надежность и доступность за счет резервирования. Балансировка нагрузки предполагает обычно наличие специального программного обеспечения или аппаратных средств, таких как многоуровневый коммутатор или система доменных имен, как серверный процесс [1].

В данный момент существует несколько глобальных подходов к распределению нагрузки в облачных вычислениях. Условно их можно разделить по логике построения: сетевая балансировка и балансировка по загруженности серверного оборудования.

Сетевая балансировка нагрузки заключена в распределении информационных потоков от пользователей (клиентов) между кластерными системами. Когда речь идет о балансировках сетевой нагрузки, термины балансировки нагрузки и прокси используются примерно одинаково. Используя модель OSI логически балансировку нагрузки можно представить на транспортном и прикладном уровнях. Транспортный уровень (L4) управляет балансировкой с помощью протоколов управления передачи. Прикладной уровень (L7) управляет балансировкой за счет протоколов прикладного уровня.

Балансировка непосредственно на кластерных мощностях осуществляется с помощью анализа загруженности конкретного оборудования в данный момент времени, а также прогнозирование загрузки. Для долгосрочных прогнозов используется машинное обучение с учителем, основанное на гауссовских процессах.

Динамическое распределение ресурсов для виртуальных машин (VM) в облачной среде объединяет локальное и глобальное распределение ресурсов VM. Локальное распределение ресурсов означает выделение общих ресурсов хоста для VM в соответствии с текущей нагрузкой. Глобальное распределение ресурсов равносильно живой миграции, в случае перегрузки или недогрузки хоста, с целью обеспечения достаточной производительности VM или уменьшение количества хостов для экономии электроэнергии. Для учёта неопределенности долгосрочных прогнозов при обнаружении перегрузок в режиме онлайн строится модель распределения ошибки предсказания с использованием метода ядерной оценки плотности вероятности.

Однако совокупность сетевой балансировки и балансировки кластеров (виртуальных машин, оборудования и т.д.) зависит от SLA (Service Level Agreement) – термин методологии ITIL, обозначающий формальный договор между заказчиком услуги и её поставщиком, содержащий описание услуги, права и обязанности сторон и, самое главное, согласованный уровень качества предоставления данной услуги. Как правило термин SLA используется применительно к ИТ и телекоммуникационным услугам. В таком соглашении может содержаться детальное описание предоставляемого сервиса, включая перечень параметров качества, методов и средств их контроля, времени отклика поставщика на запрос от потребителя, а также штрафные санкции за нарушение этого соглашения. Для того, чтобы соблюсти SLA, поставщик услуг заключает операционное соглашение об уровне услуг (OLA, operational-level agreement) с другими внутренними подразделениями, от которых зависит качество предоставления услуг [2]. Таким образом непосредственно балансировка нагрузки в облачных вычислениях, учитывая факторы загрузки сетевого и серверного оборудования решена в распределенных системах.

Список использованных источников:

1. Д.А. Варенов, А.А. Больных. Исследование подходов к разработке распределенной вычислительной системы на базе кластерной архитектуры для функционирования веб-приложений и сервисов обработки научно-образовательных ресурсов. //Труды XVII Всероссийской научно-методической конференции «Телематика'2010». Том 2. – СПб.: "Университетские телекоммуникации", 2010. – С. 378-379. – ISBN 978-5-7577-0354-1.
2. Proceedings of the twenty-second annual symposium on Principles of distributed computing / Annual ACM Symposium on Principles of Distributed Computing. – Boston, Massachusetts, USA. – 2003. – 380 с. ISBN:1-58113-708-7

SAFE МЕТОДОЛОГИЯ РАЗРАБОТКИ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

Марценюк Р.В.

Белорусский государственный университет информатики и радиоэлектроники
г. Минск, Республика Беларусь

Смолякова О.Г. – к.т.н., доцент

В настоящее время среди предприятий, занимающихся разработкой программного обеспечения, все больше начинают распространяться Agile-методологии. Одним из подвидов гибкой методологии является Scaled Agile Framework или, как его называют, масштабированный Agile.

В большинстве случаев причиной перехода на управление процессами с использованием методологии SAFe является неудовлетворенность другими подвидами гибкой методологии

(например, Scrum и Kanban) или неспособность существующих методик удовлетворить требованиям, выдвинутым к процессам.

В настоящее время все больше крупных компаний сталкиваются с ситуацией, когда над одним проектом или продуктом работает одновременно несколько команд. В таких случаях применение Scrum методологии не приносит положительных результатов, так как данная методология направлена на работу небольшого количества команд над одним проектом. Первым возможным решением может быть увеличение количества работающего персонала в пределах одной команды, однако таким способом будет ухудшено взаимодействие между членами команды. Другим решением является увеличение числа команд, но стоит заметить, что методология Scrum не предоставляет механизмов для обеспечения синхронизации между большим количеством команд.

Для решения вышеописанной проблемы была разработана методология SAFe (иногда называют фреймворком). По своему определению данная методология предназначена для разработки программного обеспечения с использованием гибких методик в командах размером 50 и более человек. SAFe методология является объединением лучших качеств и инструментов других гибких методологий.

Одним из важнейших факторов при формирования рассматриваемой методологии являлись принципы Lean. Данное понятие возникло в 1990-ых годах в автомобильной промышленности и представляет собой подход к управлению организацией, направленный на повышение качества совершаемой работы за счет сокращения потерь. Ниже представлены 5 основных принципов Lean [1]:

– определение ценности продукта с точки зрения конечного пользователя;

– определение и детальное описание всех действий в цепочке производства;

– формирование действий таким образом, чтобы между ними не было простоев и ожиданий;

– ограничение количества выполняемых задач, за счет выполнения только тех, которые важны для конечного пользователя;

– стремление к постоянному совершенствованию процессов и продукта.

Другим влияющим фактором при формировании SAFe является Agile манифест – документ, содержащий основные принципы и ценности гибкой методологии разработки. Данный документ включает описание 4 ценностей и 12 принципов. Ниже представлены некоторые основные принципы [2]:

– наивысшим приоритетом является удовлетворение потребностей заказчика;

– изменение требований приветствуется на всех стадиях разработки;

– работающий продукт следует выпускать как можно чаще, с периодичностью от пары недель до пары месяцев;

– на протяжении всего проекта разработчики и представители бизнеса должны ежедневно работать вместе;

– постоянное внимание к техническому совершенству и качеству проектирования повышает гибкость проекта.

Еще одной причиной успешного распространения SAFe является разделение фреймворка на 4 уровня: командный, программный, системный и уровень портфолио. Так как команда является основной сущностью в любом подвиде Agile-методологии, то можно сделать вывод, что командный уровень – это самый главный и неотъемлемый уровень SAFe. Команда, как и в Scrum, состоит из 10 человек, включая специалистов из различных областей, владельца продукта (Product Owner) и скрам-мастера (Scrum-master). Задачи последних двух является строго зафиксированными. Например, владелец продукта является голосом бизнеса внутри команды, определяет и принимает требования к выполняемым задачам. Скрам-мастер занимается ликвидацией препятствий для работы команды, защищает ее от внешних отвлекающих факторов, проводит и назначает митинги. Как и в Scrum, разработка происходит итеративно. Основной единицей является программный инкремент – период времени, включающий в себя 5 итераций. Каждая итерация длится 2 недели.

Присутствие следующих уровней может зависеть от потребностей и размеров предприятий.

На программном уровне обычно происходит приоритизация задач, построение планов по развитию архитектуры и осуществляется синхронизация между всеми командами. Именно на данном уровне должно происходить поддержание потока задач в таком виде, в котором это необходимо для достижения целей, поставленных заказчиком. Неотъемлемой частью SAFe процесса на программном уровне является планирование задач перед началом каждого программного инкремента.

Системный уровень служит для координации между различными отделами, их руководителями и заказчиками. На данном этапе производится анализ экономической целесообразности проведения изменений.

Уровень портфолио является самым верхним уровнем фреймворка SAFe. Здесь происходит распределение средств на различные направления в бизнесе, корректировка бюджета компании и управление человеческими ресурсами, необходимых для предоставления готовых решений.

Как утверждают создатели методологии, основными задачами SAFe являются [3]:

– улучшение производительности на 20–50%;

– улучшение качества продукта на 25–75%;
– ускорение выхода продукта на рынок на 30–75%;
– увеличение вовлеченности сотрудников и удовлетворенности работой на 10–50%.

Не стоит ожидать вышеописанных результатов сразу после внедрения SAFe в процесс производства. В ходе практического участия в SAFe процессах и проведения анализа было выявлено, что положительный эффект начинает проявляться через 1–2 программных инкремента. Особенно затруднителен переход к процессам разработки при помощи SAFe после использования таких классических методологий, как каскадная и спиральная. Это обусловлено тем, что разработчикам необходимо много времени, чтобы сформировать Lean-Agile мышление, которое позволяет понимать и применять принципы и методы SAFe.

Само название Lean-Agile характеризует два основных аспекта данного мышления. Первый аспект – это бережливое мышление. Такое мышление должно полностью отражать все принципы бережливого производства. Второй аспект – готовность к изменениям. На практике же именно данный аспект вызывает больше всего проблем. При переходе на SAFe процессы разработчикам требуется отвыкать работать согласно строго установленному набору требований и описанным спецификациям. Команды должны быть готовы в любой момент времени пересмотреть свои рабочие планы и подстроиться под текущие требования, убрать те задачи, которые не требуются для достижения целей разработки, и включить в работу новые задачи.

Еще одним фактором, замедляющим успешное внедрение методологии SAFe, является необходимость в децентрализованном принятии решений. В большинстве случаев в каждой крупной компании имеется человек или группа лиц, мнение которых является определяющим при решении задач. Любое решение, которое должно быть передано на более высокий уровень, приводит к задержке. При этом может быть ухудшено качество этих решений из-за отсутствия контекста. Децентрализация решений позволяет командам самостоятельно принимать решения в тех случаях, когда появившаяся проблема не является глобальной и времязатратной. При частом возникновении такие проблемы будут решаться локально членами команды, что уменьшит время задержки при принятии решений, улучшит процесс разработки и пропускную способность команды. Одним из главных инструментов, который может помочь в принятии решений децентрализованным способом, является доверие.

Для того, чтобы ускорить процесс внедрения SAFe происходит большое количество обучающих тренингов для членов команд, на которых происходит не только обучение принципам SAFe, но и закладываются основы Agile-Lean мышления, которое со временем необходимо развивать самостоятельно.

Безусловно, рассматриваемая методология не является идеальным подвидом Agile-методологии. Помимо длительного времени ожидания положительных результатов, к недостаткам можно отнести большое количество используемых временных и финансовых ресурсов на организацию коммуникаций между командами и их членами. Однако, при определенном уровне терпения и стремления крупные предприятия могут в полной мере прочувствовать положительный эффект от использования гибких методик SAFe методологии при разработке программного обеспечения – продукт станет качественнее, а работники почувствуют уверенность в собственных силах и удовлетворенность от работы благодаря пониманию собственной значимости в разработке и увеличению прозрачности процессов, направленных на развитие продукта.

Список использованных источников:

1. The Lean Way [Электронный ресурс]. – Режим доступа: https://theleanway.net/The-Five-Principles-of-Lean. – Дата доступа: 20.03.2019.
2. Agile Manifesto [Электронный ресурс]. – Режим доступа: https://agilemanifesto.org/iso/ru/principles.html. – Дата доступа: 20.03.2019.
3. Scaled Agile Framework [Электронный ресурс]. – Режим доступа: https://www.scaledagileframework.com/about/. – Дата доступа: 20.03.2019.