Chapter 9

Stage 6: Creating Work Products

9.1 GUERRILLA ANALYTICS WORKFLOW

Up to this point, we have discussed extracting data, loading that data, and writing code to manipulate and analyze that data. When analytics work is completed, there is something to deliver. The Work Products stage of the Guerrilla Analytics workflow is concerned with wrapping up analyses and other activities so they can be delivered either back to the team or to the customer. Furthermore, you must track those deliverables when they leave the team so that you can easily identify them should they return to the team for modification. You can see Work Products in the Guerrilla Analytics workflow of Figure 28. Note that we have skipped a stage called consolidate. We will come back to that in Chapter 12.

9.2 EXAMPLES

There is a multitude of work products that a Guerrilla Analytics team produces on a typical project. Here are just a few examples that you may recognize in your own work.

- **Data integration:** A data flow for taking a third party data source that is refreshed at some interval, cleaning and profiling this data source and joining it into the core project data for analysis.
- **Data profiling:** A summary of a new data source that has been loaded into the Data Manipulation Environment (DME). This is reported back to the team and customer on lessons learned about the data.
- **Data testing:** Checksums of some data received from the customer and a report on all checksums and exceptions.
- **Data analytics and visualization:** Code to extract a subset of sales data from a given year and identify all outliers by sales price and region. A visualization of these regional outliers using a box plot.
- **Third-party tools:** A data quality assessment of all data from a given system extract produced using a proprietary data quality tool.
- **Marked-up data:** An export of a financial data sample from the DME into a spreadsheet so that it can be reviewed and marked up. It is expected that these markups will be imported back into the DME and used in future analyses.

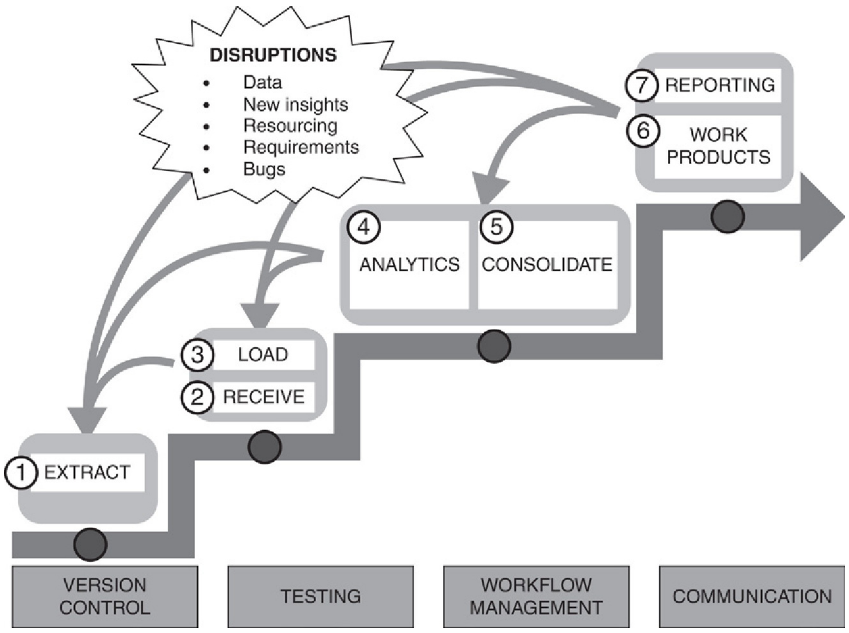

FIGURE 28 **The Guerrilla Analytics workflow**

- **Rework:** A modification of an existing work product from several weeks ago to add a new data feature to it. The new version of the work product is reissued back to the customer with a report on any implications of the new data feature.
- **Data sample:** An extract of the recently loaded address dataset so the client can confirm he delivered you the correct version of the data.

Looking through these work products examples, there are some common features to note.

- **Mix of programming languages:** You are sometimes using several programming languages in the creation of a work product.
- **Use of proprietary tools:** You will sometimes be using a proprietary tool that might have its own file formats, project files, and other ancillary materials.
- **More than just program code:** You are sometimes creating presentations, spreadsheets, and written reports. Not every work product involves program code.
- **Multiple iterations:** There are occasions where a work product is returned to the team for additions or corrections before being reissued to the customer. This raises the question of version control and impact assessments.
- **Complexity:** Some work products are complex data flows that will involve multiple code files and data steps. In many ways they are a significant piece of software in their own right and may take days to complete.

- **Simplicity:** Some work products are simply an extraction of an existing dataset – a single line of program code.
- **Data/people feedback loops:** There are occasions where data is leaving the data team, being modified in the broader team or by the customer, and then being returned into the data environment.
- **Multiple people involved:** Some work products may have multiple team members working on and reviewing them.

There is the potential for a lot of complexity here because of the wide range of activities engaged in by the Guerrilla Analytics team. Before going further it is worthwhile to define the concept of a work product.

9.3 THE ESSENCE OF A WORK PRODUCT

If we simplify things, there are three components to a work product.

- **Data Sources:** As you have seen in earlier chapters, data must be either raw data or data that the team derives from this raw data.
- **Generators:** These are the program code files and other data manipulation activities that create the work product from a data source.
- **Outputs:** These are the datasets, images, models, reports, and other derived materials that the generator creates from the data sources.

To preserve the provenance of a work product, you need the generators and their data sources. Apply the generators to the data sources and outputs should be rebuilt from scratch. Everything to do with preserving provenance in a data source is based on this simple structure. The next section discusses examples of how this structure is often broken.

9.4 PITFALLS AND RISKS

The main pitfalls to avoid when creating a work product are as follows.

- **Clutter of the DME and file system:** Where do you store all of these work products in the DME? Where do you store related work on the file system? Without guidelines, your DME and file system will be a cluttered mess.
- **Work product owner cannot be identified:** Without proper processes in place, a work product's creator and owner cannot be identified. This makes review and handover difficult and impedes your ability to manage the team's activities.
- **Work product versions confused or lost:** How do you identify versions of work products? In a cluttered DME, it becomes impossible to identify the latest version of a work product and compare it to previous versions. This is exacerbated when a work product is composed of many code files or has several outputs.

- **Work products deleted:** If you cannot identify the components of a work product on the DME and on the file system there is a higher risk that some critical component of the work product will be deleted or lost.
- **Knowledge is never consolidated:** If work products are developed in an ad-hoc fashion, it becomes difficult to identify common implementation and data needs. Team knowledge is never consolidated as every work product effectively starts from scratch. A customer can ask for a work product from one team member on one day and get a different answer from another team member on another day.
- **Team cannot identify its own work products:** Without work product tracking in place, a team cannot easily recognize a work product that is returned to the team and jump to that work product's outputs and code files for debugging or modification.

The following practice tips are guidelines to help mitigate these risks.

9.5 PRACTICE TIP 34: TRACK WORK PRODUCTS WITH A UNIQUE IDENTIFIER (UID)

9.5.1 Guerrilla Analytics Environment

The team is creating many work products and many of those work products are being delivered to the customer. Work products may be composed of multiple files including code files, images, dashboards, documents, and more. The customer will often modify the work product files they receive and perhaps return to the team to question and better understand the work product. Since these work products are leaving the team's data environment, the team has little control over their tracking and use (or abuse).

9.5.2 Guerrilla Analytics Approach

Every work product should be given its own UID. The best identifier is simply a number that increments with every new work product the team produces. With a UID established for a work product, the following should then be done.

- The UID should be embedded in the name of every work product file.
- Work product files should be saved in a folder, which also has this UID in the folder name.
- Work product activities in the DME should be stored in a location that also has the UID in its name.

Figure 29 summarizes these tips in a typical folder and DME structure. Work product 103 has its own folder on the file system and its own "folder" in the DME. Every generator and output in the file system has the UID in its file name. All intermediate and result datasets for the work product are stored in the "103" location in the DME so they are grouped together.

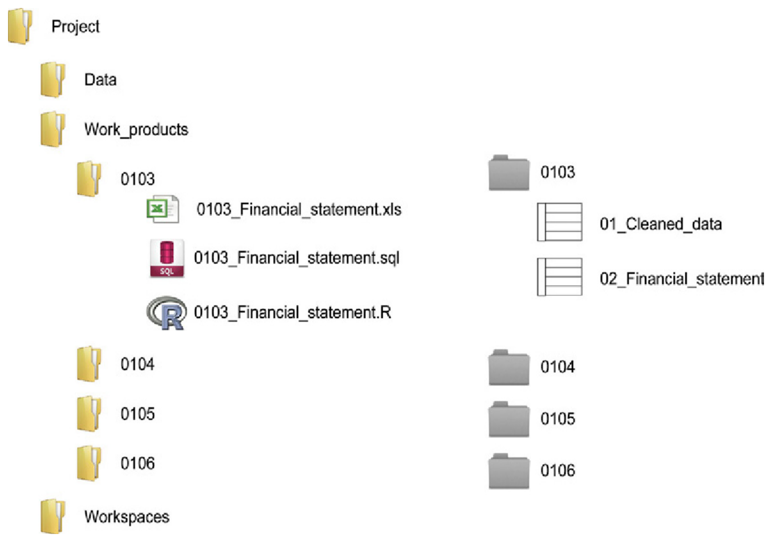

FIGURE 29 Work product UID in the file system and the DME

War Story 12: The Blame Game

Matt arrived onto a project to take over from another analytics manager. It was his second day in charge of the analytics team working in a wider project of about 50 analysts. The project was investigating the finances of a high-profile corporation that has been accused of fraud. There was a lot of pressure on the whole team with every result and report subject to scrutiny. The analytics team was overworked and being blamed for all project errors. With a project lead who didn't have an analytics background, it was difficult to gain their trust and understanding of complex analytics outputs. Matt was about to be a contestant in "The Blame Game."

He spent his first days getting familiar with work streams, the assigned data analysts, and the corresponding business analysts. All was going smoothly. Then Matt attended a meeting where a senior team member brandished an incorrect spreadsheet that they claimed to have received from the analytics team several months ago. Their accusation was that this incorrect spreadsheet had misled the wider team and almost caused the release of an incorrect report from the project. Analytics were to blame and they needed to take more care with their outputs. Diplomatically, Matt tried his best to reason with her and asked for some help tracing where exactly the spreadsheet had come from. His analytics team member claimed not to recognize it. Without an evidence chain and having only 2 days on the project, Matt was in a weak position. He took the offending spreadsheet back to his team and began looking into how the lapse in quality might have occurred.

After some "forensic" investigations of their own, Matt's team found the original spreadsheet and guess what? Somebody outside of the analytics team had changed its contents. The spreadsheet that left the team mailbox was correct. Matt's team wasn't to blame at all but had no way to back this up.

Matt immediately began a work product UID process where all files leaving the analytics team had this UID embedded in their file names. This was communicated to the wider project team. The simple process quickly put an end to blame games as there could be no doubt about what the analytics team had actually produced. It also made communication with the wider project team much easier. Instead of "spreadsheet from so-an-so.xls" the team could now talk about "work product 56." Within 2 months, work product UIDs were the standard vocabulary of the project and instrumental in planning and coordinating project phases. A simple Guerrilla Analytics tactic had protected the team and improved the wider project. Game over.

9.5.3 Advantages

The simple convention of work product UIDs has several advantages.

- **Communication:** By embedding the work product UID in the file names of work products, it is quite likely that the UID will be preserved. Better still, customers themselves will recognize the number. In projects where this is most successful, work product UIDs become part of the customers' vocabulary. They begin asking for a version of "work product 205" or ask you to send them a new version of "work product 500." This simple approach is surprisingly effective.
- **Location on the file system:** By creating a space on the file system that has the UID in its name, you can quickly locate a work product's files and know that every generator and output related to that work product must be somewhere within that file folder. You saw a similar approach with data UIDs.
- **Location in the DME:** By creating a location in the DME that has the UID in its name, you can quickly locate all datasets related to a work product and easily separate those datasets from other DME activities.

9.6 PRACTICE TIP 35: KEEP WORK PRODUCT GENERATORS AND OUTPUTS CLOSE TOGETHER

9.6.1 Guerrilla Analytics Environment

As mentioned earlier, every work product has three components.

- It has a data source.
- It has the program code and other data manipulation activities.
- It has some outputs created by the generators that are delivered to the customer or the team.

When a customer wants to discuss or modify a work product, he or she will return to the team with the work product output file. Both generators and outputs are critical to the reproducibility and data provenance of the work product since they encompass the data team's perspective and the customer's perspective, respectively.

9.6.2 Guerrilla Analytics Approach

Some teams will keep a separate folder for delivered work product files. This causes confusion. Keep work product generation code and work product outputs right beside one another in their work product folder.

Looking again at Figure 29, there are two analytics code files with the .SQL and .R extensions. There is also a spreadsheet file with the extension .XLS. The spreadsheet is a cut of data created by the analytics code files. All files are within the same work product folder and have the same file name so you know they belong together.

9.6.3 Advantages

The simple convention of keeping generators and outputs together offers several advantages.

- **Know what was issued to the customer:** Every work product has a definitive copy of what was sent to the customer. This helps resolve situations where the customer has modified a work product and there is a question over data provenance.
- **Traceability improves:** Debugging is now easy because everything that is needed to understand and reproduce a work product is located in one place. The analyst can first check the work product output file and confirm any problems. They can then follow up by examining the associated code files and DME datasets that generated the work product.

9.7 PRACTICE TIP 36: AVOID CLUTTER IN THE FILE SYSTEM

9.7.1 Guerrilla Analytics Environment

A work product has several components in terms of files that create the work product and perhaps versions of the work product. The work product's folder quickly becomes cluttered with files making it difficult to know what files to review and how to associate work product generators with work product outputs. Additional supporting materials such as emails and documentation are important but do not contribute to reproducing the work product output. How do you keep all relevant materials together but avoid accumulating confusing clutter?

9.7.2 Guerrilla Analytics Approach

A simple folder structure reduces clutter. The key insight here is that the latest version of the work product must be accessed easily and often. Older versions and supporting materials are rarely accessed and so can be filed away.

The minimal folder structure that serves the vast majority of work product scenarios is as follows.

- **Keep most recent materials in the root:** Keep only the most recent work product generator and outputs in the root of the work product folder. They are accessed most often.
- **Archive:** File away older versions of generators and outputs in a subfolder called "archive." The archiving approach need not be any more sophisticated than this as older materials are accessed less often.
- **Supporting:** File away supporting materials such as emails and documentation in a subfolder called "supporting."

Figure 30 illustrates a work product folder with this structure. Random supporting documents and emails are in the "supporting" subfolder. An older version 01 is in the "archive" subfolder.

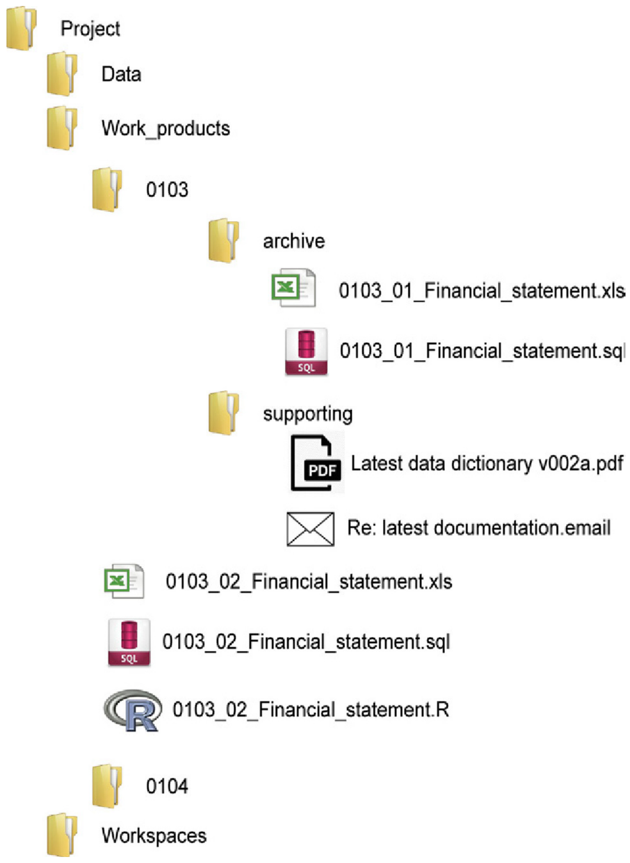
Project
Data
Work_products
0103
archive
0103_01_Financial_statement.xls
0103_01_Financial_statement.sql
supporting
Latest data dictionary v002a.pdf
Re: latest documentation.email
0103_02_Financial_statement.xls
0103_02_Financial_statement.sql
0103_02_Financial_statement.R
0104
Workspaces

FIGURE 30 A simple work product folder structure

9.7.3 Advantages

This simple filing tip offers these advantages.

- **Ease of debugging and modification:** On the majority of occasions a team will revisit a work product folder for two reasons. They want to grab the outputs so they can discuss them with the customer. Alternatively, they wish to delve into the work product code to understand how the work product was created, perhaps because it is incorrect or it is being handed over between team members. Knowing that only the latest relevant materials are in the root of the work product folder saves time sifting through older or unrelated files.
- **Versions are easily identified:** All older versions of generators and outputs are in one clearly labeled location – the "archive" subfolder.

9.8 PRACTICE TIP 37: AVOID CLUTTER IN THE DME

9.8.1 Guerrilla Analytics Environment

Much as files build up in the file system, datasets will build up in the DME. Trial analyses that were abandoned and multiple versions of a work product contribute to clutter. This clutter causes confusion and wastes time when reviewing work products and when trying to locate the source of a particular work product output file.

9.8.2 Guerrilla Analytics Approach

Two practices help with DME clutter, much like the approach taken with clutter in the file system.

- Establish a name space for a work product that is named with the work product UID.
- When a work product is executed, first clear out this name space and then rebuild all datasets generated by the work product code.

9.8.3 Advantages

This DME namespace approach combined with a tear down of all datasets has the following advantages.

- **Removes orphaned datasets:** Old orphaned datasets that are no longer part of the work product are cleared out. This makes it much easier to review and understand a work product because only relevant datasets are in the DME. Moreover, the risk of accidentally referencing an old dataset is eliminated.
- **Removes the risk of accidentally orphaning a dataset:** By requiring all derived datasets to be created when work code executes, temporary datasets that may be needed for reproducibility will be reliably present.

- **Easy location of work products:** Because the work product's datasets are located in a well-identified namespace, it is easy to find all datasets related to a given work product just by knowing its work product UID.

9.9 PRACTICE TIP 38: GIVE OUTPUT DATA RECORDS A UID

9.9.1 Guerrilla Analytics Environment

Placing a work product UID in work product file names helps identify the original output when it comes back to the team. However, what about data records within a work product file? Rows of data in a spreadsheet may be shuffled by the customer. Worse still, they may be modified or broken by the end user. Questions about dashboards and visualizations are hard to resolve when the underlying data cannot easily be referenced. Conversations with the customer are much easier if both parties have a way to precisely identify data records.

9.9.2 Guerrilla Analytics Approach

All data records in a work product output should be given a deterministic UID at the record level. As we have seen before, in the absence of primary keys, the best UID for a row of data is a hash of the entire row of data.

9.9.3 Advantages

Having a UID for each record offers the following advantages.

- **Comparison:** When a data record is returned to the team, the returned record can easily be identified and compared to the record that left the team originally.
- **Coordination:** When trying to collaborate on data, it is risky to talk about the "fifth" record or the record for "person named James Smith." If you have a record UID there can be no doubt about the record you and the customer are referring to.
- **Reintegration of data:** If data from a work product needs to be reintegrated into the DME and joined into data already in the DME, having a UID greatly helps this process.

9.10 PRACTICE TIP 39: VERSION CONTROL WORK PRODUCTS

9.10.1 Guerrilla Analytics Environment

It is inevitable that work products will go through several iterations with the customer or peer review within the team. This will happen either because requirements change as the customer explores the data with the team or it may happen because of mistakes in the work product. How should the team deal with these circumstances?

9.10.2 Guerrilla Analytics Approach

Establish a version-control process such that there are clearly identified versions of the work product generators and corresponding clearly identified versions of the work product outputs. Retain all versions of generators and outputs.

Figure 31 shows a simple example of how this can be done. On the left, files have a version number embedded in their file name. This example has a version 01 and a version 02. In the DME on the right, a version 01 and version 02 schema give name space separation to the datasets associated with each version.

9.10.3 Advantages

With a version-control process in place, much of the confusion of the Guerrilla Analytics environment is avoided. In particular, the team can show a customer the impact of changes to the work product by comparing the latest version to some previous version. The inevitable question of "why have the numbers changed" is easily answered by comparing versions of output datasets.

Second, the latest version of the work product is easily identified from the file naming and dataset namespace.

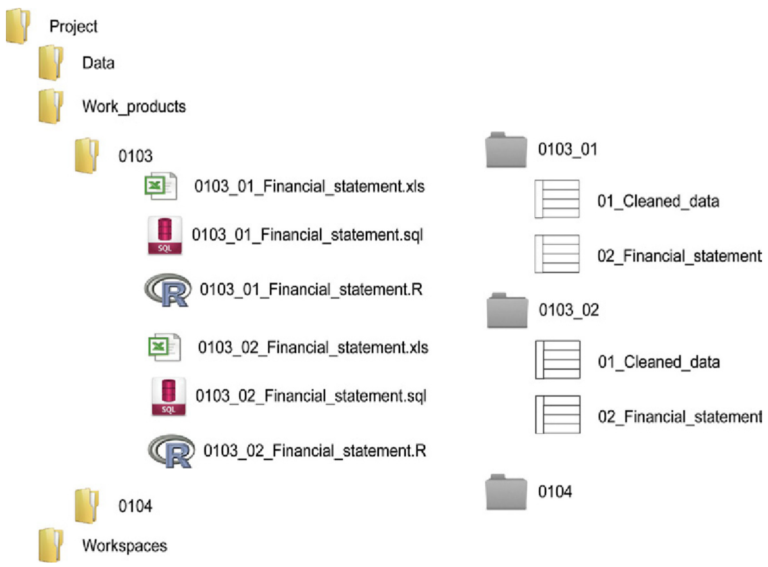
Project
Data
Work_products
0103
0103_01_Financial_statement.xls
0103_01_Financial_statement.sql
0103_01_Financial_statement.R
0103_02_Financial_statement.xls
0103_02_Financial_statement.sql
0103_02_Financial_statement.R
0104
Workspaces
0103_01
01_Cleaned_data
02_Financial_statement
0103_02
01_Cleaned_data
02_Financial_statement
0104

FIGURE 31 Simple version control in the file system and DME

9.11 PRACTICE TIP 40: USE A CONVENTION TO NAME COMPLEX OUTPUTS

9.11.1 Guerrilla Analytics Environment

Work products are composed of one or more code files and produce one or more outputs. There may be code files that only manipulate data in the DME. There might be other code files that take data from the DME and generate several plot files. There may be code files that create multiple data samples that are delivered to the customer in spreadsheets for further analysis. Without putting some type of process in place there is no easy way to know where an output came from. This situation is made worse when one generator creates several outputs.

9.11.2 Guerrilla Analytics Approach

An output should have the same name as the code that generates the output. If the code generates multiple outputs then the code file name should prefix those multiple output file names. Table 1 illustrates some examples.

- In scenario 1, there is a clear one-to-one mapping between the generator file and the output file as implied by their having the same names.
- In scenario 2, one generator is producing two output files. Since they all share the same file name prefix, it is clear where the generated files came from.
- Scenario 3 is a mix of scenarios 1 and 2. From the naming convention we can easily see that D300_data_prep.SQL generates the CSV file output. Separately D300_plotting.R generates the other two outputs.

9.11.3 Advantages

This simple naming convention has the following advantages.

- **Speed of debugging:** The code that generated one or more output files is easily identified in the work product folder.
- **Data provenance:** The link between the DME and the work product is maintained with no documentation overhead.

TABLE 1 Example Scenarios and Naming Convention for Complex Outputs

	Scenario	Generator file name	Output file name(s)
1	One code file and one output	D050_03.SQL	D050_03.XLS
2	One code file and multiple outputs	D009_01.py	**D009_01**_customers.XLS **D009_01**_addresses.XLS
	Multiple code files and multiple outputs	D300_data_prep.SQL D300_plotting.R	D300_data_prep.CSV **D300_plotting**_Bar.PNG **D300_plotting**_Scatter.PNG

9.12 PRACTICE TIP 41: LOG ALL WORK PRODUCTS

9.12.1 Guerrilla Analytics Environment

It is always useful to store some additional information about a work product. For example, it would be good to know which team member did the work as well as which customer it was delivered to. Some basic management information (MI) on date of request and date of delivery helps determine bottlenecks in delivery. What is the best way to track this type of information?

9.12.2 Guerrilla Analytics Approach

Similar to data tracking, the simple Guerrilla Analytics approach is to have a work product log in the root of the work products folder. Entries in this log use the work product UID for tracking and the team completes whatever information about the work product is deemed useful for data provenance and tracking.

9.12.3 Advantages

It is not unusual for a Guerrilla Analytics team working in a fast-paced environment to deliver hundreds of work products over several months. Having a log of who worked on what is helpful when old work products come back to haunt you or when you need to locate the most appropriate team member for a particular piece of work. The log also enables the team to clearly articulate their efforts to the customer and identify the major drivers of customer demand.

9.13 WRAP UP

This chapter has discussed creating and tracking work products in a Guerrilla Analytics environment. Having read this chapter you should now know the following.

- The risks associated with poor management of work products on the file system and in the DME. Specifically:
  - Clutter of the DME and file system.
  - Work product owner cannot be identified.
  - Work product versions confused or lost.
  - Work products deleted.
  - Knowledge is never consolidated.
  - The team cannot identify its own work products.
- Some simple practices and conventions greatly mitigate these risks. These primarily involve:
  - Giving every work product a UID.
  - Giving every data record in a work product its own UID.
  - Using sensible file naming conventions to identify versions of work product files.

- Avoiding clutter in the file system by having an archive process for older or irrelevant materials.
- Using a name space in the DME to group together all datasets related to a version of a work product.
- Versioning work products so that the impact of changes can be understood and quantified.