



The Devil's in the Decisions

Abstract

In a paper that looks at software development from a new perspective, the authors ask us to consider the central role that decision-making plays in software projects. Teams are aware of the key decisions made in a project, but many will not have considered that decision-making is the pervasive thought process that affects even the smallest detail of a project's outcome.

Given that teams that consistently make good decisions are likely to succeed, while teams that make bad decisions are likely to fail, the paper considers the factors that affect the ability of the team to make effective decisions.

Teams wanting to improve their performance need to understand the different elements that lead to bad decisions and consider ways of tuning their practices to optimize their decision making capabilities. To assist teams who are interested in improving their capabilities, an assessment tool is provided that allows teams to explore the issues that could prevent from them making effective decisions.

By understanding the factors that drive effective decision making and the relationship between those elements, project teams will be better able to identify weaknesses in their decision making capabilities and address those weaknesses before the project is compromised by a set of bad decisions.

The Devil's in the Decisions

Decision-making is so ubiquitous that we rarely stop and think about how central its role is in software projects. We are aware of the key decisions made in a project, but we don't consider that decision-making is the pervasive thought process that affects even the smallest detail of a project's outcome.

It's not a quantum leap to theorize that the success of a project is correlated to the effectiveness of the team's decision-making capabilities. Consistently make good decisions and the chances are you'll succeed. Make bad decisions and your chances of success are greatly diminished.

Traditionally, Project Management and Quality Management disciplines approach software development as a process-based activity¹. Much like manufacturing, projects are viewed as a set of discrete tasks that lead from one to the next and result in the final product. In software development, these tasks typically include Requirements, Design, Development, Test and Deployment. Depending on the methodology being used, the tasks are organized around a framework that establishes the order of the tasks, the methods to be used and whether the tasks are to be performed once, more than once, sequentially or in parallel.

While this is a convenient view, it masks the inner "atomic structure" of a project. At the atomic level, a project is more accurately represented as a large-scale complex web of inter-related decisions.

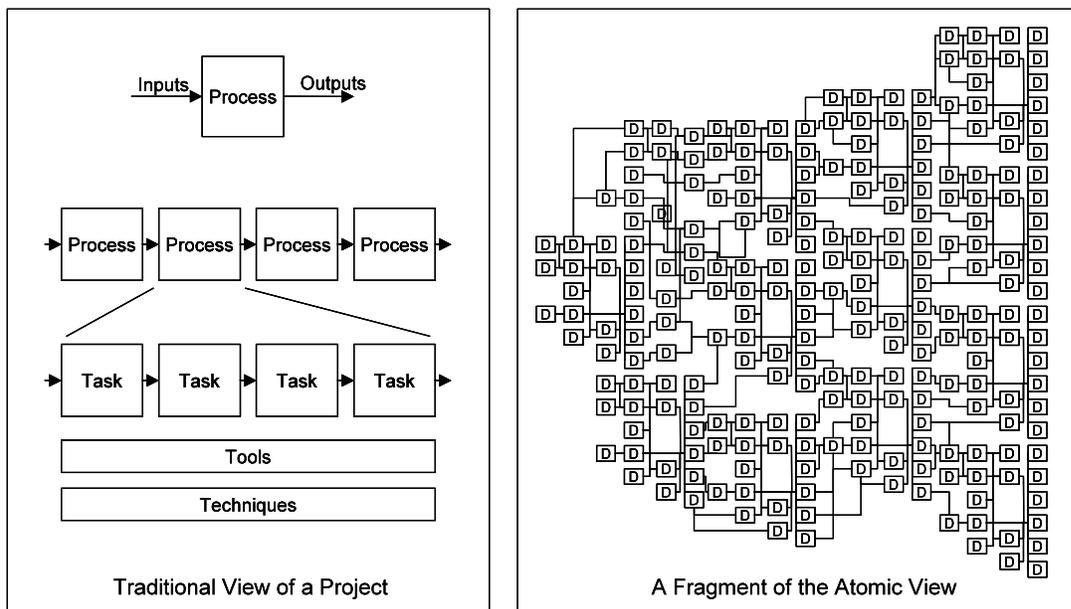


Figure 1 – A Contrast in Views

¹ Grady, Robert B, Successful Software Process Improvement, 1997, Prentice Hall

Figure 1 provides a graphical representation. The panel to the left shows the traditional view of a project. Work is represented as a set of process and tasks that are sequenced in such a way that the work follows a logical path from start to end. The panel to the right shows a small fragment of the project at the atomic level. A chain of decisions flows from left to right, with individual decisions acting as the stepping-stones that lead from project initiation to the delivery of the final product. Each box represents an individual decision and the connections between boxes show the relationship between decisions and the flows of information, as one decision becomes the basis for determining the next question or set of questions needing to be answered.

At the atomic level the clean delineation between different tasks implied by the traditional view breaks down. Although at an overall level early decisions are more likely to be requirements type decisions, while those later in the chain are more likely to be implementation type decisions, the sequence and structure of the decisions at the atomic level is more complex than the traditional view would imply. Often planning, requirements, design and even implementation type decisions are made in parallel and certain decisions may affect multiple aspects of the project rather than just one.

This heavily interconnected chain of decisions poses a significant problem for software development teams. If a bad decision is made early in the project, it can have a cascading affect for subsequent decisions that follow. The significance of a bad decision is compounded as more and more subsequent decisions are based on or influenced by the bad one².

It is important to note that for software projects, many bad decisions are ones that are not made, or are made far too late in the project. As an example, in one study the U.S. Navy found that 31% of all requirements errors were errors of omission³. Omissions are easy to make if the team lacks sufficient subject matter domain knowledge or technical knowledge in order to identify and ask the right questions at the right time.

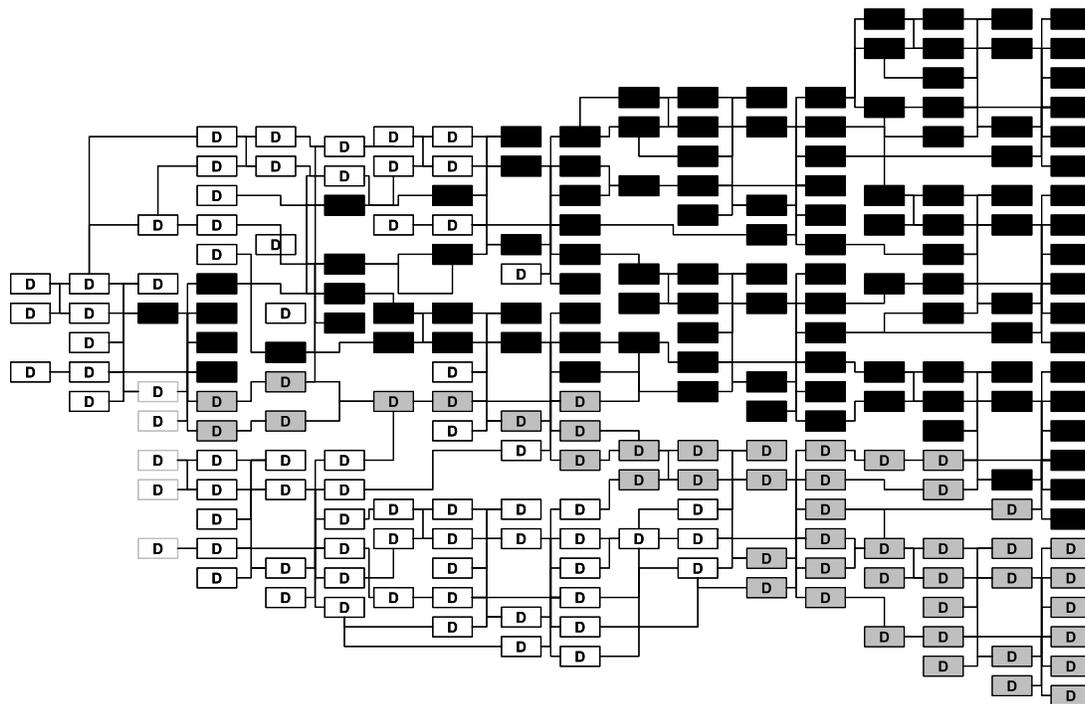


Figure 2 - Cascading Effect of a Bad Decision

² Grady, Robert B, An Economic Release Decision Model: Insights into Software Project Management (Proceedings of the Applications of Software Measurement Conference), 1999, Software Quality Engineering

³ Davis, Alan M., Software Requirements: Objects, Functions and States, 1993, Prentice Hall

Figure 2 shows the potential cascading affect of making a bad decision early in the project. White boxes represent good decisions; black boxes bad decisions and gray boxes represent sub-optimal decisions. As shown, the effect of a bad decision early in the project can have far reaching consequences throughout the project. Although ideally project teams would go back and correct every bad decisions as soon as it were identified, in practical terms teams can only move forward from where they are and projects rarely have the budget or time to go back and correct anything other than minor errors. As a result, unless caught and corrected early, many projects are compromised by the failure of one or more key decisions made early in the project.

Clearly, a key goal for software organizations is to build an environment in which individuals and teams are able to make decisions as effectively as possible and an environment where bad decisions are identified and corrected as soon as possible. Failure to do so results in a larger number of bad decisions, with the corresponding consequences.

Clearly some teams are able to make better decisions than others. Many studies have shown that some teams are more effective than others at making decisions⁴. We believe that many of the factors that have been identified as drivers for team effectiveness are key to building an infrastructure for effective decision-making. The key question to be answered is: "What are the elements that allow some teams to consistently make better decisions than others?"

A Recipe for Successful Decision Making

Over the years, the authors have participated in and observed a wide range of projects and teams. By distilling those experiences, we have identified a number of ingredients as consistent themes in the most effective project teams. These ingredients cover a number of organizational, process and people factors. Our observations show that successful projects are those where all of the ingredients are present and are sufficiently robust. As a counterpoint, our observations show that the absence of one or more ingredients has been a recurring theme in projects that have encountered serious difficulties.

By recognizing and understanding these ingredients and relationships between them, Project Managers have a powerful tool that allows them to identify where their team may encounter problems making effective decisions. The identification of such issues early in the project allows the Project Manager to address the problems, thereby avoiding the pain of having to correct bad decisions long after they have been made. These key ingredients are:

1. **Subject Matter Domain Knowledge** - Subject matter domain knowledge is a key foundation stone for successful projects. Although not every team member needs to have subject matter domain knowledge, the team as a whole needs to include the necessary knowledge (or access to it) and the flows of communications within the team need to ensure that the knowledge reaches those in the team that need it in order to perform their work. Where the team lacks such knowledge they lack the situational awareness required to ask the right questions and make the right decisions. When key questions are missed, the seeds of a troubled project are sown. Such gaps lead to flaws in the project scope, inappropriate architectures, designs that fail to meet user requirements and a host of other problems.
2. **Technical Knowledge and Skills** – The implementation of a system is dependent on the ability of the team to implement the required solution using the technical tools available to them. The stronger the team's knowledge of the technology, the faster they will be able to work, the more the team will be able to optimize their designs and the less likely the team is to encounter unforeseen technical problems.

⁴ McConnell, Steve, Rapid Development: Taming Wild Software Schedules, 1996, Microsoft Press. Steve cites numerous studies on team productivity, with variations across teams ranging from 2.6:1 to 5.6:1.

3. **Leadership and Direction** – While the two knowledge domains (subject matter and technical) fuel the project, the team needs to share a common picture of what they are trying to achieve and how they will get there if they are to make effective decisions along the way. Leadership draws together the big picture and communicates it to the team while helping the team focus their decision-making activities to align with the project goal. As changes occur, leadership alters the course as necessary and keeps the channels of communication open to ensure on-going alignment. To ensure the team are able to make effective decisions, the big picture needs to cover a number of different aspects of the project including the vision of the end product, the overall system architecture as well as the project plan of how the system will be built. Without the big picture, team members lose the ability to correctly align their decisions with those made by their colleagues, thereby damaging the integrity of the project as a whole.
4. **Ownership and Commitment** – Clear ownership of decisions and a willingness to commit to decisions are further elements required to establish a sound decision-making environment. Without the clarity of clear ownership, decisions fall through the cracks as individuals think someone else is responsible. Where there is a lack of willingness to commit to decisions, time, budget and energy are sapped from the project while the team waits for clear decisions to be made.
5. **Engagement and Participation** – Having the right stakeholders engaged in the project from the start is another key ingredient. Stakeholders need to include those providing the project resources, those affected by the project outcomes and those performing the work. As noted earlier, decisions made early in the project become the foundations for all subsequent decisions. Where key stakeholders are omitted from the decision making process early in the project, critical information can be missed. Similarly if key team members don't have sufficient time to participate effectively, either the project will struggle to gain traction, or decisions will be made superficially with the corresponding increase in risk that ineffective decisions will be made.
6. **Shared Understanding** – Everyone's been told, "communicate, communicate, communicate", but effective teams go beyond communication by building shared understandings. If a team is to make effective decisions, they need to share common understandings among themselves, with their stakeholders and with their leadership. Shared understandings require skills from both the communicator and the recipient as they both ensure they have common understandings. Shared understandings are necessary if the team is going to be able to coordinate at the detailed level. A failure to build shared understandings undermines teamwork and leads not only to bad decisions, but also problems detecting when bad decisions have been made.
7. **Integration** – The ability of a team to maintain the integrity of the whole is another critical element. Many will have seen projects in which different parts of the team have headed off in different directions. Integration problems are one of the most common failures of projects in trouble.
8. **Governance and Control** – Although words such as governance and control have negative implications for some, all projects have some form of control. The nature of the methods used may vary dramatically, but to one degree or another all successful projects have some mechanisms in place to control and govern what the team does. Such control acts as the checkpoint that allows the team to continually ensure alignment of their decisions as well helping the team in identifying where bad decisions have been made as soon as possible so that they can be corrected with minimal lost time or budget.

Decision Engine Assessment

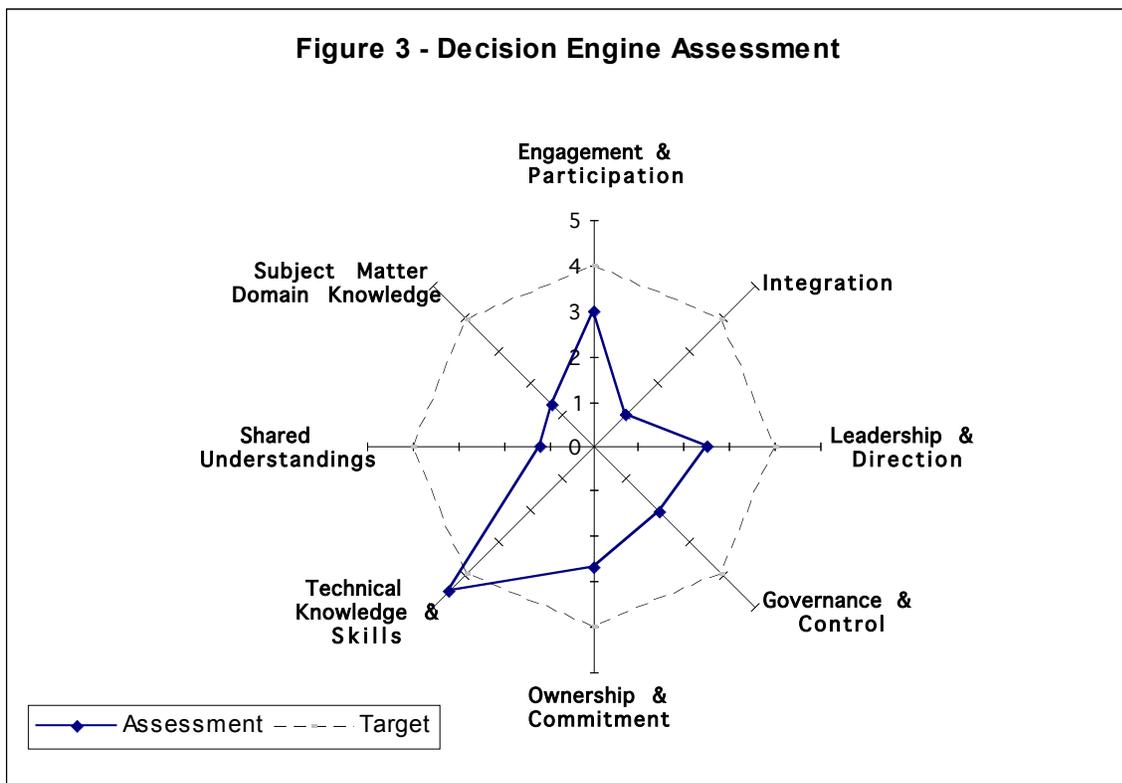
We can view these ingredients as one system in order to evaluate the ability of the team as a whole to make effective decisions. Our observations show that only when the ingredients are present and robust is the team able to make effective decisions. Weakness in any one ingredient can severely damage the ability of the team as a whole to make

sound decisions. A good analogy is to view these ingredients as an eco-system. Much like an eco-system, when all elements are strong, the whole eco-system thrives. If one or more elements are compromised, the eco-system as a whole can be impaired. We call this eco-system the Decision Engine. Many of the root causes that lie behind failed or “challenged” projects can be traced back to problems in the project’s Decision Engine.

One of the problems seen in failed projects is that the Project Manager is not aware of gaps in the project that prevents the team from making effective decisions. A flood of emails and issues swamps the Project Manager, distracting their attention such that they lose sight of the big picture for which they are responsible: building up a team that can make effective decisions.

To help maintain an awareness of the team’s decision-making capabilities, it’s a useful exercise for teams (and even organizations) to evaluate their project against the ingredients of the Decision Engine presented above. By considering the relative strength of each element, a team can gain a picture of the issues that could prevent them from making effective decisions and hence reduce their chances of success.

Figure 3 (see below) shows such an assessment. The different ingredients in the Decision Engine are shown as arms on a radar chart and are ranked with scores from one to five. A ranking of five means that the element is strong and a ranking of one represents weakness. A set of assessment questions we have successfully used to determine rankings for each ingredient is included at the end of this article. Managers, Project Managers or team members who are interested in evaluating the abilities of the team can use the tool. In addition, the tool can be used to facilitate group sessions between team members to understand differing perspectives that may further help the team as a whole gain insights to how effectively the team is working.



Our observations show that successful projects typically score a four or above for most of the elements. Hence for discussion purposes a target line has been drawn around the chart at a value of four.

The authors have worked with a number of project teams to perform the assessment. The assessment shown in Figure 3 is a typical result. The project team shown has strong technical knowledge but has weakness in the other elements. It is reasonable to predict that such a team will encounter significant “challenges” in their efforts to deliver. Using the assessment as a tool for initiating discussion and reflection among team members projects we have worked with have initiated many changes in their approach in order to address weaknesses identified through the tool. Changes made include organizational changes to ensure better oversight of work, increasing the number of reviews to validate decisions earlier, adopting a program of lunch and learns to help share domain knowledge, changes to ensure stakeholders are involved more closely with key project decisions and many other changes.

Organizations that would like to improve the performance of their teams should consider looking at the Decision Engine to determine areas where actions could improve performance the most. The tool can be used at project start up and periodically throughout the project to help the Project Manager maintain situational awareness as the dynamics of the project unfold. The tool can assess the overall project or to zoom into one particular area. Rather than using the assessment as a snapshot that may predict success or failure for a project, it can be used as a tool to identify areas to shore up in an effort to increase the team’s chances of success for a given project.

Dimensional Factors Influencing the Decision Engine

A healthy Decision Engine acts as the power plant of progress. Having looked at the ingredients in play, though, it is worth understanding that the way the engine works for two fresh college graduates building the next killer application in their basement suite won’t be the same as it is for a large team trying to put an astronaut on Mars. The key is not the level of process formality used or applying a particular organizational structure, but the appropriate tuning of the team’s practices to the needs of the project. To understand the difference between the college graduates and the Mars explorer team, we need to consider a number of dimensions.

The first of those dimensions is size (i.e the number of functions needing to be developed and the complexity of those functions). As project size increases, so too does the number of decisions needing to be made. Usually this means more people are involved and with more people comes the need for greater coordination. Communications is the oil that lubricates that coordination. Where a quick discussion and a whiteboard might suffice for the college grads, the Mars explorer team would need far more extensive structures in place to achieve the same level of communications. Similarly more formal methods of communications may be required for many other reasons such as where teams are distributed or where a project calls for coordination between organizational groups that may have different methods or styles of working.

The second dimension is time. Larger projects usually take more time. This means the timeframe between early decisions and those made later in the project is extended. Extended timeframes tax memories unless appropriate steps have been taken to ensure the project has a project memory independent of the brains of the individual team members.

The consequence of failure is the third dimension to be considered. Where failure involves loss of life, significant monetary loss or severe damage to the reputation of an organization, the practices employed need to take this into account. The usual method for preventing such failure involves increasing the number of reviews and validations that take place. Such checkpoints validate decisions and help ensure that the team is proceeding based on a solid base. Although all projects benefit from such checks, the timing and formality of the reviews is often driven by the consequences of failure.

Perhaps the final dimension is the structure of the web of decisions itself. Not all projects share a common structure. Depending on a number of requirements, design and technological considerations, some projects have broad but shallow chains of decisions while others have narrower, but deeper chains. One-way this structure becomes manifest is to think through the number of architectural decisions made in the project. Architectural decisions are significant because many subsequent decisions will be based upon them. A flaw in an architectural decision has the potential to negate all of the decisions that are built upon that decision. Projects that are heavily dependent on architectural decisions may require a different approach from those that have few such decisions.

When using the Decision Engine assessment tool, teams should consider the dimensions listed above and how these affect the scores they assign to questions in the questionnaire. In general it is reasonable to say that as project size, duration and risks rise, the level of process formality, leadership, governance and communications will also need to increase. Similarly projects that involve significant numbers of architectural decisions will also need to reflect that need when assessing rankings for individual questions. Practices that might suffice for the graduates and hence warrant a score of 4, may only warrant a score of 1 for a much larger team.

Conclusion

Although projects are rarely represented as a large-scale decision-making activity, decision-making is and likely always will be the core activity for software project teams. Effective decision-making requires both expertise and appropriate process in order to support the decision-making.

Decision Engine assessment is a useful tool to help projects or organizations examine their approach to software development, and to identify areas where changes may improve the team's capability for making effective decisions. Rather than focusing at the tool, technique or methodology level, the Decision Engine asks us to consider our problems at their most fundamental level: the decision-making level. Do you have the appropriate knowledge on the team and do you have the appropriate skills? Is your team communicating effectively and are their efforts properly aligned?

Organizations that are genuinely interested in improving their delivery capabilities need to focus on both building the expertise of their team as well as consciously managing their decision-making capabilities.

This paper was originally published at the Pacific Northwest Software Quality Conference, Portland, Oregon, 2007

Copyright © Robert Goatham & Jim Brosseau 2007

About Robert Gotham

With 20 years in the IT Industry and a broad range of experience in both Project Management and Quality Management, Robert Gotham has long been an advocate for quality. Robert has an Honors degree in Aeronautical Engineering from Kingston University, UK and became a Certified Quality Analyst in 1999. He has broad international experience and established the IT Quality function for Singapore Airlines. In addition he has played an active role in quality programs in many organizations. Always pragmatic, Robert's mix of fundamentals and from the trenches experience leaves audiences with a new and distinct perspective on the key issues affecting the IT Industry.

About Jim Brosseau

Jim has been in the software industry since 1980, in a variety of roles and responsibilities. He has worked in the QA role, and has acted as team lead, project manager, and director. Jim has wide experience project management, software estimation, quality assurance and testing, peer reviews, and requirements management. He has provided training and mentoring in all these areas, both publicly and onsite, with clients on three continents.

He has published numerous technical articles, and has presented at major conferences and local professional associations. His first book, [Software Teamwork: Taking Ownership for Success](#) was published in 2007 by Addison-Wesley Professional. Jim lives with his wife and two children in Vancouver.

About Clarrus

It is often the same class of challenges that are the root cause of most business inefficiencies. A key aspect of these challenges is that they manifest themselves differently in every organization. That's where we come in.

Clarrus is a down-to-earth, value-driven company with a focus on improving project teams and the projects they work on, across all industries.

Clarrus is unique in that it understands the interplay of human dynamics and project mechanics. Our consulting services and workshops bring a practical and proven approach to increasing the effectiveness and satisfaction of your project teams and delivering results.

Clarrus believes in the principle of teaching people to fish. Our approach nets its richest results in complex environments where multi-disciplinary teams bring diverse perspectives to the table. But, we also help small teams thrive, too. In a nutshell:

- We harness the best of best project practices and apply only what's needed for tangible short- and long-term results.
- We create an environment of trust so our sometimes tough questions can open the doors to peak performance.
- We provide inventive and practical ways for your teams to approach challenges.
- We secure the engagement of your project teams by focusing on issues where the impact is greatest.

Contact us today.

- On the web at <http://www.clarrus.com>,
- By e-mail at info@clarrus.com, or
- By phone at +1 (604) 540-6718.

January 1, 2010 - Version 1.7