ثورة تصميم البرمجيات

جمال مراد قيس

2025-11-16

في خضمّ الحقبة التي يقود فيها الذكاء الاصطناعي ثورة في تصميم البرمجيات، خرج فريق بحثي من MIT CSAIL بنموذج معماري رائد تحت اسم "البرمجيات الواضحة المُجزّأة" (Legible, Modular Software)، يضع معيارًا جديدًا لبناء الأنظمة الرقمية بحيث تكون مقروءة للبشر وقابلة للتوليد بثقة عبر نماذج اللغة الكبيرة (LLMs).

لطالما كانت البرمجيات الحديثة مصدرًا للتحفّظ: شفرة ضبابية، تراكم الديون التقنية، وتعديلات تحمل في طيّاتها احتمالًا لظهور أخطاء غير متوقّعة أو تغييرات تبعث في النظام خللاً واسع النطاق. تعرّف الباحثان Daniel Jackson و Eagon Meng هذا التحدي بـ «تجزئة الميزة» (feature fragmentation)، وهو الشرخ الذي يجعل فهم أو تعديل وظيفة بسيطة مثل "زر المشاركة" مهمة معقّدة تفوق المناقشة التقليدية.

في المقابل، يقدم النموذج المُقترَح تقسيمًا صريحًا وواضحًا للنظام إلى مكونات (Concepts) و تزامُنات (Synchronizations). فمكون "Concept" يمثل وحدة وظيفية مستقلة كأن تكون ميزة "الإعجاب" أو "المشاركة" أو "المتابعة" مع كل البيانات والحالة والإجراءات الخاصة بها. أما "Synchronizations" فهي لغة نطاق خاص (DSL) تحدّد طريقة تفاعل هذه المكونات بوضوح: القاعدة تقول ماذا يحدث عندما يفعل أحد المكونات شيئًا، وكيف يتزامن الآخر. بهذا الأسلوب، يتحوّل النظام من شبكة مبهمة إلى عقد وبينها رابط صريح يُمكن للبشر قراءته ولوّنقه الآلي فهمه.

يشير الباحثون إلى أن أهمية هذا النموذج لا تقتصر على الجانب التقني، بل تمتد إلى بُعدٍ اجتماعي-علمي أوسع، يتمثل في إعادة الثقة بين الإنسان والآلة. فالبرمجيات التي تُولّدها نماذج اللغة الكبيرة عادة ما تُعامل كـ "صندوق أسود" يصعب تفسير قراراته أو تعديلها دون خطر. لكن عندما تُبنى البرامج على منطقٍ واضح ومجزّأ، يصبح بإمكان الفرق متعددة التخصصات، من مهندسين

ومحللين ومختصي أخلاقيات – المشاركة في فهم وتطوير الأنظمة بقدرٍ أكبر من الشفافية والمسؤولية.

width= كما يفتح هذا المفهوم الباب أمام جيل جديد من أدوات التحقق والاختبار البرمجي. فحين تكون المكونات البرمجية محددة وظيفيًا ومتصلة بعلاقات صريحة، يمكن تصميم أدوات تعتمد على الذكاء الاصطناعي لفحص سلوك كل وحدة والتأكد من التزامها بالقواعد العامة للنظام. وهذا يَعِد بتحوّلٍ جذري في طريقة بناء البرمجيات عالية الحساسية، مثل أنظمة القيادة الذاتية أو برمجيات تحليل الجينوم، حيث الخطأ لا يُقاس بالزمن فقط، بل بالأثر الإنساني.

يكمن الأثر الحقيقي لهذا النموذج في عصر نماذج اللغة الكبيرة. فبينما تبرع هذه النماذج في توليد وظائف منفردة، إلا أنها تعثّرت عندما تُطلب منها الربط بين وحدات متعددة أو تعديل بنية موجودة دون أن تُخفي جوانب التفاعل بين الأقسام. هنا يأتي مفهوم "Legible, Modular Software" ليقدّم بيئة عمل أكثر طبيعية للذكاء الاصطناعي: حَدود واضحة، تصاريح معلَنة، وتفاعلات قابلة للفحص والتحليل.

مثالٌ عمليّ في الدراسة: تقسيم ميزات مثل "الإعجاب" و"التعليق" و"المشاركة" إلى مكونات مستقلة، مع تحديد قواعد التزامن الصريحة بينها بدل أن تتوزّع الوظيفة عبر خدمات متعددة بغير تنظيم واضح. النتيجة: شفافية أعلى، قابلية اختبار أفضل، وإمكانية توليد آلي موثوق للمكونات والتفاعلات.

يحمل هذا النموذج وعدًا بتحوّل ثقافي في بناء البرمجيات. لم يعد الأمر يعتمد على دمج الشيفرات وتوصيل الخدمات جزافًا، بل على اختيار مكونات جهّزت مسبقًا وكتابة قواعد تفاعلها بصيغ مفهومة. في هذا العصر الذي تشكّل فيه الأنظمة المعقّدة أساس البنى التحتية الرقمية، تصبح قِيمتان أساسيتان: الوضوح والتحفّظ الآمن.

يجدر القول إن نموذج "Legible, Modular Software" ليس عرضًا أكاديميًا فحسب، بل دعوة إلى إعادة التفكير في هندسة البرمجيات في زمن الذكاء الاصطناعي. عندما تصبح الشيفرات قابلة للقراءة، وعلاقاتها معلَنة، يمكن للذكاء البشري والآلي أن يتعاونَا بأمان وكفاءة. إنها خطوة نحو برمجيات تُكتب بلغة البشر وتنفّذ بحزم الذكاء الاصطناعي، وهذا وحده قد يشكّل مستقبل التطوير البرمجي.

المصادر

,MIT CSAIL: “MIT researchers propose a new model for legible -1
TechXplore: “Researchers propose a new model -2 ”modular software
”for legible, modular software

تواصل مع الكاتب: mohamedmouradgamal@gmail.com

/https://arsco.org/articles/article-detail-47751