10

Generic Model Base Design for Decision Support Systems in Revenue Management: Applications to Hotel and Health Care Sectors

Miguel Gutiérrez and Alfonso Durán
Área de Ingeniería de Organización, Universidad Carlos III de Madrid
Spain

1. Introduction

The sustainable development of an increasingly service-based economy requires procedures for the efficient allocation, to the various existing user classes, of non-storable service infrastructures with essentially fixed costs and whose value potential is dilapidated if not utilized. That decision is often taken in phases; for instance, assigning a hospital wing to a specific use (e.g. surgery or radiology) is a long-term decision, given the high refurbishing costs. Similarly, strategic decisions have to be made regarding the distribution of hotel rooms in single, double, suites... In a second decision stage, operational decisions will be taken, such as the individual patients or customers to whom those infrastructures will be assigned.

The aforementioned problem of infrastructure allocation is frequently addressed by Revenue Management (RM) techniques—also known as Yield Management—. RM techniques were initially developed to deal with strategies regarding the offering and allocation of flight seats. According to Zaki (2000) "the main objective of RM is to sell the right seat to the right customer at the right time for the right price to maximize the profit." In a more general approach, RM is the process of reacting or anticipating to the consumer behaviour, so that the revenue is optimized. It implies the use of dynamic forecasting models, the allocation of perishable resources/services to the diverse price categories and channels, as well as the price to charge for every category-channel (Talluri & Van Ryzin, 2004).

To accomplish these tasks, RM uses optimization heuristics, whose relative effectiveness depends strongly on the characteristics of the demand, such as predictability, differences in sensitivity to the price between the different segments of clients and temporary pattern of evolution of the relative weight of each segment of clients when the date of execution approaches.

On the other hand, RM algorithms reflect (and therefore are contingent upon) the design of the business process whose optimization they must support (in this case, infrastructure assignment) in three related ways:

- The specific traits of the business process, such as whether overbooking is allowed or not and whether two different prices can be offered simultaneously through two different channels or not. It would also encompass such aspects as alternative distribution channels through which rooms can be reserved, quota allocation and reallocation to each channel, cancellation procedures and penalizations, pricing process (prices that can only increase as the execution/ arrival date approaches vs. prices that can go either up or down).

- The decision variable(s) to be optimized. In a hotel reservation process, the decision to be taken might be when (how long before the actual stay date) to switch from a lower price to a higher price, or alternatively consider this anticipation as a fixed parameter and try to optimize which price to switch to.
- The business objective(s) to achieve. In the hotel sector it will typically be profit; given its essentially fixed costs, this translates into managing/ maximizing revenue, hence the "Revenue Management" term. In other service sectors the optimization objective is not so clear, as it is the case of the Health Care sector. Nordgren (2009) discusses the complexity of the multi-faceted concept of value in this sector and thus the number of objectives that should be simultaneously taken into account when trying to find a match between the service providers and the potential users.

Although potential benefits to be gained from the use of advanced RM techniques to guide the allocation of infrastructures are substantial, in many cases merely applying the existing algorithms adds limited value. In its thorough literature review on RM research, Chiang *et al*. (2007) highlight the resulting complexity and difficulty of developing and implementing RM projects, to the point of suggesting that these projects are viewed as strategic, as opposed to tactical activities. There is a barrier to the adoption of RM algorithms, derived from its dependency on the concrete design of the business process. Each service organization uses a different business process design for infrastructure allocations (e.g., the pricing and reservation process in a hotel). These organizations do not want to restrict themselves to predetermined business process designs just to use a specific algorithm. Since the appropriate algorithm is contingent on that design, organizations must tailor their algorithms or redevelop them from scratch. Besides, given the currently prevailing non-systematic approach to algorithm development, this adaptation requirement, both initially and whenever the business process is redesigned, imposes a stiff hindrance, particularly to the SME, and also limits its adaptability to changing market conditions.

The work presented here intends to overcome that barrier, by taking advantage of the flexibility offered by a generic modeling approach to design the model base component of a Revenue Management-based Decision Support System (DSS) aiming at the efficient allocation of the abovementioned non-storable service infrastructures.

The Model Base can be identified as the key component in RM DSS according to the definition provided by Sprague. In his classical DSS framework, Sprague distinguishes three main components in a DSS (Sprague, 1980): database, model base and software system. The software system in turn is composed of three components: the DBMS (Database Management Software), which in conjunction with the database constitutes the data subsystem; the MBMS (Model Base Management Software), which in conjunction with the model base constitutes the model subsystem; and the dialog generation and management software which forms the system user interface. The model subsystem is within the scope of a DSS what is frequently referred to as a Model Management System (MMS), which according to Muhanna & Pick (1994) can be defined with wide agreement as "a software system that facilitates the development, storage, manipulation, control, and effective utilization of models in an organization."

We can thus characterize DSS in RM problems as *Model-Driven DSS* following the framework of Power (2002). This author presents different DSS classifications and proposes a framework of DSS that includes five basic types of DSS characterized by their dominant component, user groups, purpose and enabling technology: Data-Driven, Model-Driven, Knowledge-Driven, Document-Driven and Communications-Driven. With respect to Model-Driven DSS, Power (2002) outlines that "Model-Driven DSS emphasize access to and manipulation of a model." (Power 2002)

In the next section we present the proposed generic modeling approach consisting in a hierarchy of levels of abstraction, which provides the conceptual key to design a flexible model base component of a RM DSS. The remaining sections of the chapter first describe the model associated with each of the levels of the abovementioned hierarchy and show the potential of this approach with applications to specific cases taken from the quite different in nature Hotel and Health Care sectors. One of the reasons for including the Health Sector in this analysis is to show the applicability of the approach in intrinsically multiobjective/ multicriteria settings (Gupta & Denton 2008; Nordgren 2009).

2. Hierarchical modeling approach

As stated earlier, the key question to address in order to surmount the barrier to the successful utilization of DSS in RM problems is the intrinsic linkage between the algorithms and the specific design of the business process. Our approach to tackle this barrier stems from a hierarchical generic modeling approach of the business processes. A *business process model* defines the way a particular infrastructure element is assigned to one of its potential uses. In the example of a hotel, that would be the design of the room reservation process, considering aspects like alternative distribution channels through which rooms can be reserved, pricing process (prices that can only increase as the execution/arrival date approaches vs. prices that can go either up or down), etc.

In order to explain the proposed modeling hierarchy, we establish a parallelism with a simple example of a mathematical model as can be seen in Fig.1. There are three hierarchical modeling levels:

- The **business process metamodel** level corresponds to the highest abstraction layer. The generic model of the business processes related to infrastructure assignment issues is defined, providing the basic elements and their relationships, so that, through instantiation, the set of possible models is derived. In the example of the mathematical model, in this level we would find a metamodel describing the building blocks of a mathematical equation, such as variables and operators, as well as the definition of the possible ways to combine these elements to constitute the two members of a mathematical equation.
- The **business process model** level encompasses the direct instances of the metamodel defined in the former level. In the mathematical example, in this level we would find generic equations such as that represented in the Figure: $a - b = c^2$, a, b, and c being natural numbers.
- The **business process instance** level emerges as the instantiation of the former level, in the sense that each process instance will originate from a process model by means of assigning specific values to a subset of its generic elements. The instances will become feasibility or optimization problems, with a defined objective in the latter case. The

subset of valued-assigned elements will constitute the subset of parameters of the model, whereas the non-assigned elements will form the subset of variables of the thus defined feasibility/optimization problem. In the mathematical example, we will obtain different equations with the assignment of values to a subset of the model elements (a, b, c).

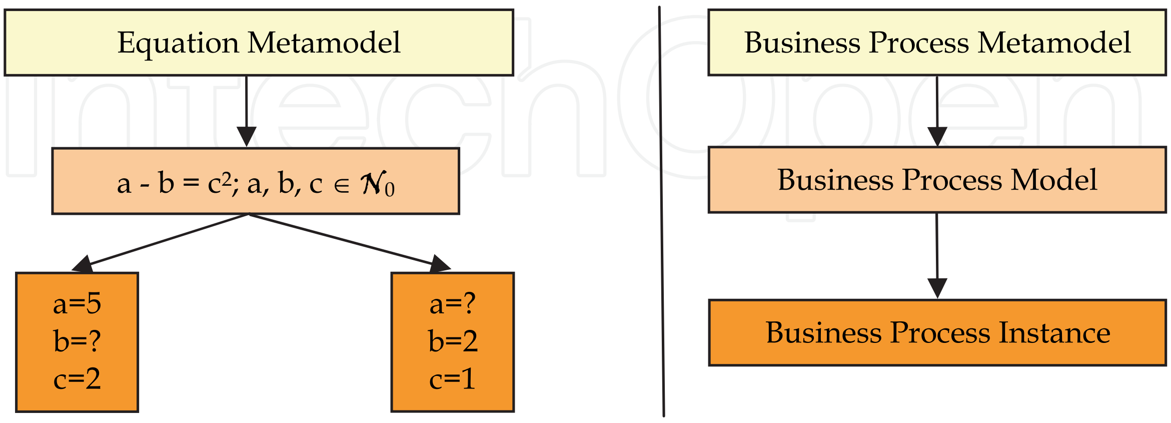


Fig. 1. Generic modeling hierarchy

Throughout the modeling process of the proposed hierarchy we make use of the standard UML, Unified Modeling Language (OMG, 2010). UML is an ongoing project developed by the OMG, Object Management Group, with the initial objective of standardizing and unifying the proliferation of object oriented modeling languages developed in the late 80s and early 90s (Fowler, 2004). According to OMG's description, UML supports the specification, visualization and documentation of software system models, including their structure and design, even though UML can also be used for business process modeling and modeling of other non-software systems too (OMG, 2011). The choice of UML derives from the confluence of these two traits: its applicability to business process modeling together with its software systems orientation (Eriksson & Penker, 2000) and the unifying standard nature with which the language was born, that has facilitated its growing usage and acceptance.

In the following sections we present the UML class diagrams that describe each of the levels of the business process hierarchy. Class Diagrams evolved from the Entity Relationship Diagrams (ERD). They describe the types of elements found in a system (entities) and the structural relationships among them. Entities are represented as square boxes and the relationships as lines that join two participating entities. There are two basic types of relationships: Association, in which participating entities are at the same level, and Generalization, graphically depicted as a line terminated in a triangle, in which one entity is a subtype or specialization of the other ("is-a-kind-of" relationship). Relationships may include a cardinality, or degree of multiplicity with which an entity participates in a relationship; it is denoted by the corresponding number, or by an asterisk to indicate any multiplicity. The default value for cardinality is one. Additionally, the role that an entity plays in a relationship might be annotated next to the entity.

3. Business process metamodel

The UML model that corresponds to the first abstraction layer is shown in Fig. 2. A *business process model* defines how a particular type of infrastructure is assigned to one of its potential uses. The proposed metamodel, as the model of the process models, aims to support the whole spectrum of problem definitions intended to be addressed.

As shown in the class diagram, Infrastructure *Access Types* are defined first, as follows: each *Customer Segment Type* accesses through a *Channel Type* a certain *Infrastructure Type*. In the most general case, all *Customer Segment Types* would be able to access all *Infrastructure Types* through all *Channel Types*; the existence of restrictions in the access to some *Customer Segment Types*, or access to certain infrastructures through some *Channel Types*, will lead to a whole set of problems to be addressed.

This problem base is reflected in the instances of the entity *Infrastructure Access Type*. In turn, each of these instances might be assigned different *Value Types*. In commercial infrastructure allocation problems, these *Value Types* will be the tariff types, e.g. reduced tariff, normal tariff, premium tariff; offer price, list price; high season, low season... The assignment of a *Value Type* to an *Infrastructure Access Type* defines a generic *Allocation Type*. Each *Allocation Type* will apply during a time interval, which will be defined by a *Generic Start Time* and a *Generic End Time*. These times are handled in the generic model layer in abstract form, without taking a specific value or time. Even if they do not take a specific value, in complex models logical relationships (*Constraints*) will have to be defined among them.

According to this approach, a *Business Process Model* is defined as a set of *Allocation Types*. The relationship that starts with the solid rhomboid is a Composition Relationship. Therefore, in this layer, the proposed metamodel encompasses in a generic manner definitions of infrastructure allocation optimization problems.

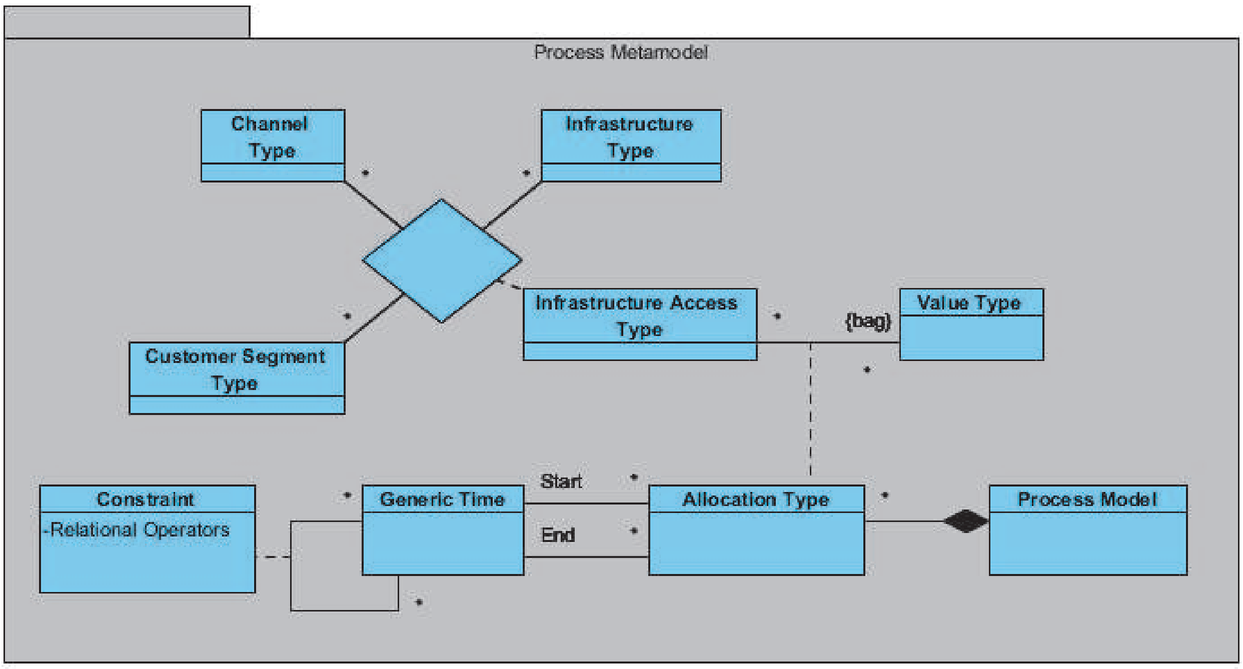


Fig. 2. UML class diagram of the Process Metamodel, the model of the process models

Model families can be derived from the instance patterns. Specifically, families can be defined if a dependency is found between two of the following entities: *Channel Type*, *Customer Segment Type* and *Infrastructure Type*; and according to which of these entities affects the definition of the prices or values, *Value Type*. As an example, there might be cases in which customers can access through various channels a given infrastructure (e.g.

reservation of a plane seat through the web or through a travel agency), and the combination of channel and infrastructure will define the types of value/ tariff that will be offered. In this case, the tariff would not depend on the customer type. Complementing the example, alternatively, the proposed model also encompasses another family of optimization models, such as when customers are segmented according to their loyalty to the company, and thus tariffs are in fact dependent on the customer segment.

4. Business process model

As stated in the hierarchy definition, the Business Process Model abstraction layer corresponds to the business process models generated through direct instantiation from the metamodel described in the former section.

We illustrate our proposal through a highly simplified model of a hotel room reservation process. In this simplified process we consider only one *Channel Type* (*CH1*) through which the customers place their reservations for a unique type of room (*R1*), which will be the only *Infrastructure Type* in this case. Customers are segmented in two types depending on whether they are only willing to make the reservation if they are offered a discounted price (*B*) or they are willing to pay the normal price (*A*; these customers would naturally also accept the discounted price). Therefore, there are two pre-specified *Value Types* (*Discounted* and *Full*) for each room. The business process designed by the hotel contemplates offering rooms initially at a specially discounted price and at some point in time, switching the price from *Discounted* to *Full*. The price will then remain *Full* for the rest of the time horizon. Table 1 shows the instances that would in this case form the business process according to the proposed model. *Generic Times* T1 and T3 define the start and the end of the process to be optimized. *Generic Time* T2 defines when the hotel switches from the *Discounted* price to the *Full* price.

Customer Segment Type	Channel Type	Infrastructure Type	Value Type	Generic Start	Generic End
A	CH1	R1	Discounted	T1	T2
A	CH1	R1	Full	T2	T3
B	CH1	R1	Discounted	T1	T2

Table 1. Example of room reservation business process model

5. Business process instance

The Business Process Instance is the third abstraction layer of the modeling hierarchy. This layer stems from assigning values to the optimization parameters linked to each model element. Optimization variables are signified by not having a value assigned in this layer. This approach allows establishing a taxonomy of infrastructure allocation problems, based on the set of parameters and variables that have been defined.

Taking the example of the former section, it is important to highlight that even such a simple business model lends itself to several optimization approaches. Specifically, it could lead to the following cases:

- Given the arrival rate for each customer type, the number of rooms to be offered, the full and the discounted price and assuming that there are no channel access restrictions, determine the optimal time T2 to maximize the hotel profit.

- Given the arrival rate for each customer type, the number of rooms to be offered, the time T2 at which the price changes and assuming that there are no channel access restrictions, determine the full and the discounted price that would maximize the hotel profit.

Both cases correspond to the same business process model. They are, therefore, instances of this process. Figure 3 depicts the UML model of process instances, in which, following the same approach as Gutierrez *et al.* (2006), each basic generic entity of the metamodel is associated to an instance that contains the parameter values and eventually the optimization variables. It is consequently at this layer that optimization problem definition takes place. There will be a group of optimization problems depending on the set of known parameters and on the set of variables to be optimized. Whereas the first two levels describe families of process models, in this level we find the organized collection of optimization problems that can be associated with those models.

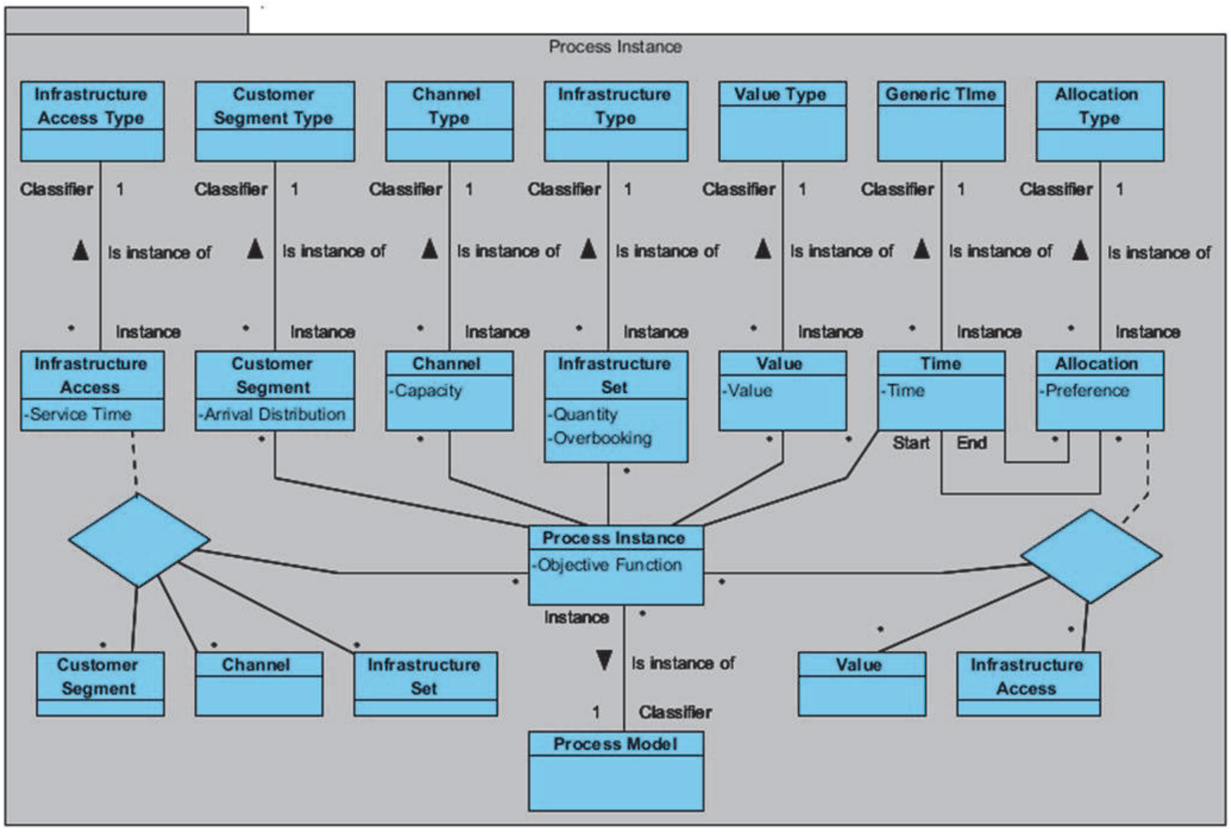


Fig. 3. UML class diagram of the Process Instance definition

We have included only a reduced number of representative parameters/variables for each entity as a means of illustration of the model capabilities. When trying to solve different infrastructure allocation problems, the necessity of further parameters will appear. Nevertheless, it is important to underline that the generic approach taken allows handling the natural extension of the group of parameters without extra coding or compilation (Gutiérrez *et al*. 2006). The user interface component of the DSS for which the model base is designed, should help the definition of entity attributes at this level. These attributes would be inserted in the corresponding database in an automatic pre-compiled procedure.

To enable the handling of problems in which a customer can be assigned to multiple intervals of time, it becomes necessary to make use of the attribute *Preference* of the entity *Allocation*. The attribute *Preference* will be used by the allocation algorithm to decide which slot to assign to a specific customer in case there are different alternatives. The DSS built upon the model base should ease the definition of a set of rules intended to define the values of the attribute.

Finally, the DSS should as well assist in the definition of an objective function, which is an attribute of the *Process Instance* entity, built upon mathematical combinations of a subset of parameters and variables.

6. Example of application to the health care sector

Based on the few cases reported of application of Revenue Management techniques to the Health Sector, and specifically on the pioneering work of Chapman & Carmel (1992) plus the above mentioned more recent work by Gupta & Denton (2008) and Nordgren (2009), an application to the dimensioning of hospital emergency wards is presented here.

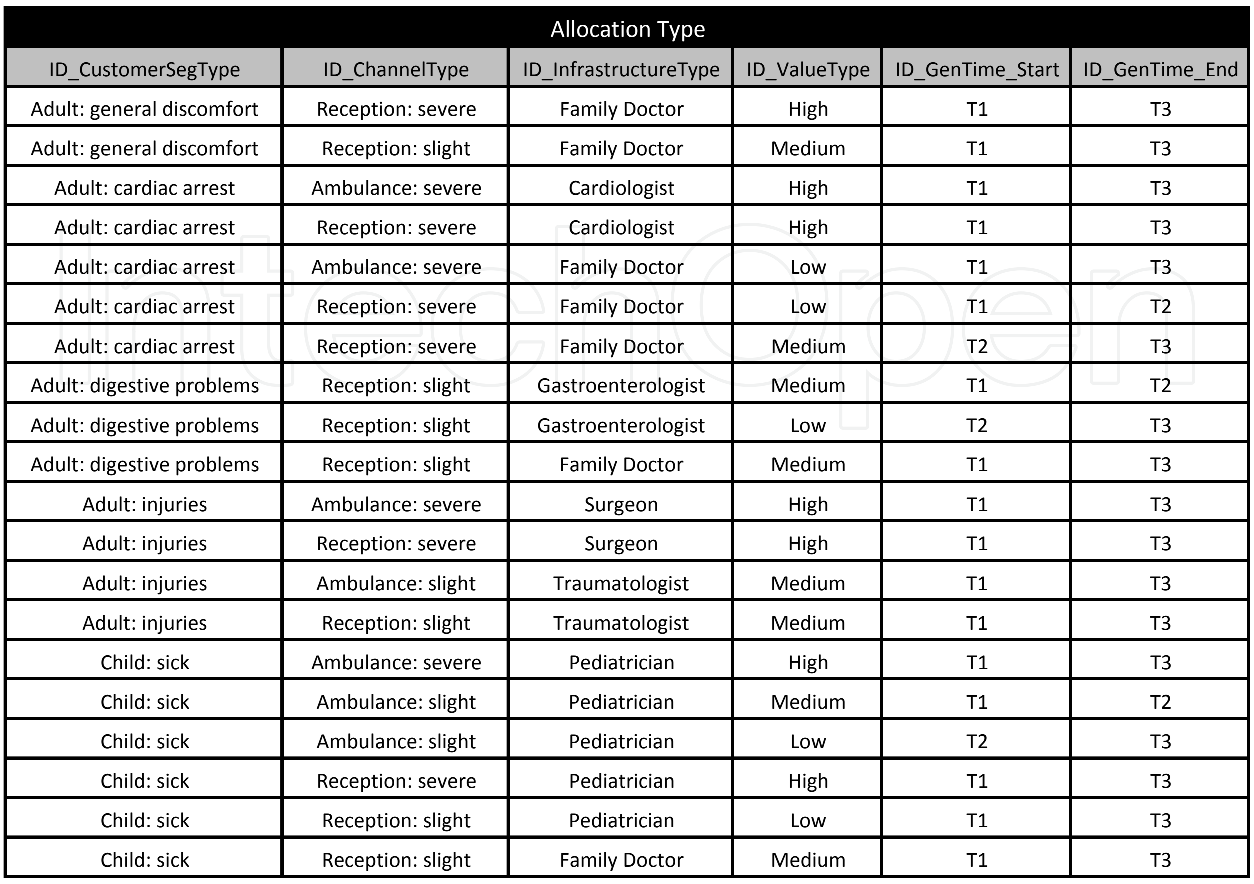
Specialized doctors are treated as the infrastructures whose assignment is to be optimized (cardiologists, general surgeons, pediatricians...). Patients (system customers) are segmented according to their age (children vs. adults) and their symptoms. Channel Types to access the emergency ward are defined as being admitted though reception or through ambulance, distinguishing two levels of severity.

Customer Segment Type	Channel Type	Infrastructure Type	Value Type	Generic Time
ID_CustomerSegType	ID_ChannelType	ID_InfrastructureType	ID_ValueType	ID_GenericTime
Adult: general discomfort	Ambulance: slight	Family Doctor	High	T1
Adult: cardiac arrest	Ambulance: severe	Cardiologist	Medium	T2
Adult: digestive problems	Reception: slight	Gastroenterologist	Low	T3
Adult: injuries	Reception: severe	Traumatologist		
Child: sick		Surgeon		
		Pediatrician		

Table 2. Health Care example direct instances

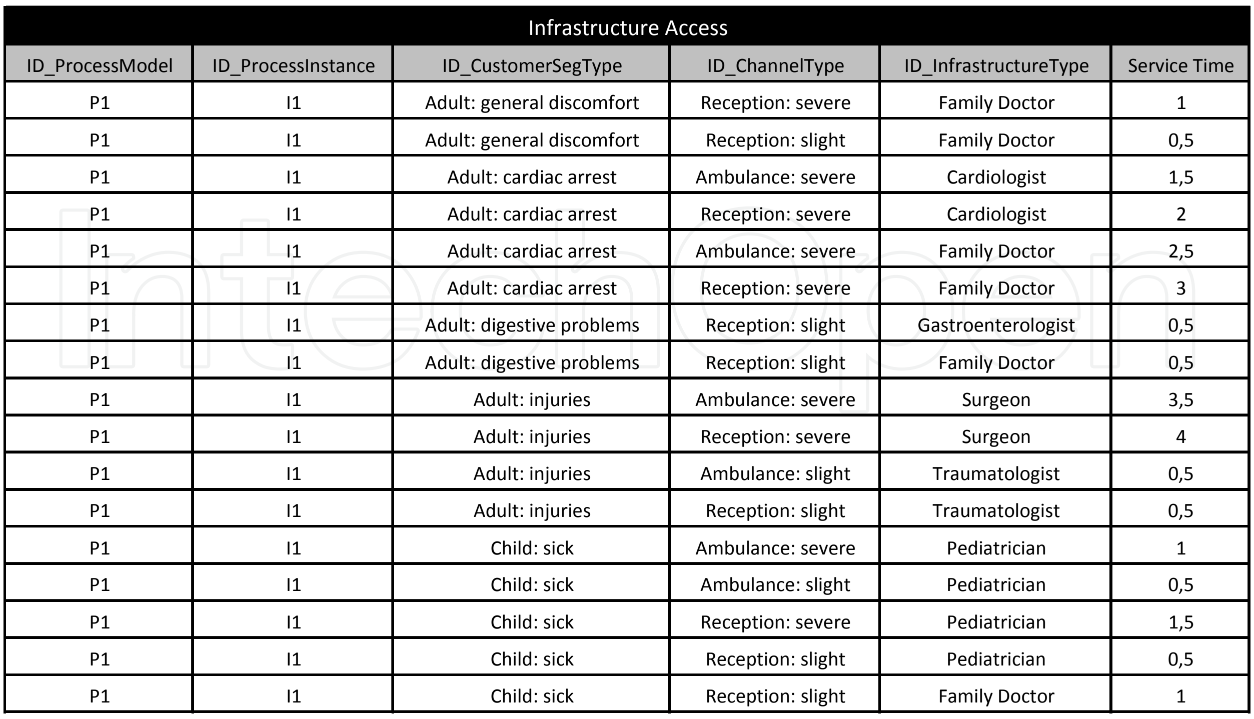
The combined consideration of these three criteria leads to the evaluation of infrastructure access types as high, medium or low value from the point of view of the system's effectiveness. The Generic Times are used to define emergency shifts, since it might make sense not to have certain specialists in some shifts. In the example, three generic times represent two different shifts. Table 2 shows the example values for the model definition.

The process model is defined as the set of possible combinations of the direct instances of the process metamodel entities. Table 3 represents the business process example. Once the process model has been defined, several optimization problems can be stated in the business process instance level. In this example, we describe an infrastructure dimensioning problem, in which the number of medical specialists of each type is to be determined. Thus, in the process instance model of Figure 3, all the attributes of the instance entities but the Quantity attribute of the Infrastructure Set, will receive a value. The Quantity attribute will thus be the variable to be optimized. Attributes that do not have to be considered in the optimization problem—in this example O*verbooking* does not make sense— will receive a default value to avoid confusion with the variables of the problem.



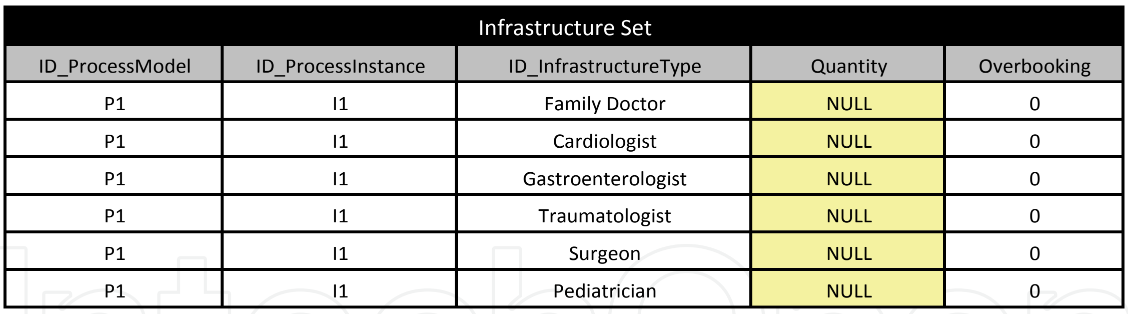
Allocation Type					
ID_CustomerSegType	ID_ChannelType	ID_InfrastructureType	ID_ValueType	ID_GenTime_Start	ID_GenTime_End
Adult: general discomfort	Reception: severe	Family Doctor	High	T1	T3
Adult: general discomfort	Reception: slight	Family Doctor	Medium	T1	T3
Adult: cardiac arrest	Ambulance: severe	Cardiologist	High	T1	T3
Adult: cardiac arrest	Reception: severe	Cardiologist	High	T1	T3
Adult: cardiac arrest	Ambulance: severe	Family Doctor	Low	T1	T3
Adult: cardiac arrest	Reception: severe	Family Doctor	Low	T1	T2
Adult: cardiac arrest	Reception: severe	Family Doctor	Medium	T2	T3
Adult: digestive problems	Reception: slight	Gastroenterologist	Medium	T1	T2
Adult: digestive problems	Reception: slight	Gastroenterologist	Low	T2	T3
Adult: digestive problems	Reception: slight	Family Doctor	Medium	T1	T3
Adult: injuries	Ambulance: severe	Surgeon	High	T1	T3
Adult: injuries	Reception: severe	Surgeon	High	T1	T3
Adult: injuries	Ambulance: slight	Traumatologist	Medium	T1	T3
Adult: injuries	Reception: slight	Traumatologist	Medium	T1	T3
Child: sick	Ambulance: severe	Pediatrician	High	T1	T3
Child: sick	Ambulance: slight	Pediatrician	Medium	T1	T2
Child: sick	Ambulance: slight	Pediatrician	Low	T2	T3
Child: sick	Reception: severe	Pediatrician	High	T1	T3
Child: sick	Reception: slight	Pediatrician	Low	T1	T3
Child: sick	Reception: slight	Family Doctor	Medium	T1	T3

Table 3. Process model of the Health Care example



Infrastructure Access					
ID_ProcessModel	ID_ProcessInstance	ID_CustomerSegType	ID_ChannelType	ID_InfrastructureType	Service Time
P1	I1	Adult: general discomfort	Reception: severe	Family Doctor	1
P1	I1	Adult: general discomfort	Reception: slight	Family Doctor	0,5
P1	I1	Adult: cardiac arrest	Ambulance: severe	Cardiologist	1,5
P1	I1	Adult: cardiac arrest	Reception: severe	Cardiologist	2
P1	I1	Adult: cardiac arrest	Ambulance: severe	Family Doctor	2,5
P1	I1	Adult: cardiac arrest	Reception: severe	Family Doctor	3
P1	I1	Adult: digestive problems	Reception: slight	Gastroenterologist	0,5
P1	I1	Adult: digestive problems	Reception: slight	Family Doctor	0,5
P1	I1	Adult: injuries	Ambulance: severe	Surgeon	3,5
P1	I1	Adult: injuries	Reception: severe	Surgeon	4
P1	I1	Adult: injuries	Ambulance: slight	Traumatologist	0,5
P1	I1	Adult: injuries	Reception: slight	Traumatologist	0,5
P1	I1	Child: sick	Ambulance: severe	Pediatrician	1
P1	I1	Child: sick	Ambulance: slight	Pediatrician	0,5
P1	I1	Child: sick	Reception: severe	Pediatrician	1,5
P1	I1	Child: sick	Reception: slight	Pediatrician	0,5
P1	I1	Child: sick	Reception: slight	Family Doctor	1

Table 4. Infrastructure Access instances in the Health Care example



Infrastructure Set				
ID_ProcessModel	ID_ProcessInstance	ID_InfrastructureType	Quantity	Overbooking
P1	I1	Family Doctor	NULL	0
P1	I1	Cardiologist	NULL	0
P1	I1	Gastroenterologist	NULL	0
P1	I1	Traumatologist	NULL	0
P1	I1	Surgeon	NULL	0
P1	I1	Pediatrician	NULL	0

Table 5. Infrastructure Set instances in the Health Care example

Tables 4 and 5 show representative database record examples. A database implementation would require the utilization of proper numerical identifiers as primary keys of the main entities. Besides, it is necessary to include both the Process Model identifier and the Process Instance identifier in all the instance entities, in order to have multiple examples handled by a unique DSS.

7. Extensions and challenges

Hotel and Health Care examples illustrate the variety of models that can be defined within the proposed generic modeling hierarchy. In this final section we first discuss further possibilities of the model base as well as its limitations, and we identify the main promising extensions of the model; in summary, the challenges to be overcome in order to develop a generic DSS capable of coping with Revenue Management infrastructure allocation problems.

The model base, as it is defined in the previous sections, allows to model infrastructure allocation optimization problems in which different segments of customers, through a set of channels, can access a system to place a request for the use of an infrastructure during an interval of time. In the previous examples, demand for an infrastructure can only take place in a unique interval of time, i.e., in case no infrastructure is available in the defined interval, there is no attempt of allocation in another interval. Nevertheless, the model can handle more generic RM scheduling problems in which each customer segment might be offered the use of an infrastructure in different time slots. Even more, it would be possible to assign different values to the different slots. For instance, in the Health Care sector it makes sense to design a system in which there are some intervals which will be preferably assigned to some segments of customers.

In practice, applicability will be restricted to RM problems that do not require handling highly detailed time slots, since using the model base for such purposes would become tedious and they are better addressed by classic schedulers. On the other hand, the model does not support multi-slot allocation systems, such as medical procedures in which it is known that the first appointment will bring along ulterior ones.

The main extension of our proposal is to build a DSS upon the model base described throughout this chapter. In terms of the Sprague's (1980) classic framework, this would imply developing, on the one hand, an algorithmic module intended to solve the optimization problems, and on the other hand, a user interface module that should ease the definition of the allocation problems. In section 5 we mentioned the main functional requirements for the user interface module. With respect to the algorithmic module, solving the RM problems can be accomplished by different techniques that may be classified in two main groups:

- Algorithmic optimization. Since the purpose of the model base is to cover a generality of infrastructure allocation problems, it is necessary to incorporate a generic algorithmic solution. Among the different optimization techniques, we find particularly interesting to explore the capabilities of Constraint Programming (CP). CP shows natural fitting with this generic approach due to three main characteristics (Tsang, 1993): in the first place, its flexibility in real-time definition of variables, constraints and objective functions; in the second place, the use of domain variables which makes it easier to handle infrastructure access constraints; and thirdly, the ease to define complex multi-criteria objective functions. Nevertheless, it appears to be a hard challenge to achieve a pre-compiled engine capable of dealing with the diversity of optimization problem families that can be modeled with the proposed model base.
- Simulation-based optimization. When stochastic parameters are involved—in RM problems, typically customer arrivals and service times, and occasionally others like customer behavior and choice preferences—simulation appears to be the natural approach. We believe that the model base is extensible to handle a generality of RM simulation models. The extension would stem from defining a 4th layer in the modeling hierarchy which would correspond to the execution level. This layer would correspond to the database model of the DSS and encompass all the execution data associated with a set of simulations of the system to be optimized.

8. Conclusions

A generic model base design for a DSS intended to deal with the problem of maximizing revenue in the allocation of service infrastructures offers great flexibility and allows overcoming the intrinsic dependency of the algorithm on the business process model. This dependency hinders the application of RM techniques through traditional DSS. The hierarchical metamodel-model-instance approach proposed to define the RM DSS model base here shows great potential and addresses the three core elements of the mentioned algorithm dependency: The specific traits of the business processes, the decision variable(s) to be optimized and the business objective(s) to be achieved. It also supports the structured definition of business process typologies along these dimensions, which can lead to applications of generalized interest to the RM community regarding the definition of taxonomies of RM optimization problems. Furthermore, this approach allows defining and including in the DSS the specific parameters required in a given model without ad-hoc coding or compilation. Examples taken from the Hotel and Health Care sectors illustrate the potential of the proposed generic approach. The modeling hierarchy can be extended to encompass the data model of the RM DSS database component.

9. Acknowledgment

This work is supported by the research project DPI2008-04872, “Optimization of service infrastructures assignment through simulation - hotel and health sectors” (Spanish National Research Plan) and by the Human Resources Mobility Action I-D+i 2008-2011 of the Spanish Ministry of Education for a visiting scholar stay at MIT Center for Transportation & Logistics.

10. References

Chiang, W.C; Chen, J.C.H. & Xu, X. (2007). An overview of research on revenue management: Current issues and future research, *Int. J. Revenue Management*, Vol.1, No.1, (January 2007), pp. 97-128, ISSN 1474-7332

Eriksson, H.-E. & Penker, M. (2000). *Business Modeling with UML: Business Patterns at Work*, John Wiley & Sons, ISBN 978-0471295518, NY, USA

Fowler, M. (2004). *UML Distilled: A Brief Guide to the Standard Object Modeling Language*, Addison-Wesley, 3rd ed., Object Technology Series, ISBN 978-0321193681, Boston, USA

Gupta, D. & Denton, B. (2008). Appointment scheduling in Health Care: Challenges and opportunities, IIE Transactions, Vol.40, No.9 (Sep. 2008), pp. 800-819, ISSN:0740-817X

Gutiérrez, M.; Durán, A. & Cocho, P. (2006). An alternative approach to the standard Enterprise Resource Planning life cycle: Enterprise reference metamodeling, *Lecture Notes in Computer Science,* Vol.3982, (May 2006), pp. 964-973, ISSN 0302-9743

Muhanna, W.A. & Pick, R.A. (1994). Meta-modeling concepts and tools for model management: A systems approach, *Management Science*, Vol.40, No.9 (Sep. 1994), pp. 1093-1123, ISSN: 0025-1909

Nordgren, L. (2009). Value creation in health care services: Developing service productivity, *International Journal of Public Sector Management*, Vol.22, No.2 (March 2009), pp.114-127, ISSN 0951-3558

OMG (May 2010). OMG Unified Modeling Language (OMG UML), Infrastructure Version 2.3, Available from:
http://www.omg.org/spec/UML/2.3/Infrastructure/PDF/

OMG (Feb 2011). Introduction to OMG's Unified Modeling Language™ (UML®), Available from : http://www.omg.org/gettingstarted/what_is_uml.htm

Power, D.J. (2002). *Decision Support Systems : Concepts and Resources for Managers,* Greenwood, ISBN 1-56720-497-X, Westport, CT, USA

Sprague, R.H. (1980). A framework for the development of Decision Support Systems, *MIS Quaterly,* Vol.4, No.4, (December 1980), pp. 1-26, ISSN 0276-7783

Talluri, K.T. & Van Ryzin, G.J. (2004). *The Theory and Practice of Revenue Management,* Springer, ISBN 1-4020-7701-7, NY, USA

Tsang, E. (1993). Foundation of Constraint Satisfaction, Academic Press, ISBN 0-12-701610-4, London, GB

Zaki, H. (2000). Forecasting for airline revenue management, *The Journal of Business Forecasting Methods & Systems,* Vol.19, No.1, (January 2000), pp. 2-6, ISSN 0278-6087