ダイアログ ボックスや Web フォームの技術的要件を指定できます。

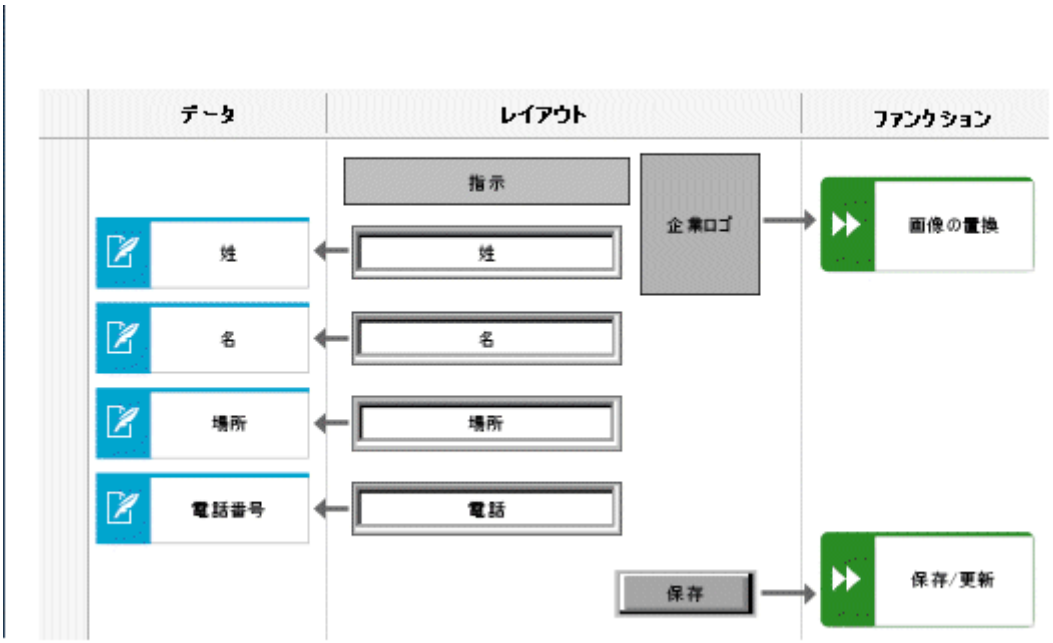
[レイアウト] 列では、ダイアログ ボックスまたは Web フォームの構造を指定します。 ダイアログ ボックスの設計手順は、開発環境でリソース エディターを使用する場合と似ています。

[レイアウト] 列には、テキスト ボックス、スピン ボックス、オプション ボタン、チェック ボックス、コンボ ボックス、ボタン、ツリー、リスト コントロール、ビットマップ、およびテキストなどのグラフィック要素を配置できます。 [タブインデックス] 属性タイプを使用して、タブ キーを使用した画面要素への移動順序を決定できます。

さまざまなデータ要素およびファンクション オブジェクトを [データ] 列および [ファンクション] 列に配置できます。 [表す] タイプの接続線によって、オブジェクトを編集中のデータ要素とファンクションに関連付けることができます。

各画面設計は、たとえば、EPC または [画面ナビゲーション] タイプのモデルなどで使用される、対応する画面オブジェクトに割り当てることができます。 さらに、画面で編集される [エンティティ タイプ]、[クラスター]、[複合オブジェクト タイプ]、[クラス]、または [ファンクション]/[IT ファンクション] にアサインすることもできます。

ScreenDesign.rsm レポートを使用して、画面設計情報をテキスト ファイルにエクスポートし、 そのテキスト ファイルを C++ 開発環境で再利用できます。



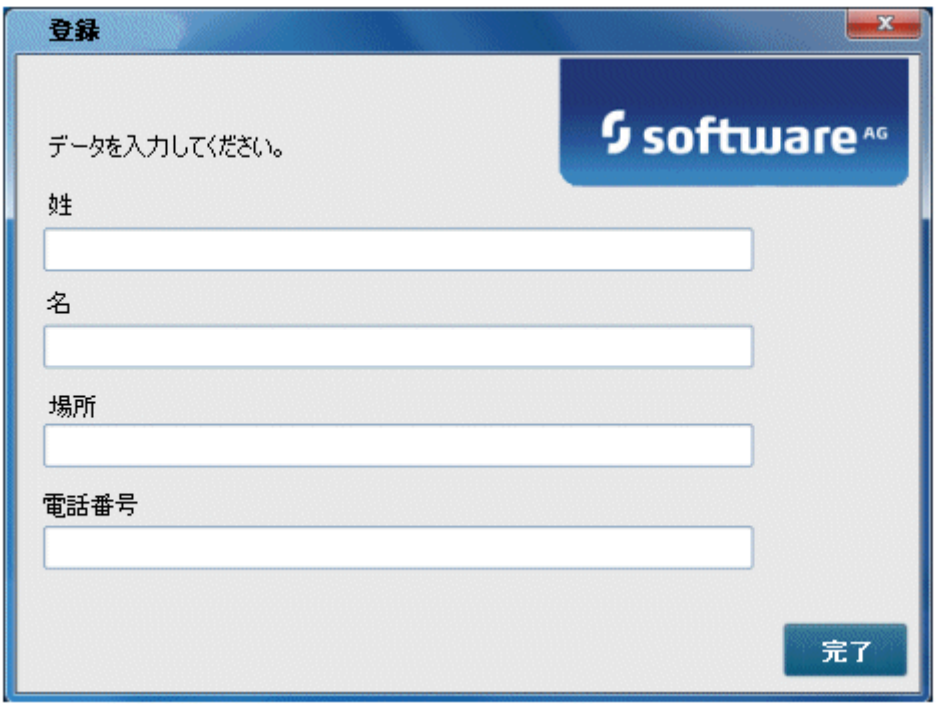


図 126: 登録ダイアログの画面設計と C++ での実装の例

4.4.1.9.13 画面ナビゲーション

[画面ナビゲーション] タイプのモデルでは、複数のフォーム フィールドやフレームがある Web サイトなど、複数のサブ画面で構成された画面の構造を指定できます。また、複数の画面間の移行順序を記述することもできます。 画面間の移行順序は、詳細に記述できます。

例

たとえば、画面要素がアクティブでないと次の画面にアクセスできないことを特に指定するとします。 この場合は、[含む] 接続線を使用して、[画面設計] モデルの起動画面アイテムを画面に割り当てます。 そして、[コールする] タイプ接続線を、画面アイテムから次に続く画面に描きます。

また、ナビゲーションがイベントに依存することを示すこともできます。 画面を閉じたときに各種イベントを開始できます。 たとえば、ユーザーがオンライン ショップの登録ページの入力を完了した際に、登録の成功/失敗を示します。 これにより、ユーザーがカタログの目次ページに進むか、登録ページに戻るかが決定されます。

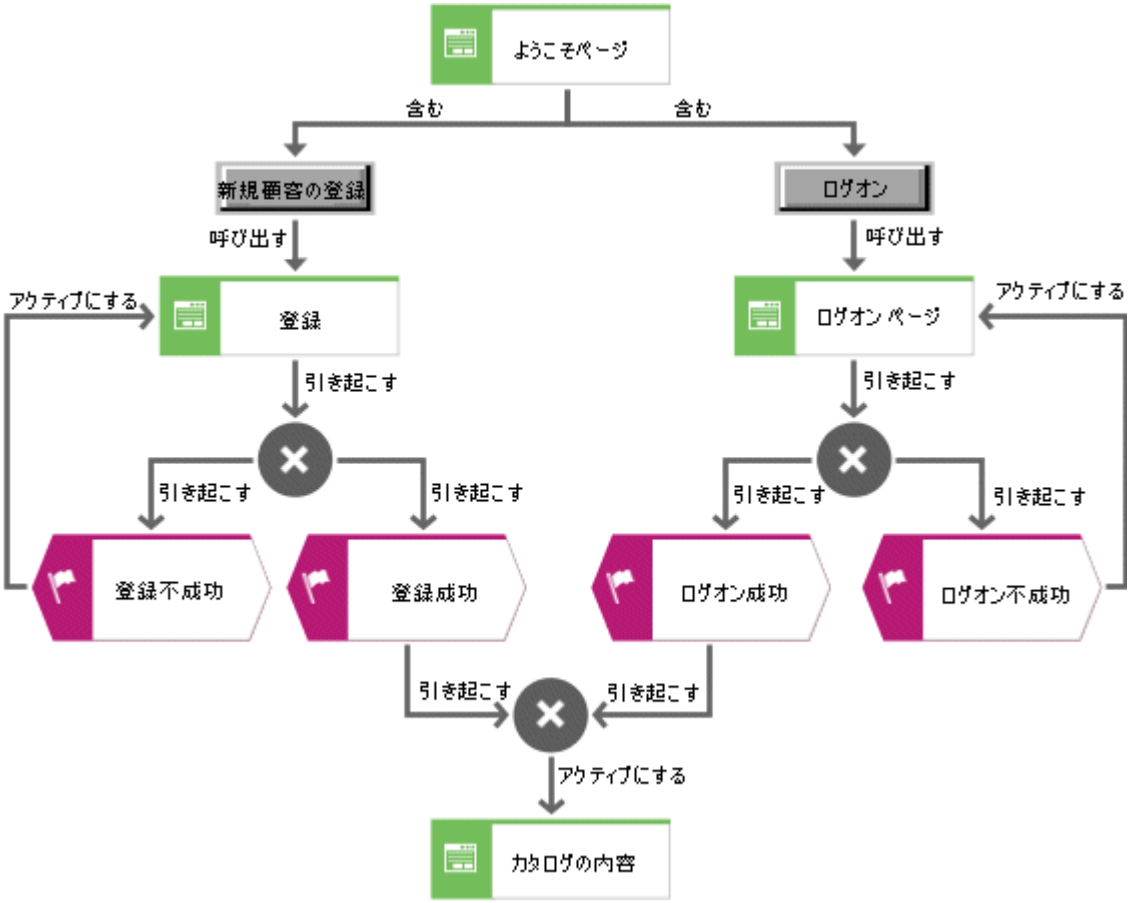


図 127: イベントによる画面ナビゲーションの例

4.4.1.9.14 事業区分マトリクス

事業区分マトリクスでは、企業が活動するさまざまな市場の概要が示され、企業の成功にとっての市場の重要性を視覚的に表わされます。

市場は次の要素によって表されます。

- 提供する製品またはサービス
- 対象とする顧客グループ

製品およびサービス ([製品/サービス] タイプのオブジェクト) は、事業区分マトリクスの 1 列目のセルに配置されます。 ターゲット グループ (さまざまな組織要素) は、1 行目のセルに配置されます。 市場を定義するには、製品の行とターゲット グループの列が交わるセルに、事業区分オブジェクトを配置します。 [事業区分に属する] タイプの非表示の関係が、製品/サービスと組織要素の間に形成されます。

事業区分の重要度を強調するために、「あまり重要でない」から「非常に重要」まで、重要度を表す 5 つのシンボルを使用できます。

モデリングの際には、事業区分はそれぞれ 1 つしかマトリクスに配置できません。

各事業区分について、企業戦略の観点からの重要度を示すことができます。 戦略とは、企業が目標の達成および競争力の獲得のために採用する長期的な計画です。

次の図は、医薬品分野における事業区分マトリクスを示したものです。

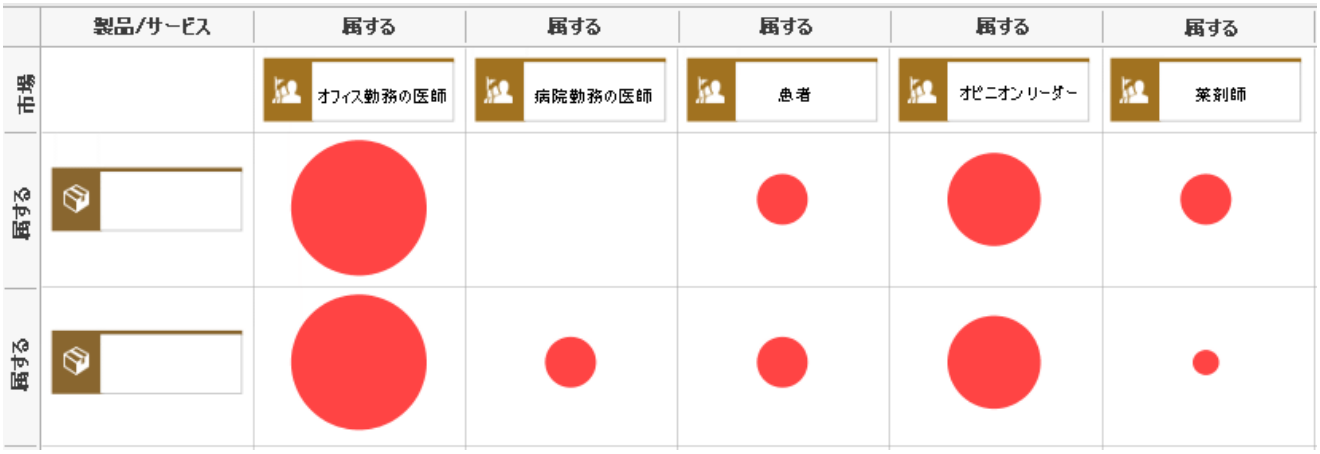


図 128: 事業区分マトリクスの例

事業区分は、目標図にアサインすることもできます。 目標図には、事業区分の目標設定、および目標の達成を助けるプロセスと成功要因が含まれます。

目標図の成功要因は、[成功 - 実績]、[成功 - 目標]、[成功 - 競合] 属性が同じ名前の属性タイプ グループに指定される場合、成功要因分析の基礎となります。 成功は「非常に低い」から「非常に高い」の 5 段階で評価できます。

手順

成功要因分析は、次の手順で実行します。

1. 事業区分のコンテキスト メニューを使用して、ARIS レポート ([評価] の [レポート]) を実行します。
2. レポート ウィザードのデフォルトのパスで [BPM] グループの MSF_Analysis(Object).rso レポート スクリプトを選択します。

レポートは HTML 形式で出力されます。

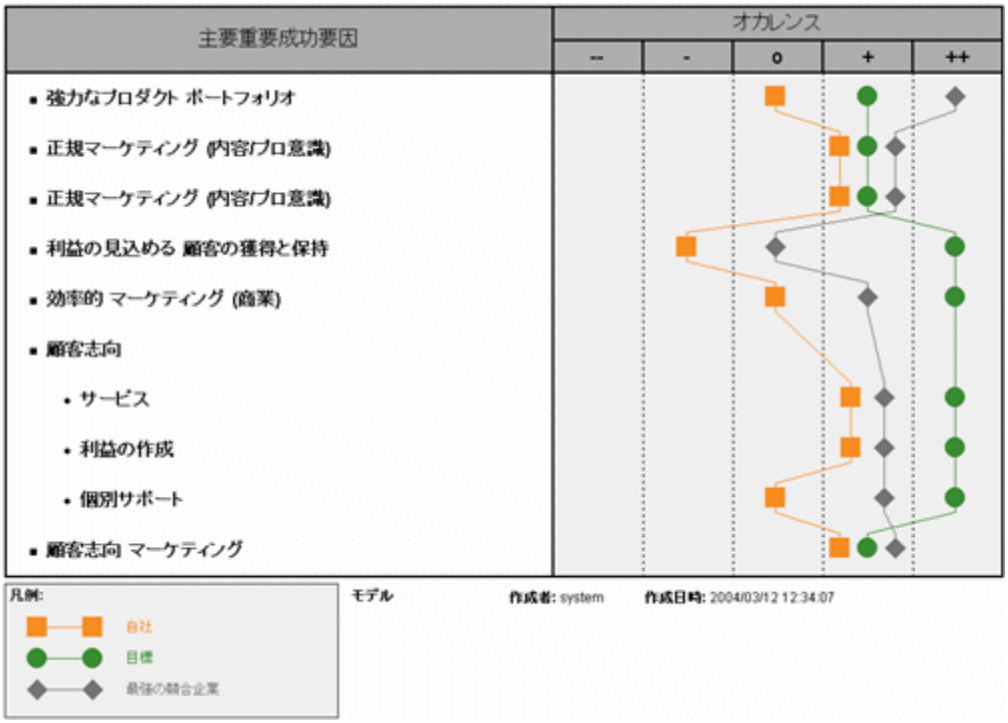


図 129: レポート

または、目標図のコンテキスト メニューを使用して成功要因分析を実行することもできます。 その場合は、MSF_Analysis(Model).rsm レポート スクリプトを選択します。

4.4.2 仕様設計

4.4.2.1 アクセス図

プロセス ビューのアクセス図には、以下の図に示した、ほかのビューの仕様設計記述で説明されたオブジェクト間の関係を含めることができます。 この図を明確に作成するには、2 つの独立した関係に分割して個別に扱う必要があります。

4.4.2.1.1 ファンクションとデータの接続

最初に、アプリケーション システム タイプ、モジュール タイプ、または IT ファンクション間の情報フローを定義できます。 このために、該当するアプリケーション システム タイプまたはモジュール タイプ間に情報フロー オブジェクトが作成されます。 より詳細にシステム タイプ間の情報フローを指定するために、eERM 図、リレーション図、またはテーブル図が情報フロー オブジェクトと関連付けられます。 したがって、情報フロー オブジェクトは要件定義レベル、仕様設計レベル、または実装レベルのいずれかで配置されます。

次の図に例を示します。

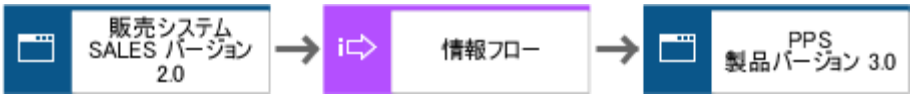


図 130: アプリケーション システム タイプ間での情報フロー

情報フローのほかに、各アプリケーション システム タイプ、モジュール タイプ、および IT ファンクション タイプの入出力データは、要件定義または仕様設計のデータ オブジェクトとして表すことができます。 矢印の向きは、データ フローの、入って来る（入力）方向または出て行く（出力）方向を示します。

次の図に例を示します。

図 131: 仕様設計レベルにおける I/O データ

4.4.2.1.2 組織とデータの接続

仕様設計レベルでデータ ビューと組織ビューを関連付けるときに実行する必要のあるタスクは、組織ユニットが企業データ オブジェクトに対して持つ責任を定義することと、組織ユニットが特定の企業データにアクセスできることを定義するアクセス権を決定することです。

組織ビューの技術オブジェクト (組織ユニット、役職、ロール、要員など) と、仕様設計レベルでのリレーション図のデータ オブジェクト (リレーション、属性、ビュー) の関係は、このようにして確立されます。 したがって、これらの関係はプロセス ビューの仕様設計レベルに割り当てられます。

リレーションや個別フィールドに対するアクセス権を定義するために、関連するデータ オブジェクトを役職とロールのいずれかに割り当てることができます。 これにより、特定の役職が特定のフィールドへのアクセス権限を持つことを規定できるだけではなく、ロールを割り当てることによって、「このフィールドは部署長のみがアクセスできる」などのビジネス ルールを定義することができます。 次の図に例を示します。

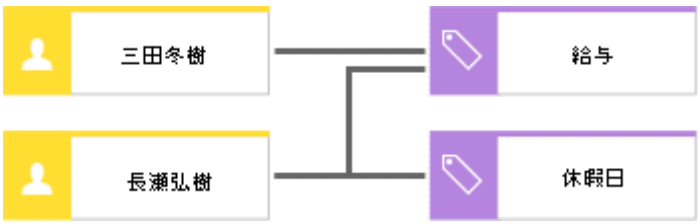


図 132: アクセス権

フィールドの内容やリレーション全体に対する責任の定義も、アクセス権の割り当てと同様に重要です。 このため、[責任を持つ] という 2 番目の接続線を、リレーション図の組織ユニットとデータ オブジェクト間に描画できます。 アクセス権とは異なり、データ オブジェクトに対する責任は、多くの場合、企業内の 1 つの役職にのみ割り当てられます。 ここでも、ロールを割り当てることによって、前述のものと同様のビジネス ルールを定義できます。 これらのルールは、データ オブジェクトに対する責任に関連付けられます。

次の図に例を示します。

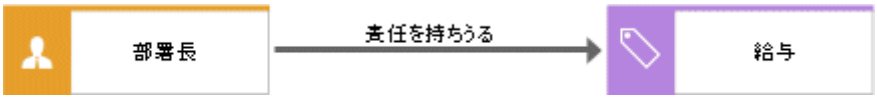


図 133: 責任の定義

4.4.2.1.3 組織とファンクションの接続

組織的な側面が、仕様設計レベルで定義されたファンクション的な側面に関連付けられるという事実は、次の疑問に対する基本的な解答になります。

- だれが（どの組織ユニット、役職、要員などが）仕様設計レベルのファンクション ビューで指定されたアプリケーション システム タイプとモジュール タイプに対する責任を持つか。また、だれがこれらのシステムを使用するのか。
- 企業内のどの場所（組織ビュー）でどのアプリケーション システム タイプまたはモジュール タイプを使用するのか。
- 企業で使用できるプラットフォーム（ハードウェア コンポーネント タイプ（組織ビュー））は、どのアプリケーション システム タイプの実行に適しているのか。

最初の疑問に答えるために、アクセス図で、組織図の組織ユニット（組織ユニット、役職、および要員）と、アプリケーション システム タイプ図のオブジェクト（アプリケーション システム タイプ、モジュール タイプ、IT ファンクションなど）間に接続線を描くことができます。 接続線を描きながら、この関係の重要性をより正確に指定できます。 次のように状況を区別します。

- 「テクニカルな観点」に関する限り、組織ユニットがアプリケーション システム タイプに対する責任を持つことができる
- 組織ユニットが、アプリケーション システム タイプの「開発」に対する責任を持つことができる
- 組織ユニットが、アプリケーション システム タイプの「ユーザー」になることができる

場所に関しては、組織ビューからアプリケーション システム タイプ、モジュール タイプ、および IT ファンクションに場所を割り当てることによって解決できます。

仕様設計では、個別のライセンスという意味で個別のアプリケーション システムを扱うことはできませんが、アプリケーション システム タイプを扱うことはできます。 つまり、実際のアプリケーション システムの場所は、この関係では定義されません（これは、実装レベルで実現されます）が、特定のアプリケーション システム タイプに対して使用できる場所は指定できます。

組織ビューの仕様設計で、企業で使用できるハードウェア コンポーネント タイプを定義します。 プロセス ビューで、これらのハードウェア コンポーネント タイプとアプリケーション システム タイプとの関係を確立できます。 このようにして、特定のアプリケーション システム タイプ、モジュール タイプ、IT ファンクション タイプを実行するためのハードウェア プラットフォームが決定されます。 この段階では、ファンクション ビューに記述されるグラフィカル ユーザー インターフェイス タイプ、オペレーティング システム タイプ、および DBMS タイプもハードウェア コンポーネント タイプに割り当てられます。

アクセス図で使用可能な関係の一覧は、インストール媒体に含まれている『ARIS メソッド マニュアル - テーブル』マニュアル (ARIS Method tables.pdf) を参照してください。

次の図に関係の例を示します。

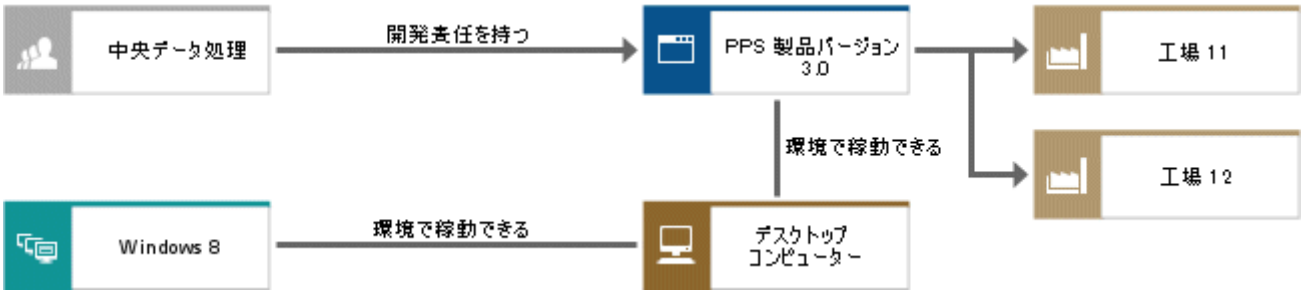


図 134: アクセス図（抜粋）

4.4.2.2 プログラム フロー図

アクセス図では、組織ビューとデータ ビューのオブジェクト タイプと、アプリケーション システム タイプ図で指定されているアプリケーション システム タイプ、モジュール タイプ、および IT ファンクション タイプとの間の関係を作成できます (「アクセス図『120 ページ 』」の章を参照)。 このモデル タイプでは、要件定義のファンクションの割り当てを直接表現することはできません。この割り当ては、[アプリケーション システム タイプ図] で実現されます。 同様に、[アプリケーション システム タイプ]、[モジュール タイプ]、および [IT ファンクション] タイプの、発生する可能性のある時系列的な操作順序も直接表現することはできません。 ARIS アーキテクチャに厳密に従って、多数のモデル タイプの間を移動するだけで、これらのリンクをたどることができます。

ただし、システム設計環境には、システム設計のあらゆる側面を統合したビューを可能にするモデル タイプ (プログラム フロー図 (PF)『123 ページ 』など) が用意されています。

ARIS に [プログラム フロー図] モデル タイプが含まれているのはこのためです。 このモデル タイプを使用すると、ARIS の分割ビューに関係なく、ほかの ARIS モデル タイプで使用可能なアプリケーション システム タイプ、モジュール タイプ、および IT ファンクション タイプへのすべての関係をモデリングできます。 さらに、前述したオブジェクト タイプの時系列な操作順序を表示できます。 このために、このモデル タイプではイベントが提供されます。 EPC でファンクションとイベントを配置する場合と同様に、プログラム フロー図で一連のモジュールを定義できます。 このコンテキストでは、イベントはモジュール タイプまたはアプリケーション システム タイプを起動するトリガーと見なされます。 分岐は、EPC で説明したルールで表すことができます。EPC とは異なり、プログラム フロー図では、イベントを使用せずに操作順序も定義できます。

4.4.2.3 プログラム フロー図 (PF)

[プログラム フロー図 (PF)] は、プログラムの処理の順序を表すために使用されます。 処理手順の順序はオブジェクト間の関係によって表されます。 この図では、データは示されません。

以下の図に、現金自動預払機（ATM）の処理順序の簡単な例を示します。 処理順序の説明では、実装に焦点を置きます。

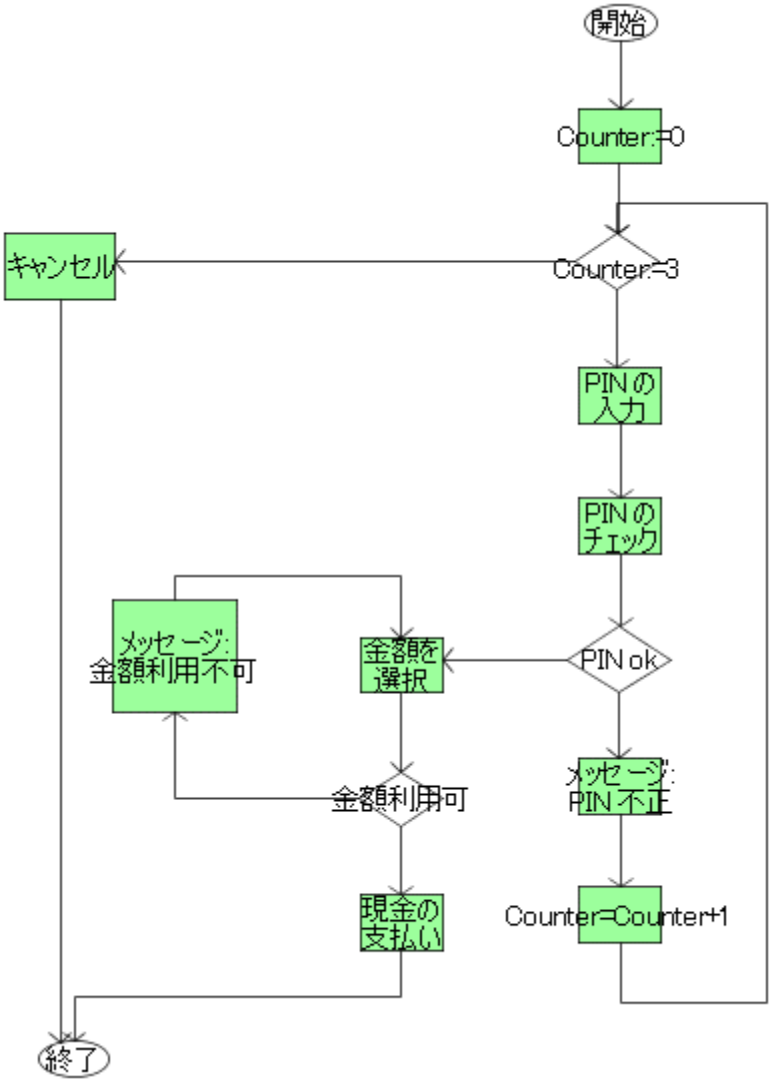


図 135: プログラム フロー図（PF）の例

4.4.2.4 画面図

画面図は、ソフトウェアの開発で使用する画面について説明します。 画面図から画面の自動生成を行うことが目的です。

したがって、画面図により構造が示され、画面の機能もある程度示されます。 左右および上下方向において、画面図の構造は記述されるインターフェイスの配置と対応します。

中央のシンボルは［画面］です。Windows の用語に従うと、これはウィンドウを表します。 このウィンドウには、複数のタブ（［ページ］シンボル）を含めることができます。 一般に、インターフェイスは、テーブル形式（行に対する［セクション］シンボル、および列に対する［列］シンボル）を使用していくつかの領域に分割できます。 複雑なインターフェイスを構成するために、［セクション］シンボルと［列］シンボルは、任意にネストさせることができます。 インターフェイスには、テーブル（［画面テーブル］シンボル）、テキスト入力ボックス（［COT 属性］シンボル）、グラフィック（［ビットマップ］シンボル）とテキスト（［テキス

ト] シンボル) を配置できます。 [レイアウト] シンボルを使用して、表示プロパティを [画面]、[ページ]、[セクション]、[列]、[画面テーブル]、[COT 属性]、[テキスト] オブジェクトに割り当てることができます。

その他のシンボルも、画面インターフェイスの記述に使用できます。

次の図に、画面図の例を示します。 2 番目の図は、画面図から抜粋した画面の説明です。

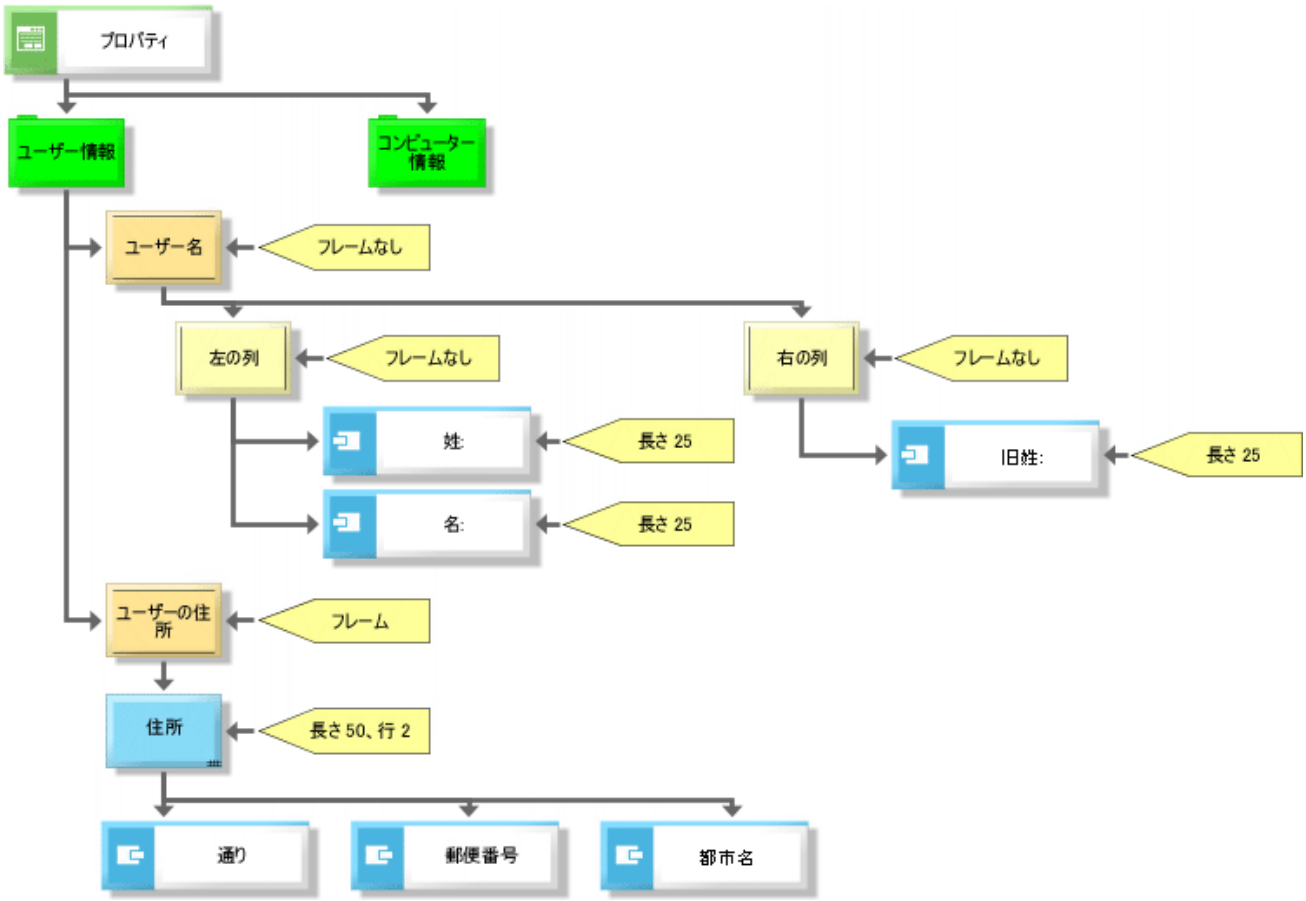


図 136: 画面図の例

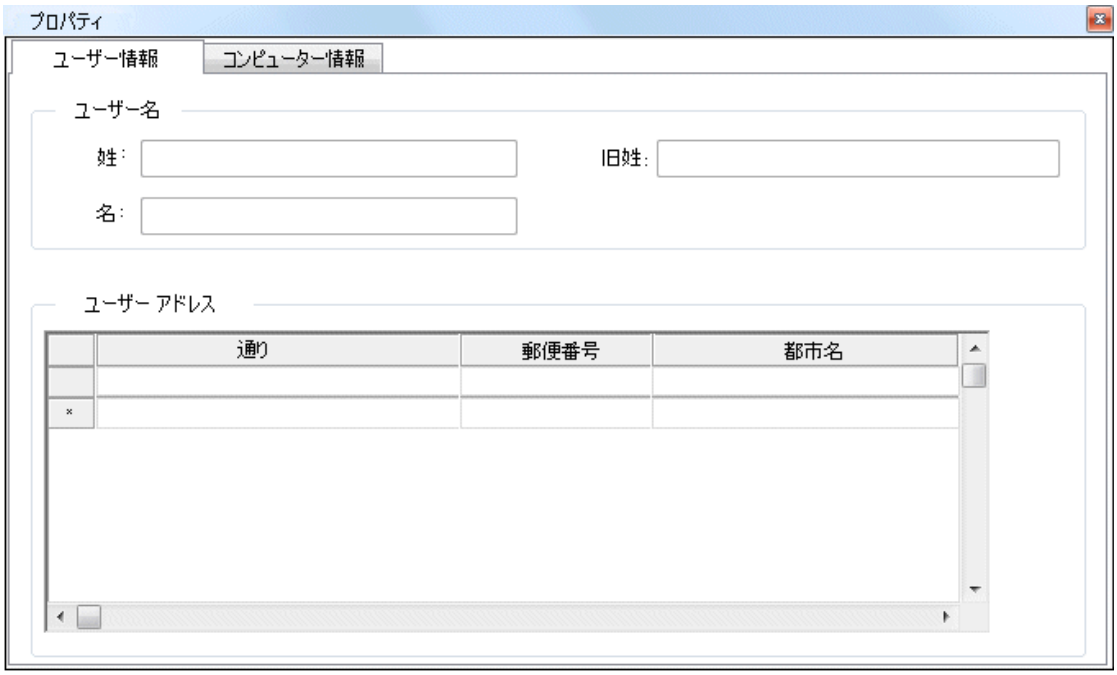


図 137: 前の図の画面図から取得した画面

4.4.2.5 SAP integration process (XI)

[SAP Integration process (XI)] タイプのモデルは、SAP Exchange Infrastructure の業務プロセスを表します。

このモデルでは、アプリケーション間でメッセージを交換するために実行する必要のあるステップを詳細に記述します。

統合プロセスには、開始オブジェクトとプロセスの終了がそれぞれ 1 つずつあります。

統合プロセスの過程で、以下の構成要素が作成されます。

- 受信
- 送信
- 待機
- ブロック範囲 (開始)
- ブロック範囲 (終了)
- 平行性 (開始)
- 平行性 (終了)
- コントロール
- 受信者の決定
- アサインメント
- 変換
- 切り替え (開始)

- 切り替え（終了）
- 空白
- ループ範囲（開始）
- ループ範囲（終了）

ブロック、平行性、切り替えまたはループの開始の後でパスが複数のパスに分割される場合、それらのパスは対応する終了シンボルに結合される必要があります。

4.4.3 実装 - アクセス図（物理的）

プロセス ビューの仕様設計で検討された疑問は、実装レベルにも関連しています。 ただし、仕様設計レベルとは異なり、オブジェクト タイプではなく、個々のオブジェクトの具体例を考察します。 たとえば、アプリケーション システム タイプと組織ユニット間に存在する関係ではなく、具体的なアプリケーション システムと組織ユニット間に存在する関係に焦点を置きます。

次の図に示す関係は、アクセス図（物理的）でモデリングされています。

4.4.3.1 ファンクションとデータの接続

アプリケーション システム間でどのデータが交換されるかを示すために、情報フロー オブジェクトをファンクション ビューのアプリケーション システム オブジェクト間で作成することができます。 仕様設計レベルのアプリケーション システム オブジェクトとは対照的に、これらのアプリケーション システム オブジェクトはアプリケーション システム タイプではなく、具体例（個別のライセンス）です。 これは、アプリケーション システム、モジュール、およびプログラム モジュール タイプを、データ フローの接続線によって連結できることを意味します。「販売システム SD バージョン 2.1」モジュール タイプが「材料管理システム MM バージョン 1.2」モジュール タイプとデータを交換できると仕様設計レベルで定義されている場合、実装レベルでは、インストールされている特定の「SD ライセンス 1234」が「MM ライセンス 2352」モジュールおよび「MM ライセンス 34234」モジュールとデータを交換することが表されます。 両方の MM モジュールは、「材料管理システム MM バージョン 1.2」モジュール タイプです。 次の図に、これを示します。

図 138: データ フロー

システム間で詳細に変換されるデータ オブジェクトを指定するために、データ ビューの対応するモデル タイプは情報フロー オブジェクトにアサインされます。

アプリケーション システム間のデータ フローだけでなく、各アプリケーション システムの入出力データも指定できます。 アクセス図（物理的）に関係を表現する場合は 2 つあります。 1 つ目の場合は、データ オブジェクトが、実装レベルのデータ ビューに配置される、テーブル図（テーブル、フィールド、ビュー（物理的））のオブジェクトである場合です。 これらのデータ オブジェクトは、入出力の関係を使用して仕様設計レベルまたは実装レベルのアプリケーション システム オブジェクトに接続できます。

2 つ目の場合は、アプリケーション システム オブジェクトが、データ ビューのオブジェクトに接続される実装レベルの、具体的なアプリケーション システムまたはモジュールである場合です。

つまり、次の一般的なルールを定義できます。

入出力の関係に関係するオブジェクト タイプのいずれかが、関連するビューの実装レベルから派生している場合、プロセス ビューの関係も実装レベル (アクセス図 (物理的)) で表現されます。

次の図に例を示します。

図 139: 入出力の関係

4.4.3.2 組織とデータの接続

ここでは、仕様設計で扱った疑問と同じ疑問が焦点となります。

- どの組織ユニットが、データ オブジェクトに対して責任があるのか。
- だれがどのデータ オブジェクトに対するアクセス権を持つのか。
- どのデータ オブジェクトがどのハードウェア コンポーネントに保存されるのか。

仕様設計の関係とは対照的に、ここでは、データ ビューの実装レベルに表示されるデータ オブジェクトへの接続から関係が構築されます。

これは、データ オブジェクトに対する責任がリレーション図のリレーションと属性に対して定義されるのではなく、テーブル、フィールドおよびその実例 (テーブル (実例)、フィールド (実例)) などの物理的な構造に対して定義されることを意味します。

これらの依存関係を表現するために、組織ビューのオブジェクト (組織ユニット、役職、要員など) と、前に説明したテーブル図のオブジェクト (テーブル、フィールド、ビュー (物理的) など) の間に接続線が描かれます。

組織ユニットとテーブルおよびフィールド間に接続線を描くとき、各関係の意味を個別に定義する必要があります。 たとえば、[責任を持つ] は、その組織ユニットが、それぞれのテーブルやフィールドの内容に対する責任を持つことを意味し、[アクセスする] は、その役職または要員が、示されているデータ オブジェクトに対するアクセス権を持っていることを意味します。

アクセス権と責任の定義に加えて、ハードウェア コンポーネント オブジェクト (組織ビュー/実装) を使用して、(たとえば資産番号で一意に識別できる) どの使用可能なハードウェア コンポーネントに企業の特定の情報オブジェクトを保存するかを定義します。 このため、[ハードウェア コンポーネント] オブジェクトは、アクセス図 (物理的) で実装レベル (テーブル、フィールドなど)、仕様設計レベル (関係、属性)、または要件定義レベル (エンティティ タイプ、クラスター/データ モデルなど) の情報オブジェクトにも接続できます。

次の図に例を示します。

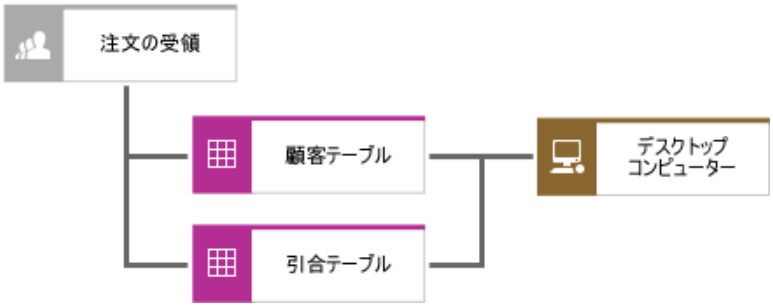


図 140: ハードウェア コンポーネントの割り当て

4.4.3.3 組織とファンクションの接続

アクセス図 (物理的) で、組織ビューとファンクション ビューのオブジェクト間で定義された関係により、次の疑問に答えることができます。

どのアプリケーション システムがどのハードウェア コンポーネントで実行されているか、どのアプリケーション システム タイプを実行できるか。

これらの依存関係を表すために、実装レベル (アプリケーション システム、モジュール、プログラム モジュールなど) または仕様設計レベル (アプリケーション システム タイプ、モジュール タイプなど) のアプリケーション システム オブジェクトと、組織ビューの [ハードウェア コンポーネント] オブジェクト タイプの間に、[プラットフォームである] 関係および [プラットフォームになりうる] 関係をモデリングできます。

次の図に例を示します。

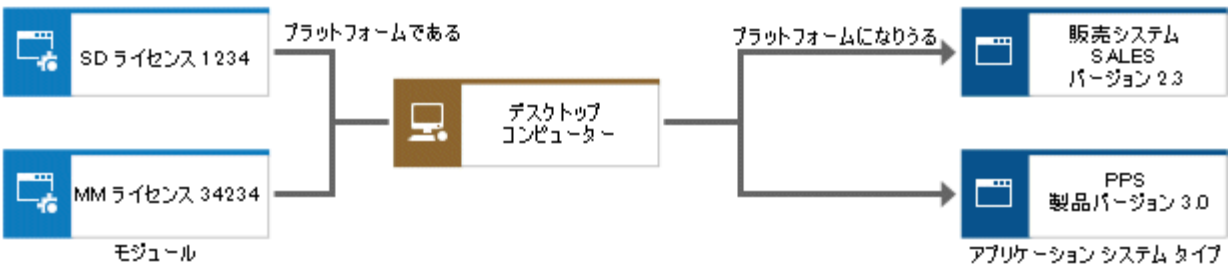


図 141: プラットフォームとしてのハードウェア コンポーネント

どの組織ユニットが特定のアプリケーション システムを使用するか。

仕様設計レベルは特定のアプリケーション システム タイプにアクセスするユーザーを定義するレベルであり、実装レベルは特定のアプリケーション システム (個別ライセンス) にこの関係を定義することができます。 たとえば、ARIS Architect アプリケーション システム タイプの複数のライセンスが 1 つの企業内で異なる設定で使用できます。 アクセス図 (物理的) には、どのユーザーがどのライセンスを使用しているかを表示することができます。 この目的のために、[使用する] 接続線を使用して [組織ユニット]、[役職]、[要員] オブジェクト タイプを [アプリケーション システム] と [モジュール] オブジェクト タイプに接続できます。 次の図に例を示します。

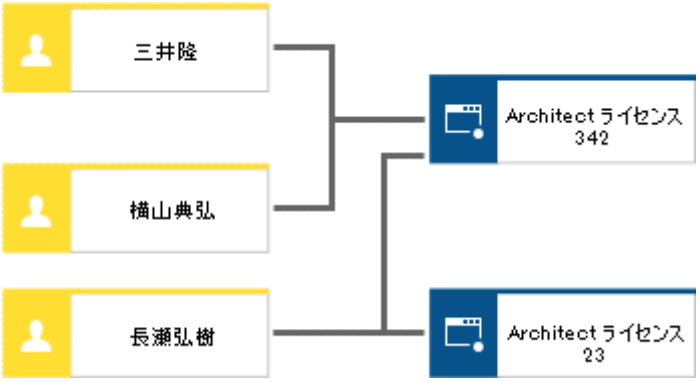


図 142: ユーザーとアプリケーション システム

企業内のどこにアプリケーション システムがあるか。

仕様設計では「アプリケーション システム タイプ」-「場所」という関係を使用して、企業内の特定の場所に配置できるアプリケーション システム タイプを定義することができます。 アプリケーション システム タイプに割り当てられている個別ライセンスの企業内での使用場所を正確に指定するために、アクセス図 (物理的) の [アプリケーション システム]、[モジュール]、および [IT ファンクション] オブジェクト タイプに場所を接続することが可能です。

次の図に例を示します。

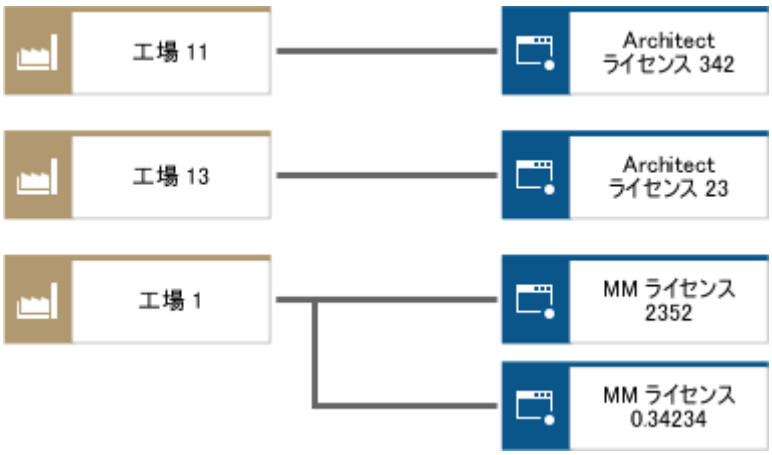


図 143: 場所の割り当て

アクセス図 (物理的) で使用可能なすべての関係の一覧は、インストール媒体に含まれている『ARIS メソッド マニュアル - テーブル』マニュアル (ARIS Method tables.pdf) を参照してください。

4.5 製品 / サービスのモデリング

ARIS では、企業が提供している製品およびサービスを表すためのさまざまなモデル タイプを用意しています。

製品またはサービスは、付加価値プロセスで生成または提供されます。 これは、人間による動作または技術的な処理の結果です。 サービスまたは製品という用語は、サービスまたは商品を提供することを意味します。

製品は、消費製品、材料タイプ、運用リソース タイプ、機材タイプ、または包装材などです。 組織ユニットまたは顧客のニーズにより、製品/サービスを作成する必要性が生じます。 製品は、実体のある商品として顧客に提供されます。

サービスは、同時に生産され消費されるという特徴を持つ無形の製品です。

サービスの提供元には、銀行、保険会社、政府機関などがあります。

製品提供元の市場区分に対する顧客の関心が高まるにつれ、その提供元にとって製品環境におけるサービスの順守と向上が重要になってきます。

そのため、ARIS で使用可能なさまざまなモデル タイプは各製品またはサービス、または製品とサービスの両方を記述するために設計されています。

製品/サービスのモデリングでは以下のモデル タイプを使用できます。

- 製品/サービス交換図
- 製品/サービス ツリー
- 製品割当図
- 製品ツリー
- 製品選択マトリクス
- 競争力モデル

4.5.1 製品 / サービス交換図

製品/サービス交換図では、製品/サービスの作成、および企業内での製品/サービスの交換を説明します。「製品/サービス」という用語は、サービスまたは製品を提供することを意味します。これはそれぞれ、対応するシンボルで表されます。 製品には、たとえば EPC (マテリアル フロー付き) で使用される、マテリアル タイプ、運用リソース タイプ、機材タイプ、包装材タイプなどがあります。 ファンクションの入力または出力、またはその両方である製品またはサービスは、これらのファンクションの開始イベントまたは終了イベントに接続できます。

業務管理ファンクション間の製品およびサービスの交換は、付加価値連鎖図と EPC の間の抽象化レベルで活用できます。 製品/サービスの交換関係は、ファンクションの視点からだけでなく、組織の視点からも表すことができます。 [製品/サービス交換図] には、これに利用できるさまざまなモデル作成オプションがあります。

次の図に製品/サービス交換図の例を示します。

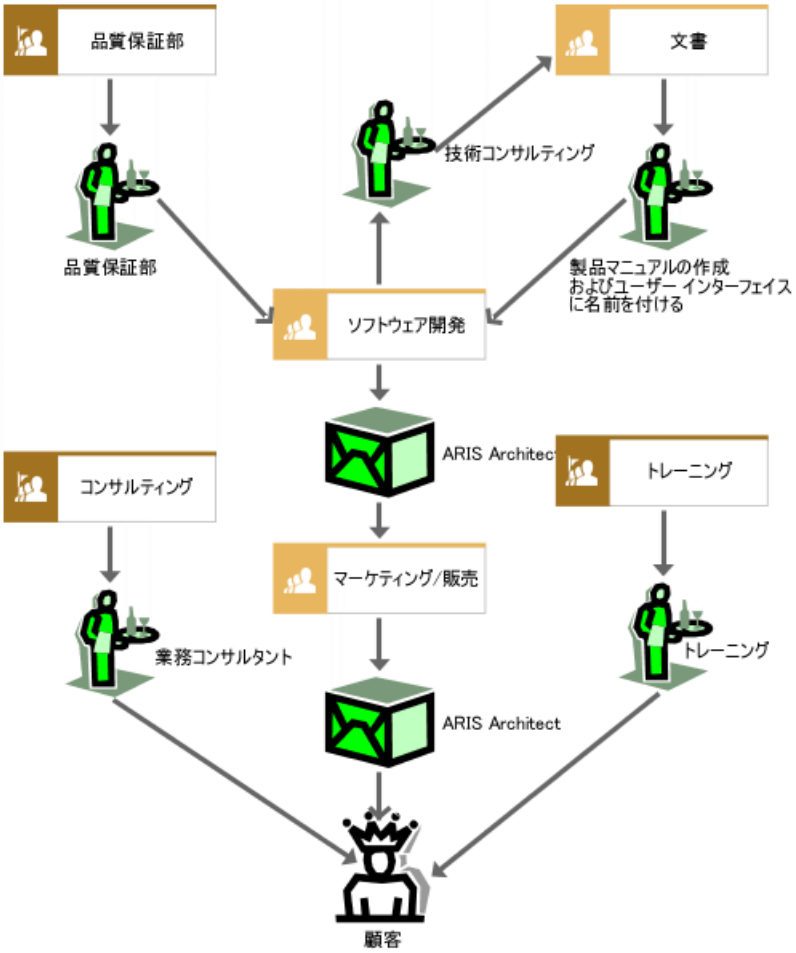


図 144: あるソフトウェア会社における製品/サービス交換の例

4.5.2 製品/サービス ツリー

製品またはサービスは、さまざまな抽象化レベルから見ることができます。 したがって、製品またはサービス全体を構成する製品またはサービスの一部を表すモデルに、これらの関係を含めると便利です。 このような静的な側面は、[製品/サービス ツリー] で表します。 たとえば、複雑な製品には複数のモジュールが含まれていることがよくあり、各モジュールはさまざまなコンポーネントで構成されています。 これらの各要素を製品またはサービスと見なすことができます。

[関係がある] 接続線を使用して、ほかの種類の依存関係を表すことができます。この接続線は、製品/サービス ツリーの製品/サービス間でも使用できます。 このような関係には、消費者ローンと返済が行われる当座預金口座間の関係が含まれます。

潜在的な代替製品またはサービスなど、ほかの製品/サービスとの代替関係も表すことができます。

また、この静的なモデルは製品またはサービスと (企業の) 目標の間の相互関係も表します。

次の図に製品/サービス ツリーの例を示します。

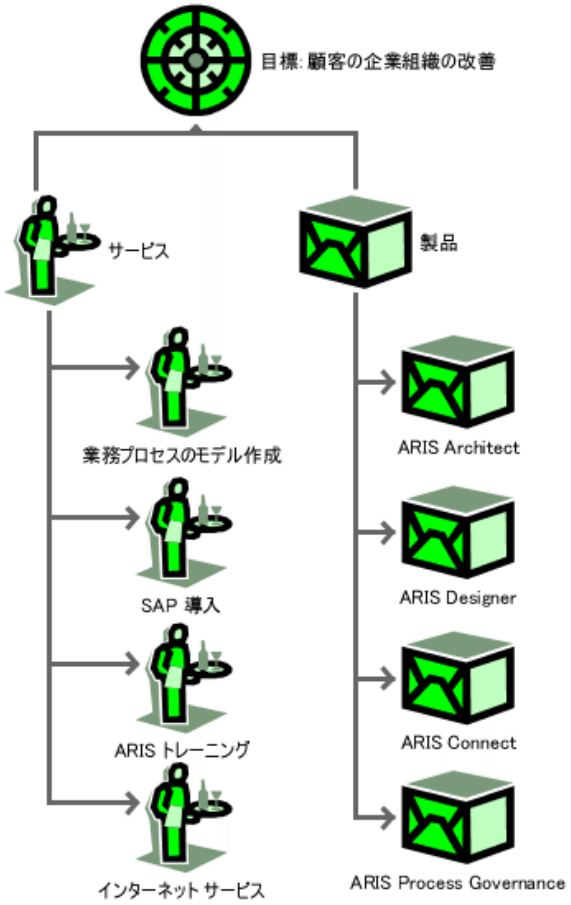


図 145: 製品/サービス ツリー

4.5.3 製品割当図

グラフィック モデルに属する全般的な製品/サービス図のほかに、抽象的な表現を実現するために製品モデルを使用できます。製品割当図は、主に公共機関における製品の生産を分析するために使用します。 製品/サービス交換図と同様に、このモデル タイプを使用して、組織ユニットが提供または使用している製品、製品の生産に必要なファンクション、製品が入力を提供しているファンクションを表すことができます。 さらに、ここには各製品の (法的な) 注文根拠を示します。 さまざまな製品を使って達成すべき目標を表すこともできます。

次の図は、政府機関向けの製品割当図の一部を示します。

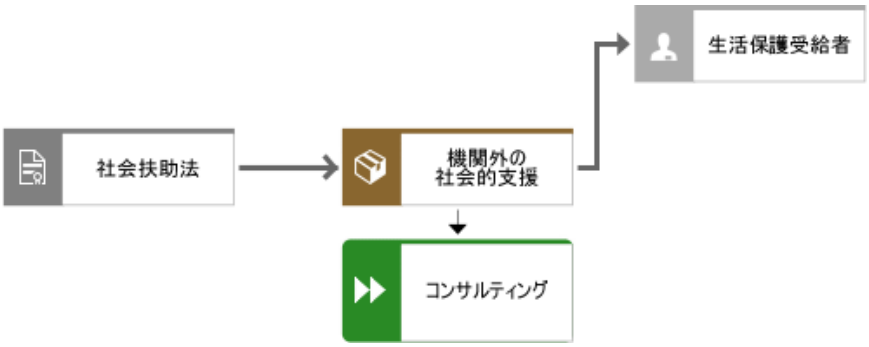


図 146: 製品割当図の例

このモデル タイプを使用して、製品のマーケティングに関係する情報を表すこともできます。

次に、銀行商品の簡単な例でこれらの側面を説明します。

過去 10 年間におけるインターネットの発展と個人インターネット ユーザーの増加により、オンライン バンキングが急速に普及しました。 同時に、青少年の財力も増加し、ターゲット層として重要な位置を占めるようになりました。

この結果、「当座預金口座」サービスは、以前とは異なる形態で提供されるようになり ました。

たとえば、銀行の支店のスタッフが預金者の応対を行うような「シニア口座」を提供することができます。 この商品は、通常、新しい技術に詳しくなく、担当者の応対やアドバイスを重視し、高齢のために行動範囲がそれほど広くない方を対象者とします。 そのような口座に請求される料金は平均以上であることがあります。

当座預金口座の別の種類として、低料金でオンラインの「ジュニア口座」が考えられます。 この商品は、インターネット技術を使って口座を利用する、利用額が少ない 12 ～ 20 才のティーンエージャを対象者とします。 このため、口座の利用料金を安価に設定する必要があります。

これらの 2 つの商品の製品割当図を以下に示します。

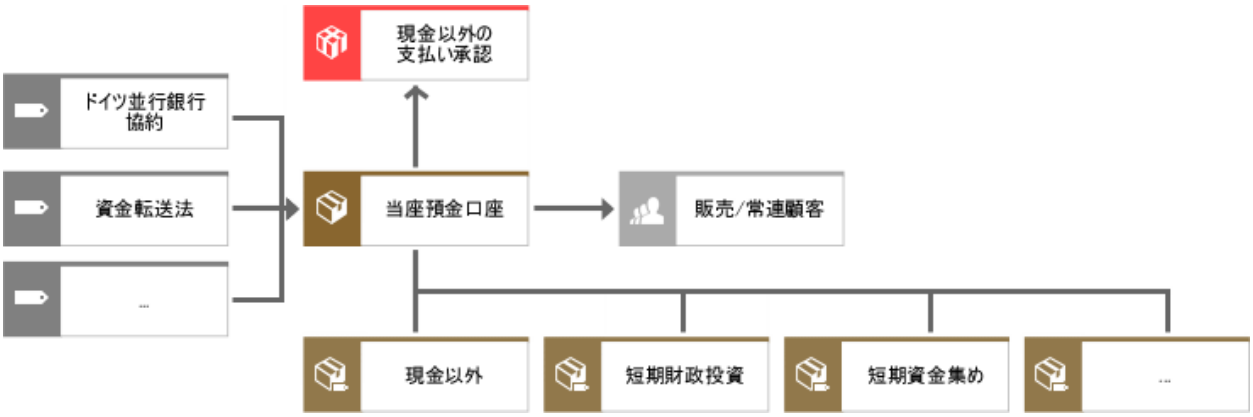


図 147: 製品割当図 - 当座預金口座

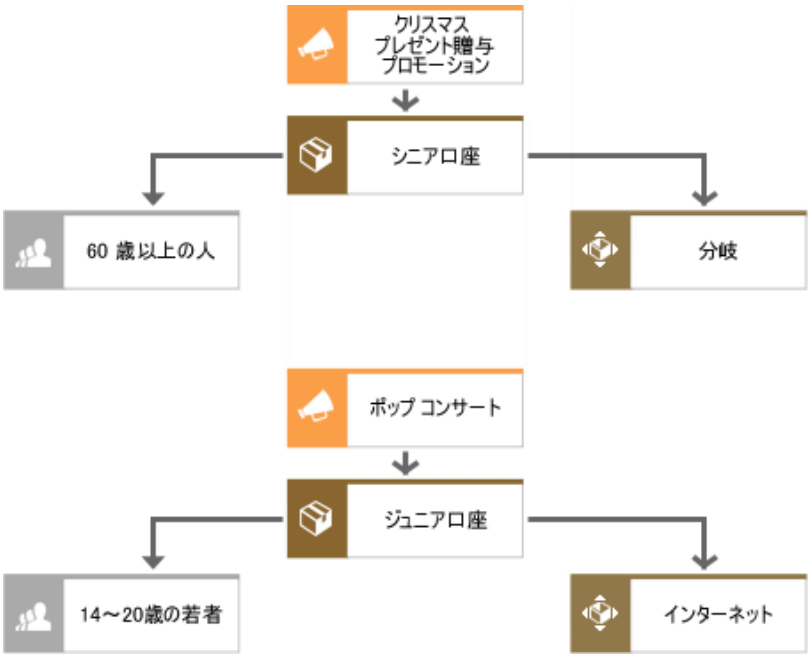


図 148: 製品割当図 - 販売品

「ジュニア口座」サービスと「シニア口座」サービスは、当座預金口座のオブジェクト バリアントとして作成され、[販売品] 属性によって識別されます。 販売品とは、ある会社により提供される製品またはサービスです。 市場区分によって異なる名前で提供されます。 一般的に、販売品ごとに異なるマーケティング手段が使用されます。

ARIS バリアント コンポーネントを使用して、ある製品から販売品を必要な数だけ作成できます。

4.5.4 製品ツリー

製品ツリーの目的は、公共機関における製品の構成を分析することです。 このモデルは、基本的に製品/サービス ツリーに対応していますが、代替製品のモデル化はできません。 製品ツリーは、製品/サービス ビューの要件定義レベルにあります。

次の図に製品ツリーの例を示します。

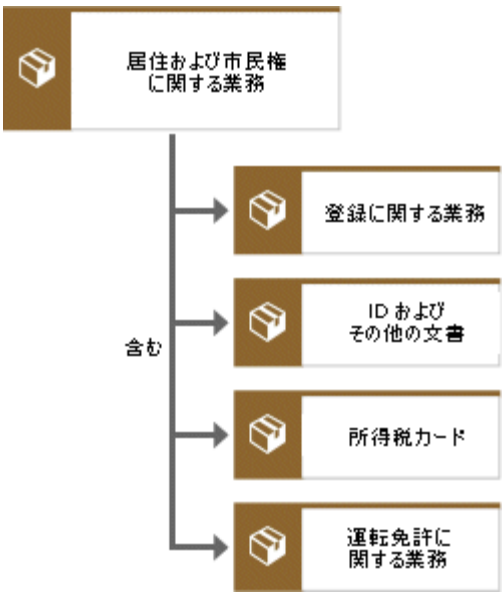


図 149: 製品ツリーを利用した、「市民登録業務」製品グループの分類

4.5.5 製品選択マトリクス

製品選択マトリクスでは、組織ユニットおよびその責任範囲となる製品を定義します。 製品の作成に必要なファンクションを製品に割り当てることができます。 このモデルは、製品の作成に関連する組織図、製品ツリー、プロセスを実現する出発点として適しています。 次の図は、製品選択マトリクスの例を表しています。

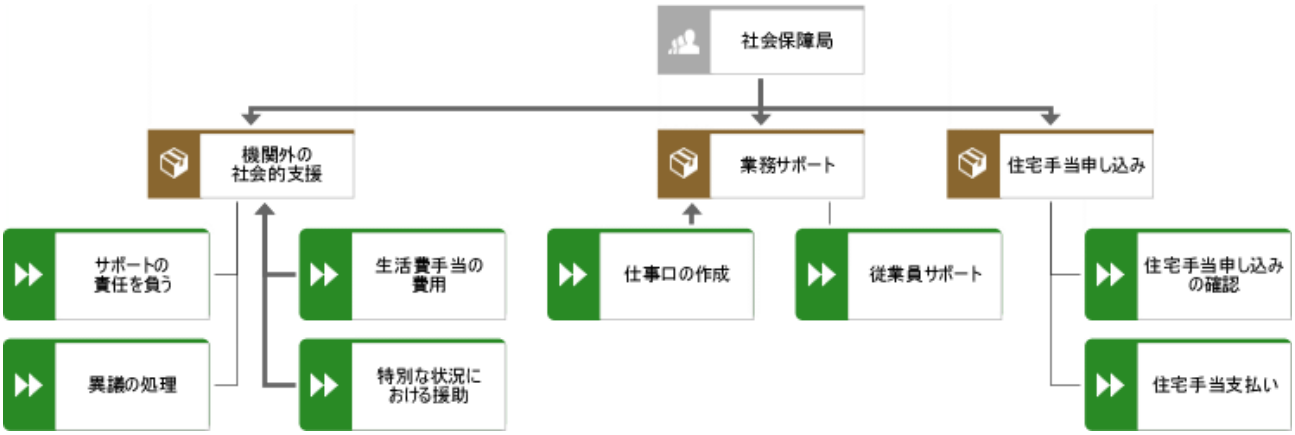


図 150: 社会福祉事務所の製品選択マトリクス

4.5.6 競争力モデル

このモデルは、企業の競合環境の分析や評価をサポートします。 産業構造は、企業がとり得る戦略に非常に大きな影響を与えます。

このモデルは、企業、最終製品やサービス、マーケット パートナーとの関係を表すために使用できます。 また、顧客が使用している製品およびサービス、納入業者が提供している製品およびサービス、(潜在的な) 競合他社が提供する代替製品およびサービスを表すこともできます。 このようにして、企業の競合状況の一部を表すことができます。

次の図に競争力モデルの例を示します。

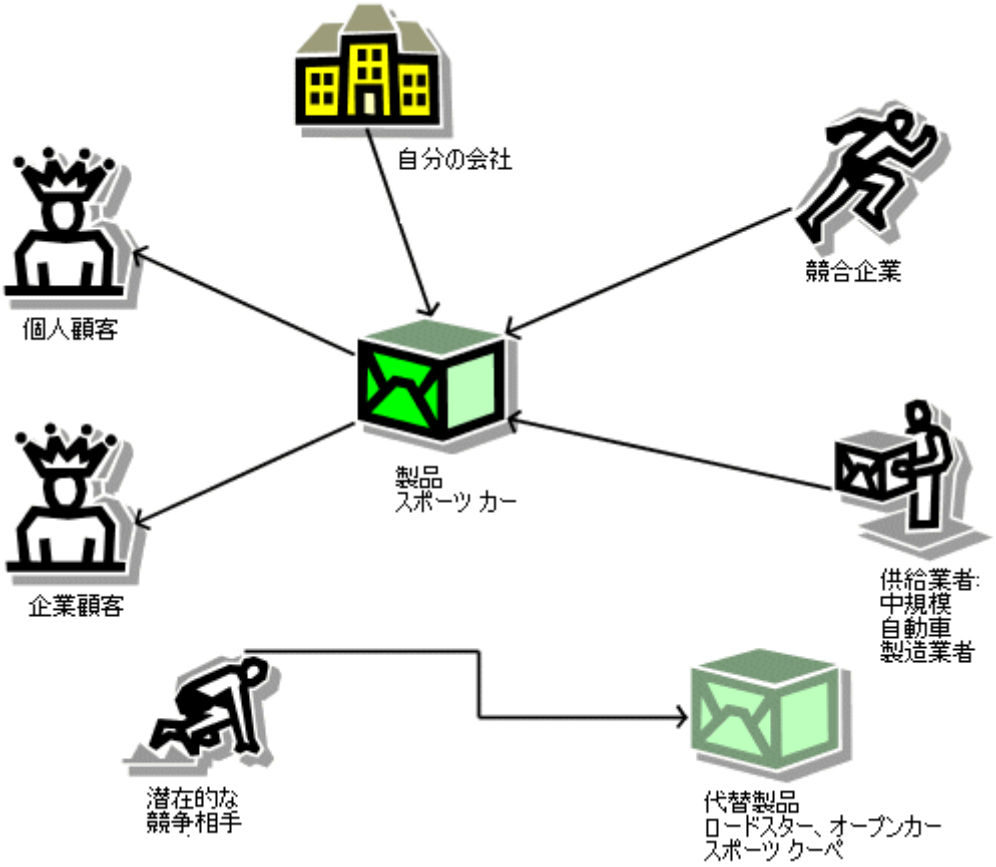


図 151: スポーツ カー市場の競争

5 ARIS の統一モデリング言語

5.1 はじめに

統一モデリング言語（UML）は、OMG (Object Management Group) ワーキング グループによって言語構造が標準化されているオブジェクト指向モデリング言語です。 UML は、OMT 法、Booch 法、および OOSE 法のオブジェクト指向アプローチに基づいています。

5.2 ARIS UML Designer - サポートされる UML 標準

ARIS UML Designer 9.x は、UML 標準 2.5 全体をサポートします。

UML 仕様は、http://www.omg.org/cgi-bin/doc?ptc/12-10-24.pdf の「Unified Modeling Language Specification, Version 2.5 FTF – Beta 1」を参照してください。

6 Object Modeling Technique (OMT)

6.1 はじめに

ARIS では、オブジェクト指向モデリングのためのもう 1 つのオプション、OMT (Object Modeling Technique) 表記法が提供されています (Rumbaugh, J.; Blaha, M.; Premerlani, W.; Eddy, F.; Lorensen, W. 共著: 『オブジェクト指向方法論 OMT モデル化と設計』トッパン、1993 年)。 また、ARIS では、OMT、Booch および OOSE メソッドを統合し、発展させた UML (統一モデリング言語) も提供されています (「ARIS の統一モデリング言語 『138 ページ 』」の章を参照)。 旧バージョンの ARIS との互換性を維持するため、OMT 法は現在もサポートされています。 ただし、OMT ではなく、統一モデリング言語を使用することをお勧めします。

以下で、OMT 法のコンポーネントについて説明します。 さらに、ARIS で使用できるモデリング手法についても説明します。 ここでの説明は、元の文献の詳細なモデリング手法の解説に取って代わるものではありません。 ARIS 内でモデリング手法をどのように扱うかの説明に主眼を置いています。

6.2 OMT 法のまとめ

OMT 法は、さまざまな観点からのシステム記述を提供するために設計されました。 この目的のために、次の方法が使われます。

- オブジェクト モデリング
- 動的モデリング
- 機能モデリング

これら 3 つのビューは互いに直交していますが、全く独立に表示できるわけではありません。

「オブジェクト モデリング」は、システムの静的かつ構造的なデータ関連のビューを表します。 ここでは、オブジェクトの構造、ほかのオブジェクトとの関係、およびそれらの属性を表現します。

「動的モデリング」は、システムの時間、振る舞い、制御に関連したビューを表します。 ここでは、操作の順序をイベントの順序を表すことによって記述します。

「機能モデリング」は、システムの一時的かつ機能的なビューを表します。 ここでは、値の変換を記述します。

これらのモデルには、ほかのモデルへのリファレンスが含まれます。 たとえば、オブジェクト モデルでは、動的モデルや機能モデルで使用するデータ構造を記述します。 機能モデルのプロセスは、オブジェクト モデルの操作に対応します。 動的モデルのステートチャート図は、オブジェクト モデルのクラス オブジェクトの振る舞い全体、または一部を記述します。

6.3 ARIS のオブジェクト モデリング手法

以下の章では、OMT 法を基にした構造の、ARIS における表現方法、使用法、および相互関係を示します。 OMT で定義されているモデリング構造 (たとえば、クラス、プロセス、状態など) は、ARIS 内のほかのモデリング構造 (イベント、ファンクション、エンティティ タイプなど) とは重複しません。つまり、これらは OMT モデルの中でのみ使用できます。 したがって、OMT は「独立した」手法として認識する必要があります。

6.3.1 OMT オブジェクト モデル

インスタンスの表現

オブジェクト指向モデリングでは、一般にオブジェクトはタイプ レベル (つまり、クラス レベル) で記述する必要があります。 ただし、個々のインスタンスをモデル化することも推奨します。 ARIS で提供されるシンボルは、角の丸い青の長方形で表されます。

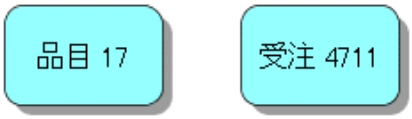


図 152: インスタンスの表現

クラスの表現

クラスは、モデルを作成する対象の基本的な領域構造を表します。 ARIS では、クラスを 2 本の水平線付きの青い長方形で表します。

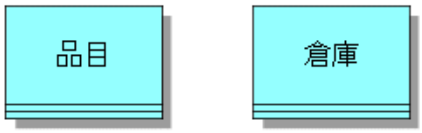


図 153: クラスの表現

クラスへのインスタンスの割り当て

インスタンスを図示すると、関連するクラスへの割り当てを示すことができます。 対応する接続線は、下に図示されるように [インスタンスである] タイプです。

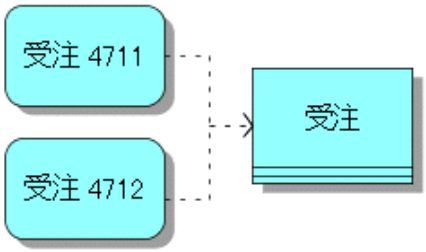


図 154: インスタンスとクラス間の接続線

クラスへの属性の割り当て

クラスの性質は、属性を使って定義します。 ARIS でモデル化する場合、属性は、[属性を持つ] タイプの接続線を使用し、別個のオブジェクト (したがって別個のシンボル) として対応するクラスに接続します (次の図を参照)。 この 2 つのオブジェクト タイプ (クラスと属性) を別々にしておくことにより、ARIS のナビゲーション機能とレポート作成機能を最大限に活用できます。

属性ごとにクラス属性 (その値がそのクラスのすべてのインスタンスに関連する) か、インスタンス属性かを指定できます。

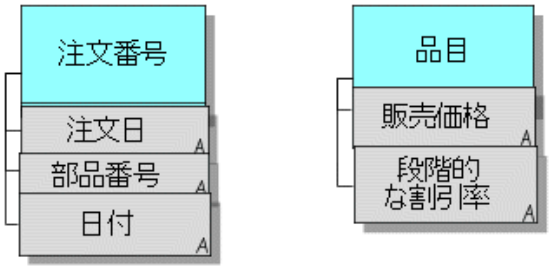


図 155: クラスへの属性の割り当て

クラスへの操作の割り当て

クラスに機能を割り当てるには、操作 (メソッド) を定義します。 ここでも別個のオブジェクト タイプを使用でき、これを [操作を持つ] タイプの接続線を使用してクラスに関連付けます (次の図を参照)。

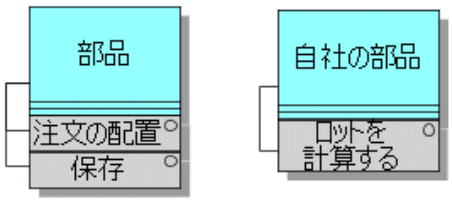


図 156: クラスへの操作の割り当て

インスタンス間の関連

個々のインスタンスは、互いに関連付けることができます。 ARIS では、これらの関連を方向性のない [結合される] タイプの接続線で表します。

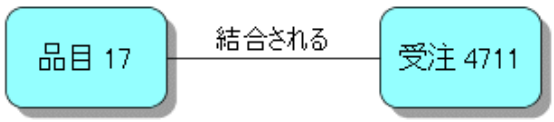


図 157: インスタンス間の関連

クラス間の関連

リンク (関連) は、クラス間にも作成できます。 これは、ERM (実体関連モデル) で頻繁に使用する関連です。 この関連は、ERM のシンボル (黄色い菱形) で表します。同様にして、n 項関係を表すことができます (以下を参照)。 接続線をクラス シンボルから菱形のシンボルまで描き、複雑度は接続線の [多重度] 属性を使用して割り当てます。 多重度として次のエントリを指定できます。多重度には、次の 4 種類があり、それに対応する図形表現を持つものもあります。

- 1
- c
- cn
- n

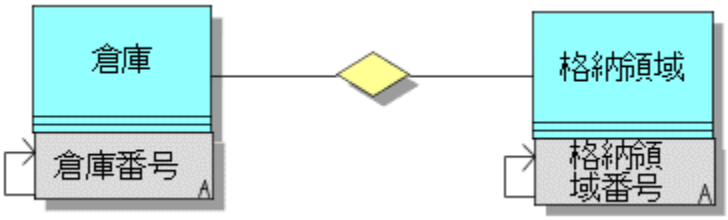


図 158: クラス間の関連

クラス間の n 項関連

3 (または n) 個のクラス間の関連は、3 番目またはこれ以上のクラスを、関連を定義する菱形に接続して表します。

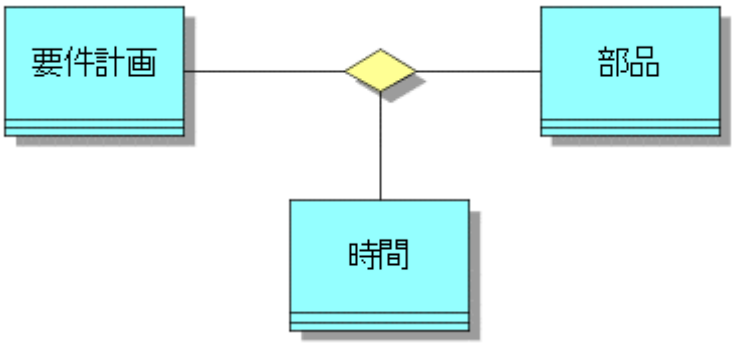


図 159: 3 個のクラス間の関係

クラスとしての関連のモデリング

関連を独立したオブジェクトと見なし、クラスとして解釈することができます。 これは、菱形シンボルからクラス シンボルへの有向接続線で表します。クラス シンボルでは、すべての属性と操作を列挙します (次の図を参照)。 この「再解釈された」クラスをほかのクラスに関連付けることももちろん可能です。

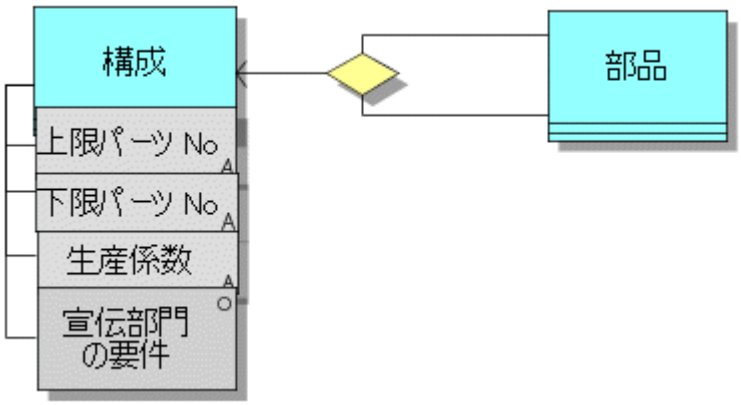


図 160: クラスとしての関連のモデリング

限定子付き関連の表現

限定子付き関連によって、通常の関連に資格情報が付加されます。 この情報は、関連の多重度を低減する付加属性です。この方法によって、関連の m 側のオブジェクトが区別されるため、1:m 関連や n:m 関連において有用です。

限定する関連は、接続線に限定情報を付加することで表します。 この目的のために、[限定子] 属性が提供されています。すべての属性と同様に、これはグラフィックでも表示できます。

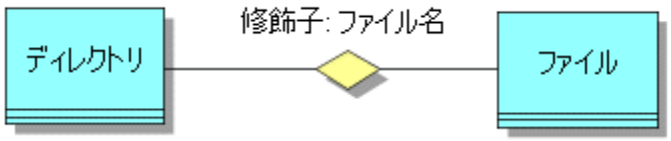


図 161: 限定子付き関連の表現

関連の順序の表現

関連の n 側のオブジェクトに順序がある場合は、これを明示的に図に表すことができます。 この目的のため、[クラス] および [関連] シンボル間の接続線には別個の属性を使用します。

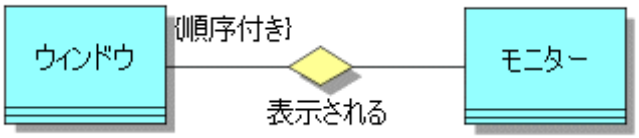


図 162: 関連の順序の表現

クラス間の集約

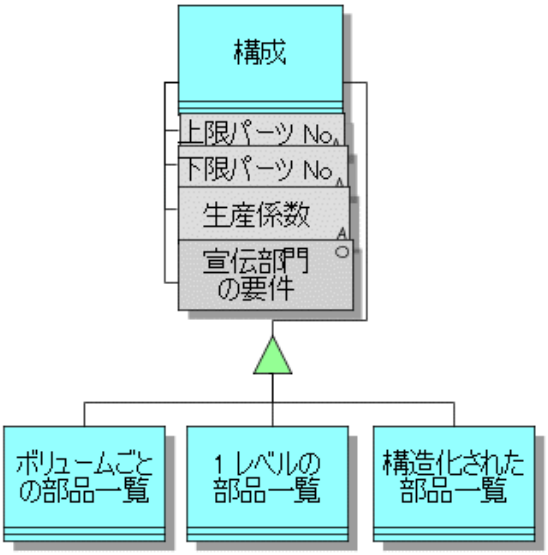
集約は部分と全体の関係を表し、関連の特別な形式と見なすことができます。 この関係は、クラス間の有向接続線 ([集約する] 接続線タイプ) としてモデル化されます。 図では、白い菱形のシンボルで「全体」(コンポーネント グループ) を表すクラスを示します。

図 163: クラス間の集約

汎化と継承

オブジェクト指向モデリングの基本的な構造の 1 つは、クラス間の階層の定義です。この階層内で、下位クラスは上位クラスの属性および操作を継承できます。 ARIS では、この目的に使用できる別個のオブジェクト タイプがあります (緑の三角形)。所属するクラスに関連しています (次の図を参照)。 多重継承も表すことができます。

汎化演算子に、汎化/特化に使用する側面を表したり、特化が分離されるか分離されないかを表したりする属性を割り当てることができます。



クラス、属性、および関連に対する制約

制約は、OMT オブジェクト モデルのクラス、属性、および関係間の機能的な関係を表します。 ARIS では、別個のオブジェクト タイプ (点) が属性の制約に定義されています。 次の図に、ウィンドウの縦横比が 0.7 から 1.7 の値をとることを表す例を示します。

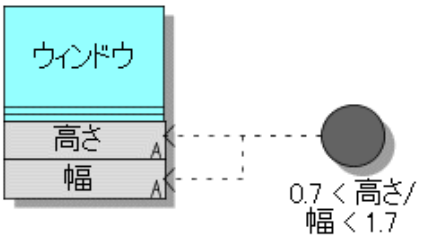


図 164: 属性の制約の表現

関連を参照する制約を定義することもできます。 次の図の例では、委員会の役員を構成するメンバーの集合は、必然的に委員会のすべてのメンバーのサブセットを表すことを示します。 この制約を関係シンボル間の有向接続線で図示します。

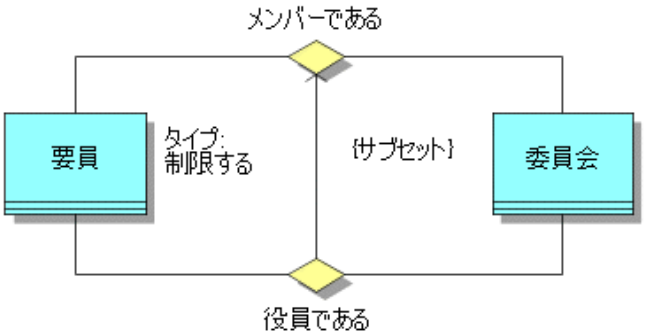


図 165: 関連の制約の表現

OMT オブジェクト モデルの例

次の図に、主なモデリング要素を含む OMT オブジェクト モデルの典型例を示します。

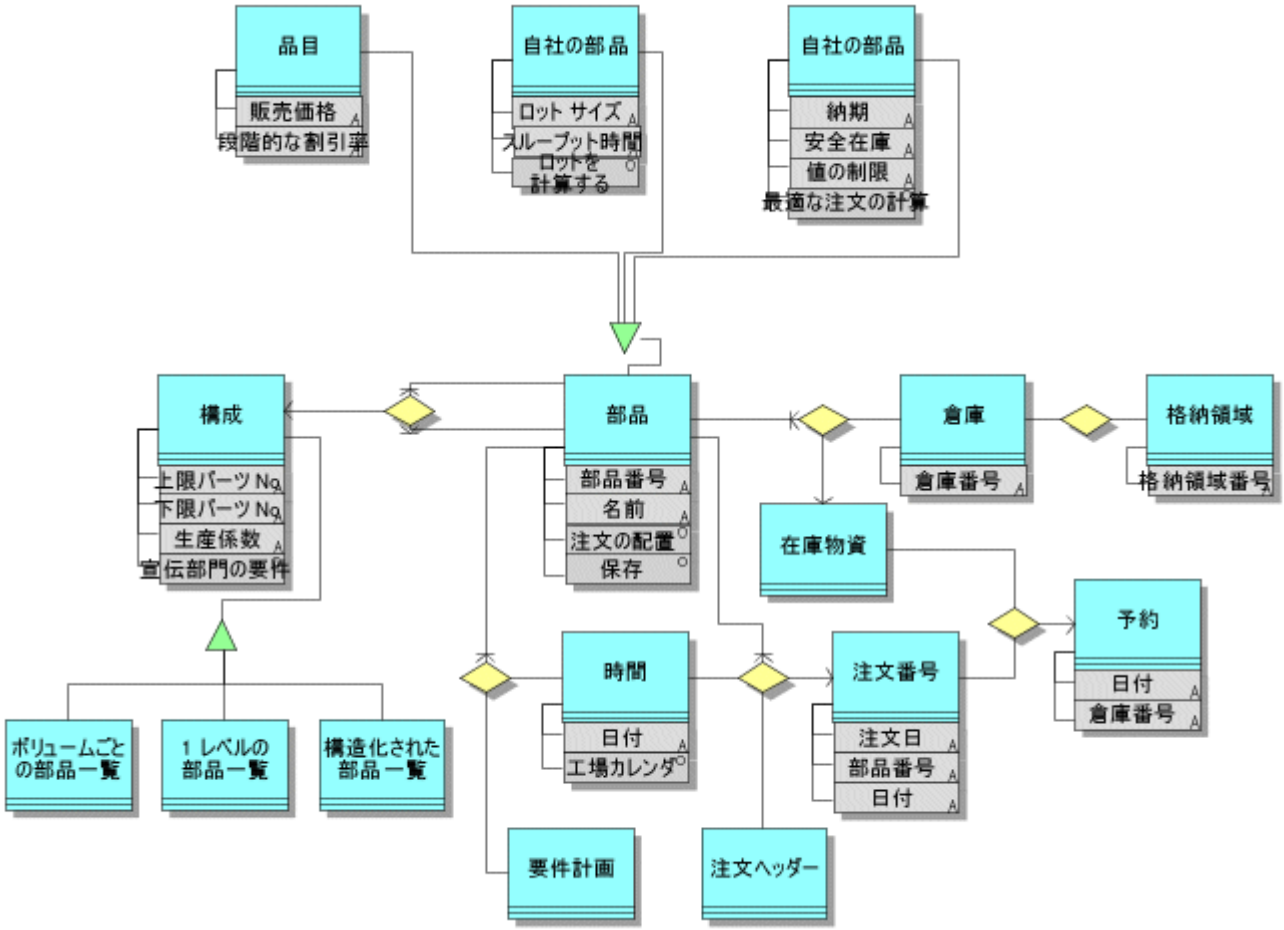


図 166: OMT オブジェクト モデルの例

6.3.2 OMT 動的モデル

OMT 動的モデルは、一連の遷移図です。通常 1 つの遷移図には、1 つのクラスの動作を定義します。 状態は、イベントを表す有向接続線によって接続されます。

初期状態、終了状態、遷移の表現

ARIS には、初期状態、終了状態、遷移を区別する 3 つのシンボルがあります。

図 167: 初期状態、終了状態、遷移の表現

状態間の遷移

状態間の遷移は、イベントによって起動されます。 2 つの状態間の接続線は、[移行する] タイプになります。

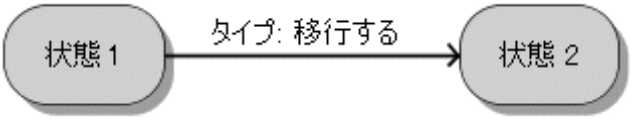


図 168: 状態間の遷移の表現

状態および遷移の両方に対して、さらに情報を付加することができます。 状態の開始アクション、終了アクション、内部アクションを記述するには、[実行/アクション]、[入状/アクション]、[退状/アクション]、[イベント/アクション]の各属性を使用します。 2 つの状態間の接続線に、より詳細な遷移条件を記述することもできます。

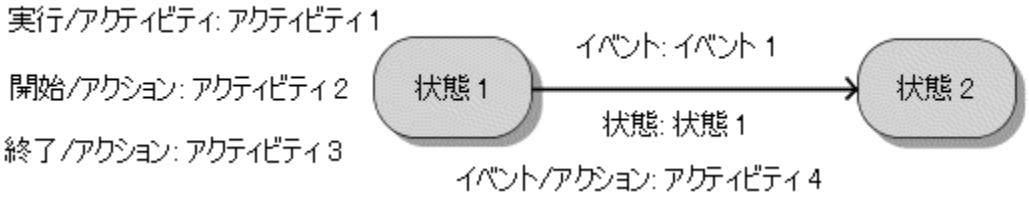


図 169: 遷移の追加情報の表現

OMT 動的モデルの例

次の図に OMT 動的モデルの典型例を示します。

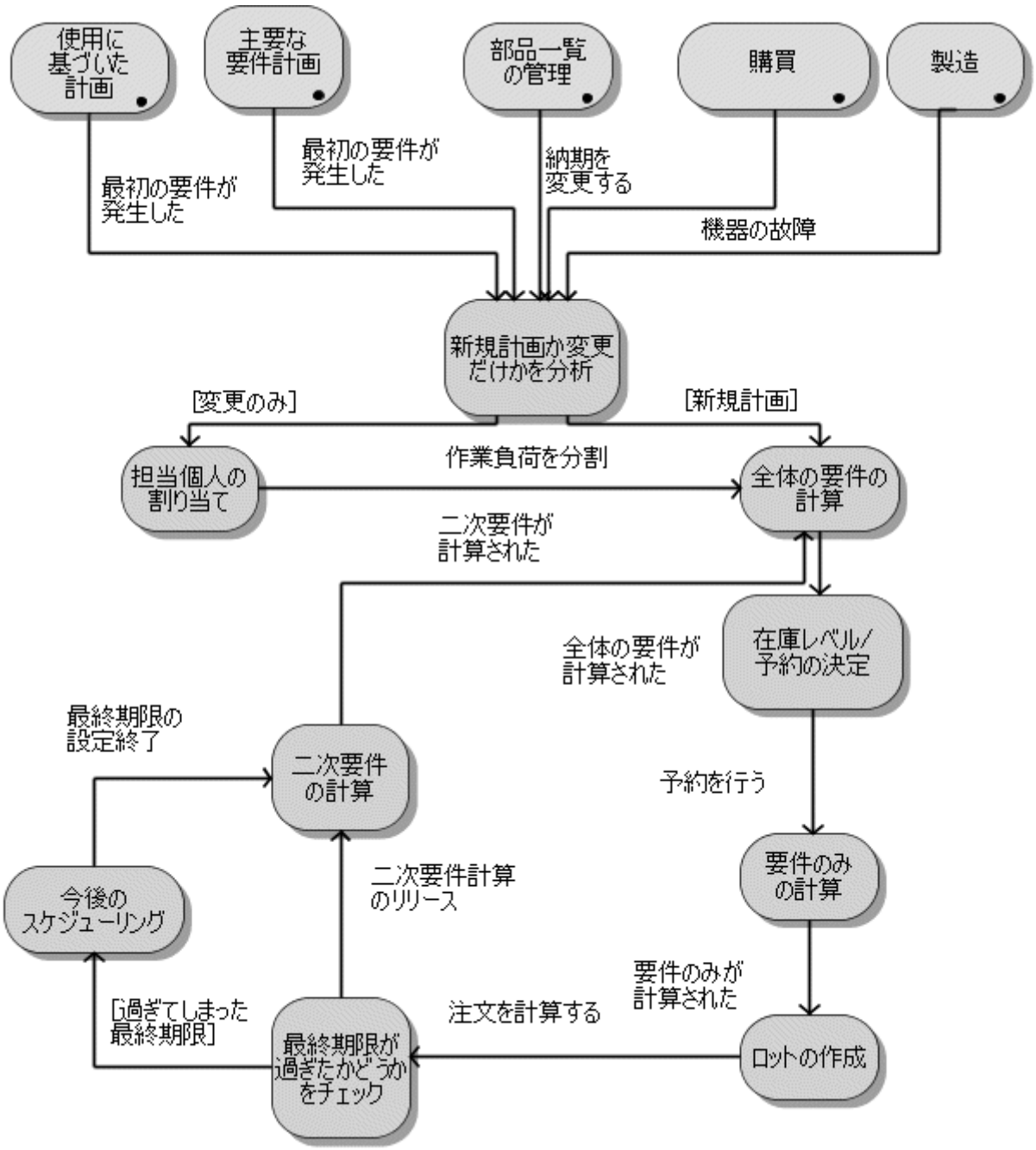


図 170: OMT 動的モデルの例

6.3.3 OMT 機能モデル

データ フロー図を使って、OMT 機能モデルでは、計算中に入力データから出力データが導き出される方法を表します。

データ ストアの表現

データ ストアは、受動的なデータの格納場所として使われます。 ARIS では、これを 2 本の水平線から成るシンボルで表します。

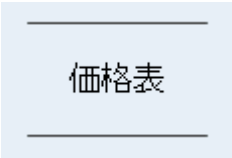


図 171: データ ストアの表現

プロセスの表現

プロセスはデータを変換します。ARIS では、これを黄色の楕円で表します。

図 172: プロセスの表現

アクターの表現

アクターは、データの値を生成、または消費することによって、データ フローを起動するオブジェクトです。 したがって、アクタは、グラフのソースまたはシンクと見なすことができます。アクタは正方形で表します。

図 173: アクターの表現

データ フローの表現

データ フローは、あるプロセスやオブジェクトの入力を別のプロセスやオブジェクトの入力に接続します。 これらは、オブジェクト間の [データ値] タイプのオブジェクトとしてモデル化されます。通常、データ フローにはデータを記述するラベルを付加します。

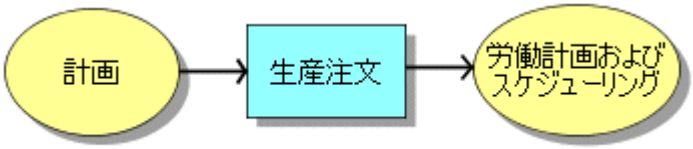


図 174: データ フローの表現

データ フローの分割

データの値が異なる場所に送られる場合は、データ フローを分割できます。 ARIS では、このような分割を表すために独自のシンボル（[接続線]）を使います。

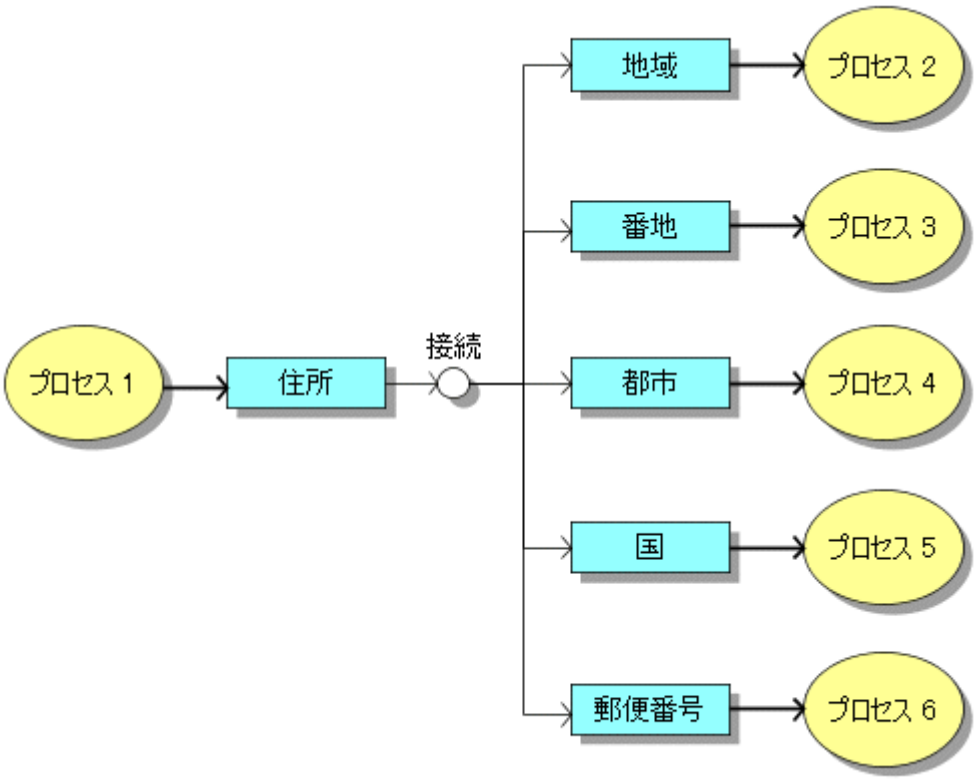


図 175: データ フロー分割の表現

OMT 機能モデルの例

次の図に OMT 機能モデルの典型例を示します。

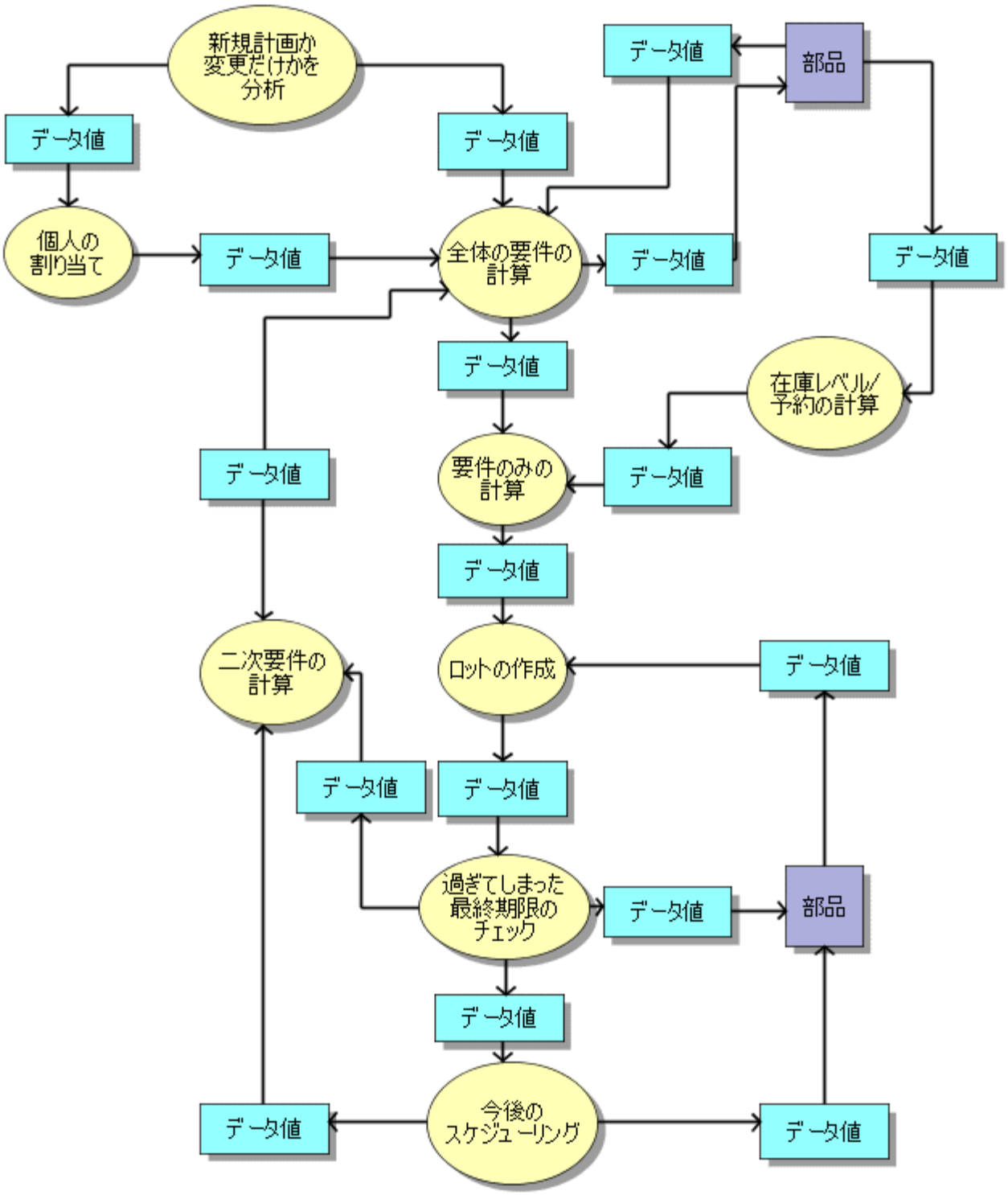


図 176: OMT 機能モデルの例

6.3.4 オブジェクトを階層的に配置する方法

- [クラス] オブジェクト タイプを、OMT 動的モデルを使用して階層的に配置することで、クラス内の遷移を記述することができます。
- [クラス] オブジェクト タイプを、OMT クラス記述モデルを使用して階層的に配置し、クラスに属性および操作を割り当てることができます。 このモデルで使用できる関係は、OMT オブジェクト モデルのサブセットを表します。
- [状態]、[最終状態]、および [初期状態] オブジェクト タイプを OMT 動的モデルで詳細化することで、さまざまな階層レベルで状態を記述できます。
- [プロセス] オブジェクト タイプを OMT 機能モデルで詳細化することで、さまざまな階層レベルでデータ フローを記述できます。
- [データ ストア] オブジェクト タイプは、OMT オブジェクト モデルを使用して階層的に配置できます。
- OMT 機能モデルの接続線タイプを、OMT オブジェクト モデルおよび OMT DataValue 分解モデルを使って詳しく表現し、データ フローの詳細を記述できます。

7 知識管理の手法

7.1 はじめに

知識管理の目的は、知識を体系的に管理することです。知識は企業のリソースであり、ますます重要度が高まっています。知識管理は、社内における効率的な知識処理のための戦略、プロセス、組織構造、および技術の、開発、監視、サポートおよび改良まで多岐にわたります。 これには、知識の獲得、準備、伝達、および活用に関するすべての活動が含まれます。 これらの知識管理活動は、一般に独立して発生するものではありません。主に、企業の業務プロセスや業務プロセスを計画する場合に発生します。 したがって、業務プロセス、知識処理、組織構造、情報システムなどの統合的なビューが必要になります。

これらのほとんどの側面は、EPC、組織図、ファンクション割当図、eERM など、確立されている ARIS メソッドを使用して表現できます。 しかし、知識処理を正確に表現、分析、および改良するには、関連する知識カテゴリ内容の識別および構造化、組織における知識の分布に関する説明、業務プロセスにおける知識の作成および活用のモデル化を行うことができる、ほかの表現手法が必要になります。

このために、2 つの新しいオブジェクト タイプ [知識カテゴリ] と [有形知識]、および 2 つの新しいモデル タイプ [知識構造図] と [知識マップ] が追加されました。 さらに、業務プロセスを表現するために使用される既存のモデル タイプ (EPC、PCD、オフィス プロセスなど) が拡張され、知識の作成や活用を扱える構造が追加されました。 新しいオブジェクト タイプとモデル タイプは、要件定義の主要なモデル タイプ (eERM、組織図、ファンクション ツリーなど) に手法として統合され、統合的な視点を提供しています。 これによって、たとえば、業務プロセス最適化プロジェクトのモデルを、知識処理の分析や改善に利用することが可能になります。

知識構造図は、要求定義のデータ ビューに配置されます。 知識マップは、業務プロセス モデリングの拡張モデル タイプと同様に、要求定義のプロセス ビューに属します。

7.2 知識処理モデリングのためのオブジェクト タイプ

7.2.1 知識カテゴリ

[知識カテゴリ] オブジェクト タイプは、特定の知識を参照する内容を持つオブジェクトを表します (図「知識構造図 『155 ページ 』」を参照) 。これは、思考を表す楕円形の吹き出しで表現します。 知識カテゴリには、プロジェクト管理に関する知識、特定の業種に関する知識、特定の技術に関する知識、顧客および競合他社に関する知識などが含まれます。これらのカテゴリは、企業の既存の知識、または必要な知識の分類に役立ちます。

特定の知識カテゴリに割り当てられる知識には、暗黙の知識、つまり、従業員やグループのスキルという形で使用できるため、完全には文書化できない知識と、説明または技術図面という形で文書化できる明示的な知識があります。 知識カテゴリには通常これらすべてが含まれます。 たとえば、プロジェクト管理に関する知識には、プロジェクト管理者の経験とプロジェクト管理マニュアルて提供される情報を含めることができます。

知識カテゴリをより詳細に説明するには、[説明]、[注釈]、[ソース] などの一般的な属性に加えて、次の特殊な属性が役立ちます。

属性名	値の範囲	説明 / 例
更新頻度	列挙タイプ: 時間ごと、毎日、毎週、毎月、毎年、ほとんどない、決してしない	更新頻度は、関連するカテゴリの知識を、最新の状態に更新する頻度を説明します。 たとえば、三角法に関する基本知識は、更新する必要はほとんどないか、まったくありませんが、特定の株価に関する知識は毎日または場合によっては時間ごとに更新する必要があります。
意味	パーセンテージ: 0 ～ 100	企業の知識カテゴリの重要性は、0% (まったく重要でない) から 100% (非常に重要である) の範囲で表します。
知識のカバー率	パーセンテージ: 0 ～ 100	企業における関連知識の現在のカバー率は、0% (まったくカバーしていない) から 100% (最大限にカバーしている) の範囲で表します。 知識カテゴリのカバー率を特定の組織ユニットまたは要員別に表すには、[自由に使える] 接続線タイプの該当する属性を使用して知識マップに指定できます。
知識の優位性	パーセンテージ: 0 ～ 100	競合他社に対する知識面での自社の相対的な優位性は、0% (競合他社が自社よりも圧倒的に優位である) から 100% (自社が競合他社よりも圧倒的に優位である) の範囲で表します。
知識の活用度	パーセンテージ: 0 ～ 100	特定の知識カテゴリの活用度は、0% (関連知識がまったく活用されていない) から 100% (関連知識が最大限に活用されている) の範囲で表します。
目標カバー率	パーセンテージ: 0 ～ 100	関連知識の目標カバー率は、0% (まったくカバーしていない) から 100% (最大限にカバーしている) の範囲で表します。
将来の重要性	列挙タイプ: 大きく減少、減少、一定、増加、大きく増加	[将来の重要性] は、企業における重要性の変化に対する知識カテゴリの今後の傾向を表します。
構造 変更速度	パーセンテージ: 0 ～ 100	構造変更速度は、関連する知識を獲得するための手法を、どの程度迅速に変更する必要があるかを表す値です (0%: まったく変更する必要がない 100%: 最大の変更速度)。

これらの属性を使用して、企業における関連する [知識カテゴリ] の重要度を評価します。 つまり、これらを、企業の知識管理を改善するための重要な、または緊急に行う必要がある処置を識別するための基礎として利用できます。 これらの値をグラフィックとして表示すると役に立ちます。 最も簡単な方法で値を図示するには、属性ウィンドウの数値をコピーし、表計算プログラムに貼り付けて、希望の図を作成します。 たとえば、対象中の [知識カテゴリ] について、棒グラフで [カバー率] の現在値と希望値を比較できます。

7.2.2 有形知識

暗黙的および明示的な知識を扱う [知識カテゴリ] オブジェクト タイプとは対照的に、[有形知識] オブジェクト タイプは、明示的に文書化されているか、原理的には文書化できる知識カテゴリのみが対象です。 マニュアルに文書化されたソフトウェア使用法に関する知識は、その一例です。 知識を知識カテゴリに割り当てる場合、一般的な知識カテゴリと有形知識を区別すると、知識処理をサポートする情報システムの可能性と限界を認識するうえで役立ちます。電子的に保存、伝達、および処理できるのは有形知識だけです。

[有形知識] オブジェクト タイプは、四角形の吹き出しで表します。「有形知識」には、「知識カテゴリ『153 ページ 』」オブジェクト タイプと同じ種類の特殊な属性タイプが含まれます。

7.3 知識処理モデリングのためのモデル タイプ

7.3.1 知識構造図

知識構造図を使用すると、知識カテゴリを内容に応じて分割できます。 次の図に、これを表す例を示します。 知識カテゴリには、ほかの知識カテゴリや有形知識が含まれることがあります。 有形知識も、いくつかの有形知識のサブカテゴリに分割できます。 ただし、一般的な知識カテゴリを含めることはできません。

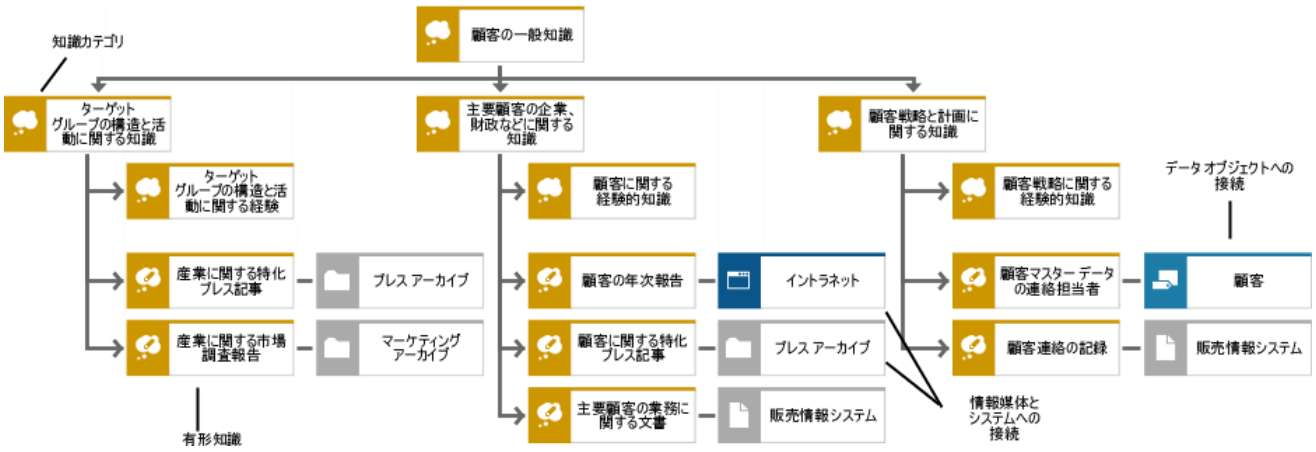


図 177: 知識構造図

有形知識の場合、知識を保存する情報媒体を知識構造図に示すことができます。また、知識の文書化に使用するデータモデルの情報オブジェクトやオブジェクト指向システムのクラスを表すことができます。 最後に、知識を管理するために使われるアプリケーション システムのタイプまたはクラスもモデル化できます。

7.3.2 知識マップ

知識マップは、知識カテゴリの組織内での分布を表します。 組織ビューの各種オブジェクト タイプ (たとえば、[組織ユニット]、[役職]、[要員]、[場所]、[グループ] など) を、[自由に使える] 接続線を使用して知識カテゴリに接続できます。 さらに、特定の要員や組織ユニットが特定のカテゴリの知識を所有しているという事実のほかに、カバー率も表すことができます。 [自由に使える] 接続線には、[カバー率] 属性が含まれます。これは、関連する組織ユニットに関して、選択されたカテゴリの知識のカバー率をパーセンテージで表したものです。 100% は最大のカバー率を表し、0% は対象のカテゴリの知識がまったくないことを表します。 これは、前述の接続線が存在しないのと同じです。 この定量的な評価のほかに、グラフ形式での表示が可能な定性的な評価も実行できます。 これが、知識の [カバー率の質] 接続線属性の目的です。これは、「低い」、「普通」、「高い」、および「最高」の値をとります。 この情報は、次の図に示すような接続線のシンボルを使って表します。 [知識のカバー率] 属性と [カバー率の質] の値には直接の関係はありません。 両方の属性を使用する場合、カバー率 25% までに対しては「低い」、カバー率が 26 ～ 50% に対しては「平均」、カバー率が 51 ～ 75% に対しては「高い」、カバー率が 76 ～ 100% に対しては「最高」を使用することをお勧めします。

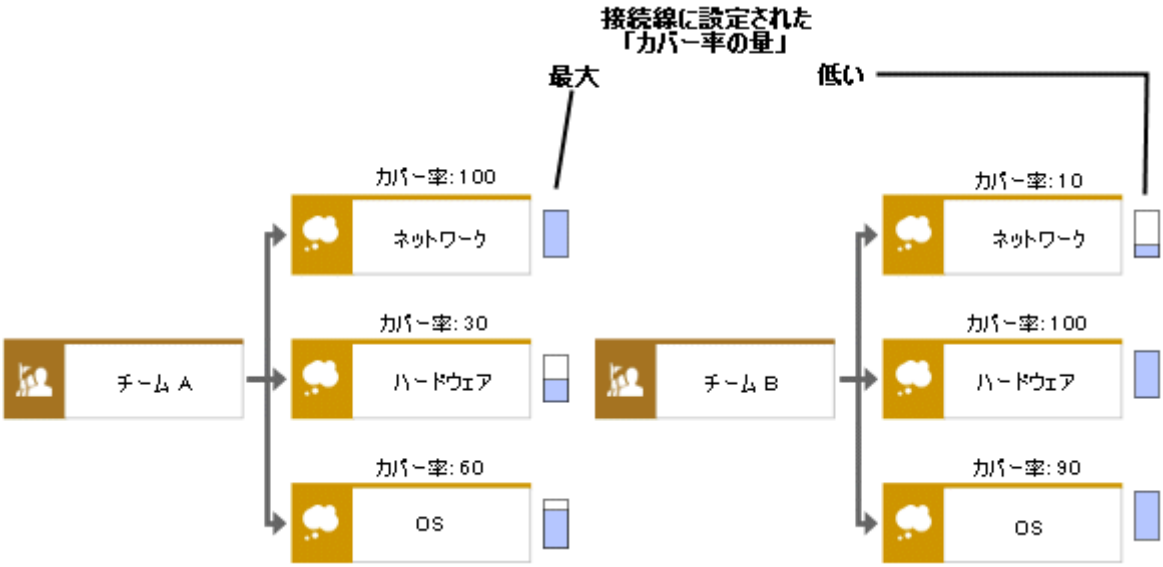


図 178: 知識マップ - 組織ユニットに関連させた場合

次の図の知識マップは、組織ユニットに注目しています。つまり、すべての知識カテゴリは各組織ユニットに指定されています。検討する中心オブジェクトとして知識カテゴリを使用して、関連するすべての組織ユニットをそれに追加することもできます。ARIS のナビゲーション オプション ([プロパティ - オブジェクト] ダイアログ ボックスの [関係] タブ) を使用すると、このいずれの場合にも、1 つの組織ユニットまたは知識カテゴリのほかの既存の関係を簡単に見つけることができます。 知識マップは、マトリクス形式で表されることがよくあります。 次の図に示すように、同じ知識カテゴリの複数のオカレンスを列形式に配置することによりマトリクス表現が可能になります。 この例では、知識カテゴリの名前だけが、表のヘッダーのように一番上に表示されています。 その他のオカレンスについては、属性配置ファンクションによって名前が削除されています。 この図では、カバー率の違いが別の方法で表現されています。つまり、知識カテゴリが異なるサイズで表現されています。

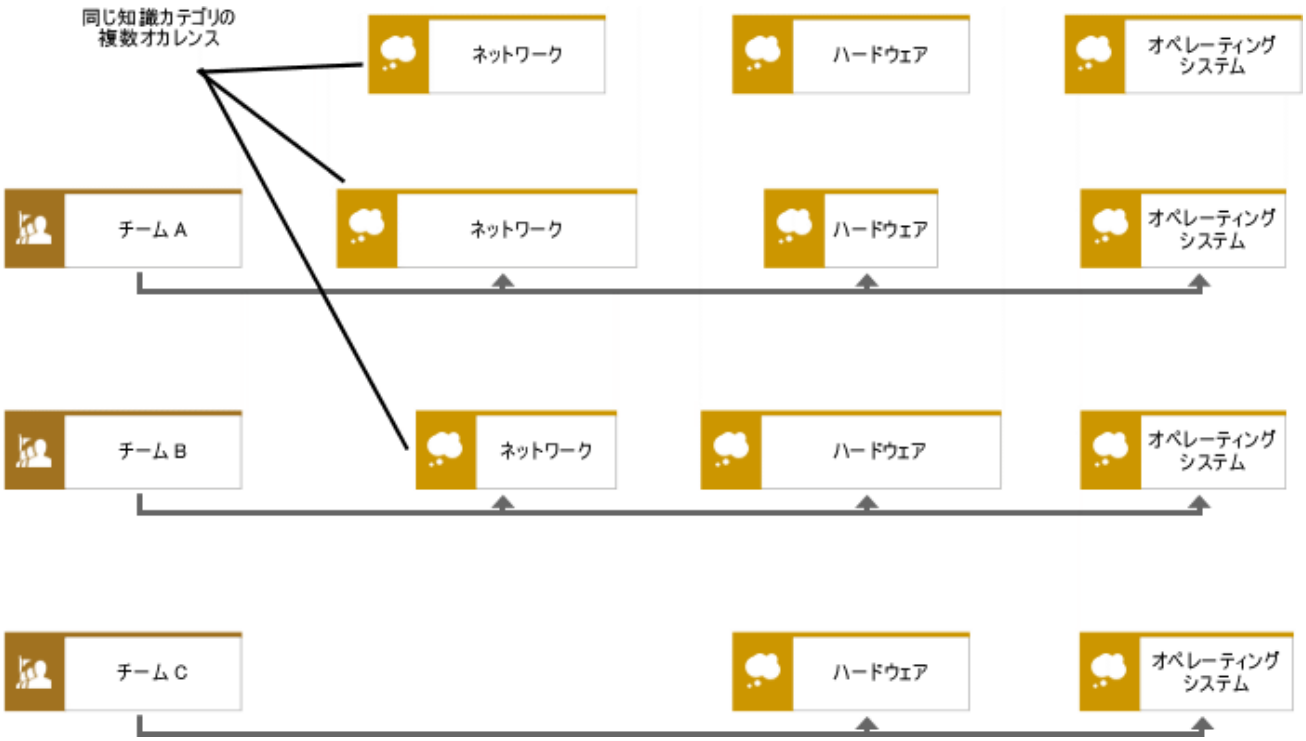


図 179: 知識マップ - マトリクス表現

7.3.3 業務プロセスにおける知識処理の表現

企業の業務プロセスにおける知識の活用と作成をモデル化するには、業務プロセスの表示に用いることができるモデル タイプ (EPC、EPC (マテリアル フロー付き)、オフィス プロセス、製造プロセス、PCD、PCD (マテリアル フロー付き)) を使用します。[知識カテゴリ] および [有形知識] は、現在これらのモデル タイプで利用可能です。 したがって、ファンクションを実行するために必要な知識の種類 (一般または有形) を指定し、実行時にどのような知識が作成または記録されるかを表現できます。この種の表現によって、知識処理を含む形で業務プロセスを検討できます。 たとえば、必要な知識の不足を明らかにできます。また、ファンクションの実行に必要な資格プロファイルを決定することもできます。

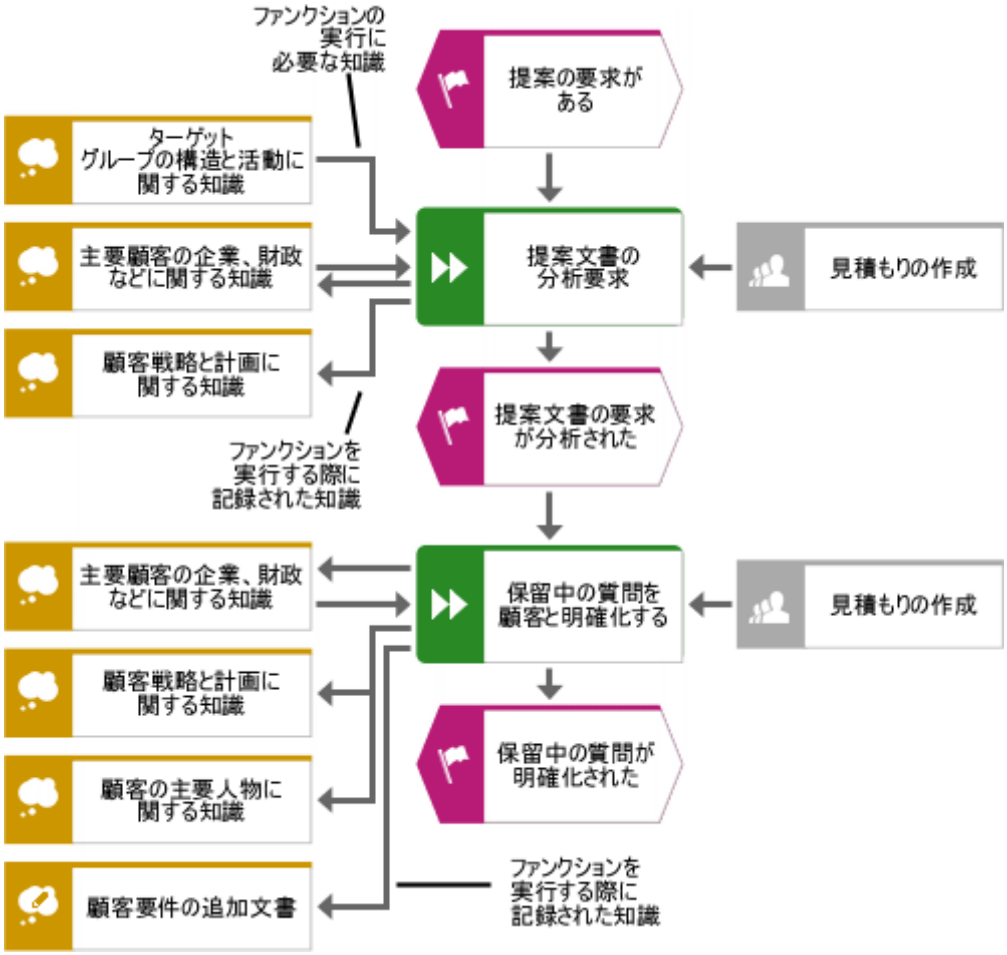


図 180: EPC における知識処理

8 ユース ケース

この章は、ビジネス管理の特定の問題に対する ARIS による適切なサポートを見つけるためのガイドとなります。 このため、章の各節はユース ケースのシナリオごとに構成されています。

各ユース ケース シナリオに関して、まずシナリオの意味と各シナリオにおける通常の活動 (アクティビティ) についての簡単な説明があります。 そのあとで、シナリオに含まれる典型的な課題 (タスク) を示します。 ARIS を使用したタスクの実行方法をタスク別に説明します。

次の表は、各ユース ケースの概要と、それぞれにおいて使用されるモデル タイプを示したものです。

シナリオ	シナリオのタスク	モデル タイプ
一般的な 文書化	業務目標の文書化 企業の付加価値の文書化 組織構造の文書化 企業のファンクションの文書化 プロセスの文書化 プロセス ウェアハウジング	目標図 付加価値連鎖図 組織図 ファンクション ツリー オフィス プロセス 製造プロセス EPC PCD
データベース管理/データ ウェアハウジング『162 ページ 』	データの構築/データベースの設計 データベース管理/アクセス管理	ERM SAP SERM SeDaM IEF データ モデル リレーション図 テーブル図 クラス図 クラス記述図
PC ハードウェアとネットワークの管理『162 ページ 』	IT インフラストラクチャの要件の定義 IT インフラストラクチャの文書化 アクセス権	ネットワーク トポロジ ネットワーク図
プロセス原価管理『163 ページ 』	プロセス構造および組織構造の説明 原価センタ分析 プロセス計算	EPC PCD 組織図 原価作用因図 原価要素図
品質管理『164 ページ 』	QM 関連文書の作成	製品ツリー

シナリオ	シナリオのタスク	モデル タイプ
	認証手続き 認証文書	製品選択マトリクス EPC PCD オフィス プロセス図 製造プロセス図 付加価値連鎖図 構造モデル 組織図
再編成 措置	プロジェクト文書 再編成の 実行	付加価値連鎖図 EPC 組織図 製品モデル 製品/サービス モデル 目標図 PCD
SAP R/3 導入『165 ページ 』	分析フェーズ、仕様フェーズ (プロジェクトの準備) 設計フェーズ (ビジネスの設計)、3 つの利用できるユース ケース	EPC 組織図
ソフトウェアの開発と導入『166 ページ 』	プロジェクト文書 アプリケーション システムおよびモジュールの指定 IT プロセスの記述 システム インターフェイスの開発	付加価値連鎖図 組織図 EPC ユース ケース図 アプリケーション システム タイプ図 プログラム フロー図 画面図
知識管理『167 ページ 』	知識マップまたはイエロー ページ 知識の分類 業務プロセスにおける知識処理	知識マップ 知識構造図 EPC PCD オフィス プロセス 製造プロセス

シナリオ	シナリオのタスク	モデル タイプ
		ファンクション割当図
ワークフロー管理『168 ページ 』	ワークフロー管理システムのプロセス カスタマイズ	権限図 EPC ファンクション割当図 アプリケーション システム図 アプリケーション システム タイプ図

8.1 一般的な文書化

各企業の特徴であるプロセス、構造、データなどは、トレーニング、プレゼンテーション、または評価などの目的に応じて、適切な形式で文書化できます。 以下に、企業における文書化でもっとも重要なタスクについて簡単に説明します。

タスク: 業務目標の文書化

ARIS サポート: 目標図を使用して、業務目標および対応する成功要因を階層的に配置できます。

タスク: 企業の付加価値の文書化

企業に付加価値を与えるアクティビティに必要なファンクションを明確にし、それを企業の意思決定の基本とします。

ARIS サポート: 付加価値アクティビティに必要な企業のファンクションは、付加価値連鎖図を使用して表します。 このモデルには、連続した一連のファンクションと、上位および下位のファンクションの両方が示されます。

タスク: 組織構造の文書化

ARIS サポート: 企業の構造は、各組織ユニットの階層と関係を示す組織図に記述できます。

タスク: 企業のファンクションの文書化

ARIS サポート: ファンクション ツリーを使用して、企業における個々のファンクションの概要を表すことができます。 これらのファンクションは、オブジェクト指向、プロセス指向、処理指向のいずれかに分類できます。

タスク: プロセスの文書化

ARIS サポート: オフィス プロセス図または製造プロセス図では、業種またはプロセス タイプに従って、メソッドの知識を必要とせずに業務プロセスを記録することができます。 メソッドの知識がある場合に、SAP アプリケーション、シミュレーション、ワークフローなどにおいてプロセス モデルをさらに活用するには、EPC によるモデリングとプロセス連鎖図 (PCD) による弱点分析をお勧めします。

タスク: プロセス ウェアハウジング

プロセス ウェアハウジングとは、業務プロセスの知識をリポジトリにおいて体系的に記録、保存、維持することを指します。

ARIS サポート: 分散ユニットにおけるプロセス知識のモデリングでは、作業を行う従業員には一般にメソッドのノウハウはないので、オフィス プロセス図および製造プロセス図の使用をお勧めします。 中央のモデル リポジトリにおける維持と管理には、シミュレーションやプロセス原価管理などの要求の厳しい評価でも使用できるように、モデルを EPC に変換し、文書、画像、動画などで補足すると便利です。

8.2 データベース管理 / データ ウェアハウジング

企業のデータをデータベースに保存することによって、冗長なデータ保存を減らせるとともに、プログラムに依存せずに社内全体で使用されるデータにアクセスできるようになります。 データ ウェアハウジングによって、基盤となるデータの品質、整合性、一貫性が保証されます。 一般に、「データ ウェアハウス (Data Warehouse)」という用語は、業務用の IT システムから独立して、あらゆる形式の管理支援システムで使用される全社的なデータ基盤を指します。 その特徴は、業務および意思決定支援のデータやシステムから完全に分離していることにあります。 データ ウェアハウスの概念では、意思決定関連のプロセスにおいて評価と分析を行うための大量のデータを、効率的に提供し処理することに重点が置かれています。

タスク: データの構築 / データベースの設計

データベースの構造を、使用するデータ モデルに応じて決定します。

ARIS サポート: もっとも広く使われているデータ モデリングの手法は、実体関連モデル (ERM) です。これは、リレーショナル データベースのインプリメンテーションの基盤となるものです。

製品または企業固有、およびその両方向けの ERM バリエーションには、SAP 社の情報モデリングに使用される SAP-ERM、BASF 社の表記法である SeDaM (semantic data model: 意味論的データ モデル)、Texas Instruments Inc. の CASE Tool のデータ モデリングに使用される IEF (information engineering facility) データ モデルなどがあります。

論理データ構造を具体的に説明するには、リレーション図を使用し、これを ERM で設計したデータ構造に追加します。

データベース システムのテーブルおよびフィールドは、テーブル図を使用して説明します。

オブジェクト指向のデータベース システムは、UML (Unified Modeling Language) または OMT (Object Modeling Technique) を使って設計できます。 UML では、クラス図によって静的なデータ関係を表示できます。これに対し、クラス記述図では属性、オブジェクト、インターフェイスなどの追加表現が可能です。

タスク: データベース管理 / アクセス管理

データベース システムにユーザーとシステム管理者を割り当てます。

ARIS サポート: アクセス図をリレーションおよびシステム コンポーネントとともに使用して、データベース システムに対する組織ユニット、役職、および要員のアクセス権を決定できます。

8.3 PC ハードウェアとネットワークの管理

ネットワーク管理とは、あるコンピューター ネットワーク内で通信を行うためのあらゆる（分散）リソース（データ ネットワーク、プロセッサ、データ、アプリケーション）の管理、監視、調整を意味します。

タスク: IT インフラストラクチャの要件の定義

既存の組織構造に基づいて、これを効率よくサポートできる通信および情報システム インフラストラクチャが導き出されます。

ARIS サポート: 情報システムの構造の要件は、ネットワーク トポロジのモデル タイプを使って表すことができます。 アプリケーション システム、ネットワーク タイプ、ハードウェア コンポーネントの表現は、個々に識別可能な実例（たとえば「在庫番号 3423 の PC」など）を示すものではなく、むしろ同じ技術に基づいて類型化したものと言えます。

タスク: IT インフラストラクチャの文書化

IT インフラストラクチャの現在の実装または実装計画を、具体的なハードウェア コンポーネント、ネットワーク、アプリケーション システムを使用して表します。

ARIS サポート: IT インフラストラクチャは、ネットワーク トポロジを具現化したものとして、ネットワーク図で図示することができます。

タスク: アクセス権

アプリケーションおよびユーザーがアクセス可能なデータおよびそのアクセス方法を示します。

ARIS サポート: アクセス図を使用すると、どのアプリケーションまたはアプリケーション モジュールが、データベースおよび情報媒体にどのような種類のアクセス権（書き込み/読み取り/変更）を許可し、データが入力または出力のどちらで動作するかを説明できます。 さらに、特定のユーザーまたはユーザー グループが、アプリケーションまたはアプリケーション モジュールに対してどのようなユーザー権限およびビューを持つかを表すこともできます。

8.4 プロセス原価管理

コスト会計は、製品やサービスの商業的な提供から発生するコストを記録し、その発生源を明らかにすることによって、スケジュール作成の基盤と制御の手段を提供するものです。 原価構造の変化、特に諸経費の増加により、従来のコスト会計手法にかわってプロセス原価管理が使用されるようになってきています。 プロセス原価管理では、原価センタにわたってプロセスの原価を決定します。 プロセス原価管理の主な利点としては、予算作成、間接費のパフォーマンスにおける原価の透明度、価格設定、「make-or-buy」の決定の支援などがあります。

以下に説明する ARIS のサポートの一部は、ARIS Optimizer でのみ提供されます。

タスク: プロセス構造および組織構造の説明

プロセス原価管理が適用されるプロセスを指定し、原価センタについて説明します。

ARIS サポート: プロセスは、EPC や PCD などの標準のモデル タイプを使って図示されます。 時間属性の指定と組織ユニットの割り当ては、プロセス原価管理において重要なステップです。

会社の組織は組織図で記述されます。この図では、組織ユニットが ([原価率] および [製品/サービス] 属性を持つ) 原価センタに対応します。

タスク: 原価センタ分析

ARIS サポート: CD 図で定義される原価作用因を原価センタの運用率の中で使用できます。

計算は、任意の数の原価要素に対して実行できます。 原価要素の構造は、原価要素図で表現されます。

さらに、分析するオブジェクトを記述するための原価要素テーブルとファンクション テーブルを作成する必要があります。

タスク: プロセス計算

ARIS サポート: プロセス原価率の決定を含む、完全な原価センタ分析が実行されていることが前提となります。 プロセス計算を実行するために、別のモデルは必要はありません。 結果は計算テーブルに示されます。

8.5 品質管理

「品質管理」(QM) という用語は、企業の品質に関する方針、目標、責任を決定するためのあらゆるアクティビティに関わります。 こうしたアクティビティを実施する手法には、品質計画、品質管理 (プロセス管理)、品質保証、品質改善 (品質推進) などがあります。

タスク: QM 関連文書の作成

社内の製品やプロセスの品質を保証するには、企業が製品やプロセスを評価、比較、改善できるような適切な文書を作成する必要があります。

ARIS サポート: 製品を効率的に分類できるので、製品マニュアルの作成には製品ツリーを使用できます。 このタイプの表現は、サービス業や、特に行政分野で使用されることが多くなっています。 さらに、製品選択マトリクスを使用して、企業のどのファンクションがどの製品の製造に必要か、どの組織ユニットが生産に責任を持つかなどを示することもできます。

このほか、EPC、PCD、オフィス プロセス、製造プロセスなどの手法で記録できるプロセスを文書化したり、レポートで評価したり、社内の文書およびアプリケーションを参照したりすることも、QM における文書化の主な目的の 1 つです。

タスク: 認証手続き

プロシージャ モデルを使用して、ISO や VDA など、国内外の標準規格の認証を受けるためのプロジェクト管理をサポートします。

ARIS サポート: 認証について説明するプロシージャ モデル (たとえば、ARIS プロシージャ モデル) は、付加価値連鎖図を使って表現できます。 個々の手順は、追加のプロセス モデルをアサインすることによって、さらに詳しく説明できます。

タスク: 認証の文書化

認証に必要となる品質管理文書を作成します。

ARIS サポート: 構造モデルでは、個々の認証基準を各コンポーネントに分割します。 構造モデルの個々のアイテムは、品質管理 のために企業モデルにアサインできます。 これらのモデルには、EPC、オフィス プロセス、製造プロセス、組織図、付加価値連鎖図などの形式のプロセス モデルがあります。

ARIS Architect を使用して、認証を目的とする品質管理マニュアルとして承認されるレポートを作成できます。

8.6 再編成措置

コストまたは時間の削減、結果または作業の品質改善のための再編成措置には、業務のプロセスを変更する方法 (プロセス再設計)、あるいは完全に新たに作成する方法 (プロセス リエンジニアリング) があります。

タスク: プロジェクト文書

再編成措置の計画、実装、および結果を文書化します。

ARIS サポート: 再編成プロセスの主要なプロジェクト フェーズは、付加価値連鎖図によるプロシージャ モデルとして説明できます。

再編成プロジェクトの個々のプロジェクト アクティビティとその操作順序は、EPC を使って文書化できます。

プロジェクトに含まれる要員およびユニットの組織上の割り当ては、組織図で表すことができます。

タスク: 再編成の実行

再編成プロジェクトでは、まずプロジェクトの準備と戦略的計画、続いて現状の分析、目標概念の開発が行われ、最後にソリューションが導入されます。

ARIS サポート: 一般的な戦略条件を文書化するには、目標図と製品/サービス図を使用します。これにより、企業の主な事業区分を、その製品、サービス、顧客グループ、重要成功要因とともに記録し、企業の目標の階層を表すことができます。

現状の分析を実行するには、まず付加価値連鎖図を使って、主要な業務プロセスを含むフレームワークを作成します。 次に、従業員との面談に基づいて、これらの業務プロセスを EPC またはプロセス連鎖図 (PCD) の形で詳しく記録します。 PCD は、メディア ブレークやプロセス担当者の変更が原因である弱点を見極める場合に、とくに適しています。

スループット時間、プロセス原価、組織ブレーク、システム ブレークやメディア ブレーク、データの重複などを考慮した弱点分析に続いて、代替の目標プロセスが定義されます。 実際のデータと共に、これらのプロセスは EPC を使ってモデル化されます。

目標概念が完成したら、システム、組織、およびデータのコンポーネントをより詳細に記述すると、実装し易くなります。 たとえば、ここでアプリケーション システム構造の「ワープロ」を「Microsoft Word」と指定できます。

注意: 弱点分析フェーズは、シミュレーションおよび ARIS Optimizer を使用した評価においてサポートされます。

8.7 SAP R/3 導入

SAP 社の標準 R/3 ソフトウェアの実装における ARIS のサポート機能では、ASAP の実装アプローチのライフ サイクルに焦点が当てられています。 また、ASAP のほかに、業務プロセスの最適化 (もっとも広義な意味で) を重視したアプローチもサポートしています。 以下に説明する ARIS のサポートの一部は、ARIS Architect extension pack SAP でのみ提供されます。

タスク: フェーズ分析、仕様 (プロジェクトの準備)

このタスクでは、企業独自のプロセスの SAP システムによる導入度を明確にするとともに、潜在的な弱点を適時に認識できるようにします。

ARIS サポート: SAP R/3 リファレンス モデルによって直接分析されない場合は、サポートする「最適な業務プロセス」は ARIS を使用してモデル化できます (「一般的な文書化」の項を参照)。 そのあとで、企業のモデルを SAP R/3 リファレンス モデルに対応させることにより、導入度の初期予測をレポートから得ることができます。

タスク: 設計フェーズ (業務の設計)

不足しているプロセスまたはファンクション、およびその両方は SAP R/3 リファレンス モデルで識別されています。

ARIS サポート: R/3 リファレンス モデルの既存のコンポーネントに基づいて、ARIS で新しいプロセスおよびシナリオのバリアントを作成できます。 さらに、プロセスおよびシナリオのコンポーネントに、新しいファンクション、イベントおよびルールを (必要な場合は新しい SAP ABAP ファンクションを作成して) 追加することもできます。

タスク: 設計フェーズ (業務の設計)

SAP R/3 と非 SAP アプリケーションとの間のインターフェイスを設計します。

ARIS サポート: 交換が必要な属性については、データまたはオブジェクトをファンクションおよびデータ モデルまたはオブジェクト モデルに割り当てることによって、ARIS リポジトリのプロセス モデルで詳しく文書化できます。 これらの情報は、レポートとして出力することもでき、インターフェイス開発の基盤を形成します。

タスク: 設計フェーズ (業務の設計)

業務プロセスおよび SAP システムの組織レベルの設計を作成します。

ARIS サポート: 企業の社会的構造および SAP 組織構造は、組織図を使って説明し、並置することができます。

8.8 ソフトウェアの開発と導入

タスク: プロジェクト文書

ソフトウェアの開発と導入の計画、手順、結果を文書化します。

ARIS サポート: プロジェクトの主要段階は、付加価値連鎖図によりプロシージャ モデルとして説明することができます。

開発および実装中の個々のプロジェクト アクティビティとその操作順序は、EPC を使って文書化できます。

プロジェクトに含まれる要員およびユニットの組織上の割り当ては、組織図で表すことができます。

タスク: アプリケーション システムおよびモジュールの指定

このタスクでは、情報システムの構造をシステム要件に基づいて示します。

ARIS サポート: 開発するソフトウェア システムの応用事例は、ユース ケース図を使って表すことができます。 また、システム ユーザーを定義して、個々のユース ケースに割り当てることもできます。 ユース ケース図は、多くの場合、詳細なプロセス モデリングの出発点となります。 プロセス モデルは、個々のユース ケースにアサインできます。

アプリケーション システム タイプ図を使用すると、タイプ レベルのアプリケーション システムの階層構造をモジュール タイプおよび IT ファンクション タイプを使用して説明できます。

特定のタイプについてより詳細に説明する具体的なオカレンスを、アプリケーション システム図で表すことができます。

タスク: IT プロセスの記述

このタスクでは、モジュール内またはモジュール間におけるプロセスの時系列の操作順序を説明します。

ARIS サポート: IT プロセスは、プログラム フロー図でモデル化できます。

タスク: システム インターフェイスの開発

このタスクでは、ユーザー インターフェイスの開発と文書化を行います。

ARIS サポート: 画面 (ウィンドウ) の構造的および機能的な構成は、画面図によって説明できます。 ERM からリレーション図への移行と同様、画面図を元にしてプログラム コードが導き出されます。

8.9 知識管理

包括的な知識管理の設計の出発点となるのは、企業において知識が事実上もっとも有力な生産要因になった、あるいはなりつつあるという認識です。 このため、知識を従来の生産要因と同じように制御可能な要素として理解することが必要となってきました。

その結果、知識管理の焦点は、知識の取得、表現および分布に置かれています。 知識管理とは、組織が知識を開発し、時間、人、場所に関係なく透過的に利用できるようにするためのあらゆる手法、手段、システムの集合です。 知識管理の目的は、知識を増やし、企業の既存の知識を最適な方法で活用することにあります。

タスク: 知識マップまたはイエロー ページ

社内で利用できる知識とその場所を示します。

ARIS サポート: [知識マップ] のモデル タイプを使って、さまざまな知識カテゴリの組織内での分布を表示できます。 ここでは、特定の知識カテゴリのノウハウを所有している組織ユニット、役職または従業員、およびそのコンピテンス レベルを表すことができます。

タスク: 知識の分類

組織の知的資本を分析的に分類します。たとえば、知識の保存構造の設計などを行うために、知識のさまざまなタイプおよびグループを説明します。

ARIS サポート: 知識構造図を使って、組織の知識基盤がさまざまな知識カテゴリに分類され、これらがさらに知識カテゴリおよび有形知識に分割される様子を表すことができます。 有形知識の場合は、この知識が保存されている情報媒体を示すこともできます。

タスク: 業務プロセスにおける知識処理

業務プロセスのどこで知識が生成、変更、および必要とされているかを示し、知識リソースをできるだけ有効に活用できるようにします。

ARIS サポート: [EPC]、[プロセス連鎖図]、[オフィス プロセス]、[製造プロセス]、および [ファンクション割当図] の各モデル タイプでは、[知識カテゴリ] および [有形知識] のオブジェクトを使用できます。 知識の構造および組織内での分布については、それぞれ知識構造図および知識マップを使って個別に説明できます。

8.10 ワークフロー管理

ワークフローは、もっとも広い意味における業務プロセスとして解釈することができます。「ワークフロー」という用語は、業務トランザクションを実行するために開始される、分業に基づくプロセスを意味します。 これには、単純な業務プロセスや複数の組織にまたがる複雑なプロセスが含まれます。 この分析では、終始一貫してプロセスの動的な流れに焦点が置かれています。 ワークフロー管理とは、ワークフローの作成、管理、および最適化に使用する手法、手段、およびシステムの集合です。

ワークフロー管理のシステムは、アクティブに稼動している、柔軟性の高いソフトウェアで、組織におけるルールのフレームワークに従って動作し、複数のワークステーションにわたる、既存の基本技術コンポーネントを統合するプロセスを制御します。 プロセス制御システムを使用することで、多数の従業員および役職が関わる複雑なタスクをサポートできます。

タスク: ワークフロー管理システムのプロセス カスタマイズ

ARIS では、一般的な業務プロセス モデルを、さまざまなワークフロー管理システムを（半）自動的に設定できるワークフロー モデルに変換する際のサポートを提供しています。

ARIS サポート: 権限図を使用して、存在するワークフロー (プロセス)、およびそれを開始する権限を持つ要員またはグループを説明します。

プロセス モデルと同様に、アクティビティ フローは EPC で表されます。 モデリングは、手法に忠実に従って行う必要があります。また、各ファンクションに対してファンクション割当図を作成する必要があります。EPC において割り当てが示されていない場合には、このファクション割当図でユーザーおよび入出力データ をファンクションに割り当てます。

データに関連するアプリケーションを実行時に自動的に起動するには、アプリケーション システム図またはアプリケーション システム タイプ図で、ファイルをアプリケーションへ割り当てる必要があります。

9 バランス スコアカード メソッド

9.1 はじめに

企業が過酷なビジネス環境と国際的な競争で生き残るには、最善の業務プロセスが必要です。 さらに、ビジネス環境の新たな展開に、自社の戦略目標に沿って迅速に対応することも同様に重要です。 このためには、企業戦略を実現して戦略目標を達成するための効率的な管理プロセスを確立し、運営指針によって日々の業務に組み込む必要があります。

従来の数多くの管理アプローチでは、戦略指向的な方法や戦略目標を達成する継続的な管理に基づいて、企業戦略の形成とその実現の間の接続性を確立することができません。

また、多くの企業は依然として財務 KPI (重要業績評価指標) 財務 KPI のみに基づいて運営されていますが、これには限界があります。なぜなら、これらは主に過去の実績を表すものであり、将来の経営に関わる情報はほとんど提供できないからです。 財務における成功 (顧客、プロセス、イノベーションなど) の理由を考えてはじめて、戦略目標の達成を妨げる可能性がある要因を早い段階で見分けることができます。

バランス スコアカード メソッドという構造化手法は理解しやすく、導入も容易であり、こうした弱点の回避に役立ちます。

9.2 バランス スコアカード メソッドの概念

9.2.1 バランス スコアカード メソッドの主な要素

バランス スコアカード メソッド (BSC 法) は、1992 年にロバート・キャプランとデビッド・ノートンが提案した戦略的管理システムです (『The Balanced Scorecard Measures that drive Performance』、Harvard Business Review、1992 年 1 月/2 月)。 これは「パフォーマンスの測定手法」をテーマとした研究の成果として開発されました。 この研究から、財務 KPI とのみ連動しているパフォーマンス測定システムを使用すると、付加価値活動がしばしば妨げられることが明らかになりました。 この事実に基づき、キャプランとノートンは革新的な企業と協力して、企業のビジョンおよび戦略の実現を最適に測定できる新しい KPI システムを開発しました。

バランス スコアカード アプローチでは、さまざまな企業の展望 (いわゆる「視点」) に KPI を関連付けます。 視点には、内部的なパフォーマンスの視点 (たとえば、学習および成長に関する視点、プロセスの視点) および外部パフォーマンスの視点 (たとえば、顧客の視点、経済/財務の視点) が含まれます。 このような KPI の割り当てにより、短期的な目標と長期的な目標、財務 KPI と非財務 KPI、事前的指標と事後的指標、内部的な視点と外部的な視点などの間で、特定のバランスを取ることができます。 業界固有の KPI を統合することにより、こうした概念にさらにベンチマーク要素が追加されます。

純粋なパフォーマンス測定手法は、企業のビジョンと個々の競争戦略から、均衡化された指標を使用した措置の策定および管理の段階までを含む、目標指向の企業経営のための包括的な管理システムへと発展しました。 したがって、バランス スコアカード メソッドは、KPI を使用するだけの単なるパフォーマンス測定システムではありません。 この手法は、企業戦略の伝達および導入に役立ち、そこから導出される戦略的学習プロセス (二重ループ学習) をサポートします。

9.2.2 戦略的管理プロセスおよびバランス スコアカード

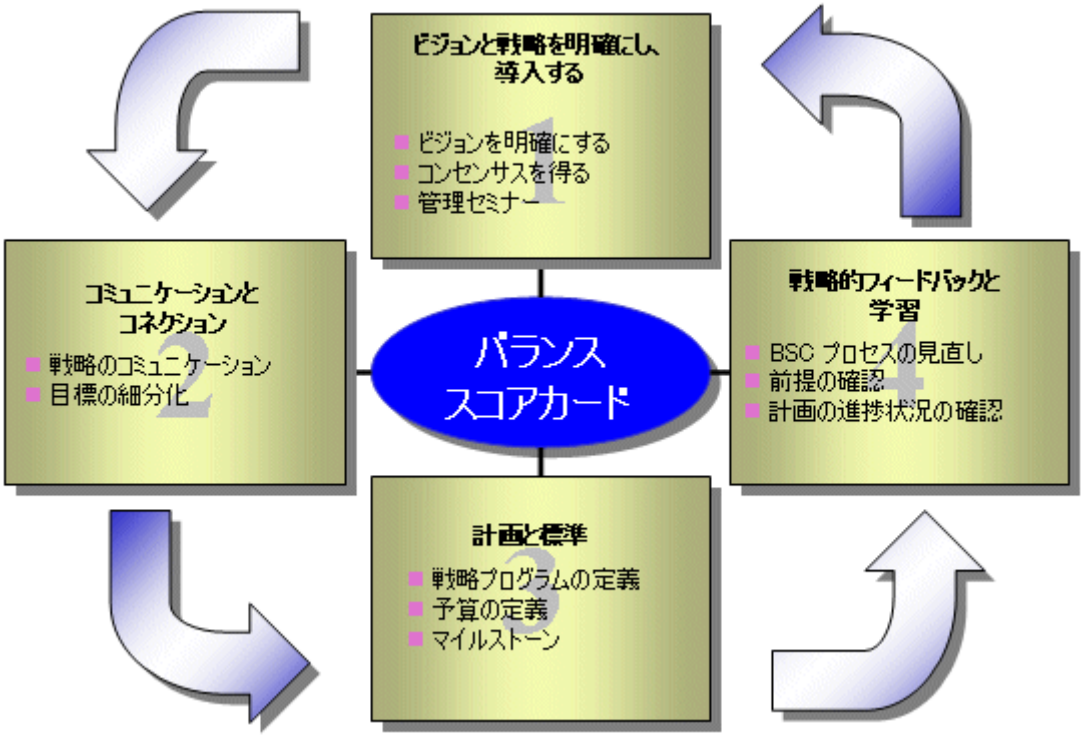


図 181: 戦略的管理構造としての BSC

バランス スコアカード メソッドにおける戦略的管理プロセスには、次の 2 つの段階があります。

1. 第 1 段階では、戦略的な分析に基づいて企業の戦略を確立する必要があります。 この分析の範囲内では、競争に関するすべてのデータを記録します。 この分析の目的は、ビジネス環境の展開にともなうトレンド、機会、リスクに関する情報を収集して評価し、自社の競争力を確定することにあります。 この段階の最後で、自社の企業戦略を定義します。
2. 第 2 段階では、企業戦略を導入します。 この段階では、BSC メソッドを活用します。 第 1 段階で策定した企業戦略は、局所戦略 (たとえば、事業分野個別の戦略) を作成する過程で詳細化します。 これにより、詳細な戦略目標が策定されます。 測定カテゴリの目標を定めて、戦略を具体化します。 この目標の達成方法は、進め方に関する手順形式のアクション プログラムを作成して規定します。 業務計画の範囲内で、このアクション プログラムを企業のさまざまな部署や部門に細分化します。 そして、予算を付けて個々のアクションをより具体化します。 個々の戦略に関連するスコアカードを使用し、作成した KPI に基づいて目標達成度を測定します。 定義済みのスコアカードに基づいてフィードバック プロセスを実施し、生じる可能性がある偏差により、具体的プログラムを追加したり、戦略を「作り直し」たりします。

9.2.2.1 ビジョンおよび戦略の形成および実現

企業の戦略はビジョンから導出されます。

ビジョンとは、対象企業の上位企業理念を表します。一般的で、印象的なステートメントであるビジョンは、不明瞭ではあるけれども、アクションのよりどころとなり、企業哲学を簡潔に表現するものです。

戦略は、個々の戦略的ビジネス ユニットそれぞれに対して策定します。 策定した戦略は、企業の業績目標に則している必要があります。 このため、BSC を導入する前に、企業の上級管理職がワークショップを開いて、戦略的ビジネス ユニットに対するビジョンとそれを実現するための戦略を策定する必要があります。 一般的に、財務目標 (財務の視点) の設定が必要となります。 焦点となる対象はさまざまです。たとえば、資本収益、ROI、株主価値、売上高、キャッシュ フローなどがあります。 財務目標を達成するには、対象となる市場 (顧客の視点) での具体的な行動が必要です。 したがって、目標を設定したら、それに基づいて適切な市場および顧客セグメントを選択します。 この顧客の視点は、戦略目標の定義および関連する KPI の決定にも関わります。 たとえば、指定した顧客層における市場専有率または成長率に関連します。

特定の市場を想定した戦略を導入するには、企業側に適切なリソース (資源) が必要です。 バランス スコアカードの作成では、リソースは次の 2 つのカテゴリに分割されます。

1. 財務および顧客目標を定義したら、次に主な業務プロセス (プロセスの視点) に注目して、目標を決定し、具体的プログラムと KPI を決定します。 通常、対象となるのはプロセスにかかる時間 およびコストです。
2. 人材リソースの開発、情報技術、革新に関する戦略目標は、いわゆる「学習および成長の視点」の一部として、財務、顧客、プロセスの各視点の戦略目標から導出されます。

この意味で、バランス スコアカードを設定する場合、定義された戦略目標はすべて相互に関連性があると言えます。これを因果関係連鎖と呼びます。

効率的な最初のステップのために、ここで重要なのは、ビジョン、戦略、ならびにそれらから導出された戦略目標について、経営陣が共通の認識を持つことです。

9.2.2.1.1 バランス スコアカードの標準的な視点

キャプランとノートンは、バランス スコアカードの作成に使用する次の 4 つの標準的な構造を提案しています。

1. 財務的な視点 - 「この戦略が財務的に与える影響は何か？」といった株主の視点です。
2. 顧客の視点 - 「対象となる市場において自社をどのように位置付けるか？」といった顧客の視点です。
3. プロセスの視点 – 「どのプロセスに戦略的重要性があるか？」といったプロセス要件です。

4. 学習の視点 - 「どのようにして学習し続ける組織にできるか？どのようにして成長を促進できるか？」といった組織の学習および革新のための要件です。

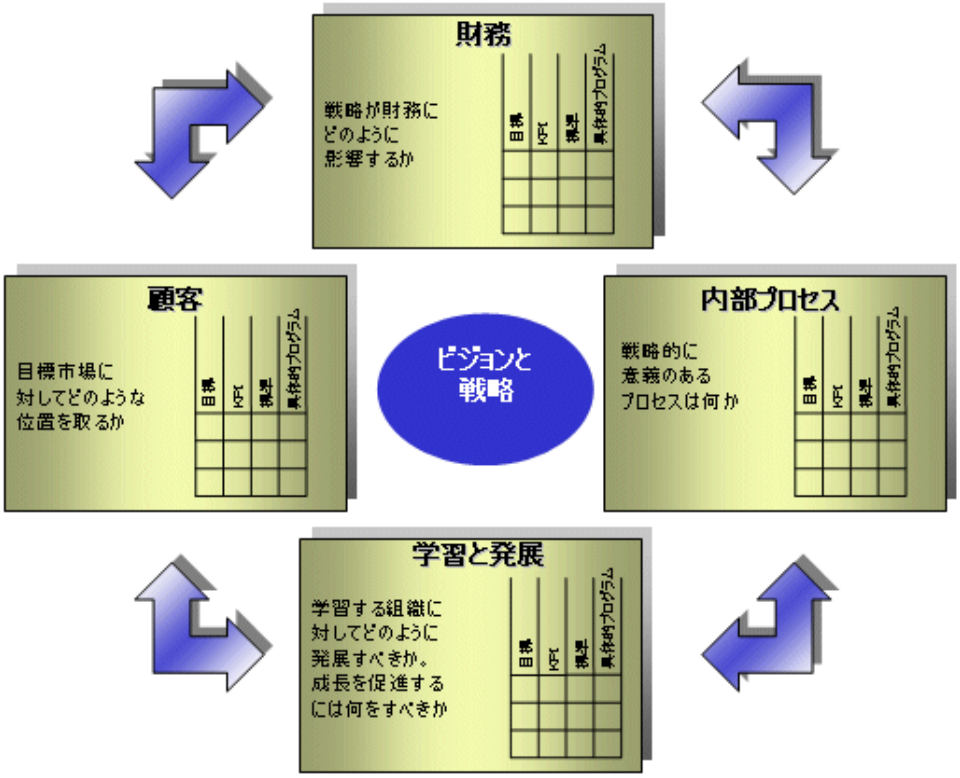


図 182: BSC の視点

標準的な視点には暗黙の論理があり、戦略の導入および個々の因果関係の形成に役立ちます。 ただし原則的に、企業戦略に関係し、目標および KPI が標準の視点に直接関係がある場合は、企業のほかの視点 (たとえば、環境的な視点) を定義することも可能です。

9.2.2.1.2 因果関係連鎖

因果関係連鎖では、戦略の個々の目標どうしの因果関係、すなわち複数の KPI の視点間の関係を定義します。 つまり、各視点の目標の達成方法を説明します。 視点自体により因果関係連鎖の一般的な条件 (「企業に大きな影響を与える領域」) が形成されます。 学習および成長の視点環境を向上させると、内部プロセスの視点の目標および KPI に直接的かつ肯定的な効果が現れると考えられます。 さらに、プロセスの視点から見た成長は、顧客の視点の目標および KPI に肯定的な影響を及ぼし、それが財務的な目標の改善につながります。

企業全体を通して有効なバランス スコアカード システムの開発の範囲内で、目標および KPI を戦略別に定義し、それらの関係を視点内だけでなく視点間でも表します。 財務の視点で指定された目標および KPI に基づいて、「肯定的な作用因」(パフォーマンス作用因) が顧客の視点で検出されます。 次に、これらを内部プロセスの視点によって分割し、学習および成長の視点へと細分化します。 このプロセスにより、企業または部署の戦略を局所的な目標および KPI に分割し、そこから業務に則した措置が導出できます。

因果関係連鎖を確立し、KPI を定義することで、事前的指標と事後的指標との間のバランスを暗黙のうちに取ることができます。

9.2.2.1.3 事前的指標および事後的指標の定義

因果関係連鎖を確立することにより、個々の目標と KPI との間に一時的な接続線を生成できます。 一般にパフォーマンス作用因（事前的指標と呼ばれます）は「学習および成長の視点」ならびに「プロセスの視点」に属し、顧客の視点および財務の視点の結果より先に現れます。 顧客の視点および財務の視点の KPI はほとんどが結果指標で、事業が成功したかどうかを測定します。これは事後的指標と呼ばれ、過去に注目する指標です。

事前的指標と事後的指標の両方を、バランス スコアカードの各視点で定義し、具体的プログラムと目標達成度の相互関係を表す必要があります。 事前的指標により、指定した目標との偏差を初期の段階で認識することができます。

9.2.2.2 詳細なスコアカードの伝達および派生

バランス スコアカードの概念を利用した企業戦略とその実現による効果は、スコアカードが全社的に認識され、受け入れられるかどうかによって異なります。 したがって、自社の企業戦略およびスコアカードを企業のあらゆる階層レベルで普及させるには、積極的な普及活動が必要となります。

トップダウン的なアプローチでは、ターゲットは企業の下位階層レベルのために上位企業戦略（企業スコアカード）から導出され、上位戦略目標を特定の部署の目標に合わせて調整します。 目標達成度を測定する KPI も作成されます。 目標を達成するには、企業のそれぞれの部署にさまざまな具体的プログラムが必要です。また、これらの具体的プログラムを基礎となる階層レベルのスコアカードに記録する必要があります。 ターゲットは、全体的な戦略と一致しない可能性のある短期的な結果を超え、長期戦略的な考慮に基づいている必要があります。 短期的な結果の例としては、短期間の生産量増加によるコストの低下などがあります。

9.2.2.3 計画とターゲット

BSC を使用して、企業のさまざま階層における戦略の導入プロセスを予算編成と組み合わせます。 この組み合わせの目的は、あらゆるリソースを企業戦略と調整することです。

BSC は、次の 4 つの段階で長期的な戦略計画および予算編成に統合されます。

1. 高い達成目標を設定し、それを全従業員に伝達し、了承を得ます。 KPI 間の因果関係を使用して、顧客の視点および財務の視点の結果に関連する KPI を識別します。
2. 現在値と目標値との間の差が大きい特定の KPI に対し、戦略に沿った具体的プログラムを導入します。 長期的には、この手順により、資本投資およびアクション プログラムが、戦略的に重要な目標値にまとめられます。
3. 企業全体にとって重要な具体的プログラムを明確にすることで、戦略目標による相乗効果が得られます。
4. 各期間（5 か年、3 か年、および年間計画）の予算に基づいて、KPI の計画値を企業計画に関連付けます。

9.2.2.4 戦略的な学習およびフィードバック

バランス スコアカードの概念は、企業における戦略的な学習に役立ち、戦略の実現に関するフィードバックを可能にします。

戦略的な学習は、すべての従業員と学習する組織としての企業全体を対象しています。 従業員は、戦略を把握し、各自が戦略に沿ったアクションを実行する必要があります。 フィードバック プロセスでは、戦略の達成度を評価するためのパフォーマンス データを収集します。 戦略目標に関する因果関係を基に以前確立した仮定を検証することにより、仮定をチェックします。

このように、二重ループ学習を実施し、戦略的目標に対してボトムアップ プロセスによる厳しいレビューを行い、新しい因果関係を生成します。 これにより、増え続ける情報を基にさらに効果的に戦略が遂行されるようになります。

したがって、フィードバックは BSC の細分化を吟味するプロセスに相当するため、非常に重要です。 フィードバックには、進捗状況の計画値という形式による KPI の報告だけではなく、追加の具体的プログラムや新しい相互関係の提案も含まれます。これにより、戦略の実施における潜在的な矛盾点を明らかにします。

9.2.3 バランス スコアカードの利点

バランス スコアカード メソッドの利点は、その一貫性のある戦略指向的な企業管理にあります。これには、企業の最下層が含まれ、そこから戦略に基づいて業務に則した具体的プログラムが導出されます。 企業のあらゆるリソースおよび従業員が、その企業の戦略に合せて調整されます。 戦略は、バランス スコアカードを使用して徹底され、監視されます。

バランス スコアカード メソッドでは、企業管理における従来の KPI システムの欠点が次のように改善されています。

- 財務 KPI 以外の指標も考慮できる
- 過去の実績にのみ焦点を置かない
- 定性的なデータも考慮している
- 複雑さが低減され、コンセンサスを構築する
- バランス スコアカード メソッドの出発点として戦略目標を利用できる
- 既存のデータではなく、経営上のボトルネックに注目している
- 必要な変更を迅速に実施できる

バランス スコアカード メソッドは、とくに戦略指向型の企業経営にとって理想的です。その理由として、次の障壁を解決していることが挙げられます。

- 具体化に関する障壁: 従来の管理アプローチでは、ビジョンおよび戦略が理論的に規定されているだけで、個々の (実施可能な) 実行計画に展開されていない。
- ビジョンに関する障壁: 戦略が実施の主体である従業員に理解されていないことがある。
- 実行に関する障壁: 戦略が個々の部署または個人の目標に関連付けられていない。
- 導入に関する障壁: 報告が経営上の財務目標指向であり、戦略的目標を意識していない。
- 経営に関する障壁: 予算編成プロセスが戦略計画プロセスと連携していない。

9.3 ARIS BSC を使用したバランス スコアカードの作成

9.3.1 用語および略語

ビジョン: ビジョンとは、企業の将来に関する戦略的な方向性および使命のことで、さまざまな戦略を使用して実現します。

戦略: 戦略は、ビジョンに基づいて作成します。 戦略は、局所的な戦略に分割されます。

戦略目標: 個々の戦略は、複数の戦略目標で構成します。 一般的に、戦略目標は経営陣がワークショップを開いて策定します。 決定した目標は、因果関係に従って時系列に相互に接続されます。

成功要因: 成功要因は企業の業務の成功に影響を与えます。 ほかの成功要因との相互作用があり、特定の効果のある特定の動作に向かわせることができます。

視点/展望: 視点とは、自社の展望をより具体的に表現したものです。 原則として、視点を選択する場合、企業の要員指向の観点、内部の観点、プロセス指向の観点、外部の観点といった観点を含める必要があります: 。

KPI: 戦略目標または成功要因にはそれぞれ KPI を割り当て、パフォーマンスおよび目標の達成度を測定します。

実績値: KPI、戦略目標、または成功要因の現在の値です。 実績値に基づいて、計画と実績の比較を行って、現在の目標達成度を導き出すことができます。

計画値: 計画値は、KPI、戦略目標、成功要因に対し、特定の期間における目標として設定します。 戦略目標の場合、各 KPI の目標達成度の分担率から導出するパーセンテージとして設定します。 計画値は個々の期間に分割したり、期間的な変動を含めたりできます。 計画値は、全体を 100% とした場合の目標達成度として表します。

目標値: 将来の特定の期間の目標として設定された値のことです。 通常、目標期間になるとこの値が計画値になります。

最小値: 目標、重要成功要因、または KPI が取り得る最小の値です。 ARIS では、最小値はデフォルトでゼロに設定されます。 この値は変更できます。

最大値: 戦略目標、成功要因、または KPI が取り得る最大の値 (上限) です。 ARIS では一般に、複数の KPI が取る値を標準化して相互に比較できるようにするために使用します。

警告限度: 警告限度は、計画値と連動しています。つまり、計画値は KPI、戦略目標、成功要因の値が、予期されている目標を下回る場合の限度に対応します。

値許容範囲: 値許容範囲とは、計画値に対する負の偏差で、この範囲内にある戦略目標、KPI、成功要因は許容されます。

警報限度: 警報限度は、値許容範囲を差し引いた計画値を示します。 警報限度を超えた戦略目標、KPI、成功要因の値はすべて、許容されなくなります。

具体的プログラム: 具体的プログラムは、単一の目標または複数の戦略目標の達成に影響します。 一般に、具体的プログラムには責任者が割り当てられ、開始日、終了日、リソースなどが含まれます。

指標タイプ: KPI は、[事前的指標] または [事後的指標] のいずれかのタイプです。 事前的指標は、パフォーマンス作用因を計測し、将来をポイントします。 事後的指標は、結果を計測し (結果指標)、遡及的に動作します。 経済/財務目標を一般的に事後的指標とし、プロセスならびに学習および成長の視点における目標を事前的指標とするケースが増えてきました。 KPI 間の因果関係を表すには、事後的指標も視点の中に含めることをお勧めします。

データ ソース: 各 KPI にはデータ ソースがあり、そこからバランス スコアカード システムにデータを転送できます。

9.3.2 ARIS BSC を使用したバランス スコアカードの作成

9.3.2.1 視点の指定

バランス スコアカード プロジェクトでは、まず戦略計画で使用する視点を指定する必要があります。 ここで指定した視点は、すべての企業スコアカードに有効です。

[視点] オブジェクト タイプは、この目的で使用します。 視点は、BSC 因果関係図でモデル化できます。

視点は、[影響を受ける] 接続線を使用して接続します。 特定のシーケンスを定義する必要はありません。 ただし、論理構造を確立すると (順番に配置された個々の視点を使用します)、モデルの自動生成が大幅に簡易化されます。 また、因果関係連鎖の論理構造がより分かりやすくなります。

9.3.2.2 バランス スコアカード システム構造の指定

バランス スコアカード システムを作成するためのフレームワークとなるモデルを次に示します。

- 組織図

バランス スコアカード システムは、企業の組織構造に応じて構成できます。 任意の数の BSC 因果関係図を組織図上のオブジェクトにアサインすることができます (バリアント作成、履歴管理などのため)。 このアサインメントにより、戦略の導入に必要な目標と対応する組織ユニットとの間の接続線が確立されます。 組織図に基づいて、企業のバランス スコアカード システム全体を、企業の最上レベルから個々の部署または従業員へと細分化できます。

- 構造モデル

  企業が組織図ではなく戦略的な事業分野に基づいてバランス スコアカード システムを作成する場合、バランス スコアカード構造の開始モデルとして構造モデルを使用できます。 この場合、任意の数の BSC 因果関係図を構造オブジェクトにアサインすることができます。

- 付加価値連鎖図またはファンクション ツリー

  バランス スコアカードは、企業のパフォーマンス管理およびパフォーマンス測定の手段として使用されるため、ARIS ではバランス スコアカード システムを付加価値基準の観点から確立できます。 これには、付加価値連鎖図およびファンクション ツリーを使用できます。

9.3.2.3 因果関係の指定

組織図のオブジェクト、および企業レベルのバランス スコアカード構造を規定する構造モデルのファンクションや構成要素を、BSC 因果関係図にアサインします。

BSC 因果関係図には、成功戦略の導入に必要な戦略目標と成功要因が定義されており、これら相互の影響はさまざまなバランス スコアカードの視点で図示されます。 ARIS では、企業のあらゆる因果関係連鎖で使用する個々の視点をオブジェクトとして作成します。

BSC 因果関係図はレーン型モデルで、[視点] オブジェクト タイプを [関連する視点] 列に、[戦略] オブジェクト タイプを [戦略] 行にそれぞれ保存します。 [目標]、[成功要因]、[KPI] オブジェクト タイプは、因果関係列にモデリングできます。 戦略目標およびその成功要因の影響の度合いは、因果関係列で説明します。 影響が強いほど、矢印が太くなります。

また、バランス スコアカードで使用する詳細な成功要因は、BSC 因果関係図で定義できます。 [成功要因] オブジェクト タイプを使用することにより、たとえば市場の成長など、企業が直接影響を与えることができないファクターおよびその KPI をバランス スコアカードに取り込むことができます。

BSC 因果関係図は、プロセス ビューの要件定義レベルで表示できます。

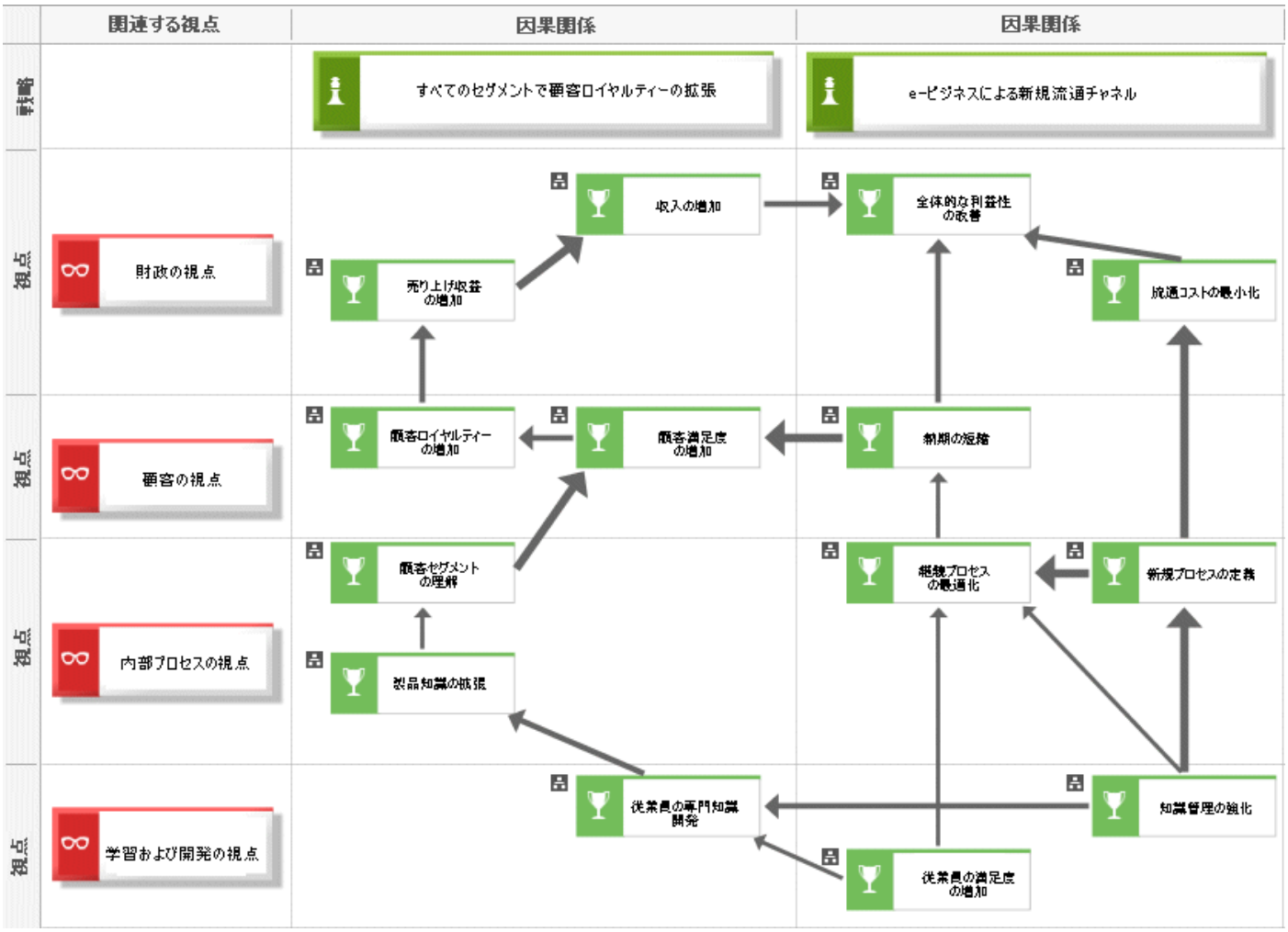


図 183: BSC 因果関係図

BSC 因果関係図のモデリング列の [目標] および [成功要因] オブジェクト タイプは、BSC KPI 割当図に対してのみアサインすることができます。

BSC 因果関係図では、次のシンボルを使用します。

シンボル	オブジェクト タイプ
視点	視点
戦略	戦略

シンボル	オブジェクト タイプ
	目標
	成功要因
	KPI インスタンス

9.3.2.4 目標監視のための具体的プログラムおよび KPI の指定

戦略目標および成功要因を BSC 因果関係図で定義したあと、BSC KPI 割当図を個々のオブジェクトにアサインします。図で目標および成功要因には KPI が割り当てられ、関連する目標または成功要因に対するベンチマークとなります。 このような KPI は、あとで識別され、チャネル化されます。 複数の KPI がオブジェクトに対して割り当てられている場合、[測定方法である/測定する] タイプの接続線を使用して KPI の分担率を指定できます。 これにより、目標または成功要因の目標達成度に関して、KPI がどの程度重要であるかが示されます。 ARIS BSC の分析スクリプトを正常に実行するため、[分担率] 属性が [測定方法である/測定する] タイプのすべての接続線に指定されている必要があります。

KPI のほかにも、BSC KPI 割当図で目標および成功要因にオブジェクトを割り当て、エンティティ タイプ、ファイル、文書、データベース、データ媒体などのデータ ソースを定義できます。 この割り当てによって、ARIS のデータ ウェアハウス メソッドとの関係を確立できます。 これにより、バランス スコアカードの各 KPI が、データ ウェアハウス メソッドのどのデータ要素と接続されているかを、正確に定義できます。

[影響する/影響を受ける] 接続線タイプを使用して KPI 間の相互作用を表すことができるので、事前的指標が事後的指標に与える影響を BSC KPI 割当図で説明できます。 この場合、BSC KPI 割当図を新たに設計し、そこでは目標または成功要因への各 KPI の割り当てに重点を置くのではなく、ほかの BSC KPI 割当図で使用している KPI のコンソリデーションおよび定義、およびその効果の説明を強調することをお勧めします。

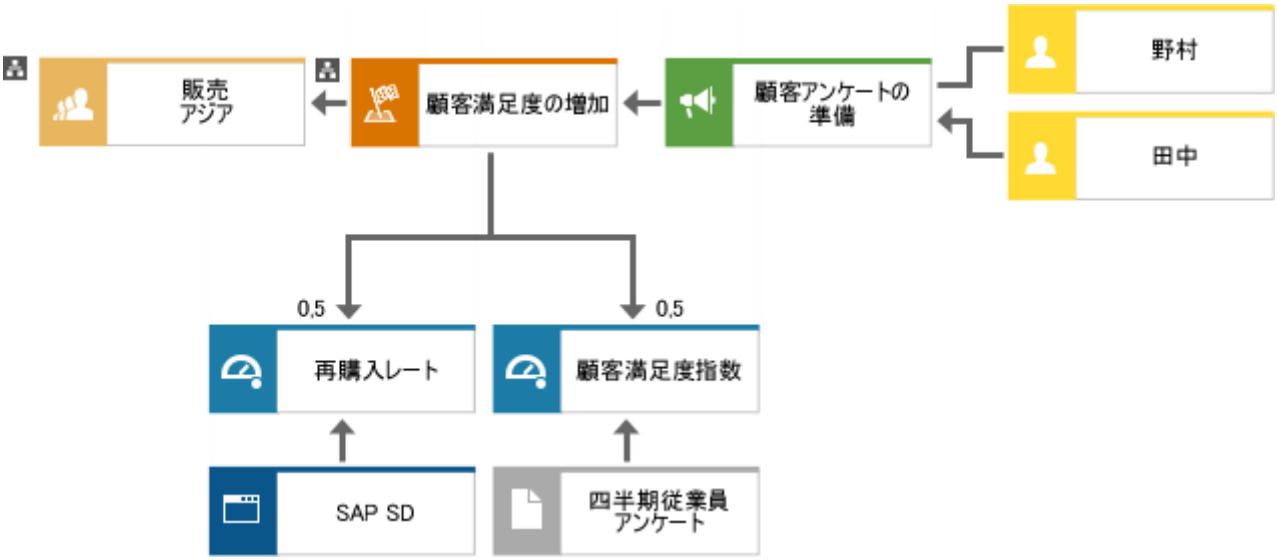


図 184: BSC KPI 割当図

BSC KPI 割当図は、プロセス ビューの要件定義レベルで表示できます。

BSC KPI 割当図では、次のシンボルを使用します。

シンボル	オブジェクト タイプ
戦略目標	目標
成功要因	成功要因
KPI インスタンス	KPI インスタンス
具体的プログラム	タスク
組織ユニット	組織ユニット
組織ユニット タイプ	組織ユニット タイプ

シンボル	オブジェクト タイプ
役職	役職
要員(女) 要員(男)	要員
ロール	ロール
グループ	グループ
コンピューター	アプリケーション システム タイプ
用語	用語
エンティティ タイプ	エンティティ タイプ
関係タイプ	関係タイプ
ERM 属性	ERM 属性
クラスター	クラスター/データ モデル
知識カテゴリ	知識カテゴリ
有形知識	有形知識
クラス	クラス

シンボル	オブジェクト タイプ
	情報媒体

9.3.2.5 KPI およびその関係の記述

戦略目標と重要成功要因を測定するための KPI を選択したあと、[KPI ツリー] モデル タイプを使用してこれらの指標とその関係を詳細に定義できます。

KPI ツリーでは、[影響する] 接続線タイプを使用して、さまざまな KPI を階層的に配置できます。 これらの接続線の [分担率] 属性を指定して、分担率から KPI ツリー内の全体 KPI を計算できます。 KPI ツリーは、全体 KPI を表す KPI インスタンスにアサインされます。 このアサインメントは、評価「BSC 管理ビューの作成」および「BSC 計画・実績比較」で含められます。

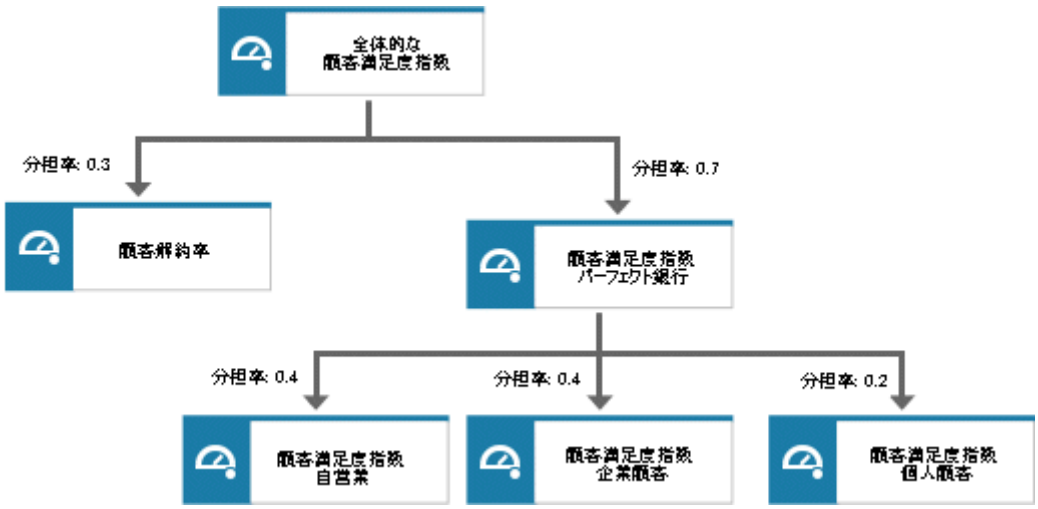


図 185: KPI ツリー

[KPI ツリー] は、[KPI インスタンス] オブジェクト タイプだけを使用します。

KPI ツリーは、データ ビューの要件定義レベルで表示できます。

9.3.3 ほかのモデルとの関係

バランス スコアカード メソッドのオブジェクトは、次の ARIS モデルに関連付けることができます。

- EPC
- 目標図
- DW 構造
- DW 変換

10 e-ビジネス シナリオ図

10.1 はじめに

企業内の業務プロセスを順調に遂行する重要性は、ますます高まっています。 一方では企業間の接点における、他方では企業とその顧客の間の接点における特定手順の業務の流れが関心の的になっています。 企業と顧客の接点に必要であるものは、透明性、迅速性、一貫性、そして方法の直接性です。

また、企業の観点からは適切なビジネス パートナーを速やかに見つけること、そして顧客の観点からは販売業者を見つけることが重要です。 これらのプロセスを最大限に最適化することが、競争上の優位につながります。 これらの複数方向の関係をサポートするための理想的なプラットフォームがインターネットです。 上記の環境におけるプロセスは大変複雑なため、「e-ビジネス」という用語を定義する必要があります。

e-ビジネスとは、企業の業務活動において、IT と通信技術の使用を表す一般的な用語です。 e-ビジネスには、電子媒体の使用を通じて、ビジネス パートナー、従業員、および顧客の 3 者の間にある関係とプロセスをサポートすることが含まれます (Herrmans, Sauter, 1999)。

したがって e-ビジネスとは、企業をプレゼンテーションするための Web サイトの作成や、インターネットを介した品物の購入、2 つの企業で共有する非常に複雑なプロジェクト、市場で出会うあらゆる数のビジネス パートナー間の多層的な関係などを意味すると言えます。

これらは、次の概念に分割することができます。

B2B (ビジネス対ビジネス)

ビジネス対ビジネスは、企業間で行われる取引を指します。 たとえば、企業のサプライ チェーンを連結することにより可能になります。

B2C (ビジネス対顧客)

ビジネス対顧客は、企業とその顧客の間で行われる取引を指します。 たとえば、オンライン ショップでの買物などがあります。

B2A / C2A (ビジネス / 顧客対行政)

ビジネス/顧客対行政は、企業や個人と行政とのあらゆる取引を指します。 企業と行政の取引は、コストを大幅に削減できる可能性を持つ領域であると考えられています。

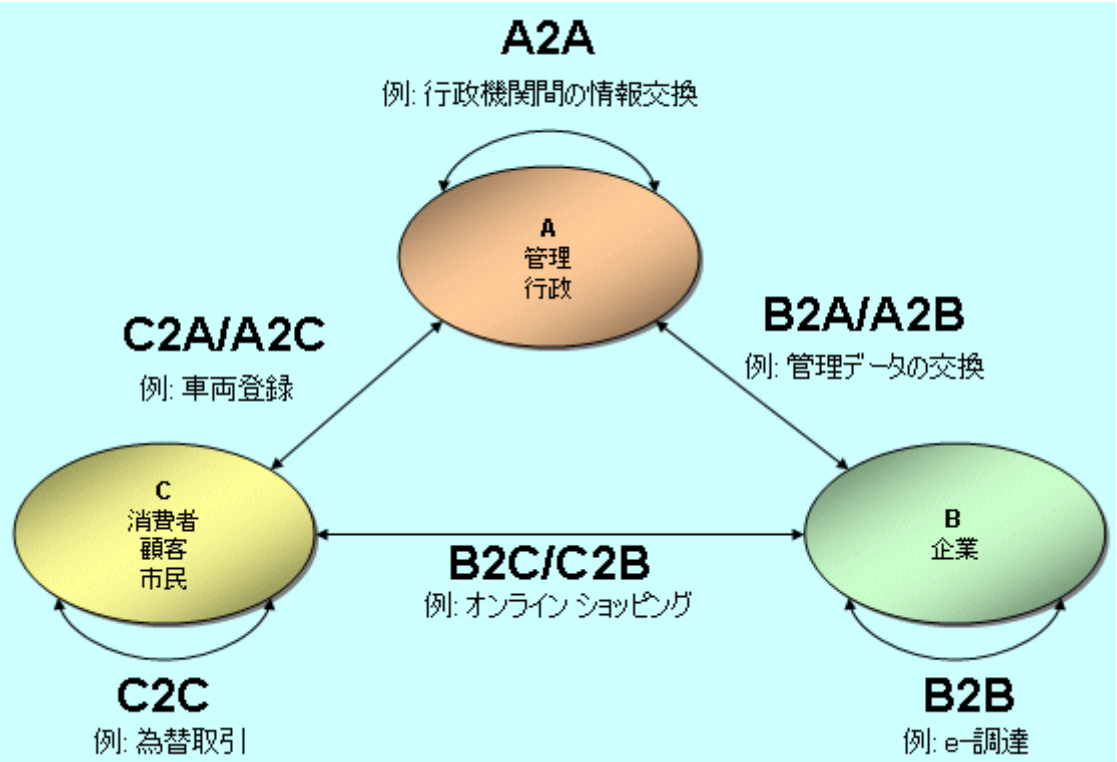


図 186: e-ビジネスで可能な取引

複数のパートナーの関係で区別するだけでなく、1 対 1、1 対多、多対多など、パートナー同士の関係の中で区別することもできます。 特に、市場シナリオは非常に重要です。

市場

電子市場とは、仮想の場所です。ここでは多数の人々によって情報が交換され、製品やサービスの売買が公然と行われています。

e-ビジネス シナリオ図は、このような e-ビジネスをサポートするために開発されました。 ARIS で提供されるその他のメソッドやさまざまなコンポーネントと併用することで、e-ビジネス プロジェクトの実装を最適な形でサポートできるようになります。 e-ビジネス シナリオ図を扱うこの章では、まずメソッドについて、すべてのオブジェクトおよび評価オプションを併せて説明し、次に、ほかのメソッドとの相互関係について説明します。 この章の最後では、ユース ケースを使用して、複雑な具体例を示します。

10.2 e-ビジネス シナリオ図手法

10.2.1 考え方

付加価値連鎖の全体を表示できること、つまりエンド ユーザーからプロセスに関与する各企業までを表示できることにより、最適化の可能性を引き出す基礎が提供されます。 目的としては、たとえば、サプライ チェーンの改善、調達コストや配布コストの低減、または情報システムのアーキテクチャの最適化などがあります。 e-ビジネス シナリオ図表現を使用すると、指定された目標を達成するために調査すべき内容をビジュアル化できます。 列で表示することにより、非常に多様なプロセス パートナーを結ぶ接点が列の枠線によって抽象化された形式で表示されます。 さまざまなレポートによってモデルが補足され、重要な分析機能が提供されます。

10.2.2 モデルおよびそのオブジェクト

モデルの中で使用されている経済主体はヘッダーに表示され、「取引先」として示されます。 これらは組織ビューから生成され、組織図をアサインすることができます。これにより、たとえば、企業構造や各列のオブジェクト間の関係をより詳細に説明することができます。

プロセス全体に関わる経済主体の個々のプロセスおよびプロセス間の接点が、モデルの中心的な構造に関連するオブジェクトです。 個別プロセスは、企業間の連係に緊密に関わるビジネス プロセスです。 このような個々のプロセスのより正確な表現と分析を得るには、プロセス モデルをアサインします。 企業で実行されているすべてのプロセスが、取引先の下の行の同じ列にモデリングされます。 企業間の連携には、各プロセスをサポートするためにさまざまな経済主体で使用されているアプリケーション システムとハードウェアを正確に分析することも必要です。このようなシステムとしては、ERP システムなどがあります。 これらは、[ビジネス コンポーネント] として表されます。 多様なこれらのコンポーネントを調整するには、システムの責任範囲を正確に指定する必要があります。 この目的のために、[組織ユニット タイプ] オブジェクトが用意されています。 プロセスに関係する従業員の役割も定義できます。 これらは、モデルでは [従業員の役割] と呼ばれます。 e-ビジネス モデリングでは、接点の統合は特別な要件です。 このコンテキストでは、列の枠線はプロセスの参加者同士を結ぶ接点を象徴するため、非常に重要です。 これらは、さまざまな観点から表示できます。

重要なポイントの 1 つは、プロセス固有の情報の転送です。 これは、XML 形式または HTML 形式が可能な、[ビジネス ドキュメント] の目的です。 ビジネス ドキュメントは、文書タイプの定義などのデータ ビュー モデルにアサインできます。 また、金銭や商品の流れを [金銭取引] オブジェクトや [商品発送] オブジェクトを使用して表示できます。

さらに重要な観点として、データのセキュリティ、特にインターネットを介した電子決済の安全性を確保する必要があげられます。これに対応するために、SET (Secure Electronic Transaction) や SSL (Secure Socket Layer) などのさまざまな符号化技術が使用されます。 セキュリティ要件は、[セキュリティ プロトコル] オブジェクトを使用して考慮できます。 トランザクションのセキュリティの責任者も定義することができ、これは [組織ユニット タイプ] で表すことができます。 さらに、より技術的な側面の分析、つまり接点でのデータ転送の技術設計に焦点を当てることができます。 この目的のため、モデルはさまざまな情報媒体を使用します。 個別プロセスは [イントラネット]、[エクストラネット] または [インターネット] を介して接続できます。 データ転送は [電子メール] を介して行われます。 携帯電話も転送媒体として普及してきています。

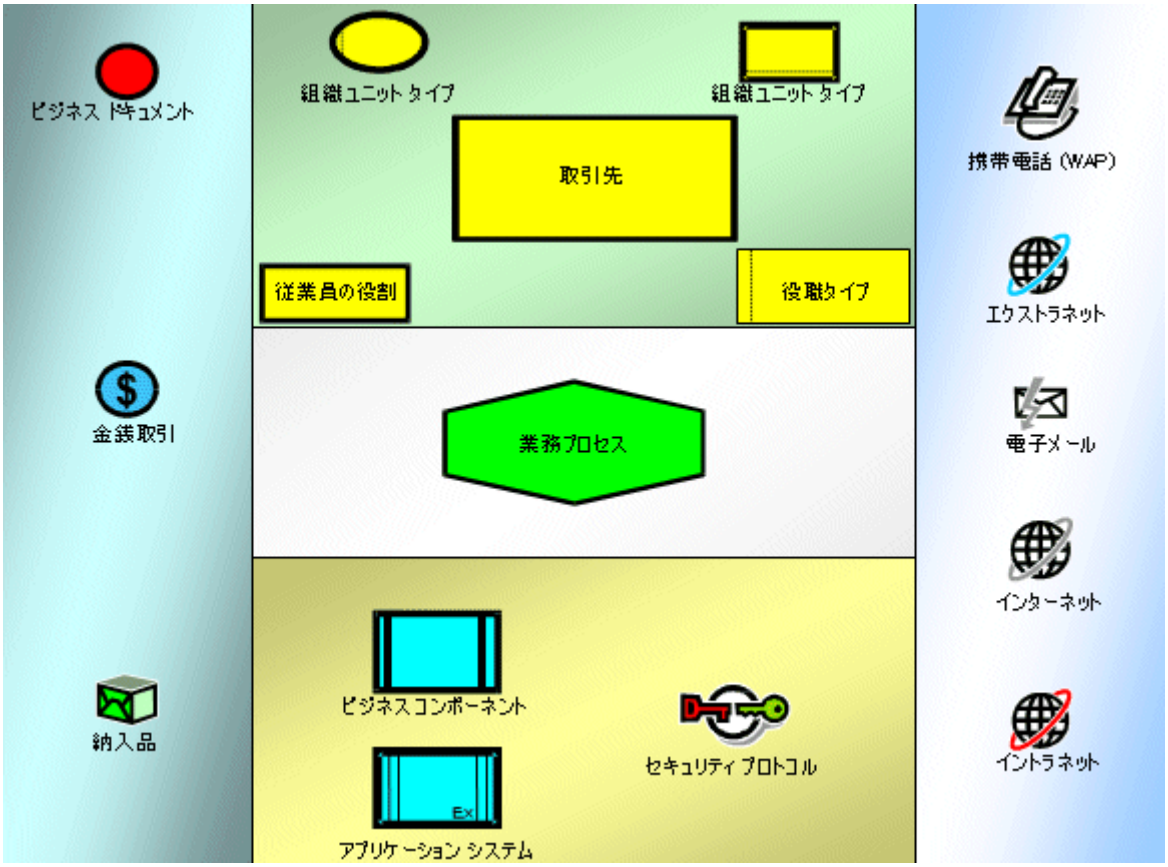


図 187: e-ビジネス シナリオ図のオブジェクト

10.2.3 [転送タイプ] 属性グループ

オブジェクトの属性を使用して、さらにモデリングされたオブジェクトを指定できます。 関連する属性は、特に、e-ビジネスの要件に合うように調整されています。

また、[ビジネス ドキュメント] オブジェクト、[金銭取引] オブジェクト、[商品発送] オブジェクトの [転送タイプ] 属性グループには注意が必要です。 転送タイプ属性に値を指定することにより、転送経路が識別されるだけでなく、取引を保護する必要も表現されます。 たとえばオンライン転送の場合、機密情報や機密データに対する上記の安全性の確保を怠らないことが重要となります。

10.3 レポートを使用した評価

e-ビジネス シナリオのモデリングをサポートするために、多数の評価オプションが提供されています。 これらの評価は、レポートの形式で行われます。 ARIS には事前に定義された多数の評価レポートが用意されていますが、ユーザー定義のレポートを作成することも可能です。 e-ビジネス シナリオには、次のレポートが事前に提供されています。

10.3.1 データ セキュリティのチェック

オンラインで転送されたデータのセキュリティは、e-ビジネスの受容に影響する最も重要な問題の 1 つです。 権限を持たないユーザーのアクセスから個人データや支払い情報を保護することは、顧客や取引先からの信頼の失墜を防止するために解決しなければならない問題です。 レポートを使用すると、交換されたすべての製品やサービス (金銭取引と商品発送) だけでなく、関連するすべてのデータ (ビジネス ドキュメント) もチェックできます。 前述の [転送タイプ] 属性グループが評価され、オンライン転送の際には、データの符号化を行うかどうかを決定するためにこの属性グループが確認されます。 このようにして、潜在的なセキュリティの穴と古くなって有効でなくなった符号化手法を特定し、排除することができます。

10.3.2 システム サポート

e-ビジネス プロジェクトで次に重要な側面は、アプリケーション システムを調和させることです。 この場合、企業は多くの問題点を考慮に入れる必要があります。 どのシステムがどのプロセスをサポートする必要があるのか。 どのユーザーが、どのシステムの操作に対して責任があるのか。 トレーニングで費用の発生する場所はどこか。 どの既存システムで、調整が必要なのか。 これらの疑問に対する解答も、レポートに表示されます。 個別のプロセスが、関連するシステムとそのシステムに対して責任のあるユーザーとともに一覧されます。

10.3.3 情報フロー

ほかのプロセス モデルとは異なり、e-ビジネス シナリオでは取引に焦点が当てられます。 交換中のデータと製品/サービスには、特別な注意が払われます。 したがって、データやサービスの交換を監視するための評価が提供されます。 重要な問題は、どのようなデータや製品/サービスがどこで生成され、どこで使用されるかということです。 必要な情報は、モデリングされたデータや製品/サービス、およびそれらを入力または出力として含むプロセスを出力するレポートを実行することにより取得できます。

10.3.4 協調ビジネス マップ

SAP で使用される協調ビジネス マップは、モデルの特別なタイプを構成するものです。 このマップにより、さまざまなパートナーのビューが強調されます。 このマップには、異なるターゲット グループに焦点を置く 2 つのビューがあります。 経営陣向けである「集約ビュー」には、ビジネス パートナーとプロセスのみが含まれます。一方、事業部門向けである「詳細ビュー」には、ドキュメントとプロセス担当者の役割が含まれます。 レポートを利用すると、e-ビジネス シナリオ図に表示されている情報を、両方のビューにいつでも転送できます。

10.4 ほかのメソッドおよびコンポーネントへの接続

ARIS Architect のさまざまなモデリング手法を使用すると、情報をさまざまな視点で見ることができ、特定の目的でターゲット グループが利用できるようになります。 e-ビジネス シナリオは、これらのビューの出発点です。 そのオブジェクトに特定のターゲット グループの詳細を指定できます。 このようにして、1 つの e-ビジネス プロジェクトを完全に表現できます。 さらに、ARIS のさまざまなコンポーネントを使用してモデルの評価を行うことができます。これらの評価により、e-ビジネス環境におけるプロジェクトに対する最適なサポートが可能になります。

例: オンライン ショップの作成

まず、計画した e-ビジネス活動で達成する目標を定義します。 この手順は、ARIS コンポーネントのバランス スコアカード (「バランス スコアカード メソッド『169 ページ 』」の章を参照) を使用して行うことができます。 これにより、目標を達成するために最適化が必要なプロセスを特定できます。 ここでの例では、新しい流通手段、つまりインターネットに目標を定めます。 この新しい経路を最適な状態で進んでいくためには、正確な文書化と計画が不可欠です。

プロセス フローそのものを考慮するだけでなく、責任者の組織や、さまざまなシステムを結ぶインターフェイス、および、データ保護も考慮する必要があります。

出発点となるのは、e-ビジネス シナリオ図です。 例に示す取引先は、ショップ システムを実装する企業と、そのサービスを使用する顧客です。 ショップに入ってからショップを出るまでのプロセス全体は、いくつかのサブプロセスに分割されます。 この表現では、顧客のビューと企業のビューが示されます。 e-ビジネス シナリオ図は、実装プロジェクトへの出発点として機能します。 次の図に、プロセス全体を分割する方法を示します。

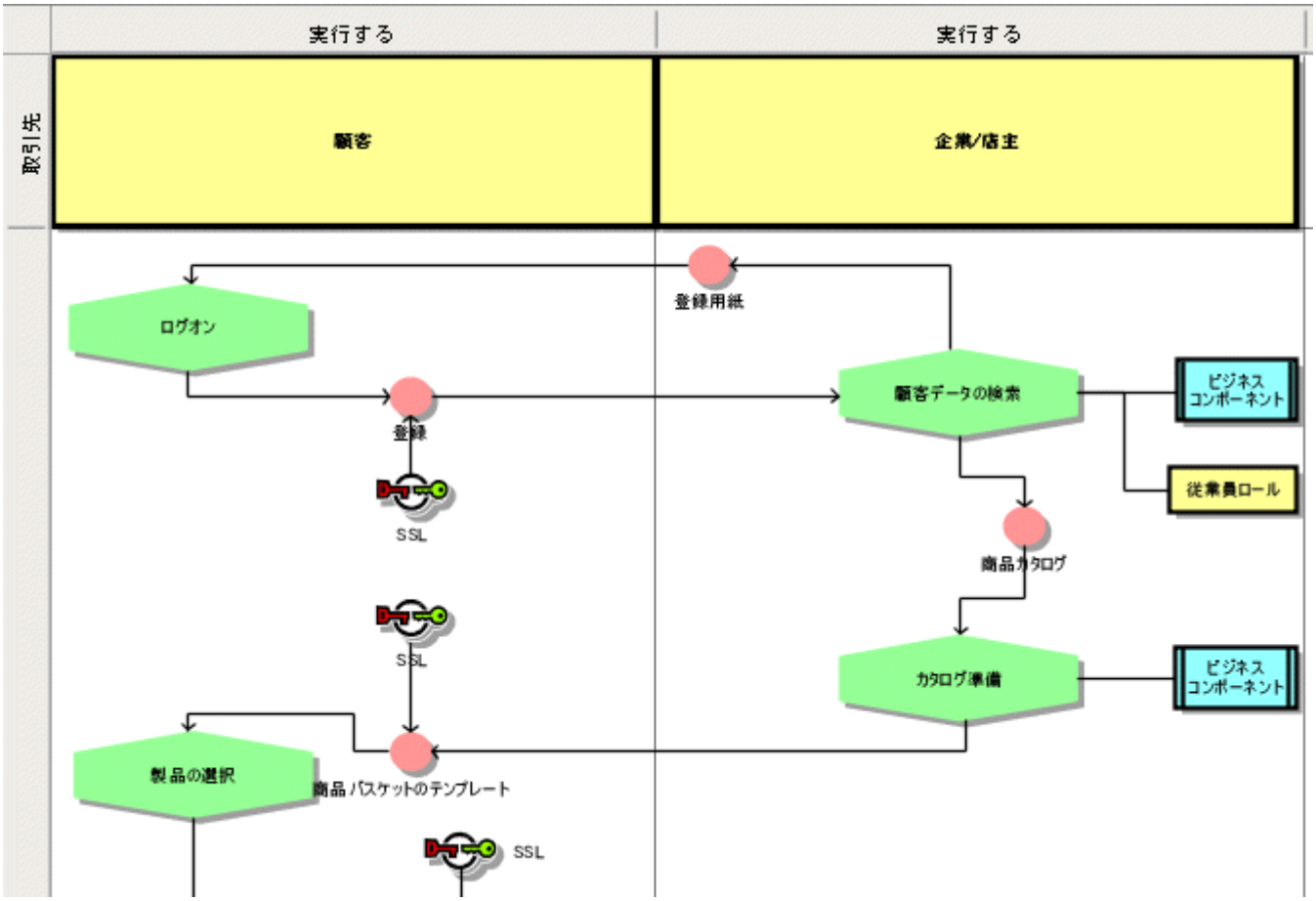


図 188: e-ビジネス シナリオ「オンライン ショップ」からの抜粋

EPC を使用して、さまざまなプロセス ステップを詳細化できます。たとえば、シミュレーション コンポーネントで検証したり、パイプライン図を使用して最適化した結果を表示したり、INTERSHOP enfinity を使用して最終的なショップ システムに変換し、さらに改良を加えたりできます。

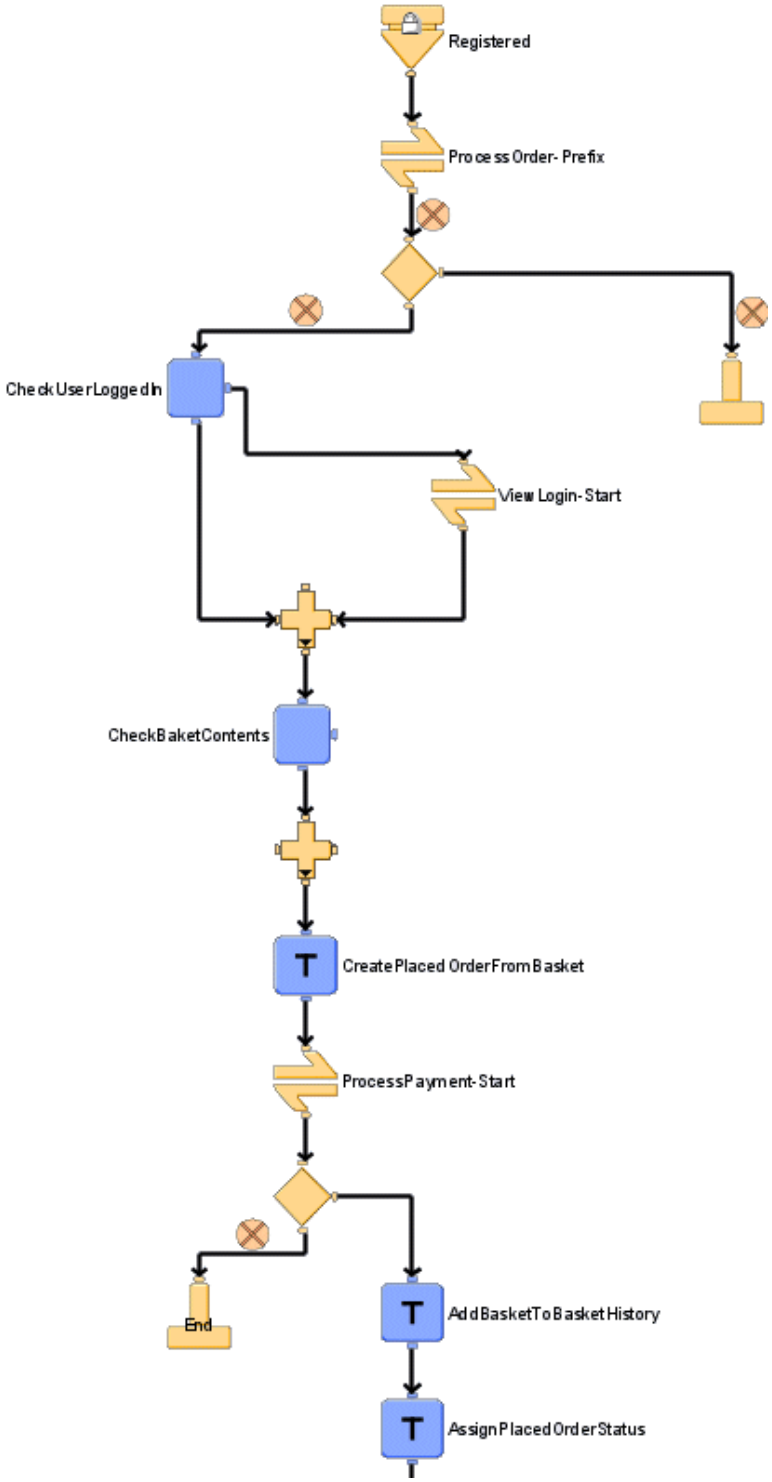


図 189: パイプライン図からの抜粋

ショップが継続して外部の ERP システム (Enterprise Resource Planning System、経営資源利用計画システム) に接続されている場合、データを適切な形式で転送する必要があります。 データ転送を行うには、さまざまなドキュメントとデータの標準化方法があります。 このような標準化の方法の 1 つとして、XML (Extensible Markup Language) を使用する方法があります。 ドキュメントの作成段階で、文書構造と内容を定義するための DTD を割り当てることができます。 XML はさまざまな経路で開発されている言語であるため、ここでは統一された基盤も必要になります。 XML の標準化には、企業と科学機関で構成される組織など、さまざまな組織が関わっています。

標準化された XML ドキュメントを使用することは、ERP システムの統合を容易にします。

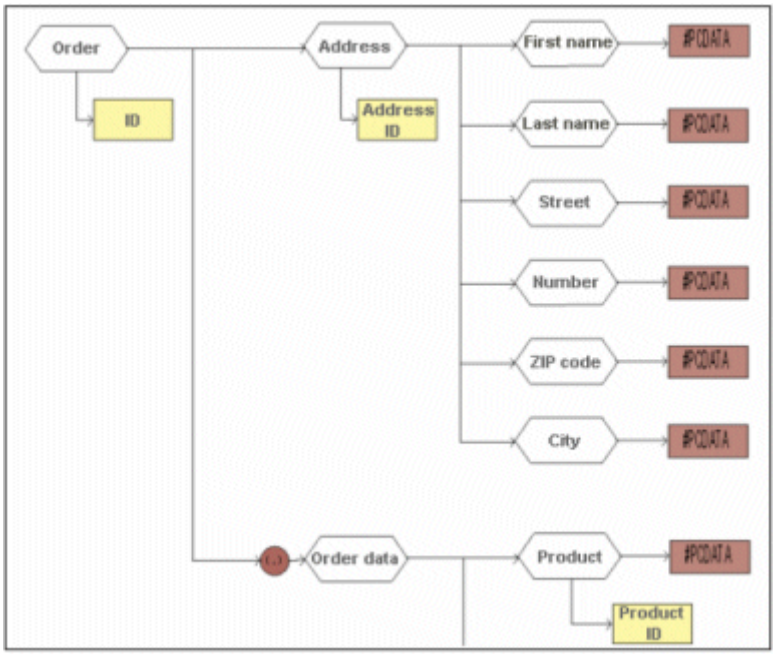


図 190: DTD からの抜粋: 順序

さまざまなコンポーネントを互いに調和させる必要性から発生した問題については、すでに説明しました。 アプリケーション システム タイプ図を使用すると、システムを視覚化することができ、モデルとしてアプリケーション システムやビジネス コンポーネントにアサインして、システムの相互関係を詳細に説明することができます。

新しい e-ビジネス活動は、組織構造にも影響を及ぼします。 新しい責任を定義したり、新規の割り当てを作成したりすることが必要になるかもしれません。 e-ビジネス シナリオ図では、プロセスの各手順に関して、役割と組織ユニットについての説明を記述できます。 企業の組織やプロセスでの役職は、組織図を使用して分析できます。

実装は、モデリングされた内容を実現することから始まります。 Intershop メソッドが使用される場合、モデリングされた内容は Intershop Enfinity により、実際に利用可能なシステムに変換されます。

11 IT 都市計画

11.1 エンタープライズ アーキテクチャおよび IT 都市計画

IT 都市計画は、1980 年代にフランス人 Jacques Sassoon により開発されたアーキテクチャ アプローチです。 IT 都市計画の目的は、システム内のアプリケーション間における情報の交換などのインタラクションを徹底的に分析することにより、多相システム ランドスケープに調和をもたらすことにあります。

都市計画に使用されるアプローチに基づく IT 都市計画設計のプロシージャは、現在だけではなく、過去 (レガシー システム) および将来の側面を考慮する、長期的で戦略的な IT 管理を有効化する考えにより推進されます。

ただし、システムをすべて再設計する必要はありません。 かわりに、プロジェクトごとの段階的なアプローチを使用します。

ARIS 製品がサポートするモデル駆動アーキテクチャ (MDA) と同様に、IT 都市計画のテーマには、技術関連情報への参照を必要とすることなく、情報システムを記述するモデルを使用して取り組むことができます。 IT 都市計画は UML を必要としないので、技術的な知識を豊富に持たないユーザーもこのテーマに容易に精通でき、受容範囲が広がります。

11.2 IT 都市計画を有効に使用できる企業

IT 都市計画は以下の企業を対象とします。

- 多種多様なアプリケーションを使用している企業
- 情報技術を長い期間にわたり使用している企業
- 情報技術に戦略的重要性がある企業
- 合併に関係している企業

IT 都市計画の目標:

- 今後起こりうる重複を回避するため、ソフトウェア リソースを再利用する。
- 「ブロック別」オーバーホール、および既存のリソースに代わるさまざまなユース ケースに対応できる新規ソフトウェア リソースを定義することで、メンテナンスのコストを削減する。
- 情報システムのコンソリデーションを行なう。
- より高度なレベルにおける EAI ソフトウェアの配置を準備する。

IT 都市計画を作成する責任は、統合コンピテンス センタが担います。 計画は、設計パターン、情報アーキテクチャおよび技術アーキテクチャの両方に対応している必要があります。

EAI = Enterprise Application Integration (エンタープライズ アプリケーション インテグレーション) の略。 企業全体にわたるアプリケーションの統合。 EAI により、e-ビジネス インフラストラクチャが提供されます。 EAI ソフトウェアは、e-ビジネス戦略を導入するための前提条件として必要な、技術ミドルウェアです。

11.3 ARIS を使用した IT 都市計画

ARIS には、情報システムの次のビューが含まれます。

- データ ビュー
- ファンクション ビュー
- 組織ビュー
- 製品/サービス ビュー
- プロセス ビュー

これらのビューは、それぞれ [要件定義]、[仕様設計]、および [実装] という 3 つの説明レベルに分かれます。 説明レベルは、情報システムのライフ サイクル、および情報技術との関連性に基づきます。

仕様設計と実装のレベルでは、基本的にソフトウェア システムを記述します。 これらのレベルの概念と用語には密接な相互関係があり、その「変換」に問題はありません。

しかし、要件定義レベルと仕様設計レベルの間の変換は違います。 仕様設計を作成するときは、ビジネス管理ビューを標準ソフトウェアに適合する必要があります。 そのためには、ビジネス管理のノウハウとデータ処理の知識が必要です (A. W. シェアー著 :『ARIS - ビジネスプロセスフレームワーク』1998 年、第 3 版、p. 7 参照)。

このとき、IT 都市計画の情報システム ビュー (IS ビュー) をレベル間の媒介として使用できます。 ARIS では、IS ビュー内のオブジェクト タイプをファンクションとアプリケーション システムの間のレベルに配置することで、ファンクション ビューが拡張されます。 ファンクションと同様に、IS 要素は、ARIS ハウスのビューのさまざまな構造に関連しています。 これらの拡張は、主にプロセス ビューとデータ ビューに関連します。 以下では、IS ビューという用語は、IS 要素間の関係、またはほかの ARIS ビューのコンテキストに示される IS 要素を詳細に記述する、ARIS ハウスのファンクション ビューとプロセス ビューのモデル タイプを指します。

アプリケーション システム タイプ、IT ファンクション タイプ、およびソケットは、IT (情報関連技術) ビューの要素として考えます。 IS ビューと同様に、IT ビューには、アプリケーション システム タイプ、IT ファンクション タイプ、および新しい [ソケット] オブジェクト タイプの間の関係を記述する、またはこれらの要素の詳細を記述するために使用されるモデル タイプがすべて含まれます。

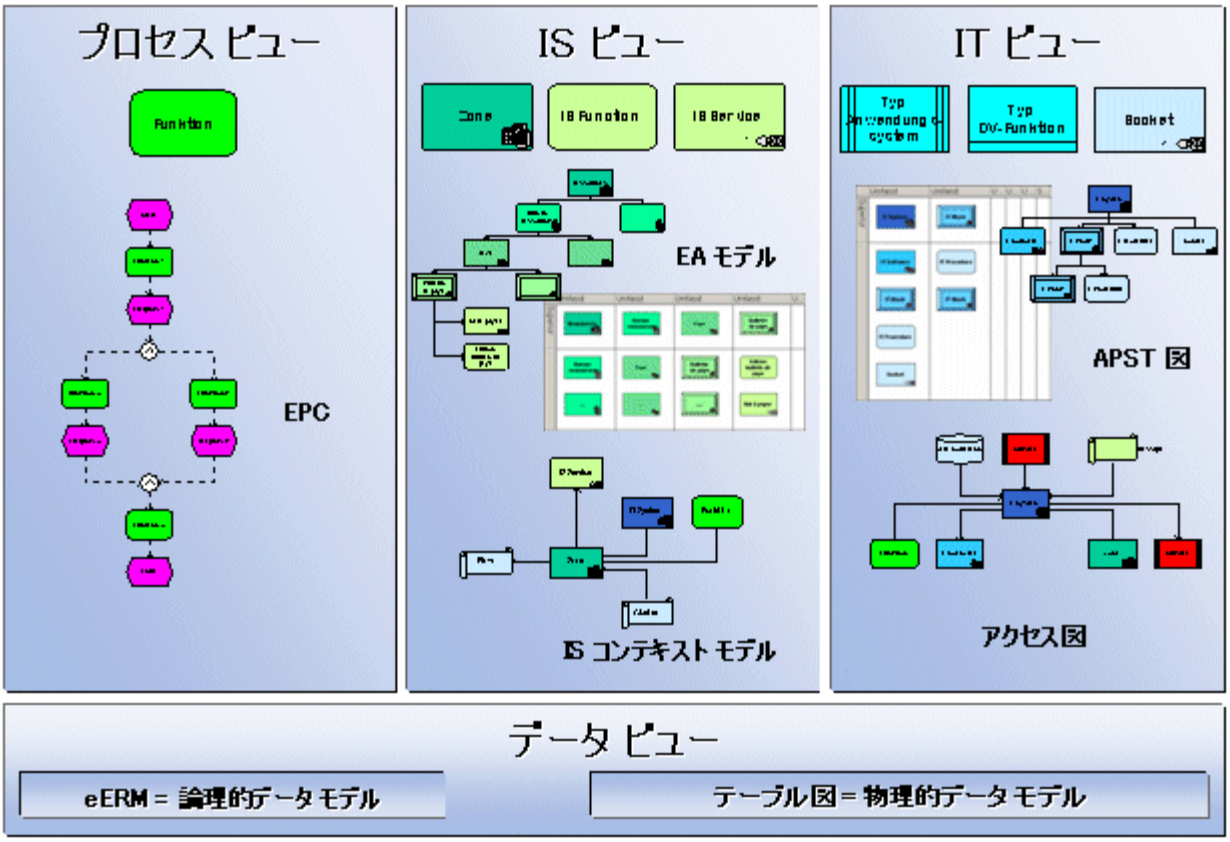


図 191: プロセス ビュー、IS ビュー、IT ビュー

11.4 IS ビュー

ARIS には、IS ビューを記述できる次のモデル タイプが含まれます。

- サービス アーキテクチャ図
- サービス アーキテクチャ図 (列表示)
- サービス割当図
- サービス コラボレーション図

この 2 つのサービス アーキテクチャ モデルにより、情報システムの構造要素が階層的に配置されます。

IS 階層には、次のレベルが含まれます。

- ゾーン
- エリア
- 構築クラスター

- 機能ブロック
- 能力
- IS サービス
- ビジネス サービス

[ゾーン]、[エリア]、[構築クラスタ]、および [機能ブロック] は、[サービス タイプ] のオブジェクト タイプです。 [サービス タイプ] は、情報システムを機能ごとに独立したユニット/ブロックに整理するために使用されます。

各サービス タイプは、それにより使用されるデータの「所有者」であり、関連する処理方法です。 その他のサービス タイプは、「所有者サービス タイプ」のサービスが呼び出される場合にのみ、これらのデータと処理方法にアクセスできます。

サービス タイプ内では、類似データが使用され、同一アクティビティおよび業務ファンクションが実行されます。

情報システムは、最上レベルでゾーンに分割されます。 [ゾーン] は、たとえば、業務および開発分野に対応します。

次の図は、企業の情報システムをどのように細分化できるかを示しています。

図 192: 企業の情報システムのゾーン

各 [ゾーン] には、1 つまたは複数の [エリア] が含まれます。

[ゾーン] 内の [エリア] は、プロセスと時間的特性 (たとえば、類似したライフサイクルおよび情報処理サイクル) に関して、多くの類似点により特徴付けられます。 エリアには、支払い条件または価格条件、人材管理、出張ガイドラインなどがあります。

「リソース」ゾーンには、「人材」エリアと「財務」エリアが含まれます。

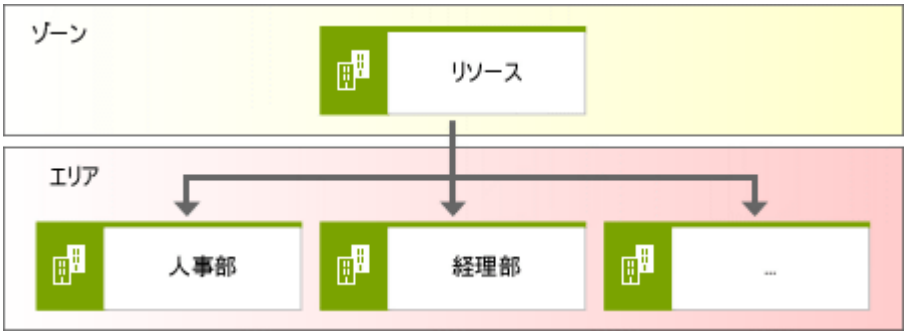


図 193: エリアに分割されたゾーン

エリア内には、給料の支払いまたは請求書の処理など、機能目的を持つ 1 つまたは複数の構築クラスターが存在します。

「人材」エリアには、「ビジネス ユニット管理」、「採用」、「人材開発」、「人材サービス」といった構築クラスタが含まれます。

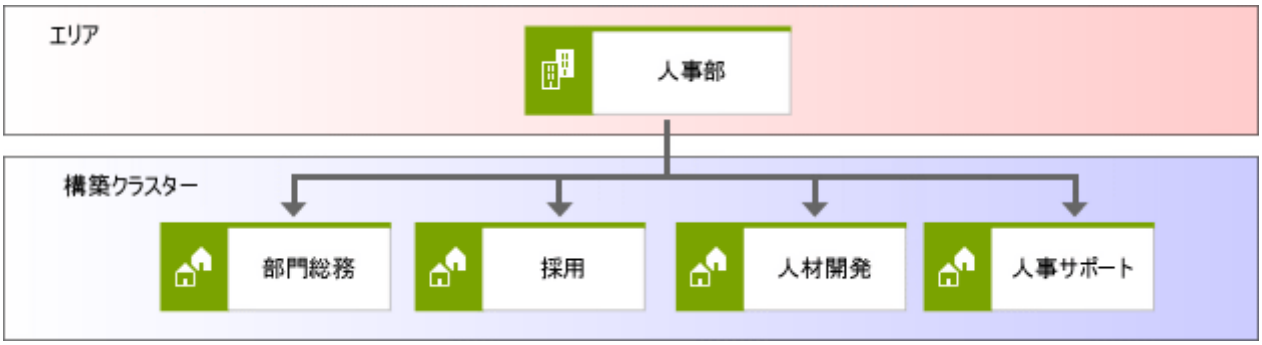


図 194: 構築クラスタに分割されたエリア

各 [構築クラスター] は、1 つ以上の [機能ブロック] を含むことができます。 [機能ブロック] は、それにより管理されるビジネス オブジェクトとイベントに関して、多くの類似点により特徴付けられます。

[機能ブロック] は、独立した再利用可能な機能コンポーネントです。 機能と IS サービスは結合して、次のルールに従った機能ブロックを形成します:

- 管理されるオブジェクトとサポートされる機能間には密接な相互関係があります。
- ほかの機能ブロックとは、最小限のやりとりのみが行われます。

上の例の構築クラスタ「人材サービス」には、機能ブロックの「マスタ データ管理」、「評価」、「管理」、「給与」が含まれます。

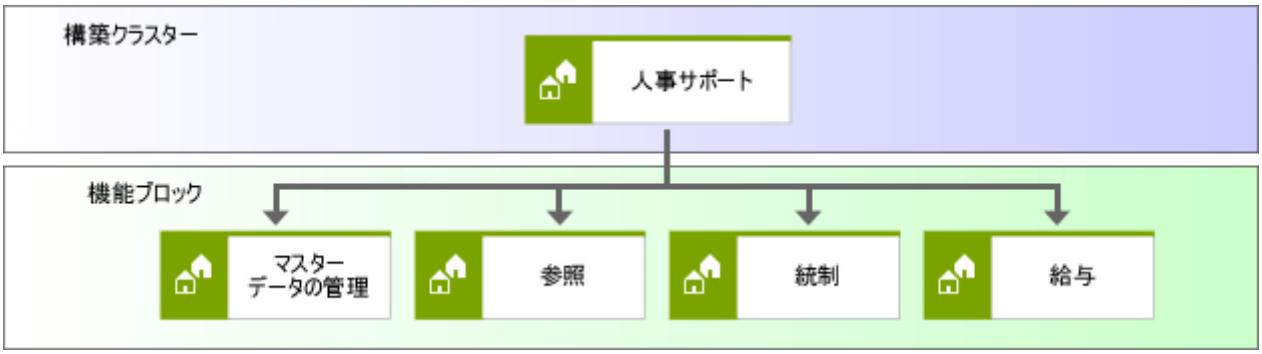


図 195: 機能ブロックに分割される「人材サービス」構築クラスタ

能力は、システム内の基本的なファンクションです。 プロセス内のアクティビティの実現がサポートされます。

IS サービスは、サービス タイプまたは能力のインターフェイスです。 また、[IS サービス] によって、サービスを提供する IS 要素のデータおよび処理方法に対する、ほかの IS 要素からのアクセスを制御できるようになります。

これらのインターフェイスは、IS ビューのほかの要素とのメッセージの交換に使用できます。

次の図は、「給与」機能ブロックの能力と IS サービスを示しています。

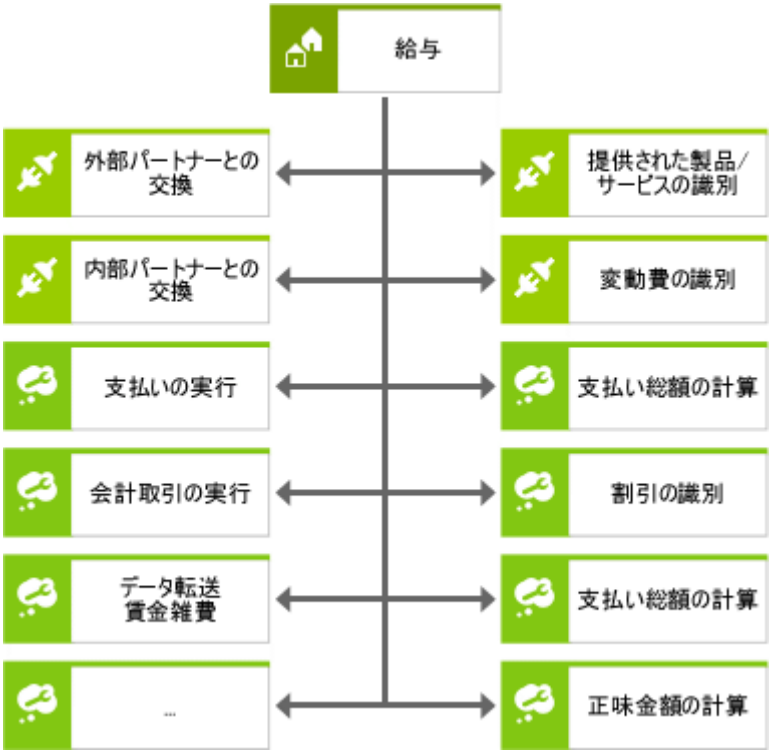


図 196: 「給与」機能ブロックの機能と IS サービス

IS 階層の記述に関しては、ここに記述されるすべてのレベルを完全にモデル化する必要はありません。「能力」および「IS サービス」IS 要素は、IT 都市計画では、都市計画の要素とは見なされません。都市計画担当者のタスクは、構築ブロックレベルまでです。能力および IS サービスは、設計者の担当です (Longépé, Christoph 著: 『Le projet d'urbanisation du système d'information』、18 ページ参照)。

11.5 サービス タイプとそのデータ

eERM を使用して、サービス タイプと能力のどちらにデータが属するか記述します。 都市計画手法のコンテキストでは、eERM によって IS ビュー シンボルが提供されます。 [所有者である] タイプの接続線は、関係とエンティティ タイプを持つオブジェクトを接続するために使用できます。

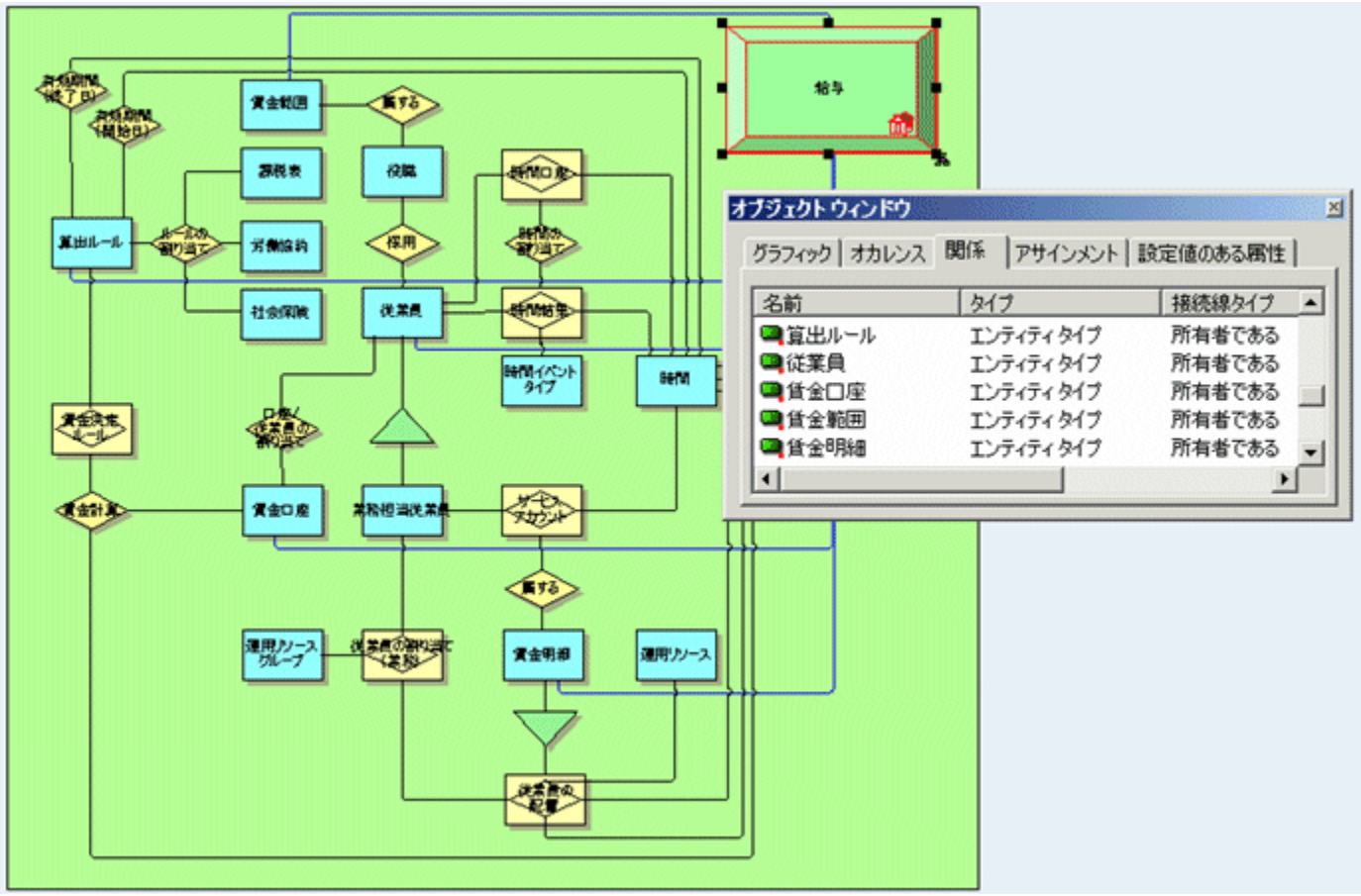


図 197: IS ビュー、関係、およびエンティティ タイプの間の [所有者である] 接続線

11.6 サービス タイプの詳細な説明

情報システムのサービス タイプと能力に関する詳細な説明を、サービス割当図で行うことができます。 これには、次の要素が含まれます。

- ブロックのインターフェイス
- ブロック間のインタラクション
- ブロックをサポートするアプリケーション システム
- ブロックによりサポートされるビジネス管理ファンクション

ゾーン、エリア、構築クラスタ、機能ブロック、および能力は、[提供する] 接続線を使用して IS サービスに接続することができます。

入出力接続線を IS 要素とデータ要素間に描画して、サービス タイプ間の情報フローを記述できます。

異なるアプリケーション システムと IT ファンクション タイプのオブジェクトを [サポートする] タイプの接続線を使用して IT ビューのオブジェクトに割り当てることができます。 都市計画を街の開発計画として解釈すると、この接続線によりどの情報システム領域がどのアプリケーション システムにより「占められている」かに関する情報が提供されます。 また、[サポートする] 接続線も IS 要素とファンクション間で使用することができます。

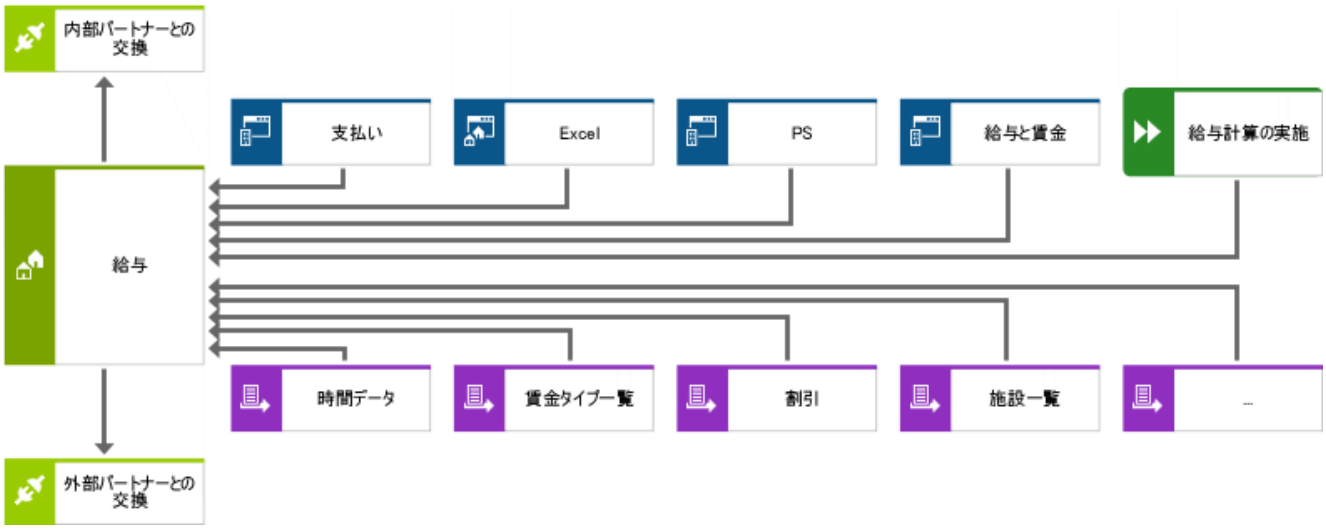


図 198: IS 要素およびファンクション間の [サポートする] 接続線

11.7 IS 要素の時系列の操作順序

サービス割当図では、組織ビュー、データ ビュー、およびプロセス ビューのオブジェクト タイプへの関係を、サービス アーキテクチャ図で指定されたサービス タイプ、能力、IS サービスに対して作成できます。 IS 要素の可能な時系列の論理的な操作順序は表現できません。

IT 都市計画では、サービス コラボレーション図を使用して IS 要素の時系列の操作順序を図示し、情報システム内の動的な側面を示します。 モデル タイプとしては、IT ビュー内のプログラム フロー図 (「プログラム フロー図 『123 ページ 』」の章を参照) と同等のものです。 イベントを使用して操作順序を表すことができます。 プログラム フロー図での IT 要素とイベントの割り当てと同様に、サービス コラボレーション図のイベントを使用して機能モジュールの順序を定義できます。 このコンテキストでは、イベントは、サービス タイプ、能力、または IS サービスをアクティブにするトリガーと見なされます。 分岐は、EPC またはプログラム フロー図で使い慣れたルールで表すことができます。 操作順序もイベントを使用せずに定義できます。

11.8 IT ビュー

IT ビューには、次のモデル タイプが含まれます。

- アプリケーション システム タイプ図
- アプリケーション システム タイプ図 (列表示)
- アクセス図
- プログラム フロー図

アプリケーション システムの階層

都市計画のコンテキストにおいて、企業内の現在のアプリケーション システムの階層は、アプリケーション システム タイプ図またはアプリケーション システム タイプ図 (列表示) を使用して表されます。アプリケーション システム タイプ図 (列表示) は、都市計画に必要なアプリケーション システム タイプ図のオブジェクト タイプ、シンボルおよび関係タイプを提供するレーン図 (列/行表示の図) です。

アプリケーション システム タイプ階層の次のレベルを表すことができます。

- IT システム
- サブシステム
- IT ソフトウェア
- IT ブロック
- IT プロシージャ
- ソケット

IT システム、サブシステム、IT ソフトウェアおよび IT ブロックは、[アプリケーション システム タイプ] オブジェクト タイプのシンボルです。階層は、[包含する] 関係タイプを使用して構成されます。

IT システムは、[アプリケーション システム タイプ 階層の最上レベルにあります。[IT システム] は、構造化された多数の IT 要素 (通常、サブシステム) で構成されています。[IT システム] の管理と操作は、指定した組織ユニットが担当します。

サブシステムは、[IT システム] のコンポーネントです。サブシステムのコンポーネントは、[IT ソフトウェア] といいます。

[IT ソフトウェア] により、同種の一連のファンクションがサポートされます。これは、ユーザー指向であり、1 つ以上の業務プロセスをサポートします。[IT ブロック] は、[IT ソフトウェア] のコンポーネントです。

原則として、[IT ブロック] には、同じデータ (データベース、テーブル、ファイルなど) にアクセスする [IT プロシージャ] が含まれます。

IT プロシージャは、[IT ファンクション タイプ] タイプのオブジェクトです。各 [IT プロシージャ] により、特定のファンクションがサポートされます。

ソケットは IS サービスに対応します。これは、ほかの IT 要素がその IT 要素のデータと処理方法にアクセスできるようにするために、IT 要素により提供されるインターフェイスです。

次の図は、DATEV システムのサブシステム構造の一例です。

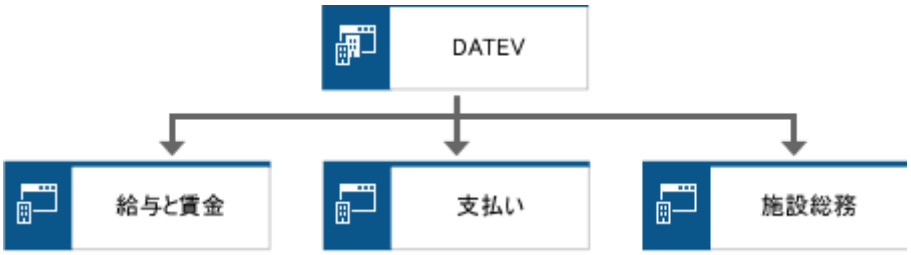


図 199: DATEV システムのサブシステム構造

11.9 IT 要素および IT 要素のデータ

eERM の IS 要素と同様に、IT 要素に含まれるデータを記述するアプリケーション システム タイプ、IT ファンクション タイプ、またはソケットと、エンティティ タイプまたは関係タイプとの間に描画できる [所有者である] 接続線を使用できます。

11.10 IT 要素の詳細記述

IT 都市計画の IT 要素は、アクセス図に詳細に記述されます。これは、IS ビューのサービス割当図に相当します。

詳細記述は、以下のアイテムに関してです。

- 該当する IT 要素の出入力関係
- サポートされている業務ファンクション
- サポートされている IS 要素
- 対象の要素による、ほかの IT 要素のアクティブ化
- IT 要素が稼動するプラットフォーム
- IT 要素のユーザー

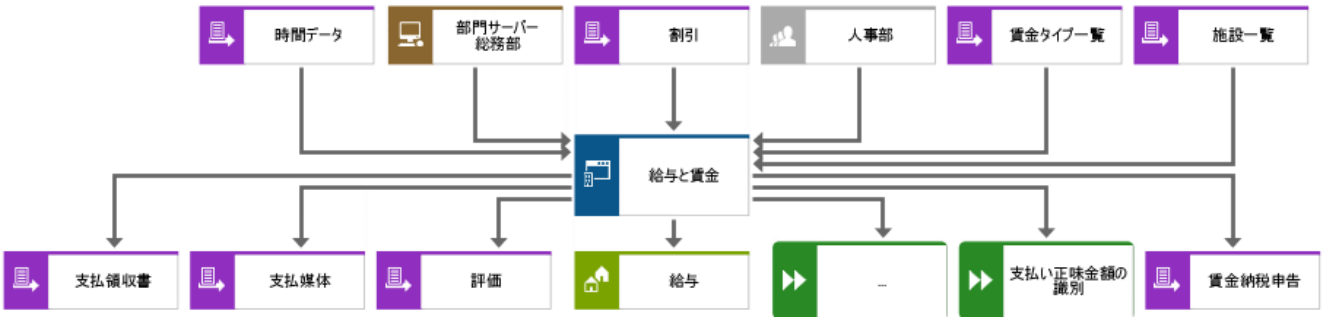


図 200: アクセス図の IT 要素の詳細記述

11.11 組織的側面

IT 要素の詳細記述には、組織ビューの情報を含めることもできます。 これには、どの組織的要素が、IT 要素のユーザーになりうるかなどの情報が含まれます。 ネットワーク図を使用して、IT インフラストラクチャの影響および効果を表すことができます。

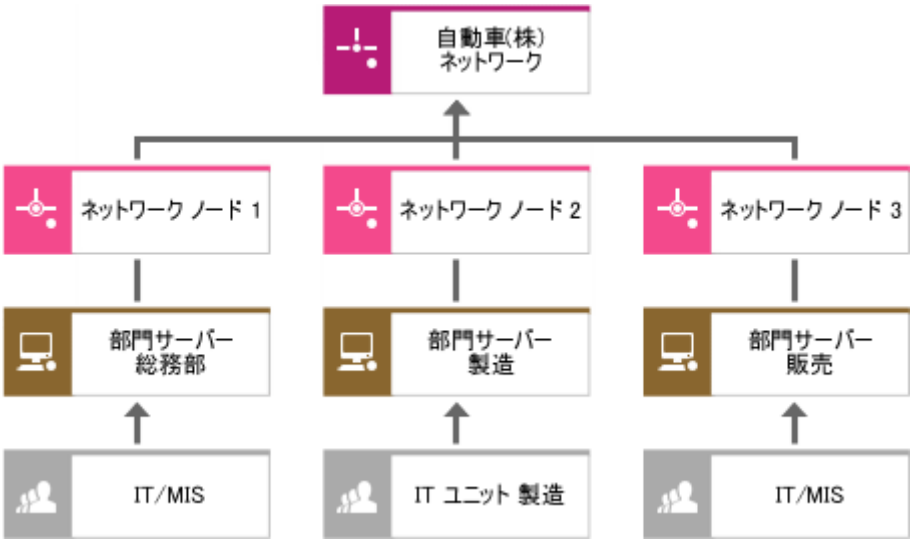


図 201: 技術的インフラストラクチャの影響および効果

11.12 IT 要素の時系列の操作順序

サービス コラボレーション図と同様に、プログラム フロー図では、[アプリケーション システム タイプ]、[IT ファンクション タイプ]、および [ソケット] の各 IT 要素の時系列の操作順序を記述します。

プログラム フロー図の詳細については、「システム インターフェイス モデル - システム属性、システム属性ドメイン 『64 ページ 』」の章を参照してください。

11.13 アーキテクチャ内の時系列の操作順序

さまざまなプロセス モデル (EPC のすべてのバリエーション) およびプログラム フロー図では、IS 要素および IT 要素をどのように時系列の操作順序で統合するか図示するために適したオブジェクトが利用できます。

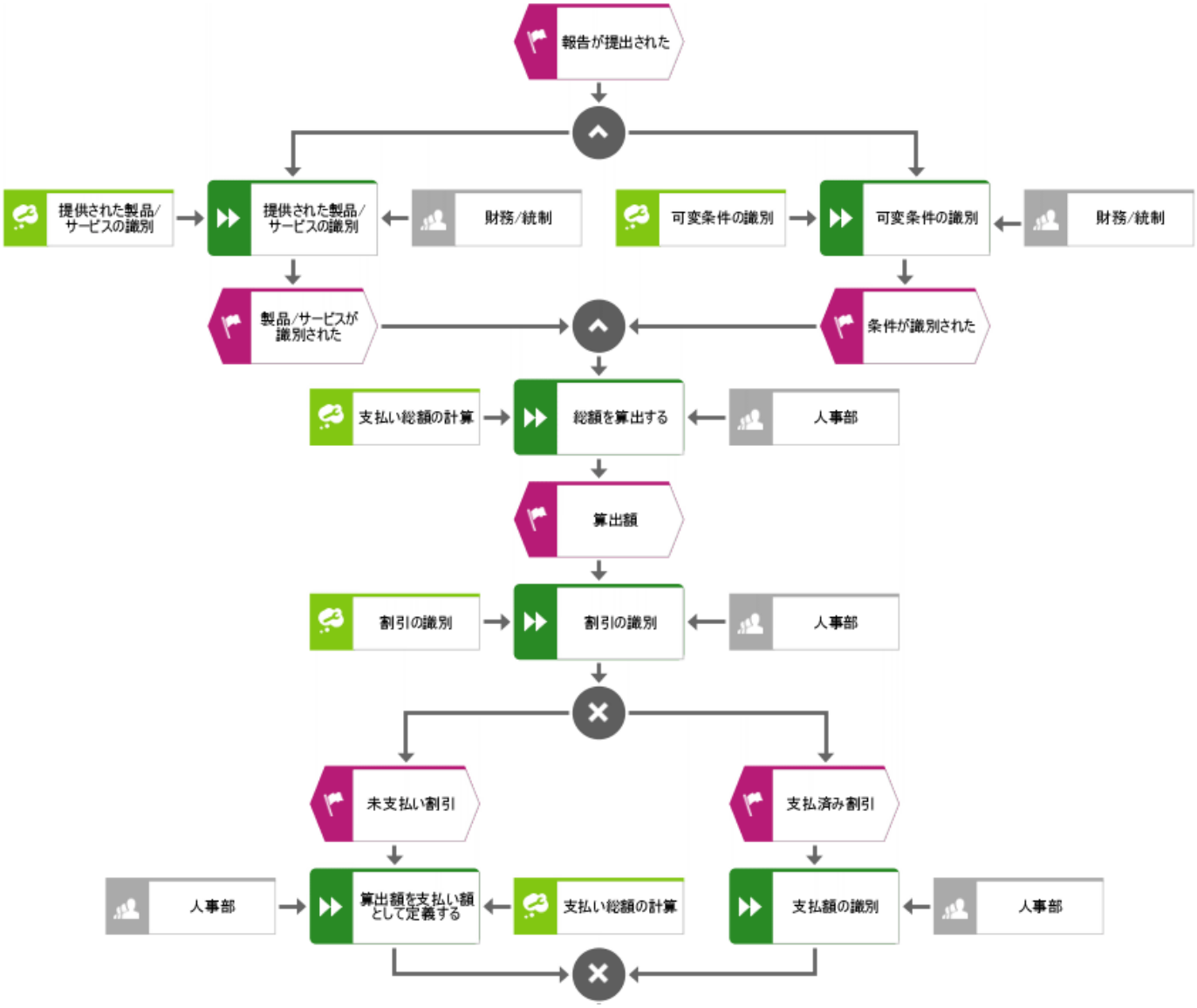


図 202: 時系列な操作順序への IS および IT 要素の統合

11.14 使用できる評価

上記の章で説明したモデリング オプションに基づいて、評価を利用して、次の項目に対する解答を得て、情報システムの構築に役立てることができます。

- 該当する IS 要素により管理されるデータは何か
- IS 要素をサポートするアプリケーション システムは何か
- 該当する IS 要素によりサポートされる機能は何か
- 該当する IS 要素の IT 要素により使用されるデータは何か
- 該当する IS 要素の IT 要素により生成されるデータは何か
- IS 要素により提供される IS サービスは何か、および IS サービスが使用されるプロセスは何か
- 特定の IS 要素のアプリケーション システムが稼動するハードウェア コンポーネントは何か

一部のアプリケーション システム タイプ、IT ファンクション タイプ、およびソケットには次の評価を使用できます。

- IT 要素により使用されるデータ
- IT 要素によりサポートされる IS 要素
- IT 要素によりサポートされるファンクション
- IT 要素によりサポートされる IS 要素に使用されるデータ
- IT 要素によりサポートされる IS 要素に作成されるデータ
- IT システムが稼動するハードウェア コンポーネント

12 業務プロセスのモデル化

会社とその取引先、納入業者、顧客の間のやり取りや取引は、これまで以上に複雑になっています。この傾向は新しい情報関連技術と通信技術によってさらに加速しています。 業務プロセスのさらなる開発と実践を進めるには、さまざまなビジネス パートナーと緊密に協力する必要があるという事実が次第に明確化しています。

企業は自身の活動とビジネス パートナーの活動をより把握することが望まれる一方、内部の変化と市場主導の変化にすばやく対応する能力も求められています。 標準化されたプロセス モデリング言語を使用することで、企業は業務プロセスを内部と外部の区別なく、明確で柔軟に記述できるようになります。 また、企業は、モデル化されたプロセスをビジネス パートナーに分かりやすく、明確に、適切な形で伝える必要があります。 つまり、すべての関係者が同じ「プロセス言語」を使用することが前提になります。

これらの目的を達成するために、Business Process Management Initiative (BPMI.org) は、標準化されたモデリング言語「Business Process Model and Notation (BPMN)」を提供しています。 BPMN は、業務プロセスを記述するためのグラフィック表記です。

表記は、すべてのユーザーが簡単に理解できなければなりません。 このため、業務プロセスの分析担当者やプロセスの監視/管理担当者だけでなく、プロセスを実行するための技術を実装する開発者も使用することができます。

さらに、業務プロセスを自動化するためには、この表記を使用して XML ベースの言語 (Business Process Execution Language for Web Service (BPEL4WS) など) を視覚化できるようにすることが重要です。

12.1 プロセス クラスと業務プロセス図

Business Process Model and Notation (BPMN) では、[業務プロセス図 (BPD)] モデル タイプを使用してプロセスを記述します。 このモデルでは、以下の 3 つの業務プロセス クラスとそれらの関係を表します。

- プライベート業務プロセス (Private business processes: 内部業務プロセス)
- 抽象業務プロセス (Abstract business processes: パブリック業務プロセス)
- コラボレーション プロセス (Collaboration processes: グローバル業務プロセス)

プライベート業務プロセスは、企業の内部のみで実行される業務プロセスです。 一般に、ワークフローや BPM プロセスとして知られています。

さまざまな内部業務プロセスは、相互作用がメッセージ フロー (Message flow) として表記される個々のプール内で、シーケンス フロー (Sequence flow) としてモデル化されます (「ARIS での BPMN の実装『207 ページ 』」の章を参照)。

BPMN は制御フローの代わりにシーケンス フローとメッセージ フローという用語を使用します。これはプロセスがイベントだけでなく交換されるメッセージによって制御されるからです。

抽象業務プロセスは、異なるプールのプライベート プロセス (Private process) 間の相互作用や異なるプールのオブジェクト間の相互作用、またはその両方を表すものです。 プライベート プロセス内のシーケンス フローの他に、個々のプロセス間のメッセージ フローは特に重要です。 相互作用はメッセージ フローを使用してモデル化されます。

抽象業務プロセスは個々のプールに組み込まれ、個別にモデル化したり、BPMN の全体図の一部としてモデル化できます。抽象業務プロセスが対応するプライベート業務プロセスと同じモデル内に発生する場合は、互いに関連付けることができます。

コラボレーション プロセスは、2 つ以上の業務エンティティ (ビジネス パートナー) の間の相互作用のみを表します。 さまざまなビジネス パートナーとのメッセージの交換パターンを反映して、アクティビティのシーケンスをモデル化できます。 シーケンス フローは、ここでは関係しません。

コラボレーション (Collaboration) に関連する言語としては、bXML BPSS、RosettaNet、W3C Choreography Working Group などがあります。 マッピング仕様は BPMN 仕様の今後のバージョンで公開される予定です。

コラボレーション プロセスはプールに組み込むことができます。 関連する取引先との相互作用はレーンでそれぞれ記述されます。これにより、プロセスを個別にモデル化したり、BPMN の全体図の一部としてモデル化できます。 コラボレーションが内部プロセスのいずれかと同じ図に発生する場合は、両者に共通するアクティビティを互いに関連付けることができます。

この 3 種類のプロセス クラスから、さまざまなタイプの業務プロセスを作成することができます。

- 上位レベルのプライベート業務プロセス
- 詳細レベルのプライベート業務プロセス (ターゲット プロセスまたは実際のプロセス)
- 詳細プロセスと外部エンティティ間で実行されているプロセス
- 詳細プロセス間で実行されているプロセス
- 詳細プロセスと抽象プロセス間で実行されているプロセス
- 詳細プロセスとコラボレーション プロセス間で実行されているプロセス
- 抽象プロセス間で実行されているプロセス
- 抽象プロセスとコラボレーション間で実行されているプロセス
- コラボレーション間で実行されているプロセス
- 抽象プロセスを介して相互作用する、複数の詳細プロセス間で実行されているプロセス
- コラボレーション プロセスを介して相互作用する、複数の詳細プロセス間で実行されているプロセス
- 抽象プロセスとコラボレーションを介して相互作用する、複数の詳細プロセス間で実行されているプロセス

以下の図は、別のプロセスがそれぞれ割り当てられている 2 つのビジネス パートナーの BPMN コラボレーション図の例です。どちらの詳細プロセスも、開始イベント、アクティビティ、シーケンス フロー接続線、終了イベントで構成されています。 2 つの詳細プロセスのアクティビティ間には、メッセージ フロー接続線が描かれています。

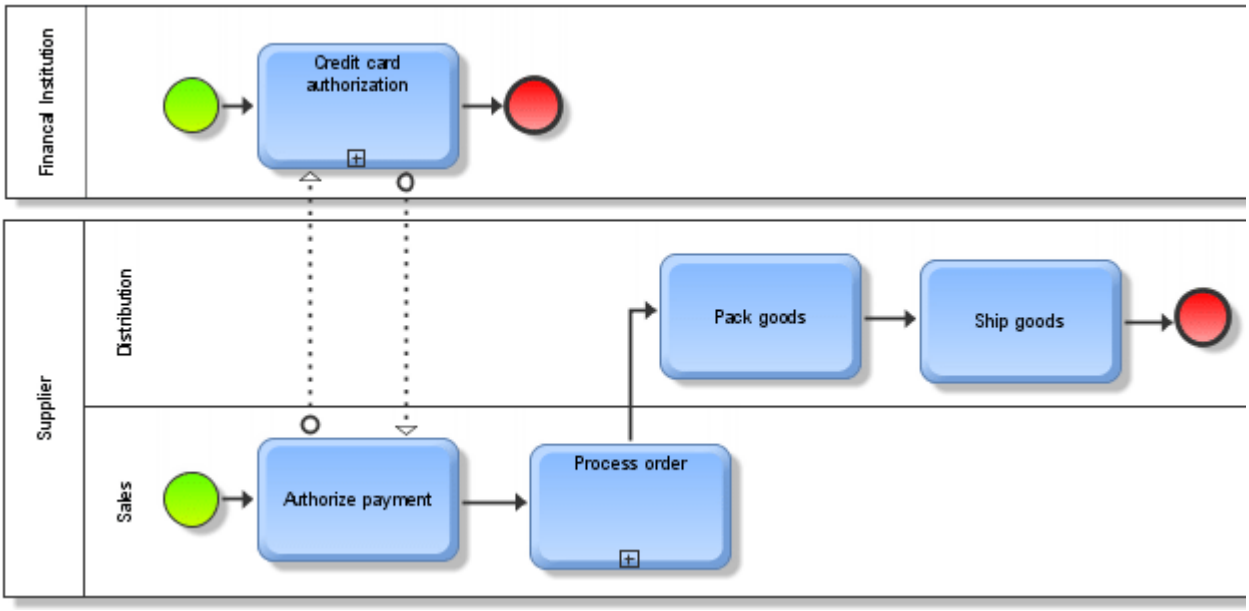


図 203: シーケンス フローとメッセージ フローを含む 2 つのプール

複数のビジネス パートナーのプロセスを 1 つの BPMN コラボレーション図で表示できるので、同じプロセスでもビジネス パートナーによって表示が異なるため、「視点」を指定すると分かりやすくなります。 BPMN では、BPMN コラボレーション図で視点を明確にする方法を定義していません。 最も簡単な方法は、割り当て済みの業務エンティティ (ビジネス パートナー) の名前を [説明/定義] 属性で指定することです (図を参照)。

12.2 ARIS での BPMN の実装

BPMN は業務プロセス図 (BPD) モデル タイプしか提供しませんが、ARIS では 2 つのモデル タイプを使用できます。EPC と新しい「業務プロセス図 (BPD)」モデル タイプです。 これにより、ARIS の既存のプロセスをプライベート プロセスとして再利用することができます。 EPC では、業務プロセス図で使用するために BPMN で指定されているすべてのモデル属性を使用できます。 業務プロセス図 (BPD) モデル タイプを使用することによって、既存の EPC タイプのモデルには B2B 特定の側面が不要になります。 このため、追加の関係タイプによって EPC モデルが複雑になることはありません。

新しい業務プロセス図は BPMN 関連のすべてのモデル属性を EPC から継承し、すべてのシーケンス フロー関連のオブジェクト、接続線、シンボルを継承します。 新しい業務プロセス図 (BPD) モデル タイプでは、シーケンス フロー関連の EPC コンセプトを再利用できます。 また、プール、レーン、メッセージ フローも表すことができます。

12.3 業務プロセス図の要素

12.3.1 プールとレーン

プールにより、プロセス図の構造が明確になります。

プールとは、業務エンティティの一連のアクティビティを組み合わせるグラフィック形式のコンテナーです。

業務エンティティとしては、ファンクション、アプリケーション システム、組織要素 (組織ユニット、組織ユニット タイプ、グループ、ロール、役職、場所、システム組織ユニット、システム組織ユニット タイプが含まれます)、またはデータ要素 (用語、クラスター/データ モデル、エンティティ タイプ、関係タイプ、ERM 属性、業務オブジェクト、複合業務オブジェクト、COT 属性、クラス、および情報媒体が含まれます) を使用できます。

BPMN では、2 つのプールによって 2 つの異なる業務エンティティを表します。 プール内でモデルを構成するテクニックは、主に B2B コンテキストで使用されます。

プールでは、レーン (Lane) によって構成および組織されたプロセス パートナーのさまざまなアクティビティを組み合わせます。 この方法で、ほかのプロセス パートナーのアクティビティと区別します (前の「シーケンス フローとメッセージ フローを含む 2 つのプール『205 ページ 』」の図を参照)。

BPD では、プールに必ずしもプロセス要素を含める必要はありません。 たとえば、関連性はあるものの詳細は不明なサブプロセス (ビジネス パートナーのものなど) の相互関係をプロセス全体に組み込む場合は、モデルに空白のプール (「ブラック ボックス (black box)」) を挿入することもできます。 複雑になりすぎるのを防ぐために、サブプロセスの詳細のモデル化を省くこともできます (「電子メール投票プロセス『216 ページ 』」の図を参照)。

プールには、少なくとも 1 つのレーンがあります。 レーンには、ネストされたレーンやマトリクスとして定義されたレーンをさらに組み込むことができます。 プールにレーンが 1 つしかない場合は、プールとレーンの名前は同じになります。 プールに複数のレーンがある場合は、レーンの名前とプール専用の名前を指定する必要があります。

図 204: BPMN による 2 つのレーンを持つプール

ARIS では、プールとレーンは最初にモデルに配置される個別のオブジェクト タイプです。 プール内で、EPC と同様の方法でプロセスをモデル化できます。 プロセスのファンクション、イベント、およびルールはすべて、プール オブジェクト上に配置されます。 これらのオブジェクトとプールとの関連性を記述するには、[属する] 接続線を使用します。 この接続線は、非表示の関係として作成することをお勧めします。 プール オブジェクトは、[描写する] タイプの接続線を使用して、組織要素、アプリケーション システム タイプ オブジェクト、データ要素、またはファンクションに接続できます。 各プールが任意のデータベース内で持つことができるこのタイプの接続線は 1 本のみである点に注意してください。 これらの関係も非表示で作成します。

BPMN 仕様では、プールはシンボルによってモデルで表示する必要はありません。 また、特に図にプールが 1 つしかない場合などには、プールの枠を非表示にできます (「電子メール投票プロセス『216 ページ 』」の図を参照)。 ただし、これらのオプションはモデルの透過性に影響を及ぼすので、複数のプールを含むモデルにはお勧めしません。

12.3.2 プールとレーンのモデリング基準

- 非表示の枠を持つプールは、図の中に 1 つしか配置できません。
- [プール タイプ (Pool type)] 属性を [コラボレーション (Collaboration)] に設定した場合は、所有者 ([責任者] 属性) を指定しないでください。
- 各レーンの上位プールは 1 つだけです。

12.3.3 シーケンス フロー

シーケンス フロー形式のプロセスでは、プロセスのアクティビティの実行順序が示されます。 シーケンス フローでは、[イベント]、[アクティビティ]、[ゲートウェイ] の各オブジェクト タイプを組み合わせます。 シーケンス フローはプール内のみで使用できます。プールの枠を超えることはできません (次の図を参照)。

シーケンス フローは黒い矢印が付いた実線で表示されます。

図 205: シーケンス フロー接続線

接続線のソースおよびターゲット オブジェクト タイプに応じて、[アクティブにする]、[評価される]、[作成する]、[結合する]、[引き起こす] などの接続線タイプを指定します。

12.3.4 シーケンス フロー接続線のモデリング基準

- XOR (データ基準) ゲートウェイまたは包含的ゲートウェイに続くシーケンス フローでは、[条件 (Condition)] 属性の値を設定してください。
- [条件] 属性の値が [式 (Expression)] に設定されている場合は、菱形シンボルを接続線の先頭に配置する必要があります。
- [条件] 属性の値が [デフォルト (Default)] で、ソース オブジェクトがファンクションの場合は、「¥」(バックスラッシュ) シンボルを接続線の先頭に配置する必要があります。
- ソース オブジェクトがゲートウェイの場合は、「¥」(バックスラッシュ) シンボルは配置しないでください。
- ソース オブジェクトが以下のいずれかのシンボルの場合は、条件を設定しないでください。
  - イベント基準ゲートウェイ (Event-based gateway)
  - 複合ゲートウェイ (Complex gateway)
  - 並行ゲートウェイ (Paralell gateway)
  - 開始イベント
  - 中間イベント
- [条件] 属性の値「デフォルト」がシーケンス フロー接続線で有効になっている場合、条件は指定できません。
- ソース オブジェクトがファンクションまたは XOR (データ基準) ゲートウェイの場合は、[条件] 属性の値に「デフォルト」を設定できます。
- [条件] 属性の値が「式」に設定されている場合は、[式] 属性も指定する必要があります。

12.3.5 メッセージ フロー

メッセージ フローは、2 つのプール間の情報交換を表します。 メッセージ フローは、2 つのプール オブジェクト間に直接配置したり、対応するプールのプロセスに組み込まれているシーケンス フローのオブジェクト間に配置できます。 プールの枠を超えて設定できるのは、メッセージ フローだけです。メッセージ フローの接続線は、同じプール内にある 2 つのオブジェクト間には配置できません (「シーケンス フローとメッセージ フローを含む 2 つのプール 『205 ページ 』」の図を参照)。

接続線は破線で表示されます。 線の始点は円でマーキングされ、終点は白い矢印で表示されます。

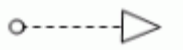


図 206: メッセージ フロー接続線

各メッセージ フローは、送信者オブジェクト、[送る] タイプの接続線、[受信される] タイプの接続線、受信者で構成されます。メッセージ フロー接続線は、開始イベントや中間イベントを起点とすることはできません。 しかし、終了イベントはメッセージ フローを受け取りませんが、自身が送信者になることはできます。 レーン、ゲートウェイ、データ オブジェクト、テキスト注釈にはメッセージ フローはありません。

12.3.6 メッセージ フロー接続線のモデリング基準

ソース オブジェクトとターゲット オブジェクトは、異なるプールに属している必要があります。

12.3.7 関連

関連 (Association) は、シーケンスまたはメッセージ フロー コンポーネントに情報を提供するために使用します。 この情報は、テキスト形式またはグラフィック形式で提供されます。 複数のプロセスが同じ図の中の一部である場合は、個々のプロセス要素を接続線で互いに関連付けることができます。

関連は破線で表示されます。必要に応じて矢印を先頭に追加できます。 これは、特に [データ オブジェクト (Data object)] タイプの成果物を割り当てる場合に使用します。

図 207: 関連接続線

接続線のソースおよびターゲット オブジェクトのタイプに応じて、[出力として持つ]、[入力である]、[入力を提供する]、[出力する] などの適当な接続タイプを指定します。

BPMN では、[データ オブジェクト] タイプの成果物をアクティビティに割り当てることが特に重要です。

この割り当てによって、情報がプロセス内でどのように使用され、どのように変更されるかが明らかになります。 この割り当ては、以下のような関連によって BPD (BPMN) で実装されます。

ファンクション - 出力する - データ要素 (特に情報媒体)

データ要素 (特に情報媒体) - 入力を提供する - ファンクション

12.3.8 イベント

イベントとは、業務プロセスの過程で発生する状態です。 イベントはプロセス フローに影響を及ぼします。 通常は、プロセス内のトリガーまたは効果に相当します。 イベントは、イベントの発生時点により、開始イベント、中間イベント、終了イベントのいずれかになります。 BPMN では、この 3 種類のイベント カテゴリをそれぞれ別のシンボルで表示します。

図 208: イベント カテゴリ

これらのカテゴリは、特化されたサブカテゴリに分割されます。 [イベント タイプ (Event type)] 属性を指定すると、3 つのイベント カテゴリのシンボルに以下の 3 つの例のような補足シンボルが追加されます。

図 209: イベント タイプの例

[イベント] オブジェクト タイプに関連するすべての属性は、[BPMN] 属性タイプ グループとして 1 つにまとめられます。

12.3.9 イベントのモデリング基準

- 「イベント タイプ」属性タイプで開始イベントに設定できる値は、「メッセージ」、「タイマー」、「ルール (Rule)」、「リンク (Link)」、「複数 (Multiple)」のうちの 1 つのみです。
- 終了イベントの場合、「イベント タイプ」属性タイプに設定できる値は、「メッセージ」、「例外 (Exception)」、「キャンセル (Cancel)」、「代償 (Compensation)」、「ルール」、「リンク」、「複数」、「終了 (Terminate)」のうちの 1 つのみです。
- 中間イベントの場合、「イベント タイプ」属性タイプに設定できる値は、「メッセージ」、「タイマー」、「例外」、「キャンセル」、「代償」、「ルール」、「リンク」、「複数」のうちの 1 つのみです。
- イベント タイプ セットによっては、追加情報を適切な属性で指定する必要があります。
- 開始イベントには、複数の出力シーケンス フロー接続線を設定できます。 これらの接続線の [条件] 属性には値を設定できません。
- 例外または代償を示す中間イベントは、ファンクションの枠に配置します。
- 中間イベントがファンクションの枠に配置される場合、「リンク」以外の値を指定する必要があります。
- プロセスの通常のシーケンス フローに配置された中間イベントには、「複数」、「ルール」、「キャンセル」を値として設定できません。
- 以下の場合、値「キャンセル」は設定できません。
  - 中間イベントがファンクションの枠に配置され、ファンクションの [トランザクション (Transaction)] 属性が有効になっていない場合。
  - イベントが、トランザクションを記述するプロセスに属していない場合。
- 中間イベントがファンクションの枠に配置される場合、シーケンス フロー接続線のターゲット オブジェクトにはできません。
- 中間イベントがプロセスの通常のシーケンス フロー内に配置され、ファンクションの枠に配置されていない場合、1 本の入力シーケンス フロー接続線を設定できます。 イベントの [イベント タイプ] 属性には、値を設定しないか、または「メッセージ」、「タイマー」、「例外」、「リンク」、「代償」のうちの 1 つを値として設定します。
- 通常のシーケンス フロー内で中間イベントの値に「リンク」を設定できるのは、ソース オブジェクトが、[ゲートウェイ タイプ (Gateway type)] 属性の値が「XOR」(イベント基準) のゲートウェイである場合のみです。
- 中間イベントには、出力シーケンス フロー接続線を 1 本のみ設定します。

- [イベント タイプ] 属性の値が「メッセージ」の中間イベントには、入力メッセージ フロー ([受信される] タイプの入力接続線) を設定できます。
- 中間イベントには、出力メッセージ フロー ([送る] タイプの出力接続線) を設定できません。

12.3.10 アクティビティ

アクティビティはプロセスの一部として実行されます。 アクティビティは単独で、または非単独 (複合) で設定できます。

BPMN では、プロセス (Process)、サブプロセス (Subprocess)、タスク (Task) という 3 種類のアクティビティ カテゴリを許可します。

BPMN ではアクティビティに対して以下のシンボルを使用します。

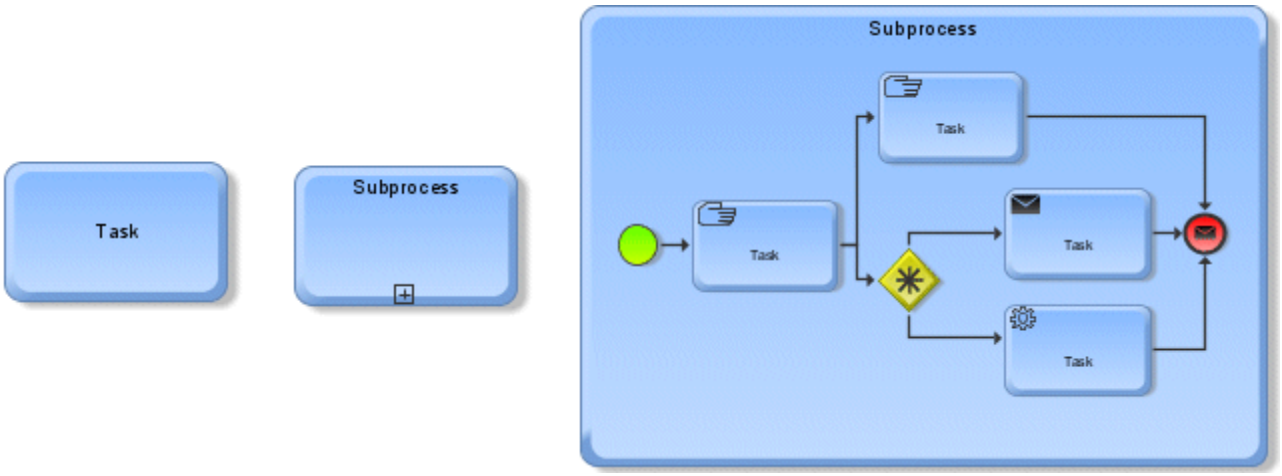


図 210: BPMN のアクティビティ

ARIS では、アクティビティは、デフォルトでファンクションとしてモデル化されます。

図 211: ARIS でアクティビティとして割り当てられたファンクション

ファンクションには、BPMN でプロセス、サブプロセス、タスクに定義するすべての属性が提供されます。 イベントと同様に、アクティビティ タイプの補足下位グループを含む [BPMN] 属性タイプ グループを使用します。

BPMN では、プロセスは企業または組織内で実行されるアクティビティを記述します。 プロセスは、さまざまなアクティビティや制御オブジェクトの集合を表すフロー オブジェクトを使用した図で記述されます。 プロセスは階層構造になっており、さまざまな詳細レベルで定義できます。 プロセスとは異なり、BPMN での業務プロセスは、企業および組織の境界を越えて実行される一連のアクティビティを記述します。

BPMN では、サブプロセスは結合されたアクティビティで、詳細な説明があります。 サブプロセスは、プロセス フロー内のオブジェクトとして発生します。

通常、サブプロセスには詳細なプロセスが割り当てられます。 BPMN とは異なり、ARIS ではアサインされたアクティビティをアサインメント アイコンで表示します。プラス記号は表示されません。

BPMN では、アサインされたファンクションを識別することができ、さらに 1 つ上のプロセス レベルの詳細なプロセスも表示できます。 詳細なプロセスを表示するにはプラス記号をクリックします。

12.3.11 アクティビティのモデリング基準

プロセス

- [アドホック (Ad hoc)] 属性が「True」の場合は、[完了条件 (Completion condition)] 属性を指定する必要があります。
- アドホック プロセスを詳細化すると、アサイン モデル内でシーケンス フローをモデル化できません。

サブプロセス

- [サブプロセス タイプ] 属性の値を「独立 (Independent)」に設定した場合は、[プロセス リファレンス] 属性も指定する必要があります。
- サブプロセスに対して [トランザクション (Transaction)] 属性を有効にした場合は、[トランザクション ID (Transaction ID)] 属性も使用できます。
- [ループ タイプ (Loop type)] 属性を指定した場合は、[ループ条件 (Loop condition)] 属性も必要です。
- モデルを BPEL4WS に転送する場合は、値が「標準」に設定されているプロセスの [ループ タイプ] 属性が [最大 (Maximum)] 属性に指定されていることを確認してください。
- [ループ タイプ] 属性の値が「標準」に指定されている場合は、[次以前にテスト (Test before)] 属性も必要です。[次以前にテスト (Test before)] 属性はデフォルトでオフになっています。
- [ループ タイプ] 属性の値が「マルチインスタンス (Multi-instance)」に指定されている場合は、[並行インスタンス生成 (Parallel instance generation)] 属性も必要です。 [並行インスタンス生成 (Parallel instance generation)] 属性はデフォルトでオフになっています。
- サブプロセスの [ループ タイプ] 属性の値が「マルチインスタンス (Multi-instance)」に設定されており、[並行インスタンス生成 (Parallel instance generation)] 属性も有効になっている場合は、[ループ フロー条件 (Loop flow condition)] 属性も指定する必要があります。
- プロセスの [ループ フロー条件] 属性の値が「複合 (Complex)」に設定されている場合は、[複合 (Complex)] 属性にサブプロセスが完了したあとに渡されるプロセス マーカーの時間と数を取得するための式を指定する必要があります。

タスク

- [タスク タイプ] 属性の値が「受信 (Receive)」に指定されている場合は、ファンクションに出力メッセージ フロー接続線を使用しないでください。
- [タスク タイプ] 属性の値が「送信 (Send)」に指定されている場合は、ファンクションに入力メッセージ フロー接続線を使用しないでください。

- [タスク タイプ] 属性の値が設定されていないか、または値「スクリプト (Script)」または「手動 (Manual)」に指定されている場合は、ファンクションで入力または出力メッセージ フロー接続線を使用しないでください。
- [タスク タイプ] 属性が「抽象 (Abstract)」に設定されているファンクションでは、[抽象タイプ] 属性も設定する必要があります。 また、これらのファンクションは [抽象] または [コラボレーション] タイプのプールでしか使用できません。

12.3.12 ゲートウェイ

ゲートウェイ (Gateway) は、シーケンス フローのプロセス内での分割または結合を表します。 ゲートウェイは、入力および出力接続線の動作を決定します。 ARIS では、ゲートウェイは [ルール] タイプのオブジェクトとして表示されます。

イベントと同様に、ゲートウェイもさまざまなタイプを指定できます。 タイプに応じて、補足シンボルがゲートウェイ シンボルの中央に表示されます。

各種ゲートウェイ シンボルの一部:

図 212: ゲートウェイ タイプ

BPMN 仕様は、各ゲートウェイで定義する必要のあるゲートの数を規定します。 ARIS では、ゲートの数は入力および出力接続線の数によって決まります。 したがって、ゲートに依存する属性は、ルールの入力および出力シーケンス フロー接続線に指定されます。

複合ゲートウェイ (Complex gateway) は特殊なケースです。この場合は、[入力条件 (Incoming condition)] および [出力条件 (Outgoing condition)] 属性が定義されます。 指定したゲートウェイに複数の入力または出力シーケンス フロー接続線がある場合は、これらの属性が必要です。 入力条件の属性には、シーケンス フロー名とプロセス プロパティ (データ) を設定することができます。 出力条件には、シーケンス フロー ID とプロセス特徴 (データ) への参照を設定します。

12.3.13 ゲートウェイのモデリング基準

- XOR (データ基準) タイプのゲートウェイ: XOR (データ基準) のゲートウェイのすべての出力接続線では、[条件 (Condition)] 属性の値を「式」に設定し、[条件式 (Condition expression)] 属性で有効な式を使用する必要があります。

シーケンス フロー (特にゲートウェイの後):

- [XOR (データ基準)] タイプのすべての XOR ゲートウェイでは、1 本の出力シーケンス フロー接続線 ([アクティブにする] 接続タイプ) に [デフォルト ゲートウェイ (Default gateway)] 属性を指定する必要があります。 いかなる場合も、この属性は複数の出力接続線に設定できません。
- [XOR (イベント基準)] タイプの XOR ゲートウェイでは、2 本以上の出力シーケンス フロー接続線 ([アクティブにする] または [引き起こす] タイプ) を使用します。

- イベント基準の XOR ゲートウェイのすべての出力接続線では、[条件] 属性の値は指定できません。 [条件式] 属性は指定できません。
- イベント基準の XOR ゲートウェイの出力シーケンス フロー接続線では、以下のターゲット オブジェクトを使用できます。
  - [受信] タスク タイプが設定されているファンクション
  - [イベント タイプ] 属性タイプの値が「代償」または「複数」以外に設定されている中間イベント
- ターゲット オブジェクト セットにファンクションが含まれている場合は、そのセットに [メッセージ] タイプのイベントを含めることはできません。
- [OR] タイプのゲートウェイに入力シーケンス フロー接続線がない場合、または 1 本だけある場合は、少なくとも 2 本の出力シーケンス フロー接続線が必要です。
- OR ゲートウェイのすべての出力シーケンス フロー接続線では、[条件] 属性の値を「式」に設定し、[条件式] 属性で有効な式を使用する必要があります。 式は現在のゲートウェイと明確に関連付ける必要があります。
- OR ゲートウェイに正確に 1 本の出力シーケンス フロー接続線がある場合は、この接続線の [条件] 属性の値は指定できません。
- [複合] タイプのゲートウェイに入力シーケンス フロー接続線がない場合、または 1 本だけある場合は、少なくとも 2 本の出力シーケンス フロー接続線が必要です。
- 複合ゲートウェイのすべての出力接続線では、特に出力接続線が 1 本しかない場合は、[条件] 属性の値を「なし」に指定する必要があります。
- 複合ゲートウェイに複数の入力シーケンス フロー接続線がある場合は、[入力条件] 属性に、シーケンス フロー名とプロセス プロパティ (データ) を参照する条件を指定する必要があります。
- 複合ゲートウェイに複数の出力シーケンス フロー接続線がある場合は、[出力条件] 属性に、シーケンス フロー名とプロセス プロパティ (データ) を参照する条件を指定する必要があります。
- AND ゲートウェイに入力シーケンス フロー接続線がない場合、または 1 本だけある場合は、少なくとも 2 本の出力シーケンス フロー接続線が必要です。
- AND ゲートウェイのすべての出力シーケンス フロー接続線では、[条件] 属性の値は指定できません。

12.3.14 成果物

成果物 (Artifact)は、プロセスに関する情報を提供します。 この情報は、シーケンス フローやメッセージ フローに属しません。成果物は、[データ オブジェクト (Data Object)]、[グループ (Group)]、[注釈 (Annotation)] の 3 つのタイプに分類されます (タイプ リストは必要に応じて拡張できます)。

データ オブジェクトは、ARIS の情報媒体またはデータ要素に相当します。 ただし広義では、データ オブジェクトにはあらゆる割り当てが含まれます。 データ オブジェクトは、シーケンス フローにもメッセージ フローにも影響しません。データ オブジェクトはプロセスの間に発生する事象に関する情報を提供します。 プロセスの間にドキュメントやデータ、その他のオブジェクトがどのように変化するかを表します。

グループは、関連付けられているプロセス要素をグラフィック形式で表したものです。 ARIS では、これには、四角形や多角形などのグラフィック オブジェクトが適しています。

また、この目的にグループも使用できます。 ただし、この方法はグループ化にグラフィックが含まれている場合のみ有効です。

注釈は、「Time out [1week]」の例のように、オブジェクトや接続線の注釈に相当します。 ARIS では、通常、[注釈/例] 属性によって実装されます。 この属性は、次の例に示されるように、[はい] と [いいえ] を使用して、モデルに配置することが重要です。

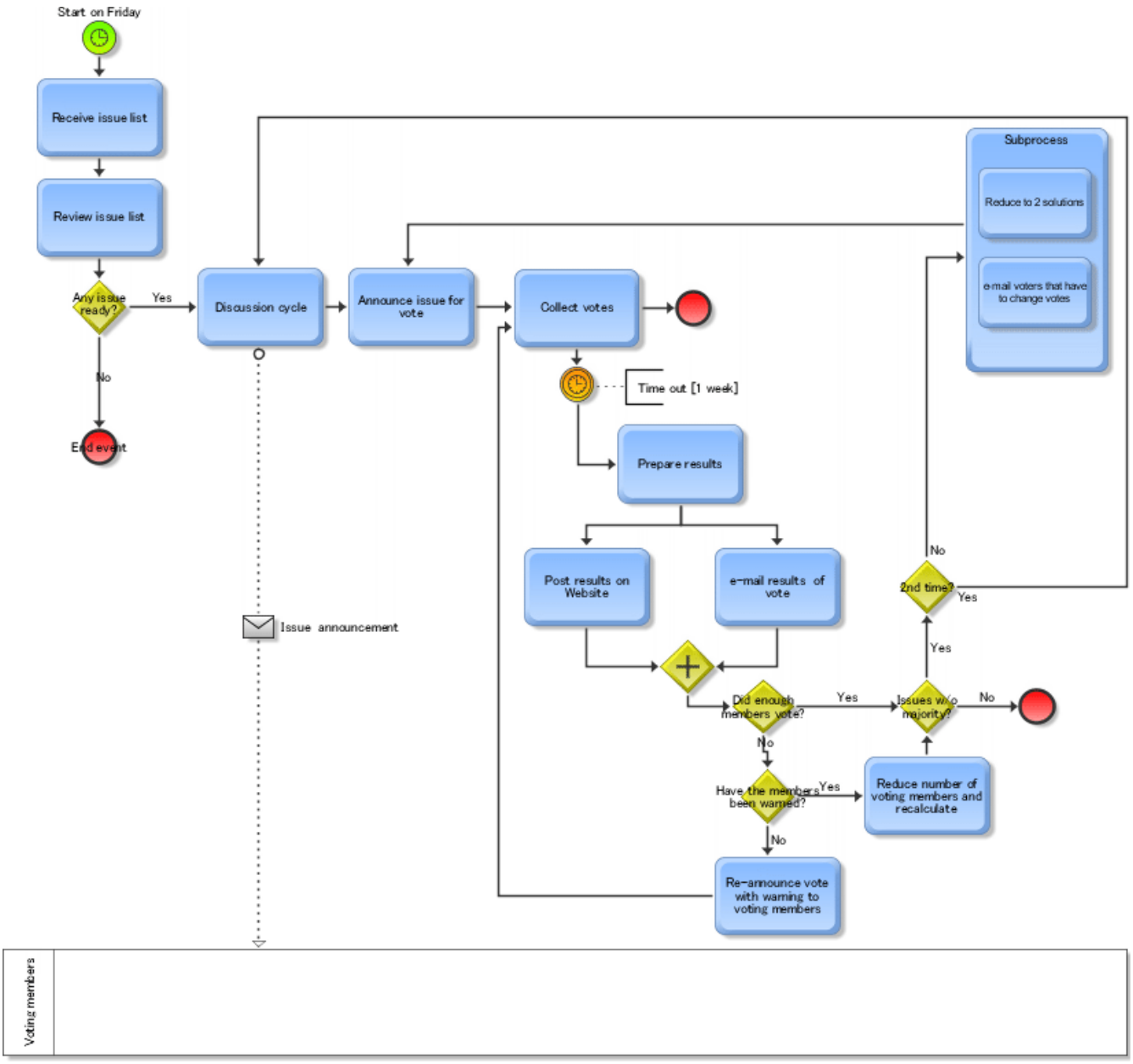


図 213: 電子メール投票プロセス

この図は、ARIS で BPMN 2.0 に準拠して業務コラボレーション図を実装する方法の例を示したものです。 図には 2 つのプールが含まれており、上のプールの境界線は非表示になっています。 下位のプールに関する個別の要素は表示されません。

12.3.15 図の出典

図「シーケンス フローとメッセージ フローを含む 2 つのプール『205 ページ 』」:

『Business Process Modeling Notation, Working Draft (1.0)』BPMI.org 著、2003 年 8 月 25 日、85 ページ

図「BPMN による 2 つのレーンを持つプール『208 ページ 』」:

『Business Process Modeling Notation, Working Draft (1.0)』BPMI.org 著、2003 年 8 月 25 日、87 ページ

図「イベント カテゴリ『211 ページ 』」および図「イベント タイプの例『211 ページ 』」:

『Business Process Modeling Notation, Working Draft (1.0)』BPMI.org 著、2003 年 8 月 25 日、27 ページ

図「BPMN のアクティビティ『213 ページ 』」:

『Business Process Modeling Notation, Working Draft (1.0)』BPMI.org 著、2003 年 8 月 25 日、28 ページ

図「ゲートウェイ タイプ『215 ページ 』」:

『Business Process Modeling Notation, Working Draft (1.0)』BPMI.org 著、2003 年 8 月 25 日、28 ページ

13 参考文献

13.1 一般的な参考文献リスト

- Brombacher, R.; Bungert, W. 共著:『Praxis der Unternehmensmodellierung』(企業モデリング実践)、1992 年。

  (企業モデリング実践)、ドイツ IDS Scheer 社によるセミナー、ドイツ Taunus/Bad Soden、1992 年 11 月 12～ 13 日
- Chen, P. P. 著:『Entity-Relationship Model』 1976 年

  『The entity-relationship model - toward a unified view of data, in: ACM Transactions on Database Systems』、Vol. 1 (1976)、Issue 1、9～36 ページ
- Hoffmann, W.; Kirsch, J.; Scheer, A.-W. 共著:『Modellierung mit Ereignisgesteuerten Prozeßketten』(イベント駆動プロセス連鎖でモデリング)、1993 年

  『Modeling with event-driven process chains』(メソッド マニュアル、1992 年 12 月号)、Scheer, A.-W. (編)、Institut für Wirtschaftsinformatik (情報システム機関) の出版物、論文 101、Saarbrücken、1993 年 1 月。
- Keller, G.; Hechler, H.-J. 共著:『Informationsmodell 1991』(情報モデル)、1991 年

  『Konzeption eines integrierten Informationsmodells für die Kostenrechnung des SAP-Systems』(SAP システムでの原価計算のための統合情報モデル設計): Scheer, A.-W. (編)、『Rechnungswesen und EDV - 12. Saarbrücker Arbeitstagung 1991. Kritische Erfolgsfaktoren im Rechnungswesen und Controlling』(第 12 回、会計と IT 会議、ザールブリュッケン、1991 年。会計と統制における重要成功要因)、Heidelberg、1991 年、67 ～ 106 ページ
- Scheer, A.-W. 著:『Architecture of integrated Information Systems』1992 年

  Architecture of Integrated Information Systems - Foundations of Enterprise Modelling (統合情報システムアーキテクチャ - 企業モデリングの基礎)、第二版、Berlin et al.、 1992.
- Scheer, A.-W. 著:『EDV-orientierte Betriebswirtschaftslehre』(EDP 指向の業務管理研究)、1990 年

  『EDV-orientierte Betriebswirtschaftslehre - Grundlagen für ein effizientes Informationsmanagement』(EDP 指向の業務管理研究 - 効率的な情報管理の基礎)、第四版、Berlin et al.、 1990.
- Scheer, A.-W. 著:『Business Process Engineering』、1994 年

  Business Process Engineering - Reference Models for Industrial Enterprises (業務プロセス エンジニアリング - 企業のリファレンス モデル)、第五版、Berlin et al.、 1994.
- Schlageter, G.; Stucky, W. 共著:『Datenbanksysteme』(データベース システム)、1983 年

  『Datenbanksysteme - Konzepte und Modelle』(データベース システム : 概念とモデル)、第 2 版、Stuttgart 1983。
- Seubert, M. 著:『SAP-Datenmodell』(SAP データ モデル)、1991 年

『Entwicklungsstand und Konzeption des SAP-Datenmodells』(SAP データ モデルの開発と設計状態)、Scheer, A.-W. (編): 『Datenbanken 1991 - Praxis relationaler Datenbanken: Vom Datenmodell zur Implementierung』(データベース 1991 年 - 関係データベース実践: データ モデルから導入まで) (1991 年 6 月 4 ～ 5 日にドイツのザールブリュッケンで開催されたシンポジウム)、Saarbrücken、1991 年、87 ～ 109 ページ

- Sinz, E. J. 著: 『Entity-Relationship-Modell』(実体関連モデル)、1990 年

  『Das Entity-Relationship-Modell (ERM) und seine Erweiterungen』(実体関連モデル (ERM) とその拡張)、「HMD Theorie und Praxis der Wirtschaftsinformatik (1990)」(業務プロセス エンジニアリングの理論と実践に関する HMD 誌)、論文 152、17 ～ 29 ページ。
- Scheer, A.-W. 著: 『ARIS - Business Process Frameworks』、第三版、Berlin et al.、1998 年
- Scheer, A.-W. 著: 『ARIS - Business Process Modeling』、第三版、Berlin et al.、1998 年
- Scheer, A.-W.、Jost, W. 共著: 『ARIS in der Praxis』(ARIS 実践)、2002 年

  「Gestaltung, Implementierung und Optimierung von Geschäftsprozessen」(「業務プロセスの設計、導入、最適化」)、Berlin、Heidelberg、New York、2002 年
- Scheer, A.-W.、Abolhassan, F.、Jost, W.、Kirschmer, M. 共著: 『Business Process Excellence 2002』

  ARIS in Practice、Berlin、Heidelberg、New York 2002 年

13.2 テーマ関連の参考文献

13.2.1 ARIS の統一モデリング言語

13.2.1.1 UML 仕様

UML 仕様: http://www.uml.org.

13.2.1.2 UML の使用

- Burkhardt, R. 著: 『UML - Unified Modeling Language, Objektorientierte Modellierung für die Praxis』(オブジェクト指向モデリング実践)、Bonn、1997 年。
- Fowler, M.; Scott, K 共著.:『UML Distilled - Applying the Standard Object Modeling Language』、Reading et al.、1997 年。
- Oesterreich, B. 著: 『Developing Software with UML: Object-oriented analysis and design in practice』、第三版、Munich - Vienna、1997 年。

13.2.1.3 UML と業務プロセス モデリング

- Ambler, S. W. 著:『What's Missing from the UML? Techniques that can help model effective business applications』、『Object Magazine』、7(1997)8
- Loos, P.; Allweyer, Th. 共著:『Process Orientation and Object-Orientation - An Approach for Integrating UML and Event-Driven Process Chains (EPC)』、Institut für Wirtschaftsinformatik (情報システム機関) の出版物、論文 144、Saarbrücken 、1998 年。

13.2.2 Object Modeling Technique (OMT)

Rumbaugh, J.; Blaha, M.; Premerlani, W.; Eddy, F.; Lorensen, W. 共著:『Object-Oriented Modeling and Design』、 Munich、1993 年

13.2.3 知識管理の手法

13.2.3.1 一般的知識管理

- Probst, G.; Raub, S.; Romhardt, K. 共著:『Managing Knowledge - Building Blocks for Success』、Frankfurt Wiesbaden、1998 年
- Bürgel, H. D. (編):『Wissensmanagement - Schritte zum intelligenten Unternehmen』(知識管理 - インテリジェントな企業へのステップ)、 Berlin et al. 1998 年

13.2.3.2 ARIS を知識管理に利用

- Allweyer, Th. 著:『Modellbasiertes Wissensmanagement』(モデルベースのナレッジベースの管理)、『IM Information Management & Consulting』、13 (1998) 1、37～45 ページ
- Allweyer, Th. 著:『Using ARIS Models for Knowledge Management』、 Scheer, A.-W. 著:『ARIS - Business Process Frameworks』、第三版、Berlin et al.、1998 年、162 ～ 168 ページ。

13.2.4 バランス スコアカード メソッド

Kaplan, Robert/Norton, David 共著:『The Balanced Scorecard - Measures that Drive Performance』、Harvard Business Review、1992 年 1 月/2 月。

13.2.5 IT 都市計画

- Schulman, Jeff 著:『A New View of Architectures Needed for New Business Drivers』、ガートナー ブリーフィング プレゼンテーション。

- Longépé, Christoph 著:『Le projet d'urbanisation du système d'in-formation』、Dunod 社、Paris、2001 年

13.2.6 業務プロセスのモデル化

『Business Process Modeling Notation, Working Draft (1.0)』BPMI.org 著、2003 年 8 月 25 日