# Extending and Automating Maturity Models for More Value

## Domain Model Driven Generic Support for Maturity Models

Graham McLeod*,a

[a] Inspired.org | University Duisberg-Essen

Abstract. *Maturity Models are widely used to quickly assess the status of a capability within organisations and to allow comparison across organisations and over time. The use of maturity models may be discouraged by friction of collecting the information, calculating ratings and managing the information over time. Value delivered is limited if only a rating results. Much higher value can be achieved if the models are extended to provide recommended actions and further still if such actions can be filtered by relative effort and sequenced to respect dependencies or resource constraints. Automation can remove friction and encourage use. This paper documents the creation of a generic meta model (domain model) for maturity models extended to support recommendations and their filtering and prioritisation. The meta model use is validated in a case showing rapid automation in a SaaS Enterprise Modeling platform. The paper concludes with considering the Return on Modelling Effort of this approach.*

Keywords. Maturity Model • Domain Model • Automation • Return on Modelling Effort • Low Code

## 1 Introduction

### 1.1 Maturity Models

Maturity models are widely used to quickly assess the status of a capability within organisations and to allow comparison across organisations and over time. Examples include the software development Capability Maturity Model [Paulk 2009] from Carnegie Mellon University, the Architecture Capability Maturity Model from the Department of Commerce (USA) [Doc 2003], The Portfolio, Programme and Project Management Maturity Model of the Office of Government Commerce [Sowden et al. 2006] and the Orange Data Management Maturity Model [Steenbeek 2019]. Indeed, [Poeppelbuss et al. 2011 ] reported on 76 maturity models used in information systems, information technology and related disciplines.

These models are generally found useful by practitioners and academics as they offer a diagnostic quickly and with little effort. The theoretical foundations are not universally sound and some models have been criticised for being simplistic. Nevertheless, they are in widespread use.

Our literature search revealed little in the way of a domain or meta model to support these kinds of models, although [Bley et al. 2020] proposed a model capable of representing typical models. We will discuss some limitations of this later.

### 1.2 The Development Organisation

Inspired.org, the organisation examined in the case, is a boutique consultancy operating in strategy, enterprise architecture and the provision of advanced methods for business, enterprise, application, information and solution architecture. It provides intellectual property (e.g. frameworks, models, training materials, methods, techniques), training and supporting software to a variety of corporate clients.

### 1.3 The Problem Addressed

The organisation has used maturity models extensively in consulting work for a number of

* Corresponding author.
E-mail. graham@inspired.org

years. Models used include *inter alia* the Carnegie-Mellon Capability Maturity Model (CM-CMM) and the Dept of Commerce Enterprise Architecture Maturity Model. The organisation has also developed its own unique models for Application Architecture Landscape Maturity and Data Management Maturity. Until recently, these models were supported by a range of presentation materials, documents and spreadsheets for gathering the data and sometimes calculating scores.

At the onset of the COVID pandemic, the leader of the organisation and colleagues developed a Pandemic Readiness Model with the dual aims of providing guidance and value to clients, prospects and the community, as well as providing useful exposure for the organisation.

There was a desire to provide the model online and on the Internet, so that potential users could easily discover it, use it and get the results immediately. Following earlier experience in devising their own models and providing guidance based upon these, there was also an intent to include advice and guidance in the results. The organisation wanted to exploit their own software platform (introduced in the next section) to support the automation. This was for reasons of economy of development effort, support and integration of models with other aspects of client architectures.

### 1.4 Available Tool Set

The organisation provides a repository backed, web user interface and highly configurable tool set dubbed Enterprise Value Architect (EVA) [Inspired.org 2022] for the support of *inter alia* strategy, business architecture, enterprise architecture, information architecture, process architecture/-modelling, application architecture/portfolio management, programme management and related disciplines. The tool set allows customisation at several levels, including:

1. Meta Model - This can be easily defined or extended by users via provided interfaces or graphical modelling facilities. Extensions are immediately active and customise provided user interfaces so that relevant data and relationships can be maintained and reports generated. [McLeod 2001]

2. User Interface - The tool allows the development of custom views which are developed in a combination of HTML, CSS and embedded Smalltalk [Goldberg and Robson 1983] code that allows accessing full facilities within the tool. Custom views can be defined at runtime and immediately deployed. They can be attached to user menus for ease of use. Views can initiate subsequent views

3. Calculated Properties - The tool supports calculated properties as part of type definitions within meta models. Any kind of calculation can be supported and these can reference properties of the item on which they are contained, properties of related items, or items elsewhere in the repository via type/item/property ids

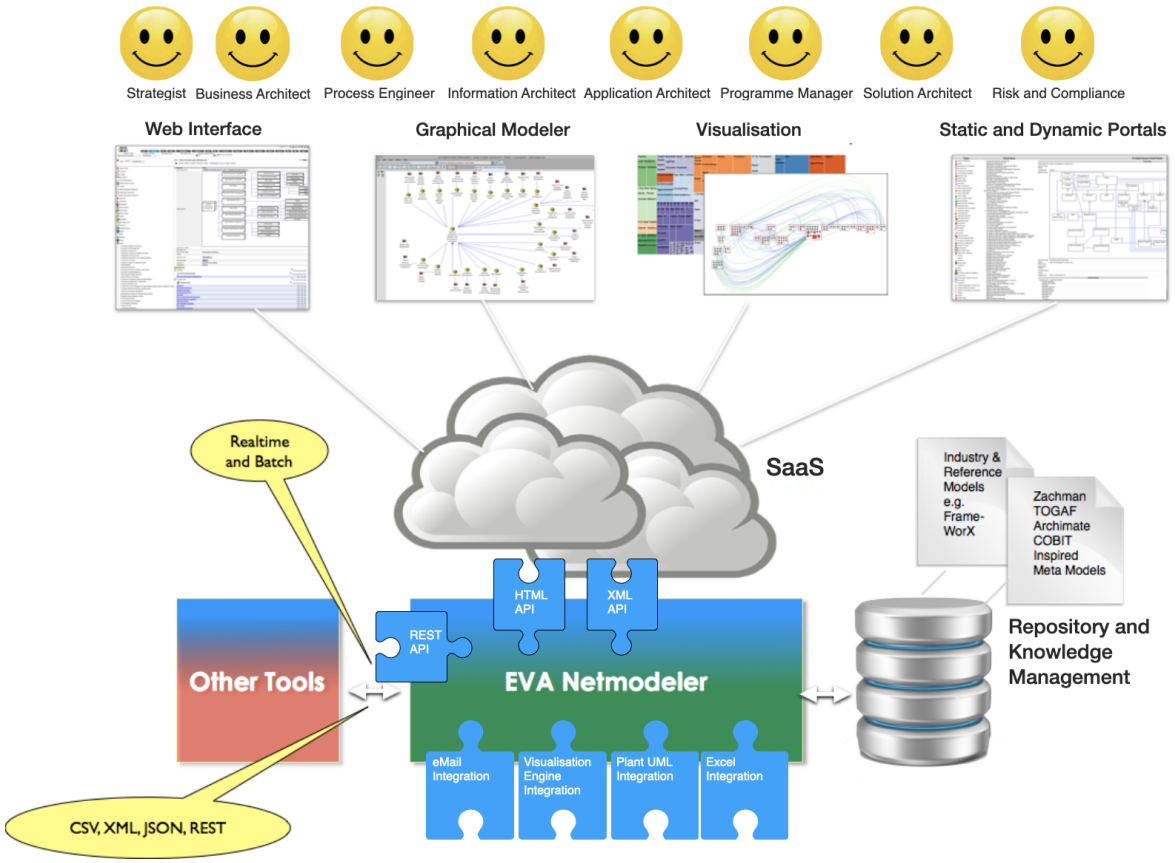
4. APIs[1] - It is very easy to create custom APIs. These are held as code in the repository. Again, API definition can occur at runtime. APIs can then be invoked in the typical fashion by a client application performing a log on to the application (thereby obtaining security credentials) and then using a REST[2] style URL[3] . Data can be returned in any required format, typically JSON[4] , CSV[5] , XML[6] or HTML[7]

Other features of the tool which provided useful facilities include:

1. HTML text fields - These store formatted text of any length which allows holding descriptions, questions and recommendations. They can also hold generated output, such as a results report

[1] Application Programming Interfaces
[2] Resource, State, Transfer
[3] Uniform Resource Locator
[4] Javascript Object Notation
[5] Comma Separated Variable
[6] eXtensible Markup Language
[7] Hypertext Markup Language

*Figure 1: Enterprise Value Architect (EVA) Architecture*




2. Picture fields - These allow storing web or print friendly pictures (e.g. .jpeg, .png, .gif). These may be used to store images for explanation or branding purposes

3. Hyperlink fields - These allow storing references to other pages, such as help text or sources of assistance (such as a consulting service or training offering)

4. Graph and visualisation plotting facilities (provided by an embedded Pharo [Open_Source 2022b] Roassal [Bergel 2022] server). This allows definition of a variety of graphs and visualisations which can be returned as images included in web output or stored in Picture fields (2)

5. A comprehensive security model allowing fine grained control over users, groups, roles, access and update permissions

6. Dynamic language and live programming environment. The tool is written in VA Smalltalk [Instantiations 2022] and exploits the dynamic typing and live programming capabilities unique to Smalltalk to good advantage. It allows code for calculated properties, custom user interfaces and APIs to be edited through the web interface and held in the repository. This code can then be executed without the need for a compile / link / deploy / launch cycle. This vastly speeds up development

7. Output Formats - The tool already supports the generation of output in various languages including: HTML/CSS; XML, CSV and JSON

## 2 Requirements

A number of requirements for a solution to support Maturity Models were identified in discussion with senior consultants within the organisation and partner organisations. These were augmented by a study of the documentation of the Maturity Models which the organisation has developed and/or uses in practice. Requirements included:

R1: The system must be able to handle a wide variety of maturity models without code change - i.e. Model properties such as concerns, maturity levels, questions, recommendations etc. should be held as user editable data in the repository

R2: It should provide support for assessment, scoring, graphing results, storing assessments

R3: Client security and privacy should be respected

R4: Scoring should allow inclusion of a radar/Kiviat[8] graph similar to Figure 3

R5: Models may include recommendations, which may have relative effort and dependencies

The above requirements were identified early on and formed a baseline. Other requirements were added as we gained experience and as we leveraged the early work to start providing the platform to commercial partners to support their own models. Further requirements included :

R6: Ability to invoke the assessment for a given model using a URL which identifies the model without requiring a login (to provide "service" models anonymously - e.g. the Pandemic Readiness Model)

R7: Ability to invoke a given model providing a unique but anonymous client identity. This identity to be used to store the assessment. An API to be provided to allow retrieval of the list of models for the client id or a given model using the id and date/time. This was necessary to meet privacy legislation and to separate management of client information (by the partner) and model execution (by our solution)

R8: Provision of summary results vs full results (to allow model providers to encourage client interaction with them before providing full results)

R9: Provide assessment scores and results data via an API to allow model providers to retrieve details for their clients and use the data to format their own custom reports (e.g. in Excel, PowerPoint or .pdf)

[8] https://en.wikipedia.org/wiki/Radar_chart

## 3 Activities and Effort

### 3.1 Gathering Representative Models

We began by identifying models typical of the range we wanted to support. These included:

1. The CMMI Maturity Model for software
2. The Data Management Maturity Model developed by Inspired based on earlier models by Data Crossroads [Steenbeek 2019] and DAMA [Cupoli et al. 2014]
3. The Application Landscape Maturity Model developed by Inspired
4. The Pandemic Readiness Model developed by Inspired
5. The Dept of Commerce Architecture Capability Maturity Model
6. The Business Architecture Guild model for Business Architecture Maturity
7. A model developed jointly by Inspired and BrightHouse for Digital Readiness

Most of these models were in use in the organisation, so effort was low, mainly just collating and reformatting.

*Table 1: Generic Structure of Maturity Models*

| | Maturity Level | | |
|---|---|---|---|
| **Concern** | Ad Hoc | Initial | Developing |
| Concern 1 | Statement A | Statement B | Statement C |
| Concern 2 | Statement D | Statement E | Statement F |

### 3.2 Drafting a Domain Model

Most of the models had a tabular presentation format, the exception being the Business Architecture Guild model. The generic structure of these is represented in Table 1.

The models as tables are used during assessment to move down row by row, then across column by column posing the statement in the cells in turn. Moving across a row, the user selects the statement which best resonates with the situation in the organisation being assessed.

The cells chosen can be translated directly to a score, or may be used as input to a more complex scoring algorithm. In particular, many models will take the selections from several rows and combine them into a scoring dimension, sometimes with a weighting for the row. Thus there could be several rows exploring aspects of a topic which then lead to a single score for a dimension of interest.

A meta model/domain model was developed using Inspired's recommended practice for domain/meta modelling, which is a conceptual modelling approach broadly analogous to UML Static Structure modelling. The process includes identifying key concepts, identity properties for these, relationships between concepts (named semantically in both directions) and then properties of interest to detail instances of the concepts. These properties can make use of the rich data types previously documented (e.g. HTML, Hyperlink, Picture, Computed Value etc. ). The tool also automatically adds a unique internal identity as well as audit properties for date and user last modifying an instance. Default relationships are also present to manage security.

Core concepts and *relationships* in the meta model (see Figure 2) include:

1. `Maturity Model` which defines a kind of maturity model. It *includes* `Maturity Model Columns` and `Maturity Model Concerns` (rows)

2. `Maturity Model Question` which holds data relevant for a cell in the table (the intersection of a column and a row)

3. `Maturity Score Dimension` which holds the formula for the scoring calculation. A score allows combining the values for selected questions on selected concerns

4. `Maturity Guidance` items hold recommendations relevant to moving from one cell to the next higher level of maturity within a concern/row. The *precedes / follows* relationship allows capture of dependencies

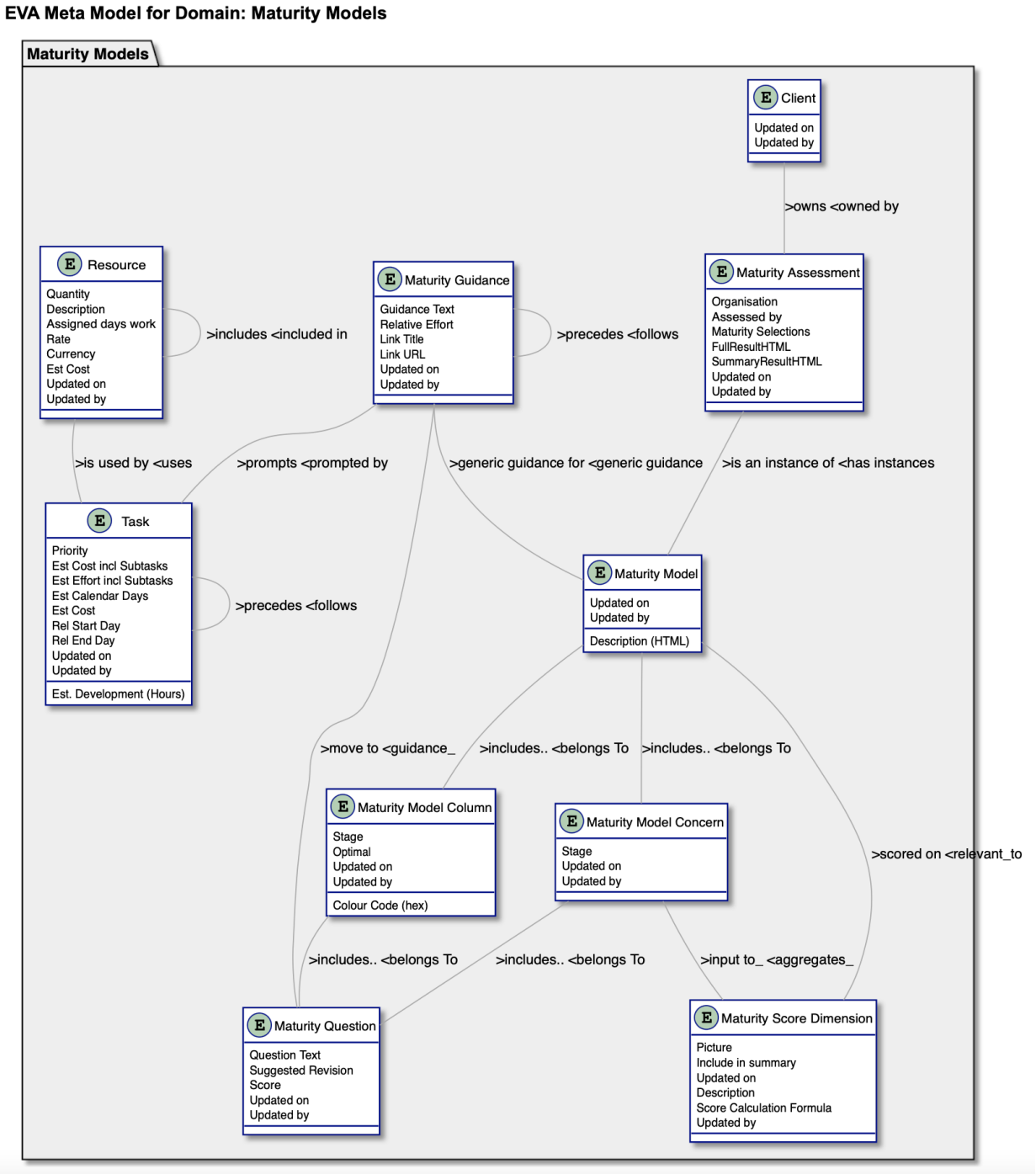
5. `Maturity Assessment` holds the chosen questions for applying a `Maturity Model` for a given `Client` on an occasion. Following the assessment it also holds the results in a summary and full report form

6. `Client` represents an organisation for whom a `Maturity Assessment` is performed

7. `Task` represents an action the client organisation chooses to take, indicated by selecting from the presented priority recommendations. The *precedes / follows* relationship allows capture of dependencies

8. `Resource` indicates resources (people or other) that will be allocated to a Task to assist in completing it

*Figure 2: Domain Model for Maturity as EVA Meta Model*




The domain model did not include cardinalities, but does name relationships explicitly in both directions. Following recommended practice, the model includes all relevant concepts, relationships and required properties, but does not include any technology specific features or behaviours/methods.

Note that in the model diagram descriptions are

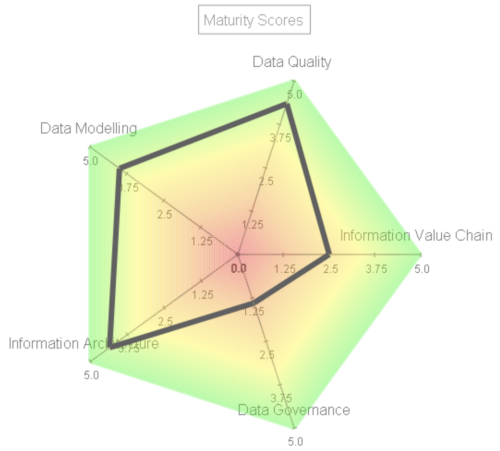
not necessary as properties, since the tool allows long descriptive names for instances by default. Unique internal identities are automatically provided for all instances as well. The Updated On and Updated by properties are also automatic and support audit trail features.

Relationships are labeled semantically (and in both directions) between the concepts.

### 3.3 Scoring

Scores can be usefully illustrated via a radar chart/Kiviat diagram with spokes for each of the scoring dimensions. This gives a visual impression of areas where the maturity is high or low. Such a diagram should be able to cope with a variable number of dimensions and different maximum scores for a dimension.

*Figure 3: Example score Kiviat Chart*




### 3.4 Recommendations

To cater for recommendations, we needed to add these to the model. Conceptually, they link between the chosen cell for a row and the following one in the same row. E.g. if we are at level "ad hoc" and the next level is "Initial" for the row related to "Method", the organisation should be trying to move from "ad hoc" to "Initial". The recommendations might include: "Select, define or adapt a suitable method". "Create a training programme and identify candidates". These recommendations would be linked from the Ad hoc cell on that row to the Initial cell on the same row.

We also provided for recommendations that were derived from the overall scores that could apply to the whole model. A recommendation could apply to multiple places on the model, and multiple recommendations could be necessary between two adjacent cells. Recommendations were provided with properties to record relative effort and a hyperlink which could point to resources for assistance.

Client details would be recorded and assessments would be grouped by client to allow comparison over time or convenient retrieval.

### 3.5 Capture Domain Model in EVA Tool

This is a familiar activity for the organisation in applying the tool to support various applications. The domain model is translated directly to a meta model within the tool. Each concept in the domain model is defined as a type in the tool; each relationship as a legal relationship and each property as a legal property on the relevant type.

The meta model was easily captured through the interactive web Type Browser interface in about two hours. The resulting model is included as Figure 2. The diagram is generated from a utility in the tool which leverages a PlantUML [Open_Source 2022a] server.

### 3.6 Capturing Initial Maturity Model

Once the meta model was captured, we proceeded to capture the Pandemic Model as an example model. This was done using the standard web form interfaces provided by the toolset. These are automatically customised by the meta model and provide facilities for creating items, capturing property values and establishing required relationships. For early models this was achieved manually.

As the number and size of models grew, we concluded that loading a model could be made easier by providing a capability to import models from a suitably structured spreadsheet. We defined a spreadsheet format with additional columns to

hold recommendations and scores per question, as well as with sufficient meta data in column and row names to recreate the model in the repository. We also wrote a custom view to facilitate importation. Importation can now be achieved for models captured in spreadsheets using the following process:

1. The spreadsheet is saved as CSV
2. Using a standard provided import facility, the CSV input is imported into the tool, creating a composite item per row
3. The custom view mentioned unpacks these items and, using the imported meta data cues, creates all the necessary items in the repository conformant with the meta model

### 3.7 User Interfaces for Assessment and Results

The overall flow of the online application comprises selecting the required model type, presenting the model, gathering the assessment responses, presenting the recommendations, capturing selected activities and adding resources. This is followed by generation of data for a project management graph (Project Schedule Bar Chart or similar). The overall flow is illustrated in Figure 4.

A custom view was created to present models for assessment. This generates a tabular view in the browser and accepts user selections. The choices are stored in the repository and fed to the next stage which calculates scores for the dimensions, invokes the graphing capability and then generates a summary report including the graph, a score table and a recommendation table.

### 3.8 Scoring Mechanisms

To facilitate flexibility of scoring mechanisms across the variety of models, we used a property on the Score Dimension to hold code implementing a scoring formula. The view mentioned earlier passes in selections made and their score as a dictionary. This was initially keyed by the cell text, but later changed to a more robust short reference.

Model designers can include any calculation required (e.g. summing, averaging, applying weights, combining scores from various concern rows) as they see fit. This is defined as a simple formula in Smalltalk, within a block which receives the dictionary as a parameter and returns the desired score for the dimension. An example is shown as Figure 5.

### 3.9 Providing Recommendations

Recommendations are derived from the position of the user selections during the assessment. They effectively connect the selected cell with the next one to its right (the next higher state of maturity on that concern). Recommendations are first returned in a tabular form showing the current state and the next maturity state, together with the recommended actions to undertake to move from the former to the latter. The table also provides for a hyperlink to direct the user to assistance (e.g. documents, web pages, training or consulting assistance). Recommendations are then processed through an algorithm which:

1. Considers the relative effort of the recommendations
2. Prioritises the recommendations which address those concerns with the lowest scores and which have relatively low effort
3. Boosts the priority of those recommendations that address multiple concerns
4. Respects dependency between recommendations if these are present in the model
5. Trims the list so that priority items can be presented without overwhelming the model user

An example of the recommendations output is included as Figure 6.

*Figure 4: Online Assessment Flow*

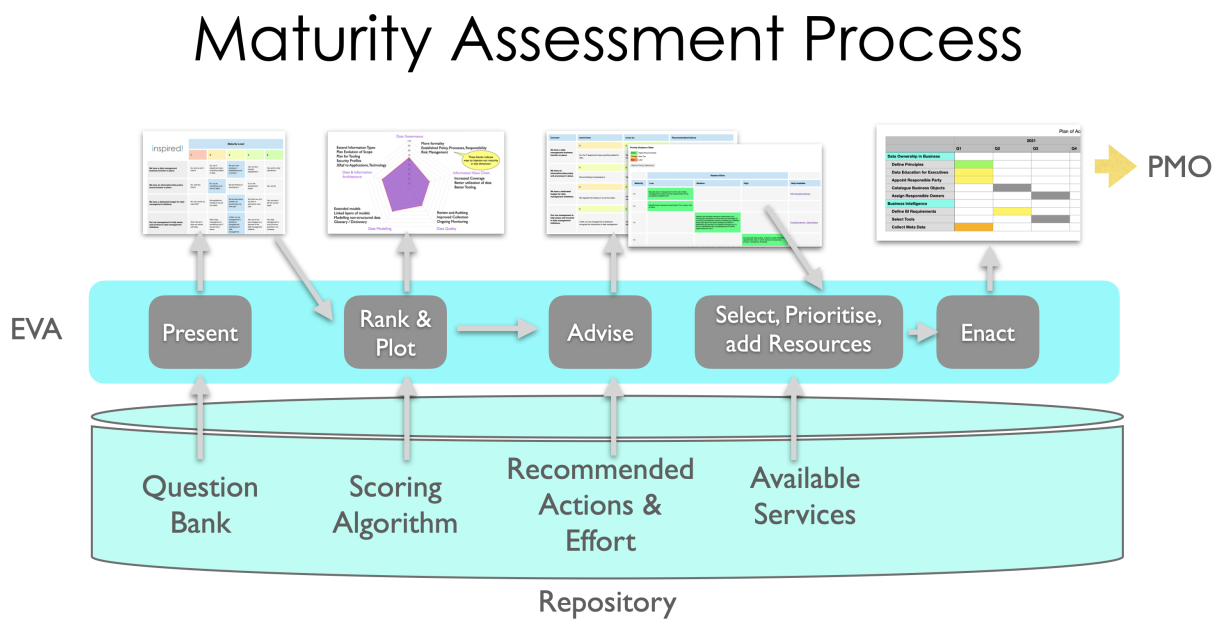



*Figure 5: Score Dimension Formula Example*

| | |
|---|---|
| **Score Calculation Formula** | **(dict at: '06') + (dict at: '07') + (dict at: '08') + (dict at: '10') + (dict at: '14') + (dict at: '29') + (dict at: '31') + (dict at: '32') + (dict at: '35')** |

### 3.10 Extending into Action

If the user desires, they are then encouraged to select recommendations from the priority result table which they would like to take action on. Once these are selected, the user can proceed to provide dates and resources to the selected items. Following this, a CSV data file is generated, suitable for creating a Gantt chart in a spreadsheet or project management package. It is also possible to generate Task items within EVA which can drive a Programme Milestone Chart for tracking purposes.

### 3.11 Validation in Use

The Pandemic model was implemented quickly and online within a week. It proved popular and useful to quite a wide global audience, while also serving a useful marketing purpose for Inspired. The platform was also soon used to support the internal Data Management Maturity model which was used with clients. This was supported by consulting facilitation and lead through. Responses were positive. Demonstration of this model lead to us being contacted by several parties interested to leverage the platform for their own purposes (using an Inspired model or their own models). The first was an international buisness/accounting practice. We worked with them to support several APIs to facilitate this.

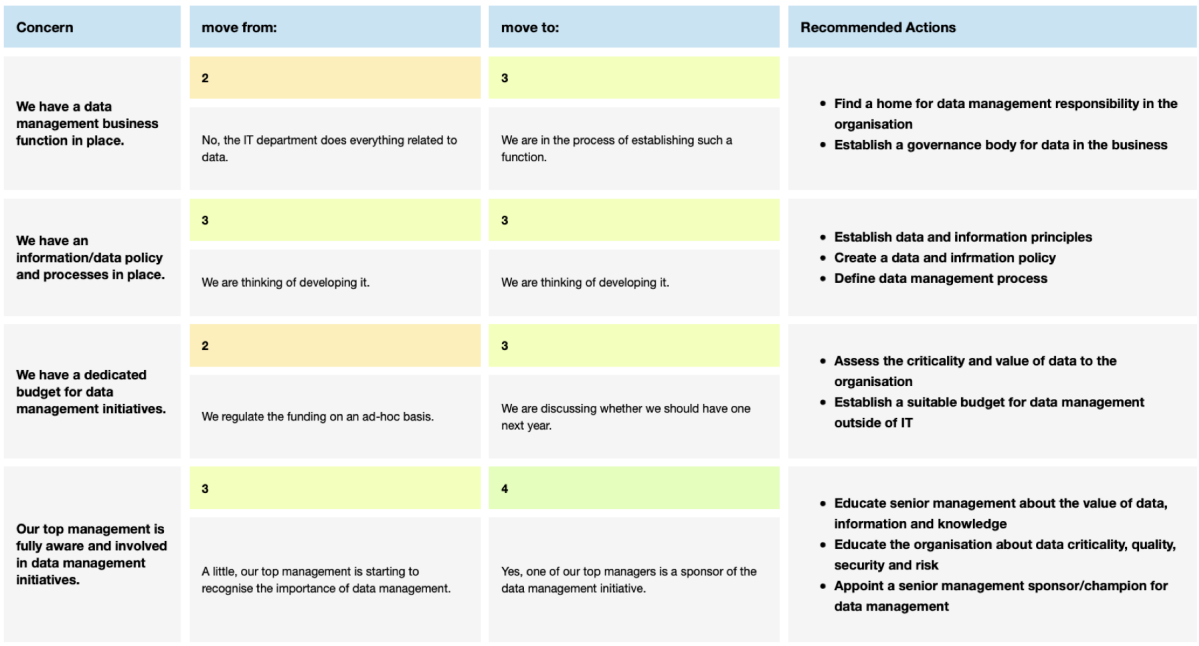
### 3.12 Adding APIs for Client Use

APIs added to the platform included:

1. Ability to launch a specified model and provide an anonymous client id. The id is used with a date and time stamp to store the assessment. Results can be returned immediately or retrieved in future

2. Ability to provide a client id and receive a list of assessments done for that client

*Figure 6: Presenting Recommendations*



| Concern | move from: | move to: | Recommended Actions |
|---|---|---|---|
| We have a data management business function in place. | 2<br>No, the IT department does everything related to data. | 3<br>We are in the process of establishing such a function. | • Find a home for data management responsibility in the organisation<br>• Establish a governance body for data in the business |
| We have an information/data policy and processes in place. | 3<br>We are thinking of developing it. | 3<br>We are thinking of developing it. | • Establish data and information principles<br>• Create a data and infrmation policy<br>• Define data management process |
| We have a dedicated budget for data management initiatives. | 2<br>We regulate the funding on an ad-hoc basis. | 3<br>We are discussing whether we should have one next year. | • Assess the criticality and value of data to the organisation<br>• Establish a suitable budget for data management outside of IT |
| Our top management is fully aware and involved in data management initiatives. | 3<br>A little, our top management is starting to recognise the importance of data management. | 4<br>Yes, one of our top managers is a sponsor of the data management initiative. | • Educate senior management about the value of data, information and knowledge<br>• Educate the organisation about data criticality, quality, security and risk<br>• Appoint a senior management sponsor/champion for data management |

3. Ability to retrieve a given assessment and results for a provided client id and date time stamp

4. Ability to retrieve assessment results as above, but as a data stream rather than rendered in the browser. This to facilitate creation of unique reports or analyses by the model provider on behalf of their client. The initial version of this can return either CSV or JSON

These APIs have taken approximately three days of effort to implement.

## 4 Return on Modelling Effort

The Return on Modelling Effort (ROME) has been very high. With the ability of the tool set to rapidly define a meta model (domain model), immediately provide interfaces for data capture, maintenance and reporting, it is easy to conceive a solution and move from idea to prototype very quickly. The ability to further define customised user interfaces, output results and APIs based upon the domain model and working at the level of the business concepts is very powerful.

The domain model was defined at a conceptual model level of abstraction. This has allowed a variety of maturity model types to be rapidly and successfully captured, presented, assessed, scored and reported on with considerable flexibility. It has allowed the rapid construction of intuitive and attractive user interfaces, graphs, reports and summaries. It has facilitated rapid delivery of APIs for tool integration with systems in partner organisations. It has also allowed quick and relatively easy implementation of an alternate capture mechanism and reprsentation in spreadsheets.

A summary of code lines in the initial facilities is given as Table 2. Additional features have been added since this analysis was done.

## 5 Reflection

We can highly recommend the practice of developing a conceptual model for a domain under consideration for automation as well as the use of suitable tools that:

1. Allow the capture of the domain model at a conceptual or semantic level

*Table 2: Feature Code Size*

| Feature | Language | | | |
|---|---|---|---|---|
| | CSS | HTML | JavaScript | Smalltalk |
| Present Questionnaire | 110 | Gen by ST | 44 | 43 |
| Generate Score Graph | 0 | 0 | 0 | 60 |
| Present Results | 160 | Gen by ST | 50 | 230 |
| Create Plan | 24 | Gen by ST | 33 | 31 |
| Import Model | 0 | 0 | 0 | 347 |
| Return Data API | 0 | 0 | 0 | 207 |

2. Facilitate the easy capture, relating, reporting and maintenance of information conformant with the domain model

3. Permit the definition of domain concept driven user interfaces, outputs and APIs

Working at a domain level greatly facilitates prototyping, rapid evolution and agility in adaptation to new requirements. Tools that operate in this manner will typically not offer equivalent performance and scalability to ones which are tailor made for a given application. They may thus not be suitable for high volume, performance critical applications, unless the domain models are used to generate efficient tailored code, but this is certainly possible.

## References

Bergel A. (2022) Agile Visualization In: Agile Visualization with Pharo: Crafting Interactive Visual Support Using Roassal Apress, Berkeley, CA, pp. 41–47 https://doi.org/10.1007/978-1-4842-7161-2_4

Bley K., Schön H., Strahringer S. (2020) Overcoming the ivory tower: a meta model for staged maturity models. In: Conference on e-Business, e-Services and e-Society. Springer, pp. 337–349

Cupoli P., Earley S., Henderson D. (2014) Dama-dmbok2 framework. In: Dama International

Doc I. (2003) Architecture Capability Maturity Model. In: Department of Commerce, USA Government Introduction

Goldberg A., Robson D. (1983) Smalltalk-80: the language and its implementation. Addison-Wesley Longman Publishing Co., Inc.

Inspired.org (2022) Enterprise Value Architect https://www.inspired.org/eva-home Last Access: 2022-05-02

Instantiations (2022) VAST Platform 2022 Documentation https://www.instantiations.com/vast-support/documentation/1100/ Last Access: 2022-05-02

McLeod G. (2001) PAMELA: A Proto-pattern for Rapidly Deliverd, Runtime Extensible Systems. In: International Workshop on Evaluation of Modeling Methods in Systems Analysis and Design (EMMSAD) In conjunction with CAiSE'01

Open_Source (2022a) Drawing UML with PlantUML PlantUML Language Reference Guide https://plantuml.com/guide Last Access: 2022-05-02

Open_Source (2022b) The Pharo Environment Documentation https://www.pharo.org Last Access: 2022-05-02

Paulk M. C. (2009) A history of the capability maturity model for software. In: ASQ Software Quality Professional 12(1), pp. 5–19

Poeppelbuss J., Niehaves B., Simons A., Becker J. (2011) Maturity models in information systems research: literature search and analysis. In: Communications of the Association for Information Systems 29(1), p. 27

Sowden R., Hinley D., Clarke S. (2006) Portfolio, Programme and Project Management Maturity Model: P3M3 Public Consultation Draft v2.0. Office of the Government Commerce

Steenbeek I. (2019) The “ORANGE” Model of Data Management. Data Crossroads