Model design and lessons from IT

What lessons are there from large IT projects that can be used to improve the design and change management processes around Financial Models?

This is a potentially huge topic ……

…… but it’s one I’ve been fascinated by for some time.

In other words, how can actuaries/modellers, change professionals and IT professionals work better together?

Presentation outline

1. The FMS team and the financial models we look after
2. The move towards ‘enterprise’ financial modelling systems
   – and some key challenges
3. How do we run model development work?
4. Lessons learnt / take-aways
1. FMS and the models we look after (1 of 2)

“We develop and support leading-edge, innovative and cost-effective solutions to empower our customers to measure and manage their risk and capital position to support decision making”

We develop and support four modelling ‘suites’ in production:

<table>
<thead>
<tr>
<th>Model</th>
<th>Business units</th>
<th>Users</th>
</tr>
</thead>
<tbody>
<tr>
<td>Capital aggregation engine (internal model)</td>
<td>7</td>
<td>c. 80</td>
</tr>
<tr>
<td>Capital aggregation engine (standard formula, PIM, QRT)</td>
<td>c. 25</td>
<td>c.100</td>
</tr>
<tr>
<td>GI risk calibration and stochastic projection systems</td>
<td>3</td>
<td>c. 30</td>
</tr>
<tr>
<td>Credit risk calibration at portfolio level (*) in process of being rolled out from central to distributed user base</td>
<td>c. 10(*)</td>
<td>c. 10(*)</td>
</tr>
</tbody>
</table>

01 December 2015

1. FMS and the models we look after (2 of 2)

We have a team drawn from Finance and IT, across two sites (London and Norwich).

The team is (mostly) co-located and covers the following disciplines:
- Management
- Actuarial
- Quants
- Operations
- Project (and test) management
- IT developers
- Compliance

01 December 2015

2. ‘Enterprise’ financial modelling systems (1 of 6)

Each of our systems has its own:
- Distributed user base who can use our production systems 24 / 7
- Service agreement, service levels and support arrangements
- Regular and controlled development and release cycles
- Governance, documentation and compliance infrastructure

01 December 2015
2. 'Enterprise' financial modelling systems (2 of 6)

An example extract from the monthly Algo (IM capital aggregation engine) service report.

<table>
<thead>
<tr>
<th>Service target</th>
<th>Service</th>
<th>SLA</th>
<th>SLA</th>
<th>SLA</th>
<th>SLA</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>SLA</td>
<td>5%</td>
<td>10%</td>
<td>15%</td>
<td>20%</td>
</tr>
<tr>
<td>Report Requests</td>
<td>99%</td>
<td>99%</td>
<td>99%</td>
<td>99%</td>
<td>99%</td>
</tr>
<tr>
<td>Report Retrieval</td>
<td>99%</td>
<td>99%</td>
<td>99%</td>
<td>99%</td>
<td>99%</td>
</tr>
</tbody>
</table>

2. 'Enterprise' financial modelling systems (3 of 6)

To support our users 24/7, the production modelling systems:

- Have segregated user access by BU, Group and Operations.
- Have segregated areas for official reporting exercises and less formal investigative work.
- Have associated FMS data preparation tools, e.g. loss function fitting, asset data preparation, GI model class inputs.
- Flex to allow users to submit, queue, prioritise and cancel run requests (to varying degrees).
- The results are easily accessible to support user analysis using standard FMS reporting tools.

2. 'Enterprise' financial modelling systems (4 of 6)

Some key challenges - I

- General purpose modelling-friendly user interface
  - We have agility in our modelling engines and flexibility and speed of change using excel for some user interfaces.
  - However, ideally we need new solutions to create more robust user interfaces and workflows/actions, with little more than drag and drop, that can be easily updated as modelling processes and data / result flows change — and that can work as a common platform across multiple financial modelling products sets.
- The actuarial platforms are starting to address this need (through products such as Integrate and Unity), but it is early days and it’s unclear who the winners will be.
2. ‘Enterprise’ financial modelling systems (5 of 6)

Some key challenges - II

• SII brings a need for more rigour to model development, which in turn requires better testing, error-trapping and analysis tools.

• Adoption of software engineering processes and tools in the financial modelling process is still in its infancy:

  - Static code analysis (e.g. Sonar)
  - Test driven development (TDD)
  - Code repositories (e.g. GitHub)
  - Continuous integration (e.g. Jenkins)
  - Code deployment (e.g. Puppet)
  - Issue management (e.g. JIRA)

2. ‘Enterprise’ financial modelling systems (6 of 6)

Some key challenges - III

• Cloud adoption (100%) is a key transition for production modelling systems over the short-term

• At Aviva, our Prophet models are all now running on the cloud

• And we plan to have all FMS products 100% cloud-based by mid-2016

• This requires a new operating model, processes and skillsets

3. How do we run development work? (1 of 5)

• There are many approaches to software development
3. How do we run development work? (2 of 5)

- Until this year we tended to use an ‘iterative waterfall’ method with some aspects of the ‘V-model’ and some items following a prototyping approach.

3. How do we run development work? (3 of 5)

- In 2015, we adopted Agile across Aviva:

**Manifesto for Agile Software Development**

We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value:

- Individuals and interactions over processes and tools
- Working software over comprehensive documentation
- Customer collaboration over contract negotiation
- Responding to change over following a plan

That is, while there is value in the items on the right, we value the items on the left more.

3. How do we run development work? (4 of 5)

**Twelve Principles behind the Agile Manifesto**

1. Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.
2. Changes are delivered frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.
3. Business people and developers must work together daily throughout the project.
4. Customer collaboration within the team is essential.
5. Responding to change is a preference to the resistance to change.
6. That is, while there is value in the items on the right, we value the items on the left more.
7. The best architectures, requirements, and designs emerge from self-organizing teams.
8. The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.
9. Administrative expenses of managing inventories of viewed elements is face conversation.
10. The team must be able to maintain a constant pace indefinitely.
11. The team should be small and cohesive.
12. At regular intervals, the team reflects on how it became more effective, then tunes and adjusts its behavior accordingly.
3. How do we run development work? (5 of 5)

With Agile, we benefit from:
- Standardized processes and support tools across our four product suites
- Clear roles and responsibilities – particularly key are the product owner and the scrum master
- Each product team is focused on fortnightly sprints covering prioritised stories (features), discussed in daily (15 minute) scrums
- Delivery of valuable features in each sprint
- Fast feedback to correct course / build incrementally on firmer foundations

Our ‘iterative waterfall’ method was trying to achieve a similar outcome, but Agile (scrum) provides a better framework for success.

4. Lessons learnt / take-aways

- A large-scale modelling systems development and support team needs to embrace a multi-disciplinary approach
- Running distributed enterprise capital models requires a focus on the support model and service levels that customers expect
- Whilst we have many and varied modelling tools that can perform the calculations, it’s the interface / input / output management that is less developed and we continue to struggle with – but we will find solution(s)
- Agile works, but it is a discipline that requires time to learn
- Cloud is where modelling will increasingly take place

The views expressed in this presentation are those of the presenter.