ENHANCED ADL

IMPLEMENTATION PLAN

Zhigang Yin

CONTENTS

Introduction

The ADL program is a fundamental part of the Information Tools business model, my software implementation aims to develop better, more industry standard ways of doing the ITL setup task.

This could be accomplished by programmatically designing either some form of enhancing component for ADL or by re-writing the ADL completely.

Different options were explored, and applicability of each alternative is discussed in details in the document –

"ENHANCED ADL DESIGN-- DOCUMENT & USER GUIDE"

Through this implementation, I hope to improve not only the ADL program itself, but also the production task in which they process data and build databases for customer analysis. This means higher efficiency and quality.

Requirements

Following section is an outline of the requirements on both theoretical and practical level.
This is also discussed in detail in the
---*"ENHANCED ADL DESIGN-- DOCUMENT & USER GUIDE"*

Theoretical:

1) **Training**
2) **Speed**
3) **Correctness**
4) **Easy to maintain and extend to meet new challenges**
5) **Retain backwards and forwards compatibility**
6) **Extra features**

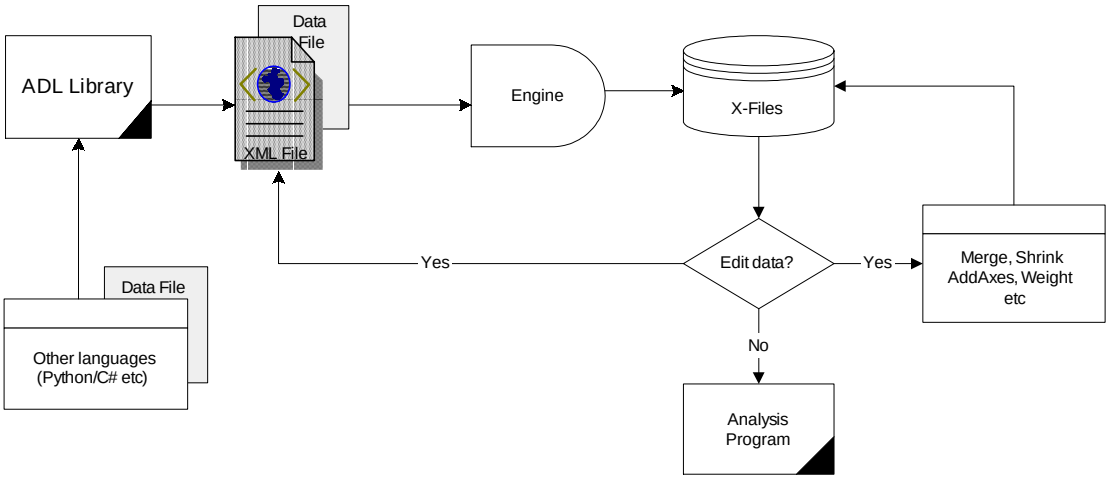
Practical:

1) **Error handling**
2) **Calculation**
3) **Security**
4) **Variable types support**
5) **Other features**

The software should be designed to address and solve as many of the problems listed above as possible.

Conceptual design models

Use another programming language



This implementation is extended by adding a GUI interface lies on top of the original model. Therefore, an end user should be able to easily generate ADL script files through the newly designed interface.

Time management

As this BTech project is a software improvement project, the main task is the software implementation (coding) part, also the part that consumes the largest proportion of time among all the parts, time management is important.

The total time allocated to this part is 160 hrs. I plan to divide them into several sections as follows:

Usage	Hours
Choose programming tool	5
Write test programmes	20
Design Main programme	20
Main programme functionality	100
Testing and debug (carried out throughout the entire coding process repeatedly)	15
Total	**160**

Testing:

At coding stage of the software, testing is done against simple valid data files. For example, the data files (data matrix) in use are normally over 100 * 100 in size, but for testing purpose, I would generate 1*1 → 10*10 matrices and use extreme values.

Once the functions are completed up to certain stage, real data files are to be used.

Training:

One of the current issues with ADL scripting language is “Training”. New staffs who have no programming background normally find ADL hard to learn as its syntax is flexible and the various functions available. Even for staffs with programming background, ADL is different enough to be treated and learned as an entirely new language.

The new programme aims to give users with basic “windows” operating skills access to produce ADL databases. The training process of this programme will involve

- Learning ADL language structure and features
- Familiarizing with the new programme functions and features

Forward / Backward adaption:

Forward:

- XML is used as the intermediate file format for storing the ADL files. The XML file should be designed carefully that it gives a clear and detailed representation of the “*.adl” file.
- XML is widely used format on the web and in many other applications. So the XML format can be easily converted to fit into other applications of to be published on the web. Even by looking at the plain XML file, user can get a good indication of the file structure.

Backward:

- By careful design, converters can be generated to convert original “*.adl” files to the XML format. Due to the original ADL syntax’s flexibility, this conversion process could well be carried out in other project.

Future Updates:

With the current design and implementation, the functionalities are still limited. Various extensions and alternations are possible. Stronger backward adoption is one of these.
Also, clients might have new requirements on their databases, or the source data file simply gets more complicated that the current design is not able to handle. Update to the application is then necessary.

Conclusion:

Once the application is completed, it would be installed and tested by users of different levels: new staff, experienced staff, software development group, etc. The result of the test use will give a fair comment about how the new application fit into the current data setup process.

The result might be **positive**, so the company is likely to carry out new projects to extend and alternate the current design, hence make this application one of the formal method of data setup process.

Or the result might be **negative**, so the application will be discarded. However, while other research projects on this topic are carried out in the future, the experience gained from this implementation will be helpful for the new project at some stage: time saving, avoidance of possible mistakes.

Reference:

- Information Tools

  --- “ADL V3 Manual” (latest version)
  --- “ADL Enhancement Proposal” by Grant Black
  --- *http://www.infotools.com*
  --- “Enhanced ADL Design --- Document & User Guide”

- “Creating A Great Design Document”
  ---http://www.geocities.com/SiliconValley/Bay/2535/design_doc.html

- *http://www.w3.org/WAI/EO/Drafts/impl/*

- *http://www.fenwicksoftware.com.au/implementation.asp*