

Adventures in API Land
Perforce User Conference 2001

Robert Cowham
Perforce Consulting Partner
Vaccaperna Systems Ltd
robert@vaccaperna.co.uk

Contents

- ➡ P4Word Origins
- ➡ Structure of the API
- ➡ Error Handling
- ➡ More advanced usage
- ➡ The Future
- ➡ Conclusion

P4Word Origins

- ☞ Why bother?
- ☞ Approaches considered
 - SCC DLL vs. API
- ☞ Chosen solution: a COM DLL

What is the API?

- ➡ As used by p4, p4win, etc.
- ➡ Tarball on Supporting Programs Page
- ➡ C++ source and header files
- ➡ Compiled libraries per platform

High Level Structure

Class ClientAPI

- Init
- SetArgv
- Run
- Final

Class ClientUser

- Inherit from and override

Top Level Procedure

```
VBEXT_API int STDCALL p4interface(int argc, char *argv[], Boolean
    UseStat)
{
    Error e;
    ClientUserDLL ui;
    ClientApi client;

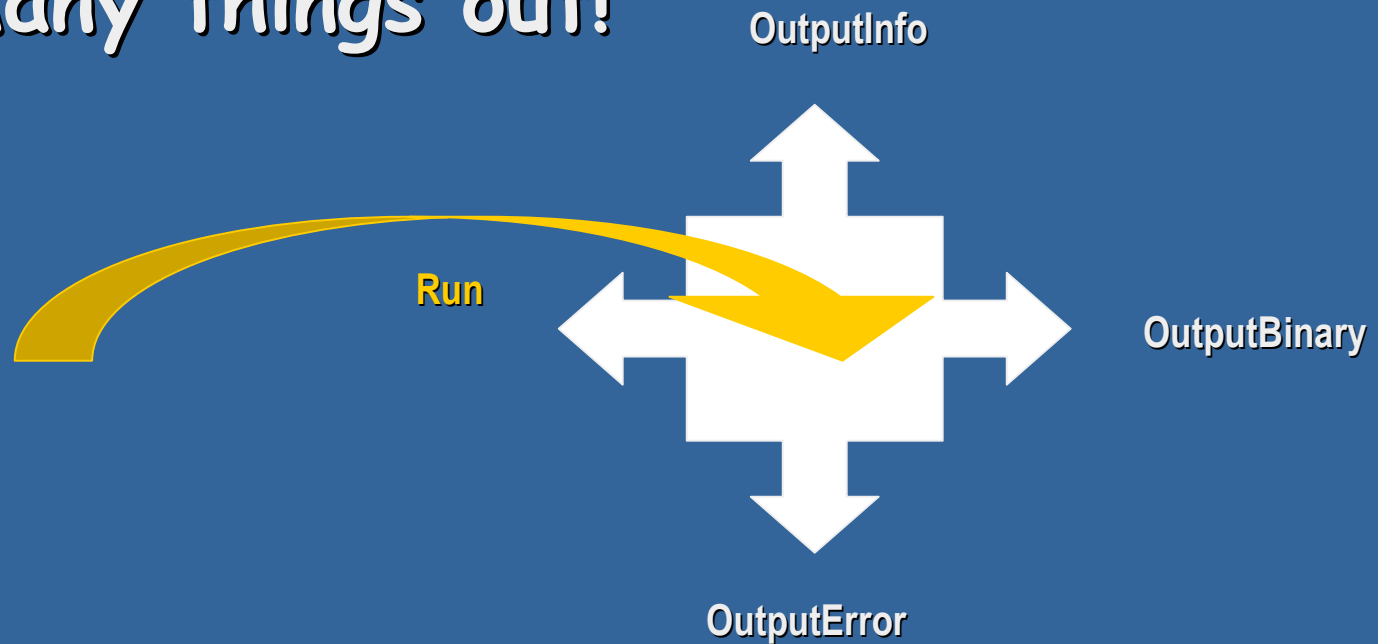
    if (UseStat) client.SetProtocol("tag", "");

    client.Init( &e );
    client.SetArgv( argc - 1, &argv[1] );
    client.Run( argv[0], &ui );

    return client.Final( &e );
}
```

Getting Results

- ➡ One thing in
- ➡ Many things out!



ClientUser Details

- ➡ Finished
- ➡ HandleError
- ➡ **OutputBinary**
- ➡ **OutputError**
- ➡ **OutputInfo**
- ➡ **OutputText**
- ➡ File
- ➡ InputData
- ➡ OutputStat
- ➡ Diff
- ➡ Edit
- ➡ **ErrorPause**
- ➡ Prompt
- ➡ Help
- ➡ Merge

ClientUser Implementation

```
class ClientUserDLL : public ClientUser {
public:
    void ErrorPause( char *errBuf, Error *e )
    {   StrBuf buf;
        OutputError( errBuf );
        MessageBox (NULL, errBuf, "p4vb", MB_OK); }
    void OutputInfo(char level, char *data)
    {   AddToArray(data, strlen(data), m_InfoArray);   }
    void OutputError(char *errBuf)
    {   AddToArray(errBuf, strlen(errBuf), m_ErrorArray);   }
    void OutputText(char *data, int len)
    {   OutputData(&m_TextFile, &m_TextFileName, FST_TEXT,
        data, len);   }
    void OutputBinary(char *data, int len)
    {   OutputData(&m_BinaryFile, &m_BinaryFileName,
        FST_BINARY, data, len);   }
}; // ClientUserDLL
```

Debugging Output

☞ C:\>p4 -s info
☞ info: User name: robert
☞ info: Client name: bruno_ws
☞ info: Client host: cowhamr-laptop
☞ ...
☞ exit: 0

☞ C:\>p4 -s files //depot/main/jam/jam.c
☞ info: //depot/main/jam/jam.c#36 - edit change 5051
(text)
☞ exit: 0

☞ C:\>p4 -s files //depot/main/jam/fred
☞ error: //depot/main/jam/fred - no such file(s).
☞ exit: 0

The VBA Interface

```
Long   p4run([in] BSTR cmd,  
            [out] ArrayString infoArray,  
            [out] ArrayString errorArray,  
            [out] BSTR * TextFileName,  
            [out] BSTR * BinaryFileName);
```

VBA example

```
Dim p4cmd As String
Dim result As Long
Dim InfoArr() As String
Dim ErrorArr() As String
Dim TextFileName As String
Dim BinaryFileName As String

TextFileName = "temp"      ` Need to initialise
BinaryFileName = "temp"

p4cmd = "print " & Chr(34) & DocPathname & "#" & ver &
Chr(34)

result = p4run(p4cmd, InfoArr, ErrorArr,
               TextFileName, BinaryFileName)
```

Error Handling

- ☞ Not always easy!
- ☞ Can require result parsing

Result Parsing

```
C:\bruno_ws\main\jam>p4 edit jam.c  
//depot/main/jam/jam.c#35 - opened for edit
```

```
C:\bruno_ws\main\jam>p4 edit jambase.c  
//depot/main/jam/jambase.c#33 - opened for edit  
... //depot/main/jam/jambase.c - also opened by mls@mls  
... //depot/main/jam/jambase.c - also opened by  
    mls@mls_oolong_nt
```

You can also get results like:

```
C:\bruno_ws\main\jam>p4 edit jam.c  
//depot/main/jam/jam.c#35 - currently opened for edit
```

Using Fstat

A file not opened for editing:

```
C:\bruno_ws\main\jam>p4 fstat jam.h
... depotFile //depot/main/jam/jam.h
... clientFile c:\bruno_ws\main\jam\jam.h
... headAction edit
... headType text
... headTime 879793859
... headRev 49
... headChange 4726
... haveRev 49
```

Using Fstat

A file which is opened for editing:

```
C:\bruno_ws\main\jam>p4 fstat jam.c
... depotFile //depot/main/jam/jam.c
... clientFile c:\bruno_ws\main\jam\jam.c
... headAction edit
... headType text
... headTime 879465964
... headRev 35
... headChange 4701
... haveRev 35
... action edit
... change default
```

Using Fstat

```
C:\bruno_ws\main\jam>p4 fstat jambase.c
... depotFile //depot/main/jam/jambase.c
... clientFile c:\bruno_ws\main\jam\jambase.c
... headAction edit
... headType text
... headTime 882925592
... headRev 33
... headChange 5028
... haveRev 33
... action edit
... change default
... .. otherOpen0 mls@mls
... .. otherAction0 edit
... .. otherOpen 1
```

The Word Side

- ➡ Add menus
- ➡ VBA functions in a template
- ➡ What functions required?
- ➡ Use of P4CONFIG
- ➡ Diffing - the “killer application”
- ➡ Demo...

Word Challenges

Installation

- References
- Template location

Differences between versions

Excel and Powerpoint

- Do they do things the same way...??

More Advanced API Usage

```
ClientApi.SetProtocol( "specstring", "" )
Void P4CmdSubmit::OutputStat(StrDict *results) {
    StrPtr *data = results->GetVar( "data" );
    StrPtr *spec = results->GetVar( "specdef" );
    SpecDataTable    specData;

    if ( ! ( spec && data) ) return;
    Error e;
    Spec s( spec->Text(), "" );
    s.Parse( data->Text(), &specData, &e );

    if( e.Test() ) { HandleError( &e );
                    return; }
    specData.Dict() ->ReplaceVar("Description",
                                changeDesc.Text() );
    s.Format( &specData, &specText );
}
```

Completing Submit

- ➡ Supply member variable (specText) to the "submit -i":

```
void  
P4CmdSubmit::InputData( StrBuf *strbuf,  
    Error *e )  
{  
    strbuf->Set( specText.Text() );  
}
```

- ➡ No temporary files

The Future

- ➡ Enhanced functionality
- ➡ P4Excel and P4PowerPoint
- ➡ Licensed to Perforce

Conclusion

- ➡ API is powerful
- ➡ Watch out for error handling...
- ➡ Perl/Python bindings
- ➡ Have a go!

Questions?