VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

VIZUÁLNÍ KONTROLA POČTU VOLNÝCH PARKOVACÍCH MÍST S VYUŽITÍM CLOUDOVÝCH SLUŽEB

VISUAL CONTROL OF THE NUMBER OF FREE PARKING SPACES USING CLOUD SERVICES

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE **VLADIMÍR HRUBAN**
AUTHOR

VEDOUCÍ PRÁCE **Ing. JAKUB ŠPAŇHEL**
SUPERVISOR

BRNO 2020

Vysoké učení technické v Brně
Fakulta informačních technologií

Ústav počítačové grafiky a multimédií (UPGM) Akademický rok 2019/2020

Zadání bakalářské práce

Student: **Hruban Vladimír**
Program: Informační technologie
Název: **Vizuální kontrola počtu volných parkovacích míst s využitím cloudových služeb**
Visual control of the number of free parking spaces using cloud services
Kategorie: Zpracování obrazu

Zadání:

1. Seznamte se s cloudovými službami a jejich možnostmi v oblasti počítačového vidění/strojového učení.
2. Vyhledejte literaturu zabývající se cloudovými aplikacemi a jednotlivými modely cloudových nasazení.
3. Prozkoumejte vhodné cloudové platformy s možností vytváření vlastních modelů strojového učení.
4. Vyhledejte přístupy pro vizuální kontrolu parkovišť a získejte datovou sadu vhodnou pro experimenty.
5. Vytvořte vlastní model strojového učení pro danou problematiku s možností nasazení v cloudu a experimentujte s ním a vyhodnoťte.
6. Vytvořte infrastrukturu cloudové apliakce s webovým frontendem pro vizuální kontrolu parkování.
7. Vytvořte plakát a video prezentující vaší práci, její cíle a výsledky.

Literatura:

- Dle pokynů vedoucího.

Podrobné závazné pokyny pro vypracování práce viz https://www.fit.vut.cz/study/theses/

Vedoucí práce: **Špaňhel Jakub, Ing.**
Vedoucí ústavu: Černocký Jan, doc. Dr. Ing.
Datum zadání: 1. listopadu 2019
Datum odevzdání: 28. května 2020
Datum schválení: 26. května 2020

Abstrakt

Cílem této práce je navrhnout a vyvinout službu běžící ve veřejném cloudu, která využívá služeb strojového učení pro vizuální kontrolu počtu volných parkovacích míst. V rámci tohoto zadání byly navrženy dvě architektury a různým podílem služeb běžících na cloudu a jedna z nich byla implementována. Zároveň vznikla uživatelská webová aplikace pro komunikaci se službou.

Abstract

The aim of this thesis is to design and develop cloud app service using public cloud services and computer vision to asses number of free parking spots at parking lot. I designed two possible architectures (using different potion of cloud services to run) and one of these was implemented. I also developed a web-app to handle user-interaction with the service.

Klíčová slova

cloud, veřejný cloud, Microsoft Azure, Custom Vision, Azure Functions, orchestrace, strojové učení, počítačové vidění, bezserverové programování

Keywords

cloud computing, public cloud, Microsoft Azure, Custom Vision, Azure Functions, orchestration, machine learning, computer vision, serverless code

Citace

HRUBAN, Vladimír. *Vizuální kontrola počtu volných parkovacích míst s využitím cloudových služeb*. Brno, 2020. Bakalářská práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Ing. Jakub Špaňhel

Vizuální kontrola počtu volných parkovacích míst s využitím cloudových služeb

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Ing. Jakuba Špaňhela. Uvedl jsem všechny literární prameny, publikace a další zdroje, ze kterých jsem čerpal.

.....
Vladimír Hruban
28. května 2020

Poděkování

Děkuji svému vedoucímu za vedení, pomoc, rady a trpělivost. Zároveň děkuji svým přátelům a kolegům za bezmeznou podporu a motivaci při psaní této práce.

Obsah

Kapitola 1

Úvod

Tato práce se zabývá využitím cloudových služeb a služeb strojového učení v cloudu pro potřeby automatické vizuální kontroly počtu volných parkovacích míst. Cílem práce je vytvořit aplikaci, která bude uživateli aplikace vracet počet prázdných parkovacích míst z určeného parkoviště. Toto parkoviště je vybavené zařízením s možností pořizovat fotografie tohoto parkoviště a tyto údaje pak odesílat ke zpracování. Uživatel tak získává informaci, zda má smysl na daném parkovišti hledat volné místo.

Osobní motivací pro tvorbu této aplikace je má vlastní zkušenost z parkoviště u mého bytového domu. To je konstruováno tak, že pokud na něj vjedu za plného stavu, není možnost se otočit, a musím tak vycouvávat poslepu do křižovatky (kde kvůli překážkám a flóře není na silnici a chodník rozhled).

V práci prozkoumávám a popisuji možnosti a specifika cloudových technologií, jednotlivé poskytovatele cloudových služeb, programování na cloudu s důrazem na moderní principy bezserverového programování, modularitu, škálování a vysokou dostupnost. Dále pak zkoumám využívání služeb umělé inteligence a strojového učení na cloudových službách – jaké zde dodavatelé cloudových technologií nabízejí možnosti, jak takový vývoj probíhá a jak se hotový model implementuje do aplikace.

V kapitole 2 práce popisuji, co to je cloud, ekonomiku fungování cloudových služeb, typy nasazení cloudových služeb, funkční modely cloudových služeb, co to je edge zařízení, možnosti služeb umělé inteligence a strojového učení v cloudu a významné dodavatele těchto technologií a jimi nabízené služby.

V kapitole 3 popisuji použité technologie, mnou navrhované dvě architektury pro implementaci aplikace (které se liší umístěním modelu strojového učení), vývoj webového rozhraní služby pro uživatele, následně se věnuji problematice dostupnosti služby, počítání SLA a predikci nákladů na službu. V poslední řadě pak popisuji možnosti dalšího rozšíření aplikace.

Kapitola 2

Technologie

V této kapitole popisuji využité technologie, principy a nástroje, které používám pro realizaci celé práce. Zabývám se zde popisem a fungováním cloudových technologií a následně implementací služeb umělé inteligence a strojového učení právě v cloudu.

2.1 Cloudové technologie

Slovo *cloud* se historicky používalo jako slangové označení pro internet, kdy byla ikona mraku – tedy cloudu – využívána v síťových diagramech jako reprezentace pro infrastrukturu internetu, kudy se mezi koncovými stanicemi přenášela data [30]. Bavíme-li se o cloudu dnes, míníme službu, která umožňuje uživatelům a zákazníkům využívat infrastrukturu, výpočetní sílu, úložiště či jiné počítačové služby, interagovat a pracovat s nimi a na tuto infrastrukturu pak dokonce nasazovat vlastní služby, aniž by tyto fyzické prostředky museli nutně vlastnit a starat se o jejich nasazení, běh a údržbu, za což dodavateli služby platí poplatek – nejčastěji formou *pay-as-you-go* neboli *PAYG*. To značí, že zákazník neplatí za služby pevný poplatek, ale je mu účtováno reálné využívání služeb (velikost uložených dat, doba běhu virtuálního stroje a tak podobně). Důležitý je zároveň systém/rozhraní pro automatické nasazování služeb – ať už pomocí webové služby nebo například aplikačního rozhraní (dále API).

Ve článku **Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility** [4] definují autoři *cloud* jako typ paralelního a distribuovaného systému, který se skládá z množství fyzických a virtualizovaných počítačů, které jsou dynamicky nasazovány a navenek prezentovány jako jeden nebo více zdrojů výpočetních služeb a poskytovány na základě SLA (neboli Service Level Agreement – dohoda o úrovni poskytovaných služeb – smlouva, která popisuje služby a vyjednané podmínky mezi poskytovatelem služby a zákazníkem – popisuje především kvalitu služby, dostupnost služby, odpovědnost smluvních stran, ale také pokuty za nedodržení podmínek a další) vyjednaného mezi dodavatelem služby a uživateli.

Mezi největší poskytovatele cloudových služeb na globálním trhu patří společnosti Amazon se svou službou Amazon Web Services, Microsoft a jeho Microsoft Azure, Alibaba se službou Alibaba Cloud Computing, Google – Google Cloud Platform a IBM se svým IBM Cloud. Podle tiskové zprávy Gartner, Inc [9] těmto dodavatelům cloudových služeb v modelu *IaaS* (viz sekce 2.1.2) v roce 2017 patřilo 72,6 % trhu a v roce 2018 dokonce 76,8 % trhu, přičemž Amazon je dominantní firmou této oblasti [9]. Na českém trhu patří mezi dodavatele cloudových služeb například společnosti WEDOS Internet se službou WEDOS

Cloud (poskytuje však služby pouze v modelu IaaS) nebo ZONER software a ZonerCloud (s modely IaaS a SaaS).

Cloudové technologie se do povědomí uživatelů, firem, vývojářů a IT profesionálů dostávají i díky možnému zrychlení vývoje, testování a nasazení služeb, odstranění potřeby vlastnictví a údržby infrastruktury a minimalizaci počátečních nákladů na nasazení služeb a aplikací. Mnoho firem již adoptovalo myšlení *cloud-first*, tedy snahu o to primárně nasazovat nové služby pomocí některého z cloudových modelů. Během posledních let můžeme u čím dál více společností podle zprávy společnosti VMware, která vychází ze zjištění analytických a auditních společností Gartner, Inc a IDC, očekávat adopci strategie *cloud-only*, tedy kompletní přechod na cloudové služby (v různých typech nasazení a i s několika dodavateli cloudových služeb) [28].

Vedle odlišnosti cloudu oproti tradičnímu využívání technologií se projevuje i rozdíl na poli ekonomického řízení podniku. Zatímco při nákupu hardwaru firma platí náklady dopředu (ve formě nákupní ceny hardwaru) a tento se pak dostává do vlastnictví firmy a patří do kategorie **CAPEX** (Capital Expenditures) neboli kapitálových (investičních) výdajů s dlouhodobými odpisy (amortizací), nákup cloudových služeb spadá do kategorie **OPEX** (Operational Expenditures) – provozní náklady s okamžitým odpisem. Zákazník však hardware (nebo IP[1] za ním) nevlastní, pouze si ho pronajímá a po ukončení spolupráce s dodavatelem k němu často ztrácí přístup. Odpadají mu však náklady spojené s vlastnictvím hardwaru, například plat či mzda techniků starajících se o údržbu, náklady na prostory a služby serveroven nebo datacenter a tak podobně.

2.1.1 Servisní modely cloudových služeb

Rozlišujeme čtyři typy nasazení cloudových služeb:

1. veřejný cloud,
2. privátní cloud,
3. hybridní cloud,
4. multicloud.

Veřejný cloud

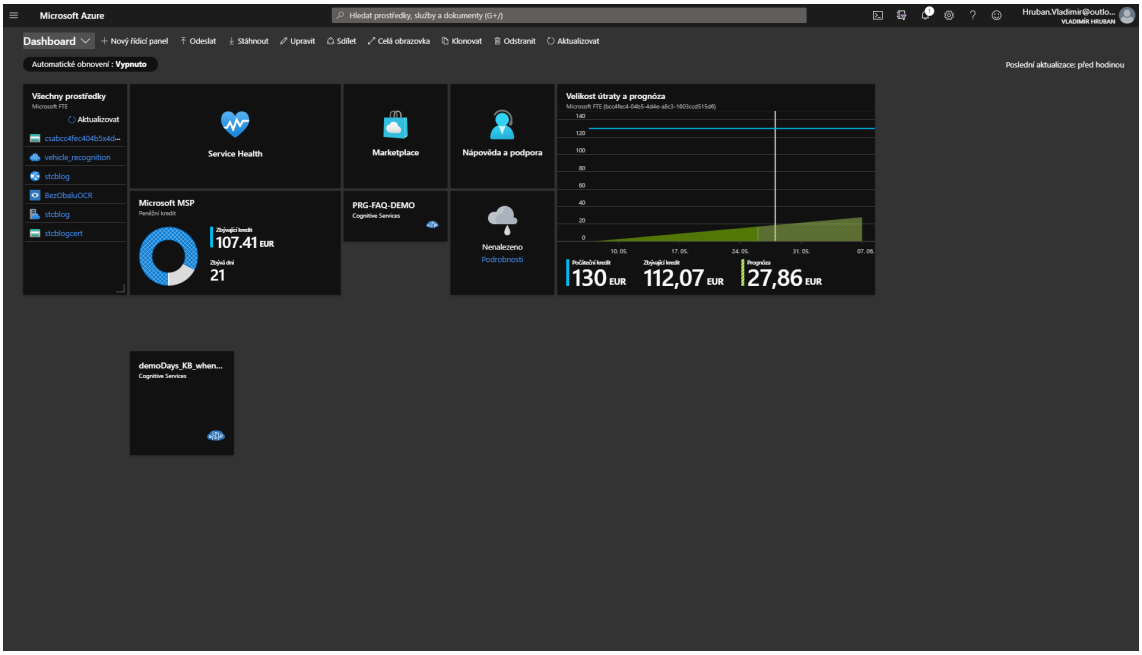
Veřejný cloud je nejčastějším servisním modelem cloudových služeb a je zároveň modelem, který umožňuje nejjednodušší škálování výpočetního výkonu a dalších zdrojů pro potřebu zákazníka. Jedná se o službu dodavatele cloudových služeb, kdy tento dodavatel vybuduje jedno nebo více datových center, která spadají pod jeho plnou správu a v nich provozuje cloudové služby (v různých modelech – od infrastruktury až po hotové aplikace) přístupné zákazníkům přes internet. Dostupnost služeb zákazníkům je pak určena pomocí SLA. Tyto služby jsou zákazníkům dodávány zadarmo či za poplatek nejčastěji spjatý s objemem využitých služeb – PAYG. Jelikož v jednom prostředí nad jedním hardwarem mohou běžet služby a prostředky více zákazníků, bavíme se o takzvané *multi-tenant architektuře*.

Jeden *tenant* si můžeme představit jako prostředí jednoho zákazníka – například všichni uživatelé řešení cloudového úložiště dat a jejich data a nastavení v rámci jedné firmy. Data těchto uživatelů mohou být poté uložena na stejném hardwaru jako data jiných zákazníků,

[1] Intelectual Property

nicméně jednotliví zákazníci si do dat svých uživatelů navzájem nevidí – právě díky multitenant architektuře. Ta přináší výhody snížení nákladů na službu, ale také jednoduchou škálovatelnost [13].

Uživatelé vytvářejí, upravují a mažou prostředky a služby přes různá rozhraní. Příkladem je webové rozhraní a API služby Microsoft Azure (viz obrázek 2.1). Uživatel může nasazovat služby a prostředky pomocí webového rozhraní Azure Portal nebo pomocí skriptovacích jazyků PowerShell či Azure CLI.



Obrázek 2.1: Prostředí Azure Portal.

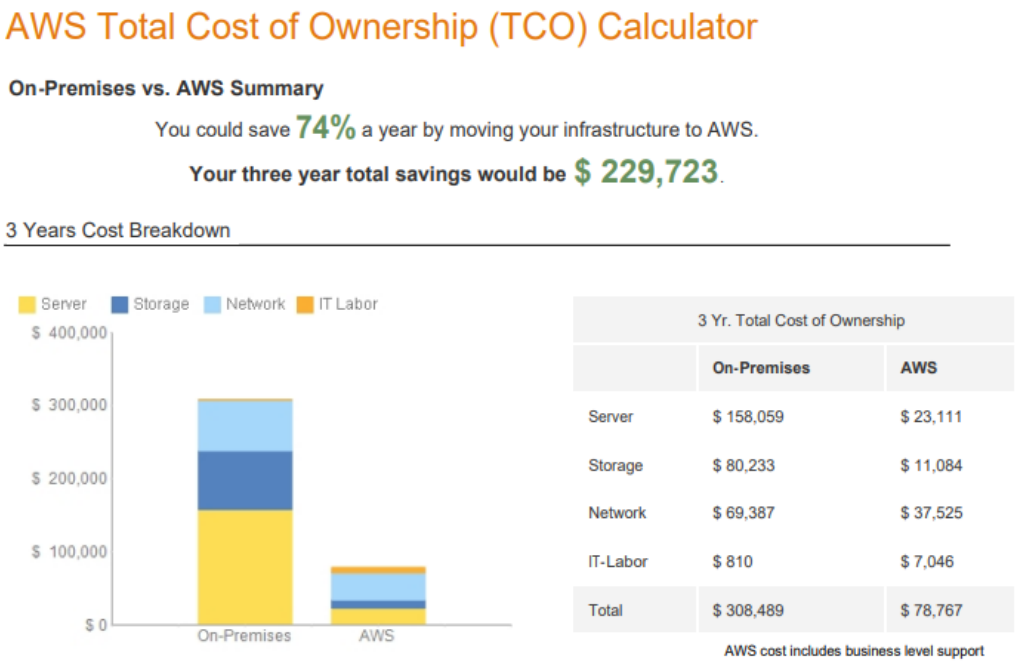
Ekonomické principy cloudu vycházejí z předpokladu, že ve veřejném cloudu může zákazník získat nejlevnější výkon. Toto je zapříčiněno tím, že velkoodběratel hardwaru a služeb dokáže stlačit cenu níže než každý jednotlivý uživatel zvlášť. Díky tomu může poskytovat služby za nižší cenu, než které by jednotliví zákazníci mohli dosáhnout sami. Pro porovnání nákladů na běh služeb na místní infrastruktuře se službami veřejného cloudu využívají dodavatelé cloudových služeb výpočet *TCO* neboli Total Cost of Ownership – výpočet celkových nákladů na vlastnictví.

Příkladem TCO kalkulačky je AWS TCO Calculator[2], který po zadání nároků na infrastrukturu vypočítává TCO jako

$$TCO = AC + OC, \tag{2.1}$$

kde AC – acquisition costs – jsou předpokládané náklady na akvizice hardwaru a OC – operational costs – jsou předpokládané provozní náklady, plat/mzda a režijní náklady v tříletém horizontu. V následujícím modelovém případu, jehož detaily naleznete v příloze C a shrnutí na obrázku 2.2, je vidět úspora až 74 % za rok při nasazení služby v Amazon Web Services [27]:

[2] https://awstcocalculator.com/



Obrázek 2.2: Výpočet TCO pomocí AWS TCO Calculator [27].

Privátní cloud

Oproti veřejnému cloudu se **privátní cloud** liší tím, že jsou veškeré prostředky tohoto nasazení dedikovány a používány pouze jedním uživatelem/zákazníkem – nad prostředky tedy běží jediný tenant. Hardware i software běží na dedikované privátní síti a zároveň na hardwaru nikdy neběží služby pro jiného zákazníka. Služby privátního cloudu poskytuje buď přímo poskytoval cloudových služeb ze svého hardwaru (za dodržení podmínek výše o plném dedikování tohoto hardwaru jedinému zákazníkovi), nebo přímo na serverech/v datacentru zákazníka. Příkladem privátního cloudu je IBM Cloud Private od společnosti IBM nebo Azure Stack od společnosti Microsoft. V případě společnosti IBM se jedná o platformu, kterou lze nasadit na jakoukoliv vlastní certifikovanou infrastrukturu (ať už postavenou na hardwaru od IBM nebo jiných výrobců), u Microsoftu se však jedná o uzavřené řešení, ve kterém je zákazníkovi dodán již hotový systém včetně hardwaru s nainstalovanou platformou, který jen zákazník připojí ke své síti. V obou případech má pak zákazník k dispozici řešení se stejnými nebo téměř stejnými možnostmi práce (nepočítáme-li případná omezení místní konektivity a omezení vlastní výpočetní síly) a stejnými službami jako ve vlastním veřejném cloudu těchto dodavatelů.

Důvodem nasazení privátního cloudu je často důraz na bezpečnost nebo regulatorní omezení u zákazníka – setkáváme se tak s ním především u finančních a státních institucí. Mezi výhody, které získává uživatel privátního cloudu, patří právě bezpečnost a kontrola nad daty a zároveň jednoduchá škálovatelnost a řízení prostředků. Nevýhodou oproti veřejnému cloudu je pak ztráta flexibility zdánlivě neomezeného množství výpočetního výkonu a případné další náklady na agendu spojenou se správou vlastního hardwaru, rovněž je privátní cloud často spojen s vyššími počátečními náklady.

Hybridní cloud

Hybridní cloud je řešení, které spojuje výhody veřejného a privátního cloudu, případně spojení veřejného cloudu s jinou místní infrastrukturou. K takovémuto nasazení obvykle přistupujeme, pokud chceme zachovat kontrolu nad kritickými nebo citlivými daty a prací nad nimi – takováto data zůstávají na privátním cloudu nebo místních databázích a veřejný cloud slouží k operacím s menším důrazem na bezpečnost – nebo například pro běh legacy aplikací[3]; při přechodu na cloud z místní infrastruktury (tzv. *on-premise*) – jedná se o stav postupné migrace do veřejného cloudu; nebo veřejný cloud využíváme jako zdroj dodatečné výpočetní síly – běžné operace běží na místní infrastruktuře a při nadstandardním vytížení jsou dynamicky vytvořeny další výpočetní zdroje ve veřejném cloudu. V hybridním nasazení se tedy schází výhoda silné kontroly a bezpečnosti, ale i flexibility.

Multicloud

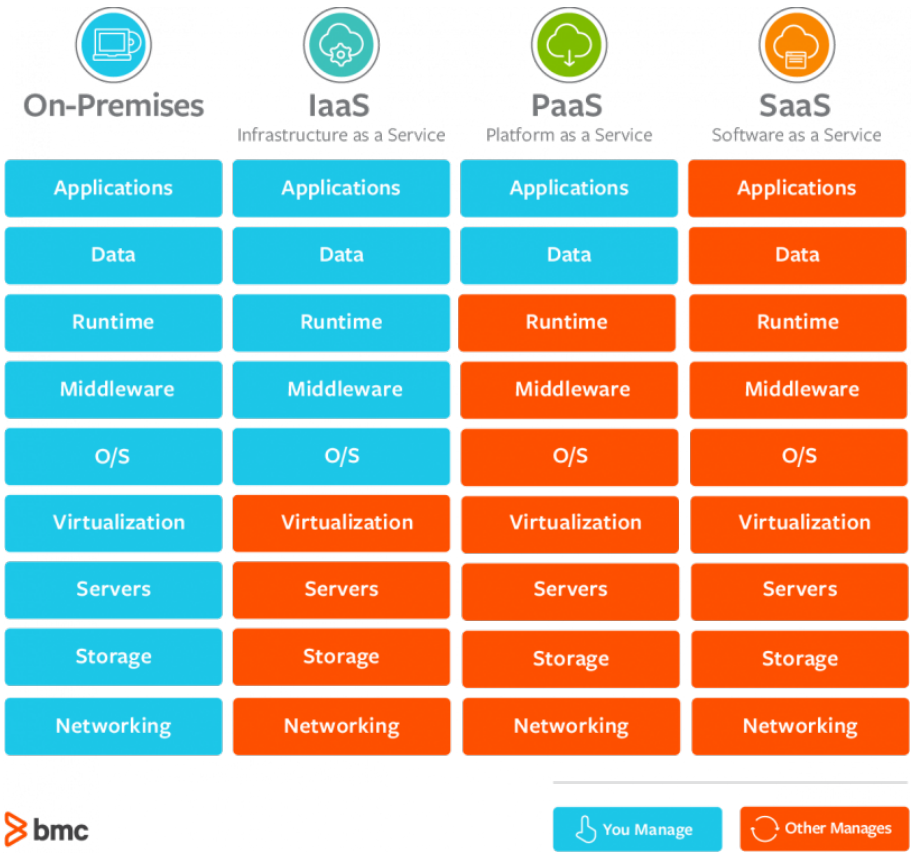
Multicloudové řešení je řešení, které využívá služeb více dodavatelů veřejného cloudu (případně může být kombinováno s hybridním scénářem). Toto uživatelům a zákazníkům přináší výhody jako minimalizace rizik, využívání možností více cloudových vendorů, snížení závislosti na konkrétním dodavateli nebo vytvoření tlaku na nižší cenu (díky kompetitivnímu prostředí) [11]. Mezi nevýhody naopak patří potřeba udržování širších znalostí cloudu jednotlivých dodavatelů, bezpečné a jednotné řízení přístupů a integrace řešení (ačkoliv cloudoví dodavatelé již pracují na jednoduchém provázání a umožnění multicloudové strategie[4]).

2.1.2 Modely nasazení cloudových služeb

Modely nasazení cloudových služeb popisují způsob, jakým jejich uživatel využívá tyto služby. Pokud budeme porovnávat jednotlivé modely s *on-premise nasazením*, jak je vidět na obrázku 2.3, zjistíme, že tyto modely popisují část služeb spravovanou poskytovatelem a samotným uživatelem. Běžně rozlišujeme tyto tři modely cloudových služeb [30]:

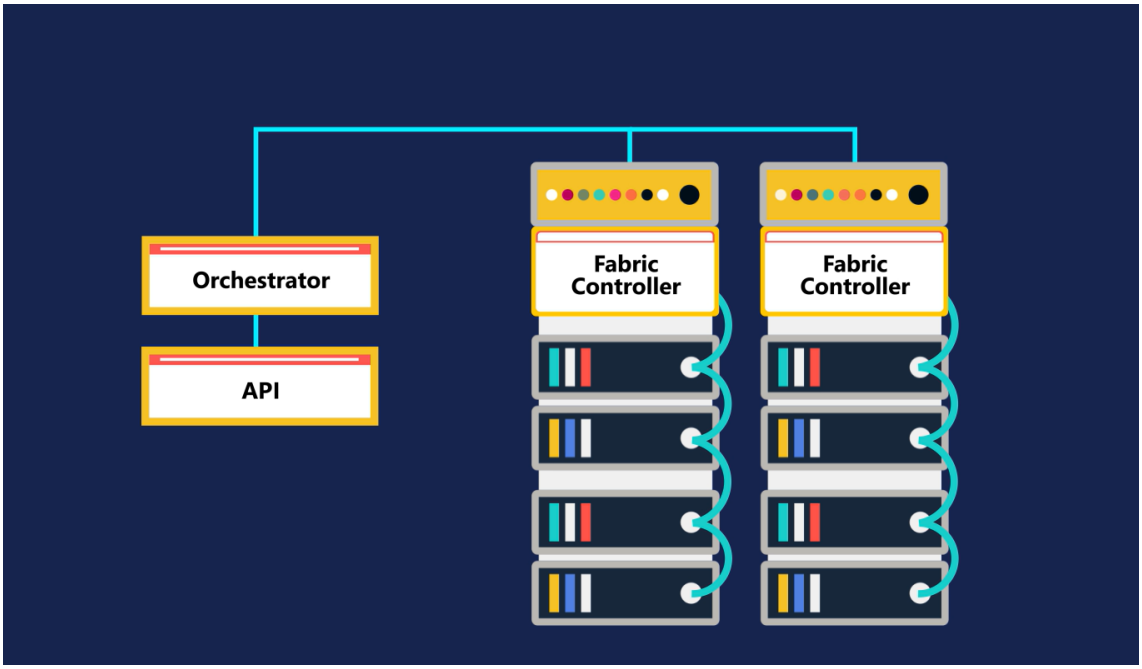
1. infrastruktura jako služba (IaaS),
2. platforma jako služba (PaaS),
3. software jako služba (SaaS).

[3] zastaralý počítačový systém nebo aplikace, které je potřeba z nějakého důvodu udržovat v chodu
[4] https://www.ibm.com/cloud/multicloud-manager



Obrázek 2.3: Spravované části datacentra v závislosti na modelu služby. Převzato z [29].

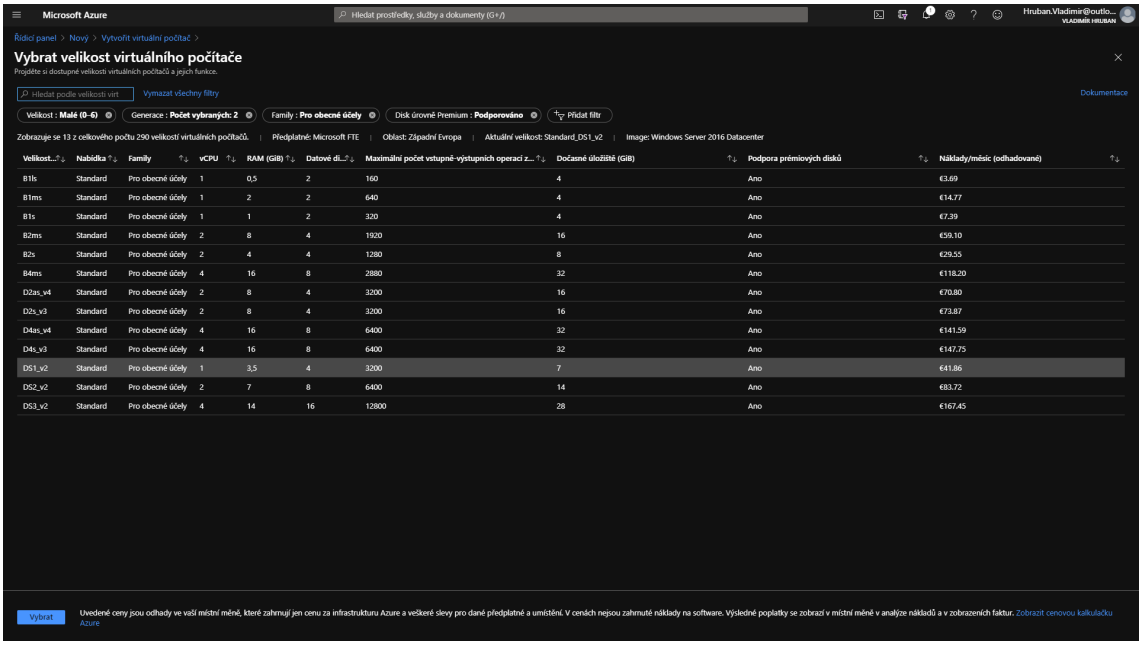
Jednotlivé modely na sebe navazují, pro příklad: pro fungování a běh PaaS i SaaS je samozřejmě potřebná infrastruktura. Modely jako takové poté ovlivňují míru možnosti přizpůsobení, standardizace prostředí, cenu běhu aplikací a komplexitu běhu služby v tomto prostředí. O nasazování prostředků na zdroje (jednotlivé servery v datacentrech) se stará služba, která vyhodnocuje, kde nejlépe prostředek nasadit (od datacentra v zónách a regionech poskytovatele až po jednotlivé serverové racky) tak, aby běžela co nejefektivněji. Například ve veřejném cloudu Microsoft Azure se tato služba nazývá *orchestrátor*, jednotlivé racky pak obsluhuje takzvaný *fabric controller* (viz obrázek 2.4) [18]. K této službě je obvykle (minimálně u majoritních poskytovatelů cloudových služeb) k dispozici API a na něj navázané nástroje (například webové rozhraní) pro vytváření a správu nasazených služeb.



Obrázek 2.4: Schéma orchestrátoru a fabric controlleru z Microsoft Azure. Převzato z [18].

Infrastruktura jako služba

Ačkoliv **infrastruktura jako služba** (anglicky *Infrastructure as a Service* nebo *IaaS*) vyžaduje nejvíce správy ze strany uživatele, je považována za nejflexibilnější model. Poskytovatel cloudových služeb spravuje samotný hardware (i proto se můžeme setkávat i s názvem *Hardware jako služba*) a virtualizaci nad tímto hardwarem pomocí hypervizoru [30]. Na uživateli pak zůstává (krom výběru samotných parametrů virtuálního stroje (viz obrázek 2.5)) volba a nastavení operačního systému (ačkoliv se v nabídkách dodavatelů služeb objevují již předpřipravené obrazy disků se systémy Linux či Windows Server, nebo dokonce hotové virtuály například určené pro práci s velkými daty), správa jeho běhu, správa dat a nasazení aplikací.



Obrázek 2.5: Nasazování virtuálního stroje v prostředí Azure Portal.

Síťové služby zajišťují především připojení serverů a dalších zdrojů, na kterých běží nasazovaná infrastruktura, do sítě datacentra a k internetu, služby DNS a vyvažování zátěže (anglicky *load balancing*) mezi zdroji tak, aby služba běžela hladce a efektivně a také v rámci SLA [30]. Síťové služby se zároveň starají o Quality of Service (QoS). Na základní síťové služby uvnitř datacentra většinou nejsou navázané přímé poplatky, tyto jsou pak započítány v poplatcích za jednotlivé služby poskytovatele.

Mezi další zdroje spravované dodavatelem služby v rámci modelu infrastruktura jako služba patří služby **úložiště**. Tyto umožňují uživatelům užívat krátkodobé i dlouhodobé ukládání dat pro navázané služby, což jim dovolí rychlé, jednoduché a relativně levné škálování (ukládání dat patří mezi nejlevnější cloudové služby), automatickou replikaci a zálohování (a to i v rámci různých datacenter s geografickou redundancí podle možností vendora cloudové služby) a často zjednodušení dodržování legislativních a jiných (například ISO) nařízení, směrnic a standardů zajištěných poskytovatelem služby. Data jsou většinou fyzicky i softwarově chráněna před neoprávněným přístupem a šifrována a procesy poskytovatele jsou nastaveny tak, že k nim nemá přístup ani poskytovatel služby. Výše poplatků za služby úložiště dat jsou obvykle navázány na:

1. typ úložiště (HDD/SSD),
2. jeho rychlost zápisu a čtení dat,
3. frekvenci přístupu k datům,
4. velikosti úložiště a množství příchozích a odchozích dat.

Servery poskytují v rámci služby nezbytný výpočetní výkon, nad kterým jsou poté vytvářeny virtuální zdroje a který je nabízen uživatelům. Uživatelé mají většinou možnost nasadit již předpřipravené šablony virtuálních strojů (často i navázané na konkrétní operační systém) nebo si vytvořit vlastní konfiguraci CPU, RAM, GPU společně s diskovým

polem. Příkladem budiž **Virtual Machines** v rámci veřejného cloudu Microsoft Azure nebo **Amazon EC2** v rámci Amazon Web Services. Cena za tyto zdroje se odvíjí od rychlosti CPU a počtu virtuálních jader procesoru, velikosti RAM, operačního systému (pokud je zajišťován dodavatelem služby), SLA a potřeby redundance. Například u Amazon Web Services a Microsoft Azure je běh účtován za jednotlivé sekundy používání a lze se zavázat k dlouhodobému užívání zdroje, čímž získává uživatel na tento zdroj slevu.

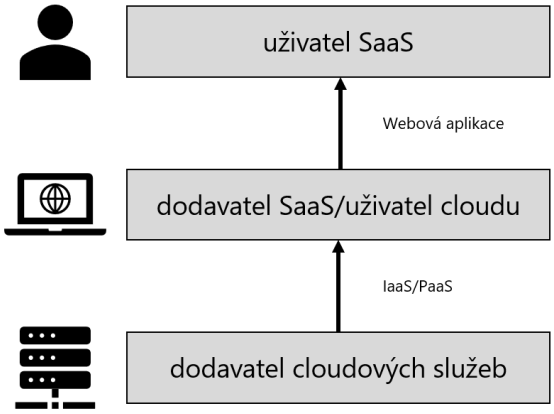
Ačkoliv je model infrastruktura jako služba nejflexibilnějším modelem stran konfigurace služby a zároveň modelem, který nejvíce reflektuje dosavadní fungování IT na vlastních zdrojích (*on-premise*), je pouze vstupní branou do možností cloudu, snižování nákladů na IT a agility v plánování zdrojů. Patří totiž k nejdražším modelům a většina správy zůstává na vlastním IT oddělení a vývojářích.

Software jako služba

Nejméně flexibilním modelem užívání cloudových služeb je model **software jako služba** (anglicky *Software as a Service* nebo *SaaS*). Zde se o veškerý vývoj, správu, údržbu a běh aplikace stará dodavatel SaaS řešení. Toto řešení je typicky provozováno přes internet (ačkoliv existují výjimky – například aplikace Microsoft 365 Apps nebo Intune patří do kategorie software jako služba a fungují i offline) a přes internet jsou také spravovány a aktualizovány. Místo prodeje a údržby mnoha verzí má dodavatel komfort centralizované kontroly verzování aplikace a uživatelům umožňuje přístup kdykoliv a odkudkoliv [2]. Uživateli jsou fakturovány poplatky za konkrétní počet licencí/přístupů nebo za verzi aplikace. Na uživateli povětšinou zůstává jen konfigurace řešení a přiřazení přístupů – jedná se tedy o typ aplikací, které jsou připraveny k bezprostřednímu užívání (anglicky *out of the box experience* a jejich škálování (například přidání více uživatelů, větší úložiště a tak podobně) je jednoduché a rychlé. Na druhou stranu není velká možnost řešení upravovat na míru konkrétnímu zákazníkovi – respektive je možnost odňata z jeho rukou. Typickými příklady řešení na modelu software jako služba jsou:

1. aplikace pro plánování podnikových zdrojů (ERP) – například Microsoft Dynamics 365 nebo NetSuite,
2. aplikace pro ukládání a sdílení souborů – například Box nebo Dropbox,
3. aplikace pro videohovory – například Zoom nebo Microsoft Teams,
4. aplikace pro produktivitu, spolupráci a sdílení – například G Suite nebo Microsoft 365,
5. aplikace pro práci s velkými daty a jejich vyhodnocení – například Tableau nebo PowerBI,
6. účetní systémy – například iDoklad.cz nebo Pohoda.

Poskytovatelé služeb v modelu software jako služba mohou využívat vlastní datacentrum a zdroje nebo využívají jiného dodavatele cloudových služeb jako poskytovatele infrastruktury (v modelu infrastruktura jako služba nebo platforma jako služba) a jsou takto připraveni na rychlé škálování svých aplikací (viz obrázek 2.6) [2].



Obrázek 2.6: Uživatelé a dodavatelé SaaS řešení. Schéma vytvořeno podle [2].

Platforma jako služba

Model poskytování **platforma jako služba** je kompromisem mezi možnostmi *IaaS* a *SaaS*. Uživatel se stará o vývoj aplikací a spravuje jejich data, nicméně nemusí brát v potaz hardware, na kterém tyto aplikace nasazuje stejně jako operační systém, middleware a runtime – pouze vybere vhodný prostředek k dosažení cíle (například volba programovacího jazyka apod.). Toto přináší výhody jako možnou ekonomickou úsporu, zrychlení a zjednodušení vývoje a nasazování aplikací ruku v ruce s nevýhodami jako například omezení možností konfigurace na straně poskytovatele služby.

Příkladem budiž nasazení SQL databáze ve veřejném cloudu Microsoft Azure. Zde máme několik možností, jak SQL databázi nasadit, nicméně pro naše potřeby porovnám dvě databáze založené na enterprise edici SQL Serveru:

1. Azure SQL Database a
2. SQL Server in Azure VM,

kde **Azure SQL Database** je příkladem platformní služby *DBaaS – databáze jako služba* a **SQL Server in Azure VM** je příkladem databáze běžící v modelu infrastruktura jako služba, tedy v rámci virtuálního stroje. Ačkoliv z hlediska administrace je Azure SQL Database volba, která snižuje náklady a investovaný čas administrátora u zákazníka, přináší omezení jako například chybějící podporu *trace flags* nebo *CLR* [18].

Dalším příkladem platformních služeb jsou například *Azure Functions* – prostředek k vytváření skriptů a programů běžících nezávisle na infrastruktuře v bezserverovém módu, případně Google App Engine – platforma pro bezserverový vývoj webových aplikací.

2.1.3 Edge zařízení

Jako *edge zařízení* označujeme zařízení, která jsou součástí cloudové infrastruktury aplikace a služeb, nicméně běží na místní infrastruktuře a je na ně offloadována část výpočetních výkonů [1]. Edge zařízení jsou potom ta zařízení, která jsou umístěna na místní infrastruktuře mezi cloud a například IoT zařízení, vedle těchto zařízení nebo jsou přímo součástí těchto zařízení. Důvod za tímto je vůbec potřebná přítomnost místních zařízení (například IoT zařízení se senzory) ve formě sběračů dat nebo ovladačů, kdy jsou tato data odesílána do cloudové infrastruktury a tam zpracovávána, nebo IoT zařízení provádí instrukce

zaslané z cloudové služby. Druhým důvodem, který navazuje na důvod první, je pak potřeba provádět některé operace přímo na edge zařízení, například kvůli nutnosti snížení latence a potřebě provádění operací v reálném čase [1]. Třetím důvodem může být stálá nebo občasná absence připojení zařízení k síti.

Úkolem edge zařízení je například sbírat, agregovat nebo zpracovávat data a ta pak posílat do cloudu nebo spouštět byznys logiku zařízení nebo být její součástí. Příkladem využití edge zařízení, které spolupracuje s cloudovými službami a místními senzory a které nemůže být neustále připojeno k síti a využívat jen cloudovou infrastrukturu, je třeba kontrola trakčního vedení pomocí bezpilotního dronu s kamerou. **Bez využití edge zařízení** by tento případ vypadal takto: operátor s dronem kontroluje trakční vedení. Toto dělá buď sám vizuálně, případně pouze nahrává obraz a výsledné video je pak zpětně nahráno do cloudu a zpracováno modelem strojového učení, který na trakčním vedení hledá anomálie. Jelikož nelze všude podél trakčního vedení předpokládat kvalitní internetové připojení s nízkou latencí, nepřipadá okamžité zpracování v úvahu. Mezi kontrolou a následným nalezením anomálie na vedení tak uběhne dlouhý časový úsek. **S nasazeným edge zařízením**, na kterém běží vyexportovaný model strojového učení z cloudové služby, kde může být model efektivně natrénován na velkém množství dat s velkým výkonem, pak samotná detekce anomálií probíhá přímo v reálném čase pomocí edge zařízení na dronu.

S využitím edge zařízení s modelem strojového učení na dronu přišli například společnosti DJI a Microsoft Corporation [15]. DJI pak vydalo SDK pro Windows, které právě takovéto případy užití umožňuje.

Dalším příkladem může být edge zařízení, které umožňuje zvýšit zabezpečení místní infrastruktury a komunikace mezi touto infrastrukturou a cloudem. Příkladem tohoto řešení je chip *Azure Sphere*[5].

Na IoT a edge zařízení samozřejmě myslí i dodavatelé cloudových řešení. Příkladem je **AWS IoT Core**, které řeší bilaterální komunikaci mezi zařízeními a cloudem Amazon Web Services[6]. Dalším příkladem je pak platformní služba **Azure IoT Hub**[7] (která taktéž řeší oboustrannou komunikaci) nebo služba **Azure IoT Central**[8], která běží v módu software jako služba – obě v cloudu Microsoft Azure.

Dodavatelé zároveň vydávají již hotový hardware pro edge zařízení, který lze používat a jednoduše napojit na jejich cloudové služby. Například výše zmíněný Azure Sphere, Vision AI Dev Kit[9] nebo Amazon Marvel MW320 IoT Starter Kit.

2.2 Umělá inteligence a strojové učení a jejich nasazení ve veřejných cloudech

Vedle poptávky po přesunu tradičních počítačových služeb do prostředí cloudové infrastruktury a služeb vzrůstá i poptávka podniků po zajištění dalších úkonů s důrazem na automatizaci nasazování a zjednodušení a zrychlení vývoje. Jedněmi z těchto služeb jsou možnosti vývoje a aplikace umělé inteligence a strojového učení a jejich integrace do aplikací. Tyto služby tak nalezneme skrze všechny modely nasazení cloudových služeb – předpřipravené virtuální stroje pro trénování a nasazení modelů strojového učení v rámci modelu infrastruktura jako služba (například **Data Science Virtual Machines** v rámci Microsoft Azure, což

[5] https://azure.microsoft.com/en-us/services/azure-sphere/
[6] https://aws.amazon.com/iot/
[7] https://azure.microsoft.com/en-us/services/iot-hub/
[8] https://azure.microsoft.com/en-us/services/iot-central/
[9] https://azure.github.io/Vision-AI-DevKit-Pages/

jsou připravené šablony pro virtuální stroje s rezervovaným výkonem a předinstalovanými balíčky nástrojů a programovacích jazyků, například Apache Spark, Anaconda Python, Jupyter Notebooks nebo Visual Studio), předpřipravené nástroje v rámci modelu platforma jako služba (jako například **Custom Vision** – součást Microsoft Cognitive Services, která umožňuje jednoduše trénovat modely pro klasifikaci obrazu nebo rozpoznání objektů na obrázku a následně model implementovat do infrastruktury aplikací), nebo dokonce hotové nástroje používané v modelu software jako služba (například **Watson Personality Insights** v rámci IBM Cloud, který umožňuje analyzovat chování a charakterové vlastnosti na základě psaného textu pomocí předtrénovaného modelu).

Ačkoliv v roce 2019 využívala možností automatizovaných služeb umělé inteligence a strojového učení ve svých aplikacích pouze 2 % týmů vývojářů, toto číslo má do roku 2023 vzrůst na 40 % a je předpoklad, že do roku 2025 bude polovina aktivit datových vědců automatizována právě pomocí umělé inteligence, což pomůže s nedostatkem kvalifikovaných pracovníků na tomto vědeckém poli [8].

Na tento předpokládaný nárůst poptávky po službách na poli umělé inteligence a strojového učení reagují i dodavatelé cloudových služeb, kteří nejenže přidávají nové služby, ale snaží se zároveň zjednodušit nasazení momentálně dostupných služeb, a umožnit tak nasazování těchto služeb širšímu publiku. Satya Nadella, Chief Executive Officer ve společnosti Microsoft, tyto aktivity nazývá demokratizací umělé inteligence. Jeho snahou (se skupinou dalších společností, například Adobe) je dostat možnosti těchto služeb k co největšímu množství vývojářů, potažmo uživatelů a zároveň zaktivizovat skupinu, která se na poli informatiky nazývá „citizen developers“ – lidé, kteří využívají takzvaných „low-code/no-code“ principů (vývoj aplikací bez využití nebo s minimálním využitím tradičních programovacích a skriptovacích jazyků) pro vývoj aplikací nebo automatizaci procesů (tzv. RPA – robotická automatizace procesů) pro své specifické osobní nebo vnitropodnikové potřeby. Nástroje umělé inteligence se tak dostávají i do těchto „low-code/no-code“ nástrojů, například do aplikací v rámci platformy **Power Platform**. Nadella popisuje svou misi a misi Microsoftu na poli AI takto:

> „Pevně věřím, že můžeme využít umělou inteligenci k tomu, abychom lidem pomohli dosáhnout čehokoli, o co usilují. Tím také pomáháme zvýšit produktivitu a růst společností.“ [21]

Mezi další velké demokratizátory umělé inteligence patří mimo jiné i společnost Google (potažmo jeho mateřská společnost Alphabet).

Tyto dvě společnosti zároveň patří ještě s Amazon Web Services a IBM podle zprávy auditní společnosti Gartner Inc z února 2020 [8] mezi lídry na trhu v oblasti cloudových AI služeb pro vývojáře (viz obrázek 2.7). Tato zpráva v hodnocení kombinuje vizi a exekuci této vize dodavatelem, mezi hodnocené služby patří zpracování textu, počítačového vidění a automatizované strojové učení, které by měly být dostupné přes API a nevyžadují velké zkušenosti vývojářů na poli datové analytiky. Nejlépe se zde umístila společnost Amazon Web Services, nejhůře pak čínská společnost Tencent. V následujícím textu se budu krátce věnovat právě nabídce čtyřky lídrů v této oblasti.

Obrázek 2.7: Gartner Magic Quadrant for Cloud AI Developer Services – převzato z [8].

2.2.1 Amazon Web Services

Amazon Web Services, jakožto dodavatel cloudových řešení s aktuálně největším tržním podílem na trhu [8], samozřejmě do cloudových služeb pro strojové učení silně zasahuje. Mezi předtrénované inteligentní služby patří API pro doporučování zboží zákazníkům (Amazon Personalize), analýza dokumentů a textu (Amazon Textract a Amazon Comprehend), speech-to-text (Amazon Transcribe) a především pro tuto práci důležité rozpoznávání objektů na obrázcích a videích a jejich klasifikace – **Amazon Rekognition**.

Mezi klíčové vlastnosti **Amazon Rekognition**[10] patří identifikace předtrénovaných objektů, vytváření vlastních klasifikátorů, detekce textu, tváří a dalších.

Vlajkovým produktem je pak **Amazon SageMaker**[11], což je nástroj na rychlé vytváření, trénování, nasazení a škálování modelů strojového učení. Zároveň s ním nabízí Amazon IDE (Integrated Development Environment neboli vývojové prostředí) pro strojové učení **Amazon SageMaker Studio** a automatizační nástroj **Amazon SageMaker Autopilot**.

[10] https://aws.amazon.com/rekognition/
[11] https://aws.amazon.com/sagemaker/

2.2.2 Microsoft

Za klíčové oblasti inteligentních služeb na svém cloudu Microsoft Azure považuje Microsoft tyto tři:

1. strojové učení,
2. dolování dat ze znalostní báze,
3. integrace AI do aplikací a agentů.

Mezi nástroje strojového učení spadá **Azure Machine Learning** nebo také **Azure Machine Learning Studio**. Tyto nástroje umožňují spravovat kompletní životní cyklus modelů strojového učení od návrhu, programování, experimentování po nasazení do produkce (včetně DevOps řešení, které se nazývá **MLOps**). Azure Machine Learning a Azure Machine Learning Studio zároveň umožňuje využití řady programovacích jazyků, nástrojů a frameworků (například *R*, *Python*, *MLflow*, *TensorFlow* ...). Zároveň jsou na tato řešení navázány integrace do dalších služeb a produktů (Power BI, Excel, SQL Server ...). Rovněž klade Microsoft velký důraz na řešení bezpečnosti a splnění regulatorních požadavků.

K nástrojům pro dolování dat ze znalostní báze – zde je vlajkovou lodí Microsftu **Azure Cognitive Search**, který kombinuje možnosti počítačového zpracování přirozeného jazyka, extrakce informací z obrázků, indexace obsahu a hledání vzorů a souvislostí mezi informacemi. Toto pak pomáhá vývojářům integrovat do svých aplikací chytré vyhledávání. Využívat k tomu mohou .NET SDK nebo REST API.

Pro integraci inteligentních služeb do aplikací mohou vývojáři také využívat **Cognitive Services** a **Azure Bot Service**. Azure Cognitive Services je set služeb a API, které pokrývají rozhodovací procesy, zpracování přirozeného jazyka, řeči, vyhledávání na internetu a také počítačové vidění – zde mohou vývojáři používat jak předtrénované modely a klasifikátory, tak modely sami natrénovat. Azure Bot Service poté umožňuje jednoduché vytváření konverzačních botů a jejich nasazení na komunikační kanály (přes web, aplikace nebo už hotová řešení jako Microsoft Teams, Slack, Facebook Messenger...).

2.2.3 Google

Google jakožto číslo tři podle Gartner Inc [8] rozděluje své AI produkty do tří kategorií [10]:

1. AI Hub,
2. AI building blocks,
3. AI Platform.

AI Hub slouží především jako repozitář řešení pro experimentování s AI na Google Cloud. Jedná se hlavně o experimenty nad daty, jejich analýzou, transformací a prezentací. K tomu jsou využívány modely TensorFlow nebo předpřipravené virtuální stroje na strojové učení.

AI building blocks jsou nástroje, které mohou vývojáři použít ve svých aplikacích, aby je tak obohatili o inteligentní služby. Jedná se o nástroje spojené s počítačovým viděním (**Vision AI** a **VideoAI**), zpracováním přirozeného jazyku (**Natural Language** a **Dialogflow**), prací s daty (**AutoML Tables** nebo **BigQuery ML**) a automatizovaným strojovým učením (nástroj **Cloud AutoML**).

AI Platform je pak celková platforma pro přípravu, běh, management a sdílení AI projektů, která využívá jak nástroje popsané výše, tak další nástroje a služby jako **Kubeflow** (open-source framework od Google), **Data Labeling Service** a další. Výsledné aplikace lze spustit přímo na Google Cloud nebo vyexportované na místní infrastruktuře.

2.2.4 IBM

Ačkoliv má společnost IBM z výše zmíněných dodavatelů cloudových služeb nejmenší tržní podíl (jen necelá 2 % [9]), je velmi aktivní na poli inteligentních služeb v cloudu a do rozvoje svých AI služeb plánuje v následujících deseti letech investovat více než dvě miliardy amerických dolarů [12].

IBM svou sadu AI cloudových služeb udržuje pod značkou IBM Watson[12] (odkazující na jednoho z nejvýznamnějších výkonných ředitelů IBM Thomase J. Watsona Sr.). Tato řešení jsou rozdělena do čtyř sekcí:

1. řízení životního cyklu AI,
2. již hotové aplikace Watson,
3. Watson APIs,
4. řešení Watson pro konkrétní vertikály.

Sekce řízení životního cyklu AI obsahuje aplikace pro tvorbu a automatizaci správy aplikací obsahujících AI a strojové učení. Patří mezi ně například produkty **Watson Studion** a **Watson Machine Learning**.

Mezi aplikace Watson patří řešení software jako service, již hotová řešení a natrénované modely, například **Watson Knowledge Studio** nebo **IBM OpenPages with Watson**.

Watson APIs obsahují platformní služby pro práci s textem a počítačovým viděním (**Watson Visual Recognition**, které umožňuje klasifikaci obrázku a práce s objekty na nich). Jsou to však právě služby pro práci s textem a služby pro zpracování přirozeného jazyka, kde je IBM podle svých uživatelů nejsilnější. Nicméně, jelikož jsou tyto služby vyvíjeny různými divizemi a týmy, jsou často nekonzistentní (z hlediska integrace služeb mezi sebe nebo například zpětné kompatibility [8].

IBM nabízí komplexní řešení pro různé firemní vertikály, například pro firmy operující ve finančnictví, IT službách nebo v odvětví péče o zákazníky. Tyto pak obsahují konkrétní služby a nasazení nejvhodnější pro dané odvětví.

2.2.5 Umělé neuronové sítě a jejich použití pro detekci objektů

Umělá neuronová síť je sítí umělých neuronů (tyto jsou inspirovány v biologii). Jednotlivé neurony jsou propojeny v síti a transformují jimi procházející signály pomocí matematické funkce. Tato transformace je ovlivňována pomocí vah jednotlivých vstupů do funkce. Tyto váhy jsou upravovány právě při učení neuronové sítě [5].

Pro detekci vozidel ve statickém obraze se nejčastěji používají konvuluční neuronové sítě (dále CNN), které se také nejčastěji používají pro rozpoznávání objektů. Pro potřeby klasifikace objektu pomocí CNN je nejdříve třeba vymezit regiony pomocí takzvaných bounding boxů. Tyto nám vymezují jednotlivé objekty na obrázku, které jsou dále klasifikovány. Pro optimální výběr těchto regionů ke klasifikaci slouží různé algoritmy. Pro příklad uvádím

[12] https://www.ibm.com/watson/products-services

metodu *YOLO* [26], tedy *You Only Look Once*. V rámci této metody je vstupní obraz rozdělen do mřížky $S \times S$ a následně je v rámci mřížky vygenerováno m bounding boxů a ty jsou poté klasifikovány pomocí CNN. Regiony oskórované CNN nad určenou hodnotu pak reprezentují detekované objekty [14].

Kapitola 3

Implementace

V této kapitole popisuji návrh dvou architektur, se kterými v rámci návrhu implementace pracuji (podle umístění modelu strojového učení v rámci architektury) – v prvním návrhu architektury běží natrénovaný model strojového učení v prostředí veřejného cloudu Microsoft Azure (kde je umístěno i řízení běhu aplikace). V druhém poté vyexportovaný model běží offline v rámci edge zařízení a cloud je pouze v roli orchestrátoru toku mezi tímto zařízením a uživatelem (kdy řídí dotazy a běh dat). Následně popisuji samotné nasazení a konkrétní řízení běhu dotazů a uživatelskou webovou aplikaci.

Back-end aplikace využívá pouze veřejný cloud – konkrétně Microsoft Azure – především služby v modelu nasazení platforma jako služba (včetně nasazení databází) a jedno edge zařízení. Tento model jsem vybral, abych minimalizoval potřeby konfigurace virtuálních zdrojů, operačního systému a jejich následné síťování a zároveň abych zajistil jednoduchou rozšiřitelnost a modulárnost tohoto systému. Jako příklad modulárnosti a možnosti jednoduchého doplňování nových funkcí jsem přidal funkci logování jednotlivých stavů parkoviště do databáze, na kterou poté může být navázáno BI řešení nebo další možnosti strojového učení, například predikce obsazenosti parkoviště (viz sekce 3.6). Ačkoliv počítám s rozšířeními, která také využívají možnosti bezserverového běhu aplikací, ke službě lze připojit i jinak běžící programy bez závislosti na modelu nasazení či konkrétním programovacím jazyku – služba je, jak budu popisovat níže, obsluhována pomocí HTTP požadavků (tak jak jsou popsány v RFC 7231[1]).

Uživatel přistupuje k webové aplikaci, která je plně responsivní a díky tomu pohodlná k používání jak na počítačích, tak na tabletech a mobilních zařízeních. Tato je pro potřeby testování této práce spuštěna v lokálním prostředí (kvůli minimalizaci provozních nákladů na testování), nicméně by stejně jako zbytek práce mohla být nasazena v modelu platforma jako služba na Azure coby Web App[2]. K využití webového rozhraní jsem přistoupil především proto, abych jednoduše zajistil dostupnost na různých platformách a typech zařízení. Oproti původně zamýšlenému využití multiplatformní technologie Xamarin[3] pro vývoj uživatelské aplikace je zde také výhoda rychlého vývoje a prototypování vedle odpadnutí nutnosti instalace aplikace do vlastního zařízení.

Edge zařízení je pro potřeby této práce simulováno pomocí jedné funkce v Azure Functions (viz sekce 3.3.2) jako jedna z nasazených funkcí, nicméně dále v této práci popisuji požadavky na takovéto zařízení pro potřeby nasazení.

[1] https://tools.ietf.org/html/rfc7230#section-3.1.1
[2] https://azure.microsoft.com/en-us/services/app-service/web/
[3] https://dotnet.microsoft.com/apps/xamarin

Prostředí a služby veřejného cloudu Microsoft Azure jsem vybral, jelikož má jednu z největších nabídek služeb v modelu platforma jako služba, ale také jelikož jeho možnosti a vhodné služby znám nejlépe z nabídky dostupných dodavatelů cloudových služeb.

Veškerý běh služeb v prostředí Microsoft Azure je hrazen z kreditů, které jsem získal z předplatného Visual Studio Enterprise[4] jako aktivní člen programu Microsoft Learn Student Ambassadors[5] a slouží k testování technologií.

Při implementaci výsledného řešení jsem vycházel ze zdrojů [18, 20, 23, 22, 7, 25, 3, 24].

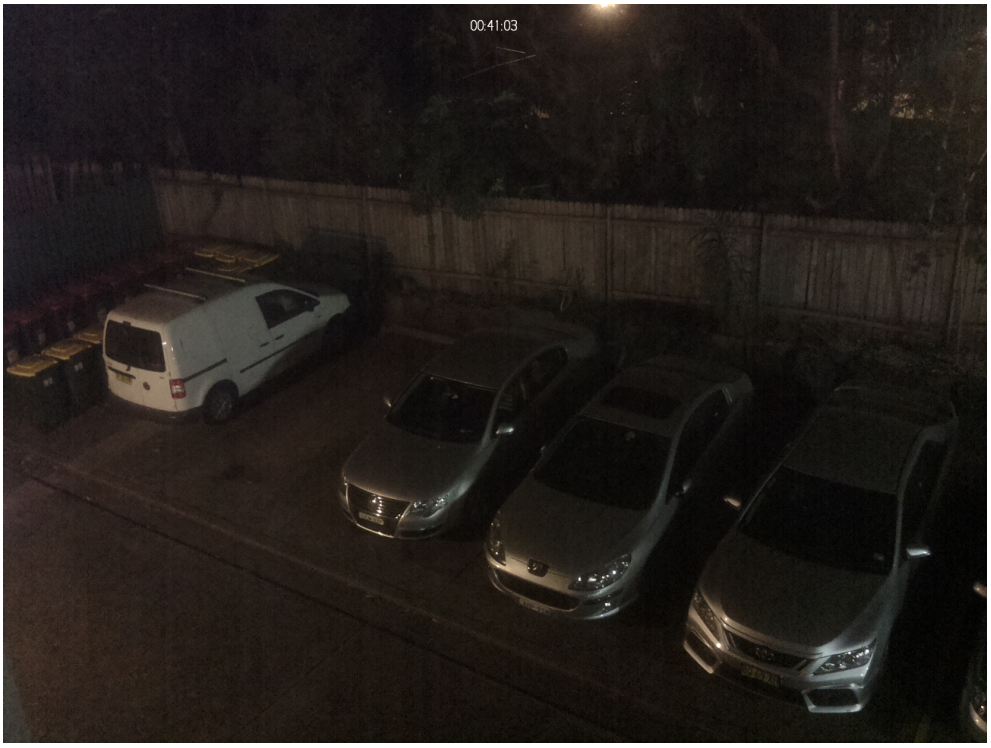
3.1 Datová sada

Pro natrénování modelu jsem využil datovou sadu **Find a Car Park** od Davida Carra z platformy **kaggle** [6], která obsahuje **3 262** obrázků formátu JPEG v rozlišení 1 296×972 pixelů. Tyto obrázky obsahují záběry na parkoviště s různými automobily za různých podmínek (viz obrázky 3.1 a 3.2). Jsou zastoupeny záběry za běžného denního světla, v noci, kamera mění pozici, v kameře nebo na sklech automobilů se objevují odlesky. Na záběrech se zároveň objevují další cizí objekty – například popelnice, plot, flóra. Do záznamu je rovněž uměle přidán čas pořízení fotografie (ve formátu HH:MM:SS).

Obrázek 3.1: Záběr na plné parkoviště. Převzato z [6].

Fotografie jsou roztříděny do dvou složek – plné parkoviště (viz obrázek 3.1) a parkoviště s různým počtem parkovacích míst (viz obrázek 3.2). Z toho záběrů na plné parkoviště je **2 195** a záběrů na parkoviště s volnými parkovacími místy je **1 067**.

[4] https://visualstudio.microsoft.com/vs/enterprise/
[5] https://studentambassadors.microsoft.com/



Obrázek 3.2: Noční záběr na parkoviště s volnými místy. Převzato z [6].

3.2 Použité technologie

3.2.1 Nástroje pro vývoj

Pro vývoj jsem používal výhradně Visual Studio Code, což je open-source editor kódu od společnosti Microsoft Corporation, s doplňky pro vývoj na cloudové platformě Microsoft Azure a pro vývoj webových aplikací, konkrétně:

1. Azure Account (v0.8.11) – doplněk na přihlašování a správu subskripcí na Microsoft Azure,
2. Azure Functions (v0.22.1) – doplněk pro rychlé vytváření a nasazení Azure Functions,
3. Live Server (v5.6.1) – doplněk pro spuštění lokálního serveru pro vývoj a debugging webových stránek,
4. Sass/Less/Typescript/Jade/Pug Compile Her (v6.8.62) – kompilační nástroj pro jazyky k vývoji webových aplikací.

Pro práci s Custom Vision modelem jsem poté používal webové rozhraní služby a dostupné API.

Jako použité jazyky jsem zvolil skriptovací jazyk JavaScript (a runtime Node.js verze 12.16.2), který využívám jak pro byznys logiku webové aplikace, tak pro vytváření Azure Functions. Dále značkovací jazyk HTML (kompilovaný z Pug.js) a jazyk CSS (respektive jeho rozšíření Sass) pro vytvoření a úpravu uživatelského rozhraní webové aplikace.

Používaná databáze je MySQL nasazená ve veřejném cloudu Microsoft Azure a pro přístup k ní využívám MySQL WorkBench 8.0 CE společnosti Oracle Corporation.

3.2.2 Datový model

Pro potřeby back-endu aplikace využívám jednu databázi *parking* se dvěma tabulkami – *location* a *log*.

Tabulka *location* obsahuje záznamy o jednotlivých lokacích. Sloupce, jež tabulka obsahuje, jsou: **id**, které obsahuje unikátní ID lokace a je zároveň primárním klíčem tabulky, **name**, které obsahuje pojmenování lokace v přirozeném jazyce; **source**, které v této verzi aplikace ukazuje na fotografii z parkoviště, jinak URI edge zařízení pro dotazování, **gps** s GPS souřadnicemi parkoviště a **spots_count** s celkovým počtem parkovacích míst na daném parkovišti.

Tabulka *log* obsahuje logy z jednotlivých volání back-endu ve sloupcích: **timestamp** – časové razítko, které je zároveň primárním klíčem tabulky; **location_id**, který odkazuje na ID parkoviště, které bylo požadováno z aplikace, a který je zároveň cizím klíčem pro tabulku *location*; **spots_taken**, který obsahuje počet zabraných míst na parkovišti v danou chvíli tak, jak byl vypočten cloudovou službou.

3.2.3 Cognitive Services

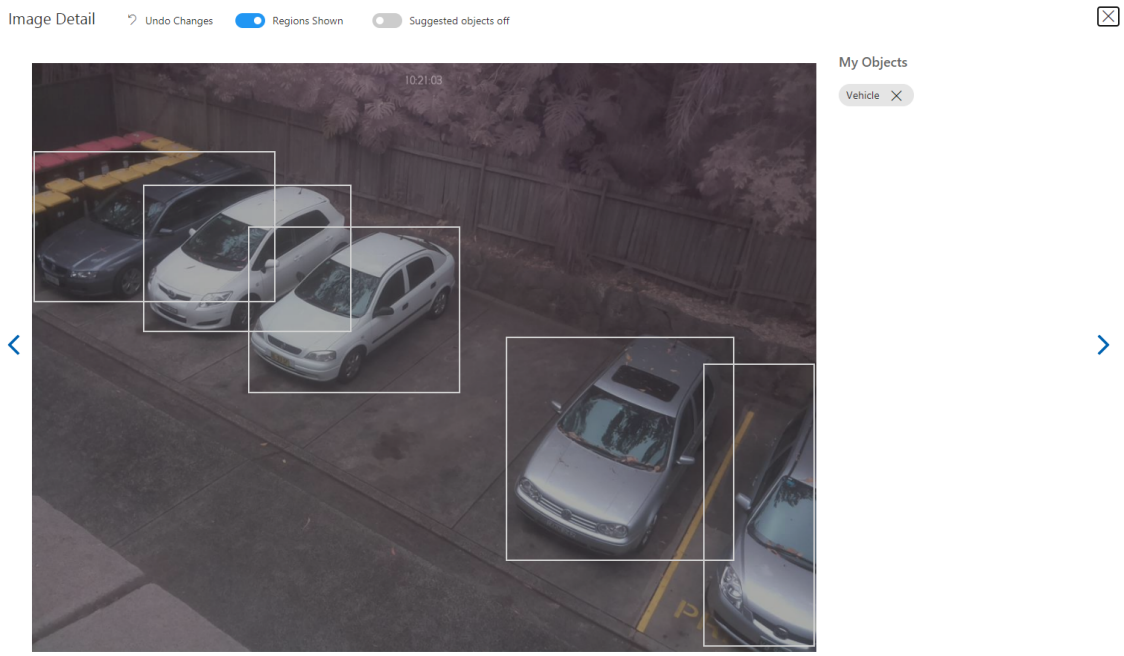
Pro potřeby detekce automobilů na snímcích používám službu **Custom Vision**, která je součástí **Azure Cognitive Services** – AI služeb ve veřejném cloudu Microsoft Azure. Tato služba umožňuje klasifikovat obrázky do kategorií nebo na nich detekovat různé objekty (pro které uživatel vytváří klasifikátory pro trénování modelu). V práci využívám obě možnosti, nicméně ve finálně implementované architektuře používám detekci objektů na obrázku, abych získal přesný počet plných (a potažmo pomocí výpočtu prázdných) parkovacích míst.

Detekce objektů

Můj projekt pro detekci objektů je založen v doméně *General (compact)*, což značí, že není předurčen k rozpoznávání určitého předtrénovovaného druhu objektu (například loga) a že natrénovaný model lze vyexportovat pro běh na lokálním zařízení. Export je možný na platformy Tensorflow, CoreML, ONNX (druhou možností by byla sada Vision AI Dev Kit[6], což je předpřipravený hardware pro běh vyexportovaného modelu včetně vhodné kamery).

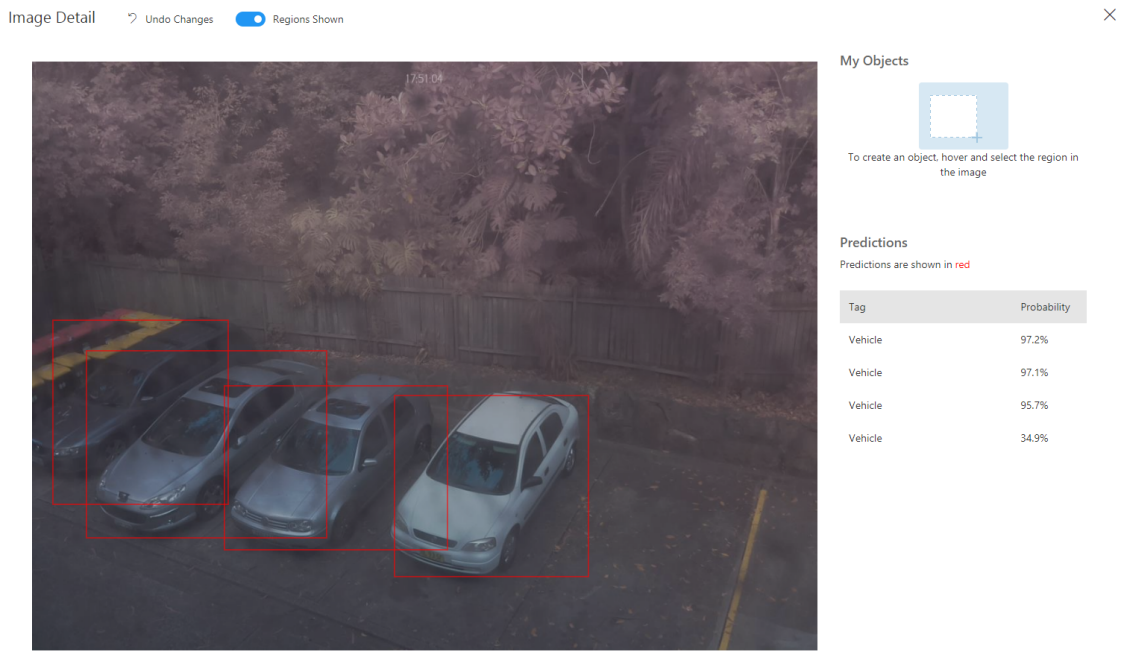
Pro nahrání a otagování záznamů a natrénování modelu jsem využil webové rozhraní nástroje Custom Vision (viz obrázek 3.3). V projektu je nahráno 649 trénovacích obrázků (z kategorií plné i prázdné parkoviště), používám jeden tag – *Vehicle*.

[6] https://azure.github.io/Vision-AI-DevKit-Pages/



Obrázek 3.3: Rozhraní pro tagování obrázků v nástroji Custom Vision.

Model při dotazování API vrací JSON objekt obsahující jednotlivé nalezené objekty na obrázku, oblast obrázku, ve které se každý jednotlivý objekt nachází (respektive souřadnice rohů ohraničení) a pravděpodobnost, s jakou podle modelu patří objekt do dané kategorie. Dotazovat lze rovněž přes webové rozhraní nástroje Custom Vision (viz obrázek 3.4).



Obrázek 3.4: Detekované objekty na obrázku v nástroji Custom Vision.

Klasifikace

Projekt je založen v doméně *General (compact)* s typem klasifikace *Multiclass*, který dovoluje přiřadit každému obrázku pouze jednu třídu. Pro potřeby trénování jsem nahrál a oklasifikoval 2 369 záznamů a tyto rozdělil do dvou tříd – plné (1 395 fotografií) a „prázdné“ (974 fotografií) parkoviště. Po natrénování dosahuje model těchto metrik:

1. precision: 95,6 %,
2. recall: 95,6 %,

kde **precision** je definováno jako

$$precision = \frac{TP}{TP + FP} \tag{3.1}$$

a **recall** jako

$$recall = \frac{TP}{TP + FN}, \tag{3.2}$$

kde *TP* značí *True Positive* (tedy počet predikcí, kdy model oklasifikuje objekt a tato klasifikace je správná), *FP* značí *False Positive* (tedy počet predikcí, kdy model oklasifikuje objekt a toto je špatně) a *FN* značí *False Negative* (tedy počet predikcí, kdy model neoklasifikuje objekt, který měl správně oklasifikovat) [5].

V mé práci může být tento model využit pro jednoduché dotazování – Je na parkovišti nějaké volné místo, nebo ne? Ačkoliv by tento model byl snazší na implementaci a údržbu, neposkytuje uživatelům takový komfort, konkrétně informace o tom, kolik je ve skutečnosti na parkovišti volných parkovacích míst. Tento model mám tedy natrénovaný k dispozici dle parametrů výše, nicméně ve finální architektuře není implementován.

3.3 Architektura řešení

Architektura řešení je rozdělena na tři části, a to:

1. uživatelská aplikace,
2. služby běžící v cloudu,
3. edge zařízení.

Jádrem mého řešení je cloudová infrastruktura, která řídí službu a je vystavěna pomocí platformních služeb, konkrétně řízení je řešeno pomocí **Azure Functions**, jimiž jsou vystavěny takzvané mikroslužby, které dohromady tvoří řídící jádro služby. O tomto jádru nadále budu referovat jako o orchestrátoru.

Mikroslužby jsou zde vhodným řešením, jelikož usnadňují škálování, jsou decentralizované, zjednodušují správu kódu a umožňují snadnou rozšiřitelnost aplikace. Azure Functions zároveň fungují v módu pay-per-run, tedy platíme pouze za spuštění služby, nikoliv kontinuální běh. Tento bezserverový vývoj mi zároveň umožnil, abych se mohl soustředit na vlastní byznys logiku aplikace bez nutnosti starat se o vlastní infrastrukturu.

Součástí infrastruktury jsou zároveň dvě služby pro uchovávání dat – Blob storage pro práci s nestrukturovanými daty a objekty a Azure Database for MySQL server, kde je jednak uchováván seznam parkovišť (společně s údaji o tomto parkovišti) a také záznamy z jednotlivých spouštění.

Všechny tyto služby jsou vytvořeny v lokaci západní Evropa (určuje lokaci datacenter), která je momentálně České republice geograficky nejblíže do otevření datacentra Microsoftu v Polsku během následujících sedmi let [17], aby tak byla zajištěna lokálním uživatelům nejnižší možná latence.

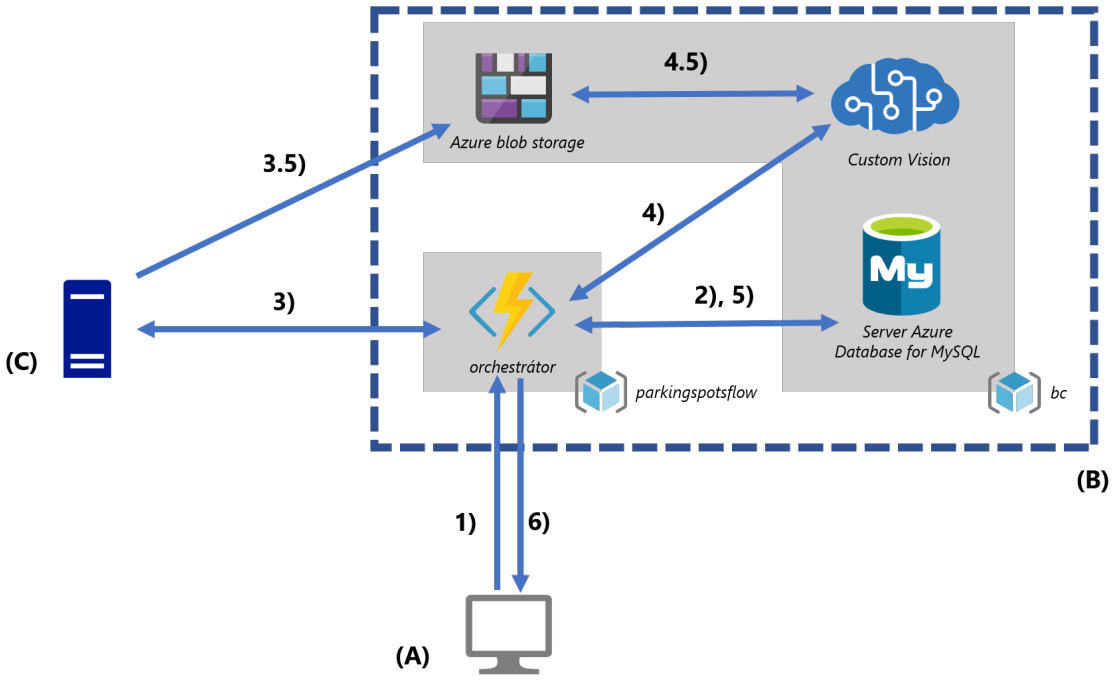
Vedle toho pak leží webová aplikace, která je jediným místem, kde uživatel může interagovat se službami na back-endu a zároveň jediným místem, odkud lze spustit celý běh, což zvyšuje úroveň zabezpečení služby a dává stupeň ochrany proti případným útokům mimo další zabezpečení už implementovaného v samotném systému dodavatele.

Edge zařízení (ačkoliv je zde simulováno pomocí další Azure Function; viz sekce 3.3.2) je obsluhováno a dotazováno z orchestrátoru a jeho úkol a funkce se liší podle architektury. V cloud-centrické architektuře (viz sekce 3.3.1) plní pouze funkci „dodavatele“ fotografie z parkoviště, kterou zachytí a nahraje do Azure blob storage, v architektuře s edge zařízením (viz sekce 3.3.3). Zároveň na tomto zařízení rovnou proběhne zpracování tohoto snímku ve vyexportovaném modelu umělé neuronové sítě.

3.3.1 Architektura 1 – cloud-centrická

V rámci této architektury běží maximum služeb ve veřejném cloudu Microsoft Azure. Ten se stará o řízení celé služby a obsloužení dotazů připojených zařízení. V následujícím textu popisuji jednotlivé části a komponenty. Tato architektura je zároveň architektura implementovaná v konečném řešení této práce (se simulovaným edge zařízením).

Cloud-centrická architektura aplikace pak vypadá následovně:



Obrázek 3.5: Architektura 1 – cloud-centrická. Pro vytvoření schématu byly použity zdroje z Microsoft knihovny symbolů [16].

Sekce (A) znázorňuje *uživatelskou aplikaci*, sekce (B) je *cloudový back-end* a sekce (C) pak *edge zařízení*.

V části (B) jsou také znázorněny nasazené *skupiny prostředků bc* a *parkingspotsflow*, což jsou logické skupiny prostředků, které v rámci Azure subskripce spravuji. Skupina prostředků *parkingspotsflow* obsahuje všechny Azure Functions a další prostředky potřebné k jejich běhu, správě a analýze – tedy Applications Insights, účet úložiště pro logování a plán služby App Service (což je definice prostředků, které jsou využívány danou App Service, v tomto případě Azure Functions). Skupina prostředků *bc* pak obsahuje služby Cognitive Services (potažmo Custom Vision), Azure blob storage pro nahrávání fotografií z parkoviště, Server Azure Database for MySQL, kde jsou strukturovaná data potřebná pro běh, dále pak potřebné další prostředky pro správu výše popsaných služeb.

Edge zařízení v sekci (C) může být jakékoliv zařízení se stálým připojením na internet, fotoaparátem nebo webkamerou a operačním systémem Linux nebo Windows Server. Jeho konfigurace závisí na počtu obsluhovaných dotazů. Zařízení je třeba konfigurovat pro dlouhý bezobslužný běh.

Běh aplikace dle schématu na obrázku 3.5 je potom následující:

1) uživatel v uživatelské webové aplikaci vybírá lokaci a kliká na tlačítko `CHECK`. Webová aplikace (dále *klient*) posílá HTTP POST požadavek na URL orchestrátoru (obsahující mimo jiné identifikátor dotazované lokace) a tím startuje tento orchestrátor (jakožto obalující Azure Function, která pouští a obsluhuje podružné služby), poté požadavek klienta dostává unikátní ID a toto je vráceno společně se stavovým kódem HTTP 202. Klient posílá pravidelně HTTP GET spolu s ID požadavku a po vyřízení dostává zpátky stavový kód HTTP 202.

2) Orchestrátor se připojuje přes MySQL API na databázový server MySQL, kde v tabulce *parking* na základě identifikátoru lokace dotazované klientem vybírá záznam obsahující informace o této lokaci a tyto navrací volající funkci orchestrátoru. Ty obsahují mimo jiné URI edge zařízení na požadovaném parkovišti.

3) Orchestrátor posílá HTTP GET požadavek na edge zařízení, které pořizuje fotografii parkoviště, tuto potom pojmenuje standardizovaným způsobem (kvůli dalšímu procesování) – `img-uuid.jpg` – a nahrává ji na Azure blob storage (krok **3.5**). Orchestrátoru vrací identifikátor nahraného obrázku (v tuto chvíli již *blobu*).

4) Orchestrátor posílá HTTP POST okamžitý požadavek s odkazem na blob s nahranou fotografií parkoviště na API služby Custom Vision (spolu s dalšími požadovanými parametry), která si pro obrázek sahá do Azure blob storage (krok **4.5**), zpracovává ho a vrací odpověď ve formátu JSON spolu se stavovým kódem HTTP 200, tato odpověď obsahuje otagované predikce modelu. Orchestrátor potom tuto odpověď zpracovává a vypočítává množství nalezených automobilů na parkovišti.

5) Orchestrátor se přes MySQL API připojuje na databázový server MySQL, kde do tabulky *log* zadává záznam obsahující časové razítko, ID lokace a počet zabraných míst na parkovišti lokace.

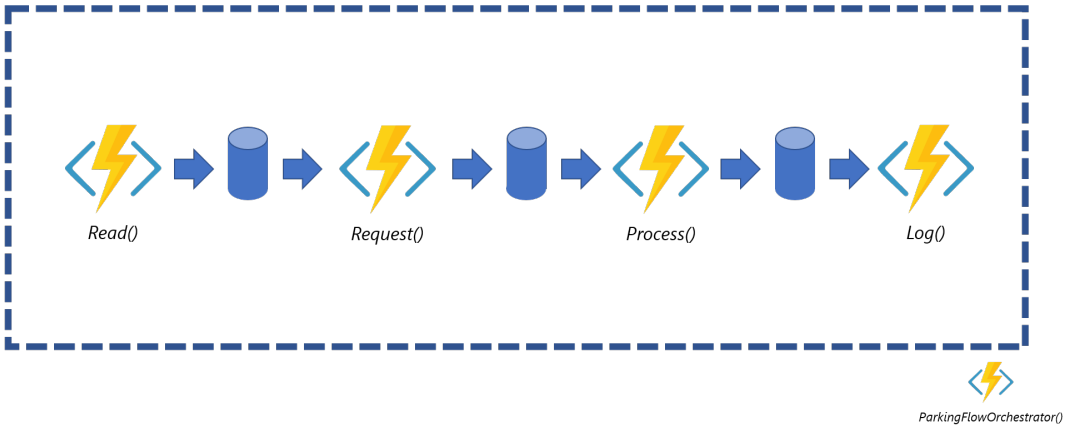
6) Klient dostává v odpovědi stavový kód 200 společně s odpovědí orchestrátoru, která obsahuje počet volných míst na lokaci. Tento počet volných míst je poté uživateli zobrazen.

3.3.2 Orchestrátor

Jak jsem již nastínil výše, orchestrátorem mám v kontextu své architektury na mysli řídící funkci cloudového back-endu. Tento jsem ve svém nasazení implementoval pomocí Azure Functions v jazyce JavaScript s rozšířením *Durable Functions*, které umožňuje volat a řídit běh více Azure Functions v rámci jednoho běhu tak, aby si mezi sebou dokázaly předávat stav a informace.

Při implementaci svého orchestrátoru jsem využil návrhový vzor Durable Functions *řetězení funkcí* [18], kdy je výstup a stav předcházející funkce vstupem funkce následující. Schéma orchestrátoru naleznete na obrázku 3.6.

Triggerem (neboli spouštěčem) orchestrátoru je HTTP POST požadavek na funkci *ParkingFlowStarter*, která následně přímo přes integrace v Microsoft Azure spouští samotnou funkci orchestrátoru – *ParkingFlowOrchestrator*. Požadavky HTTP POST na spuštění orchestrace jsou přijímány pouze z určených adres pomocí nastavení Azure Functions CORS (sdílení prostředků mezi zdroji).



Obrázek 3.6: Schéma orchestrátoru architektury 1. Pro vytvoření schématu byly použity zdroje z Microsoft knihovny symbolů [16].

Typy, vazby a triggery funkcí jsou určeny pro každou funkci v souboru `function.json`. Byznys logika funkcí je pak vždy v souboru `index.js`.

Azure Functions

Azure Functions jsou platformní službou cloudu Microsoft Azure, která umožňuje spouštět uzavřené bloky kódu bez nutnosti manuálně spravovat infrastrukturu, na které výsledný program běží. Funkce jsou pouze nasazeny do platformního prostředí a je nastaven jejich spouštěč (neboli trigger). Tento trigger může být manuální spuštění funkce, časová spoušť, která funkce spouští jednou za daný časový interval, nebo událost (například HTTP požadavek nebo volání funkce z jiné navázané funkce. Poslední zmíněné používám ve své architektuře).

Funkce mají definované jméno, vstup a výstup a možné spouštěče v konfiguračním souboru (v notaci json).

Platba probíhá pouze za reálný čas a počet běhů funkce.

Durable Functions

Durable Functions je rozšíření Azure Functions, které umožňuje běh funkcí Azure Functions ve stateful módu (tedy funkce umí udržovat stav) i přes bezserverový běh a díky tomu umožňuje orchestraci a také asynchronní běh. Rozšíření automaticky udržuje stav funkce, pozastavování a spouštění a případně restart běhu.

Jednotlivé funkce

Pro orchestraci back-endu aplikace využívám šest funkcí:

1. ParkingFlowStarter,
2. ParkingFlowOrchestrator,
3. Read,
4. Request,
5. Process,
6. Log.

ParkingFlowStarter je jediná funkce, která je dostupná z internetu, konkrétně pomocí triggeru HTTP požadavkem. Funkce dostává dotaz na požadovanou lokaci a spouští orchestraci, které přiděluje ID.

ParkingFlowOrchestrator pak již zajišťuje samotnou orchestraci. Vytváří stavové proměnné pro zbytek funkcí a následně jednotlivé funkce volá a předává mezi nimi stav.

Funkce **Read** má za úkol číst z MySQL databáze (přihlašovací údaje jsou hardcoded přímo v zdrojovém kódu, nicméně tento běží pouze v back-endu a není přístupný mimo cloud, tudíž tento přístup nepovažuji za bezpečnostní riziko) z tabulky location a vrací iniciále požadované lokace.

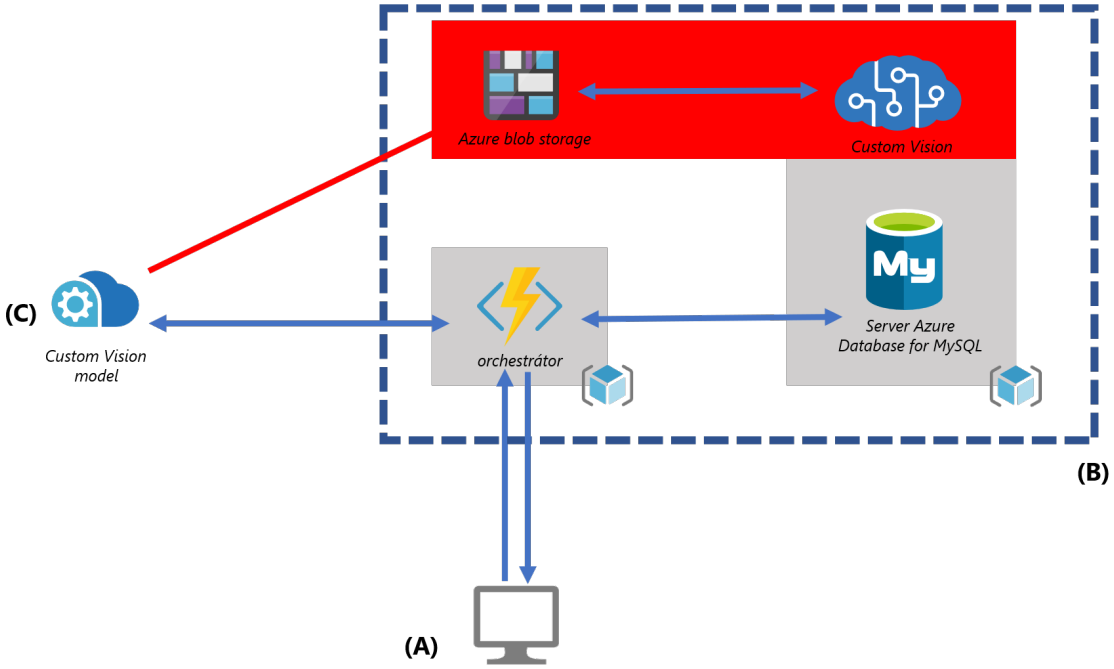
Funkce **Request** volá edge zařízení, vytváří blob z dodané fotografie a nahrává tento blob do kontejneru *parking01* na Azure blob storage. Aby bylo zajištěno unikátní pojmenování každého záznamu, vytváří funkce pro každý záznam unikátní pojmenování pomocí uuid. Vrací pak název blobu.

Funkce **Process** se stará o poslání záznamu na Custom Vision API pro klasifikaci. Toto probíhá pomocí HTTP POST požadavku na API, který obsahuje klíč konkrétního modelu a url odkaz na obrázek uložený v Azure blob storage. Odpověď je ve formátu JSON. Ten je následně pomocí parseru *parseVisual* vytěžen a vrací počet predikovaných objektů z obrázku v kategorii *vehicle* – tedy automobil.

Funkce **Log** pak zjištěný počet společně s časovým razítkem a dalšími záznamy (viz výše) zapisuje do MySQL databáze pomocí MySQL API.

3.3.3 Architektura 2 – edge zařízení

Oproti cloud-centrické architektuře se v architektuře s *edge zařízením* přenáší více služeb na místní infrastrukturu, respektive právě na edge zařízení. V tomto případě je zcela odstraněna červená část(viz schéma na obrázku 3.7) a úplně odpadá potřeba pracovat s Azure blob storage (pokud nechceme uchovávat fotografie z požadavků). Custom Vision se používá pouze pro trénování modelu, který je vyexportovaný přenesen na edge zařízení.



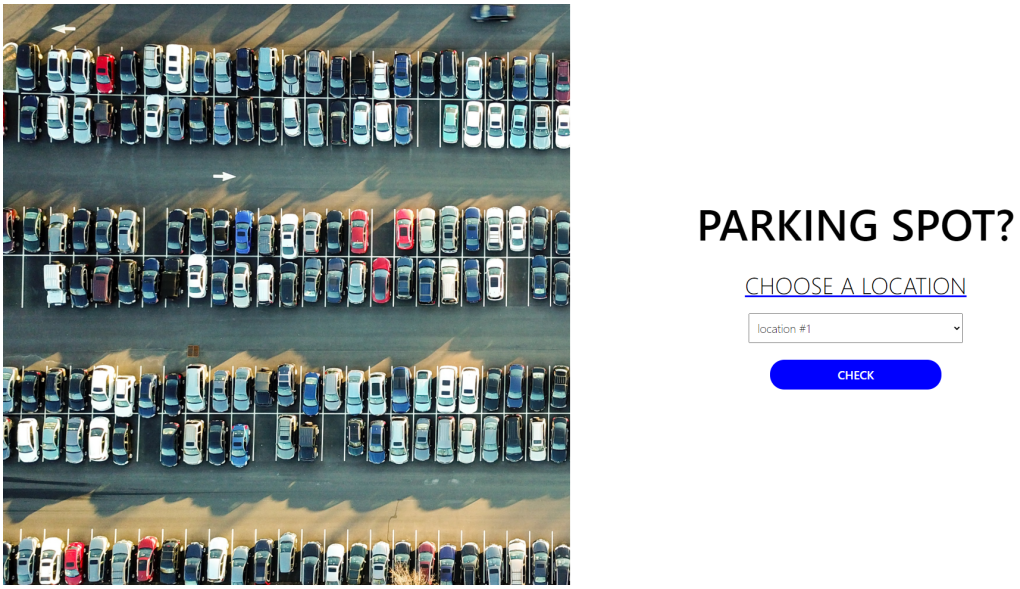
Obrázek 3.7: Architektura 2 – edge zařízení. Pro vytvoření schématu byly použity zdroje z Microsoft knihovny symbolů [16].

Na edge zařízení se právě přesouvá vyexportovaný model služby Custom Vision, který rovnou zpracovává zachycené obrázky, a vrací tak predikovaný stav na parkovišti. Tato architektura bude levnější na běh (jelikož se snižuje počet provedených transakcí v Azure Functions nad modelem, odpadá služba Azure blob storage a tak dále), nicméně bude postrádat agilitu modelu, který je trénován a rovnou provozován v cloudovém prostředí. Proto jsem vybral architekturu 1 jako architekturu, kterou finálně implementuji.

Na edge zařízení pak padají také vyšší nároky na výkon (kvůli běhu modelu) oproti předchozí architektuře, kde pouze pořizuje fotografie a ty potom nahrává do Azure blob storage. Pro tuto architekturu bych doporučil *Vision AI Dev Kit* (viz výše), který je přímo designovaný pro práci s Custom Vision modelem, nicméně jsem neměl možnost ji sám testovat, jelikož v době psaní této práce nebylo možné jej objednat.

3.4 Webová aplikace

Webová aplikace byla navržena v minimalistickém stylu. Je rozdělena do dvou částí – *header*, která má pouze estetickou funkci a obsahuje obrázek parkoviště, a *main*, která obsahuje (mimo názvu aplikace **PARKING SPOT?**) uživatelsky ovladatelné prvky, konkrétně se jedná o rozbalovací nabídku s výběrem lokace, na kterou se uživatel dotazuje, a tlačítko *CHECK*. Při zpracování dotazu je zobrazena načítací obrazovka a následně v části *main* zobrazena odpověď na dotaz (s možností vrátit se na původní výběr).



Obrázek 3.8: Uživatelské rozhraní webové aplikace v desktopové verzi.

Rozhraní je plně responzivní, kdy v maximalizované podobě jsou části *header* a *main* zobrazovány vedle sebe, *header* vlevo a *main* vpravo (viz obrázek 3.8) a v kompaktní podobě nad sebou, *header* nahoře a *main* pod ním (viz obrázek 3.9). Responzivita byla fyzicky testována na zařízení Surface Laptop 3 13.5", dále pomocí vývojářských nástrojů v prohlížeči Microsoft Edge.

Obrázek 3.9: Uživatelské rozhraní webové aplikace v kompaktní verzi.

Rozložení a obsah jsou definovány v souboru `index.pug`, který je následně kompilován do souboru `index.html`. Takto jsem postupoval pro zjednodušení a zrychlení vývoje stránky. Styly jsou poté obsaženy v souboru `index.sass`, který je přeložen do souboru `index.css`.

Byznys logika aplikace, naprogramovaná ve skriptovacím jazyku JavaScript, je v souboru `main.js`, který obsluhuje uživatelské rozhraní a zároveň posílá pomocí funkcí *sendRequest()* a *getData()* HTTP POST a HTTP GET požadavky na služby v cloudu.

Výběr lokace je nyní napevno zakódovaný do stránky, nicméně v produkční verzi by bylo vhodné tento seznam načítat dynamicky z databáze, pokud by výběr lokací rostl. Pro potřeby PoC[7] jsem však k tomuto nepřistoupil.

Webová aplikace byla testována a je funkční v prohlížečích Microsoft Edge (verze 83.x (Official build) (64-bit)) a Google Chrome (verze 83.x) v operačním systému Windows 10 (verze 1909).

[7] Proof of Concept

3.5 Cena běhu služby a dostupnost služby

Jednou z výše popisovaných výhod cloudu (viz sekce 2.1) je jednoduchá predikovatelnost nákladů na běh v závislosti na používání prostředků a zároveň garantovaná dostupnost služeb dle SLA. Cena i dostupnost služby níže je vypočítaná pro běh cloudových služeb popsaných v architektuře cloud-centric (viz sekce 3.3.1).

3.5.1 Cena běhu infrastruktury aplikace

V následujících řádcích popisuji předpokládané náklady na běh back-endu za předpokladů, že:

1. počet dotazů na službu bude právě 10 000 měsíčně,
2. souhrnná velikost uložených obrázků nepřesáhne 5 GB,
3. databáze bude mít maximální velikost 5 GB.

služba	**region**	**popis**	**odhadovaná cena za měsíc**
Azure Functions	West Europe	běh funkce 1 s, maximum 10 000 běhů měsíčně per funkce	$0
Cognitive Services	West Europe	Custom Vision [S1], 10 000 běhů měsíčně, hodina učení modelu měsíčně	$47
Azure blob storage	West Europe	kapacita 5 GB, jen místní redundance, 10 000 čtení a 10 000 zápisů měsíčně	$0,21
Azure Database for MySQL Server	West Europe	General Purpose Tier 1, 2 virtuální jádra, kapacita 5 GB, jen místní redundance, běží celý měsíc	$152,82
		celková cena	**$200,03**

Tabulka 3.1: Cenový odhad vytvořený pomocí služby Azure Pricing Calculator[8].

Celková cena (viz tabulka 3.1) popisuje předpokládané měsíční náklady na běh aplikace bez započtení ceny podpory nebo vývoje. Cena je udávána v amerických dolarech, což je měna, ve které (vedle eur) dodavatel, tedy Microsoft, fakturuje pro Českou republiku. Cena zároveň předpokládá využití tzv. web-direct cen za služby (což jsou ceny, které Microsoft uvádí na svých webových stránkách a v produktových kalkulačkách), při využití subskripce přes CSP (neboli prodejce v modelu Cloud Solution Provider – zprostředkovatel cloudových služeb Microsoftu) nebo získání cen přes EA (Enterprise Agreement – závazek k odebrání minimálního objemu služeb Microsoft Azure) mohou být ceny odlišné – nižší.

3.5.2 Dostupnost služby

Zde vypočítaná dostupnost služby v čase nepopisuje garanci běhu celé mnou vytvořené aplikace, nýbrž jejího cloudového jádra. Dostupnost je vypočítána na základě údajů v dokumentaci k produktu Microsoft Azure [19]. Pro výpočet dostupnosti byl použit vzorec reflektující cloud-centrickou architekturu (viz schéma na obrázku 3.3.1) aplikace

$$SLA = Af_1 \cdot Af_2 \cdot Af_3 \cdot Ad \cdot Af_4 \cdot Ab \cdot Af_5 \cdot Ac \cdot Ab \cdot Af_6 \cdot Ad, \tag{3.3}$$

[8] https://azure.microsoft.com/en-us/pricing/calculator/

kde SLA značí výslednou procentuální dostupnost služby v čase, Af_n značí dostupnost Azure Functions, Ad značí dostupnost databázové služby Azure Database for MySQL (Hyperscale tier bez replik), Ab dostupnost Blob Storage a Ac značí dostupnost Azure Cognitive Services. Azure Funtions mají garantovanou dostupnost 99,95 %, Azure Cognitive Services mají garantovanou dostupnost 99,9 %, Azure Database for MySQL pak 99,99 %. Výsledná dostupnost cloudového jádra aplikace je **99,38169 %**. V případě, že by tedy byla služba nedostupná déle, než **4,45 hodin** měsíčně, může být Microsoft penalizován za nedodržení SLA.

3.6 Další možná rozšíření

Z dalších možných rozšíření služby se podrobněji věnuji těmto čtyřem:

1. portál parkoviště,
2. portál služby,
3. predikce dostupnosti míst v čase,
4. vylepšená uživatelská aplikace.

Jelikož je architektura navržena modulárně, k připojení rozšíření stačí dopsat příslušné konektory a přidat jejich obsluhu do orchestrátoru cloudové služby.

3.6.1 Portál parkoviště

Portál parkoviště je rozšíření a další uživatelský portál, ke kterému může přistupovat správce daného parkoviště. Klíčové jsou čtyři funkcionality: (1) BI pohled na vytížení parkoviště jako takového, pro potřeby kontroly utilizace a plánování kapacit; (2) BI pohled na utilizaci jednotlivých míst pro potřeby údržby; (3) BI pohled na vytížení parkoviště v čase pro potřeby plánování operací na parkovišti; (4) upozornění na anomálie na parkovišti v reálném čase.

Body 1–3 by bylo možné nasadit bez potřeby zasahovat do architektury aplikace pomocí připojení nástroje pro BI k databázi s logy z parkovišť. Pro běh takové služby by však dávalo smysl tyto logy pořizovat automaticky s časovou spouští a nespoléhat jen na logy z jednotlivých spuštění uživateli, aby byly k dispozici kontinuální data v čase. Toto by však prodražilo běh služby (při spouštění každou minutu by za měsíc proběhlo navíc více než 40 000 spuštění).

Bod 4 by vyžadoval zásah do modelu a orchestrátoru, kdy by bylo třeba natrénovat různé anomálie na parkovišti (například spadlá větev na parkovací místo, rozsypané odpadky…) a zároveň model spouštět automaticky jednou za proběhnutí krátkého časového intervalu. Případně z analýzy fotografií přejít na analýzu videa.

3.6.2 Portál služby

Portál služby je rozšíření pro interakci správce parkoviště s Custom Vision. Cílem tohoto rozšíření je možnost dotrénovávat počítačové vidění (například na základě zpětné vazby uživatelů) bez potřeby zasahovat přímo do projektu v Custom Vision a potřeby přístupu do správy projektu v Microsoft Azure. Další důležitou funkcí je pak monitoring běhu služby.

3.6.3 Predikce dostupnosti míst v čase

Jelikož aplikace sbírá logy a uživatelé parkovišť (minimálně na parkovištích v obytných oblastech) se chovají relativně predikovatelně (příjezdy z práce, nákupy), po nasbírání dostatku dat by bylo možné natrénovat prediktivní model strojového učení, který uživateli umožní plánovat příjezd na parkoviště v čas, kdy je pravděpodobné, že budou k dispozici volná místa.

Stejně jako v případě portálu parkoviště by bylo ideální sbírat logy automaticky v časovém intervalu.

Vhodným cloudovým nástrojem pro implementaci této funkce by bylo Microsoft Machine Learning Studio[9], které umožňuje low-code/no-code přístup a snadné napojení na služby Microsoft Azure.

Implementace funkce by také vyžadovala úpravu existující uživatelské aplikace.

3.6.4 Vylepšená uživatelská aplikace

V momentální webové aplikaci kontroluje uživatel dostupnost parkovacích mít manuálně. Další iterace by mířila ke zlepšení uživatelského zážitku a možnosti nastavení časové spouště nebo volání notifikace při přiblížení se k vybrané lokaci (za využití takzvaného *geofencingu*) a následné posílání notifikace na mobilní zařízení. Jelikož by však tyto funkce vyžadovaly přístup k systémovým funkcím zařízení (například údaje o poloze uživatele), bylo by potřeba vyvíjet nativní aplikaci pro platformy iOS a Android.

[9] https://studio.azureml.net/

Kapitola 4

Závěr

Cílem práce bylo vytvořit aplikaci s využitím cloudových služeb a služeb počítačového vidění, která bude sloužit k automatické vizuální kontrole parkoviště a která bude uživatelům vracet počet prázdných parkovacích míst. Tyto jsou počítány pomocí predikce modelu strojového učení implementovaného na cloudu.

Abych dosáhl tohoto výsledku, vytvořil jsem dvě možné architektury k implementaci pracující s cloudovými službami, využitím edge zařízení a uživatelskou webovou aplikací. Tyto architektury se liší v podílu běhu služeb ve veřejném cloudu a služeb offloadovaných na edge zařízení. Vhodnější architekturu s větším množstvím služeb v cloudu jsem naprogramoval a nasadil do veřejného cloudu Microsoft Azure s maximem služeb běžících v modelu platforma jako služba.

Zároveň jsem natrénoval pomocí služby Custom Vision v rámci Azure Cognitive Services dva modely strojového učení. Jeden model klasifikuje obrázky z parkovišť do dvou kategorií – parkoviště plné a parkoviště s volnými místy. Druhý pak přímo detekuje automobily na fotografiích a z návratové hodnoty pak funkce v orchestrátoru vypočítává množství prázdných parkovacích míst. Tento model s detekcí objektů jsem implementoval v rámci výše zmíněné architektury. Oba tyto modely jsou dotazovatelné přes API Custom Vision.

Pro trénování modelů jsem využil dataset čítající 3 262 fotografií parkoviště. Tento jsem měl rozdělen na dvě kategorie – část s plným parkovištěm a část s různým počtem volných míst.

Pro uživatelskou interakci se službou jsem vytvořil responzivní webovou aplikaci, která je jediným bodem, odkud lze s cloudovou službou interagovat.

Dalšími kroky by bylo vytvoření nativních aplikací pro mobilní platformy, které by umožnily příjemnější interakci se službou a implementace rozšíření směrem ke zpracovávání a využití zachycovaných logů z parkovišť.

Za úspěch považuji vytvoření infrastruktury aplikace bez potřeby nasazení virtuálních strojů, správy sítě a operačních systémů a za využití platformních služeb a bezservevrového principu programování s vysokou dostupností a škálovatelností. Zároveň také prozkoumání využití a nasazení služeb počítačového vidění bez nutnosti hluboké odborné znalosti programování strojového učení a neuronových sítí.

Díky možnostem vývoje aplikací a služeb v cloudu lze dnes vyvíjet výpočetně náročné služby bez nutnosti vysokého počátečního kapitálu a nutnosti vlastnit dostatečně výkonný hardware, který by byl jinak potřebný. Zároveň díky možnostem bezserverového programování odpadá nutnost hluboké znalosti správy serverů a infrastruktury.

Literatura

[1] Agaba, I. *Adaptive Process Distribution at the Edge of IoT using the Integration of BPMS and Containerization*. Tartu, 2017. Diplomová práce. University of Tartu.

[2] Armbrust, M., Fox, A., Griffith, R., Joseph, A. D., Katz, R. et al. *Above the Clouds: A Berkeley View of Cloud Computing*. Technická zpráva UCB/EECS-2009-28. University of California at Berkeley, 2009.

[3] Aurer. *SVG Loading icons* [online]. 2014 [cit. 2020-05-27]. Dostupné z: https://codepen.io/aurer/pen/jEGbA.

[4] Buyya, R., Yeo, C. S., Venugopal, S., Broberg, J. a Brandic, I. Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility. *Future Generation Computer Systems*. 1. vyd. 2009, roč. 25, č. 6, s. 599–616. ISSN 0167-739X.

[5] Bíl, T. *Hluboké neuronové sítě pro analýzu medicínských obrazových dat*. Brno, 2019. Bakalářská práce. Vysoké učení technické v Brne.

[6] Carr, D. *Find a Car Park* [online]. kaggle.com, červen 2019 [cit. 2020-05-27]. Dostupné z: https://www.kaggle.com/daggysheep/find-a-car-park.

[7] Catlin, H., Weizenbaum, N., Eppstein, C., Anne, J. et al. *Saas – Sass Basics* [online]. 2020 [cit. 2020-05-27]. Dostupné z: https://sass-lang.com/guide.

[8] Gartner. *Magic Quadrant for Cloud AI Developer Services* [online]. Stamford: gartner.com, únor 2011 [cit. 2020-05-27]. Dostupné z: https://www.gartner.com/doc/reprints?id=1-1Y6KI01L&ct=200122&st=sb.

[9] Gartner. *Gartner Says Worldwide IaaS Public Cloud Services Market Grew 31.3% in 2018* [online]. 2019 [cit. 2020-05-27]. Dostupné z: https://www.gartner.com/en/newsroom/press-releases/2019-07-29-gartner-says-worldwide-iaas-public-cloud-services-market-grew-31point3-percent-in-2018.

[10] Google. *AI and machine learning products* [online]. 2020 [cit. 2020-05-27]. Dostupné z: https://cloud.google.com/products/ai/#tab5.

[11] Gupta, L. *Management And Security Of Multi-Cloud Applications*. Washington, 2019. Disertační práce. Washington University in St. Louis.

[12] IBM. *Governor Cuomo Announces IBM Investment To Create Artificial Intelligence Hardware Center At SUNY Poly Albany Campus* [online]. 2019 [cit. 2020-05-27]. Dostupné z:

https://newsroom.ibm.com/2019-02-07-Governor-Cuomo-Announces-IBM-Investment-To-Create-Artificial-Intelligence-Hardware-Center-At-SUNY-Poly-Albany-Campus.

[13] IBM. *IBM Cloud Learn Hub*. Květen 2020.

[14] Mao, W. *Vision-Based Vehicle Detection and Tracking in Intelligent Transportation System*. Espoo, 2019. Diplomová práce. Aalto University.

[15] Microsoft. *DJI and Microsoft partner to bring advanced drone technology to the enterprise* [online]. 2018 [cit. 2020-05-27]. Dostupné z: https://news.microsoft.com/2018/05/07/dji-and-microsoft-partner-to-bring-advanced-drone-technology-to-the-enterprise/.

[16] Microsoft. *Microsoft Azure Cloud and AI Symbol / Icon Set – SVG* [online]. 2019 [cit. 2020-05-27]. Dostupné z: https://www.microsoft.com/en-us/download/confirmation.aspx?id=41937.

[17] Microsoft. *Microsoft announces a $1 billion digital transformation plan for Poland, including access to local cloud services with first datacenter region* [online]. 2020 [cit. 2020-05-27]. Dostupné z: https://news.microsoft.com/europe/2020/05/05/microsoft-announces-a-1-billion-digital-transformation-plan-for-poland-including-access-to-local-cloud-services-with-first-datacenter-region/.

[18] Microsoft. *Microsoft Docs – Technical Documentation, API and code examples*. Květen 2020.

[19] Microsoft. *Service Level Agreements* [online]. 2020 [cit. 2020-05-27]. Dostupné z: https://azure.microsoft.com/en-us/support/legal/slahttps://azure.microsoft.com/en-us/support/legal/sla.

[20] Mozilla a contributors individual. *MDN Web Docs* [online]. 2020 [cit. 2020-05-27]. Dostupné z: https://developer.mozilla.org/.

[21] Nadela, S. *Democratizing AI: Satya Nadella on AI vision and societal impact at DLD* [online]. 2017 [cit. 2020-05-27]. Dostupné z: https://news.microsoft.com/europe/2017/01/17/democratizing-ai-satya-nadella-shares-vision-at-dld/.

[22] npm. *Npm* [online]. 2020 [cit. 2020-05-27]. Dostupné z: https://www.npmjs.com.

[23] Nwamba, C. *Stateful Serverless with Durable Functions* [online]. 2019 [cit. 2020-05-27]. Dostupné z: https://dev.to/azure/stateful-serverless-with-durable-functions-2jff.

[24] Pug. *Pug – Getting Started* [online]. 2020 [cit. 2020-05-27]. Dostupné z: https://pugjs.org/api/getting-started.html.

[25] Rana, O. *Aerial view of assorted cars in parking lot* [online]. 2018 [cit. 2020-05-27]. Dostupné z: https://unsplash.com/photos/rZ7aO1lorLU/info.

[26] Redmon, J., Divvala, S., Girshick, R. a Farhadi, A. You Only Look Once: Unified, Real-Time Object Detection. In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2016.

[27] Services, A. W. *AWS Total Cost of Ownership (TCO) Calculator* [online]. 2020 [cit. 2020-05-27]. Dostupné z: https://awstcocalculator.com/.

[28] VMware. *Cloud 2.0: Companies Move From Cloud-First To Cloud-Only* [online]. 2019 [cit. 2020-05-27]. Dostupné z: https://www.forbes.com/sites/vmware/2017/04/07/cloud-2-0-companies-move-from-cloud-first-to-cloud-only/#2a01d8b84d5e.

[29] Watts, S. a Raza, M. *SaaS vs PaaS vs IaaS: What's The Difference and How To Choose* [online]. 2019 [cit. 2020-05-27]. Dostupné z: https://www.bmc.com/blogs/saas-vs-paas-vs-iaas-whats-the-difference-and-how-to-choose/.

[30] Šimonfy, A. *Cloud computing and its introduction in small and medium sized enterprises.* Brno, 2015. Bakalářská práce. Mendlova Univerzita v Brne.

Příloha A

Instrukce pro spuštění a nasazení

Prerekvizity:

1. máte nainstalované Visual Studio Code ve verzi 1.45.1 s doplňky **Azure Account, Azure Functions a Live Server**,
2. máte účet pro přístup do Microsoft Azure a k němu přiřazenou subskripci (pouze pro nasazení vlastních Azure Functions),
3. na používaném zařízení máte stabilní připojení k internetu,
4. máte nainstalovaný Node.js ve verzi 12.16.2,
5. máte nainstalované npm ve verzi 6.14.4,
6. potřebné node moduly jsou již staženy ve složce *node_modules*, pokud však budete během používání vyzváni k doinstalaci modulů, prosím, učiňte tak.

Postup pro spuštění aplikace s existující infrastrukturou:

1. zkontrolujte, že splňujete prerekvizity výše,
2. otevřete Visual Studio Code a v něm složku *parking_app*,
3. myší označte složku *parking_app/webapp*,
4. ve spodní liště klikněte na tlačítko **Go Live**,
5. v otevřeném okně prohlížeče klikněte na **webapp**,
6. ujistěte se, že se webová stránka spustila z adresy http://127.0.0.1:5500/webapp/, jinak upravte nastavení doplňku Live Server ve Visual Studio Code,
7. aplikace nyní běží.

Postup pro nasazení vlastních Azure Functions:

1. otevřete Visual Studio Code,
2. vyberte složku *parkingapp*,
3. otevřete doplněk pro Azure (klávesová zkratka (Ctrl+Shift+A),

4. v horním panelu vyberte možnost *Deploy to Function App...*,
5. vyberte subskripci ve vašem účtu pro přístup do Microsoft Azure, do které chcete aplikaci nahrát,
6. vyberte možnost *Create new Function App in Azure*,
7. funkci pojmenuje a potvrťe stisknutím klávesy Enter,
8. vyberte verzi Node.js 12,
9. vyberte lokaci pro spušťení zdrojů West Europe,
10. počkejte na nasazení aplikace,
11. v portálu Azure otevřete skupinu prostředků se jménem vaší aplikace,
12. otevřete App Service,
13. Přejděte do nastavení API –> CORS,
14. do povolených zdrojů přidejte adresu webové aplikace, ze které budete Azure Functions spouštět (v mnou popisovaném případě: http://127.0.0.1:5500),
15. v souboru *webapp/main.js* upravte na řádku 11 adresu na vaši funkci (získáte po provedení kroku 10).

Příloha B

Plakát

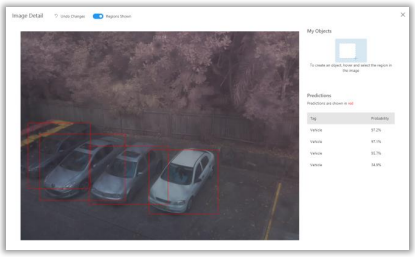
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

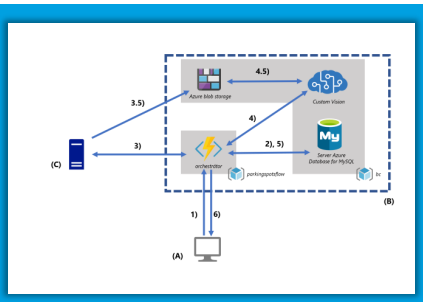
Vizuální kontrola počtu volných parkovacích míst s využitím cloudových služeb

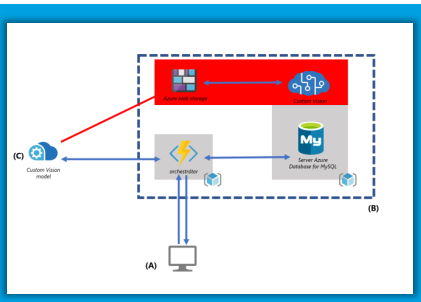
Autor: **Vladimír Hruban**
Vedoucí práce: **Ing. Jakub Špaňhel**

2020

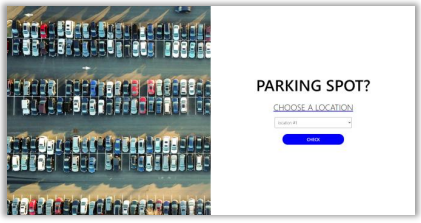
Cílem této práce je navrhnout a vyvinout službu běžící ve veřejném cloudu, která využívá služeb strojového učení pro vizuální kontrolu počtu volných parkovacích míst. V rámci tohoto zadání byly navrženy dvě architektury a různým podílem služeb běžících na cloudu a jedna z nich byla implementována. Zároveň vznikla uživatelská webová aplikace pro komunikaci se službou.







Architektury



Výsledkem práce je hotová a implementovaná infrastruktura ve veřejném cloudu Microsoft Azure, webová aplikace a dva modely strojového učení, z nichž jeden klasifikuje snímky do kategorií *plné/neplné* parkoviště a druhý přímo detekuje a počítá vozidla na parkovišti.

Příloha C

Modelový příklad ekonomického dopadu nasazení cloudových služeb

Total Cost of Ownership (TCO) Comparison

This report includes a total cost of ownership (TCO) comparison between running your application in an on-premises or colocation infrastructure and AWS. The on-premises/colocation infrastructure is based on the description you provided in the online tool. The AWS infrastructure is an approximation of the infrastructure you described. These calculations use third-party estimates and assumptions. This calculator provides an estimate of usage charges for AWS services based on certain information you provide. Your monthly charges will be based on your actual usage of AWS services and may vary from the estimates the calculator has provided.

Notices

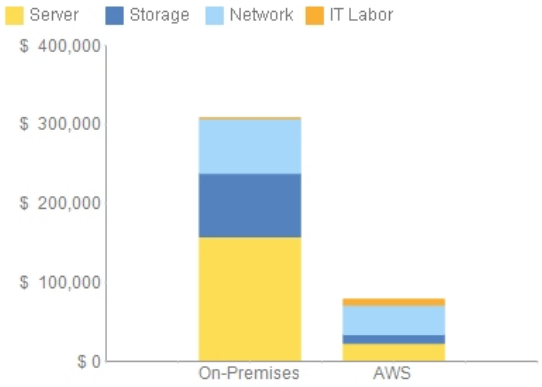
AWS Total Cost of Ownership (TCO) Calculator

On-Premises vs. AWS Summary

You could save **74%** a year by moving your infrastructure to AWS.

Your three year total savings would be $ 229,723.

3 Years Cost Breakdown



3 Yr. Total Cost of Ownership		
	On-Premises	**AWS**
Server	$ 158,059	$ 23,111
Storage	$ 80,233	$ 11,084
Network	$ 69,387	$ 37,525
IT-Labor	$ 810	$ 7,046
Total	$ 308,489	$ 78,767

AWS cost includes business level support

Environment Details

Currency: United States Dollar

Your Virtual environment

Environment: Virtual					
# of VMs	**CPU Cores**	**Memory (GB)**	**OS**	**VM Usage (%)**	**Optimize by**
5	4	64	Windows	100%	CPU

Storage (TB)		
SAN	**NAS**	**Object**
10	0	0

Bandwidth (Mbps)	
Pipe Size	**Peak/Average Ratio**
1,000	3

Your AWS environment : EU (Frankfurt)

Closest AWS Instances					
# Instances	**Instance**	**vCPU**	**RAM (GiB)**	**Optimize by**	**Instance Type**
5	r5.large	2	16	CPU	3 Yr. Partial Upfront EC2 Instance Savings Plan

EC2 Instance Mapping Criteria	
Optimize by	**Description**
CPU	Optimized EC2 instance costs with the right-sizing approach at 50% utilization of compute and DB servers
RAM	Option matches by RAM size and then finds the lowest priced EC2 instance from the available choices
Storage IO	Option matches by I/O requirements and then finds the lowest priced EC2 instance from the available choices

Cost Breakdown

Your On-Premises Cost Breakdown

Server

Hardware : $ 64,740 [41%] Software : $ 525 [0%]
Overhead : $ 92,794 [59%]

Network

Bandwidth : $ 60,000 [43%] Gear : $ 9,387 [7%]
Admin : $ 69,387 [50%]

Your AWS Cost Breakdown

Compute EC2

3 Yr EC2 Instance Savings Plan : $ 21,024 [100%]
On Demand : $ 0 [0%]

EBS

IOPS : $ 0 [0%] EBS Volumes : $ 9,887 [98%]
Snapshot : $ 190 [2%]

Server

Input

On-Premises Server Configuration							
App Name	**# of VMs**	**CPU Cores**	**Memory (GB)**	**Hypervisor**	**Guest OS**	**VM Usage (%)**	**Virtualization Host**
Demo	5	4	0	Hyper-V	Windows	0	Host 1: 2 CPU, 8 Cores, 96 GB RAM

Modified Assumption	
Parameter	**Value**
Host 1: 2 CPU, 8 Cores, 96 GB RAM	
VMware License cost ($)	
Windows License cost ($)	
Windows Assurance ratio (%)	
Metered Power cost/kWH	
Cost to operate a rack/mo	

Output

On-Premises - Server Costs

Server Hardware Costs

Virtual Host Sizing for Virtualized Environments						
App Name	**# VMs**	**Host Type**	**# Cores**	**RAM**	**# Servers**	**VM Density**
Demo	5	Host 1	16	96	4	2

Server Hardware Costs							
# Servers	# of Cores	RAM (GB)	Units (U)	Power (KW)	Unit Cost	Unit Discount	Total Cost
4	16	96	8	3	$ 10,189	25%	$ 30,567
4			8	3			$ 30,567

Total Server Hardware cost	$	30,567
Server Hardware Maintenance cost for 3 Yrs. (@15%/Yr.)	$	13,755
Total number of Racks required (1 Rack=42U, 28U occupied by servers, 4U by ToR switches and PDUs)		1
Total Peak power consumed (kW)		3

Rack Infrastructure Costs

AWS - EC2 Costs

EC2 Instance Costs (3 Yr.) – On-Demand and Reserved Instances

3 Yr. Partial Upfront EC2 Instance Savings Plan			
AWS Instance	**Upfront**	**Hourly**	**Total Costs**
r5.large	$ 10,512	$ 0.16	$ 21,024
Total Cost:			**$ 21,024**

Total costs = (hourly cost*8,760 hours/yr.*3 years)* # of instances.

On-Demand			
AWS Instance	**Upfront**	**Hourly**	**Total Costs**
r5.large	0	0.24	$ 31,536
Total Cost:			**$ 31,536**

Total costs = (hourly cost*8,760 hours*3 years utilization)* # of instances (Hourly usage fee charged for each hour you use the instance)

Lowest Priced Instance

Rack Chassis with PDU (@$3500/rack) cost	$	1,850
PDUs, dual 280V per rack (@$540 each, 2/rack for HA) cost	$	920
Top of Rack Switch (48-port 10/100/1G, $5,000 each, 2/rack for high availability)	$	10,000
Rack and Stack one-time deployment cost ($250/server)	$	1,000
Provision for spare servers for 3 Yrs. (@5% spare capacity/Yr.)	$	6,648
Total Rack costs (rack infrastructure and server hardware)	**$**	**64,740**

Virtualization Software Costs

Hyper-V		
Total number of Microsoft Hyper-V licenses req		8
Microsoft Hyper-V license costs	$	300
Microsoft Hyper-V SA costs	$	225
Total Microsoft Hyper-V license + SA costs (3 Yrs.)	$	525
Total virtualization license and maintenance cost (3 Yrs.)	**$**	**525**
Total 3-Year Database Software License Cost	$	-
Total Server Costs (Hardware and Software) - 3 Yr.	**$**	**65,265**

Facilities Costs (data center space, power and cooling) - On-Premises

Total Power consumed by servers (kW)		3
Metered cost per kWH	$	0.36
Estimated power cost/month	$	777.60
Monthly cost to operate a rack	$	1,800.00
Total rack costs/month	$	1,800.00
Total monthly Facilities costs	$	2,577.60
Facilities costs - On-Premises (3 Yr.)	**$**	**92,794**

Server cost break-down

Server cost break-down

Category	Cost	% of Total Cost
Hardware	$ 64,740	41%
Software	$ 525	0%
Operating Costs (3 Yrs.)	$ 92,794	59%

Instance	Cost	Type
r5.large	$ 21,024	3 Yr. Partial Upfront EC2 Instance Savings Plan
Total Cost:	**$ 21,024**	

EC2 Costs (3 Yr.) **$ 21,024**

EC2 Reserved Instances discounts (if Applicable)

EC2 Reserved Instances

AWS Instance	Pricing model	# Instances	Upfront fee	Total cost
r5.large	3 Yr. Partial Upfront EC2 Instance Savings Plan	5	$ 10,512	$ 21,024

Total fee	**$**	**21,024**
Discount Tier Applicable		0%
AWS Business Support (EC2)	**$**	**2,087**
EC2 Costs (3 Yr.) after discount	**$**	**23,111**

Total	$ 158,059	100%

Total server cost, including operational cost (3 Yr.)	**$**	**158,059**

Storage

Input

On-Premises Storage Configuration				
Storage Type	**Raw Storage Capacity (TB)**	**Max IOPS for Application**	**Backup % / Month**	**Disk Type**
SAN	10	0	0	

Modified Assumption	
Parameter	**Value**
SAN	
Number of TB in a rack	

Output

On-Premises - Storage Costs

Only raw capacity specified, no IO requirements; use HDD by default

SAN Cost

Starting capacity/raw capacity (TB) user provided		10
Starting capacity/raw capacity (GB)		10,240
Capacity after OS Penalty (~7%, capacity OS recognizes) (GB)		9,216
Usable capacity based on RAID (RAID 10 assumed) configuration (GB)		1,843
$/raw GB purchase price	$	4.40
Discounted $/raw purchase price (50% storage hardware discount applied)	$	2.20
Acquisition Cost of SAN storage	**$**	**22,528**

Storage backup cost

Total amount of storage to be backed up (TB)		10.00
Total amount of storage to be backed up (GB)		10,240
Type of Tape Library used		LTO-6
Max uncompressed speed (MB/s) for Tape Library		160
Max uncompressed speed - TB/day		13.18
Backup Window Time(hr.)		8
TBs processed/drive for backup window		4.39
Number of Tape drives required		3
Tape Library price/drive	$	2,300
Backup cost (3 Yr.)	**$**	**6,900**

Storage Overhead (data center space, power,

AWS - Storage Costs

EBS Storage - Only Standard EBS used with no IOPS requirements

EBS Costs - Equivalent to On-Premises SAN environment

Starting capacity (GB)		1,843
Equivalent EBS storage volume		General Purpose (SSD)
Number of EBS volumes required		2
EBS volumes cost/month	$	274.64
Initial snapshot cost(one-time)	$	189.85
EBS incremental snapshots cost/month	$	-
Total EBS cost /month	$	275
EBS Costs (3 Yr) - no IOPS	$	10,077
EBS Costs (3 Yr.)	**$**	**11,084**
AWS Business Support (EBS)	**$**	**1,008**
EFS Costs (3 Yr.)	**$**	**{TotalCostForThreeYears}**
AWS Business Support (EFS)	**$**	**{EFSSupportCost}**
Total AWS Storage Costs (3 Yr.)	**$**	**11,084.45**

cooling, storage administrator)

Typical TB managed by storage admin/Yr.		1000
Storage Admin Costs (3 Yr.)	**$**	**405**

Amount of TBs hosted by a single rack (TB)		1000
Number of racks required		1
Monthly cost to operate a rack	$	1,400
Total data center space, power, cooling costs (3 Yr.)	**$**	**50,400**

Storage cost break-down

Storage cost break-down		
Category	Cost	% of Total Cost
Raw Capacity (Incl. IOPS)	$ 22,528	28%
Backup	$ 6,900	9%
Overhead (excl. storage admin)	$ 50,400	63%
Storage Admin	$ 405	1%
Total	$ 80,233	100%

Total Storage Costs (3 Yr.)	**$**	**80,233**

Network

Input

Data Center Bandwidth (Mbit/s)	1000
Peak/Average Ratio	3

Modified Assumption	
Parameter	**Value**
Network overhead as % of Hardware costs (%)	

Output

On-Premises - Networking Costs

Networking Hardware and Software Costs

Network overhead cost as a % of server hardware acquisition cost		10%
Network hardware and software cost	$	6,474.05
Network hardware and software maintenance/Yr.		15%
Maintenance cost (3 Yr.)	$	2,913.32
Total Network Hardware and Software costs (3 Yr.)	**$**	**9,387**

Bandwidth Costs (On-Premises)

Size of Network Pipe (Mbps)		1000
Peak/Avg. Ratio		3
Average Bandwidth		333
On-premises Bandwith costs/Mbps	$	5.00
Bandwith costs/month	$	1,666.67
Avg. data transferred per month (TB)-Inbound + Outbound		103
Avg. data transferred per month (TB)-North/South		20.6
Avg. data transferred per month (TB) - Outbound		10.3
On-premises Bandwith costs/Mbps	$	5.00
Bandwith costs/month	$	1,666.67
Bandwith costs - On-Premises (3 Yr.)	**$**	**60,000.00**

Network Admin Costs

Network admin effort as % of total IT admin effort		8%
Avg. burdened salary for your Network Admin	$	13,500
IT labor cost (1 Yr.)	$	-
Network admin costs (1 Yr.)	$	-

AWS - Data Transfer Costs

Monthly Data Transfer Out (TB) 10.3

Data Transfer Costs			
	EU (Frankfurt)	**Tier (GB)**	**Monthly Cost**
First 1 GB per month	$ -	1	$ -
Up to 10 TB per Month	$ 0.09	10240	$ 921.60
Next 40 TB per Month	$ 0.09	306	$ 26.00
Next 100 TB per Month	$ 0.07	0	$ -
Over 350 TB per Month	$ 0.05	0	$ -

Total monthly data transfer costs $ 947.60

AWS Business Support (data transfer) **$ 3411.36**

Data Transfer Costs (3 Yr.) **$ 37,525**

Network admin costs (3 Yrs.)	$	-
Total Networking Costs (3 Yr.)	$	**69,387**

IT Labor

Input

Provide average salary for your data center staff	$ 13500
Provide Number of VMs per Admin	250

Modified Assumption

Parameter	Value
On-Premises Server Admin Ratio	13500

Output

On-Premises- IT Labor Costs

Number of VMs managed by an Admin		250
Avg. burdened salary for your IT Admin	$	13,500
Number of VMs in your current environment		5
Admin effort required for your current environment		2%
Total IT Admin Costs -based on number of VMs/Servers (1 Yr.)	$	270
Total IT Admin Costs -based on number of VMs/Servers (3 Yr.)	**$**	**810**

AWS - IT Labor Costs

EC2 Admin Costs

# of EC2 instances managed by an admin		375
Avg. burdened salary for your IT Admin	$	13,500
# of EC2 instances in your environment		5
Admin effort required for your current environment		2%
IT labor costs (1 Yr.)	$	180
IT labor costs (3 Yr.)	$	540
Total IT Admin Costs -based on number of VMs / Servers (3 Yr.)	**$**	**540**

AWS SUPPORT

Modified Assumption	
Parameter	**Value**
AWS Server Admin Ratio	
AWS support Included	Y
1 Yr. or 3 Yr. Reserved Instances	3

AWS - Support Costs

Monthly EC2 Spend	$	292.00
Monthly EBS Spend	$	279.91
Monthly S3 Spend	$	-
Monthly S3IA Spend	$	-
Monthly Data Transfer Spend	$	947.60
Total AWS Spend - Month	**$**	**1,519.51**

Support Costs - All Services	
Business Level Support	**Cost**
10% of monthly AWS usage for the first $0 - $10K	$ 151.95
7% of monthly AWS usage for the first $0 - $10K	$ -
5% of monthly AWS usage for the first $0 - $10K	$ -
3% of monthly AWS usage for the first $0 - $10K	$ -

AWS Support for all services - Month	**$**	**151.95**
AWS Support for all services (3 Yr.)	**$**	**5,470.24**
EC2 Reserved Instances Upfront cost after discount	**$**	**10,512**

Support Costs - Reserved Instances	
Business Level Support	**Cost**
10% of monthly AWS usage for the first $0–$10K	$ 1,000.00
7% of monthly AWS usage from $10K–$80K	$ 35.84
5% of monthly AWS usage from $80K–$250K	$ 0.00
3% of monthly AWS usage over $250K	$ 0.00

AWS Reserved Instance Support cost (One-Time)	**$**	**1,035.84**
Total AWS Support (Business)	**$**	**6,506**

METHODOLOGY

The AWS TCO calculator uses the following methodology when calculating on-premises, colocation, and AWS costs.

Our methodology defines Total Cost of Ownership (TCO) as below –

TCO = Acquisition Costs + Operational Costs

Operational costs include labor cost to manage the data center operations as well as overhead cost associated with running the data center equipment. A standard 3 year time frame is used for our calculations as the useful life for the data center equipment.

The following graphic shows the major cost categories in on-premises and colocation environments

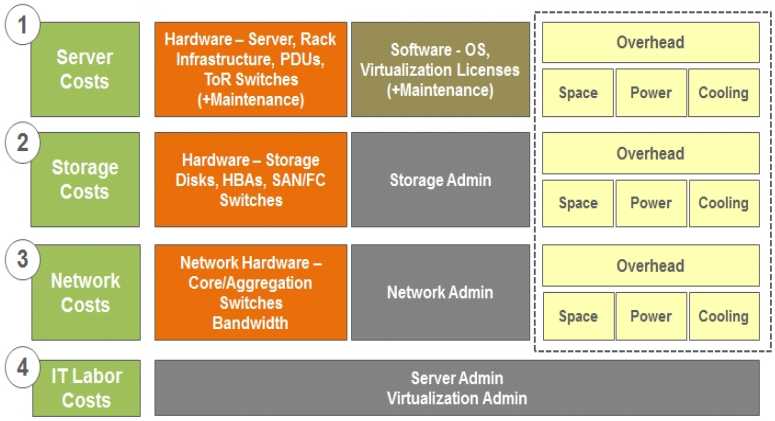
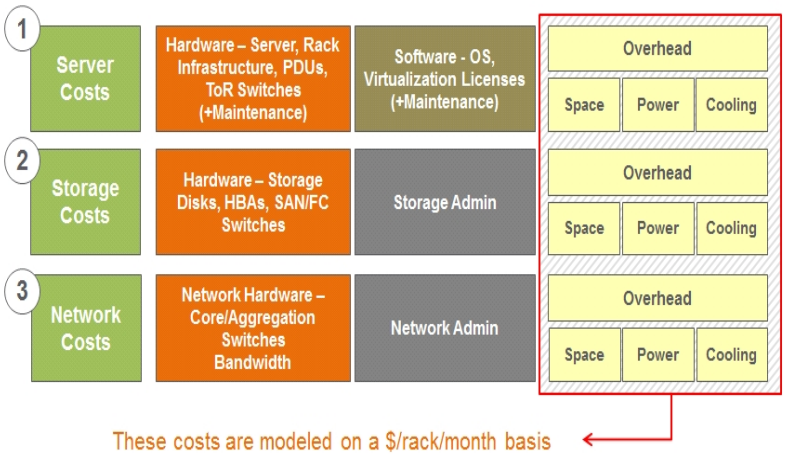


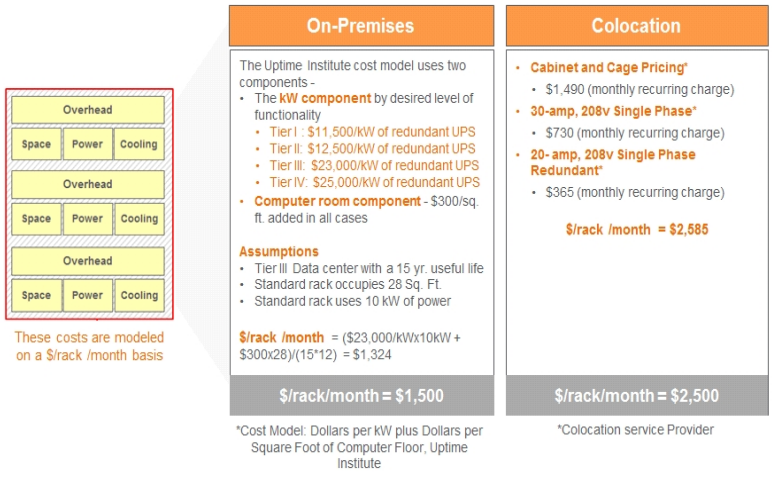
Diagram doesn't include every cost item in data center. E.g. software costs can include database, systems management, middle tier software costs. Facilities cost can include costs associated with upgrades, maintenance, building security, taxes etc. IT labor costs can include security admin and application admin costs.

For On-Premises/Colocation, TCO = Server Costs + Storage Costs + Network Costs + IT Labor Costs

For on-premises and colocation environments, each of the major cost categories (server, storage, and network) include the cost of hardware, software (where applicable), and overhead costs. Overhead costs include the cost of data center floor space, and power and cooling required for data center equipment. For our calculations, a "standard rack" is considered to be the typical 19 inch rack that has a rack footprint of 28 sq. ft. (actual area covered by the rack) in the data center. Additionally, we assume average power density per rack to be 10kW in an on-premises data center and a cabinet to have a primary 20 amp, 120V single phase circuit and a redundant 20 amp, 120V circuit in a colocation environment. We use Uptime institute cost model to calculate overhead costs for on-premises and a publicly available price quote from a global colocation, interconnection, and managed IT infrastructure service provider for colocation environment.Since power and cooling expenses are billed on a monthly basis, we calculate our overhead costs on a monthly basis. We also use a "standard rack" as a common point for calculating overhead costs.



For On-Premises and Colocation environments, the $/rack/month is calculated differently -



As shown above, the logic by which the overhead cost is calculated for on-premises and colocation environments is different. Most of the other cost categories are handled similarly between these environments. On the network side, a colocation environment incurs recurring bandwidth costs where as an on-premises environment also incurs network capital expense and network operation expense.

	On-Premises	Colocation
Overhead (Space, Power, Cooling)	Default Cost/Rack/month = $1,500 Power/cooling charged separately	Default Cost/Rack/month = $2,500 Power/cooling included in this cost
Server (excl. overhead)	Same	Same (unless HW is leased)
Storage (excl. overhead)	Same	Same (unless HW is leased)
Network (excl. overhead)	Flat Bandwidth charge; Network overhead	Tiered Bandwidth charge; no network overhead
IT Labor Costs	Same	Same

Finally, on AWS side overhead costs is included in the publicly listed prices and customers don't have to pay extra for space, power, and cooling as shown below.

	Server & Network Hardware	OS + Virtualization Software	Data center /Colocation Floor Space	Power Cooling	Data Center Personnel	Storage Redundancy	Resource Mgmt. /SW Automation	Software Defined Networking
								
amazon web services	✔	✔	✔	✔	✔	✔	✔	✔
Hardware Vendor Offering	✔	✘	✘	✘	✘	✘	✘	✘

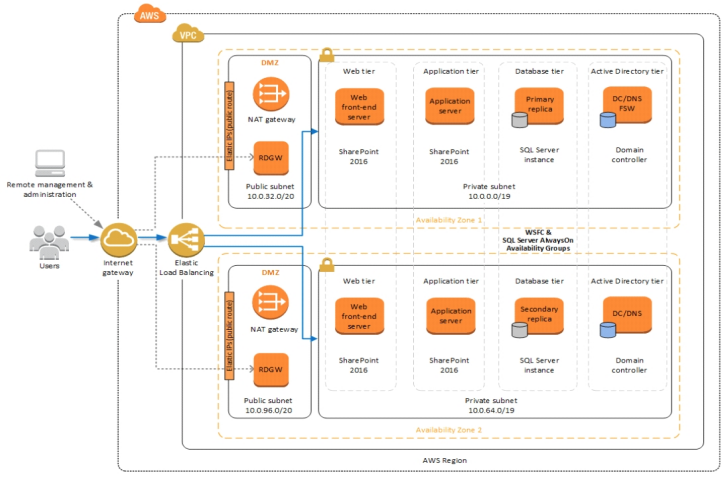
With AWS, we include AWS Business level support costs. AWS Business level support includes guidance on optimizing AWS products and configuration to meet your specific needs. Business level support provides discounts as your AWS usage grows

TCO Methodology for RDS

	On-Prem/Colo:	AWS: DB on EC2	AWS: RDS
Server Costs Hardware: Database Pricing:	Server Costs: Commericial List prices, less expected discount, plus operational burden for power/ cooling/ floor space. Database Costs Compare Database Pricing to comparable edition (non-Enterprise)	EC2 Pricing Follows Existing Methodology: Server requirements Mapped to EC2 based on RAM/CPU/Storage constraints DB Mapped to AWS Support & CloudWatch: + AWS Business Level Support + 2 CloudWatch Custom Metrics (2x: Query Analyzer, Replication)	RDS Pricing for DB • User defined Single v Multi-AZ replication • RI used for comparison DB Mapped to AWS Support: + AWS Business Level Support
Block Storage:	DB Storage (SAN/NAS) Follows Existing methodology NAS/SAN Raw storage penalized for OS, RAID 10, over provisioning, and h/w performance capacity requirements to convert to usable. Backup: Follows Existing TCO Methodology: Assume Tape Backup	Mapping SAN/NAS to EBS (current methodology): Backup: Initial backup to S3 (100%), then incremental snapshots based on input for backup/month.	Multi-AZ vs. Single AZ Deployment: If Production, Multi-AZ; otherwise single-AZ. Backup: Same as EC2 Scenario.
Labor:	Compute, Storage, Admin: Follow existing TCO methodology DBA: 1 DBA can manage up to 40 DBs, under 1 TB in size Colo only: 10% additional labor cost for 'on-call DBA' for emergency maintenance issues	DBA Labor Savings of 10%, as a result of decreased time spent security planning, and performing installation, upgrading, migrating. 1 DBA can then manage 45 DBs on EC2.	DBA: Labor savings of 60%, as a result of decreased time spent doing patch mgmt., managing backups, performance, troubleshooting, and storage provisioning. 1 DBA can manage 100 DBs on RDS, versus 40 on prem.

SharePoint Architecture

Amazon Web Services – Microsoft SharePoint Server 2016 on the AWS Cloud May 2016



ASSUMPTIONS

The AWS TCO calculator makes the following assumptions for on-premises, co-location, and AWS environments.

On-Premises and Co-location Assumptions

1. Servers and Racks:

- On-premises and co-location server prices are based on Dell PowerEdge Rack servers and HP ProLiant Rack servers.
- Dell PowerEdge prices available here.
- HP ProLiant Rack servers prices available here.
- Servers can be physical or virtualized. Currently the tool supports VMware vSphere, KVM and Xen hypervisors.
- For virtualized environments, two virtualization host configurations are supported –
- Host 1 - 2 processors with 8 cores each and 96 GB RAM.
- Host 2 – 4 processors with 8 cores each and 256 GB RAM
- VM density is calculated based on the virtual RAM and virtual cores allocated to VMs.
- Server and rack hardware are discounted by 25% off the publicly available list prices.
- A "standard rack" is considered to be the typical 19 inch rack that has a rack footprint (actual area covered by the rack) in the data center as defined here. Standard rack assumed to consist of 42 rack units (42U).
- On average each rack is filled up to 75% of capacity (i.e. for a 42U rack, 32U is actually used)
- Dell PowerEdge Energy Smart 4620S Rack Enclosure used to hold data center equipment. A base price of $3,499 assumed as per the published price here.
- Every rack consists of two top of rack Switches (ToR) for Redundancy. Cisco Catalyst 2960 48 port switches used in calculations with the following configuration- 48 x 10/100/1000 - PoE+ 525Watt + 4 x SFP, LAN Base Layer 2. here.
- Every rack consists of 2 Power Distribution Units (PDU) for high-availability. APC Metered Rack PDU - power distribution strip used with a base price of $545 as per the published price here.
- A 15% hardware maintenance/year applied to the server and rack hardware.
- 5% of all server capacity assumed to be hot spare servers. A hot spare or hot standby is used as a failover mechanism to provide reliability and high-availability in data center environments.
- BYOL Windows EC2 will be 2.3 times more then Linex EC2 cost.

2. Software

- VMware vSphere Enterprise Plus licenses assumed for customers using VMware environments. A base price of $3,495 per licenses and $874 for 1 year support and subscription assumed as per the published prices from VMware here.
- KVM is free software released under the GPL as described here. Vendors like RedHat offer KVM hypervisor under a subscription model that includes product access, all updates, and support. Details can be found here.
- Xen hypervisor is covered by the GNU general public license as described here and available for free. Vendors like Citrix offer a free and paid version of XenServer. Details can be found here.
- Software licenses and maintenance are discounted by 25% off the publicly available list prices.
- Windows OS license cost assumed $100 per core per year, user has a choice to modify this value from modify assumption section.
- The Linux distributions used in our model are available for free as described here. Our model doesn't use the paid Linux distributions like SUSE Enterprise Linux and Red Hat Enterprise Linux.

3. Storage

- Our model assumes RAID 10 configurations for on-premises and colocation storage. RAID 10 details are available here.
- Hard disk manufacturers measure drive capacity differently than operating systems. Hard disk manufacturers use a “base-10” measure, whereas operating systems use a “base-2” measure as described here. This mismatch causes a 7% penalty on the disk capacity.
- Raw capacity is defined as the disk capacity in the box or frame while usable capacity is the disk capacity after RAID protection and available for host allocation.
- Average Solid State Devices (SSD) and Hard Disk Drives (HDD) price per GB available here. Our model assumes higher prices as we include the price of Host Bus Adapters (HBA), Fiber Channel Adapters, Optical or Fiber Channel Cables and other storage equipment.
- A discount of 50% is applied by default to the SSD and HDD price per GB.
- The model assumes that a storage admin manages 1000 TB of data on an annual basis.
- The model assumes that a single storage rack contains 1000 TB of disks.
- The model assumes LTO-5 tapes used for backups. Details available here.
- A base price of $1800/drive assumed for the cost of LTO-5 drive taking tapes of up to 1.5 TB true (uncompressed) capacity as per the published price here.

4. Network

- The model assumes a 20% network overhead for on-premises environments. The network overhead is calculated by taking a 20% overhead on server and rack hardware cost.
- For a colocation facility, no network overhead is assumed as colocation providers would bundle this cost in their prices.
- Traffic in the data center assumed to be both north-south (between servers inside the data center and end points outside the data center) and east-west (between components in the data center). Our model assumes 80% of all data center traffic is “east-west”.
- For on-premises environment, Bandwidth costs also include cost of WAN Optimizers and MPLS VPN.
- Average colocation bandwidth pricing is tiered – our model assumes $30/Mbps at the lowest tier and $7/Mbps at the highest tier. More details can be found here.
- Average network admin effort is around 8% of total IT administration effort as per this report.

5. Power and Cooling

- Average Retail Price of Electricity in the US can be found here.
- The average cents per kilowatt-hour in the US commercial segment for January 2014 was 10.34 cents.
- The model assumes a standard on-premises/colocation PUE of 1.5 and a base kWh (kilo watt hour) price of 10 cents/hr. This gives us a total electricity charge of 15 cents per kWh (10*1.5).

6. Data Center Space

- The model assumes colocation providers charge a fixed $ per kW rate that covers the cost of all power and

space contracted. Every rack assumed to have a primary and redundant power supply.

- The model assumes a separate charge for space, power and cooling for on-premises environments.

7. IT Labor

- Average data center admin salaries by region available here.

AWS Assumptions

1. Compute

- Both On-Demand and 3 Yr. EC2 Instance Savings Plan types used for AWS compute.
- EC2 instances are matched with on-premises servers and VMs based on CPU, RAM, or storage I/O.
- EC2 instance size are optimized with the right-sizing approach assuming 50% utilization of compute and DB servers.
- The number of EC2 instances is the same as the physical servers or VMs on-premises meaning we don't apply any cost optimization on AWS side. In real-life situations, customers would change instance sizes up or down based on monitoring various AWS resource metrics like CPU utilization, free memory, or disk usage.
- Amazon EC2 provides a wide selection of instance types optimized to fit different use cases. Our model uses General purpose, Compute-optimized, Memory-optimized, and Storage-optimized instances.
- The model uses only the current generation of instances- General purpose (m3), Compute-optimized (c3), Memory-Optimized (R3), and Storage-optimized (I2).
- Current generation of instances provide faster, newer Intel Ivy Bridge processors, SSD-based instance storage, Higher performance Enhanced Networking, and advanced features such as support for HVM AMIs.
- The model assumes EC2Instance Savings Plan volume discounts as described here.

2. Storage

- On-premises SAN and NAS storage systems are represented in AWS as EBS volumes.
- On-premises Object storage is represented in AWS as S3.
- S3 offers multiple storage classes – S3 Standard, RRS, and Glacier depending on how S3 handles data. Our model assumes S3 standard only.
- Model assumes RAID 0 for EBS volumes as described here.
- Multiple EBS volumes can be striped together to achieve up to 48,000 IOPS when attached to larger EC2 instances as described here.
- For backup, we use EBS snapshots to S3 for calculating backup costs on AWS.
- Backup%/month is the amount of data that changes every month. So a 50% number means that 50% of data changes every month and is backed up.
- Model calculates EBS-optimized instance cost separately, but doesn't add it to the total storage cost.

3. Network

- Model assumes publicly available Data Transfer OUT tiered rates From Amazon EC2 to Internet as described here.

4. IT Labor

- Model assumes that an IT admin can manage 400 EC2 instances.

5. AWS Support

- Model assumes Business-level support for AWS as described here.

SharePoint Assumptions

1. SharePoint Server 2016 on AWS

- The Amazon Web Services (AWS) cloud provides a suite of infrastructure services that enable you to deploy SharePoint Server 2016 securely, affordably, and with high availability.Running SharePoint Server on the AWS cloud gives you flexibility and agility, and you can fully customize and extend SharePoint for your business processes. This Quick Start implementation guide walks you through the steps to automatically deploy an enterprise SharePoint Server 2016 architecture in your own AWS account

2. Cost and Licenses

- You are responsible for the cost of the AWS services used while running this Quick Start reference deployment. There is no additional cost for using the Quick Start itself. The AWS CloudFormation template for the SharePoint Server 2016 Quick Start includes configuration parameters that you can customize, and some settings, such as the instance types and the number of instances, can greatly affect the cost of the deployment. AWS has published a whitepaper that shows how to estimate the cost of your SharePoint deployment. You have a wide array of options for building your SharePoint farm, and it's not possible to cover them all in that whitepaper or in this guide. The following table offers a model based on some key assumptions

3. Deploying Microsoft software on AWS

- Microsoft on AWS here.
- Secure Microsoft applications on AWS here.
- Microsoft Licensing Mobility here.
- MSDN on AWS here.

4. Microsoft SharePoint Server

- Configure SQL Server 2012 Always On Availability Groups for SharePoint 2013 here.
- Windows Server Failover Clustering and SQL Server AlwaysOn Availability Group here.

FAQ

1. What is the purpose of the AWS TCO calculator?

You can use the AWS TCO calculator to compare the cost of running your applications in an on-premises or colocation environment to AWS. The tool produces a detailed cost comparison with AWS based on the infrastructure details you provide.

2. What assumptions do you make?

We make several assumptions based on third party analyst and industry research as well as data from hundreds of AWS customers. Please refer to the Assumptions section of the TCO output page.

3. What is the difference between an on-premises data center and colocation facility?

An on-premises data center is a brick and mortar structure that contains all the required systems / facilities to house computing infrastructure running 24 x 7 x 365. An on-premises data center is owned and operated by the owners of the computing infrastructure. A colocation facility is usually offered by a provider that owns their own "data center" and rents out rack space and/or computing hardware. These environments have very different cost structures and your TCO for the same application would vary between these environments.

4. How are you calculating on-premises (or colocation) server infrastructure costs?

The calculator averages market rate pricing from multiple enterprise server vendors to determine an average price for a server based on CPU, RAM, and storage configuration. In addition, the tool adds licensing cost for Operating System and virtualization licenses as well as rack infrastructure costs. Rack infrastructure costs include cost of power distribution units, top of rack switches, rack chassis and one-time server deployment.

5. Do you apply any discount to on-premises (or colocation) server hardware, storage hardware, and software acquisition costs?

Yes, by default server hardware is discounted by 25%, Operating System and Virtualization licenses by 25% and storage hardware by 50%. These closely resemble industry standard pricing policy.

6. What types of storage are you using to compare?

On-premises Storage Area Network (SAN) and Network Attached Storage (NAS) are mapped to Amazon EBS, while on-premises object storage is mapped to Amazon S3. The amount of on-premises storage specified in the calculator input represents the amount of raw physical storage capacity purchased. AWS storage that is calculated is not equal to the amount of raw physical storage capacity, but rather a percentage of the storage that is actually utilized for on-premises data storage. Primary and secondary research indicates that this is ~50-60% of the raw physical storage purchased after taking into account RAID and Operating System overhead.

7. How are you calculating on-premises (or colocation) storage costs?

We are using industry standard pricing based on $ per raw GB purchase. We use a combination of Solid State Drives (SSD) and Hard Disk Drives (HDD); SSDs are used in case customers need higher IOPS for their on-premises/colocation environments. The tool uses RAID 10 configurations for on-premises and colocation storage. A discount of 50% is applied by default to the SSD and HDD price per GB.

8. How are you calculating on-premises network costs?

This is calculated as a percentage (20%) of server hardware acquisition cost. The network cost includes the cost for network interface cards, host bus adapters, core switches and other networking gear.

9. How are you calculating bandwidth costs?

We use a sampling of average market rates for data center telecom from major markets (typically measured in $/Mbps). Traffic in the data center is assumed to be both north-south (between servers inside the data center and end points outside the data center) and east-west (between components in the data center). We also assume a high percentage of data center traffic (~70-80%) to be “east-west”.

10. What is the overhead cost? How are you calculating it for on-premises and colocation environments?

Overhead costs include the cost of data center floor space, and power and cooling required for data center equipment. It can also include other ongoing data center expenses such as maintenance and physical security. For the sake of simplicity, we only consider data space, power, and cooling when calculating overhead costs. The way overhead cost is calculated is different for on-premises and colocation environments. We leverage the publicly available Uptime institute’s cost model to calculate overhead costs for on-premises data center and a price quote from a global colocation and interconnection infrastructure service provider for colocation environment. Since power and cooling expenses are billed on a monthly basis, we calculate our overhead costs on a monthly basis. We also use a “standard rack” as a common point for calculating overhead costs.

11. How do you calculate IT Labor costs?

You can specify the fully burdened cost of your IT admin resources. We leverage industry and third party analyst research for server: admin ratios. For physical environments, we assume one FTE IT admin per 100 physical servers or 250 virtual machines. We use these two variables and the rest of the inputs of the tool to calculate the percentage administrative cost. For storage, we assume a storage admin to manage 1 PB of storage annually and for network we assume network admin effort to be ~8% of the total IT admin effort.

12. How are the Amazon EC2 compute costs computed?

The tool automates the task of selecting the right AWS instance type based on the information you provide; you can input your physical or virtual infrastructure details and the tool will provide the equivalent AWS instance types that meet your requirements. The calculator uses current generation Amazon EC2 instances (except GPU instances) to calculate AWS compute costs. The tool also takes into account your on-premises usage/utilization rate. You can think of on-premises usage/utilization rate as the total desired uptime for your servers or VMs. For example, over a

3 year time-period, a 10% usage rate implies that your servers are running 10% of the time (or 3.6 months).

13. Why do you ask for usage/utilization rate?

If you don't need your on-premises VMs or physical servers up and running 24/7/365 you could save lot of money by powering off the VM/servers when not used. AWS provides multiple pricing models to optimize your spend for variable or steady-state workloads. The calculator will let you change the usage % (uptime) and automatically select the right AWS pricing model that will meet that uptime at the lowest price. If you don't know your usage/utilization rate, the tool defaults to 100% (which means that your servers/VMs are running 24/7/365).

14. Why can't I run calculations with previous generation Amazon EC2 instances?

We encourage you to use the latest generation of Amazon EC2 instances to get the best price for performance ratio. The calculator uses the current Amazon EC2 instances - M3, C3, R3, I2 and HS1 (and m1.small and t1.micro). You can find more details about the previous generation Amazon EC2 instances here.

15. Do you use S3 Reduced Redundancy Storage and Amazon Glacier?

No, currently the tool only uses S3 standard. For backup of EBS data, the tool uses EBS Snapshots to Amazon S3 charges.

16. Are data transfer costs included in the AWS storage cost?

No, all AWS data transfer costs are included under the network charges for AWS. The data transfer cost is calculated based on the EC2 data transfer out from Amazon EC2 to the internet. Since the calculator doesn't assume cross-AZ deployments, it doesn't calculate data transfer charges between Amazon EC2 instances in another Availability Zone in the same region.

17. What is the AWS server to admin ratio?

We have found a 400:1 Server to Admin ratio to be appropriate when running your apps on AWS. This is a conservative ratio based on AWS customer engagements. (Note: for on-premises we assume a 100:1 server admin ratio for physical servers and 250:1 ratio for virtual machines)

18. Why do you include AWS support costs?

We recommend all AWS customers use AWS Support to ensure a seamless experience leveraging AWS infrastructure services. We have created multiple tiers to fit your unique technical needs and budget. AWS Basic Support offers all AWS customers access to our Resource Center, Service Health Dashboard, Product FAQs, Discussion Forums, and Support for Health Checks – at no additional charge. Customers who desire a deeper level of support can subscribe to AWS Support at the Developer, Business, or Enterprise level. in our TCO model, we assume Business-level support costs