In this chapter you will learn:

1. About the documentation needed for deployment
2. How training is managed
3. About data migration
4. What a beta release is
5. About different deployment strategies
6. About managing the risk of deployment
7. About organizing support and enhancement

The development team has produced the product and the testing team has provided approval; now it is time to release the product on the unwitting users. There is still much to be done. A fair proportion of this will be carried out in parallel with the earlier stages. The product is not just an item of software. It comes with documentation, training and, in many cases, revision of business practices. There will have to be regular backups and a support organization to handle the product while it is live.

At this point, the weary adventurer will be greeted with plaudits or with jeers. Either the process that has led up to this point has produced something that will be loved and cherished, or there will be nothing but moans and complaints. All fresh deployments have teething troubles, but the pain that has to be endured and the success of the delivery depends on the quality of the software development process and the individuals that produced it.

As with implementation, technical deployment is as diverse as the technologies that might be used. There is little we can discuss here in specific terms, though there are some aspects of deployment strategies that can be considered. Needless to say there is a lot of physical work involved in deployment, including installing computers and networks, installation of operating systems and other support software, and the installation of the application itself.

Deployment of the software is, however, only one small part of the deployment process. In fact, most of the cost and effort of deployment can go into related tasks, such as training, production of documentation and the setting up of a support facility. For new software there may well be a complete reorganization of business alongside the software installation.
In this chapter, a flavour of the aspects of deployment is given. We will look at aspects of documentation, training, migration of data, deployment strategies, risk management, support and enhancement. These are all non-trivial aspects of any project, and can be critical to the success of the project. Figure 18.1 shows the primary inputs and outputs of the deployment stage. The early requirements capture, in terms of business processes and use case descriptions, form the basis for training and user manuals. The integrated system is deployed. Once deployed, faults and change requests will come back through the help desk.

18.1 Documentation

A number of stakeholders require documentation for the support of a product, namely the operations team that keep the system going, the support team that deal with problems, the development teams that will enhance the system in the future, and last but not least the users of the application. For a product that is to be sold, there will also need to be marketing documentation.

Operations documentation needs to explain where to find software for deployment, how to deploy the software, how to start and stop the software, what backups of the data need to be taken, how to restore data, and what precautions and recovery operations are needed to cope with disasters such as a fire. This documentation need not be excessive, but it will take time to produce. Usually the development team will do this, as they have closest knowledge of the application structure. Operations on the whole need not know much about the functionality of the software; just how to get it up and running, and to keep it that way.

Support documentation needs to explain to the support team the actions needed if a user presents a problem. Simple problems can usually be handled by the support team directly, but anything that involves problems with the software will usually require assistance from other parties, usually the developers. Support
documentation is usually lightweight, and just needs to identify sources of assistance and provide emergency routines if there are problems that need to be handled urgently.

Development teams need to keep the outputs of analysis and design so that future enhancements to a system can be made. In an incremental delivery, the application will continue to be developed as parts of it are deployed. Keeping good records and maintaining them in a current form is a non-trivial task, and large teams often allocate someone to take prime responsibility for this. It can take up a substantial amount of time to maintain good analysis and design documentation.

User documentation is a description of how the application is to be used. You will be familiar with this type of documentation from software that you have bought for your PC. Following the development process described in this book, the basis for user documentation has been provided in the form of use case descriptions. Screenshots from the application can be used to supplement this. For a substantial application, it is common to utilize technical writers who can take the analysis and design documentation and construct a more accessible description of the system for the users. Developers are notoriously bad at producing this type of documentation, as they tend to have a focus on how it works rather than how it is used.

18.2 Training

Before deploying a system, various stakeholders need training. Training of the operations and support teams is usually done informally through a brief workshop. Training of users, however, can take up a considerable amount of time. They are the ones that need to know all about how to use the system.

Training material can be developed from the user manual or from the use case descriptions. For anything other than a simple system or a system with few users, a trainer is used rather than a member of the development team. The process of training will begin with briefing the trainer. Trainers provide a nice bridge between the technology experts and the end users.

Training will require a version of the system set up for training purposes. It is not sensible to train on the live system, as there may be inadvertent damage to the data. The system will need to be set up with dummy data, and usually there will need to be some process of restoring the system to a common starting point.

For very large applications, such as in call centres, a separate training facility is usually set up. There is usually enough turnover in the workforce for there to be regular training needed for new staff. Users need to be given plenty of time to explore and use a system in a training environment before they use it operationally.

Training is costly. There is the cost of the trainer and the training facility, but most of the expenditure is in terms of the time spent by users in learning the new system. To get some idea of the cost, suppose a call centre with 1000 staff takes on a new system that requires two days training, and that each member of staff costs €100 a day to employ; the total cost of training in terms of lost time and training facilities is likely to be of the order of €250,000!
The cost of not training can be higher than the cost of training. Though there is not the obvious impact of people being taken off productive work, a 20% reduction in productivity caused by introducing a new system without training could, over a month, cost twice as much as training the staff in the first place.

The logistics of training are problematic. Training needs to be as near to the initial use of a system as possible. If large numbers of people need to be trained, then it will need careful scheduling, and inevitably some people may have to be trained well before they can actually use the system live. Much depends on the deployment strategies.

18.3 Data Migration
Replacement applications will usually need to acquire the data from the applications that they replace. For example, a new customer service system would need the customer records from the previous customer service system. This will usually involve some form of conversion of the data.

If the applications both use databases of similar types, the conversion may not be too difficult. However, it would be surprising if the new application did not record some information that the old system did not, and it is highly likely that the structure of the data is different. Encoding of some of the data may differ too.

There will be a need to write special routines to transfer data. Knowledge of the data content for both applications is necessary, and it is usually the database administration team that undertakes this task. The transfer needs to be tested in just the same way that the application is tested, and there will need to be trial runs before the actual transfer of live information into the new system.

18.4 Beta Releases
For products sold in a market place or widely distributed in a large and dispersed organization, a process called beta testing is commonly used. This acknowledges the fact that live use of an application inevitably uncovers problems. After acceptance testing has given the go-ahead, the application is deployed to a limited set of users who use it for real work, accepting a higher risk of failure.

Beta testing will be supported as a live release, with suitable support. Defects will be identified and rectified ready for the full release of the application. In return, beta testing groups will usually be rewarded in some way, perhaps in terms of reduced licensing costs or closer involvement in package specification.

18.5 Deployment Strategies
There are really only two deployment strategies. Either the system is deployed to all the users in one go (known as ‘big bang’), or there is an incremental rollout. Wherever possible, incremental rollout should be used. This minimizes risk and allows for rollback if the deployment is unsuccessful or the application has serious problems.

Big bang, however, is sometimes the only possible solution. This is the case when an old system is being replaced, and the new and old systems cannot coexist. Big bang can involve a massive transfer of data to facilitate the change over.
are contemplating a big bang installation of a substantial system, then all of the quality assurance procedures, such as testing, need to be as rigorous and thorough as possible.

A failed big bang installation can have dramatic consequences. A famous example of a failed installation was for the London Ambulance Service Computer-Aided Despatch System in 1992. The system was designed to facilitate the speedy allocation of ambulances to emergency calls. For a variety of reasons the system rapidly ran into problems and had to be switched off, but only after ambulances had been despatched wrongly and emergency calls not responded to, with obvious dire consequences. Another example of big bang installation was the switch of the London Stock Exchange in 1986 to screen-driven trading, and although this had problems it did succeed.

Incremental installation makes more sense wherever possible. This allows for a small number of users to adopt a system initially, and for that number to grow over time. This way, any problems with the application can be ironed out before it has a massive impact on the operation of the business. It also allows for training of the users in line with the installation.

18.6 Risk Management

Deployment is the riskiest point in any system development. Even a minor fault in the application or the deployment process can cause the system to fail, with dramatic consequences. Thus care needs to be taken that a failure causes the minimal amount of damage. No matter how carefully the application has been tested, the deployment will uncover cracks. If the ship is soundly constructed, a few minor leaks can be tolerated and repaired while still afloat, but if a gaping hole appears then the ship needs to be quickly returned to dry dock before it sinks.

The most important thing to consider is what to do if the deployment fails, or the application fails shortly after deployment. The usual answer is to resort to the previous system. Ideally the old system should be on hot standby, waiting to be called up.

A deployment failure needs diagnosis, repair and then (hopefully) redeployment. Too many failed deployments, however, will ultimately destroy confidence and may result in the cancellation of the application.

18.7 Support and Enhancement

The application is live and the users are happy; now comes the long process of keeping them happy. This is the point at which software developers, and most books on software development, like to leave off. In fact, the vast majority of development work is not on new systems but on continual minor adjustments to existing systems. Applications can last twenty years or more, though in that time they may go through many transformations. At the very least they will be migrated from one hardware and operating system platform to another. Additionally they will go through endless changes, in response to the changing and growing needs of their users. Ninety per cent or more of the development effort may go into the so-called maintenance phase.
I prefer to use the term enhancement, as maintenance has some implication of ‘wearing out’, as with the maintenance of a motor vehicle. Software does not wear out, but the needs that it is intended to meet do change, and the expectations of the stakeholders change.

Change comes about in many ways. Minor changes come from day-to-day operational problems. A user may ask for a small amount of additional information to be displayed on a screen, or for the layout of a screen to be changed to improve a task. Major changes come about when there are changes in the business, such as adding a new product line that the sales system cannot support.

An organization needs to capture these changes, consolidate them, select the worthwhile changes, prioritize them and implement them. Briefly we will look at the way organizations do this.

18.7.1 SUPPORT
The first point of call for changes is usually the support or help desk. The sort of changes initiated there are basically of two types. Either the system is not doing something properly or well, or there are repeated requests from users asking how to do something that the system does not do.

Typically, a user will ring the help desk and the operator will take details of the problem or issue. The next stage is to prioritize the problem. Three levels of priority are usually assigned. Critical issues will involve the help desk in seeking immediate action. Urgent issues will require examination quickly. Normal issues can be considered in the due course of time.

User support will deal with an issue by firstly checking that a simple problem has not arisen, such as the computer not being powered on or having been unplugged from the network. They will also usually deal with routine matters, such as forgotten passwords. If the problem cannot be solved quickly and easily, they will then determine who best to call to rectify the problem. They will usually have a list of contacts that they will call to seek a resolution to the problem.

Once someone has been identified to check out the problem, the support operator will record the details of the issue and who is acting on it. They will then monitor at regular intervals the progress towards resolving the problem. When a problem is resolved, they will then advise the person who raised the problem and close the issue.

Support logs need to be reviewed regularly for possible enhancements to a system. Frequently recurring problems need action, as they are likely to be causing operational problems for the business and they waste support time and energy.

18.7.2 CHANGE CONTROL
A live application will result in requests for change from many sources. Users and their supervisors will request minor changes to make their day-to-day tasks simpler. Business managers will usually request more substantial changes in anticipation of major changes in the way a business operates. Senior management may be the ones to propose very radical changes in the light of changes in business strategy.

Change proposals need to be recorded and reviewed. The common method is to have a change request form (paper or electronic), that specifies the change, the
reasons for change, and the benefits that the change will provide. Change proposals will come from a number of sources within the business.

The change proposals need to be reviewed regularly. Firstly the development team need to group them, looking for related changes. They then need to provide provisional estimates of the cost of making the changes. Once the changes have been checked for viability and costed, a review panel will consider all the changes.

The review panel will consist of various stakeholders in the system, including business managers from areas where the system is used. Inevitably the number of changes will exceed the ability of available resources. The panel must prioritize the changes and choose those that need to be implemented. The review panel will take into consideration all the issues we covered in project inception, focusing on cost and benefits.

Implementation of changes basically means either revising existing use cases or adding new ones. The processes that we have discussed for development of use cases are entirely appropriate for managing the changes. Revised or new use case specifications need to be produced. Any necessary systems analysis for substantial changes is carried out. Design or redesign is undertaken. Then follows construction, testing and deployment. Changes to documentation and necessary training will be incorporated in the deployment.

18.8 Conclusion

Getting the completed system onto users’ desks, making sure they are happy, and keeping them that way, is the most important step. It can be quite messy for a new system, as lots of people will be doing things for the first time. It requires careful planning from the project manager and other managers involved in the deployment.

As with implementation, this is very dependent on organization. Therefore the case studies and exercises have been omitted. You may like to consider the issues around the deployment of the trading and retail outlet systems.

**REVIEW QUESTIONS**

1. What is the source material for training and user manuals?
2. What problems do replacement systems introduce?
3. Why are big bang deployments risky?
4. How are support and enhancement organized?
5. Once a system is deployed, how are changes to the system organized?