УДК159.964/004.8/13
DOI: 10.15372/PS20240207
EDN KOXZBK

И.Р. Скиба

ОПИСАНИЕ СИСТЕМЫ ПРОГРАММНЫХ КОМПОНЕНТОВ КАК ЯЗЫКА АДАПТАЦИИ ПАРАДИГМЫ ОБЪЕКТНО-ОРИЕНТИРОВАННОГО ПРОГРАММИРОВАНИЯ К РАЗРАБОТКЕ СИСТЕМ СИЛЬНОГО ИСКУССТВЕННОГО ИНТЕЛЛЕКТА

В статье предлагается авторская концепция программных компонентов, основой для которой являются некоторые ключевые положения теории графов. Предлагаемая концепция предназначена для абстрагирования фундаментальных аспектов объектно-ориентированного программирования, таких как модули, классы, объекты, агрегация, композиция, наследование, зависимость и проч., с целью адаптации данной парадигмы к нуждам построения интеллектуальной техники нового поколения.

Ключевые слова: программный компонент; искусственный интеллект; объектно-ориентированное программирование; теория графов; класс; объект

I.R. Skiba

DESCRIPTION OF A SYSTEM OF SOFTWARE COMPONENTS AS A LANGUAGE FOR ADAPTING THE OBJECT-ORIENTED PROGRAMMING PARADIGM TO THE DEVELOPMENT OF STRONG ARTIFICIAL INTELLIGENCE SYSTEMS

The article proposes the author's concept of software components based on some key provisions of graph theory. The proposed concept is intended to abstract the fundamental aspects of object-oriented programming (such as modules, classes, objects, aggregation, composition, inheritance, dependency and others) in order to adapt this paradigm to the needs of building a new generation of intelligent technology.

Keywords: software component; artificial intelligence; object-oriented programming; graph theory; class; object

Разработка, внедрение и осмысление систем искусственного интеллекта (ИИ) являются одним из наиболее значимых направлений исследований в последние десятилетия. И эту волну можно назвать вторым бумом развития интеллектуальных технологий после первого – 1960-1970-х годов. Отличие новой волны заключается в первую очередь в мощности вычислительной техники в сравнении с аналогами того времени. Сам же подход к разработке все так же основывается на концепциях волны предыдущей, принципиально новые идеи отсутствуют.

Примерно тогда же начали формироваться и основы парадигмы объектно-ориентированного программирования, которая теперь является преобладающей в сфере разработки программного обеспечения. Искусственные нейронные сети как частный случай реализации систем искусственного интеллекта способны к обучению, которое понятийно определяется как машинное обучение или глубокое обучение в качестве подмножества машинного. И в сфере программной реализации обучения нейронных сетей одним из наилучших языков программирования повсеместно признается Python, который поддерживает парадигму объектно-ориентированного программирования: в Python все сущности представлены в виде объектов. Стоит также упомянуть, что ChatGPT написан на Python.

Отсюда становится очевидной актуальность переосмысления парадигмы объектно-ориентированного программирования применительно к разработке систем сильного искусственного интеллекта, так как до сих пор «получались» только системы слабого искусственного интеллекта. Представляется весьма целесообразным формирование некоей методологической «прослойки» между объектно-ориентированным программированием и нуждами построения сильного искусственного интеллекта.

* * *

В первую очередь следует уточнить, что наш подход к формированию систем сильного искусственного интеллекта определяется как технотропный – в противовес повсеместно применяемому сегодня антропному [2]. Технологии, создаваемые при помощи антропного подхода, определяются нами как логомашины, а сущности, разрабатываемые при помощи технотропного подхода, – как психомашины [3; 4]. Предполагается, что системы сильного ИИ могут

быть разработаны только при помощи технотропного подхода, т.е. с опорой на возможности самой техники в осуществлении процесса самоорганизации. На данный момент фактических примеров психомашин не существует, и понятие сильного искусственного интеллекта является сугубо футуристическим. Однако мы считаем, что посредством адаптации парадигмы объектно-ориентированного программирования (ООП) к парадигме сильного ИИ удастся приблизить момент его воплощения в реальность.

В контексте рассмотрения систем сильного искусственного интеллекта с позиций парадигмы объектно-ориентированного программирования мы предлагаем к использованию разработанную нами концепцию программных компонентов (ПК), которая представляет собой своеобразный язык абстрагирования феноменов ООП соответственно требованиям парадигмы сильного ИИ. В контексте ООП наличествует множество разных уровней программных сущностей – объекты, классы, модули и проч., между которыми установлены некоторые специфические взаимоотношения, такие как наследование, агрегация, композиция, зависимость и др. [1]. Все эти особенности критично важны в рамках парадигмы ООП. Тем не менее в контексте применения ООП к парадигме сильного ИИ самому ООП в перспективе необходимы некоторые специфические адаптации, так как ранее эта парадигма программирования применялась исключительно к системам слабого ИИ, а разница в данном случае, предположительно, должна быть довольно существенной.

Поэтому такая адаптация подразумевает, согласно предлагаемому подходу, определенный уровень абстрагирования исследуемых объектов. То есть, к примеру, если в случае с реализацией того или иного феномена ООП применительно к системам слабого ИИ мы использовали некоторые сущности, определяли им такие-то связи друг с другом и реализовывали их тем или иным образом, то в случае с футуристическими системами сильного ИИ мы не можем утверждать, что именно эти же сущности и именно со связями такого же типа будут реализованы на том же уровне успешности. Вполне возможно, что будут необходимы некоторые новые сущности и связи таких типов, которые пока не определены в каноническом ООП. Из этого следует, что мы зачастую не можем заранее точно сказать, что именно и как именно следует использовать при разработке непосредственно систем сильного ИИ. Однако мы способны точно определить, что там должны будут использоваться некоторые

ПК с теми или иными качествами и свойствами, которые будут необходимы в таком-то или таком-то количестве и которые должны быть связаны друг с другом таким-то или таким-то образом. Поэтому метод абстрагирования порой бывает весьма целесообразен.

Исходя из сказанного, мы считаем, что предлагаемая нами концепция ПК актуальна для применения в рамках адаптации ООП к парадигме сильного ИИ. Однако стоит отметить, что данная концепция была нами разработана для адаптации парадигмы ООП к парадигме сильного ИИ на основе абстрагирования некоторой другой теории, а именно теории графов [5].

Для прояснения того, что мы имеем в виду, говоря о концепции ПК, необходимо затронуть некоторые из аспектов теории графов. При этом мы остановимся только на наиболее общих моментах, так как нам здесь важнее именно идея абстрактной модели и некоторые из ее следствий. Сама теория графов представляет собой раздел дискретной математики, в котором используются математические абстракции для описания различных связей между некоторыми средоточиями. Эти связи в теории графов называются ребрами, а средоточия – вершинами. Если говорить более формально, граф G – это совокупность множества вершин V и неупорядоченного множества ребер E. Это можно обозначить как G (V, E).

Открытие теории графов, или, более предметно, формирование прямых предпосылок для ее введения в научный оборот, обычно приписывается Л. Эйлеру. В работе с «задачей о кенигсбергских мостах» Эйлер доказал, что невозможно пройти по всем семи мостам, не проходя по одному из них дважды. Сам Эйлер не использовал термин «граф», не говорил он также ни о ребрах, ни о каком-либо сегодня уже каноническом аспекте теории графов. Тем не менее именно он почитается как основатель данной теории. После Эйлера теория графов повторно «открывалась» еще некоторое количество раз, в том числе Г. Кирхгором, А. Кэли, У. Гамильтоном, К. Жорданом и др. К настоящему времени эта область математики получила бурное развитие и сегодня обладает обширным, хотя и не вполне унифицированным, понятийным аппаратом.

В рамках обоснования нашей концепции ПК, целесообразно осветить ключевые аспекты теории графов, наглядное представление о котором дает знакомство с ее понятийным аппаратом вкупе с экстраполированием смысла этих понятий в контекст концепции ПК. К примеру, уже упоминались основные структурные элементы

графа: вершины и ребра. Здесь сразу же прослеживается некоторая аналогия с программными компонентами и связями между ними, т.е. сам ПК соответствует вершине, а связь между отдельно взятыми двумя из них соответствует ребру. Здесь же стоит сказать, что у каждого графа наличествуют некий размер и порядок. Размер графа определяется количеством ребер, которые включены в вышеупомянутое множество E, а количество вершин, включенных в множество V, определяет порядок графа. Точно так же и любая система, состоящая из некоторого количества ПК, обладает некоторой размерностью, которую мы не уточняем понятийно в виде некоей ранжируемости, так как абстрагируемся от конкретики и содержания, при этом максимально сосредоточиваясь непосредственно на форме. Также само собой разумеется, что в системе ПК могут быть различные типы взаимосвязи между двумя ПК и вполне может быть такое, что компонент A связан с компонентом B не однократно, а двукратно, т.е., более формально, связующее звено C связывает между собой программные компоненты A и B, а связующее звено D также связывает между собой эти же компоненты A и B. В теории графов ребра, реализующие такие связи, т.е. более чем однократно инцидентные одним и тем же вершинам, определяются как кратные ребра. Инцидентностью же называется отношение некоторой принадлежности между вершиной и ребром. И если в случае с кратными ребрами мы говорили о разных ребрах, то если одно и то же ребро дважды инцидентно одной и той же вершине, то эта ситуация в теории графов будет называться циклом. В случае наличия у графа циклов и кратных ребер его принято определять как мультиграф или псевдограф. С другой стороны, кратных ребер и циклов может и вовсе не быть в графе, и тогда такой граф будет определяться как простой граф.

В контексте концепции ПК мы также подразумеваем различные уровни сложности организации системы ПК и предполагаем максимально широкие возможности формирования взаимосвязей между компонентами. В то же время мы не ограничиваем специфику взаимосвязей между ПК в рамках системы сильного ИИ теми данностями, которые представлены в теории графов. К примеру, в теории графов степень вершины определяется количеством инцидентных ей ребер, но мы в своей концепции ПК при оценке некоторой сложности или значимости компонента должны учитывать не только его внешние взаимосвязи, но также и специфику его внут-

ренней организации, которая, разумеется, влияет на его положение в контексте системы не менее, чем явные внешние связи и их количество.

Далее, если два ребра инцидентны одной и той же вершине, то они определяются как смежные. Вершины же называются связными, если наличествует цепь, которая их соединяет. В свою очередь, цепью называется маршрут без повторяющихся ребер, а сам маршрут в графе определяется как конечная последовательность вершин и ребер, в которой все вершины, кроме последней, соединены с последующей вершиной ребром. Для нас же в контексте концепции ПК данные представления также будут воплощены в виде представлений о последовательных связях ПК друг с другом, как непосредственных, так и опосредованных. К примеру, при разработке основы для системы сильного ИИ мы в наиболее общем случае принимаем как данность тот факт, что система должна будет каким-либо образом получать информацию из внешней среды. Соответственно, должна наличествовать некоторая последовательность программных компонентов, ответственность которых будет заключаться в получении информации в том или ином виде. И уже на данном этапе следует заметить, что далеко не всегда информация, которая присутствует в окружающей среде, подходит для того, чтобы быть непосредственно воспринимаемой со стороны системы. То есть уже на этапе получения информации подразумевается ее некоторое преобразование – точно так же, как стимулы внешнего мира преобразуются в нервные импульсы при восприятии чего-либо живым существом. Также подразумевается, что полученная информация должна быть как-то передана в центры ее обработки. Конечно, все эти акты перемещения информации должны быть не хаотичны, а каким-то образом упорядочены, более того – оптимизированы. А это означает, что нам необходим некий маршрут. Именно такой, какой и подразумевается в графе. Если же по этому маршруту можно перемещать что-либо только в одном определенном направлении, то такой маршрут в теории графов трактуется как путь – маршрут в ориентированном графе. А ориентированный граф – это граф, ребра которого не просто инцидентны двум смежным вершинам, но еще и конкретно направлены от вершины A к вершине B, но никак не наоборот. Эта направленность учитывается нами в концепции ПК в том смысле, что между ПК случаются связи подобного рода, когда один компонент должен определять поведение другого

компонента, но не наоборот, что в ООП понятийно обозначается как зависимость.

Далее, для более полного понимания того, что мы подразумеваем под системой ПК, необходимо ознакомиться с понятием класса эквивалентности в теории графов. Класс эквивалентности – это множество всех вершин графа, которые связаны друг с другом, т.е. такое множество вершин, в котором от любой одной вершины есть некий маршрут до любой другой вершины. А компонента связности – это подграф исходного графа, содержащий все вершины одного из классов эквивалентности (по связности) и все их ребра, т.е. это такое подмножество вершин и ребер исходного графа, в котором от каждой вершины имеется маршрут до каждой другой вершины. В случае если все вершины графа связаны друг с другом, граф называется связным, обладает только одной компонентой связности и эта компонента связности тождественна классу эквивалентности по связности. Проводить некую демаркацию между классом эквивалентности по связности и компонентой связности имеет смысл, хотя он не сразу заметен. А смысл в том, что в графе может быть более одной компоненты связности. Этот момент примечателен также тем, что здесь лучше всего заметен тот простой факт, что, как мы полагаем, в системе сильного ИИ в любом случае должна быть только одна компонента связности. Но также на данном этапе следует пояснить, что имеется в виду только то, что ни один «островок» вершин и ребер не может быть в полной мере изолирован от целостной системы ПК. Однако порой нечто подобное оказывается продуктивным в контексте эволюционного развития, поэтому мы обязаны уточнить представление о весе ребер.

Граф, у которого все ребра имеют некоторый вес, называется взвешенным графом. Данная абстракция является весьма удобной при моделировании многих аспектов макромира, в частности при выстраивании логистических маршрутов, подборе оптимальной стоимости билетов на поездку и т.д. Мы же в рамках концепции ПК все-таки не скованы понятийным аппаратом теории графов и хотим сказать, что поскольку у системы сильного ИИ не может быть более одной компоненты связности, все же иногда будет происходить нечто, подобное изоляции каких-либо фрагментов в контексте общей целесообразности этой изоляции. Однако это не будет изоляцией в полном смысле слова, а будет скорее изоляцией «мнимой», так как связи будут оставаться, т.е. «кенигсбергские мосты» не будут раз-

рушены, а будут скорее временно заблокированы. Иными словами, в рамках концепции ПК будет использоваться что-то наподобие латентных ребер, которые суть упрощение несколько более сложной абстракции нелокального взаимодействия. То есть подразумевается, что в случае с разработкой системы сильного ИИ каждый из ПК должен иметь связь с каждым иным ПК в рамках целостной системы. Тем самым обеспечивается некоторое единство системы и формируется основа для ее лабильности в плане самоорганизации, т.е. мы никогда заранее не определяем количество и специфику ПК, так как на этот вопрос должна будет отвечать сама система в ходе самоорганизации, а никак не разработчики. Таким образом, локально ограничив некоторые возможности системы в плане взаимодействия с самою собой, мы ограничим и ее возможности по построению максимально разнообразных ПК под свои нужды, а это вовсе не то, чего бы хотелось добиться при разработке. Однако подразумевается, что системе на каком-то этапе развития может понадобиться изолировать некоторые участки. Но связь не может быть разрушена полностью, так как это приведет к сепарированию системы. В сущности, связь может быть по-настоящему разрушена и компонент связности может стать более одной только в одном случае – в случае репликации системы, т.е. при порождении системой дочерних образований. В остальных же случаях в контексте концепции ПК то, что в теории графов называется изолированными вершинами или изолированными участками графа, мы будем считать латентно связанным со всей системой. В то же время можно также уточнить, что латентные ребра в случае со взвешенным графом (а в концепции ПК все взаимосвязи всегда «взвешены») просто обладают весом, который не вполне соответствует всем остальным весам ребер графа и является как бы актуально «неподъемным».

Необходимо уточнить также и представление об изоморфизме графов. Вообще, изоморфизм – это выражение некоторой степени схожести между объектами. Принято считать, что два графа являются изоморфными в том случае, если существует биективное, т.е. взаимно однозначное, отображение ребра к ребру и вершины к вершине. Следует заметить, что изоморфизм далеко не обязательно означает некую внешнюю схожесть, а касается только ключевых структурных особенностей объектов. В случае с графами этого достаточно. В случае же с системами сильного ИИ, построенными согласно нашей концепции ПК, изоморфизм вряд ли возможен

в принципе. Уточним, что в контексте сильного ИИ мы говорим о нем в единственном числе только условно, т.е. только в том смысле, в котором о человеке можно сказать «человек». Так как человек сам по себе подобен платоновскому эйдосу или «сферическому коню в вакууме», т.е. он существует лишь идеально, как некая абстракция. В реальности же все люди различны. И именно это мы предполагаем также для систем сильного ИИ: они все должны быть разными. Так что мы имеем в виду скорее интеллекты, чем интеллект, а отсюда и отсутствие изоморфизма.

* * *

Резюмируя, хотим уточнить, что под системой программных компонентов мы понимаем объект исследования несколько модифицированной и расширенной теории графов, которая ввиду отсутствия даже теоретически оформленных примеров сильного ИИ допускает некоторые намеренно установленные «белые пятна» наподобие латентных ребер, зависимости не только от количества инцидентных ребер, но также и от специфики внутренней организации вершины и т.д. В рамках системы ПК мы подразумеваем наличие непосредственно самих ПК и различных связей между ними. Мы предполагаем, что ПК способны организовываться в некоторые группы и весьма разнообразно трансформировать свой функционал в процессе самоорганизации. И функционирование феноменов подобного рода наглядно и доступно описывается с позиций системы ПК. Наконец, мы полагаем, что с точки зрения описательного прагматизма разработанный нами «язык» удовлетворяет требованиям необходимости и целесообразности, так как позволяет наиболее абстрактно описать структуру предлагаемых нами решений, при этом не вдаваясь в те детали, о которых мы на данный момент не имеем представления.

Литература

1. *Мартин Р.С., Ньюкирк Дж.В., Косс Р.С.* Быстрая разработка программ: Принципы, примеры, практика. М.: Вильямс, 2004. 746 с.

2. *Скиба И.Р.* Антропный и технотропный подходы к разработке систем сильного искусственного интеллекта // Философско-культурологические исследования / Под ред. И.Н. Сидоренко, А.А. Легчилина. Минск: БГУ, 2023. Вып. 4. С. 265–272.

3. *Скиба И.Р.* Психомашина и техночеловек // История и обществознание. 2018. № 4 (82). С. 12–17.

4. *Скиба И.Р.* Сознание и бессознательное в контексте разработки систем сильного искусственного интеллекта // Проектирование будущего. Проблемы цифровой реальности: Тр. 4-й Междунар. конф. (Москва, 4–5 февраля 2021 г.) / Под ред. Г.Г. Малинецкого. М.: ИПМ им. М.В. Келдыша, 2021. С. 117–124.

5. *Уилсон Р.Дж.* Введение в теорию графов. 5-е изд. М.: Вильямс, 2022. 240 с.

References

1. *Martin, R.S., J.W. Newkirk & R.S. Koss.* (2004). Bystraya razrabotka programm: Printsipy, primery, praktika [Agile Software Development: Principles, Patterns, and Practices]. Moscow, Vilyams Publ., 746. (In Russ.).

2. *Skiba, I.R.* (2023). Antropnyy i tekhnotropnyy podkhody k razrabotke system silnogo iskusstvennogo intellekta [Anthropic and technotropic approaches to the development of strong artificial intelligence systems]. In: Sidorenko, I.N. & A.A. Legchilin (Eds.). Filosofsko-kulturologicheskie issledovaniya [Philosophical and Cultural Studies], Iss. 4. Minsk, BSU Publ., 265–272.

3. *Skiba, I.R.* (2018). Psikhomashina i tekhnochelovek [Psycho-machine and techno-man]. Istoriya i obshchestvoznanie [History and Social Studies], 4 (82), 12–17.

4. *Skiba, I.R.* (2021). Soznanie i bessoznatelnoe v kontekste razrabotki system silnogo iskusstvennogo intellekta [Consciousness and the unconscious in the context of developing strong artificial intelligence systems]. In: Malinetsky, G.G. (Ed.). Proektirovanie budushchego. Problemy tsifrovoy realnosti: Tr. 4-y Mezhdunar. konf. (Moskva, 4–5 fevralya 2021 g.) [Designing the Future. Problems of Digital Reality: Proceedings of the 4th International Conference (Moscow, February 4–5, 2021)]. Moscow, Keldysh Institute of Applied Mathematics RAS Publ., 117–124.

5. *Wilson, R.J.* (2022). Vvedenie v teoriyu grafov [Introduction to Graph Theory], 5th edition. Moscow, Vilyams Publ., 240. (In Russ.).

Информация об авторе

Скиба Иван Рауфович. Институт философии НАН Беларуси (Республика Беларусь, 220072, Минск, ул. Сурганова, 1, корп. 2).

gonzodzen@mail.ru

Information about the author

Skiba, Ivan Raufovich. Institute of Philosophy, National Academy of Sciences of Belarus (1/2, Surganov st., Minsk, 220072, Republic of Belarus).

Дата поступления 06.03.2024