

FINANCIAL MODELLING: TOPICAL ARTICLES, TUTORIALS, CASE STUDIES AND NEWS

WELCOME

We are pleased to open this issue with an article on the RiskAgility™ Economic Capital Aggregator (RiskAgility™ EC). Armed with tools such as RiskAgility™ EC, insurers will be better prepared to meet the evolving requirements of regulators, rating agencies and investors.

We continue this issue with an article covering the latest release of Towers Perrin's C-ALM suite of MoSes applications. We also include a follow-up to our tutorial article from the last issue covering an introduction to submodels. This time we take a look at cloned submodels and output groupings.

We finish with the usual Q&As and the Back Page packed with interesting news topics from the world of financial modelling.

WHAT IS ECONOMIC CAPITAL?

Economic capital provides a realistic economic measure of the amount of capital that a firm needs to hold to cover losses at a certain risk tolerance level. A growing number of insurers use economic capital as an important metric for quantifying risk within their financial and strategic decision making.

Economic capital captures the myriad of risks, dependencies and complexities to which an insurer is exposed, and expresses these as a single understandable number.

RISKAGILITY™ ECONOMIC CAPITAL AGGREGATOR

A RiskAgility application for embedding economic capital into an insurer's decision making.

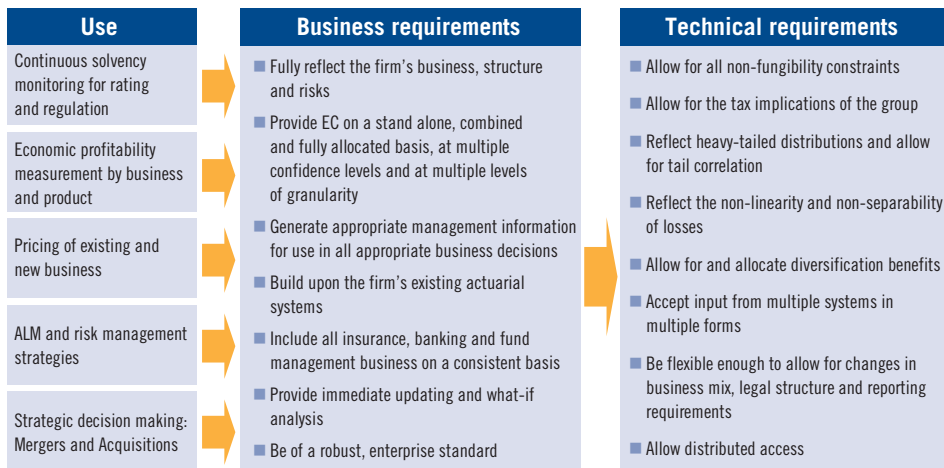
The Importance of economic capital

Economic capital is gaining momentum. It is at the heart of the Solvency II framework in Europe, a number of insurance regulators have adopted it in advance and rating agencies are increasingly accepting it as a measure of solvency. Consequently, there are real capital advantages to having an effective economic

capital model and many insurance firms are now rushing to implement it.

At its most powerful, economic capital can assist management to decide whether to enter a particular market and, if so, the nature and volume of business they can write. They will be able to understand how each risk diversifies against other risks, whether the business is profitable on a risk-adjusted basis and the mix of business that will optimise that profitability. Furthermore, it enables the effective assessment of alternative risk management and hedging strategies.

FIGURE 1 – What do insurers require from an economic capital system?



IN THIS ISSUE

RiskAgility™ Economic Capital Aggregator	1
Latest release of the Towers Perrin C-ALM application now available	4
In-depth tutorial: Further aspects of submodels	5
Q&As: Common questions on submodels	8
The Back Page	9

RiskAgility EC calculates and aggregates a group's economic capital from the marginal risk distributions, loss functions and correlation structure. It allocates capital by risk type at multiple levels of granularity, allowing for the group's legal structure.

Regulation and economic capital

Economic capital underpins the drive towards risk-based regulation. It has given rise to the UK's Individual Capital Assessment, and is enshrined Europe-wide in Solvency II via the internal capital model option: essentially a firm's calculation of the capital needed to withstand a 1-in-200 year event over a one year time horizon. Economic capital principles also underlie the approach to the standard Solvency Capital Requirement under Solvency II.

Current implementations face daunting practical and theoretical problems

At present, most firms are geared up to calculate economic capital only once a year. This is an unacceptable frequency, either for effective risk management or for risk-based decision making. The technical complexities involved in modelling a firm's business structure, and the panoply of risks it faces, stretch systems to their limits. While the Solvency II 'Standard Approach' represents a more easily achievable calculation, it frequently fails to provide sufficient accuracy or granularity for taking business decisions.

Existing methodologies represent economic capital theory in a state of evolution. Firms' current models are often based on unempirical thin-tailed risk distributions and use simplistic assumptions of linear correlation. Indeed, 'risk management's failure to consider extreme exposures' was cited by the International Association of Insurance Supervisors as a lesson to be learned from the financial crisis of 2008. In essence, fundamental features of insurance risks have been assumed away for mathematical convenience.

The RiskAgility EC solution

Robust answers to the above problems will result in materially different financial decision making. The RiskAgility Economic Capital Aggregator ('RiskAgility EC') is not driven by a desire to be mathematically elaborate; rather, it highlights the unwitting assumptions implicit in simpler economic capital models, and puts business decision making on a firm footing. When data is scarce and risks poorly understood, management must not seek false security in easy assumptions. What is needed, and what RiskAgility EC provides, is a rigorous framework for understanding how what we assume affects the capital we must hold.

Features of the RiskAgility EC solution include:

■ Capturing the company structure:

RiskAgility EC models the full group structure of a firm, along with the associated fungibility constraints, tax and reinsurance. It allocates capital by risk type across life, non-life and asset management business units, allowing for diversification effects.

■ **Updating on demand:** In order to be used in the decision-making process and pass the Solvency II use test, an internal model must be able to produce results, sensitivities and updates in timescales appropriate to those decisions: close to real time for market risk management. RiskAgility EC has functionality to support rapid updating and analysis without the need to re-run underlying ALM/DFA models, moving the calculation of economic capital from an annual to a monthly or daily activity (where needed).

■ **Reader-friendly reporting:** RiskAgility EC's interactive reports produce available capital and required capital measures on both a stand-alone and fully diversified basis, allocating diversification benefits back to their sources. It supports full drill-down capabilities, so results can be as granular as the underlying inputs.

■ **Expanding the extremes:** For mathematical ease, traditional models have assumed thin-tailed normal distributions, where the probability of extreme events tails off exponentially. RiskAgility EC recognises that, in insurance and finance, extreme events occur more frequently than this, and supports the use of fat-tailed distributions. In addition, it allows Value-at-Risk to be calculated simultaneously at multiple confidence levels, thereby preventing catastrophes from being 'hidden in the tail'.

■ **Going beyond linear correlation:** Use of linear correlation coefficients does not permit the observable characteristics of extreme scenarios to be appropriately simulated. In these extremes, otherwise uncorrelated risks tend increasingly to occur together. RiskAgility EC captures this feature using a copula approach.

■ **Reflecting non-linear losses:** Via stress tests on firms' underlying models, RiskAgility EC describes in full the non-linear dependence of losses on risk variables. Furthermore, it recognises the non-separability of risks, namely that their combined impact may differ from the sum of those risks considered in isolation.

WHAT ARE COPULAS?

Copulas are a powerful mathematical tool for imposing a dependency structure within the risk model. Various families of copulas exist to capture all manner of dependencies. They allow variables to exhibit different degrees of dependency under different circumstances; in particular, differences between benign and extreme adverse conditions.

At its simplest, our EC methodology reproduces the ‘Sum of Squares’ calculation of Solvency II’s ‘Solvency Capital Requirement’. However, it has the ability to go much further.

RiskAgility EC builds upon existing projection systems

RiskAgility EC does not seek to supplant companies’ existing business unit ALM/DFA models, with which management is familiar and whose results it consequently trusts. Instead, these existing models inform the calibration of the aggregator.

RiskAgility EC and the RiskAgility Platform

Drawing upon data from multiple business units, group-wide calculations of economic capital require robust information security, high standards of auditability, and strong process controls. As a fully integrated and supported application of the RiskAgility enterprise technology platform, RiskAgility EC possesses all of these qualities, enabling economic capital to be embedded into the firm’s decision making. Together with the

appropriate risk governance processes, RiskAgility EC forms the basis of a firm’s ‘internal model’, as defined by Solvency II.

RiskAgility takes advantage of 64-bit server architecture and distributed processing to efficiently meet the computational challenges posed by EC calculations, and thus enables the industrial-strength production of economic capital results. The models, assumptions and output are stored centrally, from which sophisticated reporting and analytics can be generated.

The future of economic capital

RiskAgility EC is the catalyst to rapidly transform the use of economic capital by insurers in risk-based decision making. Forthcoming regulation and modern risk management concur that insurers must embed this approach into their business

processes, including strategic and business planning, product design and performance measurement. Combining Towers Perrin’s experience in economic capital design and implementation with the company’s robust RiskAgility enterprise platform, RiskAgility EC enables insurers to make this a reality.

To find out more about the capabilities of RiskAgility EC, please contact your local Towers Perrin consultant or one of the following:

CONTACTS

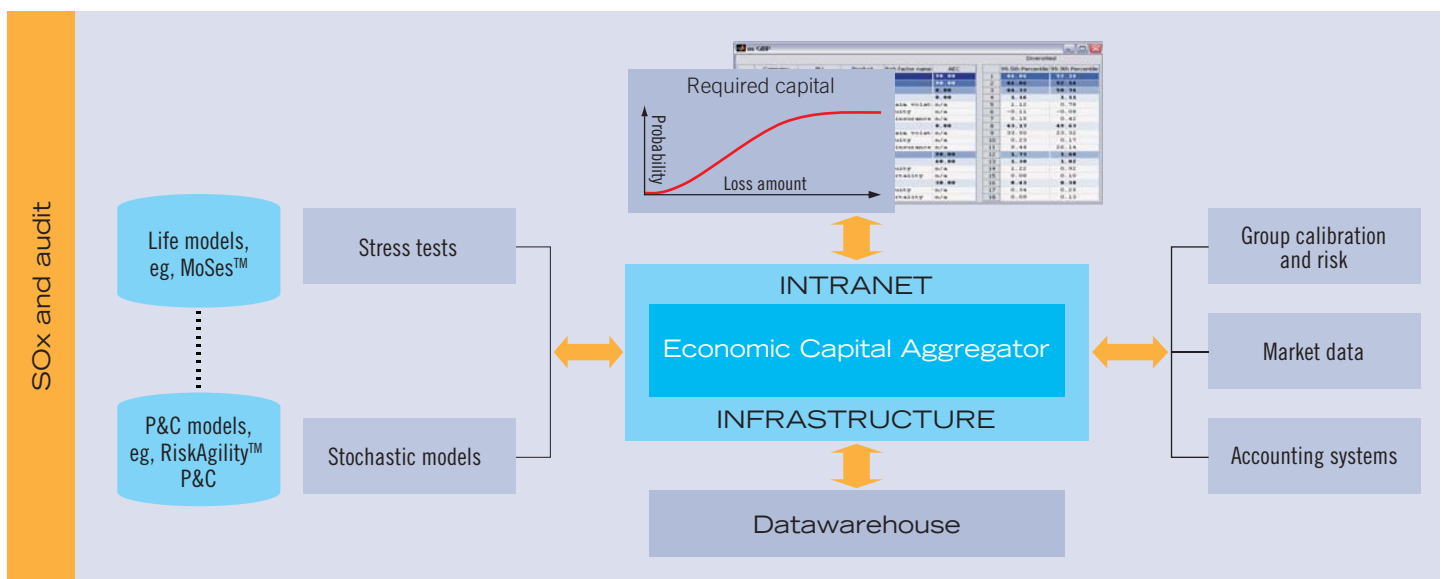
Europe

David Dullaway +44 (0)20 7170 2220
David Tonner +44 (0)20 7170 2577

Asia-Pacific

Lynda McCarthy +61 2 8198 9000

FIGURE 2 – RiskAgility EC fits into and supports key business processes and systems



C-ALM is a detailed, optimised, MoSes ALM application

LATEST RELEASE OF THE TOWERS PERRIN C-ALM APPLICATION NOW AVAILABLE

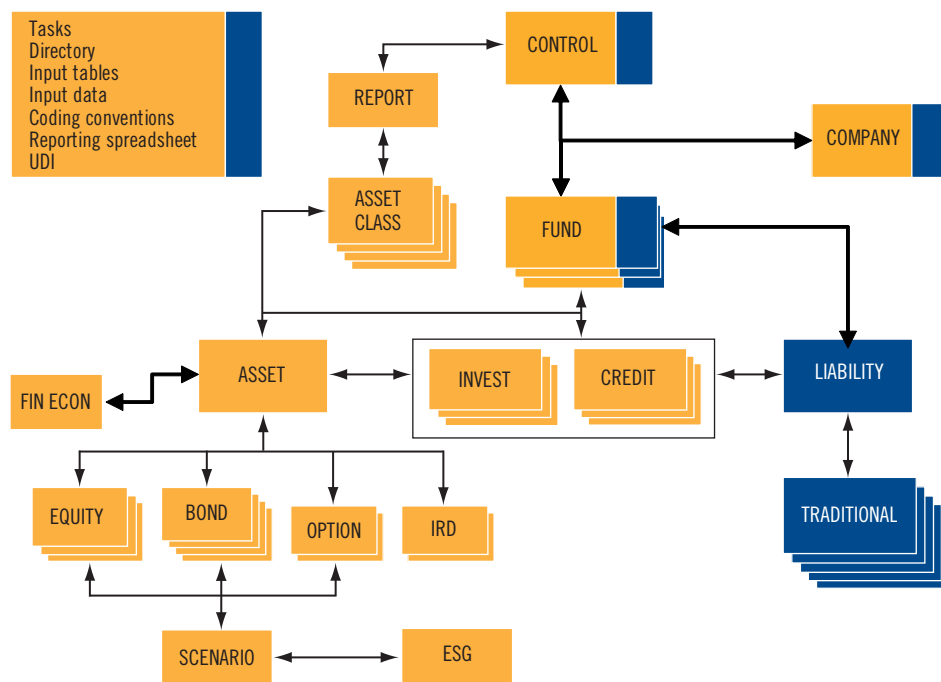
The C-ALM or Central Asset Liability Model is a detailed, optimised, MoSes ALM application, designed for the advanced modelling of profit-sharing business. It is already a fully functional application, but its modular design means it can easily be adapted to specific requirements for companies wishing to produce tailored ALM and realistic reporting applications (see Figure 3).

C-ALM includes the basic architecture needed to produce a full set of company results, enabling the measurement of value, capital and risk. The application provides, “out of the box”, a standardised set of assumption inputs, company reports and dashboard tools using either MoSes built-in User Defined Views or Excel reporting sheets. Calculation code is also provided for a variety of liability products and asset classes. A full set of MoSes tasks are supplied to enable each of the different projections to be performed.

Liabilities can be projected stand-alone or as part of realistic reporting, for statutory valuation, MCEV, economic capital or Solvency II. Asset classes modelled cover all of those commonly used by insurers: equities and real estate, corporate and government bonds, floating rate notes, equity put and call options, stand-alone caps, floors and swaps, and swaptions. C-ALM then allows dynamic management and policyholder behaviour to be modelled and the impact measured, using stochastic projections and sensitivities.

The generic C-ALM application has been designed to work with many of the product designs and the standard valuation rules used across Europe. However, due to the differing legislation and country specific products, different versions have been produced for specific territories:

FIGURE 3 – Towers Perrin’s C-ALM application framework



- German specific C-ALM
- French specific C-ALM
- Italian specific C-ALM
- Training or appraisal C-ALM – A cut down version for learning about the architecture.

We already have a number of clients using these C-ALMs across each of the countries specified, with varying levels of specialisation. The generic C-ALM application also works well for insurers in other countries across Europe and Asia, including those in Spain.

The recent updates have built on the existing functionality and added a number of new features:

- New business projections available
- MCEV calculations and reporting included “out of the box”
- Solvency II-ready functionality
- Wider variety of User-Defined View and Excel Reporting reports available
- Dynamic asset allocation and bundled “mutual” unit funds

- Intra-year valuation periods
- More country-specific calculations eg, Italian double accounting
- Updated to the latest version of MoSes

For further information on the design and features of C-ALM please see the article on C-ALM version 1 in the August 2007 issue of FMU available online.

To find out more about the capabilities of C-ALM and to arrange a demonstration, please contact your local Towers Perrin consultant or one of the C-ALM specialists:

C-ALM SPECIALISTS

Cologne

Christian Naecker +49 221 9212 3426

London

David Pond +44 (0)20 7170 2521

Milan

Simona Parise +39 02 7787 2329

Paris

Guillaume Beneteau +33 1 5393 1403

Asia

Soo Meng Foo +822 773 1157

Clone submodels simplify the process of projecting cashflows on more than one basis

IN-DEPTH TUTORIAL: FURTHER ASPECTS OF SUBMODELS

Introduction

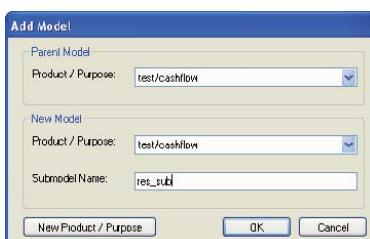
In the last issue, we introduced submodel types including Standalone, Fixed Arrays, Data Driven Arrays and User-Defined Arrays. This issue builds on this by discussing cloning submodels and how to use User-Defined Arrays to generate different output grouping.

Cloning

Clone submodels are a powerful feature of MoSes, which simplify the process of projecting cashflows on more than one basis using the same projection formulae. For example, calculating an Embedded Value with cashflows on an experience basis requires the change in reserves each period. The reserves are typically calculated on a more prudent basis but use the same basic cashflow formulae. A clone model in MoSes is simply an exact copy of the base model. It has the same model class, so it uses the same formulae and variables but is a submodel of the original model. Any changes in the base model are automatically reflected in the clone. This means there is less ongoing maintenance and reduced risk of the models being out of line.

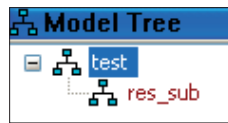
To create a clone, right-click the model you wish to clone on the model tree, and select Model -> Add. In the *Parent Model* box ensure the model you wish to clone is the parent and ensure the *New Model* has the SAME Product/Purpose, as shown in the screenshot (see Figure 4).

FIGURE 4 – Add model dialogue box



MoSes automatically marks the submodel as a clone (indicated by highlighting the model name in red on the model tree).

FIGURE 5 – Example model tree showing a clone submodel



By default, all variables, including data, have the same value in both the parent and its clone. This means the user can be sure that when a model point is calculated the same calculations are being performed within both the parent and the clone.

To allow a different basis to be set for the clone the variables making up the basis must have their variable property *Same Across Model* set to “No”. The variable values can then be set directly in the interface. Alternatively, MoSes code can be written to set the values of the relevant variables in the base and clone models.

In our model, we have created a string-type indicator variable *model_basis*. We can use this to tell each model which calculations it should perform. This has a choicelist of “Experience” and “Reserve”. In the Task View, we must set the Variable Datasource to Assumption; this then allows us to change the *Same Across Model* property to “No”. We can now use the drop-down box to select “Experience” as the value in the top model and “Reserve” for the submodel (see Figure 6).

FIGURE 6 – Task View showing different assumption values across clone submodels

Name	Value
initial_variable	0
model_basis	Experience

Name	Value
initial_variable	0
model_basis	Reserve

Syntax

We need to add some code within our application to use the clone model. It is important to remember that the clone has the same model class. Therefore, any changes made to the parent also change in the clone. If the parent model uses submodel syntax to refer down to the clone, this must not be called from the clone because an error will occur when MoSes looks for a further submodel beneath the clone. Our indicator variable is useful here as we can use code such as this for the reserve column:

```
if (eq(model_basis, "Experience"))
    return sm_res_sub->reserve(t) * surv(t)
/ sm_res_sub->surv(t);
```

We may also want additional code for the reserve model; for example, where no future surrenders are allowed on the reserve basis, the *surrender_rate* column may use:

```
if (eq(model_basis, "Reserve"))
    return 0.;
```

```
return surrender_tbl(prod_code,
policy_year(t));
```

Projection Task Setup

As well as creating the clone model and setting up the basis, the projection task property *Cloning Wanted* under *Run Control* must be set to “Yes”. This allows the user to switch off the clone for testing. It is good practice to include a check in the startup column which throws a *FatalError* if this property is not set to

Submodels can specify their own subgrouping of output

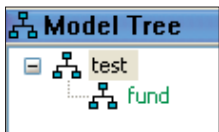
“Yes”. This is because, depending on the code written, either the clone results will be ignored or MoSes will give a run-time error when it tries to access the clone. Putting the check up front stops the run immediately rather than continuing and then skipping every policy or iteration.

Output grouping

MoSes allows output results to be grouped by specified criteria. The grouping specified in the top model (either through the iteration number or the group field on the data file associated with the top model) provides the highest level of grouping of results. Individual submodels can specify their own particular subgrouping of output within this highest grouping level. By using various submodels, it is possible to display output at different levels.

In the example below, a unit-linked policy has links to two funds. A policy may invest in both funds. Each fund may have different charges and the premium allocation may be different, hence their quite different size in the example.

FIGURE 7 – Example model tree



Using an arrayed submodel below the main policy, we can break each policy into its two funds and maintain a separate unit-fund roll-up for each. To demonstrate this, we will use the following simple data:

FIGURE 8 – Example data used to demonstrate output grouping

Policy No.	Product_code	Tax_status	Unit_fund_1	Unit_fund_2
polno001	prod1	Life	10,000	20,000
polno002	prod1	Pensions	50,000	80,000

We have linked data variables in the top model to the fund_1 and fund_2 unit values. In order to separate the funds, we have created a User-Defined Arrayed submodel (“fund” on the model tree in Figure 7). Within code, we create two elements in the fund model using:

```
sm_fund.resize(2);
```

The first element (element 0) will be for fund_1 and the second for fund_2. There is a single variable for unit value in the fund model which is populated from the top model, using code:

```
sm_fund[0]->units = units_1; // units_1
is linked to data for Unit_fund_1
sm_fund[1]->units = units_2; // units_2
is linked to data for Unit_fund_2
```

Submodel Output

By default, the two elements in the submodel will have the same group as the top model. So, if we look at the submodel output, we will just see the total result across all funds.

FIGURE 9 – Default output grouping

	A	B
1	Period	Unit_value
2	0	160,000

We can use the setGroup function to change the group for each element, eg,

```
sm_fund[0]->setGroup("Fund_1");
sm_fund[1]->setGroup("Fund_2");
```

The group field now shows the data grouping, with our additional grouping beneath it. The value of Fund_1 will be the sum of all the Fund_1 components for all the records in that product code, just as the output for the group in the top model shows the total result for that product code. Note both funds are branches below prod1, which is the group field on the data.

FIGURE 10 – Output grouping by fund number

	A	B
1	Period	Unit_value
2	0	160,000
3		

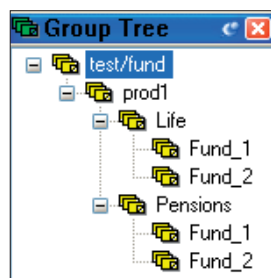
	A	B
1	Period	Unit_value
2	0	60,000
3		

Additional report models can be used to provide different levels of output from the same basic calculations

Complex subgroups can also be created for any submodels. Adding a “|” (vertical bar – SHIFT+“\” on a UK keyboard for example) into the group adds a new branch to the group tree within the same submodel. MoSes automatically groups by branch. By adding further branches it is possible to break the results down to lower levels. We could split out Life and Pensions results using the variable `tax_status` linked to the `tax_status` field on the data.

```
sm_fund[0]->setGroup(tax_status +
"/Fund_1");
sm_fund[1]->setGroup(tax_status +
"/Fund_2");
```

FIGURE 11 – Example group tree



It is also possible, either in MoSes (right-click on the Group Tree and select Add Combined Group) or in Excel Reporting, to use combined groups. For example, we could see just the total for Fund_1 by using grouping of `*|Fund_1` which will sum all groups but for the Fund_1 components only.

Additional Report Models

Often a single projection run needs to provide different levels of report output from the same basic calculations. By setting up additional submodels, MoSes allows this degree of flexibility without having to duplicate the calculations themselves. This is achieved simply by setting the group field for each of the report models appropriately from the top model. For example, this could be used during testing to view ungrouped output at policy level without editing the data file. It could also be used in production to view results by commencement year cohort (for GAAP purposes), while the main reporting output remains at product level.

In our example model shown in Figure 12, we have added a submodel “report” with a model type of User-Defined Array.

Only this model type is able to run data which may be grouped by product code but produce output grouped by a different measure. We have resized the array to one element in code at first use. We will use `setGroup` to view output at policy level (this could create extremely large output files if all time periods and all policies are used, but for a small subset the output files will be kept to a reasonable size).

```
sm_report[0]->setGroup(pol_no); // this
code needs to be run for every policy
```

For each policy, this sets the group field of the single element to the current policy number, thereby creating a new output group.

In the report model, we have also added a column for the unit value. We would need to add additional columns for any other output we require. These columns then need to contain code to pull the results from the calculation models, eg,

```
return test->sm_fund->units_e(t); // sum all
funds to give the total policy result
```

This is the output from our model with the output for Policy 2 shown:

FIGURE 12 – Output grouping in report model by policy number

	A	B
1	Period	Unit Value
2	0	130,000
3		

FURTHER INFORMATION

We have included some Q&As at the end of this issue focused on the topic of cloning and producing output.

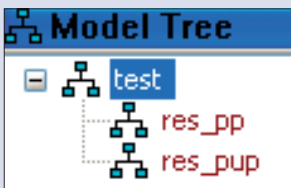
More information about submodel types and syntax can be found in the MoSes Online Help under **User Manual > Models, Model Classes & Submodels** and also in previous issues (March 2009 available online, and April 2002, June 2002 and June 2003 available by sending an email request to david.wright@towersperrin.com).

Q&A'S – COMMON QUESTIONS ON SUBMODELS

Q) Is it possible to have more than one clone of a model?

A) Yes. You can have many clones of a model. It is common, for example, for premium-paying policies to calculate reserves assuming the policy continues paying premiums and assuming it is made paid-up immediately (ie, premiums stop, a reduced benefit level is set in line with the proportion of the benefit which has been paid for and the policy remains in force), with the maximum of these two calculations being taken as the reserve. The indicator variable becomes very useful in this case as we have Experience, res_pp and res_pup cases all using different bases.

FIGURE 13 – Example model tree with two clones

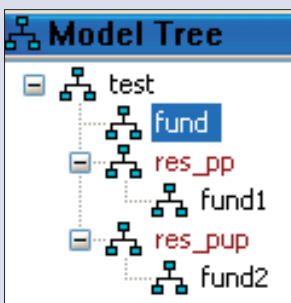


```
if (eq(model_basis, "Experience")
    return max(sm_res_pp->reserve(t), sm_res_pup->reserve(t));
```

Q) What happens if you clone a more complex model?

A) MoSes automatically clones the complete model structure from the cloned model, including all the children of the cloned model. This is where using submodel names is important to maintain referential integrity.

FIGURE 14 – Example model tree for a more complex cloned model



If the original code in the top model ("test") used the unique name for the submodels (eg, return fund->cashflow(t)), then the res_pp and res_pup submodels in this example would both, when cloned, not refer to their own subfund ("fund1" and "fund2" respectively); instead, they would refer to the "fund" beneath the experience model. This would require code changes such as:

```
if (eq(model_basis, "Experience")
    return fund->cashflow(t);
```

```
if (eq(model_basis, "Reserve_PP")
    return fund1->cashflow(t);
```

```
if (eq(model_basis, "Reserve_PUP")
    return fund2->cashflow(t);
```

No changes would be required, however, if the submodel name had been used as this would remain valid in all cases and is the key strength of coding using submodel names rather than unique names. The code would require no changes and be much simpler:

```
return sm_fund->cashflow(t);
```

Q) How can I ensure all the columns in a report model are calculated without having to list all the columns and force them to be called?

A) You can use the column interface functionality to loop over all columns in a model. This is commonly called from finalise so the order of calculation of the main cashflows is unaffected by the calls from the report models (as long as the cashflows called by the report model itself already exist). Here is some example code for finalise of the top model. It loops over all the columns in the report model. For each column, it loops over all time periods from zero until the final projection period (*last_model_period* is a MoSes system variable for the "proj period to" projection task property).

```
double test = 0.;
for (int col_num = 0; col_num < sm_report.columnCount(); col_num++){
    for (int time = 0; time <= last_model_period; time++){
        test = sm_report->Column(col_num).Value(time);
    }
}
```

News from the world of financial modelling

MoSes Version 6.3 and MoSes HPC Integration

MoSes Version 6.3 is now available on request from your Regional Support Centre.

MoSes HPC is also available for purchase. This is a new product that incorporates support for Microsoft's HPC compute cluster platform and 64-bit computing. To find out more, please contact David Tonner (david.tonner@towersperrin.com) in Europe or Lynda McCarthy (lynda.mccarthy@towersperrin.com) in Asia-Pacific.

Training Course Dates

Below is a list of training courses scheduled in our London office in 2009. We will be introducing a new Advanced Developer course this year, so watch this space for further details.

Please contact Merryl John (email: merryl.john@towersperrin.com, phone: +44 (0)20 7170 2537) to register your interest or request further details.

Dates	Users	Foundation developer	Intermediate developer
Q3 2009	7 – 8 September	10 – 11 September	14 – 15 September
Q4 2009	7 – 8 December	10 – 11 December	14 – 15 December

MoSes Japan User Group Conference – July 2009

We are looking forward to welcoming you to the 2009 Japan MoSes User Group, to be held on Tuesday 14 July 2009 in Tokyo Kaikan, Tokyo. This conference will provide you with an opportunity to learn how to get greater value and benefit from MoSes, exchange ideas with other users, interact with our staff, as well as receiving an update on the latest TPSS's (Towers Perrin Software Solution's) software developments such as MoSes HPC and RiskAgility™ offerings. Registration closes on Friday 3 July. For further details and registration please contact louise.birch@towersperrin.com.

ABOUT TOWERS PERRIN

Towers Perrin is a global professional services firm that helps organisations improve their performance through innovative human capital and risk and financial services solutions. www.towersperrin.com

Recent Towers Perrin papers

The start of 2009 saw the publication of many new Update papers and other publications at Towers Perrin. The following give a flavour of those most relevant to financial modelling:



May 2009 – The Solvency II directive agreed: Where to now?

Towers Perrin, recognised leader in Solvency II consulting, comments on the changed Solvency II Directive that has been cleared and, with 2012 implementation approaching, how European insurers should prepare.



Our quarterly Emphasis magazine

contains a number of articles relevant to financial modelling. The latest issue, published in Q1 2009 contains, amongst other interesting topics, articles on economic capital and enterprise risk management.

All publications are available by clicking the above icons. Alternatively, go to the Towers Perrin website, www.towersperrin.com, and follow the link to Risk & Financial Services / Tillinghast Insurance Consulting. From the Tillinghast page, you can then go to "Our Latest Thinking – See More".

FINANCIAL MODELLING UPDATE – EUROPE/ASIA-PACIFIC EDITION – CONTACT LIST

Regional Support Centres

Europe, Middle East and Africa

Tel: +44 207 170 3000 Email: support_eu@towersperrin.com

Asia Pacific

Tel: +61 2 8198 9008 Email: support_apac@towersperrin.com

FMU Europe/Asia-Pacific Editor

Joel Fox Email: joel.fox@towersperrin.com

Lucy Vickers Email: lucy.vickers@towersperrin.com

Asia-Pacific contacts

Asia-Pacific (excluding China)

Richard McPherson Email: richard.mcpherson@towersperrin.com

China

Roger Chan Email: roger.chan@towersperrin.com

European/EMEA contacts

France

Guillaume Beneteau Email: guillaume.beneteau@towersperrin.com

Germany/Austria/Switzerland

Christian Naecker Email: christian.naecker@towersperrin.com

Italy

Simona Parise Email: simona.parise@towersperrin.com

Netherlands/Belgium

Frank den Bieman Email: frank.den.bieman@towersperrin.com

Nordic region

Hanna Jacobsson Email: hanna.jacobsson@towersperrin.com

Spain/Portugal

Juan Ipiña Email: juan.ipina@towersperrin.com

UK, Middle East and South Africa

Joel Fox Email: joel.fox@towersperrin.com