The 5 Reasons Why

Most Projects Fail

And what steps you can take to prevent it

By Mark A. Taylor, President & CEO,

TAYLOR Systems Engineering Corporation

Presented by
TAYLOR Systems Engineering Corporation
40800 Five Mile Road
Plymouth MI 48170

(734) 420-7447
www.shippingsys.com

According to research by The Standish Group, **74% of all information technology projects fail**.[1] A survey by the Warehouse Education Research Council (WERC) and Arthur Anderson further concluded, "Only 40% - 4 out 10 companies were satisfied with the installation of their projects."

If you are considering a new project, it will probably fail. Everyone will be impacted – the CEO, management, and your customers. The result is that you will waste money, time, and resources. I have seen it happen. I have witnessed the upset, frustration, lawsuits, bankruptcies, and despair.

Over the past 13 years I have led over 2,000 systems projects with a 99.99% success rate. As I reflect on those experiences and the lessons learned, I have found that there are 5 primary reasons why most projects fail. My intention in writing this paper is to share that knowledge with you so that you can apply the positives and avoid the "pitfalls" in order that you can make a difference and a positive contribution for your organization.

The number one reason that most projects fail: Lack of Leadership.

It is difficult to believe that an organization can commit thousands and even hundreds of thousands of dollars to a project and then fail to lead that project to success. Yet, as I look at the reasons projects fail or succeed, leadership is the one common denominator. What is leadership? Professor Warren Blank provides a powerful metaphor:

> "Leadership and physics share a common focus. Physics explains the energy, matter, and motion that define how the universe works. In the same way, leadership is the power that galvanizes human energy and translates it into action. So the exercise of leadership can be viewed as the practice of human physics."[2]

The leader is responsible for the vision of the future and inspiring a shared vision with all the people that are affected by the project. This is not easy but is critical to the success of the project. It does not necessarily have to come from the CEO but if the CEO does not share the vision, the project will fail.

Last summer I was involved in a project to implement new project management software. The CEO's vision was that this software would improve customer service by giving clients visibility into the status of a project at any time through the web. It would also provide him with a means to see how his company was doing on all projects at any time.

---

[1] PC Week, December 5, 1999
[2] Blank, Warren, The 9 Natural Laws of Leadership, 1995, p. 27

In fact, it included a digital dashboard that displayed the status on a daily basis in bold color. He could look on his computer and see projects that were on time designated with green lights, those that had failed to deliver a task on time were in red, and those that were close to a deadline were in yellow. Unfortunately, his engineering manager did not share his vision. In my opinion, the engineering manager did not want the CEO to see the status reports because he would no longer be able to cover his butt. They were definitely not on the same team; the project was doomed to failure and ultimately did fail to get implemented. Professor Blank goes on to say the following:

> "Natural Law 1: A Leader Has Willing Followers
> No leader exists without gaining the support of others. Typically all glory and grandeur goes to the leader, and being a follower is usually thought of as a second-class or lower-status role. The first natural law of leadership changes our view of followers because it recognizes the collegial, partnering role they play. Followers are allies who represent the necessary opposite side of the leadership coin. "[3]

My experience over the years has convinced me that for a project to succeed all parties impacted need to be a part of the project team or represented by someone on the team. And yes, this especially includes the end users! If an individual does not share the vision and is not in a partnering role, they will invariably find a way to sabotage the project. The leader must align the team and insure that all members are on the same page and committed to the project's success. The leader must take care of the politics and the team dynamics. George Eckes, the primary consultant for General Electric in their Six Sigma Quality initiative agrees:

> "Concern 4—Ignoring Team Dynamics as a Root Cause of Project Failures
> By far the greatest source of team failures is poor team dynamics and poor facilitative leadership behaviors. Approximately 60 percent of teams that fail have these as their major reasons. I am reminded of an old adage spoken by some of my organizational friends. "The hard stuff is the easy stuff; the easy stuff is the hard stuff." The most common problem areas we have encountered are:
>
> *Meeting skills*.
> *Setting agendas*.
> *Determining the meeting's roles and responsibilities*.
> *Setting and keeping ground rules*.
> *Facilitative behaviors*."[4]

[3] Blank, Warren, The 9 Natural Laws of Leadership, 1995, p. 11

[4] Eckles, George, The Six Sigma Revolution, John Wiley & Sons, 2000, p. 244

The second reason that projects fail: Lack of Clarity.

This is related to the first reason in that leadership is about providing a clear, compelling reason why a particular project or initiative should happen. And, as you will see, all of the reasons that projects fail are related. However, clarity stands on its own because projects fail to deliver the results when the results have never been specified.

> "In fact, it is in step 2 that many projects go wrong. I often say that projects don't fail at the end, they fail at the beginning—right here in step 2. The reason is that we take for granted that we all understand the problem perfectly, when this is not the case at all."[5]

About five years ago I met with an electronics wholesale distributor that wanted to implement a bar code system. The project team had representation from the warehouse, shipping, receiving, accounting, information services (IS), an outside consultant that was an expert in their enterprise resource planning (ERP) system, and the vice president of operations, the team leader. Every group that was impacted was to be represented at all of the meetings.

A conceptual design was developed that integrated bar codes with the ERP system. The system worked in such a way that when product was received it was bar coded and entered into the ERP inventory system, and when it was shipped out it was scanned and taken out of inventory. After the design was approved, a detailed functional specification was written and signed off by all team members. The system was designed and went into production; the system was tested, accepted, and the users were trained. The project came in on time and on budget.

Everything *seemed* perfect. However, soon after implementation, the president of the company called me and asked me to meet with her. She was very upset. The bar code system was not fulfilling her expectations, and not working as she had envisioned. As such, she felt it was, "a complete waste of money". What went wrong? The problems were many. First, she had not been present at any team meetings. As such, her vision for the goal of the technology was not communicated and the software was not programmed to do what she wanted it to do.

A survey conducted by Compass for the London School of Economics reports the following:

> "**CEO's felt that only 33% of their IT spending resulted in an improvement to the bottom line**."[6]

[5] Lewis, James P, Mastering Project Management, McGraw Hill Professional Book Group, 1998 p 63
[6] Computing Canada, June 8, 1998 v24 n22 p.9

In order for projects to be effective, it is vital that an expected result be clear. Dr. E. Rechtin, professor of systems architecture and engineering at the University of Southern California elaborates in the following way.

> "The purpose of systems is to produce added value, system functions, that none of elements alone, could produce. Indeed, the reason that systems are built at all is to achieve those results. As examples, the principal system function of an automobile is transportation; the principal system function of the human body is life. Now, although each of the elements (heart, lungs, brain and so on) has its value, none of them produce the system function and, in many cases, none of them can operate by itself. Each must be interconnected to others in order to function at all. The added value of a system, therefore, is not only in providing system functions but in also providing the infrastructure necessary for its elements to operate."[7]

The third reason that projects fail: Little or no Due Diligence is performed.

Sometimes we need to have a project implemented *yesterday,* yet we don't have the time or resources to first do our homework. While it is certainly possible to "fast track" a project, many projects fail because somebody did not do the research necessary to make proper decisions.

A few months ago I received a telephone call from an operations manager of a fulfillment company that needed a system within 30 days in order to secure new client business. He informed me of his plan to implement a software package that would produce a compliant bar code label for each package that was to be shipped. I asked him why he had chosen that particular software package and he said that it was certified and was recommended by one of his current vendors. As I asked him additional questions, I discovered that he was going to fulfill 50,000 packages at a time, in a batch. I suggested that the certified package that he was considering was designed to process packages one at a time and inquired if that was the process he really wanted. After our conversation he realized that it would have cost him a fortune in labor to go with that certified package and that a system that could produce mass quantities of labels at one time would be a much better solution. He was fortunate that we spoke because I am certain that if he had gone the other way his project would have failed.

In every project there is the risk of failure. The best way to reduce risk is to do the research and discover as much as possible about the project up front. Many times we do not have the time nor expertise and need to consult with an expert, or a consultant that we

[7] Rechtin, Eberhardt, The Journal of the National Council on Systems Engineering, July/September 1994, p. 36

trust. An outside, third-party observer that can add value by providing information that we don't even know that we don't know about a particular project.

The Johari Window[8], named after its creators Joseph Luft and Harry Ingham, depicts aspects of a relationship that I equate to risk in an easy to understand graphic (see table 1.) Imagine a window with four panes. You are the customer and I am the expert.

1. The OPEN pane of the window are the things about the project that I know about and that you know about. Particulars such as, specific results, resources, and the budget. These are the things that are openly revealed. This corresponds to the win-win situation in game theory, because the more knowledge we have in common, the greater the chances are of a successful project/end result.

2. The BLIND pane of the window is where much of the risk is. You don't know what you *don't know*, but I know factors that could have an impact because of my experience, competence, research, and/or time that I have dedicated to the project. The advantage of utilizing an expert is that it reduces the risk of the project failing. As you gain wisdom due to my knowledge your BLIND quadrant shrinks. By the way, this is also the reason to have a project team that includes everyone potentially impacted by the implementation of the project. An end user may see something that no other person has seen or even considered. A team, with representatives from the varying disciplines, can also reduce blind spots.

3. The HIDDEN pane of the window are considerations that you, the project owner, keep to yourself. These might include your fears about whether the project will succeed, your relationship with your boss, your insecurity about your competence; the conversations you may consider private. You know these things and I don't. As we get to know and trust each other, you will then feel more comfortable disclosing more intimate details about yourself. This process is called: "Self-disclosure."

4. The "unknown" quadrant represents things that neither I know, nor you know. For example, you were working with a project team and one of the members disclosed that they knew how to place a 2 dimensional bar code on a sheet of paper with their laser printer. You didn't know that they knew how to do this because you did not know and they did not know that it was useful or important. This enabled us to design the project in a way that neither of us imagined. As we both attempted to understand its significance, a new awareness emerged, known to neither of us before the conversation took place. This is where speculation in a spirit of openness with an intention of resolution can often reveal new possibilities, not previously known to self or others.

[8] Luft, Joseph (1969). "Of Human Interaction," Palo Alto, CA:National Press

Table 1

	Known to self	Not known to self
Known to others	1. **Open**; I know; you know	2. **Blind;** I don't know; but you know
Not known to others	3. **Hidden**; I know; you don't know	4. **Unknown**; I don't know; and you don't know

The other advantage of using an expert may be that they can focus on the project. Most projects are not a person's full time job. It is easy to get caught up in what I call the "tyranny of the present". You know what I mean, the everyday emergencies that seem to prevent getting done what needs to be accomplished for the project to be successful. It is what Stephen Covey refers to the urgent, but not the important.[9] (See table 2)

[9] Stephen R. Covey, A. Roger Merrill, Rebecca R. Merrill, First Things First. To Live. To Love. To Learn. To Leave a Legacy, Simon & Schuster, 1994

Table 2

I Urgent - Important • **Crises** • **Pressing problems** • **Deadline-driven projects, meetings, preparations**	**II Not Urgent - Important** • **Preparation** • **Prevention** • **Values clarification** • **Planning** • **Relationship building** • **True re-creation** • **Empowerment**
III Urgent - Not Important • **Interruptions, some phone calls** • **Some mail, some reports** • **Some meetings** • **Many proximate, pressing matters** • **Many popular activities**	**IV Not Urgent - Not Important** • **Trivia, busywork** • **Junk mail** • **Some phone calls** • **Time wasters** • **"Escape" activities**

A project's success depends on the important tasks getting done on time with the time that it deserves.

The fourth reason projects fail: Lack of Accountability.

Most vendors, consultants, or project managers will promise to produce a system that fulfills a functional specification - but not a specific business result. So this relates to the second reason, clarity. By accountability, I mean that there is a promise to produce the results. In other words, the project is complete when the results are visible, which means that they also have to be measurable.

A number of years ago I became involved in a situation with an automobile parts distributor that was angry because the system he had paid a lot of money for was not working. The IT vendor that provided the system claimed that it was working exactly as was written in the functional specification that the CEO had signed. Unfortunately, it was not printing the invoices, which was the result he desired. The IT vendor said it was the client's responsibility to get the printer driver and this CEO did not have a clue what a printer driver was. The vendor was able to manipulate the client because he knew that the CEO didn't know. The CEO felt helpless and frustrated. This is not uncommon; in fact, The Wall Street Journal reports,

> **"Eighty percent of CEO's recently surveyed in The Wall Street Journal say they cannot adequately evaluate their chief information officer's performance because they do not understand what the CIO does."[10]**

The problem is that most vendors do not take responsibility for the result. And they get away with it.

What if his vendor was willing to really put its neck on the line and guarantee that the system solution would produce the business results that the CEO envisioned/hoped for?

What if that company took responsibility for the entire system (i.e. software, processes, people and environment); would the project succeed?

What if there was no risk on the part of the CEO making the decision? Would he be able to make a decision faster and sleep better at night?

What if that vendor were willing to guarantee that the system would produce the specific, measurable results that were determined and agreed upon at the beginning of the project?

You get the idea. Someone has to be accountable for the results.

Finally, the fifth reason that projects fail: Lack of a Proven Process.

In our fast food, instant answer society, we tend to skip processes. We want the answers now and we are not always patient.

A month ago I came down with a cold. Everyone knows that you don't always go to the doctor because you have a simple cold. You simply drink plenty of fluids, get some rest, and it will heal itself. Well, this cold did not go away. After about two weeks, I finally went to the doctor. Of course, I would rather have been able to simply call the doctor and have him prescribe cold medication over the phone. Instead, this doctor had a nurse take

[10] Inbound Logistics, July 2000, p 160

my temperature, blood pressure, listen to my chest, ask me several dozen questions, swab my throat, run a culture, and x-ray my sinuses. As it turned out, I had a sinus infection and not a cold. He gave me the proper medication and in a few days I had recovered. I say that the reason his project succeeded was because he had a proven process. He took the time to find out what my problem really was so that he could prescribe the appropriate remedy.

In my business, we use a system called the systems engineering process. I am not claiming that this is the only or the best process in the world, or that you need to implement this particular process for your project to be a success. However, you do need to have in place *some* process in order to maximize results. In an extensive survey conducted by the Gartner Group, META Group and Standish Group, the following was found to be among the principle reasons for successful projects:

> "Institutionalize the processes, metrics and tools. This is an integral part of successful IT project management. It improves sponsorship, minimizes risks and improves the probability of success."[11]

I have always had a preference for systems engineering because it looks at the whole picture. According to Professor Blanchard, the chairman of systems engineering at Virginia Polytechnic Institute, systems engineering is defined as follows:

> "Broadly defined, systems engineering is the effective application of scientific and engineering efforts to transform an operational need into a defined system configuration through top-down iterative process of requirements definition, functional analysis and allocation, synthesis, design optimization, test, and evaluation."[12]

Systems engineering is a process that is evolutionary in nature, considers the whole system, and involves integrating the hardware, software, people, process, and environment to address a specific objective or result.

[11] Bounds, Gene, Principles of Effective IT Project Management, Federal Times, 3/27/2000, Vol 36, p. 16
[12] Blanchard, Benjamin, The Journal of the National Council on Systems Engineering, July/September 1994, p. 123

It consists of the following 7 phases:

1. Requirements and Specifications (What)
2. Preliminary Conceptual Design (How)
3. Design and Architecting (Details of How)
4. Production and Testing (Preparation)
5. Operational Implementation (Delivery and Integration)
6. Evaluation and Modification (Change Orders)
7. Deployment & Maintenance (Production and Support)

Each of these phases contains four different components: activities, deliverables, milestones, and a review. Let's look at each of these steps individually and see what specific actions you could take to insure the success of your project.

The first step is defining the requirements and specifications of the system.

This is the most critical phase of any project. It is also known as the definition or discovery phase of a project. You have to understand the problem in order to solve it. An elegant solution to the wrong problem is useless. At this stage, you must define precisely **what** the system will do in terms of the results that are expected, the cost, and the time required for the project.

Sometimes people try to take shortcuts here and that is a huge mistake. Changing the requirements later in the development of the system can have a significant impact on the cost and its ultimate outcome.

You also must have agreement from all stakeholders on the requirements. A stakeholder is anyone who has the ability to impose requirements on the system. This includes end users and the owner. (Remember my bar code systems experience?)

One method that is successful in getting everyone on the same page is to create a project charter. A charter contains the problem statement, goal statement, constraints, team members, and preliminary plan.

> "Jon Skeels, the National Interagency Resource Ordering and Status System project team leader says a large percentage of IT projects fail because charters aren't constructed to show stakeholders how the projects will be financed, what the deliverables are and when the milestones would be. A charter signed by all the participants is the best way to keep everybody on board."[13]

[13] Computerworld, 7/16/2001, Vol.35 Issue 29, p. 26

So, how do you do it? These are the seven activities for the first phase:

1. Identify and meet with the people that are impacted by the problem that you want to solve. Define the problem and its cause. Gain as much clarity as possible on the specific results that you need to produce.
2. Define the mission and goal of the new system. Create the big picture. Develop a shared vision.
3. Identify the program constraints. These may include budget, operating system, interface into other systems, operational environment, and time frame or schedule.
4. Prepare a business case or return on investment to justify the system.
5. Determine if the project makes sense and is possible to execute. If it does…
6. Put it in writing. Develop a preliminary project plan that clearly identifies the problem, the goal, constraints, major functions necessary for the system to achieve the goal, time frame, and budget. This document is sometimes referred to as the system requirements document or the statement of work.
7. Submit the document to your team and gain agreement. Make any revisions necessary for clarity. Validate that the document is complete and accurate.

Sometimes customers don't know what they want or need. I have found that one of the best ways to discover the requirements is to conduct a site survey and interview the potential users in the warehouse. I have always found that they were able to reduce the risk of project failure by informing me what I don't know that I don't know about a project's requirements.

The deliverable and milestone for the first phase is the signed and approved written requirements document.

The second phase is the preliminary conceptual design of the system.

This is where **how** the system will work is determined in order to accomplish those "WHATS". The purpose of this phase is to define all the functionality that the system will provide and determine what the best means of accomplishing it is. Professor Rechtin of the University of Southern California explains:

> "The first system model to be built is usually a conceptual sketch of the system structure (architecture). It is best constructed through free discussions between client and architect. It should contain all the features that are critical and as few possible of those that appear irrelevant or self-evident, at least in the beginning. It should show how its elements are aggregated (clustered) and partitioned (interfaced). The client should be able to study it to make value judgments, the architect to make technical choices. It should provide visual cues to analysts. It

should at least help preclude major misconceptions or future disasters. More than one first model and more than one perspective of each is certainly desirable."[14]

The activities involved in this phase could include the following:

1. White board discussion with team members on different design approaches.
2. Block diagrams or functional flow diagrams of the components, subassemblies, interfaces, inputs and outputs.
3. Early prototypes of the system.
4. Pilots.
5. Tests and evaluations of design alternatives.
6. Trade-off studies and selection of design.

The deliverable is an agreed upon design of how the system will work. At the end of this phase, four questions should be asked:

- Will this design fulfill all of the requirements?
- Can it be done within the constraints of the project, such as budget and schedule?
- Have we identified and resolved all of the risks?
- Is it supportable?

The third phase of the systems engineering process is the detailed design and architecting of the system.

This is where the details of how of the system will work are specified. The deliverable and milestone for this phase is the signed and approved written functional specification document. The functional specification should fully describe the system architecture, all of the components of the system and how they interface. This includes all the hardware, peripherals and software. All deliverables should be specified. This is the blueprint of how the system will be created. A well-written functional specification would allow a builder of the system to produce it without having to ask questions of the designers.

The detailed design should include the following:

- Overview—a reiteration of the project charter, the problems to be solved and the goal of the system
- Environmental description—a specification of the environment in which the system must operate.
- Design diagrams—showing the subsystems and components.

[14] Rechtin, E, The foundations of systems architecting, The Journal of NCOSE, Volume 1, Number 1, July-September 1994, p. 39.

- List of standards to be used; this should describe how the inputs, outputs, and error conditions should be passed between the components along with any standards to be followed in production.
- A definition of each component and their inputs, outputs, and error conditions.
- An acceptance test plan—a plan that will test all the functionality of the system. The acceptance test plan allows the client to verify and accept the system. Each test in the acceptance test plan should contain the purpose of the test, the function tested, and the input and output that is expected.
- An implementation plan—a plan that details the roles and responsibilities of each of the members on the implementation team, the time frame or schedule for completed activities, and the flow or order that the activities will take place.

The fourth phase of the systems engineering process is the production and testing of the system.

The purpose of this phase is to build the system. The activities in the production phase include acquiring, building, constructing, coding and assembling the components of the system (hardware, peripherals, and software). Each component should be unit tested and tested again as a part of the system. Quality should be controlled to ensure that all components meet specifications. The entire system should be tested to assure completeness and that functional and performance measures are satisfied. The only way to tell whether a component, subsystem, or the whole system can accomplish its intended functions and has the correct performance features is to test it. During this phase, any deficiencies found in components or processes need to be corrected or resolved. The acceptance test plan should be updated to reflect any changes to the design or functionality.

The deliverable is a built system.

The fifth phase of the systems engineering process is the operational implementation of the system.

The purpose of this phase is to deliver and implement the system in the environment in which it is to be operated. Sometimes, not all aspects of a system can be tested in the laboratory environment so this phase is a critical test of how it will really work in the environment in which it will work.

I spoke with a vice president of a large food distributor that recently implemented a bar code scanning system in a freezer environment. While the specification of the bar code scanner/terminal said that it would work in a 20 degree below zero environments, it didn't mention that the display would fog up.

This is also the phase when the acceptance test plan is executed. It is here when all the functionality that was agreed to in the third phase is demonstrated to the customer or the

review team. It is important that this review takes place and the system is accepted before it goes into production. The last thing in the world that you want to do is to rush a system into full-scale production and then have to pull it out because of something that was not completely tested and accepted at this phase. When that happens, all confidence and trust in the system is lost, and regaining momentum can be a monumental task.

The deliverable is a written sign off on the acceptance test plan.

The sixth phase of the systems engineering process is the evaluation and modification of the system.

In my experience of implementing systems, no matter the planning and testing conducted up-front, nothing ever goes absolutely perfect. There is always something that someone wants tweaked or modified after seeing the whole system operate. Generally, it is one of those details that no one has thought of until that very moment. They say in mountain climbing that you can't see what is on the other side of the mountain until you get to the top of the one you are climbing. Now, we are on the top of the mountain. We are just about ready to go into production and have tested every function when someone asks if we can make a change. At that point, a decision has to be made as to how critical the change is in relation to the time it will take to change and test it and how large or small it is. We always send an engineer to the site to implement a system for this very reason. If it is something that is minor and can be changed on the spot, then we do it. If it is more involved, we make a note of it for later consideration, and move on to the next phase.

This is also the place to write down or diagram the process for the operation of the system. This is an invaluable tool for the training process.

The deliverable is the final approval of the system and the documented process.

The seventh and final phase of the systems engineering process is the deployment and maintenance of the system.

This is when we train the users on how the system operates and put the system into production. I have found that the best way to train a user is to let them operate the system. You don't learn to dance by reading a book about it or sitting in a chair watching. You learn by doing and having a teacher coach you on what you are doing. It is very important to have the users run the system and to have the instructor there for a few hours to a few days depending on the complexity of the system to coach the users in every aspect of its operation. If there are many users, then it generally works best to train a trainer and then coach the trainer in training the users. I firmly believe that most support issues come from not investing enough time in the training process. This is not an area to shortcut if a system is to succeed.

Conclusion

In summary, the secret to succeeding in any project is to use a proven process that will assure that you get the results that you desire and deserve. This paper has provided you with the five biggest reasons that projects fail and given you steps and actions that you can take to prevent yours from failing. The very fact that you took the time to read this paper tells me that you are the type of professional that is more likely to succeed; you are open to new ideas, perspectives and methods of operation.

The last piece of advice that I can give you is that after you have successfully implemented your project--celebrate! You have succeeded where most have failed. Congratulations!

About The Author

Mark A. Taylor, MBA, is the nation's leading authority on utilizing technology to increase productivity in the warehouse, with 25 years experience consulting more than 10,000 organizations. He is the president and CEO of TAYLOR Systems Engineering Corporation in Plymouth Michigan and the author of the only two books ever written on the subject of computerized shipping systems, his latest is the *New Millennium Edition of Computerized Shipping Systems: Increasing Profit and Productivity Through Technology.* Mr. Taylor is a regular speaker at major industry events including the Parcel Shipping and Distribution Expo, the National Conference on Operations & Fulfillment, the National Postal Forum, Council of Logistics Management and the Parcel Logistics Expo. He is also a frequent contributor to numerous industry publications, including Operations and Fulfillment, MAIL, and Parcel Shipping and Distribution magazines. He is a member of the International Council on Systems Engineering, the Council of Logistics Management and The Executive Committee (TEC).

He can be reached at (734) 420-7447 or MTaylor@TAYLORSystemsEngineering.com.

For more information contact

TAYLOR
SYSTEMS ENGINEERING
CORPORATION
PROFIT THROUGH PROVEN TECHNOLOGY

(734) 420-7447

www.TAYLORSsystemsengineering.com