

A Systematic Method for the Intentional Modelling and Verification of Business Applications

I. Gmati¹, M. Missikoff², S. Nurcan¹

Abstract Recently, we have witnessed a growing need to involve business people in the early stages of Enterprise Information Systems development. The MAP methodology appears to be a good candidate for involving business people in the early modelling of business applications, reducing the risk of business-IT systems misalignment. Furthermore, in the context of Model Driven Engineering, such a methodology perfectly fits in the upper CIM level. In this paper we revisit the MAP to propose a formal approach capable of providing solid basis to it, necessary when developing automatic tools aimed at supporting the modelling activity and the verification of the produced map diagrams.

Introduction

In the recent period, there has been the increasing need to foster the direct involvement of business experts in the development and maintenance of enterprise information systems (EIS), aiming at reducing the gap in the alignment of information systems and business needs. To bridge this gap, the OMG³ has proposed the Model Driven Architecture (MDA) [6] approach that places the modelling activities at the centre of the EIS development. MDA is organised along three main modelling levels: CIM (Computational Independent Model), PIM (Platform Independent Model), and PSM (Platform Specific Model), with a progression that goes from business oriented models (CIM) to technology oriented models (PSM) of the enterprise software.

Traditionally, the intermediate level, PIM, has been characterised by extensive modelling, aimed at software design specifications. The top level, CIM, concerning the business modelling activities, is mainly expressed in an intuitive,

¹ Univ. La Sorbonne, Paris1

² CNR-IASI, Rome

³ Object Management Group, a major international standardization organization participated by all the primary computer industries

informal fashion, with a prevalence of textual documents, often incomplete and ambiguous. Such imprecise specifications are among the causes of problems in the alignment of business requirements and EIS [10].

Experience proved that the early introduction of formal methods in business modelling has little success. It is better to start modelling business in an intuitive way and then progressively introduce precision and rigour, with an incremental approach. Along this line is positioned the MAP method [3], where a business application is modelled in terms of intentions (goals) and strategies (activities) represented in a diagrammatic form, as nodes and arcs, respectively. The MAP method has been adopted and fruitfully used since long time. In this paper we revisit it, starting from a previous work [5] to further elaborate a formal framework and a new approach to model the contexts, i.e., the involved business objects with their states. Another key contribution of this paper is represented by the use of IF-THEN production rules to model MAP sections. Finally, having introduced an explicit representation of the pre- and post-conditions of a MAP section, we propose a formal approach aimed at a systematic validation of MAP models.

Related work

In the literature, there are several methods for early modelling of enterprise knowledge, such as Zachman [7] or TOGAF [8]. They extensively model an enterprise but they are mainly conceived to be used by people, not by computers. When focusing on enterprise strategies, there are approaches based on “means-end” trees, representing goals and means to reach such goals, in an hierarchical decomposition structure [2]. Another approach is represented by the *i** method [1], where goals are categorised into *soft* and *hard* ones, and it is possible to specify the actors and resources necessary to achieve such goals, in a network of dependencies. All these methods are essentially ‘structural’, in the sense that they do not capture the sequencing of goals and strategies (describing the business logic), as the MAP method does. Another important added value of the MAP methodology is on the explicit modelling of business objects that are involved in different strategies.

The rest of the paper is organised as follows. In Section 2 we introduce a running example, used throughout the paper. In section 3, we provide a first description of the MAP methodology, while in Section 4 we provide a preliminary proposal of a formal framework for its core part, including the principles for the systematic verification of MAP diagrams validity. Finally, Section 5 concludes the paper.

A practical example of MAP usage

In this section a business case is first reported in a narrative, textual form and then as a Map diagram, with a number of associated tables, Section Specification Table (SST), that allow the business context to be represented, i.e., the result of the enactment of a path, in terms of involved business objects with their states.

The business case: loan handling in a bank

Our business case refers to the loan handling process in a bank. A customer presents a loan request to a loan service clerk, in charge of setting up a file with the corresponding information (loan amount, rate, customer account situation, etc.). Then, the clerk performs a first formal check on the loan request and if it evidently exhibits a high risk then it is not further processed. In case the first filter is passed, the request needs to be evaluated, either by the loan service clerk himself (in the simple cases) or by the experts of the financial department (if an advanced evaluation is needed). The outcome of the evaluation is analysed by the loan manager who has the possibility either (i) to accept the loan request, or (ii) to ask the loan service clerk to further review it, or (iii) to ask an extensive re-evaluation to the financial department. If the final decision is positive, the clerk assistant sends to the customer a proposal of loan, specifying the amount, the duration, and the refunding modalities. Then, the customer has to sign the contract in the indicated time span, otherwise the offer is cancelled. When the decision is negative, the clerk assistant notifies the customer with a refusal letter.

This brief text represents a typical preliminary description performed by a business expert. Then, starting from such a kind of statement, the business is modelled by a MAP diagram (a map, for short) as shown below.

A MAP diagram

The Figure 1 represents the business case just illustrated, where the strategies and intentions are respectively indicated by arcs and nodes of a directed multi-graph. In the MAP diagram we also have sections. A section is represented by a source and a target intention, and a strategy connecting the two. Finally, it is necessary to indicate when a section will be activated and what are the effects induced: such knowledge is expressed by pre- and post-conditions, respectively. A Section Specification Table (SST) complements the map diagram.

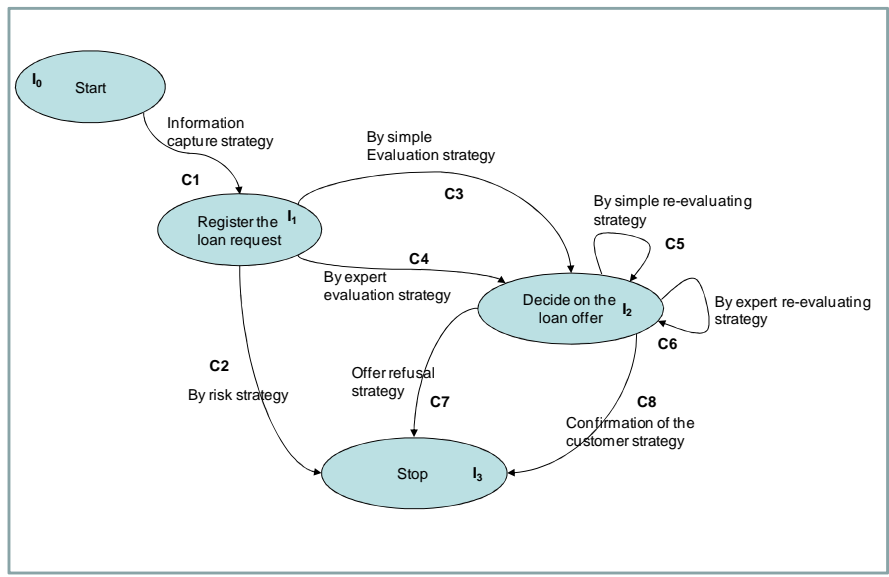


Fig. 1. The business map: Manage loan requests

A Section Specification Table

To complete the specification of a Map diagram we need to provide further knowledge, concerning the objects, with their states, that characterise the different intentions and the involved sections. This knowledge is structured in the Section Specification Tables (SST, one for each section) which are organised in two parts: one part specifies the (source and target) intentions description, with their objects and states; the other part is related to the description of section’s preconditions and post conditions. We consider the example of SST1 which reports the knowledge associated to the section S1.

Table 1. the specification of Section 1 (SST1)

SST	Intentions	Objects	States
S1	(source) Start	loanReq	Arrived
	(target) Register the loan request	loanReq	Registered
	Strategy	Precond	Postcond
	By information capture	LoanReq::arrived	loanReq::registered

A table is an intuitive structure easy to be filled and managed by a business expert. In the next Section 3 we will provide a more precise and rigorous specification of the core elements of the MAP methodology.

The MAP Methodology

The MAP⁴ Meta model

The MAP methodology allows business people to model the high level objectives and strategies of an enterprise, expressed in intentional terms. A MAP diagram represents a first specification of a business application; it contains a finite number of paths from ‘Start’ to ‘Stop’, each of which describes a way to develop a business product (for instance a service to be delivered). Such an application, at a later stage, will be modelled by Business Process diagrams.

An intention represents a desired state of the world, modelled in terms of a set of object states and, at the same time, a starting point for the activation of a strategy necessary to achieve the next intention. A strategy is used to achieve an intention, producing (part of) the desired object states. Since there can be multiple edges entering a node, the MAP is capable of representing different strategies that can be used for achieving, either fully or partially, an intention. Such strategies can be complementary (if more than one is needed to fulfil the intention) or alternative.

A map consists of several sections, each of which encapsulate a strategy, according to a five-tuple:

$$S = \langle prc, psc, I_i, I_j, C_{ij} \rangle,$$

where I_i is the source intention, I_j is the target intention, C_{ij} is the strategy linking the two intentions, and prc and psc are the pre- and post-conditions, respectively. There are two distinct pre-defined intentions, called *Start* and *Stop*, that represent the intentions to start navigating in the map and to stop doing so, respectively. In general, there are several paths in a map diagram from Start to Stop.

The pre- and post-conditions of a section are both defined in the form of Boolean expressions over object states. A section can be activated if its prc is verified. Then, after a finite time, the section is enacted, ensuring that the post-conditions is true. Please note that in this paper we consider a simplified view in

⁴ Please note that we will use the term MAP (full capital) to denote the methodology, while the term “map” will be used to denote a specific MAP diagram.

which: (i) all the preconditions must be verified before a section is activated, therefore we do not accept delayed pre-conditions, becoming true during the course of actions; (ii) we assume an optimistic approach, therefore when a section is activated it will eventually terminate, producing the effects specified in the post-condition. In essence, at this stage we are not dealing with notions such as exceptions, failures, or abort conditions. In general, the achievement of a target intention, using a strategy from a source intention (i.e. the enactment of a section), produces a state change on the involved objects, the creation of new objects, or the destruction of existing objects, according to the post-condition of the said section. We also assume that what is not explicitly indicated will remain as it was before the activation of the strategy (here we wish to recall the ‘frame problem’ [9], guaranteeing the absence of ‘hidden’ side effects).

It is useful to introduce the notion of a *context* as a coherent set of object instances with their states. Each time a section terminates, the context evolves into a new state. The evolution over the time of the context takes place following the navigation through the map.

In Fig. 2 the meta-model of map is reported.

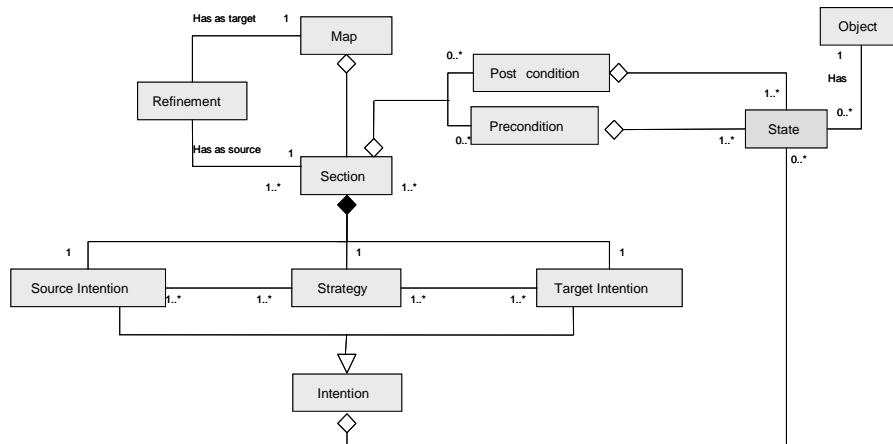


Fig. 2. The MAP Meta-model

The MAP as an intentional process model

As a concrete example, the map shown in Figure 1 has ‘Register the loan request’ and ‘Decide on the loan offer’ as intentions. Both express what is wanted, a sequence of expected results, disregarding the considerations about who, when,

where and how. A strategy is an approach, a manner to achieve an intention (typically implemented by a business process). In Figure 1, we have the section:

$SI = \langle prc, psc, Start, RegisterLoanRequest, InformationCaptureStrategy \rangle$, which links a source intention (*Start*) to a target intention (*RegisterLoanRequest*) through a strategy (*InformationCaptureStrategy*). Furthermore, we assume for instance that $prc = 'Loan\ request\ has\ arrived'$ and $psc = 'Loan\ Request\ Registered'$.

A section of a map can be refined into a more detailed map. This happens when it can be decomposed into more refined intentions and associated strategies. This feature will not be dealt with in this paper. Detailed information can be found in [3] and [12].

Towards a formal specification of MAP

Motivation

In this section we will revisit the MAP methodology with the objective of proposing a formal framework covering its core modelling notions. Such a formal framework will support the map developer on two levels. Firstly in the design process providing a rigorous, unambiguous grounding when building a complex map diagram. Secondly, a formal representation is needed to develop computer supported services; for instance for the simulation of a map diagram, even at a preliminary development stage, or for checking its validity.

In the following, the key notions of the MAP methodology, such as intentions, strategies, sections, pre- and post-conditions, will be formally specified.

MAP Basic definitions

Def: Map. A map is a business application model expressed in intentional terms. It provides a business logic driven representation based on a non-deterministic ordering of intentions (I), strategies (C), and sections (S) [3]:

$$Map = (I, C, S).$$

Example:

$I = (RegisterTheLoanRequest, MakeTheLoanOffer, Start, Stop)$

$C = (InformationCaptureStrategy, ExpertEvaluationStrategy, \dots)$

$S = (\langle prc1, psc1, Start, InformationCaptureStrategy, RegisterTheLoanRequest \rangle, \dots)$

Def: Business Object (BO). It is the conceptual specification of a passive entity that characterizes a given Business Domain (BD) or application. They appear in a business scenario (BS). A BO specification includes a set of attributes that, when fully instantiated, define a business object instance. The attributes of an object are also used to define the object states (see later).

Def: Business Object State (BOS). It is an intentional notion represented by a pair: $(BO, sdef)$, where the state definition ($sdef$) is a labelled Boolean expression defined over the properties of one or more⁵ BOSs. For short, when possible, we will use a synthetic label, bos , to indicate a Business Object State pair. A BOS is in fact a condition that can be checked against a BO instance, therefore: $bos(oi)$ is true, if and only if the BO instance oi is in the $sdef$ state⁶.

Example

$BOS = \{(request, underEval), (loan, granted)\};$

For conciseness, we may adopt for the BOS pair the following compact syntax:

$BOS = \{request::underEval, loan::granted\}.$

Def: Intention. It represents a (intermediate) goal, a desired state of affairs defined by a set of business object states⁷, typically used in a conjunctive Boolean expression (Bex). When there are more than one set of object states (SOS) that satisfy an intention, its Boolean expression is disjunctive⁸.

Example:

$DecideOnLoanOffer = \{(LoanReq::accepted, LoanOffer::issued), (LoanReq::refused, RefusalLetter::sent)\}.$

This is an example of a disjunctive Intention, where the goal is achieved if one (or more) section produces the first pair or the second pair.

Def: Strategy and Section. A strategy is the active element of a map, capable of achieving a goal. It contributes to define a Section that includes the source and

⁵ Note that the state of an object may be endogenous, depending on its own attribute(s), or exogenous, depending on the state of another object. E.g., a LoanRefusal depends on the LoanEval.

⁶ In other terms, the predicate $bos = (bo, sdef)$ can test if the oi is of type bo and its attribute values satisfy the $sdef$.

⁷ Please note the compact syntax: $\langle object \rangle :: \langle state \rangle$ to represent an object state

⁸ Please note that in case of disjunction the objects of each SOS must be related, i.e., there is a unique set of BO for each Intention, and the alternatives are based on alternative states of the same objects.

target intentions, and the pre- and post-conditions (*prc* and *psc*), respectively. A strategy starts from an input context (satisfying the section's pre-conditions) and produces a new context satisfying the section's post-conditions.

In our proposal, a Section can be represented by a labeled (non-deterministic) production rule *S*, of the form:

$$S = IF\ prc\ THEN\ C\ AND\ psc$$

Applying the above rule pattern to the third section of our running example we have the following

Example:

$$S3 = IF\ LoanReq::registered\ THEN\ SimpleEvaluationStg\ AND \\ (LoanReq::accepted,\ LoanOffer::issued)\ OR\ (LoanReq::refused,\ LoanOffer::cancelled)$$

Def: Context. A context is a coherent set of object instances with their states. It is updated by strategies and evolves while a map diagram is traversed. There is an initial Context that then evolves according to the state of affairs determined by the enactment of a sequence of strategies. Each time a strategy terminates, the context is updated with the new object states produced by the former (according to its post-conditions).

The sections' sequencing between the Start and Stop intentions constitutes a "path" in the map. The "context" evolves with the enactment of a path. The context is thus the set of BOS resulting of a path enactment.

Having completed the definition of the key notions of a map model, in the following section we will address issues concerning the verification of validity of the built map.

Verification of a MAP Diagram

There are two main approaches to map diagram verification: a static and a dynamic one. The former is performed by statically analysing the sections with their pre- and post-conditions (as reported in the SSTs) and the intention definitions. In fact, not all possible diagrams, with strategies and intentions, are valid map models. There are a number of basic validity rules to be respected, as reported in [4] and [11], sketchily recapped below:

- Every intention in a map is the source of a strategy except the Stop intention.
- Every intention in a map is the target of a strategy except the Start intention.
- Two contiguous intentions are always connected by (at least) a strategy.

Therefore:

- Maps are connected graphs; there is no isolated intention or dangling strategies.
- There is always (at least) a path from Start to Stop.
- Any section belongs at least to a path between Start and Stop (i.e., there are no extra 'sink' or 'source' nodes).
- No strategy in the map can be defined as a sub-part of another strategy.
- No intention in the map can be defined as a sub-part of another intention.
- Intentions having as result the same part of product should be merged.

Focusing on intentions, according to the definitions given above, we can provide further MAP validity rules [5], to be used in the validity verification method:

- No intention should have object states that do not belong to a precondition of an outgoing strategy (except the “stop” intention).
- No intention should have object states that are not produced by an incoming strategy (except the “start” intention).

The latter requires that we adopt an operational view and then we analyse the diagram by traversing it, with a “enactment” logic. In this perspective the idea is that the control is initially positioned in the Start node and then the navigation (enacting sections) evolves until the Stop node is reached.

By traversing a map diagram we have a sort of “live” validation of the correct sequencing of goals and strategies. This validation is mainly done by business experts who can confirm that the business actually behaves (or it is wished it does) according to the map.

Here we report a sketch of an algorithmic method for the verification that all the intentions of a given map are achievable.

```

Begin
  Forall intentions  $I_i$  in map do
    Extract the business object states  $SOS_i$ 
    Forall  $SST_j$  having  $I_i$  as the target intention do
      Extract the post-condition  $psc_j$ 
       $ProdOS_i = \cup_j psc_j$ 
      If  $SOS_i - ProdOS_i = \emptyset$  then intention  $I_i$  is achievable
      else intention  $I_i$  is not achievable
      fi
    od
  od
End

```

Fig. 3. The map method for intentions' achievability validation

The above validation method addresses only one dimension of the map validation requirements, in particular the last bullet of the list of criteria reported above. Another dimension is represented by the possibility of an interactive validation performed by the business experts. Having introduced a rule-based formalization, it is possible to use a Rule System (such as Drools⁹) to support the dynamic validation. This Rule System approach is based on two concepts: (i) the context and (ii) the production rules.

Conclusion

In this paper we presented a first approach towards a formalization of the MAP methodology, based on production rules to represent sections and a set-theoretic method to represent intentions and contexts. The formal approach is useful to (i) provide a formal semantics to map diagrams and (ii) to lay the basis for the development of automatic tools aimed at their verification and validation. Future activities will concentrate on the formalization of additional parts of the MAP methodology, excluded here for sake of space, and on the implementation of a first prototype aimed at map diagrams validation, along the line of the method represented in the Figure 3 of the previous section,

Bibliography

1. Estrada, H., Martinez, A., Pastor, O., [Mylopoulos](#), J. and [Giorgini](#), P. (2008) A Service-oriented Approach for the i*Framework, *Proceedings of the 3rd International i* Workshop*, Recife, Brazil.
2. Anton A.I. (1996) Goal Based Requirements Analysis, in *Proc. Second Int. Conference on RequirementsEngineering.*, ICRE '96, pp. 136–144.
3. Rolland, C., Prakash, N., Benjamin A. (1999) A Multi-Model View of Process Modeling, *Requirements Engineering Journal*, 4(4) pp 169-187.
4. Rolland, C., Salinesi, C., Etien, A. (2004) Eliciting Gaps in Requirements Change”, *Requirement Engineering Journal* Vol. 9, pp1-15.
5. Rolland, C., Soffer, P. (2005) Combining Intention-Oriented and State-Based Process Modeling, *Conference on Advanced Information Systems Engineering*.
6. Bézivin, J., Gérard, S., Muller, P -A., and Rioux, L. (2003) MDA components: Challenges and Opportunities, in *Proc. Of Metamodeling for MDA*, York.

⁹ <http://jboss.org/drools/drools-guvnor.html>

7. Pereira C.M., Sousa P. (2004) A Method to Define an Enterprise Architecture, *Proceedings of the ACM SAC*, Nicosia.
8. Saha, P. (2004) Analysing The Open Group Architecture Framework from the GERAM Perspective, Tech. Rep. of Open Group.
9. Scherl, RB., Levesque, HJ. (2003) Knowledge, Action, and the Frame Problem, *Artificial Intelligence*, 144, Elsevier.
10. Standish Group (1995) The Chaos Report, www.standishgroup.com.
11. Rolland C, Prakash N. (2001) Matching ERP system functionality to customer requirements. *In: Proceedings of RE'01, 5th international symposium on requirements engineering*, Toronto, Canada, pp 66–75.
12. Nurcan, S., Etien, A., Kaabi, R., Zoukar, I. and Roland, C. (2005) A strategy driven business process modelling approach, *Business Process Management Journal* Vol. 11 No. 6, pp. 628-649.