Institution: University of Edinburgh		
Unit of Assessment: 11		
Title of case study: Effect handlers for increased reach, effectiveness, and future-proofing in software products and services		
Period when the underpinning research was undertaken: 2009 – 2019		
Details of staff conducting the underpinning research from the submitting unit:		
Name(s):	**Role(s) (e.g. job title):**	**Period(s) employed by submitting HEI:**
Gordon Plotkin	Professor	1971 – present
Samuel Lindley	Research Fellow	2010 – 2012
	Senior Research Fellow	2013 – 2019
Nicolas Oury	Research Fellow	2009 – 2012
Ohad Kammar	Senior Research Fellow	2019 – present
Period when the claimed impact occurred: 2014 – 2020		
Is this case study continued from a case study submitted in 2014? No		

1. Summary of the impact

Researchers at the University of Edinburgh (UoE) invented the Effect Handler programming construct, and proved it to be a versatile, highly expressive programming abstraction. Companies worldwide have adopted Effect Handlers for software infrastructure, including GitHub, Uber and Facebook. Across these companies, Effect Handlers have brought commercial benefits ranging from improved developer productivity, in turn yielding more cost-effective working practices (for GitHub and Uber); to increased product performance (for Uber and Facebook). Effect Handlers have been strategically important for the three companies, and ultimately benefit a diverse population of billions of end-users, from developers collaborating on GitHub, to commercial and private users of Facebook's underpinning software architecture, React.

2. Underpinning research

Programming language researchers investigate new programming constructs, through programming language modelling, design, and implementation. Desirable properties of such constructs include the ability to make modular abstractions, allowing software development teams to scale and achieve larger goals; expressivity, enabling software to evolve alongside developer needs; and runtime efficiency, allowing software systems to scale to larger tasks and throughput. Over the last decade, researchers in the University of Edinburgh's (UoE) Laboratory for the Foundations of Computer Science's (LFCS) programming language group invented and implemented a modular control-flow programming construct, the Effect Handler. In 2009, Professor Gordon Plotkin and his PhD student Dr Matija Pretnar presented an algebraic treatment of exception handlers, and introduced handlers for other computational effects representable by an algebraic theory [3.1]. In this formalism, programmers can define *effects* and invoke them to interrupt the program's execution and invoke an appropriate handler code. However, unlike exception handlers, an Effect Handler can resume the original point of execution, sometimes repeatedly. Effect Handlers promise a highly modular and expressive programming style, and Plotkin and Pretnar gave a theoretical demonstration of their versatility [3.1].

Subsequently, Dr Sam Lindley, Dr Nicolas Oury, and Dr Ohad Kammar (then a UoE PhD student) popularised Effect Handlers by giving simpler descriptions for them and demonstrating how to incorporate them into existing programming languages [3.2]. Effect Handlers were originally presented by Plotkin and Pretnar in a style that required considerable mathematical sophistication to understand, namely categorical algebra and denotational semantics. The researchers gave an operational description using a high-level state machine. They also implemented Effect Handlers in existing programming languages (Haskell, SML, OCaml, Lisp, Racket), giving programmers and programming language researchers a hands-on opportunity to learn about and experiment with them.

Lindley and Kammar explored the expressive power of Effect Handlers, and proved they are highly expressive [3.4]. These theoretical results make Effect Handlers a future-proof programming abstraction. Moreover, this theoretical development incorporates several source-to-source translations that form the basis for the more practical work, including Lindley's *continuation passing style* translations, which offer a practical implementation technique of effect handlers that does not require special support in a target language's runtime [3.3].

Lindley and his PhD student Daniel Hillerström showed how to implement Effect Handlers efficiently in web applications [3.3]. This technique formed the basis for the programming language OCaml's run-time for Effect Handlers, which Hillerström continued in a research internship at OCaml Labs Cambridge during his PhD. Lindley continued to develop implementation techniques and type systems for Effect Handlers with PhD students Hillerström [3.5] and Dr Craig McLaughlin [3.6].

Plotkin, Lindley and Kammar led and grew the Effect Handlers research community and its ties with industry by participating and co-organising international research meetings such as Dagstuhl (#16112 and #18172) and Shonan (#146), attended by academic and industrial researchers.

3. References to the research

3.1. Plotkin, G., & Pretnar, M. (2009). Handlers of Algebraic Effects. In *Castagna G. (eds) Programming Languages and Systems. ESOP 2009* (pp. 80-94). (Lecture Notes in Computer Science, vol 5502). Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-00590-9_7 **(188 citations)**

3.2. Kammar, O., Lindley, S., & Oury, N. (2013). Handlers in action. In *Proceedings of the 18th ACM SIGPLAN international conference on Functional programming* (*ICFP '13*). (pp. 145-158). Association for Computing Machinery, New York, NY, USA, 145–158. https://doi.org/10.1145/2500365.2500590 **(based on research conducted at UoE in 2012; 143 citations)**

3.3. Hillerström, D., Lindley, S., Atkey, R., & Sivaramakrishnan, KC. (2017). Continuation passing style for effect handlers. In *Miller D. (ed.) Proceedings of the 2nd International Conference on Formal Structures for Computation and Deduction. FSCD 2017* (pp. 18:1-18:19). (Leibniz International Proceedings in Informatics (LIPIcs), vol 84). Schloss Dagstuhl—Leibniz-Zentrum fuer Informatik. Dagstuhl, Germany. https://doi.org/10.4230/LIPIcs.FSCD.2017.18 **(32 citations)**

3.4. Foster, Y., Kammar, O., Lindley, S., & Pretnar, M. (2017). On the expressive power of user-defined effects: effect handlers, monadic reflection, delimited control. In *Proceedings of the ACM on Programming Languages (PACMPL), 1(ICFP)*, 13:1-13:29. [13]. https://doi.org/10.1145/3110257 **(42 citations)**

3.5. Hillerström, D., & Lindley, S. (2016). In *Proceedings of the 1st International Workshop on Type-Driven Development* (*TyDe 2016*). Association for Computing Machinery, New York, NY, USA, (pp. 15–27). https://doi.org/10.1145/2976022.2976033 **(70 citations)**

3.6. Lindley, S., McBride, C., & McLaughlin, C. (2017). Do be do be do. In *Proceedings of the 44th ACM SIGPLAN Symposium on Principles of Programming Languages* (*POPL 2017*) (pp. 500-514). Association for Computing Machinery, New York, NY, USA, https://doi.org/10.1145/3009837.3009897 **(POPL 2017 acceptance rate 20%)**

Citation data from Google Scholar, 2020-12-17.

Key research grants
EPSRC: From Data Types to Session Types (EP/K034413/1, GBP4,945,110)

4. Details of the impact

The Effect Handlers abstraction has been adopted for software infrastructure by companies worldwide, contributing to their business strategy, enabling increased productivity, and leading to resource-use efficiency improvements. Three notable companies – GitHub, Uber and Facebook – used Effect Handlers to benefit an exceptionally large number of users.

GitHub, Inc. hosts the largest collection of programming language source code repositories in the world. The company's Semantic team develop code intelligence tools for semantic analysis, like semantic code navigation. Central to the code intelligence tools is an Effect Handler library that the GitHub team developed, underpinned by the core abstractions first developed by UoE (3.1, 3.2) [5.1, para. 2].

Semantic's Effect Handlers operate at scale, analysing thousands of pull-requests per day, continuously [5.1, para 1]. Effect Handlers enable Semantic's analysis tools to support multiple programming languages, giving these tools extraordinary reach. Semantic's tools cover over 56% of the thousands of pull requests per day in these languages, across 100,000,000 repositories [5.1, paras. 1, 4]. This wide reach is only possible through supporting a diverse collection of languages, since each language covers under 25% of the total repositories on GitHub [5.1, para. 1]. Moreover, the support enabled by Effect Handlers is unusually cost-effective: the Semantic team confirm they can "offer [their] users a service that would otherwise be prohibitively expensive [...keeping] development and maintenance costs remarkably low", with only 8 developers on the team. Effect Handlers also give GitHub a strategic advantage, as they confirm that "a future-proof Effect Handlers library contributes to GitHub's long-term strategy for offering code intelligence products 'out of the box'" [5.1, paras. 4-5].

Effect Handlers are central to these benefits. Analysing a language involves controlling a combinatorial explosion of thousands of subtle inter-related parameters. The company confirms:

> *Without a modular abstraction like Effect Handlers, we would likely need a dedicated team for each combination of language and analysis parameters, rendering the endeavour too costly.* [5.1, para. 4]

With Effect Handlers, a small development suffices. The strategic advantage, future-proofing the product, is one of the Effect Handler library's design goals, "making it easy to add new languages as well as new analyses in order to support more advanced language features." [5.1, para. 4]

Uber Technologies, Inc. provides a diverse number of services from vehicles for hire to food and package delivery. It operates the largest ride-hailing platform in the world. Uber faces many machine-learning problems, involving large amounts of data. In 2017, grounded in their desire to advance the state of the art in machine intelligence, Uber released the Pyro open-source programming language embodying how they "believe statistical modelling should be done" [5.2, para. 2].

Uber used Pyro to debug, prototype and automate statistical modelling pipelines, freeing staff from having to constantly monitor and maintain these systems manually, avoiding wastage of dollars [5.2, para. 6]. The production system requires very little oversight, enabling teams to focus on higher level strategy and execution instead of repetitive monitoring tasks. Uber credits Effect Handlers with enabling these benefits from Pyro. They note that implementing a system like Pyro without Effect Handlers would have been impractical due to code duplication, resulting in many bugs [5.2, para. 5]. Effect Handlers allow modellers to change inference algorithms with low effort, without changes to the model. For example, Pyro's conjugate marginalisation Effect Handler "led to a 10x faster performance than comparable models in Stan – the other principal industrial standard probabilistic programming language" [5.2, para. 7]. Pyro's reference manual [5.3, final page] refers readers directly to University of Edinburgh's underpinning research (3.1, 3.2). This library lets users "answer complex questions with minimal additional code." [5.2, para. 5] The modularity of Pyro, achieved through Effect Handlers, future-proofs these statistical pipelines, which is "an important strategic consideration" for Uber [5.2, para. 7].

One concrete success story at Uber is a Pyro model deployed at scale in advertising budget allocation, running automatically, continually, and daily. This marketing problem is strategically important for Uber due to marketing's direct contribution to the vitality of their ride hailing platform and the company's global volume of operation: Uber works in approximately 10,000 cities across 69 countries [5.2, paras. 2 and 8]. The company's total marketing and sales budget in 2019 was USD4,600,000,000 (09-2020), a USD1,500,000,000 (09-2020) increase from 2018. Even a 1% increase in return on advertising spend translates into millions of US dollars per annum [5.2, para. 8]. Uber confirms:

> *Pyro has more than delivered on its promise of making probabilistic modelling and inference easier, and these models have been used by the marketing team to manage important campaigns.* [5.2, para. 8]

Facebook, Inc. forms the largest social media conglomerate corporation in the world, developing and maintaining multiple social media platforms reaching billions of users. Key products include the Facebook app (more than 2,000,000,000 users), Messenger, and Instagram (more than 1,000,000,000 users, each). The user-interface in these platforms is based on the Facebook's React JavaScript library [5.4, para. 1].

React used Effect Handlers in the design of its API, in the form of two new programming concepts: Hooks and Suspense. Both make user-interface code simpler and cheaper to develop and extend, and the user interface more reactive. Facebook has made using Hooks mandatory across its codebase, including its influential Messenger and Instagram platforms, and most notably in the website design for the Facebook platform [5.4, paras. 6-8]. Moreover, React is a popular library outside Facebook, with 2,000,000 active developers worldwide [5.4, para. 1]. It is widely adopted across diverse business sectors by companies such as Twitter, Airbnb, Uber, Dropbox, Microsoft Office, online education platforms Khan Academy and Codecademy, and news outlets such as the New York Times and the BBC [5.4, para. 2]. A recent survey conducted in the React community showed that 70% of respondents prefer to use Hooks, and only 7% prefer not to use them, and that 50% use Hooks in production [5.4, para. 9]. Hooks have been adopted as the new industry-standard for web user-interface design, with all competing frameworks directly referencing Hooks as a feature they want to support [5.5-5.8].

The React team relies on the expressivity and compositionality of Effect Handlers to future-proof their API design, describing the UoE programming construct as "strategically important" [5.4, penultimate para.]. They credit Effect Handlers with enabling improvements to React's compositionality, and also state:

> *Effect Handlers are an expressive concept based on rigorous mathematical models. These properties give us further assurance that they will continue to deliver flexible designs and utilise resources efficiently as our needs develop, allowing us to make more informed strategic decisions today.* [5.4, penultimate para.]

React developers attribute these benefits directly to the UoE research, citing Effect Handlers as a "key concept inspiring the current design of the React API" [5.4, final para.]. Their documentation references Multicore OCaml's algebraic effects [5.9, final para.], an implementation of Effect Handlers derived from UoE's underpinning research (3.1-3.3). Facebook concludes:

> *The React ecosystem — the largest on the web — has widely embraced this Effect-Handler-based design, and it is used in production code in an essential way.* [5.4, final para.]

Overall, Effect Handlers underpin technologies used by companies across the software industry, reaching billions of users and clients daily. The programming construct developed at UoE has improved production efficiency, and empowered teams to develop clean, flexible, and expressive code. Effect Handlers has enabled companies to provide future-proof products and services, and drives strategic decision-making.

5. Sources to corroborate the impact

5.1. Letter of corroboration from the Senior Engineer Manager of Semantic, GitHub, Inc.
5.2. Letter of corroboration from a Senior Engineer Manager, formerly at Uber AI and a core member/maintainer of the Pyro project.
5.3. Uber Technologies, Inc. (2018). Poutine: A Guide to Programming with Effect Handlers in Pyro. Retrieved December 7, 2020, from http://pyro.ai/examples/effect_handlers.html
5.4. Letter of corroboration from the React team, Facebook, Inc.
5.5. Vuejs. (2020, November 17). Vuejs/rfcs: Introducing the Composition API. Retrieved February 9, 2021, from https://github.com/vuejs/rfcs/blob/master/active-rfcs/0013-composition-api.md
5.6. Harris, R. (2019, February 06). 100% true. When hooks were first announced, it forced me to acknowledge the flaws in the original design for Svelte components… Rising tide lifts all boats! Retrieved January 6, 2021, from https://twitter.com/rich_harris/status/1093260097558581250?lang=en
5.7. Richardson, L. (2019, July 12). React, Meet Compose. Retrieved January 6, 2021, from https://speakerdeck.com/lelandrichardson/react-meet-compose?slide=19
5.8. Wadhwa, A. (2020, September 06). Hooks in Flutter. Retrieved January 7, 2021, from https://medium.com/flutterdevs/flutter-hooks-67b24d47cb36
5.9. Facebook, Inc. (2020). Hooks FAQ. Retrieved January 7, 2021, from https://reactjs.org/docs/hooks-faq.html#what-is-the-prior-art-for-hooks