Communication is key to distributed systems

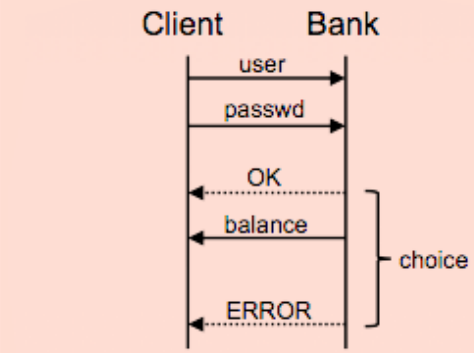
Modern life depends on large-scale distributed software systems. The key to making them work is to ensure that their components communicate with each other correctly. **Dr Simon Gay** tells us how COST Action IC1201 (BETTY) aims to use behavioural type theory as a basis for new programming languages and software development methods

For a long time data processing was the primary focus in computing development, with isolated machines running their own software systems. However, with modern society growing increasingly dependent on distributed software systems, computing is changing to reflect new priorities. "Most computing now is not based on isolated systems doing data processing. Today we have large networks of communicating systems; things like web applications, online banking and online shopping all make use of distributed computing systems. The key to making distributed computing systems work is to make sure that all the components are communicating with each other correctly," explains Dr Simon Gay, the coordinator of COST Action IC1201: Behavioural Types for Reliable Large-Scale Software Systems (BETTY). Behavioural type theory, which encompassess a range of concepts, including interfaces, communications protocols and contracts, holds real potential in this area. "The overall aims of the BETTY project are to continue developing the theory of behavioural types, to try and push behavioural types out into practice, and to make them applicable in practical software development," continues Dr Gay.

Many of the standard programming languages in use today were designed with data processing firmly in mind. It was recognised early on in development that the structure of data, or data types, is very important to the effectiveness of a program. "Programmers need to know for example that a person's address consists of a number of fields, what kind of data is in each field, and what the correct format of that is. Essentially all programming languages have very strong support for these data types. This means that when you're writing a program, you can write declarations in your program of the data types, the data formats, that you're working with. Then the programming language system checks that you're using all parts of the data correctly," explains Dr Gay. Data types are quite static however, so aren't well-suited to today's distributed systems. "Data types describe the structure of data and the relationships between data, but they don't say anything for example about sequences of operations," continues Dr Gay. "Most computing now is not about isolated systems processing data, but rather large networks of communicating systems."

Distributed systems

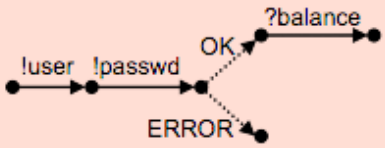
The designer of a distributed system has to consider communication protocols, the sequence and format of messages that are being exchanged between different components. Current standard programming languages don't offer effective support for the definition of these communication protocols, says Dr Gay. "Although the programming language may help people get the format of each message correct, it doesn't help them get the sequence of messages correct, so the programmer often has to test it manually," he outlines. Programmers have become accustomed to development environments where the system constantly analyses and checks the code that they write, helping them identify errors, but this is not yet available for communication programming. "As soon as you type something on your keyboard which looks like applying an operation to a nonsensical piece of data, you'll get a little red line under it telling you you've done something wrong. So it provides immediate feedback," says Dr Gay. "At the moment that's just not available for errors in communication programming."



(a) sequence diagram

```
ClientType =
  !user . !passwd .
  &{ OK: ?balance . end ,
     ERROR: end }
```

(b) behavioural type



(c) transition system

This is an issue which Dr Gay and his colleagues are working to address. The underlying idea behind the project is to think of the communication structures in a distributed system as a type of information, analogous to data types. "We aim to identify all the theories, techniques, tools, programmer support that we like from data types, and then to apply them to behavioural types," says Dr Gay. A lot of theoretical research has been done on behavioural types, but it is only in the last five years or so that significant progress has been made on putting these ideas into prototype programming languages. "There are some prototype programming languages in which you can indeed define these behavioural types, use them in programming and get automatic checking. But these programming languages are either small prototypes or experimental extensions of existing languages – they're not yet ready for full-scale use by industrial software developers," continues Dr Gay. "Part of our aim is to move closer to a world where these kind of ideas are suitable for use in industrial software development."

Today we have large networks of communicating systems; things like web applications, online banking and online shopping all make use of distributed computing systems. The key to making distributed computing systems work is to make sure that all the components are communicating with each other correctly

General approach

A lot of work has been done in this area, yet Dr Gay believes one standardised, general approach is required to put these ideas on behavioural types into practice. There are also several other issues to consider before the project's ideas can be applied practically. "One thing we need to put these ideas into practice is to look at error recovery and how to handle interaction with components that are outside the nice, perfect design. We're starting to see some work on issues like error handling, but I think we need more of this type of research," says Dr Gay. One area that he believes warrants further exploration is time-outs. "In practical systems, there's always the possibility that the component you're trying to talk to has somehow failed, and you can't wait forever for it to complete your interaction. So usually there's some notion of a time-out," points out Dr Gay. "A customer might wait five seconds for a response from the bank for example, and if it doesn't come then you decide that the bank isn't operating."

These types of large, distributed systems

generally work very well, yet there is still scope for further improvement. One of the working groups within BETTY specifically concerns security, which is a particularly important issue in online banking. "A lot of work is ongoing on techniques for analysing security software, as well as research into integrating security analysis with behavioural type descriptions. We want to encourage more work in that direction, so that a behavioural type communication between the client and the bank should not only talk about the format of the messages that are being exchanged, but also guarantee security properties at certain points," outlines Dr Gay. The project's ultimate aim is to help software companies produce reliable, communication-based software more quickly and easily. "Some of the participants in BETTY have their own research projects, some with industrial collaborators. They're starting to apply behavioural typing ideas, and they're already starting to see productivity benefits," says Dr Gay.

This of course is an important issue for the commercial sector, suggesting that behavioural type theory will have a major part to play in the long-term future of software development. However, Dr Gay says the project will have to consider the techniques that the software industry is currently using if it is to have a wider impact. "The way to do that is to try to develop improvements through existing methodologies. We have the idea of behavioural types, now we want to deploy that in the programming languages people are already working with," he explains. This will be an important part of the overall research agenda, both within the original term of the project and beyond. "We will spend the four-year term of the BETTY project really trying to consolidate the community and setting up new collaborations, which we hope will then continue after the end of the project itself," says Dr Gay.

At a glance

Full Project Title
COST Action IC1201: Behavioural Types for Reliable Large-Scale Software Systems (BETTY)

Project Objectives
To use behavioural type theory as the basis for new foundations, programming languages, and software development methods for communication-intensive distributed systems. Behavioural type theory encompasses concepts such as interfaces, communication protocols, contracts, and choreography. As a unifying structural principle it has the potential to transform the theory and practice of distributed software development.

Project Funding
€130k - €150k per year over 4 years (October 2012 - October 2016).

Project Partners
European Cooperation in Science and Technology (COST)

Contact Details
Project Coordinator
Dr Simon Gay
School of Computing Science
University of Glasgow
Glasgow G12 8QQ, UK
T: +44 141 330 6035
E: Simon.Gay@glasgow.ac.uk
W: www.behavioural-types.eu

Dr Simon Gay

Project Coordinator

Dr Simon Gay is a Senior Lecturer in the School of Computing Science at the University of Glasgow, Scotland, where he has been working since 2000. In addition to chairing COST Action IC1201 (BETTY), he is involved in two research projects funded by the UK EPSRC. Both projects concern structured communication: one for distributed computing systems, and one for many-core microprocessors.