

How to Build Tramonto

Contents:

- Prerequisites
- Building Tramonto
- Adding New Code
- Running the Graphical User Interface

Prerequisites:

Tramonto requires several supporting software packages to build and run properly. Following is a list of the required packages and a web address from which you can download them. Some of the basic packages like BLAS and LAPACK may already be available on your computer in a format optimized for your cpu. Before downloading and installing these two packages, ask your system administrator about local versions of these libraries.

Package	Web Reference
BLAS	http://www.netlib.org
LAPACK	http://www.netlib.org
LAM-MPI	http://www.lam-mpi.org
Trilinos (4.0 or later)	http://software.sandia.gov/trilinos

Of these packages, Trilinos is the most complex and the one you will most likely have to build yourself. Trilinos contains several packages of which Tramonto requires aztecoo, triutils, ifpack and epetra. While Trilinos has its own detailed instructions on building you will find help in the following section.

Tramonto was developed using LAM-MPI, but you may use MPICH in its place (<http://www-unix.mcs.anl.gov/mpi/mpich/>). The section on building Tramonto will offer some guidance.

Building Trilinos:

As of July 2004, Tramonto requires Trilinos version 4.0 or later. Therefore, please ensure that you have a recent version.

Trilinos is a complex set of packages, that may appear difficult to build, but its integrated configure system helps. To set up Trilinos to build on your computer you will need to download the source code, uncompress, configure, compile and finally install it. There are many options available in configuring Trilinos before one builds it, but some of the most important one are:

```
--prefix=  
--enable-mpi  
CC=  
CXX=  
F77=
```

The "--prefix=" option lets you specify a location in which to install Trilinos. If you don't want to install it in /usr/local/lib and /usr/local/include, or if you don't have permission to do so,

you should pick a location in your home directory. For example, I use the following on my linux computer:

```
--prefix="/home/rlschie/arch/intelLinuxParallel"
```

When Trilinos is installed, all of its libraries go in /home/rlschie/arch/intelLinuxParallel/lib and its include files go in /home/rlschie/arch/intelLinuxParallel/include.

The "--enable-mpi" option builds a parallel version of Trilinos. Since Trilinos requires a parallel computer to run, you will need a parallel version of Trilinos.

Finally, the CC=, CXX= and F77= options let you specify which C, C++ and Fortran compilers to use. These variables are very useful on high performance computers where the compilers may be located in non-default locations.

Following are examples of ways to invoke the configure script to properly configure Trilinos. You can find more examples in the "sampleScripts" directory withing the Trilinos distribution.

Linux - parallel

```
../configure \  
  --prefix=/home/rlschie/arch/intelLinuxParallel \  
  --enable-mpi \  
  --with-mpi-compilers
```

Mac OSX - parallel

```
../configure \  
  CC=mpicc CXX=mpic++ F77=mpif77 \  
  FLIBS="-L/sw/lib/ -lg2c" \  
  LDFLAGS="-framework vecLib" \  
  --prefix=/Users/rlschie/arch/MacOsXParallel \  
  --enable-mpi
```

Cygwin Parallel

```
CC=mpicc CXX=mpic++ F77=mpif77 \  
../configure \  
  --prefix=/home/rlschie/arch/cygwinParallel \  
  --with-ldflags="-L/home/rlschie/arch/cygwinParallel" \  
  --with-lapack=/home/rlschie/arch/cygwinParallel/lib/lapack.a \  
  --enable-mpi
```

Ross - Parallel

```
../configure \  
CXXFLAGS=-O3 CPPFLAGS=-D__USE_STD_Iostream \  
TRILINOS_TEMPLATE_OBJS="./cxx_repository/*.o" \  
--with-cflags=-O3 --with-fflags=-O3 --host=alpha-unknown-linux \  
--with-mpi=/usr/local/cplint/ross/current \  
CXX="/usr/local/cplint/ross/current/bin/c++" \  
CC="/usr/local/cplint/ross/current/bin/cc" \  
F77="/usr/local/cplint/ross/current/bin/f77" \  
--prefix=/home/rlschie/arch/ross \  
--enable-mpi
```

West - Parallel

```
../configure \  
  --enable-mpi
```

```
CXXFLAGS=-O3 CPPFLAGS=-D__USE_STD_Iostream \  
TRILINOS_TEMPLATE_OBJS="./cxx_repository/*.o" \  
--with-cflags=-O3 --with-fflags=-O3 --host=alpha-unknown-linux \  
--with-mpi=/usr/local/cplant/west/current \  
CXX="/usr/local/cplant/west/current/bin/c++" \  
CC="/usr/local/cplant/west/current/bin/cc" \  
F77="/usr/local/cplant/west/current/bin/f77" \  
--prefix=/home/rlschie/arch/west \  
--enable-mpi
```

Once you have configured Trilinos, build it by typing "make". If the build is successful, then install it with "make install"

Building Tramonto:

Once you have the prerequisites at hand, building Tramonto is straightforward. As with Trilinos, you will have to first configure the code and then build it. Since the fully built Tramonto is just one executable, there is no need to install it -- you can just copy the executable to where it you will use it.

To configure Tramonto, you will need to tell the configure script where it can find the required libraries and compilers. There are several useful configure options:

```
--with-cflags=  
--with-cxxflags=  
--with-ldflags=  
CC=  
CXX=  
F77=  
--host=
```

The first three options identify where libraries like Trilinos and it's include files can be found. If you followed the example from the previous section, then these options will have values like:

```
--with-ldflags="-L/home/rlschie/arch/intelLinuxParallel/lib"  
--with-cxxflags="-I/home/rlschie/arch/intelLinuxParallel/include"
```

Additionally, if you are using MPICH you can specify the location of the MPI header files and libraries with:

```
--enable-cflags="-I/usr/MPICH/SDK.gcc/include" \  
--enable-ldflags="-L/usr/MPICH/SDK.gcc/lib -lmpich"
```

As with Trilinos, the options CC=, CXX= and F77= allow you to select specific compilers for your system.

The final option, --host=, is normally not needed unless you're cross-compiling. On cross-compiling system, you compile your program on one type of machine and run it on another. Typically, the compiled program won't run on the compiling machine, and this can cause the configure script to think that the compilers are broken. Specifying any value for host causes the configure script to ignore this issue.

Next you will find a set of configure invocations as examples and to

help you see how to configure Tramonto for your computer. These scripts can be found in the directory "sampleConfigs" in the Tramonto distribution.

Linux

```
./configure \  
  --with-ldflags="-L/home/rlschie/arch/intelLinuxParallel/lib" \  
  --with-cxxflags="-I/home/rlschie/arch/intelLinuxParallel/include"
```

Cygwin

```
./configure \  
  --with-ldflags="-L/home/rlschie/arch/cygwinParallel/lib -lf77blas -latlas -lg2c" \  
  --with-cxxflags="-I/home/rlschie/arch/cygwinParallel/include"
```

Mac OS X

```
./configure \  
  FLIBS="-lg2c" \  
  --with-ldflags= \  
  "-Wl,-framework -Wl,vecLib -L/sw/lib -L/Users/rlschie/arch/MacOsXParallel/lib" \  
  --with-cxxflags="-I/Users/rlschie/arch/MacOsXParallel/include"
```

Cplant Ross

```
./configure \  
  --host=dec \  
  CC=/usr/local/cplant/ross/current/bin/cc \  
  CXX=/usr/local/cplant/ross/current/bin/c++ \  
  F77=/usr/local/cplant/ross/current/bin/f77 \  
  --with-ldflags="-L/home/rlschie/arch/ross/lib -lmpi" \  
  --with-cxxflags="-I/home/rlschie/arch/ross/include"
```

Cplant West

```
./configure \  
  --host=cplant \  
  CC=/usr/local/cplant/west/current/bin/cc \  
  CXX=/usr/local/cplant/west/current/bin/c++ \  
  F77=/usr/local/cplant/west/current/bin/f77 \  
  --with-ldflags="-L/home/rlschie/arch/west/lib -lmpi" \  
  --with-cxxflags="-I/home/rlschie/arch/west/include"
```

Once the configure script completes, you can type "make" to build Tramonto. After make finishes, a copy of the Tramonto binary, "dft" will be placed in this directory.

To clean up after a build type "make clean" after which configure can be run again to build a different version of Tramonto.

Adding New Code:

If you add new code to this project, then please do the following to add it to the make files.

1. Edit the Makefile.in in the src to include your new source file in the build process.
2. If you've edited "configure.ac" then run "autoconf" to update the configure script.

4. Now you should be able to run configure and make as normal to build the project.

Running the Graphical User Interface (NEEDS TO BE UPDATED 10/2003)

The graphical user interface is run through a program called "Maui". Maui reads a series of configuration file that describes the interface and then helps you enter valid simulation parameters for your problem of interest.

To run Maui do the following.

1. Define the environment variable IDEA_HOME to point to the location of the top level Tramonto directory. For example, if you've unpacked Tramonto in /home/myAccount/src/, so that there is now a directory called /home/myAccount/src/Tramonto, then set IDEA_HOME to /home/myAccount/src/Tramonto.

2. Add to your path the location of Maui's startup script by adding \$IDEA_HOME/Idea/Java/bin to your path. You may have to type "source ~/.cshrc" or "source ~/.bash_profile" or even logoff and back on again to get your changes to your path to show up in your current shell.

3. Start Maui by typing "Maui.e"

4. On your first time running Maui you will have to tell it where to find the Tramonto configuration files. When Maui starts up click on the "Configure Maui" button.

5. In the window that just opened, click on the "Package Names" tab and then click on the "Add" button. In the dialog box that just popped up, type "Tramonto" in the box that's labeled "package". Note, that the case of "Tramonto" is important, so typing "tramonto" won't work. Click "Ok" to close this dialog box.

6. Next, click on the "Services to Add" tab and then click on the "Add" button. In the dialog box that just opened, in the box labeled "service" type your value for IDEA_HOME followed by /guiSupport/Tramonto.xml. If IDEA_HOME was "/home/myAccount/src/Tramonto" then you will type "/home/myAccount/src/Tramonto/guiSupport/Tramonto.xml" In the box labeled "label" type "Tramonto". Make sure the check box "To be included" is checked and then click "OK" to close this dialog box.

7. Next, click on the tab labeled "Classpath entries", click on the button labeled "Add" and in the dialog box that just opened type in your value for IDEA_HOME followed by /guiSupport/Tramonto.jar". Thus if your value for IDEA_HOME was "/home/myAccount/src/Tramonto" then you would type "/home/myAccount/src/Tramonto/guiSupport/Tramonto.jar". Now click on "Ok" to close this dialog box.

8. Well, that was the worst of it. Now click on "Save Config" then "Apply Settings" and then "Close" Maui will ask if you're

sure you want to close this window, so click on "Yes". That was the end of the configuration work -- if you don't move the Tramonto or Idea directories, you will never have to do steps 1 through 8 again and can just skip to step 9.

9. To start the Tramonto part of the Maui interface click on the "Start New Session" button.

10. In a few moments, a window will pop up on the second line of which is a label saying "Select subclass" and next to it a pull down box. From the pull down box select "Tramonto". You will now have the Tramonto interface up and you can begin entering values. To save your work click on "Save XML" and to restore it click on "Read XML". When you're ready to convert your input to a file that Tramonto can understand click on "Submit".