Outline

- Introduction
- Traditional Requirement Engineering (TRE)
- Agile Requirement Engineering (ARE)
- TRE vs ARE
- Shift from TRE to ARE
- Why this shift should help?
- Case Study
- Summary
Introduction

- Requirement Engineering applies different techniques and methods for the requirement analysis during development of software.

- TRE - complicated process

- Need - Flexible and speedy process

- Solution - Agile Requirement Engineering
Traditional Requirement Engineering (TRE)

- Identifying, modeling, communicating and documenting the requirements for a system

- Paetsch et al. [2] mentioned that:
  - Customer interaction only in early stages
  - Describes what is to be done than how to do
  - Prevents costly rework
Traditional Requirement Engineering Phases

- Elicitation – interviews, use-case, focus groups, brainstorming, prototyping
- Analysis and Negotiation – Joint Application Development (JAD), prioritization, modelling
- Documentation
- Validation
- Management
A traditional linear iterative requirements engineering model (Batool et al. [1])
Agile Requirement Engineering (ARE)

- Batool et al. [1] regard ARE as:
  - More flexible and quicker.
  - Benefit of constant communication between customers and developers.
  - Result: System delivered on time with customer’s expectations and better business value.
Agile Requirement Engineering - Methodologies

- Extreme Programming (XP)
- Agile Modelling
- Scrum
- Feature Driven Development (FDD)
- Dynamic Systems Development Method (DSDM)
- Adaptive Software Development (ASD)
Agile Requirement Engineering

An agile collaborative and innovative framework (Batool et al. [1])
## TRE vs ARE

<table>
<thead>
<tr>
<th>Traditional RE</th>
<th>Agile RE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Relies on Documentation</td>
<td>Face to Face interaction</td>
</tr>
<tr>
<td>Predictive</td>
<td>Adaptive</td>
</tr>
<tr>
<td>Process Oriented</td>
<td>People oriented</td>
</tr>
<tr>
<td>Include Use-Cases</td>
<td>Includes User Stories (business centric)</td>
</tr>
<tr>
<td>Realistic view of customer</td>
<td>Assumes customer knows everything</td>
</tr>
<tr>
<td>Customer involved only in the start</td>
<td>Customer is involved throughout the SDLC</td>
</tr>
<tr>
<td>Properly defined techniques</td>
<td>Techniques defined vaguely</td>
</tr>
</tbody>
</table>
Shift From TRE to ARE

A view of documentation within traditional and agile software developments (Batool et al. [1])
Why this shifting would help?

<table>
<thead>
<tr>
<th>Traditional RE</th>
<th>Agile RE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Blunt Planning</td>
<td>Flexibility/ Adoptability as per user needs and expectations</td>
</tr>
<tr>
<td>Highly technical/ unproductive(complex documentation)</td>
<td>Simpler</td>
</tr>
<tr>
<td>Lack of capability/ ability to respond to evolving requirements/learning</td>
<td>Easy to grasp evolving requirements, welcomes new requirements (which are consistent with old ones) at any stage in SDLC</td>
</tr>
<tr>
<td>Difficult to Re-organize documentation (wastage of time)</td>
<td>No time waste in building huge and complex documentation</td>
</tr>
</tbody>
</table>
Case Study

• A project of Hospital Management Information System (HMIS) has been developed by software team at some company.

• Applied:
  • Traditional Requirement Engineering
  • Agile Requirement Engineering

• 2 Data base administrators, 2 Managers, 3 Developers, 3 Technical Writers and 2 QA experts.

• Evaluation of the results on the basis of their expert opinions/responses.
Case Study: Critical Factors (For Evaluation)

- Interviews with the experts that why they moved to agile development:

1. Small Duration Project (SDP)
2. Project Team With Expertise (PTWE)
3. Up front Risk Analysis (URA)
4. Good Customer Relationship (GCR)
5. Face-To-Face Communication (FTFC)
6. Right Amount Of Documentation (RAOD)
7. Flexibility (FLXB)
8. Responsive To Change (RTC)
9. Correct Integration Testing (CIT)
10. Effective Delivery Management Process (EDMP)
Case Study: Results

Snapshot of Comparison for Critical Success Factor (Batool et al. [1])
Case Study: Results

Graphical Representation of Positive and Negative Responses in Traditional RE and Agile RE (Batool et al. [1])
Summary

• Growing shift from Traditional RE to Agile RE.

• Agile RE is likely to perform better than Traditional RE in large organizations where changes evolve throughout the development phase of software life cycle. (Batool et al. [1])
References

