## производителност

Времето за отговор е важен показател в сферата на софтуера и системното инженерство, представляващ продължителността между заявката на потребителя и последващия отговор на системата. Той служи като осезаема мярка за ефективността на системата и често пряко се свързва с удовлетвореността на потребителите. Бързото време за реакция обикновено означава оптимална производителност на системата, което води до положително потребителско изживяване, докато продължителната реакция може да е показателна за неефективност в системата.

Следното уравнение е общ метод за концептуализиране на производителността:

*Време за отговор = Време за обработка + Време на изчакване*

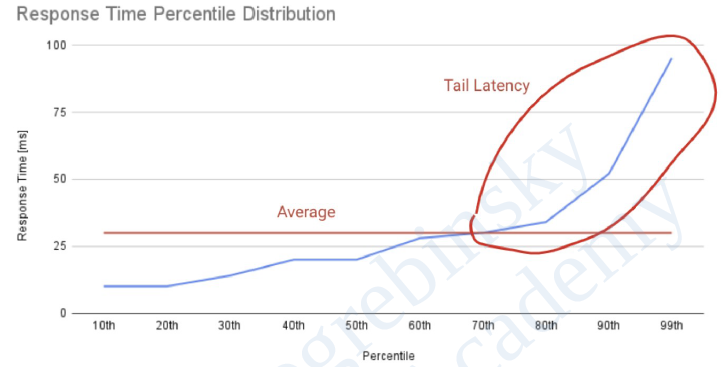
Време за отговор: Това е общото време, необходимо от момента, в който потребителят изпрати заявка до момента, в който получи отговор. По същество това е времето, което потребителят чака да види резултат след започване на действие.

Време за обработка: Това е действителното време, необходимо на системата за изчисляване на резултата след получаване на заявката. Той включва задачи като изчисляване на данни, запитване към бази данни и всяка друга работа, която системата извършва, за да изпълни заявката.

Време на изчакване: Това представлява времето, което заявката прекарва в чакане в опашки, преди да бъде обработена. В система с голям трафик могат да постъпят няколко заявки едновременно. Ако системата не може да ги обработи наведнъж, някои заявки трябва да изчакат, като по този начин се увеличава времето за изчакване.

Чрез разбирането и разделянето на времето за реакция на неговите компоненти, системните администратори и разработчиците могат да определят областите за подобрение. Например, ако времето за обработка е дълго, може да е необходима оптимизация на алгоритми или код. Ако времето за изчакване е дълго, това може да е знак, че системата се нуждае от по-добро балансиране на натоварването или увеличен капацитет за обработка.

Ниво на латентност е от съществено значение, тък като гарантира, че преобладаващата част от клиентите получават последователно и отзивчиво обслужване.



Вместо да разчитат единствено на средни стойности, крива на разпределение предоставя по-изчерпателна картина на ефективността. Примерът на фиг.1.Х показва, че 95% от заявките се обработват в сравнително оптимално време, докато 5% отнемат повече. Често процентилите се използват за определяне на цели за ниво на обслужване (SLO) и споразумения за ниво на обслужване (SLA) за производителност на системата. Това е така, защото няколко изключително дълги или кратки времена за реакция могат да изкривят средната стойност, докато крива на разпределение осигурява по-стабилна метрика, която е по-малко податлива на изкривяване от извънредни стойности.

Крайната латентност е ефикасността на заявките в горните процентили (като P95, P99 и P99.9). Акцентът е върху най-слабите заявки. Голямото забавяне може да означава проблеми, които възникват само при определени условия, като конкуренция за ресурси, забавяне на рамка за изпълнение, хардуерни проблеми и т.н.

## Мащабируемост

Трафикът и натоварването са динамични, влияещи пряко върху производителността на системата и възможността за разширяване на бизнеса. Мащабируемостта се отнася до способността на системата да управлява ефективно увеличеното работно натоварване. Познати са три измерения на мащабируемост:

Вертикална мащабируемост се случва чрез надграждане на хардуер като процесор, памет или честотна лента на мрежата, позволявайки да обработва повече трафик. Въпреки че осигурява повишаване на производителността, съществуват ограничения за това колко може да се „надстрои“.

Хоризонтална мащабируемост се постига чрез добавяне на ресурсни единици. Вместо да се подобрява един сървър, множество сървъри се създават, за да разпределите натоварването. Този подход може да предложи висока достъпност и толерантност към грешки. Това въвежда сложности, особено при координацията между ресурсите.

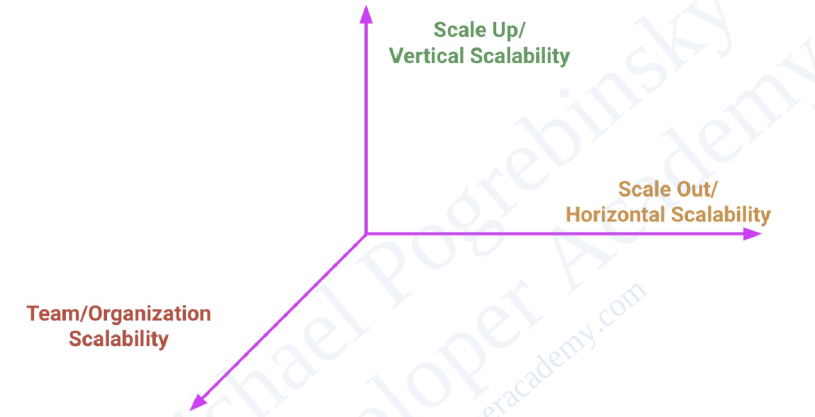
Фигурата показва пример.



Фиг.1.

Екипна или организационна мащабируемост: От гледна точка на разработчиците, мащабирането не е свързано само с производителността на системата, но и със скоростта на разработка на софтуер. С нарастването на инженерните екипи производителността може да намалее поради фактори като по-чести срещи, конфликти в кода и други.

Обща визуализация на типовете мащабируемост:



## Висока наличност

Наличността е първостепенен атрибут на качеството. Наличността на системата оказва влияние както върху потребителите, така и върху бизнеса: потенциална загуба на приходи и загуба на клиенти.

Наличността е частта от времето, през което дадена услуга е функционална и достъпна. Тя може да бъде изразена като процент от времето на работа спрямо сумата от времето на работа и времето на в застой:

*Availability = uptime / (uptime + downtime)*

Също така може да се оцени с помощта на показатели като MTBF (средно време между отказите) и MTTR (средно време до възстановяване), с формулата:

*Availability = MTBF / (MTBF + MTTR)*

Тази формула подчертава, че намаляването на времето за възстановяване (MTTR) може да подобри наличността.

Абсолютната 100% наличност е нереалистична поради необходимостта от поддръжка и надстройки. 90% наличност се равнява на над 2 часа ежедневен престой или 36 дни годишно. 95% се равнява на около час дневно или 18 дни годишно. Индустриалните стандарти, особено определени от облачните доставчици, се движат около 99% до обикновено 99,9% (наричани „три деветки“). Това представлява престой от по-малко от 1,5 минути дневно.

## Толерантност към грешки

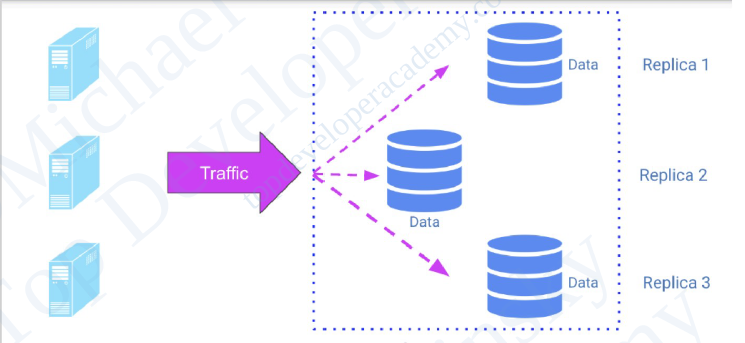
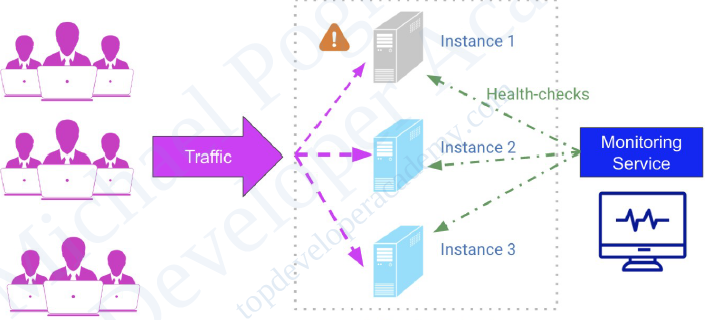
Постигането на висока наличност осигурява поддържането на непрекъснати бизнес операции. Източници на грешки могат да бъдат различни, като някои от тях са:

Човешка грешка: внедряване на неизтестван софтуер, неправилни конфигурации и изпълнение на неправилни команди.

Софтуерни грешки: проблеми като застой, получен от работните рамки, сривове поради липса на памет, изключения за нулев указател и грешки в сегментирането.

Хардуерни повреди: повреди на сървъри, рутери или устройства за съхранение поради ограничен срок на годност, природни бедствия, причиняващи прекъсване на захранването, и мрежови повреди поради инфраструктурни проблеми.

Тактики за толерантност към грешки:

* Предотвратяване на повреда: Елиминиране на единични точки на повреда: Чрез излишък и репликация. Техниките включват изпълнение на реплики на приложения на множество сървъри и реплики на бази данни на различни компютри. 
* Откриване и изолиране на повреда: Мониторингът се свързва с Инструменти, които изпращат периодични проверки, вземайки предвид процентите на грешки, времето за реакция и други показатели за ефективност, за да открие потенциални повреди. 
* За да се поддържат системните операции, като след откриването на дефектен екземпляр трафикът трябва да бъде пренасочен. Също така дефектна система може да бъде рестартирана за потенциално разрешаване на определен проблем. Като алтернатива, връщането към предишна версия може да отстрани проблеми със софтуера.