



INTELLIGENT TEXT PROCESSING

ITP

From OfficeVision/400 to ITP

2002-05-15

How to migrate OfficeVision/400 text-data merge applications to ITP.



Table of contents

- 1. Introduction 2
- 2. Introduction to ITP/CS 3
 - 2.1 The architecture of ITP/CS 3
 - 2.2 Database abstraction..... 4
 - 2.3 Model documents, models and result documents..... 4
 - 2.4 The ITP document development process 4
 - 2.5 Producing result documents..... 5
 - 2.6 Printing ITP output..... 5
- 3. The ITP Instruction language 6
- 4. ITP/CS vs. OfficeVision/400 11
 - 4.1 A comparison chart..... 11
- 5. From OV/400 to ITP 15
 - 5.1 Migrating existing OV/400 (output) documents..... 15
 - 5.1.1 ITP/O₂O..... 15
 - 5.2 Migrating OV/400 Merge documents..... 16
 - 5.2.1 Plain migration (ITP/OMA) 16
 - 5.2.2 Reimplementation 17
 - 5.3 OV/400 API Replacement..... 18
 - 5.3.1 WindowsEnabler/400 19
 - 5.3.2 ITP/Document Services..... 19
 - 5.3.3 ITP OfficeVision/400 Integration library..... 19



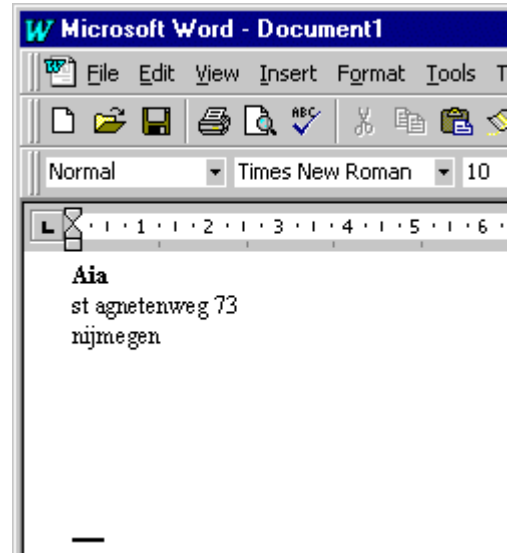
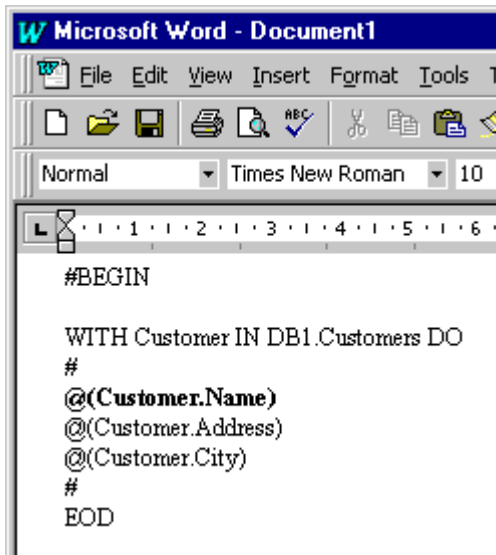
1. Introduction

This white paper describes the ITP/CS product and the process to go through when migrating text/data merge applications from OfficeVision/400 to ITP/CS.

Now that the days of OfficeVision/400 are numbered, people using the text/data merge facilities of OV/400 are wondering: what is my alternative? The solution to this problem is ITP/CS. ITP has been around since 1990 and was developed as a more flexible and high-performance alternative to OfficeVision/400 text/data merge. In 1994 Aia introduced ITP/CS, the client/server successor to ITP.

ITP/CS is a client/server software solution that offers unprecedented flexibility in creating text/data merge applications for modern word processors, like Microsoft Word and Corel WordPerfect. ITP/CS connects these word processors to your AS/400 databases. It does so by adding a flexible and easy-to-use instruction language to your word processor of choice.

White paper





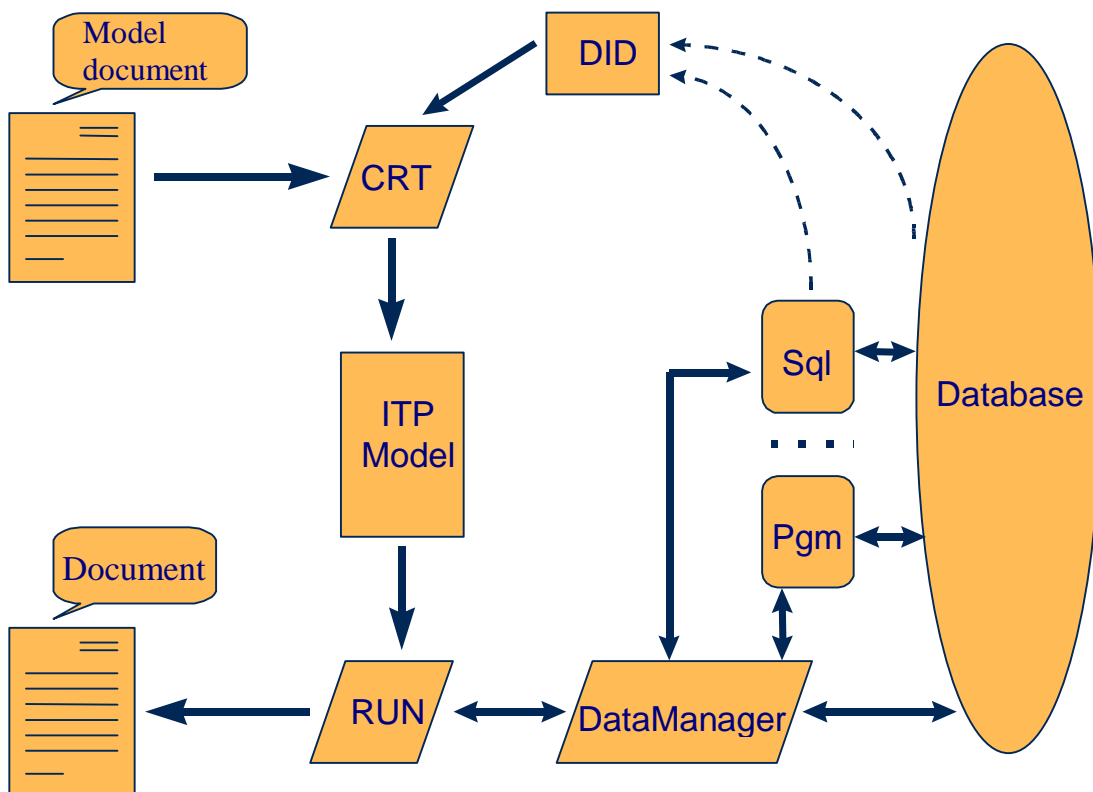
2. Introduction to ITP/CS

With ITP the model documents, the merge documents, are developed in the word processor of your choice. After development of a model document, ITP creates the final output document(s) based on this model document. The word processor itself is not used during this creation process.

ITP combines all familiar word processor facilities, both textual as well as layout, in a seamless way with data from your corporate databases. Text and layout in the documents can be conditioned based on the data from the database. ITP accesses the database in real-time, so no downloading, pre-processing or copying of data is required.

Document production can be document driven, where the output is based on conditions and calculations coded in your document, that may or may not use database access. It can also be data driven, where the output is based on the contents of the database. If necessary, the document production or parts of it can be controlled by user interaction.

2.1 The architecture of ITP/CS



This figure describes all basic ITP processes:

1. A description of the database is created, the DID. The DID describes how to access the database using for example SQL queries or external data retrieval programs.
2. Model documents are developed in the word processor of your choice. These documents contain ITP instructions with references to the ITP database description. Model documents are compiled to ITP Models.



3. ITP Models are invoked to produce the actual result documents. During this process data are accessed using the SQL queries, programs etc. which were described in the DID. This step is the actual production process.

2.2 Database abstraction

ITP uses its own database descriptions for database access. In ITP language these are called DIDs: **Database Interface Definition**. These database descriptions can contain multiple definitions of sets of related data (e.g. a database record) and the information required to retrieve this data. Relations between these definitions can be defined as well.

Through these definitions ITP models can access data without any knowledge of where the data resides or how to access it.

The DID is in fact an abstract view on a, possibly imaginary, database that maps to database files, external data retrieval programs or SQL queries. This mapping can be one-on-one, meaning that the abstract view closely mimics the original database, but it can also be completely different, offering a much more abstract (high-level) view of the database, which simplifies the development of model documents, especially when dealing with highly normalised databases.

By mapping the abstract view on the database to external programs, ITP enables the integration of self-written programs into the data retrieval, and therefore document production, logically. It furthermore allows developers to fine-tune the data retrieval process for special purposes, such as highly controlled data retrieval, mixed environment/dbms data retrieval or high-performance data retrieval.

The fase of developing and implementing this abstract view is separated from the creation and execution of models, the document development and usage fase, enabling organisations to limit the data that are accessible from documents. This abstract view can also be modular, so that the DID or parts of the DID can be reused by other collections of documents.

The program which is used for this part of the ITP development process is called the ITP SDK.

2.3 Model documents, models and result documents

ITP works with three kinds of files in the text/data merge process:

- **Model documents:** These are word processor documents that can contain all kinds of text with all layout facilities offered by the word processor, mixed with ITP instructions. The ITP instructions enable data retrieval using external retrieval programs or logic, manipulation of data and merging those data into the document text.
A model document is the equivalent of a program source.
- **ITP Models:** These are binary files generated from model documents, containing an optimised form of the model document, including all text, layout and ITP instructions. Model documents are compiled to ITP models. The model is the equivalent of a program.
- **Result documents:** These are normal word processor documents that contain the final document text, merged with -possibly manipulated- data. These documents are produced by executing ITP models.

2.4 The ITP document development process

Developing a document in ITP means writing model documents (in your favourite word processor), compiling them to ITP models and testing them.

ITP program logic is written in the model document using the ITP instruction language, a powerful yet easy to use language, which has two very special properties, concerning input/output:



ITP – From OfficeVision/400 to ITP

- Input is always done through a highly abstract and structured interface to the database or to external programs. The actual database access is not programmed in the model itself. Furthermore some user interaction can take place for additional input. All other input for the result document is already present in the model document itself! Other means of input are not (and do not have to be) available.
- Output is the result document itself, which means that all output produced is text and layout for the result document(s). Both may of course depend on database content. No special output facilities are needed.

The ITP instruction language furthermore offers all the usual programming facilities: functions, if-then clauses, while-loops, variables, arrays etc. It also offers powerful calculation, data and text manipulation and conversion functions. A model can be designed to produce one or more result documents.

The ITP instruction language is in fact a kind of programming language on word processor document level. Contrary to a normal programming language it is very easy to use (even by trained end-users), because of the special ways in which is dealt with input and output and because of the focus on text and layout in the word processor's native environment.

ITP supports model documents in several word processor formats: Word For Windows 6.0/95/97/2000/XP, WordPerfect 6.1 and up and Ami Pro 3.0/3.1. Lotus WordPro is support through the Ami Pro document format. Support is also available for HTML documents and ASCII documents.

2.5 Producing result documents

Result documents are generated by executing the corresponding ITP models; this is called the runmdl process. This process can be initiated from the ITP user interface or it can be integrated in an application, using ITP's runmdl API.

While running the model users can be asked to make (database) selections or to provide additional input. This is the case with interactive models. Models can also be designed to run without any intervention by the user.

During one run of the model, one or more result documents can be produced.

The runmdl process is such a highly optimised process, that performance is very good, limited by how fast data can be retrieved.

The result document is in the same word processor format as the original model document.

2.6 Printing ITP output

ITP produces standard word processor documents. These documents are printed using the word processor itself as the print engine, thus allowing the total layout flexibility of modern word processors for your automated output production.

The word processor software uses the operating system's printing architecture to print documents. ITP can therefore be used with all printer-drivers that are available for the operating system. It can - for example- produce Postscript, PCL5, AFP and PDF output, or any other format generated by a Windows printer driver.

This open architecture also enables easy integration of ITP with -for example- fax solutions, web publishing solutions, archiving software and print servers.



3. The ITP Instruction language

The ITP instruction language can roughly be divided into three parts:

1. instructions for the creation of text
2. instructions for data retrieval
3. instructions for data manipulation

To distinguish between text that has to appear in the result document and ITP instructions, the hash symbol is used as a “tumble switch”; each time a hash symbol is encountered, ITP switches from “instruction-mode” to “text-mode” or vice versa. A model document will always start in text-mode, so before the first instruction a hash symbol has to be placed.

A very simple model document could look like this:

```
#  
BEGIN  
  
#  
This is a very simple model document.  
#  
  
END  
#
```

The BEGIN and END instruction indicate the begin and end of the ITP instruction sequence.

This would result in a document containing the following text:

```
This is a very simple model document.
```

Since no data is being merged, this is of course not a very useful document.

If data has to be merged into the text, this can be done using the @-construct:



ITP – From OfficeVision/400 to ITP

```
...  
#  
Dear Ms. or Mr. @(Cust.Surname),  
  
In reference to you writing...  
#  
...
```

In the above example the field *Surname* from the record *Cust* will be merged into the document. But where does this record come from?

Data retrieval is achieved through the use of Entries, abstractions of the database which are defined in the DID. In the following example the entry *Customer* will retrieve data from the database. The fields that are retrieved are grouped in a 'record' which will be referred to as *Cust* and which is available between the ITP instructions DO and OD. The fields that are contained in the record have been defined in the entry definition in the DID. Since the method to actually retrieve the data has also been defined in the DID, no more specifications are needed in the model document.

```
...  
WITH Cust IN EXP.Customer DO  
#  
Dear Ms. or Mr. @(Cust.Surname),  
  
In reference to your writing...  
#  
OD  
...
```

The WITH construct will retrieve one record from the database. If multiple records should be retrieved you can use the FORALL construct. The DO ... OD part of the FORALL construct will be executed for each record that has been retrieved.

Again, the method to retrieve the data has been defined in the DID.



ITP – From OfficeVision/400 to ITP

White paper

```
...
WITH Cust IN EXP.Customer DO
#
Dear Ms. or Mr. @(Cust.Surname),

In reference to your writing....

A list of the ordered articles follows:
#
  FORALL Art IN Cust.Ordered_articles DO
#
  @(Art.Number_of_articles) @(Art.Article_description)
#
  OD (* FORALL Art IN Cust.Ordered_articles *)

OD (* WITH Cust IN EXP.Customer *)
...
```

The relation between *Customer* and *Ordered_articles* has been defined in the DID. So, the developer of model documents only need to know that these relations exist, not how they are implemented.

To clarify ITP coding in the model document, comments can be added. Comment starts with (*) and end with *). Most word processors can also aid in the creation of easily readable documents by using different styles, colours, etc.

Of course the contents of a document should differ based on -for example- whether the customer is male or female.

```
...
Dear #
  IF Cust.Sex = "M" THEN # Mr. #
  ELIF Cust.Sex = "F" THEN # Ms. #
  ELSE # Ms. or Mr. #
  FI
# @(Cust.Surname),

In reference to your writing...
```

Although this gets the job done, it is not quite obvious what text will be produced in the result document. Introducing variables will make this somewhat easier.



```
...  
TEXT ms_mr  
  
IF Cust.Sex = "M" THEN  
  ASSIGN ms_mr := "Mr."  
ELIF Cust.Sex = "F" THEN  
  ASSIGN ms_mr := "Ms."  
ELSE  
  ASSIGN ms_mr := "Ms. or Mr."  
FI  
#  
Dear @(ms_mr) @(Cust.Surname),  
  
In reference to your writing...
```

In this example a variable of type TEXT is declared. Then a value is assigned to this variable based on the contents of the field. Finally the contents of the variable are merged into the text using the @-construct. As a result, by separating the logic as much as possible from the text, the structure of the result text can be more *wysiwig*.

More then 60 build-in functions are available to format, convert or otherwise manipulate both numerical and text values.

```
...  
#  
@(Cust.Name)  
@(Cust.Street) @(Cust.Housenumber)  
@(uppercases(Cust.City))  
  
Nijmegen, @(date(today))  
  
...  
#  
...
```

Might result in:

```
Aia Software b.v.  
Kerkenbos 10 - 129  
NIJMEGEN  
  
Nijmegen, 12 April 2001  
  
...
```



ITP – From OfficeVision/400 to ITP

In the above example three built-in functions are used: *uppercases*, *date* and *today*.

Uppercases will convert a text into the same text, but -as the name suggests- all into uppercases.

Today has no parameters and will result in a numerical representation of the current date. In the above example this is 20010412.

The function *date* will print a date with the name of the month “in words”. Of course the result of the date function depends on the language being used, something that can be changed while producing the text. This is especially useful for internationally operating companies.

It would go beyond the scope of this short introduction to show examples of all the other facilities the ITP instruction language offers.

White paper



4. ITP/CS vs. OfficeVision/400

ITP/CS is not only a mere substitute for OV/400 text/data merge, but it also offers a lot of additional features:

- Access to multiple databases from one document
- Support for modern word processors
- A complete, though easy to use, instruction language
- Support for modern publishing media, like HTML and PDF

Owing to the support for modern word processors you can do many layout-related things with ITP that are impossible or only very hard to do with OV/400.

In the next section you will find a comparison chart with the most important OfficeVision/400 features compared to ITP/CS.

4.1 A comparison chart

OV/400	ITP
Multiple Letter merge: without record/item selection	ITP allows you to process a main file <ul style="list-style-type: none"> • completely • based on user selection of a particular item/record (no need to create a separate database file)
Column List: only one line can be repeated	nested data processing spans from words within a line to several pages if needed.
Difference between Multiple Letter Merge-data and Column list data	Input allowed from as many files as you want. All input data is handled the same way.



OV/400	ITP
<p>Conditional text</p> <p>.bct (expression) -></p> <p>.ect -></p> <p>maximum nesting level: seven .bct statements</p> <p>Data fields from Column list are not allowed</p> <p>Conditionals are difficult to read</p>	<p>IF-statement.</p> <p>IF (expression) THEN ELSIF (expression) THEN ELSE ... FI</p> <p>NO maximum nesting level</p> <p>All data available can be used within the expression</p> <p>IF statements are very readable</p>
<p>Expressions are simple</p>	<p>Expressions can be combined using (.), AND, OR and the NOT-operator.</p>
<p>Variables and set-instructions</p> <p>.set var, expression</p> <p>Data fields from Column list are not allowed</p>	<p>Variables, arrays (multiple variables sharing the same name)</p> <p>ASSIGN var := expression ASSIGN array_name[index] := expression</p> <p>all data available can be used within the expression</p>
<p>Numeric values Alphanumeric values</p>	<p>Numeric (called NUMBER) Alphanumeric (called TEXT) true, false (called BOOL)</p>
<p>F5, F9, F6 format data options</p>	<p>built-in functions to format data, convert data, manipulate data (like substring, ..)</p> <p>Built-in feature to add functions on document level.</p>
<p>Output is not completely revisable</p>	<p>Output is a document in either RFTDCA, Word (6.0 and up), WordPerfect (5.1 and up) or Ami Pro (WordPro) format. Section and document protection may be used to prevent user from editing (portions of) the document.</p>



ITP – From OfficeVision/400 to ITP

White paper

OV/400

ITP

OV/400 .inc structure	__INC(docname)
.INC instruction only used with fully qualified folder names	you can chose to use <ul style="list-style-type: none"> • a fully qualified directory name • a relative path • only the document name
(you can not copy an entire tree of documents)	(you can setup different environments)
Errors in the Shell Document-coding are only found at merge-time	Errors are found during a separate phase: create ITP-model. ITP-models, a binary file format, are a more efficient way to store the coded document.
No counterpart in OV/400	All kind of additional clauses, like WHILE DO..OD, FOR counter DO.. OD etc.
Runs only on the AS/400	Runs on a Windows or OS/2 PC, connected to the AS/400 via SNA or TCP/IP.
Uses normal user-authority on the AS/400 data objects	Uses normal user-authority on the AS/400 data objects
Prints to AS/400 configured printers	Prints to any printer visable in the LAN/WAN. This includes PCL, Postscript and AFP printers. Printing to queues as well as to files is possible. Printing to PDF format is another way to publish the result documents.
Coding of the instructions is done using the OV/400 editor	Coding of the instructions is done using <ul style="list-style-type: none"> • OV/400 (ITP currently supports OV/400) • Word (6.0 and up) • WordPerfect (6 and up) • Ami Pro • WordPro • Plain ascii, like notepad
All OV/400 features may be used.	All word processor features are 'part' of the instruction language, like creating tables, shading, font switches etc, etc.
Instructions are only allowed in the body of the text	Instructions are allowed in the header, footer and body of the text. This includes both ITP instructions (like ASSIGN) as wel as printing in the specific document-part.



ITP – From OfficeVision/400 to ITP

OV/400

Merge process is started either from OV/400 or from commands.

ITP

The merge process starts either from an AS/400 5250 session, an AS/400 batch job, a PC-application or the desktop.

White paper



5. From OV/400 to ITP

5.1 Migrating existing OV/400 (output) documents

OV/400 (merge) documents can be opened and manipulated in most Windows word processors. The word processor converts the document and you can save it in the native word processor format (e.g. Word 97 or WordPerfect 8). The quality of the resulting document depends on the quality of the converter used. Aia Software B.V. supplies a more advanced and flexible conversion software package, called ITP/Office to Office or ITP/O₂O.

5.1.1 ITP/O₂O

ITP/O₂O is a batch converter that converts IBM OfficeVision/400 RFT-DCA (OV/400) documents to Microsoft Word 97/2000/XP (Word) documents. ITP/O₂O also provides a basic conversion of OV/400 data/text merge instructions to ITP.

When you convert your documents using Word's standard RFT-DCA convertor you will soon run into problems:

- Conversion is often unreliable. All but the most trivial layout options require manual adjustment of every converted document or is even lost.
- Conversion is a very tedious process. You will have to convert each document individually.
- Merge instructions are not converted. They are simply disposed off in the conversion process. You will have to reimplement your data-text merge from scratch.
- All OV/400 document attributes are lost in the conversion process.

To attack these problems Aia Software developed a product called the ITP/Office to Office Convertor (ITP/O₂O).

ITP/O₂O is a Windows program, requiring Windows 98 or ME, NT 2000 or XP, for batch conversion of IBM OfficeVision/400 RFT-DCA documents to Microsoft Word 97/2000/XP format. ITP/O₂O is developed with two goals in mind: visual resemblance and maintainability of your documents. It offers unprecedented flexibility in the conversion process of your documents:

- Program features
 - With ITP/O₂O you can test and configure your conversion process on a number of representative documents and then convert large sets of documents at once. The number of documents that can be converted at once is only limited by your PC's system resources.
 - ITP/O₂O creates an extensive log. The log displays which features could not be converted. Use this to optimize your conversion settings.
- Basic layout conversion
 - ITP/O₂O converts all basic layout features, like bold, italic, underline and tab settings.
 - ITP/O₂O attaches a configurable Word template to converted documents.
 - ITP/O₂O maps OV/400 document attributes to Word document properties.
 - ITP/O₂O converts running headers.
 - ITP/O₂O converts cursor movements.
 - ITP/O₂O converts footnotes.



ITP – From OfficeVision/400 to ITP

- ❑ ITP/O₂O maps stopcodes to Word FILLIN form fields.
- ❑ ITP/O₂O supports custom page and margin settings for the first page.
- ❑ ITP/O₂O supports all four column types.
- ❑ ITP/O₂O converts .inc instructions to Word include fields or ITP __INC instructions.
- Configurable conversion features
 - ❑ In ITP/O₂O you can specify your own font mappings.
 - ❑ In ITP/O₂O you can specify a mapping for OV/400 paper bins to Word paper trays.
 - ❑ In ITP/O₂O you can specify a language mapping.
 - ❑ ITP/O₂O maps OV/400 symbols (from different codepages) to configurable Unicode symbols.
 - ❑ ITP/O₂O converts automatic numbering (.nl) to a configurable type of Word numbering.
- ITP support
 - ❑ ITP/O₂O maps data/text merge instructions to ITP instructions. This conversion assumes a DID fitting the document. This DID still has to be defined. See the next section for more information.

You can download a free demo version of ITP/O₂O after registration on our web site: www.aia-ntp.com/products/ntp/itpo2o.html or you can install ITP/O₂O from the ITP 2.1 CD. Through our website you can order the full version.

5.2 Migrating OV/400 Merge documents

There are two main options of migration OV/400 merge documents to ITP:

1. Plain migration
2. Reimplementation

In a plain migration, you use the ITP/Office Migration Assistant (ITP/OMA) to analyze your OV/400 merge documents and convert them to identical ITP model documents. When you do a reimplementation you redesign your documents and implement them again by hand as ITP model documents. The advantage in doing a reimplementation is that you can use the complete feature set of ITP. Since ITP is more versatile, you can create complex documents with a relatively small amount of work or documents that you were unable to produce with OV/400. With ITP, you can use arbitrary one to many relationships any which way you want. You are not limited to column lists. You can also nest relationships to an arbitrary level. In some applications, heavy use of queries is made in order to be able to produce the desired results. In ITP you can still do this, but it is no longer necessary and can in most cases be easily avoided.

The disadvantage of doing a reimplementation over plain migration is that it takes more time. How much depends on your current setup and the amount of ITP model documents that you need to build.

5.2.1 Plain migration (ITP/OMA)

When you want to perform a one-on-one migration from OV/400 merge documents to ITP model documents you can use ITP/OMA. ITP/OMA performs four steps:

1. Analyze the OV/400 merge document.
2. Create an ITP Database Interface Definition (DID) based on the results of the analysis.
3. Call ITP/O₂O to convert your OV/400 merge document to a Microsoft Word based ITP model document.



ITP – From OfficeVision/400 to ITP

4. Inserts the data definitions from the DID into the ITP model document.

After this step, you can compile the model document into a model and you are ready to create output documents.

Availability and more information

ITP/OMA is available free of charge. You can download ITP/OMA after registration on our web site: www.aia-ity.com. A complete manual describing the use of ITP/OMA is included with the software.

5.2.2 Reimplementation

ITP uses a Database Interface Definition (DID) to define its database access. Such a DID is based on the ITP Entry concept. In the DID you define the fields and the parameters (keys) for an Entry and you define how the data should be retrieved. You need the ITP/SDK AS/400 in order to define a DID. In most cases you can use built-in ITP data retrieval mechanisms, in other cases you must write your own programs to retrieve the data based on the parameters and to send the result fields to ITP. Users mostly write programs if they want to remove complexity from the ITP Model. This inherent complexity is then implemented in the data retrieval programs. When ITP should just retrieve fields from a record from a specific file, based on the keys of that file, you can use a built-in data retrieval mechanism. The advantage of this method is that you do not have to maintain a set of data retrieval programs.

Reimplementing your current OV/400 merge documents as ITP model documents is basically the same as starting with ITP from scratch. You start with analyzing the desired documents: which data from which files is needed in the documents. Based on this analysis you create a DID. This is a one-time process. Once you have a DID you can use it to create any number of model documents.

Migration database access

You can implement database access for OV/400 merge documents in several ways. The most common implementations are based on:

- direct database file access and
- QRY/400 queries.

Database files

When your OV/400 merge documents are based on direct database file access it is easy to create an Entry. The ITP/SDK AS/400 can read the file definition from a physical or logical file and automatically create an Entry for that file. If you use the built-in data retrieval mechanism you now just have to download the DID to the PC and are ready to define and run ITP Models based on the file for which you just created an entry.

Queries

When your OV/400 merge documents are based on QRY/400 queries you have two options:

1. Create an entry based on the query.
2. Switch to file/program based entries.

The first option, creating an entry based on the query has an obvious advantage:

- It takes little development time. A program that runs the query and sends back the result data to ITP is, with the help of the ITP/SDK AS/400, written in approximately half an hour.
- Reuse your existing queries.

This method also has some disadvantages:



ITP – From OfficeVision/400 to ITP

- You must write a specific query for every type of output-document or you can write a single query that retrieves the data for a lot of document-types. Using multiple queries means a lot of maintenance; using a single query means introducing a lot of unnecessary overhead.
- One-to-many relations that have to appear in your document are not easily expressed with the help of queries.

The second option, switching to file/program based entries, and thus building a more generalized ITP DID has the following advantages:

- You can easily define and use one-to-many relations in your documents;
- Every entry can be used in every ITP Model; you only have to define the data retrieval for a specific set of data once and you can (re)use it many times;
- You can offer the document developer a simplified view of the database; you can hide implementation details;
- You can reuse data retrieval and data manipulation programs that you have already written; you can use, for example, existing interfaces to a Customer database, instead of directly reading from the files;
- Once you have defined the Entries that you need in your documents, you can easily create new ITP Models based on these Entries. You do not have to create new queries or adjust existing queries.

Migrating documents

After creating the DID you can start creating your model documents. It is a good idea to use ITP/O2O to convert your current OV/400 merge documents to Word documents, since large portions (especially text paragraphs) will probably be reusable in your ITP model documents.

5.3 OV/400 API Replacement

OV/400 offers several APIs that you can use to integrate OV/400 into your AS/400 application, e.g.: EDTDOC, PRTDOC, DSPDOC, MRGDOC. When migrating from OV/400 to ITP you need replacements for these APIs. With respect to ITP, you can divide these APIs into two groups: APIs that create an output document and APIs that post-process an output document, e.g. edit or print. ITP output documents are regular word processor documents and ITP only creates those documents; it does not post-process these word processor documents itself. In order to work with output documents the word processor is used; there is no difference between documents created by ITP and documents created by hand in a word processor. Please note that ITP does not use the word processor while creating documents; the word processor does not even have to be installed on the PC. You only need the word processor when you want to process output documents.

The Run Model process that produces an output document is performed through a Windows program. ITP supplies Windows API's to start this program. When using the ITP API you must supply:

- the name and path of the ITP model to execute and
- the name and path of the output document to be created.

Optionally you can supply additional parameters that can be used within the ITP Model. This means that you can supply the key information needed for the contents of the document, e.g. an invoice-number or a policy number. The ITP model will then, based on this key, retrieve all necessary information.



ITP – From OfficeVision/400 to ITP

When you want to create an ITP output document you need to execute a Windows program in order to start the ITP Run Model process. It is hard to control Windows applications, such as word processor, from the AS/400 in order to perform the required operations on ITP output documents. ITP however offers several solutions for this problem.

5.3.1 WindowsEnabler/400

With WindowsEnabler/400 you can start arbitrary PC commands on a PC. Commands that are started from a terminal session are executed on the PC that is running the terminal emulator. This means that interactive users can create, open, edit and print Word documents on their PC controlled from an AS/400 application or program.

With WindowsEnabler/400 you can execute a single PC command line and wait for the completion and result of the PC program. You must implement queuing, logging and error handling yourselves on the AS/400. WindowsEnabler/400 has built-in commands for file handling, such as print, edit, copy and delete and built-in support for ITP.

5.3.2 ITP/Document Services

With ITP/Document Services you can submit requests to an ITP/Document Services server. ITP/Document Services is intended for standalone (batch) use. It offers all the non-interactive features of WindowsEnabler/400 and much more, such as built-in queuing, logging and error handling. ITP/Document Services also has built-in capabilities for e-mailing documents or uploading documents to a web server.

ITP/Document Services and WindowsEnabler/400 can be used as complementary products; where WindowsEnabler/400 is used to handle interactive requests to create, edit or print documents and ITP/Document Services is used to handle non-interactive (batch) requests to create and print documents.

5.3.3 ITP OfficeVision/400 Integration library

The ITP OfficeVision/400 Integration Library (ITPOVINT) is a library that contains a front-end for either WindowsEnabler/400 or ITP/Document Services. The library contains a set of commands and programs together with their sources that mimic the behavior of the OV/400 APIs as close as possible. You can choose to use either WindowsEnabler/400 or ITP/Document Services to actually execute the command.

ITPOVINT was specifically created to work in combination with documents migrated with ITP/OMA, but it can also be used without ITP/OMA.

Availability and more information

ITPOVINT is available free of charge. You can download ITPOVINT after registration on our web site: www.aia-ity.com. A complete manual describing the use of ITPOVINT is included with the software. You can also examine the source of the programs and use these sources to create your own programs.



ITP is developed by

For more information please contact us.

Telephone: +31 24 371 02 30
Fax: +31 24 371 02 31
WWW: <http://www.aia-ntp.com>
Email: info@aia-ntp.com
Postal Adres: POBox 38025
6503 AA Nijmegen
The Netherlands
Visiting Adres: Kerkenbos 10-129
6546 BJ Nijmegen
The Netherlands