It is also necessary however for the user to provide domain specific code to implement task functionality. That is, the user must provide code for all of the functions for computing costs for a primitive task, and for providing callbacks 555 to control external systems. This is done using the code generation editor 360.

This is the only place where the developer needs to define any executable code. CABS can define prototypes for these functions and also provides an API library 380 for use by developers in their code production.

The output of CABS 100, in the form of software agents, is distributed to the user environment via scripts 390. Existing systems can then be linked to the agents using an API of a wrapper class, defined by the system, which wraps code inherited from the component library together with the user-supplied data.

## 5. DEBUGGING AND VISUALISATION

It is difficult to create any single program, even a program which interacts with no other programs, which has no faults or errors in it. It is an order of magnitude more difficult to analyse and debug distributed software which contains multiple agents. The behaviours that emerge from the overall distributed software may not be at all what was expected. Further, the communication is often at such a high level that it is not possible to look directly at data involved. Even an object based system can be designed to allow the programmer to look at the actual data involved in a sequence of events but agents, using message-based communication, may simply not enable the programmer to do so.

Clearly, it is important to analyse what is going wrong so that it can be corrected. Approaches used in singular 'agent' applications may be used but are frequently inadequate in analysing and debugging distributed software systems. For example, fault analysis can be done as a "post mortem" exercise where the data is stored at the time of failure which later provides a "snap shot" of the prevailing circumstances, from which an attempt can be made to find out what caused the failure. A known debugger, described in international patent application No: W093/24882 in the name of the present applicant, shows how such data can be captured in a history file which can later be 'played', 'replayed', 'rewound', 'fast forwarded', etc. - video style.