Institution: University of Oxford		
Unit of Assessment: 11 Computer Science and Informatics		
Title of case study: Improving Software Security via Variant Analysis		
Period when the underpinning research was undertaken: 2001 – 2007		
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:**
Professor Oege de Moor	Professor of Computer Science	1994 – 2015
Ganesh Sittampalan	PDRA	2001 – 2004
David Lacey	PDRA	2001 – 2004
Eric Van Wyk	PDRA	1999 – 2001
Damien Sereni	PDRA	2006 – 2008
Period when the claimed impact occurred: 1 August 2013 – 31 July 2020		
Is this case study continued from a case study submitted in 2014? Y		

1. Summary of the impact

The Oxford programming tools team led by Oege de Moor developed a novel approach to program analysis that allows for automated application to complex problems and extensive amounts of code. Based on this research, de Moor and colleagues founded the successful spinout company Semmle. Semmle's solution for "variant analysis" has benefitted product security teams at Google, Microsoft, Dell, Credit Suisse, Nasdaq, Uber, and many other organisations since 2013. The company's open, community-driven approach to security – its free service for open source had a user community of over 700,000 developers in 2019 – has also provided wider economic and public benefit by helping to secure open source code. The success of this approach led to Semmle's 2019 acquisition by the leading developer community platform GitHub, for an undisclosed sum estimated at over USD400,000,000. Semmle employed over 80 people across six sites worldwide at the time of the acquisition.

2. Underpinning research

In the early 2000s, the programming tools team at Oxford made considerable advances in a line of research on declarative program analysis. The initial emphasis was on functional programming. A major step, achieved in 2001 [**R1**], was the development of a language for implementing optimising compiler transformations based on a combination of rewriting and side conditions phrased in temporal logic. A salient feature was the ability to bind free variables while matching the temporal logic formulae. This built on prior work by Sittampalam and de Moor, who had already created an efficient higher-order matching algorithm that was an example of selecting a pattern language in the sweet spot between expressivity (a bit more than 2nd order but not all of 3rd order) and efficiency. The implementation also heavily relied on ideas from logic programming. The results in [**R1**] sparked a lot of interest in the research community, with groups led by Neil Jones (Copenhagen) and by Craig Chambers (University of Washington) taking up and further developing the theme. The link to restricted forms of logic programming became even more apparent at this point, and joint work by Sittampalam, Lacey, De Moor, and others explored this connection to express particular kinds of recursive analyses as logic programs that query control flow graphs [**R2**]. Again, the key was to strike the right balance between expressivity and efficiency, by not using the full power of a Turing-complete language.

At that time 'aspect-oriented programming' was coming into prominence, and it appeared a perfect area of application for the technologies developed in the programming tools team. Briefly, aspects are a means to specify a declarative monitor to check properties in the execution of a software system. In 2003, we showed that the earlier work with Sittampalam could achieve dramatic speedups in the execution of aspect-oriented programs [**R3**]. This became one of the most cited papers in aspect-oriented programming, and provided the foundation for our implementation of a new optimising compiler for the most popular aspect-oriented programming language, AspectJ, demonstrating that the theoretical speedups could be achieved at an industrial scale [**R4**].

Meanwhile, a number of other research groups, including Laurie Hendren's at McGill and Monica Lam's at Stanford, had also started using Datalog, a much-restricted logic programming language, for program analysis. Hendren spent a sabbatical at Oxford in 2004, and her implementation via binary decision diagrams was used in the above work on aspect-orientation. At Oxford, de Moor's group also provided a Datalog implementation for experimentation using traditional database technology instead of binary decision diagrams, building on previous work by Thomas Reps at Wisconsin (who also used it for program analysis applications) and others. It became apparent that no one understood the semantics of the event patterns (called "pointcuts") of AspectJ properly, and therefore a clear semantics of its pattern language was required. The programming tools team addressed this question by translating that pattern language into Datalog [**R5**], and giving a simple implementation through that translation [**R6**]. By this point, the potential of Datalog as a basis for software analysis was evident.

3. References to the research

[**R1**] D. Lacey, O. de Moor: Imperative Program Transformation by Rewriting. Compiler Construction (CC), 2001: https://doi.org/10.1007/3-540-45306-7_5.

[**R2**] O. de Moor, D. Lacey, E. Van Wyk: Universal Regular Path Queries. Higher Order and Symbolic Comp., 2003: https://doi.org/10.1023/A:1023063919574.

[**R3**] D. Sereni, O. de Moor: Static analysis of aspects. Aspect-oriented Software Dev. (AOSD), 2003: https://doi.org/10.1145/643603.643607.

[**R4**] O. de Moor, D. Sereni, G. Sittampalam et al.: Optimising aspectJ. Prog. Lang. Design and Implementation (PLDI), 2005: https://doi.org/10.1145/1065010.1065026. *Submitted to RAE2008*.

[**R5**] O. de Moor, D. Sereni et al.: Semantics of static pointcuts in aspectJ. Principles of Prog. Lang. (POPL), 2007: https://doi.org/10.1145/1190215.1190221.

[**R6**] E. Hajiyev, M. Verbaere, O. de Moor: codeQuest: Scalable Source Code Queries with Datalog. Eur. Conf. on Object-Oriented Prog. (ECOOP), 2006: https://doi.org/10.1007/11785477_2. *Submitted to RAE2008*.

4. Details of the impact

Pathway to impact. Semmle was founded in 2006 to create the novel technology that realises the potential of Datalog for software analysis demonstrated in the Oxford team's research, widening the scope of application to business intelligence rather than just program analysis. Six US patents were filed by Semmle to protect these further advances after the creation of the company [**E1**]. In September 2014 Semmle raised USD8,000,000 Series A investment from the leading venture capital firm Accel Partners. This allowed the team to expand from its Oxford office to two new sites in Copenhagen and New York, and to make the product enterprise-ready. The company grew from 18 employees in 2014 (FTEs: 18), to 35 following Series A investment (FTEs: 35), to 48 by 2018 (FTEs: 48). In August 2018 USD21,000,000 Series B funding was raised from a consortium led by Accel Partners. By the time of its 2019 acquisition by GitHub, a Microsoft subsidiary since its USD7,500,000,000 acquisition in 2018, Semmle employed 81 staff (FTEs: 81) across six offices in Oxford, San Francisco, New York, Seattle, Valencia, and Copenhagen [**E2**, **E14**.3]. It is not possible to give confidential information about company revenue, or the sale price. However, business insights company Owler estimates 2019 annual revenue at USD10,000,000, and financial data company PitchBook estimates the sale price at USD410,000,000 [**E14**.1, **E14**.2].

Semmle's technology is built on a novel approach to program analysis that combines two disparate disciplines, object-oriented programming and database logic. By treating source code as a relational database, and analysis problems as queries against a database, deep semantic analyses can be expressed as concise queries in an object-oriented query language. The creation of such queries is an order of magnitude quicker than previous methods of creating code analyses. The query language is an object-oriented form of

Datalog, whose evolution can be traced in [**R1**–**R6**]. Cutting-edge techniques are used to make the queries perform well. Semmle's analysis engine thus makes software easily and accurately searchable, allowing complex questions to be asked at a previously unachievable pace and scale [**E3**, and below]. This REF period coincided with the application of Semmle's technology to its 'killer' commercial use case, software security. Semmle's unique solution for variant analysis – when a new vulnerability is identified in a software system, to find all occurrences of the same logical mistake in the same code base, or indeed in a portfolio of codebases – has delivered business-critical benefit to numerous major global clients since August 2013. These include Behavox, Credit Suisse, Dell, Google, Microsoft, Mozilla, Murex, NASA, Nordea, Uber, and other clients who cannot be named publicly. Three examples are described below. The period also saw the growth of Semmle's wider contribution to software development and integrity through its open and shared approach to security. It made its CodeQL analysis engine freely available through an open analysis platform, LGTM.com, used by over 700,000 developers [**E7**], and worked closely with top commercial security teams and the open source community to find and report vulnerabilities in widely used software [**E5**.1–3, **E6**].

Software security and development in industry. Microsoft's products and services are used by billions of individuals and millions of companies every day. When a software vulnerability is identified or reported, failure to find and patch all variants at the same time increases the risk of code being exploited in the wild. Since 2017 Microsoft has incorporated Semmle automation into its code review processes, using it to analyse bugs reported in a given component (for example the JavaScript engine of the Edge browser) and define queries that can find similar patterns in other components. A single such query has in many cases "provided a number of actionable results across multiple codebases". Microsoft then store these queries in a central repository, to be re-run periodically by security teams across the organisation. In addition to variant analysis, Microsoft's software researchers use Semmle's technology proactively to review source code and identify vulnerabilities. Its ability to identify even the most complex semantic patterns means that an "explorative approach" can be maintained while automating what would otherwise be a "tedious and error-prone" process of manual audit. "Using Semmle to scale up our code review capabilities was an easy choice" (Security Software Engineer, Microsoft Security Response Center) [**E3**, **E4**].

Nasdaq Corporate Solutions provides business and market intelligence through software and consultative services to thousands of organisations globally. After benchmarking static analysis tools and software analysis products, Global Corporate Solutions Technology (GCST) at Nasdaq chose Semmle's LGTM product in 2018 as the only solution capable of running the continuous and accurate analytics needed to monitor development standards across a growing and highly complex application portfolio. Nasdaq have used Semmle analysis to provide actionable insights for strategic decision-making and budgeting across the organisation, enabling improved tracking of development projects, increasing the efficiency of engineering teams, and benefitting the individual developers who use Semmle's tools. As a result, Nasdaq "completely changed the way people are writing code" (SVP of Global Corporate Solutions Technology), and brought down "technical debt" across the application portfolio (i.e., the implied future time and development cost accrued by using limited or imperfect solutions in the short term) by more than 75% within one year. This reduced software risk, and freed developer time for value-creating tasks [**E3**].

BlackLine is a leading global provider of cloud financial software, serving over 2,300 organisations in more than 150 countries. After its initial public offering in late 2016, BlackLine turned to Semmle to meet the challenge of increasing the complexity of its code architecture by the addition of new software features and developments, while maintaining the security and integrity of its clients' critical financial data. Blackline used the technology to identify complex data integrity issues, automate code review, and implement robust coding standards across the organisation. Blackline's Director of Software Development was "blown away by [Semmle's] power"; "no automated solution we had tried was able to find problems of this complexity, but using Semmle we became able to quickly find these types of issues across our portfolio. Within weeks this eliminated manual efforts that were consuming significant cycles of our SDLC" (Software Development Life Cycle). Using Semmle to enforce the organisation’s coding standards, moreover, enabled BlackLine “to reduce our maintenance spend drastically”. “On top of that, our developers are relieved of routine tasks related to maintenance work and can now focus on the creative side of their work” [**E3**].

Securing open source software. Almost every software product today relies on open source code at some point in its supply chain. Through its commitment to security as a shared responsibility, Semmle has made a significant contribution to reliably securing such code, benefitting both developers and the consumers of software, and helping to ensure the continued growth and sustainability of open source. Semmle’s technology was made freely available to open source developers through its LGTM.com analysis platform, allowing them to run shared queries against their code or create their own. As of January 2020, the service analysed every commit on over 135,000 projects worked on by a user community of over 700,000 developers [**E7**]. Security research findings contributed by Semmle’s security research team and by its customers were shared through an open Git repository, amplifying the expertise of top security researchers across the world [**E5**.1]. Semmle’s security research team also directly discovered and reported many vulnerabilities in open source projects [**E5**.2]. By September 2019, when Semmle was acquired by GitHub, the team had responsibly disclosed over 100 CVEs (Common Vulnerabilities and Exposures) in high-profile projects like U-Boot, Apache Struts, the Linux Kernel, Memcached, VLC, and Apple’s XNU, allowing project developers to patch vulnerabilities and protect users [**E5**.2]. As GitHub’s SVP of Product noted, Semmle’s unique approach to code analysis “makes Semmle far more effective, finding dramatically more issues and with far fewer false positives”: “just as relational databases make it simple to ask very sophisticated questions about data, Semmle makes it much easier for researchers to identify security vulnerabilities in large code bases quickly… Because QL [Semmle’s analysis engine] is declarative and object-oriented, creating a new analysis is much easier than with traditional code analyzers. Customers frequently find vulnerabilities they couldn’t find with other tools and accomplish tasks that used to take weeks or more in hours” [**E10**.3]. Some of the vulnerabilities discovered using the technology have had drastic data security implications. In 2017 and 2018, for example, Semmle security researchers discovered critical remote code execution (RCE) vulnerabilities in Apache Struts, a widely used framework for Java applications [**E8**]. Equifax’s failure to patch a similar RCE vulnerability in Apache Struts earlier in 2017 led to a serious data breach that had directly cost the company USD440,200,000 by the end of 2018 [**E9**]. According to some expert commentators, the 2018 Semmle-discovered vulnerability was potentially even more concerning than the previous RCE vulnerabilities, including the one exploited in the Equifax breach, because it “operates at a far deeper level within the code” [**E6**.2, at pp. 15–17: synopsys article on CVE-2018-11776, August 2018].

The success of its open, community-driven model allowed Semmle to double its customer base and to increase open source usage tenfold in the year 2018/19, resulting in its acquisition in September 2019 by GitHub, the leading global platform for open source resources and collaboration. GitHub announced their excitement in “bring[ing] the world’s most powerful semantic code engine to the world’s largest developer community” [**E10**.1], describing the move as “a big step in securing the open source supply chain” [**E10**.2]. GitHub’s CEO called Semmle’s engine “revolutionary”, and the SVP of Product stated that “no other code analysis tool has a similar success rate” in finding vulnerabilities [**E10**.2 & 3]. The CEO stated: “Semmle’s community-driven approach to identifying and preventing security vulnerabilities is the very best way forward… As a community of developers, maintainers, and researchers, we can all work together toward more secure software for everyone” [**E10**.2]. Following the acquisition, the Semmle security research team formed the core of a new GitHub Security Lab, which was launched in November 2019 with the mission to find vulnerabilities in open source projects and to make tools available “to make it easier for others to find those vulnerabilities within their own codebases” [**E11**.1 & 2]. Semmle’s “industry-leading semantic code analysis engine”, CodeQL, was accordingly released as a GitHub product in November 2019 [**E11**.2, **E12**.1]. It is free for research and open source, and is available to all public repositories and enterprise customers via GitHub’s continuous integration and deployment service [**E12**.2]. As of July 2020 GitHub reported having over

50,000,000 developer accounts, over 2,900,000 enterprise accounts, and over 100,000,000 repositories worldwide [**E13**].

5. Sources to corroborate the impact

[**E1**] US patents (2008–12). US20090240649 A1: http://tiny.cc/e8hukz; US20090177640A1: http://tiny.cc/faiukz; US20090234801A1: http://tiny.cc/ldiukz; US20120016912A1: http://tiny.cc/80jukz; US20130055205A1: http://tiny.cc/yyjukz; US20130232160A1: http://tiny.cc/xsposz.
[**E2**] Corroborator 1: Semmle company information will be corroborated by the former CEO of Semmle.
[**E3**] Approved public customer case studies (https://semmle.com/case-studies). Microsoft: http://tiny.cc/my79iz; Nasdaq: http://tiny.cc/x079iz; BlackLine: http://tiny.cc/zx79iz.
[**E4**] Microsoft Security Response Center blog posts on their use of Semmle technology: (1) http://archive.ph/QzDZ2; (2) http://archive.ph/8rmZs; (3) http://archive.ph/3mxJC.
[**E5**] Semmle security research: https://semmle.com/security. *Note that Semmle Security Research has been absorbed by GitHub, and links now resolve to GitHub Security Lab pages.*
(1) Semmle open query library (a large library of re-usable queries covering known vulnerabilities, contributed by Semmle and its customers and partners): https://github.com/Semmle/ql.
(2) 100+ CVEs found and reported by Semmle in open source projects (page harvested on 4 March 2020): https://lgtm.com/security/#disclosures (https://archive.vn/SETK7). *Note that this list has migrated from Semmle to the GitHub Security Lab domain;* [**E7**.2 & 3] *confirm the provenance in Semmle research. Each CVE links to its National Vulnerability Database entry, with details of advisories and patches.*
(3) CVE proof of concept videos and technical deep dives: https://semmle.com/security.
[**E6**] Web media coverage of selected Semmle-discovered CVEs (2017–2020): (1) Apple OS XNU kernel; (2) Apache Struts; (3) Facebook Fizz; (4) Google Chrome; (5) U-Boot loader (commonly used with Linux kernel); (6) VLC.
[**E7**] LGTM.com (http://archive.ph/IstFg): Semmle code analysis platform, free for open source projects, showing usage and project stats (as of January 2020, over 39,000,000 commits by more than 700,000 developers analysed for 135,821 open source projects).
[**E8**] RCE vulnerabilities in Apache Struts: CVE-2017-9805 and CVE-2018-11776.
[**E9**] Equifax 4th quarter results, 2017: https://archive.vn/Er5sx; 2018: https://archive.vn/l0FNv.
[**E10**] GitHub communications regarding acquisition of Semmle: (1) Tweet from GitHub account (http://archive.ph/RowTx), and blog posts by (2) GitHub CEO (http://archive.ph/IPIRK), and (3) GitHub SVP of Product (http://archive.ph/ujuKV).
[**E11**] GitHub Security Lab: (1) team: https://securitylab.github.com/; (2) launch blog post, November 2019 (https://archive.vn/F3tFT).
[**E12**] GitHub CodeQL (1) product (https://securitylab.github.com/tools/codeql), and (2) documentation (https://archive.vn/GWPvT).
[**E13**] GitHub data (page archived 26 July 2020): https://archive.vn/NFyVN.
[**E14**] (1) PitchBook estimate of Semmle sale price: http://archive.ph/kRecU. (2) Owler estimate of Semmle annual revenue 2019: http://archive.ph/pkjse. (3) TechCrunch article reporting Series B (and Series A) investment rounds: https://archive.vn/uM0hf