

A MILLIMAN GLOBAL FIRM



Milliman USA

Consultants and Actuaries

SOA San Antonio Meeting 2004
Actuaries vs. IT – The Perpetual Dilemma

Emil B. Kraft, ASA, MAAA, MCSD

Actuaries vs. IT –
The Perpetual Dilemma?

There is hope.

Presentation Theme

- Many conflicts between actuaries and IT staff are due to the nature and structure of systems
- If you change system structure to be consistent with the nature of the business problem much of the adversarial quality of relationships can be avoided

Rating Systems – A Case Study

- Historically very painful due to their analytic nature
- Usually Actuarial/Underwriting vs. IT. vs. Sales
- Billions of dollars of revenue can be driven by rating processes
- A number of technologies have been introduced recently to help address the nature of these systems

Rating System Alternatives

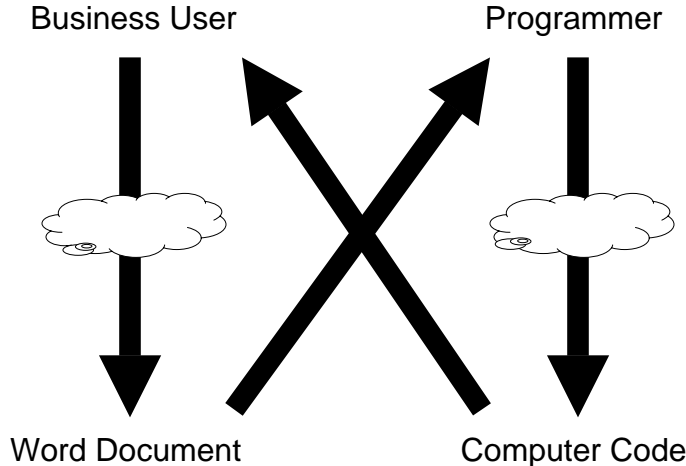
	Excel/Word Processes	Production Systems
Automation		X
Transparency	X	
Easy to Develop	X	
Confidence in Accuracy	X	
Flexibility	X	
Language Independence	X	
Scalability		X
Reporting		X
Centralized Security		X
Versioned Updating		X
Single Installation		X

Areas of Business Control

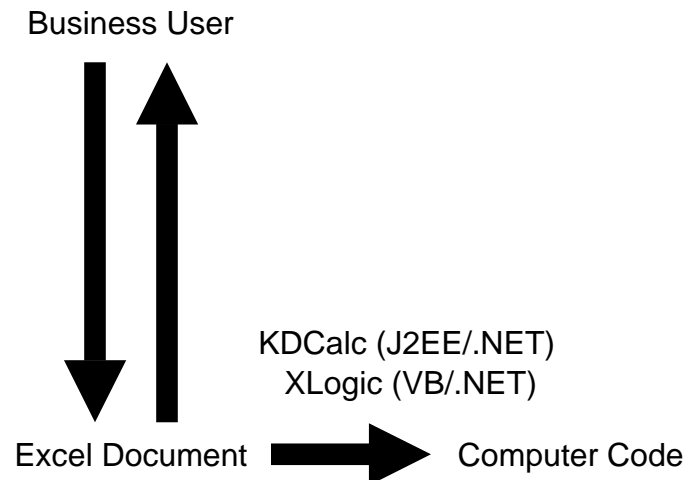
- Rating formula calculation engine
- Analytic environments
- Dynamically generated documents
- Parameter tables

Rating Formula Calculation Engine

The Problem



The Solution



Central Benefits

- Rating Logic started in 1990 Lotus 123 workbook
- 12 years and many owners later it had evolved into an unintuitive 6 MB Excel workbook
- Client desired a migration of its mainframe rating system to the Internet

Central Benefits

- Client changed and tested the workbook
- Milliman used XLogic to generate 60,000 lines of error free code from workbook in 1.5 hours
- Integrated engine into web-based rating and quoting portal

Central Benefits

- Maintenance continues in Excel
- Excel became the design, development, testing, and maintenance environment for production system calculation algorithms

Analytic Environments

The Problem

- Limiting, opaque, inflexible analytic interfaces

The Solution

- Dynamically generated Excel-based interfaces:
 - Generated by production system
 - Downloaded by business users
 - Uploaded when analysis is complete
 - Content of analysis extracted by the production system

Blue Cross Blue Shield of RI

- Achieved consensus on a highly generalized data feed to support analysis
- Agreed on an interface standard
- Built macro libraries to help retrieve and manipulate data along lines of expected use

Blue Cross Blue Shield of RI

- IT programmed production systems to generate Excel workbooks with specified data feed
- Actuarial and Underwriting programmed virtually all analysis worksheets against specified data feed using macros

Blue Cross Blue Shield of RI

- Actuaries and underwriters perform their analyses in Excel while enjoying the benefits of a production system
- Actuaries and underwriters also design, develop, test, and maintain their analytic environments completely disconnected from IT

Dynamically Generated Documents

The Problem

- After days of sophisticated analysis, little is communicated to sales or the client
- Dynamically generated documents tend to be limited and manually generated documents are time consuming and error prone

The Solution

- Use a dynamically generated document engine based that parses Microsoft-Word templates using a markup language that references the Excel workbook standard underlying the calculation engine










Nippon Life of America

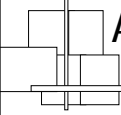
- Actuary makes a change to the Excel workbook underlying their rating system
- Actuary designs and develops accompanying dynamic document changes to the Word-based dynamic document template

Nippon Life of America

- Actuary tests dynamic document changes against Excel workbook
- Excel workbook is compiled, dynamic document template is moved into production, both changes are implemented

Conclusion

	Develop	Test
Rating Engine		
Analytic Environment		
Proposal Format		 
Parameter Tables		




Actuaries vs. I.T. – The Perpetual Dilemma

SOA San Antonio Meeting
June, 2004

Susie Lee FSA, MAAA
Allstate Financial


1



Actuaries vs. I.T. – the Perpetual Dilemma

- Many conflicts between actuaries and IT staff are due to the ~~nature~~ *and lack of understanding* of the structure of systems.
- If you ~~change~~ *align* system structure to be consistent with the nature of the business ~~problem~~ much of the adversarial quality of relationships can be ~~avoided~~ *minimized*.

2




Actuaries vs. I.T. – the Perpetual Dilemma

- There are two key phases to a software development project: Design & Construction
 - Design: Difficult to predict; requires creative people.
 - Construction: Easier to predict; can employ less skilled individuals (think automation).

→ Explore elements of design and construction that can be modified to bridge gap between expectations and results.


3



Actuaries vs. I.T. – the Perpetual Dilemma

- Actuaries Expectations of IT:
 - Adjust priorities to quickly implement changes ~ Design
 - Fix bugs and technical difficulties ~ Construction
 - Design programs that easily incorporate modifications ~ Design
 - Deliver error free systems ~ Construction


4



Actuaries vs. I.T. – the Perpetual Dilemma

- IT Staff Expectations of Actuaries:
 - Write specs that include all reasonably foreseeable modifications ~ Design
 - Write specs IT can program from ~ Design
 - Rigorously test all system calculations ~ Construction


5



Actuaries vs. I.T. – the Perpetual Dilemma

- Design Fundamentals
 - Accept that design will change. Fundamental business forces do not stand still. (IT1: Write specs that include all reasonably foreseeable modifications)
 - Focus on what is predictable. Adopt processes to control the unpredictable aspects. (A1: Adjust priorities to quickly implement changes)
 - Shift emphasis of deliverable. Did you get more value from the end result than you put into it rather than was the project on-time and on-budget. (A2: Design programs that easily incorporate modifications)


6



Actuaries vs. I.T. – the Perpetual Dilemma

- Design Fundamentals (continued)
 - Emphasize relationship management. Stress partnership through feedback mechanism. (IT2: Write specs IT can program from & IT1: Write specs that include all reasonably foreseeable modifications)
 - Modify management measurement. Delegate rather than dictate. Facilitate communication.

7



Actuaries vs. I.T. – the Perpetual Dilemma

- Construction Fundamentals
 - Reduce complexity; automate
 - Anticipate diversity (A1: Adjust priorities to quickly implement changes)
 - Structured for validation (IT3: Rigorously test all system calculations)
 - Follow established standards (A2: Fix bugs and technical difficulties & A4: Deliver error free systems)

8

Actuaries vs. I.T. – the Perpetual Dilemma

- Software Development Methodologies
 - Predictive:
 - History in civil and mechanical engineering; designed to minimize the use of, at the time, very expensive computing resources
 - Key issues are amendable to mathematical analysis (vs. reliance on peer review)
 - Better suited to projects with a small design element relative to construction (or if believe developers lack motivation)
 - Examples: Sequential V, Waterfall, Boehm Spiral Model

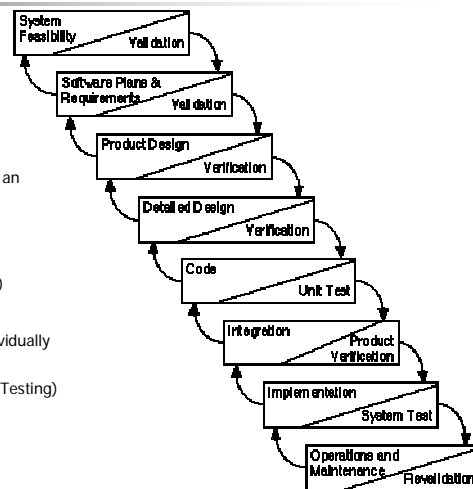
9

Actuaries vs. I.T. – the Perpetual Dilemma

■ Waterfall

The standard waterfall model for systems development is an approach that goes through the following steps:

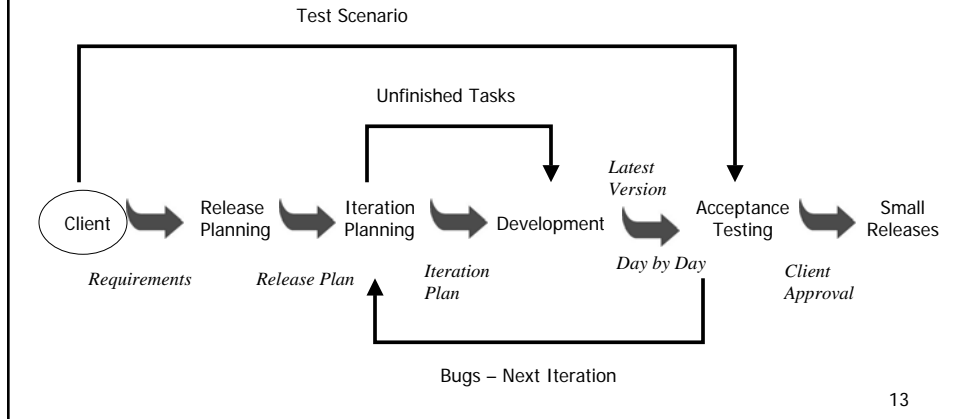
1. Document System Concept
2. Identify System Requirements and Analyze Them
3. Break the System into Pieces (Architectural Design)
4. Design Each Piece (Detailed Design)
5. Code the System Components and Test Them Individually (Coding, Debugging, and Unit Testing)
6. Integrate the Pieces and Test the System (System Testing)
7. Deploy the System and Operate It



10

Actuaries vs. I.T. – the Perpetual Dilemma

■ Extreme Programming (XP)



13

Actuaries vs. I.T. – the Perpetual Dilemma

■ Extreme Programming (continued)

Core practices of XP are:

1. **Small Releases:** Start with the smallest useful feature set. Release early and often, adding a few features each time.
2. **Refactoring:** Make small changes to your code, called refactorings, to support new requirements and/or to keep your design as simple as possible
3. **Simple Design:** Always use the simplest possible design that gets the job done. The requirements will change tomorrow, so only do what's needed to meet today's requirements.
4. **Continuous Testing:** Before programmers add a feature, they write a test for it. When the suite runs, the job is done. Tests in XP come in two basic flavors: Unit & Functional.
6. **Collective Code Ownership:** No single person "owns" a module. Any developer is expect to be able to work on any part of the codebase at any time.
7. **Continuous Integration:** All changes are integrated into the codebase at least daily. The tests have to run 100% both before and after integration.
8. **On-site Customer:** Development team has continuous access to a real live customer, that is, someone who will actually be using the system. For commercial software with lots of customers, a customer proxy (usually the product manager) is used instead.
9. **Coding Standards:** Everyone codes to the same standards. Ideally, you shouldn't be able to tell by looking at it who on the team has touched a specific piece of code.

14

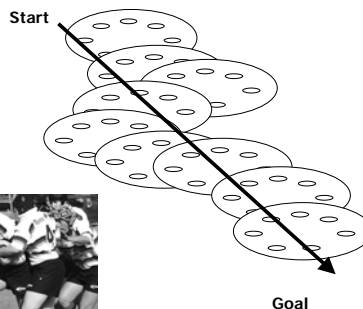
Actuaries vs. I.T. – the Perpetual Dilemma

■ Scrum

(A scrum is a team of 8 individuals in rugby; the pack works together to move the ball down the field)

Principles of a SCRUM Model:

1. Always have a product that you can theoretically ship: "done" can be declared at any time
2. Build early; build often
3. Continuously test the product as you build it.
4. Assume requirements may change. Have ability to adapt to marketplace changes during development
5. Small teams work in parallel to maximize communication and minimize overhead.




15

Actuaries vs. I.T. – the Perpetual Dilemma

■ Construction Process

- Specification – comprehensive statement of the problem
- Analysis – layout problem in a form that will lend itself to arithmetic and/or logical analysis
- Flow diagram – operations and sequence
- Encoding – converting operations into computer language
- Debugging – locating errors
- Documentation


16



Actuaries vs. I.T. – the Perpetual Dilemma

- Construction Styles
 - Linguistic Methods
 - Formal Methods
 - Visual Methods


17



Actuaries vs. I.T. – the Perpetual Dilemma

- Observations
 - If dissatisfied with results then perhaps using an inappropriate development methodology or inappropriate construction style.
 - Are all parties aware of their roles? Do team members have the correct skill set?
 - Communication can not be stressed enough!


18



Actuaries vs. I.T. – the Perpetual Dilemma

- Reminders to Actuaries – Are you guilty of the following?
 - Noise
 - Silence
 - Over-specification
 - Contradiction
 - Ambiguity
 - Forward Reference
 - Wishful Thinking

19



Actuaries vs. I.T. – the Perpetual Dilemma

- Reminders to IT Staff – solution should have following attributes:
 - Portable
 - Flexible
 - Complete & Comprehensive
 - Dynamic & Responsive
 - Consistent
 - Clear & Concise
 - Simple
 - Provides Suitable Controls

20

Case 1

In a conversion, one life is over 100 years old
System designed with 2 digit age field through out
Cost of change \$300,000
Policy for \$1000
Demand to Systems - put policy in
Reply: pay off the policy and use for publicity
Moral: cheaper ways may exist

Case 2

Actuarial division announces new products 3 weeks before year end
System requires 10 weeks to be updated and tested
Results: - policies had to be manually valued at year end
- Systems informed of every idea considered in last 3 years
Moral: right amount of communication at right points best

Case 3

Systems analyst designs system without input from line area
Demands approval on 4 days notice
System unable to produce all the output required for the Annual Statement
Systems' chief incensed and then humiliated
Moral: Systems does not always know best and must involve the other areas

Case 4

Group annuity system designed assuming no deferred lives over age 88
Found when system run for year end that 94 year old has not yet retired
Moral: find out the extremes

Case 5

Actuarial department uses 12 feet of approximations to develop reserve for a supplemental benefit
Systems area wants to do valuation exactly in a new system
Actuarial wants continue the use of the approximation
Costs: approximate , \$300,000 vs exact \$50,000
Moral: sometimes exact is cheaper than current approximations

Case 6

CPP conversion
Actuarial department wants a quotation system able to handle variety of integration methods
Time is extremely tight - system must be ready in 6 weeks
Actuarial division wants to modify current premium rate methods
At first meeting Systems department wants parameters of new rates
Actuarial division says method will not be ready for 8 weeks
Moral: do not try too much at once

Case 7

Actuarial prepares specifications for disability system
Systems prepares programming specifications and Actuarial signs off
System programmed and tested by Systems
Actuarial receives first run 12 months later at peak time
Actuarial does review 3 months later and finds problems
Systems notified and takes 12 months to reply - again at peak period
Cycle repeats three times - in some cases, repairing one problem discloses another and then another and then yet another
System abandoned as Actuarial was “not co-operating”
Moral: Systems must understand Actuarial’s schedule and also once a project is begun not see it as a new project each time there are corrections

Case 8

Actuarial division develops new wording for forms and sends to Translation
Systems division looks at wording and decides to use its own translations
Field force delighted with Systems’ translations - first time anyone understood French
Moral: need to use best available source of ability

Case 9

System for group annuities designed but outstanding premia incorrect
Systems refuses to let Actuarial prepare test data or work through cases
Problem persists at huge cost to Actuarial
System finally abandoned and replaced at a cost of \$5,000,000
Moral: work together and do not defend “turf”

Case 10

New system installed for accounting but not valuation
Accounting division wants to use accounting entries up to 31/12
Actuarial division must cut off for reserves 15/12 due to use of old system
Initial Annual Statement figures show a large loss
Computer division blamed for resulting year end errors
Moral: Understanding needed by all parties of interrelationships of data

Case 11

Chief actuary wants sample rates for a cash value disability product
Systems analyst programmes system using advanced Mathematical techniques
Programme set running Saturday am and 18 hours later Computer Room staff wants to go home
Programme stopped - premium narrowed to between \$ 40 and \$140
Chief actuary asks for results Monday morning and is told what happened
Chief actuary asks what starting value used - Told 0
Chief actuary contemplates maple tree out side office for 30 seconds and suggests \$85 as a starting value
Next Saturday programme again started. Ends 2 minutes and 15 seconds later. Value \$84.27
Moral Communication

Case 12

Computer system designed with 250 consistency checks
In practice 3 checks produce over 10,000 messages a month
Actuarial division says nothing but internally mocks Systems
Moral: Communicate problems and be sure error messages meaningful