ABCD Report: 2017

Simon Gay, Philip Wadler, Nobuko Yoshida

7th December 2017

We summarise this year's activity, organised into sections according to the original ABCD workplan. The report only covers 2017, so it is not a complete report on progress since the beginning of the project.

P1: Industrial Case Studies

WP1-1 Amazon Web Services

WP1-2 Distributed Enterprise Applications

WP1-3 Distributed Applications for Advanced Cyberinfrastructure

A journal version of "Monitoring networks through multiparty session types" on a new theory for dynamic run-time monitoring by Bocchi (Kent), Chen (Darmstadt), Demangeon (UPMC), Honda (QM) and Yoshida (Imperial) has been published in Theoretical Computer Science [2].

WP1-4 Messaging

The work by Neykova and Yoshida (Imperial) [29] is using Python Celery task queue for implementing multiparty session actors.

The work by Neykova, Bocchi (Kent) and Yoshida [32] is using AMQP/RabbitMQ to implement multiparty session types with timers in Scribble and Python.

WP1-5 Review and Organisation of Use Cases

The repository of use-cases has been maintained at `https://github.com/epsrc-abcd/session-types-use-cases`.

P2: Inter-language Interoperability via Session Types

WP2-1 Scribble and Scribble Tool Chain

Hu and Yoshida (Imperial) extended Scribble to support protocols with optional and dynamic participants. The extensions allow endpoints to be connected and disconnected, which is a common pattern in real-world protocols, but were not treated in existing MPSTs. A new bounded global models for MPSTs were developed to validate MPST safety and progress of these enriched protocols, as conservative syntactic restrictions used to ensure safety in standard MPST were insufficient. The work was published in FASE 2017 [16].

WP2-2 Programming Interface, APIs and Runtimes

As the final output of COST Action IC1201 (BETTY), Gay (Glasgow) and Ravara (who was an academic visitor to the ABCD group in Glasgow during 2016/17) edited the book "Behavioural Types: from Theory to Tools" [12]. It consists of sixteen chapters about programming languages and tools based on session types and other forms of behavioural types. Eight of the chapters have authors or co-authors from ABCD, including Glasgow, Edinburgh and Imperial. This shows the substantial influence of ABCD on the European research community. The shift of focus towards programming language and tool implementations has been a notable development in the BETTY community during the last five years, and the work of ABCD has been a large part of it.

Hu (Imperial) contributed a chapter "Distributed Programming Using Java APIs Generated from Session Types" [31] to the book mentioned above, detailing the API generation approach for Java based on Scribble.

Perera and Gay (Glasgow) have collaborated with Lange (Imperial, now Kent) and Morris (Edinburgh, now Kansas) on a theory of communicating objects regarded as automata, with an interactive IDE for checking communication safety. An earlier version of this work was presented at PLACES 2016, and the latest version will be submitted to a conference in 2018.

Neykova and Yoshida (Imperial) developed a runtime monitoring framework for project partners OOI. The design and implementation of the python-based library for runtime monitoring was contributed as a chapter "How to Verify Your Python Conversations" [32] to the Behavioural Types: from Theory to Tools book.

WP2-3 Communication Libraries based on Session Types

A journal version of Neykova and Yoshida's (Imperial) work "Multiparty Session Actors" on applying MPST to actors, has been published in LMCS [29].

WP2-4 Outreach to Developers

Ng (Imperial) spoke at this year's Golang UK 2017 developer conference.

P4: Session Types for Mainstream Programming Languages

WP4-1 Session Types and Typestate in Java

The Glasgow team has continued to work on the Mungo typestate checker for Java, and the StMungo tool that translates Scribble local protocols into typestate specifications. The journal version of our PPDP 2016 paper has been accepted for Science of Computer Programming [20]. We have also contributed a chapter [7] to the book "Behavioural Types: from Theory to Tools".

Weber (Glasgow) has developed a tool for specifying concrete message formats in Scribble, and generating corresponding parsers and formatters for use with the StMungo tool.

Gay and Kouzapas (Glasgow), collaborating with Ravara (Nova University of Lisbon, Portugal; academic visitor to Glasgow in 2016/17), have begun work on inferring typestate specifications for a simple object calculus. The aim is to take dereferencing a null pointer as the semantic error from which constraints on sequences of method calls are derived.

WP4-2 Gradual Session Types in Python

Igarashi (Kyoto University), Thiemann (University of Freiburg), Vasconcelos (University of Lisbon), and Wadler (Edinburgh) have developed a theory of gradual typing for session types. Their work introduces Gradual GV, an extension of the GV language as first described by Wadler, with gradual types, allowing the safe mediation between untyped and session-typed code. This was published at ICFP 2017 [17]. Gradual session types are crucial to the widespread adoption of session types in practice, as they allow session types to be incorporated without rewriting an entire system.

WP4-3 Session C, Many-core, and MPI

Ng and Yoshida (Imperial) contributed a chapter "Protocol-Driven MPI Program Generation" [30] on Parameterised Scribble-based MPI code generation to the book "Behavioural Types: from Theory to Tools"; and collaborating with Vasconcelos, Martins and Marques (ULisbon)

Chapter "Deductive Verification of MPI Protocols" [37] was contributed to the same book, as a tutorial to the MPITypes MPI/C deductive verification framework.

Lange (Kent), Ng, Toninho and Yoshida (Imperial) implemented a static analyser for the Go programming language using a type-based approach to extract and analyse communication deadlock and liveness issues in "Fencing off Go: Liveness and Safety for Channel-based Programming" which was published at POPL 2017 [24] with an evaluated artifact.

WP4-5 Global Types and Scribble

Scalas (Imperial), Dardha (Glasgow), Hu (Imperial) and Yoshida (Imperial) have collaborated on theory and implementation of session types in Scala. The paper was published at ECOOP 2017 [33] with an evaluated artifact [34].

Neykova (Imperial), Bocchi (Kent) and Yoshida (Imperial) worked on an extension of Scribble with timers and published in FAOC [27].

WP4-6 A Website for Session-Typed Languages

Fowler compiled a list of implementations of session types in programming languages, providing links to the tools and to their associated publications. This list is now maintained on the ABCD project website [1].

(Other implementations, which don't fit neatly into work packages)

Dardha and Gay (Glasgow), Fowler (Edinburgh) and Harvey (National Institute of Informatics, Tokyo) are collaborating on the design and implementation of a session type system for Harvey's actor-based language Ensemble. This language has several interesting features including dynamic service discovery, explicit locations and mobile agents, and runtime adaptable code. The session type system provides static guarantees of runtime safety in the presence of these dynamic features. A paper will be submitted to ECOOP 2018.

Yoshida (Imperial) collaborated with Imai (Gifu University, Japan) and Yuen (Nagoya University, Japan) on an implementation of session types in OCaml, but statically enforcing linearity and delegation. Such strong properties were not enforced in any other previous implementations [18].

Orchard (Kent) and Yoshida (Imperial) collaborated to implement session-typed communication in Haskell with static linearity checks. The details of the implementations are published in the "Session Types with Linearity in Haskell" chapter [10] of the Behavioural Types: from Theory to Tools book.

P5: Session Types for Web Applications

WP5-1: Add Session Types to Links

Lindley and Morris (Edinburgh) have implemented asynchronous session-typed channels in the Links functional programming language [26]. Their implementation is grounded in a formal calculus, System FST, which integrates session types, linear types, and row polymorphism.

WP5-2 Exploit Session Types for Distribution

Work at Edinburgh by Fowler, Lindley, Morris, and Decova (an intern) has concentrated on extending the Links language to allow distributed session-typed communication in the web-based setting. A key concern is exception handling. A draft paper currently under review details the design and implementation of the distribution extension to Links, providing a core calculus allowing exception handling in the higher-order, asynchronous setting. Formal properties include global progress in the presence of channel cancellation. Exception handling is implemented using recent work by Lindley and collaborators [14, 15] which extend Links with handlers for algebraic effects.

WP5-3 Reliability and Recovery

In Glasgow, Gay, Kouzapas and Voinea, in collaboration with Gutkovas (Nova University of Lisbon), have continued to work on session types for unreliable broadcast communication. The calculus includes a recovery mechanism for re-synchronisation after undelivered messages. A paper has been submitted to FOSSACS 2018 [22].

Neykova and Yoshida (Imperial) developed a static recovery framework for Erlang by analysing the multiparty session types describing the communication flow of the program. The framework extracts dependencies between processes and localises failures using the session type, and allows program to recover from failure at a computed safe state. This was published at CC2017 [28].

WP5-5 Case Studies and Empirical Evaluation

Most of the work published by the Imperial team contains case studies and evaluations [28, 16, 34, 33, 32, 19, 10, 31, 30, 37, 18, 32, 29, 24].

WP5-6 Infrastructure

At Edinburgh, the Links team has continued to make regular releases. Developers can easily install the latest version of Links using OCaml's `opam` tool.

WP5-7 Tutorial and Dissemination

Yoshida and Neykova (Imperial) taught the pi-calculus and session types in Imperial to Masters and Undergraduate students.

Yoshida taught the pi-calculus and session types in Keio Univeristy in Japan to Masters and Undergraduate students.

Yoshida and Neykova will give a tutorial at CGO/CC/PPPoPP'18 in Vienna.

In 2017, the Imperial group gave over 20 seminars/distinguished lectures/invited keynote talks in various places in UK/EU/Japan/Hong Kong in addition to conferences listed in this document.

Scalas (Imperial) gave a lecture to undergraduate students to introduce ECOOP'17 work.

Most of the talks by the Imperial team are published at `http://mrg.doc.ic.ac.uk/talks/`.

P6: Environments, Modelling, and Empirical Studies

WP6-1 Environments

WP6-2 Modelling Techniques and Session Types

WP6-3 Empirical Studies

P7: Foundations of Session Types

WP7-1 Embracing Races and Deadlock

Kokke's (Edinburgh) MSc thesis describes an extension of CP, inspired by bounded linear logic, which is expressive enough to capture nondeterminism and races whilst retaining deadlock-freedom. A conference submission stemming from Kokke's MSc work, joint with Morris and Wadler (Edinburgh), is currently under review.

In Glasgow, Kouzapas and Gay, in collaboration with Philippou (University of Cyprus), have developed an extension of multiparty session types with non-deterministic behaviour. This allows the specification of interaction scenarios resembling interrupts. A paper has been submitted to FOSSACS 2018 [21].

WP7-2 Multiparty Session Types

A journal version of "Monitoring networks through multiparty session types" on a new theory for dynamic run-time monitoring by Bocchi (Kent), Chen (Darmstadt), Demangeon (UPMC), Honda (QM) and Yoshida (Imperial) has been published in Theoretical Computer Science [2].

A journal version of "Certifying data in multiparty session types" by Toninho and Yoshida (Imperial) has been published in JLAMP [36].

Scalas and Yoshida (Imperial) developed a general framework for multiparty sessions which can type more processes than usual projection-based approaches in the literature is proposed in [35]

The first work which proposes linear-logic-based multiparty session types was selected for the special issue of CONCUR 2017 and published in Acta Informatica [3].

WP7-3 Productive Streams

WP7-4 Extensions to match BPMN 2.0

WP7-5 Synthesis

Based on the relationship between choreographies, CFSMs and session types, Lange (Kent), Tuosto (Leicester) and Yoshida (Imperial) contributed a chapter [19] to the "Behavioural Types: from Theory to Tools" book describing the choreography analysis and synthesis tool ChorGram. This work also relates to WP7-4.

WP7-6 Relationship with other formalisms

Lange (Kent) and Yoshida (Imperial) showed the undecidability of asynchronous session subtyping, the prove leverages the relationships between session types and CFSMs. The work was published in FoSSaCS 2017 [25].

(Other P7 papers, which don't fit neatly into WPs)

A journal version of "On the Preciseness of Subtyping in Session Types" which studied subtyping preciseness by Chen (Darmstadt), Dezani (Torino), Scalas and Yoshida (Imperial) has been published in LMCS [5].

Kouzapas (Glasgow), Perez (Groningen) and Yoshida (Imperial) published journal version of Characteristic Bisimulation for Higher-Order Session Processes [23] .

Fowler, Lindley, and Wadler (Edinburgh) have developed core calculi comparing type-parameterised channels and actors, thereby dispelling confusion and developing formal connections between the two models. This was published at ECOOP 2017 [11]. They show the issues with naïvely adding types to actors, and argue the difficulties of adding behavioural types to mailboxes directly. Finally, they show that their minimal actor calculus can encode complex actor-based features such as selective receive, as found in Erlang.

In Glasgow, Dardha and Gay have developed a variant of classical linear logic that combines the CP type system and Kobayashi's type systems for deadlock-freedom based on priority tags. This gives a Curry-Howard correspondence for a session type system which types more deadlock-free processes than CP does. A paper has been submitted to FOSSACS 2018 [6].

The journal version of Dardha's paper "Session Types Revisited", with Giachino and Sangiorgi, has been published in Information and Computation [8].

Dardha's work on semantic subtyping for objects, with Gorla and Varacca, has been published in the Computer Journal [9].

Graversen, Philipps and Yoshida (Imperial) study categorical representation of reversible event structures [13] and Castellan (Imperial) published a survey work on Concurrent Game Semantics [4]. The Imperial team is currently working to apply event-structure based game semantics to session pi-calculi.

Outreach to Developers

Wadler (Edinburgh) spoke on category theory to QCon Sao Paulo, QCon San Francisco, and Apple Computer, and on domain specific languages to Google X. Wadler will deliver a keynote at Lambda Days, Krakow, February 2017.

Other activities

Gay, Vasconcelos, Wadler and Yoshida organised a Dagstuhl workshop on Theory and Applications of Behavioural Types, which took place in January 2017. We were limited in the number of people who could be invited from ABCD, but in addition to the organisers, Kouzapas (Glasgow), Morris (Edinburgh), Ng (Imperial) and Scalas (Imperial) attended. Other participants included former ABCD members Bocchi (Kent) and Orchard (Kent), and ABCD advisory board members Dezani (Torino) and Pfenning (CMU).

References

[1] Session Types in Programming Languages: A Collection of Implementations . `http://groups.inf.ed.ac.uk/abcd/session-implementations.html`, 2017.

[2] L. Bocchi, T.-C. Chen, R. Demangeon, K. Honda, and N. Yoshida. Monitoring networks through multiparty session types. *Theoretical Computer Science*, 669:33 – 58, 2017.

[3] M. Carbone, F. Montesi, C. Schürmann, and N. Yoshida. Multiparty session types as coherence proofs. *Acta Informatica*, nov 2016.

[4] S. Castellan. Concurrent Structures in Game Semantics. *The Concurrency Column, EATCS Bulletin*, pages 1–31, 2017.

[5] T.-c. Chen, M. Dezani-Ciancaglini, A. Scalas, and N. Yoshida. On the Preciseness of Subtyping in Session Types. *Logical Methods in Computer Science*, Volume 13, Issue 2, Jun 2017.

[6] O. Dardha and S. J. Gay. A New Linear Logic for Deadlock-Free Session-Typed Processes. Submitted to FoSSaCS 2018.

[7] O. Dardha, S. J. Gay, D. Kouzapas, R. Perera, A. L. Voinea, and F. Weber. Mungo and stmungo: tools for typechecking protocols in java. In S. Gay and A. Ravara, editors, *Behavioural Types: from Theory to Tools*, River Publishers Series in Automation, Control and Robotics, pages 309–328. River Publishers, June 2017.

[8] O. Dardha, E. Giachino, and D. Sangiorgi. Session types revisited. *Inf. Comput.*, 256:253–286, 2017.

[9] O. Dardha, D. Gorla, and D. Varacca. Semantic subtyping for objects and classes. *Comput. J.*, 60(5):636–656, 2017.

[10] Dominic Orchard and Nobuko Yoshida. Session Types with Linearity in Haskell. In Simon Gay and Antonio Ravara, editor, *Behavioural Types: from Theory to Tools*, chapter 10, pages 219–242. River publishers, 2017.

[11] S. Fowler, S. Lindley, and P. Wadler. Mixing metaphors: Actors as channels and channels as actors. In *ECOOP*, volume 74 of *LIPIcs*, pages 11:1–11:28. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2017.

[12] S. J. Gay and A. Ravara, editors. *Behavioural Types: from Theory to Tools.* River Publishers, 2017.

[13] E. Graversen, I. Phillips, and N. Yoshida. Towards a Categorical Representation of Reversible Event Structures. In *10th International Workshop on Programming Language Approaches to Concurrency- and Communication-cEntric Software*, volume 246 of *EPTCS*, pages 49–60. Open Publishing Association, 2017.

[14] D. Hillerström and S. Lindley. Liberating effects with rows and handlers. In *Proceedings of the 1st International Workshop on Type-Driven Development - TyDe 2016.* ACM, 2016.

[15] D. Hillerström, S. Lindley, R. Atkey, and K. Sivaramakrishnan. Continuation passing style for effect handlers. In *FSCD*, volume 84 of *LIPIcs*. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2017.

[16] R. Hu and N. Yoshida. Explicit Connection Actions in Multiparty Session Types. In *20th International Conference on Fundamental Approaches to Software Engineering*, LNCS. Springer, 2017.

[17] A. Igarashi, P. Thiemann, V. T. Vasconcelos, and P. Wadler. Gradual Session Types. *Proceedings of the ACM on Programming Languages*, 1(ICFP):38, 2017.

[18] K. Imai, N. Yoshida, and S. Yuen. Session-ocaml: A session-based library with polarities and lenses. In J.-M. Jacquet and M. Massink, editors, *Coordination Models and Languages: 19th IFIP WG 6.1 International Conference, COORDINATION 2017, Held as Part of the 12th International Federated Conference on Distributed Computing Techniques, DisCoTec 2017, Neuchâtel, Switzerland, June 19-22, 2017, Proceedings*, pages 99–118. Springer International Publishing, 2017.

[19] Julien Lange and Emilio Tuosto and Nobuko Yoshida. A Tool for Choreography-Based Analysis of Message-Passing Software. In Simon Gay and Antonio Ravara, editor, *Behavioural Types: from Theory to Tools*, chapter 6, pages 125–146. River publishers, 2017.

[20] D. Kouzapas, O. Dardha, R. Perera, and S. J. Gay. Typechecking protocols with mungo and stmungo: a session type toolchain for java. *Science of Computer Programming*, October 2017.

[21] D. Kouzapas, S. J. Gay, and A. Philippou. Non-deterministic Multiparty Session Types. Submitted to FoSSaCS 2018.

[22] D. Kouzapas, R. Gutkovas, A. L. Voinea, and S. J. Gay. A Session Type System for Asynchronous Unreliable Broadcast Communication. Submitted to FoSSaCS 2018.

[23] D. Kouzapas, J. A. Pérez, and N. Yoshida. On the relative expressiveness of higher-order session processes. In *Programming Languages and Systems*, pages 446–475. Springer Nature, 2016.

[24] J. Lange, N. Ng, B. Toninho, and N. Yoshida. Fencing off Go: Liveness and Safety for Channel-based Programming. In *Proceedings of the 44th ACM SIGPLAN Symposium on Principles of Programming Languages*, pages 748–761. ACM, 2017.

[25] J. Lange and N. Yoshida. On the Undecidability of Asynchronous Session Subtyping. In *20th International Conference on Foundations of Software Science and Computation Structures*, LNCS. Springer, 2017.

[26] S. Lindley and J. G. Morris. Lightweight functional session types. In S. Gay and A. Ravara, editors, *Behavioural Types: from Theory to Tools*, chapter 12, pages 265–286. River publishers, 2017.

[27] R. Neykova, L. Bocchi, and N. Yoshida. Timed runtime monitoring for multiparty conversations. *Formal Aspects of Computing*, 29(5):877–910, Sep 2017.

[28] R. Neykova and N. Yoshida. Let It Recover: Multiparty Protocol-Induced Recovery. In *26th International Conference on Compiler Construction.* ACM, 2017.

[29] R. Neykova and N. Yoshida. Multiparty Session Actors. *Logical Methods in Computer Science*, Volume 13, Issue 1, 2017.

[30] Nicholas Ng and Nobuko Yoshida. Protocol-Driven MPI Program Generation. In Simon Gay and Antonio Ravara, editor, *Behavioural Types: from Theory to Tools*, chapter 15, pages 329–352. River publishers, 2017.

[31] Raymond Hu. Distributed Programming Using Java APIs Generated from Session Types. In Simon Gay and Antonio Ravara, editor, *Behavioural Types: from Theory to Tools*, chapter 13, pages 287–308. River publishers, 2017.

[32] Rumyana Neykova and Nobuko Yoshida. How to Verify Your Python Conversations. In Simon Gay and Antonio Ravara, editor, *Behavioural Types: from Theory to Tools*, chapter 4, pages 77–98. River publishers, 2017.

[33] A. Scalas, O. Dardha, R. Hu, and N. Yoshida. A linear decomposition of multiparty sessions for safe distributed programming. In *31st European Conference on Object-Oriented Programming, ECOOP 2017, June 19-23, 2017, Barcelona, Spain*, volume 74 of *LIPIcs*, pages 24:1–24:31. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2017.

[34] A. Scalas, O. Dardha, R. Hu, and N. Yoshida. A linear decomposition of multiparty sessions for safe distributed programming (artifact). *DARTS*, 3(2):03:1–03:2, 2017.

[35] A. Scalas and N. Yoshida. Multiparty Session Types, Beyond Duality. In *10th International Workshop on Programming Language Approaches to Concurrency- and Communication-cEntric Software*, volume 246 of *EPTCS*, pages 37–38. Open Publishing Association, 2017.

[36] B. Toninho and N. Yoshida. Certifying data in multiparty session types. *Journal of Logical and Algebraic Methods in Programming*, 90:61 – 83, 2017.

[37] Vasco T. Vasconcelos and Francisco Martins and Eduardo R.B. Marques and Nobuko Yoshida and Nicholas Ng. Deductive Verification of MPI Protocols. In Simon Gay and Antonio Ravara, editor, *Behavioural Types: from Theory to Tools*, chapter 16, pages 353–372. River publishers, 2017.