The Business Intelligence Model: Strategic Modelling

version 1.0

Daniele Barone[1], John Mylopoulos[1], Lei Jiang[1], and Daniel Amyot[2]

[1]Department of Computer Science, University of Toronto, Toronto, ON, Canada
barone/jm/leijiang@cs.toronto.edu
[2]SITE, University of Ottawa, Ottawa, ON, Canada
damyot@site.uottawa.ca

April 14, 2010

Contents

Business Intelligence (BI) consists of a range of technologies intended to assist large organizations in determining the state and quality of their operations. BI activities are meaningful in the context of a business organization and its objectives, strategies and tactics, as well as a broader (external) context involving regulations, competitors, customers, markets, etc. This business context (internal and external) defines the effectiveness of business processes, and the things to monitor to ensure that business objectives are being met and regulations and policies are complied with. The Business Intelligence Model (BIM) provides a set of constructs for modeling and analyzing a business context consisting of intentions, situations, processes, actors, influences, key performance indicators, and more. It is intended to support the modeling and analysis of a business organization at both a strategic and a tactical level. BIM schemas can be used for governance activities, including analysis, monitoring and auditing. This report presents some of the main innovations of BIM, including its primitive concepts, its structuring mechanisms, analysis examples, as well as an overview of an illustrative case study.

1 Introduction

Business intelligence (BI) consists of a range of technologies for using information within organizations to ensure compliance to strategic and tactical objectives, as well as to laws and regulations. As a research field, it encompasses data and knowledge management, modeling of processes and policies, data quality, data privacy and security, data cleaning and integration, data exchange, inconsistency management, information retrieval, data mining, analytics, and decision support.

This interest in technologies and services that improve organizational governance has caused dramatic growth for the BI market and the industry that serves it. By now, most competitive organizations have a significant investment in BI, much of it technology-related, based on software tools and artifacts. However, as summarized by Gartner [35], one important problem of BI technologies is that information generated by BI systems and other decision inputs are rarely linked to business decisions and outcomes. In addition, business people – be they executives, managers, consultants, or analysts – are in general agreement that what they are looking for is not new gadgets producing a dizzying array of largely useless statistics. Instead, they are interested in having their business data analyzed in their terms: strategic objectives, business models and strategies, business processes, markets, trends and risks. This gap between the worlds of business and data remains today the greatest barrier to the adoption of BI technology, and the greatest factor in the cost of applying BI technology.

We propose to bridge this gap by extending the notion of conceptual schema to include concepts beyond entities and relationships. In particular, we propose the Business Intelligence Model (BIM) as a business-level counterpart to the Entity-Relationship or Relational Model, so that strategic objectives, business processes, risks and trends can all be represented in a business schema. Users can query this schema, much like conventional database schemas but with business terms, to perform analysis, to track decisions and their impacts, or to explore suitable strategies to problems at hand. Such queries are to be translated through schema mappings into queries defined over databases and data warehouses, and the answers are to be translated back into business-level concepts.

The main objective of this report is to introduce BIM's constructs for modeling business organizations at a strategic level. In particular, we present a set of primitive concepts consisting of actors, intentions (e.g., goals), situations (strengths / weaknesses / opportunities / threats, a.k.a. SWOT), influences, processes, key performance indicators and more. These concepts can be used in tandem with abstraction mechanisms, such as generalization, aggregation and classification, to develop global models of business organizations for purposes of analysis, monitoring and auditing.

Our work is founded on modeling techniques from diverse sources. Abstract concepts for describing all things are inspired from DOLCE[11]. The intentional and social concepts used in BIM are adopted from concepts in Goal-Oriented Requirements Engineering, notably [7, 42, 18]. The notion of influence is adopted from influence diagrams [16], a well-known and accepted decision analysis technique. Concepts related to SWOT analysis [9] and others have also been adopted from OMG's Business Motivation Model standard [3].

As shown in Figure 1.1, the final aim of the BIM is to represent the internal and external business and environment, and to support managers in making decisions at each level of management providing answers to questions such as "What will happen next?". Indeed, the BIM is based on the idea that you cannot measure what you cannot represent, and you cannot improve what you cannot measure.

As defined in [28, 24] and summarized in Table 1.1, people at different levels in an organization have different types of decision-making responsibilities.

Strategic decisions, which are typically made by executive managers, affect the long-term direction of an organization and are often complex and characterized by uncertainty due to the limited availability of information. Usually, managers at this level depend on their past experiences and instincts for making a decision.

Examples of strategic decisions can be to decide whether it is time to discontinue a product line or to launch a new one.

Tactical decisions regard more intermediate-term issues and are typically made by middle managers. The decisions made at this level attempt to move the organization closer to reaching the strategic goals.

Examples of tactical decisions can be to hire an advertising agency in order to promote a new product or to create an incentive plan for encouraging employees in increasing the organization's production.

Operational decisions concentrate on day-to-day organization activities and are typically made by lower-level managers. Decisions made at this level attempt to ensure that the daily activities are conform with respect to business targets and standards in order to achieve the strategic goals.

Examples of operational decisions include scheduling employees, purchasing raw materials needed for production, or answering questions such as "Do we extend credit to this customer?".

Level of Management	**Core Requirement**	**Nature of the Decision**
Executive	Strategic Planning	Long term, unstructured, difficult to develop specific decision models
Mid-level	Management Control	Shorter term, semi-structured, modeling possible.
Operational	Operational Control	Short term, structured, modeling possible.

Table 1.1: A taxonomy of management decision-making.

In this report, we focus on a particular use of the model for supporting the *strategic planning* process, which is the process of defining an organization's strategy (or

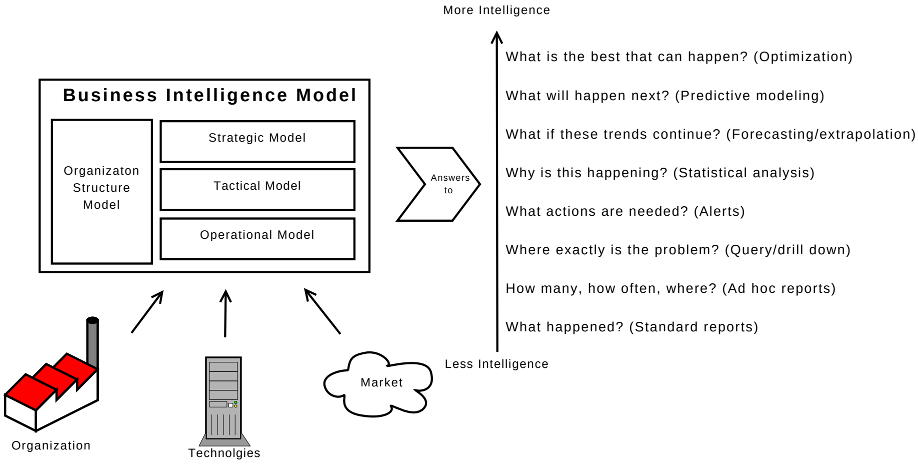

Figure 1.1: The *Business Intelligence Model* ("Questions" have been adapted from [8]).

direction) and making decisions on allocating the organization's resources in order to pursue this strategy.

In particular, an organization can follow different approaches for strategic planning such as the *Situation-Target-Path*, the *Draw-See-Think-Plan*, the *See-Think-Draw*, etc. [36] which can be summarized by the following activities:

1. *Formulate Vision & Mission*: an organization must clearly define its Vision and Mission statements and the associated hierarchy of goals and objectives;

2. *Situational Analysis*: an analysis of the organization and its environment must be conducted in order to identify influences among external and internal factors and the organization goals, e.g., the SWOT analysis [9].

3. *Develop Strategies*: a set of alternatives for the fulfillment of goals must be formulated in terms of actions and processes to be taken to achieve goals and in terms of the resources required to execute these actions;

4. *Implement Strategies*: the "best" strategies must be chosen and implemented using organizational processes and resources;

5. *Evaluate and Monitor Performance*: the implemented strategies must be evaluated to see if they are working successfully.

In the next sections we present the BIM metamodel and how it can support the above strategic planning activities.

In particular, Section 2 describes BIM's primitive concepts and their use for strategic modeling. Section 3 presents the mechanisms (generalization, aggregation, classification) used to structure BIM models. Section 4 offers an overview of how BIM models can be used to support a range of analyses, such as strategic map, goal analysis and SWOT analysis. In Section 5, we present an illustrative case study, while Sections 6 and 7 respectively discuss related work and conclusions.

2 Primitive Concepts

Table 2.1 introduces the primitive concepts in BIM for modeling strategic objectives and strategies.

As an ongoing example, we introduce *BestTech Inc.*, a company in the market of cellular phones and home computers[1]. Figure 2.1 shows its Strategic Map (SM) [22] and Balanced Scorecard (BSC) [21], which are strategic planning instruments commonly used in industry.

The former (SM) is a visual representation of the strategy of an organization showing plans used to achieve missions and visions. In particular, it illustrates the cause-and-effect relationships between different strategic goals and associated measures, the key performance indicators (KPIs) [33].

These measures are included in the latter, the BSC, which represents a "balanced" range of metrics against which to measure the Organization's performance. "Balance" here means that the broader view of leading performance indicators includes also non-financial concerns, such as "learning and growth of employees" and "customer satisfaction". Both SMs and BSCs describe and measure organizational performance across four balanced perspectives: *financial*, *customers*, *internal business processes*, and *learning and growth* [22, 21].

The following subsections detail *how* BIM's primitive concepts are used to describe BestTech and its business environment, from a strategic viewpoint.

2.1 Situations, Intentions and Processes

Primitive concepts are represented as metaclasses (having names that end in `-Class`). In particular (Figure 2.2 [2]), `Intention` can be used to define a hierarchy of *Vision*, *Strategic/Tactical goals*, etc., which represents the desired end state of an organization. `Process` models an organization's *Mission* and the different *Strategies* and, at tactical and operational levels, their decomposition into *Business processes* and *Activities*. `Resource` models what resources are required to execute such strategies.

In any strategic planning setting, a scan of the internal and external environment is a fundamental issue [9]. Accordingly, BIM provides the `Situation` concept for modeling internal and external situations that can be helpful or harmful to an organization's goals. In addition, we adopt the SWOT classification [9] (see Section 4) in order to define the types of influence that a `Situation` may exercise on an `Intention`.

[1] Adapted from a generic strategy map for a credit card company provided by the Balanced Scorecard Institute – http://www.balancedscorecard.org (2010)

[2] Dashed inheritance arrows indicate the existence of hidden metaclasses in between.

Concept	Description	Example	Superclass
`Thing`	BIM's most general concept. This abstract metaclass has everything as an instance.		
`Object`	Abstract metaclass whose instances persist/endure over time [11].		`Thing`
`Event`	Instantaneous happening (perdurant) that changes an `Object`; can be described by a `Proposition` that was false before the `Event` and is true after [11, 15].	*New order received*	`Thing`
`Situation`	Partial state of the world described by a `Proposition`. `Situations` can have structure consisting of relations and `Things` standing in those relations [27].	*Christmas season*	`Object`
`Proposition`	Describes a `Situation`. In general, `Propositions` are true/false/undefined in a `Situation` [27].		`Object`
`Intention`	A `Proposition` an `Actor` wants to make true [42].	*More products sold*	`Proposition`
`Domain Assumption`	A `Proposition` assumed by an `Actor` to be true for purposes of fulfilling an `Intention`.	*Market increases at least 10% annually*	`Proposition`
`Directive`	A `Proposition` prescribed by an authority intended to constraint, guide, govern, or influence elements of an organization such as `Actors` and `Processes` [3].	*Pizza delivery in 30 mins.*	`Proposition`
`Entity`	`Object` whose existence does not depend on that of others.		`Object`
`Actor`	`Entity` that carries out `Actions` to achieve `Intentions` [42].	*Sales manager*	`Entity`
`Action`	`Entity` performed by an `Actor` that produces `Events` and can have pre- and postconditions [7].	*Deliver product*	`Entity`
`Process`	`Entity` consisting of coordinated `Actions` to achieve an `Intention` [3].	*Sales process*	`Entity`
`Resource`	`Entity` of value to an `Actor` [42].	*Money, information*	`Entity`
`Indicator`	Measurable `Object` that gives information about the state of an associated `Object`. Can be used to quantify the satisfaction level of an `Intention` or the operational performance of an `Actor`, a `Process` or a `Resource` [33].	*Number of products sold per week*	`Object`
`Relationship`	An `Object` that relates two or more `Things` and whose existence depends on that of the `Things` it relates [5].		`Object`
`Influence`	A `Relationship` between two `Things` t_1 and t_2, where the state of t_1 constrains the state of t_2 in a probabilistic or causal sense.		`Relationship`

Table 2.1: BIM primitive concepts.

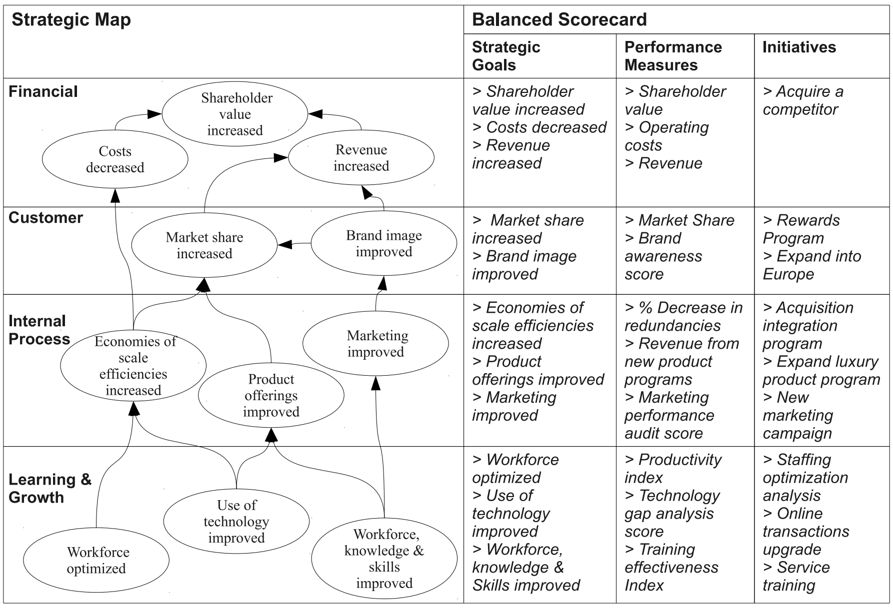

Figure 2.1: A strategic map and a balanced scorecard for BestTech Inc.

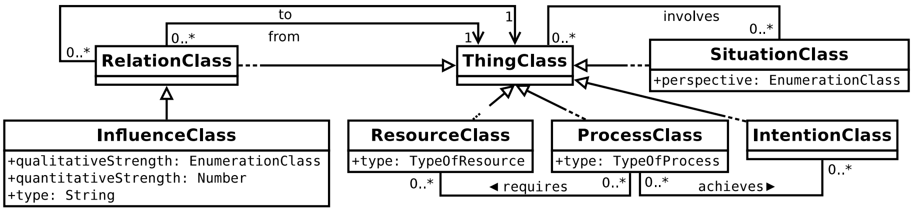

Figure 2.2: `Intention`, `Process` and `Situation` metaclasses and associated relationships in the BIM metamodel.

In this classification, internal factors are situations further classified into *Strengths* or *Weaknesses* while external factors are *Opportunities* or *Threats*. For example, a *Threat* for Wal-Mart may be "Exposure to political problems in the countries that we operate in". Notice that a strength with respect to one `Intention` may well be a weakness for another.

Figure 2.3 shows a small example that instantiates the metaclasses of Figure 2.2. Goals (i.e., `Intentions`), such as "Shareholder value increased" are decomposed into subgoals. Subgoals may be *mandatory* or *optional* (the notation used here is adapted from feature diagrams [37]). Goals may be achieved by carrying out `Processes`. In turn, `Processes` require `Resources`; there are various types of `Resources` with associated icons. `Situations` can influence goals but also `Processes`, `Resources`, etc. Opportunities and weaknesses are particular kinds of influences. Influence relationships [3] can actually exist between any two `Things` and play a pivotal role in governance models.

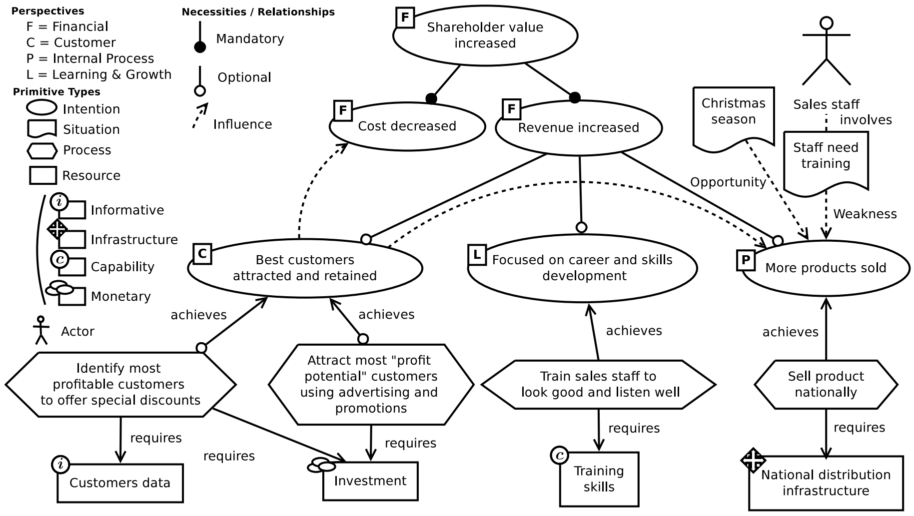

Figure 2.3: An example of `Situation`, `Intention` and `Process` concepts at schema level.

2.2 Actors

Figure 2.4 shows the `Actor` type and its relationships. An important relationship is the "responsible for" that defines which `Actor` is liable to be called on to (legally) answer when `Intentions`, `Processes`, `Resources`, and `Directives` are not acting in accordance with the business of an organization. For example, an `Actor` can be responsible for enforcing a work-site safety policy (i.e. a `Directive`). Indeed, if an accident occurs due to inappropriate safety conditions or there is a problem during a safety inspection, the responsibility will fall upon the `Actor`.

`Actors` must also comply with `Directives`, e.g., an employee must wear the protective equipments, and are able to define them, e.g., the executive board can define a policy such as "Pizza must be delivered within 20 minutes from the order or it will be free.".

[3]The instantiation of the *EnumerationClass* type for the *qualitativeStrength* attribute can vary depending on the nature of `Things` involved in the relationship.

An `Actor` can desire `Intentions` which are satisfied by other `Actors`; for example, the "More products sold" `Intention` is desired by the executive board and is satisfied by the sales staff under the responsibility of the sales manager. An `Actor` is also capable of a `Process` (or an `Action`) that can actually performing or not.

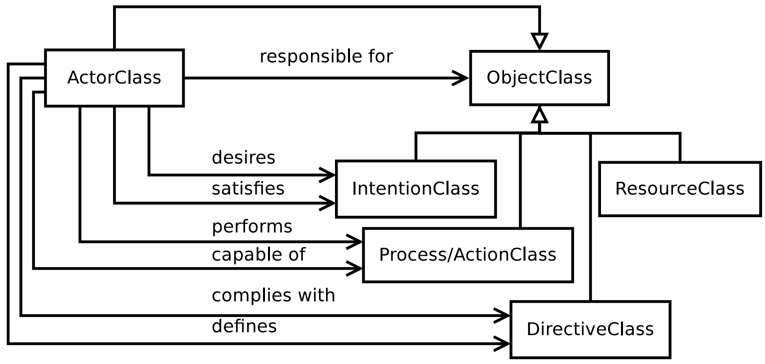

Figure 2.4: The `Actor` primitive type and the associated relationships.

Figure 2.5 shows an example of `Actors` and their relationships with the business environment.

Notice how cardinality values can be used for enriching the semantic of relationships in order to describe the minimum and maximum number of associated elements within a set. For example, in the figure, the sales department is constituted by a sales manager and from a minimum of six to twelve sales employees. In particular, regarding the "Package product" `Action`, a minimum of three employees are capable of the "Package product" `Action` and at least two must perform this `Action` on a maximum of ten that can be allocated to such `Action`.

Moreover, as described in Figure 2.6, an `Actor` can be specialized in `Agent`, `Role`, and `Position`. As defined in i^* [41], a `Role` is an abstract characterization of the behavior of a social `Actor` within some specialized context or domain of endeavor, and a `Position` is a set of `Roles`, usually played by one `Agent`. An `Agent` is an `Actor` with concrete, physical manifestations, such as a human individual or an artificial component of a system (hardware/software agents). Finally, an `Agent` can occupy a `Position`, while a `Position` covers a `Role`. Figure 2.6 shows an example of `Agent`, `Role` and `Position`.

2.3 Indicators

`Indicators` evaluate the quality of `Objects` to ensure compliance to internal policies and external directives.

Figure 2.8(a) shows the `Indicator` metaclass, consisting of attributes such as *target*, *thresholds*, *extreme values*, etc.[33]. The attributes 's description is provided in Table 2.2.

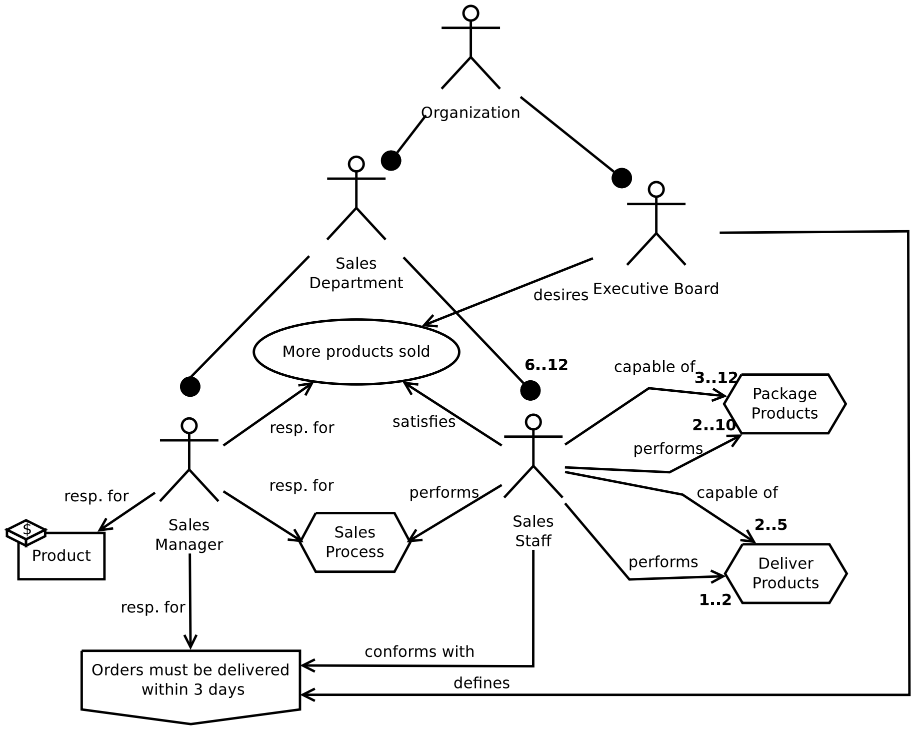

Figure 2.5: An example of `Actors` and their relationships with the business environment.

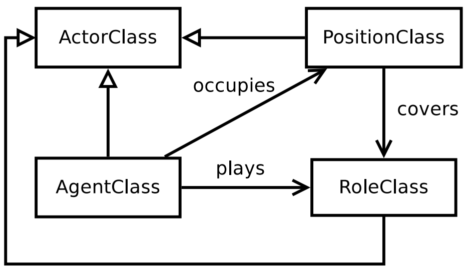

Figure 2.6: `Agent`, `Role` and `Position` in the BIM metamodel.

Notice that, in Table 2.2, the *evaluationTime* represents the timestamp associated to a specific current value instance, i.e., the time when the instance is calculated and created (see Section 2.5 for more details). This "system" time is different from the "domain" time (e.g. a Time dimension) used to navigate and calculate `Indicators` in a historical period of time, e.g., the *Time* dimension's value "March, 2009" to calculate

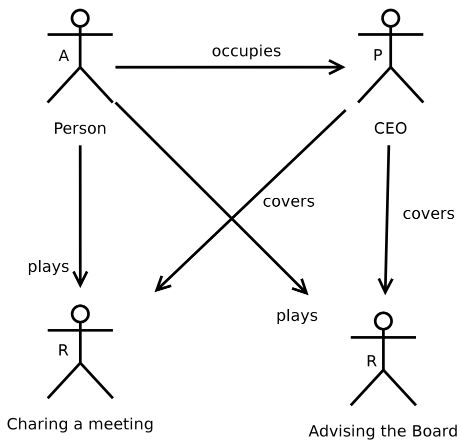

Figure 2.7: An example for `Agent`, `Role` and `Position` at schema level.

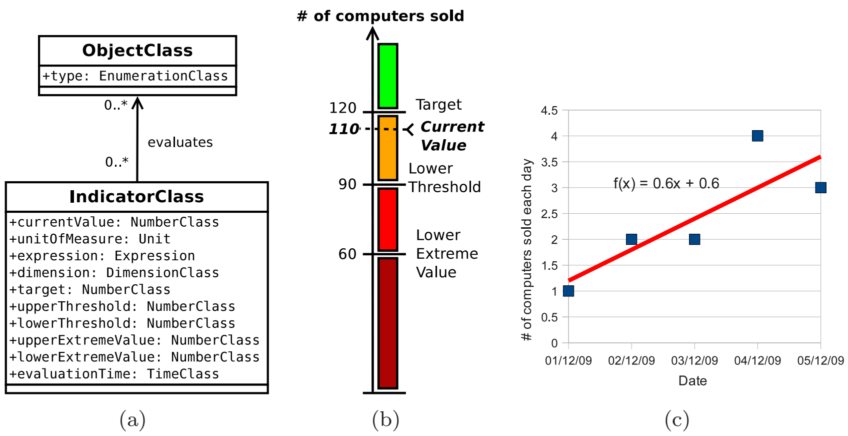

Figure 2.8: (a) The `Indicator` primitive type in the BIM metamodel. (b) An example of visual notation for `Indicators` at instance level. (c) An `Indicator` 's Trend example.

the number of products sold in March.

An example of `Indicator` and the use of its attributes is shown in Figure 2.8(b). Notice how the current value (*currentValue*) is found between *Lower Threshold* and *Target*; therefore, the company has almost reached its target in selling computers. The current value is calculated at *evaluationTime* using the metrics defined in the *expression* field. In general, BIM relies on the *Object Constraint Language* (OCL) [31]

Meta-attribute's name	Description
currentValue	the current value of an `Indicator` which is calculated through the evaluation of a metric's expression.
unitOfMeasure	the unit of measure associated with the current value of an `Indicator`.
expression	the metric's expression used to calculate the current value of an `Indicator`.
dimension	It allows to navigate (calculate the value of) an `Indicator` along specific directions of interest.
target	It is a value which allows to quantify the satisfaction level of an `Intention` (or, more in general, a performance level desired for a property of an `Object`). If an `Indicator` reaches the Target the associated `Intention` is satisfied.
(upper & lower) *threshold*	It defines a threshold (upper or lower) value that can be assumed by an `Indicator` that represents a critical situation for the Organization's business.
(upper & lower) *extremeValue*	It defines the worst (upper or lower) value that can be assumed by an `Indicator` which represents a critical (extreme) situation for the Organization's business.
evaluationTime	The time in which the current value of an `Indicator` is calculated.

Table 2.2: The description of `Indicator`'s attributes.

[4] to define such expressions. Moreover, one or more *dimensions* can be used to calculate and navigate an `Indicator` along specific directions of interest, e.g., the number of "Apple" computers sold.

BIM also supports the definition of operations, such as *Trend*, *Risk*, *Reward* or *Confidence*, at the class level (i.e., at the schema level), which calculate additional information (*meta values*) for the current values assumed by an `Indicator`. The result is in-depth information on the subject of an `Indicator`. Table 2.3 provides a brief description of such methods while Figure 2.8(c) is an example of a *Trend* operation based on a linear regression method. The result is a trend with a slope of +0.6 which means that: i) the trend is *positive*, and ii) the prediction of computers sold on Dec. 6, 2009 is about four units.

In order to better understand the use of `Indicators`, take a look at the following example:

Suppose that a Car Dealer has in store 16 units of "Luxury car" that it desires to sell over a period of one month (from December 1^{st}, 2009 to December 31^{st}, 2009).

[4] The OCL is a declarative language for describing rules that apply to UML models developed by the *Object Management Group* (OMG) and now part of the UML standard.

Method's name	Description
trend	It represents a general movement over time of a statistically detectable `Indicators`'s change.
risk	It provides the actual loss associated to the current value assumed by an `Indicator`.
reward	It provides the actual gain associated to the current value assumed by an `Indicator`.
confidence	It provides the actual confidence associated to the current value assumed by an `Indicator`. The confidence is based on i) the quality of the information used to calculate the current value, ii) the reputation of the source who provided the information and iii) the reliability of the method used to calculate the current value.

Table 2.3: The description for the `Indicator`'s operations.

Suppose also that the actual (partial) state of the world, i.e. on December 6^{th},2009, is as described in table 2.4 in which each car is identified by a Vehicle Identification Number (VIN).

(Desired) State of the world	Evaluation
Car (VIN 1HGBH41J8MN109180) Sold	**True**
Car (VIN 1HGBH41J8MN109181) Sold	False
Car (VIN 1HGBH41J8MN109182) Sold	**True**
Car (VIN 1HGBH41J8MN109183) Sold	False
Car (VIN 1HGBH41J8MN109184) Sold	**True**
Car (VIN 1HGBH41J8MN109185) Sold	False
Car (VIN 1HGBH41J8MN109186) Sold	False
Car (VIN 1HGBH41J8MN109187) Sold	**True**
Car (VIN 1HGBH41J8MN109188) Sold	False
Car (VIN 1HGBH41J8MN109189) Sold	**True**
Car (VIN 1HGBH41J8MN109190) Sold	False
Car (VIN 1HGBH41J8MN109191) Sold	**True**
Car (VIN 1HGBH41J8MN109192) Sold	**True**
Car (VIN 1HGBH41J8MN109193) Sold	False
Car (VIN 1HGBH41J8MN109194) Sold	False
Car (VIN 1HGBH41J8MN109195) Sold	**True**

Table 2.4: The (desired) states of the world for the "Car sold" `Intention`.

We have the following `Indicator` as described in Figure 2.9 and in Table 2.5. Notice the use of the cardinality (0..16) to express the number of cars the organization desires to sell. Moreover, Figure 2.10 shows the different zones defined by using the *extreme*, *threshold* and *target* values; and the current value for December 6^{th}, 2009 lying on the red zone.

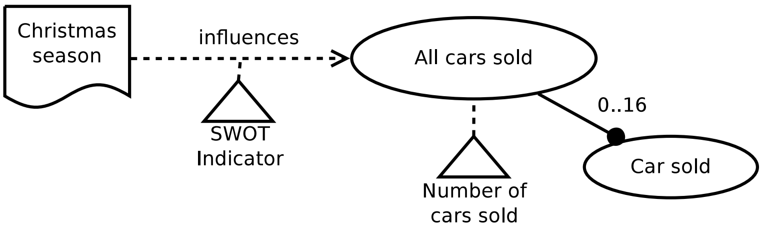

Figure 2.9: An example of `Indicators` at schema level.

Meta-Level Indicator	**Schema-Level** Number of Cars Sold	**Instance-Level** Inst 1:Number of Cars Sold
currentValue	Integer	8
unitOfMeasure	cars sold	-
expression	"*context* Car *select* (car | car.status='Sold' $\wedge$ car.type='Luxury car') $->$ size()"	-
target	Integer	13
(Lower) threshold	Integer	10
(Lower) extremeValue	Integer	7
dimension	TypeOfCar	"Luxury Car"
evaluationTime	Time	Dec 6, 2009

Table 2.5: An example of an `Indicator` at different levels of modelling.

Figure 2.11 and Figure 2.12 show examples of `Indicator`'s operations. In particular, the former is an example of a Trend *Operation* based on a linear regression [5] method. The operation calculates the slope (or gradient) of the trend line fitting the time series provided in input, where the time series are the number of cars sold from Dec. 1^{st} to 2009 to Dec. 6^{th}, 2009. The result of the operation is a slope of -0.7 which means that on Dec. 6, 2009 the Trend for the number of cars sold is negative.

The latter, Figure 2.12, is an example of risk and reward operations which allow to associate specific loss or gain values to any current value. In fact, if we suppose that 30,000 USD is the total cost sustained for a car while 50,000 USD is the revenue obtained by its selling, we have that: i) a total gain of 170,000 USD is obtained when

[5] Notice that, different ways for evaluating a Trend exist, such as *logarithmic*, *exponential*, etc.; in this case we use the simple one, namely the *linear regression*.

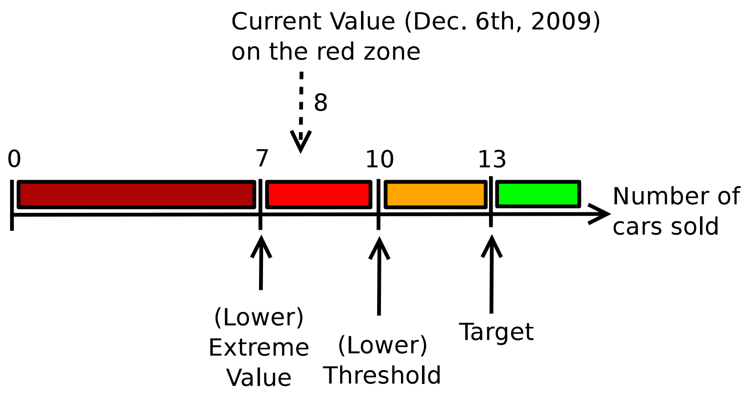

Figure 2.10: The current value of the “Number of cars sold” `Indicator` on December, 6^{th}.

the target is reached, ii) a loss of 130,000 USD is endured when the current value assumes the extreme value and iii) a loss of 80,000 USD is endured on the current value calculated on December 6, 2009.

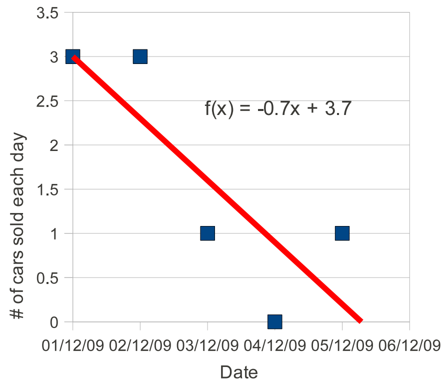

Figure 2.11: The Trend for the “Number of cars sold” `Indicator`.

Figure 2.9 shows another `Indicator`, namely the SWOT Indicator, which is used to evaluate the influences among `Situations` and `Intentions`. As described in Section 2.1, we have four kind on influences, namely, *Strength*, *Weakness*, *Threat*, and *Opportunity*. For each type, it is possible to define a quality or quantity *power scale*, e.g., $< high, medium, low >$ or $[0, 100]$, in order to determine the degree of the influence. Therefore, a SWOT Indicator can assume a current value that ranges among values of the power scale and can have a unit of measure equal to *Strength*, *Weakness*, *Threat*, or *Opportunity*. As described in Section 4, the SWOT Indicator's current value is used

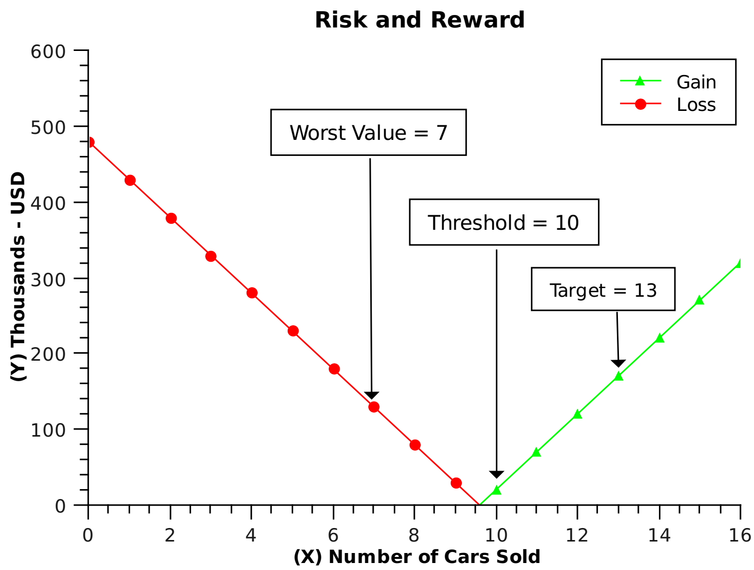

Figure 2.12: The *Loss* and *Gain* values for the "Number of cars sold" `Indicator`.

by the *InfluenceClass* metaclass to support different kind of analysis.

2.4 Objects' Life-cycle and Behavior: A Finite State Machine Model

A *Finite State Machine* (FSM) [13] provides a simple and effective means to control the **life-cycle** and overall **behavior** of BIM's `Objects`. In fact, a FSM is an *abstract computational model* which allows to define for each `Object`: i) a set of different *states* assumed by `Objects`' instances in the real world; ii) the *transitions* among these different states in order to define the (computational) behavior; iii) the *events/inputs* which express the stimuli taken into account; and, iv) the *actions/outputs* which are the possible *responses* that can be generated.

Formally, a FSM is a multi-tuple $FSM = (\Sigma, \Lambda, S, s_0, \delta, \omega)$, where:

- (Σ) is the **input alphabet** of symbols representing external stimuli (inputs or events) that are used by transition functions;
- (Λ) is the **output alphabet** of symbols representing responses (outputs or actions) that are provided by output functions;
- (S) is the set of possible **states** which are conditions of the state machine at a certain time;

- ($s_0 \in S$) is the **start state**;
- ($\delta : S \times \Sigma \rightarrow S$) is the state **transition function**. Based on the *current state* $s_c \in S$ and an input symbol $i \in \Sigma$, it computes the *transition* to the next state $s_n \in S$;
- ($\omega : S \times \Sigma \rightarrow \Lambda$) is the **output function** (as defined in the *Mealy model*).

As shown in the following subsection, the BIM also allows to associate to each state a *Time* attribute which stores a *timestamp* value representing the last time in which an `Object` enter in that specific state.

In general, for all `Objects`, it is possible to identify a *START* state and a (possible) *END* state. The *START* state allows to define when an instance of the real world becoming an instance of a specific type of the BIM. For example, the "Car VIN=1HGBH41J8MN109180 sold" instance stars to be an `Intention`'s instance when the car dealer defines a clear statement to sell that specific car, i.e. the car with VIN=1HGBH41J8MN109180. Than, the car dealer will be able to pursue it, i.e., the state *BEING PURSUED*.

In the opposite way, the *END* state allows to define when the same instance stops to be an (active) instance of that type [6]. For example, the "Car VIN=1HGBH41J8MN109180 sold" *Intention*'s instance stops to be an `Intention`'s instance when the car dealer stops to pursued it either because it is achieved, failed or aborted (i.e.,the car dealer has changed his mind).

Moreover, for each specific `Object`, such as the `Intention`, it is also possible to define specific states which have a particular semantic into the business environment. For example, the above *BEING PURSUED* state can be defined for `Intentions` to express the continuing activity by an `Actor` to achieve a specific `Intention`.

Figure 2.13 and Figure 2.14 show, respectively, possible FSM diagrams for `Intention` and `Resource`. Notice that, a *Vision* cannot be fitted in the diagram illustrated in Figure 2.13 since, usually, it is not possible to define a SATISFIED state or an END state for such a concept. Therefore, if it is necessary, a specific FSM diagram must be defined or, alternatively, the same FSM diagram can be used in which some states will be never assumed by the instances.

Moreover, it is also important to notice the two "pass Deadline" and "pass Expiring date" events which, respectively, lead to a FAILED `Intention`'s state and to an EXPIRED `Resource`'s state.

These events are fired when: i) the current time in the system is greater than, respectively, the "deadline date" and the "expiring date" defined in the business environment (see next subsection); and, ii) the actual states of `Intention`'s and `Resource`'s instances are, respectively, BEING PURSED and BEING CONSUMED. FSMs allows to define these events as *guards* which can be expressed by using constrains.

In this way, for example, it is possible to represent situations where a goal's deadline has passed and, since it is not longer important whether or not the goal will be satisfied, we transit to a FAILED state.

[6] Stop to be an "active" instance means that the instance is still recorded in the system for "history" purpose (i.e., for analysis and queries) but is no more used for operative tasks.

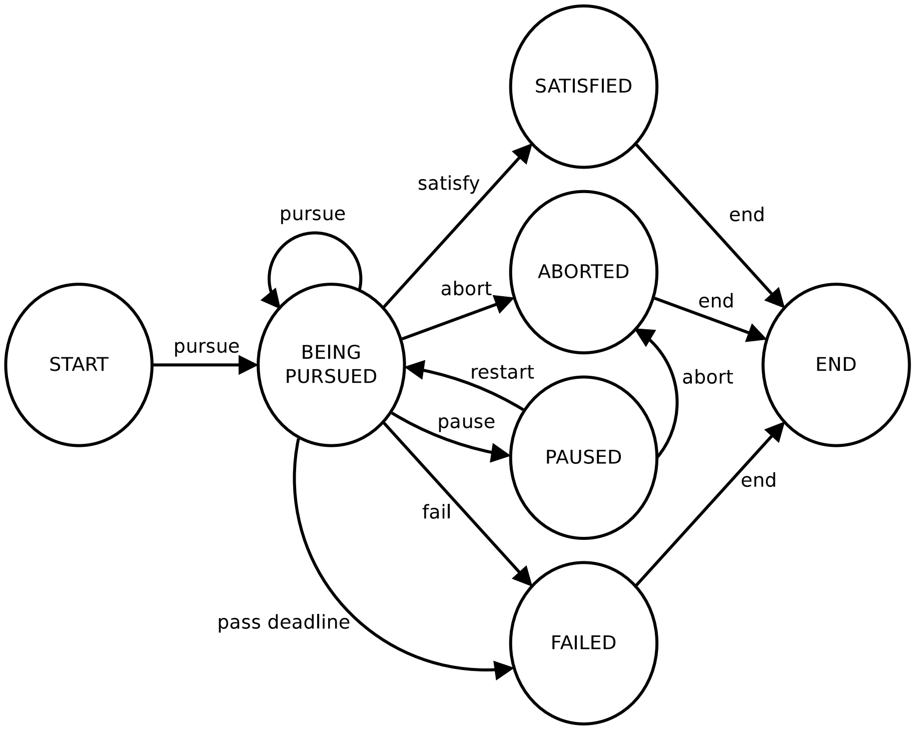

Figure 2.13: A FSM diagram for `Intention`.

A similar situation exists for `Resources` in the case we have passed the expiring date. In fact, the `Resource` stops to be a valid `Resource` loosing its `intrinsic` properties that make it such as a `Resource` in the business environment; however, we can still consume it with all the relative consequences.

The use of states and timestamps enable the definition of interesting queries over `Objects` belonging to a schema. For example, we can express queries such as (1)="List all goals that are not yet satisfied" or (2)="Show the time in which <All car sold> was satisfied" that can be defined as:

$$\begin{aligned}(1) = &\ context\ Intention\\ &\ select\ (intention \mid not(intention.state =' SATISFIED'))\end{aligned}$$

$$\begin{aligned}(2) = &\ context\ Intention\\ &\ select\ (intention \mid intention.name =' All\ car\ sold'\ and\\ &\ intention.state =' SATISFIED').time\end{aligned}$$

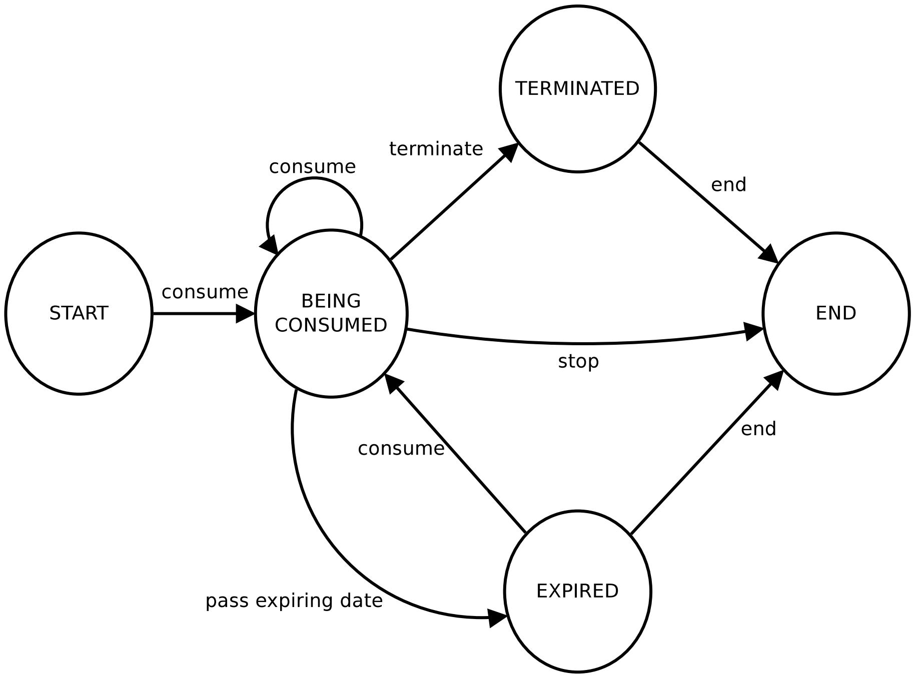

Figure 2.14: A FSM diagram for `Resource`.

2.5 Evolution Timeline and Time Constrains for Objects

The previous subsection presented the concept of *State* to address issues related to the life cycle and behavior of `Objects`. However, since we want to describe the "evolution" of `Objects` within a business environment, we need to introduce the concept of `Evolution Timeline`. Figure 2.15 shows the fragment of the BIM metamodel aimed to describe such aspect.

As described above, each object can have an `Evolution Timeline` which represents its "lifetime". On this lifetime line, two types of `Timepoints` can be defined: i) *time constrains* for the definition of time constrains, such as a deadline for an `Intention`, and ii) *timestamps* to define when an instance enters into a specific state, such as when an `Intention`'s instance assumes a SATISFIED state.

Notice that, for now, we are not constraining an `Evolution Timeline` to be associated to only one `Object`. In fact, as shown by the model, a same timeline can be shared among different `Objects`. Indeed, we think that having a unique timeline for the entire organization on which `Timepoints` belonging to different `Objects` can coexist, will be useful for analysis activities.

Finally, notice the relationship among `State` and `Timepoint` which was also described in the previous subsection. Each `State` can have multiple `Timepoints` representing all the times (timestamps) an `Object` entered in that particular `State`. More-

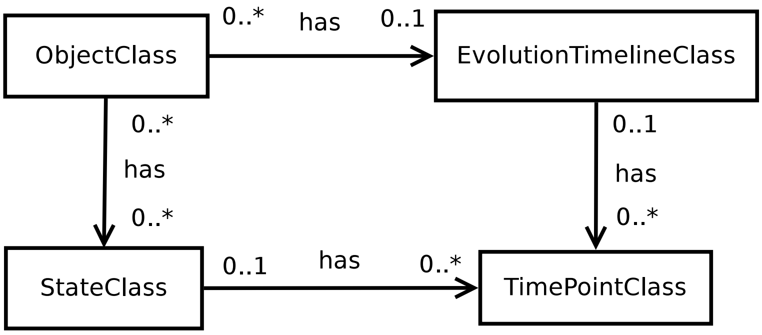

Figure 2.15: The `Evolution Timeline` concept.

over, a `State` might not have a `Timepoint` associated, i.e., the `Object` never assumed that `State`, and a `Timepoint` might not have a `State` associated, i.e., the `Timepoint` represents a time constrain for the associated `Object` and not a timestamp for a particular `State`.

Figure 2.16 shows an example describing the above concepts with respect to the `Intention` primitive type.

In this example, we can see how an `Intention` was defined on March 1, 2009 with a deadline fixed on March 30, 2009. Moreover, the `Intention` was started to be pursed on March 3, 2009 to be paused on March 11, 2009 and finally satisfied on March 23, 2009 (after it was re-pursed on March 13, 2009). Notice that, all the above `Timempoints` represent timestamps with the exception of the deadline `Timepoint` constrain.

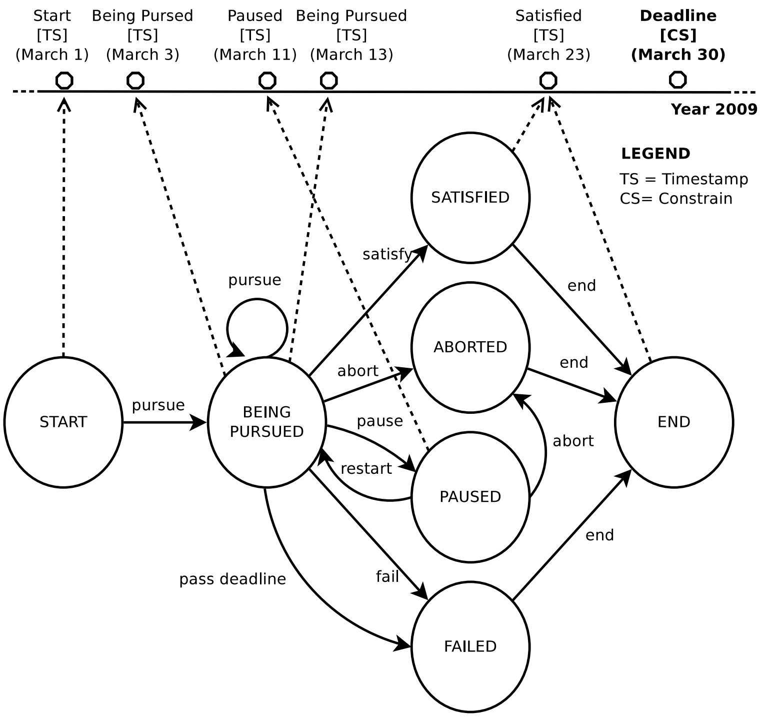

Figure 2.16: An example of `Evolution Timeline` for the `Intention` primitive type.

3 The BIM Repository and the Abstraction Mechanisms

A BIM Repository is a persistent location in which organization and business data are stored and maintained in order to be fetched to perform some particular task, e.g., analytics tasks (see Section 4). In particular, a BIM Repository consists of structured *classes*(or *types*) and *objects* (or *instances*) which are defined using the BIM metamodel.

In general, as with other modelling languages, classes and objects can be organized along the three dimensions of *aggregation*, *classification* and *generalization* (see [17]). As described in [20], the act of "abstracting a collection of units into a new unit is called aggregation"; indeed, an aggregation is a special type of association in which objects or classes are assembled or configured together to create a more complex object or class.

For example, the *John* object can be aggregated into the *Pizza Pizza Sales Department* object and respectively, the *Employee* class can be aggregated into the *Department* class. As we will describe in Section 3.1, we adopt the feature model [20] in order to allow users to be flexible during the aggregation activity.

The classification dimension calls for each object or class to be *an instance* of one or more generic classes or *metaclasses*. In fact, referring to the previous example, *John* is an instance of *Employee* while *Employee* is an instance of the *ActorClass* metaclass. The Classification dimension is also used for relationships belonging to the model; in fact, we can have an *HomeAddress* object which can be classified by an *Address* class which, similarly, can be classified by an *AddressClass* metaclass. More details about Classification is provided in Section 3.1 which describes the four meta layer metamodelling architecture of the *Meta Object Facility* (MOF) [30] and the features inherited from *Telos* [29].

Classes and Metaclasses can be specialized along *generalization* or *ISA hierarchies*. As defined in [20], generalization is the act of "abstracting the commonalities among a collection of units into a new conceptual unit suppressing detailed differences". For example, an `Employee` class may have subclasses such as `Clerk`, `Sales Person`, etc.; similarly, an *ObjectClass* metaclass may have submetaclasses such as *EntityClass*, *SituationClass*, etc. Notice that ISA hierarchies are orthogonal to the classification dimension; therefore, all the above subclasses, i.e., `Clerk`, `Sales Person`, etc., should be instances of the `ActorClass`.

In the next subsections, further details of aggregation, classification, and generalization are provided. In particular, in Section 3.4 a possible implementation of the abstraction mechanisms for the BIM metamodel is illustrated and described.

3.1 Aggregation: Mandatory, Optional, OR and Alternative

The BIM provides a flexible way for the aggregation activity that allows to have a direct control in the choice of the parts (i.e., the "partOf" relationship) constituting an `Object`.

Indeed, the BIM aggregation mechanism uses the same approach described in [20] in which the relationships between a parent and its children are categorized as:

- *Mandatory* – a child is required,
- *Optional* – a child is optional,
- *Or* – at least one of the children must be selected,
- *Alternative (xor)* – exact one of the children must be selected.

Figure 3.1 provides an example which shows the visual notation used to aggregate different `Intentions`. The example, also show how cardinalities can be used to add more semantic during the aggregation activity; e.g., the "iPods produced" aggregates from one to one million "one iPod produced". Moreover, since the "one iPod produced" is mandatory the range of cardinality must start from "one".

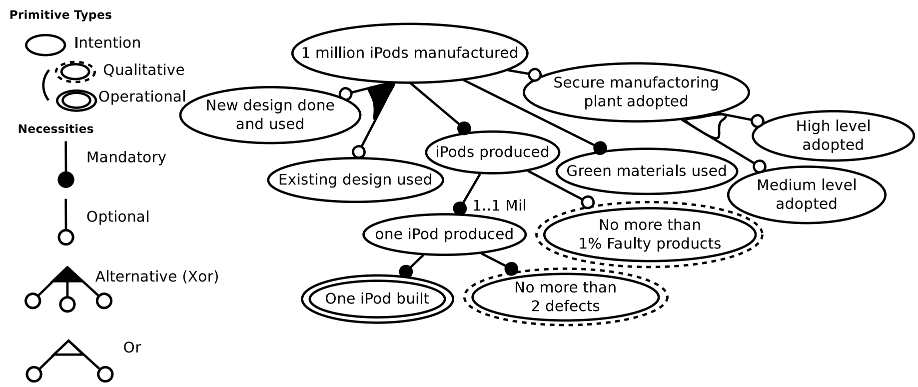

Figure 3.1: An `Intention` aggregation hierarchy.

In Section 3.4, an underlying model for the aggregation mechanism is described.

3.2 Classification: The OMG Four-Layer Metamodel Architecture

The BIM metamodel is designed to be aligned with the OMG four-layer metamodel architecture [30] which is summarized in Table 3.1.

Layer	Description	Example of model elements
M3: meta-metamodel	Defines the language for specifying metamodels.	*MetaClass, MetaAttribute, MetaOperation*
M2: meta-model	An instance of a meta-metamodel. Defines the language for specifying a model.	*Class, Attribute, Operation, Component*
M1: model	An instance of a metamodel. Defines a language to describe an information domain.	*StockShare, askPrice, sellLimitOrder, StockQuoteServer*
M0: user objects (user data)	An instance of a model. Defines a specific information domain.	$< Acme\_Software\_Share\_98789 >$, 654.56, $sell\_limit\_order$, $< Stock\_Quote\_Svr\_32123 >$

Table 3.1: The OMG four-layer metamodel architecture.

In this architecture, a model at one layer is used to specify models belonging to the layer below. Similarly, a model at one layer can be seen as an instance of a particular model in the layer above.

Usually, models at M_n layer have an higher level of abstraction and are typically more compacts than models at M_{n-1} layer. In fact, models at M_{n-1} are more elaborate than the models at M_n layer that describe them. Figure 3.2 shows an example where the top layers M3 and M2 of the architecture are represented and specified, respectively, by the MOF meta-metamodel and by the UML metamodel.

The BIM metamodel is at the same layer of the UML metamodel, i.e, at the M2 layer. Therefore, we would consider the BIM metamodel to be an instance of the MOF meta-metamodel. We want this for two reasons: i) MOF enables the interoperability of model and metadata driven systems; and, ii) MOF is quite spread across industry. In this way, our model can exploit the interoperability provided by MOF in order to facilitate its eventual integration with industry's models and systems.

However, a major problem is that MOF (and UML) suffers from the "shallow" instantiation problem [6].

Basically, a class can only define the semantic of its direct instances, but it has no effect on entities created by further instantiation steps.

This is caused by the old "two-levels only" modeling philosophy which does not adequately support a multi layer architecture. In fact, although model elements in a multiple layer architecture can represent both objects and classes, i.e., an object at M2 layer can be seen as a class for objects at M1 layer, a class can never receive attributes and associations from its classifier, but only slots and links, thus leading to the shallow instantiation problem (see [6] for further details).

The BIM metamodel needs to be able to influence both the M1 layer and the M0 layer in order to constrain designers in the choice of domain concepts and relationships at M1 layer but also to propagate such "semantic" constrains (when requested) on instances at M0 layer.

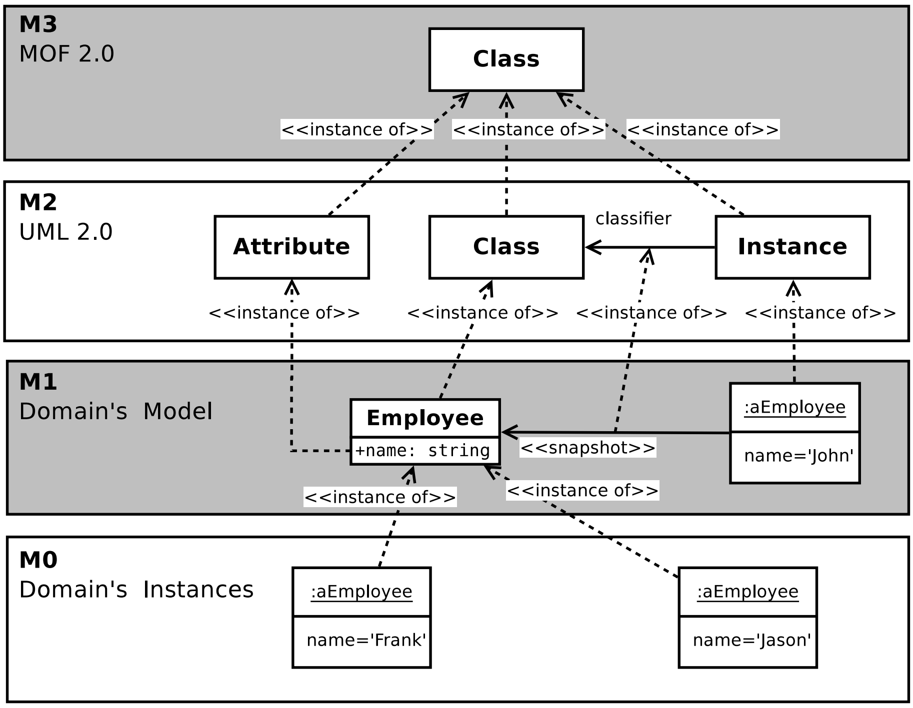

Figure 3.2: An example for the four-layer model architecture which uses MOF and UML.

Figure 3.3 shows an example of such *deep* instantiation in which some "instance of" links are missing to simplify the illustration. Notice how:

1. *associations*, such as "evaluates", can be propagated across multiple layers;
2. some *attributes*, such as "currentValue", can be propagated across multiple layers while refining their types, e.g., the type of *currentValue* is *NumberClass* at M2 while is *Integer* at M1;
3. some *attributes*, such as "metric", can be limited to specific layer, i.e., metric is instantiated at M1 while is disappearing at M0;
4. some *attributes*, such as "director", can be freely defined by the designer at M1 but are not specified at M2, i.e., they are domain specific and are not mandatory by the BIM metamodel.

In order to support such deep instantiation, the BIM is inspired by the Telos language [29] in which classes, attributes, and associations [1] are collectively referred to

[1] In Telos associations are represented using attributes which are binary relationships between entities, i.e. classes, or other relationships.

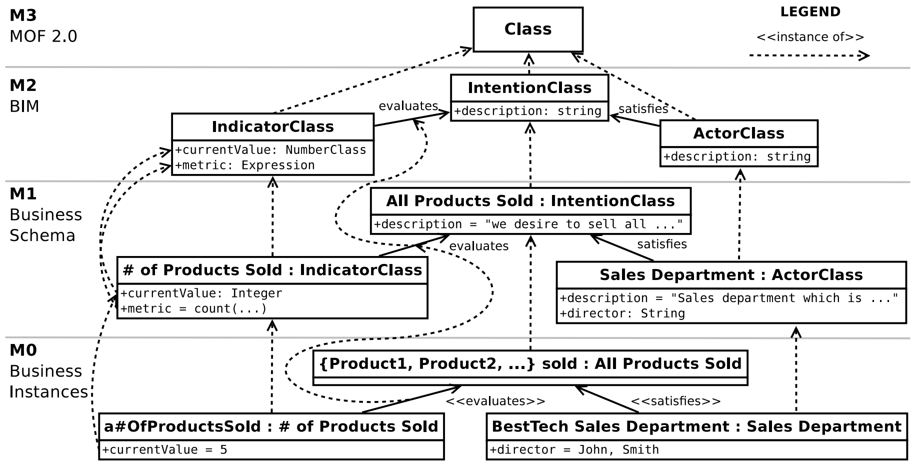

Figure 3.3: An example of the *deep* instantiation concept required in the BIM.

by the term "proposition" and are treated uniformly by the structuring mechanisms of aggregation, classification, and generalization. Therefore, as shown in Figure 3.4, we can have metaclasses, classes, objects; but also: i) metattributes, attributes, slots; and, ii) metassociations, associations, links [2].

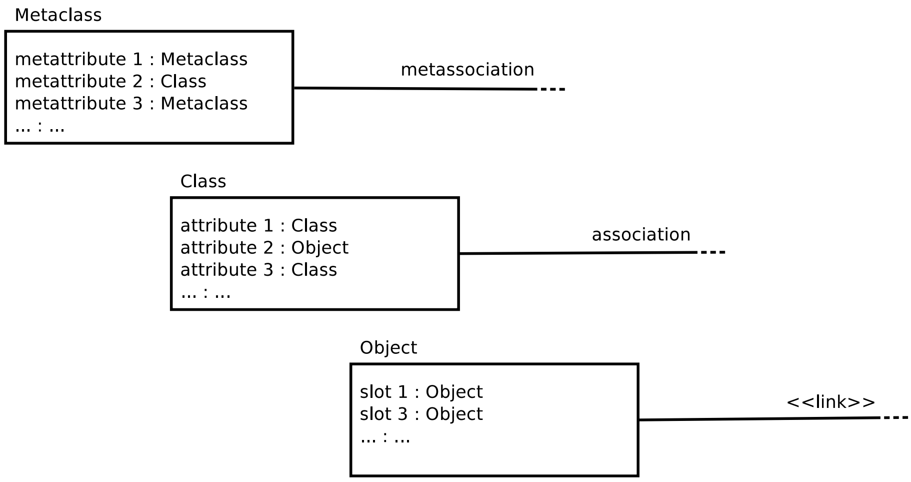

Figure 3.4: The Metaclass, Class and Objects instantiation.

A careful reader can observe that the UML metamodel introduces *Instance* metaclasses in order to "link" objects of different types at the M0 Layer. Although this can

[2] In Telos is also possible to have meta-metaclasses, meta-metattributes, meta-metassociations and so on (although for our aim it is not necessary).

resolve the instantiation of associations at the M0 layer, this approach arises issues such as the "ambiguous classification" and the "replication of concepts" [6]. Moreover, it can increase the complexity of the model and lead to inconsistencies and the losing of precision; and, it does not satisfy the "need" of propagating attributes across different layers.

3.3 Generalization: Subtypes and Business Terms Specialization

The primitive types presented in Section 2 represent the model elements provided by the BIM metamodel for the description of the different concepts belonging to a generic business environment. Moreover, in order to describe particular instances of the real world, the BIM metamodel defines a set of *subtypes* whose semantic is described in Table 3.2.

Moreover, in order to cover and map to business terminologies, such as a vision, mission, or strategy, we use *metaproperties* such as (i) short-/long-term, (ii) many/few instances, (iii) formal/informal definition, and (iv) chances of success. Clusters of terms from a business glossary, such as *Vision*, *Strategic/Tactical goal*, *Softgoal*, *Objective* are then represented in terms of a single BIM primitive concept (`Intention`) but each has different combinations of values for the four metaproperties. For example, a *Vision* is a long-term `Intention` without a formal definition, which is likely to only have a few instances (usually one) whose chances of success are low (depending on many uncertain factors).

To represent business terms, the BIM metamodel defines an attribute associated to the `ThingClass` metaclass, called `type` which is inherited by all the other metaclass in the metamodel to store specific terminology for each `Thing`. Table 3.3 [3] provides common clusters for some of BIM's primitive concepts.

Figure 3.5 shows an example of subtypes and business terms specialization for the `Intention` primitive type.

3.4 An UML Class Diagram for the BIM's Abstraction Mechanisms

In this section we present a possible UML class diagram for the BIM metamodel, with respect to the aggregation, classification, and generalization mechanisms described in the previous sections.

The model is shown in Figure 3.6. The `RefinementLinkClass`, the `NodeElementClass` and the `AggregationClass` are parts of the aggregation mechanism. In particular,

[3] To define the set of terms illustrated in Table 3.3 we analyzed both the scientific literature, e.g., the Business Motivation Model (BMM) [3], and the business world, e.g., the www.businessdictionary.com [4] site; however, the business terminology can be easily customized for the domain at hand.

Primitive Type	Subtype	Subtype Description
Intention	Operational Intention	An atomic `Intention` which has a very strict and clear logical criterion of satisfiability and can be achieved by an operational process or activity.
	Qualitative Intention	An atomic `Intention` which has not a clear-cut criterion for its satisfaction and can be claimed only when there is sufficient positive and little negative evidences (or unsatisfaction in the opposite case).
Actor	Agent	Actor with concrete, physical manifestations, such as a human individual. We use the term agent instead of person for generality, so that it can be used to refer to human as well as artificial (hardware/software agents). An agent has dependencies that apply regardless of what roles he/she/it happens to be playing. These characteristics are typically not easily transferable to other individuals, e.g. its skills and experiences, and its physical limitations [1].
	Role	Abstract characterization of the behavior of a social actor within some specialized context or domain of endeavor. Its characteristics are easily transferable to other social actors. The dependencies associated with a role apply regardless of the agent who plays the role [1].
	Position	Intermediate abstraction that can be used between a role and an agent. It is a set of roles typically played by one agent (e.g., assigned jointly to that one agent). We say that an agent occupies a position. A position is said to cover a role [1].

Table 3.2: Subtypes belonging to the BIM metamodel.

BIM Concept	Business Terms
Intention	Vision, Strategic/Tactical Goal, SoftGoal, Objective
Process	Mission, Strategy, Tactic, Initiative, Business Process, Activity
Actor	Organization, Business Unit, Human person, System Application
Resource	Monetary / Infrastructure / Economic Good / Information / Human / Capability Resource
Directive	Policy, Rule

Table 3.3: An example of business terms captured with the `Thing`'s attribute *type*.

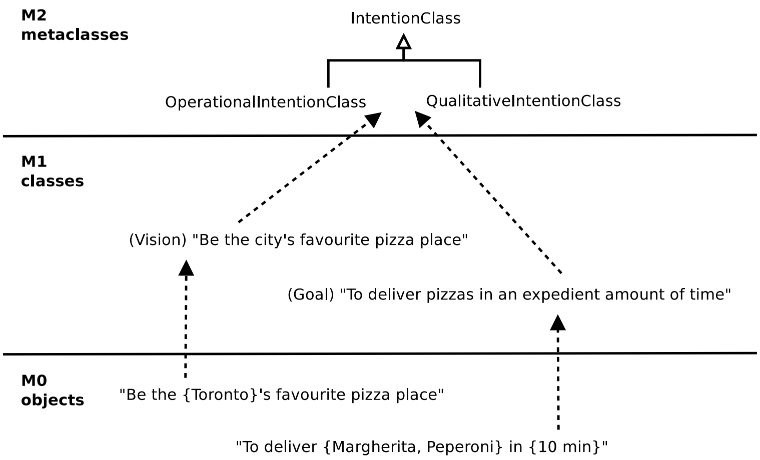

Figure 3.5: An example of subtypes and business terms specialization for the `Intention` primitive type.

the attribute `mandatory` of the `RefinementLinkClass` allows to specify if the "refiner" component in the refinement relationship is mandatory or optional; while, the attribute `type` of the `AggregationClass` allows to specify the type of the aggregation, i.e., OR, AND, XOR, Alternative, performed on the sub-components.

In regard to the generalization mechanism, the class named "..." (specializing the `ThingClass`) represents the different primitive types presented in Section 2. Instead, the attribute `type` belonging to the `ThingClass` is used to specify the business terminology shown in table 3.3.

Finally, the right part of the model represents the state and the evolution timeline concepts describe in Subsection 2.4 and Subsection 2.5.

Figure 3.7 shows an example of how the model works underlying the feature model visual notation (see Subsection 3.1).

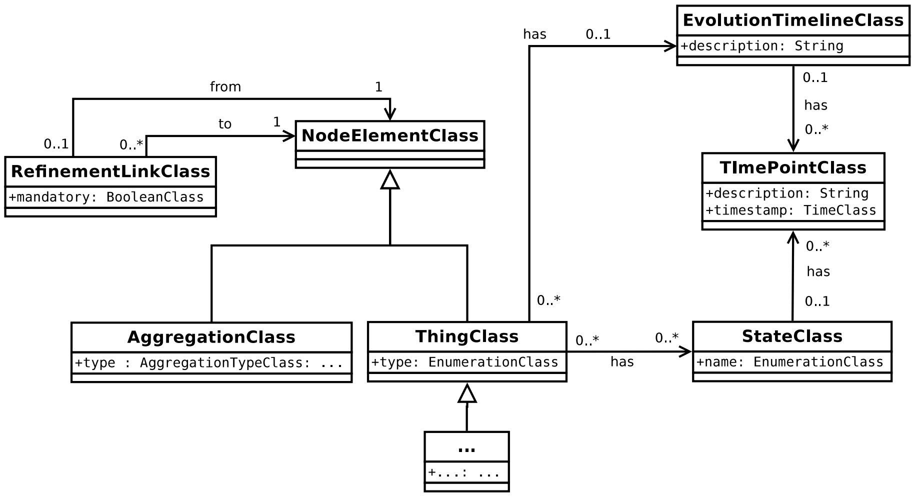

Figure 3.6: An UML Class diagram for the BIM's abstraction mechanism.

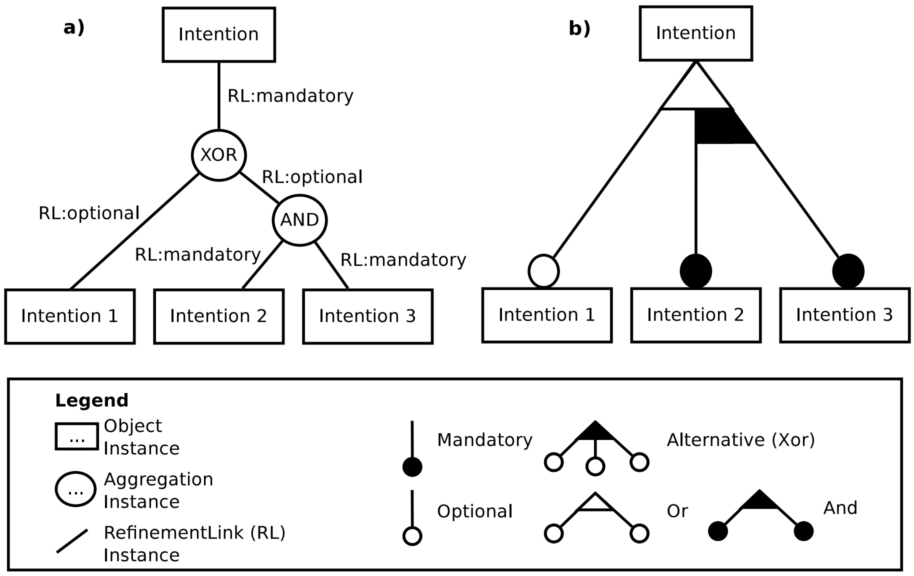

Figure 3.7: An example of aggregation using (b) the feature model visual notation and (a) the underlying UML Class Diagram.

4 Strategic Analysis through Mappings

We illustrate how the richness and flexibility of BIM can be used to represent widespread strategic planning models. Moreover, since the final aim of BIM is to support analysis activities for answering questions such as "What will happen next?" or "Where exactly is the problem?" [8], we describe how BIM can be projected onto different analysis models.

In particular, the following target models are considered:

- a goal reasoning model based on a formal goal model [14],
- the SWOT analysis model [9],
- the Strategic Map [22] model,
- the Balanced SCorecards (BSCs)[21](and Key Performance Indicators [33]) model.

4.1 Goal (Intention) Reasoning: A Formal (Axiomatic) Model

In the BIM, `Intention` analysis and reasoning are given a prominent role to help stakeholders in the definition of their intentions and relationships among them, such as conflicts and negative or positive contributions.

As we described in Section 2.1, the *Intention* primitive type can be used to define the hierarchy of the Vision, Goals, and Objectives of an organization in which nodes can be connected by influence links.

In this section, we want to provide an underneath algorithm which enables the reasoning on the `Intentions` belonging to such a hierarchy.

At this aim, we project the BIM toward the goal reasoning model described in [14]. In this work, the authors adopts a formal goal model to make the goal analysis process concrete through the use of forward and backward reasoning. Notice that, the model is used in the context of the Tropos methodology [2] which adopts the i^* [41] modelling framework for analyzing requirements (*Early Requirements* and *Late Requirements* [1]).

In particular, the formal model goals is used by the software engineer to cope with qualitative relationships and inconsistencies among goals during the early requirements phase.

[1] The former is concerned with understanding the organizational context within which the system-to-be will eventually function; the latter, on the other hand, is concerned with a definition of the functional and non-functional requirements of the system-to-be.

The formal model description resides in the definition of the notions of goal graphs and the axiomatic representation of goal relations. The goal graphs is defined trough a set of goal nodes G_i and of relations $(G_1, ..., G_n) \overset{r}{\mapsto} G$ over them, including the (n+1)-any relations *and*, *or* and the binary relations +S, -S, +D, -D, ++S, - -S, +,-++, - -. For a in depth description we remand to [14] while here we briefly recall the intuitive meaning of such relationships.

For *and* and *or* we have that:

- $(G_1, ..., G_n) \overset{and}{\mapsto} G$ means that G is satisfied (resp. denied) if all $G_1, ..., G_n$ are satisfied (resp. if at least one G_i is denied);

- $(G_1, ..., G_n) \overset{or}{\mapsto} G$ means that G is denied (resp. satisfied) if all $G_1, ..., G_n$ are denied (resp. if at least one G_i is satisfied);

For the other binary relationships, an example is provided by: $G_2 \overset{+S}{\mapsto} G_1$ (resp. $G_2 \overset{++S}{\mapsto} G_1$) means that if G_2 is satisfied, then there is some (resp. a full) evidence that G_1 is satisfied, but if G_2 is denied, then nothing is said about the denial of G_1.

To generalize the previous $G_2 \overset{+S}{\mapsto} G_1$ relationship, we said that, the "S" (resp. "D") symbol denotes the fact that the satisfiability (resp. deniability) value of the source goal, e.g., G_2, is propagated; the "+" (resp. "-") symbol denotes the fact that the propagation is positive (resp. negative), in the sense that satisfiability propagates to satisfiability (resp. deniability) and deniability propagates to deniability (resp. satisfiability).

Finally, the relations +, -, ++, - - are defined such that $G_2 \overset{r}{\mapsto} G_1$ is a shorthand for the combination of the two corresponding relationships $G_2 \overset{r_S}{\mapsto} G_1$ and $G_2 \overset{r_D}{\mapsto} G_1$, e.g., $G_2 \overset{+}{\mapsto} G_1$ is a shorthand for the combination of $G_2 \overset{+S}{\mapsto} G_1$ and $G_2 \overset{+D}{\mapsto} G_1$. The first kind of relationships are called *symmetric* and the latter two *asymmetric*.

Now, a set of four distinct predicates over goals are introduced to be used with ground axioms in order to reasoning on the goal model. They are: $FS(G)$, $FD(G)$ and $PS(G)$, $PD(G)$; which mean, respectively, that there is (at least) *full* evidence that goal G is satisfied and that G is denied, and that there is at least *partial* evidence that G is satisfied and that G is denied. In their work, the authors provide a set of ground axioms for the propagation rules which are *soundness* and *completeness*. An example of of relation axiom is: $G_2 \overset{+S}{\mapsto} G_1$: $PS(G_2) \rightarrow PS(G_1)$.

Given a goal graph and an initial values assignment to some goals, the underlying algorithm exploits the ground axioms for *forward* and *backward* reasoning tasks. In particular, for the forward reasoning the assigned goals are called input goals (typically the leaf goals) while for the backward reasoning the assigned goals are called *target goals* (typically root goals).

The aim of the forward reasoning is the propagation of initial values (i.e., the input goals) to all other goals of the graph; the user can look the final values of the goals of interest (i.e., the target goals).

Instead, the aim of the backward reasoning is the *backward search* of the possible input values (i.e., the input goals) leading to some desired final value (i.e.the target values), under desired constrains, e.g., avoiding conflicts among goals.

In general, the forward reasoning is used for evaluating the impact of the adoption of the different alternatives with respect to the root goals; while, the backward reasoning, is used to analyze goal models and find the set of goals at the minimum costs that if achieved can guarantee the achievement of the desired top goals and softgoals.

The algorithm of the formal goal model can be used within the BIM to allow such reasoning. In fact, the goal relationships are accounted for within BIM through the *InfluenceClass* metaclass illustrated in Figure 2.2.

Notice how the *qualitativeStrength* and the *quantitativeStrengh* allow, respectively, to record the qualitative (e.g., + or - -) or quantitative (e.g., 0.7 or -0.3) strength of an influence. The *type* attribute allows to specify whether the satisfiability or deniability is propagated, i.e., S or D. The `StateClass`, which is inherited from the `ObjectClass`, is used to record the four states associated to $FS(G)$, $FD(G)$, $PS(G)$ and $PD(G)$ predicates.

Finally, the `ResourceClass`, can help in the backward "search" when we desire to find the set of `Intentions` at the minimum cost that, if achieved, can guarantee the achievement of the desired top `Intentions`. Indeed, the `ResourceClass` can represent the monetary resource required for the achievement (through a `Process`) of an `Intention` which is used in the minimum cost analysis[2].

Figure 4.1 shows an example of `Intentions` reasoning with respect to the example described in Figure 2.3.

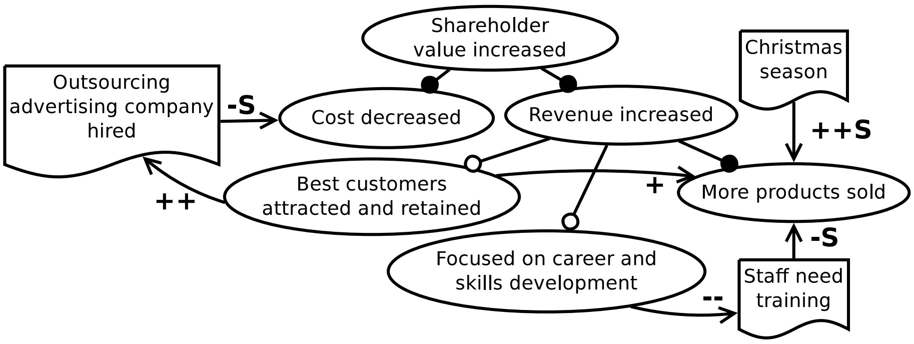

Figure 4.1: An example of `Intentions` reasoning with BIM.

Notice how the formal model is used for both `Situations` and `Intentions`. In the figure, the semantic of the influence relationships is the following:

- the satisfiability of "Outsourcing advertising company hired" `Situation` is propagated negatively (-S) to the "Cost decreased" `Intention`; this means that if the former holds the latter is partial denied; nothing is said about the denial of the "Outsourcing advertising company hired";

[2]Alternatively, a redundant attribute called *cost* can be added in the definition of the `IntentionClass` metaclass.

- the satisfiability (resp. deniability) of “Best customers attracted and retained” `Intention` is propagated positively (++) to the “Outsourcing advertising company hired” `Situation`; this means that if the former is satisfied (resp. denied) the latter holds (resp. does not hold);

- the satisfiability (resp. deniability) of “Best customers attracted and retained” `Intention` is propagated positively (+) to the “More products sold” `Intention`; this means that if the former is satisfied (resp. denied) the latter is partial satisfied (resp. denied);

- the satisfiability of “Staff need training” `Situation` is propagated negatively (-S) to the “More products sold” `Intention`; this means that if the former holds the latter is partial denied; nothing is said about the denial of the “Staff need training” `Situation`;

- the satisfiability of “Christmas season” `Situation` is propagated positively (++S) to the “More products sold” `Intention`; this means that if the former holds the latter is (at least partial) satisfied; nothing is said about the denial of the “Christmas season” `Situation`;

In order to show an example of forward reasoning on the model defined in Figure 4.1 [3], we input such a model in the same tool used in [14]. The result is as shown in Figure 4.2.

Table 4.1 shows the results obtained by applying forward reasoning. The first three rows correspond to `Situations`, followed by three rows for the top `Intentions` and three rows for the bottom `Intentions` with respect to the `Intentions` hierarchy. In the table, three experiments are described through initial values (*Init*) and final values (*Fin*) for satisfiability (S) and deniability (D) of `Situations/Intentions`. In particular, these values can be: full (F); partial (P); an *empty* cell when the corresponding element is not involved in the reasoning; or, a question mark symbol (**?**) when a result cannot be calculated.

A brief description of the experiments is the following:

- **Exp 1** : The “Christmas season” `Situation` is satisfied (see the F value for the *[S]-Init* column) so is the “Best customers attracted and retained” `Intention`. As result, we have that the “Shareholder value increased” is partial denied due to the partial denying of the “Cost decreased” `Intention`.

- **Exp 2** : The “Christmas season”, the “Staff need training”, and “Focused on career and skills development” initial values are set to full satisfied (F). As result, we have a full “Revenue increased” satisfaction (see below for the semantic of the conflicts) but no information for the “Shareholder value increased” `Intention` (represented by the question mark). This result is due to the fact that the ground

[3]Notice that: i) we need to add an extra node (namely, “-”) to simulate the feature model approach for the decomposition; ii) we need to select at least one of the OR sub-`Intentions` to properly use the tool while preserving the semantic of our model.

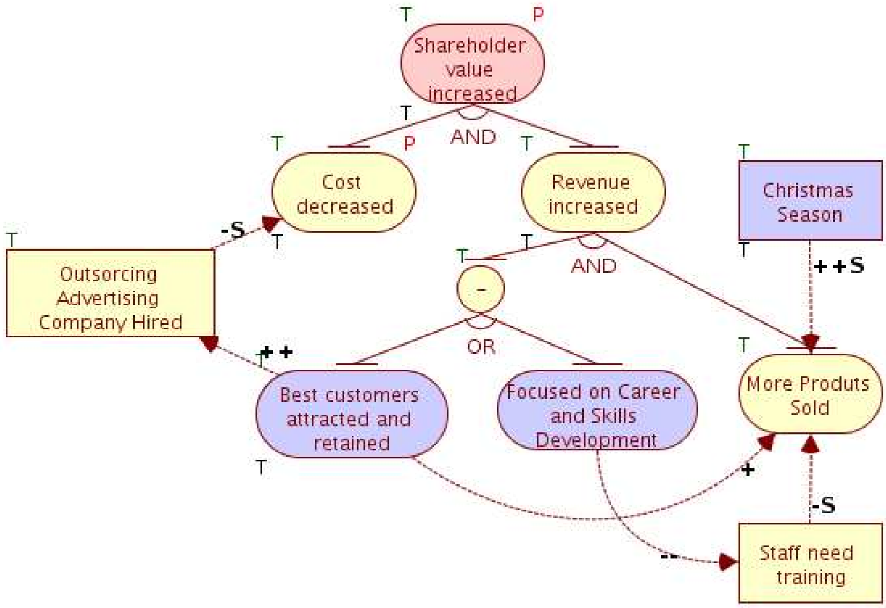

Figure 4.2: An example of `Intention` reasoning using the tool described in [14].

axioms, in this case $(G_1, ..., G_n) \overset{and}{\mapsto} G$, are not able to work with uncertainties (see Subsection 4.4 for how to address such issues). In fact, we have no information (see the question mark symbol **?**) for the "Cost decreased" `Intention`.

- **Exp 3** : The "Christmas season", the "Staff need training", "Best customers attracted and retained" and "Focused on career and skills development" initial values are set to full satisfied (F). As result, we have that the "Shareholder value increased" is partial denied due to the partial denying of the "Cost decreased" `Intention` and a conflict (i.e., full satisfied and partial denied) on the "Revenue increased" `Intention`.

Therefore, in the three experiments we use different strategies to satisfy the top "Shareholder value increased" `Intention` which lead to different results.
To conclude this section a final observation regarding the influence from `intentions` towards `Situations` must be made. In fact, an `Intention` can lead to (++) or avoid/mitigate (- -) a `Situation`.
A clear example is shown in the analysis performed in Table 4.1 where the "Outsourcing advertising company hired" can hold as the result of the satisfaction of the "Best customers attracted and retained" `Intention`; vice-versa, we have also that "Staff need training" `Situation` is avoided or mitigated by the satisfaction of "Focused on career and skills development".
The latter is the semantic associated to the conflicting values (S=F and D=F) for

Situation / Intention	**Exp 1**				**Exp 2**				**Exp 3**			
	Init		Fin		Init		Fin		Init		Fin	
	S	D	S	D	S	D	S	D	S	D	S	D
Outsourcing advertising company hired			F								F	
Christmas season	F		F		F		F		F		F	
Staff need training					F		F	F	F		F	F
Shareholder value increased				P				?				P
Cost decreased				P				?				P
Revenue increased			F				F	P			F	P
Best customers attracted and retained	F		F						F		F	
Focused on career and skills development					F		F		F		F	
More products sold			F				F	P			F	

Table 4.1: A formal forward reasoning example.

the "Staff need training" `Situation` which is propagated towards the"More products sold" `Intention`. Notice also that, the formal model is not able to deal with with uncertainty when some `Intentions` have not an initial value since the ground axioms require a complete information for the reasoning algorithm.

Finally, a similar analysis for the backward reasoning can be performed using similar experiments as shown in [14] both considering or not a cost criteria.

As a summary, we can said that the `Intention` reasoning model enables to:

- perform forward reasoning, in order to evaluate different strategies for the satisfaction of top `Intentions` elements;
- perform backward reasoning (considering also cost constrains), in order to evaluate the optimal input values leading to some desired final value;
- perform analysis on `Intention` inconsistencies and conflicts in the `Intention` hierarchy.

4.2 The SWOT Analysis with the BIM

The SWOT analysis [9] is a strategic planning method which is used to evaluate the **S**trengths, the **W**eaknesses, the **O**pportunities, and **T**hreats which are involved in a business environment. The purpose of the analysis is to specify the goals of the organization, business venture or project and identifying those internal and external factors that are favorable and unfavorable to achieve these goals.

Since a scan of the internal and external environment covers an fundamental role in the strategic planning process, the SWOT analysis can be considered as the first stage of such a process in which an organization is helped to focus on key issues.

Therefore, a SWOT analysis starts with the definition of a desired state of the world in terms of a set of strategic goals. Than, the identification of SWOTs with respect to the these strategic goals is performed. The result is an essential information which helps the decision makers in understanding the attainability of the selected strategic goals given such SWOTs. If the goals are not attainable different objectives must be selected and the process repeated.

In detail, the description of SWOT factors is:

1. *Strengths* are resources and capabilities of an organization which can be used as a basis for developing a competitive advantage since they are are helpful to achieve the strategic goals;

2. *Weaknesses* are absence of (certain) strengths as resources and capabilities which may be viewed as a weakness since they are are harmful to achieve strategic goals;

3. *Opportunities* are external conditions which can be helpful to achieve the strategic goals since represent favorable circumstances for profit and growth;

4. *Threats* are external conditions, usually due to changes in the external environment, which can be harmful to the strategic goals.

The results of a SWOT analysis are often presented in the form of a matrix as illustrated in Figure 4.3.

Notice how, some factors may be viewed as strengths/opportunities or weaknesses/threats depending upon their impact on the organization's goals, e.g, the opportunity or threat "changing of customer tastes".

Another way to use SWOT is for the *matching* and *converting* activities. The matching is used to find competitive advantages by "matching" the strengths to opportunities, while converting is the act of guide strategies in order to convert weaknesses or threats into strengths or opportunities. Usually, if the threats or weaknesses cannot be converted an organization should try to minimize or avoid them.

In particular, an organization can use a SWOT analysis to define:

- *S-O strategies*, which pursue opportunities that fit good to the organization's strengths.

- *W-O strategies*, which overcome or avoid weaknesses to pursue opportunities.

- *S-T strategies*, which identify ways to use organization's strengths to reduce its vulnerability to external threats.

- *W-T strategies*, which establish a defensive plan to avoid that organization's weaknesses accentuate external threats.

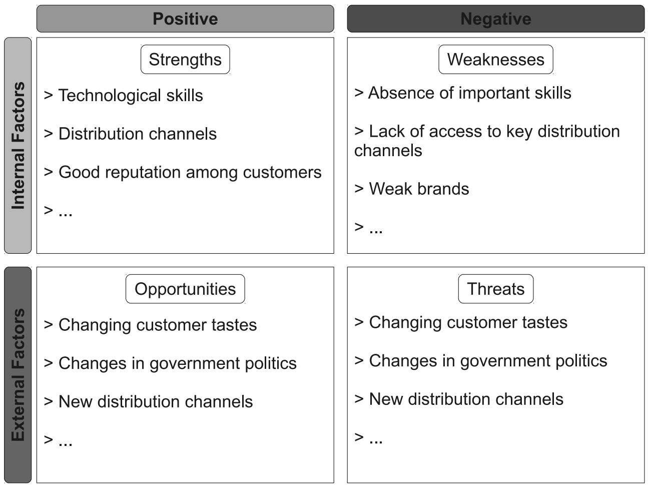

Figure 4.3: An example of SWOT matrix.

The BIM provides a formal way to perform the SWOT analysis since: i) allows to link the SWOT factors directly to the strategic goals they impact upon; ii) allows a formal reasoning on the set of strategic goals, SWOT factors and influences relationships among them.

The latter can be very useful for the definition of S-O, W-O, S-T, W-T strategies since make feasible the exploration of the different alternatives relying on the forward reasoning and backward reasoning approaches presented in Subsection 4.1.

As shown in Section 2.1, we use `Situation` to represent those internal and external factors which can contribute positively or negatively to the achievement of `Intention`, i.e., strategic goals.

Notice that, as described in Subsection 4.1, when a schema is defined, some `Intention` can be introduced to mitigate or avoid some `Situations` and some (harmful) `Situations` can arise due to the presence, in the schema, of specific `Intentions`.

Moreover, it must be said that, in the BIM, we characterized as strength, weakness, opportunity or threat the "influence" that exist from a `Situation` to an `Intention`. This allows to represent those cases in which the same `Situation` can represent, for example, a strength with respect to an `Intention` while representing a weakness with respect to another.

Table 4.2 shows how to map the SWOT influences to the formal model presented in Subsection 4.1, while Figure 4.4 illustrates an example.

In the figure, the "More products sold" is defined to exploit the "Christmas season" external opportunity. This opportunity is matched by the"Efficient and effective distribution channels" internal strength that allows to deal with the high demand during

SWOT Influence	Formal Model Influence
Strength	+S, ++S
Weakness	-S, –S
Opportunity	+S, ++S
Threat	-S, - -S

Table 4.2: SWOT and formal model mapping.

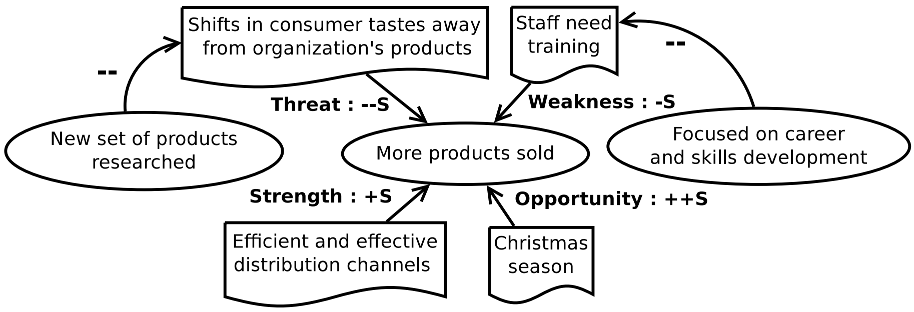

Figure 4.4: A SWOT analysis example with BIM

Christmas.

Moreover, in order to avoid and mitigate weaknesses and threats, two strategic goals are also defined in the schema, namely “Focused on career and skills development” and “New set of products researched” strategic goals.

The former attempts: i) to mitigate the lack of Staff’s skills in order to be prepared for the Christmas; and ii) to reduce the organization vulnerability to the external threat helping the Staff to turn the customer’s taste toward the Organization’s products.

The latter, the “New set of products researched”, is defined and pursued as a defensive plan to match the new customer’s taste.

4.3 Define Strategic Map, Balanced Scorecard and Key Performance Indicators with BIM

Important instruments for strategic planning are Strategic Maps (SMs) [22] and Balanced SCorecards (BSCs)[21]. The former are visual representation of the strategy of an organization which shows organization plans used to achieve missions and visions. In particular, a Strategic map illustrates the cause-and-effect relationships between different strategic goals and the associated measures, the key performance indicators (KPIs).

These measures are included in the latter, the BSC, which represents a “balanced” range of metrics against which to measure the Organization’s performance. The meaning of “balance” is provided by the fact that the broader view of leading indicators of performance includes also non-financial metrics, such as "learning and growth of employees", "customer satisfaction", etc.

The combination of SMs and BSCs follows the principle of "you cannot measure what you cannot describe". In fact, SMs aim to describe the direction of an organization while BSCs aim to define a comprehensive set of performance measures that provides the framework for a strategic measurement and a management system.

Both the SMs and BSCs describe and measure organizational performance across four balanced perspectives: financial, customers, internal business processes, and learning and growth (for their descriptions and further details see [21] and [22]). In general, these perspectives, allow to see the organization and the business environment from different viewpoints and not only from the financial aspects.

As described in [21], the four perspectives have been found to be robust across a wide variety of companies and industries but should be considered a template. Indeed, no mathematical theorem exists to proof that four perspectives are both necessary and sufficient.

Within each of the four perspectives, the organization must define the following elements:

1. *Strategic goals* [4] – strategies which must be achieved in that perspective;
2. *Measures* – the progresses toward that particular strategic goals;
3. *Targets* – the target value sought for each measure;
4. *Initiatives* – what should be done to facilitate the achievement of the target;
5. *Cause-effect relationships* – influences among strategic goals (or measures).

Figure 4.5 illustrates an example of such elements in which only *Targets* are missing. A typical target can be, for example, a value of $10,000 for the *Revenue* measure for satisfying the "Revenue increased" strategic goal.

A common approach to evaluate the performance of an organization and how successful it is in achieving short and long-term goals, is the use of KPIs [33]. KPIs are quantifiable measurements which reflect the performance of an organization towards its goals. Therefore, BSCs can express measures and targets through a set of KPIs.

BIM integrates in a single conceptual framework the primitive concepts that characterize SMs, BSCs and KPIs, as well as requirements models in Software Engineering. Through projection mappings on a global BIM model, it is possible to obtain partial models that can be analyzed through SM, BSC, KPI and formal goal reasoning techniques [14] as described in previous sections.

Using the fragment in Figure 2.2 and the *IndicatorClass* described in Figure 2.8(a), we are able to represent both SMs and BSCs (i.e., a set of KPIs). In particular, `Intentions` and `Indicators` represent strategic goals, their measures and associated targets. `Processes` with the *type* attribute set to *Initiative* (see Table 32) describe

[4] We use the strategic goal term instead of the objective as used in the BSC.

initiatives used to reach targets. Instances of the `Influence` metaclass address cause-effect relationships (both in quantitative and qualitative ways). Finally, the *perspective* attribute helps to characterize elements along the four different perspectives.

An example of such mapping is shown in Figure 4.5, corresponding to the model of Figure 2.1.

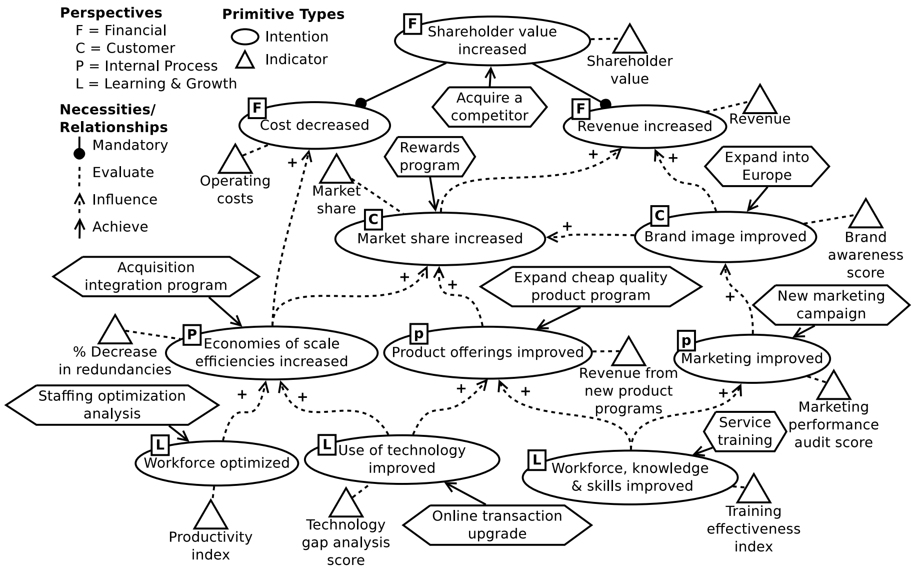

Figure 4.5: The BestTech Strategic Map and Balanced Scorecard defined with BIM.

Notice how, the BIM model can represent a possible underneath formal schema for the SM and BSC described in Figure 2.1. Therefore, SM and BSC should be used for illustration purpose, since familiar to executives, middle managers, etc., while BIM should be used to formalize such abstracted human-language to a machine-readable language on which queries, in depth analysis, etc., can be performed.

In conclusion we can affirm that, as SMs and BSCs do, the BIM is: i) a way of providing a macro view of an organization's strategy using the `Intention` primitive type to describe strategic and tactical goals; and ii), a way of constructing metrics to evaluate performance against these strategies using the `Indicator` primitive type.

However, at the contrary of SMs and BSCs, the BIM allows more in depth analysis on the schema obtained after the designing activity.

4.4 Probabilistic Graphical Model for Intention Reasoning

In Section 4.1, we described a solution based on formal logic model to provide a reasoning mechanism on `Intentions`. However, we also highlighted that such kind of model is able to provide only partial results in condition of *uncertainty*. For example, we would recall the experiment two in Table 4.1 in which a question mark symbol (`?`) was introduced for the "Cost decreased" and "Shareholder value increased" `Intentions`.

The issue of treat with uncertainty is an inescapable aspect of most real-world applications; indeed, it is quite common to have not a complete information during an analysis activity. Future works for BIM, include the investigation of probabilistic (graphical) models [23], which make the uncertainty explicit and provide models that are more faithful to reality.

Probabilistic graphical models are approaches model-based which allow interpretable models to be constructed and then manipulated by reasoning algorithms. These models can be defined by an analyst or can be learned automatically from data in order to facilitate their construction when a manual design is difficult or even impossible. Different Probabilistic graphical models have been defined in the scientific community, such as Bayesian networks, undirected Markov networks, Influence Diagrams, etc. (see [23] a comprehensive discussion).

One of our goals within BIM, is to adapt such models in order to manage uncertainty to perform causal reasoning and decision making under such circumstances. In particular, we are concentrating on Bayesian networks and in providing a first step toward the use of Influence Diagrams.

5 A Case Study

In this section, we sketch a case study for BestTech Inc. for which we constructed a complete BIM schema. Part of the schema is shown in Figure 5.1. This schema provides a comprehensive description of the business and its environment, balanced along the four perspectives discussed earlier. For example, from the Financial Perspective, the top-level intention is *Shareholder value increased*; one of its sub-intention *Cost decreased* is further refined into *Management cost decreased* and *Supply chain cost decreased*. In general, for each perspective, `Intentions` have their associated `Indicators`, e.g., *Market share* for *Market share increased* (from the Customer perspective), and they are related to high-level processes (strategies), e.g., *Rewards program*.

The BestTech schema can be queried by the business analyst to answer questions such as "Which are the influencers and sub-intentions for *Revenue Increased*", or "Which are the `Intentions` whose performance is poor (red zone) and whose deadline is at the end of the month". Since data often resides in and scatters across databases, such queries are translated through schema mappings into database queries, and the answers are then translated back into business-level concepts. Schema mapping between a BIM schema and database schema is a ongoing research in our group. Moreover, this schema can be projected along different views. An example is illustrated by the SM of Figure 4.5 which is a useful view when communicating an organization's strategies with the BestTech executives. Moreover, if the need is to perform analysis, we can project the schema towards a variety of analysis models, as discussed in Section 4. With such projections, we can respond to queries such as "Show me all the `Intentions` which are in conflict with at least one other `Intention`" or "Show me the impact of denial of the *Marketing improved* `Intention`".

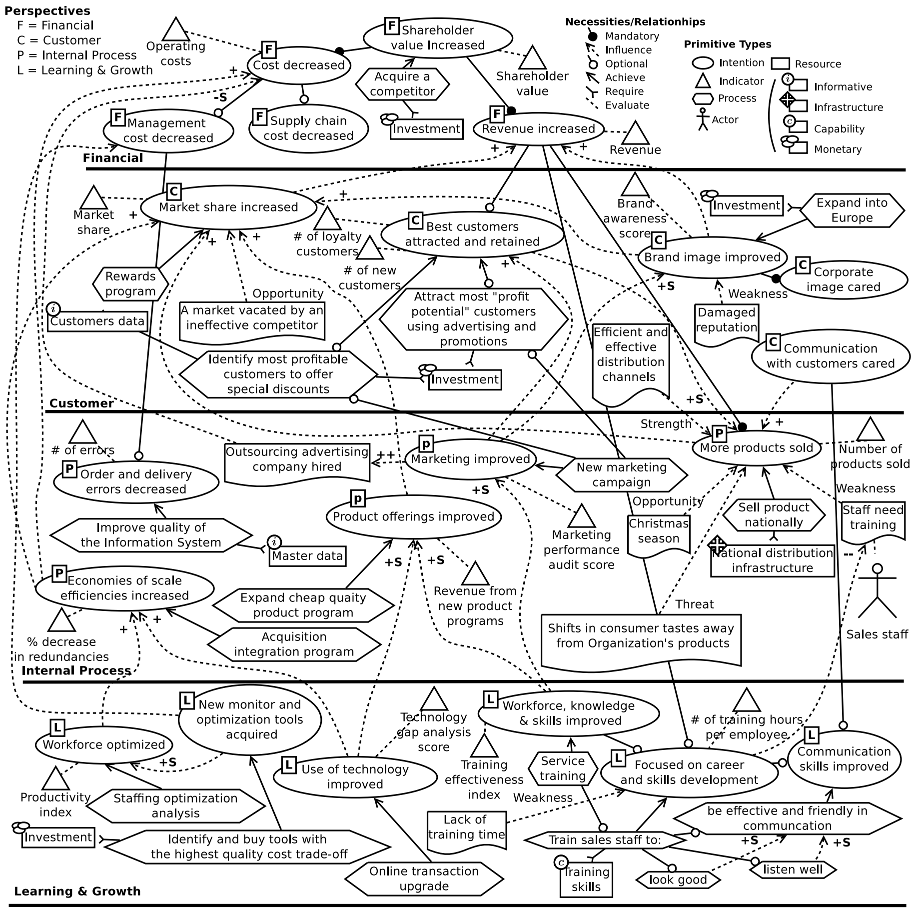

Figure 5.1: Part of the BestTech BIM schema.

6 Related Work

The use of business-level concepts—such as business objects, rules and processes—has been researched widely for at least 15 years and is already practiced to some extent in both Data Engineering and Software Engineering [38, 25, 19]. These efforts have more recently resulted in standards, e.g., OMG's Business Process Modeling Notation (BPMN) [32]. Such proposals focus on modeling objects and processes, with little attention paid to objectives.

Enterprise modeling languages (enterprise *ontologies*, to some) have also been researched for a long time, with the express intention of aligning business and IT concerns. Examples of this line of research include TOVE [10], REA [26] and the Zachman Framework for Enterprise Architecture [43], as well as TOGAF [39]. Of those, BMM [3] is closest in spirit to BIM. Our proposal places the BIM concepts we adopted from BMM on an ontological foundation adopted from DOLCE [11] and also integrates those with state-of-the-art abstraction mechanisms.
Notably, our concept of `Situation` is akin to the notions of description and situation proposed in [12], but the authors there envisioned semantic web applications, rather than business ones.

The Zachman Framework for Enterprise Architecture is one of the oldest proposals for enterprise modeling. The framework consists of a table of 5 rows and 6 columns. The rows define an IT system and its context from different perspectives ranging from scope (top row), to business model, information system model, technology model and detailed description (lowest row). Each row of the table uses a different language. Columns define common questions that need to be answered about each perspective: what, how, where, who, when and why. The public part of the Zachman framework consists of this table, with no stand taken on what notation or modeling method to use. Issues of notation and method to use are addressed in the proprietary part. This modeling framework has had considerable influence on enterprise modeling practice, including recent work on Service-Oriented Architectures (SOAs). BIM fits within the Zachman framework, focusing on the *why* column, but offers a different set of primitive concepts for capturing *why* concerns than other proposals in the literature.

As indicated in the introduction, the other modeling proposals that relate to our work are i* [42], URN/GRL [18] and KAOS [7, 40], all from the general area of Goal-Oriented Requirements Engineering. From these we have adopted intentional and social concepts. These models lack primitive constructs for influence relationships, indicators, and various types of situations integrated in the BIM modeling framework. Recent proposals extending URN do include indicators [34], but BIM's indicators are more general and they can be used to measure any model object, including other indicators.

From a business perspective, BIM models can capture what is commonly found in

Strategic Maps and Balanced Scorecards. They can also be mapped to other languages that enable goal analysis and SWOT analysis, and we expect other mappings to probabilistic frameworks such as Bayesian networks and Analytics [8] to enable reasoning under uncertainties.

7 Conclusions

One important problem of Business Intelligence technologies is that information required and generated by such technologies is rarely explicitly linked to business concepts, decisions and outcomes, and is therefore hard to interpret and use. In this report we have proposed the Business Intelligence Model (BIM), as first step towards bridging the gap between the worlds of business and data analytics. The proposed model extends the notion of conceptual schema to accommodate business concepts such as strategic objectives, business processes, influences, indicators, risks and trends. We have showed, through examples taken from a case study how a BIM schema can support governance activities, including monitoring, auditing and analysis at the strategic level. As mentioned before, for BIM to be useful, we also need technologies for translating queries specified over a BIM schema into queries over database schemas, also for translating answers back into business terms. Such work is being carried out within the context of the strategic network for Business Intelligence, funded by the Natural Sciences and Engineering Research Council (NSERC) of Canada [1].

As for future work, along one direction, we are further evaluating and refining BIM with a large scale, real-world case study. Along another, we are extending to cover the tactical level of business organizations, and along a third, we plan to extend our model to incorporate uncertainty in strategic modeling and analysis through the use of Bayesian networks. This will enable BIM to support statistical decision making [23] and will complement the logic-based analysis techniques currently within BIM's scope.

Acknowledgments

This work was supported by the Business Intelligence Network (BIN) and NSERC. We are grateful to G. Mussbacher, G. Richards, E. Yu and many others for useful discussions.

[1] http://bin.cs.toronto.edu/home/index.php and
http://www.nserc-crsng.gc.ca/Partners-Partenaires/Networks-Reseaux/BIN-RVE_eng.asp

8 Appendix

8.1 BIM Taxonomy

Refer to Table 2.1 and Table 3.2 for the taxonomy's description.

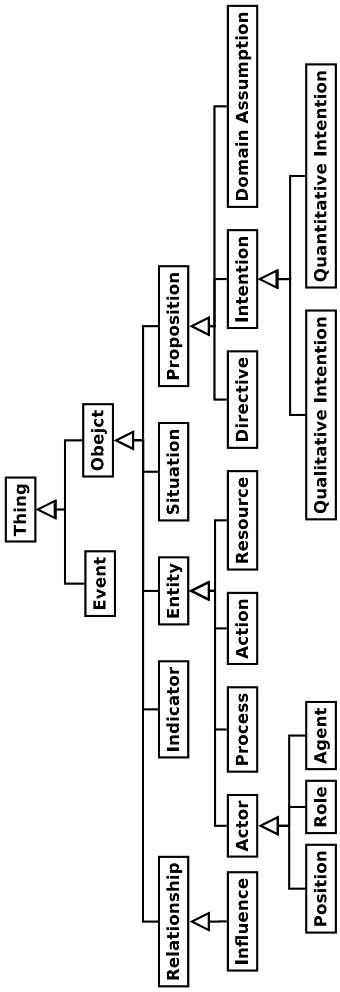

Figure 8.1: The BIM 's taxonomy.

Bibliography

[1] istarquickguide. http://istar.rwth-aachen.de/tiki-index.php?page=iStarQuickGuide, 2009.

[2] P. Bresciani, P. Giorgini, F. Giunchiglia, J. Mylopoulos, and A. Perini. Tropos: An agent-oriented software development methodology. *Autonomous Agents and Multi-Agent Systems*, 8(3):203–236, 2004.

[3] Business Rules Group. The business motivation model: Business governance in a volatile world. http://www.businessrulesgroup.org/bmm.shtml, 2007. Release 1.3.

[4] BusinessDictionary.com. Businessdictionary. http://www.businessdictionary.com, 2009.

[5] P. P.-S. Chen. The entity-relationship model—toward a unified view of data. *ACM Trans. Database Syst.*, 1(1):9–36, 1976.

[6] A. Colin and K. Thomas. The essence of multilevel metamodeling. In *Proc. of the 4th Int. Conf. on The UML, Model. Lang., Concepts, and Tools*, pages 19–33, London, UK, 2001. Springer-Verlag.

[7] A. Dardenne, A. van Lamsweerde, and S. Fickas. Goal-directed requirements acquisition. *Sci. Comput. Program.*, 20(1-2):3–50, 1993.

[8] T. H. Davenport and J. G. Harris. *Competing on Analytics: The New Science of Winning*. Harvard Business School Press, March 6, 2007.

[9] T. R. Dealtry. *Dynamic Swot Analysis*. Dynamic Swot Associates, 1994.

[10] M. S. Fox. The tove project towards a common-sense model of the enterprise. In *IEA/AIE '92: Proceedings of the 5th international conference on Industrial and engineering applications of artificial intelligence and expert systems*, pages 25–34, London, UK, 1992. Springer-Verlag.

[11] A. Gangemi, N. Guarino, C. Masolo, A. Oltramari, and L. Schneider. Sweetening ontologies with dolce. In *Proc. of the 13th Int. Conf. on Knowl. Engineer. and Knowl. Manage. Ontol. and the Semant. Web*, pages 166–181, London, UK, 2002. Springer-Verlag.

[12] A. Gangemi and P. Mika. Understanding the semantic web through descriptions and situations. In *Proceedings of ODBASE03 Conference*, pages 689–706. Springer, 2003.

[13] A. Gill. *Introduction to the Theory of Finite-State Machines*. McGraw Hill, 1962.

[14] P. Giorgini, J. Mylopoulos, and R. Sebastiani. Goal-oriented requirements analysis and reasoning in the tropos methodology. *Eng. App. Artif. Intel.*, 18:159–171, 2005.

[15] C. Heitmeyer. Software cost reduction. In *Encyclopedia of Software Engineering, Two Volumes*. Wiley, January 2002.

[16] R. Howard and J. Matheson. Influence diagrams. *Readings on the Principles and Applications of Decision Analysis, Vol. II*, 1984.

[17] R. Hull and R. King. Semantic database modeling: survey, applications, and research issues. *ACM Comput. Surv.*, 19(3):201–260, 1987.

[18] International Telecommunication Union: Recommendation Z.151 (11/08). User Requirements Notation (URN) – Language definition. http://www.itu.int/rec/T-REC-Z.151/en.

[19] S. Jablonski. On the complementarity of workflow management and business process modeling. *SIGOIS Bull.*, 16(1):33–38, 1995.

[20] K. Kang, S. Cohen, J. Hess, W. Novak, and A. Peterson. Feature-oriented domain analysis (foda) feasibility study. Technical report, CMU/SEI-90-TR-021, SEI, Carnegie Mellon University, November 1990.

[21] R. S. Kaplan and D. P. Norton. *Balanced Scorecard: Translating Strategy into Action*. Harvard Business School Press, 1996.

[22] R. S. Kaplan and D. P. Norton. *Strategy maps: Converting intangible assets into tangible outcomes*. Harvard Business School Press, 2004.

[23] D. Koller and N. Friedman. *Probabilistic Graphical Models: Principles and Techniques*. The MIT Press, August 2009.

[24] N.-S. Koutsoukis and G. Mitra. *Decision Modelling and Information Systems: The Information Value Chain*, volume 26. Springer: Operations Research/Computer Science Interfaces Series, 2003.

[25] P. Loucopoulos and E. Katsouli. Modelling business rules in an office environment. *SIGOIS Bull.*, 13(2):28–37, 1992.

[26] W. E. McCarthy. The rea accounting model: A generalized framework for accounting systems in a shared data environment. *The Accounting Review*, 1982.

[27] Metaphysics Research Lab. Stanford Encyclopedia of Philosophy. http://plato.stanford.edu/, 2010.

[28] H. Morris. Analytic applications and decision-centric bi. Information Management Magazine, June 1 2004.

[29] J. Mylopoulos, A. Borgida, M. Jarke, and M. Koubarakis. Telos: representing knowledge about information systems. *ACM Trans. Inf. Syst.*, 8(4):325–362, 1990.

[30] Object Management Group. Meta-object facility (MOF) 2.0 specification. http://www.omg.org/spec/MOF/2.0/, 2006.

[31] Object Management Group. Object constraint language. http://www.omg.org/spec/OCL/2.0/, May 2006.

[32] Object Management Group. Business process modeling notation (bpmn). http://www.omg.org/spec/BPMN/1.2/, January 2009. version 1.2.

[33] D. Parmenter. *Key Performance Indicators*. John Wiley & Sons, 2007.

[34] A. Pourshahid, D. Amyot, L. Peyton, S. Ghanavati, P. Chen, M. Weiss, and A. J. Forster. Business process management with the user requirements notation. *Electronic Commerce Research*, 9(4):269–316, 2009.

[35] R. L. Sallam and K. Schlegel. Overcoming the gap between Business Intelligence and Decision Support. http://www.gartner.com/it/page.jsp?id=927913, 2009.

[36] P. K. Saxena. *Principles of Management: A Modern Approach*. Global India Publications Pvt Ltd., 2009.

[37] P.-Y. Schobbens, P. Heymans, J.-C. Trigaux, and Y. Bontemps. Generic semantics of feature diagrams. *Comput. Netw.*, 51(2):456–479, 2007.

[38] J. Sutherland. Business objects in corporate information systems. *ACM Comput. Surv.*, 27(2):274–276, 1995.

[39] The Open Group. TOGAF 9 - The Open Group Architecture Framework Version 9, 2009.

[40] A. van Lamsweerde. *Requirements engineering: From System Goals to UML Models to Software Specifications*. John Wiley & Sons, 2009.

[41] E. Yu. *Modelling strategic relationships for process reengineering*. PhD thesis, University of Toronto, Dept. of Computer Science, University of Toronto., 1995.

[42] E. Yu. Towards modelling and reasoning support for early-phase requirements engineering. In *Proc. 3rd IEEE Int. Symp. on Requirements Engineering, IEEE CS*, Washington, USA, 1997.

[43] J. A. Zachman. A framework for information system architecture. *IBM Systems Journal*, 26(3):277–293, 1987.