Exporting search results with Mascot Parser

ASMS 2003

MATRIX SCIENCE

Exporting search results with Mascot Parser

- Why is it needed?
- What does it do?
- What doesn't it do?
- What platforms / tools can be used
- Case study

ASMS 2003

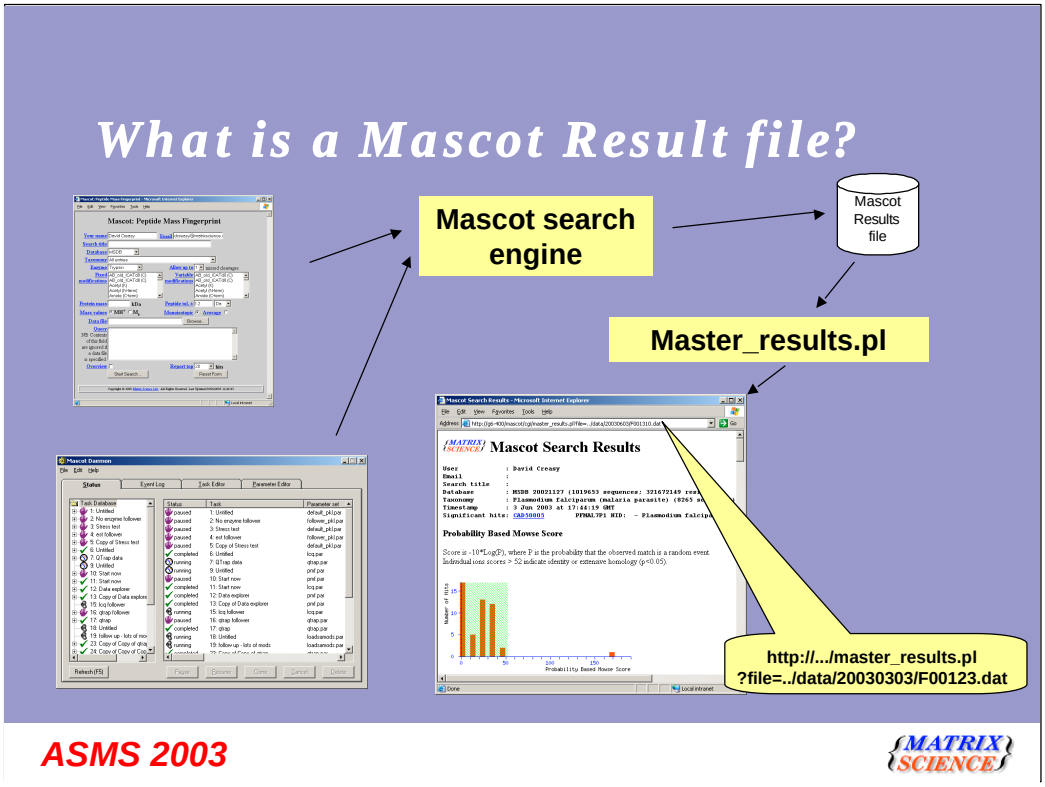
MATRIX SCIENCE

In this session, I'm going to describe Mascot Parser - a toolkit to help you take Mascot results files and export the data in them to a database or an Excel spreadsheet.

So, I'll be covering what it does and equally importantly what it doesn't do.

I'll next give a quick run down of the programming tools that can be used with the toolkit,

Finally, I'll give a very simple case study example of a project using Mascot Parser.

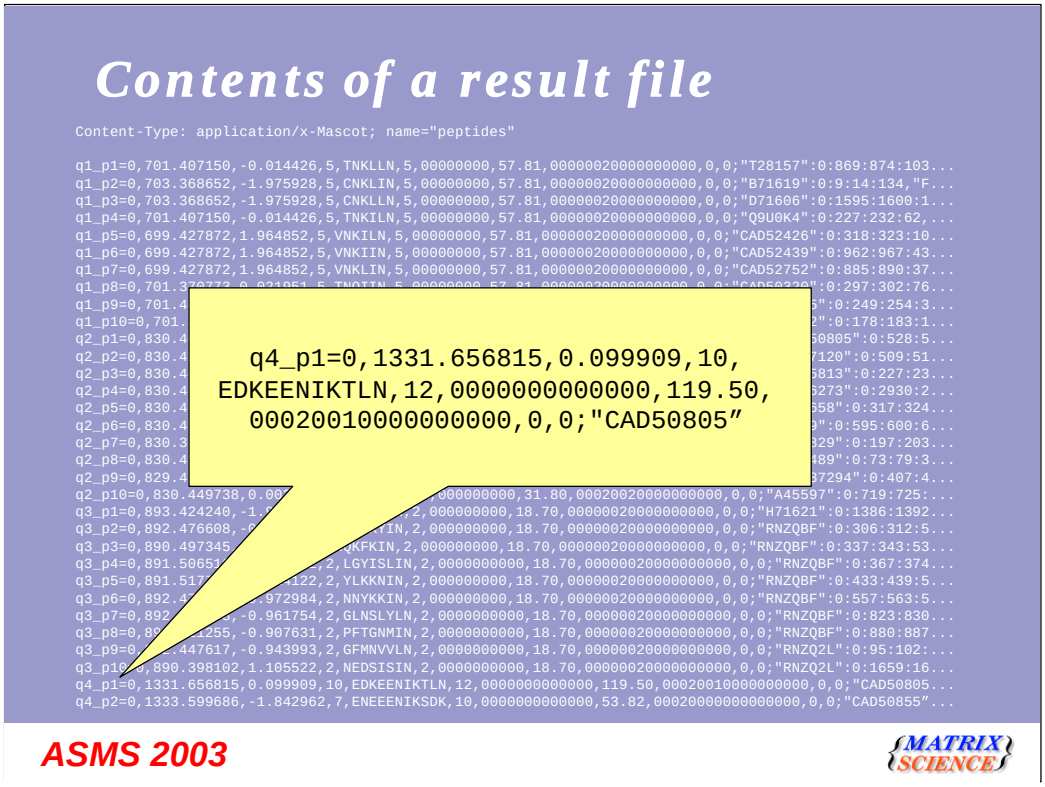


I should just briefly explain what a Mascot Results file is, because this won't be obvious to some of you.

When a search is submitted, either from a search form, or from Mascot Daemon, the search engine runs and creates a results file on the server. It is important to note that the search engine does not output the search results that you see on the screen. A perl script on the mascot server then reads the results file and creates html output.

This means that you can go back and look at old results without re-running the search.

It is also important that you don't write programs that screen scrape. By this mean I that you shouldn't look for specific text in the html output and then 'parse' that text. Suppose for example that you looked for the text "Significant hits:" here, and one day we decided to tidy up the grammer and if there is only one hit we put the text "Significant hit" - then the screen scraping would not work.



We don't have time to look at a results file in detail, but this is the peptides section of a results file, and this is what is used to create the peptide summary report

Problems with parsing results file

- Protein summary is quite easy
- Concise protein summary groups similar proteins
- Peptide summary groups all the peptides into proteins and sorts by score
- Mascot 2.0 reports will use Mascot Parser

ASMS 2003

MATRIX SCIENCE

The protein summary is quite easy to extract from the results file, and quite a lot of Mascot users have done this. If that is all you need to do, then it's quite likely that you don't need Mascot Parser.

The concise protein summary is a little harder. This was introduced in Mascot 1.9, and basically groups similar proteins together.

The peptide summary report is quite involved. In the previous slide I showed the peptides section of a results file, and this is what is used to create the peptide summary report. There's no information directly here about what the top scoring protein is. It has to be determined by calculating the score for each protein that has one or more peptide hits. The list then has to be sorted by score and similar proteins have to be grouped together. Furthermore, it has to be done efficiently and clearly, this is more than a couple of lines of code.

With Mascot 2.0, we are using Mascot Parser, and this speeds up the loading of large reports.

So, finally, what does Mascot Parser Do?

Enables a programmer to *easily* gain full access to Mascot results

Enables a programmer to put results into a database

Enables a programmer to *easily* display custom reports

What doesn't it do

- Make the tea / coffee
- Access configuration files and utilities such as ms-getseq and ms-gettaxonomy
- Cannot be used with data on Matrix Science public web site
- Absolutely no use to anyone who cannot write a small computer program

ASMS 2003

MATRIX SCIENCE

I wouldn't be being fair if I didn't say what it doesn't do.

Somewhat obviously, it doesn't make the tea - although, being English, we are working on that aspect.

Secondly, it doesn't currently allow access to configuration files such as mascot.dat - that is changing, and we are adding support for this in the next major release. Also, it cannot return for example the taxonomy ID for a particular protein - it doesn't call the utility to do this

Finally, it is of no use to anyone who cannot program - it is a toolkit.

I can't program - can I fall asleep now?

- **Do you have access to any programming resource?**
- **Have you ever written a few lines of Perl?**

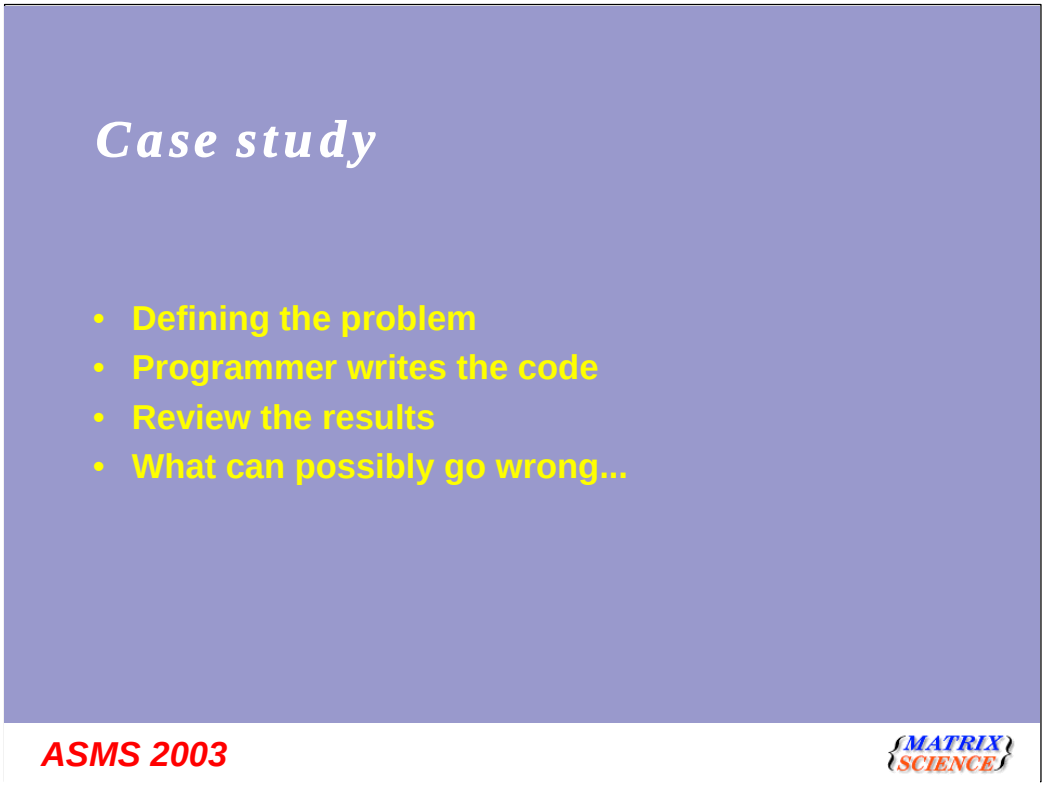
ASMS 2003

MATRIX SCIENCE

Don't let me keep you awake - after all you have several hundred talks and 5000+ posters to look at over the next few days, so now mught be a good time for a nap.

However, just before you nod off, can I ask two simple questions? The first is "Do you have access to any programming resource? If the answer to this is "yes", then the rest of this talk is actually directed towards you. I will show a couple of slides with code very briefly - but the rest of the talk is more relevant for someone trying to get a programmer to developa tool

Or, have you ever written a few lines of Perl? I am continually amazed at the resourcefulness of people who would chiefly consider themselves to be biochemists ormass spec experts - so many people seem able to fix computers, replace chips on PCBs, fix pumps, and even program a little. If I've not convinced you to stay with me, then all I ask is that you don't snore.



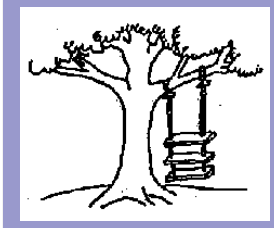
To make this slightly more interesting, we are going to provide a follow a story of how some software might get developed for your lab. The steps are:

The problem is defined - hopefully by someone who actually understands what the issues are.

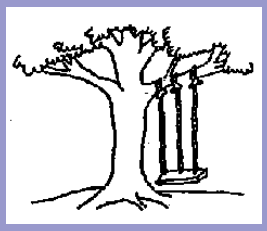
The programmer performs some system analysis, and then writes some code.

We will then review the results? So, what can possibly go wrong?

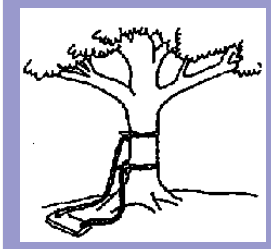
Software development process



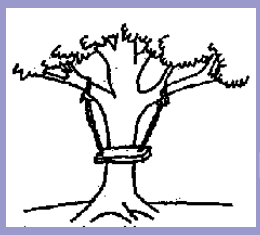
As proposed by the project sponsor.



As specified in the project request



As designed by the senior analyst



As produced by the programmers

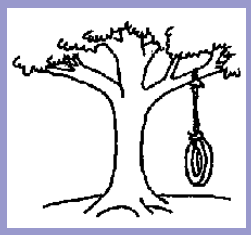
ASMS 2003

MATRIX SCIENCE

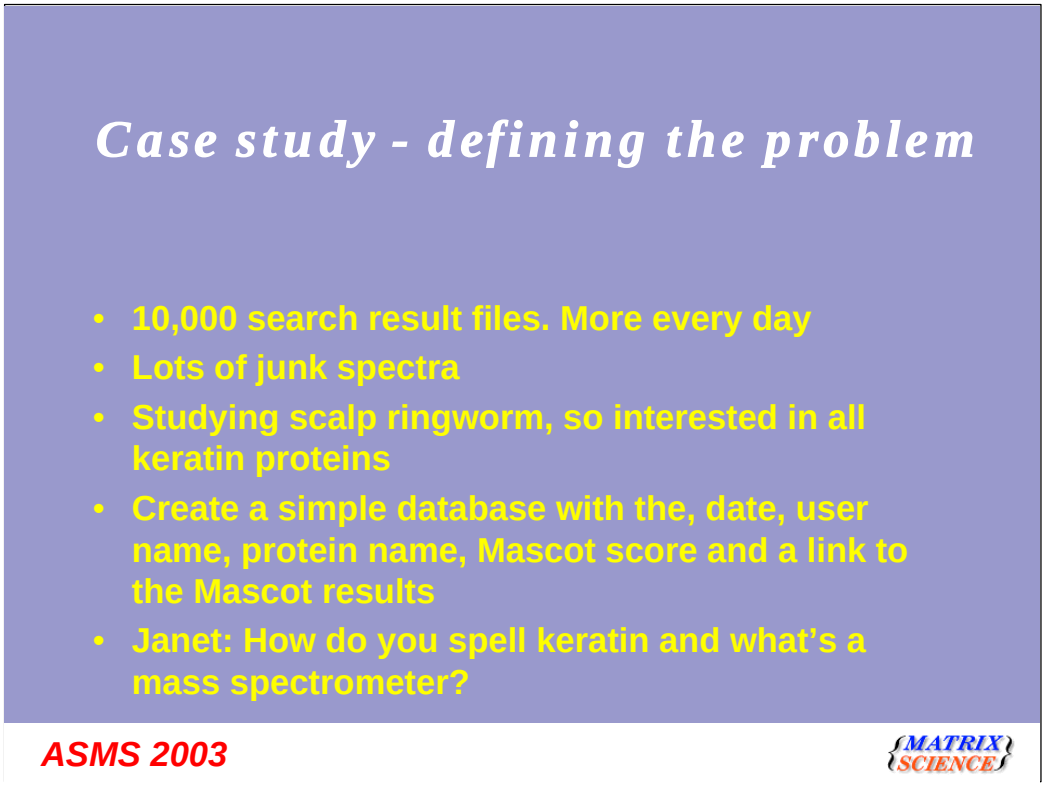
These pictures really show my age - about years ago, it was common to see this sequence in most software development books.

Software development process

As installed at the user site



What the user wanted



So, bearing in mind what I've just said, you can probably see that this project is not going to go very smoothly. I must state that this project has no resemblence to any project that we know about, but to protect the innocent, names, identities and sample source code have been changed.

Mass spec expert 'Harry ' explains the problem to programmer who shall call 'Janet'

Harry explains that they have 10,000 search result files, many of them contain junk spectra.

Harry: Due to a recent discovery, I really just want to see all the keratin proteins - if I could just search for that in these 10,000 results I'd be very happy.

Janet: So that's all that you want to see?

Harry: As long as I can get a link back to the Mascot results page, this will be fine for a first step.

We will just have to break from our case study for a moment to see what options are available for Janet the programmer,

So, the programming languages supported are C++, Perl and Java. The Java version is not quite available yet, but we have been uing it in house for the last couple of months.

It's a technical detail, but most C++ compilers are supported, and you can use both types of library.

Most versions of perl are supported and we will do our best to help you with less common ports of perl

And it will run on all the platforms supported by mascot

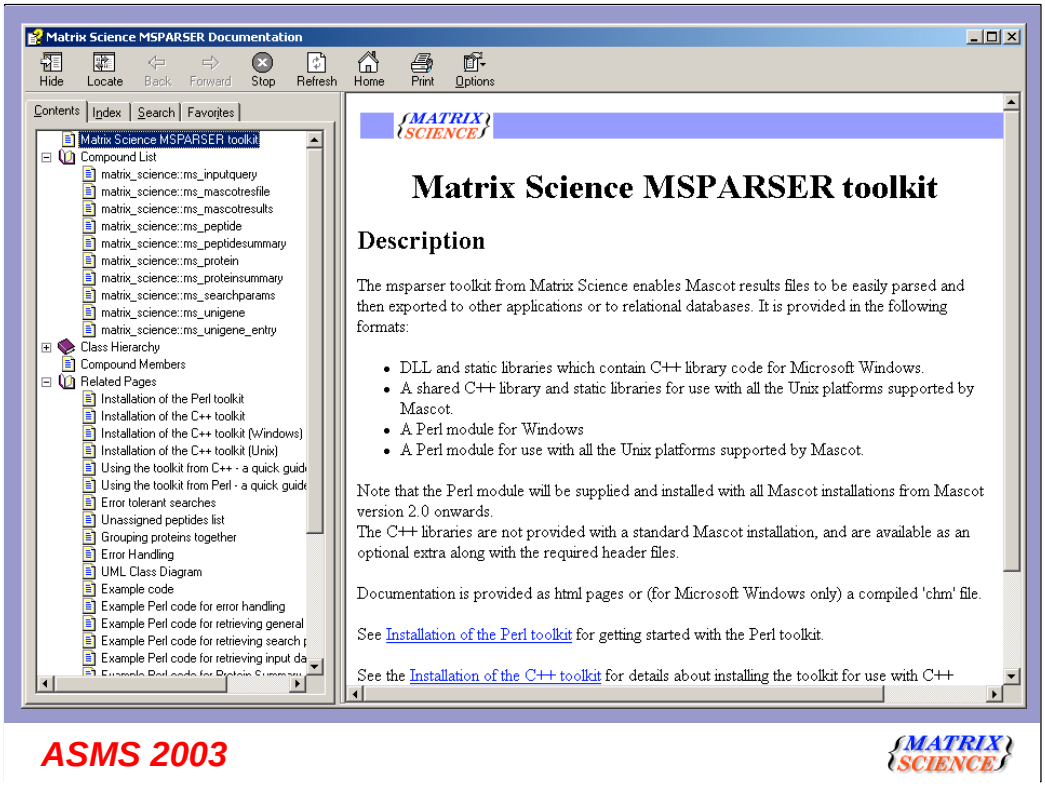
What's included in the toolkit

- Libraries for Perl, C++ and Java for your platform
- Example code for all three languages
- Online documentation - available as a compiled help file for Windows or html pages for Unix
- Free support

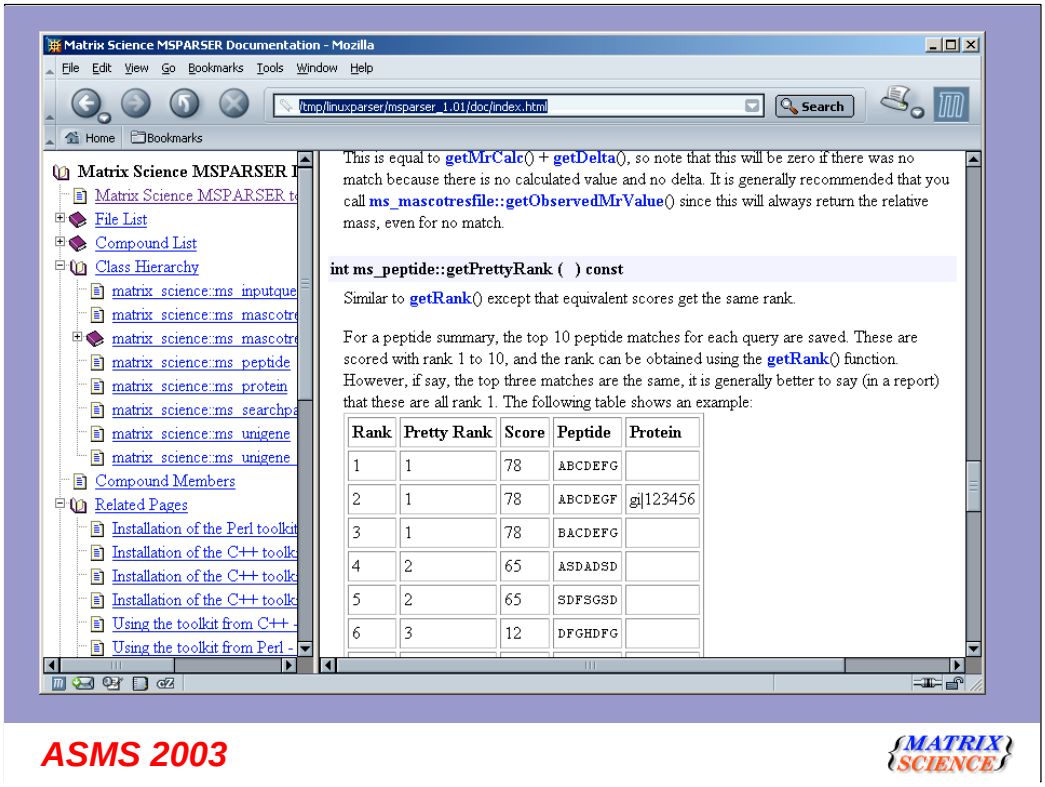
ASMS 2003

MATRIX SCIENCE

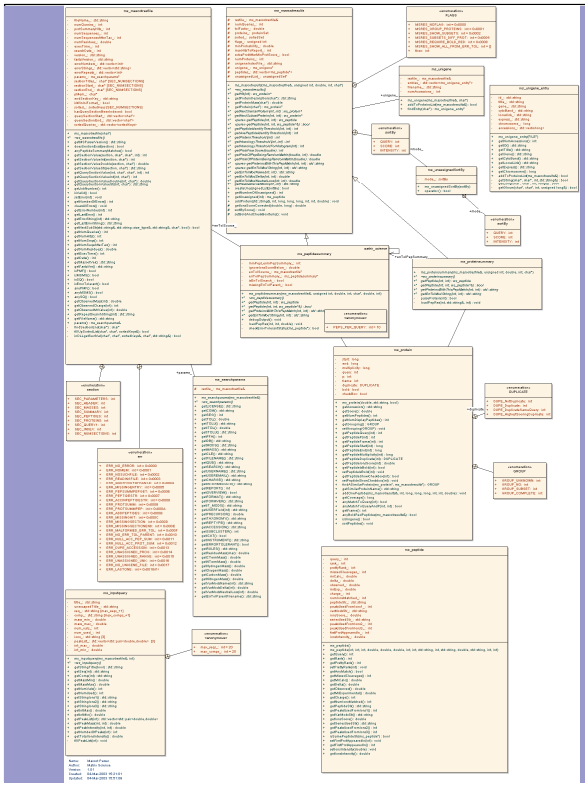
The package contains all that you need - the libraries for the relevant language, example code and online documentation



There is extenisive help - either in a compiled help file for Windows, or



Html help for other platforms



- ‘Simple’ object oriented design that allows access from multiple languages and platforms
- 10 ‘Classes’ to represent proteins, peptides, results, input parameters etc.
- Support for all search types and reports including Unigene, error tolerant searches

The toolkit uses a very simple object oriented design. In order to support a wide range of languages, and to simplify the learning process, we have kept the design simple.

There are 10 classes to represent a variety of things such as proteins, peptide, results input parameters etc.

It fully supports all the current search search types and reports including Unigene and error tolerant searches

One of the key dewsign goals was that the same user code could be used for different types of reports - for example, if you want to group est results using a unigene index, then this can be done by simply passing the name of the unigned index file as a parameter - no other changes to your code are required.

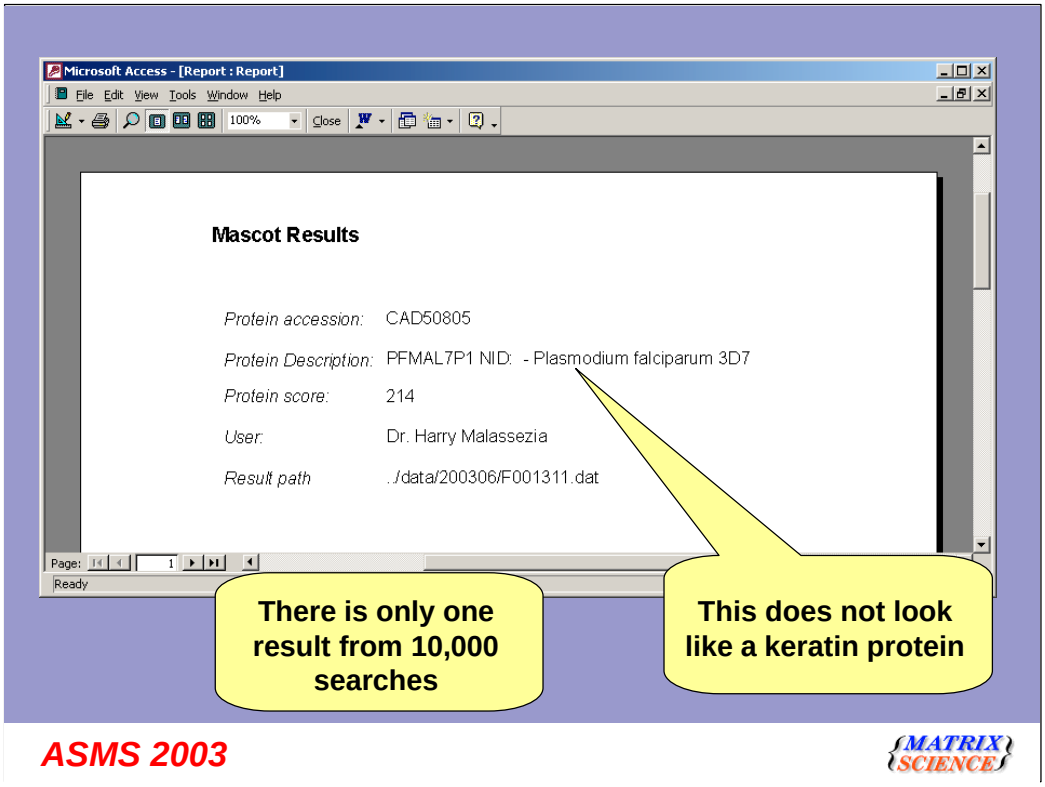
Programmer writes the code...

- Janet's system has Microsoft Office, so she decides to use Microsoft Access and progam in 'C++'
- Installs her copy of a 'C++' compiler and Mascot Parser for Windows
- The following morning, she has something ready to show...

So let's get back to our case study programmer Janet.

Although she's not a great fan of Microsoft Access, she has this on her system and since she is familiar with C++ she decides to use that.

Janet is one of those rare breed of highly productive programmers, and by the following morning (well actually it's early afternoon - most programmers seem to work best at night) she has something ready to show.



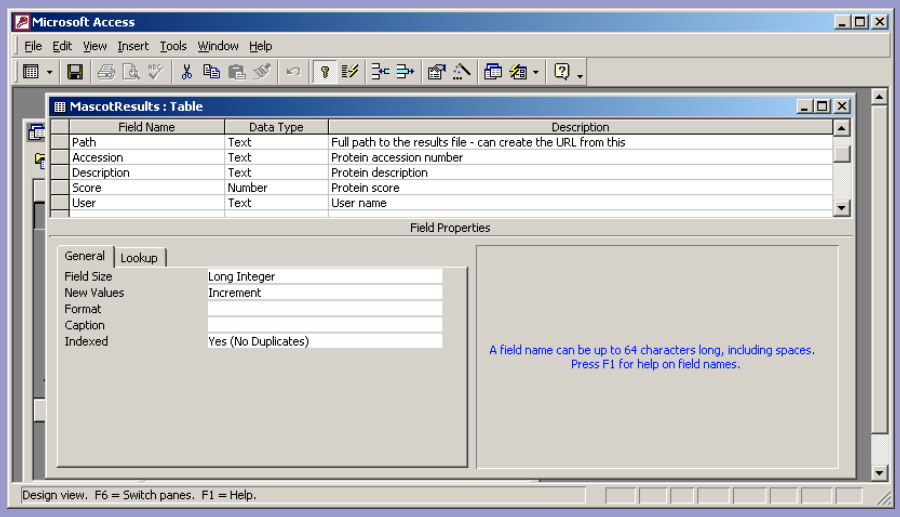
This is the Microsoft Access report that she shows to Harry. She explains that the link back to the results file doesn't quite work yet, and mutters something to Harry about embedded something or other. However, she shows him how to take what is displayed and copy it into a URL to display the results, and since Harry is not a demanding sort of person he is happy with this temporary solution.

Harry then looks more closely at the result on the screen and says that this is not a keratin protein - in fact Plasmodium falciparam doesn't have any skin. However, he is not disheartened and says "Let's look at some other results and see if they are keratin".

No, that is it - there is just one result... Janet says smiling. She fails to understand why Harry is not pleased with this

Now, it is at this point that we at Matrix Science sometimes get a call or an email for help. So, lets see what Janet did, and maybe see if we can help her out.

Database 'design'



ASMS 2003

MATRIX SCIENCE

Firstly, she designed the database. Design is rather a strong word to use here. All she has is a single table with the name of the results file, the accession number for the protein, the description of the protein, the score and the user name. She appears to have done exactly what harry has asked here.

She next sets up an ODBC provider to link to the database - it's only a few mouse clicks, so I won't bore you with that.

Now, you need to take a deep breath, because the next two slides contain some code. I promise that there are only two slides and it will all be over in less than 4 minutes

```
int main(int argc, char * argv[])
{
  if (argc == 2) {
    ms_mascotresfile resfile(argv[1]);

    if (resfile.isValid()) {
      ms_mascotresults * pepsum = new ms_peptidesummary(resfile);
      ms_protein * protein = pepsum->getHit(1);
      if (anyKeratinInProtein(pepsum, protein)) {
        TRY {
          CDatabase db;
          db.Open("MascotResults");
          std::ostringstream s;
          s << "INSERT INTO MascotResults(Path, Accession,Description,Score,User)"
            <<" VALUES('" << argv[1] << "','"
            << protein->getAccession() << "','"
            << pepsum->getProteinDescription(protein->getAccession().c_str())<<"','"
            << protein->getScore() << "','"
            << resfile.params().getUSERNAME() << "')";
          db.ExecuteSQL(s.str().c_str());
        }
        CATCH(CDBException, e) {
          std::cout << "Error: " << (LPCTSTR)e->m_strError << std::endl;
        }
        END_CATCH
      }
    }
  }
}
```

ASMS 2003

MATRIX SCIENCE

This is almost the complete code that Janet wrote. I hope you can read it - lets look at what it does.

The first couple of lines are mumb jumbo C, and take the parameter that you pass to the program - this is a results file name. She then creates a mascot results file object and then a peptide summary. See it's quite easy really.

She then gets the top scoring protein from the peptide summary. Well, how was she to know that Harry wanted her to look at all significant scoring proteins. Clearly she just has to add a little loop here.

The next line is a call to "anyKeratinsInThisProtein" - this is some code that Janet has written - we will look at that later.

All of the rest of the code is making a SQL command and putting it into the database.

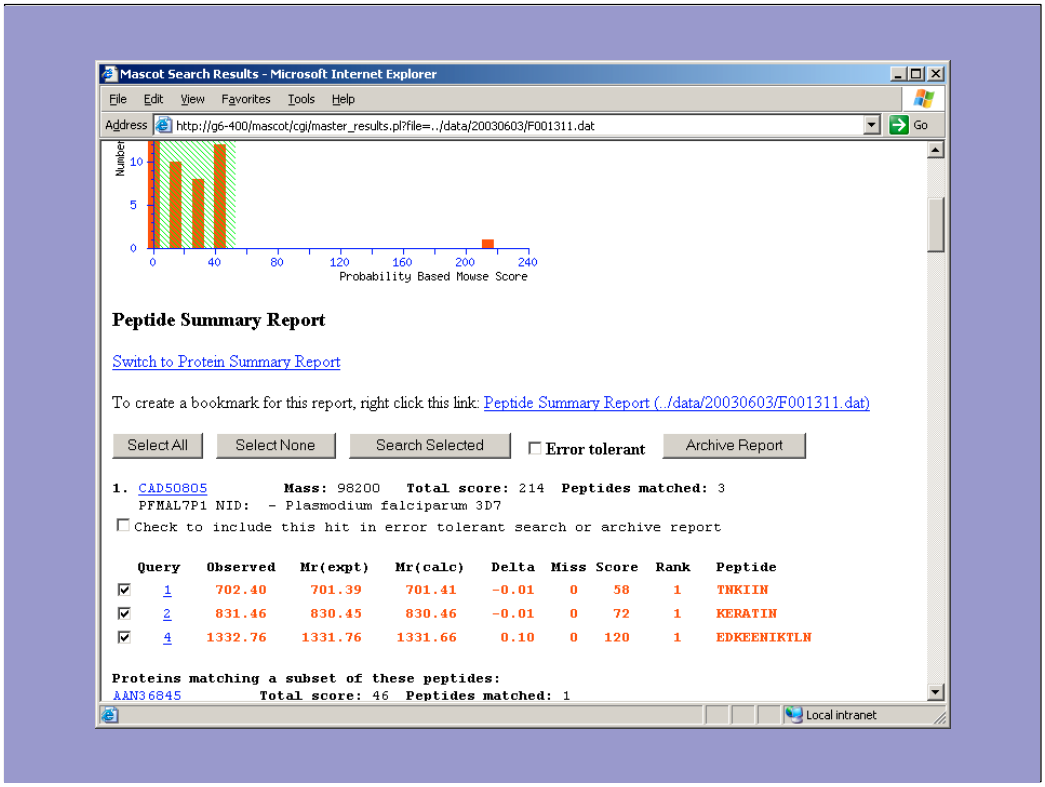
So, we've not seen anything really wrong so far. Let's have a look at the anyKeratinsInThisProtein function.

```
bool anyKeratinInProtein(ms_mascotresults * pepsum,
                         ms_protein * protein)
{
  bool found = false;
  if (protein) {
    for (int i=1; i <= protein->getNumPeptides(); i++) {
      ms_peptide * pep;
      if (pepsum->getPeptide(protein->getPeptideQuery(i),
                             protein->getPeptideP(i), pep)) {
        if (pep->getPeptideStr() == "KERATIN") {
          found = true;
        }
      }
    }
  }
  return found;
}
```


Here we have a loop that is checking each peptide in the protein.

This function returns the amino acid sequence from the peptide. Oh, and she appears to be looking for the text “KERATIN”…

I suspect that Harry didn’t want her to do this.



If we go back and look at the one result that she has, yes there is a KERATIN peptide.

Back at Matrix Science, we have a little laugh, and suggest that she shows this result to Harry and ask him to explain in a little more depth what he really wants.

However. I would not describe this as a bad experience. It only took one day for Janet to do this. She now understands a little more about what needs doing. Harry realises that he needs to explain in a little more detail.

In the swing example, the end user would have to wait a month to get what is required. You won't know what you need to data mine until you start. So, let's give Janet some advice.

Case study - fixing the problem

- Remove the test for 'Keratin'
- Put all the data in the database - including the peptides for each protein hit
- Create a query in Access - you don't know what you might need to query in future

ASMS 2003

MATRIX SCIENCE

Firstly, she must remove the test for keratin. Harry may only want keratin proteins today - tomorrow I suspect he may want other proteins...so, put all the results in the database - possibly leaving out junk hits.

She could then write an access query to search for the text keratin in all the description lines.

Support and licensing

- Support is provided by email or telephone
- A one-off fee per developer
- Updates and support will be provided as long as there is a Mascot support contract in place
- Free to use the application that you develop anywhere within your organisation

ASMS 2003

MATRIX SCIENCE

One thing that has come up in this case study is that Janet needed to get support from Matrix Science.

So, support is primarily provided by email.

The keratin problem was easy to find having seen the source code.

So it is a one off fee per development toolkit.

Updates and support will be provided as long as there is a mascot support contract in place.

And you are free to use the application that you develop anywhere in your organisation.

In summary

- it is a programming tools to speed up development, and is of no use to someone who cannot program

- it is available for all platforms supported by Mascot

- it can be used with C++, Perl and Java

- it is safer to use Mascot Parser than write your own code - support for new report styles as they become available.

Thank-you for your attention, and please wake up anybody next to you who chose to sleep.