



IST-2001-32133

GridLab - A Grid Application Toolkit and Testbed

## D 3.6 Mobile Communication By Email

---

Author(s):	Matthew Shields, Ian Wang and Ian Taylor
Document Filename:	GridLab-3-D3_6-0001-MBCE
Work package:	WP3 Work-Flow Application Toolkit (TGAT)
Partner(s):	University of Wales, Cardiff
Lead Partner:	University of Wales, Cardiff
Config ID:	GridLab-3-D3_6-0001-1.0-Draft_A
Document classification:	INTERNAL

---

**Abstract:** This document outlines mobile communication by email within Triana

## Contents

<b>1</b>	<b>Introduction</b>	<b>2</b>
<b>2</b>	<b>Reasoning</b>	<b>2</b>
<b>3</b>	<b>Implementation</b>	<b>2</b>

## 1 Introduction

As part of the project proposal Triana needed the ability to communicate with a user or users via email. A user may be contacted with interim or final results of a taskgraph execution or asked to respond with input back before a taskgraph can continue execution. Triana now has the ability to send and receive email messages through the use of two new units, *SendMailMsg* and *RecvMailMsg*.

## 2 Reasoning

Executing taskgraphs in Triana can take anything from a few seconds to an unlimited amount of time to complete. In the case of short running applications the user would typically be sat in front of a monitor waiting for the results but in the case of long running or even never ending application the user would not want to wait for results. In these cases it would be useful for Triana to be able to communicate results via email when execution has finished or when an interesting event occurs. Further we may have cases where the user wants to be notified of an event and modify the application's behaviour based on that information.

For example, if Triana is being used to monitor a data stream such as the one that comes from the gravitational wave detector used by the GEO 600 project [1], Triana would be running continuously. The scientist using Triana would not necessarily be sat in front of the application however he or she may want to be notified if an event is found in the analysed data. The taskgraph could then pause waiting for a reply as to whether the event warrants further investigation and run one of two paths in the taskgraph based on the user's emailed response. One path would ignore the event and the other would perform further analysis.

## 3 Implementation

The implementation of the email mechanism could have been done as an extension to the core Triana application but since this form of communication would not be used by all users of Triana it was better served by implementing two new units or components that a user can select to use in the same manner as any other unit within Triana. The two units, *SendMailMsg* and *RecvMailMsg* are self contained units that perform sending and receiving of email messages respectively. The units are Java programs that make use of the JavaMail API [2] from Sun Microsystems to implement the standard *SMTP*, *POP* and *IMAP* mail protocols. The two units and their parameter user interfaces can be seen in the screen shot [figure 1].

As can be seen from the figure, the send mail unit has a number of input parameters that the user sets: mail server; from email address; recipient mail addresses; subject; message ID; and the message contents. The unit uses the mail server that the user specifies, via the *SMTP* mail protocol, to send the message. All of the other parameters can be entered by hand via the user interface, passed from other units as parameters or even set from input data. The unit accepts as input the Triana data type *MailMsg*, a sub type of the *Document* data type which contains all of the information needed to be a valid email message apart from the mail server information. The unit outputs sent email as a *MailMsg* which can be used or ignored in the taskgraph.

The receive mail unit also has a number of parameters: mail server; user name; user password; mail folder; receiving protocol; and message ID. The user must enter these parameters or have

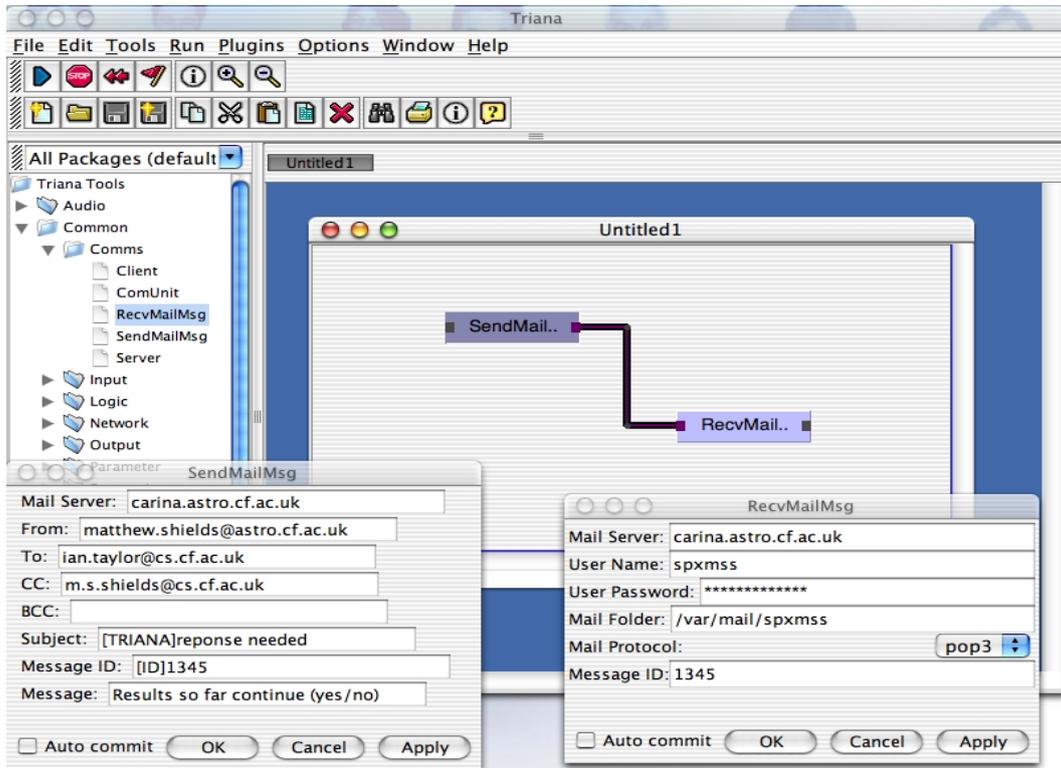


Figure 1: A screen shot of the send and receive units in use

them passed from other units. The unit uses the users mail account, via either the *POP* or *IMAP* mail protocols, to search for emails. As can be seen in the send mail unit interface on the screen shot, the subject line starts with the string “[TRIANA]”, the receive mail unit uses this string to filter the incoming mail, recognising only subjects starting with this string as valid emails. If there is more than one match on the incoming mail then the message ID can be used to further filter the email. The message ID can be passed from the send mail unit as a parameter.

When the receive unit is used in a constructed taskgraph, it will pause until a valid email has been received and then output that message as a *MailMsg* data type. Further units can then access the message parameter using a comparison unit to branch the taskgraph one way or another.

One potential downside in the mind of users is that Triana has access to their mail folder however this can be easily overcome by setting up an email account specifically for use by Triana or using a standalone mail server.

The two units that implement the send and receive email functionality are available now as standard units in the Triana toolbox. They can be found in the package *Common.Comms* in the toolbox tree.

## References

- [1] The GEO 600 Project <http://www.geo600.uni-hannover.de/>
- [2] The JavaMail API, <http://java.sun.com/products/javamail/index.html>