The Progress Electronic Magazine

In this issue:

Publisher's Statement:

We all know how to pass information to modules of code using INPUT/OUTPUT parameters and SHARED variables.

But what happens when the module of code changes in functionality over time? And you don't have the option of changing the code calling that module all the time? Maybe you just don't WANT to change the code calling the module all the time.

This is a condition that has been appearing as we work on smtpmail.p over the years. As new features are added, the parameters have become kludges. This is so the signature of the code doesn't change and people can just throw in the module, recompile, and be done with it. Some changes are important – like bug fixes! Others are more optional in nature.

In the C++ and Java world, one can usually change the module/object to include some additional methods and properties to enhance it for a section of code. Other parts of the application that are not interested in those changes need not be touched. We need something like this for the Progress 4GL world and an answer appears below!

Alas, I have finished up a contract with the Superior Court of California and am available once again to do work. If any of you have some work that can be done in the United States or via the wire, take a gander at my resume: http://www.amduus.com/Resumes/ScottAuge.html. I tend towards web based applications on UNIX/Linux operating systems. My skills include law enforcement, manufacturing, and service oriented companies and their problems. I also offer Webspeed training.

Coding Article: Flexible Parameter Passing With The 4GL

By Scott Auge

As smtpmail.p has evolved over the years, it's parameters have become more of a kludge as people contribute additional functionality to the module that requires inputs.

I have seen this happen over the years to other modules – especially modules of code that are complicated. As the business changes, these modules of code need to change.

In the best of worlds, re-writing the code that calls the modules would be done. But sometimes that doesn't happen for what-ever reason. Sometimes one doesn't have access to the code calling the module, there is no time, or no one knows the code well enough to comfortably do so. I am sure there are other reasons.

A way needs to be developed to be able to call modules of code without having to re-write the calls to it.

In the past, this was done with shared variables. However, keeping track of what variables do what (as well where they are assigned) can become tricky in an application. And there is the risk of the variables purpose changing when someone realizes it "just happens" to have a value that they need in computation in that module.

And of course with shared variables, there is the whole what to re-compile problem again. (I have seen apps that take over two hours to compile.) There is the idea that they are frowned upon also – I don't feel this way, but one does have to be strict about shared variable management.

So, we need a method of setting up parameters, where by:

- New parameters can be introduced into the procedure
- No re-compilation is needed in calling programs (unless they choose to use those new parameters)
- If people are not interested in using those parameters they don't need to set up dummy variables as place holders
- Changes don't require modification of code when new optional features are added
- Bug fixes can still be plugged in and re-compiled
- New output parameters can be set up
- Output parameters can be ignored in the code without coding for their existence

The answer – some code I have included in runapi.i listed below. But first, lets see how to conveniently use this code… .

Testing Code

We have three pieces of code here. One is called A. Sometimes A needs to be called with four parameters. Other times, A needs to be called with only three parameters. Notice also, that not only the number of parameters needed change, but that the parameters are different!

> *Before thinking this might not be so real world, consider smtpmail.p – a program that sends out mail. Sometimes one only needs to send out Subject, To, From, and Body – four parameters. Other times, one may need to send out a Subject, To, From, CC and Body – five parameters. In the C++ world, this is called overloading where the same method has different numbers and types of arguments.*

Now we have two other programs, CallA.p and CallA2.p that use A to achieve something.

CallA.p

This is an example program that calls A.p with four arguments called Parm1, Parm2, Parm3, and Parm4. We can easily add an argument to A.p without updating CallA.p and have it work just fine (unless that argument is required.)

```
{runapi.i NEW}

SetParm("A.p", "Parm1", "1").
SetParm("A.p", "Parm2", "2").
SetParm("A.p", "Parm3", "3").
SetParm("A.p", "Parm4", "4").

RUN A.p.
```

Note that in order to use this parameter methodology, we need to include the `runapi.i` file into the application.

This file should be included in ALL programs that use the parameter passing methodology described in this article.

When to NEW the include file

The problem that the programmer is faced with, is to determine if that include file should be have the pre-processor argument NEW included or not. As you will see in the following code, A.p does NOT have NEW in it's invocation of the include file.

The rule to determine if the include file has NEW in it is this:

> *If the program is NEVER called by another program using this methodology, it should have NEW as a pre-processor argument.*
>
> *If the program is DEFINITELY called by another program using this methodology, it should NEVER have NEW as a pre-processor argument.*
>
> *Modules that are called by modules do not have to include NEW in the invocation of the include file.*

CallA2.p

Here we have a call to A.p with only three arguments and parameters named Parm1, Parm3, and Parm5!

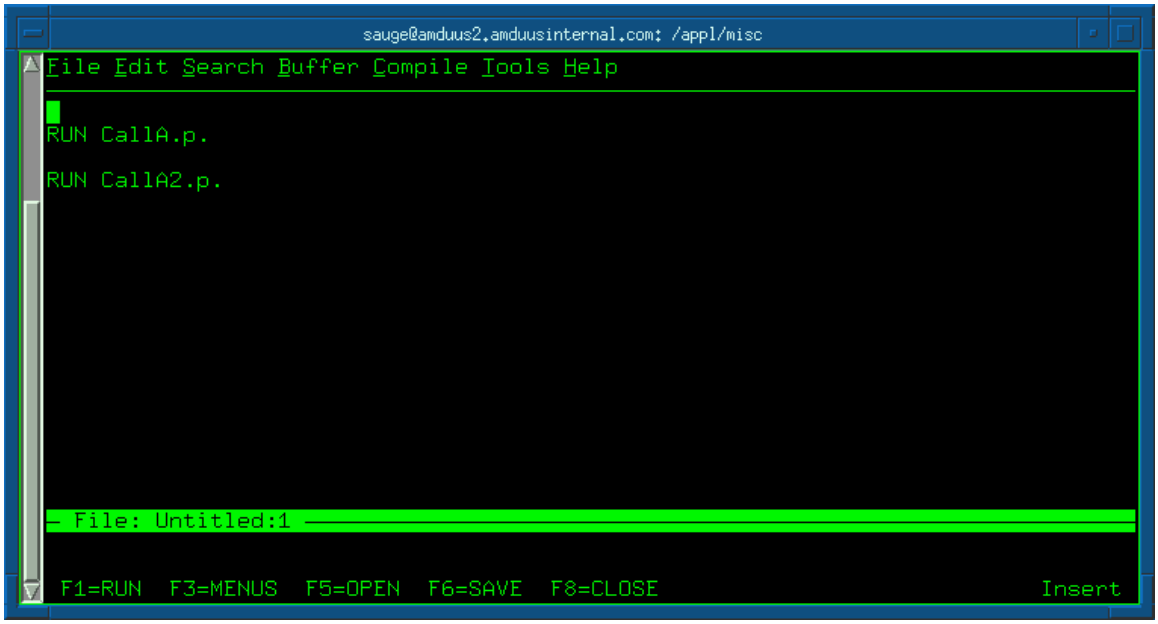
```
{runapi.i NEW}

SetParm("A.p", "Parm1", "1").
SetParm("A.p", "Parm3", "3").
SetParm("A.p", "Parm5", "5").

RUN A.p.
```

A.p

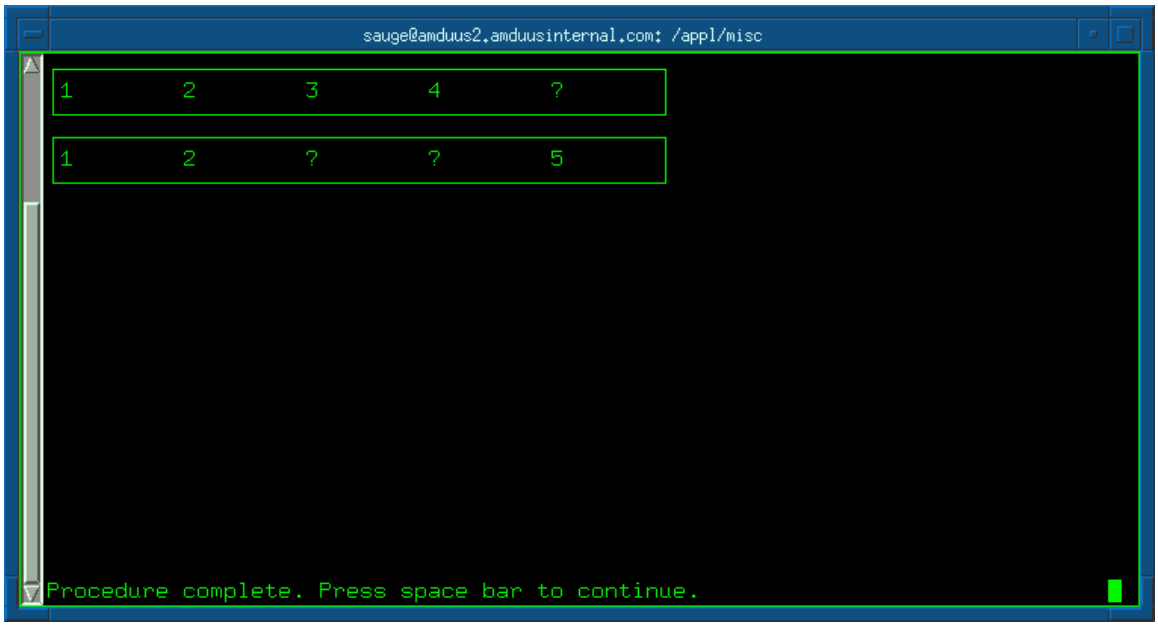
Here is the code for A.p. It mostly focuses on displaying the parameters sent to it.

```
{runapi.i}

DISP GetParm("A.p", "Parm1").
DISP GetParm("A.p", "Parm2").
DISP GetParm("A.p", "Parm3").
DISP GetParm(PROGRAM-NAME(1), "Parm4").
DISP GetParm(PROGRAM-NAME(1), "Parm5").
```

Below we run the programs listed above. The first screen is the program in the editor buffer calling out to the other programs which will call A.p in turn.



Here we have the output which is generated by A.p. Note that the parameters not passed have ? (the Progress NULL character) presented.



Things to think about

Output Parameters

One of the exciting things about this, is that output parameters can be set! This could be used for returning a set of data, or an error condition of some form. GetParm() works just as well in calling program as SetParm() does.

Object Oriented Behavior: Inheritance

This set of data could be viewed up n calling programs with a given name from a module. It could also present the program with error conditions as well output and input values from sub modules called. One can think of this as "inheritance" where a module will inherit the inputs and outputs of the modules below it. One may want to call the data the expected calling tree:

An instance called `workorder.credit.check.myinstancekey` might refer to data created in workorder.p -> credit.p -> check.p while `workorder.myinstancekey` might refer to data handled purely in workorder.p.

Another thing is that since the parameters pass through the various programs, if the application uses "global" parameters, they can be set somewhat up the calling tree and still be available to the module. They can exist or not exist based on the module's need.

Object Oriented Behavior: Instancing

By slightly changing the name of the value's module name, one can make an "instance" of the module. For example, instead of:

```
SetParm("A.p", "Parm5", "5").
```

One might:

```
SetParm("A.p", "WorkingInstance", "Part.2876").
SetParm("A.p-Part.2876", "Parm5", "5").
```

This in effect allows multiple pieces of data to be related to manipulation in A.p.

Then have something like this in the module:

```
GetParm("A.p-" + GetParm("A.p", "WorkingInstance"), "Parm5").
```

By using the WorkingInstance parameter, it tells the module which data set to work with.

Note that “privacy” of this instance data is practically nil.

Here is a program that creates two “instances” and calls a “method” called ObjA.p. ObjA.p merely displays the data (or “properties”) of the current instance.

CallObjA.p

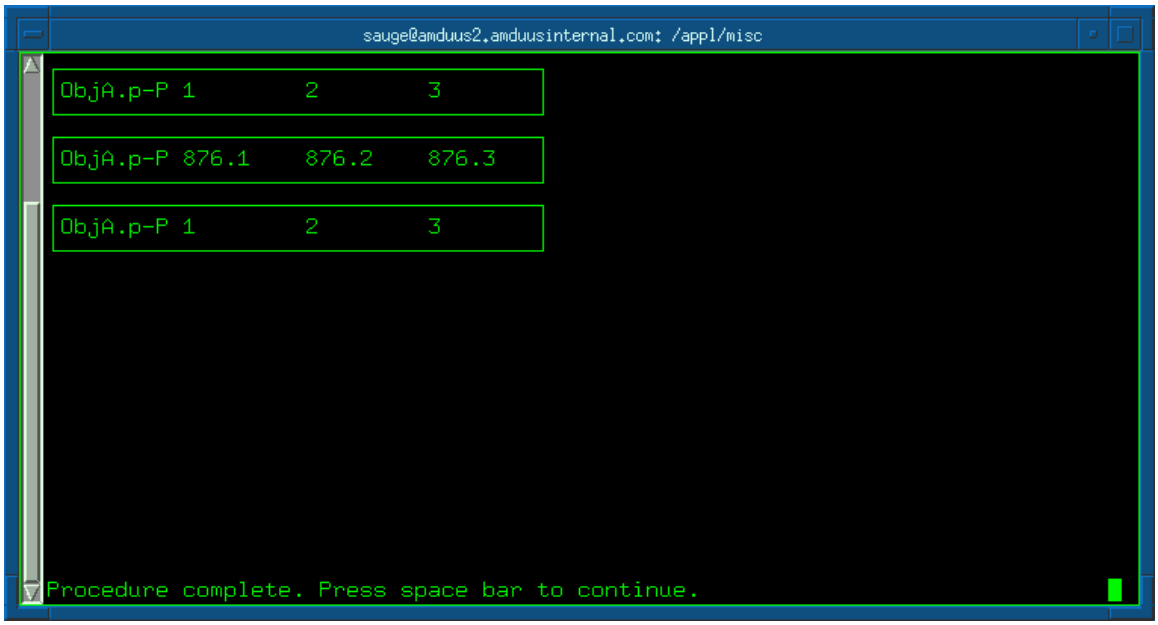
```
{runapi.i NEW}

DEF VAR MyInstanceName AS CHARACTER NO-UNDO.

/* Identify the "instance" we are working with */

ASSIGN MyInstanceName = "ObjA.p-Part.3456".
SetParm("ObjA.p", "WorkingInstance", MyInstanceName).

/* "Construct" the instance */

SetParm(MyInstanceName, "Parm1", "1").
SetParm(MyInstanceName, "Parm2", "2").
SetParm(MyInstanceName, "Parm3", "3").
SetParm(MyInstanceName, "Parm4", "4").

/* Call a "method" */

RUN ObjA.p.

/* Identify the "instance" we are working with */

ASSIGN MyInstanceName = "ObjA.p-Part.876".
SetParm("ObjA.p", "WorkingInstance", MyInstanceName).

/* "Construct" the instance */

SetParm(MyInstanceName, "Parm1", "876.1").
SetParm(MyInstanceName, "Parm2", "876.2").
SetParm(MyInstanceName, "Parm3", "876.3").
SetParm(MyInstanceName, "Parm4", "876.4").

/* Call a "method" */

RUN ObjA.p.

/* Switch "Instances" */

SetParm("ObjA.p", "WorkingInstance", "ObjA.p-Part.3456").

RUN ObjA.p.
```

ObjA.p

```
{runapi.i}

DEF VAR MyInstanceName AS CHARACTER NO-UNDO.
```

```
ASSIGN MyInstanceName = GetParm("ObjA.p", "WorkingInstance").

DISP GetParm("ObjA.p", "WorkingInstance").

DISP GetParm(MyInstanceName, "Parm1").
DISP GetParm(MyInstanceName, "Parm2").
DISP GetParm(MyInstanceName, "Parm3").
```

Screen showing the output of CallObjA.p using ObjA.p



Grouping Data

Of course, most people use WORK-TABLES to group related data. Just being complete!

runapi.i

Here is the program that works the magic. The basis of holding the information is in a Work-file. This is in-memory and should be quite fast. If the data increases past such limits, one can re-set it into a TEMP-TABLE. Do not use a common table in the database, but a table construct local to the user such as a TEMP-TABLE or a WORK-TABLE.

The two functions SetParm() and GetParm() merely present a convenient means of manipulating entries in the table.

```
/*
 * Removal of this header is illegal.
 * Written by Scott Auge scott_auge@yahoo.com sauge@amduus.com
 * Copyright (c) 2002 Amduus Information Works, Inc.  www.amduus.com
```

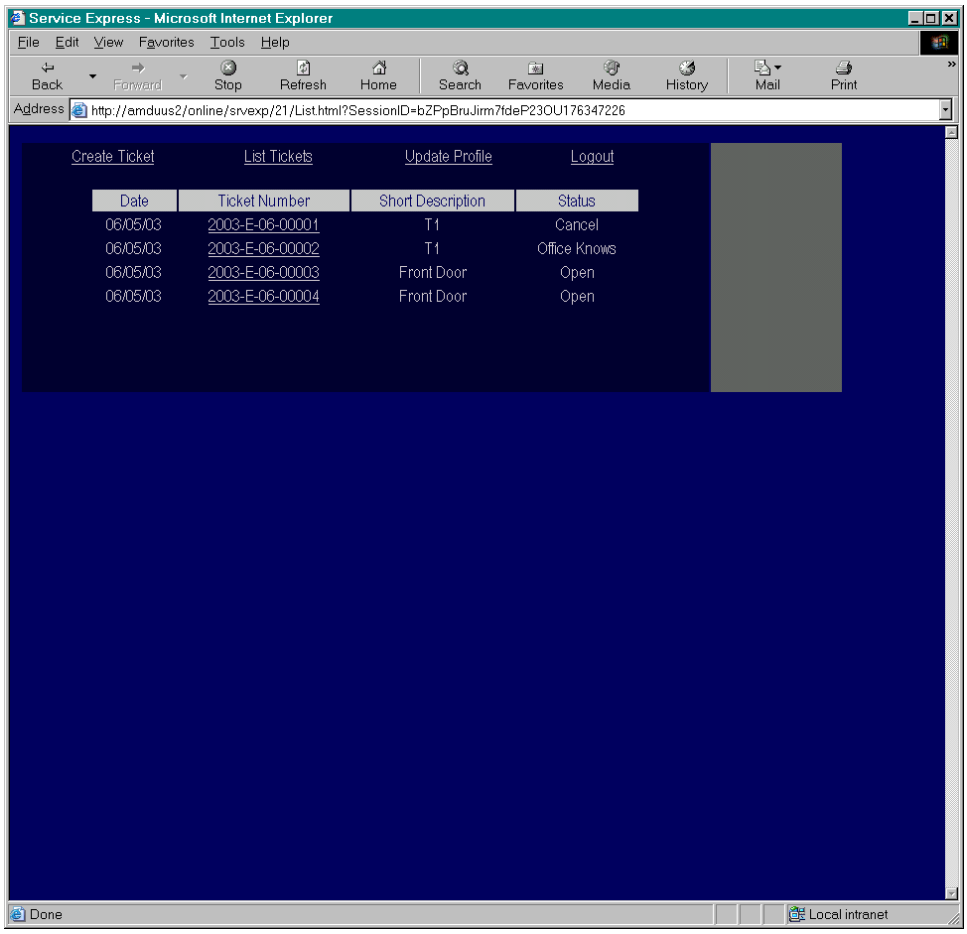
```
 *
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions
 * are met:
 * 1. Redistributions of source code must retain the above copyright
 *    notice, this list of conditions and the following disclaimer.
 * 2. Redistributions in binary form must reproduce the above copyright
 *    notice, this list of conditions and the following disclaimer in the
 *    documentation and/or other materials provided with the distribution.
 * 3. All advertising materials mentioning features or use of this software
 *    must display the following acknowledgement:
 *      This product includes software developed by Amduus Information Works
 *      Inc. and its contributors.
 * 4. Neither the name of Amduus Information Works, Inc. nor the names of
 *    its contributors may be used to endorse or promote products derived
 *    from this software without specific prior written permission.
 *
 * THIS SOFTWARE IS PROVIDED BY AMDUUS AND CONTRIBUTORS ``AS IS'' AND
 * ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
 * IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
 * ARE DISCLAIMED.  IN NO EVENT SHALL THE AMDUUS OR CONTRIBUTORS BE LIABLE
 * FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
 * DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS
 * OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
 * HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT
 * LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY
 * OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
 * SUCH DAMAGE.
 *
 */

DEF VAR RCSVersion AS CHARACTER INIT "$Header$" NO-UNDO.

/*********************************************************************/
/* Routines to prep a property (as in C++ and Java) API type interface */
/* to progress procedures.                                           */
/*********************************************************************/

/*********************************************************************/
/* APIName      - If table is shared with multiple procedures, recom-  */
/*                procedure's name.                                  */
/* APIParmName  - Name of the parameter used in the module           */
/* APIParmValue - Value to send to procedure - character only!       */
/*********************************************************************/

DEF {1} SHARED WORK-TABLE API_Parm
  FIELD Name      AS CHARACTER
  FIELD ParmName  AS CHARACTER
  FIELD ParmValue AS CHARACTER.

/*********************************************************************/
/* Use this to set a parameter value                                 */
/*********************************************************************/

FUNCTION SetParm RETURNS LOGICAL
(INPUT cName AS CHARACTER,
 INPUT cParmName AS CHARACTER,
 INPUT cParmValue AS CHARACTER):

  FIND FIRST API_Parm NO-LOCK
  WHERE API_Parm.Name BEGINS cName
    AND API_Parm.ParmName =  cParmName
```

```
  NO-ERROR.

  IF NOT AVAILABLE API_Parm THEN CREATE API_Parm.

  ASSIGN
  API_Parm.Name = cName
  API_Parm.ParmName = cParmName
  API_Parm.ParmValue = cParmValue.

  RETURN TRUE.

END. /* FUNCTION SetParm () */

/*********************************************************************/
/* Use this to get a parameter value                                 */
/*********************************************************************/

FUNCTION GetParm RETURNS CHARACTER
(INPUT cName AS CHARACTER,
 INPUT cParmName AS CHARACTER):

  FIND FIRST API_Parm NO-LOCK
  WHERE API_Parm.Name BEGINS cName
    AND API_Parm.ParmName =  cParmName
  NO-ERROR.

  IF NOT AVAILABLE API_Parm THEN RETURN ?.

  RETURN API_Parm.ParmValue.

END. /* FUNCTION GetParm () */
```

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*

Reference Partners Wanted

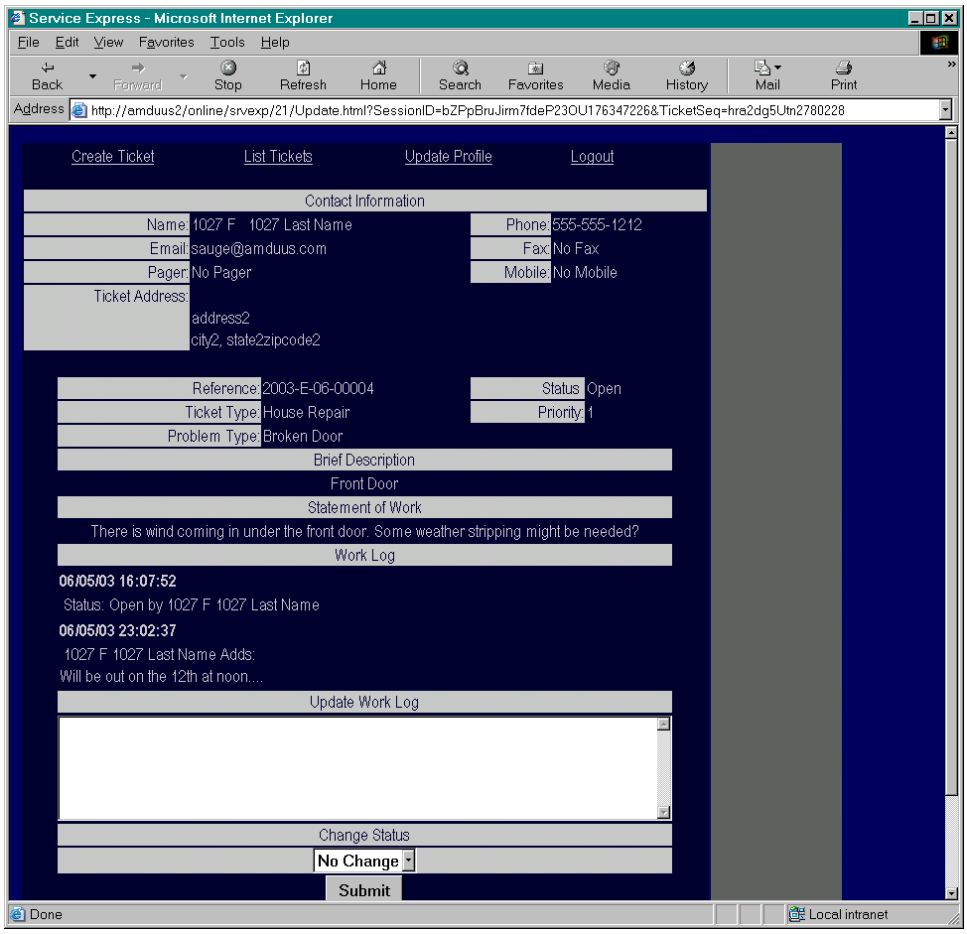
Amduus Information Works, Inc. finally has a working version of the Service Express available!



A listing of tickets belonging to a customer

What is it?

What is it? It is a work order/issue tracking system. It lets your customers (whether they be your co-workers, or the company's customers) create work orders (tickets) to solve problems. The tickets can be categorized by types and problems; And a work flow can be developed to track the status of the ticket. All of this is configurable to match your industry or user base.



A customer's ticket information and work log

Who!

We would like to make it available to five companies/organizations that might be willing to offer suggestions for improvement and are willing to be referenced as user's of the software.

The company/organization can be governmental, non-profit, private, or a corporation.

Qualification Questions For Free Access To Service Express	
Market Served By Your Company?	
Is your organization non-profit?	Yes No
Is your organization a corporation?	Yes No
Would external customers use your site?	Yes No
Would internal customers use your site?	Yes No
Would the site be integrated into an existing site?	Yes No
Types of problems the site would aid in managing?	
Contact Person's Name	
Contact Person's Phone	
Contact Person's E-Mail Address	
Would your company be willing to act as a reference for the software?	
Would you be able to spend a day or so to learn how to and to perform configurations for the software?	
Would you be willing to pay a nominal fee to have Amduus configure the software?	
Estimated number of tickets created per day?	
Country of Company/Organization? *Currently Service Express is available only in English*	

How!

You can reach the public portion of a demo for the application here:

https://www.amduus.com/cgi-bin/se0001pub/21/index.html

You would be acting as a customer of a company called "Demo" with relations to the organization running the web site. Your company name would be Demo, and to prove that you are indeed an employee/representative of that company, you would know that the authorization code is Demo.

Let me point out that you could be an internal customer, such as HR approaching IT about setting up a new employee; Or, you could be an external customer approaching the organization with a request for a service to be performed.

This could be used by a manufacturing company for repairs/over-hauls of their equipment. An apartment complex for handling property issues. Any company/organization that wishes to interact with it's customers in a manner that needs to be defined and tracked for a process of completion.

Be sure to see the accompanying Power Point presentations for more information.

Please contact Scott Auge at sauge@amduus.com if you have additional questions!

Publishing Information:

Scott Auge publishes this document. I can be reached at sauge@amduus.com.

Amduus Information Works, Inc. assists in the publication of this document by providing an internet connection and web site for redistribution:

Amduus Information Works, Inc.
1818 Briarwood
Flint, MI 48507
http://www.amduus.com

Other Progress Publications Available:

This document focuses on the programming of Progress applications. If you wish to read more business oriented articles about Progress, be sure to see the Profile's magazine put out by Progress software http://www.progress.com/profiles/

There are other documents/links available at http://www.peg.com .

There is a web ring of sites associated with Progress programming and consultants available at http://i.webring.com/hub?ring=prodev&id=38&hub .

Products/Services Available From Amduus:

Amduus Information Works, Inc. is a Progress reseller and ASPen partner. We primarily develop UNIX/Linux based applications with web interfaces for manufacturing, service, and law enforcement communities.

We also perform integration of Progress applications to non-Progress applications through such languages and tools as MQ Series, C, and C++.

Amduus provides support for the following applications: Blue Diamond, Denkh, Denkh HTML Reporter, Red Arrow Portal (CMS), Survey Express and other software.

Amduus is looking for consultants who might want to promote the use of our tools at user groups and companies they might work in. Send some information to sauge@amduus.com to let me know you are out there!

Article Submission Information:

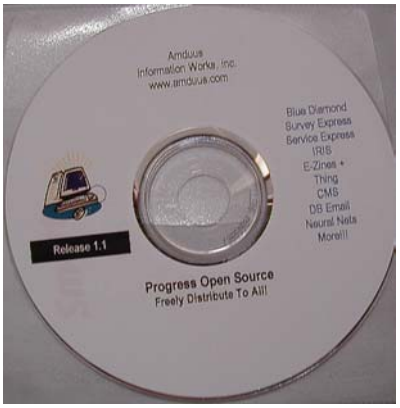
Please submit your article in Microsoft Word format or as text. Please include a little bit about yourself for the About the Author paragraph.

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

Send your articles to sauge@amduus.com! Thanks!

Order Form for Progress Open Source CD-ROM

COUPON 001A



This is an offer for the CD-ROM at lower than list savings!
This is a great way to support the E-Zine too!

Mail this form to:
Amduus Information Works, Inc.
1818 Briarwood
Flint, MI 48507

Please send _____ copies of the Open Source CD-ROM at $25.00 per disk to:

Name _____
Company _____
Address _____
City _____
State _____ Country _____
Zip _____

Please make your checks/money orders out to: Amduus Information Works, Inc. Cash works too!
This offer only valid in the United States of America and those countries with postal agreements with the United States Post Office.

The CD-ROM includes (all source code included):

- Blue Diamond/IRIS – Webspeed alternatives
- Survey Express – easily create text templates of surveys and then have the program generate the web pages automatically
- Service Express – Web based Help Desk.
- The Progress E-Zines, books on learning to program in Webspeed (PDF/Word/HTML)
- Denkh HTML Reporter – web based report writer
- CMS – a web content management system
- DB Email – Use pop3 to download emails into a Progress database
- Neural Networks – experiments in spam recognition and text message classification
- Denkh – create PDF file reports for Webspeed/UNIX CHUI!

- More!