Тема: Build2019, осмысляем увиденное

Буквально на прошлой неделе прошла самая большая конференция Microsoft для разработчиков Build2019. Поехав туда, я преследовал 2 цели.

* Первая цель - понять, куда идет Microsoft с точки зрения разработки и какие технологии и подходы продвигает.
* Вторая цель - понять состояние сообщества вокруг Microsoft. Build - публичная конференция - куда лучше дает об этом представление, чем внутренние конференции Microsoft для своих сотрудников. На публичном мероприятии можно делать выводы даже по количеству людей, пришедших на сессию.

Для себя я выделил несколько ключевых тем, по которым хочу высказаться:

* **.Net Framework Roadmap**
* **Kubernetes**
* **ServerLess**
* **Edge Computing**
* **Big Data, Machine Learning, Artificial Intelligence**
* **Windows Presentation Platform**

**.Net Framework Roadmap**

Мои наблюдения с момента анонса .net core 3.0 в следующем: создание .net core/.net standards было очень правильными, но в .net, как и в других платформах, накоплено очень много кода, который нельзя вот так взять и перенести. А значит надо привнести плюшки .net core/standard в старые проекты с минимальной модификацией. WinForms/WPF вынесли в opensource, следующий релиз будет уже с .net standard под капотом, можно будет забыть про старую структуру csproj файлов. Вещи безусловно правильные, но вызваны они в том числе тем, что legacy надо тоже поддерживать.

А вот, что мне не понравилось на этой сессии, так это настойчивые попытки прописать всем .net/C# разработчикам необходимость знать, понимать и на ежедневной основе использовать Machine Learning, Cognitive Services и т.п. Безусловно, толковый разработчик должен такие вещи понимать, изучать, по возможности практиковать. Но в 99% случаев, в проекте это не требуется и столь настойчивое евангелизирование, откровенно говоря, раздражает.

**Service Fabric vs Kubernetes**

3 года в Microsoft и уже почти полгода в EPAM я не могу дать нормального технического объяснения зачем нужен Service Fabric (и вы не разработчик из Microsoft), если есть Kubernetes.

Долгое время толмачи от маркетинга рассказывали, что это равные платформы, но это не так (гуглится элементарно). Затем были разговоры, что SF более зрелый для Windows development, а Kubernetes для Linux.

Ну и заодно, мол, на SF ваши разработчики смогут использовать свои навыки работы в Windows Environment, а в K8s надо изучить Linux и переучиваться. Согласен, legacy код и необходимость переучивать-переучиваться – это серьезный аргумент, а с другой стороны - SF была достаточно сложной, и на мой взгляд ничуть не легче чем K8s. Ну а Linux с одной стороны изучать и знать надо в любом случае даже закостенелому “виндузятнику”, а с другой - Kubernetes достаточно сильно от этого абстрагирует.

На Build2019 я для себя поставил крест на SF, т.к. по нему был 1 доклад (хотя в названии и было что-то про Mission Critical \*\*\*), а по Kubernetes штук 20-30 во всех разрезах (разработка, облака, devops, best practices, enterprise security). На мой взгляд, тут Microsoft сдались и приняли реальность - SF сообщество, по сути, не адаптировало, с чем я нас с Вами и поздравляю.

**ServerLess всем и каждому.**

По-прежнему очень много говорится про ServerLess вычисления. Но если раньше под этот соус заворачивали только Azure Functions и Logic Apps, то сейчас зацепило даже Azure SQL Database ServerLess (когда увидел анонс, очень удивился т.к. уж что-что, а ServerLess SQL – это звучит интересно).

Суть SQL Serverless в том, что Вы можете указать нижнюю и верхнюю границу потребляемых ресурсов и платить будете обязательно по нижней границе и плюс то, что потребите между нижней и верхней границей. Мелкую фичу, что можно ставить SQL на паузу (и не платить за compute ресурсы), если база не используется (на 6 часов можно ставить на паузу сейчас), я бы пока не рассматривал, т.к. чтобы пару часов вообще ни одного запроса к базе не было - это очень странная ситуация, ведь есть же хотя бы мониторинг, да и старт у базы (как обещали) будет за 30-60 секунд, что тоже важно.

С другой стороны - Serverless можно запустить не только в Azure, но и локально в контейнере. Таким образом, Serverless становится более понятным и менее волшебным (инженеры обычно не любят волшебство, т. к. не понимают, как оно работает… а если не работает, то, как волшебство починить).

**Edge Computing по подписке или Cloud не только в Microsoft Data Center**

* Когда-то давно, в 2008-2010 годах, Microsoft говорил примерно так: «вот вам наши magic Cloud Service, перепишите ваши приложения под них (Web/Worker Role) и будет у Вас счастье». Счастья не получилось, т.к. Legacy все портило.
* Потом появились виртуальные машины, но сейчас это не важно.
* Затем, когда стало понятно, что даже теперь мало кто может полностью переехать в Cloud, надо все-таки делать Hybrid решения. В начале, под этим подразумевались VPN до Azure. Затем появился Azure Stack (программно-аппаратный комплекс, который по API похож на Azure и можно было на нем держать Production нагрузку, если регулятор не разрешает публиковаться в Public Cloud, а dev/test с анонимизированными данными держать в Public Cloud)
* Потом стали модными и популярными всякие IoT решения (для Microsoft и других больших игроков… а телеметрию-то давно писали). Поначалу в сценарии Connected Factory (Завод/ сборочная линия), предлагали использовать решения из Azure Cloud, но индустрия-таки объяснила, что Latency между их заводом может быть сильно больше допустимого и останавливать сборочную линию из-за “моргнувшего” интернета – это не жизнеспособное решение.
* В итоге, сначала появился Azure IoT Edge, который хоть и скачивался из Azure, но работал и обрабатывал данные на вашем железе (нужен был Docker совместимый хост, который хоть иногда в интернет подключался), но на нем поначалу был совсем скромный набор возможностей.
* Потом, когда IoT Edge показал себя полезным (для потребителей и для финансов Microsoft), стали как грибы появляться сервисы, которые раньше считались Cloud Only. Например, Cognitive Services пришли на Edge. Визуальный контроль качества продукции на сборочной линии или транспортере делать из локального docker container сильно проще, чем собирать latency до Azure Data Center.
* А вот на Build2019 многие из этих сервисов вышли в общий доступ (из preview вышли), а заодно был анонсирован SQL Server он Edge. Нет, SQL 2017 уже давно запускался на Linux в Docker, но требовал 3.5 GB памяти. Эта версия оптимизирована в том числе по памяти, для Edge устройств (на них памяти обычно мало).
* Кроме того, начали показывать Hardware на базе Azure Sphere для Secured IoT решений.
* Показали Azure Edge compute оборудование, что уже совсем не оставляет сомнений, что тема под пристальным контролем.

**Big Data, Machine Learning, Artificial Intelligence**

Я не самый большой специалист в этих областях, но и писать про них буду с позиции бывшего разработчика (хотя бывших разработчиков не бывает).

Очень часто встречал “интересную” позицию некоторых заказчиков, что IaaS надо делать в Aws, PaaS в Azure и Data в GCP. Нельзя не согласиться, что исторически каждый из провайдеров специализировался на чем-то, но сейчас, если смотреть по-крупному, во многих сервисах паритет достигнут (разница в нюансах, конечно, осталась). Интеграция решений между разными Cloud провайдерами – это, конечно, интересная работа и золотая жила, но как правило, крайне неэффективная для самого заказчика (за исходящий трафик всегда надо платить, а есть же еще latency и complexity/сложность решения).

Наверное, самым перспективным направлением для Microsoft/Azure на ближайшее время с точки зрения новых денег является работа с данными – “привет” Spark/Hadoop/Kafka, Data lakes и еще много интересных сервисов,а заодно с горячи темами (кто-же не любит на хайпе делать продажи) типа Artificial intelligence (Опять нам показывают Кортану, опять нам показывают умные автомобили). Количество анонсов, а главное кейсов (иногда синтетических, а иногда даже показывают счастливого клиента) по использованию Big Data сервисов в Azure на конференции просто зашкаливало. Большинство этих технологий open source, они все community driven и существуют во всех 3 больших провайдеров. Пару лет назад такого активного подхода к Data сервисам я, признаться, не припомню.

**Windows Presentation Platform**

Первая сессия, которую я бы хотел отметить - это сессия про Windows Presentation Platform (WPP). Если быть честным, я по невнимательности считал, что это про Windows Presentation Foundations (WPF), но зато приятно удивился, что тема была шире, ожидаемой.

Когда-то в далеком 2009 году, когда WPF только появился, была популярна тема: хостинг WinForms controls в WPF приложениях или наоборот - пытаться вставить WPF controls в WinForms. Это все было от того, что взять и махом переписать огромные куски приложения на новый framework было часто невозможно. В первых релизах такие особенности не рекламировали, но когда стало очевидно, что обратную совместимость надо тянуть, то начали очень активно пиарить.

Так вот, в 2019 году обсуждается как хостить Unified Windows Controls в WPF (UWP islands), и как развязать UI components и операционную систему (т. е. поставлять controls/compiler/runtime в качестве nuget packages и не ждать Windows Update. Идея отличная, но делать надо было еще в самой первой версии UWP, т. к. к тому времени команда Entity Framework вынесла EF из состава .net Framework, чтобы перестать зависеть от Windows Updates.)

Для сессии был отведен большой зал, но свободных мест в зале хватало (Разработку под Windows уже лет 10 хоронят), но дело не в этом. Молодых лиц в зале было крайне мало (сравниваю с сессией по Asp.Net Core или тем более Machine Learning). Это лишь мои субъективные наблюдения, я не занимаюсь похоронами разработки под Windows.

**Microsoft Ecosystem для разработчиков**

Работая полевым инженером в Microsoft, я официально поддерживал только то, что сделал сам Microsoft. От такого подхода атрофируются знания всего остального. Сейчас, когда я ушел из Microsoft и стал Solution Architect, стало жизненно важно заполнить этот пробел. А на Build была целая выставка решений партнеров для разработки и разработчиков, где можно было подойти к любому стенду и позадавать вопросы, попросить демку. Я сэкономил недели, посещая эти стенды.

* По-прежнему можно использовать SonarQube для управления техническим долгом (качество кода, а не архитектура конечно-же)
* Есть специализированные решения по DevOps на Kubertenes, которые сильно проще использовать, чем Azure DevOps
* Есть интересные решения типа Aqua/Snick, по аудиту содержимого docker images и уже запущенных контейнеров.
* Есть интересные решения по мониторингу-трассировке распределенных приложений, у которых есть преимущества перед банальным ELK или Azure Monitor (Application Insights + Log Analytics) типа \*FX
* Я уже молчу про R# и среду разработки на .net от Jetbrains (Про R# я помнил, но не использовал, т.к. код писал мало и соответственно про среду разработки особо не слышал).

Было полно всяких решений на базе AI/ML/Data и т.п., но я к ним особо не ходил, т.к. не имею практического опыта в промышленных Big Data решениях и не смогу оценить преимущества.

**Анонсы**

Если честно, то лично меня анонсы этого года не впечатлили, и я особо их не запомнил. На квантовых компьютерах хорошо хайпиться, но не заметно, что это технология сегодняшнего дня. А больше ничего толком и не анонсировали.

**Послесловие**

Если честно, то я ожидал большего количества представителей не itшных индустрий, т.к. Microsoft давно заявляет, что стремится именно помочь бизнесу с проектами, а не itшникам. Да, конечно, были доклады с участием Airbus, BMW… но я ожидал большего количества.

Я не большой специалист по office365, поэтому решил даже не пробовать рассматривать эту тему.

Все вышесказанное - это мое личное мнение, сделанное на базе моих наблюдений и моего предыдущего опыта. На объективность не претендую. Обсудить можно любой тезис.