CLASSIC
SYSTEM SOLUTIONS, INC.

8 Stupid Things Teams Do To Mess Up Their Software Projects

By James Hobart
President & CEO

In 2002, the United States wasted over $55 billion on failed and poorly run software projects according to the Standish Group. Ten years ago, they reported similar numbers with 53% of projects overrunning their estimates and only 61% of features being implemented.

Why have things changed so little over the past decade despite the great advances of the web, technology innovations and a brutal economic downturn that should have forced efficiencies as a matter of survival?

How about your projects? Are they on track for success or doomed to fall into the chasm of late and cancelled projects? Here are 8 pitfalls to avoid and solutions to try on your next project.

Stupid Thing #1) Use the wrong process.
The process people will tell you this is the key root of the problem. What do you expect? They're process people. We have found that processes do not make great software. Great software is created by a highly focused group of people with a clear vision, the right skills, and an environment that allows them to accomplish their goals. Agile, RUP, Six Sigma, and other processes all add rigor to a naturally chaotic and creative process. We use pieces of each of these methods on all of our projects. The problem with systematically following a rigid process is that people try to substitute the 'processes' for 'thinking'. Completing the tasks and following a process will not in itself lead to a successful completion of a project

Solution – There is no perfect process.
Take the best ideas from several mainstream methodologies and implement a pragmatic set of techniques using an iterative model. Tailor your process to each project based on size and complexity. The biggest problems with project delays arise out of 'Missed steps'. These can result in fatal blows later in a project. A big 'Missed step' is the user study. Did you really study how the users work and understand their needs? User studies can truly unite business and IT team members on the key business issues they are trying to solve. Measure your results of usability testing using Six Sigma techniques and create use cases in a standard format (RUP). Finally use an iterative approach emphasizing whiteboards (Agile) and prototyping to zero in on the true requirements.

Stupid Thing #2) Sweat the small stuff
How many times have you been in a design meeting where the argument has degraded down to hour long discussions of where the button goes or whether it should be labeled 'Save' or 'Submit'? Other common sidebar discussions concerning color or use of a specific font or disabling the 'Back button' can easily put a halt to a productive design session. All these are important issues, however, when each team re-examines UI standards for each project it becomes an incredible waste of time and defocuses the team from the important business issues they set out to solve.

Solution: *Implement design standards*.
Implement design standards with sound user interface design principles and evolve these into visual design patterns as you test and validate what works with your users. Make them available across the organization via a web-based repository and keep them updated and flexible to meet the needs of different users and technologies.

Stupid Thing #3) Solve the wrong business issues.
True business issues are often not addressed even if the project meets deadlines for implementation. This results in additional product releases before the users can functionally use the system. Stakeholders are sensitive to this and now factor these subsequent releases into the project cost often resulting in little or no ROI (Return on Investment) associated with the project, thus putting future project funding for other IT initiatives at risk.

Solution - *Identify the true business issues*.
Hang out with users, talk to customers, and dig up the root cause of issues they face. Ask "how and why" questions and implement these solutions into your use cases. When writing use cases, try a two column use case format vs. typical one column format where you can document intent of the user on the left and system response on the right. Focus your initial efforts on the high impact use cases and primary success scenarios. Don't get caught up in all the exception cases until you have really nailed down the key tasks performed 80% of the time.

Stupid Thing #4) Have the wrong team mix.
Many teams have missing players or the roles assigned are not understood. Great products need balanced input from at least three perspectives: (IT, Product Mgmt, and Stakeholders). Often a 4th perspective is required: Usability. Usability runs the middle ground between product management and IT to ensure the stakeholders can use the delivered solution. Without the right mix, a natural imbalance will occur and a technology-centered solution will often emerge along with high training and support costs. Even if balance exists, the short-term nature of today's products often results in a lack of vision. If a visionary is brought into the mix, the team immediately pushes back in fear of putting their pre-determined project deadlines in jeopardy. Unfortunately, the team may meet their project release date but miss the mark on solving the true business issues or capitalizing on a new business opportunity.

Solution*: **Build the right team*** or acquire skills to fill the gaps.
A 'usability person' often has very different skills than an interaction designer. Usability people find defects and interaction designers create solutions. Consider your team lucky if you have one person who can fill both roles. Business analysts often fill the role of a product manager but may not fully understand what drives revenue. A product manager may have the vision but may lack the skills or political influence to bridge the gap between stakeholders and IT. Bring in a person with vision to help really solve the key issues and the political clout to bring about change. The visionary sees where the end-game needs to be played whereas the IT group often needs to stay focused on the next release in order to keep their sanity and meet deadlines. Create a project roadmap so all parties see where the product is going and how it will get there via the iterative process. Avoid myopic thinking.
Jeff Hawkins, the inventor of the original Palm demonstrated his vision recently with the recently released Treo 600. After he met with users in New York he had a clear vision on

user needs and had the team stop mid-stream on PDA design and refocus their efforts on a combined Palm, Phone and Blackberry. After much resistance, the team got clear on the vision and now has a market leading product.

Stupid Thing #5) Bog your team down in meetings.
We still have too many meetings. Combine poor team communication with offshore outsourcing and some scaring things are ending up on the user's desktops. As teams become more geographically disbursed, the need to clearly articulate specific design details is more important. One estimate we've seen is that the cost to fix a design flaw is 200x's higher in UAT (User acceptance testing) than during the initial design phase.

Solution: *Enable rapid sharing of team knowledge.*
One way to keep people focused on deliverables rather than meetings is to use a team collaborative web-based portal. It should allow easy access to design documents, prototypes and models and allow teams to easily review deliverables and post current issues. Project status, due dates, etc should be easily available and not hidden in the dark corner of a manager's local drive in an inaccessible MS project file doled out during the weekly 'team status' inquisition. If you are using these collaborative tools effectively, there will be less team 'meetings' and more actual work getting completed.

Stupid Thing #6) Create the wrong deliverables.
Most people are visual learners. Creating 400 pages of text is a sure way to ensure no-one truly understands what is needed and what will be built. Taking the "Sign here, here and here…" approach with numerous user signoffs is not the answer. Not that we don't like signoffs, they are a necessary part of every development process. Unfortunately this can become the 'Prime directive' rather than ensuring the users really understand what they are getting.

Solution: *Create useful deliverables*.
Use a visual model driven approach. Start off building 3 key visual models; The presentation model which identifies key objects, interactions, the 'look and feel' and various major assumptions about how the application will be delivered to the user.
The navigation model visually displays how users will navigate key tasks as defined by the primary use cases. Finally, key screen layouts will storyboard the interaction and provide the visual anchors for users to actually experience how those key tasks will be performed. Ideally these models will easily transform into a working prototype which brings all the design and requirement sessions to life in a set of real world congruent models. Once an agreement has been made at this level, finalize the use cases and develop the detailed specifications as supported by the visual models.

Stupid Thing #7) Use the wrong tools.
Tool vendors will always be standing by to sell you another tool to fix the problems with your project. Tools can help you succeed as long as they provide real value and the people using them are properly trained in their usage. Just like process, tools don't create great software, people do. Trouble begins when tools are forced onto teams who have neither the time nor the management support to properly use them. Over the years we have seen millions of dollars wasted on development and testing tools that barely made it beyond their initial Installshield glimpse of life. Worse yet, we are still seeing technology decisions that are not appropriate to solve the true business problems facing the project. For instance, if your business problem is to provide real-time collaboration and instant response time for

loosely connected clients, it may be time to revisit that corporate mandate stating that all new applications be developed using HTML.

Solution: ***Business drives technology choices.***
Technology decisions should be driven by a key understanding of the business issues you are trying to solve and how user performance can be optimized. While web technology like HTML is appropriate for a wide range of needs, enterprise internet applications may require a richer set of interactions simply not attainable with HTML. As you develop expertise in a range of technologies, you will be able to design and build reusable visual components on top of existing frameworks like .Net, J2EE or Struts and apply them where needed based on the business problems uncovered during design.

Stupid Thing #8) Have Bad timing
Completing the right task at the wrong time can be a recipe for disaster. When to do a usability test, when to begin coding, etc. is often determined by past projects regardless if they were successful. It seems many project plans were initiated as a result of 'Save As…' from the file menu rather than careful consideration of what worked well on past projects.

Solution: ***Improve your timing***
Rearrange your key project milestones and implement an iterative measurable process. Don't start coding before you understand what you're trying to solve. Don't put usability testing in with QA testing at the end of the project as this will just annoy the developers when they are totally focused on getting the product across the finish line. Allow user studies, iterative design and whiteboarding to occur in earnest during the requirements gathering phases and embrace user feedback as the visual models evolve and are presented. Continue gathering user feedback and get agreement among all parties (IT, stakeholders and business units) before moving to the next level of iteration in your design.

Summing it up…

Many of the solutions mentioned here are things you 'know how to do'. But do you consistently '***Do what you know***'? The numbers speak for themselves. Despite great advances in technology, basic principles still apply: Understanding users, implementing a self-correcting iterative process and being smart in how we create deliverables that can be understood by all team members will go a long way toward improving the project success numbers over the next ten years.

Our focus is helping teams succeed in creating great software. We have training, GUI standards and a proven process all based on real-world experience.

About Classic System Solutions

Classic System Solutions has been designing Enterprise Internet Applications and creating visual design patterns with Global 2000 clients for over a decade and we offers practical enterprise design solutions in our consulting practice, training, and design guidelines. Give us a call or e-mail info@classicsys.com, and we'll provide you with a detailed set of critical success factors for building enterprise internet applications along with a free usability kit.