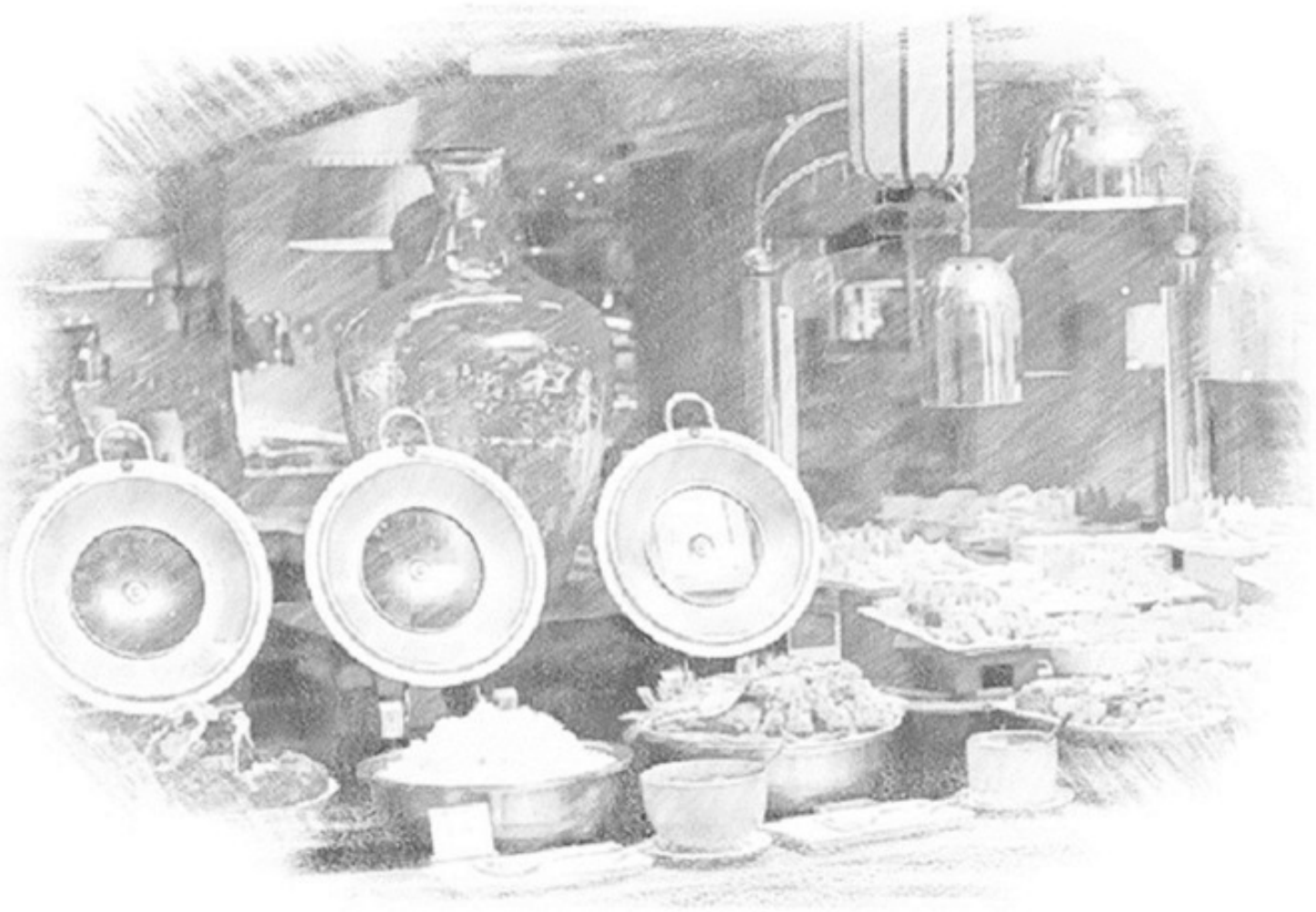
Анатолий Левенчук



СИСТЕМНАЯ ИНЖЕНЕРИЯ 2022

Учебник

Анатолий Левенчук

Системная инженерия – 2022

«Издательские решения»

Левенчук А.

Системная инженерия – 2022 / А. Левенчук — «Издательские решения»,

ISBN 978-5-00-590127-9

Учебник «Системная инженерия — 2022» продолжает учебники «Практическое системное мышление — 2022» и «Методология 2022» (обязательные пререквизиты), рассказывая о практиках современной системной (основанной на системном мышлении) безмасштабной (для систем самого разного вида) и непрерывной (эксплуатируемая система постоянно развивается) инженерии. Учебник базируется на совмещении изложения лучших практик программной инженерии, классической «железной» системной инженерии и инженерии предприятия.

ISBN 978-5-00-590127-9

Содержание

Системная инженерия – 2022

Анатолий Левенчук

ISBN 978-5-0059-0127-9
Создано в интеллектуальной издательской системе Ridero

Книга/курс системной инженерии учит фундаментальной практике системной инженерии, как способу изменения мира к лучшему. Поначалу системная инженерия развивалась главным образом в сложнейших аэрокосмических проектах, в инженерии транспорта, отличаясь от традиционных видов инженерии тем, что опиралась на системный подход, позволяющий увязать в одном инженерном проекте мышление об объектах самого разного масштаба – от маленького винтика в самолёте до огромного аэропорта, поддерживающего эксплуатацию множества самолётов. Сегодня практически все инженерные разработки в мире для систем самого разного масштаба (от отдельных веществ в химическом синтезе или деталей из какого-то однородного материала до обществ или даже человечества в целом, что позволяет говорить о «безмасштабности») делаются с использованием подходов системной инженерии: в них выделяются разработка концепции использования, концепции системы, архитектуры системы, управление версионированием и конфигурацией, делаются инженерные обоснования. Кроме безмасштабности современная системная инженерия опирается на третье поколение системного мышления, которое выходит за рамки одного жизненного цикла на идеи эволюции/развития, и поэтому в учебнике описана непрерывная инженерия: практика непрерывной разработки, непрерывного принятия архитектурных решений, непрерывного ввода в эксплуатацию системы, которая развивается в непрерывно меняющемся окружении, не просто однократно разрабатывается, эксплуатируется и затем забрасывается, а постоянно меняется в ходе своей эксплуатации, получает частые обновления, инженеры-разработчики, архитекторы и изготовители системы не прекращают свою работу с началом эксплуатации.

Книга/курс даст представление о том, чем занимаются инженеры в своей работе, даже если они не называются «инженеры» (например, менеджеры – это инженеры предприятия, врачи – инженеры человеческого тела, агрономы – инженеры урожая).

Книга/курс уникален, ибо на данный момент большинство современных учебников «железной» системной инженерии хорошо доносят идею безмасштабности как применимости одних и тех же методов разработки, основанных на применении системного подхода в инженерии для самых разных систем самых разных масштабов, в том числе традиционно не понимаемых как инженерные. Но они основаны на идее однократной сдачи системы в эксплуатацию, однократном прохождении жизненного цикла системы с упором на разовое введение в строй какой-то большой системы (например, атомной станции). Современные учебники программной инженерии, наоборот, хорошо передают непрерывный характер разработки, идею непрерывного уточнения изменяющегося окружения уже работающей системы, уточнения потребностей потребителей/пользователей/заказчиков и непрерывный ввод в эксплуатацию всё новых и новых возможностей для уже работающей системы. Но они абсолютно провальны в показе общепременимости идей инженерной разработки к самым разным видам систем, ограничиваются только программными системами. Наш курс/учебник даёт гармонизированное представление как о безмасштабности системной инженерии, так и о её непрерывности в ходе инженерных проектов. Конечно, через некоторое время появится много учебной литературы, которая закроет этот разрыв, но пока «лучшее из двух миров» собрано именно в нашем курсе/книге.

В курсе даётся основное представление об инженерных практиках на самом высоком уровне абстракции, удобном для соотнесения этих практик со многими видами целевых систем, начиная от традиционных для инженерии прошлого простых механических систем через сегодняшние киберфизические системы к живым организмам, личности, сложным социотехническим системам (предприятиям) и далее социальным системам. Излагается не конкретный метод системной инженерии, принятый, например, в NASA или MITRE, или полностью соответствующий учебнику системной инженерии INCOSE, по которому ведётся сертификация профессионалов в системной инженерии. Излагается не метод, который полностью соответствует духу и букве какого-то конкретного одного стандарта практик системной инженерии, например ISO 15288:2015. Нужно помнить, что все эти международные стандарты и публичные документы привязаны к работе в сложнейших аэрокосмических проектах (системный уровень киберфизической системы) и подразумевают обучение на многочисленных специализированных курсах в порядке магистерской подготовки профессиональных системных инженеров киберфизических систем.

Материалы курса могут оказаться неожиданными для инженеров, получивших классическое инженерное образование. Так, в курсе отражается изменение в разделении труда между прикладными разработчиками и архитекторами, появление DevOps (с многочисленными их вариациями в самых разных видах инженерии), и исчезновение по факту инженерии требований (что вызывает максимальное отторжение у привыкших к работе с требованиями инженеров, но жизнь изменилась, и мы не можем излагать в нашем новом учебнике то, чему учили в вузах нас самих, не можем излагать опыт старых лучших инженерных проектов. Мы консультировались с системными инженерами из компаний-единорогов, чьи продукты используются миллионами людей, и они подтвердили: за последние пять лет ситуация с разработкой требований кардинально изменилась. Так что мы просто добавили разъяснения по этому вопросу, но инженерию требований не излагаем, как устаревшую уже практику).

Наша книга/курс даёт только кругозор по системной инженерии и примерно соответствует содержанию относительно коротких курсов «Foundations of Systems Engineering», которые дают введение в специальность системного инженера. Особенность книги/курса в том, что она даёт кругозор по созданию систем самых разных видов, которые не всегда сегодня ассоциируются с инженерной работой: простых механических систем без встроенных компьютеров (уровень «вещества»), киберфизических систем (со встроенным компьютером), живых существ (от бактерий и растений через животных до человека, взятого как биологическое существо, включая генную инженерию и медицину как инженерные практики), личностей (психология и образование как инженерные практики), организаций (системный менеджмент как инженерия предприятия), сообществ (строительство сообществ, маркетинговые кампании для инженерии клиентуры), обществ (госстроительство, политика как социальная инженерия), человечества в целом.

Тем самым пройти наш курс предлагается не только «классическим» системным инженерам киберфизических систем, которые планируют далее пройти ещё много других прикладных курсов, чтобы профессионально заниматься инженерной работой по созданию гаджетов, ракет и термоядерных реакторов, но и менеджерам, интересующимся тем, что делают в проекте инженеры (и менеджеры обнаружат, что они заняты по сути дела системной инженерией непрерывно изменяющихся организационных систем) и предпринимателям-основателям предприятий (они, оказывается, тоже занимаются инженерией предприятий, но в предметах их интереса концепция использования целевой системы этого предприятия), а также представителям всех остальных профессий, которые занимаются изменениями мира: от агрономов и маркетологов до разработчиков софта и политиков.

Лучшие инженерные практики появляются сегодня сначала в программной инженерии корпоративных информационных систем, потом заимствуются с некоторым лагом (5—

10 лет) оттуда в классические проекты аэрокосмической и транспортной системной инженерии, а затем оказываются общеприменяемыми в самых разных проектах: непрерывное уточнение потребности в системе и разработка концепции использования с определением границ целевой системы и функций системы в её рабочем окружении в рамках стратегирования, непрерывная разработка самой системы прикладными инженерами, разработка системной архитектуры, обязательность инженерных обоснований, в том числе испытаний и частей системы, и системы в целом, непрерывная интеграция результатов работы самых разных разработчиков, и так далее.

Все эти практики звучат уже абсолютно привычно для программных инженеров, успешно осваиваются работающими классическими «железными» инженерами, но для будущих инженеров, предпринимателей, менеджеров и работников самых разных других специальностей нужно объяснить эти практики, дать их примеры, дать попробовать их применить в рабочем проекте. Первому знакомству с этими практиками безмасштабной непрерывной инженерии и посвящён наш курс/книга.

Нашего курса будет недостаточно для профессиональной системноинженерной работы в проектах (для этого нужно будет пройти прикладные курсы инженерии конкретного вида систем какого-то определённого системного уровня, и уж как минимум изучить полтора десятка книг, которые рекомендуются в курсе и адаптировать тамошние практики к условиям конкретного вашего рабочего проекта), но достаточно для того, чтобы иметь фундаментальные знания в части SoTA системной инженерии как универсального метода изменения мира к лучшему.

В прикладной трудовой кругозор входит знакомство с самыми разными прикладными инженерными практиками: кругозор саморазвития (инженерии личности), менеджерский кругозор (инженерии организации/предприятия как системы создания других систем), опционально кругозоры программной инженерии и инженерии киберфизических систем. После кругозорного образования, конечно, нужно получать профессиональное образование: в реальных проектах на одном кругозоре далеко не уедешь. Но даже прикладные кругозоры строятся на основе настоящего фундаментального курса «безмасштабной непрерывной» системной инженерии, они основаны на владении фундаментальным, трансдисциплинарным способом рассуждения о том, как нужно изменять мир.

Хотя наш фундаментальный курс системной инженерии предполагается первым из кругозорных курсов прикладных инженерных практик для отдельных видов систем (перед курсами системного менеджмента, программной инженерии, инженерии киберфизических систем, инженерии мастерства и т.д.), прохождение его возможно только после прохождения курсов по всем нижележащим мыслительным практикам интеллект-стека, в частности, курсов онтологики и коммуникации, системного мышления, методологии (а если эти курсы окажутся трудными, то и подготовительного курса системного саморазвития). Из других пререквизитов, не вошедших в курсы по дисциплинам интеллект-стека, укажем только на необходимость владения английским языком: вся оригинальная современная литература по системной инженерии доступна только на английском, на русском есть только переводы немногих классических (и поэтому уже устаревших) работ великих системных инженеров прошлого, причём нужно учитывать, что эти работы великих системных инженеров прошлого главным образом будут инженерными работами для относительно простых систем уровня вещества и киберфизической системы. Наш курс системной инженерии безмасштабный в части целевых систем и даёт возможность работать и с системами более высоких уровней: живыми существами, людьми, а также организациями, сообществами и обществами людей.

Большинство понятий, с которыми работают системные инженеры, вводится ещё в курсах системного мышления и методологии. Но в курсе практического системного мышления ничего не говорится о практиках, выполняемых создателями, а в курсе методологии об этих

практиках говорится, но их состав уточняется. Так что ещё нужно отдельно уточнить SoTA практики, которые позволяют организации проекта двигать альфы проекта по их состояниям. Надо как-то описать систему в её окружении, затем описать её устройство, попробовать сделать начальную систему, а затем уточнять эти описания и улучшать/развивать систему, выпуская новые и новые её версии, меняя её какими-то инкрементами, реализуя всё новые и новые «фичи» (и тогда окажется, что движутся по состояниям уже не столько альфы всей системы в целом, сколько альфы отдельных инкрементов системы, реализующих эти «фичи» – это тоже нужно описать).

Как именно это всё делается в инженерном проекте (то есть практически любом проекте по изменению мира к лучшему, даже если его не принято называть «инженерным»), какие есть примеры всех этих практик для систем самого разного масштаба? Методам непрерывной разработки нужно учить специально, методам принятия архитектурных решений нужно учить специально. Нужно знать не только то, что «в проекте обязательно нужно иметь концепцию использования», но и как эту концепцию использования разработать для самых разных видов систем, а потом ещё и реализовать, причём не разово на какой-то момент времени, а адаптируя получившуюся систему ко всё новым и новым условиям окружения, используя всё новые и новые идеи для концепции системы, всё новые и новые организационные возможности, появляющиеся у создателей системы.

Изложение опирается на знакомство с фундаментальными мыслительными практиками интеллект-стека (данный курс завершает знакомство с этими практиками).

Материал излагается в форме, уже знакомой студентам курсов практического системного мышления и методологии. В частности, используется аннотация типами мета-мета-модели системной инженерии (типами из нашего курса/учебника) понятий мета-моделей предметных областей (учебники прикладных инженерных практик как «мета-модель из культуры», стандарты предприятия как «локальная/ситуационная мета-модель») для показа особенностей использования абстрактных понятий курса в конкретных ситуациях.

Концептуально все разделы курса согласованы между собой, хотя в изложении явно прослеживается двойственность использования и более старых идей «водопадной» (одноцикловой), и современных идей непрерывной инженерии.

Основной материал курса изложен с использованием не столько стандартов системной и программной инженерии (они вышли до 2015 года, поэтому не отражают произошедшего с тех пор сдвига в непрерывную инженерию, при этом они продолжают использоваться в сильно зарегулированных госорганами больших инженерных проектах создания кибер-физических систем типа атомной электростанции и авианосца, изменений инженерии там можно ожидать очень нескоро), сколько по современной литературе 2017—2022 года. В нашем курсе/книге даны ссылки на почти два десятка учебников системной, программной инженерии и инженерии предприятия, разъясняющих подробно те или иные положения курса. Больше половины этих учебников отсылают к примерам из программной инженерии, но мы надеемся, что при их изучении будут учтены идеи безмасштабности из нашего учебника (а идеи непрерывности в них всех уже есть).

Для примеров в книге/курсе берутся главным образом практики программной инженерии и инженерии киберфизических систем, изредка практики инженерии предприятия. В следующих версиях предполагается увеличить число примеров других систем. Курс не делает акцента на моделеориентированную[1] системную инженерию, ибо сегодня другой системной инженерии уже не бывает, вопрос лишь в выборе уровня формальности моделирования, опоры на самые разные языки моделирования. Эти языки системного моделирования отнюдь не все относятся к визуальным языкам начала 21 века, принятым для больших окологосударствен-

[1] Без различения model-based и model-driven.

ных проектов классической системной инженерии. Материал по моделированию, поднимаемый в рамках курсов-пререквизитов по онтологике и коммуникации, системному мышлению, методологии, и примеры построения различных инженерных моделей даются в нашей книге/курсе в их текущем разнообразии. Но упор делается на то, чтобы использовать для инженерного моделирования подручные low code системы (productivity tools), в том числе системы табличного моделирования (типа coda.io, notion.so и их варианты), а также всевозможные прикладные инженерные моделеры различных прикладных инженерных дисциплин.

Материал книги может быть использован в качестве основы для кругозорного семестрового или даже двухсеместрового курса системной инженерии для технических университетов, материал курса включает в себя также «рабочую тетрадь» со множеством тестовых вопросов, упражнений на моделирование, заданий для тренировки инженерного мышления. Тем самым наша книга/курс может служить основой для самостоятельного изучения в варианте онлайн-курса, основой для группового изучения в study group с инструктором или без, для практикума с преподавателем (семинарские занятия с консультациями по выполнению упражнений, в рамках смешанного/blended обучения с использованием компьютера для лекционных материалов и в качестве тренажёра, но возможностью получить консультацию преподавателя), для отдельного курса с участием преподавателя, в составе других длинных программ (в том числе магистерских программ системной инженерии, где материал курса дополняется другими прикладными курсами). Возможно также прохождение курса с ментором.

История курса/книги начинается с 2012 года, когда автор начал чтение односеместрового курса «Введение в системную инженерию» (Foundations of Systems Engineering) на межфакультетской (впоследствии – межвузовской) кафедре технологического предпринимательства МФТИ. Буквально на следующий год курс был поделен на семестровый курс системного мышления, и отдельно – семестровый курс системной инженерии. Обучение на кафедре по этим курсам затем продолжили выпускники программы (Михаил Бухарин и Илья Бурдин). В Школе системного менеджмента работа над курсом продолжилась, несколько лет там курс «Системная инженерия» на базе учебника системного мышления читал Вячеслав Мизгулин. В 2022 году было окончательно сформировано видение опоры курса системной инженерии на мыслительные практики интеллект-стека и необходимость безмасштабного курса, а также отражение непрерывного характера инженерии, окончательный отход от «водопадной» модели с её одним стадийным «жизненным циклом».

Это новое видение нашего курса/книги отразило опыт многочисленных попыток преподавать какие-то основы системной инженерии в самых разных вузах: поскольку часы в программе на пререквизиты к курсу не были предусмотрены, большинство этих попыток оказались неудачными. Как минимум, для курса оказались нужны знания онтологики, основ системного подхода, методологии, а также безмасштабное изложение материала, чтобы решить вопрос о применимости системной инженерии к проблемам инженерии предприятия и согласовать материал с трендами в INCOSE VISION 2035, которые явно указывают на рост внимания системных инженеров к системам уровня живых существ, личности, организации, сообществ, обществ. В программной инженерии массовые изменения по линии непрерывности разработки прошли с переходом на микросервисные архитектуры, это стало массовым и отразилось в учебной литературе приблизительно с 2017 года (хотя сами идеи обсуждались ещё с момента выхода Agile manifesto в 2003 году). Системное мышление, которое в третьем своём поколении вышло за пределы жизненного цикла и включило в себя идеи эволюции, базирующиеся на общих физических законах, как раз помогает в понимании этого направления развития инженерных практик.

В кооперации между Школой системного менеджмента и Русским отделение INCOSE была создана рабочая группа из инженеров, имеющих и опыт преподавания системной инженерии, и опыт руководства системноинженерными проектами, и опыт применения мыслитель-

ных практик интеллект-стека к системам самых разных уровней. Рабочую группу возглавил К. Гайдамака, научным руководителем стал А. Левенчук. Книга/курс – стал результатом работы этой группы, в которой прошло множество плодотворных обсуждений. Текст в итоге написан А. Левенчуком. Спасибо Роману Варьянко за корректуру и ценные содержательные замечания.

Курс можно найти тут: https://system-school.ru/engineering, чат поддержки https://t.me/systemsthinking_course, там вы можете не только задать вопросы студентам и преподавателям курса, но и дать предложения авторам курса. Курс регулярно обновляется, ибо системная инженерия быстро развивается, а курс уделяет особое внимание изучению современных практик, а не старинных (в инженерии старинными будут практики буквально десятилетней давности).

1. Безмасштабная непрерывная системная инженерия

Инженерия

Инженерия – это практика изменения физического мира к лучшему. По факту это означает или создание каких-то новых, ранее не существовавших систем, или изменение каких-то уже существующих систем. Иногда «изменение» означает и уничтожение, если это как-то меняет мир к лучшему. Как понимается «изменение к лучшему»? Тут есть несколько вариантов:

• Этические ответы, что такое «изменения к лучшему» (этика – это фундаментальная практика, входящая в трансдисциплинарный интеллект-стек и выполняемая ролью совести).

• Изменения к лучшему – это многоуровневые эволюционные непрерывные оптимизации, отражающие неустроенности/frustrations от конфликтов между системными уровнями (было более подробно рассмотрено в курсе практического системного мышления). Ключевое тут – это понимание, что инженерный проект проходит как часть техноэволюции, часть бесконечного развития цивилизации, часть процесса деятельного (изменяющего мир!) познания, тем самым он не заканчивается разовым проведением какой-то системы по жизненному циклу, а непрерывно вводит в эксплуатацию новые и новые изменения, ввод в эксплуатацию непрерывен/continuous, он не происходит один раз за время проекта, это не «водопад» для одного организма, а эволюция для технического вида (и неважно, сколько там было организмов в ходе эволюции: один с заменяемыми частями, или множество последовательных). В инженерии, как и биологической эволюции, доступны множество «почти одинаково выгодных» решений, выгода всегда многоуровневая, ухудшения часты, нейтральные решения более редки, крупные улучшения встречаются очень редко.

• Изменения к лучшему – это действия по минимизации свободной энергии целевой системы и/или её надсистемы, согласно теории деятельного рассуждения/active inference. Другое дело, что при этом неизбежно будут оставаться и даже множиться (эволюционный рост сложности, увеличение числа системных уровней!) неустроенности в целевых системах, и приходится снова и снова инженерно/деятельно/практично решать сложную оптимизационную задачу на многих системных уровнях, опять всё упирается в непрерывность этих изменений. Ключевое «к лучшему» тут – это более длительные сроки удержания целостности какой-то системы за счёт постоянных изменений состояния этой системы перед лицом много превосходящих внешних воздействий, грубо говоря «к лучшему» – это когда меньше ожидания незапланированного разрушения системы, меньше байесовских неприятных «сюрпризов» типа «тебя съели», «ты умер от голода», «ты не смог размножиться» (конечно, это просто метафора для инженерных систем, но спросите у серийных предпринимателей, насколько они воспринимают такие высказывания как метафоры по отношению к их бизнесам по изменению мира к лучшему, и вы поймёте, что в каждой шутке есть доля шутки. Про эту долю шутки мы ещё не раз вспомним в этом курсе).

Конечно, есть и множество других вариантов понимания «к лучшему», но мы ограничимся этими. Главное, что все эти понимания предусматривают физические изменения в мире как конечный результат (появление, изменение или исчезновение какой-то системы или целого ряда систем), а также указание на то, что это изменение было «к лучшему», то есть целенаправленное, а не любое изменение.

Современная (SoTA, state-of-the-art, лучшая из известных на сегодняшний момент, и именно о такой мы будем говорить в нашем курсе) инженерия является **системной**, ибо эво-

люционирующий усложняющийся мир имеет много системных уровней, число которых непрерывно растёт, и каждый из которых требует каких-то особых приёмов работы систем создания, методов изменения мира на каком-то уровне масштаба и на этом уровне масштаба часто ещё и есть различия методов работы с разными видами систем (так, с мышами работают не так, как с тиграми – хотя в части генетики они очень похожи).

Согласно приведённому определению, **SoTA инженерия должна быть системной и не просто многоуровневой, но и безмасштабной (вообще не привязанной к каким-то уровням), и непрерывной. Безмасштабность (общность основных положений инженерии для всех возможных масштабов, отсутствие указаний на масштаб в инженерной дисциплине) гарантирует, что инженеры/деятели/практики не остановятся в своей оптимизации мира «к лучшему» на границе какого-то масштаба. Непрерывность означает, что каждая «умная мутация» в порядке попытки оптимизации не будет разовой, поток этих мутаций их влияния на успешность системы будет в проекте нескончаем, а успешность системы будет непрерывно отслеживаться во времени, а не удостоверяться разово на момент единичного за время жизни системы ввода в эксплуатацию.**

Безмасштабность означает, что инженеры (которые, конечно, не будут называться инженерами, а будут называться самыми разными другими ролями) не изменят личность так, что от этого развалится общество, не сделают киберфизическую систему такую, что она уничтожит всё живое в радиусе трёх километров. Можете подумать о том, как по-разному аргументировали это безмасштабное «изменить мир к лучшему» инженеры, создававшие атомную бомбу, на каком уровне они останавливали свои рассуждения, примат какого уровня был у них в голове – вещества, киберфизической системы, существа, личности, организации, сообщества, общества, человечества?

Непрерывность означает, что одно разовое изменение вряд ли возможно, и нужно непрерывно подстраивать систему под стремительно меняющееся окружение, непрерывно вводить в эксплуатацию всё новые и новые «фичи» как результат «умных мутаций». Какие-то из этих новых фич позволят системе выжить среди систем-конкурентов, какие-то окажутся ненужными и временными, но непрерывность означает, что инженерия не разово делает системы, а evolve/«эволюционирует» их. По-русски это лучше будет говорить «развивает» (ибо «выращивает» сложно говорить про какой-нибудь ледокол-атомоход, но можно сказать, что этот ледокол-атомоход развивается. Впрочем, идея непрерывности инженерии до атомоходов ещё не дошла, её мы больше наблюдаем на примере программного обеспечения. Посмотрите, как часто обновляется Windows или даже BIOS в вашем ноутбуке).

Альтернативная формулировка инженерии – это **создание успешных систем** (также было подробно разобрано в курсе «Практическое системное мышление»). Успешность определяется как соответствие ожиданиям внешних проектных ролей, которых множество самых разных (и это требует безмасштабности в работе с системами, интересными для разных ролей) и ещё эти ожидания меняются (что требует непрерывности изменений целевой системы, её развития под эти изменяющиеся ожидания). Это определение более традиционно и основывается на классических рассмотрениях проектов системной инженерии. Если со всеми договорились (с учётом соображений безопасности и того, кого считать врагом, а кого считать другом и клиентом), и продолжают договариваться всё время существования системы, то система успешна. Если не договорились, и при изменениях ситуации не успевают передоговариваться (непрерывное договаривание, а не разовое) – не успешна. Тут по факту тоже оптимизационная задача, но рассмотрение не столько многоуровневое/безмасштабное системное, сколько деятельное/методологическое/практическое по линии разнообразия ролей, предметов их интереса и интересов, а также выходящее за рамки одного жизненного цикла техноэволюционное рассмотрение.

«Успешность системы» – в разы более простой критерий: «если все внешние проектные роли, которые мы выявили, непрерывно соглашаются с проектом, то это и означает, что мы мир изменяем к лучшему – наша система непрерывно успешна». Классическая системная инженерия ещё несколько лет назад даже выкинула бы слово «непрерывно», ибо в головах системных инженеров прошлого система изготавливалась однократно, и больше не менялась по ходу эксплуатации, не нужно дальше ни с кем договариваться, не нужно менять систему. Модернизации системы, конечно, предусматривались, но они были абсолютно автономными тоже разовыми проектами, а не частью непрерывно ведущейся инженерии. Ближе тут, конечно, была концепция апгрейда (например, апгрейд компьютеров, который был популярен в 90е годы прошлого века). Но это была «инженерия пользователя» для редкого класса систем, а не штатный способ организации инженерной деятельности. В телефонах наоборот: апгрейд аппаратный был запрещён (невытаскиваемая батарейка, отсутствие слота для внешней памяти), зато расцвёл апгрейд прошивок и апгрейд приложений. Телефон перестал быть неизменяемым объектом, даже когда попадал к покупателю. Это, конечно, создаёт и множество проблем (так, чей это теперь телефон, если его купили, но продавец продолжает его «производить»? Ответ не такой очевидный[2]).

Но тут кроется и много проблем: разработка обычно центрируется на поведении целевой системы в составе непосредственной её надсистемы, то есть главным образом на двух системных уровнях. Если в таком подходе разрабатывается киберфизическая система, то могут не учитываться соображения более высоких системных/эволюционных уровней – включая уровни эко-системы (сообщества), общества, человечества.

Необязательно речь об оружии, где мнение противника о том, что считать «успешным оружием», заведомо будет учтено с обратным знаком. Например, представьте какую-нибудь социальную сеть или новостной портал, их влияние на общество в целом как систему: как его учесть, кто из внешних и внутренних проектных ролей этим озабочен? Конечно, идеал для таких сервисов – это максимизация времени пребывания на этом портале всех, кто туда зачем-то зайдёт, алгоритмы настроены именно на это. И эти алгоритмы постоянно перенастраиваются по мере изменения интересов широкой публики, чтобы никто от этой максимизации не убежал (инженерия непрерывна). Но полезно ли это для общества в целом? Сутками «тупить в вконтакт/тикток/телеграм» хорошо ли для личности, сообщества, общества, человечества? А ведь именно длительность времени «тупления в соцсеть/портал» будет критерием успешности разработчиков соцсети/портала и акционеров фирмы-провайдера соцсети/портала как успешного сервиса по возможности «тупить» и потреблять при этом рекламу вместо чего-то полезного (да и что тут будет «полезным»?). И искусство пиара/пропаганды тут подсказать клиентам сети/портала, что «тупление в соцсетях» надо называть «развлечением и отдыхом», так называемый «рефрейминг», переформулирование зла в добро.

Могут не учитываться соображения внешних проектных ролей и по более низким системным уровням, чем целевой. Если это идеологический геополитический или религиозный проект, то вполне может зайти речь об игнорировании интересов огромного числа обычных граждан – их жизни могут приносить в жертву каким-то геополитическим или религиозным целям. «Обществу» или «сообществу» будет ой как хорошо (например, какое-то общество выиграет в войне), но вот нескольким миллионам человек этого общества вдруг может оказаться ой как плохо по сравнению с вариантом без войны, но они будут проигнорированы, ибо геополитические или религиозные проекты не работают с масштабом отдельных людей! А потом пройдёт немного времени, и мнения людей поменяются. Если идеологический проект

[2] https://ailev.livejournal.com/1106188.html – и это текст ещё 2014 года. События 2022 года показали, что в этом мире «непрерывной разработки» отключить миллионы людей от пользования самыми разными системами стало не просто легко, а очень легко.

не организован как непрерывная инженерия, это не будет отслежено, и принесение в жертву одних людей вдруг окажется принесением в жертву совсем других людей: «хотели как лучше, получилось как всегда».

Системность инженерии как задающая её безмасштабность и непрерывность

Особенность современной инженерии в том, что она по факту стала системной. Это означает, что она сразу принимает постулаты системного подхода в явной (классическая системная инженерия) или неявной форме (разные другие виды инженерий):

• Изменение мира происходит на нескольких системных уровнях, выделяемых по отношению «часть-целое» несколькими разными способами в момент эксплуатации, это тезис первого поколения системного мышления (это всё подробно разъяснялось в курсе практического системного мышления).

• Практики работы по созданию системы (второе поколение, появились создатели системы) на разных системных уровнях разные, у них даже могут меняться имена. Если «железную» деталь «изготавливают», то мастерству человека (или даже робота) «обучают» или его «осваивают» (в зависимости от внешней или внутренней позиции восприятия, у детали ведь нет внутренней позиции восприятия, а у человека есть, у робота – можно уже обсуждать, там «философская серая зона»). Описание поведения «железной» системы будут называть концепцией использования, у предприятия это будут называть стратегией. Но всё это одно и то же: практики изменения мира, причём целенаправленного, «к лучшему».

• Третье поколение системного мышления говорит, что однократным созданием системы дело не обходится, а речь идёт о техноэволюции, которая подчиняется тем же самым законам, что эволюция. Непрерывно оптимизируем конструкцию системы и подстраиваем её функции, чтобы как-то адаптироваться к непрерывно меняющемуся окружению.

Если принять эти положения в полной мере, то системная инженерия будет безмасштабна/scaleless: не зависеть от масштаба/размера систем, эволюционного системного уровня. Термин этот идёт из физики, иногда об этом говорят как многомасштабность/multiscale, но это менее точно: применимость ко всем масштабам (безмасштабность) – это вовсе не учёт только нескольких выделенных заранее масштабов, но отсутствие привязки к масштабу в рассуждениях.

Безмасштабность означает «вынос за скобки» того общего, что имеет инженерия как изменение мира к лучшему на всех масштабах, всех уровнях эволюционного стека, попытка учесть все неустроенности, оптимизировать все конфликты между всеми уровнями (даже теми, о которых мы не знаем! Как минимум, сделать попытку узнать о них!). И современные инженерные проекты будут это делать не разово в попытке угадать какой-то «оптимум» для «текущей ситуации», а довольно долго, развивая систему для подстройки под непрерывно меняющиеся обстоятельства. Впрочем, если считать «безмасштабность» применимой и ко времени (масштаб времени эксплуатации, масштаб времени жизненного цикла, масштаб эволюционного времени многих жизненных циклов), то непрерывность инженерии тоже следствие безмасштабности: учитывается не один масштаб времени, а все.

В нашем курсе мы довольно грубо и почти произвольно в методических/учебных целях определяем уровни этого системноинженерного/эволюционного стека физических масштабов как (от более крупных к более мелким масштабам):

• **Вселенная** (пока разговоры об инженерии для вселенной в целом носят сугубо теоретический характер, а ещё ведь есть и концепция «мультиверса», как множественности миров. Но приводим тут как предельный масштаб/размер)

• **Инопланетные цивилизации в целом** (тут тоже чисто гипотетическая инженерия, уровень привели только для того, чтобы лучше показать принцип выделения уровней по отношению часть-целое)

• **Человечество** – мы тут одни на маленьком одном глобусе (хотя есть мысли уже насчёт Марса), и неплохо бы изменить всё человечество к лучшему (в том числе защитить от астероидных опасностей, вымирания от техногенных катастроф, барьеров роста из-за ограничений в экологии и т.д.)

• **Общество** – каталлактическая самоорганизация по фон Хайеку[3] (не знаем контрагентов в лицо, ибо их слишком много, не знаем интересов этих контрагентов, ибо они могут занимать самые разные роли), но общество в целом тоже может быть достаточно организованным и некаталлактически (то есть через силовые органы власти как организации с понятным подчинением принуждать всех остальных исполнять приказы нескольких человек), чтобы выставлять границы на свою территорию и противодействовать своему исчезновению при агрессивных изменениях окружения (скажем, войны с другими обществами) или изменениях в своей организации (скажем, захват власти религиозным фанатиком или диктатором и последующая сознательная или даже неосознаваемая их работа по уничтожению этого общества в силу каких-то иррациональных убеждений)

• **Сообщество** – определения subculture, counterculture, community, community of practice, learned society, school of thought из википедии: все говорят об одном и том же, подчёркивая разные стороны. И мы тоже можем говорить разным языком, подчёркивая разные стороны. У нас субкультура, контркультура, сообщество практики, сообщество наученных (деятелей, не учёных!), «незримый колледж». Но можно говорить и о племени (очень модно!), и о «муниципальном образовании» типа «деревня, где все друг друга знают», землячестве бывших (или даже нынешних) членов одного общества в каком-то другом обществе. Ориентируемся на число Данбара[4] (150), но с учётом современных компьютерных средств («записных книжек», социальных сетей) человеческая «память на лица» быстро растёт. Но в сообществе всё равно может быть и больше людей/личностей, чем можно помнить даже с помощью записной книжки. Например, число системных инженеров киберфизических систем, которые вступили в Международный совет по системной инженерии на 2022 год – более 19000, это означает, что эти люди считают себя членами сообщества системных инженеров, тем самым примерно известны их ролевые интересы, но это не означает, что речь идёт о традиционной организации, которая создаёт какую-то целевую систему, или что все там примерно знакомы друг с другом и доверяют друг другу сильно больше, чем своим соседям по улице. Сотрудники штаб-квартиры INCOSE, однако, занимаются построением сообщества системных инженеров как целевой системы (об INCOSE в нашем курсе будет отдельно сказано через несколько разделов). А маркетологи пытаются делать сообщества потребителей каких-то продуктов, «клиентуры» как множества клиентов.

• **Организация** – это проекты (организованные коллективы с их компьютерами, то есть с понятными ролями и полномочиями каждого агента в коллективе). Инженерия предприятия как «организовывание» работает с системами этого уровня, инженерия на стадии эксплуатации получившейся организации – это операционный менеджмент. Есть отдельный курс системного менеджмента[5], а связь прикладной инженерии целевой системы (разработки) и инженерии предприятия-создателя системы (системного менеджмента) обсуждается в нашем курсе в разделе непрерывной разработки.

---

[3] https://ru.wikipedia.org/wiki/Каталлактика

[4] https://ru.wikipedia.org/wiki/Число_Данбара

[5] https://system-school.ru/systems-management

• **Личность** – и тут «инженерия личности» (хотя так и не говорят, но мы советуем думать именно так!) сводится к разным психотерапиям («ремонт» личности), коучингу (помощь в стратегировании), образованию, просвещению. Включать ли сюда разные варианты обучения «разумных роботов», то есть личностей не на базе homo sapiens (xGI, разные виды general intelligence[6])? Почему бы и нет. Поэтому тут можно рассматривать и проекты AGI (artificial general intelligence), и проекты создания киборгов в части экзокортекса (грубо говоря, «добавка памяти и возможности вычислений», ибо добавка искусственного зрения, печени и т. д. – это уже более низкий системный уровень, «существа»).

• **Существо** – тут разведение живых существ, включая «системную биологию» как разработку и производство полностью искусственной жизни. Но в целом тут инженерия и вирусов, и бактерий, и растений (агрономия), а дальше червей, рыб и до зверей (включая приматов и даже человека вне аспектов его разума). Медицина/ветеринария, фермерство, генная инженерия – это всё оказывается инженерными дисциплинами, «изменением мира к лучшему». Сюда же трансгуманизм как отрыв личности от обязательного использования генома homo sapiens (хотя усиление интеллекта живых организмов уровня выше возможности планировать представляется принадлежащим к уровню личности. Граница существа и «разумного существа» весьма размыта, как и граница «робота», «киборга», «существа»).

• **Киберфизические системы** (физические системы, включающие софт на базе универсального компьютера). Тут классическая системная инженерия с её примерами ракет и самолётов, атомных и солнечных электростанций, умных (то есть обвешанных датчиками и кое-какими исполнительными устройствами) домов и прочих традиционных для инженерии объектов.

• **Косное вещество** (молекулы, простые физические детали): тут от органического синтеза до инженерии мыльниц и электролампочек, то есть инженерия простых систем без сложного управления с контроллерами-компьютерами в их составе.

Конечно, в системноинженерном/эволюционном стеке все обычные проблемы представления системных уровней какой-то системы в виде даже не дерева, а «стека»:

• Каждый уровень включает не одну систему, а множество их самых разных, это больше «иерархия/дерево», а не «стек».

• Легко находится множество подуровней внутри одного уровня, и может существовать множество вариантов такого стека

• Не оговорено, это функциональное или конструктивное разбиение, а также возможность одновременной множественности таких разбиений (считаем, что на границах уровней оно совпадает, но это не всегда факт: так, внутри одного и того же общества легко выделить разные сообщества, составленные из одних и тех же людей, и даже разные организации. СМД-методология называет это «популятивными объектами»[7], когда люди одновременно входят в состав разных их устойчивых групп, термин «популятивности» позаимствован был из биологии).

Тем не менее, мы принимаем за основу курса именно такой системный/эволюционный стек. Главное в курсе – научить компактному типовому инженерному рассуждению о том, как создавать системы, которое можно применять рекурсивно (к разным уровням систем) и ите-

[6] См. обсуждения в постах «Онтологический статус интеллект-стека: мы не устраняем inductive bias, мы приветствуем его!», https://ailev.livejournal.com/1598826.html; «Инженерия и исследования xGI: учитесь сами, чтобы научить и людей, и нежить», https://ailev.livejournal.com/1600567.html; «Мои претензии в xGI: я не натуральное хозяйство, я участник разделения труда», https://ailev.livejournal.com/1600861.html.

[7] https://fondgp.ru/publications/problemy-postroeniya-sistemnoy-teori/

ративно (в разные моменты времени создания и существования системы), независимо от масштаба системы и её функционального назначения.

Масштабы времени, важные для выхода в непрерывную разработку:

• Эксплуатация одной версии системы
• Жизненный цикл одной версии системы
• Непрерывная инженерия множества версий (множество жизненных циклов «замысел-проектирование-изготовление-эксплуатация-вывод из эксплуатации»).

Конечно, вы можете, и даже должны адаптировать эти стеки для ситуации вашего проекта. Самый простой способ это сделать – это выписать конкретные масштабы и виды систем из вашего проекта, аннотировав их типами из нашего курса/учебника (помня при этом, что эти уровни выделены более-менее произвольно, для целей упрощения понимания). Например, если вы создаёте таблетку из лактобактерий как биодобавку, вам потребуется как-то изменить (в том числе уговорить принимать эту таблетку!) системы на следующих уровнях стека:

• Потребнадзор::сообщество и общество
• Клиентура::сообщество (маркетинг)
• Клиент::личность
• Желудок и кишечник::существо
• Таблетка::вещество
• Лактобактерия::существо

В этом примере есть даже два «перескока уровней»: лактобактерии производятся как живые системы, а затем высушенные становятся частью вполне неживой таблетки. Потребнадзор тут половинчат (обычное свойство госорганов): отражает мнение сообщества медиков с одной стороны и вроде как мнение общества в целом. Его тоже нужно изменить (уговорить считать таблетку безопасной). Лактобактерии – это биоактивные добавки (грубо – сушёный кефир), они не требуют медицинского лицензирования. Если бы это было лекарство, то ситуация была бы более сложной.

Суть всего этого представления – это понимание, что инженерная работа будет на всех этих уровнях. Поэтому в проекте придётся обратиться к самым разным инженерам: которые умеют работать с существами, работать с разнообразными предметами из косных веществ, работать с личностями, работать с сообществами и даже обществом в целом (если выяснится, что это как-то затрагивает общественные интересы и нужно вести какие-то общественные дискуссии, менять законодательство и тем самым заниматься политикой).

Потребуется каждый раз описывать функциональность системы и определять приоритеты в архитектурных характеристиках (надёжность, изменяемость и т.д.) для систем из каждого уровня, принимать архитектурные решения, разрабатывать концепцию системы и изготавливать её, вводить в эксплуатацию и заниматься всеми остальными инженерными практиками, хотя они и будут носить разные имена и для веществ, и для существ, и для личностей, и так далее. И это простой и обозримый пример, хотя уже и в нём нужно затрагивать проблемы высоких уровней. Скажем, все штаммы молочнокислых бактерий, грубо говоря, «штаммы кефира и йогурта разных сортов», обладают примерно одинаковой полезностью по влиянию на микрофлору и уж точно не являются лекарствами, их даже «биодобавками» считать сложно – если вы упакуете кефир или йогурт в капсулы, они ж не станут от этого «биодобавками»? Но вы можете использовать «связи в министерстве», чтобы этот кефир с йогуртом в капсулах назвать лекарством, получить лицензию и устроить на этом маркетинг! Это этично, или нет? Хорошо ли от этого будет людям (они получат плацебо!), а хорошо ли будет (гражданскому) обществу?

А теперь представьте реальную разработку: вряд ли проект закончится выпуском партии таблеток. Скорее всего, команда будет готовить другие версии – улучшать упаковку, вещество, варьировать цену, увеличивать разнообразие вариантов в надежде на привлечение покупателей, менять название и рекламные слоганы, получать новые сертификаты и лицензии, собирать статистику по итогам применения, и т. д. Инженерия таблетки оказывается не разовой, а непрерывной, при этом работа идёт отнюдь не только с веществом таблетки, она вполне многомасштабна.

Если вы хотите изменить мир хоть неживой, хоть живой, хоть в небольших масштабах, хоть в больших – вы должны будете предположить функцию вашей системы, описать конструкцию, изменить физический мир, чтобы реализовать конструкцию, а потом подстраивать получившуюся систему к непрерывно меняющимся условиям, причём делать это на множестве системных уровней на множестве масштабов времени. Неважно какими вы словами это называете, насколько различны те системы, которые вы затрагиваете своими изменениями мира к лучшему, и насколько сильны традиции работы с этими системами (в этих традициях может что-то не учитываться из всего перечисленного, они же именно «традиции», поэтому могут не учитывать знаний современной инженерии). Вы должны всё это делать, и делать на многих масштабах/системных уровнях как вещества, так и масштабах времени (эксплуатация, один жизненный цикл системы или фичи, эволюция как множество жизненных циклов). Поэтому такой подход и называется системной инженерией, а не просто инженерией. Безмасштабность и непрерывность просто характеризуют её современное состояние, отвечающее третьему поколению системного мышления, появившемуся по историческим масштабам совсем недавно, в десятые годы 21 века.

Инженерия и эволюция

Как соотносятся инженерия и эволюция? Технический прогресс/техноэволюция, которая делается инженерами – это просто часть эволюции? Или это не эволюция, а просто инженерия, «практика агентов-людей с приданными им компьютерами»? Как об этом думать? Участвуют ли инженеры в эволюции, или они и есть эволюция, или эволюция сама по себе, а инженеры творят сами по себе, вне эволюции?

Думать об этом нужно как об эволюции через интеллект (evolution through intelligence). Инженеры реализуют эволюционный алгоритм, но в этом алгоритме есть оптимизации, связанные с использованием интеллекта как общего мастерства решения самых разных проблем, которые не встречались раньше, подробней это раскрывается в курсе «Образование для образованных»[8].

Как описывалось в курсе «Практическое системное мышление», во вселенной действует эволюционный физический процесс, который можно представить как (вполне деятельное, то есть физичное) оптимизационное вычисление, биологическая/дарвиновская эволюция тут часть этого общего эволюционного физического процесса. Алгоритм эволюции как оптимизационного вычисления кратко для случая биологии выражается в центральной догме молекулярной биологии[9]: в ходе эволюции обязательно появляется медиа с возможностью цифровой записи информации репликаторов. Цифровая запись гарантирует точную многократную репликацию без накопления ошибки (в случае аналоговой записи ошибки накапливаются, точная репликация становится невозможной).

На цифровом носителе в ходе эволюции записываются программы генотипа (речь на Земле идёт о ДНК и отчасти РНК, которые хранят информацию в цифровой форме), затем эта цифровая информация разворачивается в уже аналоговый фенотип, и далее она проходит от генов через проявления в фенотипе на уровень популяции – в том числе разделение на два пола, стайный образ жизни, особенности воспитания детей, поведения по терраформированию (например, строительство плотин бобрами) и т. д.

Это (от генотипа к фенотипу и далее, включая популяционные уровни и даже социальную эволюцию, включая техноэволюцию) прямой ход накапливающегося в ходе эволюции оптимизационного знания, приводящего к меньшему влиянию сюрпризов окружающей среды на агентов/IPU. А вот назад в гены полученный в ходе жизни организмов и популяций опыт идёт совсем другим способом, симметрия тут нарушена: в гены удачные модификации попадают только в ходе мутаций, и если они хороши, то репликация оригинала с мутацией дальше происходит, а если не очень хороши, то не происходит, ибо фенотип (вместе с его популяцией, если она оказалась недостаточно разнообразна в части мутаций) вымирает.

Какие цели эволюции, что она оптимизирует? Она реализует физический принцип минимального действия, который в данном случае трактуется как информационный: минимизация свободной энергии, что можно перефразировать как минимизация неприятного сюрприза, который агент/IPU получит от окружающей среды. То есть эволюция борется с энтропией, которая рано или поздно присылает какое-то существенное изменение внешних условий. Достаточно подумать о более длительных масштабах, нежели «догнали и съели» или «умер от голода»: удары астероида, взрывы сверхновых и т. д. Не вопрос, будет ли катастрофа. Вопрос в том, когда будет, и насколько удастся жизни проэволюционировать к этому моменту, чтобы не исчезнуть. Пока достижений эволюции хватило примерно на 3.5 миллиарда лет репликации механизмов, похожих на клетки, но репликация людей пошла уже немного по-другому: они

[8] https://system-school.ru/uptodate

[9] Помним, что вся литература и более подробное изложение есть в «Практическом системном мышлении».

получились очень устойчивыми, и хорошо размножились. Население Земли сегодня составляет порядка 8 миллиардов человек[10]. Курс «Практическое системное мышление» начинается как раз с того, что сравнивает массу всех людей с биомассой всех остальных видов, а также оценки массы преобразованной «неживой» части земли (автор тут задумчиво смотрит в окно и оценивает массу домов, домашней утвари, дорожных покрытий, производств, еды и отходов в радиусе 100 км вокруг себя, а живёт он в центре Москвы).

Эволюция усложняет и усложняет реплицирующихся (то есть «создающих копии себя») агентов/IPU как «создателей», чтобы эти репликаторы были более и более устойчивы к воздействиям окружающей среды, чтобы были способны ко всё более длительному выживанию во всегда в конечном итоге враждебной среде. Муравейник потыкать палкой – он за ночь восстанавливается, но и если город потыкать какой-нибудь огромной палкой (печальные прецеденты с атомными бомбами, да и просто тысячами тонн взрывчатки были), то город тоже восстанавливается, причём масштабы этого «потыкать» несопоставимы. Город восстанавливается во много бóльших масштабах, и речь идёт именно о количественных характеристиках: масса, энергия и скорость (учёт массы, преобразуемой за какую-то единицу времени).

Итак, эволюционный процесс имеет целью получить устойчивый репликатор (минимизация свободной энергии сводится тут к минимизации байесовского сюрприза от внешней среды – это и есть цель эволюции), который копируется и копируется, выживая всё круче и круче на всё новых и новых уровнях квазиустойчивости. При этом устойчивость к внешним воздействиям и адаптивность резко повышается время от времени за счёт роста сложности в больших эволюционных сдвигах: многоклеточные организмы, популяции, паразитизм и симбиотика (когда паразит и хозяин вдруг находят взаимную, а не одностороннюю выгоду. Когда-то наши митохондрии были паразитами в клетках).

Между системными уровнями есть конфликты, квазиоптимальных оптимизационных/архитектурных решений на предмет минимизации негативных сюрпризов от проявления этих конфликтов много и от этого возникает неустроенность (буйство самых разных видов, которые примерно одинаково выживают, это относится и к буйству моделей автомобилей, телефонов и т. д. – характеристики их выживаемости в природе или на рынке примерно равны, равно как и страны с постоянно меняющимися границами имеют более-менее одинаковые условия для жизни, они существенно отличаются только в деталях, сравнивать нужно тут просто с достаточно далёкими моментами в истории, например со средневековьем или даже с каменным веком, разница сегодняшнего дня в части устойчивости к массовому вымиранию людей будет хорошо видна).

Иногда в ходе эволюции возникают существенные оптимизации, типа тех самых переходов от одноклеточных к многоклеточным, или появление хорошей цифровой памяти типа мозга с ручкой-бумажкой, или на следующем этапе типа компьютеров с интернетом. Мы просто продолжаем пересказывать идеи работ Ванчурина-Кацнельсона-Вольфа-Кунина, опять отсылаем для более обстоятельного рассказа об эволюции к «Прикладному системному мышлению», подробности тут приводить не будем.

Эволюционный алгоритм можно ускорить через моделирование, то есть реплицировать только важное, а потом фенотип (включая популяции!) заставлять в самых важных аспектах проживать свою жизнь, доказывая свою живучесть, в компьютере/виртуальном мире, достаточно большом мире, чтобы вмещать популяции и моделировать более-менее точно эффекты от взаимодействия популяций с окружающей их богатой средой, возможно содержащей и другие популяции. Если иметь достаточные вычислительные мощности, то делать это можно быстрее, чем проживать популяциям полную жизнь в реальном мире. Можно считать это «ускоренным воспроизведением», да ещё и параллельно можно пробовать множество разных вариантов,

[10] https://countrymeters.info/ru/World

но опять же, если хватает вычислительных мощностей. Эволюция крайне затратна в вычислениях!

Переход к «проживанию модели в компьютере» это вроде как уже инженерная работа, «применение вычислителей для генерации догадок о полезных мутациях, а потом фильтрации догадок об удачных мутациях». Инженерия тут в том, что физически создаётся вычислитель виртуального мира, а в нём создаются процессы «проживания» для моделей агентов/IPU (включая популяции!) с потенциально интересными мутациями. Это отличный способ ускорить прогресс, но у него есть существенные ограничения: требуются немыслимо большие вычислительные мощности на моделирование N миров, в которых живут и размножаются организмы и популяции, на которых мы пробуем те или иные мутации. С этим боремся так: уменьшаем объём моделирования (скажем, пытаемся вычислить только догадку о мутации, но не моделируем выживание), увеличиваем доступную компьютерную мощь, увеличиваем эффективность эволюционного алгоритма в целом (качество генерирования догадок, точность моделирования и т.д.).

Например, в статье Evolution through Large Models[11], предлагается использовать внутри эволюционного алгоритма вместо случайных мутаций «умные», то есть «разработанные/вычисленные», а вместо человека-инженера, высказывающего догадки о полезных мутациях, использовать нейронную сетку большой языковой модели (модели языка и мира, выученной нейронной сеткой определённой архитектуры).

Основная проблема в инженерии путём генерации новых архитектур нейронными сетями в том, что архитектура должна давать вариант многоуровневой оптимизации конфликтов между системными уровнями, при этом нейросеть обычно не может сгенерировать оптимизацию, которая выходит за рамки тех примеров, которые ей показывали в ходе обучения. А эволюционный алгоритм принципиально может. Поэтому статья предлагает оставить снаружи эволюционный алгоритм, принципиально дающий новизну решений за пределами того, что уже видела нейронная сеть, но вставляем внутрь «умный мутационный оператор» на основе нейросети, который предлагает потенциально не смертельную мутацию. Помним, что в работах по эволюции как многоуровневой оптимизации подчёркивалось, что всё в ходе эволюции давно уже квазиоптимизировано, какие-то локальные оптимумы для минимума свободной энергии целевой системы достигнуты, поэтому большинство случайных мутаций будут смертельными (выходим за рамки локального оптимума в менее благоприятные зоны), немного их будут нейтральными (находим другой квазиоптимум, они очень близки обычно), редко что-то приводит к маленькому улучшению текущего локального оптимума, и совсем уж редко что-то радикально приближает к обычно недостижимому глобальному оптимуму.

Если в основу эволюции брать не проверки по большому объёму бессмысленных смертельных мутаций, а проверки по менее большому объёму заведомо более осмысленных предложений, эволюция в целом пойдёт быстрее. Статья демонстрирует это на примере эволюционного алгоритма для генерации программного кода, порождаемого нейросетью ровно как это делают системы подсказок для кода программ: GitHub Copilot и Amazon CodeWhisperer. Если грубо, то статья предлагает вместо случайных перестановок текста в генетических алгоритмах использовать подсказки всех этих Copilots и CodeWhisperers[12]. И там же даётся ещё много разных других способов ускорить эволюцию (ибо много ещё мест, где «универсальный аппроксиматор/оптимизатор» типа нейронной сетки можно задействовать в эволюционном алгоритме).

Если представить, что у всех инженеров в голове есть вычислители этих самых «умных мутаций» и других оптимизационных вставок в эволюционный алгоритм, да ещё в последнее время эти «вычислители в голове» умощняются компьютерами и средствами компьютерной

[11] https://arxiv.org/abs/2206.08896

[12] https://techcrunch.com/2022/06/23/amazon-launches-codewhisperer-its-ai-pair-programming-tool/

связи для объединения вычислений многих людей и компьютеров, то мы смело можем считать инженеров всей Земли ускорителями эволюции. Мы берём инженеров вместе с их моделерами и заводами: моделеры документируют мутации, а заводы производят какие-то продукты, которые потом «пробуются жизнью на соответствие среде», некоторые из них оказываются удачны настолько, что предложенные в них «мутации» оказываются достойными повторения. А поскольку в нашем подходе инженеры – это те, кто что-то делают/практикуются/трудятся, то всё человечество оказывается мощным вычислителем-ускорителем эволюции, то есть работает на то, чтобы минимизировать негативные сюрпризы от окружающей среды, это и есть «изменение жизни к лучшему». Сложность окружающего мира продолжает расти, но уже не только за счёт классической биологической эволюции, но и за счёт технологической эволюции, которую производят люди, и которая просто часть общей «физической» эволюции.

Итак, если инженеры угадали техническое решение, то «бинго, вымри твой вид продукта сегодня, а мой вид завтра». Если не угадали, то будет наоборот: финансирование работ тех, кто не угадал, будет прекращено. В эволюции биологической вид вымирает, и эта неудачная ветвь просто прекращает существование. В технологической эволюции ничего страшного не происходит: все люди и их компьютеры обычно остаются живыми и просто производят следующий вариант «умной мутации техносреды», пробуют что-то ещё.

Так что для уменьшения бизнес-неудач остаётся поднять вероятность угадывания хороших концептуальных (найти аффорданс/affordance: как какую-то функцию реализовать каким-то объектом из окружения, решения по концепции использования и концепции системы) и архитектурных решений (способ разбиения конструктивных объектов на части-модули и организация взаимодействия их такие, что архитектурные характеристики становятся «наименее плохие из возможных») как «умных мутаций», и это ровно то, что должны бы делать нейросетки: 1. Для языковых моделей типа T5 или GPT-3 поднимать степень осмысленности того, что они выдают в ответ на какие-то промпты/prompt[13], то есть улучшать и нейросети и способы генерирования промптов как запросов к нейросети на продолжение промпта (принцип работы моделей типа T5: они просто продолжают какую-то строку-промпт, например промпт «2x2=» наверняка получит своим продолжением «4») и 2. учитывать для генерации «не совсем случайной мутации» максимум информации о мире. Ровно это и происходит: большие модели, выдающие эти самые мутации, могут использовать знание о мире, которое воплощено не только в инженерных кодах, компьютерных программах или информационных моделях (корпусная инженерия[14]), но и в просто текстах на естественном языке, а также фотографиях и других изображениях[15] и даже геноме человека[16], 3. Не останавливаться в своих попытках на первой же, а продолжать (непрерывная инженерия), пока позволяют ресурсы.

Дальше можно думать о том, чтобы запускать алгоритмы «умной мутации», то есть алгоритмы архитектурных решений для того, чтобы улучшать и эволюционные алгоритмы, и для того, чтобы улучшать мутационный оператор, и для того, чтобы улучшать моделирование мира для определения того, выживет ли индивид с фенотипом, определяемым мутацией в генотипе, в виртуальном мире, чтобы уменьшить время экспериментирования и ресурсы, требуемые для проверки выживаемости в физическом мире. Тут есть и альтернативные подходы, которые прямо говорят о генерации каких-то оптимальных архитектурных технических решений, без связи этого с идеями эволюции (то есть рассматривается один жизненный цикл, а не то, что

---

[13] https://en.wikipedia.org/wiki/Prompt_engineering

[14] https://ailev.livejournal.com/1009201.html

[15] Например, вот такие работы по мультимодальным вычислениям в нейронных сетях – Allen AI & UW Propose Unified-IO: A High-Performance, Task-Agnostic Model for CV, NLP, and Multi-Modal Tasks, https://arxiv.org/abs/2206.08916

[16] GENA_LM – первая в мире языковая модель ДНК, обученная на самой полной версии генома человека (T2T-CHM13), которая была опубликована в конце марта 2022 года, https://huggingface.co/AIRI-Institute/gena-lm-bert-base/, https://github.com/AIRI-Institute/GENA_LM

параллельно идёт конкуренция с другими проектами, которые предлагают другие варианты архитектурных решений какой-то проблемы). Скажем, можно «смягчать»/relax формулирование архитектурных проблем из языка дискретных решений так, чтобы получать набор вроде бы непрерывных, то есть дифференцируемых функций, искать архитектурный оптимум на них нейросетевыми или даже какими-то другими алгоритмами, а потом возвращать в дискретную область архитектурных решений для формулирования ответа на вопрос об оптимальной архитектуре[17].

[17] https://ailev.livejournal.com/1464563.html

Этические/политические проблемы естественной эволюции, которые можно решать инженерно

При таком подходе к архитектурной работе в инженерии как оптимизации конфликтов системных уровней в ходе эволюции появляется возможность по-новому взглянуть на старые этические/политические проблемы. Скажем, если мы применим «умные мутации» к человеческому геному, то получим евгенику[18] средствами генетической инженерии. Далее можно задать вопрос: сколько людей в идущей в том числе и сейчас биологической эволюции человека умрёт до появления «усиленной версии человека будущего» в ходе естественных случайных мутаций, и сколько людей умрёт в ходе вот такой «инженерной эволюции», где мутации будут предлагаться на основе какой-нибудь языковой модели генома на базе нейронной сети? Ведь вся суть предложений «смарт мутаций» как раз в том, чтобы уменьшить порождение потенциально вымирающих от несоответствия/misfit окружающей среде индивидов, которым суждено быть просто «неудачными ветвями эволюции»! Это алгоритмы увеличения количества живых, которые будут более живучи в мире, но при этом минимизирующих количество мертвецов, на которых пробовали разные мутации. Эволюция человека тут просто предельный случай, который сразу поднимает жёстко сформулированные этические вопросы жизни и смерти. Можно выделить два предельных мнения:

• «Руки прочь от естественной эволюции, что-нибудь повредите, всякие генетические инженерии и социальные инженерии опасны!», – это подход охранителей с их скрепами. Они достаточно сильны, в США недавно принято судебное решение как раз в области ограничения естественных влияний на рождение людей, а именно, запрет абортов[19].

• «Вы что, сдурели оставлять всё естественной эволюции, у неё ж на поиски решения уйдёт миллион лет и миллиард смертей! А то и вообще всё сдохнет, это ж эволюция! Срочно зовите инженеров!», – эту точку зрения представляет, например, John Doyle[20].

Генная инженерия поэтому сразу сталкивается с подобными вопросами, иногда это называют «биоэтика». Но вот эволюция той же нейронной сетки тут пример поближе к «обычной инженерии», но и тут могут быть вопросы (например, «можно ли выключать нейронную сеть, если мы заметили, что у неё есть сознание?»), или эволюция робота (и вопросы свободы воли робота), или эволюция атомных реакторов («безопасность человечества против доступа к дешёвой электроэнергии»).

Ещё жёстче стоит вопрос при выходе на уровни социальной инженерии, и тут уже работать могут не судебные механизмы против отдельных людей, а военные механизмы против больших групп людей (да, автор понимает, что привлечение примеров из политики может существенно отвлечь внимание и ему знаком текст «Политика – убийца разума»[21], но надеется на то, что из примеров будет вычитываться не собственно политика и политические решения по конкретным кейсам, а будет вычитываться применение принципов инженерии к сообществам, обществам, человечеству). Так Касым-Жомарт Кемелевич Токаев (президент Казахстана) сказал[22]: «Подсчитано, что если право наций на самоопределение в реальности будет реализовано, то вместо 193 государств, входящих в состав ООН, на земле возникнет более

[18] https://ru.wikipedia.org/wiki/Евгеника

[19] https://www.nbcnews.com/politics/supreme-court/supreme-court-wipes-away-constitutional-guarantee-abortion-rights-over-rcna18718

[20] https://ailev.livejournal.com/1622346.html

[21] https://lesswrong.ru/w/Политика_убийца_разума

[22] https://tass.ru/mezhdunarodnaya-panorama/14956829

500 или 600 государств, конечно же, это будет хаос. По этой причине не признаем ни Тайвань, ни Косово, ни Южную Осетию, ни Абхазию. И, по всей видимости, этот принцип будет применен в отношении квазигосударственных объединений, коими, на наш взгляд, являются Луганск и Донецк».

Тут даже неважно, если дать право людям на свободную сецессию[23], будет ли 500—600 государств, а не 3500. И почему сегодня в мире не хаос, когда всякие спорные полугорячие точки вроде Тайваня, Северного Кипра и так далее (включая потенциально страну басков, время от времени желающий отделиться Техас и сотни других регионов) можно войной или мирно забирать в другие страны, или наоборот, не отпускать, если считать их сепаратистами, или же считать «борцами за независимость» или «возвращающимися в родное старое государство» и это не ведёт к хаосу, а каким-то чудом его предотвращает, а 600 государств вдруг будут хаосом, если отпускать без войны. Война уничтожает людей, но мирное изменение границ государств само по себе не уничтожает людей, и уж точно не приводит к «хаосу», просто чьи-то амбиции по рулению себе подобными ущемляются, а чьи-то нет. Хаос возникает именно от войны, в том числе военного и изменения границ, и военного неизменения границ. Нужно чётко понимать, что при изменении границ нужно обсуждать проблемы разных системных уровней. Когда-то Россель, будучи губернатором Свердловской области, высказал те же аргументы Ельцину по поводу автономии для области, что Ельцин высказывал Горбачёву для автономии РСФСР от СССР. Абсолютно те же аргументы, но Росселю и его сторонникам автономизироваться никто не дал, а Ельцин у Горбачёва и СССР в точно такой же ситуации выиграл. Нужно понимать, что это всё попытки инженерной деятельности, по изменению устройства общества. Принципы, которые учитывают одноуровневую оптимизацию (границы государств, например, и только), а не многоуровневую (структуры уровня мельче государств и уровня крупнее государств) плохие, они не ведут к изменению жизни к лучшему, они не дают многоуровневой оптимизации. Они пытаются содействовать выигрышу какого-то одного уровня в межуровневых конфликтах. Но это невозможно, ибо эволюция стремится к нахождению многоуровневого оптимума, а не одноуровневого.

В случае границ государств как оформляющих границы разных общественных организованностей (обществ с разной одинаковостью его членов) нужно помнить о неустаканенностях/frustrations таких же, какие наблюдаются в спиновых стёклах[24].

В учебнике «Практическое системное мышление» это рассказывается подробнее, даются ссылки на литературу по эволюции (например, почему существует множество похожих по качеству решений оптимизационной проблемы, и легки переходы между состояниями системы, соответствующими этим решениям[25]). При этом остановить историю не получится, социальная инженерия непрерывна: «добиться достижения цели» нельзя, ибо сама цель непрерывно движется, и обстоятельства её достижения меняются, а ещё системы неэргодичны, у них есть память и нельзя просто «вернуть всё назад», поддерживать равновесие, гомеостаз. Развитие не гомеостатично, эволюция это не гомеостаз (хотя удерживание какой-то границы между отдельной системой и её окружением и можно с трудом назвать гомеостазом, но акцента на изменениях в подстройке к происходящему и убеганию от ожидаемых опасностей в будущем больше, чем акцента на неизменности, на которой центрируется гомеостаз).

Так что в инженерии можно действовать методом чисто эволюционным: проб и ошибок, и в социальной инженерии тоже так можно делать. Но ценой этого будет много смертей, и хорошо если это будут смерти фиктивных сущностей (скажем, если умрёт государство, то

[23] https://ru.wikipedia.org/wiki/Сецессия

[24] https://en.wikipedia.org/wiki/Geometrical_frustration – и обязательно смотрите там видео Frustrated magnetism in solids, ибо нужно понимать динамику происходящего, она лучше даётся в видео.

[25] https://www.pnas.org/doi/10.1073/pnas.1807890115

некоторым людям будет просто очень-очень обидно, но если умрёт множество людей вместе с государством, то это уже не «просто обидно», это невосполнимые потери). Если нет нормальной объяснительной контрфактуальной порождающей теории/модели, то ничего сделать в политике нельзя: любой пробегающий мимо специально обученный риторике как убедительной речи «пропагандист» даст более яркую картинку, и уведёт за собой толпу людей. Часть толпы, правда, останется. А ещё часть разбредётся. Без SoTA теории в политику как социальную инженерию идти нельзя, как и в любую инженерию. Нужны сущностные аргументы, а не просто «пропаганда» и «яркий образ». Дело тут даже не в строгости логики (можно обсуждать, какой логики: булевой, байесовской или квантовоподобной, этим занимается фундаментальная дисциплина рациональности[26]), но хоть какой-то объяснительной теории того, что происходит. Если такой теории нет, то мы имеем дело с безжалостной естественной эволюцией, она принесёт нам много нового интересного, но это будет долго и дорого, никакое лыко не будет в строку. При этом если есть труды типа тех же трудов Ванчурина-Кацнельсона-Вольфа-Кунина, то текущие политические теории о той же сецессии нужно признать прокритикованными (они объясняют меньше и хуже, чем новые теории). И честно вести другую политику.

Основная идея в том, что при многоуровневой организации общества (хоть сколь-нибудь устойчивые структуры разных уровней размера, составленные из людей и их вещей, включая землю и недра) неизбежны конфликты между целями разных уровней, а главная цель – «сохраниться, чтобы не съели и не умерли от голода, холода и болезней», и делать это с минимальными усилиями. Чистая физика-математика, а не идеалы из какого-то «исторического прошлого».

Есть множество субоптимальных решений, которые примерно одинаково неоптимальны, пока не появляется (редко!) какое-то крупное новшество, обычно приводящее к появлению нового системного уровня (типа одноклеточные стали многоклеточными, а многоклеточные образовали популяции с двумя полами). Это означает, что есть огромное количество разных конфигураций «стран» с самыми разными вариантами границ между ними и устройством жизни в этих странах, и эти варианты конфигураций деления мира на страны все будут примерно одинаковы в части успешности межуровневых конфликтов – хоть этих стран будет как сейчас 193, хоть 600 или даже 3500.

Ситуация может резко поменяться, если будут изобретены какие-то новые типы межстрановых объединений (так, Единая Европа, НАТО, ОДКБ тут примеры попыток социальной инженерии этого межстранового уровня, но можно думать и о других вариантах конструкции межстрановых объединений), новых типов стран (федеративное и конфедеративное государство когда-то было таким новым типом, тоже признающим, что оно состоит как целое из частей) и новых типов частей стран (областей, графств, штатов или как они там могут называться). Более того, никакое одно решение одного уровня не пройдёт, чтобы границы стран можно было переустраивать без смертей, нужны решения на всех уровнях сразу.

Конечно, президент «страны» плевать хотел на президента «графства», это ж понятно – он это просто по должности должен делать, защищать интерес этого уровня, «держать и не пущать», а также «строить империю, собирать земли». Но иногда графство/область вдруг начинает дружить «через голову» с другими президентами стран и даже страновыми объединениями, тогда страновое объединение уже выращивает свою бюрократию и плевать хотело на того президента, который против. Всё это непрерывно перемешивается и обладает памятью, как спиновые стёкла: возврата назад в какие-то прежние границы регионов и в теории не бывает, и в истории наблюдаем ровно это. Нужно крепко думать над словами тех, кто хочет стабилизировать неустроенности/неустаканенности/frustrations, обзывая их чистым хао-

[26] https://ailev.livejournal.com/1619025.html

сом, а не проявлением общих эволюционных законов. Если «заморозить границы по их сегодняшнему состоянию», то жди беды рано или поздно.

Какие могут быть инженерные решения? Демократия – это пока ещё относительно плохой и плохо проработанный, но всё-таки способ бескровной смены правителя. Без неё диктатора можно было бы менять только убийством диктатора и его сторонников. А при демократии при голосовании сразу видно, много ли сторонников, и поэтому никого убивать не надо, можно позволить всем жить. На уровне стран могла бы быть какая-то подобная процедура, и следы этой процедуры даже обговариваются, это идеи сецессионных или объединительных референдумов. Но дальше, как всегда, есть противники самих этих референдумов (и они же могут себя считать сторонниками демократии, не забываем, что у агентов разных системных уровней будут разные интересы).

По факту вся инженерия (как классическая, так и просто создание и изменение самых разных систем в части попыток изменить мир к лучшему) оказывается ускорением эволюции жизни как таковой. Инженерия тут просто увеличивает вероятность неумирания биологической жизни (как минимум, homo sapiens) при резких негативных изменениях в окружающей среде: люди выживают сегодня в ситуациях, где ещё сто лет назад точно бы умерли, их изделия инженеров и от болезней вылечивают, и от голода спасают, и от холода, и (после распространения кондиционирования воздуха) от жары.

В техноэволюции (и тем самым общей эволюции) активно участвуют не только инженеры классические (изобретатели архитектурных новаций в киберфизических системах), но и основатели/founders предприятий, которые строят предприятие на основе какой-то «предпринимательской гипотезы», ибо эта гипотеза как раз и есть предложение «умной мутации»: новый продукт, который сможет хорошо размножиться/реплицироваться в окружающей его среде. Предпринимательские гипотезы выдвигаются и маркетологами, и простыми людьми, которые строят свои действия на догадках о том, как изменить свои модели мира, модели себя, а также изменить мир и себя для того, чтобы избежать неприятных последствий. Некоторые из этих предпринимательских гипотез оказываются настолько интересными, что они реализуются хотя бы единожды, а некоторые настолько интересные, что их реализацию пробуют реплицировать/размножить. Так что вся инженерия оказывается частью «умной эволюции», а концепция использования (предпринимательская гипотеза), концепция системы, архитектурные решения оказываются «умными мутациями», причём важно отслеживать не только каждую такую мутацию, но и «умную эволюцию»/развитие: длинные цепочки во времени с опробыванием множества мутаций и получения быстрой обратной связи, и ещё как-то пытаться учитывать сложное взаимодействие этих мутаций друг с другом.

Если выйти на уровень социальной инженерии, то плохое понимание эволюционных процессов также ведёт к войнам в буквальном смысле слова, а это плохая инженерия.

Взгляд на системную инженерию как часть «умной эволюции» упирается в пока ещё недостаточное понимание современной науки, что такое системное мышление и откуда физически берётся эмерджентность и как она может быть описана традиционными для физики математическими средствами. Уже понятно, что:

• все описания эволюционирующего мира оказываются квантовыми в том смысле, что физические процессы излагаются как информационные (если можем что-то изменить при взаимодействии, то система генерирует ноль или единицу, квант/бит информации, это и есть «новая квантовость»[27] как информационный взгляд на физику, а не как особенности физики микромира с «квантовыми явлениями»).

[27] первый абзац в https://ailev.livejournal.com/1621262.html

• мы имеем дело с одними системами, которые являются моделями других систем, то есть их описаниями (ДНК – это описание организма, и даже мини-популяции, нужной для репликации фенотипа – мужчин и женщин). То есть работа с описаниями-репликаторами оказывается тесно связана с физикой как мы обычно это себе представляем (а физика теснее связана с информатикой, чем раньше казалось тем же физикам).

• в информационных системах количество неожиданно переходит в качество, то есть эмерджентность возникает при количественном росте вычислителя. С этим вообще малопонятно, что делать, просто дадим ссылки на литературу[28].

• Без обсуждения трёх масштабов времени (жизнь как «эксплуатация», мутации и разворачивание генотипа в фенотип как «разработка и изготовление») и результат множества мутаций (эволюция видов как «непрерывная инженерия», «развитие технических систем») никаких рассуждений проводить нельзя.

Чтобы разбираться дальше, потребуется хорошенько разобраться со всем интеллект-стеком, всем набором фундаментальных дисциплин: там и про эволюцию, и про этику, и про инженерию, и про «движущую силу эволюции» как те самые конфликты между системными уровнями, и про многое другое. Учитесь, а не то вас достанут или (генные или социальные) инженеры, или эволюция. И даже союзы в последней фразе нужно менять с «или» на «и» – и непонятно ещё, какой вариант будет хуже. Или лучше.

[28] Очень интересные исследования про порог эмерджентности при масштабировании в обучающихся/эволюционирующих системах, и помним, что нейросети, термодинамика, эволюция имеют общую теорию/математику (в курсе «Практическое системное мышление» про это рассказывается и даются ссылки на работы Ванчурина-Кацнельсона-Вольфа-Кунина): Emergent Abilities of Large Language Models, резкие пороги в работоспособности нейросетей в зависимости от числа вычислений, размеров и прочего масштабирования, https://arxiv.org/abs/2206.07682 гифка дерева возможных приложений у сети с масштабируемостью (пороги возможностей с ростом числа параметров сетки): https://ai.googleblog.com/2022/04/pathways-language-model-palm-scaling-to.html резкое появление самых разных свойств у случайных графов (порог эмерджентности при росте связности графа): https://www.quantamagazine.org/elegant-six-page-proof-reveals-the-emergence-of-random-structure-20220425/

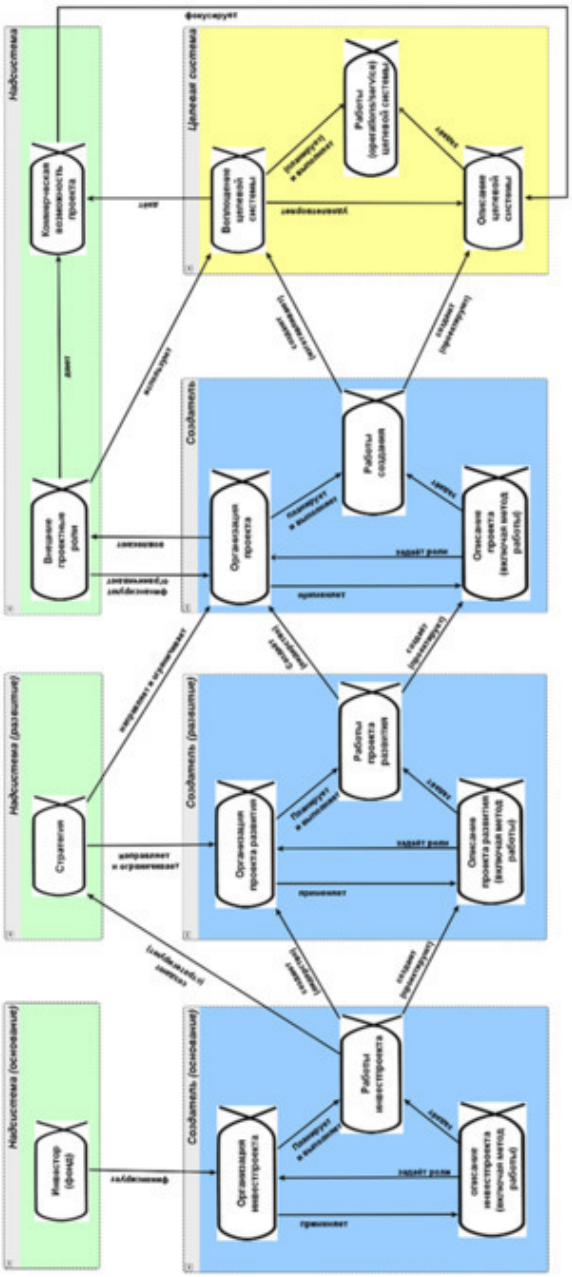
Специализация/конкретизация практик системной инженерии

Понимание системной инженерии можно разбить на три основных уровня прикладности/специализации/конкретизации (не путать с системными уровнями), каждый из которых можно далее разбивать на подуровни.

1. Уровень специализации безмасштабной системной инженерии

Этому уровню и посвящён наш курс/книга. Он приложим ко всем масштабам системноинженерного стека и описывает многоуровневую работу системного инженера. Если вы хотите проложить трамвайный маршрут по городу, вам придётся как инженеру учитывать и общественное мнение, и особенности городского планирования и налаживания пассажиропотока, и особенности электротранспорта и его технической инфраструктуры, и особенности вагонного парка, включая отслеживание трендов по беспилотному транспорту (включая особенности беспилотности рельсового транспорта, который не может «уклониться» от внезапно появившегося сбоку препятствия). Это типичная задача системного инженера (это роль, исполняется человеком или организацией из многих людей), который будет работать с самыми разными прикладными инженерами (роль, будет исполняться человеком или организацией из многих людей) для этих самых разных предметным областей. Системный инженер будет выстраивать проектную коммуникацию для согласований по всем этим вопросам.

На этом уровне вы будете отслеживать объекты внимания в проекте на основе обобщённой схемы альф инженерного проекта из OMG Essence так, как она показана в курсе методологии. Именно так в общем случае и организована деятельность по изменению мира, с учётом довольно длинных цепочек (на самом деле, конечно, деревьев, а не линейных цепочек) создания:



Система тут – просто «целевая система», без особой конкретизации, описание системы (включая «метод её работы», описание практики функционирования, хотя об этом довольно трудно говорить для «неживых» или «социальных» систем, но довольно просто для личностей и организаций), а описание проекта включает метод работы организации проекта. Даже такое общее представление о проекте изменения мира к лучшему позволяет точнее планировать (включая и agile «планирование на лету») и вести многомасштабный инженерный проект. Многомасштабный – это признание того факта, что вам не удастся охватить своим проектом

улучшение мира на многих масштабах, и придётся ограничиться несколькими масштабами из всех эволюционных. Но не одним!

Ещё эту диаграмму лучше бы считать

• или диаграммой общего жизненного цикла, надолго застревающей на состоянии воплощения системы «80% сделано» (и каждый день там будет что-то доделываться новое и что-то удаляться из уже не нужного, а ещё рекомендуемый норматив времени на работу с так называемым «техническим долгом»[29] – 20%, так что в любой момент времени оказывается, что готово только 80%, а ещё 20% нужно «отдать должок»). В жизни часто всё ещё менее однократно: представьте себе, что вы готовите круглосуточный шведский стол в курортном пансионате (не только подачу в этом стиле[30], но вместе с работой кухни, «под ключ». Ни в один момент времени вы не можете сказать, что закончили работу: что-то подъели, и нужно приготовить дополнительную порцию, что-то наоборот, испортилось несъеденное, и нужно убрать. А ещё нужно планировать сезонную смену меню, отвечать на жалобы тех, у кого болит живот (ни у кого не болит, а вот у этого одного заболел!) и кто нашёл жучка в тарелке (никто не нашёл, а вот этот один нашёл!) и следить, чтобы не уносили слишком много еды с собой. Это явно не описывается движением кейса «шведский стол» к приёмке-сдаче системы. Закрытие кейса – это будет как раз неуспех! Трудно представить, но операционная система Windows это примерно такой же «шведский стол». И к этой модели стремится огромное число других систем. Даже если вы делаете революцию в обществе, это тоже будет только «открыть двери накрытого шведского стола», а потом этот кейс нельзя будет считать закрытым, закрыть после этого двери – это неуспех проекта, аварийная ситуация.

• или диаграммой, описывающей кейс одной фичи, на каждом звене цепочки создания. Это много ближе к современному прочтению такой диаграммы. Кто-то что-то захотел изменить в системе, описал это, воплотил в жизнь, ввёл в эксплуатацию. Продолжаем и продолжаем. То есть в случае шведского стола это «давайте заменим крем в эклерах, будет и вкуснее, и дешевле, и готовить его быстрее». Lead time тут будет от появления этой идеи до момента, когда первый клиент откусит эклер с новым кремом, только в этот момент «ввели новую фичу в эксплуатацию».

Мы ещё много раз вернёмся к этой идее шведского стола как современной метафоре разработки. Уровень специализации безмасштабной системной инженерии как раз и нужен для того, чтобы вы понимали одинаковость инженерии шведского стола, инженерии ракет, инженерии корпоративного софта, генной инженерии, инженерии сообществ. Вот это всё нужно сделать, пополнить, проследить, чтобы не было просрочено, что-то доохладить, что-то держать горячим, а ещё нужно сделать так, чтобы всем желающим хватило, чтобы не нужно было долго ждать, и затем ещё быстро убрать объедки, не забывая при этом про улучшение раскладки и найм персонала. Это и есть современная инженерия, нацеленная на непрерывный ввод в эксплуатацию, continuous delivery:

[29] https://en.wikipedia.org/wiki/Technical_debt

[30] https://www.hospitality-school.com/buffet-meaning-table-setting/

2. Уровень специализации инженерии систем одного системного уровня.

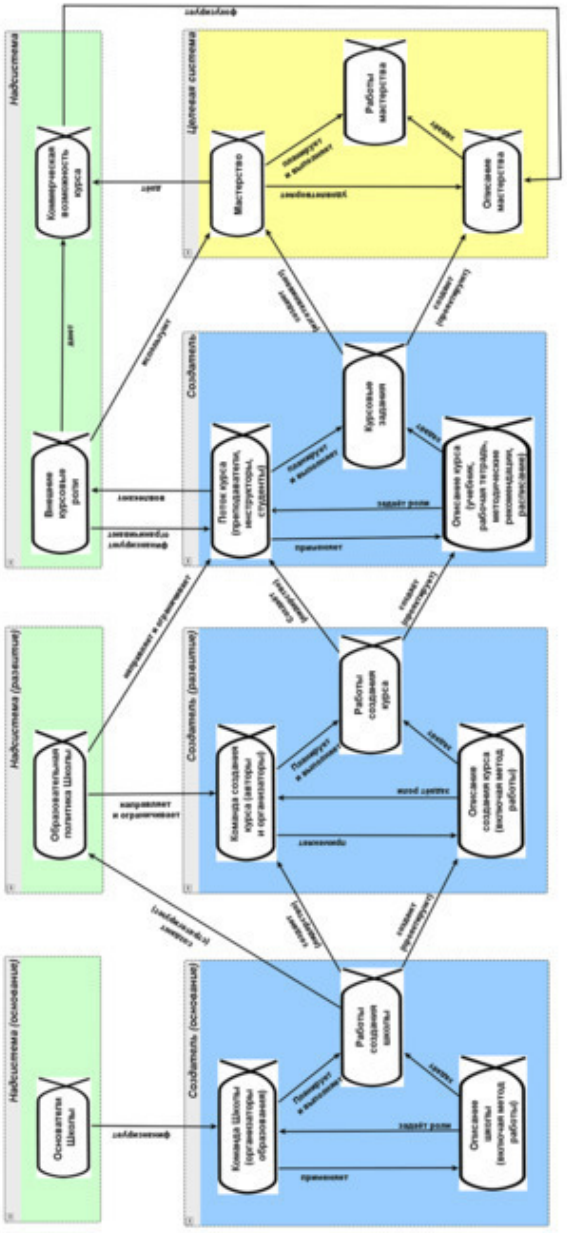
В многомасштабном проекте, тем не менее, на каждом системном уровне нужно каким-то образом вести свой локальный проект создания успешной системы, использующий практики системной инженерии с их особенностями для объектов данного уровня. Воплощение системы оказывается или веществом, или киберфизической системой (или её частью, например, компьютерным софтом), или личностью (или её частью, например, мастерством), или организованной группой людей (организацией или её частью, например, департаментом), и так далее.

Безмасштабная и непрерывная системная инженерия при этом уточняется и по факту становится **прикладной (к определённому масштабу систем, но всё-таки непрерывной) инженерией**. Так, классическая системная инженерия – это (прикладная) инженерия киберфизических систем, развивавшаяся главным образом на примере информационных систем (компьютеры и операционные системы), аэрокосмических систем (гражданская и военная авиация и ракетная техника) и других транспортных систем (автомобили, поезда, военный транспорт). Или программная/software инженерия, занимающаяся информационными системами как компьютерными программами (без включения аппаратуры в программно-аппаратном комплексе). Или инженерия предприятия, часто понимаемая как менеджмент (с операционным менеджментом как эксплуатационной инженерией).

Все эти отдельные виды (прикладной) инженерии имеют свою специфику, свои учебные курсы, но все следуют одному и тому же образцу/нормативу инженерного действия, взятому из безмасштабной системной инженерии. Хотя исторически было ровно наоборот: это безмасштабная системная инженерия была сформулирована главным образом на базе инженерии киберфизических систем, когда пришлось включать в эту инженерию людей, заниматься системноинженерным менеджментом и поэтому формулировать инженерную практику в наи-

более общем виде. Но потом эти общие безмасштабные идеи из *системной инженерии/systems engineering* пошли в самые разные прикладные *инженерии систем/engineering of systems*.

Скажем, если взять целевой системой мастерство как часть личности, то можно представить себе модификацию системной схемы проекта для проекта учебного курса:



В этой схеме уже не абстрактное «воплощение системы», а конкретизированное «мастерство», не абстрактная «организация проекта», а «поток курса», не просто «описание, включающее метод работы в проекте», а «описание курса, включающее методические рекомендации».

Этот проект создания мастерства может быть как частью какого-то многомасштабного проекта (например, вы хотите в проекте создания предприятия как проекте создания системы уровня организации научить людей работать по-новому, то есть внести в головы людей какое-то мастерство работы на новом типе станков или новым программным обеспечением, то есть хотите предпринять проект создания системы уровня части личности сотрудника). Системный инженер должен провести безмасштабное рассуждение, чтобы увязать эти проекты разных уровней. Понятно, что человек в роли такого системного инженера будет рассуждать на языке системной инженерии, общем для всех проектов создания чего бы то ни было.

Опять же, схема должна рассматриваться двумя путями:

• Создание целевого мастерства как чего-то конечного (при полном понимании, что конечным оно никогда не будет, всё время надо будет чему-то доучиваться, что-то переучивать, а что-то срочно забывать как «уже не SoTA»). В этой модели «готового мастерства» не бывает, но оно вполне себе может одновременно и разрабатываться (пишутся новые материалы курса), и изготовляться (будет идти обучение), и эксплуатироваться (обученный мастер будет вести ра

• Создание одного конкретного умения в рамках полного мастерства, одной «фичи» мастерства. Придумали чему-то конкретному научить – и научили, «ввели кусочек мастерства в эксплуатацию». И это введение в эксплуатацию разных новых и новых фич будет происходить долго-долго в ходе проекта, «непрерывный ввод в эксплуатацию».

Проект создания мастерства должен предусмотреть и более-менее автономное рассмотрение системы целевого уровня (а часто и не только целевого). И тут окажется, что для такого сорта систем как мастерство, даже если принять «инженерный подход к делу», а не «педагогический» или какой-нибудь менее определённый «культурных изменений» или чего чего-то более размытого в плане указания на цели и способы работы, есть много особенностей. Чтобы научиться такой «инженерии уровня личности», нужно освоить какую-то довольно специфическую практику инженерии (инженерия мастерства, которая даже называться будет «обучение», а не «инженерия»). Обучение, в свою очередь, как метод изготовления мастерства, будет разбито на множество разных практик. Это практики культуртрегерства как создание концепции использования мастерства в его «личностном окружении» (начало разработки – старт проекта, культуртрегер тут как product-manager), методологии и методики как разработки мастерства, преподавания как DevOps (изготовление и проверка частей, сборка из частей и проверка целого, введение в эксплуатацию – вынос в жизнь), мастерского выполнения деятельности как эксплуатация мастерства. И это не разовое прохождение жизненного цикла: нужно ещё и обновлять мастерство в уже выученных мастерах, как сейчас обновляют прошивки телефонов в давно изготовленных телефонах.

Эта адаптация методов системной инженерии для самых разных её приложений происходит для всех масштабов систем, всех видов систем одного масштаба. Концепция использования, а также use cases к театральной постановке, ледоколу, парламенту будут обязательно нужны (и это говорится в системной инженерии, мы особо отметим это в нашем курсе), но они и называться будут все по-разному, и разрабатываться по-разному (и это будет говориться в прикладных «инженерных» курсах. Слово «инженерный» тут в кавычках, чтобы напомнить: не все эти инженеры будут соглашаться на то, что они будут называться именно «инженерами». Но мы будем их называть именно так).

По каждой такой конкретизации/адаптации, в каждом приложении системной инженерии к какому-то виду целевых систем нужно владеть прикладной практикой инженерии, часто

называемой «инженерия таких-то систем» в случае уровней косного вещества и киберфизических систем (например, инженерия систем управления, control systems engineering, и помним, что systems engineering отличается от engineering of systems[31]) и уж совсем по-разному в случае более высоких системных уровней. Например, курс по инженерии мастерства (с неминуемым охватом не только ведения учебного процесса как создания мастерства, но и создания учебного курса со всеми его материалами как системы в цепочке создания мастерства) является именно конкретизацией/адаптацией курса системной инженерии. Пример такой конкретизации для курса инженерии нового мастерства как части общего мастерства уже образованных людей доступен в Школе системного менеджмента под названием «Мастерство обучать образованных»[32] (по названию инженерного мастерства, которому учит курс – «обучение образованных»). Конечно, эта конкретизация тоже не «законченный продукт», курс постоянно обновляется, «непрерывная инженерия».

3. Уровень специализации инженерии конкретной системы

Даже при том, что безмасштабная и непрерывная системная инженерия конкретизируется для какой-то прикладной инженерии систем определённого масштаба и вида (авиация, образование, медицина, фермерство, госполитика), для каждого конкретного проекта будет ещё одно уточнение для того конкретного типа системы, который будет создаваться и затем развиваться в единичном экземпляре или выпускаться серийно (включая особенности разных провайдеров сервиса). Так, легко представить себе, что конкретизации инженерии мастерства для мастерства системного фитнеса, мастерства игры на фортепиано, мастерства решать дифференциальные уравнения при помощи программ компьютерной алгебры и численных методов, мастерства безмасштабной системной инженерии (кругозорный курс по которой вы сейчас проходите) будут совсем разными для конкретных учебных проектов.

Скажем, игра на фортепиано – это мышечная практика и по факту она может быть надстроена над мастерством системного фитнеса, но не только – в это мастерство входит как минимум ритмика, которой нет в системном фитнесе и можно ещё обсуждать промежуточную практику музицирования, которая неразличима для вокала, игры на фортепиано и игры на скрипке, и учёт всех этих особенностей даст совсем разные практики игры на фортепиано, которые поразному могут преподаваться в музыкальных школах, но неизбежно приведут к более-менее похожим результатам. Помним про неустроенности и эволюцию: есть множество выживающих почти одинаковых вариантов оптимизаций для межуровневых конфликтов, а резкие улучшения довольно редки. Это означает, что есть множество самых разных вариантов «инженерии музыки на инструментальной базе фортепиано», они дают все примерно одни и те же результаты, плохие варианты рыночно вымрут, а улучшения будут, но крайне редки. Скажем, улучшение типа игры на клавишном синтезаторе, или магнитофон, а затем и плеер, который позволяет не играть пьесу каждый раз, а просто воспроизводить записанное. Эволюция практик происходит постоянно, а практики – это и есть деятельности/виды труда, и они же – виды инженерии. И если какая-то команда занялась инженерией такого вида систем, то она будет заниматься этим видом систем непрерывно, подстраивая варианты целевой системы под изменения окружения и используя всё новые появляющиеся альтернативы по концепции системы как способы реализации функций какими-то интересными новыми конструктивными элементами.

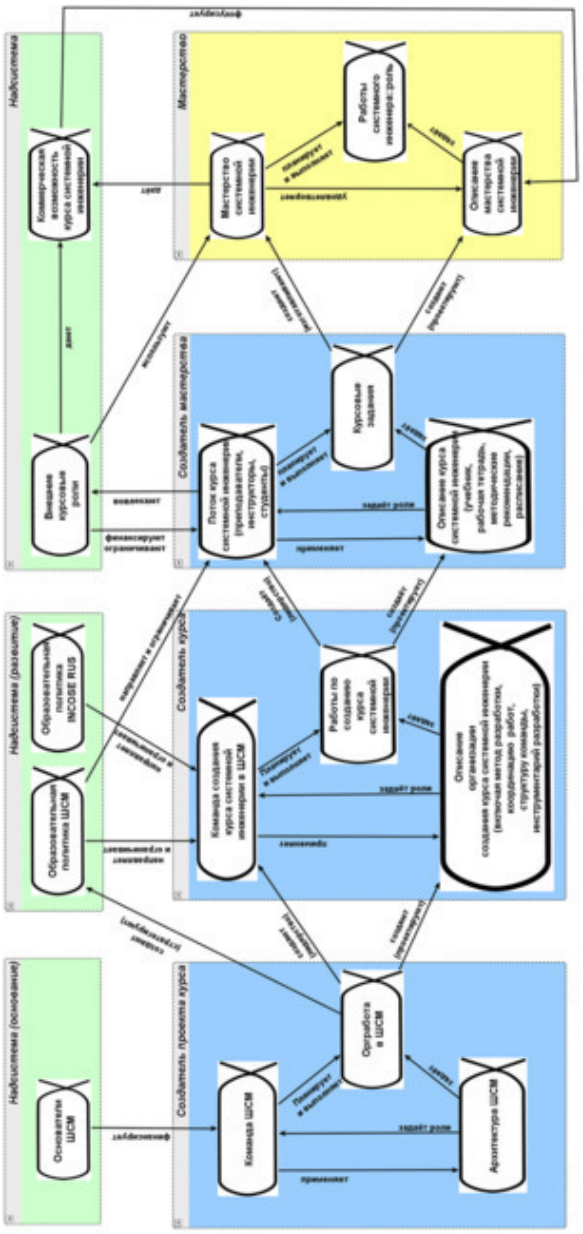
На этом уровне специализации инженерии конкретной системы можно использовать или общую схему проекта и альфы с их подальфами, адаптированную для уровня мета-модели (предметная область какой-то инженерии «как в учебнике») с предыдущего уровня, или всё-

[31] If there is an issue between SE and EoS, it may be semantic, rather than grammatical. Classically, the first word moderates the second: so, in SE, 'systems' appears to moderate (describe the type or flavour of) engineering; and in EoS, 'engineering' appears to moderate (describe the type and flavour of) systems. – из prof. Dereck Hitchins, «SYSTEMS ENGINEERING VS. ENGINEERING OF SYSTEMS – SEMANTICS?», https://systems.hitchins.net/profs-blog/systems-engineering-vs.html

[32] https://system-school.ru/teaching

таки конкретизировать эту общую схему проекта для создания конкретного вида киберфизической системы (авиалайнера конкретной модели или игрушечного самолёта определённой модели), создания конкретного предприятия (конкретного стартапа на пять человек или конкретного агропромышленного холдинга), создания конкретного мастерства у конкретных людей (мастерства игры на фортепиано или мастерства безмасштабной системной инженерии, и ещё тут разница – на уровне хобби/кругозора, или профессиональном уровне). Все эти уточнения потребуют адаптации системной схемы проекта, использования самых разных практик работы в рамках одного конкретного метода создания конкретной целевой системы.

Скажем, для создания текущего курса «Системная инженерия» можно использовать конкретизированную схему «образовательного проекта»:: «проекта создания»/«инженерного проекта» (и тут даже приведены названия конкретных организаций, которые связаны с проектом, ШСМ и INCOSE RUS, при этом в области интересов надсистемы/окружения могут находиться альфы, как-то связанные с разными системами в окружении, например альфы и связанные с ШСМ, и связанные с INCOSE RUS):



Эта схема также может читаться двумя способами:

• создание и непрерывное ведение курса (иногда такое называют «вечная бета») в целом, со всеми его версиями

• создание и введение в эксплуатацию какого-то одного **инкремента** курса в порядке его развития. Тогда при желании можно выполнить ещё такт адаптации, дав название этому инкременту. Помним, что диаграммы в части их модификации – это зло, поэтому такие схемы в жизни будут существовать в табличном виде, и каждый инкремент легко может получить своё

название. С диаграммами такое не пройдёт, если вы выпускаете по паре инкрементов в день, просто сил на перерисовку не хватит!

Если вы будете создавать «систему найма сотрудников»:: «проект оргразвития», то это третий уровень конкретизации, конкретный проект в организации, ситуационная мета-модель уровня стандарта предприятия и модель конкретной проектной ситуации с учётом экземпляров (операционным учётом). На втором уровне конкретизации (культурная мета-модель) это будет предметная область на каком-то достаточно абстрактном уровне учебников менеджмента как прикладной дисциплины инженерии предприятия, то есть «проект оргразвития»:: «проект создания»/«инженерный проект». А сам «проект создания»/«инженерный проект» – это самый общий уровень, наименее конкретный, безмасштабный/фундаментальный, то есть создание чего угодно, уровень рассмотрения нашего курса системной инженерии, уровень мета-мета-модели. И не путайте системные уровни (отношение часть-целое) и уровни абстракции (отношение принадлежности к классу, конкретизация как противоположная направленность абстрагирования как раз тут).

Такой подход многоуровневой конкретизации инженерного метода решает и традиционные проблемы классического подхода к инженерии как отдельной от менеджмента. В классической системной инженерии как инженерии сложных киберфизических систем обычно вынуждены описывать менеджмент проектов создания самолётов и подводных лодок как «системноинженерный менеджмент». Если понимать, что речь идёт об инженерии организации, которая будет заниматься созданием киберфизической системы, то это просто описание двух инженерных проектов в цепочке создания: создание киберфизической системы (самолёта или подводной лодки – концепция использования, use cases для них, архитектурные решения, и т.д.) и создания организации проекта (стратегия этой организации как концепция использования и use cases организации/команде/коллективу/предприятия/«производственной кооперации» проекта, решения по концепции системы, архитектурные решения для предприятия как архитектурные решения по организации проекта, и т.д.). Эти проекты существенно связаны в части архитектуры целевой системы и организации (через закон Конвея и обратный манёвр Конвея), и эту связь реализуют архитекторы. Архитекторы целевой системы и организации согласуют их архитектуры и меняют их по мере развития целевой системы.

А чему учить тогда системных инженеров, например, классических киберфизических систем? Учить нужно так же многоуровнево:

• общей безмасштабной и непрерывной системной инженерии, это и есть предмет нашего курса/книги

• затем инженерии киберфизических систем (кругозорный уровень предметной области для какого-то эволюционного уровня),

• затем конкретизации этой инженерии для самолётов (конкретная предметная область, и это будет весьма разная инженерия для одномоторного одноместного самолёта и авиалайнера на три сотни пассажиров – domain), и доводят это до узкой практики, которая используется в конкретном проекте (и тут говорят про ограниченную предметную область, workflow/practice как bounded domain).

• и обязательно менеджменту, то есть организационной инженерии с конкретизацией для инженерных организаций в авиации.

При этом есть всегда надежда, что при знании общего подхода к инженерии (например, хорошего понимания дисциплины безмасштабной системной инженерии) обучение каким-то особенностям для какого-то класса систем не только займёт меньше времени, но и пригодится, когда будут происходить события техноэволюции. А они будут происходить! Скажем, когда

самолёты вдруг будут заменяться или электросамолётами для ближних рейсов, или многоразовыми ракетами для дальних рейсов: основное знание «как создать киберфизическую систему» останется, но придётся поменять представления о том, что там нужно будет делать конкретно для изменившегося вида систем. А когда пойдут организационные изменения (а они обязательно будут!), то у инженера-авиастроителя будет понимание происходящего, хотя бы в общих чертах.

Ответственность системных инженеров за целокупность систем. Фундаментальность/трансдисциплинарность системной инженерии

Ответственность за всю систему на многих уровнях как целое (whole system) и связанная с этим фундаментальность/трансдисциплинарность/transdisciplinarity подхода к другим инженериям (механической, электрической, программной, предприятия и т.д.) отличают системную инженерию от всех её частных/предметных/domain конкретизаций. Представим себе ледовую буровую платформу:

Сотни тысяч тонн металла, бетона, пластмассы, компьютеров (самих по себе сложно устроенных) и оптоволокна, необходимых расходных материалов, обученная вахта должны собраться вместе далеко в море среди льда. В строго определённый момент эта огромная конструкция должна начать согласованно работать – и не просто работать, а приносить прибыль и обеспечивать безопасность в части загрязнения окружающей среды и здоровья находящейся на платформе вахты, а также в чётком согласовании со службами берегового обеспечения (нефтепереработка, провайдеры связи, метеорологические службы, государственные надзоры по самым разным линиям и т.д.).

Какая инженерная/деятельностная/практическая/трудовая дисциплина должна учесть результаты работ всех других инженерных дисциплин, работающих на самых разных системных уровнях – собрать в единое целое данные ледовой обстановки, санитарных норм в помещениях для обслуживающего персонала, обеспечение электричеством попавших туда компьютерных серверов, необходимые характеристики этих серверов и программное обеспечение с нужными функциями и нужной надёжностью? Кто озаботится учётом в конструкции платформы изменений в длине металлоконструкций за счёт разницы суточных температур и одно-

временно установкой акустических датчиков на трубах, которые прослушивают шорох песка, чтобы по этому шороху нейросетевые алгоритмы определяли износ труб и предлагали редкий и нужный «ремонт по состоянию» вместо относительно частого и бесполезного «планового/профилактического ремонта»?

Системная инженерия как раз и является той дисциплиной, которая ответственна за обеспечение целостности в инженерном проекте: именно системные инженеры в их разных ролях проектируют нефтяную платформу как изменяющее мир к лучшему или хотя бы просто успешное (безопасное, надёжное, прибыльное, ремонтопригодное и т. д. – разным внешним проектным ролям нужно от этой платформы разное) целое, потом раздают части работы «прикладным инженерам»/«инженерам по специальностям» (инженерам-строителям, машиностроителям, инженерам-электрикам, компьютерщикам/айтишникам и т.д.), а затем собирают результаты их работ так, чтобы получить работоспособную и надёжную систему. Агенты, исполняющие роли прикладных инженеров самых разных специализаций по факту владеют на каком-то (кругозорном) уровне и системной инженерией, чтобы понимать происходящее в проекте как в части его целевой системы, так и в части организации проектной работы.

Это главный признак, отличающий **системных** инженеров «по роли» от всех других **прикладных** инженеров: они отвечают за целевую систему в целом, как в части деталей-частей (разные системные уровни требуют разных практик для работы с типовыми для них системами – с микросхемами работают не так, как с компьютерами, а с компьютерами не так, как с системами управления, в состав которых входят компьютеры), так и в части использования детальных знаний отдельных инженерных практик (разные свойства систем какого-то типа требуют знания разных практик: если вам нужно сделать корпус компьютера, то нужны и знания инженера-механика, чтобы корпус был прочным и лёгким, и знания инженера-теплотехника, чтобы компьютер не перегревался и корпус обеспечивал достаточную вентиляцию). Главная задача системного инженера – чтобы не было пропущено какой-нибудь мелочи, ведущей к провалу.

Авиалайнер – это много-много кусков металла и пластика, синхронно летящих вместе с людскими телами и чемоданами на скорости 900км/час (0.85 от скорости звука, это типовая скорость Boeing 787 Dreamliner) на высоте 11км. Системный инженер в его прикладной специализации «авиационный системный инженер» – это тот::роль, кто придумал, как обеспечить их надёжный и экономичный совместный полёт, увязав самые разные интересы самых разных ролей (грузоподъёмность, расход топлива, дальность полёта, шум при взлёте и посадке, ограничения по длине разбега и посадки, необходимость лёгкого обслуживания на земле, отсутствие обледенения, безопасность людей на борту, и т. д. и т.п.), при этом эти интересы отстаивались самыми разными исполнителями внешних проектных ролей, представлявшими самые разные профессиональные и общественные группы исполнителей тех или иных практик (экологи, военные, урбанисты, коммерсанты, и т.д., и т.п.). Для авиалайнера примерно шесть миллионов деталей замысливаются, проектируются, именуются, изготавливаются, проверяются, доставляются в одно место, собираются в одно изделие, принимаются как целый авиалайнер и продаются, в аэропорту эти собранные детали как одно целое обслуживаются, заправляются топливом и едой, заполняются пассажирами и грузом, диспетчируются на взлёт – и вот эти шесть миллионов деталей летят, обеспечивая комфорт пассажирам и прибыль владельцам.

Но это даже не вся история: авиазавод тоже делает и дальше развивает системный инженер! Элон Маск не боится компаний, которые делают прототипы электромобилей. Даже для ракет создать серийное производство ракет вдесятеро трудней, чем создать саму ракету. Для автомобилей создать производство автомобилей в сто раз трудней, чем создать автомобиль[33].

[33] «It's relatively easy to make a prototype but extremely difficult to mass manufacture a vehicle reliably at scale. Even for rocket science, it's probably a factor of 10 harder to design a manufacturing system for a rocket than to design the rocket. For cars it's maybe

Это тоже делают (системные) инженеры::роль. Для любого производства киберфизической системы нужно учесть потребности внешних проектных ролей, разработать концепцию использования и концепцию системы, принять архитектурные решения, создать информационную модель с детальностью, достаточной для изготовления, далее выполнить все закупки комплектующих, построить завод, запустить/наладить его оборудование, нанять и обучить людей, потом управлять тем, чтобы этот завод работал «как машина» (несмотря на то, что работают на нём не только машины, но и люди). Это всё требует многоуровневого (системного) учёта мельчайших деталей нижних уровней.

Если где на конвейере завода неудачно выпадет винтик из-за того, что что-то там было плохо спроектировано, потом плохо смонтировано, поломка плохо диагностирована, может неожиданно остановиться весь завод. Ну, или работники завода могут забывать вести учёт изменений оборудования этого завода – и будет непонятно уже, какое там стоит оборудование, и как его ремонтировать. Это кажется не таким важным для автомобильного завода, но в США как-то была закрыта атомная станция из-за того, что состав оборудования «в жизни» через несколько лет работы стал существенно отличаться от состава оборудования «по документам»: никто не давал гарантии, что будет понятно, как быстро отреагировать на проблемную аварийную ситуацию, разобраться «по чертежам» уже стало невозможно, эксплуатацию сочли далее очень опасной, атомную станцию закрыли. Это проблема в организации труда инженерной бригады, и в конечном итоге это тоже проблема инженерии, но инженерии предприятия, называемой обычно «менеджментом» («организационным менеджментом» в части создания и модификации/модернизации организации, «операционным менеджментом» в части эксплуатации созданной организации).

Всё то же самое относится к «нежелезным» системам. Если вы работаете с генной инженерией, то используете вирусы. Это означает, что вам нужно соответствующим образом простроить работу исследовательской лаборатории, работу производственных мощностей, организовать работу с персоналом и ещё отвечать на многочисленные вопросы по биоэтике, а также делать это достаточно долго, всё не закончится первым же продуктом, будь он удачным или не очень удачным. То есть работать нужно отнюдь не только с самими вирусами! Это всё проблемы «системного инженера»:: роль (роль, а не «должность»!). Такую роль могут выполнять десятки людей в большом проекте. Системный инженер держит все (и в окружении, и в цепочках создания) системы в проекте взаимоувязанными, ибо именно у него (или у них, если проект большой) в голове и компьютерах (основной инструмент – моделер для системного моделирования) практика раскладки всей сложной производственной системы в её окружении на уровни, а также раскладки рассмотрения разных интересов в рамках одного уровня на разные практики, многоуровневой оптимизации решений для удовлетворения всех интересов (конфликтов/противоречий между решениями разных уровней избежать нельзя, нужно проявлять изобретательность, и всё равно в результате будет неустроенность/неустаканенность в конструкции системы, будет проявляться субоптимальность в решении этих конфликтов). В помощь системным инженерам тут компьютеры, позволяющие удерживать всю эту сложность в усиленной компьютером памяти, проводить имитационное моделирование для экономии времени и материалов по сравнению с пробами и ошибками в реальной жизни. Компьютеры помогают удерживать коллективную собранность: проявлять коллективное внимание к каждой мелочи нижних уровней разбиения систем, не теряя коллективного внимания ко всему огромному целому на верхних уровнях, глубоко уходящих в окружение, и это коллективное внимание будет удерживаться компьютерами также и на этих верхних, надсистемных уров-

---

100 times harder to design the manufacturing system than the car itself. The issue is not about coming up with a car design – it's absolutely about the production system,» Musk said. «You want to have a good product to build, but that's basically the easy part. The factory is the hard part.» – https://www.businessinsider.com/elon-musk-says-building-factory-100-times-harder-than-making-car-2019-3

нях окружения, а также по необходимости на всех уровнях систем и их окружения по цепочке создания. Ответственный за организацию коллективного многоуровневого внимания к самым разным системам как раз системный инженер. «Системный» тут как раз отсылает к многоуровневости внимания, в отличие от внимания прикладных инженеров к одной предметной области какого-то одного системного уровня для систем какого-то одного вида.

Роль (системного) инженера

Все рассуждения про роли и агентов/IPU как актёров/деятелей/практиков:: «исполнителей ролей» полностью применимы к системным инженерам как деятелям/практикам, занимающимся практиками системной инженерии. Причём для системных инженеров как «деятелей/практиков вообще» будет два вида конкретизации, если нужно разбираться с ролями в жизни:

• **Конкретизация по линии масштабов и видов систем в каждом масштабе.** *Системный инженер* обычно занимается несколькими масштабами системы/продукта в её окружении, а *прикладные инженеры* по конкретным системам занимаются каким-то одним видом системы, одной предметной областью. То есть системный инженер – инженер-робототехник (скажем, должность «ведущий робототехник»), врач (инженер человеческого организма, чаще всего на стадии жизненного цикла «ремонт» и в случае паталогоанатома – «вывод из эксплуатации»), инженер предприятия (скажем, на должности «директор по развитию» или «менеджер проекта»), инженер общества (скажем, должность «депутат Госдумы»). Эта конкретизация требует, конечно, обучения особенностям каких-то конкретных систем, в приведённых примерах нужно хорошо разбираться с дисциплинами, объясняющими работу роботов, предприятий, общества. Если ограничиваться только общими знаниями по безмасштабной системной инженерии (например, знанием текущего курса) и ничего не знать про особенности целевой системы, то будет «хотели как лучше, а получили как всегда». Уже давно нет «врачей вообще», ровно так же как и «инженеров вообще» (хотя сто лет назад были и просто врачи, и просто инженеры) а есть врачи по отдельным подсистемам человеческого организма: отоларинголог, уролог, стоматолог, клинический психолог или даже психиатр. Ещё есть варианты тоже по линии изменения состояния целевой системы (педиатры, патологоанатомы), или состава целевой системы (семейный врач). Вот это деление врачей по их целевым системам, как ни странно, ровно такое же, как и инженеров по их видам целевых систем (подсистем: например, специалисты по фюзеляжам самолёта или инженеры-двигателисты в авиапромышленности). Врачи – это тоже инженеры, они тоже меняют физический мир к лучшему!

• **Конкретизация по линии разных видов инженерных практик для одного и того же сорта систем** – и вот тут системные инженеры занимаются не столько разработкой (это делают прикладные инженеры-разработчики, часто разрабатывая отдельные модули системы, ответственные за реализацию отдельных функций, максимально независмо друг от друга), сколько архитектурой, которая позволяет трудам прикладных инженеров-разработчиков интегрироваться в надёжную, масштабируемую, производительную, легкодоступную, выдерживающую случайные отказы (это мы всё перечисляем разные предметы интереса архитектора, об этом будет отдельная глава нашего курса), также многочисленными практиками непрерывного ввода в эксплуатацию, начиная с управления конфигурацией и заканчивая организацией разворачивания результатов работы команд разработчиков и обеспечения доступа пользователей к системе (transfer, ввод в экслупатацию). Этих системных инженеров называют DevOps, о них тоже будет большая глава.

• У врачей тоже есть подобное разделение. Например, по типам практик вмешательства (стадия «изготовление»): терапевт (неинвазивные вмешательства и сборка разных других видов вмешательств в работающее целое, «ввод в эксплуатацию»), хирург (инвазивные вмешательства), врач функциональной диагностики (нет вмешательства, только диагностика как часть проектирования будущего вмешательства). И тут обычно обязательно будет добавляться вид системы, ибо вид вмешательства/изготовления существенно может отличаться в зависимости от вида целевой системы. Без того, с какой системой работаешь, о практике мало что можно сказать, хотя и тут можно вывести за скобки немало. Так, хирург лет двести назад

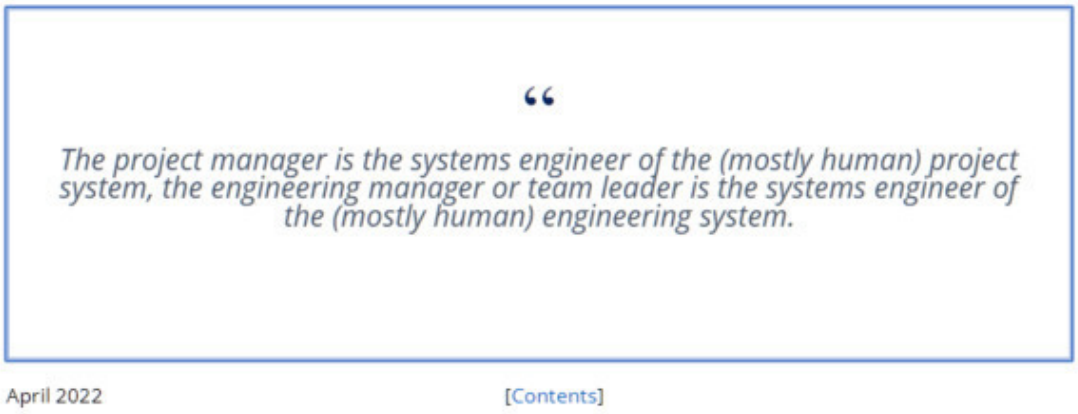
ещё не был ярко выраженной подролью врача, это была просто одна из врачебных практик, лет пятьдесят назад хирургия была уже отдельной практикой, а сейчас практически независимо практикуется нейрохирургия и кардиохирургия, да и в целом уже не встретишь «хирурга общего профиля», кроме самых захудалых сельских больниц. Конечно, на какой-нибудь подводной лодке можно встретить даже «врача» по всем вопросам, а в деревне «фельдшера» (даже не совсем врача, но тоже по всем вопросам), но это уже не типичная ситуация, не отражает «лучших известных на сегодня практик»/state-of-the-art (SoTA). В инженерных проектах по созданию киберфизических систем можно встретить инженера-электротехника, инженера-теплотехника, инженера-электронщика, но в маленьком стартапе на трёх человек можно встретить и просто «инженера», а исполнитель этой роли ещё будет прихватывать и другие роли – продавца, проектного менеджера и даже уборщика помещения. Так и в системной инженерии бывает «просто системный инженер», а бывает системный архитектор или девопс (по-старинке иногда называемый системным администратором, хотя он давно уже ничего не администрирует, а создаёт «автоматического администратора»).

Всё разнообразие ролей, которые получаются конкретизацией по обеим линиям (масштабов и видов систем, а также видов инженерных практик) будет выполняться какими-то агентами (обычно людьми и их организациями), имеющими должности или составляющими какие-то оргзвенья. Не нужно думать, что системный инженер будет называться именно так. Он может быть на должности «ведущий инженер», может называться «завуч» в случае образовательных учреждений, может быть «координатором», смотреть нужно на содержание его деятельности/практики. Если это практика, описанная в нашем учебнике как архитектурная или DevOps, то она идёт с целой системой, так что это системный инженер.

Чтобы сочетать самые разные инженерные роли, делают гибридные должности типа «продакт-менеджер» или сокращённо «продакт», который с одной стороны ответственный за реализацию бизнес-модели проекта (грубо говоря, «чтобы проект деньги зарабатывал, а не проедал»), а с другой стороны озабочен связью с инженерами, которые должны реализовать его идеи по поводу концепции использования системы. Поскольку концепция использования системы связана с целой системой, то часть деятельности продакт-менеджера – это системная инженерия (про него мы ещё упомянем, когда будем говорить про концепцию использования). Но часто он и инженер предприятия (ибо именно он должен организовать разработчиков, выполняет практики организационного менеджмента, собирает команду) или даже операционный менеджер (например, разработчиков организовал не он, а какой-нибудь TeamLead, но именно он определяет загрузку разработчиков, так что он «инженер по эксплуатации группы разработчиков»). Всё это вариации ролей системного инженера, поэтому не так страшно их объединение в одном лице. И помним, что по поводу всех этих терминов никакого согласия нет, в разных организациях и разных предметных областях они означают совершенно разное (TeamLead в веб-разработке и где-нибудь в лесотехническом машиностроении может означать очень разные обязанности, ещё и существенно зависящие от страны и организации, а также от года, в котором ведётся обсуждение: в прикладных видах инженерии, включая инженерию предприятия всё стремительно меняется, включая «модные» и «локально правильные» названия типовых должностей).

Позиция единства инженерного взгляда как минимум на инженерию и менеджмент уже давно признаётся системными инженерами. Вот в журнале «Project Performance Engineering Systems Engineering News» в апреле 2022 это даётся на седьмой странице обведённым в рамочку[34], чтобы системные инженеры об этом не забывали:

[34] https://issuu.com/ppisyen/docs/ppi_syen_111_april_2022/29



Человек на должности инженера может сутки напролёт играть роль менеджера, планируя работы, и наоборот, человек на должности менеджера может сутки напролёт разбираться в инженерных задачах. Элон Маск сообщает о себе, что 70% времени занят решением инженерных задач «по железу», то есть играет роль инженера киберфизических систем, и даже необязательно системного инженера, но и просто инженера-разработчика, который что-то изобретает и принимает трудные проектные решения по концепции системы. Элон Маск – это агент/актёр, и оказывается, что роль предпринимателя и менеджера (он ведь CEO и в Tesla, и в SpaceX) он исполняет меньше времени, чем роль инженера «по железу»! Да и эти 30% «менеджмента и предпринимательства» оказываются инженерией предприятий, то есть инженерией всевозможных цепочек создания.

Если вы понимаете, как устроена инженерия на самом высоком уровне абстракции (наш курс как раз про это), вам будет легче разобраться в неизбежном разнообразии видов деятельности по созданию самых разных видов целевых систем и их частей в сложно устроенном окружении, в длинных и запутанных цепочках создания этих систем. Оказывается, все виды деятельности по созданию чего бы то ни было, если брать их в более-менее субоптимальном варианте (SoTA, а не любом варианте!), выдержавшем эволюционный отбор, похожи друг на друга, это просто варианты безмасштабной непрерывной системной инженерии, которую вы проходите в нашем курсе. Те же объекты внимания, те же операции с ними, хотя и по-разному называемые и данные в некотором разнообразии. Разные виды одного рода!

Вовсе не факт, что вы в жизни будете встречаться с классическими «железными» проектами инженерных систем или даже не такими классическими проектами программной инженерии. Если вам встретится какой-нибудь детский садик Монтессори или организация выборов в какой-то профессиональной ассоциации или даже муниципалитете, вы сможете разобраться быстрее, если будете понимать, что это проекты по изменению мира, то есть инженерные проекты. Практики, которые вы будете выполнять в этих «неинженерных проектах» – это таки инженерные практики, а не какие-то особые экзотические практики, которые дано понять только посвящённым людям. Даже если будет казаться, что проект состоит только из практик по описанию мира (реверс-инженерия уже существующей в мире системы в её окружении, или даже будущей системы, которую только потом нужно изготовить), а не изменению, это будет просто часть какого-то проекта по изменению мира, то есть часть какого-то инженерного проекта. В любом случае, вы сможете задать правильные вопросы к проекту, разобраться в происходящем. **Все сегодняшние проекты по изменению мира к лучшему различаются и по их форме, и по их содержанию, но одновременно и все проекты оказываются в чём-то одинаковыми: они являются разными вариантами системноинженерных проектов.**

Если оказалось, что речь идёт не о проектах по изменению мира к лучшему (то есть не об инженерных работах), то такие работы выполнять не нужно. Скажем, инженерная ли работа, в которой я поднимаю штангу по вечерам? Да, это проект инженерии себя. Можно пообсуждать, как распределены инженерные роли между мной, врачом, тренером, диетологом и внешними проектными ролями – включая внешние проектные роли тех, кто воспользуется результатами этого инженерного проекта, в число этих людей можете попасть не только вы, но и члены вашей семьи, а если ваша работа связана с физической культурой, то и рабочее окружение. Если вы художник, или политик, или музыкант, то тоже полезно задумываться, что вы меняете в окружающем мире, и происходят ли эти изменения к лучшему. Никакие названия должностей, ролей (и связанные обычно с названиями ролей названия профессий и квалификации), названия практик и «исторически сложившиеся традиции» не должны препятствовать такому инженерному взгляду на мир.

Конечно, в этом разделе мы кратко описываем ещё и процесс непрерывного углубления разделения труда, ибо появляются всё новые и новые способы изменить мир к лучшему – инженеры придумывают всё новые и новые способы удовлетворять интересы, которых раньше и не было. Так, у неандертальцев не было интереса к мастерству системной инженерии, да и самой системной инженерии ещё не было, а затем появилась «просто инженерия», потом в середине прошлого века появилась системная инженерия, в конце 20 и начале 21 века она сосредоточилась на киберфизических системах, а в последний десяток лет оказалось, что речь идёт об огромном разнообразии прикладных инженерных практик для самых разных масштабов и видов систем. Развилась методология, и стало очевидно, что все эти практики изменения мира чем-то похожи (как минимум, все они работают с какими-то системами), но в чём-то и радикально непохожи. Это происходит и у «железных» инженеров, и у программных инженеров, и у врачей, и у деятелей искусства. Достаточно посмотреть на титры какого-то кинофильма, чтобы оценить масштаб разделения труда в киноискусстве. Конечно, при создании «железных» систем разделение труда не меньше, просто не принято так подробно выписывать участников инженерной работы, как это принято в кинопроизводстве.

Так что число инженерных практик и связанных с ними инженерных ролей будет продолжать непрерывно расти, но можно смело опустить слово «инженерных»: число практик (инженерных, то есть изменения мира) и связанных с ними ролей будет продолжать расти. Тем более важно понимать, что в их основе лежит системная инженерия, все эти практики являются или модификациями системной инженерии для какого-то сорта систем (скажем, селекция и производство коров как для производства молока, так и для производства мяса в политической оппозиции веганам – и в этой части это будет социальная инженерия), или частями общей практики системной инженерии (скажем, той частью, которая занимается архитектурой, для каких бы систем эти архитектурные решения ни принимались и как бы ни назывались).

Системный инженер как технический лидер

Системные инженеры-генералисты в отличие от инженеров по более узким инженерным специальностям должны технически направлять инженерные коллективы, так прописано в документах системных инженеров для киберфизических систем. Врачи-терапевты как системные инженеры уровня существа (а с учётом клинической психологии и уровня личности) должны быть лидерами во врачебных коллективах с врачами более узких специальностей. Они поделят врачебный труд (направят к хирургу и на ЛФК), обеспечат стык этих практик, оттестируют и выпишут из клиники, а потом ещё и будут «вести» несколько лет всю семью (непрерывная инженерия).

Политики «по всем вопросам» должны быть лидерами в политических коллективах (партиях), где могут быть политики с более узкой специализацией – это всё одно и то же.

Понятие **технического лидерства** (technical leadership, отличается от **организационного лидерства**, organizational leadership) означает помощь в организации коллективной мыслительной работы по отношению к той или иной технической идее: все участники проекта должны делать одну и ту же систему, а не разные. Отличия **технических лидеров** от «технических евангелистов»: евангелисты – это проповедники чужих идей на предмет их воплощения в самых разных проектах инженерной компании или даже отрасли, а системные инженеры как технические лидеры – сами себе «технические иисусы христы» в отдельных конкретных проектах, они сами поставщики тех технических решений, в которых потом они должны уметь убедить других людей в их конкретном проекте. Системные инженеры не просто берут идеи от одних инженеров, а потом убеждают других инженеров их принять.

Системные инженеры генерируют технические идеи самостоятельно, и чаще всего это архитектурные идеи и идеи организации непрерывного выпуска. Системные инженеры в отношении надзора и менторства по отношению к инженерам-разработчикам. Они принимают решения, разъясняют их инженерам-разработчикам, проверяют выполнение этих решений, менторят инженеров-разработчиков, если тем трудно выполнять решения.

Организационные лидеры занимаются тем, что помогают людям занять свои роли в организации, они катализируют сотрудничество, делают организацию целой. Технические лидеры делают целевую систему целостной. Какую целевую систему? Разных уровней. Политические лидеры как технические лидеры уровня общества должны делать общество целостным, чтобы оно функционировало без каких-то существенных коллизий.

В давней (это статья 2009 года. Помним, что первый iPhone появился в 2007 году. В инженерии 2009 год – это очень давно! Всё в инженерии происходит быстро, и не только в железной инженерии) статье «Наука и искусство системной инженерии»[35], отражающей опыт космической системной инженерии NASA, приводится сравнение системного инженера со специально обученным и талантливым дирижёром симфонического оркестра, который творит Симфонию. Системный инженер, как дирижёр, налаживает работу «симфонического оркестра» из многих инженеров-разработчиков, и даже часто не отдельных людей-разработчиков, а целых команд разработчиков. Это одновременно и правда, и неправда. **Правда в том, что системные инженеры – это технические лидеры. Неправда в том, что системный инженер – это один человек, «рулящий» огромными коллективами других инженеров.**

Как в инженерии произошло разделение на инженеров по специальности (механиков, электриков, программистов, теплотехников и т.д.), как в западной медицине произошло раз-

[35] http://www.worldscinet.com/srf/03/0302/free-access/S1793966609000080.pdf

деление врачей по разным врачебным специальностям, и врачи редко работают поодиночке, так подобное **углубление**[36] **разделения труда** уже произошло и в самой системной инженерии киберфизических систем, и во многих других областях более высоких системных уровней, в которых слово «инженерия» не употребляется. Системных инженеров разной специализации может быть в одном проекте целая команда, которая делится, как мы помним и по видам подсистем (у самолётов это могут быть двигателисты, у организации – бухгалтеры. То, что подсистема в одной системе, на другом уровне имеет свою целостность и будет требовать архитектурной проработки и какой-то организации введения в эксплуатацию), и по практикам (прежде всего это деление на разработчиков, включая занимающихся функциональностью системы в целом, архитекторов, отвечающих за архитектурные характеристики, девопсов, отвечающих за минимизацию времени ввода в эксплуатацию каждого инкремента системы, который сделали разработчики).

Вспомните ситуацию начала времён WWW, когда появилась и начала бурно развиваться профессия веб-мастера в середине 90х прошлого века. Трудно уже вспомнить, но всего двадцать лет назад был один человек, который занимался для вебсайтов всем: программировал движок, разрабатывал арт-дизайн, пришивал его к движку, наполнял вебсайт материалом, продвигал его в Сети и т. д. Один человек, который совмещал в себе всё разнообразие прикладной инженерии для сайтостроения. Сейчас «вебмастера» уже нет, а есть отдельно программисты CMS (content management system) или программисты бэк-энда, дизайнеры, верстальщики (которых уже так не называют, это программисты фронт-энда), тимлиды и DevOps, редакторы (в вариантах editor и content manager, причём второго уже и редактором не называют, а раньше называли), администраторы, модераторы, специалисты по пользовательским интерфейсам и по пользовательскому опыту (раньше их не различали, теперь даже пишут UI/UX), SEO (search engine optimization) и после этого появившиеся спецы по SMM (social media marketing), и это еще не полный список (вы заметили, что в этом списке не хватает архитектора? Он обязательно должен быть!).

При развитии практически любой профессии она дробится на различные профессиональные/деятельностные/инженерные позиции, которые соответствуют различным практикам, которые играются какими-то профессиональными/деятельностными/организационными/проектными ролями.

Один человек, даже если он гений, не в состоянии удержать целостность в современных сложных проектах: целостность удерживается в современном мире только командно и уж точно не людьми, а хорошо налаженными компьютерными моделями, которые точно не забудут ни одной детали, ни одной строчки ни одного документа. От играющих самые разные роли и поэтому имеющих самые разные интересы людей в инженерном проекте требуются две вещи:

• **Согласовывать свои решения. Не договорились – не делаем!** Если делаем ракету, то нужно договориться, сколько у неё ступеней – две или три. Если делаем рыбу, то нужно договориться, какой она должна вырастать длины. Если делаем общество, то нужно договориться, какое именно общество (при этом договариваться в случае общества нужно не только со своей командой социальных инженеров!). И хорошо бы разделить проблемы так, чтобы нужно было поменьше согласовывать решения (например, диаметр болтиков на плате и цвет корпуса прибора можно не согласовывать между собой, эти решения могут принимать разные

[36] Обычно deep, «глубина» в речи означает рост числа уровней какой-то иерархии. Скажем deep learning – это многоуровневое обучение, то есть обучение нейронных сетей с числом скрытых уровней/слоёв больше одного. В углублении разделения труда речь идёт об увеличении числа уровней иерархии по *отношению специализации* (род-вид) между практиками.

команды разработчиков независимо, но если болтиком занимаются две команды, то их решения придётся между собой согласовывать). Этим занимаются инженеры-архитекторы.

• **Дисциплина всё записывать/учитывать: договорились и делаем – записать, о чём договорились, что делаем, что сделали, чем обосновывали. Не записали – это не договорились, так что – не делаем! Памяти не верим, верим записям.** Современная инженерия без компьютеров не делается! Внимание людей к деталям проекта и целокупности проекта поддерживается компьютерами! И личная собранность и организационная/командная/коллективная собранность сегодня держится не на биологической памяти, а на компьютерной памяти! Голыми руками и голым мозгом не работаем! То, что «все ходы записаны» и легко разобраться в ситуации, за это отвечают девопсы (управление версионированием и конфигурацией у них).

Есть ли «системный инженер» в сайтостроительстве как отдельная должность? Может быть (хотя называться он может как-нибудь по-другому, например тимлид или «архитектор проекта», а иногда и «ведущий аналитик», у которого и права принимать решения нет, только «право давать рекомендации начальникам»), а может и не быть. Есть ли «системная инженерия» в сайтостроительстве? Безусловно, есть: это и обсуждение сайта как целого (создание концепции использования), и практики создания сайтовой архитектуры, и практики технического и организационного лидерства как поддержки совместного технического творчества инженеров как в технической части, так и в организации труда этих инженеров (это практика DevOps).

Это сравнение системного инженера с дирижёром можно прокритиковать и с другой стороны. Метафора симфонического оркестра соответствует административной модели управления, «писанной музыке», и чуть ли не «руководству» как буквально «руками водству». Может возникнуть впечатление, что системный инженер – это человек с должностными полномочиями командовать какими-то «несистемными» инженерами, то есть прикладными инженерами-разработчиками разных специализаций, которые слыхом не слыхали о системных уровнях, архитектуре и непрерывном вводе в эксплуатацию. Вроде как это такой «дирижёр», который прямо диктует музыкантам, когда и какую ноту играть каждому, как это происходит иногда в симфонических оркестрах с авторитарным дирижёром, заботящимся о каждой ноте, чтобы она соответствовала именно его авторскому замыслу. И да, раньше и в инженерии (помните Туполева, Королёва, Мессершмита?), и в политике (вождизм – это как раз от этого, несменяемые цари-диктаторы как раз таковы) так и было. Но ведь ещё в прошлом, XX веке музыка по факту пошла другими путями:

• в симфонической музыке за счёт компьютерной поддержки композитора оказались выкинутыми и дирижёр, и его оркестр, а «симфонические» фонограммы к современным фильмам и играм готовятся самим композитором буквально в одиночку – но на компьютере. Компьютер настолько увеличивает возможности одного человека, у которого есть в голове полный замысел, что других людей можно смело выкинуть из процесса. Музыка оказалась несложной, умещается в одной голове, если ей помогает компьютер. Симфонический оркестр с нотной записью партитуры для каждого музыканта и дирижёром по факту оказался не нужен.

• в музыкальной мейнстримной живой культуре сегодня преобладает джаз, подразумевающий совсем другие принципы коллективного творчества: импровизация плюс взаимоподстраивание (рок – это развитие того же джаза, ибо в рок-группах никакого «дирижёра», а вместо «нот» используется звукозапись);

• симфонические оркестры остались как очень дорогое средство антикварного хранения традиции, и от них никакого развития музыки не ожидается. Эпоха великих композиторов и великих дирижёров окончилась. Развитие музыки и музицирования идёт, при этом идёт достаточно бурно, но в совершенно других формах. Например, дети массово заканчивают

сегодня не столько музыкальную школу, сколько школу диджеинга – и там игра идёт не отдельными нотами, а более крупными блоками (сэмплами, треками), и на выходе не только «треки», но и отдельные «сеты» (аналоги альбомов звукозаписи или концертных программ в живой музыке).

Если посмотреть на то, что происходит в инженерии (в том числе и программной, и «железной», и инженерии организации, и врачевании, и социальной инженерии как политике, и любых других видов инженерии), то тренд к «джазовой» организации деятельности несомненен. Все движение agile – это движение именно в эту сторону «джазовости» как импровизационности и согласованию деятельности «на лету». Это и понимается как «гибкость», непредзаданность последовательности шагов, уменьшение размера планируемой заранее работы по принципу small batch size[37], введение не всей системы с тысячей функций одним шагом (и этот шаг с подписанием акта приёмки-сдачи будет последним шагом проекта), а инкрементами, функция за функцией. Как ни странно, такой «постепенный» подход заодно улучшает скоростные характеристики разработки за счёт того, что раньше получается отклик от задействования целевой системы в её окружении и поэтому раньше исправляются ошибки, меньше тратится времени на проработку заведомо неудачных идей.

Все остальные примеры новинок в организационных дисциплинах (например, переход от акцента на administration/management к leadership, предложение блокчейна в финансовой инженерии) тоже идут именно в эту сторону отсутствия «единоличного лица, принимающего все ответственные решения». Ситуация, при которой все главные решения принимаются одним лицом, которое всеми «дирижирует», опасна. «Дирижёр всего» потенциально создаёт угрозу появления в проекте «бутылочного горлышка», существенно замедляющего принятие инженерных решений (ибо решений много, дирижёр один, все решения он не то что скоординировать – он просто познакомиться с ними не успеет толком)! Но главное тут даже не в замедлении работы: одному человеку иметь образование и опыт во всех дисциплинах, в которых принимаются важнейшие решения по проекту невозможно: гений в одних вопросах вполне закономерно может быть полным идиотом в других вопросах. Метафора «великого вождя» в системной инженерии не соответствует духу времени.

Это соответствует и идеям, о которых говорит John Doyle в его работах по системам управления[38]: для устойчивого (не срывающегося в широком разнообразии режимов) скоростного и точного управления нужно иметь большое разнообразие датчиков, актуаторов, обработчиков информации, которые организованы многоуровнево, системно. В этой структуре при этом множество самых разных обратных связей, сложно устроенных (и просто устроенных прямых связей, команды «просто исполняются», а вот информация от датчиков обрабатывается сложно). Крупные быстрые элементы дают скорость, мелкие медленные – точность, и компромисс между скоростью-точностью будет только при сочетании достаточного числа вот этих разных масштабов, появлении множества мест принятия решений. По большому счёту именно такое и происходит в «джазовом» проекте, в котором непрерывно меняется как будущее окружение (его оценки! Его же ещё нет, системы ещё нет!), так и в ответ на это структура целевой системы (которая только-только проектируется и изготавливается, её тоже зачастую ещё нет), так и в ответ на это цепочка создания (по мере изменений целевой системы требуется изменять и практики её создания, то есть менять создателей).

Системные инженеры как люди, занимающие позицию в проекте в профессиональной роли отвечающего за какую-то целокупность всей системы (помним, что мы говорим о системе

---

[37] См. доклад Влада Свяжина рассуждениями об этой идее, с 1:04:11, https://www.youtube.com/watch?v=lpgpnCoV14w&t=3851s

[38] https://ailev.livejournal.com/1622346.html

самых разных уровней – это может быть бактерия для генного системного инженера, или городской квартал в «умном городе» для инженера-строительного девелопера, или общество какой-то страны для политика) специализируются тем самым до лидеров инженерной разработки/development (самый верхний уровень, концепция использования и самые верхние уровни концепции системы), и часто на этом самом верхнем уровне это неотличимо от работы инженеров-разработчиков, инженеров-архитекторов и DevOps (разработчиков инфраструктуры непрерывного введения в эксплуатацию, аналог «управления жизненным циклом» в ситуации множественности этих жизненных циклов). Системные инженеры не руководят («руками водят», то есть дают поручения на выполнение работ по развитию системы) прикладными разработчиками, но принимают решения по принципам их работы и структуре развиваемой/evolving разработчиками системы, а также создают условия (в том числе инфраструктуру) для работы всех разработчиков. Это не руководство или управление/management, это governance.

Команды системных инженеров не столько руководят специализированными на работе в каких-то прикладных инженерных областях (domains) людьми, выполняющими ещё более узкие инженерные практики, сколько организуют их взаимодействие по поводу разделения их труда, выполняя свой кусок инженерной работы в части целевой системы и технологий, используемых в проекте создания (прежде всего технологии непрерывного ввода в эксплуатацию).

Вместо «дирижёрской» поэтому лучше использовать «джазовую» метафору описания деятельности, это подробней будет изучаться в курсе системного менеджмента/инженерии предприятия. Так, звукорежиссёр из записанных в студии разными музыкантами в разное время отдельных треков делает окончательную запись. Но он не предписывает, какую музыку играть музыкантам. Звукорежиссёр тут – DevOps. Или руководитель джазового ансамбля, который выбирает, какую мелодию будут играть – но он не командует кому, когда и что играть, и не выдаёт точные ноты мелодии, не указывает точную гармонию или ритмический паттерн. Это разработчик концепции использования. Или художественный руководитель, который определяет, кто вообще будет играть в ансамбле и устраивающий разнос музыкантам по поводу плохого качества их игры. Это архитектор. Люди, исполняющие роли системных инженеров в команде тоже имеют различающиеся функции, в самых разных их сочетаниях. Но как музыкантов из ансамбля называют «музыкант» (исполнителей ролей музыкантов, отождествляя их с ролями), так и мы системных инженеров из команды/коллектива проекта будем называть «системный инженер» для краткости. Курсы «Онтологика и коммуникация», «Практическое системное мышление», «Методология» помогут вам тут разобраться с различиями ролей, должностей, исполнителей ролей (как отдельных людей, так и организованных в плане распоряжения своим и чужим трудом, инструментами и материалами групп людей). В курсе системного менеджмента как инженерии организации вы получите дополнительный опыт того, как строить системы из людей так, чтобы в их состав входили все нужные виды инженеров (входили люди, исполняющие все необходимые для создания целевой системы практики, исполняя все нужные инженерные роли, включая роли в системах цепочек создания).

Разные виды системных инженеров имеют и разные акценты в их образовании. Так инженер-архитектор знает множество архитектурных паттернов для того вида целевых систем, который развивается в проекте. И ещё он имеет опыт разработки, чтобы не отрываться от реальности и не требовать от разработчиков невыполнимого. DevOps хорошо понимает в операционном менеджменте, ибо его главная задача – уменьшать Lead Time (этих Lead Time есть ещё и много разных вариантов), в общем случае определяемого как время от момента, когда появилась идея очередного инкремента системы (например, реализующего какой-то запрос клиентов на новую фичу) до момента, когда потребители смогут воспользоваться этим инкрементом в составе эксплуатируемой системы, и всё это должно происходить быстро, но без роста числа дефектов эксплуатирующейся системы (да, это умение выполнять модернизацию

двигателя автомобиля не выключая двигатель, и даже не прекращая движения! И тут только доля шутки).

По большому счёту, такие акценты в образовании, какие нужны архитекторам и девопсам (впрочем, и разработчикам, им же нужно изобретать в рамках их рабочих обязанностей!), не помешают каждому человеку, ибо рано или поздно в своей жизни каждый человек столкнётся с необходимостью быть системным инженером, то есть с необходимостью отвечать за всю систему в целом (даже если это маленькая подсистема в составе какой-то большой системы).

И инженеры-разработчики, и инженеры-архитекторы должен владеть какими-то практиками перевода проблем (которые непонятно, как решать) в последовательность задач (которые понятно как решать) в ходе многоуровневой оптимизации неизбежных конфликтов между системными уровнями в разрабатываемой системе: архитектура ведь подразумевает модульный синтез, который вовсе неочевидно как сделать, ибо учитывать приходится минимально четыре разных описания системы (функциональное, конструктивное, размещение, стоимостное), да ещё и на многих системных уровнях, да ещё и по цепочке создания. В настоящем курсе будет дано представление о таких практиках разработки как практиках системного творчества и об архитектурных практиках как достижении оптимального значения так называемых архитектурных характеристик (в том числе и такой характеристики, как возможность менять систему без полной её переделки, evolvability).

Главным же критерием отнесения какой-то инженерной специальности к (безмасштабной/фундаментальной/трансдисциплинарной) системной инженерии, а не (прикладной) инженерии систем является то, что системный инженер думает о всей системе на многих системных уровнях в целом, а не о каком-то её аспекте (механическом, электрическом, программном и т.д.) на каком-то одном системном уровне. Именно этот критерий даёт основание David Firesmith относить инженеров по безопасности и защите (вроде как прикладных инженеров) к системным инженерам: несмотря на то, что инженеры по безопасности имеют свои ВУЗы, профессиональные организации и конференции, они думают обо всей системе в целом, и на этом основании их вполне можно считать системными, а не прикладными инженерами. В нашем курсе не обсуждается их практика, но она оказывается в чём-то похожа на практику архитекторов, только набор характеристик там другой (не архитектурные, а безопасности и защиты). Всё это, конечно, более-менее условно, в том числе и потому, что в проекте люди обычно выполняют множество ролей и быстро переключаются между ними.

В любом случае, название «системный инженер» будет использоваться в курсе как название роли (тип мета-мета-модели, объект из нашего курса, а не из жизни), но нужно учитывать, что в жизни в разных предметных областях (культурная мета-модель, отражающая domain и ситуационная мета-модель, отражающая принятый способ работы в конкретном проекте) вы будете слышать самые разные другие имена и ролей и тем более должностей. Вы должны будете догадаться, что речь идёт о системной инженерии (вообще-то **всегда** идёт речь о системной инженерии!), и тогда вам пригодится знание материала нашего курса, в самых разных проектах. И это может быть отнюдь не SoTA версия инженерии, то есть инженерия в вашем проекте может оказаться не системной, не безмасштабной, не непрерывной (зато люди будут гордиться, что они заняты «искусством», проявляют находчивость и изобретают велосипеды, а лучшие из них будут считать себя «ремесленниками», а не инженерами. Десяток лет в программной инженерии такое отношение к методам собственной работы было даже модно, сейчас об этом предпочитают не вспоминать).

В мире существует несколько профессиональных организаций системных инженеров, из них наиболее значительной является международный совет по системной инженерии (INCOSE, International Counsil of Systems Engineering, http://incose.org).

В этой организации есть индивидуальное членство ($50-$160 в год) и корпоративное членство ($4750—7250 в год), в 2022 году примерно 19 тысяч индивидуальных членов во всех стран мира[39].

Цели INCOSE – распространять знания по системной инженерии, обеспечивать международное сотрудничество и обмен опытом системных инженеров, устанавливать стандарты профессионального мастерства и проводить сертификацию системных инженеров на соответствие этим стандартам, обеспечивать поддержку корпоративных и правительственных образовательных программ в области системной инженерии.

Деятельность INCOSE проходит главным образом в рамках деятельности региональных (уровня штата или города в США, страны или даже группы стран в Европе и Азии) отделений.

Есть Русское отделение INCOSE (INCOSE Russian Chapter). Русское – это название по языку, а не по стране. Члены этого отделения из России, Украины, Белоруссии. Это похоже на Немецкое отделение (INCOSE German Chapter), где члены из Германии и Австрии. Русское отделение входит в INCOSE EMEA sector (отделения INCOSE из стран Европы, ближнего Востока и Африки). Русское отделение INCOSE проводит заседания раз в две недели, на этих заседаниях заслушиваются доклады о современных практиках системной инженерии, прорабатывается использование русской терминологии, ведётся обсуждение исследований[40]. Раз в год в подмосковном Бекасово проходит трёхдневная Рабочая встреча Русского отделения INCOSE по проблемам системной инженерии (до 2022 года было проведено 12 таких встреч[41]).

INCOSE разработало этический кодекс системных инженеров[42], активно участвует в разработке международных стандартов, сертифицирует системных инженеров (по факту – инженеров киберфизических систем, хотя это явно не декларируется).

Тем не менее у этой организации есть существенные проблемы:

• Исторически там собрались системные инженеры киберфизических систем, и значительная часть их занимается государственными инфраструктурными и военными проектами.

[39] https://www.incose.org/about-incose

[40] Исторические видеозаписи этих заседаний и демонстрирующиеся там слайды можно найти на http://incose-ru.livejournal.com/, сегодня работа ведётся главным образом в телеграм-канале https://t.me/incose_rus.

[41] На 12 встрече как раз обсуждался и текущий курс, см. заметки об этом: https://ailev.livejournal.com/1623518.html

[42] https://www.incose.org/about-incose/Leadership-Organization/code-of-ethics

Это означает, что меньше всего там используются принципы lean, ведущие к минимальному дублированию работ (госбюджет заплатит! Это не рынок!) и agile (государство строго планово! Никакого «планирования на лету» и тем более «неудачных экспериментов»!). По факту варианты системной инженерии, поддерживаемые этой организацией, базируются на идеях «водопада» и не учитывают идей непрерывной инженерии (в общих словах – признают и учитывают, но при рассмотрении сути практик в публикуемых учебниках, документах, стандартах – нет). Ровно по этой причине частная компания SpaceX смогла за несколько лет сделать в космических проектах больше, чем все традиционные контракторы NASA (достаточно сравнить проект Starship с проектом SLS по скорости разработки и новациям). И таких «частных компаний» в организации явно недостаточное представительство, больше представителей компаний с крупными военными проектами. SpaceX активна в INCOSE была где-то в 2007 году, но не сейчас.

• INCOSE признаёт, что системная инженерия безмасштабна и используется для всех системных уровней (это ясно видно по INCOSE Vision 2035[43]). Но исторически всё равно центрируется на системно-эволюционном уровне киберфизических систем как основных для членов этой организации.

• Также INCOSE признаёт, что «инженеры систем» и прикладные инженеры по отдельным прикладным инженерным практикам (электрики, теплотехники, метрологи) по факту тоже используют системное мышление, и произошло проникновение системной инженерии во все «другие инженерии». Но всё равно организация поддерживает главным образом инженеров-генералистов для киберфизических систем (робототехников, автостроителей, авиастроителей, ракетостроителей).

Повторим, что совсем необязательно называться системным инженером, чтобы им быть. Так, архитекторы софта (software architects) вполне себе системные инженеры, специализированные на программных системах. Архитекторы предприятий – системные инженеры, специализированные на организационных системах. Дело совсем не в названии должности или практики, не в членстве в ассоциации, дело в содержании инженерной практики.

Мы тут использовали термин «системная инженерия» в двух разных значениях, так он используется и в жизни:

• Любая инженерия, ибо без системного подхода, учёта самых разных систем в окружении и цепочках создания, учёта эволюции и непрерывной подстройки под изменения нельзя менять мир. Как бы ни называлась практика, она оказывается инженерной, тем самым системноинженерной. Любой инженер – это системный инженер. Это соответствует определению «**системная инженерия – это практика инженерии с системным мышлением в голове инженера**».

• Те практики системной инженерии, которыми занимались люди, считавшиеся системными инженерами в инженерии киберфизических систем: на сегодня это главным образом архитекторы и DevOps, а также создатели концепции использования (ConOps, OpsCon). Эти люди получают специальное обучение по своим поддисциплинам. Это соответствует определению «**системная инженерия – это практика поддержания целостности системы, в отличие от прикладных инженеров-разработчиков, системный инженер занимается системой в целом, а не какими-то частями системы или прикладными особенностями системы**».

[43] https://www.incose.org/about-systems-engineering/se-vision-2035

Какое значение правильное? Оба, и жизнь покажет, какое из них выживет. Пока же используются оба значения, в нашем курсе/книге тоже так.

Инженерия и наука

Нельзя путать роли инженера и исследователя. Мы тут говорим слово «исследователь», а не «учёный» главным образом потому, что научное/рациональное мышление как построение объяснительных (про причины-следствия) порождающих (generative, пригодных для «обратного моделирования», предсказания состояний мира, конкретизации/рендеринга) теорий сегодня используется отнюдь не только в «науке» как особом роде деятельности «учёных», но повсеместно в самых разных деятельностях/практиках/видах труда.

Исследователи (это роли, а не люди!) ровно обратны инженерам: если инженеры делают реальные материальные вещи, опираясь на какие-то (необязательно «научные»!) описания, то исследователи делают объяснительные теории физической и абстрактной (математики как раз занимаются исследованием поведения абстрактных/ментальных объектов) реальности. Исследователи получают компактные, понятные и формальные описания действительности. Слово «реальность» тут означает именно физический и ментальный мир, а вот «действительность» – мир, преломлённый нашими понятиями, отражающими то, что нам об этом мире уже известно и что мы можем выразить по поводу этого мира (скажем, если человечество пару веков назад не знало о существовании электронов, то в реальности они были, а в действительности их не было). Конечно, инженерия и исследования тесно переплетены:

Конец ознакомительного фрагмента.