Cost of Software Obsolescence Resolution of Real-Time Software
The QinetiQ Team

Sanathanan Rajagopal – QinetiQ Fellow
Estimating Manager- Cyber, Information and Training

- ICEAA Certified Cost Estimator / Analyst with the Parametric Specialism (ISPA)
- Chairman and member of the board of the Society for Cost Analysis and Forecasting (SCAF)
- Fellow of Defence Equipment and Support, MoD, UK
- Recipient of the ICEAA Technical Achievement of the year 2017 award
- Member of EPSRC Centre of Innovative Manufacturing (Through Life Engineering Services, Cranfield University)

Publications:
- Supported in the development of modules for ICEAA Software CEBoK
- Contributor to a study undertaken by Institute for Defence Analysis (US DoD) sponsored by Defence Logistics Agency (DLA, DoD) on Obsolescence titled “A Research and Development Investment Portfolio for Diminishing Manufacture Sources and Material shortages”
5

5 fundamental mobile phone technologies including Touch Screen and Liquid Crystal Displays (LCD) developed by QinetiQ experts.

40

40 organisations, including the Royal Navy, involved in 6 weeks of operations during the Unmanned Warrior Exercise.

50+

including 50+ unmanned vehicles operating in the air, land and sea.

12

12 Empire Test Pilot School students have become astronauts including Major Tim Peake.

85+

locations worldwide.

1,300+

patents (including 300+ pending).

3 secs

Every 3 seconds a Boeing aircraft takes off or lands that has been tested in QinetiQ’s low speed Wind Tunnel.

1,850 km

1,850km of the TANAP pipeline will be protected by OptaSense®.

£833 m

FY2018 revenue.

6,000+

people with unique science and engineering expertise.

16

Our Ocean Basin in Gosport, UK contains enough water to fill 16 Olympic swimming pools.

We are QinetiQ
Agenda

1 Research Aim
2 Definitions
3 Introductions
4 Software Obsolescence Cost Analysis Framework
5 Software Obsolescence Resolution Cost Optimization Model
6 Summary
7 Conclusion
Software Obsolescence

Research Aim
Research Aim

“To develop a cost analysis framework to estimate the cost of Software Obsolescence Resolution of a bespoke real-time software in defence and aerospace”
Software Obsolescence

Definitions
Software Obsolescence is defined as “what happens when the original and authorised third party ceases to provide support with regular update, upgrade, fixes or due to the changes in target or operating environment, systems or hardware which makes the software unusable”

-S Rajagopal et al; (2014)
## Software Obsolescence vs Software Maintenance

<table>
<thead>
<tr>
<th>Software Maintenance</th>
<th>Software Obsolescence</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bug fixes</td>
<td>Replacement of entire application if need be to a new one</td>
</tr>
<tr>
<td>To address fault/Failures, security patches etc.</td>
<td>To address the issues with the application in totality</td>
</tr>
<tr>
<td>Maintenance is the review of the stored files to ensure they are still useable</td>
<td>Solves unavailability of fixes, licenses, permission and upgrades</td>
</tr>
<tr>
<td>Software maintenance takes care of the current versions to ensure that its up and running and meeting the requirements</td>
<td>Software Obsolescence management looks forward the industry standards and other software to continue supportability of the software</td>
</tr>
<tr>
<td>Maintenance deals with the upgrading the software to enhance capability</td>
<td>Obsolescence management deals with enforced changes in the environment</td>
</tr>
</tbody>
</table>
Software Obsolescence

Introduction
Introduction

The need for a Software Obsolescence Cost Analysis Framework

- High Dependency on Software in Defence and Aerospace
- Constant Changes in Hardware
- Software Obsolescence is unavoidable
- Software is a key (cost and Schedule) Driver
- Long Support Contract

Unclassified-QinetiQ Proprietary
Software Obsolescence Cost Analysis Framework
Software Obsolescence Cost Analysis Framework

Following process was undertaken to develop the Framework

- 6 x Defence Case Studies
- 120 Responses from Online Survey
- Literature Searches
- Software Obsolescence Cost Analysis Framework
- Over 20 SME Interviews
- 6 x Cognitive Case Studies

Unclassified-QinetiQ Proprietary
**Research Methodology**

**Problem Statement**: Understand the problem and requirement for the need of S/W Obsolescence Resolution Framework

**Problem Validation**: Obsolescence
- Literature Review
- Reliability and Maintenance
- Cost Estimating
- Cognitive Case Study
- Obsolescence Management

**Hypothesis Analysis**: Software Obsolescence Cost Analysis Framework (SOCAF)
- Requirement Capture from Industry
- Research Gap Analysis
- 6 Cognitive Case Study
- 6 Expert Interviews

**Model Development**: Process and Procedures Development
- Models and Tools Development
- One Hour Structure Interview x 6
- Validation Case studies x 6

**Validation**
- Process Validation
- Model Validation

---

**Collecting**
- Analysis of current situations

**Proposing**
- Requirement Derivation

**Building**
- Framework Development

**Testing**
- Verification and Validation
Software Obsolescence Cost Analysis Framework

The framework has the following attributes

- This framework is in its final iterations.
- This framework’s foundation is based on the Literature Searches, Case Studies, Online Survey results, SME Interviews and Cognitive Case Studies.
- This framework has several attributes that can be mapped across from and to, to the software estimating principals.
- This framework looks at the Cost Risk and Uncertainty.
Software Obsolescence Cost Analysis Framework

- **Project and System Information**
  - Project Data
  - Project Schedule Data
  - Software Application Data
  - Software Sizing Matrix
  - System Details
  - Software Development Data

- **Obsolescence Management Level**
  - 1.As Is-State
  - 2.Resolution Strategy

- **Resolution Approach**
  - Identification of Resolution Techniques

- **Software Obsolescence Complexity Level**
  - Software Obsolescence Resolution Profile
  - Software Obsolescence Key Cost Drivers
  - Risk and Uncertainty Modelling

- **Software Obsolescence Management Systems**
  - Software Obsolescence Resolution Cost Modelling
  - Cost of Software Obsolescence Resolutions

- **Historic Data (Maintenance Data)**
  - ISBSG Data

- **CER based on Historic Data (ISBSG)**
  - Representation
  - Approach
  - Estimates
1. Identification of the software Obsolescence Resolution Approach
2. Compile Software Obsolescence Strategy
3. Generates Key Cost Drivers and Potential Risk/Uncertainties
Software Obsolescence Management Level

**Reactive**
- S/W Obs Mgt Level 1
  - Deal with Software Obsolescence Reactively
  - No Obsolescence Management Strategy
  - Freeze and do nothing
  - CMMI Level 1
  - Low TOMCAT Score

**Proactive**
- S/W Obs Mgt Level 2
  - No Software Obsolescence Management Strategy
  - Reactive but dealing with Software Obsolescence by reverse engineering and code conversion
  - CMMI Level 2
  - Low TOMCAT Score

- S/W Obs Mgt Level 3
  - Deploy software Obsolescence Monitoring process or tool if available
  - Monitoring software Supply Chain
  - Monitoring skills and technological insertions
  - Deploy software Obsolescence professional
  - Monitoring software Obsolescence Proactively
  - CMMI Level 3
  - Medium TOMCAT Score

- S/W Obs Mgt Level 4
  - Deploy S/W Obs Mgt Strategy
  - Proactive Mgt of update, upgrade and migration
  - Employ and deploy appropriate skills in-house
  - Mgt of Software Supply Chain and monitoring any technology insertions
  - Escrow agreement in place or third party partnership in place
  - CMMI Level 4
  - Medium TOMCAT Score

- S/W Obs Mgt Level 5
  - Proactive Mgt of S/W Obs
  - Deploy effective Mgt of S/W Obs Mgt Strategy
  - Continuous Monitoring of S/W Obs
  - Management of S/W obsolescence as BAU
  - Considering software Obsolescence at the design and development stages
  - CMMI Level 5
  - High TOMCAT Score
Software Obsolescence Complexity Level

High Risk Software

- Custom Software
- Real Time Software
- Custom Middleware
- Custom Glue Code
- Safety Critical Software
- High Requalification and testing requirement
- Require high end hardware
- Single Source
- Low Reliable suppliers
- Machine Code, 1st and 5th generation language
- No backward or forward compatibility
- Not easy to emulate

Medium Complexity

- Medium level of requalification and testing
- Non Safety Critical Software
- 2nd and 3rd generation language
- Readily available but requires minor re-design
- Easy to adapt
- Easy to emulate

Low Complexity

- Standard software
- Standard middleware
- Low requalification and testing requirement
- 4th generation language
- Readily available
- Backward and Forward compatible

Low Risk Software

Proactive

Reactive
Software Obsolescence Key Cost Drivers

Key drivers are determined by the following

• The resolution approach
• The level of integration (these are both software to software and software to hardware integration)
• Software/System multipliers
• Type of platforms
• Testing and requalification
• Level of modification required
Software Obsolescence key Cost Drivers

Resolution Approach
- Technical
- Functional
- Logistical

Level of Integration
- Low
- Medium
- High

Types of Platforms
- Commercial Systems
- Air
- Land
- Maritime

Testing and Requalification
- Low
- Medium
- High

Level of Modifications
- Low
- Medium
- High

Software Parameters
- Software Complexity
- Software Dependencies
- Number of Applications
- Length of Support Contract
- Software Languages
- Software Development Environment
- Software Development Life Cycle
- Target and Operating Environment
Software Obsolescence and Maintenance
Types of Software Maintenance

There are four different types of Software Maintenance. (from ISBSG)

• Perfective Maintenance: - Perfective maintenance is the modification of a software application, after delivery, to improve performance or maintainability

• Preventative Maintenance: - The modification of a software application after delivery to detect and correct latent faults in the software product before they become effective faults

• Corrective Maintenance: - The reactive modification of a software product performed after delivery to correct discovered problems.

• Adaptive Maintenance: - Enhancements necessary to accommodate changes in the environment in which a software product must operate
Maintenance Vs Obsolescence

Corrective Maintenance
Adaptive Maintenance
Perfective Maintenance
Preventative Maintenance

Reactive Management
Proactive Management

High Risk Software
Low Risk Software
Case Study – Software Maintenance Vs Software Obsolescence

• On Software with larger applications, about 95% of the time is spent on corrective maintenance.
• This indicates that more time is spent on reactive management of the software.
• In order to reduce software obsolescence, more time should be spent on preventive and perfective maintenance.
Software Obsolescence Resolution Cost Optimisation Model (SRCOM)
Integration of all the Models

Software Obsolescence Data Collection Form

Software Obsolescence Resolution Model

Software Obsolescence Resolution Cost Model

Software Obsolescence Optimization Model

Optimized Through Life Resolution Profile with Cost
Model build ups

• The model is in three parts
  – Data Collections
  – Resolution model
  – Cost model /Optimisation Model

• The data collection form is based on the back of the literature review, case studies, expert interviews and cognitive case studies.

• There is about 150 questions been asked on these forms.

• High level validation on these questions have been undertaken by,
  – Cross checking these with the outputs from Case studies and online surveys
  – Questions were send out to the Software Project managers in QinetiQ
This model will recommend the best resolution approach based on the cost

Optimisation is undertaken using MatLab Optimisation toolbox

This model is at a very early stages of its development and

This methodology involve diagrammatical representation of the cost

Diagram represents decompositions of cost from output node through successive levels to individual inputs nodes

This decomposition is the focus of the diagram and technique

The diagram will be converted into a set of inputs and calculations

Each nodes have inputs (Min, ML and Max)
Software Obsolescence Resolution Cost Optimisation Model - Nodes

- System/WBS/Component/PBS Nodes
- Resolution Profile Level 1
- Resolution Profile Level 2
- Resolution profile Level 3/Component Nodes
- Notes and Audit Nodes
- Information only Nodes
Software has several components and each component undergoes different obsolescence resolution profiles which can be diagrammatically represented using above methodology.

Diagram represents decomposition of cost from output node (on the left) through successive levels to individual input nodes (on the right);

This decomposition is the focus of the diagram and technique.
• Input values are entered on the right of the diagram and aggregate through the model to give the vignette cost.

• Each node has a specific formula for combining inputs, which are defined in the diagram.

• Module nodes allow repeated use of common elements, such as Software Engineers rates.
Software Obsolescence Resolution Cost Optimisation Model – Example
Optimisation Options

• In order to undertake Optimisation, it is necessary to identify the following
  – External Variables
  – Objective Functions
  – Constraints

• It is also necessary to develop a formula that uses the above variables, constraints and Objective functions in order to generate optimisation of
  – Cost
  – Resolution Strategy
Optimisation Formula

\[ \text{Pc} = [(\text{Tp} \times \text{In}) + (\text{TsIr} \times \text{Md}) + (\text{Ssp})] \times \text{Rs} \]

- **Tp** = Type of Platform
- **In** = Level of Integration
- **TsIr** = Testing and Requalification
- **Md** = Level of Modification
- **Ssp** = Software System Parameter
- **Rs** = Resolution Strategy Cost

**Objective Functions**

**Constraints**

**External Variables**
Optimisation Formula

Where,

\[ S_{sp} = (m_5C \times m_6D) + (m_7Ap + m_8Cl) + (m_9L \times m_{10}Dc) + (m_{11}De \times m_{12}Te \times m_{13}Oe) \]

Constraints

External Variables

- \( C \): Software Complexity
- \( D \): Software Dependency
- \( Ap \): Number of Applications
- \( Cl \): Software Contract Length
- \( L \): Software Language
- \( Dc \): Software Development Cycle
- \( De \): Software Development environment
- \( Te \): Target Environment
- \( Oe \): Operating environment
Optimisation Formula

\[ P = [(m_1 T_p X m_2 I_n) + (m_3 T_r X m_4 M) + (m_5 C X m_6 D) + (m_7 A_p + m_8 C_l) + (m_9 L X m_{10} D_c) + (m_{11} D_e X m_{12} T_e X m_{13} O_e)] X R \]
## Optimisation – Test Case

<table>
<thead>
<tr>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type of Platform</td>
<td>Air</td>
</tr>
<tr>
<td>Level of Integration</td>
<td>Medium</td>
</tr>
<tr>
<td>Testing and Requalification</td>
<td>High</td>
</tr>
<tr>
<td>Level of Modification</td>
<td>Medium</td>
</tr>
<tr>
<td>Software Complexity</td>
<td>Low</td>
</tr>
<tr>
<td>Software Dependency</td>
<td>Low</td>
</tr>
<tr>
<td>Number of Applications</td>
<td>2</td>
</tr>
<tr>
<td>Software Contract Length</td>
<td>6-10 years</td>
</tr>
<tr>
<td>Software Language</td>
<td>2nd Generation</td>
</tr>
<tr>
<td>Software Development Cycle</td>
<td>Stable</td>
</tr>
<tr>
<td>Software Development environment</td>
<td>Stable</td>
</tr>
<tr>
<td>Target Environment</td>
<td>Stable</td>
</tr>
<tr>
<td>Operating environment</td>
<td>stable</td>
</tr>
</tbody>
</table>
Optimisation – Test Case (Resolution Strategies)

Software xyz

Application 1
- Tech Res
  - Support
    - Maint
      - Corrective
    - Perfective
  - Tech
    - Update
      - upgrade

Application 2
- Func Res
  - Compatibility
- Tech
  - Support
  - Purchase
    - License
    - 1st Line
    - 2nd Line

Optimisation Level 1
Optimisation Level 2
Optimisation Level 3
Optimisation Level 4
Optimisation – Results

£6.200.000
£6.400.000
£6.600.000
£6.800.000
£7.000.000
£7.200.000
£7.400.000
£7.600.000

Corrective  Perfective  Update  Upgrade  1st Line  2nd Line

Total Resolution Cost
Summary

Software Obsolescence is an emerging issues and it is important to understand how much Software Obsolescence is going to cost at a very early stages of development life cycle. In order to do so we need to

- Define what Software obsolescence is
- Understand the difference between Software Maintenance and Obsolescence
- Identify how Software Obsolescence is triggered
- Have a framework to manage software obsolescence proactively
- Identify the key Software Obsolescence Resolution approaches
Conclusions

• Software plays an important role in defence. Almost every project in defence has software elements with various degrees of complexity and dependencies.

• In order to understand and see the bigger picture and challenges; software developers and the customers need to foresee the following issues that drive the whole life cost and should be in a position to develop innovative means to mitigate these issues by:

  – Anticipation of the Software Obsolescence at a very early stage of projects.
  – Understanding the technology insertion, technology update requirement.
  – Understanding the relationship between Software Maintenance and Software Obsolescence.
  – Anticipation of future capability integration to the existing platforms taking into account systems of systems, software to software and software to hardware integrations.
  – Formulation and evaluation of alternative architectural framework to inform the software designers that recognises the key market and cost drivers.
Thank you –Any Questions?