The Progress Electronic Magazine

In this issue:

Did you sign up to receive this E-Zine? Send email to sauge@amduus.com to subscribe or fill out the forms at http://www.amduus.com/online/dev/ezine/EZineHome.html ! It's free! (Though donations are certainly welcome – whatever you feel is fair!)

Publisher's Statement:

Welcome to issue six of the Progress E-Zine. There is some good stuff in here for not only the programmer, but the manager.

First I will give you some news on the home front and that is I have gone full time on Amduus Information Works, Inc. Myself and a few other programmers are clamoring away on software for resale to the marketplace. In particular is the survey presentation software – a web based interface for a company to ask the world "what are you thinking?"

I would like to add we are looking for resellers for this software. So if you are Progress contractor or VAR and need something to approach clients with – go ahead and make contact with me at sauge@amduus.com so we can get you informed about what we are trying to achieve and how to work with us. We are also willing to let the software be rented off our machines for those who want to present a survey without the hassle of buying software for it.

For the programmers, this issue documents the new Open Source product from Amduus Information Works, Inc. called the Configuration SDK (Software Development Kit). This is released under the Berkley Open Source license which means you can include it in your commercial applications, just be sure to acknowledge that Amduus Information Works, Inc. has contributed to the effort. The Configuration SDK is based on the previous work regarding the Parms table found in earlier issues of this E-Zine. It allows the programmer to create a .conf or .ini file that their application can reference on how to handle default behaviors.

For the managers, there is the Formula's for Quality Software article. By quantifying some of the attributes of one's organization, one might be able to identify where to invest in order to achieve more productivity. It is heavily based on a book I read some time ago that I feel still has a lot to offer.

Finally, the mailing list for this publication is getting to big to manage by hand, so I have written some programs for people to subscribe, change their address, and unsubscribe via a web site. The address is as follows:

http://www.amduus.com/online/dev/ezine/EZineHome.html

The source code for this is available for sale at $50.00 per site. It was developed on UNIX and would be a good way to see how a Progress Webspeed/Amduus Blue Diamond program is written – as well as being functional for your own company to send out mailings. If your

interested, email me at sauge@amduus.com for payment arrangements. This software can also be rented from our machines for those who wish to have a mailing list but not the hassle of buying software.

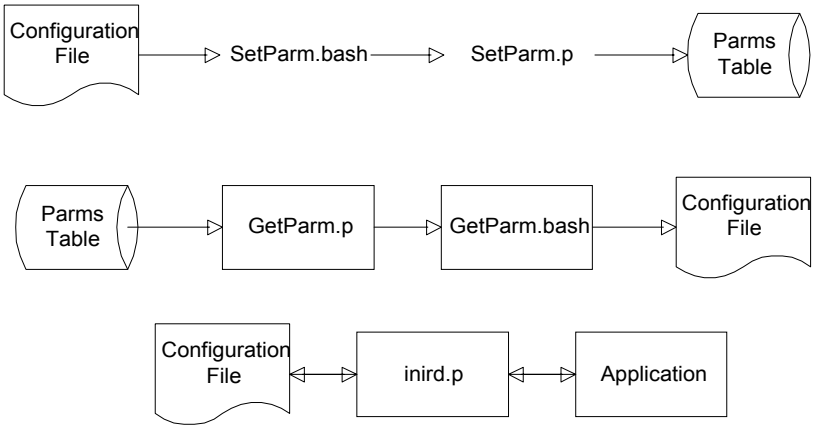
To your success,

Scott Auge
Founder, Amduus Information Works, Inc.

Coding Article: Creating configuration files for your software

Written by Scott Auge sauge@amduus.com

Theory of Operation



Configuration File Syntax

The configuration file files the commonly used syntax in UNIX .conf files and Windows .ini files. The SDK uses # as the comment symbol, and currently any line containing a # is considered a comment. So be forewarned that values and names cannot use the # sign.

The GroupName field is set to the encounter of those values found in []. Thereafter, the records created based on the name/value pair will be under that GroupName.

Finally, those items containing an = sign will be considered an NVP (Name/Value Pair.) The ParmName field of the record would be set to the left side of the = sign, and the ParmValue field would be set to the right side of the = sign. Note that one can have an = sign in the value portion.

```
# Automatically Generated 19/08/01 11:11:34
# File Name: /appl/sdk/listmngr/script/listmngr.conf
# Generated by code derived from open source of
# Amduus Information Works, Inc.  www.amduus.com
```

```
[Server]
DoDelete=NO
ServerName=localhost

[User]
Password=sample
UserName=sauge
```

So for the records created, they would read as follows:

Application	GroupName	ParmName	ParmValue
See SetParm.bash	Server	DoDelete	NO
See SetParm.bash	Server	ServerName	localhost
See SetParm.bash	User	Password	sample
See SetParm.bash	User	UserName	sauge

The Application field is set according to the value passed in the SetParm.bash script.

The GetParm.bash script

This script will call the GetParm.p program to retrieve the values in the Parms table within the connected database. A file will be made for each Application field value. No changes to the database are made by this program.

CONFFILEDIR varible is used to describe where the .conf files should be placed. The names of the files will be the value in the Application field of the Parms table postfixed by .conf. The program will over-write existing files if they are present and it has permission to do so.

```
. setenv.bash
export PROPATH=$PROPATH:/appl/sdk/listmngr/src
# Used to identify the configuration file
export CONFFILEDIR=/appl/sdk/listmngr/config
# END OF CONFIGURATIONS
$DLC/bin/pro -b -pf /db/common/parm/commons.pf -p GetParm.p
```

The SetParm.bash script

This script calls the SetParm.p program. This program will create or modify records in the Parms database based on the values given in the configuration file.

Note the CONFFILE describes where to find the configuration file[1]. This file will be parsed and records placed into the Parms table.

The APPLICATION variable is used to place the value in the Application field of the Parms table. This is separated out so that if one has multiple applications using the same database, or multiple versions of the application using the same database, one can set the field separately.

```
. setenv.bash
export PROPATH=$PROPATH:/appl/sdk/listmngr/src
# Used to identify the configuration file
export CONFFILE=/appl/sdk/listmngr/script/listmngr.conf
# Used to identify the application (because the same DB might be used by
# multiple applications)
export APPLICATION=listmngr
# END OF CONFIGURATIONS
$DLC/bin/pro -b -pf /db/common/parm/commons.pf -p SetParm.p
```

Code Tour

`GetParm.p`

Acts as a wrapper for the GetParm.bash shell script. Simply retrieves the values from the environmental variables and calls iniwrt.p.

`SetParm.p`

Acts as a wrapper for the SetParm.bash shell script. Simply retrieves the values from the environmental variables and calls iniset.p.

`inird.p`

This routine will parse the configuration file for the given Application, GroupName and ParmName values. It does not use or effect the database table Parms at all.

`iniset.p`

This routine will parse the configuration file and apply the settings to the Parms table. If the Parms table is missing a record for the given setting, a record will be created. If the record exists,

[1] One may want to make this a parameter argument, ie $1 or the like under UNIX shell scripts.

then the record will be modified with any new values. The program does not eliminate records that are not in the configuration file, that is, it does not delete existing records.

`iniwrt.p`

This routine will create the .conf files named on the Application field and stored in the directory sent in.

The code can be found at http://www.amduus.com/OpenSrc/SrcLib/Config/ and the FreeFramework web site.

About the author: Scott Auge is the founder of Amduus Information Works. He has been programming in the Progress environment since 1994. His works have included E-Business initiatives and focuses on web applications on UNIX platforms.
sauge@amduus.com

Management Article: What is quality software?

Written by Scott Auge sauge@amduus.com

The open source movement has been a major factor is building quality software. Just recently on the Progress E-Mail Group (www.peg.com), there was a small contest to improve the speed of a program that base-64 encoded an attachment for email. There was a program already available made up of many lines of code and did not function very fast. After word was sent out about a free steak dinner, within hours the program had been re-written using perhaps 50%-60% fewer lines of code and nearly 400% faster in execution.

Quality software can be viewed as a combination of the following factors:

Quality = Functionality + Reliability + Integrity + Efficiency + Maintainability + Ease of Use + Flexibility + Uniqueness + Client Satisfaction

Examining this event, one can see from the above formula how the base-64 encoding software changed in quality. The areas of the formula touched where Efficiency, Maintainability, Ease of Use, and Client Satisfaction. By increasing the speed it could encode attachments, the efficiency of the software certain became a higher value. The reduction in source code made the program more maintainable because less analysis would need to be performed to understand how the software worked. With the increased speed of the source module, ease of use and client satisfaction became elements of an improvement in quality.

Examining the elements of quality software, one knows right away functionality plays a major role. If the software does not do what a user needs it to do, then the software does not fit the scope of the work it is to do. Traveling down the slippery slope, one can also say that even though the software does what the user wants it to do, it may not do it the way the user was hoping it would do. So in defining a metric for the quality of software base on a ranking system, at least these values are needed: one if the software actually does it, another for the software does it, but not in a desirable manner, and a third for software that does it in a desirable manner.

Another facet of quality software is reliability. How often does the software crash, come up with the wrong derived data[2] or abend[3]? If the software does stop undesirably, are the error conditions spelled out and the corrective actions documented to get the software going again?

[2] Derived data is computations and transformations performed on input data to yield new data.

[3] Abnormal End – ended with an error of some kind – that is the software is fragile.

In this day of go-go-go, software efficiency needs to be examined. Not only the efficiency of computation, but the efficiency the software provides to the user. This should be examined on two fronts. One is computational efficiency. The other is the efficiency of the work flow process it is used within the company. A piece of software might be able to allocate inventory for a large BOM ten minutes faster than another, but the work required to get the software to do that in "people" time could ruin the efficiency of the software in the expense of time and user frustration. Other areas of efficiency that can be measured is the size of disk space and memory required for a certain application to function properly.

The maintainability of the software is an important aspect of quality software. I knew of one program where the programmer wrote with the variable names fork, knife, and spoon! Whether the source is viewed by the world or by new programmers within a company, it should be written so it can be easily digested and improved upon. There are many differing opinions on how to achieve this however and beyond the scope of this article!

If software is hard for a user to use, then Ease of Use becomes an issue. Of course, if the user must jump through hoops to get something done, then the issue becomes relevant. As an example, under SAP, placing an order for a part requires the part to be set up else it fails. The process of creating a part within the application is one of the most complex and frustrating processes in the application. It would be better if the software realized that the part was not available, and would prompt the user with the basics needed to create the part. Another Ease of Use issue is the requirement of users to remember various codes. There was a ticket system at one of my work places where by the drop down boxes where filled with letters and numbers that had little meaning to me. Needless to say, the phone was picked up more often than the software used.

Flexibility of the software also plays a role in quality these days. The business market place changes daily and the processes and procedures needed to handle that market place need to be able to change with it. As the company learns it's market place, and the needs of the customers in it, software should not stop the company from modifying it's processes to take advantage of behaviors in the market place by it's customers! (In fact, the configuration article written above is meant to help software developer's achieve quality software by making the software more pliable and flexible by changing certain values that might be part of the business decision making process – i.e., one of the configurations could be Accounts Receivable accumulated by a customer before a manager needs to make an approval of additional purchases.)

Product differentiation has always been the key to the auto industry selling different kinds of cars – hence software uniqueness is important also. Software is less of a product, than it is a service within an organization. If one thinks about it, it acts a lot like a "knows it" guy one can ask a

question of about something happening in the company, or performs automated tasks for a set of users – i.e., a service. One of the benefits of writing one's own software or modifying existing software is the profit motive of running your company different from others. By differentiating your company process from that of others, allows you to discover and utilize new cheaper/more productive ways of doing things (compared to those competitors who use off the shelf software – if one knows what they use, one has a good idea how their company works!) It is the differentiation of the software that prevents the "buy at the lowest price" argument by potential purchasers.

And of course, satisfaction of using the software. This is not only that the software does what the user wants, how the user wants it, but the interaction between the user and the software provider has been a good and rewarding experience.

A lot of these ideas come from a book entitled "Measuring & Motivating Maintenance Programmers" by Landsbaum & Glass. ISBN 0-13-567827-7 Unfortunately is it out of print!

About the author: Scott Auge is the founder of Amduus Information Works, Inc. He has been programming in the Progress environment since 1994. His works have included E-Business initiatives and focuses on web applications on UNIX platforms.
sauge@amduus.com

Contractors/Consulting Companies Available:

If you do work in the Progress world – let me know and I will be able to include you! Price is $10.00 per listing per issue.

Analysts Express Inc.

The Reliable Source for I.T. Consulting and Recruiting
Contact: James Arnold @ 888/889-9091

02-1

Amduus Information Works
4506 Carlyle Court Suite 712
Santa Clara, CA 95054
408-980-8447
scott_auge@yahoo.com sauge@amduus.com

Creation of modules and products for re-sale as well customized Internet/Intranet programming for E-Business in the marketing/manufacturing/service and law enforcement industries.

Community Announcements:

A place to announce Progress User Groups, Open Source Exchanges, etc. No charge!

Wonder where to find those Progress marketing articles?

http://www.progress.com/analyst/
http://www.progress.com/profiles/index.htm
http://www.progress.com/success/index.htm

Product Announcements:

Survey Software

Amduus Information Works, Inc. is creating survey software. This software can be used on a web site to query a population of people about their views and needs. The population could be internal to a company or external to yield a better understanding of the marketplace. Documentation for the application will be available at http://www.amduus.com for free download.

The software ships with source code for better adaptability to your company's application landscape and needs. The software was developed and designed on Linux with Progress Version 9 and Webspeed 3.1. The software can operate on AIX, Linux, Solaris, HP-UX and UNIXWare. The software will also run on Blue Diamond available for free from Amduus Information Works, Inc.

Customers and resellers are welcome to contact Scott Auge at sauge@amduus.com for more information. Street price is $1,000 per machine without Progress licenses.

The software can be rented out at $100.00 per survey per week of taking results.

Mail List Software

The publishing of this E-Zine has created the need to create some software that can handle the creation of mailing lists, web pages to subscribe and unsubscribe as well as a command line to distribute this file.

This software is available for $50.00 per site.

Publishing Information:

Scott Auge publishes this document. I can be reached at scott_auge@yahoo.com.

Currently there are over 300 subscribers and companies that receive this mailing! This mailing is not sent unsolicited, so it is not SPAM.

Amduus Information Works, Inc. assists in the publication of this document:

Amduus Information Works, Inc.
4506 Carlyle Court Suite 712
Santa Clara, CA 95054

Article Submission Information:

Please submit your article in Microsoft Word format or as text. Please include a little bit about yourself for the About the Author paragraph.

Looking for technical articles, *marketing Progress* articles, articles about books relevant to programming/software industry, white papers, etc.