

# Preface

**M**ost organizations have suffered through their “Titanic Project,” a project that results in a catastrophic failure. Most Project Managers will admit to living through at least one of these in their career, and from my own personal experience I have seen my share of these during my career as a consultant. There are different types of project failures, however:

1. Some projects are obvious failures, particularly those cancelled or aborted during the project. The Standish Group brought this to light in their landmark “Chaos” report. (Research shows a staggering 31.1% of projects will be canceled before they ever get completed. Further results indicate 52.7% of projects will cost 189% of their original estimates).<sup>1</sup>

---

<sup>1</sup> Source: <http://www.standishgroup.com/index.php>

## *Titanic Lessons for IT Projects*

2. More critical failures are those projects that fail on or around implementation. The solution is already built and has undergone testing, but fails while being deployed.
3. The most critical failures are those projects that fail weeks or months into production, after the project has been deemed as completed. These are unpredictable, unexpected and by far the most costly.

In my encounters with this third type of failed projects, typically the project team has been disbanded and allocated to new projects long before the problems emerge. In addition, the operations group has been grappling with an unknown solution as they had been brought in much too late into the project to appreciate and understand the solution well enough to manage it adequately.

One of the challenges in investigating these types of failures is convincing senior level executives that the root cause of the failures is poor decision making in the project itself. Very often decision making was hampered by poor investments in the project and a focus on functional (rather than non-functional) requirements. (That is, *what* the solution does rather than *how* it does it.)

This also raises the question of when should a project end, and when can it be deemed a success or failure. As I began to understand these problems more deeply, I started to draw parallels from some of the notable disasters of the 20<sup>th</sup> century: *Titanic*, *Hindenburg*, Three Mile Island, Chernobyl, and space shuttles *Challenger* and *Columbia*.

*Titanic* kept coming to the forefront of my thinking as the volume of research material on the disaster is phenomenal. With *Titanic*, the 4 days of operations were chaotic and littered with mistakes: the breakdown in early warning systems, the failure to manage incoming intelligence, and succumbing to business pressures to better *Olympic*'s best Atlantic crossing time. Once the collision with ice occurred, the compromises in the project's design and poor decision making doomed *Titanic*. Most importantly, all these operational problems were rooted in the shortcomings of the ship's construction project.

My initial book, *On-line, On-time, On-budget: Titanic Lessons for the e-business Executive*, was created to help readers highlight how IT problems in operation or production correlate back to the initiating IT project. As a result, this culminates in a better understanding of risks and decision making, and assists in the better running of IT projects. Since the book's publication, I have had many conversations with readers, mostly Project Managers. The overwhelming feedback I have on the book indicated the need for a more high-level, shorter, less technical version that captures the essence of the *Titanic* story, which was proving to be the central feature. To respond to this growing demand for a simpler book, I have prepared the volume you are now reading

Mark Kozak-Holland.

## *Titanic Lessons for IT Projects*

# **The Story Begins**

**M**ost people are very familiar with the *Titanic* story from James Cameron's 1997 movie, or from documentaries on television. Typically, these focus on the last two days of the voyage and the last hours of the disaster itself. But what about the four-year construction project? Was it significant? What impact did it have on the disaster? What can we learn from that project that we can transfer to today's information technology projects?

Let's go back to 1909 and the business situation that faced White Star, the company that owned the fateful ship. (See Figure 1.1.) Their aging fleet of liners was grossly inadequate to compete with stiff growing competition. White Star embarked on a strategy to invest in new emerging technologies and create three super liners (two of which are

## *Titanic Lessons for IT Projects*



*Figure 1.1: The infamous White Star Line logo.*

shown in Figure 1.2). These were major investments as the liners were likely to be in service for at least 20 years. So for the designers it was critical to get the design right. They proceeded with a design strategy of luxury over speed where the ship's second class was equivalent to first class on other ships, and third class to second class.

To match the luxurious splendor (or the *functional requirements*) investments also had to also be made into *non-functional requirements* —everything that supports the functionals including performance, safety and capacity.

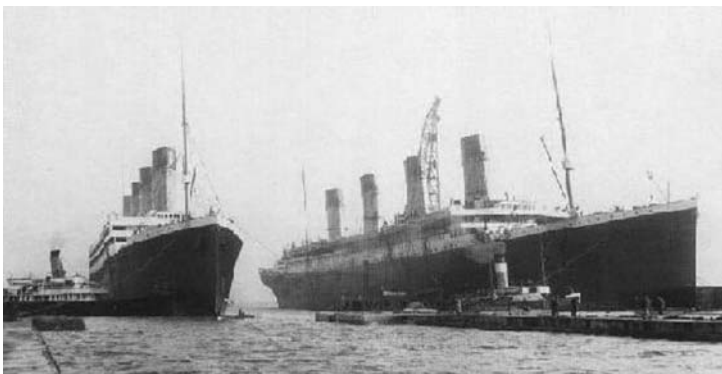
From the outset, no expenses were spared when investing in the latest emerging technologies for the non-functionals like the safety systems, which included a double skin hull (the bottom space divided into 73 watertight compartments), 15 bulkheads and electric doors, 48 lifeboats, and advanced water pump technology. However, as in many projects, a struggle took place within the project team where the success of the business strategy overrode other considerations.

One by one, compromises were subtly made in the non-functional requirements as the focus stayed on the functional requirements. Non-functional requirements were

less visible, so this “corner cutting” went unnoticed. For example, the functional requirements for a spacious ball room resulted in four of the bulkheads not extending to the top deck, severely compromising the ability to self contain flooding. It was not only the business executives (principally Director Bruce Ismay) who were responsible for this but also the technical people—both the White Star architects and the Harland-Wolff ship builders.

By the end of the project construction stage, most of these safety features had been compromised. The height of the bulkheads was only 10 feet above the waterline in some places. The logical explanation for these compromises is that assumptions were made by the White Star architects that the “aggregated” safety features remaining would protect *Titanic* from whatever nature handed out.

By the end of project, the team believed that the safety levels were still maintained at the initial levels. So this set a high level of confidence for the maiden voyage (or



*Figure 1.2: Sister ships Olympic and Titanic in port.*

## *Titanic Lessons for IT Projects*

*production*). The arrogant view evolved that *Titanic* was a huge lifeboat. The *Titanic* project team made the mistake of believing the initial design assumptions, and not testing these far enough. Such was the confidence in the safety of the ship that by the end of the project, disaster recovery and business continuity plans were considered superfluous.

In short, the people “who should have got it”—the architects—allowed the compromises to pass. As the ship went into operation, a perception emerged that even if things did go wrong operationally the ship had enough safety features to protect it. This instilled a mindset in the crew and passengers that the ship was unsinkable. Why else were 53 millionaires aboard?

The ship set forth after a grossly inadequate “testing” phase, and enormous operational risks were taken. The ship’s speed was steadily increased as it approached the ice field. Compromises were then made operationally as an ice detection test was fudged (see Chapter 7 for details), radio ice warnings were not passed to the bridge in a timely fashion, and a minimum number of lookouts were posted without binoculars. The ship’s officers failed to piece together the extent of the ice field and understand the true danger as the feedback systems went awry.

Bring this story forward to today and there are many comparatives that can be made to modern IT projects, from construction right through to production. For example, there are many similarities with how IT project problems and issues surface days, months or even years after the project is completed and in production.

IT projects may be successful on deployment and pass a broad number of “standard” tests (system, performance and acceptance tests) yet still fail catastrophically when in operation. After all, only 25 percent of all IT projects are successful, a figure that has been continuously verified in various surveys.<sup>1</sup>

The success of IT projects should not be measured at deployment, but rather after the solution has been in production for a while and carefully measured. Metrics should be closely tied to the overall impact to the business. The *Titanic* story helps us better understand the relationship between functional and non-functional requirements, the interplay of compromises in the project and why things go horribly wrong in operation.

---

<sup>1</sup> Only 26 percent of all IT projects finish on-time, on-budget and with all the features and functions originally specified according to “Chaos, a recipe for success,” Standish Group, 1994, 1996, 1998.