Software Project Management
Towards Failure Avoidance

Thang N. Nguyen
Department of Information Systems, California State University Long Beach, California, U.S.A.

Keywords: Software Project, Management by Exception, Management Decision Evaluation, Biologically-inspired Application.

Abstract: Development tasks, in the thousands or more, are involved in a complex system/software project. These tasks aim at three objectives in the project, namely *on budget, on time and on performance*. Behind these tasks are decisions made on them to move the project forward. Project failure is an aggregated and cumulative failure of these tasks due to errors in tasks and/or decisions made. The project errors from faulty strategies to wrong builds, collectively called *exceptions* are much harder to detect than cost overrun (amounts spent) or time delay. The paper suggests a biologically-inspired approach to project failure avoidance which is different than most. It focuses on exposing *exceptions* as they occur and understanding the *decisions* made on them. Thus, there are two pieces needed for the proposed approach. The base piece is a *management by exception* (MBE) framework to monitor development exceptions occurred for management attention. The second piece is an adapted measurement method which will elicit, analyse and evaluate the decisions made on the reported exceptions. We argue that using the proposed approach, failure avoidance is possible and software project performance (in scope, intended features and desired quality) would be under control, and so are expected cost and time.

1 ON SOFTWARE PROJECT FAILURE

From the project perspective, according to Calleum Consulting which combined various facts and figures on why software fails (Calleum, 2014), it was reported that "*IT projects run 45 percent over budget and 7 percent over time, while delivering 56 percent less value than predicted*" (Calleam, 2014; McKinsey and Co. survey), "*Fuzzy business objectives, out-of-sync stakeholders, and excessive rework mean that 75% of project participants lack confidence that their projects will succeed*" and "*a truly stunning 78% of respondents reported that the "Business is usually or always out of sync with project requirements*" (Calleam, 2014; Geneca survey).

The KPMG survey showed "*An incredible 70% of organizations have suffered at least one project failure in the prior 12 months* and *50% of respondents also indicated that their project failed to consistently achieve what they set out to achieve*" (Calleam, 2014; KPMG survey). In change management, software projects experienced "*40% of projects met schedule, budget and quality goals, best organizations are 10 times more successful than worst organizations, biggest barriers to success listed as people factors: changing mindsets and attitudes – 58%, and corporate culture – 49%, lack of senior management support – 32%, underestimation of complexity listed as a factor in 35% of projects*" (Calleam, 2014; IBM survey). Other statistics (Mieritz, 2012) as well as investigations on what happened, why they happened, lessons learned etc. appeared in the literature (Heusser, 2013; Galorath, 2011; Krigsman, 2008; Charette, 2005).

In this paper, we approach what make projects fail differently i.e. from the perspectve of preventing them. Our purpose is simply to detect *exceptions*, whatever they are (*monitoring issue*), expose them to appropriate parties (*transparency issue*) for proper decision on them by the responsible parties, whoever they are (*control issue*), and evaluate whether the decisions are arbitrary, risky or otherwise (*justification issue*) towards failure avoidance.

Nguyen T..
Software Project Management - Towards Failure Avoidance.
DOI: 10.5220/0004992605600567
In *Proceedings of the 9th International Conference on Software Engineering and Applications* (ICSOFT-EA-2014), pages 560-567
ISBN: 978-989-758-036-9

In a complex system/software project, there exist thousands or more tasks. These tasks aim at three objectives in a project, namely *on budget, on time and on performance* (in this paper, *on performance* implies: in-scope, intended features or functionalities, and desired quality). Behind these tasks are decisions made on them by decision makers (DMs), i.e. management and subject matter experts (SMEs), exercising some decision making schemes.

In terms of success-failure, with pure monitoring of the funding amounts spent, task start dates and completion dates, the cost and time of the software project will reflect cost overrun or time delay for corrective decision making. Software performance, however, is more complex to monitor and measure in every phase of development. The project errors from faulty strategies to wrong codes, hereafter collectively called *exceptions* are commonly much harder to detect until they occur. These, if not detected and/or if occurred but not addressed properly for any reasons, could aggregate and accumulate into more critical wrongdoings (exceptions) and would bring the project to failure.

By and large, the success or failure of any tasks can be attributed to the decisions made by the management (including top executives) and software developers (including other SMEs such as business analysts, etc.). They are the responsible parties. They are the people who execute one task to the next and/or sign off the specifications, documents or other artifacts.

We propose to monitor the *exceptions* during the life of the project, which might be resulted from *decisions,* arbitrary or otherwise. Since all project tasks are linked in a complex decision network in a critical path-like, the overall failure can be initiated by the first wrongdoing initiated by some decision contributing to the final failure of the software project. We attempt to understand and to measure these decisions on exceptions made by the management-leadership team and the developers-SMEs responsible for the project.

Thus, there are two major pieces of our proposed solution. The base piece is a *event-driven framework which will house a amangement by exceptions (MBE) application to monitor and to expose all development exceptions* occurred during the development cycle for management and developers attention. The MBE is responsible by an Oversight organization (or Committee). This organization can then requests management-SMEs-developers to look at the decisions they make on the exceptions.

Thus, the second piece is a *measurement method which elicits, analyses and evaluates the decisions made on the reported exceptions and consequences*. Applying this measurement scheme for timely correction, we argue that we could possibly, at some confidence level, avoid failure, and software performance would be at the expected level in terms of scope, intended features and desired quality, and so are cost and time.

2 A FRAMEWORK FOR ENTERPRISE-WIDE INFORMATION MANAGEMENT BY EXCEPTION

For small size projects with fewer managers and developers, an MBE application can be simply daily or weekly meetings where exceptions are reported. For mid-size project, a software development decision model for managing software development projects as suggested by (Nguyen, 2006) or any project management tool in the market could be appropriate. For larger or very large project such as (1) the now-defunct Future Combat System of the US Army in the 2000's (GAO, 2009), (2) the US Air Force Expeditionary Combat Support System – ECSS, and (3) the US Marines Global Combat Support System - GCSS (GAO, 2012; Kanaracus, 2012), or the recent difficulties experienced by Healthcare.gov (Schadler, 2013), we would ask if a different scheme is possible for detecting exceptions and measuring decisions made on them.

To that end, we exploit a couple of considerations towards a framework for an information management by exception which is *biologically-inspired.* We discuss the rationale of such framework in some details in this section.

The initial consideration stems from the biological spectrum (Figure 1) which consists of *protoplasm* component at the lowest end to the *biosphere* component at the highest end (Alberts et al., 1998; Raven, 2008).

Some part of this biological spectrum has been the source for insights by different researchers during the last century. At the cell level, for example, there have been the Computer and The Brain and the Theory of automata by John Von Neumann (Von Neumann, 1966), and the theory of autopoesis (Maturala and Varella, 1980), to name just a few. Institution (as community) and business

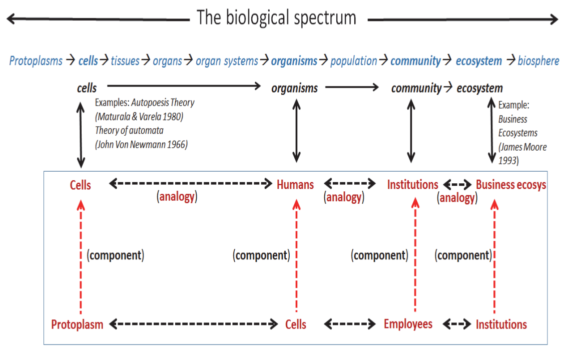

The biological spectrum
Protoplasms → cells → tissues → organs → organ systems → organisms → population → community → ecosystem → biosphere
cells
organisms
community → ecosystem
Examples: Autopoesis Theory (Maturala & Varela 1980) Theory of automata (John Von Newmann 1966)
Example: Business Ecosystems (James Moore 1993)
Cells
Humans
Institutions
Business ecosys
(analogy)
(analogy)
(analogy)
(component)
(component)
(component)
(component)
Protoplasm
Cells
Employees
Institutions

Figure 1: The biological spectrum.

ecosystems in the sense of James Moore (Moore, 1996) are part of the spectrum.

We look at the spectrum a little differently, however. We wonder if both human (as organism) and institution (as community) can be viewed analogously (the *blue* box with red text in Figure 1). That is if an institution is considered as analogous to a human body (structural, functional or behavioral) then the employees of the institution are analogous to the cells of the human body.

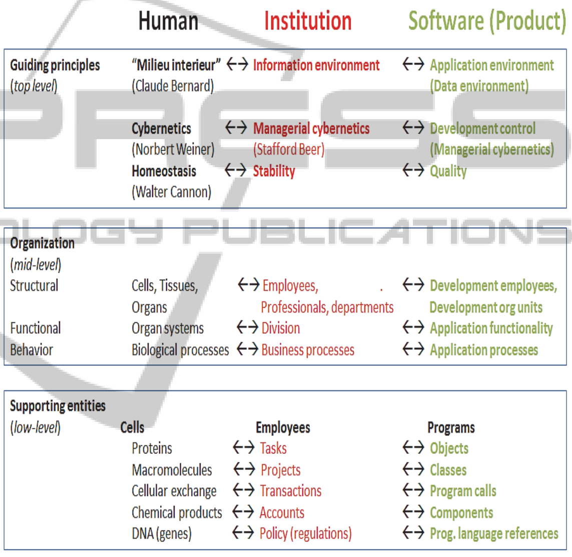
From an *exception* perspective, cells in the human body can turn abnormal or cancerous. If the abnormal cells grow uncontrollably, invade nearby tissues, termed as malignant tumor, and subsequently proliferate to other organs, the tumor can bring death to the human (King, 1996). Note that an infectious disease by a deadly virus in a host cell would also cause death.

Analogously, if a group of people in an institution turned abnormal for any reasons (commonly greed, power, growth, risk, etc.), they can become an "*institution malignant tumor*". If this group is funded and exercises their influence to other organizational units, they can bring collapse to the institutions. Examples of cancer-like wrongdoings which led to collapses in the financial circle are Enron, WorldCom, Adelphia, Parmelat, and Lehman Brothers (Foster, 2010).

In software development projects circle, issues leading to exceptions such as *ill-understood business problems, risky contracts with client, out-of-sync between stakeholders, underestimated complexity of solutions,* and others (Pressman, 2010) are originated from top management-SMEs-developers' decisions. These decisions, if arbitrary or turned bad, when aggregated, might bring failure to the project.

We can extend the analogy between *human* (organism) and *institution* (community) to *software product* (these are products of humans in the institution) into perspective as shown in Figure 2. The analogy does not have to be perfect as long as it can offer some insights into the making of our proposed solution, subject to verification and validation.

For humans, at the higher level there are biological guiding principles that govern the structure, functionality and behavior of a human body at the middle level. They are: (1) the "*milieu interior*" of Claude Bernard (Gross, 1998) in which all cells, tissues, organs and organ systems of the body reside, (2) the principle of *cybernetics* (Wiener, 1948) which controls the human functionality and behavior, and (3) a condition called *homeostasis* where the human body maintains its equilibrium (Cannon, 1963).



	Human	Institution	Software (Product)
Guiding principles (*top level*)	"Milieu interieur" (Claude Bernard)	↔ Information environment	↔ Application environment (Data environment)
	Cybernetics (Norbert Weiner)	↔ Managerial cybernetics (Stafford Beer)	↔ Development control (Managerial cybernetics)
	Homeostasis (Walter Cannon)	↔ Stability	↔ Quality
Organization (*mid-level*)			
Structural	Cells, Tissues, Organs	↔ Employees, Professionals, departments	↔ Development employees, Development org units
Functional	Organ systems	↔ Division	↔ Application functionality
Behavior	Biological processes	↔ Business processes	↔ Application processes
Supporting entities (*low-level*)	Cells	Employees	Programs
	Proteins	↔ Tasks	↔ Objects
	Macromolecules	↔ Projects	↔ Classes
	Cellular exchange	↔ Transactions	↔ Program calls
	Chemical products	↔ Accounts	↔ Components
	DNA (genes)	↔ Policy (regulations)	↔ Prog. language references

Figure 2: Human-Institution-Software analogy.

The corresponding analogous principles of the institution, considered as a community are namely, *information environment*, *managerial cybernetics* (Beer, 1972) and *stability* as shown in the middle column. At the lower end, we recognize the five elements of the human body supporting the structure, function and behavior of the human. Analogously, we list the corresponding analogous elements supporting an institution. Those of software products are shown on the right side of Figure 2.

Figure 2 will not help much in our attempt to modelling, unless we re-arrange all the *guiding principles* at the top level, the *structural, functional and behavioural* aspect at the middle level, and all the *supporting elements* at the lowest level of Figure 2 to reflect *activities* and *events* that occur as shown in the dotted box of Figure 3 and in the control mechanisms.

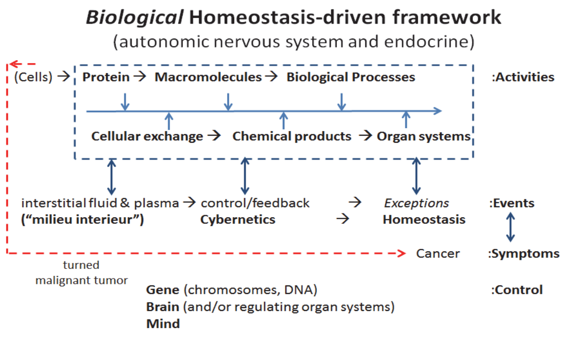


Figure 3: Biological framework.

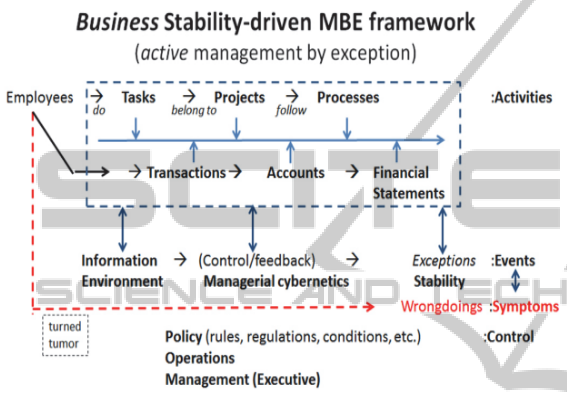


Figure 4: Business framework analogous to biological framework.

From this rearrangement, we can formulate two other frameworks which are considered analogous, to some extent, to the *biological framework* (Figure 3). They are: the *business framework* (Figure 4) and the *development framework* (Figure 5). Note that in Figure 3, the set of proteins, macromolecules, cellular exchanges and DNA/genes are analogous to those in Figure 4, with projects, tasks, transactions, accounts and policy, and in Figure 5, with objects, classes, program calls, components, and program language references. All three frameworks share a common goal: the *detection of exceptions*.

The monitoring and detection in the last two frameworks (*business* and *development*) is performed by an MBE application as exemplified in the application menu shown in Figure 6.

The MBE captures software project data, e.g. at the management level, it involves *Project, Tasks, Transactions, Accounts, and Policy* (e.g. System Operating Procedures or high-level strategy), and at the supporting development level it involves *objects, classes, program calls, components,* and *standard operating procedure (SOP)* or *programming language references*, with the intention to detect exceptions much like the detection of malignant tumor in the human body.

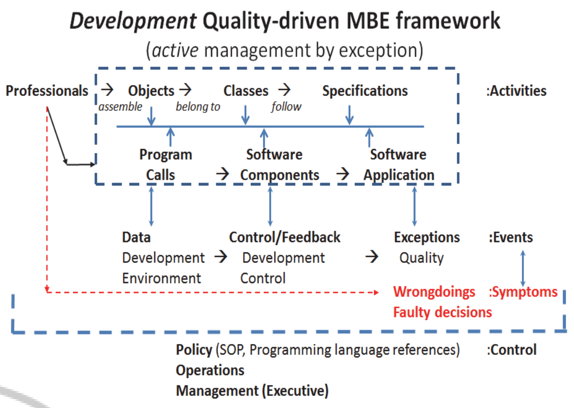


Figure 5: Software project framework analogous to biological framework.

DATA ACQUISITION	ANALYSIS	REPORTING
Task	Data analysis	Standard report
Project	Process analysis	Periodic report
Transaction	Event analysis	Exception report
Account	Cluster analysis	Custom report
Policy	Correlation analysis	Other

Figure 6: MBE application menu.

The data acquired (*Data Acquisition* on the left of Figure 6) are subject to different analysis methods listed in the middle part of Figure 6 (*Analysis*). Different exception reports (*Reporting*) can be requested and produced.

Note that we don't have to draw the cancer analogy to arrive at an MBE for software projects as proposed. However, the cancer analogy gives insights into the criticality of symptoms of a malignant tumor. Cancerous symptoms are quite often hidden until later stages (by the time they surface, it is too late, the human would die). This can be parallel to the hidden wrongdoings in software project failures or in corporate collapses. The analogy gives another advantage: it is possible that some known biological processes would give insights into the formulation of additional analysis schemes or methods since it is the humans who create institutions and software products, or any other man-made products.

The three *umbrella* frameworks depicted by Figure 3-5 might have some weaknesses. They might need some fine tuning to explore the details of the analogical aspect. But for our purpose, they are sufficiently adequate. They show commonalities in terms of the *activities, events and control* mechanisms guided by *policy* for the detection of exceptions to be exposed to management as sketched in Figure 6. The exceptions are considered as symptoms of wrongdoings on which decisions by the responsible parties are made.

It is known that the responsible parties do not always properly act on exceptions as expected. At times, decision makers might intentionally involve in arbitrary decisions. They can also purposely ignore, avoid, alter or hide the symptoms which could aggravate or lead to more catastrophic situation. Therefore exposing symptoms of wrongdoings and making them transparent are necessary but not sufficient. There must be a way for the Oversight organization to “force” the responsible parties to take actions and measure their decision’s effectiveness. This is the topic of the next section.

3 MEASURING DECISION EFFECTIVENESS

Decisions made by Good *management-leadership* and Good *employees-SMEs* would ensure *success* (quadrant 2 of Figure 7). This is contrasted and opposed to Bad *management-leadership* and Bad *employee-SMEs* (quadrant 4). The latter would possibly bring to project failure. If one of the two is bad while the other one is good, it will be more complex to measure or label the failure-success (quadrant 1 or 3) of the decisions.

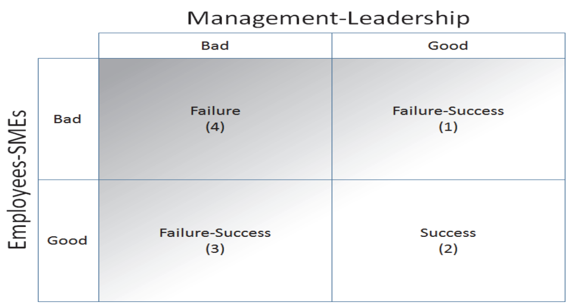


Figure 7: Management-Leadership top grid.

In all four cases (quadrants) we wish to know how to provide a measure of the decision maker’s effectiveness on a scale of failure-success in the software development management. The top grid of Figure 7 actually embodies underneath a hierarchical decision grid network which can be very complex.

In this grid, we think of employees-SMEs as primarily concerned with operational and some tactical decisions in an institution. On the other hand, management-leadership is primarily involved with strategic and some tactical decisions, and corporate vision.

Thus, within the context of this top grid of Figure 7, we need to drill-down any exceptions in software development projects, on which decision makers made decisions.

Example *decisions* which might cause exceptions are: *A sales person closes a development contract with minimum involvement of technical personnel, leading to under-estimated size and complexity. A business analyst insists on unrealistic requirements. A developer manager approves the use of open-source for cost reasons.* Example *exceptions* include**:** *A tester reports a memory leak. A client experiences a deadlock between multiple users using the same input form against the same policy or rule on data inputted.* From this set of exceptions, we need to elaborate all possible attributes of the decisions on exceptions, which we will call *constructs*.

The proposed process to arrive at constructs is adapted from George Kelly’s Personal Construct Theory (PCT) (Kelly, 1963), Valerie Stewart’s repertory grid (RG) (Stewart, 1981, 2010) and other variations (Smith et al., 1996).

We ask the decision makers to qualify the *exceptions* as business (and/or technical) *elements* (projects, tasks, transactions, accounts, and policy) associated with the exceptions. They have to select three exceptions at a time (called a triad), identified as crucial *in terms of decision made on them*. We will ask:

- In what way two of the elements have in common, (*emergent construct*) and
- In what way both of them differ from the third (*implicit construct*, opposite to emerging)

We build a detailed grid with columns headings as the exception elements identified by triad, and with rows as the hierarchical structure of *decision constructs* in opposing pairs underneath the top grid. Hinkle’s laddering up and Landfield’s laddering down (Fromm, 2004) and other recent modifications to laddering are used to elicit other constructs (Korenini, 2012).

We ask them to associate each decision construct with a ranking or better yet a rating. Specifically, they will be asked to rate each and every construct in a 5-point (or 7-point scale), e.g. from 1-5, with 1 being at one end of the construct dipole (emergent) and 5 being its opposite (implicit). Content analysis and/or cluster analysis are used as exemplified in (Stewart, 1996) and (Bourne and Jenkins, 2005), and others.

The purpose is to identify the significant decision constructs involved and their rated values for all elements. From the grid, a final and combined measure of all evaluations expresses a level of success (or failure) in the overall failure-success dipole. This is the indicator of how effective the management-leadership and SMEs-developers team member is.

The elicitation, analysis and evaluation is responsible by a separate organization called an Oversight organization or Committee. The skills set for elicitation, analysis and evaluation needs be developed.

4 DISCUSSION

In section 2, we proposed a biologically-inspired framework (event-driven, Figure 5) and the MBE application (Figure 6) for the detection and exposition of exceptions in an institution by the human-institution analogy. One would expect that when examining, in section 3, the decisions on the exceptions we would look into neuroscience (Kandel, 2000) or cognitive neuroscience (Koch and Davis, 2003) for analogy insights into the institution's decisions made by its management-leadership-SMEs team.

The bottom-up path (anatomy-physiology-neuroscience) however is too complex for us to handle and offers no guarantee that we can reach the top grid constructs for success-failure (Figure 7) since decision topics in neuroscience is still under investigation, even with the best know technique, fMRI. Instead, we chose to address *decision on exceptions* top-down, from a psychological aspect in the sense of George Kelly. This is investigated with the hope that the top-down approach to decision would lead to a construction system which would identify some core constructs eventually delineated by cognitive neuroscience findings. This is to establish an integration link between the exceptions (low-level events) and the decisions on them (high-level action). Thus, the repertory grid in section 3 helps reveal the *what, why and how* (in what way) of a hierarchical decision construct model by which the responsible parties in an institution, individually and collectively, respond to different exception situations.

Our thinking is that if we can discover and understand the construction system of the decision makers (by elicitation as discussed in section 3) on exceptions in this particular domain: software development and management, then not only we can (1) explain what caused software failure, (2) predict the consequences, (3) take remedy actions, but also (4) avoid future crises.

There are two scopes in pursuing the top-down approach. First, by performing the elicitation *individually*, the Oversight organization begins to pay attention to the DM responsible or involved in the exception for recommended remedy action and prediction measure to avoid future crisis or failure. Secondly, by analysing the elicited data *collectively*, some patterns characterizing the institution's decision model can be discovered.

Since the repertory grid technique has been investigated over the last five decades and have been applied to many domains successfully, even though it is complex and requires excellent interview skills, we can be assured at some level of confidence that the technique would work in our domain of interest: software development and management.

Thus, the main issue is not the power of the repertory technique but a thorough understanding of the domain so that we could adapt the technique successfully and effectively.

So, one of the relevant questions is "*what is involved in this domain, and how can we apply the RP technique*?" The software development, like any other development projects is creativity-driven. The main component is termed as peopleware (Demarco and Lister, 2012). The average person are highly trained and well experienced. But as in any situation, there exist two other groups: the outstanding versus the weak, the quality-driven versus the error-prone, the greedy versus self-content, the quick versus the slow, the abuser versus the abused, etc. all represent the bipolar concept on constructs. What we finally construe is a collection of constructs revealed by individuals which can organized collectively. We would be able to characterize the institution as a whole, within the context of the four quadrants of the top grid (Figure 7).

There are a couple of issues (not exhaustive) on the construct system under investigation.

Selection of triads: One would think that since the elicitation is based on a specific exception that occurs during the life of the project, how would the decision maker (DM) select a triad of exceptions to work with, let alone many different triads? First, note that the DM is not anybody in the institution. The DM is either a top management member, or a SME with a wide range of responsibilities. The DM influences the success-failure of the project, since the DM heavily is involved in the project in many managerial, business and technical aspects of the project: planning, scheduling, funding, personnel, technology, training, support, skills, etc. The DM should and would be familiar with many prior exceptions during his tenure with the institution or elsewhere. So, there won't be problems that the DMs identify the other elements to work with the exception at hand.

The interviewer: According to many RG authors, the interviewer must be skilful. The interviewer can't suggest any construct. The interviewer's questions are based primarily on the DM's responses to further explore the DM's decision model in terms of laddering up or laddering down. The interviewer must faithfully scribe the information as it is given.

Rating the grid: If 1-5 scale rating is used instead of the two-value scheme suggested by George Kelly, is it possible that the DM fakes the rating? On a scale of 1-5, 3 being neutral or not applicable (N/A), 4 and 5 are normally assigned to the emerging construct, with 1 and 2 assigned to the implicit construct. The faking of rating, if occurred, will not impact the final analysis a great deal.

Analysis and evaluation: This is the task of the Oversight organization. Analysis can be done using a commercially available tool. Content analysis or statistical tools such as cluster analysis can be used to evaluate similarity and difference among the set of decisions on exceptions. The expected result from this analysis and evaluation offers an understanding on the consensus or otherwise among DMs on the issues, circumstances, actions, consequences, and values, and other factors surrounding the exceptions. We would see whether the decision is arbitrary or not, or whether decision makers are high-risk driven, etc. Again, the grid also can expose individual perception and thoughts of the decision maker on the exceptions. Thus, it could help discover any improper intention. That's the basis of our proposed psychological method adapted from Kelly and others.

5 CONCLUDING REMARKS

Our biologically-inspired investigation is driven by insights into the biological spectrum. On the surface, it appears that it is similar to previous investigations in the past in terms of insights and/or metaphors. For example, one could say that the Object-Oriented (OO) *inheritance* concept is drawn from Gregor Mendel (Alberts, 1998) who discovered the principle of inheritance, basis of genetics. The window *icon* was after Charles Peirce's theory of signs (Stanford 2006). Many OO *design patterns* were after Christopher Alexander's creation in the field of architecture (Alexander, 2002).

However, our approach is different in that the biological spectrum as a whole offers a global and integral scope. In the software project management domain, we can extent further to include exceptions caused by suppliers (e.g. primary or subcontractors) and other stakeholders in an IT development project (such as (1) the case of US Army Future Combat systems, (2) the US Air force ERP project which could have been pulled out sooner, rather than waited until after $1B already spent, or (3) the Marine Corps' GCSS project which could have avoided delay and budget overrun (Kanaracus, 2012).

We can include medical providers, insurers, patients, etc. such as in the case of Healthcare.gov, (Heusser, 2013) where each group is considered as population or community, and the whole system of systems as a business ecosystem. Interesting insights can be gained from this perspective. It allows the concept that humans (as organism) assembled in institutions (as a community) which run business as part of (business) ecosystems within some economy (biosphere).

Furthermore we couple the biologically-inspired framework with a psychologically-driven technique on decisions to characterize decisions, because we recognize that a wrongdoing, if not properly handled, can lead to other wrongdoings of higher criticality, and therefore aggravates the project health. This is worse especially when the decision makers try to hide their bad decisions in the process for one reason or another.

In the financial world such as the Barings Bank case, Nicolas Leeson was able to hide the loss of his first trading transaction between Osaka exchange and Singapore exchange (SIMEX) in the error account 88888 (Leeson, 2012), without management knowledge (maybe his Singaporean subordinates knew but did not report). He was also responsible for both the front office as a trader and the back office as a general manager of Barings Singapore (an oversight by his managers and executives in London Office). Both should be exposed earlier as exceptions for proper decisions, and the bank could have avoided collapse.

In the case of Enron, CFO Andrew Fastow, was able to offset losses in the Enron financial statements over many years using a complex structure of Special Purpose Entities (SPE) as hedging scheme (Powers, 2002). If the first warning (Mack in 1993) to Enron CEO Ken Lay, or if the following warnings reported by Enron's own accountants on the use of SMEs were not overridden or actually received proper attention, additional wrongdoings by Andrew Fastow and his team would have been avoided (Powers, 2002).

Other solutions such as prevention of corporate fiascos, can be formulated by looking more closely at the biological spectrum as we have discussed in section 2 and exemplified in the case of Barings Bank and Enron bankruptcy, and also as found in (Nguyen, 2014).

REFERENCES

Alexander, Christopher, 2002. *The Nature of Order*, Pub. The Center for Environmental Structure.

Alberts, Bruce, Dennis Bray, Alexander Johnson, Julian Lewis, Martin Raff, Keith Roberts and Peter Walter. (1998). *Essential Cell Biology*, Garland Publishing.

Beer, A. Stafford,1972. *Brain of the firm: Managerial Cybernetics of Organization*, The Penguin Press, London 1972.

Bourne, Humphrey and Mark Jenkins, 2005. Eliciting Managers' Personal Values: An Adaptation of the Laddering Interview Method, *Organizational Research Methods* 2005 8: 410

Calleam, 2014. Why Projects Fail: Facts and Figures, *Calleum Consulting*. .http://calleam.com/WTPF/?page_id=1445

Cannon, Walter, 1963. *The wisdom of the body*, The Norton Library, Norton & Company, 1963

Charette, Robert N., 2005. Why Software Fails? *IEEE Spectrum*. Volume 52, Issue 9. September 2005.

Demarco, Tom and Timothy Lister, 2012. *Peopleware: Productive Projects and Teams*, http://javatroopers.com/Peopleware.html.

Foster School of Business, Washington, 2010. A decade's worst financial scandal*s*, April 2010

Fransella, Fay and Bannister, Don, 1977. *A Manual for Repertory Grid Technique*, Academic Press, 1977.

Fromm, Martin, 2004. *Introduction to the Repertory Grid Interview*, Waxmann Verlag.

GAO, 2009. *Review of Future Combat System Is Critical to Program's Direction*, http://www.gao.gov/new.items/d08638t.pdf.

GAO, 2012. 12-565R: *DOD Financial Management* http://gao.gov/assets/590/589796.pdf

Galorath, Dan, 2012. Software Project Failure Costs Billions. http://www.galorath.com/wp/software-project-failure-costs-billions-better-estimation-planning-can-help.php June 2012

Gross, C. G., 1998. "Claude Bernard and the constancy of the internal environment" *Neuroscientist* 4 (1)

Heusser, Mathew, 2013. 6 Software Developments Lessons from Healthcare.gov's Failed Launch, http://www.cio.com/article/743366/6_Software_Development_Lessons_From_Healthcare.gov_s_Failed_Launch

Kanaracus, Chris, 2012. The Scariest Software Project Horror Stories of 2012. http://www.cio.com/article/print/721628

Kandel, Eric and James Schwartz, Thomas Jessell, 2000. *Principle of Neuroscience*, 4th edition, (Eds), McGraw-Hill, 2000

Kelly, G., 1963. *A Theory of Personality: The Psychology of Personal Constructs*. New York: W. W. Norton & Company

King, Roger J.B., 1996. *Cancer Biology*, Longman, 1996.

Christof, Koch and Davis, Joel L., 2003. *Large-Scale Neuronal Theories of the Brain*, Christof Koch and Joel L. Davis (Eds), MIT Press

Korenini, Bojan, 2012. Conducting Consisting Laddering Interviews Uisng CLAD, *Metodološki zvezki*, Vol. 9. No.2.

Krigsman, Michael, 2008. 68% of IT Project Fail, Beyond IT Failure, December 2008, http://www.zdnet.com/blog/projectfailures/study-68-percent-of-it-projects-fail/1175

Leeson, N., 2012. *Rogue Trader*, Little Brown Book Group.

Mack, T., 1993. Hidden Risks, *Forbes*.

Maturana, Humberto R. and Francisco J. Varela., 1980. *Autopoesis and Cognition: The Realization of the Living,* D. Reidal Publishing Company.

Mieritz, Lars, 2012. Gartner Survey Shows Why Project Fails? June 2012, http://thisiswhatgoodlookslike.com/2012/06/10/gartner-survey-shows-why-projects-fail/

Moore, J.F., 1996. *The Death of Competition – Leadership and Strategy in the Age of Business Ecosystems*, Harper Business, 297.

Nguyen, T. N., 2014. A Different Approach to Information Management by Exceptions: Towards the Prevention of another Enron, *Information & Management*, Vol 51 Issue 1.

Nguyen, T. N., 2006. A Decision Model for Managing Software Development Projects, *Information & Management,* Vol 43 Issue 1.

Pressman, Roger S., 2010. *Software Engineering: A Practitioner's Approach.* 7e. McGraw-Hill.

Powers Jr., W. Report of Investigation, 2002.

Stewart, Valerie., 2010. http://valeriestewart-repertorygrid.blogspot.com/

Raven, Peter H. George B. Johnson, Jonathan Losos, Susan Singer, 2008. *Biology*, 8e, McGraw-Hill

Reilly, Sean, 2012. How the Air Force blew $1B in a dud system. Federal Times, http://www.federaltimes.com/article/20121126/DEPARTMENTS01/311260009/How-Air-Force-blew-1-billion-dud-system

Schadler, Ted, 2013. Healthcare.Gov's Failure Starts with Leadership, Not Technology, http://www.forbes.com/sites/forrester/2013/10/25/healthcare-govs-failure-starts-with-leadership-not-technology/

Stanford Encyclopaedia of Philosophy, 2006. http://plato.stanford.edu/entries/peirce-semiotics/

Stewart, Valerie, Andrew Stewart with Nickie Fonda, 1981. *Business Application of Repertory Grid*, McGraw-Hill Books Co.

von Neumann, 2000. *The Computer and The Brain*, 2e, Yale University Press.

von Neumann, John; Burks, Arthur W., 1966. *Theory of Self-Reproducing Automata*, University of Illinois Press.

Weiner, Norbert, 1948. *Cybernetics or Control and Communication in the animal and the machine*, The Technology Press.