ITC303 – Team Orange E2 Iteration Plan

1. High Level Objectives/Milestones

ID	Objectives	Date
-	Iteration start	24/04/2017
1	Implement Base project components	07/05/2017
2	Implement request contact base functionality.	07/05/2017
2	Implement read contact list base functionality.	07/05/2017
2	Implement amed contact list base functionality.	07/05/2017
3	Implement Contacts Class	07/05/2017
4	Implement remove Contact from roster	07/05/2017
5	Create chat use case	07/05/2017
6	View chats use case	07/05/2017
7	Research and Implement MVC architecture	07/05/2017
8	Create GUI	07/05/2017
-	Iteration End	01/05/2017
8	Review Iteration	02/05/2017

2. Work Item assignments

The following Work Items will be undertaken in this iteration to complete the above listed Milestones:

ID	Work Items	Measure	Outcome	Assigned	Completed by	Hours worked
1.1	Implement Development Branch.	New Branch in project created	Branch E2 created.	All	07/05/2017	
1.2	Implement Base Class.	Base Class constructs Subclasses, seen here.	Babbler XMPP base class implemented (refer to diagram)	All	07/05/2017	
2.1	Implement Contacts class function to Create new Roster	New Roster assigned to Server.	Not required, Rosters are created we contact is created.	Shaun	07/05/2017	0.5
2.2	Implement Contacts class function to request contact added to Roster.	Request for Roster membership can be sent to Contact.	Implemented in ContactManager type, method addContact(),	Shaun	07/05/2017	3
2.3	Implement Contacts class function to accept contact request.	Request for Roster membership can be read and responded to.	Found additional dependencies/complexity to implement full functionality. Will delay to next iteration to allow function to work with other event driven functions. Although Implemented base methods for use with event handling in ContactManager type, method approveContact(), denyContact(), addListenerPresence(), removeListenerPresence(), getPresenceOfEvent(), getContactOfEvent().	Shaun	07/05/2017	2
2.4	Implement Test Script 2.1	Test outputs results to console.	Built and executed TestScript2.1, see console output.	Shaun	07/05/2017	1

2.5	Update TestPlan2 Doc and iteration plan with evidence,	Doc updated	Completed.	Shaun	07/05/2017	0.5
3.1	Implement Contacts class function to create list of Groups User assigned to.	List created of Groups assigned to Contact	Successful, implementation in ContactManager type, method getContactGroups()	Shaun	07/05/2017	3
3.2	Implement Contacts class function to create list of Contacts in Roster.	List created of Contacts in Roster.	Successful, implementation in ContactManager type, method getContacts().	Shaun	07/05/2017	2
3.3	Implement Test Script 2.2	Test outputs results to console.	Built and executed TestScript2.2, see console output.	Shaun	07/05/2017	1
3.4	Update TestPlan2 Doc and iteration plan with evidence,	Doc updated	Completed.	Shaun	07/05/2017	0.5
4.1	Implement Contacts class function to remove Contact from Roster.	Contact removed from Roster.	Successful, implementation in ContactManager type, method removeContact()	Shaun	07/05/2017	2
4.2	Implement Test Script 2.3	Test outputs results to console.	Built and executed TestScript2.3, see console output.	Shaun	07/05/2017	1
4.3	Update TestPlan2 Doc and iteration plan with evidence,	Doc updated	Completed.	Shaun	07/05/2017	0.5
5.1	Retrieve a user (or users details e.g. JID) [This will be normally implemented through contacts, so we probably just have to know a user's JID for now.]		Unnecessary, just starting chats based on username for now until contacts is fully implemented.	Tim/Murray	07/05/2017	
5.2	(Optional) Set up a chat bot on the server that will respond to a received message. This will make testing chat functionality much easier.	A chat bot is set up on the server that will respond to a received message.	Incomplete. Couldn't find a good one. Will keep looking.	Murray	07/05/2017	1
5.3	Create a chat with a user	Chat is created for both users (creator and other)	A user can create an AppChatSession with another user, and this will send a message to the other user, telling them to create a corresponding chat session with the same thread. Only works if the other user is online at the time.	Tim	07/05/2017	2
5.4	Send a message to a user in the chat	The other user receives the message	Same as bellow.	Tim	07/05/2017	2

5.5	Receive a message from another user	The sent message is received	I created a chat session on my desktop, joined it on my laptop, and sent messages between the two successfully. However receiving the message to create a corresponding chat session and receiving chat messages only works if the other user is online, as messages are not saved on the server and sent the next time the user is online.	Tim	07/05/2017	2
5.6	Notify of new Chat Event (new Chat, new Message)	System knows when a new chat event occurs (a callback handler function is called)	A message listener is alerted whenever there is a new message, and the message is handled accordingly. When user A creates a new ChatSession with user B, user B receives a message (not a chat message) telling them to create a corresponding chat session.	Tim	07/05/2017	2
5.7	Notify of Chat status (read/unread)	The status of a chat is held correctly. Starting as unread, becoming read when the message is seen, and handling the case of being read instantly (received when that chat is open).	Incomplete. Did not get up to this stage.	Tim	07/05/2017	
5.8	Pass all of TestPlan3	All steps of all test functions in TestPlan3 are successfully completed.	TestPlan3 3.1 and 3.2 have been completed. 3.3 has not been attempted yet. Murray is working on automation of all tests via JUnit/Mockito.	Tim/Murray	07/05/2017	0
6.1	Retrieve a list of all chats. (id, users between)	A list of chats can be retrieved, and a given chat can be identified from this list.	When a chat is created it is represented by the model AppChatSession, and is saved in the model AppChats	Tim	07/05/2017	3

6.2	Retrieve a list of messages for a given chat, from the most recent message back X messages.	A list of messages can be retrieved from a given chat.	Chat model has been created called AppChatSession, and when a message is sent or received in that chat, it is saved to its list of AppChatSessionMessage. However these chats and messages are not yet saved on the server.	Tim	07/05/2017	3
7.1	Research and implement MVC Architecture	MVC Architecture is implemented in project	MVC architecture has been implemented, as can be seen here. Views and Models don't know about each other or Controllers. Controllers know about the Views and Models that they need.	Tim	07/05/2017	2
8.1	Create log in / create account gui	A view exists with corresponding controllers for the LogIn and Create Account use cases.	AccountScreen accepts username and password input. Then calls LoginController if login button is pressed, or CreateAccountController if create account button is pressed.	Tim	07/05/2017	2
8.2	Create Chat gui	A view exists with a corresponding controller for the create chat use case.	HomeScreen accepts the name of a user. Then when the create chat button is pressed, it calls the CreateChatController.	Tim	07/05/2017	2
8.3	Chat gui	A view exists with a corresponding controller for the chat use case.	ChatScreen allows a message to be typed in, and when the send button is pressed, it calls the ChatController. The message is sent and saved within models, however it is not displayed in the gui yet.	Tim	07/05/2017	2
8.4	Create Base gui structure.	Screens are tracked, displayed, and implemented nicely.	Screen is a class that all screens extend. ScreenEnum is used to track which screen we are on, and change screens. GuiBase is used to display screens.	Tim	07/05/2017	2

9.0	Create new more detailed master test plan	Implement template provided in the resources section	MasterTestPlan2	Murray	09/05/2017	3