BEIGEL
TECHNOLOGY
CORPORATION

PHONE: 760-633-3868
FAX: 760-633-3819
E-MAIL: mike@beigeltech.com

www.beigeltech.com
308 VIA JULITA, ENCINITAS CA 92024

The following article, Exception Management for RFID Systems by Mike Beigel, Samuel Marcus and Grant Ballard, was published in the most recent IDTechEX Web Journal, Issue 36, January 2004, Smart Labels Analyst.

IT IS NOT FOR DISTRIBUTION

Issue 36 • January 2004

The IDTechEx Web Journal

Smart Labels Analyst

Website: www.idtechex.com Email: info@idtechex.com

Exception Management for RFID Systems

Report by Mike Beigel, Samuel Marcus and Grant Ballard, Beigel Technology Corp

"There's an exception to every rule," the saying goes. Unless your RFID system can handle every exception you can't build a fully automated system. So choose your rules carefully and design exception management techniques into your system.

We assume that your new RFID project is useful and viable. The purpose of this article is to help you make it more robust through attention to exception management. We use as examples some of our experiences in designing, constructing, deploying, debugging and optimizing a RFID "penguin weighbridge" project in Antarctica starting in 1994.

Most marketing scenarios today would paint a picture of RFID as a perfect tool for automation, be the application inventory tracking, drive-through merchandise checkout, or supply chain management. A general sense of optimism pervades the field. However, this romanticized view of the capabilities of RFID distracts from some hard realities encountered in building automated RFID systems that really work. Full automation can only be achieved when the system design and implementation can handle every possible situation.

Many exceptions can be avoided through the patient application of common sense. Taking time during the design phase to anticipate what can go wrong and take it into account will result in a more robust system development and implementation phase. A RFID system designed to function under less than optimal conditions will outperform a system designed to take the idealized case for granted. Principles like these are easy to advocate in the abstract, but actually applying them is another matter.

System Function Levels

In this article we examine the design, creation and operation of RFID systems at differing levels corresponding to project design and implementation. The exceptions we examine will be categorized according to which levels

Continued >>

Smart Labels Analyst · Issue 36 · January 2004

Exception Management for RFID Systems (Cont)

they affect.

- Project Level: The explicit goals, purposes, and functions of the system.
- Model Level: The manner in which the above purposes of the system will be fulfilled and the ways in which the system will be used. The "world view" or paradigm of the system.
- System Level: The system is the data infrastructure of the project- all the components and subsystems and communications channels.
- Hardware Level: A subset of the System level, this deals with all the physics and mechanics of running the RFID aspect of the system. It includes RFID tags, readers, computers, fixtures, etc.
- User Level: The range of possible interactions between active users (e.g., technicians, customers) and passive users (e.g., shipping pallets).

Two penguins enter RFID weighbridge, causing multiple exceptions to RFID, direction tracking and weighing rules.
Source: Beigel Technology Corp

The RFID system and implementation are proposed, developed, and deployed according to the assumptions based on the categories above, and then the "fun" begins.

Exceptions and Failure Modes

Project Level Exceptions:
Exceptions occur on the Project Level when performance expectations are unrealistic or poorly defined, or when communication between development teams and management are inadequate. Errors in planning, the early design phase, and in general problems in the process of actually designing and creating the system are all Project Level exceptions. To avoid these problems it is important to use an open and cautious approach to system design and to make sure that everyone involved in the project is aware of the requirements and proposed design.

Model Level Exceptions:
Exceptions on the Model Level are errors in the understanding of how the system is going to work. Since the model serves to define the reaction of the system to any given situation, a faulty model will lead to a system that behaves incorrectly or erratically. The only way to avoid Model exceptions is to thoroughly analyze and test the model and prototype systems before full implementation.

Continued >>

Exception Management for RFID Systems (Cont)

System Level Exceptions:
Exceptions on the System Level are hardware failures and design flaws. They can also occur when communications between components fails or there is miscommunication. Power outages, computer crashes, and any other sort of disruption in the process are all common System Level exceptions. System exceptions can be partly eliminated during the design phase with redundant and rugged engineering, and can be avoided somewhat during run-time by educating users on the correct use of the system.

RFID Level Exceptions:
The RFID subsystem is susceptible to all the exceptions that can befall the system in general, as well as a another set of limitations and problems peculiar to the field of RFID. RFID specific problems tend to befall one of the either the tag or the tag reader, though some problems like general interference or miscommunication affect both. The tags themselves may be lost or misapplied, or their hardware components may fail, ceasing function or corrupting data. The readers may be unable to read at some ranges, or be unable to process more than a certain number of tags at once.

User Exceptions:
There are several types of user and each can cause exceptions, intentionally or not, by simply acting outside the bounds of user behaviour predicted by the system designers. Operators and technician administrating the system and can cause exceptions if they improperly handle the system. Other users can be active (a customer) or passive (a shipping pallet), helpful or hostile (as when a shoplifter tries to circumvent store security), but any user can cause exceptions which unpredictable action. These can be abated by taking as wide a range of potential user behaviour as possible into account when designing the system.

In addition to the mundane sorts of issues that can come up during the operation of a system, sometimes extraordinary adverse events occur. For some systems it will be important to be able to operate despite true disasters: earthquakes, power outages, and even war. Disasters can disrupt both corporate and global infrastructure, isolating key subsystems from the whole. If a system is vital then it must be equipped to handle such exotic exceptions as man-made and natural disasters. Exceptional circumstances need not be fully fledged disasters; they can also take the form of extremely harsh weather, excessive background RF radiation, and so forth.

In short, an exception is what happens when events take circumstance outside the realm of the expected system operation. By learning to anticipate and expect likely problems, the number of exceptions can be reduced. Actually fixing exceptions, on the other hand, requires analyzing the problem, creating a solution and then implementing it. It is in general less expensive to anticipate and eliminate an exception than to be forced to and handle it after production and during operation. Of course, it's impossible to predict which exceptions will occur in order to catch them all, so exception handling in the implementation and operation phase is just as important as exception prediction and reduction in the design phase.

Simple Examples Taught by Penguins

In 1994 we were retained to design and fabricate an automatic RFID based system to track and weigh penguins as they moved back and forth between their nests on the beaches of Antarctica and their aquatic feeding territories. By tracking the weight of tagged penguins on the way to and from their nests it would be possible to determine how much food they were taking back to their young, and to determine how long it took them to find this food. The principles which guided the project are illustrative for any sort of tracking system- whether it tracks shipping pallets, customers, library books, or any set of objects.

The plan was to find points the penguins had to travel through on their journeys and to place weighbridges—corridors with an assortment of built-in sensors and RFID tag readers—at these locations to monitor their movements. A penguin entering or leaving the corridor would trigger an optical sensor, get weighed as it walked on an electric scale, have its RFID tag (which had been implanted previously)

Continued >>

Exception Management for RFID Systems (Cont)

read, and then trip a second optical sensor (to confirm the direction they were heading) on its way out of the weighbridge. The weighbridge would process the data and then store it until downloaded by the scientists. It was a simple plan and should have gone off without a hitch. This meant, of course, that we hit every stumbling block, conceivable or not, in Antarctica.

The Project Level

Our project goal was to build corridors with and RFID tag reader, optic and weight sensors, and a microcomputer. Because we chose to use proprietary RFID tags we needed to build special readers for the weighbridge. The project was a bit rushed, so the components were amassed, tested, and flown to Antarctica to be assembled on-site. Since things were running late there was only time to test components individually and it came as some surprise when it turned out that the RFID reader field interfered with the optic sensors, a problem compounded by the impossibility of obtaining new parts in a remote site in Antarctica.

Since the system software was dependent on the concurrent operation of the RFID reader and the optic sensors, the only solution was an unforgiving debugging and software re-design session on-site in a tent pitched on thawing penguin guano.

The design choice of a corridor was less than perfect to begin with—a simple hoop would have worked better in terms of RFID reading, but it was initially assumed that penguins would be too wary to walk through a hoop. After four years of improving the system, it was discovered by practical observation that penguins were more likely to go through a hoop than a corridor anyway, and the change in structure significantly increased the read rate for the system. The component interfer-

Continued >>

Weighbridge test at Sea World: captive penguins don't act like wild penguins
Source: Beigel Technology Corp

Continued >>

Exception Management for RFID Systems (Cont)

ence problem could have been easily avoided with thorough testing but it was a rushed job and in the spirit of getting it there on time it couldn't be done. The second problem was a fundamental and probably unavoidable misapprehension by scientists, engineers and business people about the "best mode" of system design.

The Model Level

The operation of the weighbridge seemed fairly straightforward. A penguin would approach and enter the device from one side and walk straight through and out the other. This would in turn trigger an optical sensor, the RFID reader and scale, and the other optical sensor, in order. Of course, the penguins didn't have to do that, but we saw no reasons for them not to. The reality of the situation couldn't have been farther from our assumptions of course and so our system, which was basically a concrete implementation of how we had built our model, needed extensive reworking on site. As is typical of the model level, the problems that these misunderstandings caused actually occurred on other levels, confounding the RFID system and requiring hardware changes.

The Systems Level

Because of the remote locations of the penguin colonies and the 24 hours of sunlight during the study period, we opted to use solar panels to power the weighbridges. Using solar power, convenient and politically correct as it was, introduced power efficiency concerns not present in most "on-grid" application. We switched to a slower, low power microprocessor to analyze the incoming data, and we turned the reader off when the optical sensors indicated a period of inactivity. A chain reaction of complications ensued.

Because of the lower powered microprocessors, storage capacity and processor speed were compromised. The original design allowed ID, direction, weight and time measurements to be made in real time. However, now the data analysis could not be performed in runtime without causing backlogs in the system, so the raw data had to be archived and then computed and analyzed after the fact. This meant storage issues as the solid state NVRAMs were asked to hold more than they had originally been intended to, and required a higher frequency of data downloading by the scientists to prevent them from filling up, sometimes more than once a day.

Additionally, the weight calculating algorithms turned out to be far more complicated than anyone had predicted, which mean correspondingly complicated software, which became increasingly difficult to mange on the weaker processors. Ultimately, all data needed to be offloaded to full-fledged personal computers and processed after the fact.

Sometimes penguins would simply squat in front of the optical sensors and end up being read three or four times, or they would trip one, turn around and wander off. The weighbridge could only properly determine the direction of the penguin if it hit each of the optical sensors, and this caused some problems. The issue was basically fixed in software, where the weighbridges were instructed to ignore events outside a specific set of complete event profiles. If need was present, humans could go back and look at the records with some hope of reconstructing the anomalous events, but there were more than enough valid readings to make up for the loss of this minority (helped by the fact that in this scenario one valid read was fundamentally as good as any other—in a different setting this fix would not have worked).

The RFID Level

What happens when two penguins cross the bridge at the same time?

Continued >>

Exception Management for RFID Systems (Cont)

What if a penguin's tag isn't read properly or just doesn't work? This second problem is less an issue than the first, since it was impossible to tag the entire population anyway- the majority of the penguins had no tags at all. A penguin with a broken tag just shifted to this category, ultimately to be replaced with the tagging of a new penguin. Other problems like tag misreads and the "two headed penguin" when the weighbridge was forced to try and read two tags at once, were more complicated. Since the RFID tag reader had been reprogrammed to interrogate the tags in rapid pulses it could usually determine which two penguins were crossing (if both had tags) but it was difficult to determine their weights individually. To a certain extent this could be resolved by manually going back through the data, but it was easier, as in the case of the squatting penguins, to ignore nonstandard event profiles. Again, it was the nature of the project that allowed this solution. It would be quite unwise to implement a similar workaround in any system where users could be expected to be antagonistic, or where even a single event-loss would not be acceptable.

Corridor weighbridge works, but RFID performance is marginal
Source: Beigel Technology Corp

The User Level

By far the aspect of the project for which we were least prepared was the penguins themselves. The engineers knew very little penguin psychology, we had no idea what to expect from them. The scientists knew penguin behaviour very well but not in the context of the new weighbridge setup. On some level somewhere we took it as granted that penguins would queue up to walk through the weighbridge in a calm and orderly fashion. This was completely unfounded. A penguin would walk halfway through the bridge, change its mind and walk back the way it came, yielding half complete data. Some penguins would cross so quickly that only partial reads could

Exception Management for RFID Systems (Cont)

be taken. Others would cross so slowly that other penguins would start and finish crossing in the meantime. We didn't know in advance that the floor of the weighbridge was hard for penguins to walk on and had to be covered with a mat.

As we observed the behaviour we began system design changes, from complex electronic or software modifications to physical changes to the weighbridge setup and apparatus. In order to control the speed at which the penguins made the crossing, we placed bamboo sticks low across the entrances to the bridge. Like speed bumps, the sticks forced the penguins to slow down to hop over, and this resulted in more accurate weighing.

"The problem with this utopian vision of RFID is that it's just not that simple"

Special Circumstances

Antarctica is an inhospitable environment. Intense cold and high winds dominate the area. The cables and connectors we used at first simply weren't robust enough to withstand the temperature extremes and corrosive climate at the nest areas. The light sensors also had to cope with the extreme seasonal variations in lighting conditions that could make it difficult to determine just when the sensor was actually detecting a penguin passing in front of it and when the result was an artefact of the lighting. And of course, on-site maintenance in Antarctica isn't the simplest matter. Replacement parts can't be easily found nearby and there are only certain times in which untrained personnel should visit the continent.

We were lucky in that the penguin project continued over a long enough time to find and iron out many of the technical wrinkles. Every year between the nesting seasons we were able to evaluate our results. We could modify our hardware and software, and then work in the field during nesting season to observe the results of our efforts, make adjustments on-location, and once again plan for improvement.

The first deployment was one weighbridge in December of 1994, and the completion of three optimized deployed systems was accomplished by 1998. The system is still in use by the scientists in Antarctica.

Large Scale Implications of RFID System Implementation

A RFID system offers several key advantages over a barcode system. Unlike a barcode, a RFID reader does not need line-of-sight with its tags. Each tag can potentially hold more information than a barcode, and this information need not be read-only; the flow of data may be in more than one direction. A RFID reader can also communicate with more than one tag at a time or be used to locate an object in space. Like the barcode revolution, RFID has potential to increase efficiency (of tracking inventory, purchases, location, etc) by eliminating the need for a human to deal with every unit being tracked. Rather, they just wave a RFID reader at a pile of goods (or in a more idealized model, the goods are read automatically as they pass through a door, for instance) and the miracle of RFID handles the rest. It seems that the future is indeed only a few years away.

The problem with this utopian vision of RFID is that it's just not that simple. To replace barcodes with RFID tags actually means adding to potential problems, not eliminating them. This is because everything that can go wrong with a barcode can go wrong with a RFID tag, and the RFID system itself will have additional idiosyncratic problems to deal with. True- RFID does not require line-of-sight. There is a popular misconception, however, that a RFID tag within a reader's range will automatically be read. Rather this is a probabilistic event, dependent on the conditions and a variety of scenario parameters (in much the same way that a barcode may not be read properly even if it is in line-of-sight). The ability to read more than one tag at a time also increases the time it takes to read each individual tag significantly.

Like a barcode, a RFID tag may be incorrect or misread- but it's not being read by a human who knows that the can of tomatoes they just scanned isn't

Smart Labels Analyst · Issue 36 · January 2004

Exception Management for RFID Systems (Cont)

a bottle of bleach. A RFID system ultimately has no way to check if there has been a mislabelling or misreading short of human intervention, and if a supermarket RFID checkout operates transparently as customers leave the store they may never get a chance to correct the bill. Additionally a RFID tag, like a barcode, may fail to read entirely and thus requires a clerk on hand to step in and take control. And because we ask RFID to perform more than we do of barcodes, it presents more potential failure modes. Signal interference, unsavoury customers using electromagnetic shielding to foil the system in some way, and importantly a number of unresolved privacy issues stand ready to prevent the creation a fully automated RFID system in a store near you.

Even self checkout, a presently implemented intermediate stage between the now standard barcode checkout process and a 'walk-thru' system has its problems. Clerks are still required to be on-hand in case something won't scan or a customer doesn't understand the complicated instructions for scanning, weighing, and bagging their merchandise. If check-out proceeds without a hitch the clerk still needs to verify that everything was paid for.

For every step of a process in which we replace human operators with automation, we reduce the process' ability to cope with situations beyond some predefined set of assumptions, circumstances and rules. Machines cannot at this time improvise, and so they can never deal with the full spectrum of possible exceptions in an automatic way. Any system which is going to function under all circumstances will need human elements included to take over when automation collides with exception. Human supervision and occasional intervention is still necessary for automated systems, and our recognition of this will enable a more robust path of growth towards the large scale RFID systems we envision.

Hoop-coil weighbridge optimizes RFID reading by respecting physics and penguin behaviour
Source: Beigel Technology Corp

Making Robust Systems

There is no simple solution to making a truly robust automated system, but to close we present a few key suggestions to help with the development of yours.

- Study the problem your system aims to solve. Observe the situation before making conclusions about its nature.
- Time and money spent up-front develop and refinement of the model and system design will save even more time and money later.
- It is absolutely crucial to test as much as possible before deployment.
- Dedicate a team to start-to-finish system engineering, development, and optimization.
- Prepared for human intervention in system operation during development.
- The system automation will evolve from the interactive process of its use.