Project: mod_openopc_2

Codename: Bad Medicine

Version: 3.3-2
Build #: 66
Date: 2016_0621

Release: July 19, 2016

FORKED PROEJCTS

Project Name	*http://*	*Contact*	*License*
OpenOPC for Python	openopc.sourceforge.net	Barry Barnreiter	GNU GPL (Perpetual)

LICENSE

GNU GENERAL PUBLIC LICENSE
Version 3, 29 June 2007

Preamble

The GNU General Public License is a free, copyleft license for software and other kinds of works.

The licenses for most software and other practical works are designed to take away your freedom to share and change the works. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change all versions of a program--to make sure it remains free software for all its users. We, the Free Software Foundation, use the

GNU General Public License for most of our software; it applies also to any other work released this way by its authors. You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for them if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs, and that you know you can do these things.

To protect your rights, we need to prevent others from denying you these rights or asking you to surrender the rights. Therefore, you have certain responsibilities if you distribute copies of the software, or if you modify it: responsibilities to respect the freedom of others.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must pass on to the recipients the same freedoms that you received. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

Developers that use the GNU GPL protect your rights with two steps: (1) assert copyright on the software, and (2) offer you this License giving you legal permission to copy, distribute and/or modify it.

For the developers' and authors' protection, the GPL clearly explains that there is no warranty for this free software. For both users' and authors' sake, the GPL requires that modified versions be marked as changed, so that their problems will not be attributed erroneously to authors of previous versions.

Some devices are designed to deny users access to install or run modified versions of the software inside them, although the manufacturer can do so. This is fundamentally incompatible with the aim of protecting users' freedom to change the software. The systematic pattern of such abuse occurs in the area of products for individuals to use, which is precisely where it is most unacceptable. Therefore, we have designed this version of the GPL to prohibit the practice for those products. If such problems arise substantially in other domains, we stand ready to extend this provision to those domains in future versions of the GPL, as needed to protect the freedom of users.

Finally, every program is threatened constantly by software patents. States should not allow patents to restrict development and use of software on general-purpose computers, but in those that do, we wish to avoid the special danger that patents applied to a free program could make it effectively proprietary. To prevent this, the GPL assures that patents cannot be used to render the program non-free.

The precise terms and conditions for copying, distribution and modification follow.

TERMS AND CONDITIONS

0. Definitions.

"This License" refers to version 3 of the GNU General Public License.

"Copyright" also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

"The Program" refers to any copyrightable work licensed under this License. Each licensee is addressed as "you". "Licensees" and "recipients" may be individuals or organizations.

To "modify" a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a "modified version" of the earlier work or a work "based on" the earlier work.

A "covered work" means either the unmodified Program or a work based on the Program.

To "propagate" a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

To "convey" a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays "Appropriate Legal Notices" to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

1. Source Code.

The "source code" for a work means the preferred form of the work for making modifications to it. "Object code" means any non-source form of a work.

A "Standard Interface" means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

The "System Libraries" of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A "Major Component", in this context, means a major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

The "Corresponding Source" for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work's System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require, such as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work.

2. Basic Permissions.

All rights granted under this License are granted for the term of copyright on the Program, and are irrevocable provided the stated conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with you.

Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

3. Protecting Users' Legal Rights From Anti-Circumvention Law.

No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures.

4. Conveying Verbatim Copies.

You may convey verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

5. Conveying Modified Source Versions.

You may convey a work based on the Program, or the modifications to produce it from the Program, in the form of source code under the terms of section 4, provided that you also meet all of these conditions:

a) The work must carry prominent notices stating that you modified it, and giving a relevant date.

b) The work must carry prominent notices stating that it is released under this License and any conditions added under section 7. This requirement modifies the requirement in section 4 to "keep intact all notices".

c) You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.

d) If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an "aggregate" if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation's users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

6. Conveying Non-Source Forms.

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

a) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.

b) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or customer support for that product model, to give anyone who possesses the object code either (1) a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical medium customarily used for software interchange, for a price no more than your reasonable cost of physically performing this conveying of source, or (2) access to copy the Corresponding Source from a network server at no charge.

c) Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.

d) Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source may be on a different server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the object code saying where to find the Corresponding Source. Regardless of what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements.

e) Convey the object code using peer-to-peer transmission, provided you inform other peers where the object code and Corresponding Source of the work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A "User Product" is either (1) a "consumer product", which means any tangible personal property which is normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, "normally used" refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

"Installation Information" for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as part of a transaction in which the right of possession and use of the User Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source code form), and must require no special password or key for unpacking, reading or copying.

7. Additional Terms.

"Additional permissions" are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

a) Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this License; or

b) Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or

c) Prohibiting misrepresentation of the origin of that material, or requiring that modified versions of such material be marked in reasonable ways as different from the original version; or

d) Limiting the use for publicity purposes of names of licensors or authors of the material; or

e) Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or

f) Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for any liability that these contractual assumptions directly impose on those licensors and authors.

All other non-permissive additional terms are considered "further restrictions" within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document, provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

8. Termination.

You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

9. Acceptance Not Required for Having Copies.

You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

10. Automatic Licensing of Downstream Recipients.

Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

An "entity transaction" is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party's predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

11. Patents.

A "contributor" is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor's "contributor version".

A contributor's "essential patent claims" are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For purposes of this definition, "control" includes the right to grant patent sublicenses in a manner consistent with the requirements of this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor's essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a "patent license" is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to sue for patent infringement). To "grant" such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. "Knowingly relying" means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient's use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work and works based on it.

A patent license is "discriminatory" if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license (a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses to infringement that may otherwise be available to you under applicable patent law.

12. No Surrender of Others' Freedom.

If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

13. Use with the GNU Affero General Public License.

Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU Affero General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the special requirements of the GNU Affero General Public License, section 13, concerning interaction through a network will apply to the combination as such.

14. Revised Versions of this License.

The Free Software Foundation may publish revised and/or new versions of the GNU General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU General Public License "or any later version" applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU General Public License, you may choose any version ever published by the Free Software Foundation.

If the Program specifies that a proxy can decide which future versions of the GNU General Public License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

15. Disclaimer of Warranty.

THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. Limitation of Liability.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

17. Interpretation of Sections 15 and 16.

If the disclaimer of warranty and limitation of liability provided above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an absolute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively state the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

```
    <one line to give the program's name and a brief idea of what it does.>
    Copyright (C) <year>  <name of author>

    This program is free software: you can redistribute it and/or modify
    it under the terms of the GNU General Public License as published by
    the Free Software Foundation, either version 3 of the License, or
    (at your option) any later version.

    This program is distributed in the hope that it will be useful,
    but WITHOUT ANY WARRANTY; without even the implied warranty of
    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
    GNU General Public License for more details.

    You should have received a copy of the GNU General Public License
    along with this program.  If not, see <http://www.gnu.org/licenses/>.

Also add information on how to contact you by electronic and paper mail.

  If the program does terminal interaction, make it output a short
notice like this when it starts in an interactive mode:

    <program>  Copyright (C) <year>  <name of author>
    This program comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
    This is free software, and you are welcome to redistribute it
    under certain conditions; type `show c' for details.
```

SCOPE

mod_openopc is a cross platform utility / package / program (whatever you may wish to call it) that has the following goals:

1 – function primarily in a Linux kernel 2.6 or greater environment, secondarily function in other environments (including MS Windows [2k kernel or greater, which means Windows 2000 and up]), Macintosh, and any other UNIX or UNIX-Like Operating System (BSD / SunOS [Solaris] / etc...)

2 – adapt the complex functions of 'Open OPC for Python' into a transparent implementation.

3 – serve as a central point of contact for the following discrete items, and integrate them:

A OPC Servers
B MySQL Servers and Databases
C Web pages issuing read/write commands to OPC Devices
D Any other 3rd party program or device which issues read/write command to an OPC Device

4 – be self-correcting* with regard to communication between the following:

A OPC Server (and Gateway) and mod_openopc itself
B OPC Device and OPC Server
C MySQL and mod_openopc

• correction takes the form of 'attempting reconnection perpetually until successful', 'breaking connection and reconnecting', or (after attempting the former) faulting out of the routine and generating a fault / error to the system log.

5 – function in the form of an 'instance based' application; that is, each 'task' (for lack of a better term) is launched independently, creating multiple threads (one per instance), which allows for excellent scalability and inherently allows test or problem instances to be killed / terminated independently of the many other running instances.

6 – function as a bridge between two OPC Devices on the same OPC server. For

example, an OPC Device on Ethernet may not typically be able to communicate with an OPC Device on a serial network (without expensive proprietary hardware and program modifications). mod_openopc shall enable fast (relative) read and writes between these devices (messaging).

7 – function as a bridge between two OPC Devices on different OPC servers / networks. For example, an OPC Device on Ethernet, such as a Koyo Direct Logix, cannot typically communicate with an Allen Bradley SLC Series PLC also on Ethernet, or a Modicon on MODBUS. However, provided an OPC Server exists for the devices, then mod_openopc can jump the gap between servers and effectively message between.

8 – run constantly in an industrial environment without the requirement for program maintenance or 'babysitting'. mod_openopc should (once setup properly) recover from communication errors (which are a fact of life on any communication network under any protocol), recover from system restarts due to power outage or similar, and recover from read / write OPC errors. This will not necessarily happen gracefully, but it must happen without human intervention.

SYSTEM REQUIREMENTS – HARDWARE

mod_openopc is intended to be run as a server-side application. This means that a dedicated machine should be setup solely for it. This is not due to compatibility issues or anything of that nature. Rather, it simply will require a lot of "horsepower" to run as intended, when under load and scaled.

Absolute Minimum Hardware

Architecture	i386
Processor	2.0 – 2.2 GHz AMD Athlon Thunderbird / Barton 3.0 GHz Intel Pentium 4 Single Core
Memory	2 GB DDR PC3200 400 MHz
Disk Space	… depends on the size of your database (how much data you wish to store). Plan on no less than 1 GB per machine per year of retention, at an absolute minimum. Your mileage ***will*** vary greatly.

Suggested Deployment (500,000 sq. ft. facility):

Architecture	x86_64 (note: this is for the host OS, your OPC Server should reside on a dedicated MS-Windows based machine, or in a MS-Windows based virtual machine within your host OS. This is for OPC Automation compatibility assurance). ** Minimum Windows version is XP Pro Service Pack 2
Processor	2 each – AMD Opteron 3rd Generation Barcelona Quad Core @ 2 GHz or better.
Memory	32 GB DDR – 800 Mhz ECC Registered Server Memory
Disk Space	… depends on the size of your database (see above).
Virtual Machine	at this time, we are recommending Sun Virtual Box, as VMWare Virtual Server is no longer supported under newer Linux / Unix releases due to new glibc builds and the

decision by VMWare to phase out Virtual Server Free Edition. Sun Virtual Box is easier / faster / and ultimately better suited anyway (in our subjective opinion)... and it is also freeware / cheap-ware.

Warning:

We do NOT advise runing mod_openopc itself (any command but 'GATEWAY_DAEMON') in a virtual machine, without ensuring clock integrity.

The system clock under all versions of Linux that we've tested with a 2.6.18 or better kernel (up to 2.6.23 at least) has failed to stay sync'd properly. Similar issues plague Windows guest operating systems; we've not checked a true UNIX. That said, mod_openopc relies heavily upon the system clock for automated run, and can fault out, causing unpredictable behavior, if the clock 'suddenly stops'. ...and, for the validity of record time stamps, it is suggested 'dalight-savings-time' be disabled, else you'll end up with duplicate records on that one-day-a-year when an hour is repeated on a DST clock.

SYSTEM REQUIREMENTS – SOFTWARE

mod_openopc was developed under Fedora Core 7 [Moonshine], and was intended to be run upon a Linux 2.6 or later-kernel based Operating System. However, it has also been re-developed (version 3 and later) under MS Windows Server 2003 R2, and while performance is degraded (slower) under non Unix/Linux OS's, it functions properly. mod_openopc is compatible with any Operating System that can run Python and MySQL.

*** however we still strongly recommend a UNIX or Linux Operating System, regardless ***

http://www.oracle.com/solaris	SOLARIS
http://www.redhat.com	RHEL Linux
http://www.oracle.com/us/technologies/linux	Unbreakable Linux
http://www.scientificlinux.org	Scientific Linux
http://www.centos.org	CentOS
http://www.freebsd.org	Free BSD

1 – Python Programming Language **http://www.python.org**

- version 2.6 or greater (tested against 2.6 on WIN and UNIX platforms)
- note: version 2.7-2 is provided

2 – Oracle/Sun Microsystems MySQL **http://dev.mysql.com**

- version 5.0 or greater (we developed against version 5.1 and 5.0.4)

3 – Python Remote Objects “PYRO” **http://pyro.sourceforge.net**

- version 3.7 or greater
- note: version 3.15 is provided

4 - MySQL-Python **http://mysql-python.sourceforge.net**

- version 1.2 or greater
- note: version 1.2.3 is provided

5 – py-setproctitle for Python

- version 1.0 or greater **http://pypi.python.org/pypi/setproctitle/**
- note: version 1.1.3 is provided

6 – setuptools for Python

- version 0.6 or greater **http://pypi.python.org/pypi/setuptools**
- note: version 0.6-11 (*nix) and 0.6-12 (Win) provided.

7 – For your Windows based OPC Server (whether live machine or virtual machine)...

- PyWin32 for Windows version 2.14 or greater **http://pywin32.sourceforge.net**
- note: version 2.16 is provided

8 – Not Required, but Nice to Have...

- Oracle/Sun Virtual Box **http://www.virtualbox.org**
  (lets you host your OPC Server as a virtual server inside of your main OS / Server)

BE ADVISED!

- All required dependencies are included in the folders...
  - /mod_openopc_2/help/dependancies
    - /unix Linux and Unix source files
    - /win32 Win32 binaries (do not mix with 64)
    - /win64 Win64 binaries (do not mix with 32)

- If you do not already have one, or you have an incompatible one, there is a known working and free-ware (open-ware) Automation DLL (DCOM Interface) in the following folder...
  - /mod_openopc_2/help/dependancies
    - /automation_dll
      GrayBox DLL version 1.2.8-508

BASIC INSTALLATION

After satisfying all of the hardware and software requirements, you can now install mod_openopc.

A legitimate and current source shall remain at:

http://download.spinellicreations.com/mod_openopc

*** The following instructions assume a Linux Host OS, however, they are identical (sans file-paths) for other operating systems ***

Decompress the contents of your downloaded file to whichever directory you desire that mod_openopc actually run in. For example, we suggest decompressing the entire structure to *nix filetree...

```
       /opt/mod_openopc_2/
```

You should end up with a hierarchy that is...

```
      /opt/mod_openopc_2/
                        ./gwcomm
                        ./help
                        ./options
                        ./prog
                        ./server_reset
                        ./temp
```

While this is the age of 'security' and such, mod_openopc has not been tested or designed to be 'permissions friendly'. That is to say, it was developed, and assumes, that all files within itself are RWX available to all users. To ensure this, change permissions for the tree to 0777, such as... (this does not apply in a WIN environment)

```
      [root@server] chmod 777 -R /opt/mod_openopc_2
```

Open ./options/options.opt and modify as follows:

> *** if you mess up, that's ok... there is a 2nd copy, called default_options.opt which you can simply copy and paste as 'options.opt' and then re-start the editing process ***

```
[global_runtime]
MINIMALRESPONSE:YES
      # minimal system echoes back for pulling in data and spitting out
      # data to the opc server.
      # -- use YES for minimal
      # -- use NO for verbose

[global_throttle]
GROUPBUILD_TIMEOUT_OVERRIDE:30
      # timeout value, in seconds.
      # -- suggested as 30 seconds or greater.

[global_network]
MYIP:10.1.12.253
      # what is your CU static IP address?
MYDEFAULTGATEWAY:10.1.12.1
      # what is your CU default gateway IP address?

[global_openopc]
OPENOPC_TIE_IN:GATEWAY
      # do not modify unless building custom routine

[auto_launch]
AUTO_LAUNCH:READ,mypreset1,30|BRIDGE,mypreset2,60|
      # define mod_openopc routines to auto launch
      # -- form is ...
      # COMMAND,preset-file-name,argument1,argument2,argument3|
      # COMMAND,preset-file-name,argument1,argument2,argument3|
      # -- MUST INCLUDE TRAILING PIPE!
      # -- DO NOT use any spaces.
      # -- if a COMMAND does not have any arguments, then it would
      #    simply be...
      # COMMAND,preset-file|COMMAND,preset-file,argument1|
      # COMMAND,preset-file,argument1,argument2|
      #     ... you get the idea.
      #     ... ultimately everything between 'pipes' is passed as a
      #         newly spawned process
      #         where commas are replaced by whitespace.  So, the
      #         example above would be passed as...
      #         --> mod_openopc.py COMMAND preset-file
      #         --> mod_openopc.py COMMAND preset-file argument-1
      #         --> mod_openopc.py COMMAND preset-file argument-1 argument-2

[gateway_reset]
GATEWAY_LIST_TO_RESET:thundercracker,30,24
      # define mod_openopc gateways to auto reset
      # -- form is GWNAME,DISCONNECT-DELAY,RECYCLE-FREQUENCY-IN-
      #    HOURS|GWNAME2,DISCONNECT-DELAY2,RECYCLE-FREQUENCY-IN-
      #    HOURS2
      # -- DO NOT use any spaces.
      # -- DO NOT use a trailing pipe.
      # -- for example...
      #     if you have 2 gws, gw1 and gw2, and you want to reset
      #     gw1 every 2 days with a delay of 30 seconds to allow
      #     clients to clean up, and gw2 every 1 day with a 60   second
      #     disconect delay, then you'll use...
      #           gw1,30,48|gw2,60,24
```

Setup a MySQL database for your mod_openopc data to be stored in, and be sure to grant privileges to a MySQL user with a password:

```
      mysql> CREATE DATABASE mod_openopc;
      mysql> GRANT ALL ON mod_openopc.* TO vince@localhost IDENTIFIED BY
              “wesmokekools”;
      mysql> GRANT ALL ON mod_openopc.* TO vince@* IDENTIFIED BY
              “wesmokekools”;
      mysql> FLUSH TABLES;
```

Your MySQL server should be setup by a qualified DB administrator, but in addition to the 'usual' configuration steps, the MySQL 'my.ini' (WIN) or 'my.cnf' (UNIX) configuration file MUST include the following flags...

```
      old_passwords=1
      set-variable = wait_timeout=150000
      set-variable = interactive_timeout=150000
```

** THESE FLAGS ARE REQUIRED TO ALLOW mod_openopc TO SELF-POLICE A CONNETION TIMEOUT versus an OPC_TOPIC BEING DOWN.

Setup a new database preset in mod_openopc;

Copy ./options/sql_configs/template.sql to a new database preset name, in case of our example, above, we'll copy to...

```
./options/sql_configs/mod_openopc.sql
```

… and yes, this is all case sensitive and the preset name must match the sql db name.

```
[sql_server_configs]
MYSQLDB:mod_openopc
      # DATABASE NAME
MYSQLIP:localhost
      # IPADDRESS, should be "localhost" or actual IP address
      # -- MAY HAVE TO USE "QUOTES" FOR IF RUNNING THIS PROGRAM UNDER
      #    UNIX
      # -- DO NOT USE QUOTES IF RUNNING UNDER WIN
MYSQLFAULT:mod_openopc
      # DATABASE THAT HOLDS FAULT TABLE
FAULTTABLENAME:system_faults
      # FAULT TABLE NAME
MYSQLUSER:vince
      # YOUR MYSQL USERNAME
MYSQLPASS:wesmokekools
      # YOUR MYSQL PASSWORD
COMMITTRANSACTIONS:YES
      # COMMIT DATABASE TRANSACTIONS AFTER EXECUTION
      # -- YES or NO
      # -- TRANSACTIONAL DATABASES SUCH AS INNODB REQUIRE
      #    THIS FUNCTIONALITY, FUTURE MYISAM WILL ALSO
      # -- 'STOCK' MyISAM databases as of January 2010 do not require
      #     this.
MYSQLRETENTION:4
      # DB RETENTION TIME FOR RECORDS IN YEARS
FIELDRETENTION:DATESTAMP
      # FIELD TO CARRY OUT RETENTION QUERY ON (what we judge age by)
MYSQLMAINTTABLES:TABLENAME1|TABLENAME2|system_faults
      # TABLE NAMES (CASE SENSITIVE) TO PERFORM PERIODIC
      # MAINTENANCE ON WHEN CALLING 'MAINT_DB' FUNCTION
      # -- DO NOT INCLUDE ANY TABLES THAT ARE SUPPOSED
      #    BE STATIC
      # -- DO NOT INCLUDE TRAILING PIPE!
      # -- FORM IS TABLE1|TABLE2|TABLE3|system_faults
      # -- -- NO QUOTES, NO SPACES (TABLE NAMES MUST BE SINGLE STRING
      #       WITH NO SPACES)
      # -- -- SHOULD ALWAYS INCLUDE system_faults
      #       UNLESS YOU ARE RUNNING A CUSTOM SCHEME
```

Setup a new server (OPC Server) preset in mod_openopc...

Copy ./options/server_configs/template.opc to a new opc server preset name. Let's continue with our example, and assume that we have a virtual guest OS running Windows with an OPC Server and the Gateway. And let's say that we're going to call that “opc_server_1”. So, copy...

```
./options/server_configs/template.opc
                         ← to →
                              ./options/server_configs/opc_server_1.opc
```

```
      [opc_server_configs]
      IP_OF_GATEWAY_FOR_SERVER:10.1.12.250
            # OPC SERVER IP ADDRESS
      SERVER_NAME:RSLinx OPC Server
            # OPC SERVER SOFTWARE NAME WHEN QUERIED
      SERVER_TEST:[TEST]ST9:0
            # DESIGNATED TEST PLC AND TAG TO READ
            # —- ASSIGN A STRING OR VALUE TO SOME KNOWN PLC OR OPC
            #    DEVICE SO THAT 'TEST' BATTERY CAN RUN AGAINST IT.
      MINIMUM_SCAN_INTERVAL:5
            # INTEGER INDICATES SMALLEST AMOUNT
            #   OF TIME BETWEEN READ ROUTINES THAT
            #   ARE ON A SCHEDULE SO WE DON'T
            #   POUND A SERVER UNDER HEAVY LOAD
      OPC_DEVICENAME_START_TRIM:1
      OPC_DEVICENAME_END_TRIM:1
            # NUMBER OF CHARACTERS TO TRIM FROM THE
            #   START AND END OF AN OPC DEVICE NAME
            #   FOR EXAMPLE...
            #   -  (RSLinx)
            #      [MYPLC1]N7:10
            #      - DEVICE NAME IS ACTUALLY "MYPLC1"
            #        NOT [MYPLC1], SO WE SET VALUE OF "1"
            #        FOR START_TRIM and VALUE OF "1" FOR
            #        END_TRIM.
            #   -  (Kepware)
            #       CH1.MYPLC1.N7:10
            #       - DEVICE NAME IS ACTUALLY "MYPLC1"
            #         NOT "CH1.MYPLC1.", SO WE SET VALUE
            #         OF "4" FOR START_TRIM AND VALUE OF
            #         "1" FOR END_TRIM.
      SERVER_STOP_CMD_LINE_INPUT:net stop RSLinx
            # COMMAND ISSUED VIA COMMAND LINE (SHELL, POWER SHELL, DOS
            # PROMPT / ETC) ON REMOTE MACHINE TO STOP OR KILL THIS OPC
            # SERVER.
            #   - enter "none" if not using SERVER_RESTART_WITH_GATEWAY
      SERVER_START_CMD_LINE_INPUT:net start RSLinx
            # COMMAND ISSUED VIA COMMAND LINE (SHEL, POWER SHELL, DOS
            # PROMPT / ETC) ON REMOTE MACHINE TO START UP THIS OPC
            # SERVER.
            #   - enter "none" if not using SERVER_RESTART_WITH_GATEWAY
      SERVER_RESTART_WITH_GATEWAY:yes
            # WOULD YOU LIKE TO RESTART THIS SERVER (STOP / START)
            # ALONG WITH THE GATEWAY, WHEN DOING A GATEWAY_RESET ?
            #     - "yes" OR "no"
```

Setup a fault container:

*** If you are using the S.E.E.R. II front-end, then DO NOT PERFORM THIS STEP. Instead, proceed to S.E.E.R. Install and configuration, at which point you can create the fault container automatically when creating the 'mod_openopc Database' from the Settings Tab. ***

mod_openopc needs a table created within itself called 'system_faults' . This is where any problems encountered (typically communication faults) are logged when any instance of mod_openopc is working with an opc server that is having data logged. So, continuing with our example...

```
mysql> USE mod_openopc;
mysql> CREATE TABLE system_faults(DATESTAMP VARCHAR(20),
            INDEX(DATESTAMP),
            TYPE VARCHAR(15), INDEX(TYPE),
            ROUTINE VARCHAR(55), INDEX(ROUTINE),
            PARTNER VARCHAR(20), INDEX(PARTNER),
            ACKNOWLEDGED VARCHAR(30), INDEX(ACKNOWLEDGED)
            );
```

If you choose to use the InnoDB database engine (allows hot backups – consult your database admin first – but yes, we DO RECOMMEND IT!)...

```
mysql> ALTER TABLE system_faults CHANGE ENGINE=InnoDB;
```

Open up ./options/presets/system_faults.pre and edit as follows:

```
[your_server]
YOUROPCSERVER:none
      # required placeholder
YOURSQLSERVER:mod_openopc
      # preset sql server name
YOURSQLTABLE:system_faults
      # preset sql table
COMMENTENABLE:no
      # required placeholder
YOURSQLCOMMENTTABLE:none
      # required placeholder
YOURSQLFILLERCOUNT:0
      # required placeholder
YOURSQLCOLUMNCOUNT:0
      # required placeholder

[your_read]
YOURLEAFERS:none
      # required placeholder
```

SETUP YOUR OPC SERVERS AND THE GATEWAY

Before we can go any further, we must setup an OPC Server and mod_openopc_2 in GATEWAY_DAEMON mode on our guest (or other machine) Windows based OS.

Assuming you have already setup a WinXP or Server'03 guest OS (*32 bit is advised for OPC Server support – at least until the industry catches up with modern computers. mod_openopc is fine regardless of architecture.*) you can proceed as follows...

> *** Be aware, this section is where we get the information to enter into the previous section, where we talk about “Setup a New OPC Server Preset in mod_openopc”... ***

Install your OPC Server software (RSLinx, Kepware, Matrikon, etc...)

MAKE SURE YOUR OPC SERVER SOFTWARE IS SETUP AS A WINDOWS SERVICE! (else it must be manually started at boot PRIOR to mod_openopc in GATEWAY_DAEMON mode)

Give your guest OS (or physical machine) a unique IP address.

Following our example, let's use 10.1.12.250.

In your OPC Server Setup, create a new TOPIC called “opc_server_1_test”.

For further examples and to allow us to show you other functions later, let us also create a second TOPIC called “plc_hydraulic_system”, and map that to a PLC at some address, let's say 10.1.12.60 (obviously, use a real address of one of your real PLC's).

Copy the entire mod_openopc distribution over to your OPC Server as... C:\mod_openopc_2

Install the Automation DLL of your choice (if you have not already). If you do not have one, or do not have a supported DLL, you can use the included GrayBox DLL:

Open a command prompt (DOS Shell → Start / Run / cmd)...

```
C:\>  cd C:\mod_openopc_2\help\dependancies\automation_dll
C:\>  regsvr32 graybox_opc_automation_v-1-2-8-508.dll
```

** Command is appropriate for Win2k / WinServer2003 / WinXP
** You should check for Win7 / WinServer2008 (PowerShell)

Go to Start → Control Panel → Administrative Tools → Services …

Find the service for your OPC Server Software (RSLinx, Kepware, etc...) and edit properties as follows:

- startup type =	AUTOMATIC
- enable / disable =	ENABLED
- 1st failure =	RESTART SERVICE
- 2nd failure =	RESTART SERVICE
- Subsequent failure =	RESTART SERVICE

Click “OK” to save your changes.

Create a shortcut to the following, and place it in your Windows START folder so that it may be launched when your OPC Server Operating System starts up...

```
C:\[python]\python.exe C:\mod_openopc_2\prog\mod_openopc.py GATEWAY_DAEMON
```

This will monitor via network port 7767, on all interfaces, for valid mod_openopc requests for reset by the SERVER_RESET subroutine running on your primary (in this example, Linux) machine.

** *Note, it is highly suggested that you set the OPC Server guest WIN OS to auto-login on boot. Refer to your Windows documentation on how to do this...* **

Create a USERNAME / PASSWORD for auto-login under Windows...

Go to START → CONTROL PANEL → USERS.

Create User =	modopenopc
w/password =	communicate
w/level =	Administrator

You should now reboot your WIN guest OS or machine. When it boots up, you will see the OPC Server automatically start up, and you will also see the mod_openopc GATEWAY_DAEMON... It will launch the Gateway Service about 30 seconds after it loads itself. The windows will not close, and you should NOT close it. It needs to run constantly. If you're firewalled, make sure ports #7766 and #7767 are open for both TCP/UDP and RAW connections, as we've yet to decide on a formal protocol, so if you jump versions, the comm type might change.

USING PRESETS

The best way to use mod_openopc is with preset files.

Preset / Command
BRIDGE →

Allow you to READ a value from one opc device and write that value back to another opc device on the same opc server. This takes the place of (for example) PLC Messaging, and allows you to bridge networks that otherwise couldn't 'talk' to each other, such as Allen Bradley Data Sidewalk and Allen Bradley Ethernet/IP.

```
In ./options/presets,
      copy ./template/template_bridge.brg to...

      ./options/presets/my_first_bridge.brg

Edit my_first_bridge.brg as follows...

[your_server]
```

```
YOUROPCSERVER:grimlock
DATA_SOURCE:cache
      # CHOICES ARE 'cache' or 'hybrid' (CASE SENSITIVE)
      # -- hybrid USES DIRECT DEVICE READS (HIGH CPU USAGE)
      # -- cache USES OPC SERVER OPTIMIZED CACHE READS (BEST
      #    PERFORMANCE)
YOURSQLSERVER:modopenopc
      # SQL SERVER JUST FOR LOGGING FAULTS

[your_bridge]
YOURLEAFERS:XXX[MY_PLC_1]YYYF11:11&XXX[MY_PLC_1]YYYF11:41&|
YOURLEAFERS2:XXX[MY_PLC_2]YYYF11:48&XXX[MY_PLC_2]YYYF11:49&|
YOURBRIDGELENGTH:2
      #      YOURLEAFERS - the SOURCE data points or registers
      #      YOURLEAFERS2 - the TARGET data points or registers
      #
      #      preset leaves to bridge, follow form ...
      #        "LEAFIDENTIFICATION&|
      #      where & delineates columns in the table and |
      #         delineates rows
      #      TYPICALLY, LEAFS ARE IN THE FORM OF ...
      #             XXX[TARGETNAME]YYYTAG
      #      SUCH AS...
      #             XXX[MY_PLC]YYYN7:42
      #
      #      BE ADVISED... YOU MUST ENSURE REGISTER (TAG)
      #        COMPATABILITY BEFORE YOU BRIDGE.  mod_Open_Opc
      #         will not check!
      #             For Example...
      #                   x - Integers can copy to Floats
      #                   x - Floats can't copy to Integers
      #                          (they should round though)
      #                   x - DINT's can copy to INT's
      #                   x - INT's can't copy to DINT's
      #                   x - STRING can copy to STRING
      #                   x - STRING can't copy to anything else
      #                       you get the idea...
      #                   x - BIT is cleanest at word level but
      #                      can be discrete
      #                   x - INT and DINT should copy at word
      #                      level but you can
      #                      perform discrete copies as well.
      #                   x - ELEMENTS (such as T4:10.PRE) can
      #                          copy directly, you don't have to
      #                          bridge the entire Timer.
      #
      #      BRIDGE LENGTH ... the number of leaves in a bridge
      #        set.  All sets must be of the same length, or else
      #        you will have to not 'use' sets, and, rather
      #        just have one very big set.  Using sets gives
      #        you fault detection and the ability to jump
      #        over leaves attached to equipment that may be
      #        powered down or not working.
      #        A set is the full string of leaves up until
      #        the '&|' separator.  Counting starts
      #        at 1, not zero, so if you have 9 items, then
      #        list it as "9", not "8".
```

Save the file, and you can now launch it with...

```
[root@server] ./prog/mod_openopc.py BRIDGE [preset file basename] \
[root@server] [scan_interval] [OVERRIDE | --blank-- ]
```

Where...

```
preset file basename =  my_first_bridge
scan interval =         seconds (suggest 2 to 10) to wait between each
                        bridge execution
overrride =             OVERRIDE (or blank), this allows us to override
                        the minimum scan interval variable from the
                        mod_openopc options.opt file.  If you choose a
                        scan time of less than the minimum scan
                        interval, but do not specify OVERRIDE, then the
                        minimum scan interval will remain in effect.
```

Example real world call...

```
[root@server] ./prog/mod_openopc.py BRIDGE my_first_bridge 2  OVERRIDE
```

Preset / Command
SPACE_BRIDGE →

Allow you to READ a value from one opc device and write that value back to another opc device on A DIFFERENT opc server. This takes the place of (for example) PLC Messaging across expensive 3rd party Gateway Devices, or simply enables messaging where it was otherwise impossible. Networks that otherwise couldn't 'talk' to each other, such as Allen Bradley Ethernet/IP and Modicon MODBUS.

```
In ./options/presets,
      copy ./template/template_space_bridge.sbrg to...

      ./options/presets/my_first_spacebridge.sbrg
```

Edit my_first_spacebridge.sbrg as follows...

```
[your_server]
YOUROPCSERVER:grimlock
DATA_SOURCE:cache
      # CHOICES ARE 'cache' or 'hybrid' (CASE SENSITIVE)
      # -- hybrid USES DIRECT DEVICE READS (HIGH CPU USAGE)
      # -- cache USES OPC SERVER OPTIMIZED CACHE READS (BEST
      #    PERFORMANCE)
YOURSQLSERVER:modopenopc
      # SQL SERVER JUST FOR LOGGING FAULTS
YOURSPACEBRIDGE:grimlock_hmi
      # mod_openopc WRITE_DAEMON instance where the
      #    gathered data shall be exported to.

[your_bridge]
YOURLEAFERS:XXX[MY_PLC_1]YYYF11:11&XXX[MY_PLC_1]YYYF11:41&|
YOURLEAFERS2:XXX[MY_PLC_2]YYYF11:48&XXX[MY_PLC_2]YYYF11:49&|
YOURBRIDGELENGTH:2
      #      YOURLEAFERS - the SOURCE data points or registers
      #      YOURLEAFERS2 - the TARGET data points or registers
      #
      #      preset leaves to bridge, follow form ...
      #        "LEAFIDENTIFICATION&|
      #      where & delineates columns in the table and |
      #         delineates rows
      #      TYPICALLY, LEAFS ARE IN THE FORM OF ...
      #             XXX[TARGETNAME]YYYTAG
      #      SUCH AS...
      #             XXX[MY_PLC]YYYN7:42
      #
      #      BE ADVISED... YOU MUST ENSURE REGISTER (TAG)
      #        COMPATABILITY BEFORE YOU BRIDGE.  mod_Open_Opc
      #        will not check!
      #             For Example...
      #                   x - Integers can copy to Floats
      #                   x - Floats can't copy to Integers
      #                         (they should round though)
      #                   x - DINT's can copy to INT's
      #                   x - INT's can't copy to DINT's
      #                   x - STRING can copy to STRING
      #                   x - STRING can't copy to anything else
      #                       you get the idea...
      #                   x - BIT is cleanest at word level but
      #                      can be discrete
      #                   x - INT and DINT should copy at word
      #                      level but you can
      #                      perform discrete copies as well.
      #                   x - ELEMENTS (such as T4:10.PRE) can
      #                          copy directly, you don't have to
      #                          bridge the entire Timer.
      #
      #      BRIDGE LENGTH ... the number of leaves in a bridge
      #        set.  All sets must be of the same length, or else
      #        you will have to not 'use' sets, and, rather
      #        just have one very big set.  Using sets gives
      #        you fault detection and the ability to jump
      #        over leaves attached to equipment that may be
      #        powered down or not working.
      #        A set is the full string of leaves up until
      #        the '&|' separator.  Counting starts
      #        at 1, not zero, so if you have 9 items, then
      #        list it as "9", not "8".
```

Save the file, and you can now launch it with...

```
[root@server] ./prog/mod_openopc.py SPACE_BRIDGE \
[root@server] [preset file basename] [scan_interval] \
[root@server] [OVERRIDE | --blank-- ]
```

Where...

```
preset file basename =  my_first_spacebridge
scan interval =         seconds (suggest 2 to 10) to wait
                        between each bride execution
overrride =             OVERRIDE (or blank), this allows us
                        to override the minimum scan interval
                        variable from the mod_openopc
                        options.opt file.  If you choose a scan
                        time of less than the minimum scan
                        interval, but do not specify OVERRIDE,
                        then the minimum scan interval will
                        remain in effect.
```

Example real world call...

```
[root@server] ./prog/mod_openopc.py SPACE_BRIDGE my_first_bridge 2  OVERRIDE
```

Preset / Command
WRITE →

Allow you to WRITE a value from a preset file to an OPC Device. This is excellent for loading recipes, which can take doing a machine product changeover from editing 20 different settings to simply allowing an operator to press one button and let mod_openopc do the rest.

```
In ./options/presets,
      copy ./template/template_write.wrt to...

      ./options/presets/my_first_write.wrt
```

Edit my_first_write.wrt as follows...

```
[your_server]
YOUROPCSERVER:opcname
      # preset opc server name
YOURSQLSERVER:sqlname
      # preset sql server name
      # strictly for reporting faults

[your_write]
YOURLEAFERS:LEAFIDENTIFICATION&VALUETOWRITE&|
             2NDLEAF&VALUETOWRITETO2NDLEAF&|
      #      preset leaves to bridge, follow form
      #         "LEAFIDENTIFICATION&|
      #      where & delineates columns in the table and |
      #      delineates rows
      #
      #      TYPICALLY, LEAFS ARE IN THE FORM OF ...
      #             XXX[TARGETNAME]YYYTAG
      #      SUCH AS...
      #             XXX[MY_PLC]YYYN7:42
      #      SO...
      #             XXX[MY_PLC]YYYN7:42&2.5&|   ... would
      #            write value 2.5 to leaf N7:42 of MY_PLC
      #
```

Save the file, and you can now launch it with...

```
[root@server] ./prog/mod_openopc.py WRITE [preset file basename] [scan_interval]
```

Where...

```
preset file basename =  my_first_write
scan interval =         time, in seconds for a timeout
                        reference, this may eventually be
                        eliminated, as it serves very little
```

purpose. Typically, set to 30 seconds... if it takes more than 30 seconds for a WRITE, then something is severely wrong with your setup or network.

Example real world call...

```
[root@server] ./prog/mod_openopc.py WRITE my_first_write 30
```

Preset / Command
READ →

Allow you to READ, repeatedly, from an OPC Device, and log that data to a MySQL database table.

In ./options/presets,
copy ./template/template_read.pre to...

```
./options/presets/my_first_read.pre
```

Edit my_first_read.pre as follows...

```
[your_server]
YOUROPCSERVER:opcname
      # preset opc server name
DATA_SOURCE:cache
      # CHOICES ARE 'cache' or 'hybrid' (CASE SENSITIVE)
      # -- hybrid USES DIRECT DEVICE READS (HIGH CPU USAGE)
      # -- cache USES OPC SERVER OPTIMIZED CACHE READS (BEST
      #    PERFORMANCE)
YOURSQLSERVER:sqlname
      # preset sql server name
YOURSQLTABLE:sqltable
      # preset sql table
COMMENTENABLE:no
      # 'yes' or 'no', if yes, then set
      #  below to presetname_comment, or name
      #  of table holding the comments.
      #  if no, set below to 'none'
YOURSQLCOMMENTTABLE:none
      # preset sql table
      # correspondes to comments for this preset.
      # comment column counts as a filler column,
      # so do NOT include a comment column as
      # part of the sqlcolumncount, rather, count
      # it as part of the fillercount.
YOURSQLFILLERCOUNT:0
      # integer number, corresponding to the
      # number of empty cells of columns left
      # at the right hand side of your table.
      # this is useful for inserting partial
      # records.  set to 0 if you're not
      # using it, or if your leafers will
      # fill up entire row in table.
YOURSQLCOLUMNCOUNT:0
      # integer number, corresponding to the
      # number of columns per row of your sql
      # table... note that this requires all
      # rows to fill with the same number of
      # columns. DO NOT INCLUDE FILLER COLUMNS!
[your_read]
YOURLEAFERS:LEAFIDENTIFICATION&ANOTHERLEAF&|
                         2NDROWLEAF1&2NDROWLEAF2&|
      #     preset leaves to read, follow form
      #           LEAFIDENTIFICATION&|
      #     where & delineates columns in the table and |
      #       delineates rows.

      #     TYPICALLY, LEAFS ARE IN THE FORM OF ...
      #           XXX[TARGETNAME]YYYTAG
      #     SUCH AS...
      #           MY_FAVORITE_PLC, TAG N7:42 would be
      #           listed as
      #           XXX[MY_FAVORITE_PLC]YYYN7:42
      #     FOR EXAMPLE...
      #           YOURLEAFERS:XXX[MY_PLC_1]YYYN7:20&
      #                             XXX[MY_PLC_1]YYYN7:21&|
```

Save the file, and you can now launch it with...

```
root@server] ./prog/mod_openopc.py READ [preset file basename] [scan_interval]
```

Where...

```
preset file basename =  my_first_read
scan interval =         time interval of scans (every X
                        seconds)
```

Example real world call...

```
[root@server] ./prog/mod_openopc.py READ my_first_read 30
```

Additionally, any 'READ' preset file may be modified with the addition of the following lines to the [your_server] section...

```
      YOURSQLCOLUMNDATESTAMP:DATESTAMP
            # case sensitive name of the column in your
            # mysql database which contains the datestamp
            # of records logged - this should be the first
            # column!
      YOURSQLCOLUMNLEAFNAME:MACHINENAME
            # case sensitive name of the column in your
            # mysql database which contains the leaf_name
            # (also called 'leaf_sql_name' or 'machine name')
            # of records logged - this should be the second
            # column, and will be easily identified as it is
            # the column where all of your OPC_TOPIC names
            # are logged.
```

This will allow you to call the 'READ' subroutine with the additional command line argument of 'UPDATE', for example...

```
root@server] ./prog/mod_openopc.py READ [preset file basename] /
root@server] [scan_interval] UPDATE
```

Where...

```
preset file basename =  my_first_read
scan interval =         time interval of scans (every X
                        seconds)
argument =              UPDATE
```

Example real world call...

```
[root@server] ./prog/mod_openopc.py READ my_first_read 30 UPDATE
```

The result of the 'UPDATE' argument will be that only ONE record will be maintained for any given LEAFNAME (also called 'machine name' or 'leaf_sql_name' – this is the value of the OPC Topic name you declare in the list of values to be read). This is useful for HMI's or other interfaces where you are either already logging data in a different format, or you simply want to display a bunch of different data points that you do not necessarily want to record. When building HMI's – it is best to pair 'READ' (with 'UPDATE') along with 'WRITE_DAEMON' (where appropriate flat files are dumped as needed.

Preset / Command
READ_ONE_SHOT →

Exactly the same as READ, except it is only performed *once*.

Make and edit a presetfile the same as for a READ...

Save the file, and you can now launch it with...

```
[root@server] ./prog/mod_openopc.py READ_ONE_SHOT [preset file basename]  \
[root@server] [scan_interval]
```

Where...

```
preset file basename =  my_first_read_one_shot
scan interval =         time, in seconds for a timeout
                        reference, this may eventually be
                        eliminated, as it serves very little
                        purpose.  Typically, set to 30 seconds...
                        if it takes more than 30 seconds for a
                        READ, then you may have a loaded system,
                        or a bottleneck somewhere.
```

Example real world call...

```
[root@server] ./prog/mod_openopc.py READ_ONE_SHOT my_first_read_one_shot 30
```

Preset / Command
READ_DAEMON →

Once launched, this deamon scans a folder for file system events. When one occurs, the new file is parsed for information, and then a READ_ONE_SHOT is performed using a regular READ preset, which has been defined by the file system event.

Let's say we want to create a READ_DAEMON for the opc server that we call “opc_server_1”.... .... For this, we will choose to accept flat files in /[path]/mod_openopc_2/gwcomm/opc_server_1_read

*** NOTE – you have the option of creating the folder (in our example “opc_server_1_read” under the /[path]/mod_openopc_2/gwcomm/ directory yourself, and giving it permissions of 0777; or mod_openopc will create it for you the first time you run this command with your new 'my_first_read_daemon' preset. mod_openopc will continually check for and create if necessary this folder ***

```
      In ./options/presets,
            copy ./template/template_read_daemon.rdm to...

            ./options/presets/my_first_read_daemon.rdm

      Edit my_first_read_daemon.rdm as follows...

      [your_server]
      YOUROPCSERVER:name_of_opc_server
            # preset opc server name
      YOURSQLSERVER:name_of_sql_server
            # preset sql server name
            # strictly for reporting faults

      [your_daemon]
      YOURDAEMON:subdirectory_in_GWCOMM_directory
            # where flatfiles will be exported to by
            # other programs, in order to call for a read.
```

Save the file, and you can now launch it with...

```
[root@server] ./prog/mod_openopc.py READ_DAEMON [preset file basename]
```

Where...

preset file basename = my_first_read_daemon

Example real world call...

```
[root@server] ./prog/mod_openopc.py  READ_DAEMON my_first_read_daemon
```

```
      FLATFILE form for event files is...

            # START READ_DAEMON EVENT FILE
            [your_read]
            YOURREAD:any_read_preset_basename
            #-- READ PRESET FROM FILE
            # END OF FILE

            Where all end-of-lines are assumed to be UNIX friendly
            “\n”.
```

Preset / Command
WRITE_DAEMON →

Once launched, this deamon scans a folder for file system events. When one occurs, the new file is parsed for information, and then a WRITE is performed using EITHER OF a regular WRITE preset, which has been defined by the file system event, or DYNAMIC values, declared by the file system event.

Let's say we want to create a WRITE_DAEMON for the opc server that we call “opc_server_1”.... For this, we will choose to accept flat files in /[path]/mod_openopc_2/gwcomm/opc_server_1_write

*** NOTE – you have the option of creating the folder (in our example “opc_server_1_write” under the /[path]/mod_openopc_2/gwcomm/ directory yourself, and giving it permissions of 0777; or mod_openopc will create it for you the first time you run this command with your new 'my_first_read_daemon' preset. mod_openopc will continually check for and create if necessary this folder ***

```
In ./options/presets,
      copy ./template/template_write_daemon.wdm to...

      ./options/presets/my_first_write_daemon.wdm
```

Edit my_first_read_daemon.rdm as follows...

```
[your_server]
YOUROPCSERVER:name_of_opc_server
        # preset opc server name
YOURSQLSERVER:name_of_sql_server
        # preset sql server name
        # strictly for reporting faults

[your_daemon]
YOURDAEMON:subdirectory_in_GWCOMM_directory
        # where flatfiles will be exported to by
        # other programs, in order to call for a WRITE.
```

Save the file, and you can now launch it with...

```
[root@server] ./prog/mod_openopc.py WRITE_DAEMON [preset file basename]
```

Where...

```
      preset file basename =  my_first_write_daemon
```

Example real world call...

```
[root@server] ./prog/mod_openopc.py  WRITE_DAEMON my_first_write_daemon
```

FLATFILE form for event files is... { DECLARED VALUES }

```
      # START WRITE_DAEMON EVENT FILE
      [your_write_type]
      YOURWRITETYPE:DECLARED
      # -- WRITE DECLARED VALUES TO OPC TARGET
      [your_leafers]
      YOURLEAFERS:LEAFIDENTIFICATION&VALUETOWRITE&|
                  2NDLEAF&VALUETOWRITETO2NDLEAF&|
      YOURWRITEPRESET:NONE
      # -- NAME OF PRESET FILE TO WRITE
```

```
# END OF FILE
```

Where all end-of-lines are assumed to be UNIX friendly "\n".

FLATFILE form for event files is... { PRESET VALUES }

```
# START WRITE_DAEMON EVENT FILE
[your_write_type]
YOURWRITETYPE:PRESET
# -- WRITE DECLARED VALUES TO OPC TARGET
[your_leafers]
YOURLEAFERS:NONE
YOURWRITEPRESET:any-existing-mod_openopc-WRITE-preset-basename
# -- NAME OF PRESET FILE TO WRITE
# END OF FILE
```

Where all end-of-lines are assumed to be UNIX friendly "\n".

Preset / Command
MAINT_DB →

Performs daily maintenance on your MySQL database tables. Deletion and cleanup of NULL records and deletion of records that are older than the "RETENTION TIME" declared in your SQL configuration file. In the least, perform once weekly.... schedule it with WIN task scheduler or CRON in UNIX.

Tables to be cleaned up are pulled from the global options file (options.opt), as whatever you've declared them to be.

```
[root@server] ./prog/mod_openopc.py MAINT_DB [sql_preset_name] \
[root@server] [OPTIMIZE |-blank- ]
```

Where...

OPTIMIZE executes a full re-index / re-order optimization on a table, which takes a LOT of horsepower, a lot of time, and is mostly unnecessary.

Example real world call...

```
[root@server] ./prog/mod_openopc.py  MAINT_DB  mod_openopc
```

Preset / Command
AUTO_LAUNCH →

Launches all of your mod_openopc presets, and server daemons for all of your opc servers (you decalred which ones you wanted auto launched in the global options [options.opt] file – see section at the bottom 'Auto Launch').

This works wonderfully for system startup, as you can add it to your 'rc.local' start file or your START folder.

***On Linux / Unix flavors, it should be called with...*

```
[path-to]nohup [path-to]python [path-to]mod_openopc.py AUTO_LAUNCH CONFIRM 2>/dev/null 1>/dev/null &
```

*… this will properly fork the entire process and all subthreads to the background, and out of your console window (or the startup TTY).***

```
[root@server] ./prog/mod_openopc.py AUTO_LAUNCH [CONFIRM | -blank- ]
```

Where...

CONFIRM actually executes the auto start function...
anything else, or blank, will result in simply displaying what WOULD HAVE BEEN DONE; which is nice if you're looking to make sure your updates to a running system have gone through as your intended.

Example real world call (such as that which would be added to a startup file)...

```
[root@server] /usr/bin/nohup /usr/local/bin/python \
[root@server] /opt/mod_openopc_2/prog/mod_openopc.py \
[root@server] AUTO_LAUNCH CONFIRM 2>/dev/null 1>/dev/null &
```

Preset / Command
GATEWAY_RESET →

Peforms a systemwide, controlled, reset of the mod_openopc Gateway which runs on the guest (or other physical system) OS. This is used as a preventative measure   to deal with the pywin32-to-OPC Automation DLL memory leak under Windows. It is a workaround, not a fix.

In most installations, you can get away with a once weekly reset. However, just as a precaution, we recommend resetting once PER DAY. This does not restart any computer OS. It simply drops the Gateway Service. All client threads (instances) of mod_openopc are put into a hold state, where they disconnect from their respective Gateways, wait for the cycle, and then come back up. The whole process takes anywhere from 30 seconds to a minute or so, depending on what you set your “delay” variable to.

Be advised, the “delay” variable should be set to no less than 15 seconds, and mod_openopc enforces this! - set it to “2” and mod_openopc will auto-bump it up to 15.

*** Calling the GATEWAY_RESET function can be be done manually, however the SERVER_DAEMON will do this for you, auto-magically at your desired interval if you've set up your options.opt file correctly! ***

Structure...

```
[root@server] ./prog/mod_openopc.py GATEWAY_RESET [opc_server_preset] [delay]
```

Example real world call...

```
[root@server] ./prog/mod_openopc.py  GATEWAY_RESET myopcserver1 25
```

Preset / Command
GATEWAY_RESET_DAEMON →

Auto-magically calls GATEWAY_RESET subroutine on the specified OPC Server at the specified interval, so that you don't have to schedule it as a job.

This subroutine is invoked by AUTO_LAUNCH on all of the OPC Servers you declared in the global options (options.opt) file. You can, however, invoke it manually. Be advised, though, that you should NEVER have two instances of this daemon running for the same OPC Server!

Structure...

```
[root@server] ./prog/mod_openopc.py SERVER_DAEMON \
[root@server] [opc_server_preset] [delay] [recycle time]
```

Where...

```
opc_server_preset  =    name used for opc server *.opc
                  preset file, without extension.
delay =           delay time to allow running subroutines to
                  disconnect from the OPC Server before
                  resetting it (in seconds)
                        * must be >= 15
recycle_time =    hours between resets... so for a daily
                  reset, this would be “24”, or for a weekly
                  reset this would be “168” (in hours)
```

Example real world call...

```
[root@server] ./prog/mod_openopc.py SERVER_DAEMON \
[root@server] thundercracker 25 48
```

Preset / Command
GATEWAY_DAEMON →

Runs ONLY on the OPC Server virtual guest OS (or dedicated machine), and listens for network communication from SERVER_RESET subroutines.

Resets mod_openopc Gateway.

*** You do NOT need a completed global options (options.opt) file in order to run this subroutine. It will function “all on its own”. The comm part is 7767.*

*** Make sure ports 7766 and 7767 are OPEN for all protocols between your OPC Server virtual guest (or dedicated machine) and the Primary Machine running mod_openopc.*

*** If you –REALLY-- want to change these ports, you can do so in editing mod_openopc_common.py, and must be done on both OS instances. This is NOT recommended!*

Structure / Call...

```
[root@server] ./prog/mod_openopc.py  GATEWAY_DAEMON
```

Preset / Command
HELP →

Displays the help file with all availabe commands, switches, and their structure. Similar to this section, but less fleshed out. It's you're basic help command.

Structure / Call...

```
[root@server] ./prog/mod_openopc.py  HELP
```

Preset / Command
SETTINGS →

Displays all settings within your global options (options.opt) file, as you have entered them, for your reference or for debugging of your preset files.

Structure / Call...

```
[root@server] ./prog/mod_openopc.py  SETTINGS
```

Preset / Command
TEST_FOR_ECHO →

Peforms a health assessment on your installation, this is an invaluable troubleshooting tool!

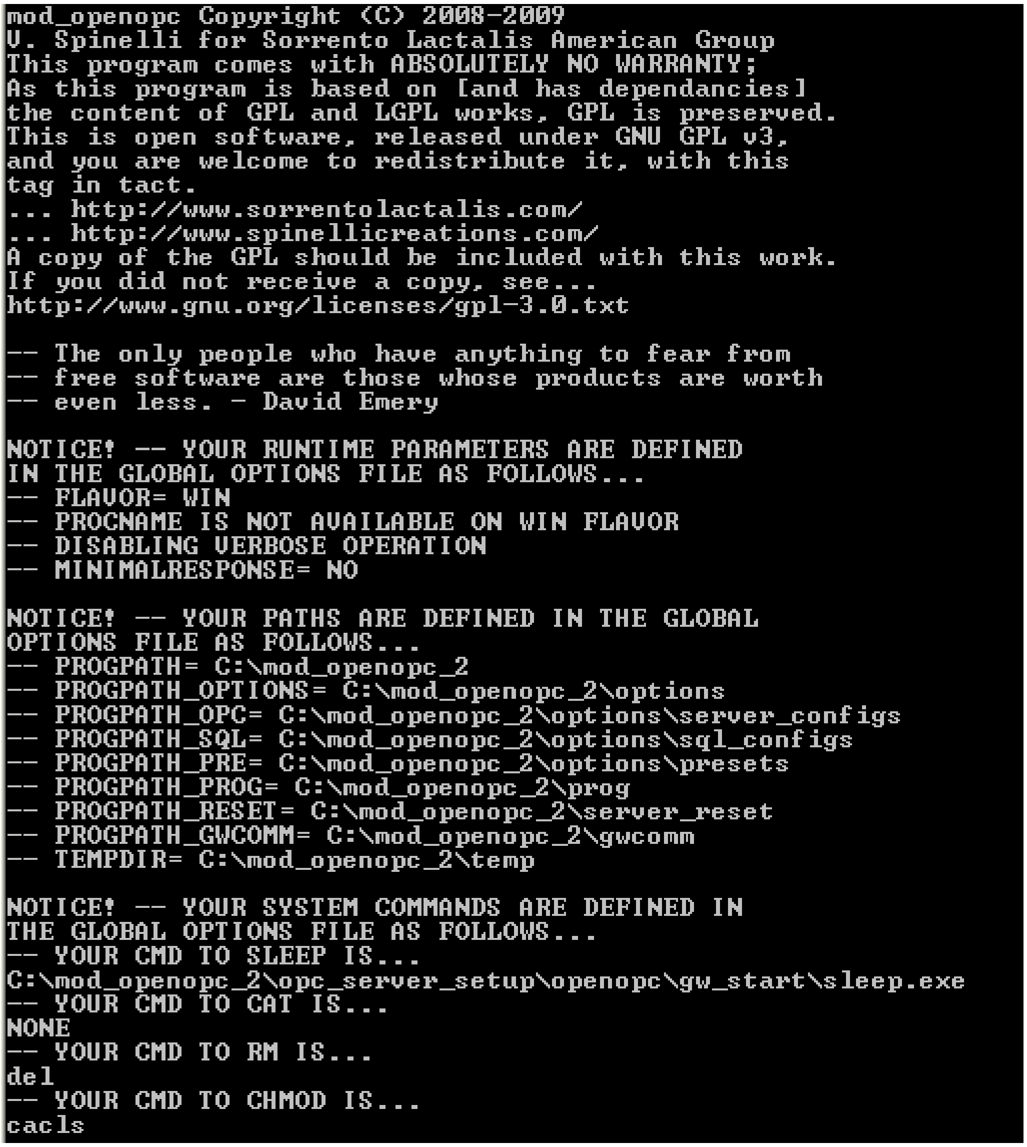
Structure...

```
[root@server] ./prog/mod_openopc.py  TEST_FOR_ECHO [opc_servername]
```

Example real world call...

```
[root@server] ./prog/mod_openopc.py  TEST_FOR_ECHO opc_server_1
```

** Below is a sample output when running under Windows Server 2003 R2 32 it **



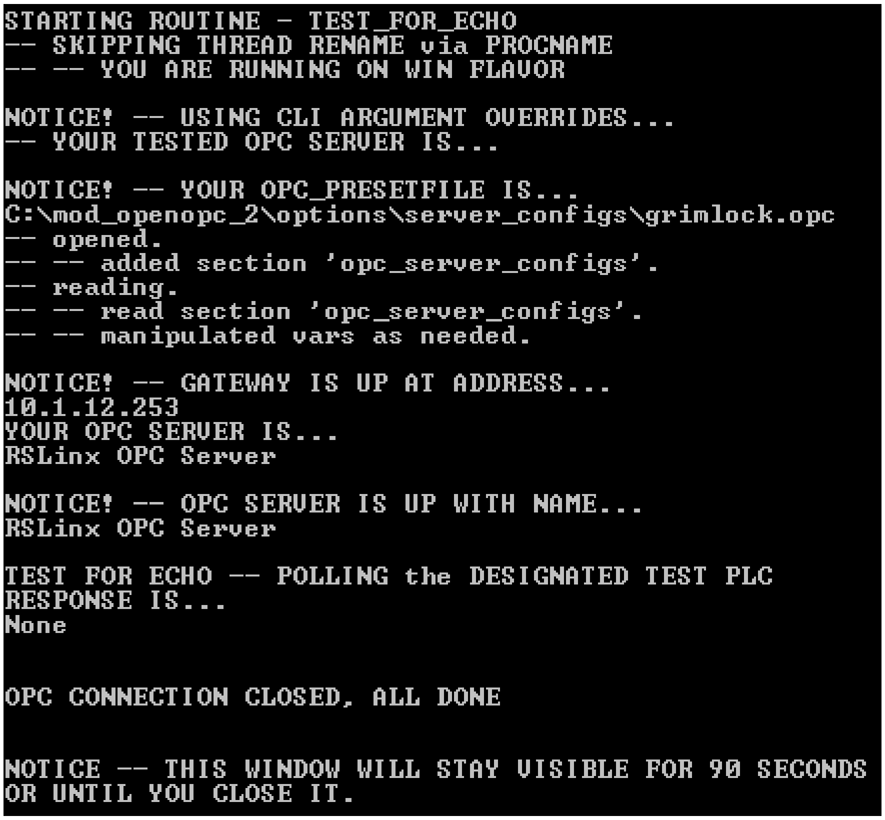
```
mod_openopc Copyright (C) 2008-2009
V. Spinelli for Sorrento Lactalis American Group
This program comes with ABSOLUTELY NO WARRANTY;
As this program is based on [and has dependancies]
the content of GPL and LGPL works, GPL is preserved.
This is open software, released under GNU GPL v3,
and you are welcome to redistribute it, with this
tag in tact.
... http://www.sorrentolactalis.com/
... http://www.spinellicreations.com/
A copy of the GPL should be included with this work.
If you did not receive a copy, see...
http://www.gnu.org/licenses/gpl-3.0.txt

-- The only people who have anything to fear from
-- free software are those whose products are worth
-- even less. - David Emery

NOTICE! -- YOUR RUNTIME PARAMETERS ARE DEFINED
IN THE GLOBAL OPTIONS FILE AS FOLLOWS...
-- FLAVOR= WIN
-- PROCNAME IS NOT AVAILABLE ON WIN FLAVOR
-- DISABLING VERBOSE OPERATION
-- MINIMALRESPONSE= NO

NOTICE! -- YOUR PATHS ARE DEFINED IN THE GLOBAL
OPTIONS FILE AS FOLLOWS...
-- PROGPATH= C:\mod_openopc_2
-- PROGPATH_OPTIONS= C:\mod_openopc_2\options
-- PROGPATH_OPC= C:\mod_openopc_2\options\server_configs
-- PROGPATH_SQL= C:\mod_openopc_2\options\sql_configs
-- PROGPATH_PRE= C:\mod_openopc_2\options\presets
-- PROGPATH_PROG= C:\mod_openopc_2\prog
-- PROGPATH_RESET= C:\mod_openopc_2\server_reset
-- PROGPATH_GWCOMM= C:\mod_openopc_2\gwcomm
-- TEMPDIR= C:\mod_openopc_2\temp

NOTICE! -- YOUR SYSTEM COMMANDS ARE DEFINED IN
THE GLOBAL OPTIONS FILE AS FOLLOWS...
-- YOUR CMD TO SLEEP IS...
C:\mod_openopc_2\opc_server_setup\openopc\gw_start\sleep.exe
-- YOUR CMD TO CAT IS...
NONE
-- YOUR CMD TO RM IS...
del
-- YOUR CMD TO CHMOD IS...
cacls
```



```
STARTING ROUTINE - TEST_FOR_ECHO
-- SKIPPING THREAD RENAME via PROCNAME
-- -- YOU ARE RUNNING ON WIN FLAVOR

NOTICE! -- USING CLI ARGUMENT OVERRIDES...
-- YOUR TESTED OPC SERVER IS...

NOTICE! -- YOUR OPC_PRESETFILE IS...
C:\mod_openopc_2\options\server_configs\grimlock.opc
-- opened.
-- -- added section 'opc_server_configs'.
-- reading.
-- -- read section 'opc_server_configs'.
-- -- manipulated vars as needed.

NOTICE! -- GATEWAY IS UP AT ADDRESS...
10.1.12.253
YOUR OPC SERVER IS...
RSLinx OPC Server

NOTICE! -- OPC SERVER IS UP WITH NAME...
RSLinx OPC Server

TEST FOR ECHO -- POLLING the DESIGNATED TEST PLC
RESPONSE IS...
None


OPC CONNECTION CLOSED, ALL DONE


NOTICE -- THIS WINDOW WILL STAY VISIBLE FOR 90 SECONDS
OR UNTIL YOU CLOSE IT.
```

Preset / Command
SERVER_SEEK →

Peforms a a check for the SERVER_NAME(s) of OPC server software packages running on the same machine that the GATEWAY service is on. Sometimes, server software packages don't use common or simple names, and sometimes they're not consistent. The Windows Service, SERVER_NAME, and Administrative Tools entries do not necessarily match… so it is up to you to check (here) and be sure.

Structure...

```
[root@server] ./prog/mod_openopc.py  SERVER_SEEK [gateway IPv4 address or FQDN]
```

Example real world call...

```
[root@server] ./prog/mod_openopc.py  SERVER_SEEK 192.168.10.5
```

win32_bootup.py

Some users may choose to run mod_openopc GATEWAY service and client operations on the same operating system instance. Basically, this would be the case of an “all in one” machine, running everything. This has been accomplished, and can be easily reproduced, on systems running Windows Server 2008 (64 and 32 bit) and Windows 7 (64 and 32 bit). It *should* (but has not been tested to) work well with Windows Server 2012 and forward, as well. Your mileage may vary – the preferred environment is still a Unix / Linux deployment. However, the portability of Python makes this a no-brainer for small / stand-alone applications.

In order to facilitate this, we have to ensure that the GATEWAY_DAEMON and the AUTO_LAUNCH routine are started in the correct order and their persistence is ensured.

A simple script was created – located in the “prog” folder – and called “win32_bootup.py”.

Adding this script to your Windows Start Menu / Startup programs will ensure a trouble-free experience.

The shortcut (in your “Startup” list) should be:

```
C:\[path-to-python]\python.exe C:\mod_openopc_2\prog\win32_bootup.py RUN
```

ADDITIONAL HELP

- For additional help with this package, setting it up, or just general “what is … xyz?”, contact the author at...

  Vince@SpinelliCreations.com

- For the latest downloads / patches / etc. publicly released, visit...

  http://download.spinellicreations.com/mod_openopc

- For the latest news on this project, go to...

  http://spinellicreations.com/spark/project_mod_openopc.php

DOCUMENT REVISION HISTORY (THIS DOC)

VERSION	***DATE***	***BASED on BUILD***	***NOTES***
1	2009-04-05	#23	- first draft
2	2010-01-11	#35	- massive rework of program and this document.
3	2010-05-04	#40	- minor rework of document, based upon new method of file-system path import.
4	2010-11-13	#49	- large rework of program to eliminate need for server reset delay to be based upon the scan interval of other subroutines, all helper scripts eliminated (integrated into main body of program), and the Gateway Monitor has become the GATEWAY_DAEMON subroutine.
5	2011-04-12	#51	– small rework / enhancement of program with drastic effects; you should review the revision history at this point for more details. However the documentation has been updated accordingly.
6	2011-05-03	#52	– MAINT_DB subroutine call structure corrected.
7	2011-12-19	#55	- addition of 'UPDATE' command line argument to 'READ' subroutine.
8	2012-01-19	#58	- complete program audit and restructure for unity.
9	2012-06-17	#61	- ability to reset OPC Server by declared command (as part of GATEWAY_RESET) to allow for flushing OPC Server of Zombie Groups (for misbehaving servers).
10	2016-07-19	#66	- addition of SERVER_SEEK command & win32_bootup.py script.