A Case-Based Inquiry into the Decision Model and Notation (DMN) and the Knowledge Base (KB) Paradigm

Marjolein Deryck*[1], Faruk Hasić[2], Jan Vanthienen[2] and Joost Vennekens[1]

[1] Department of Computer Science
KU Leuven, Campus De Nayer, Sint-Katelijne-Waver, Belgium
{marjolein.deryck;joost.vennekens}@kuleuven.be
[2] Department of Decision Sciences and Information Management
KU Leuven, Leuven, Belgium
{faruk.hasic;jan.vanthienen}@kuleuven.be

Abstract. Modelling decisions in organisations is a challenging task. Deciding which modelling language to use for the problem at hand is a fundamental question. We investigate the Decision Model and Notation (DMN) standard and the IDP knowledge base system (KBS) in their effectiveness to model and solve specific real-life case problems. This paper presents two cases that are solved with DMN and IDP: (1) Income taxation for foreign artists temporarily working in Belgium; and (2) Registration duties when purchasing real-estate in Belgium. DMN is used as a front-end method, assisting the business analyst in the analysis and modelling of the business domain and communication with the domain expert. It is complemented with the representation of the logic in IDP as back-end system, which allows more forms of inference.

Keywords. Decision Modelling, Decision Model and Notation, DMN, Knowledge Base Paradigm, IDP

1 Introduction

Today companies face a high level of complexity: omnichannel distribution, geographical dispersion, customised product offers, customer's pressure for short delivery cycles, etc. Moreover, regulations on topics like reporting (e.g., Sarbanes-Oxley Act), consumer data processing (e.g., EU General Data Protection Regulation), product quality (e.g., EU General Product Safety Directive) are becoming ever more stringent [1–3]. To top it all off, these challenges are in themselves subject to rapid change. In light of this complex, changing and highly demanding environment, coherent, traceable and adaptable operational decisions are a must. Consequently, automated correctness and consistency checking should be common practice [4]. In practice, however, the bulk of business knowledge exists only in informal policies, operating procedures, the source code of legacy systems, system parameters or even in experts' heads [4, 5]. In other words, a

* Corresponding author.

separate database of business rules that can be maintained and managed as a whole, is rare. This is surprising, as there exists a large variety of business rule methods and tools. Perhaps a lack of insight in the applicability and pros and cons of these methods can explain this enigma. Or perhaps the ideal method, that combines user-friendliness with versatile use is lacking up to now. In this paper we investigate the combination of two decision logic methods: Decision Modelling (DM) that excels in user friendliness, and the Knowledge Base Paradigm (KBP) that excels in versatility. In two case studies we apply both methods and evaluate their respective contributions. Both methods describe the problem domain, each in their own way, without imposing a particular execution strategy on this knowledge. By separating the business knowledge from the execution procedure they produce models that are more easily readable and adjustable, which improves the agility of the business logic.

For decision modelling we use Decision Model and Notation (DMN), a standard published by the Object Management Group (OMG) [6]. Because of the intuitive graphical notation, decisions in DMN format can easily be read and validated by business users. The KBP-system IDP uses a derivative of First Order Logic (FO(.)) to formalise knowledge of the problem domain. Afterwards different algorithms use this information to find suitable solutions for the user. As such the use of the KBP offers a company the possibility to use the information in ways that were perhaps not envisaged beforehand [7].

This paper is structured as follows. In Section 2, relevant work on decision modelling and knowledge representation is discussed. Section 3 presents two real-life cases that are solved with DMN and IDP, while Section 4 outlines advantages and disadvantages of each approach and provides a broader discussion on the comparison between the two paradigms, i.e. the decision modelling paradigm and the knowledge base paradigm. Finally, Section 5 provides conclusions and directions for future research.

2 Decision Modelling and the Knowledge Base Paradigm

For our case studies we selected DMN and KBP as methods to model the decision logic. Both methods are declarative in nature, meaning that only what needs to be achieved is formalised, but not how this should be done.

2.1 Decision Modelling

An increased interest in modelling decision is present in scientific work in the field of process management, as illustrated by the body of recent literature on Decision Model and Notation (DMN) [6, 8–11]. DMN is a declarative decision language and it does not provide a decision resolution mechanism, which is left to the invoking context. The same holds for the processing and storage of outputs and intermediate results. With this recently introduced OMG standard for modelling decisions, it has become possible to extract decisions from processes and to model them separately according to the *Separation of Concerns* paradigm, hence enhancing the understandability, scalability, and maintainability of processes, as well as that of the underlying decisions [12–15]. DMN consists of two levels. First, the decision requirement level takes the form of a Decision Requirement Diagram (DRD) that depicts decisions and subdecisions, business knowledge models, input data, and knowledge sources. It is used to portray the requirements of decisions and the dependencies between the different constructs in the decision model. Second, the decision logic level is used to specify the underlying decision logic. It is usually represented in the form of decision tables. The standard also provides a formal expression language FEEL (Friendly Enough Expression Language) that allows the execution of decision tables on a decision engine, as well as boxed expressions and a metamodel and schema. Decision tables are considered the core concept of DMN, and they contain the necessary information to automate decision-making. The DRD is mainly used to get a high-level understanding of the structure of the problem domain, but does not contain additional information.

Literature has focused mainly on the integration of decision and process models, as well as on decision services based on DMN [8,14,16,17]. Little attention is given to the interaction of DMN with other paradigms, e.g. the knowledge base paradigm [9]. This paper aims at closing that gap in the following sections.

2.2 Knowledge Representation and Reasoning

The field of Knowledge Representation and Reasoning offers many different approaches and systems. In this paper, we have chosen to make use of the IDP system, a state-of-the-art implementation of the Knowledge Base Paradigm [7,18]. IDP allows domain knowledge to be expressed in the FO(.) language, a rich, typed extension of classical logic. It then allows a variety of different inference tasks to be applied to such a knowledge base. Probably the most used inference task is that of *model expansion*: given an interpretation I for part of the vocabulary of a given theory T, the IDP system is asked to compute a interpretation J for the remaining part of the vocabulary such that the two interpretations together satisfy the theory, i.e., $I \cup J \models T$.

For instance, in order to compute a coloring of a graph, we might consider the typed vocabulary consisting of types $Node$ and $Color$, the relation $Edge(Node, Node)$ and the function $Color : Node \rightarrow Color$. We can then express the required domain knowledge by the theory T that consists of the following formula, stating that two connected nodes must have a different color:

$$\forall x\ y : Edge(x, y) \Rightarrow Color(x) \neq Color(y).$$

We can then provide the IDP system with an interpretation I for the types $Node$ and $Color$, and the predicate $Edge$, e.g.:

$$\begin{aligned} Node^I &= \{A, B, C\} \\ Color^I &= \{R, G, B\} \\ Edge^I &= \{(A, B), (B, C), (A, C)\} \end{aligned}$$

and ask it to compute an interpretation J for the function Color such that $I \cup J \models T$. Alternatively, we may also provide an interpretation for the function Color and ask IDP to compute a corresponding interpretation for the relation Edge; in other words, the same IDP theory can be used for different input/output behaviors. The IDP systems contains a number of different algorithms and techniques from domains such as Answer Set Programming, Constraint Programming and Logic Programming that allow it to efficiently implement this flexible behavior.

In addition to model expansion, IDP also allows other inference tasks to be solved for a given theory. For instance, it also supports the task of *optimising* the value of a given term. If we consider the term $\#\{x : \mathit{Color}(x) = R\}$, which represents the cardinality of the set of nodes that are red, we can ask IDP to compute not just any model expansion of the theory T and a particular input interpretation I, but the model expansion J for which the interpretation t^J of this term t is minimal. In this way, we compute the graph coloring in which as few nodes as possible are colored red.

As a final example, we mention the inference task of *propagation*: given an input interpretation I and a theory T, compute all of the consequences of I according to T, formally defined as the set of all atomic properties that are either true in all model expansions of T in I, or false in all of them. Informally this means that, given an input interpretation I, forced and forbidden values are derived and uncertain values are reported.

3 Case Studies

In this section we provide two case studies that have been tackled by both decision modelling and knowledge representation. The first case deals with income tax policies for visiting artists in Belgium and is implemented in the Avola decision management tool. The second case concerns registration duties when buying property in Belgium. It is implemented in the open source tool OpenRules. This tool is Excel-based, which is an advantage for the notary who is already familiar with the spreadsheet program. For both cases, the advantages and disadvantages of each solution are elaborated upon.

3.1 Income tax management

Background. Like researchers, artists often travel to other countries to present their work. They are remunerated in the country where they perform,which raises questions regarding income taxation. The legislation on income tax for these specific situations is quite complex, and so are the double taxation agreements between different countries, as they all tend to differ from each other quite significantly. Hence, there is a need for transparency, not only for the artists who need to know how much they can earn, where to file for taxes and under which tariffs, but also for arts and culture institutions inviting foreign artists to their country, production studios invoking the help of foreign artists in their productions, and even government employees charged with tax collection. A decision model can be offered as a service to all parties involved in the tax management problem: government agencies, artists subject to taxation, production studios, and arts and culture institutions and federations. This case was put forward by oKo, a Belgian industry federation for the arts, and was carried out in collaboration with Avola, a decision management tool vendor. The problem was modelled in the Avola tool and implemented as a web service.

Decision Modelling. The resulting DMN model distinguishes eight core decisions needed for the implementation of the tax regulations. A simple decision requirements model can be found in Figure 1. `Income Tax` is the top-level decision and it provides the category of income tax that the travelling artist belongs to, e.g. fully taxable, not taxable, partially taxable, or specific tax rates. `Subsidised Exemption` determines whether the artist can enjoy tax exemptions based on subsidies allocated to the artist's organisation by the government. `Individual Exemption` investigates whether the artist can enjoy tax exemptions based on his personal situation, e.g. depending on the number of full time equivalent days that he worked in the country of the performance. `Organisational Exemption` determines whether the artist can enjoy tax exemptions based on potential special treatments and agreements that the artist's working organisation enjoys from the government of the country of performance. `Fiscal Articles` figures out which fiscal articles are applicable to the artist. `Double Tax Agreement` indicates the logic and the rates of the double tax agreement between the originating country and the country of performance. `Initiation of Double Tax Agreement` gives the date of validity of the double tax agreement between the originating country and the country of performance. Finally, `Application Area` evaluates whether the time the artist worked in the country he visited falls under a period of double tax agreements between the artist's country of origin and the country of performance.

Followed approach. The information relevant for the tax decision is available in the form of law texts and procedures. Hence we primarily need to construct decision tables from the textual descriptions. The decision tables can then be checked for completeness, correctness and possible contradictions. After this post processing step, the decision tables can be simplified and represented in an unambiguous manner.

However, the legal texts were not sufficient to model the tax management problem, as they are at times rather complex to understand without the necessary legal training. The modellers have an information systems background and not a legal one. Therefore, the models obtained from the texts were built in close interaction with domain experts. In a dialogue mode, the modeller and the experts gradually discover relevant criteria and outcomes and refine the tables until a full description of the decision logic is obtained. For this tax management problem, we had multiple iterative meetings to validate, amend, clarify,

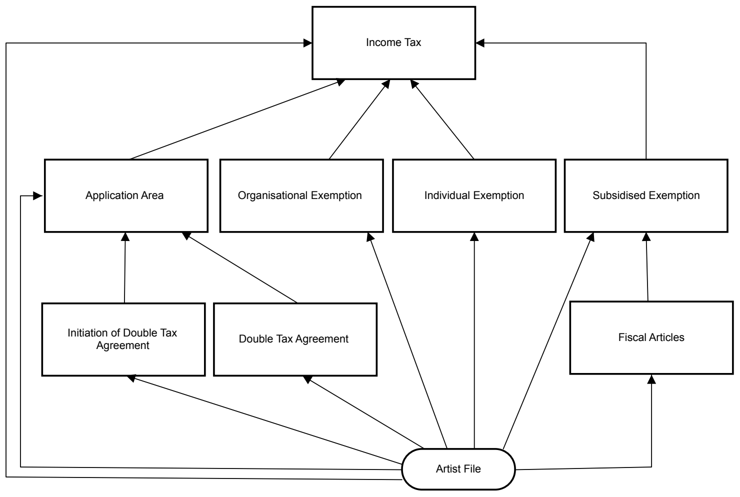


Fig. 1: Tax management decision model.

and improve the decision logic obtained in the modelling cycles from the law texts. The knowledge authorities that were consulted are legal experts attached to either a Belgian industry federation for performing artists or the ministry of culture and media. The experts in both industry and government provided valuable input towards understanding all legal requirements specified in laws, tax agreements, and government policies. This ensured the correct translation of legal requirements into decision rules.

Hence, this decision elicitation process happened in an iterative manner in the following steps:

1. Obtain conditions, condition intervals and outcomes of the law texts.
2. Formalise the conditions and outcomes into decision rules.
3. Aggregate the specified rules into decision tables.
4. Check for completeness, correctness and contradictions.
5. Simplify the decision table and display it.
6. Present the decision tables to the domain expert for validation.
7. If needed, go back to the previous steps until a full description of the decision logic is obtained that is validated by the domain experts.

Especially Step 6 is an important step to ensure correct interpretation of law texts, i.e., using decision tables as a communication tool for logic discussion and validation with domain experts. This is due to the fact that decision tables are relatively easy to read and to understand, even for non-expert decision modellers.

Evaluation. Based on the decision model, the Avola decision management tool allowed us to easily deploy a decision web service that returns the decision outcome of the top-level decision *Income Tax* given a number of input data entries provided by the user of the service. This service can be used by artists to query in which income tax category they belong, by industry organisations and production houses that wish to employ foreign performing artists, as well as by government employees for decision support in tax collection. That way, tax rule compliance becomes transparent to all parties and ambiguities are avoided. We briefly sum up the data input fields needed for the service invocation: the country in which the artist is performing (currently only implemented for Belgium), the origin country of the performing artist, type of legal personality of the organisation or artist, the origin country of the performing organisation, income earned by the artist in the visiting country in the past year, whether all performing artists of the performance share the same origin country within the organisation, date of performance, whether the artist resided for more than 183 days over the past year in the country that he is visiting, the country of fiscal domicile of the artist, and the percentage of government subsidies allocated to the performance. Based on these ten input values, the web service invokes its underlying decision model and returns the tax policy category.

In analogy with the model constructed for Belgium, the service can be expanded for other countries of the European Union and offered in different languages. Currently, oKo is looking to share this model with the European industry federation which will then disseminate the approach to all national and regional industry federations in order to achieve unanimity in tax transparency across the European Union.

Knowledge Representation. Once the DMN tables were completed, they were used as starting point for the development of the knowledge base in IDP. The lessons learned from this case are overlapping with the lessons learned from the next case, and will be presented there.

3.2 Registration duties

Background. When purchasing real-estate in Belgium, registration duties must be paid. Duties differ between regions and may deviate depending on characteristics of the buyer, the property, the seller and the location of the real-estate. The decision model of the applicable law needs to translate all legal requirements in a user-friendly, comprehensible overview. As some of the exemptions are not cumulative, the implementation should allow for the optimisation of the amount to be paid. It should also avoid asking the user for information that is difficult to retrieve, if it is already clear from the available information that an exemption cannot be granted. Therefore the system should allow flexible and minimal information input. The request for this application originated from a notary's office, with the explicit demand to focus on exceptional cases and on reasoning with partial information.

Decision Modelling. The calculation of registration fees depends on multiple sub-decisions: the value of the property taking into account possible discounts, called *abattement*, the applicable tax rate, and portability of earlier paid registration duties. The use of the DRD to have a visual overview of the different sub-decisions and their dependencies is a convenient starting point to understand the problem at hand, and to structure the development of both the decision tables and the IDP model.

Followed approach. OpenRules was used as decision engine as it is available for free and has a low entry effort because it is based on Microsoft Excel files. Although the development of a DRD is recommended as a first step, the program itself does not contain DRD-drawing functionalities. The DRD can be included in the executable file as documentation, but this is optional as the DRD does not impact execution. For this case, multiple overview models with different levels of granularity were developed. At first a rather high-level DRD (upper right panel of Figure 2) was developed.

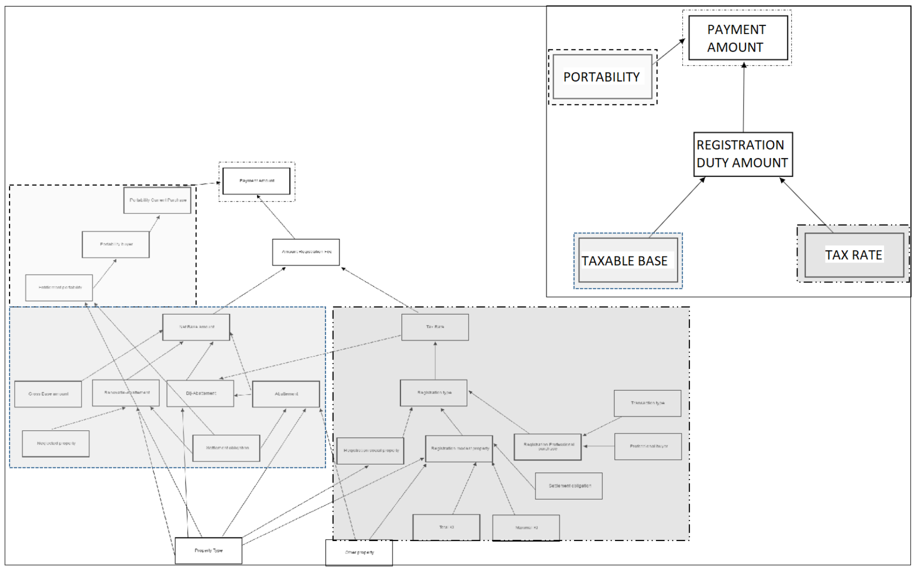
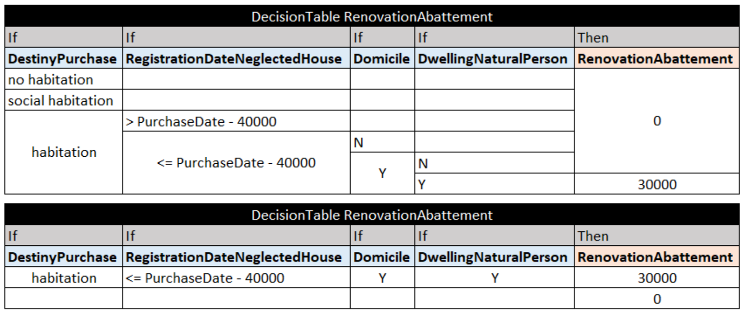


Fig. 2: DRD registration duties with abstract (top right) and expanded (bottom left) view.

The advantage of this figure is its simplicity: for readers it is easy to understand which elements impact the the registration duties. However, the abstraction level is too high to use it as a practical input for the decision tables. Therefore a second DRD with all decision nodes was developed (see left panel of Figure 2. Although this one is complete, it loses the advantage of comprehensibility of the high level model. Of course both DRDs are interrelated in the sense that the elaborate DRD detailed the decisions that were captured in one decision node in the high level DRD. It is an intuitive extension of the DMN standard to expand the abstract top right DRD in the bottom left finer-grained DRD.

Next, the decision tables were constructed. Each row describes a combination of conditions, i.e., one condition per column (*if* or *condition* column) that all need to be applicable, and leads to one or multiple output columns (*then* or *conclusion* columns). It is considered a good design principle to represent all possible combinations and to ensure mutual exclusivity of all rows, which results in single hit tables with the *Unique* hit policy [19]. The relevant legislation identifies several discounts which are each applicable in a specific situation that is characterised by a conjunction of multiple conditions. In this case, tables with the *Unique* hit policy become long and triangularly filled, as shown by the upper table in Figure 3. An empty cell in such a table means that any value can be chosen.



DecisionTable RenovationAbattement

If	If	If	If	Then
DestinyPurchase	**RegistrationDateNeglectedHouse**	**Domicile**	**DwellingNaturalPerson**	**RenovationAbattement**
no habitation				0
social habitation				0
habitation	> PurchaseDate - 40000			0
habitation	<= PurchaseDate - 40000	N		0
habitation	<= PurchaseDate - 40000	Y	N	0
habitation	<= PurchaseDate - 40000	Y	Y	30000

DecisionTable RenovationAbattement

If	If	If	If	Then
DestinyPurchase	**RegistrationDateNeglectedHouse**	**Domicile**	**DwellingNaturalPerson**	**RenovationAbattement**
habitation	<= PurchaseDate - 40000	Y	Y	30000
				0

Fig. 3: Decision Tables representing the same logic to grant a renovation abatement modelled with resp. Unique and First hit policy.

Alternatively this table can be compressed by using a table with the *First* hit policy. In such a table multiple rows may be applicable, but only the first of these is actually applied. The second table in fugure 3 shows that such a table can be used to compactly represent the same information as in the first table. However, from the point of view of verification and validation this solution is considered bad practice. At one hand the use of *First* hit policy means that the order of the rows in the table affects its correctness. At the other hand the use of a blank row always leads to a complete table, but prevents the detection of missing combinations in the other rules of the table. The explicit assignment of hit policies is not possible in OpenRules, but the execution of single hit tables finishes after a hit is found, i.e. de facto first hit policy.

Evaluation. The *First* hit policy can be used to create concise and easily readable tables. Users should be aware that DMN offers more possible hit policies, and that the use of *First* hit policy is debatable, as this complicates the (visual) consistency and completeness checking of tables [19]. Moreover, it goes against the declarative assertion of DMN, because it means that the order in which conditions are checked is relevant [20].

OpenRules evaluates tables in a context-dependent way. For instance, to determine if a buyer can register for *modest housing*, a certain condition must be checked for all previously acquired properties of the buyers. This is modelled with an iterate table. For each property some information is introduced, e.g., if it is a plot or a house, and the value assigned to the real-estate, called *cadastral income*. This information is also needed for the actual purchase itself. The same variable name can be used in both situations and the meaning is derived from the context. It is not possible to refer specifically to the *cadastral income* of a particular property.

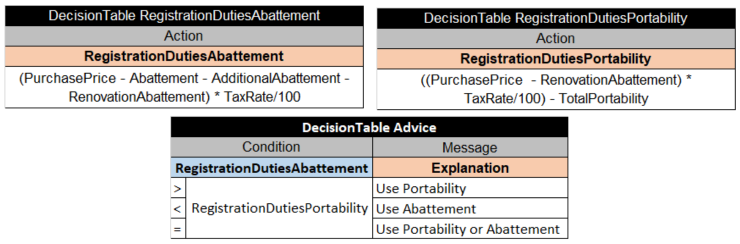
OpenRules executes the specified rules in a predefined order. When information is missing, the output value will be put at -1. Rules that can be executed with the available information will be executed, which may lead to a solution that simply ignores unknown facts. Although this way of working might be desirable in some situations, it does not comply with the way the notary wants to use partial information. He prefers to use available information to narrow down the complete domain of possible solutions, keeping visible other options on which no information is currently available.

The decision tables were instrumental to make sure the complex juridical jargon was understood correctly, as the tables are easily readable by domain experts. From the modeller's point of view it was easy to start creating the application, as it is straightforward for (business) people familiar with Excel. The possibility to reuse the outcome from one table in others is essential for this. The requirement to formalise all variables in a glossary helps to keep overview.

The data modelled in OpenRules can be used to find a solution based on the available inputs. A distinction is made between input variables, internal variables (i.e. intermediate results), and output variables. The value of an internal variable cannot be forced manually. Instead, values provided for for internal or output variables will be checked against the calculated outcome, and deviations will be reported as an error. The format allows manual checks for correctness and completeness and proves to be useful in the process of knowledge elicitation. DMN tables in theory allow for reverse reasoning from output to input variables, but this functionality is not supported by OpenRules. The optimisation of registration rights is done by introducing different variables for the different options and comparing them in a later stage, as demonstrated in Figure 4.

Knowledge Representation. As the functions of reasoning with partial information and optimisation of registration rights that are important for the notary's office are not performed satisfactorily by OpenRules, we also tried to use the IDP-system.

Followed approach The DMN tables were transferred to IDP-definitions one by one. Here the definition of default values was used frequently. For instance, the table in figure 3 was formulated as:



DecisionTable RegistrationDutiesAbattement
Action
RegistrationDutiesAbattement
(PurchasePrice - Abattement - AdditionalAbattement - RenovationAbattement) * TaxRate/100

DecisionTable RegistrationDutiesPortability
Action
RegistrationDutiesPortability
((PurchasePrice - RenovationAbattement) * TaxRate/100) - TotalPortability

DecisionTable Advice		
Condition		Message
RegistrationDutiesAbattement		**Explanation**
>	RegistrationDutiesPortability	Use Portability
<	RegistrationDutiesPortability	Use Abattement
=	RegistrationDutiesPortability	Use Portability or Abattement

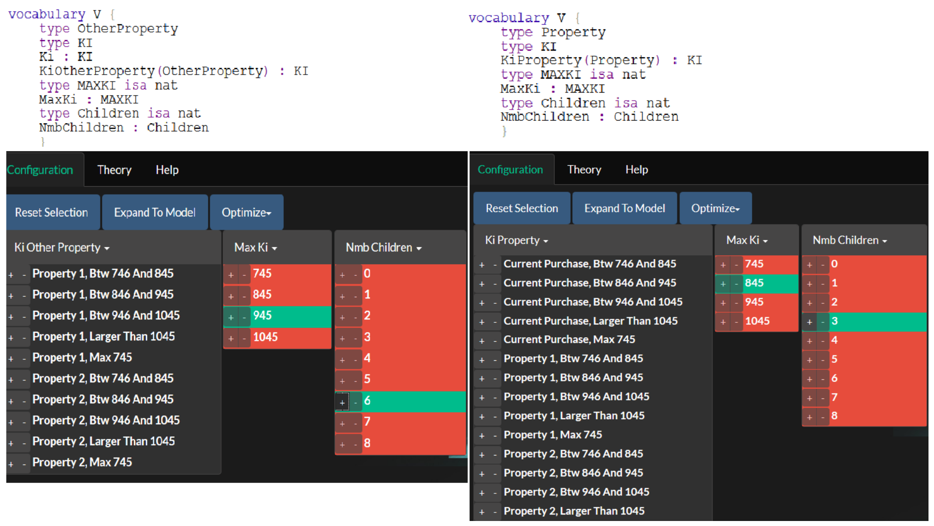
Fig. 4: Optimisation in OpenRules.

```
{  RenovationAbattement = -30000 ← DestinyPurchase = habitation &
                                   RegistrationDateNeglectedHouse ≥ PurchaseDate - 40000 &
                                   Domicile &
                                   DwellingNaturalPerson.
RenovationAbattement = 0 ← RenovationAbattement ≠ -30000.  }
```

This results in a correct and concise representation. With *model expansion* all models that comply with the given information can be obtained. To handle the requirements of reasoning with incomplete information, we used the approach from [9]. The inference task *propagation* is used to derive all consequences of the information that is already available. Based on this, the GUI shown in figure 5 is updated: mandatory values are highlighted in green and impossible values in red. As long as no colour is assigned, information relevant for this decision is missing. The translation of variable types from OpenRules to IDP can happen in multiple ways. Consider for instance the *cadastral income* of a specific property: It was captured as the variable Ki in OpenRules, and depending on the context this pointed to the Ki of the current purchase, or the Ki of previously acquired real estate. In IDP this can be solved by creating a function that maps all properties to their *cadastral income*, as shown in the right panel of Figure 5. This option seems logically consistent, but it could lead to the use of (less preferred) partial functions for attributes of the property that are relevant for the current purchase but not for the other real estate. A way to circumvent this, is to create a function specifically for previously acquired real estate, and to create an additional constant to reflect the *cadastral income* of the current purchase, as shown in the left panel of Figure 5. It has an additional advantage of a clearer distinction between data from both types of purchases. The effect of this choice on the GUI is shown in the screenshots below.

Another form inference that was used is the *optimisation* of selected terms. In many cases buyers want to minimise current total registration duty payment. This can be done by minimising following term:
$termRegistrationPayment : V\{(BaseAmount * TaxRate) - TotalPortability\}$
It is possible to create other terms to optimise other criteria.

Evaluation. The fact that the analysis of the problem domain was already done with the DMN-tables allowed the modeller to focus exclusively on the formula-



```
vocabulary V {
    type OtherProperty
    type KI
    Ki : KI
    KiOtherProperty(OtherProperty) : KI
    type MAXKI isa nat
    MaxKi : MAXKI
    type Children isa nat
    NmbChildren : Children
    }
```

```
vocabulary V {
    type Property
    type KI
    KiProperty(Property) : KI
    type MAXKI isa nat
    MaxKi : MAXKI
    type Children isa nat
    NmbChildren : Children
    }
```

Ki Other Property: Property 1, Btw 746 And 845; Property 1, Btw 846 And 945; Property 1, Btw 946 And 1045; Property 1, Larger Than 1045; Property 1, Max 745; Property 2, Btw 746 And 845; Property 2, Btw 846 And 945; Property 2, Btw 946 And 1045; Property 2, Larger Than 1045; Property 2, Max 745

Max Ki: 745; 845; 945; 1045

Nmb Children: 0; 1; 2; 3; 4; 5; 6; 7; 8

Ki Property: Current Purchase, Btw 746 And 845; Current Purchase, Btw 846 And 945; Current Purchase, Btw 946 And 1045; Current Purchase, Larger Than 1045; Current Purchase, Max 745; Property 1, Btw 746 And 845; Property 1, Btw 846 And 945; Property 1, Btw 946 And 1045; Property 1, Larger Than 1045; Property 1, Max 745; Property 2, Btw 746 And 845; Property 2, Btw 846 And 945; Property 2, Btw 946 And 1045; Property 2, Larger Than 1045

Max Ki: 745; 845; 945; 1045

Nmb Children: 0; 1; 2; 3; 4; 5; 6; 7; 8

Fig. 5: Translation of variables and impact on interface.

tion in the FO(.) language. Therefore the development of the IDP model was relatively easy. The GUI is derived automatically from the IDP model and requires no extra work. The interface in which the propagation of available information is visible, is perceived as very user friendly. The notary sees at a glance which information needs to be requested from the buyer, and which information is superfluous. Therefore the application provides useful support during client meetings. Further improvements in usability can be obtained by further exploring the possibilities of separated vocabularies. For the discounts for modest housing, *abattement* and portability certain settlement restrictions are in place. These restrictions differ slightly for the different discounts. As a result three variables with almost the same name are created. By working with different vocabularies groups of related variables can be grouped together, making it easier to visually relate the correct settlement variable with the intended concept.

Inherent to the concept of a KBS is the absence of a distinction between input and output variables. Besides the different inference possibilities this also means in this case that variables referred to as *internal* by OpenRules can be selected manually. An example is the definition of a professional seller. If this is a large organisation or a frequent client of the notary, the notary probably knows it is a professional seller without checking the different conditions. However, for a new professional buyer/seller the determination of the legal statute should happen by defining the prerequisite conditions. IDP allows the notary to choose if he wants to manually select the value *Professional Buyer*, or if he wants to check the preconditions.

4 Insights on the Modelling and Usage of DMN and IDP

By applying DMN and KRR in these two cases, some insights became apparent. To start with, it turned out to be less straightforward than expected to describe the decision rules. Beforehand, we assumed that laws and regulations would be easy to formalise as they state clear rules. In practice, however, laws are often formulated cryptically and vaguely, which make them hard to understand for unversed readers and open to interpretation. Law rulings may give or change such interpretations, which makes legislation a living matter rather than a fixed fact. Changes and exceptions occur most often at a detailed level. Another difficulty is that rules that apply to a subject are often impacted by rules that apply to the group of which the subject is part, e.g., an artist's taxation depends on the company. This phenomenon occurs in both cases studies. The general structure of the decision tree is not immediately impacted, but at the most fine-grained level the number of columns in decision tables tends to expand for each exception. As living matter, the exceptions are hardest to model, and at some places the outcome of a decision is to check with an expert. Sadly, the largest added value of an automated system would be to give a certain decision for these difficult cases. The conclusion from the case studies suggests that rule-based systems for legislation should either be conceived as decision support systems, or that experts themselves should continue to model their knowledge for the very exceptional cases. It also suggests that the quality and consistency of law texts could benefit from the use of decision tables in their creation process.

Another concern from a modelling point of view is the use of the DRDs in DMN. Currently the modeller has the choice between a high level DRD that shows the large structure of the problem domain at a glance, and a detailed DRD that can be used as the basis for the decision tables. Although a mapping between these two levels is possible, it would be a natural extension of DMN to have a separate notation for high level decision nodes that can be expanded into lower level decisions, analogous to the formulation of sub-processes in BPMN [21].

IDP and DMN prove to be rather compatible as DMN decision tables can be used as a starting point for the creation of IDP rules. We consider this good practice as DMN tables allow for the verification and validation of decision rules, e.g., identifying missing or overlapping rules [22, 23]. IDP, however, does not provide such verification and validation mechanisms. Rather, in IDP errors need to be discovered by checking the different solutions. Another incentive to start from DMN when building IDP rules is that DMN, unlike IDP, boasts a visual notation that is understandable to the domain experts. As such, DMN decision tables can be used as a means of communication between the modeller and the domain experts for the purpose of decision rule checking and validation as often proved necessary in our two case studies.

When it comes to the use of DMN and IDP, both are declarative, which means that they are not reserved for a specific inference form. However, DMN tool developers usually stick to simple forward chaining approaches with complete and correct input data, thus only answering a single question: given the input data, what is the outcome of the top-level decision? The DMN standard [6] allows for the use of different inference strategies, with missing data or top-down approaches such as backward chaining. Until now, the implementation of such inference strategies is largely lacking. In IDP, on the other hand, more algorithms have been developed to allow different forms of inference and answer different kind of questions based on the same model.

5 Conclusion and Future Work

In this paper, we have assessed the use of the Decision Model and Notation (DMN) and the Knowledge Base (KB) paradigm in the form of IDP through two case studies. Takeaways on the modelling and the use of both DMN and IDP were inferred from the case studies: (1) Income taxation for foreign artists temporarily working in Belgium; and (2) Registration duties when purchasing real-estate in Belgium. Based on these case studies, the advantages, disadvantages, as well as synergies of DMN and IDP were discussed. The analysis shows that both DMN and IDP are very suitable when it comes to solving decision-dependent problems and that DMN and IDP are often similar and compatible with each other. However, some dissimilarities were identified as well, as DMN and IDP are not always usable for the same kind of problem solving.

In future work, we will analyse to which extent the existing DMN tools adhere to the DMN standard by assessing their level of support of DMN features and concepts relating to the decision requirements diagram, decision logic specifications and the (S)FEEL expression language. Additionally, we will look into the automatised transformation of DMN decision tables into IDP rules. Finally, we will focus on the complexity of DMN decision models in relation to their granularity, i.e. a trade off between large DRD models accompanied by smaller decision tables versus simpler DRD models with larger and more complex decision tables at their core.

Acknowledgements. We would like to thank Avola for giving us access to their decision management tool and the following case study participants for their assistance with the analysis, interpretation and completion of the case studies: the Belgian industry federation for the arts (oKo), Kunstenloket, and notary Van Pelt.

References

1. Trunomi: Gdpr portal. `https://www.eugdpr.org/`
2. 107th Congress: Public law 107-204. `https://www.gpo.gov/fdsys/pkg/PLAW-107publ204/html/PLAW-107publ204.htm`
3. Commission, E.: General product safety directive. `https://ec.europa.eu/info/general-product-safety-directive_en`
4. Mitra, S., Chittimalli, P.K.: A systematic review of methods for consistency checking in sbvr-based business rules. In: CEUR Workshop Proceedings. Volume 1819., CEUR-WS.org (2017)

5. Nelson, M.L., Peterson, J., Rariden, R.L., Sen, R.: Transitioning to a business rule management service model: Case studies from the property and casualty insurance industry. Information & Management **47**(1) (2010) 30 – 41
6. OMG: Decision Model and Notation 1.1 (2016)
7. Denecker, M., Vennekens, J.: Building a knowledge base system for an integration of logic programming and classical logic. Volume 5366., Springer (2008) 71–76
8. Hasić, F., Devadder, L., Dochez, M., Hanot, J., De Smedt, J., Vanthienen, J.: Challenges in refactoring processes to include decision modelling. In: Business Process Management Workshops. LNBIP, Springer (2017)
9. Dasseville, I., Janssens, L., Janssens, G., Vanthienen, J., Denecker, M.: Combining dmn and the knowledge base paradigm for flexible decision enactment. In: RuleML 2016 Supplementary Proceedings. (2016)
10. De Smedt, J., Hasić, F., Vanthienen, J.: Towards a holistic discovery of decisions in process-aware information systems. In: Business Process Management. Lecture Notes in Business Information Processing, Springer (2017)
11. Horita, F.E., de Albuquerque, J.P., Marchezini, V., Mendiondo, E.M.: Bridging the gap between decision-making and emerging big data sources: an application of a model-based framework to disaster management in brazil. Decision Support Systems **97** (2017) 12–22
12. Biard, T., Le Mauff, A., Bigand, M., Bourey, J.P.: Separation of decision modeling from business process modeling using new decision model and notation(dmn) for automating operational decision-making. In: Working Conference on Virtual Enterprises, Springer (2015) 489–496
13. Hu, J., Aghakhani, G., Hasić, F., Serral, E.: An evaluation framework for design-time context-adaptation of process modelling languages. In: Practice of Enterprise Modelling (PoEM), Springer (2017)
14. Hasić, F., De Smedt, J., Vanthienen, J.: Augmenting processes with decision intelligence: Principles for integrated modelling. Decision Support Systems **107** (2018) 1 – 12
15. Hasić, F., De Smedt, J., Vanthienen, J.: An Illustration of Five Principles for Integrated Process and Decision Modelling (5PDM). Technical report, KU Leuven (2017)
16. Boumahdi, F., Chalal, R., Guendouz, A., Gasmia, K.: Soaˆ\mathrm {+ d}: a new way to design the decision in soabased on the new standard decision model and notation (dmn). Service Oriented Computing and Applications **10**(1) (2016) 35–53
17. Hasić, F., De Smedt, J., Vanthienen, J.: A service-oriented architecture design of decision-aware information systems: Decision as a service. In: OTM Confederated International Conferences "On the Move to Meaningful Internet Systems", Springer (2017) 353–361
18. de Cat, B., Bogaerts, B., Bruynooghe, M., Denecker, M.: Predicate logic as a modelling language: The IDP system. CoRR **abs/1401.6312** (2014)
19. Vanthienen, J., Dries, E.: Developments in decision tables: evolution, applications and a proposed standard. DTEW Research Report 9227 (1992) 1–40
20. Silver, B.: Dmn method and style., USA, Cody-Cassidy Press (2016) 305
21. OMG: Business process model and notation (BPMN) 2.0 (2011)
22. Vanthienen, J., Dries, E.: Illustration of a decision table tool for specifying and implementing knowledge based systems. International Journal on Artificial Intelligence Tools **3**(2) (1994) 267–288
23. Calvanese, D., Dumas, M., Laurson, Ü., Maggi, F.M., Montali, M., Teinemaa, I.: Semantics, analysis and simplification of dmn decision tables. Information Systems (2018)