Transformation from Linear Development Model to Iterative Development within a Waterfall Environment: A Case Study

THESIS

Presented in Partial Fulfillment of the Requirements for the Degree Master of Science in the Graduate School of The Ohio State University

By

Abhijit Rajiv Shirke

Graduate Program in Computer Science and Engineering

The Ohio State University

2012

Master's Examination Committee

Dr. Jay Ramanathan, Advisor

Dr. Rajiv Ramnath
Copyright by
Abhijit Rajiv Shirke
2012
Abstract

The application of agile software development is primarily an art. Organizations are adopting agile software development methodologies through a combination of bottom-up adoption and top-down change. However, reality of agile adoption has diverged from the original ideas described in the Agile Manifesto, with many actual adoptions resembling what is labeled as Water-Scrum-Fall by Forrester Research. In large organizations, development teams have adopted some variant of agile, for their product development efforts, what happens before and after this effort is still very traditional and waterfall. Not only is the performance of successive iterations difficult to compare, the results of agile adoption case studies are also difficult to compare as they vary in the scope of application of agile methods (most use only a subset of practices). Thus this thesis follows an exploratory case study approach to document the root cause for the water-scrum-fall phenomenon at a large scale Enterprise and presents the observations of a new initiative at the Enterprise to expand the scope of Agile development. Further, the aims of this case study thesis are to understand the limitations of Water-Scrum-Fall, and push back on the Water-Scrum side of the model. To do this, we model the as-is process (ie. the traditional handoffs model) and the to-be process (ie. Iterative requirements specification process) within an overall predominantly waterfall-based delivery model. The to-be initiative is still in its infancy, yet it provides new insights that we capture by a precise workflow for Iterative Solution Scoping to Deployment (ISSD) model.
Through the use of the ISSD model on a pilot project being implemented at the Enterprise we identify lessons learned thus far.

Specifically through the means of the ISSD, we show that the expanded scope facilitates:
1) regular feedback into the decision process for Continuous Improvement of Iterative Solution Scoping process and illustrate its performance-based utility, 2) better negotiations through interactions between the client and the development team by increasing collaboration, 3) define precise points of metrics for data collection which helps in better understanding the sources of performance deficiency towards managing variability, 4) change of mindset in letting go of the waterfall-related idea of "planning to commit," where development teams plan as much as possible upfront before committing to those plans, and instead shift to "commit to planning", where planning is a constant process throughout the development lifecycle, and 5) frequent and coordinated release cycle that results in a higher business value delivery and reduced risk.
Dedication

This thesis is dedicated to my family and friends.
Acknowledgments

I would like to thank Dr. Jay Ramanathan for her motivation, support, guidance and patience throughout the work in this thesis. Thanks to Dr. Rajiv Ramnath and Mr. Perumal Ramasamy, for their help and enthusiasm, particularly during initial phases of this work. Their continued support and assistance, thesis related and otherwise are highly appreciated.

I would like to mention special thanks for Mr. Hari Narayanswami, whose insight about the problem provided a direction for this case study. I wish to express sincere thanks to Mr. Charles Ward and Mr. Jawahar Talluri for their valuable guidance and contribution in data collection, understanding the process and discussions about the project, all of which have definitely helped to shape up this thesis.
Vita

2005 to 2009 …………………. Bachelor of Engineering (B.E.) in Computer Engineering, University of Pune, India

July 2009 to July 2010 …………… Persistent Systems, Pune, India

Apr 2011 to Jun 2012 …………… Nationwide Insurance – Application Development Center, Columbus, Ohio, USA

Sep 2011 to Jun 2012…………….. Graduate Research Fellow, CERCS for Enterprise Transformation & Innovation Lab, Dept. of Computer Science & Engineering, The Ohio State University, Columbus, Ohio, USA

Field of Study

Major Field: Computer Science and Engineering
# Table of Contents

Abstract........................................................................................................................................... ii

Dedication .......................................................................................................................................... iv

Acknowledgments .............................................................................................................................. v

Vita ..................................................................................................................................................... vi

List of Figures ..................................................................................................................................... xii

I. Introduction ................................................................................................................................ 1

a. Motivation .................................................................................................................................... 1

i. Water Scrum Fall is a reality ......................................................................................................... 3

ii. Challenges in Water-Scrum-Fall ................................................................................................. 4

b. Case : An insight into the Enterprise ........................................................................................... 5

c. Problems Explored by this Case Study ....................................................................................... 8

d. Goals of the Case study ............................................................................................................... 9

e. Contributions of Case Study ...................................................................................................... 10

f. Scope of Case Study .................................................................................................................... 11

II. Related Work ............................................................................................................................... 13
a. Empirical Studies on Comparison of Models..............................................13

b. Problem Analysis................................................................................................15

c. Project Negotiations and Handoffs .................................................................16

III. As- is : Traditional Model .................................................................19

    a. Transition phase increases cost of release one .........................................19

    b. Traditional Handoff Model ........................................................................21

    c. Water- Scrum-Fall at Enterprise ...............................................................23

IV. To- be : Iterative Solution Scoping to Deployment Model .........................27

    a. Need for Project Management strategy change ......................................28

    b. Solution Approach : Iterative Solution Scoping Process .......................29

        i. Principles of the Iterative Solution Scoping Process ............................30

        ii. Goals of the ISSD Process ..................................................................30

        i. Increased Collaboration by implementing ISSD ..................................32

        ii. Impact of ISSD implementation ..........................................................33

V. Modeling the Workflows ..............................................................................35

    a. Workflow Design using IDEF 0 notation ...............................................35

    b. As is : Traditional Linear Model for Software Delivery .........................37

    c. To - be : ISSD Model ..............................................................................39
d. Metrics to differentiate As-is and To-be model...............................................43
i. Survey Results used to differentiate as-is vs to-be model .................................44
ii. Metrics for Individual Phase Analysis..................................................................46
iii. Total time for project delivery based upon data input ........................................50
VI. Observations .........................................................................................................53
   a. Using Velocity to measure the Productivity of an Agile Development Team.......54
   b. Qualitative improvements due to ISSD Workflow..............................................54
      i. How has ISSD Approach Affected Team’s Productivity? ..............................55
      ii. How has ISSD Approach Affected Quality of Software produced...............55
      iii. How has ISSD Approach Affected Cost of Software Development .............56
      iv. How has ISSD Approach Affected Business Stakeholder Satisfaction.........57
   c. Timing of Value Delivery ....................................................................................58
   d. Estimating progress of development team ..........................................................59
   e. Calculating the ROI of Implementing Agile Practices – Agile increases output
      and efficiency ........................................................................................................63
         i. Metrics to measure the operational costs/savings associated with improved
            efficiency ........................................................................................................63
VII. Conclusion.............................................................................................................66

Risks & Opportunities...............................................................................................67
VIII. Future Work ...........................................................................................................69

References ......................................................................................................................70

Appendix A :Survey Questions .....................................................................................74

Measuring the performance of Iterative Solution Scoping Strategy .........................74

a. Introduction ..............................................................................................................74

b. Personal Data .........................................................................................................75

c. Survey 1 : Questions for this Iteration .................................................................76

d. Survey 2 : Iterative Solution Scoping Performance .............................................77

e. Survey 3 : Stakeholder Satisfaction .....................................................................78

f. Qualitative Analysis ...............................................................................................79

g. Survey 4 : Iterative Scoping Process Feedback ....................................................81
List of Tables

Table 1 Project release data for pilot project ...............................................................46
Table 2 Time and SLOC data for multiple releases of pilot project .................................47
Table 3 Individual phase analysis ..................................................................................47
Table 4 Sample Data for project depicting EV, PV and variance ....................................62
Table 5 cost per sprint......................................................................................................64
Table 6 Financial measure of efficiency .........................................................................64
List of Figures

Figure 1 ESDm Life Cycle- Enterprise software delivery process[23] ..........................6
Figure 2 A flexibility diagram depicts constraints, drivers, and degrees of freedom for a project. ........................................................................................................................................18
Figure 3 Current Timeline for New Projects in Development Center .................................20
Figure 4 Traditional Handoff Model showing rework ..........................................................23
Figure 5 Chart representing quadrants which correspond to different project strategies ...29
Figure 6 Impact on Solution Scoping phase as depicted in [23] ..............................................33
Figure 7 IDEF 0 Activity notation .........................................................................................37
Figure 8 As-is : Traditional Handoff Model .............................................................................38
Figure 9 ISSD process - High level Model .............................................................................39
Figure 10 To-be model : ISSD workflow with decomposed activities .................................41
Figure 11 Time vs Delivery Chart for Releases ......................................................................48
Figure 12 Time (hours) vs SLOC chart ...............................................................................49
Figure 13 SLOC per hour for different releases .....................................................................50
Figure 14 Working of a Iterative Solution Scoping to Deployment model .........................52
Figure 15 Impact on Team's productivity .............................................................................55
Figure 16 Impact on Quality of Software produced ............................................................56
Figure 17 Impact on Cost of Software Development ..........................................................57
Figure 18 Impact on Business Stakeholder Satisfaction ........................................................58

xii
Figure 19 Operational Savings as in [26] .................................................................65
I. Introduction

So far, only few studies have focused on the direct comparison of process models in general, and between traditional, linear process and iterative agile approaches in particular. Thus, it is not made explicit what the effect is of moving from one model to another model. Also, there is limited evidence on impact of adopting agile software development, specifically in the context of large-scale Enterprise. The objective of the thesis is to investigate how the perception of bottlenecks, unnecessary work, and rework changes when transitioning from a traditional to an iterative software development approach with agile practices. Semi-structured interviews were conducted in order to investigate the perceived changes during transition, the survey results being an important source of insights. It is concluded that the recent introduction of iterative process brings added value in comparison to the traditional approach, and this is evident from the absence of critical issues that are encountered in traditional, linear process.

a. Motivation

Agile is a software development methodology that focuses on high customer interaction, scenario-based development, user stories, continuous integration and code-centered development and documentation, refactoring and paired programming. In large organizations, development teams have adopted some variant of agile, for their product
development efforts, but what happens before and after this effort is still very traditional and waterfall. The front-end of a project still has the approval stage and the non-agile interactions with Stakeholders in the company. The back-end of every iteration, too, is burdened with non-agile steps and interactions (which are project handoffs). This is a big impediment to agile that requires a working solution to be ‘released’ at the end of each iteration, providing quick feedback and input to the next iteration. This results in software delivery model, where in application development professionals need to carefully consider and make the right decisions about where the lines fall between Water-Scrum and Scrum-Fall. Otherwise, they are unlikely to realize agile's business benefits, such as faster time-to-market, increased business value, and improved flexibility and responsiveness.

The motivation behind this case study is to understand the impact of the change from traditional development models to agile-based iterative methods. Specifically, to understand the impact of transition to an iterative agile method, which requires the comparison of agile with other approaches in different contexts. For example, how does traditional development perform in comparison to iterative development in different domains and different system complexities. The effect was captured in terms of advantages and issues for the situation before and after the migration.

Let us see the real-world problems faced within a larger enterprise shared IT services organization, after adoption of agile methodologies in Waterfall environment:
i. **Water Scrum Fall is a reality**

"Water-Scrum-Fall is the Reality of Agile for Most Organizations Today", is the norm of most organizations as said in [3]. Next we will present brief description of different stages in Water-Scrum-Fall scenario as described in [3].

1. **Water** – Defines the upfront project planning process that typically happens between IT and the business. Governance rules require many organizations to define requirements and plans before starting work. In some companies, these plans form the basis of a contract between the business and IT that defines project direction, timeline and budget.

2. **Scrum** – The practice of frequently releasing software using scrum methodology, makes intuitive sense. Implementation is based upon an iterative and adaptive approach to achieving the overall plan that was first laid out in the 'Water' stage.

3. **Fall** – A controlled, infrequent production release cycle that is governed by organizational policy and infrastructure limitations. Frequent software releases to the customer enable rapid feedback and ensure that valuable software is being used as early as possible. However, most organizations do not have the architecture required to support dynamic, flexible releases; instead, they do infrequent releases backed by heavy processes and governance.
ii. Challenges in Water-Scrum-Fall

1. Traditional Plans drive funding

Delivery organizations in large enterprises have built up decades of traditional management tools and processes — budgets, plans, architecture road maps, and requirements specifications — that make it challenging to move agile beyond the software development phase and into these aspects. Organizations rely more on traditional Waterfall processes as Project plan includes a detailed description of the tasks, resources, and time the project requires, translated into cost and time estimates.

2. Requirements handoff issues

Business Users trained in business analysis cannot always communicate effectively their requirements in technical terms. Hence many times they only understand the problem after they have seen a solution that was not quite right. Often the business people who are involved are not the same as the users, and they are not domain experts or design experts. Hence development teams must understand the business and the users better than they understand themselves.

Another problem with spending a great deal of time writing down detailed requirements is that users get to see the whole solution after some delay and only then can provide feedback. Feedback late in the cycle tends to be less welcome, as the team has to balance the feedback with the significant cost of changing the application (we depict the cost for change curve later in thesis). This waste due to re-negotiations not only stops the team from spending time on more-important requirements but also increases project complexity. Thus handoff from requirements to development may encourage teams to
focus less on customer value and more on fulfilling their contract. The resulting process looks more like a game in which the business tries to get all its requirements in and IT tries to avoid including any requirements that might increase risk to the project and add very little to overall business value.

The above points motivate this case study, the goal being to understand and characterize at a workflow level, all of the various problems and inefficiencies that riddle agile adoption and fail to realize agile's implementation benefits. This case study will be a starting point to introduce the problem of Water-Scrum-Fall and will focus on documenting a possible solution which is a process change within the traditional software delivery model that can effectively lead to pushing more agility into the traditional planning and estimation phases.

b. Case: An insight into the Enterprise

The organization in which this case study is set is a leader in Insurance sector of the economy. The company has currently approximately 36,000 employees and a yearly revenue of about 20 billion Dollars. Within this company, there is an IT organization of 5,000 employees responsible for developing and providing IT services. The internal IT organization is the preferred IT provider. Within the IT organization, the unit in charge of application development, is experimenting with Iterative software development, even though the normal software process in the company is a waterfall-like one. The agile based method was tailored in order to comply with the project control issued by the IT organization.
The said Enterprise has adopted the Enterprise Solution Delivery methodology (ESDm), which is traditionally Waterfall like, for commercial software development and is based upon IBM's Global Services Method. The ESDm organizes projects into five distinct development phases as a means to consistently describe the levels of solution definition.

IBM's Global Services method has been the foundation on which the framework of ESDm is defined.

The following diagram describes the 5 stages of ESDm:

![Figure 1 ESDm Life Cycle- Enterprise software delivery process](image)

The diagram above highlights the ESDm components:

1. Five project phases: Initiate, Solution Scoping, Design, Develop and Implement
2. Four dimensions: People, Process, Technology and Program Management
3. Seven Disciplines which make up the dimensions: Organization change, Requirements, Architecture, Application, Test, Production Readiness and Project Management.
All of ESDm work products and deliverables map to the defined seven disciplines. ESDm defines the IT Capability of Enterprise, which comprises of People, Processes, Context, Technology, Assets and Business Partners that provide outputs of value to customers.

A brief background of the software delivery life cycle at the Enterprise:

The project initiation phase for any new project is completed at Business site and the project is handed over to Development Center after the Solution Scoping phase. This case study will focus on highlighting the issues of transition phase that flows across two different software development methodologies ie. Waterfall, which governs the Planning and Scoping phase and agile which is adopted for the development phase. Every phase has its distinct characteristics that define the progress of a software project. There is lot of friction during project transition due to involvement of different and conflicting methods, strategies; for example: the Business might estimate based upon roles and the Development team might estimate based upon team size, which leads to a considerable variance between estimates for the same software effort.

In order to reduce friction and smoothen transition of projects into development center, an iterative solution scoping strategy is being implemented for a pilot project. Once the project is completed, we will be able to perform an in-depth assessment of the effectiveness of this new strategy based upon quantifiable values. For the purpose of this case study we rely upon data gathered for few releases of the pilot project, which were done by adopting the new strategy.
c. Problems Explored by this Case Study

Traditional software delivery process evolved in response to business’ growing demand for IT to deliver software using a contract based handoff approach with a focus on interim documents to describe the problem and its solution. Agile evolved in response to uncertainty, helping teams deliver value while working in a world of changing understanding and priorities. Most IT organizations don’t follow a strictly traditional or strictly agile approach but fall somewhere in the middle.

Our aim is to formalize the non-standardized adoption of agile into a Waterfall based software delivery life cycle and study the implementation of iterative scoping of the requirements and creation of project artifacts during the Solution Scoping phase, which continues all the way up to software delivery. To precisely model the Iterative Solution Scoping process for Continuous Improvement, we would like to conceptualize the Iterative Solution Scoping to Deployment model (ISSD), which will aim to scrap the boundaries of Water-Scrum-Fall and help to push back more agile into the Water-Scrum side of project life cycle. To summarize the problem to be studied, we see that transition from waterfall to agile software development requires careful planning, collaboration and change management. When implemented correctly, the agile model can result in the project team accelerating production efforts and working more cohesively.

The project management [23] plans to develop the process using an approach that:

a. Forces Conversations between stakeholders and development teams.

b. Generates Consensus on the project expectations.
c. Breaks down silos within the organization.

d. Builds trust and confidence between Business and IT teams.

d. Goals of the Case study

This case study will focus on standardizing the proposed Iterative solution scoping method, which will facilitate introducing more agility into project planning phase. A major goal of this study is to highlight the importance of business team scoping out requirements in phases to speed up start of iterative, incremental development lifecycle of software. This initiative will result in change of mindset to let go the waterfall-related idea of "planning to commit," where development teams plan as much as possible upfront before committing to those plans, and instead shift to "commit to planning", where planning is a constant process throughout the development lifecycle. At a more granular level we summarize the goals of the implementing the ISSD model as:

a. Define a process for Solution Scoping phase that:

1. Enables incremental delivery of software through iterative scoping and development phases.

2. Enables dynamic feedback to the development team and prioritize the functionality to provide early benefits to stakeholders.

b. To identify metrics for measurement and points of data collection for the ISSD model to measure performance of Iterative Solution Scoping strategy.
c. Comparative analysis of Traditional handoffs model and the Iterative Solution Scoping model involving interactions among stakeholders aiming to eliminate handoffs.

d. Data collection to extrapolate iteration results to Continuous Improvement decision-making.

e. Contributions of Case Study

The contributions of the paper and case study are:

1. Illustrates an industry approach of using iterative agile practices and structuring the observations using workflows.

2. Identify and gain an in-depth understanding of the most important issues in relation to process performance in traditional development and the process used after introducing iterative and agile practices.

3. The outcomes of the situation before (as-is) and after the migration (to-be) were compared and discussed. This information was captured through interviews, and thus illustrates the perception of the effect of the migration.

4. Precise workflow modeling of proposed iterative solution scoping process and current traditional handoff model, which allows the capture change of mindset to let go the waterfall-related ideas and adapt to changing business conditions and provide maximum business value within given time and cost constraints.

5. Provide process performance measurements on the to-be development approach as an additional source of evidence to support or contradict the primary evidence in the form of qualitative findings from the interviews.
Thus the overall contribution of this case study is to precisely model the As-is and To-be workflows for Traditional process and ISSD process respectively, to provide a framework for decision making for CI. Also with this the case study itself can be used:

1. As an example to introduce the challenges of Water-Scrum-Fall at large scale Enterprises.
2. To estimate progress of development team based upon pre-defined performance metrics to highlight the advantage of implementing new ISSD process.
3. As a reference to conduct an in depth analysis of the proposed Iterative Solution Scoping process once the pilot project has completed its final delivery.

f. **Scope of Case Study**

The study was performed in a division of an IT department of a large multinational company. Sets of people in different roles in a given department of this organization were interviewed. We triangulated the results with the opinion of experts from other departments and existing literature. The research defines several factors for which the use of iterative development is preferable over waterfall development in terms of business value, however there are also some clear costs and limitations that should be explicitly considered by any company planning on introducing an iterative development process. For this case study, we limit our scope of data collection for a single pilot project at the development center. Project management tools such as "Clarity" and "Slim" were source of data for analysis of as-is and to-be methods in the scope of case study. We have used IDEF 0 modeling notations to define the workflows which can be used as a reference for specifying points of data collection.
The research was conducted in a limited environment over a relatively short period of time, limiting its ability to make any strong statements on organizational improvement at an enterprise level. The resources and time required to conduct such an extended study were well beyond our means.

For the confirmation of our proposed ISSD model we present observations based on a Qualitative Assessment of the Iterative Solution Scoping approach based upon results of survey interviews. Hence even the limited scope of this exploratory case study calls for considerable amount of future work.
II. Related Work

Studies have investigated the advantages and disadvantages of plan-driven and agile processes. However, few studies present a comparison of the models in general, and the effect of moving from one model to the other. This section summarizes the results of existing empirical studies on comparison of process models. This section will also provide in depth analysis of problem domain at the Enterprise. We conclude this section by introducing key factors affecting negotiations during handoffs and project drivers, constraints and degrees of freedom for project "Interactions".

a. Empirical Studies on Comparison of Models

In waterfall development the requirements are specified upfront, even the requirements that are not implemented later (due to change). The introduction of an iterative approach to requirements specification, reduces the impact of change requests on a project. Furthermore, the increments can be delivered to the customer more frequently demonstrating what has been achieved. This also makes the value of the product visible to the customer early in development [19]. Furthermore, several studies indicate that agile companies are more customer centric and generally have better relationships to their customers. This has a positive impact on customer satisfaction.
However, a drawback of agile development is that team members are not as easily interchangeable as in waterfall-oriented development. A study reported a productivity gain (LOC/Effort) of 337% [21], was a multi-project experiment. Regarding time consumption, the results show that XP saves time on requirements, but requires more time for verification and validation. In coding, no major time differences were discovered. The huge gain in productivity was due to more code developed. However, it is important to mention that this does not imply that the team delivered more functionality. Ilieva et al. (2004) measured productivity for each iteration and compared the productivity of a baseline project (characterized as heavyweight and documentation driven) with an XP project. The productivity gains are the highest for the first two of three iterations. However, the last iteration did not lead to gains as only bug fixing and modifications were requested in this iteration. Layman et al. (2004) compared two releases with each other, one developed with traditional methods and one using XP. The results show improvement in programmer productivity by 46%. However, as pointed out in the study the increase on productivity can also be influenced by the gained experience during the development of the first release.

With regard to quality [22] reported positive effects of the introduction with regard to reduction in the number of defects discovered. The number of defects pre-release was reduced by 65% and the number of defects post-release was reduced by 35%.

Given the results of the related work it becomes apparent that benefits reported were not identified starting from a baseline, i.e. the situation before the introduction of agile was not clear.
Hence, little is known about the effect of moving from a traditional linear process to an iterative agile approach. Furthermore, evaluating the baseline situation is important to judge the improvements achieved through the migration. In response to the research gap this study investigates the baseline situation to judge the effect of the migration towards the iterative and agile approach.

b. Problem Analysis

This section will discuss the problem of Water-Scrum-Fall in depth, which is the focus of this case study. In the traditional processes adopted by most Enterprises, where agility is espoused by management, there is still tremendous up-front planning of requirements[3]. Dealing with the unknown is not a characteristic of a waterfall process, hence the business stakeholders still want to reduce risk surface area by doing as much planning up front as possible. This plan defines the project and it includes a detailed description of the tasks, resources, and time the project requires, translated into cost and time estimates. Though agile projects do encourage having an upfront plan, that plan often includes far fewer details than traditional approaches require[1].

At a high level, the waterfall approach infers structured and linear project cycles. Detailed project specifications form the contract during handoff and waterfall model mandates that this plan is adhered to without deviation. Thus teams tend to work on their piece of the software deliverable in isolation, and then they pass it along to the next team in the sequential production cycle. Using this approach, feedback and revisions are slated as the final phases of production, once the entire product has been completed. This means changing software functionality can be costly and cumbersome, because even a small
change at the end of a project may have implications to many parts of a finished product. One more shortcoming is the limited collaboration between different teams which leads to frequent clarifications and renegotiations during the project transition phase from Solution Scoping to development.

c. Project Negotiations and Handoffs

Let us have a look at some factors, which affect negotiations and handoffs during project transition phase in any industry.

i. Project Negotiations during handoff [11]:

1. Negotiation is required whenever there is a gap between the schedule or functionality the key project stakeholders demand and development team’s best prediction of the future.

2. Principled negotiation involves four precepts [16]:

   a. Separate the people from the problem.

   b. Focus on interests, not positions.

   c. Invent options for mutual gain.

   d. Insist on using objective criteria.

ii. Project drivers, Constraints and Degrees of freedom for Project Interactions - Following is list of basic parameters which define and govern the development of a software product [16]:

16
1. Functionality
2. Staffing
3. Cost
4. Schedule
5. Quality objectives.

iii. Constraints on IT Project

1. Uncertainty in mapping features to design
2. Limited resources and time
3. Ever changing technologies
4. All contributing to unachievable demands

A “flexibility diagram” [16] such as that shown in Figure 2 visually depicts - constraints, drivers, and degrees of freedom. A constraint gives the project manager no flexibility in that dimension, so it is plotted at the zero value on its axis. A driver yields a small amount of flexibility, so its point is plotted a bit higher than zero. Degrees of freedom provide varying degrees of latitude. They represent parameters the project manager can adjust to achieve the project’s success drivers within the limits imposed by its constraints. Connecting the five plotted points creates an irregular pentagon. The smaller the area inside the pentagon, the more constrained the project is.
Figure 2 A flexibility diagram depicts constraints, drivers, and degrees of freedom for a project.
III. **As- is : Traditional Model**

In this section, we present our case study observations of current software development process (As-is). This will identify the core issues which arise during the transitioning of a project from Business Solution Analyst handled Solution Scoping phase to Development phase (Water-Scrum) in as well as during the release phase transitioning from development to Business (Scrum-Fall).

We now focus on presenting the issues in current model.

**a. Transition phase increases cost of release one**

A major effect of the problems in transition phase to be considered is felt on the estimated date of delivery for "release one". A big reason to worry for majority of projects is "Why Does Release One Cost More? ". Let us have a look at the activities which define release one and other project activities which form a part of release one, leading to a longer time to release. Release one versus Subsequent Releases efforts is described in the Figure 3.and we can infer that-

a. Release one of a project can cost up to 10 times more than later releases.

b. Release one has a lot of overhead that is not accounted for in high level estimates.
The above Figure 3 shows development activities (in Green) that form a part of release one of project. These are specifically writing Gherkins ie. test scenarios, Developing Code and Test cases. These usually depend upon the completion of some activities from earlier project phases like Initiate and Solution Scoping (Solution Architecture, Solution Approach, etc). Since the estimates do not account for this delay in completion of activities in release one schedule of development phase, it leads to delay and also increase in cost. For new projects, most commonly infrastructure and architecture work spills over into development. According to a study by IT Architecture team at the Enterprise[18], specific problems are listed below which arise due to Water-Scrum transition:
a. DEV/CI setup often takes longer than Iteration Zero.

b. Build and deployment processes are refined in earlier iterations.

c. Application architecture is still being defined.

d. Data model has not started and will be refined during development.

e. Interfaces to other systems take a long time to be established.

The above-mentioned problems are recurring ones, seen in most of the projects inside development center. Next, we present the as-is traditional software delivery process and highlight the problems within it by referencing the Water-Scrum-Fall scenario at Enterprise.

b. **Traditional Handoff Model**

Let us now narrow down to specific problems narrated by project management team in [23], which arise during the traditional handoff process involving a linear Solution Scoping phase. As we can see during the handoff stage, detailed Solution Scoping artifacts ie. requirements and solution architecture documents etc. are provided to development team which then will analyze and build estimates using their own pre-defined process.

Below is a detailed explanation of issues from [23] in the current handoff model:

i. **Competing objectives can lead to chaos in projects during handoffs** - The Business stakeholders always like to scope out all requirements based upon their own estimates and costs, but the development team will seldom agree with the estimates of
Business and this leads to conflict in handoff stage. Business will try to squeeze in more requirements while development team will only commit on specific number of requirements which is achievable for them. Thus competing objectives lead to complications and increased friction, which will reduce collaboration among Business team and Development team.

ii. **Collaboration suffers when teams do not act like a team "collaboration over contracts"** - Ideally Business and Development team should function as "one team" to facilitate the two-way sharing of best practices. This is not the common case in many projects, leading to finger pointing during any crisis.

iii. **Lack of 3 important drivers : "Common understanding, common goals, common direction"** - This can be achieved when Business team and Development team are involved in Solution Scoping activity from start and focus should be more on making decisions with achievable target points.

iv. **Several questions and clarifications lead to rework** - The business team will hand over a detailed set of requirements to development team, which then will analyze those requirements. Since business teams do not have the expertise to give a comprehensive and complete set of requirements for the given problem, development teams will spend a lot of time in analyzing the requirements and even have to go back to business stakeholders to get clarifications. This leads to re-negotiations between business and IT until every requirement has been understood by development team and signed off to deliver. This results in a longer period of project transition phase.
and will delay the start of development activity. Thus handoffs and subsequent rework cause a lot of waste during the project initiation phase which has direct impact on project delivery schedule.

The above figure highlights the problem of waste generated in projects due to rework. The arrow going back from inspection of requirements towards solution scoping team depicts the rework done due to "back and forth" situation i.e., development team has to re-negotiate on the terms of contract in case of improper requirements specification. This will most commonly impact the schedule of release one. After analysis of the traditional model, we will present the reason behind these problems by referencing the Water-Scrum-Fall scenario at our Enterprise.

c. Water-Scrum-Fall at Enterprise
Let us focus on the actual problem areas in adoption of agile practices for development within a traditional Waterfall based model at a large-scale enterprise. This analysis was
presented by the project management team, which was responsible for initiating the new to-be process (which will be introduced later) in [23].

i. **Development teams not involved until Solution Scoping is close to completion**- The fact that Business teams responsible for Solution Scoping do not involve the Development teams nor the Application Development Leads, early enough in the phase leads to potentially conflicting project estimates as regards to Time/Schedule and Cost. It is a proved beyond doubt that the estimation accuracy of development teams is much higher as compared to estimation accuracy of business team during Solution Scoping phase. The root cause of potential conflicts in later stages of project development can be avoided if both the business and development teams are involved together in all discussions and focus is more on limiting action items and forcing decisions.

ii. **Big Requirements Up Front Approach** (BRUF)- Traditional project teams take a serial approach to development in which requirements are defined and documented early in the project, they may or may not be reviewed and accepted by the project stakeholders, and they’re provided to developers. It leads to significant wastage, thus an Iterative development model is less financially risky than serial development approach, and projects should prefer iterative approach to requirements [12].

   a. Wastage due to BRUF:
1. One major factor to be considered is how much required functionality wasn't delivered as the result of a BRUF approach.

2. The requirements change and that leads to rework.

3. People’s understanding of the requirements change.

4. People make up requirements.

Very often a BRUF approach defines the best that will be delivered by the team and then throughout the project team will negotiate downwards from there as it begins to run out of resources or time. Studies [Standish Group Survey results] show the results for analysis of problem: “Of the functionality which was delivered, how much of it was actually used?” prevalent in industry across multiple domains: an astounding 45% of the functionality was never used, and a further 19% is rarely used among successfully delivered projects.

iii. **Scoping to Iteration Zero can lead to hand-offs and confusion** - Once requirements are scoped, requirements specification document is handed over for development, which forms the contract between Business and Development team. The terms of negotiation will depend on the estimates derived by development team and clarity of requirements specified which leads to friction between Business and Development team. Any possibility of requirements creep will lead to more confusion and add up the complexity of handoff negotiations.

iv. **ADC lines are often engaged during scoping causing charges to projects when they are not delivering value to project** - Traditionally Solution Scoping
would involve elaborate scoping of all requirements leading to a time period of about 8-9 weeks for mid size projects or even extending up to about 3-6 months on large projects. The development team is usually involved during the final stages of Solution Scoping and would be responsible for "Iteration -1 " ie. team formation and preparation for development phase. One major issue is that the development team is not producing any working solution during this time, even though the project is being billed for its time. Hence this is a huge burden on project in terms of money with no potential value being added.
IV. To-be: Iterative Solution Scoping to Deployment Model

In previous section, we have covered major issues identified by project management team [23] in the traditional handoff model at the Enterprise. This section illustrates an industry approach of using iterative agile practices and structuring the observations using workflows. Precise workflow modeling of proposed iterative solution scoping process and current traditional handoff model allows us the capture of advantages and issues for the situation before and after the transition. Following are major observations:

1. We discuss the need for change of strategy through background work, in order to address the problems during handoff within the traditional linear model. Iterative solution scoping model to is a better approach. There is a conscious effort to move away from "handoffs" and implement continuous "Interactions" between the Business and Development team all the way through entire period of software delivery.

2. Next we analyze the possible solution approach to the problem of traditional handoffs, with a precise model of the Iterative solution scoping process using IDEF 0 notation. The model will provide us a platform to differentiate as-is and to-be process and analyze results in order to support the management's claim that Iterative solution scoping model in better than traditional linear model.
a. Need for Project Management strategy change

There is a need for corporations to change from a hierarchical approach to project management to being more collaborative as knowledge work has grown in importance. Project Stakeholders must have flexibility in a project in order to be able to adjust constantly to emerging challenges and opportunities. Traditional Linear project model should migrate towards an Iterative and Adaptive strategies defined by Wysocki [17]. In addition, Wysocki [17] has identified an approach which identifies project characteristics in a quadrant and then matches those to different project management strategies. Project characteristics are classified based on a certainty to uncertainty continuum for the project goal (ends) and project solutions (means). After identifying the quadrant (see Figure 6 below) that a project is in, specific, situational project strategies can be selected to match project management strategy with problem type. If there is doubt about the quadrant a project belongs to, the less risky strategy is to err on the side of choosing a higher numbered quadrant. However, depending on how the project evolves, it may be advisable to adjust the management of the project to the strategies of a lower quadrant if the project characteristics change.
A Traditional, linear strategy fits in Quadrant 1, which represents the current Delivery model at Enterprise. It consists of dependent, sequential phases with handoffs after every phase and the development phases are executed with no feedback loops until the project solution is delivered in the final phase. The newly proposed Iterative strategy consists of a number of repeated phases that include a feedback loop after a group of phases is completed. The Iterative strategy is a learn-by-doing strategy that uses intermediate solutions as a pathway to discover the details of the complete solution i.e. each iteration’s feedback adjusts the next iteration so that a solution will be converged upon.

b. Solution Approach : Iterative Solution Scoping Process

This section of case study will present the implementation strategy for the proposed solution approach. We present the to-be Iterative Solution Scoping to Deployment (ISSD) process and its features. Next we will present a workflow model of the ISSD process using IDEF 0 notation. A precise model of ISSD process aids in defining metrics for data
collection which is imperative and additional analysis that will help us improve the efficiency of ISSD process.

i. **Principles of the Iterative Solution Scoping Process**

The working of ISSD process is governed by the following principle as defined in [23]:

1. Iterative and incremental Solution Scoping approach regularly produces high quality solutions in a cost effective and timely manner via a risk and value driven life cycle.

2. It is performed in a highly collaborative, disciplined, and self-organizing manner within an appropriate governance framework, with active stakeholder participation to ensure that the team understands and addresses the changing needs of its stakeholders to maximize business value provided.

ii. **Goals of the ISSD Process**

A department within the Enterprise, which specialized in process change, initiated a project to develop and implement a Iterative process for Solution Scoping Phase that:

1. Is a flow with no wastes - Aim is to avoid rework which arises due to unclear requirements and lot of re-negotiations during handoffs.

2. Predictable- Facilitate better negotiation between the Business and the Development team by bringing justification through traceability of decisions.

3. Measurable - So that we can continuously improve it by effective feedback mechanism and use of performance metrics.
4. Is based on ESDm - As our Enterprise has adopted ESDm as standard for software delivery, the proposed process should be a "fit" within the bigger process.

5. Enables incrementally delivering: Plan to deliver what is more valuable to business than lesser value functionality, under given cost and schedule constraints.

6. A process which uses an approach that:

   a. **Forces conversations** - It increases collaboration between the Business and Development teams and meetings are more focused on decision making than generating documents.

   b. **Generates consensus** - As both Business and Development teams are involved in the solution scoping process, it helps to achieve -Common understanding, Common goals, and Common direction - between all parties involved and leads to less chaos in projects. It also helps build trust and confidence for the development team.

   c. **Breaks down silos** - Reason for breaking down silos is reducing duplication of effort. Plan is to get a better output by incorporating the mix of roles on the team (Development and Business) and deliver in a consistent manner while identifying and sharing best practices. Lastly,
breaking down silos enables expertise to be leveraged across the entire enterprise so that this process can be scaled across multiple projects.

The results of implementing ISSD process in the pilot project should reflect some of the goals defined for the process. Realization of all process goals is not observed in the initial pilot project. We will next present the result of implementing the ISSD process in the pilot project:

i. **Increased Collaboration by implementing ISSD**

The Figure 6 below depicts the Iterative Solution Scoping phase activities as been documented by the team monitoring the progress of ISSD approach in pilot project[24]. This process has led to increased collaboration between Business and Development team which results in effective project delivery. As shown in figure, individual roles from Business and Development side pair up with each other so that they are on a common ground as regards to project expectations and they can exchange ideas and share best practices.

There are 5 high level activities shown in the figure which are part of the "High Level Estimation" phase of the project where the scope of every delivery phase is decided and requirements are prioritized and distributed across different iterations of solution scoping phase. Collaboration creates one team vs. a project handing off to a line. By doing pieces iteratively, stakeholders will see working software as they are still scoping the remaining features.
ii. Impact of ISSD implementation

We now present in brief, many of the features observed as a result of implementing ISSD process in the pilot project.

a. **Evolutionary** - This process enables working in a non-linear manner, ie. at any given time the development team can start developing solution for given requirements while the Business team will start scoping out requirements for next iteration.

b. **Risk and value driven life cycle**

   a. Processes are well defined to produce visible value in the form of working software on a regular basis throughout the life cycle.

   b. Active risk mitigation early in the life cycle thereby reducing significant business risk.
c. Advantage of building a potentially shippable solution every iteration is that teams produce concrete value in a consistent and visible manner throughout the life cycle.

c. **Highly collaborative**

a. Facilitated sessions with all key members (from Business, Development, Management), where decisions are made and actions items are kept to a minimum.

d. **Cost effective and timely manner**

a. Implement functionality in priority order with the priority being defined by the stakeholders to maximize the return on investment (ROI) and thereby increasing cost effectiveness.

b. Short iterations reduce the feedback cycle, improving the chance that agile teams will discover problems early and thereby enable them to address the problems when they’re still reasonably inexpensive to do so.

e. **Reduced Waiting time for Development team during the Solution Scoping**

Adopting ISSD approach has resulted in early involvement of development team in the project life cycle. Team leads from development side are now a part of requirements analysis phase. This will change the current scenario of teams spending "idle" time without delivering value, as it takes minimal time to start development activities.
V. Modeling the Workflows

To make our case observations precise, we need a modeling notation to design a workflow which can effectively capture the process details being followed into a model which is self explanatory. This case study aims to understand and characterize at a workflow level, all of the various problems and inefficiencies that riddle agile adoption and fail to realize agile’s implementation benefits. In this section we model the workflow for Traditional process and ISSD process using IDEF 0 modeling notations. The motivation of this study to design a model, is to facilitate the use of workflow as reference to determine performance measure metrics for data analysis. Data analysis results can provide a feedback mechanism in decision making for Continuous Improvement. The data presented here is one of major contribution's of the case study.

We first mention the usefulness of IDEF0 for workflow modeling followed by workflow design of As-s and To-be process. Lastly we present data analysis results for metrics defined to analyze implementation benefits of ISSD.

a. Workflow Design using IDEF 0 notation

The IDEF0 method is designed to model the decisions, actions, and activities of a system and is targeted to communicating and analyzing the functional perspective of a system.
IDEF0 can be used to analyze the functions the system performs using defined metrics and to record the mechanisms by which these are done.

Thus we have used IDEF method to model the as-is workflow based on traditional handoff model and the to-be workflow based upon Iterative Solution Scoping model. As a limitation of this study, we have studied just a single project implementing the as-is and to-be models during its delivery life cycle. Few early releases of the project were subject to traditional linear model and 3 latest releases have been performed using Iterative Solution Scoping process.

i. **As-is project model based upon - Pilot project Releases in 2011**

ii. **To-be project model based upon - Pilot project Releases in 2012**

An activity within the given process that we want to model can be represented by a rectangular box with the left arrow in Figure 8. that shows the **Input** to the activity in box, the top arrow represents **Constraints** on the functioning of given activity, the bottom arrow represents the **Roles and mechanism** which help in functioning of activity and right bound arrow indicates the **Output** generated by activity.
Additionally, the description of the activities of a system can be easily refined into greater and greater detail until the model is as descriptive as necessary for the decision-making task at hand. The hierarchical nature of IDEF 0 facilitates the ability to construct (As-is) models that have a top-down representation. In order to represent an enterprise's process transformation (ie. To-be modeling), top-down construction is usually more appropriate. Beginning with the top-most activity, the To-be process can be described via a logical decomposition. The process can be continued recursively to the desired level of detail.

b. As is : Traditional Linear Model for Software Delivery

We model the as-is traditional linear model using IDEF 0 notations with an overlay of metrics which define 4 critical project success criteria: Time/Schedule, ROI, Value and Quality of software. Figure 8. illustrates working of as-is model for software delivery at Enterprise which is based upon the ESDm phases [Figure 1]. This model effectively describes every activity in process in terms of the constraints acting on it and the roles
which own the responsibility to complete the activity. For modeling the workflow, we show project success criteria as the constraints acting on each of the activity. The activity number will be referenced later when presenting the metrics for data collection and recommendations.

Figure 8 As-is: Traditional Handoff Model

We can correspond every activity in the model to the ESDM phase using the activity number. For understanding the transition from traditional model to iterative model we focus on activities A1, A2, A3 and A4 which correspond to Solution Scoping, Design, Development and Implementation phase respectively. The figure depicts the time taken (around 8-9 weeks on average) to initiate a project and time taken (around 20 weeks on average) for first major release of the project using traditional model. Project transition from one phase to another is linear in nature. ie. only after completion of all work within a phase, the project responsibility will be transferred to next phase.
c. To - be : ISSD Model

We present a meta model for the proposed ISSD process first and this will be logically decomposed to show more details. The model here is a high level IDEF 0 diagram which depicts project success criteria and constraints being applied to ISSD process. We can measure the same metrics applied to the as-is model using the given meta model.

![ISSD Meta Model Diagram](image)

Figure 9 ISSD process - High level Model

The above figure, highlights iterative nature of ISSD process through a feedback loop from the output to input. The nature of ISSD approach is to scope out requirements in
iterative phases and speed up the development activity leading to more frequent releases. This serves the purpose of ISSD, which is to facilitate minimal time to start development and provide a frequent and consistent feedback loop. Thus the iteration feedback can be used as negotiation factor by the stakeholders when they will be processing requirements for a new phase. Next we decompose the Figure 9 into a IDEF 0 diagram depicting detailed activity flow inside the model from Analysis phase to Release.
Figure 10 To-be model: ISSD workflow with decomposed activities
Figure 10. depicts the hierarchical decomposition from a meta model ie. Activity "A0n" into a detailed workflow model for ISSD process ie. Activity "A0" and further illustrates Solution Scoping workflow by expanding activity A1n into A00.

Described below is the working of ISSD process as referenced from [23]:

According to the process documented in [23], ISSD model comprises of 4 major phases of software delivery - Solution Scoping - A1, Design - A2, Development - A3 and Implement - A4. Once a project "Initiate" phase has completed and project plan is handed over to Business team responsible for scoping the solution, marks the start of ISSD model. Business team and Development team are involved in a series of meetings during the High Level Estimation phase which is a part of A1 phase, with a plan to estimate the number of Iterations of ISSD that will be sufficient to accommodate all of the requirements. Scope change can be accommodated between iterations. Hence it can adapt to changing business requirements.

Next is a series of meetings - i) Tech Alignment ii) Scope Alignment which focus on decision making and limiting action items for individual phases of solution scoping. No time is wasted on preparing elaborate documents to note the requirements as this is a highly collaborative and interactive model.

The development team will immediately start working on 1st set of requirements and meanwhile the business team will continue working on scoping requirements for phase 2. As soon as the development team will hand over the software for testing phases they can start development for phase 2 given that requirements for phase2 are ready from Analysis
side. Each iteration feedback adjusts next iteration so that solution will be converged upon.

d. Metrics to differentiate As-is and To-be model

We next identify data collection points within the ISSD process which can be effectively used in future to measure project performance. The workflow models provide us a efficient framework for metrics identification. The details of workflow model depicted by IDEF notation are effective tools to identify the performance measure metrics and subsequent data collection points. Once data can be collected at start of iteration for planned release ie. when business team will hand over first set of requirements, next data collection can happen at end of development when the development team will hand over software for testing. The analysis of data collected can be then used to show the variance between the estimated performance and actual performance of current release. This feedback will serve as a input to the Solution Scoping activity of next phase and the differentiating factors form the negotiation points between business team and development team during the next scheduled interaction.

We now present survey data and analysis results to illustrate difference between as-is and to-be models. Objective of this survey interviews was to gather data to measure the performance of Iterative Solution Scoping strategy being implemented in ADC as an initiative to reduce the Business-IT alignment issues.
i. **Survey Results used to differentiate as-is vs to-be model**

The design of the interview consisted of five parts. In the first part of the interview, an introduction of the study’s goals was provided. Furthermore, the interviewees were informed why they had been selected for the interview. It was also made clear that they were selected specifically as they were a part of pilot project. In the next part, the interviewees were asked for their experience and background regarding work in agile development approaches in particular.

First part of Questionnaire focused on assessing -

1. **Use of project success criteria metrics to assess impact of ISSD process**

A survey was conducted at Enterprise, to gauge the user feedback on the priority they assign to the project success criteria's [24]. These questions which are based on [25], will try to understand the user perception of what he thinks is more important in ensuring success of a project and we will summarize the highest priority or highest chosen activities within each criteria. Next we ask the same set of interviewees about the possibility of ISSD adoption being able to achieve those project success criteria that they have identified in earlier section.

The results of survey indicate that:

i. **Time/ Schedule Metrics:** 58% believe that delivering on schedule is more important than delivering when the system is ready

ii. **Return on Investment (ROI) Metrics:** 70% believe that providing the best ROI is more important than delivering under budget.

44
iii. Value Metrics: 83% believe that meeting actual needs of stakeholders is more important than building the system to specification

iv. Quality Metrics: 82% believe that delivering high quality is more important than delivering on time and on budget

We now present results of second set of questions to gauge if the ISSD process being implemented in the pilot project is able to match up the project success criteria in terms of its performance.

Results:

1. Almost 70% of people feel Iterative Solution Scoping process does meet the time/schedule requirements.

2. Almost 60% people feel Iterative Solution Scoping process does provide effective return on investment (ROI).

3. Almost 90% of people feel Iterative Solution Scoping process does have ability to deliver a solution which meets the actual needs of stakeholders.

4. Almost 50% people feel Iterative Solution Scoping process does provide high quality solutions.

Analyzing the results we can observe that

1. We need to identify a large data set of metrics to assess the implementation impact of ISSD.

2. The impact of ISSD was not felt at the lower level in project hierarchy ie. developer or tester level.
3. We cannot establish based upon just qualitative analysis, the true value of ISSD adoption.

ii. Metrics for Individual Phase Analysis

Based upon observations and project data obtained for different releases of pilot project, we would like to assess the performance of individual phase of project and use the result to compare as-is vs to-be model.

1. Data Sources

<table>
<thead>
<tr>
<th></th>
<th>Analysis Started</th>
<th>Development Started</th>
<th>Implementation Started</th>
<th>Total Hours</th>
<th>Features</th>
</tr>
</thead>
<tbody>
<tr>
<td>Release 1</td>
<td>1-Jan</td>
<td>1-Feb</td>
<td>9-May</td>
<td>1490</td>
<td>2</td>
</tr>
<tr>
<td>Release 2</td>
<td>13-Feb</td>
<td>13-Mar</td>
<td>6-Jul</td>
<td>5330</td>
<td>5</td>
</tr>
<tr>
<td>Release 3</td>
<td>1-May</td>
<td>1-Jun</td>
<td>7-Sep</td>
<td>2834</td>
<td>3</td>
</tr>
</tbody>
</table>

Table 1 Project release data for pilot project

Project data for multiple releases of pilot project with count of number of hours per release as well as effective Lines of Code (SLOC) per release:
<table>
<thead>
<tr>
<th>Model</th>
<th>Effective SLOC</th>
<th>C&amp;T Effort</th>
<th>SLOC/ Hour</th>
</tr>
</thead>
<tbody>
<tr>
<td>ISSD</td>
<td>72641</td>
<td>1490</td>
<td>48.75</td>
</tr>
<tr>
<td>Traditional</td>
<td>16139</td>
<td>4561</td>
<td>3.53</td>
</tr>
<tr>
<td>Traditional</td>
<td>47785</td>
<td>2930</td>
<td>16.3</td>
</tr>
<tr>
<td>Traditional</td>
<td>135023</td>
<td>6990</td>
<td>19.31</td>
</tr>
</tbody>
</table>

Table 2 Time and SLOC data for multiple releases of pilot project

Results of individual phase analysis:

<table>
<thead>
<tr>
<th>Project Phase</th>
<th>Factors affecting phase</th>
<th>Traditional Model</th>
<th>ISSD model</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initiate phase</td>
<td>Domain Complexity, Technical Complexity, Agreement between stakeholders</td>
<td>Based on given data, average time for Analysis was around 8-9 weeks or even 3-6 months.</td>
<td>Based on given data, average time taken to complete analysis phase was about 4 weeks.</td>
</tr>
<tr>
<td>Development phase</td>
<td>Velocity of team, Resources, Requirements Clarity</td>
<td>The first 3 releases of pilot project, the hours are much higher and LOC is lower.</td>
<td>For latest release of pilot, the LOC is high and the hrs are low.</td>
</tr>
</tbody>
</table>

Table 3 Individual phase analysis
The above figure illustrates the results from previous section regarding time required for various phases of the project involving two different approaches ie. traditional and ISSD. This figure shows that business can achieve break-even point early in the project as they will get most valuable features delivered early in project cycle.

We now illustrate the difference in output of team which followed traditional method first and later switched to ISSD process. This chart is based upon Table2.
Result analysis in Figure 13 is based on Table 2 and Figure 14. The effective SLOC/hour is much higher for the release which had adopted ISSD model as against the earlier releases which were based upon Traditional model.

It was observed that the LOC/hour count for the release cycle that adopted ISSD approach was significantly higher than other releases which adopted traditional method of development. We summarize the reasons for higher LOC count:

1. The iterative scoping process allowed team to quickly get through scoping thus causing less hrs.

2. The iterative process allowed the line to focus on small pieces, so even though the coding lines were high, it did not take as much time because the line had a focus on a small set of features compared to a massive backlog.
3. Feedback loops were constant in this process thus causing re-work to minimal.

![SLOC per hour for different releases](image)

Figure 13 SLOC per hour for different releases

iii. Total time for project delivery based upon data input

We have earlier highlighted the time schedule metrics on As-is and to-be models and presented the time required for every phase. We can now calculate total time for every release of the project and compare the timings across as-is and to-be processes. These results can be extrapolated to provide more accurate estimation of subsequent duration of project iterations and can be really helpful in release planning process.

For a Traditional Model:

By referring to Figure 9. we can formulate total time for release of a given project as (using IDEF activity notation corresponding to phases)
Total time for 1 Release(T) = Time for Initiate (TA0) + Time for Analysis(TA1) + 
+ Time for Design(TA2) + Time for Development (TA3) 
+ Time for release (TA4)

TA2 + TA3 = Idle time spent by Development team while waiting for Solution Scoping to get over + Time required to design and develop code and send to release stage.

For a ISSD Model:

We refer to Figure 15. to formulate total time for 1 release of project,
Here we define time required for 1 release following ISSD approach. We have assumed that multiple stages of solution scoping, design and development form a release. Hence we use "n" to highlight the multiple iterations.

Total time for 1 Release(T1n) = Time for Initiate (TA0) + Time for Analysis(TA1n) 
+ Time for Design(TA2n) + Time for Development (TA3n) + Time for release (TA4n)

The equations need to be validated by actual project data. Although now we can appropriately conclude that total time required for completion of project following ISSD is less than project following traditional approach. This is one of the biggest value addition of ISSD approach. One of estimates by a Project Manager at the Enterprise say the project will stand to save around 2-3 months when it will implement ISSD process.
The figure describes a working of ISSD process with project schedule overlay for 3 releases. Activity A1 from Figure 1 is decomposed into sub-activities A11, A12 and A13, each leading to release after A2 and A3 are executed. Activities A2n, A3n and A4n depict the development team work which starts after analysis done in A1n (N=1,2,3.. represents a release cycle).
VI. Observations

The discussion is structured around the reported consequences of the implementation of ISSD process in [23]. In the previous section, we presented model for To-be process and defined metrics for the same. Now we will present key observations and recommend metrics which can be applied to To-be model. It was found in[23] that the iterative nature of the software development approach had major impact on the project execution. In the case it was found that the iterations provided the project management with a smaller control unit that enabled project management to divide the project execution in smaller pieces of work that could be better controlled.

The study in [23] reported that the explicit prioritization and mitigation of risks in ISSD in the early phases increased predictability of the project. Also we observe from [23] that the use of the iterations as the control unit for the project execution enabled the project management to plan the project execution on an iteration basis, thus making it easier to incorporate changes to the process (requirements and artifacts) and the project execution.

One of our first observations is that the team velocity (number of story points completed in given time) goes on increasing with successive iterations. This attribute can be effectively applied to differentiate As-is vs To-be model or can be used to quantify the benefits of using ISSD mode. Next we present Qualitative Assessment of the impact of
ISSD model on team members of pilot project. We conducted a survey for understanding the feedback from development team after it adopted ISSD process.

a. Using Velocity to measure the Productivity of an Agile Development Team

Teams can use velocity to help track the relative productivity of the same team from one iteration to another assuming the value of a point stays the same over time. As velocity measures the quantity of work being delivered by a team, if velocity increases, then the team is delivering more, which leads to increased productivity. One might argue that the team could be producing more work at lower quality –however, this is somewhat protected by the fact that we only count Story Points towards a team’s velocity if they are accepted by the Product Owner.

b. Qualitative improvements due to ISSD Workflow

A survey was designed to analyze the Qualitative impact [25] of ISSD process on the development team as regards to project performance measures. We have structured the interview to gauge the user perception about effectiveness of ISSD to impact pre-defined measures. The intention of questions was to check how effective agile teams are in practice, so we asked about productivity, quality, stakeholder satisfaction, and cost. We summarize the results below using visual representation. As seen in results, pilot team reported significant improvements in productivity, quality, and stakeholder satisfaction,
and reasonable improvements in cost. We now present the results with approximate percentage values. For survey questions please refer to Appendix A.

i. How has ISSD Approach Affected Team’s Productivity?

The figure below summarizes pilot project team members response as regards to ISSD being able to impact team's productivity in positive way. Few of the responses were from members at lowest level of hierarchy who did not have a clear idea of the impact. Hence we see large number of users saying impact was "Somewhat higher".

![Figure 15 Impact on Team's productivity](image)

ii. How has ISSD Approach Affected Quality of Software produced

The figure illustrates that maximum number of users are not clear how precisely ISSD affected quality of code produced. This is dependent on number of factors like defects found, re-work done etc.
iii. How has ISSD Approach Affected Cost of Software Development

This question was difficult to answer for team members as it was more specific towards the Project Management team. Hence we got a lot of response saying that users were not aware how ISSD adopting affected project cost. Project Manager would be the most ideal person to give an precise answer to this question.
iv. How has ISSD Approach Affected Business Stakeholder Satisfaction

This question can also be said to be targeted towards Project Management, but still we decided to take a opinion of team members. Team members have a rough idea of how satisfied their stakeholders are based upon the feedback they get consistently. This has resulted in maximum users stating that ISSD has resulted in their stakeholders being happier than usual.

Figure 17 Impact on Cost of Software Development
Observations from Survey Questionnaire:

The impact of process transition from traditional to iterative was not really seen at developer or tester level as the work they did earlier did not change at all. This is because for the developers and testers point of view, they still receive a requirements document from the analysis team, which they need to develop. A significant change would be surely experienced by the Project Management team responsible for requirements specification.

c. Timing of Value Delivery

In the case study it is observed that a different timing in value delivery was related to the iterative development approach. In particular, study reported [23] that workable software can be delivered earlier and on a more frequent basis. However, delivering software iteratively also brings along costs in terms of extra effort in terms of getting the feedback from user representatives.

Figure 18 Impact on Business Stakeholder Satisfaction
We will move on to presenting our first recommendations for ISSD process, which are based upon the detailed observations made in comparing the performance of 2 process models. Firstly we recommend that performance metrics be defined to "Estimate progress of development team ". We recommend that metrics to "measure the operational costs/savings associated with improved efficiency" should be defined, to be referenced by Project Managers. This section holds importance as it is meant to be persuasive enough to convince the audience that implementing ISSD will benefit their respective teams.

d. Estimating progress of development team

We use the modeled workflows to identify points of measurement and metrics to be used as feedback from one release (workflow activities A1n to A3n) to identify pro-active actions for subsequent releases (e.g. workflow activities A1n+1 until A3n+1) and aid in decision making for Continuous Improvement.

To achieve this, we would like to define a set of metrics based upon the EVM (Earned Value Management) strategy[14], which integrate the areas of technical performance, schedule and actual cost to provide metrics for work actually accomplished. The metrics will provide effective program management technique that uses “work in progress” to indicate what will happen to work in the future. This gives managers greater early insight into potential risk areas and with a clearer picture allowing managers to create risk mitigation plans based on actual cost, schedule and technical progress of the work.

By comparing the earned value (EV) with the planned value (PV) the actual progress on the project is compared against the expected progress, which yields valuable information. Using this information, metrics assessing cost efficiency and schedule efficiency are
calculated. Metrics forecasting the expected cost to complete a project and total expected
cost of a project based on past project performance are derived. The iteration burn-down
and burn-up charts do not provide at-a-glance project cost information and the metrics
defined here will therefore be a excellent extension to the information provided by burn-
down charts.

Planned Value is the value of the work planned for a certain date. It is the entire budget
for work to be completed at the planned date. ie. it is the sum of the estimated feature
sizes for all the features up until the planned date.

Earned Value is the value of work completed at the same date as used for PV. Earned
Value is the sum of the estimated story points for the features up until the calculation
date. Actual Cost is what the name implies: the cost in dollars to complete a set of
features.

EV, PV  are computed based on measurements at the start of each A1n in the workflow
[Figure 9]. We are using three measurements points to establish the initial baseline:

i. The number of planned iterations in a release;
ii. The total number of planned story points in a release;
iii. The planned budget for the release.

In order to calculate the metrics, there are four measurements needed at activity A3n:

i. The total story points completed;
ii. The number of Iterations completed;
iii. The total Actual Cost;
iv. The total story points added to or removed from the release plan.

Apply metrics to release one for Pilot project: In our example, after release one, we should be at 25% complete and assuming our total budget for project is $ B1. Planned Value for a given Release is the Expected Percent Complete multiplied by the Total Budget:

$$ PV1 = \frac{25}{B1} \times 100 $$

Actual Percent Complete equates to the total number of story points completed divided by the total number of story points planned:

$$ APC = \frac{\text{Story points completed in Release1}}{\text{Total Story points planned for project}} \times 100 $$

Earned Value is calculated by multiplying Actual Percent Complete by the Total Budget:

$$ EV1 = \frac{APC}{B1} \times 100 $$

Comparing the current release plan against actual work performed:

Variance factor = $EV1 - PV1$
The **Cost Performance Index** (CPI) gives a measure of efficiency. It shows how efficiently team is actually spending the allocated dollars compared to how efficiently it planned to spend them. It is calculated by dividing Earned Value by the Actual Cost.

\[ \text{CPI} = \frac{\text{EV1}}{\text{Total Cost for Release}} \]

A CPI of 1 indicates spending of budget to accomplish work at the rate that you had planned to spend it. A CPI less than 1 means team is over budget i.e. spending budget less efficiently than planned because Earned Value is larger than the Actual Cost.

The **Scheduled Performance Index** (SPI), which gives a measure of project delivery accuracy, compares Earned Value with Planned Value and is calculated by dividing the Planned Value into the Earned Value:

\[ \text{SPI} = \frac{\text{EV1}}{\text{PV1}} \]

<table>
<thead>
<tr>
<th>Release</th>
<th>Total Budget</th>
<th>Planned Value</th>
<th>Earned Value</th>
<th>Actual Cost</th>
<th>CPI</th>
<th>SPI</th>
<th>Estimate at Complete</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$300k</td>
<td>$150k</td>
<td>$150k</td>
<td>$150k</td>
<td>1</td>
<td>1</td>
<td>$300k</td>
</tr>
<tr>
<td>2</td>
<td>$1,000k</td>
<td>$575k</td>
<td>$500k</td>
<td>$625k</td>
<td>.8</td>
<td>.86</td>
<td>$1,250k</td>
</tr>
</tbody>
</table>

Table 4 Sample Data for project depicting EV,PV and variance
e. Calculating the ROI of Implementing Agile Practices – Agile increases output and efficiency

One of the many benefits of developing software in an iterative way is that Business can start realizing the benefit of development work before the project is officially ‘finished’. The objective is to front-load the project with the most profitable or highest value deliverables and release them as soon as possible so that they can start making money for Business.

i. Metrics to measure the operational costs/savings associated with improved efficiency

These metrics [26] can be applied to One Iteration of project, which followed Traditional Model and second Iteration data of project, which had adopted Iterative Solution Scoping process.

1. Calculate the total weekly cost of development team (sum of all members of the Development team including employers costs). Assume for pilot project, the standard per hour rate of work is $68.
2. Calculate the total cost per Iteration.
3. Calculate the total cost per velocity [Velocity is a measure of how much work a team can complete in a set period of time and is measured in Story Points] point for each Iteration over time (cost per Iteration /velocity point). Calculate the cost savings associated with the reduced cost per sprint i.e. how much would you have had to pay last sprint in order to achieve the current velocity.
4. Calculate the cumulative cost savings over time.

<table>
<thead>
<tr>
<th>Total Annual Cost</th>
<th>Total Weekly Cost</th>
<th>Sprint Duration (weeks)</th>
<th>Sprint Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>$ X</td>
<td>$ 40<em>68</em>y</td>
<td>2</td>
<td>5440*y</td>
</tr>
<tr>
<td>0</td>
<td>2</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

Table 5 cost per sprint

\[ y = \text{no of resources} \]

<table>
<thead>
<tr>
<th>Iteration Number</th>
<th>Velocity</th>
<th>Cost Per Point</th>
<th>Financial Return</th>
</tr>
</thead>
<tbody>
<tr>
<td>Iteration 1</td>
<td>12</td>
<td>72</td>
<td>0</td>
</tr>
<tr>
<td>Iteration 2</td>
<td>14</td>
<td>62</td>
<td>144</td>
</tr>
</tbody>
</table>

Table 6 Financial measure of efficiency

**Financial Return** = (Velocity of Iteration2 \* Cost/point of Iteration1) - (Velocity of Iteration2 \* Cost/point of Iteration2) ... Assumption will be that velocity increases over time thus reducing the cost per point over time.

If you want to measure the change from pre-ISSD to post-ISSD implementation then:

Ask the team to retrospectively assign story point estimates to a ‘iteration-worth’ of work delivered pre-change e.g. if you’re now running two-week sprints, have the team estimate the work achieved in the two weeks prior. This can actually be used to help establish team velocity in the first place. Using team and stakeholder feedback after comparing the data, it can be used as sufficient evidence to prove the point.

Based upon the data collected by analyzing the Financial gains over multiple iterations we can plot a graph which depicts that savings go on increasing in Iterative model as iterations go forward.
The figure below shows the increased savings over successive iterations:

![Graph showing increased savings over iterations](image)

Figure 19 Operational Savings as in [26]

In the next part of this section, we present the observations of impact caused by ISSD process in terms of value addition to project.
VII. Conclusion

This case study investigates the effect of introducing iterative and agile practices in an organization that has been working in a waterfall driven way. Thus the exploratory study is aimed at identifying the advantages, disadvantages and risks of implementing iterative solution scoping process for software development. To this end, we collected information of the introduction of the process in a large company and defined the ISSD model for same. This led to a series of relationships between characteristics and consequences of ISSD model. This research shows that ISSD has a number of advantages, disadvantages and risks.

The ISSD approach has a positive impact on the project execution control[23]. The activities performed during the inception and elaboration phases increase the predictability of the project execution by explicitly addressing business and technical risks. ISSD enables the development team to deliver value earlier and on a more frequent basis as a consequence of the iterative solution scoping approach. This also had an impact on the amount of feedback that the development team receives from the customer, which helps the team to adjust the software to better match to the customer’s expectations.

Thus, in summary the main improvements identified on basis of feedback from project management review [23] are -
1) ability to increase release frequency and shorten requirements lead-times; 2) significant reduction waste and better reflection of the current customers’ needs measured as reduced number of change requests; 3) improvements in software quality for basic testing(unit and component testing) and overall system quality, and 4) improved communication which facilitates better understanding and allows to reduce documentation.

To summarize the impact of ISSD, based upon the pilot project team review document we can say, ISSD adoption which leads to:

i. Thorough Discussions: Improved decision making in Solution Scoping meetings.
ii. Line Involvement: Increased collaboration between business and development teams
iii. Faster Turnarounds: Working software released to customer in every release
iv. Issues are addressed quickly: short feedback time ensures early issue resolution
v. Wonderful business support: stakeholders get early benefits from project
vi. No more hand-offs: Interactions have replaced handoff and its fallacies

Risks & Opportunities

Some of the risks are related to the introduction of iterative solution scoping process which are identified in [23]. Introducing ISSD is a an organizational change in which
business processes have to be adapted, and the people and organizational issues have to be actively dealt with. In particular, the most recurrent people and organizational risk is related to how stakeholder involvement is managed.

Studies reveal few risks and improvement opportunities in the working of iterative solution scoping so that it can be scaled across multiple projects:

1. Code Management is important – team can be working multiple releases

2. Further proves the need for Accepted Test Driven Development – lines should be doing this to see full benefit of iterative scoping

3. Architecture impacts – currently being analyzed by the architecture team in the black belt project
VIII. Future Work

This case study – based research hints at some interesting research opportunities in several aspects. In future work, more qualitative as well as quantitative studies are needed to compare development models for large-scale development.

One aspect is to further understand is how different factors that influence the outcomes of adopting ISSD model interact with each other. Finally, this research also elucidates a series of risks involved (in terms of progress measurement and stakeholder involvement), which provide an opportunity for researching how to extend iterative solution scoping process, in order to overcome its limitations.

This case study will encourage to build a process which is:

**Scalable** - Scale the ISSD process to multiple projects inside the Development center.

**Predictable** - Standardize the adoption of ISSD process to make the process implementation results more predictable.

**Measurable** - Define measurement criteria for the process so that we can continuously improve it.
References


17. **Wysock Robert K.** Effective Project Management: Traditional, Agile, Extreme [Book]:

18. Nationwide proprietary Document- Architure Adoption Document


25. Scott Ambler’s February 2008 Agile Adoption Survey posted at www.agilemodeling.com/surveys/
Appendix A : Survey Questions

Measuring the performance of Iterative Solution Scoping Strategy

a. Introduction

Goal of the study:

1. Survey results will be a part of my Thesis under the Masters program in Computer Science Department. My thesis research revolves around the BSA-ADC issues during the transitioning phase of a project from Solution Scoping phase to Design phase.


3. Objective of this Survey is to gather data to measure the performance of Iterative Solution Scoping strategy being implemented in ADC as an initiative to reduce the BSA-ADC alignment issues.

4. Data analysis performed will help us quantitatively establish the superiority of Iterative Scoping strategy.
b. Personal Data

Personal Information asked to gauge the experience level of individual and decide the overall diversity of interviewees.

1. Which best describes your current position?
   i. Business Stakeholder
   ii. Data Professional
   iii. Developer
   iv. IT Manager
   v. Project Manager
   vi. QA/Test
   vii. Other

2. How many years of work experience do you have?
   i. None
   ii. Less than 2 years
   iii. 2 to 5 years
   iv. 5 to 10 years
   v. 10 to 20 years
   vi. 20+ years

3. Number of IT people in your team
   i. 1 to 10
   ii. 11 to 50
   iii. 51 to 100
   iv. 100+
c. Survey 1: Questions for this Iteration

Project Success Criteria

1. Which do you understand to be more important?
   i. Delivering on or under budget
   ii. Providing the best return on investment (ROI)
   iii. Both are equally important
   iv. Neither are important

2. Which do you understand to be more important?
   i. Delivering on time according to the schedule
   ii. Delivering when the system is ready to be shipped
   iii. Both are equally important
   iv. Neither are important

3. Which do you understand to be more important?
   i. Building the system to the specification
   ii. Meeting the actual needs of stakeholders
   iii. Both are equally important
   iv. Neither are important

4. Which do you understand to be more important?
   i. Delivering systems on time and on budget
   ii. Delivering high quality, easy to maintain systems
   iii. Both are equally important
   iv. Neither are important
d. Survey 2: Iterative Solution Scoping Performance

1. When it comes to time / schedule what is your experience regarding the effectiveness of iterative solution scoping development process?
   
i. Effective
   
ii. Neutral
   
iii. Ineffective
   
iv. Very Ineffective

2. When it comes to effective return on investment (ROI), what is your experience regarding the effectiveness of iterative solution scoping development process?
   
i. Effective
   
ii. Neutral
   
iii. Ineffective
   
iv. Very Ineffective

3. When it comes to ability to deliver a solution which meets the actual needs of its stakeholders, what is your experience regarding the effectiveness of iterative solution scoping development process?
   
i. Effective
   
ii. Neutral
   
iii. Ineffective
   
iv. Very Ineffective

4. When it comes to the quality of the system delivered, what is your experience regarding the effectiveness of iterative solution scoping development process?
   
i. Effective
   
ii. Neutral
   
iii. Ineffective
iv. Very Ineffective

e. Survey 3: Stakeholder Satisfaction

1. What strategies does your team follow to provide value to your stakeholders?
   i. We are producing working software every iteration during construction phase.
   ii. There were one or more iterations at the start of the project where we did not produce working software due to project initiation considerations.
   iii. At the start of the project we identified our key stakeholder groups and their goals and we have a definition of what needs to be done.
   iv. We are implementing improvements to the business process.
   v. Other

2. What strategies does your team follow when working with your stakeholders?
   i. We demo the solution to stakeholders every iteration/sprint during construction.
   ii. We have a written requirements specification which defines what we need to deliver.
   iii. Stakeholders work with business analysts who provide requirements to our product.
   iv. We did some initial requirements envisioning with our stakeholders to identify the scope and to populate our backlog at the start of the project.
   v. Other
f. Qualitative Analysis

1. Has your team adopted Iterative Scoping method?
   i. Yes
   ii. No

2. How has Iterative Scoping approach affected your productivity?
   i. Much higher
   ii. Somewhat higher
   iii. No change
   iv. Somewhat lower
   v. Much lower
   vi. Don't know

   Explain Why (in short):

3. How has Iterative Scoping approach affected the quality of the systems produced?
   i. Much higher
   ii. Somewhat higher
   iii. No change
   iv. Somewhat lower
   v. Much lower
   vi. Don't know

   Explain Why (in short):

4. How has Iterative Scoping approach affected the cost of development?
   i. Much higher
   ii. Somewhat higher

79
iii. No change
iv. Somewhat lower
v. Much lower
vi. Don't know

Explain Why (in short):

5. How has Iterative Scoping approach affected stakeholder satisfaction?
   i. Much higher
   ii. Somewhat higher
   iii. No change
   iv. Somewhat lower
   v. Much lower
   vi. Don't know

Explain Why (in short):
g. Survey 4: Iterative Scoping Process Feedback

<table>
<thead>
<tr>
<th>Statement</th>
<th>Strongly agree</th>
<th>Agree</th>
<th>Neutral</th>
<th>Disagree</th>
<th>Strongly disagree</th>
</tr>
</thead>
<tbody>
<tr>
<td>Iterative Scoping doesn’t support sufficient Documentation</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Iterative Scoping doesn’t sufficiently address architecture issues</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Iterative Scoping doesn’t sufficiently address analysis</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Iterative Scoping doesn’t plan sufficiently</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Iterative Scoping based software development is undisciplined</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Iterative Scoping is only for small projects as large projects are difficult to govern</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>