Predicting Outcomes for Big Data Projects: Big Data Project Dynamics (BDPD)

Research in Progress

David K. Becker
David Becker - Systems Analysis and Consulting
Beavercreek, OH
dbecker2017@outlook.com

Abstract— **The number and importance of Big Data projects is increasing, but unfortunately, a large proportion of Big Data projects are failing. The ability of organizations to manage these projects has so far not kept pace – they need better ways to analyze the behavior of their Big Data projects and positively affect outcomes. The objective of this paper is to identify the important dynamic characteristics of Big Data projects, and explore how a modeling and simulation technique called system dynamics (SD) can be applied these characteristics. The approach draws from applicable concepts in the domains of traditional project management, Agile software development and Lean product development, and proposes to develop a model called Big Data Project Dynamics (BDPD) incorporating these insights. The BDPD model is organized into sectors: Core Rework Cycle, Iterative & Incremental, Exploration & Learning, Economic Value, and Policy Actions & Consequences. Given ADPD, practitioners can simulate Big Data project behavior from initial conditions, and probabilistically predict project outcomes.**

Keywords: Big Data Project, System Dynamics, Prediction, Modeling & Simulation, Project Management

I. INTRODUCTION

As documented by numerous industry analysts and studies, Big Data is becoming increasingly important to various organizations as they seek to gain advantage in a highly competitive and challenging operational environment.

Spending on big data initiatives is growing rapidly based on the promise of yielding new insights, making better decisions, and optimizing business processes. Originally, organizations thought they could achieve these objectives very simply using Big Data tools and techniques because they were so easy to assemble and get up and running.

However, it quickly became obvious that the complexity of the data, tools, techniques and resources involved required some level of project management for the organization to take full advantage of Big Data. They had to establish a project to assemble the necessary collection of data, tools, and resources, and then direct this assemblage toward a specific goal to be accomplished. Unfortunately, even with some level of project management, Big Data projects have still been experiencing very high failure rates.

One way for dealing with the management of Big Data projects is a technique called systems dynamics (SD). This paper attempts to show how SD might be applied to the management of Big Data Projects.

II. BACKGROUND

A. What is a project?

What is a project? According to the Project Management Institute (PMI) a project can be defined as: "a temporary endeavor undertaken to create a unique product or service" [1]. A project:

- Consists of a series of activities or tasks
- Has a specific objective (scope)
- Is expected to be completed within certain specifications (requirements)
- Has defined start and end dates
- Has funding limits
- Consumes and/or utilizes resources

One major category of projects is those dealing with what is called information technology (IT) or systems engineering (SE). This category includes projects comprising the orchestration of people, processes and technology to develop, maintain and use systems involving computer hardware, computer software, data storage, input, output and telecommunications. IT projects include software or application development. One subset of IT projects is called "data intensive projects" – projects involving software applications that are limited less by raw CPU power than by three other key constraints: "the amount of data, the complexity of data, and the speed at which it is changing" [2].

A Big Data project can be considered a data intensive project which has very large-scale issues with the three constraints (which for Big Data are called the three V's – Volume, Variety, and/or Velocity). Within Big Data projects a further distinction can be drawn between whether the project is dealing with:

- Big Data infrastructure – technology development initiative that cuts across one or more of three dimensions [3]: time (affecting multiple releases of a system), scope (affecting multiple systems), or organization (affecting multiple teams or programs)
- Big Data applications – development of a Big Data system for a particular analytical task
- Big Data analytics – the analysis of the data

For the purposes of this paper, it makes sense to consider Big Data infrastructure and application projects as traditional IT projects, while Big Data analytics projects typically employ discovery techniques. These two approaches may require somewhat different ways of thinking about project management.

B. What does it mean for a project to fail?

Many software engineering projects suffer from outdated or ineffective systems engineering and acquisition processes that often result in failures:

- Cost overruns
- Schedule slippages
- Deficiencies in delivered capability
- Catastrophic program failure and cancellation

The 2014 Standish Group Chaos Report stated that project statistics are bad and getting worse [4]:

- 16% – Project Success (cost/schedule/performance objectives met)
- 53% – Project Challenged (cost/schedule/performance shortfalls)
- 31% – Project Impaired (cancelled)

C. How about Big Data Projects?

A study by Price Waterhouse and Iron Mountain [5] found that, while 75 percent of business leaders feel they are making the most of their information assets, in reality only 4 percent are set up for success. Overall, 43 percent of companies surveyed obtain little tangible benefit from their information, while 23 percent derive no benefit whatsoever.

Here are some of the definitions of failure for Big Data Projects [6]:

- One common failure is a perpetual scope-creep of the solution. The project just never seems to end
- Frequently the Big Data solution does not result in surprising or useful information
- Other times the Big Data project fails to produce actionable results, used in decision support for the organization
- In many cases, the project is simply abandoned, and the organization just walks away from it

According to an article from Gigaom, 55% of Big Data projects fail [7]. That is an extremely high number. Furthermore, Gartner predicts that 2017 will see 60 percent of Big Data projects fail. They won't go beyond piloting and experimentation phases, and will eventually be abandoned [8]. Furthermore, as Crawford states [9]:

> "Various sources report that 65-100% of Big Data Analytics projects fail:
>
> - Incomplete
> - Out of time
> - Over budget
>
> Sources reporting less failure than 65% include Proofs of Concept (POCs), Proofs of Technology (POTs), Proofs of Value (POVs), and pilots in this estimate."

Not all projects problems will display all of these symptoms. Some problems may become more severe at different stages in the project, particularly toward the end.

D. Reasons for Failures

According to Philip, a key part of the problem is that too many companies jump into projects without really understanding why they're using Big Data and what they want to achieve. It goes to illustrate the enigma that Big Data can be for many companies who don't take the time to understand its capabilities. [10]

A survey of commentaries written by 19 different Big Data practitioners and industry analysts listed a total of over 100 different causes behind the large rate of failure being experienced in Big Data project efforts [11]. The causes fell into 17 major clusters or areas of concentration. Listing these areas in decreasing order of frequency as cited by the experts:

Wrong/Inadequate Skills	14
Incorrect Business Objectives	13
Insufficient ROI/Business Case	9
Technology Complexity	8
Data Integration	8
Data Management	8
Management & Cultural Resistance	7
Improper Scope	7
Incorrect Project Structure	6
Inadequate Management & Governance	6
Enterprise Strategy Match	5
Poor Communication	5
Incorrect Use of Technology	5
Technology Architecture & Infrastructure	5
Problem Avoidance	2
Technology Change	1

Table 1 – Causes of Project Failure

Not all causes will occur on all projects. Some causes may become more pronounced in different project stages. Actual project behavior is very dynamic and non-linear, and trying to infer causes from this behavior can be challenging.

III. METHODOLOGY

The method described in this paper is to develop an SD model of the structural dynamics of Big Data project management that drives Big Data projects behavior. Once this model is validated, it can then be used to simulate how a Big Data project will behave over time as the project unfolds given its starting conditions. Modeling and simulation (M&S) of existing conditions for a project can provide probabilistic predictions of the project's expected outcome. M&S of modifications of existing conditions can permit exploration of what-if scenarios for different policy options. By making portions of the model conditional upon

their selection, the manager can evaluate their contribution to the overall behavior of the project. These conditional portions can be matched to the reasons for failure listed in section II.D.

In order to develop the SD model, we need to understand the and special characteristics that constitute the structure of Big Data projects. We first explore the commonalities and differences between traditional IT projects and Big Data projects (focusing primarily on Big Data analytics projects). From this comparison, we identify specific dynamics, and explore each factor as it has been elucidated in various project development practices (agile and lean). We then explore the structure of SD models that have already been developed for traditional project management.

From this exploration, we propose an SD model specific to Big Data projects that incorporates the dynamics identified as important to both traditional project management and Big Data project management, as well as those specific to Big Data projects.

IV. BIG DATA PROJECTS VS REGULAR IT PROJECTS

It is then important to compare regular IT projects with Big Data projects to determine which characteristics are common and can be modelled by core project dynamics techniques, and which IT projects what are different

A. Commonalities

Big Data projects share many characteristics with other types of projects. They consist of a series of activities or tasks (work to be done). They typically have a specific objective (scope) to be completed within certain specifications (benefits to be achieved). They frequently have defined start and end dates (schedule). And they consume and/or utilize resources (costs) that operate at some defined level of productivity.

Unfortunately, another commonality of Big Data projects to traditional IT projects is that they also experience failure at a rate roughly equal to traditional IT projects involving cost overruns, schedule slippages, deficiencies in delivered results, and catastrophic program failure resulting in abandonment or outright cancellation.

These commonalities imply that general project dynamics, described above as a means of modeling both the positive and negative behaviors experienced by projects, should also be applicable to Big Data Projects.

While Big Data projects do share many characteristics with other types of projects, such similarities have less to do with technology than with how projects are governed in relationship to other enterprise processes and systems [12].

B. Differences

So, what are the factors that make Big Data projects different from other IT related projects?

Werner Vogels, CTO of Amazon.com Inc., once said in an interview [13] that: "one of the core concepts of Big Data is being able to evolve analytics over time …. in the new world of data analysis your questions are going to evolve and change over time and as such you need to be able to collect, store and analyze data without being constrained by resources". This is the fundamental difference with traditional Business Intelligence (BI).

So, requirements are typically not well known at the beginning and cannot be easily defined. And the requirements will change over time. In many cases, the Big Data project doesn't really have an end. It can lead in many different directions, while at the same time becoming an integral part of ongoing operations. This is a situation similar to what is called Exploratory Testing [14]:

- You cannot easily estimate your time for the exploration
- You cannot develop a detailed plan for the exploration testing, as you do not have defined expected results
- There is no defined scope for the exploration
- The developer, tester, product owner, and other members of the team may not be in control of the direction the exploration will take
- Measures of progress are difficult to establish; when is exploration enough

Another complicating factor for Big Data projects is, of course, scale. Many of the parameters that drive the operation of Big Data projects will have to be scaled up to Big Data proportions: complexity, skills mismatch, data integration, data management, etc. The very large size of these parameters will frequently result in impacts on project performance that are completely out of proportion to the inputs. For traditional projects, most of these parameters operate just fine at smaller scales, but as the V's of Big Data (Volume, Velocity and Variety) scale up, they eventually cause a bend in the performance curve after which the performance decays exponentially. While this type of behavior has been ameliorated using new Big Data technologies, these factors still need to be incorporated into the modelling effort.

C. Applicable Concepts from Agile

While Big Data projects exhibit many commonalities with and differences from regular IT projects, the Big Data projects must begin to adopt more appropriate Big Data project management techniques to overcome some of the problems currently being experienced.

Two useful sources of new techniques that can be brought to bear are a) agile software development, and b) lean product development.

One of the most popular agile approaches is called Essential Scrum [15]. Of its list of underlying principles, several are particularly applicable for Big Data project dynamics: Exploration, Iterative Development, Incremental Development, and Learning.

1) Exploration

Big Data Analytics projects are essentially an exploration. You fundamentally don't know what the outcome of the exploration will be, or at least you can't be certain that what you find will confirm your preconceived expectations. As Rubin states:

> "Plan-driven, sequential processes focus on using (or exploiting) what is currently known and predicting what isn't known. Scrum favors a more adaptive, trial-and-error approach based on appropriate use of exploration to discover the best questions to ask, and find the best answers to the questions being posed.
>
> "Exploration refers to times when we choose to gain knowledge by doing some activity, such as building a prototype, creating a proof of concept, performing a study, or conducting an experiment. In other words, when faced with uncertainty, we buy information by exploring.
>
> "Our tools and technologies significantly influence the cost of exploration. Historically software product development exploration has been expensive, a fact that favored a more predictive, try-to-get-it-right-up-front approach. Fortunately, tools and technologies have gotten better and the cost of exploring has come way down. There is no longer an economic disincentive to explore. In fact, nowadays, it's often cheaper to adapt to user feedback based on building something fast than it is to invest in trying to get everything right up front. Good thing, too, because the context (the surrounding technologies) in which our solutions must exist is getting increasingly more complex.
>
> "If we have enough knowledge to make an informed, reasonable step forward with our solution, we advance. However, when faced with uncertainty, rather than trying to predict it away, we use low-cost exploration to buy relevant information that we can then use to make an informed, reasonable step forward with our solution. The feedback from our action will help us determine if and when we need further exploration."

2) Iterative Development

A second applicable concept from Essential Scrum [15] is iterative development.

> "Iterative development acknowledges that we will probably get things wrong before we get them right and that we will do things poorly before we do them well. As such, iterative development is a planned rework strategy. We use multiple passes to improve what we are building so that we can converge on a good solution."
>
> "Iterative development is an excellent way to improve the product as it is being developed. The biggest downside to iterative development is that in the presence of uncertainty it can be difficult up front to determine (plan) how many improvement passes will be necessary."

3) Incremental Development

A third important concept accompanying Iterative Development from Essential Scrum is Incremental Development [15].

> "Incremental development is based on the age-old principle of "Build some of it before you build all of it." We avoid having one large, big-bang-style event at the end of development where all the pieces come together and the entire product is delivered. Instead, we break the product into smaller pieces so that we can build some of, learn how each piece is to survive in the environment in which it must exist, adapt based on what we learn, and then build more of it.
>
> "Incremental development gives us important information that allows us to adapt our development effort and to change how we proceed. The biggest drawback to incremental development is that by building in pieces, we risk missing the big picture (we see the trees but not the forest)."
>
> "Scrum leverages the benefits of both iterative and incremental development, while negating the disadvantages of using them individually. Scrum does this by using both ideas in an adaptive series of timeboxed iterations called sprints."

4) Learning

A fourth applicable concept from Essential Scrum [15] is learning: "There is learning that occurs when using sequential development. However, an important form of learning happens only after features have been built, integrated, and tested, which means considerable learning occurs toward the end of the effort. Late learning provides reduced benefits because there may be insufficient time to leverage the learning or the cost to leverage it might be too high.

> "In Scrum, we understand that constant learning is a key to our success. When using Scrum, we identify and exploit feedback loops to increase learning. A recurring pattern in this style of product development is to make an assumption (or set a goal), build something (perform some activities), get feedback on what we built, and then use that feedback to inspect what we did relative to what we assumed. We then make adaptations to the product, process, and/or our beliefs based on what we learned."

Big Data Analytics projects certainly need the flexibility and learning inherent in iterative and incremental development approaches due to the exploratory nature of the work. Big Data application and infrastructure projects also need this flexibility due to the different and complex technologies involved. Iterative and incremental development permits maximization of learning as early in the development process as possibility.

D. Applicable Concepts from Lean

One of the most powerful lean approaches is called Second Generation Lean Product Development [16]. Of its very long list of underlying principles (175 principles in 8 sections), many are similar to or the same as those listed above for Agile. In fact, Agile has drawn heavily from Lean concepts. However, several specific Lean concepts are particularly applicable for Big Data projects: Economics, Variability, Fast Feedback.

1) Economic Value

First and foremost, Lean Product Development demands that we quantify the overall economic impact of all pending decisions. We must set up an economic framework within which we can properly understand the cost differences between alternatives under consideration. Product and project attributes that have economic value should be quantified in terms of a standard and useful unit of measure: life-cycle profit or mission impact. This common unit of measure permits an analysis of the tradeoffs between cycle time, product cost, product value, development expense, risk, or other measures of technical performance.

Perhaps the most important variable to quantify is the Cost of Delay (COD). It provides a way of "sharing and understanding the impact of time against forecasted outcomes. It provides the means to calculate and compare the cost of not completing something now, by choosing to do it later" [17]. COD allows for a straightforward prioritization of work to be performed. More formally, it is the partial derivative of the total expected value with respect to time. Cost of Delay is expressed in units of $/time. The Delay Cost incurred (as a result of a delay) is found by integrating Cost of Delay over a specific time period [18].

Cost of Delay combines urgency and value, two things between which humans are not very good at distinguishing. To make decisions, we need to understand not just how valuable something is, but how urgent it is. The immediate and full cycle costs and benefits must be considered.

Also, since Big Data projects typically push our decisions toward extreme values, we may only need to know which direction maximizes performance, not necessarily exact values, in order to correctly trade off variables. Even when we are operating in trade off situations, highly accurate information is not necessarily needed to improve economic decisions, as opposed to making them perfect.

The issue of product value-added should not be left just to an informed customer as the sole judge, a practice typically adopted by Agile. Value added should be quantitatively determined as the difference in price or mission outcome that a rational buyer would pay for a work product before and after the activity is performed.

As important as it may seem to focus on the Pareto 20% of the problems (that provide 80% of the value), there is usually more actual opportunity in the undermanaged 80% of the problems. These greater in number but smaller in impact individual decisions are better handled in small batches as long as they can be structured to require a low fixed transaction cost per decision. These many small tradeoff decisions must be made continuously at random times throughout the development process. This approach allows for the rapid incorporation of changing information about the economics of our project, and the exploitation of perishable opportunities, rather than blindly conforming to an original project plan that may no longer represent the best economic choices available. In this way, by essentially buying information based on economic value, we can successively improve the conditional probability of our expectations, assess the value of risk reduction, and thus better manage risk.

Finally, all economic decisions should incorporate not just the value of the decision, but also the life-cycle cost. We should compare marginal value to marginal cost. All decisions must be time-based. This is frequently a good strategy for dealing with feature creep, as well as assessing work in process and sunk costs by only considering remaining costs.

2) Variability

A blind statement that "all variability on product development is to be avoided" is not correct. Some variability can actually create value, and it is important to distinguish between variability that increases economic value and variability that decreases it. It we accept the notion that our initial project plans are not 100% perfect (and in fact may be considerably less), then we must be able to embrace that variability is inevitable, and we must incorporate project mechanisms that can generate and capture new information and take advantage of it. Further, by combining the probability function of a decision with its payoff function, asymmetric payoffs in product development can be revealed where higher variability can be exploited to achieve higher economic value.

In exploratory experimentation and testing, maximum information generation occurs when the probability of failure equals 50% [16]. Because significant effort is expended to minimize failure, more time and money are expended than should be. The information content of a test (not counting simple validation or regression testing) comes from the degree of surprise associated with each possible result. So, exploration should be performed keeping in mind that economic value created by the exploration (expected value of the new information) should always exceed its cost.

In the past variability in forecasting and estimating in product development was highly undesirable. But uncertainty is proportional to the square of time. The longer term the plan, the more wildly uncertain it becomes. It turns out that extensive efforts should never be expended to develop long term plans. It is much more beneficial to only develop detail plans for the near future. This can best be accomplished by either limiting the scope of the project or by it into a number of smaller efforts. As it turns out, many

small experiments will produce much less variation than one big one.

3) Fast Feedback

Merriam Webster defines feedback as "the transmission of evaluative or corrective information about an action, event, or process back to the original or controlling source" [19]. Feedback is critically important to the product development process because of the dynamic nature of requirements and emergent information about the cost and benefits of development activities within a dynamic environment. Feedback is the basic input to the control systems which are set up to manage the product development process.

Corrective actions based on feedback should be prioritized based on the economic impact of a deviation [16]. It is first necessary to identify the magnitude of a deviation that is worth controlling, and then determine the change in the variables under observation that would produce the economic impact. These deviations could lead to both positive opportunities to be pursued as well as negative outcomes to be avoided. One should never establish product development controls to a static upfront plan, because the plan will almost always be based on unstable assumptions. Rather than trying to close the gap by moving the development toward the goal, it is usually more desirable to adjust the goal since the feedback will provide much more clarity in the direction to be taken.

Given this potential impact, we should be attempting to shorten the time between development cause and project effect. The more we can speed up the feedback process, the more we can speed up learning. In this way, we can generate more information and learn new things. By decreasing the size of each development effort, we will naturally speed up the feedback process and thus increase learning.

V. General Project Dynamics

So how should Big Data project problems be addressed? One way is to consider the Big Data project as a system, to use "systems thinking" to evaluate how it operates, and to employ systems dynamics (SD), the technique used to implement systems thinking, to do the analysis.

Project conditions and the project's performance evolve over time because of accumulations of project progress and resources, and because of feedback responses involving nonlinear relationships inherent in the project's structure.

A. Systems Thinking

As defined by the Institute for Systemic Leadership, "systems thinking is a management discipline that concerns an understanding of a system by examining the linkages and interactions between the components that comprise the entirety of that defined system" [20].

B. System Dynamics

SD is the study of feedback loops as an explanation for complex system behavior through cause and effect analysis. The simulation of SD models enables observation of dynamic behavior or output of a system over time, and it provides a way to find points of leverage where small, focused actions can produce significant, enduring improvements [21]. SD can also establish an appropriate modeling and simulation capability for Big Data project management while providing the organization with the management dashboards and predictive analytics tools it needs to more effectively explore these issues.

System Dynamics (SD) employs a diagramming language using systems diagrams that provide a way to sketch out systemic interrelationships in a system or organization. A systems diagram presents a series of elements that are connected to one another with directional cause and effect links. As the value of an element goes up or down, the element(s) it points to will also go up or down either in the same direction or the opposite direction depending on whether the causal element has a positive or negative impact on the effected element. These graphical interrelationships among several elements can eventually lead back to the original causal element thus creating a feedback loop. These feedback loops can be positive reinforcing feedback loops if there are an even number of cause and effect links, or negative balancing feedback loops if there are an odd number of cause and effect links. As these feedback loops proceed over time, they can cause the system to exhibit nonlinear behavior, such as exponential growth or collapse, oscillation, goal seeking, or erratic movements.

A cause and effect link can sometimes have a delay which slows down the system and can lead to unexpected consequences. Other elements can represent constraints which influence a balancing loop by providing hidden limits that the system will adhere to, but which can also provide significant leverage points for a change initiative.

C. Core Project Dynamics

SD has been successfully applied by consultants and practitioners in thousands of project management situations conservatively saving clients $ Billions on improved project budget performance involving avoidance of cost and schedule overruns and resultant disputes. Savings from earlier completion and higher delivered quality further increase the benefit to clients [22].

Traditional project management exhibits some real-time feedback structures that drive project success and/or failure (project features, rework cycles). Further, project dynamics can be used to represent the policy-based behaviors of the participants (project control actions, policy resistance, and unintended consequences).

Project Dynamics, the foundation for describing actual experiences of software development projects, begins with the simple work model where there is first a stock of Work to be Done which gradually flows through Work Being Done to a stock of Work Really Done. The rate at which work is done is controlled by a) the number of

People/Resources applied to the work, and b) their Productivity.

When work is performed at less than 100% Quality, rework will be needed. But it must be discovered first. Thus, there will be a stock of Undiscovered Rework created. As this rework is discovered, say by testing or through product failure, a stock of Known Rework will be accumulated which must then be integrated back into the primary work stream. This is called the Rework Cycle [23]. Furthermore, if requirements change due to customer requests or environmental changes, work that was already completed will become obsolete, and must also be redone [20]. These dynamics are all depicted in Figure 1.

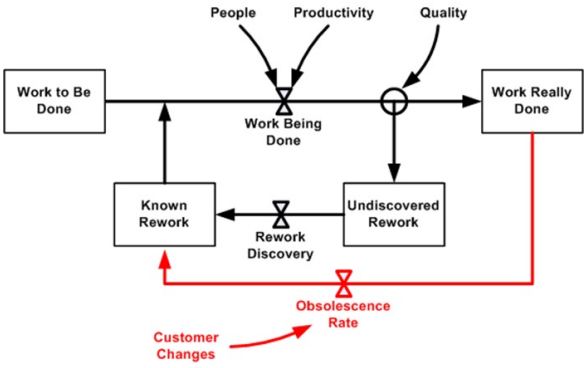

Figure 1 - Rework Cycle

Project performance is typically measured in terms of schedule, cost, quality and scope. Management actions to control a project's performance are modeled as efforts to close the target-performance gap in one or more of the performance dimensions. For example, typical control actions for achieving a target schedule:

- Move targets toward project behavior:
  - Slip deadline, increase budget, reduce scope, or accept higher fraction of flaws in final product
- Move project behavior closer to targets:
  - Work overtime, add people, work faster, slack off, or gold plating

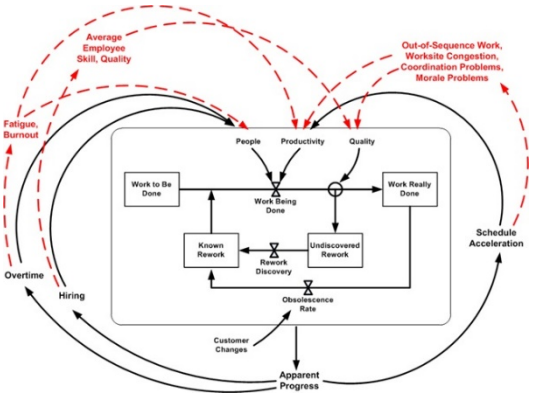

Figure 2 - Ripple Effects & Unintended Consequences

When apparent progress starts to slip, management's typical response is to increase overtime, add more people, and accelerate delivery of certain items not yet completed. Unfortunately, these actions taken to close a gap between project performance and targets often have unintended side effects (see Figure 2) that generate policy resistance: experience dilution, too big to manage, burnout, haste makes waste. What is happening is that these typical control actions are setting in motion negative feedbacks that are working against their intended consequences – what is called "Brooks Law" [23]

These negative feedback ripple effects can themselves generate secondary and tertiary feedbacks and side-effects with even more unintended consequences [24]: haste creates out-of-sequence work, errors build more errors, errors create more work, people begin to exhibit hopelessness. Adverse impacts through clients and customers can initiate or amplify external project dynamics: scope creep, customer changes & obsolescence, contractor change orders, increased oversight, client intransigence, project conflict/restructure, etc. (see Figure 3).

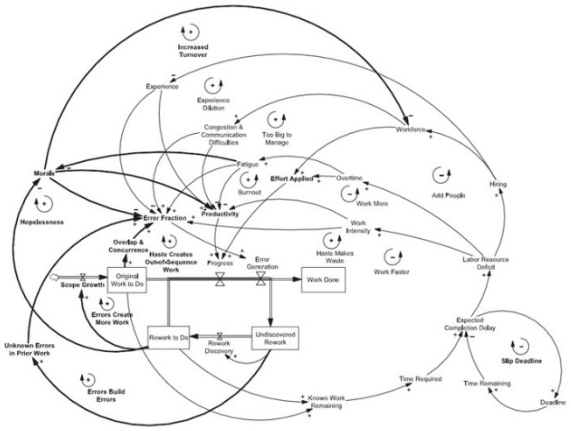

Figure 3 - Knock-on Effects

If the Rework Cycle is recognized, and the impact of ripple and knock-on effects on performance is acknowledged, management can reconsider their mental models of how things operate, modify their typical knee-jerk responses, and take more appropriate actions to minimize the consequences: Improve quality and reduce errors, discover rework faster, don't start downstream work too soon, improve project performance communication, inform management of ripple and knock-on effects.

VI. Big Data Project Dynamics

In order to address Big Data project problems, we need to consider the Big Data project as a system, to use "systems thinking" to evaluate how it operates, and to employ systems dynamics (SD) to do the analysis.

From the discussion above of the important dynamics of Big Data, we can begin to see the outlines of an SD model that we will call Big Data Project Dynamics (BDPD). The

BDPD model should address the five key sectors depicted in Figure 4: Core Rework Cycle, Iterative and Incremental, Exploration and Learning, Economic Value, and Policy Actions & Consequences.

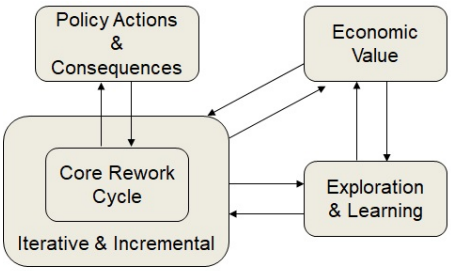

Figure 4 - The BDPD Model

The following subsections describe each of these component sectors in more detail.

A. Core Rework Cycle

The first component to include in the BDPD model is the core rework cycle from Section V.C above. All of the pieces of the basic rework model need to be adopted whole cloth into BDPD because they represent universal dynamics related to the management of work and quality. These pieces include the simple work model consisting of the following (see Figure 1): Work to be Done, Work Being Done, Work Really Done, People/Resources Productivity, Quality, Undiscovered Rework, Rework Discovery, Known Rework, Customer Changes and Obsolescence Rate.

These dynamics are entirely applicable to any type of work that is being performed at any scale in any area of endeavor. They are particularly applicable to Big Data projects which are typically characterized by great complexity. When properly considered, they offer the opportunity to greatly improve the speed with which feedback is achieved, and can make a Big Data project much more manageable.

B. Iterative and Incremental

Traditional project management attempts to minimize or eliminate rework by emphasizing spending much more time up front explicitly detailing requirements and developing very detailed plans. Unfortunately, this approach has exacerbated the very problems they intend to resolve. The Agile community has instead successfully adopted a number of techniques (here called "Iterative and Incremental") that attack rework discovery with a much different approach. This approach is characterized by dramatically scaling down the size of each unit of work to be operated on, performing the work in a repeating sequence of short systematic steps, and continuously integrating the results.

The Agile Project Dynamics (APD) model was developed to "gain insight into the dynamics of how Agile development compares to classic "waterfall" approaches by constructing an SD model for software projects" [25]. APD presents a framework of characteristics representing different Agile methods (which APD calls genes: micro-optimizing, refactoring, continuous integration, team dynamics, customer involvement). Because APD incorporates the Core Rework Cycle dynamics discussed in Sections I.A.2), I.A.3), and VI.A, we have used the APD model extensively in BDPD.

APD assumes a hierarchy of work decomposition where product is implemented through a series of releases, a release is implemented through a series of sprints, and a sprint consists of a series of work tasks). The APD model looks something like Figure 5.

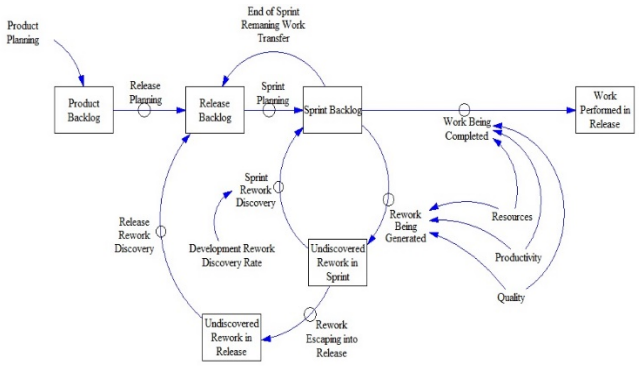

Figure 5 - The BDPD Iterative and Incremental Model

Iterative and Incremental includes rework cycles for both sprints and releases. Various important dynamics can then be added to this core model component as positive or negative reinforcing loops.

C. Exploration and Learning

One of the major differences between traditional project management and Big Data projects is the need for extensive exploration and learning involved in using Big Data analytical tools on large collections of data.

The Agile Iterative and Incremental sector already has several built-in learning loops [26]: a) a daily scrum that can reorganize the upcoming day's tasks, b) an end of sprint review that demos and inspects the product increment produced during the sprint and adapts what we build next by grooming the product backlog for the upcoming sprint, and c) a sprint retrospective that focuses on how the process is operating – what went well, what went wrong and what could be done differently.

To augment these learning loops, BDPD adds an Exploration and Learning sector (see Figure 6) to the Iterative and Incremental sector.

The Exploration and Learning component will operate as follows. The output of the Iterative and Incremental sector is a release of completed product. Usage of this completed work will generate a set of actual results. We can then compare the actual results against a set of expected results and evaluate the results gap. A large gap will increase the Pressure to Make Project Changes. A small gap will decrease this pressure. The size and nature of this pressure

can in turn a) lead to a revision of the original question underlying the analytics which will be fed back into the backlog as rework, or b) generate a completely new question that can be added directly to the analytics backlog. This will permit the project to move where the results are indicating will have the greatest impact.

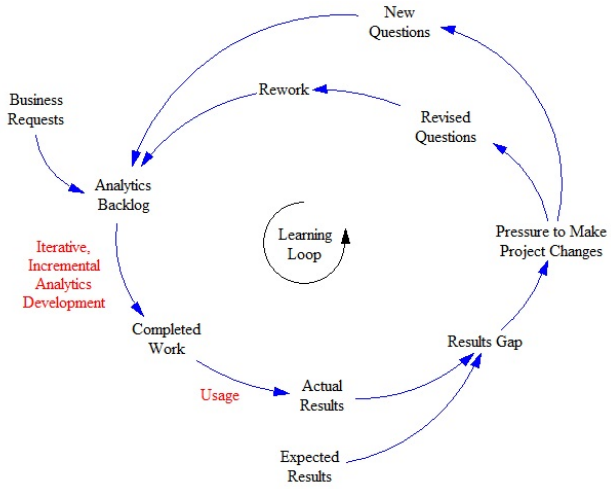

Figure 6 - BDPD with Learning Loop

D. Economic Value

Perhaps the most important focus of all activities in both Agile and Lean is to optimize the value of the work to be done and delivered to the customers. For the reviews and evaluations in the Iterative and Incremental sector, and the gap analysis in the Exploration and Learning sector, to operate properly, the decisions must be based on economic value considerations.

For the Exploration and Learning sector, the actual results achieved by the completed work must be compared to the expected results (see Figure 7). When we have actual results to look at, we will generate better information with which to calculate Economic Value. This in turn will increase the Conditional Probability of Expected Value. The Business Case will be calculated by subtracting the Life Cycle Cost from the Expected Value, combined with Urgency (cost of delay), and discounted by the project's Risk and by the Conditional Probability of Expected Value. This Business Case, when combined with the Results Gap will determine how much Pressure to Make Project Changes can be expected.

The Economic Value sector will also feed into the Iterative and Incremental sector. In this case, the Economic Value of the Expected Results of each alternative to be considered must be assessed in a manner similar to how the business case was calculated for Exploration and Learning. Then by comparing the Expected Results of the alternatives (stories) to one another, we can prioritize the work in the backlogs to be undertaken. This economic value calculation would need to be performed in the Release Planning and Sprint Planning steps.

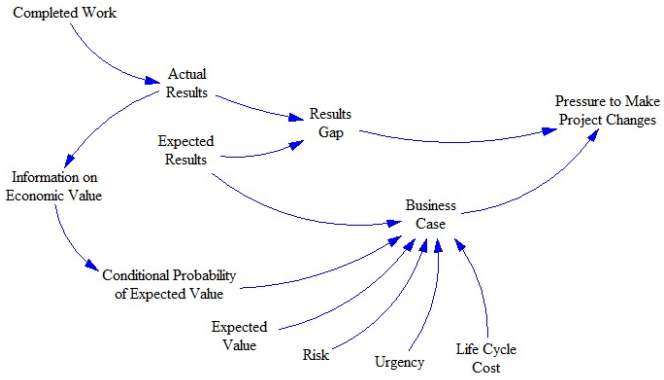

Figure 7 - BDPD Economic Value

E. Policy Actions & Consequences

Given the BDPD model developed so far, we have a platform for managers to make various policy decisions and take actions to address situations they may confront as the project evolves over time.

For example, as managers evaluate apparent progress in the project, and they discover shortfalls from their original plans, their typical mental models of how to react would immediately drive them to begin moving project behavior closer to their originally planned targets. This would involve taking actions to increase the resources available to the project such as scheduling overtime or adding people. However, as discussed in Section V.C above, these actions will lead to a series of ripple effects and unexpected consequences that will negatively affect the resources available to the project, the productivity of those resources, and the quality of the work they perform. The "system" will, in effect, begin pushing back.

Being aware of this effect, and being able to model and simulate it, gives managers an opportunity to expose their typical mental models to understand how they may be driving the counterintuitive behavior of the system they are attempting to manage. They can then see how modifications to their normal responses (in essence, changing their mental models) might lead to better outcomes. By working on the leverage points within the model (such as by introducing better ways to add resources to the project without burdening existing team members with additional training, communication and coordination duties), the manager can begin to move the project toward improved outcomes.

So, because we know that, as the project progresses, the original project plan will typically no longer represent the best information and economic choices available, managers can treat actual results as valuable new information with which they can revise their original plans. They can make more informed decisions, introduce much more intelligently devised policy actions, and adjust their expected outcomes accordingly.

VII. Future Directions

The next chore for this research effort is to get the BDPD model fully developed. There are many components of the model that are readily reusable from prior referenced work, particularly the Core Rework Cycle, the Iterative and Incremental, and the Policy Actions and Consequences sectors. But these components need to be reworked to integrate properly with one another. Additional components need to be created for the Exploration and Learning and for the Economic Value sectors.

The full BDPD model itself will be developed iteratively and incrementally. As model development proceeds, the model will be validated against some real Big Data project data.

Once BDPD is operating well, various more sophisticated representations of the reasons and causes for failure can be incorporated into the model, for example: Wrong/Inadequate Skills, Technology Complexity, Data Integration and Data Management, or Management & Cultural Resistance. BDPD can then be used as a platform for specific problem exploration. This would permit more flexible and selected representation of different Big Data project conditions. This work could be prioritized based either on the frequency of representation of the reasons as listed in section II.D above, or on some importance scheme yet to be determined.

A number of other extensions can also be considered. For example, the BDPD Economic Value sector can be expanded to include a connection to the dynamics of business models.

Another consideration that could be addressed is How will we productively put the BDPD model into the hands of the managers that need it? This would be accomplished using dashboards and predictive analytics driven by a standard off-the-shelf SD tool running the BDPD model. Given this set-up, managers and practitioners could much more easily simulate Big Data project behavior from initial conditions that they would enter or modify as often as they needed. Each simulation run would generate a probabilistically predicted project outcome right from the dashboard.

Yet another possible activity would be connecting Big Data project critical success factors (CSFs) directly to the BDPD model [27]. This would be based on equating a CSF to a constraint in an SD model, where the constraint is the actual level of the factor for the project, and is compared to the level of the factor required for the project to operate successfully. The resulting gap and its magnitude is one of the things that is driving project behavior.

As is obvious, this paper is just the beginning of a number of very interesting

VIII. References

[1] PMI. (2013). A Guide to the Project Management Body of Knowledge (PMBOK Guide). Fifth Edition. Project Management Institute.

[2] Kleppman, M. (2017). Designing Data-Intensive Applications: The Big Ideas Behind Reliable, Scalable, and Maintainable Systems. O'Reilly Media.

[3] Leffingwell, D. (2011). Agile Software Requirements: Lean Requirements Practices for Teams, Programs, and the Enterprise. Addison Wesley

[4] Standish. (2014). Chaos Report 2014. The Standish Group. https://www.projectsmart.co.uk/white-papers/chaos-report.pdf.

[5] White, S. (2015). Study reveals that most companies are failing at big data. IDG Communications, Inc. http://www.cio.com/article/3003538/big-data/study-reveals-that-most-companies-are-failing-at-big-data.html.

[6] Wynkoop, S. (2017). Big Data Projects Failure Rate? SSWUG.ORG. https://www.sswug.org/swynk/editorials/big-data-projects-failure-rate/.

[7] Bodkin, R. (2013), The big data Wild West: The good, the bad and the ugly. Gigaom, 2017 Knowingly, Inc. https://gigaom.com/2013/09/14/the-big-data-wild-west-the-good-the-bad-and-the-ugly/.

[8] Agarwal, S. (2017). Why big data projects fail and how to make 2017 different. Network World. IDG Communications, Inc. https://www.networkworld.com/article/3170137/cloud-computing/why-big-data-projects-fail-and-how-to-make-2017-different.html.

[9] Crawford, T. (2013). Big Data Analytics Project Management: Methodologies, Caveats and Considerations. PMI Silicon Valley Chapter, PM Symposium. https://pmisv.org/document-repository/special-folder/symposium/2013-symposium-presentations?limit=20&limitstart=20.

[10] Philip, N. (2014). The Key to Success with Big Data Projects (Updated). Qubole, Inc. https://www.qubole.com/blog/big-data-project-success.

[11] Becker, D. (2017). Reasons for Big Data Project Failures. Unpublished. Internal Research Report, DB-SAC.

[12] McDonald, D. (2015). Planning and Managing Big Data Projects: Selected Articles - Understanding the Challenges of Big Data Project Management: "The Data Must Flow". Dennis D. McDonald Website. http://www.ddmcd.com/flow.html.

[13] Zicari, R. (2011). On Big Data: Interview with Dr. Werner Vogels, CTO and VP of Amazon.com. ODBMS.org. http://www.odbms.org/blog/2011/11/on-big-data-interview-with-dr-werner-vogels-cto-and-vp-of-amazon-com.

[14] Oluwole, D. (2013). Agile Methodology Is Not All About Exploratory Testing. Scrum Alliance. https://www.scrumalliance.org/community/articles/2013/march/agile-methodology-is-not-all-about-exploratory-tes.

[15] Rubin, K. (2012). Essential Scrum: A Practical Guide to the Most Popular Agile Process. Addison-Wesley Professional.

[16] Reinertsen, D. (2009). The Principles of Product Development Flow: Second Generation Lean Product Development. Celeritas Publishing.

[17] Huether, D. (2015). An Introduction to Cost of Delay. LeadingAgile. https://www.leadingagile.com/2015/06/an-introduction-to-cost-of-delay/.

[18] Wikipedia. (2017). Cost of Delay. Wikimedia Foundation, Inc. https://en.wikipedia.org/wiki/Cost_of_delay.

[19] Merriam-Webster. (2017). Feedback. Merriam-Webster, Inc. https://www.merriam-webster.com/dictionary/feedback.

[20] Systemic Leadership. (2017). Basic principles of systems thinking as applied to management and leadership. The Institute for Systemic Leadership. http://www.systemicleadershipinstitute.org/systemic-.leadership/theories/basic-principles-of-systems-thinking-as-applied-to-management-and-leadership-2/
[21] Sterman, J. (2000). Business Dynamics: Systems Thinking and Modeling for a Complex World. McGraw-Hill.
[22] Lyneis, J., Cooper, K., Els, S. (2001). Strategic management of complex projects: a case study using system dynamics. System Dynamics Review Vol. 17, No. 3, 237–260. John Wiley & Sons, Ltd. http://onlinelibrary.wiley.com/doi/10.1002/sdr.213/pdf.
[23] Cooper, K., Mullen, T. (1993). Swords and Plowshares: The Rework Cycles of Defense and Commercial Software Development Projects. American Programmer. Volume 6, Issue 5.
[24] Lyneis, J., Ford, D. (2007). System dynamics applied to project management: a survey, assessment, and directions for future research. System Dynamics Review. Vol. 23, No. 2/3, p157–189.
[25] Glaiel, F., Moulton, A., Madnick, S. (2013). Agile Project Dynamics: A System Dynamics Investigation of Agile Software Development Methods. https://web.mit.edu/smadnick/www/wp/2013-05.pdf.
[26] Rubin, K. (2016). Stop Calling Your Sprint Review a Demo—Words Matter! Scrum Alliance Inc. https://www.scrumalliance.org/community/spotlight/ken-rubin/january-2015/stop-calling-your-sprint-review-a-demo%E2%80%94words-matte.
[27] Saltz, J., Shamshurin, I. (2016). Big Data Team Process Methodology: A Literature Review and the Identification of Critical Factors for a Project's Success. 2016 IEEE International Conference on Big Data. School of Information Studies, Syracuse University.