European Cooperation in Science and Technology

BETTY — a COST action

Behavioural Types for Reliable Large-Scale Software Systems

Tihana Galinac Grbac

Faculty of Engineering, University of Rijeka, Croatia

14th Workshop on Software Engineering Education and Reverse Engineering, Sinaia, Romania, 24-30 August 2014

The problem

Symptoms

- In the past, computing consisted of isolated computers processing data. Programming languages adopted data types in order to codify the structure of data and support the development of reliable data-processing software.
- Modern society is increasingly dependent on large-scale software systems that are distributed, collaborative and communication-centred.
- Correctness and reliability of such systems depend on compatibility between components and services that are newly developed or may already exist.
- The consequences of failure are severe, including security breaches and unavailability of essential services.

The problem

Symptoms

- In the past, computing consisted of isolated computers processing data. Programming languages adopted data types in order to codify the structure of data and support the development of reliable data-processing software.
- Modern society is increasingly dependent on large-scale software systems that are distributed, collaborative and communication-centred.
- Correctness and reliability of such systems depend on compatibility between components and services that are newly developed or may already exist.
- The consequences of failure are severe, including security breaches and unavailability of essential services.

The problem

Symptoms

- In the past, computing consisted of isolated computers processing data. Programming languages adopted data types in order to codify the structure of data and support the development of reliable data-processing software.
- Modern society is increasingly dependent on large-scale software systems that are distributed, collaborative and communication-centred.
- Correctness and reliability of such systems depend on compatibility between components and services that are newly developed or may already exist.
- The consequences of failure are severe, including security breaches and unavailability of essential services.

The problem

Symptoms

- In the past, computing consisted of isolated computers processing data. Programming languages adopted data types in order to codify the structure of data and support the development of reliable data-processing software.
- Modern society is increasingly dependent on large-scale software systems that are distributed, collaborative and communication-centred.
- Correctness and reliability of such systems depend on compatibility between components and services that are newly developed or may already exist.
- The consequences of failure are severe, including security breaches and unavailability of essential services.

The problem

Diagnosis

- The large-scale software systems we rely upon crucially depend on communication between co-operating components.
- Current software development technology is not well suited to producing these systems, due to the lack of high-level structuring abstractions for complex communication behaviour.
- The reliability of these systems (also) depend on the abstraction mechanisms to represent and reasoning about their behaviour.

The problem

Diagnosis

- The large-scale software systems we rely upon crucially depend on communication between co-operating components.
- Current software development technology is not well suited to producing these systems, due to the lack of high-level structuring abstractions for complex communication behaviour.
- The reliability of these systems (also) depend on the abstraction mechanisms to represent and reasoning about their behaviour.

The problem

Diagnosis

- The large-scale software systems we rely upon crucially depend on communication between co-operating components.
- Current software development technology is not well suited to producing these systems, due to the lack of high-level structuring abstractions for complex communication behaviour.
- The reliability of these systems (also) depend on the abstraction mechanisms to represent and reasoning about their behaviour.

A step towards a cure

Type systems

- As computing moves from the data-processing era to the communication era, we need to codify the structure of communication to support the development of reliable communication-oriented software.
- Data type systems are very useful to statically prevent operations from "going wrong".
  Type systems have been consistently developed since there are high-level programming languages.
- Is it possible to encode as types the communication structure of modern computer systems and statically verify behavioural properties about them?

A step towards a cure

Type systems

- As computing moves from the data-processing era to the communication era, we need to codify the structure of communication to support the development of reliable communication-oriented software.
- Data type systems are very useful to statically prevent operations from "going wrong".

  Type systems have been consistently developed since there are high-level programming languages.
- Is it possible to encode as types the communication structure of modern computer systems and statically verify behavioural properties about them?

A step towards a cure

Type systems

- As computing moves from the data-processing era to the communication era, we need to codify the structure of communication to support the development of reliable communication-oriented software.
- Data type systems are very useful to statically prevent operations from "going wrong".
  Type systems have been consistently developed since there are high-level programming languages.
- Is it possible to encode as types the communication structure of modern computer systems and statically verify behavioural properties about them?

A step towards a cure

Behavioural types

- To specify, characterize and reason about dynamic aspects of program execution, one needs to formalize and codify communication structures.
- Behavioural types capture these aspects and can form a basis for both static analysis and dynamic monitoring.
- Concrete approaches are:
  - Session types, by Honda, Kubo, Takeuchi, and Vasconcelos
  - Contracts, by Castagna, Gesbert, and Padovani
  - Typestates, by Aldrich, Bierhoff, DeLine, Fähndrich, Strom, and Yemini

A step towards a cure

Behavioural types

- To specify, characterize and reason about dynamic aspects of program execution, one needs to formalize and codify communication structures.
- Behavioural types capture these aspects and can form a basis for both static analysis and dynamic monitoring.
- Concrete approaches are:
  - Session types, by Honda, Kubo, Takeuchi, and Vasconcelos
  - Contracts, by Castagna, Gesbert, and Padovani
  - Typestates, by Aldrich, Bierhoff, DeLine, Fähndrich, Strom, and Yemini

A step towards a cure

Behavioural types

- To specify, characterize and reason about dynamic aspects of program execution, one needs to formalize and codify communication structures.
- Behavioural types capture these aspects and can form a basis for both static analysis and dynamic monitoring.
- Concrete approaches are:
  - Session types, by Honda, Kubo, Takeuchi, and Vasconcelos
  - Contracts, by Castagna, Gesbert, and Padovani
  - Typestates, by Aldrich, Bierhoff, DeLine, Fähndrich, Strom, and Yemini

A step towards a cure

Behavioural types

- To specify, characterize and reason about dynamic aspects of program execution, one needs to formalize and codify communication structures.
- Behavioural types capture these aspects and can form a basis for both static analysis and dynamic monitoring.
- Concrete approaches are:
  - Session types, by Honda, Kubo, Takeuchi, and Vasconcelos
  - Contracts, by Castagna, Gesbert, and Padovani
  - Typestates, by Aldrich, Bierhoff, DeLine, Fähndrich, Strom, and Yemini

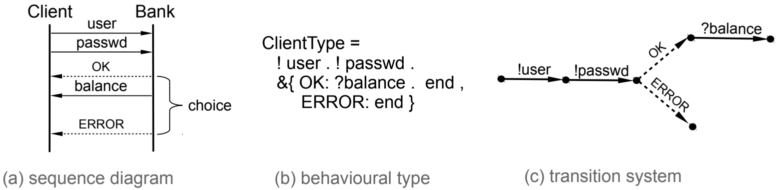
A step towards a cure

Behavioural types

- To specify, characterize and reason about dynamic aspects of program execution, one needs to formalize and codify communication structures.
- Behavioural types capture these aspects and can form a basis for both static analysis and dynamic monitoring.
- Concrete approaches are:
  - Session types, by Honda, Kubo, Takeuchi, and Vasconcelos
  - Contracts, by Castagna, Gesbert, and Padovani
  - Typestates, by Aldrich, Bierhoff, DeLine, Fähndrich, Strom, and Yemini

A step towards a cure

Behavioural types: a simple example



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

(a) sequence diagram

(b) behavioural type

(c) transition system

A step towards a cure

Behavioural type theory

- Behavioural type theory is 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 will transform the theory and practice of distributed software development.

A step towards a cure

Behavioural type theory

- Behavioural type theory is 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 will transform the theory and practice of distributed software development.

The BETTY project

Aims

- To develop the domain of certified software for global services, by incorporating behavioural types into programming languages and software engineering tools for automatically checking behavioural properties of communicating software systems.
- To co-ordinate European research activity on the theory and application of behavioural types, and the deployment of programming languages and tools based on them.
- To build an effective working community of European researchers in this area.
- To encourage the industrial adoption of advanced programming languages and tools.

The BETTY project

Aims

- To develop the domain of certified software for global services, by incorporating behavioural types into programming languages and software engineering tools for automatically checking behavioural properties of communicating software systems.
- To co-ordinate European research activity on the theory and application of behavioural types, and the deployment of programming languages and tools based on them.
- To build an effective working community of European researchers in this area.
- To encourage the industrial adoption of advanced programming languages and tools.

The BETTY project

Aims

- To develop the domain of certified software for global services, by incorporating behavioural types into programming languages and software engineering tools for automatically checking behavioural properties of communicating software systems.
- To co-ordinate European research activity on the theory and application of behavioural types, and the deployment of programming languages and tools based on them.
- To build an effective working community of European researchers in this area.
- To encourage the industrial adoption of advanced programming languages and tools.

The BETTY project

Aims

- To develop the domain of certified software for global services, by incorporating behavioural types into programming languages and software engineering tools for automatically checking behavioural properties of communicating software systems.
- To co-ordinate European research activity on the theory and application of behavioural types, and the deployment of programming languages and tools based on them.
- To build an effective working community of European researchers in this area.
- To encourage the industrial adoption of advanced programming languages and tools.

Working groups

WG 1: Foundations
Developing the theory of behavioural types in order to support their applications.

WG 2: Security
Integrating behavioural types with techniques for security analysis.

WG 3: Languages
Implementing behavioural types in practical programming languages.

WG 4: Tools and Applications
Developing software engineering tools and applying them to realistic case studies.

Working groups

WG 1: Foundations
Developing the theory of behavioural types in order to support their applications.

WG 2: Security
Integrating behavioural types with techniques for security analysis.

WG 3: Languages
Implementing behavioural types in practical programming languages.

WG 4: Tools and Applications
Developing software engineering tools and applying them to realistic case studies.

Working groups

WG 1: Foundations
Developing the theory of behavioural types in order to support their applications.

WG 2: Security
Integrating behavioural types with techniques for security analysis.

WG 3: Languages
Implementing behavioural types in practical programming languages.

WG 4: Tools and Applications
Developing software engineering tools and applying them to realistic case studies.

Working groups

WG 1: Foundations
Developing the theory of behavioural types in order to support their applications.

WG 2: Security
Integrating behavioural types with techniques for security analysis.

WG 3: Languages
Implementing behavioural types in practical programming languages.

WG 4: Tools and Applications
Developing software engineering tools and applying them to realistic case studies.

Recent and next events

- WG meetings: ETAPS, March 2013; SEFM, September 2013; ETAPS, April 2014; and CONCUR 2014, September 2014.
- 1st International Workshop on Behavioural Types (BEAT) at POPL, January 2013.
- 2nd International Workshop on Behavioural Types (BEAT 2) at SEFM, September 2013.
- 3rd International Workshop on Behavioural Types (BEAT 3) at CONCUR, September 2014.
- International Summer School on Behavioural Types, 2014.
- Further International Workshops on Behavioural Types in 2015 and beyond.
- Funding for Short-Term Scientific Missions between participating countries.

Recent and next events

- WG meetings: ETAPS, March 2013; SEFM, September 2013; ETAPS, April 2014; and CONCUR 2014, September 2014.
- 1st International Workshop on Behavioural Types (BEAT) at POPL, January 2013.
- 2nd International Workshop on Behavioural Types (BEAT 2) at SEFM, September 2013.
- 3rd International Workshop on Behavioural Types (BEAT 3) at CONCUR, September 2014.
- International Summer School on Behavioural Types, 2014.
- Further International Workshops on Behavioural Types in 2015 and beyond.
- Funding for Short-Term Scientific Missions between participating countries.

Recent and next events

- WG meetings: ETAPS, March 2013; SEFM, September 2013; ETAPS, April 2014; and CONCUR 2014, September 2014.
- 1st International Workshop on Behavioural Types (BEAT) at POPL, January 2013.
- 2nd International Workshop on Behavioural Types (BEAT 2) at SEFM, September 2013.
- 3rd International Workshop on Behavioural Types (BEAT 3) at CONCUR, September 2014.
- International Summer School on Behavioural Types, 2014.
- Further International Workshops on Behavioural Types in 2015 and beyond.
- Funding for Short-Term Scientific Missions between participating countries.

Recent and next events

- WG meetings: ETAPS, March 2013; SEFM, September 2013; ETAPS, April 2014; and CONCUR 2014, September 2014.
- 1^{st} International Workshop on Behavioural Types (BEAT) at POPL, January 2013.
- 2^{nd} International Workshop on Behavioural Types (BEAT 2) at SEFM, September 2013.
- 3^{rd} International Workshop on Behavioural Types (BEAT 3) at CONCUR, September 2014.
- International Summer School on Behavioural Types, 2014.
- Further International Workshops on Behavioural Types in 2015 and beyond.
- Funding for Short-Term Scientific Missions between participating countries.

Recent and next events

- WG meetings: ETAPS, March 2013; SEFM, September 2013; ETAPS, April 2014; and CONCUR 2014, September 2014.
- 1*st* International Workshop on Behavioural Types (BEAT) at POPL, January 2013.
- 2*nd* International Workshop on Behavioural Types (BEAT 2) at SEFM, September 2013.
- 3*rd* International Workshop on Behavioural Types (BEAT 3) at CONCUR, September 2014.
- International Summer School on Behavioural Types, 2014.
- Further International Workshops on Behavioural Types in 2015 and beyond.
- Funding for Short-Term Scientific Missions between participating countries.

Recent and next events

- WG meetings: ETAPS, March 2013; SEFM, September 2013; ETAPS, April 2014; and CONCUR 2014, September 2014.
- 1^{st} International Workshop on Behavioural Types (BEAT) at POPL, January 2013.
- 2^{nd} International Workshop on Behavioural Types (BEAT 2) at SEFM, September 2013.
- 3^{rd} International Workshop on Behavioural Types (BEAT 3) at CONCUR, September 2014.
- International Summer School on Behavioural Types, 2014.
- Further International Workshops on Behavioural Types in 2015 and beyond.
- Funding for Short-Term Scientific Missions between participating countries.

Recent and next events

- WG meetings: ETAPS, March 2013; SEFM, September 2013; ETAPS, April 2014; and CONCUR 2014, September 2014.
- 1^{st} International Workshop on Behavioural Types (BEAT) at POPL, January 2013.
- 2^{nd} International Workshop on Behavioural Types (BEAT 2) at SEFM, September 2013.
- 3^{rd} International Workshop on Behavioural Types (BEAT 3) at CONCUR, September 2014.
- International Summer School on Behavioural Types, 2014.
- Further International Workshops on Behavioural Types in 2015 and beyond.
- Funding for Short-Term Scientific Missions between participating countries.

Data

Duration

from October 26, 2012 to October 25, 2016

22 participating countries

BA, CY, DE, DK, EE, ES, FR, GR, HR, IE, IT, LT, MK, MT, NL, NO, PL, PT, RO, RS, SE, UK.

Contact details

Chair of the Action: Simon Gay (Simon.Gay@glasgow.ac.uk)

Website: www.behavioural-types.eu

Data

Duration

from October 26, 2012 to October 25, 2016

22 participating countries

BA, CY, DE, DK, EE, ES, FR, GR, HR, IE, IT, LT, MK, MT, NL, NO, PL, PT, RO, RS, SE, UK.

Contact details

Chair of the Action: Simon Gay (Simon.Gay@glasgow.ac.uk)

Website: www.behavioural-types.eu

Data

Duration

from October 26, 2012 to October 25, 2016

22 participating countries

BA, CY, DE, DK, EE, ES, FR, GR, HR, IE, IT, LT, MK, MT, NL, NO, PL, PT, RO, RS, SE, UK.

Contact details

Chair of the Action: Simon Gay (Simon.Gay@glasgow.ac.uk)

Website: www.behavioural-types.eu

In a nutshell

What is it?

COST is an intergovernmental framework for European Cooperation in Science and Technology, allowing the coordination of nationally-funded research on a European level.

Mission and goal

Contributes to reducing the fragmentation in European research investments and opening the European Research Area to cooperation worldwide.

9 Key domains, including

Information and Communication Technologies.

In a nutshell

What is it?

COST is an intergovernmental framework for European Cooperation in Science and Technology, allowing the coordination of nationally-funded research on a European level.

Mission and goal

Contributes to reducing the fragmentation in European research investments and opening the European Research Area to cooperation worldwide.

9 Key domains, including

Information and Communication Technologies.

In a nutshell

What is it?

COST is an intergovernmental framework for European Cooperation in Science and Technology, allowing the coordination of nationally-funded research on a European level.

Mission and goal

Contributes to reducing the fragmentation in European research investments and opening the European Research Area to cooperation worldwide.

9 Key domains, including

Information and Communication Technologies.

Modus operandis

COST funds pan-European, bottom-up networks of scientists, called 'COST Actions', promoting international coordination of nationally-funded research.

The role of COST

COST does not fund research itself, but provides support for networking activities, open to researchers, as well as to NGOs, industry and SMEs, with a four-year duration and a minimum participation of five COST Countries.

A networking tool

COST Actions are active through meetings, workshops, conferences, training schools, short-term scientific missions (STSMs) and dissemination activities.

Modus operandis

COST funds pan-European, bottom-up networks of scientists, called 'COST Actions', promoting international coordination of nationally-funded research.

The role of COST

COST does not fund research itself, but provides support for networking activities, open to researchers, as well as to NGOs, industry and SMEs, with a four-year duration and a minimum participation of five COST Countries.

A networking tool

COST Actions are active through meetings, workshops, conferences, training schools, short-term scientific missions (STSMs) and dissemination activities.

Modus operandis

COST funds pan-European, bottom-up networks of scientists, called 'COST Actions', promoting international coordination of nationally-funded research.

The role of COST

COST does not fund research itself, but provides support for networking activities, open to researchers, as well as to NGOs, industry and SMEs, with a four-year duration and a minimum participation of five COST Countries.

A networking tool

COST Actions are active through meetings, workshops, conferences, training schools, short-term scientific missions (STSMs) and dissemination activities.

Submission of proposals

Open call

http://www.cost.eu/participate/open_call

Basic facts

- Two collection dates a year.
- Response within 8 months.
- Successful proposals "kick-off" within three months thereafter.

More information

http://www.cost.eu/about_cost