Česká zemědělská univerzita v Praze

Provozně ekonomická fakulta

Katedra informačního inženýrství

Bakalářská práce

IT architektura pojišťovací společnosti

Radek Marčan

ČESKÁ ZEMĚDĚLSKÁ UNIVERZITA V PRAZE

Katedra informačního inženýrství

Provozně ekonomická fakulta

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Marčan Radek

Informatika

Název práce

IT architektura pojišťovací společnosti

Anglický název

IT architecture of an insurance company

Cíle práce

Cílem práce je vytvořit sadu konkrétních příkladů a aplikací pro informační systém pojišťovací společnosti z pohledu jednotlivých vrstev IT architektury. Součástí práce bude také literární řešerše o IT architektuře obecně.

Metodika

Nastudování teorie z odborné literatury či případně z internetových zdrojů. Posléze nastudování konkrétního příkladu z praxe a následná aplikace do bakalářské práce.

Harmonogram zpracování

říjen - listopad - nastudování teoretických znalostí
prosince - leden - seznámení s praxí včetně prostudování
únor - závěrečné úpravy, formování finální verze

Rozsah textové části
25-35 stran

Klíčová slova
it architektura, informační systém

Doporučené zdroje informací

Informační systémy v podnikové praxi - Sodomka, Klčová
Tvorba informačních systémů : Principy, metodiky, architektury - Buckner, Voříšek, Buchalcevová
Architektury informačních systémů - Dohnal, Pour
Podnikové informační systémy - Molnár

Vedoucí práce
Merunka Vojtěch, doc. Ing., Ph.D.

Termín odevzdání
březen 2014

Elektronicky schváleno dne 30.10.2013

Ing. Martin Pelikán, Ph.D.
Vedoucí katedry

Elektronicky schváleno dne 5.12.2013

Ing. Martin Pelikán, Ph.D.
Děkan fakulty

V Praze dne 5.12.2013

Čestné prohlášení

Prohlašuji, že svou bakalářskou práci "IT architektura pojišťovací společnosti" jsem vypracoval samostatně pod vedením vedoucího bakalářské práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou citovány v práci a uvedeny v seznamu literatury na konci práce. Jako autor uvedené bakalářské práce dále prohlašuji, že jsem v souvislosti s jejím vytvořením neporušil autorská práva třetích osob.

V Praze dne 13. 3. 2014 _____

Poděkování

Rád bych touto cestou poděkoval doc. Ing. Vojtěchu Meruňkovi, Ph.D. za připomínky a odborné vedení.

IT architektura pojišťovací společnosti

IT architecture of an insurance company

Souhrn

Tato bakalářská práce se zabývá vysvětlením co si pod pojmem IT architektura představit a co vše to obnáší. Nejprve je zde vysvětlení základních pojmů, jako například kdo je to architekt a co vše tato pozice obnáší. Další částí této práce je proč je vhodné a potřebné IT architekturu zvolit a zavést do společnosti. V teoretické části jsou vysvětleny technologie, které se při tvorbě a správe IT architektury využívájí, například TOGAF, MDA a podobně. V praktické části je řečeno proč by se měla architektura v pojišťovací společnosti využít a dále rozepsání do jednotlivých vrstev. U každé vrstvy je uvedeno co v daném podniku či organizaci obnáší včetně konkrétních příkladů. Dále je zde uveden souhrn, ve kterém je řečeno od jakých vrstev by se mělo postupovat v případě budování nové organizace, či její pobočky.

Summary

This Bachelor work deals with explanation of what you should imagine under IT architecture and what it all involves. At first there is an explanation of basic term, for example who is architect and what this position involves. In the next part of this work is explanation why it is appropriate and necessary to use IT architecture and bring it up to company. In theoretical part are explained technologies which are used in development and management, for example TOGAF, MDA and so on. In practical part is told why should be IT architecture used in insurance compandy and further there is a breakdown into individual layers. For every layer there is an explanation what is inside this layer and also there are specific examples. There is also summary in which is said from which layers should be started in case of building new organization or branch office.

Klíčová slova:

IT architektura, pojišťovací společnost, vrstvy architektury, podniková architektura, ICT služby, IT architekt, TOGAF, architektonické rámce, Zachmanův rámec, modelem řízená architektura

Keywords:

IT architecture, insurance company, layers of architecture, enterprise architecture, ICT services, IT architect, TOGAF, architectural Framework, Zachman Framework, model driven architecture

Obsah

Seznam obrázků

Seznam tabulek

1 Úvod

Tématem mé bakalářské práce je IT architektura společnosti pojišťovací společnosti. Toto téma jsem si zvolil, jelikož v dnešní době není tato problematika natolik známá, jak by měla být. Problémem je, že mnoho lidí studujících či již pracujících v oboru IT nemají ponětí ani tušení, co pojem IT architektura obnáší.

Vzhledem k neustálému rozvoji a vývoji technologií ICT je nezbytné, aby společnost, jež chce efektivně působit na trhu a využít maximum svého potenciálu využila vhodně navržené IT architektury. Vzhledem k rozvoji technologií je stale větší provázanost, složitost a rozsah těchto technologií.

V této práci se budu zabývat především úvodem do tajů IT architektury a příklady uvádět na pojišťovací společnosti. Pojišťovací společnost jsem vybral z toho důvodu, že zde lze dobře a názořně ukázat konkrétní příklad problému, jenž může nastat anebo naopak problem, jenž je zde řešen. Rozhodl jsem se začít od základních pojmů jako je pojem architetk, co tato pozice zastává, jaké jsou nezbytné požadavky a co si v první řade pod tímto pojmem představit. Dále je vysvětlím co je to IT architektura, proč je dobré to ve firmě využít a jaká kritéria je vhodné při výběru využít. Samozřejmostí je, že dochází k situacím, kdy je volba závislá na daném architektovi, nebo na vedení společnosti a proto se mohou lišit.

Krom různých rámců a technologií se zde také zaměřím na dílčí architektury. Na dílčí architektury se zaměřím především proto, že na nich lze jednoduše demonstrovat co se v této architecture vyskytuje. V praktické části jsem se rozhodl vysvětlit, proč je vhodné využít IT architektury a jaké kritéria je vhodné v případě pojišťovací společnosti využít.

Dále zde uvedu, co se v jednotlivé architecture vyskytuje. Z názvu každé dílčí architektury lze odvodit, co by v ní mohlo nejspíše být, ale vzhledem k závádějícím názvům, či rozsáhlosti daného názvu jsem se rozhodl uvést konkrétní příklady pro větší názornost.

Na závěr praktické části jsem uvedl souhrn, jak by se dalo postupovat při tvorbě pobočky, nebo nového pojišťovací společnosti. Ovšem je nezbytné vše brat s jistou rezervou, jelikož je v tomto souhrnu postup vysvětlen velice zběžně a není to rozvedeno do velikých detailů, jak by bylo nezbytné při opravdové tvorbě. Cílem této práce tedy je informovat o podstatě IT architektury včetně vysvětlení technologií a vrstev, včetně příslušných příkladů v rámci pojišťovací společnosti.

2 Cíl práce a metodika

Cílem bakalářské práce bylo vytvořit práci, v níž je vysvětleno, co je to IT architektura obecně. Čím se zabývá, co si pod tímto pojmem představit a kdy je vhodné, či dokonce nezbytné ji využít. Cílem je zmínit nejen dílčí vrstvy architektury, ale take různé technologie, které se využívají při tvorbě architektury, například Zachmanův rámec. Dále v praktické části je cílem ukázat konkrétní příklad náležící do dané dílčí architektury a vysvětlení co se v dané architektuře nachází.

Použitá metodika bakalářské práce je založena na studiu a následné analyze informačních zdrojů. Po nastudování odborné literatury je následná aplikace získaných informací na konkrétní příklady z pojišťovací společnosti. Po nastudování odborné literatury jsem dlouho přemýšlel jak nejvhodněji aplikovat praktickou část až jsem se rozhodl ji aplikovat na dílčí vrstvy IT architektury pojišťovací společnosti.

3 Co je architektura

S pojmem architektura se můžeme setkat jak v IT, tak například i ve stavebnictví. Pod pojmem architektura si představíme obor, zabývající se návrhem či designem. Architektura je jedním z klíčových nástrojů tvůrců Informačních systémů. Pod pojmem architektura IT si lze představit návrh, jak zrealizovat funkční model organizace, kde je optimalizované využití technologií IT. Též lze si to představit jako plán, jak vybudovat IT zázemí společnosti, aby vše bylo plynulé, propojené a funkční. V architektuře by mělo být zahrnuto vše od umístění kabeláže přes hardware až po software. Člověk zabývající se architekturou se nazývá architekt. Architektura slouží při vývoji informačního systému podniku jako jeden z klíčových nástrojů pro tvorbu informačních systémů.

3.1 Kdo je architekt a jak se pozná dobrý architekt?

Architekt je pozice, která figuruje v životním cyklu informačních systémů. Činnost, kterou architekt zastává je široká, proto existují různé specializace. V dnešní době najdeme stejné specializace s různými názvy a významy takže je v tom občas nějaké nedorozumění. Stále platí, že architekt je dominantní role při budování, provozování a řízení IT systémů. Náplň role architektura je návrh a řízení architektury. Využívá k tomu různé nástroje a prostředky. Je vhodné, aby měl odpovídající pozici v organizaci a tím pádem i dostatečné pravomoci.

3.2 Co je náplní práce architekta?

Hlavní náplní práce neboli primární rolí architekta je návrh a řízení architektury v organizaci. Architekt má různé nástroje a prostředky, které může a měl by využívat při návrhu a řízení architektury. Je potřeba, aby architekt, měl dostatečné pravomoci odpovídající pozici v organizaci, jinak může dojít k problému, že nebude mít dostatečné pravomoci či oprávnění na provedení potřebných operací. Velmi často se práce architekta zaměřuje na malování modelů. Například může nastat situace, kdy architekt dostane od týmu otázku „Jaké modely máme tvořit a používat.“, zde je nutno podotknout, že není tak moc důležité jaké modely využívají, ale jestli se především zabývají řízením architektury.

Řízení architektury znamená manažerskou složku v práci architekta a pod tímto pojmem si lze představit, že je potřeba pochopit situaci, podmínky a potřeby. Dále je nutné si umět opatřit potřebné informace a umět se správně dotazovat. Správná formulace dotazu je

například „Vyhovovalo by vám, kdyby to řešení vypadalo a fungovalo takto? “ na místo dotazu „Jaké jsou vaše požadavky?“.

3.3 Vhodné řešení

Je to takové řešení, které vyhovuje potřebné situaci (dané situaci), parametrům, standardům a je optimalizován z ekonomického hlediska i z hlediska kvalitativních parametrů. Dále by toto řešení mělo mít nějakou „vizi“ rozvoje odpovídající rozvoji potřeb uživatelů dané organizace. Víceméně se nejedná o finální řešení, jak by se mohlo zdát, ale je to spíše návrh či plán rozvoje architektury organizace.

3.4 Validace řešení

V případě, že již máme nějaké řešení, je potřeba provést validaci tohoto řešení. To znamená, že je zapotřebí zkontrolovat jestli vše odpovídá vypracovanému modelu a jestli provozní parametry odpovídají našim potřebám.

Mezi časté chyby lze uvést následující příklad: Po povodních byla a je při pojišťování staveb náklonnost k ověřování zda budova stojí v rizikovém pásmu či nikoliv. Tyto informace se nacházejí v systému GIS (Geografický informační systém). V tomto případě architekt propojí systém na sjednávání on-line pojištění (web service) se systémem GIS v dané organizaci. Zde nastává problém při špatné validaci řešení v tom, že systém GIS musí být nainstalován na dostatečně výkonném stroji (serveru). Proč? Jelikož v případě, že by systém byl nainstalován na PC, které by leželo v kanceláři tak by tento PC nezvládal v nějakém obstojném režimu poskytovat službu, tudíž by to bylo velice zdlouhavé a v případě veliké vytíženosti by mohlo dojít až ke kolapsu stroje a systém by čekal a čekal pořád na GIS systém. Jedná se tedy o to, aby vše bylo domyšleno a správně aplikováno. Čím více návrh ověřím, tím větší šanci mám, že můj návrh je dobrý a na nic jsem nezapomněl.

3.5 Komunikace řešení

Pokud již mám navržené řešení dobře ověřené neboli validované tak jej musím tzv. komunikovat. Je to většinou formou dokumentů neboli závazných pravidel a standardů. Dále je to i pomocí osobních prezentací. Jednoduše řečené je nutné, aby člověk, který bude uvedený design architektury implementovat, tomu rozuměl a měl vše potřebné.

3.6 Kontrola

Je nutné i samozřejmé, že je potřeba zajistit, aby se vývoj udával správným směrem, který architekt určil. Pokud je cokoliv jinak, je potřeba zjistit proč a případně proti tomu něco podniknout. Například nechat opravit vyvíjené řešení a případně řešitelskému týmu poskytnout dostatečnou podporu formou rozhodnutí, rad či konzultací. Samozřejmě může chyba být v samotném návrhu architektury a pak je potřeba ji opravit.

4 Význam architektury

Architektura informačního systému by se dala přirovnat důležitostí k plánům pro stavbu či opravu baráku. Zvláštností je, že při stavbě či opravě baráku jednotlivcem, nikoliv firmou je samozřejmostí využití alespoň jednoduchých náčrtků dané nemovitosti, zatímco v případě architektur podniku (organizací) se můžeme setkat s tím, že při tvorbě složitějších informačních systémů dochází někdy k tomu, že je systém vybudován bez jakékoliv představy architektury. V tomto případě dochází k tomu, že do podniku se nakupuje různorodý hardware, software (základní a aplikační) se nakupuje od různorodých dodavatelů nikoliv podle plánu, ale podle okamžité potřeby uživatele a není zde řešeno, zdali budou všechny komponenty dohromady kolaborovat a tvořit integrovaný systém. Vlastně by bylo možné v přirovnání ke stavbě nemovitosti zjednodušeně říci, že je potřeba aby dům, který stavíme, byl obyvatelný a dalo se v něm pohodlně a pokud možno levně bydlet. Když zůstanu ještě u přirovnání ke stavbě budovy tak si to lze též představit jako situaci, kdy v jednom baráku více majitelů bytů nakupují materiál a staví podle své okamžité potřeby. V tomto případě může docházet k následujícím situacím:

- Materiál nebo část materiálu je nepoužitelný, jelikož se při stavbě ukázalo, že materiál není na stavbu vůbec vhodný, anebo jej není potřeba takové množství. Například

software, který je nakoupený, ale není nikým využíván, jelikož není v daném podniku vůbec potřeba.

- Může nastat stav, kdy nám schází jiný materiál. Neboli v daném informačním systému schází software na pokrytí daných požadavků uživatelů.
- Dále může dojít ke stavu, kdy nám neschází pouze materiál, ale také stavební stroje a specialisté v dané oblasti. V případě informačního systému by se jednalo o málo výkonné vývojové prostředí bez nástrojů typu CASE, bez specialistů na tvorbu informační strategie, bez specialistů na počítačové sítě a tak podobně.
- Také může dojít k tomu, že se často musí bourat, jelikož nikoho ve vhodné chvíli nenapadlo, že někde musí být prostup pro vodovod, elektřinu a další instalace. V rámci informačních systému se jedná o nákup softwarových komponent, jejichž integrace není možná, anebo realizace požadavků jsou v rozporu s logikou aplikace či s požadavky jiných uživatelů.
- Dochází ke vzniku tzv. nepoužitelných prostorů. Prostorů, kde nelze zabudovat okna ani dveře. Neboli nakoupený (drahý) software a hardware, ale z důvodu nekompatibility jej nelze využít a stává se nevyužívaným.
- Nedochází k využití standardů, tudíž jeden byt má například zásuvky podle české normy, jiný zas podle americké či britské normy a tak podobně. Jedná se tedy o různé uživatelské rozhraní (GUI), různé provozní platformy daných aplikací a podobně.
- Když dům po dlouhé době (delší než se předpokládalo) a po vyčerpání rozpočtu, i několikrát navyšovaného zdánlivě k nastěhování, tak jde o stavbu, na které se podepsaly rozličné stavební styly. V rámci informačních systému si to lze představit tak, že dochází k použití vzájemně těžko integrovaných metodik a nástrojů.
- Jestliže byl průběh stavby strastiplným obdobím tak dochází k tomu, že bydlení v domě se stává zcela nesnesitelným. Pokud dojde k havárii (praskne voda a podobně) tak nikdo neví, kde je hlavní uzávěr vody, nebo hůře ani neví, jestli nějaký uzávěr vody existuje. Jelikož k domu neexistuje žádná dokumentace tak jej nelze opravovat. Tedy jedná se o obtíže s údržbou softwaru za předpokladu, že dokumentace neexistuje či není aktuální.

Kvalitní architektura informačního systému / ICT je do jisté míry lék na výše popsané problémy, které mohou nastat. Je zapotřebí zdůraznit, že je potřeba, aby architektura byla kvalitní a došlo k ošetření problémů, které mohou nastat a dalším možným lékem je multidimenzionální přístup k vývoji informačních systému.

5 Podstata a účel architektury

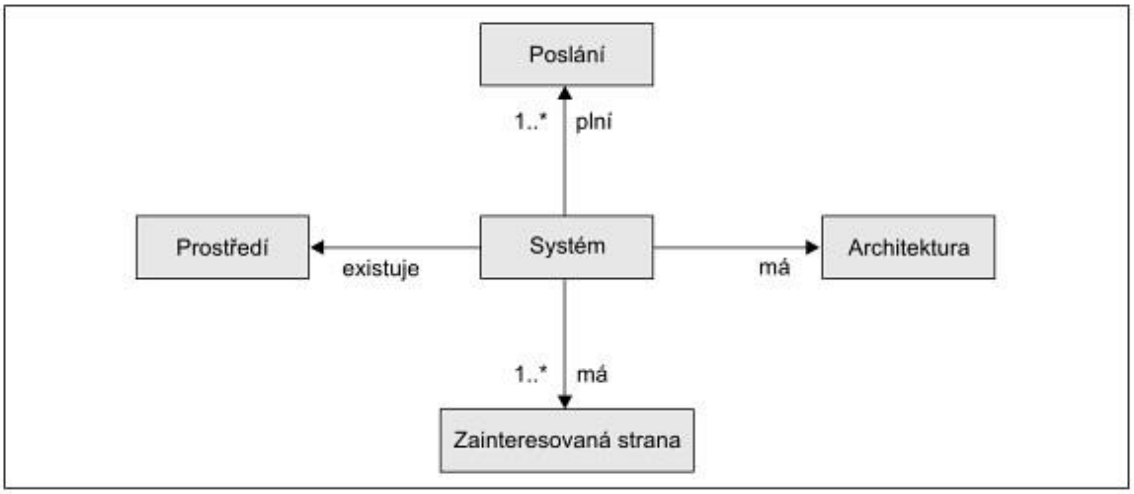
Jeden z náhledů na architekturu IT je definován jako *„Schéma zohledňující všechny podstatné dimenze návrhu informačního systému“.*[1]V současné době představují prostředky IS (informační systém) / ICT jeden za faktorů, které ovlivňují úspěšnost a s tím související výkonnost organizací. Při tvorbě je potřeba respektovat řadu hledisek a vazeb, aby došlo k plné podpoře podnikových procesů a bylo možné realizovat jejich potenciál. Architektura systému je prostředkem, který umožňuje zachytit a jasně popsat vztahy pro různé typy zainteresovaných pracovníků ICT, ale i ze strany byznysu. Architektura IS/ICT je při řešení integrovaných informačních systémů významná z následujících hledisek:

- Vytváří poměrně stabilní rámec řešení IS/ICT kam se v průběhu času (vývoje) zařazují postupně jednotlivé komponenty podle plánu, který je předem připravený. Komponenty se začleňují podle technologických, ekonomických a dalších možností, avšak s vazbami, které jsou předem definovány na ostatní komponenty IS/ICT.
- Je to významný komunikační prostředek mezi vedením organizace, byznys analytiky a vývojáři. Jedná se o komunikaci při formulaci představ o koncepci řešení IS/ICT. Také zajišťuje vzájemné porozumění investorů, uživatelů a řešitelů. Porozumění v tom smyslu, které aplikace, data a rozhraní budou implementovány v tomto okamžiku.
- Zajišťuje stabilitu vývoje IS/ICT je pokud je architektura navržena dostatečně otevřená a předvídající změny, které mohou nastat a mohou být předpokládány. Je potřeba vzít v potaz rychle se vyvíjející technologickou ICT oblast. V případě, že jedna z komponent informačního systému nevyhovuje tak ji vyměníme za jinou, která je vyhovující. Tato procedura nesní zapříčinit zhroucení celé stavby IS/ICT!
- Umožňuje zohlednit hlavní požadavky na vlastnosti IS/ICT již v počátku a tudíž je možné z ní pak odvíjet soudržný specifikace jednotlivých ICT projektů.
- Je významná z ekonomického pohledu. Umožňuje minimalizaci nákladů za chybné projekty či chybně zadané projekty. Také nám to umožňuje minimalizaci nákladů na celkovou rekonstrukci IS/ICT v důsledku další neudržovatelnosti.

[1] Dohnal J., Pour J.: Architektury informačních systémů v průmyslových a obchodních podnicích, 1997

Ohledně architektury při vývoji a jeho následné údržbě panuje v odborné komunitě shoda, ale dlouho neexistovala přesná definice pojmu architektura systému a také shoda o tom, jak by měla být popsána. V roce 2000 byla přijata organizací IEEE norma, která formuluje doporučené postupy pro popis architektury systému IEEE-Std-1471-2000 [2]. Norma byla převzata organizací ISO a díky tomu se stala základem normy ISO/IEC 42010:2007[3].

Tato norma má za cíl standardizovat praktiky a prvky popisu architektury a tím usnadnit komunikaci zainteresovaných stran o prioritách a struktuře informačního systému. Následující obrázek vyobrazuje základní pojmy a principy související s architekturou informačního systému a jejím popisem podle této normy.



Obrázek 1 konceptuální model systému a jeho architektury[4]

- **Systém** je to soubor komponent uspořádaných účelově k dosažení určitého cíle či skupiny cílů.
- **Prostředí** jedná se o kontext, který určuje okolnosti a nastavení vývojových, provozních, politických, sociálních, regulačních a jiných kritických vlivů na systém a jeho další rozvoj.
- **Zainteresované strany** jsou to jednotlivci (jednotlivci, týmy, organizace), kteří mají zájem na systému, anebo jsou ve vztahu k systému. Mezi zainteresované strany například patří uživatel, zákazník, vývojář, manažer podniku, poskytovatel služby, dodavatel a další.

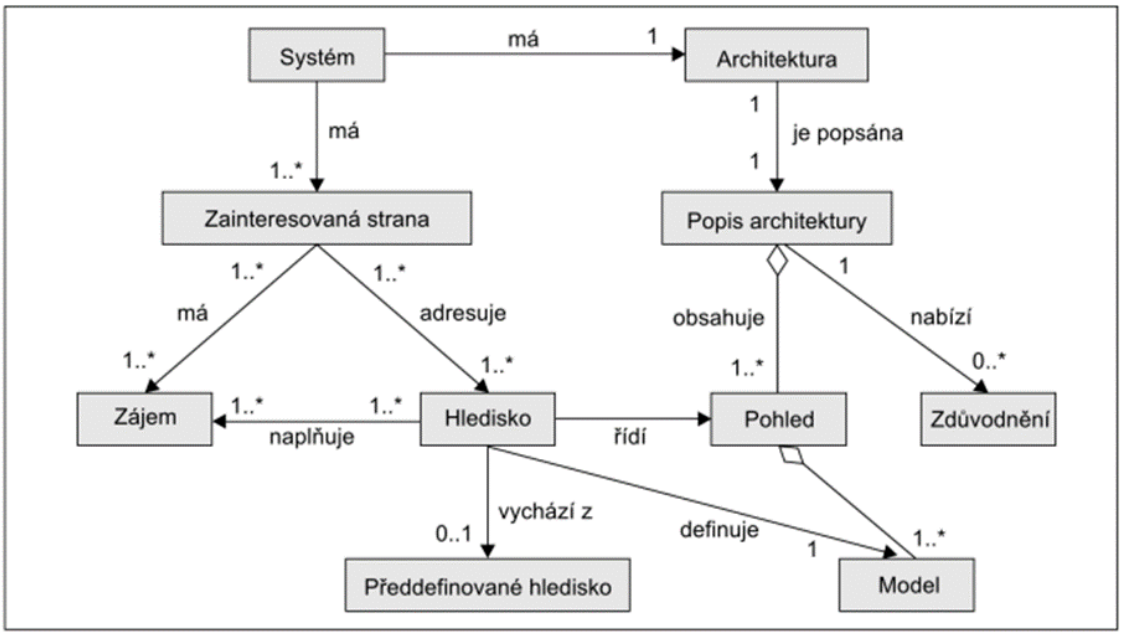
[2]IEE-Std-1471-2000 : Recommended Practice for Architectural Description of Software-Intesinve Systems.

[3] ISO/IEC 42010:2007 Systems and software engineering – Recommended practice for architectural description of software-intensive systems.

[4] Buckner T, Voříšek J., Buchalcevová A., Stanovská I., Chlapek D., Řepa V.: Tvorba informačních systémů: Principy, metodiky, architektury, 2012

- **Poslání** - systém je vytvořen proto, aby plnil definované poslání.
- **Architektura** je takové uspořádání systému, které tvoří komponenty a vztahy mezi nimi, včetně vztahů k prostředí a také principy, které řídí jeho návrh a rozvoj.

Architektura je dokumentována pomocí popisu architektury (Architectural description). Norma rozlišuje mezi popisem architektury (konkrétní informační artefakty) a mezi architekturou systému (konceptem). Na obrázku 2 můžeme vidět zachycení konceptuálního modelu popisu architektury. Popis je složen z částí, jež se nazývají architektonické pohledy. Každý pohled (View) určuje určité zájmy zainteresovaných stran na vlastnostech systému. Architektonické hledisko je respektováno tzv. architektonickými pohledy, jež představují dílčí architekturu. Pohled je reprezentován modely (Model). Modely, které jsou součástí pohledu, je definováno pomocí hlediska (ViewPoint). Architektonické hledisko specifikuje konvence pro vytvoření pohledu a jeho použití. Můžeme jej chápat jako šablonu nebo vzor, na jehož základě vytváříme pohled. Hledisko nám definuje účel, pro koho je daný pohled určen a techniky pro jeho vytvoření a analýzu. Hlediska mohou být definována předem a uložena v knihovně. Pro popis dané architektury formuluje zdůvodnění neboli Rationale.



Obrázek 2 Konceptuální model popisu architektury[5]

Když potřebujeme popsat architekturu, postupujeme následovně: Nejprve definujeme systém, jehož architekturu budeme popisovat (to jsou prvky/komponenty, ze kterých se skládá

[5] Buckner T, Voříšek J., Buchalcevová A., Stanovská I., Chlapek D., Řepa V.: Tvorba informačních systémů: Principy, metodiky, architektury, 2012

a samozřejmě vazby mezi nimi). Dále definujeme zainteresované strany a jejich zájmy. Tímto jsou určeny pohledy. Každému pohledu definujeme hledisko. Poté vytvoříme modely, které tvoří pohled. Souhrn všech těchto pohledů nám pak tvoří popis architektury, který opatříme příslušnou identifikací a formulujeme účel a zdůvodnění.

6 Kritéria uplatňovaná při posuzování architektur

Za předpokladu, že bychom zadali navrhnout architekturu informačního systémů jedné organizace více nezávislým týmům, s velmi vysokou pravděpodobností dojde každý z týmů k jinému výsledkům, které se budou od každého odlišovat. Zde nám ovšem vzniká otázka, která z daných architektur bude nejlepší či nejvýhodnější pro danou organizaci. V tomto případě pro posouzení kvality architektury musíme definovat kritéria hodnoceni a váhy kritérií. Váhy kritérií musí být odlišené, jelikož cíle podnikových informatik i prostředí, ve kterých působí, se liší tak se musí lišit i váhy daných kritérií. Hodnotící kritéria, ale mohou býti analogická, protože vycházejí z požadovaných vlastností systému. Kritéria mohou zahrnovat například:

- **Efektivita** – uskutečňuje informační systém vybudovaný podle dané architektury podnikové cíle? Uskutečňuje je dobře?
- **Nástroj komunikace** – uskutečňuje navrhovaná architektura vhodný nástroje pro komunikaci mezi byznysem a informatikou a mezi jednotlivými řešiteli informačního systému?
- **Náklady tvorby, údržby a provozu** – budou celkové náklady na tvorbu, údržbu a provoz informačního systému vybudovaného na základě architektury úměrné přínosům?
- **Čas** – je možné dosáhnout s danou architekturou požadovaných cílů v požadovaném čase?
- **Kvalita a bezpečnost** – zajistí nám daná architektura dostatečnou a požadovanou kvalitu ochrany a bezpečnost ICT služeb?
- **Flexibilita** – je možné daný informační systém s příslušnou architekturou pružně přizpůsobovat měnícím se požadavkům?

- **Škálovatelnost –** je možné přizpůsobovat výkon podnikové informatiky, zejména objem poskytovaných služeb výkyvům v potřebách byznysu?
- **Rizika –** pomáhá nám architektura řešit identifikovaná rizika provozu a rozvoje informačního systému? Jaká nová rizika jsou spojená s novou architekturou?
- **Soulad se standardy –** je daná architektura v souladu se závaznými standardy?

7 Přístupy k tvorbě architektur

V posledních dvou desetiletí byla vyvinuta řada přístupů k návrhům IS/ICT. Zde popíši nejvýznamnější z nich.

7.1 Podniková architektura (Enterprise Architecture, EA)

Podle Molnára[6] je možné se na podnik dívat ze dvou hledisek. Častější pohled na podnik je jako na racionální systém, jenž klade důraz na organizaci jako celek a na účel, pro jenž byla vytvořena.

Podniková architektura vznikla koncem 80. let minulého století. Cílem bylo řešit vznikající problémy v IS/ICT. Mezi první problémy patřilo, jak zvládat rostoucí složitost informačních systémů a dalším problémem bylo, jak využít stále se zvětšující potenciál ICT pro byznys a jak sladit možnosti ICT s potřebami byznysu.

Historie Podnikové architektury neboli EA se začal psát v roce 1987. V tomto roce vyšel v publikaci IBM Systém Journal článek nazvaný „Rámec pro architekturu informačních systémů. V tomto článku byly popsány výzvy a vize podnikové architektury. Zásadní výzvou bylo zvládnutí zvětšující se komplexity distribuovaných systémů. Zachman měl vliv na jeden z prvních pokusů vytvořit Podnikovou architekturu (Enterprise Architecture) pro ministerstvo obrany USA. Tento pokus byl znám jako „Technical Architecture Framework for Information Management“ neboli TAFIM a byl představen v roce 1994. Podniková architektura slibovala podstatně lepší sjednocení projektů technické povahy s potřebami byznysu. Využil toho americký kongres, který v roce 1996 schválil zákon známý jako Clinger-Cohenův, jenž je označován také jako Information Technology Management Reform Act. Vyžadoval, aby všechny státní agentury podnikaly stanovené k zefektivnění jejich investic v ICT. Dozor nad

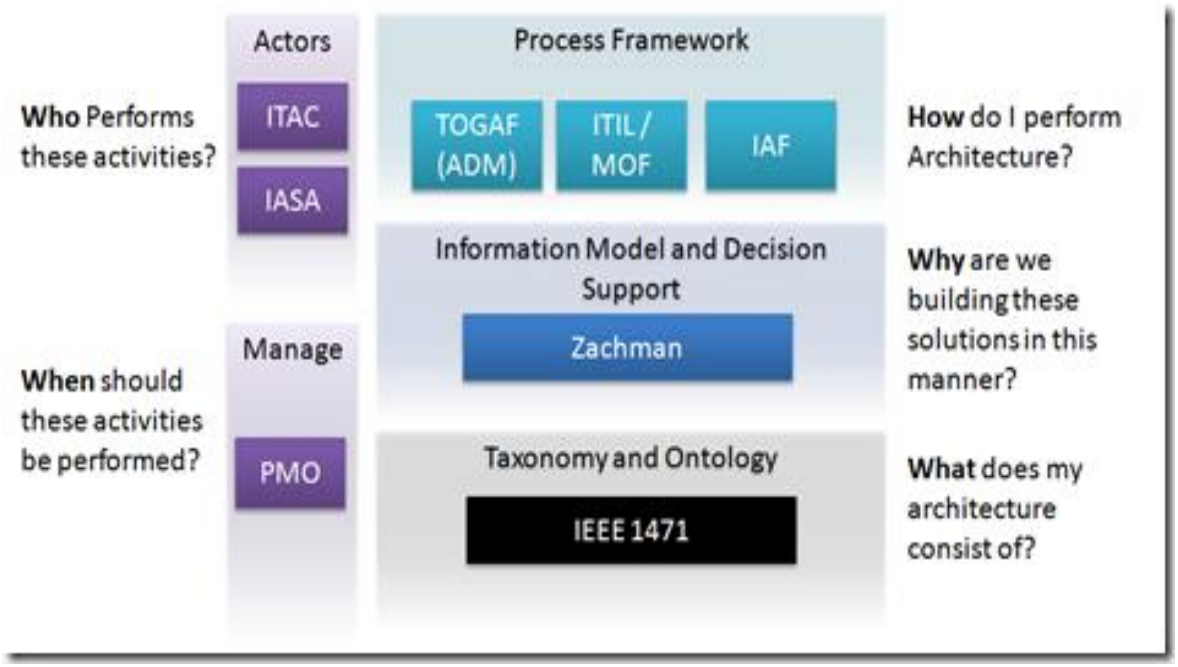
[6] Molnár Z.: Podnikové informační systémy, 2009

touto snahou byl tvořen tzv. CIO sněmem, který se skládal z CIO všech hlavních státních institucí. CIO sněm začal v roce 1998 pracovat na prvním velkém projektu známém jako Federal Enterprise Architecture Framerwork neboli FEAF. TAMIF a veškerá práce odvedená na tom nepřišla vniveč, ale přešla na Open Group. Tato skupina jej přetvořila na standard známý jako The Open Group Architectural Framework (TOGAF).

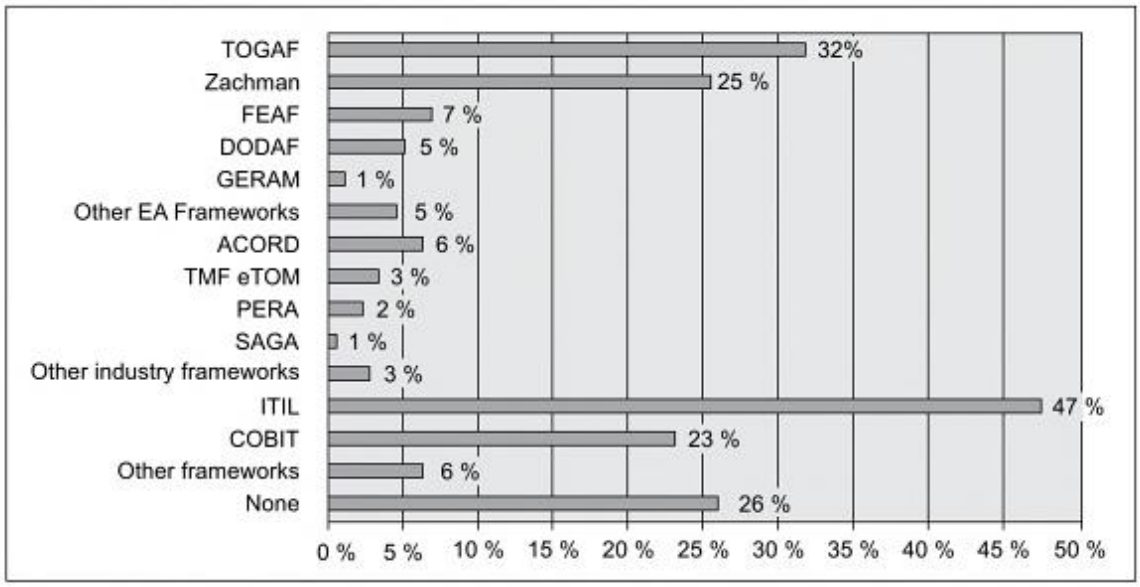
Podniková architektura (Enterprise Architecture, EA) je koncept, přístup, prostředek a nástroj sloužící pro vyjádření fundamentálního uspořádání vztahu mezi informačním systémem a byznysem. Podniková architektura nás vede k naplnění poslání organizace a respektuje okolní prostředky, konzistentně dodržuje formulované principy návrhu a rozvoje.

7.2 Architektonické rámce

Architektonický rámec je množina zájmů, zainteresovaných stran, předdefinovaných hledisek a pravidel definující vazby hledisek, jež byly definovány pro popis dané architektury ve specifické oblasti. Architektonický rámec je nástroj pomáhající řešitelům v návrhu a budování Enterprise Architecture. Použití architektonického rámce zrychluje a zjednodušuje celý proces tvorby architektury. Snižuje nám riziko opomenutí nějakého důležitého prvku architektury. Obsah rámců se může lišit v závislosti na přístupu k podnikové architektuře – viz obrázek 3. Vždy není ideální využívat pouze jeden rámec. Vhodným řešením je, či může být kombinace rámců ták, aby pokryly optimálně potřeby budování informačního systému organizace.



Obrázek 3 StandardyEA
(zdroj:http://blogs.msdn.com/blogfiles/mikewalker/WindowsLiveWriter/ArchitectureStandards_DEEC/image_thumb.png)



Obrázek 4 Standardy EA graf[7]

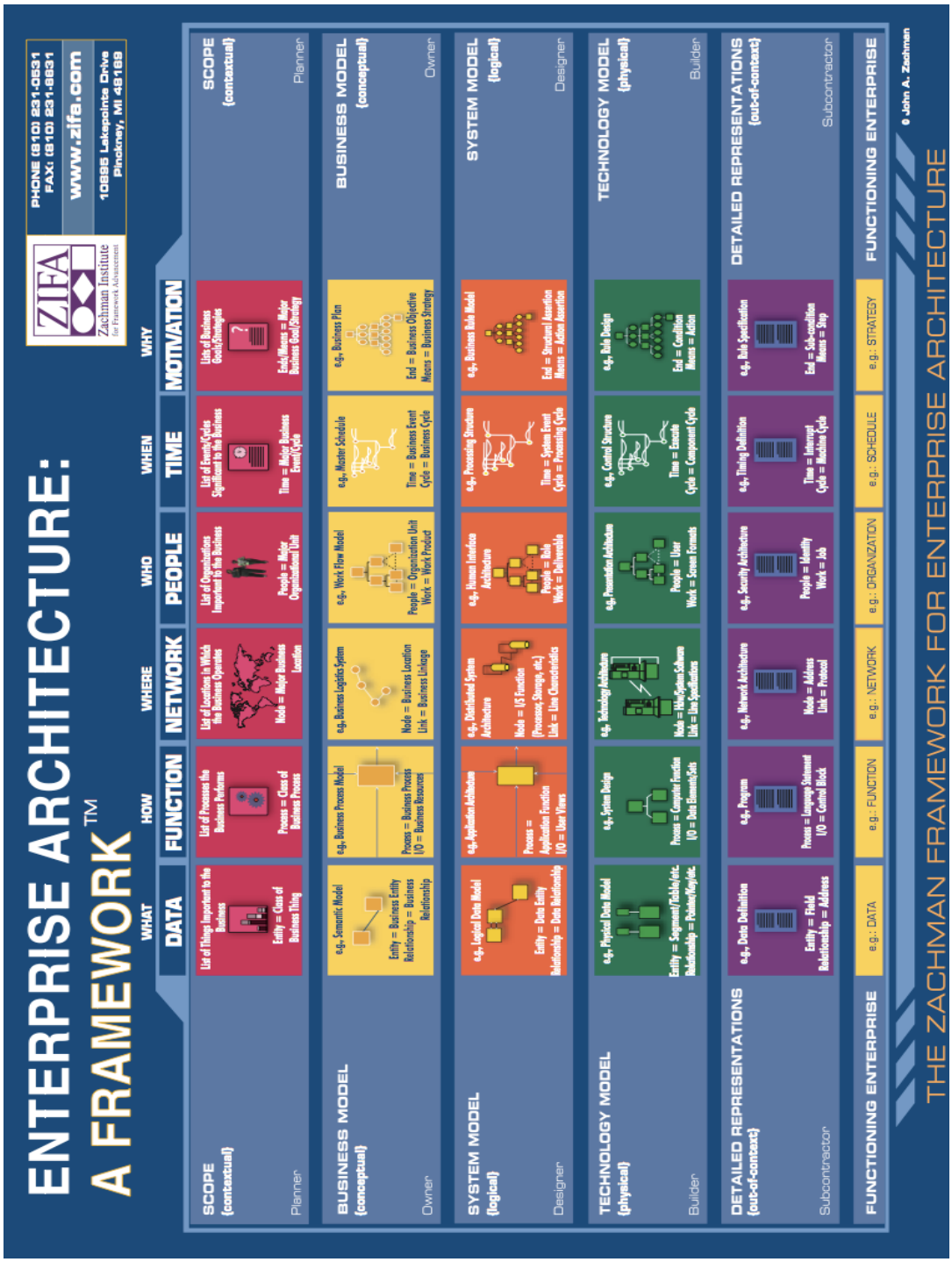
Na obrázku 4 lze vidět, že v současné době nejpoužívanějšími rámci jsou ITIL a TOGAF. Je to poprvé, kdy rámec TOGAF předstihl Zachmanův rámec v použití. Dokumentuje nám to posun akcentů pří řízení IS/ICT více k byznysu. Zajímavý z hlediska budoucnosti je podíl ITIL. ITIL jako takový má jiné zaměření než TOGAF. ITIL je primárně zaměřen na IT procesy související s řízením životního cyklu ICT služeb a neakcentuje podnikovou architekturu. Vzhledem k tomu, že se ITIL a TOGAF vhodně vzájemně doplňují

[7] Buckner T, Voříšek J., Buchalcevová A., Stanovská I., Chlapek D., Řepa V.: Tvorba informačních systémů: Principy, metodiky, architektury, 2012

a že podíly jak ITIL tak TOGAF v budoucnu zřejmě ještě porostou, bude vhodné vymyslet způsob jak tyto dva rámce vzájemně integrovat.

7.3 Zachmanův rámec

Zachmanův rámec poskytuje jednoduchou a zároveň logickou strukturu pro organizaci a klasifikaci celkového pohledu na podnik a jeho informační systém. Základem rámce je dvourozměrná matice, kterou můžeme vidět na obrázku 5. Řádky matice tvoří role, jež řeší IS/ICT na různých úrovních abstrakce.



Obrázek 5 Zachmanova matice *(zdroj: http://www.embt.cz/data/wysiwyg/img/ERS_XE/Zachman.png)*

Zachmanem doporučené role jsou:

- **Projektant (Planner)** zabývá se předmětem a cíli podnikání, vymazuje podnikové procesy, lokality, klíčové podnikové útvary a klíčové události, na které podnik musí reagovat.
- **Vlastník (Owner)** definuje podnikovou strategii, procesy a jejich vzájemné vztahy (konceptuální modely byznysu).

- **Návrhář (Designer)** definuje nám aplikační a informační architekturu neboli logické modely a modely zachycující implementaci podnikových pravidel (business rules).
- **Stavitel (Builder)** definuje fyzickou strukturu dat, technologickou infrastrukturu, softwarovou architekturu a uživatelské rozhraní aplikace (fyzický model technologie).
- **Dodavatel (Subcontractor)** převádí předchozí úroveň do implementační úrovně.

Sloupce představují různé obsahové dimenze, které by měly být modelovány a popsány (data, funkce, síť, lidé, čas, motivace). Každá buňka matice reprezentuje danou obsahovou dimenzi modelovanou na dané úrovni abstrakce. Síla rámce je v tom, že poskytuje způsob uvažování podniku v organizované podobě. Vlastně jde o to, že podnik může být popsán a analyzován z různých pohledů a různých úrovní detailu. Dovoluje řešitelům informačního systému podniku se zaměřit na vybrané aspekty podniku a vidět je v kontextu celku. Slabinou Zachmanova rámce je jeho přílišný důraz na ICT problematiku. Je to důvod, proč je tento rámec poslední dobou na ústupu.

7.4 TOGAF

TOGAF si lze představit jako manuál, který nám radí, jak strategicky a dlouhodobě plánovat a řídit podnikovou architekturu. Tento rámec je v současné době k dispozici v jeho deváté verzi a neustále se vyvíjí. TOGAF je založen na definici čtyř základních architektur a jejich vztahů, jenž popíši níže.

- Byznys architektura – definuje nám byznys strategii, procesy a organizační strukturu.
- Architektura informačního systému složenou z:
  - Aplikační architektury – popisuje aplikace, vztah aplikací k byznys modelu a jejich vzájemné vazby.
  - Informační architektury - popisuje nám strukturu a organizaci dat v podniku.
  - Technologické architektury - popisuje hardware a základní software, jenž jsou potřeba pro vzájemnou komunikaci/integraci aplikací a pro běh aplikací.

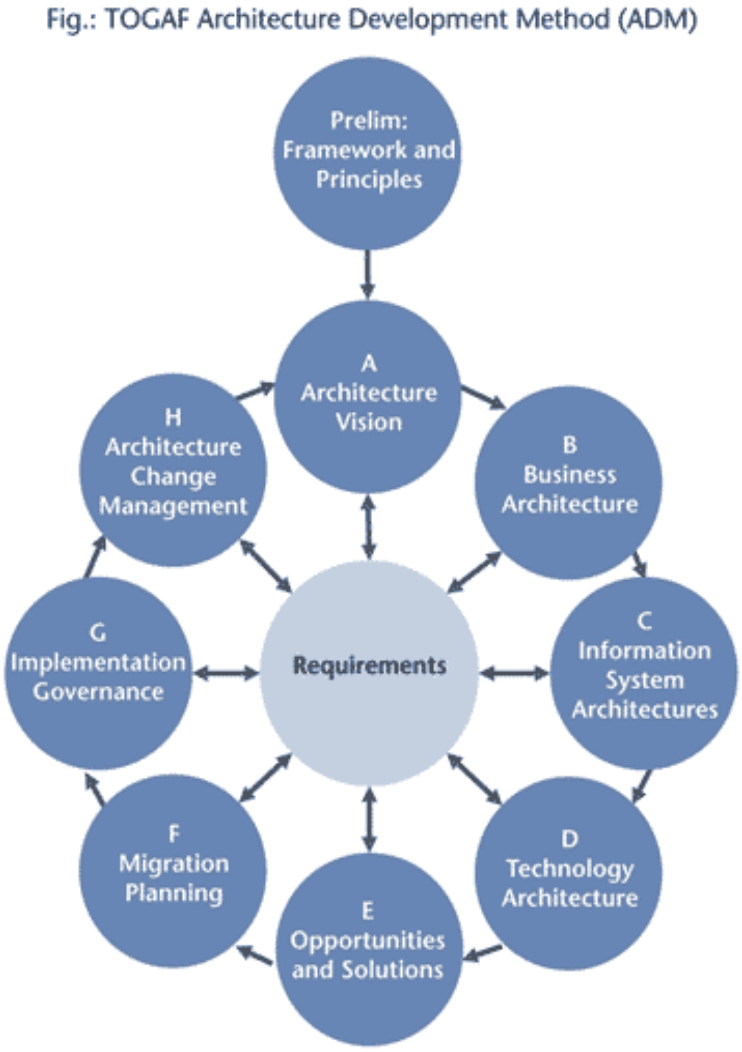
Součástí je metoda vývoje podnikové architektury ADM (Architecture Development Method). ADM rozděluje řízení podnikové architektury do osmi fází. Znázorněno je to na obrázku 6.

- **Příprava** - zde, v této fázi dochází k seznámení všech zainteresovaných s plánem vývoje podnikové architektury.

A. **Vize architektury** – formuluje rozsah, vizi a strategii vývoje podnikové architektury.
B. **Byznys architektura** – popisuje cílovou a stávající byznys architekturu. Ukazuje a zachycuje rozdíly mezi těmito dvěma architekturami.
C. **Architektura informačního systému** – navrhuje aplikační a informační architekturu informačního systému.
D. **Technologické architektura** – definuje nám technologickou infrastrukturu, na které budou provozovány aplikace informačního systému.
E. **Příležitostí a řešení** – zde se určuje, co se bude vytvářet, kupovat a znovu využívat a jak se budou implementovat architektury popsané v předchozích fázích.
F. **Plánování migrace** – tato fáze vytváří plán migrace a seřazuje podle důležitosti jednotlivé projekty.
G. **Zavedení governance** – formuluje, jak se bude realizovat ohled nad implementací návrhu.
H. **Řízení změn architektury** – v této fázi se dohlíží na běh systému, a pokud se objeví nějaká nezbytná změna, zahájí celý cyklus znovu.

ADM definuje doporučené pořadí fází v rozvoji architektury, ale již nám neurčuje, do jakého detailu má být podniková architektura propracována. Stanovit si to musí každá organizace sama. Zde se dostáváme k problému, zde je v TOGAF počet architektur dostatečný a vhodný pro modelování celého podniku a jeho informačního systému.

Vzhledem k tomu, že byznys architektura je mapována přímo do aplikační, informační a technologické architektury zde vyvolává otázku, jak zachytit různé varianty sourcingu ICT služeb, ve kterých podnikoví architekti nemohou ovlivnit architektury, jež souvisí s externě poskytovanými službami. Například, když podnik využívá SaaS, pak řešitelé IS/ICT v dané organizaci již nemají kompetence k tomu, aby ovlivnili aplikační a technologickou architekturu aplikačních služeb, jež jsou dodávané externími dodavateli. OpenGroup si je tohoto nedostatku vědoma a proto bude zajímavé sledovat, jak tuto problematiku vyřeší další verze TOGAF. Metodika MMDIS řeší tento problém vsunutím architektury ICT služeb mezi byznys architekturu a dílčí IT architektury.



Obrázek 6 Fáze ADM *(zdroj: http://www.qualified-audit-partners.be/user_images/Logos/togaf_method.gif)*

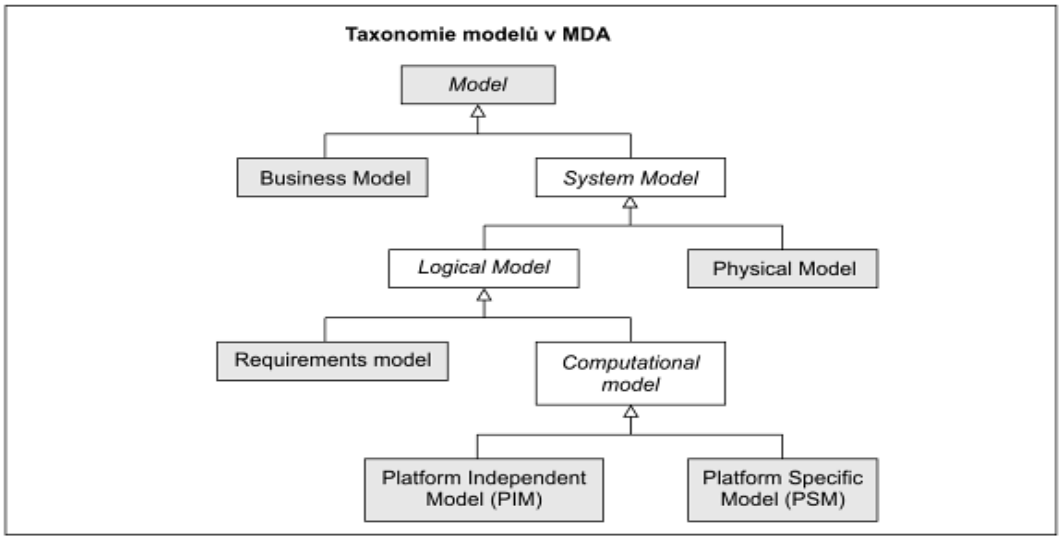
7.5 Modelem řízená architektura

Mezi další významné přístupy k tvorbě architektur IS/ICT patří modelem řízená architektura (Model Driven Architecture, MDA). Myšlenky, na nichž je tato architektura založena nejsou nijak nové. Příležitost oprostit se od technologických problémů a zaměřit se na věcné problémy sebou již přinesl vznik jazyka Cobol a rozvoj strukturovaných metod v 60. letech minulého století. Myšlenka oddělení konceptuálního (analytického) pohledu od pohledu návrhu a implementace je také jedním ze stěžejních principů metodiky MMDIS (princip tří architektur). MDA vychází ze skutečnosti, že s postupem na vyšší úrovně abstrakce množství změn v systému klesá. Dopad těchto neustálých změn technologií je možné omezit jen na část modelu, tedy na jeho nižší vrstvy.

Při vývoji MDA se nejprve vytvoří platformově nezávislý model (Platform Independent Model – PIM), jenž představuje věcnou funkcionalitu a chování systému. Pomocí nástrojů MDA se PIM mapuje na zvolenou platformu (například Corba, Java/EJB, XML/SOAP) a generuje se platformově specifický model (Platform Specific Model – PSM). Na závěr se generuje implementační kód pro příslušnou technologii. Nástroje podporující MDA také umožňují zpětné inženýrství (reverse engineering), a díky tomu je možné vytvořit modely již existujících systémů například pro účely integrace aplikací. MDA generátory aplikují současné i architektonické vzory a návrhové vzory. Na obrázku 7 lze vidět znázorněnou hierarchii modelů využívajících v rámci MDA.

Modely vyznačené výplní představují konkrétní modely, které se při MDA vývoji vytvářejí. Ostatní modely jsou abstraktní a představují pouze logické členění. Byznys model (Business Model) popisuje věcné aspekty problémové oblasti bez ohledu na to, zda budou automatizovány. Model systému (Systém Model) pak popisuje počítačový systém. Logický model (Logical Model) zaznamenává logiku systému prostřednictvím modelu tříd a modelu chování, zatímco Fyzický model (Physical Model) popisuje fyzické artefakty a zdroje používané při vývoji a provozu – soubory s modely, soubory zdrojového kódu, spustitelné soubory, archivní soubory.

Model požadavků (Requirements Model) popisuje počítačový systém z uživatelského pohledu a neber v úvahu technologické aspekty řešení, ale Výpočetní model (Computational Model) popisuje počítačový systém včetně technologických aspektů řešení. Hlavní (klíčovou) úlohu v MDA plní platformově nezávislý model (Platform Independent Model, PIM) a platformově specifický model (Platform Specific Model, PSM). PIM představuje konceptuální model dané problémové oblasti, jenž je nezávislý na platformě. Při mapování Platform Independent Model na konkrétní platformu vznikají nejen artefakty příslušné platformy (například Interface Definition Language, IDL), ale též platformově specifický model, který mnohem lépe zachytí sémantiku řešení pro specifickou platform.



Obrázek 7 Taxonomie modelů v MDA[8]

Platformově specifický model představuje jak technologickou, tak věcnou sémantiku aplikace. Je to model UML vyjádřený v dialektu UML, které se označuje jako UML profil. Profil odráží technologické prvky cílové platformy. Základem MDA je modelovací jazyk UML (Unified Modeling Language). Silná stránka jazyka UML spočívá v jeho metamodelu a v možnosti převodu modelů do XML na základě standardu XML. UML je základ pro tvorbu platformově specifických modelů. Příkladem profilů je například UML profile for Corba, jenž nám definuje mapování z PIM do PSM pro technologii Corba. Dalším standardem, na němž je MDA postavena je MOF (Meta Object Facility). MOF je jazyk pro vytváření konstruktů modelů, tedy metajazyk. Používá stejné modelovací konstrukty pro diagram tříd jako UML, takže je možné používat pro jeho tvorbu běžné modelovací nástroje podporující UML.

MOF akceptuje realitu, tedy bere v potaz, že různé modely vyjadřují různé pohledy na systém. Architektura MOF je tvořena čtyřmi metaúrovněmi.

- Úroveň M3 MOF, množina konstruktů pro definici metamodelů
- Úroveň M2 metamodely definované pomocí konstruktů MOF
- Úroveň M1 modely tvořené instancemi konstruktů M2 metamodelu, například třída Zákazník
- Úroveň M0 objekty a data, neboli instance konstruktů M1 modelu, například zákazník Petr Macek

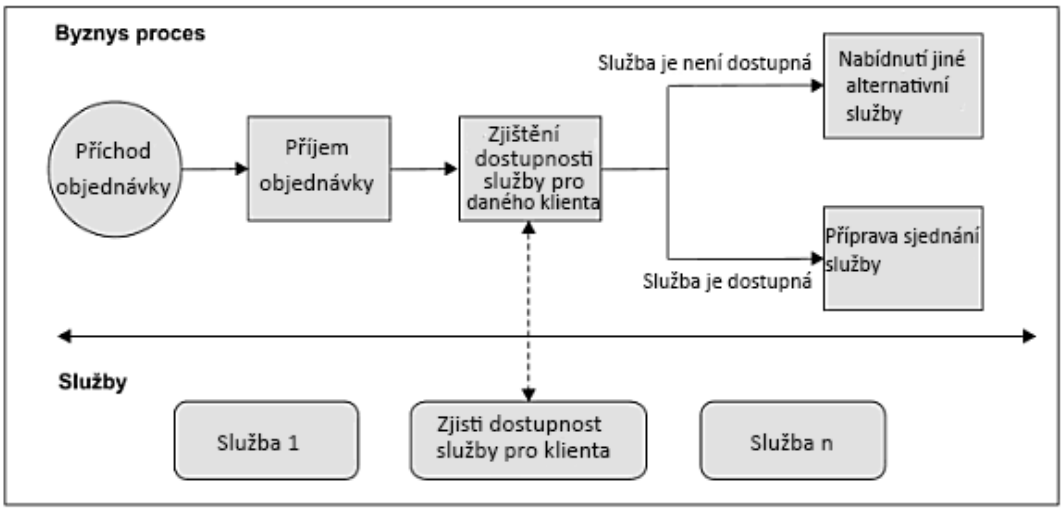
[8] Buckner T, Voříšek J., Buchalcevová A., Stanovská I., Chlapek D., Řepa V.: Tvorba informačních systémů: Principy, metodiky, architektury, 2012

Strategická inciativa OMG MDA představuje převratný vývoj v technologiích a silný důraz na efektivnost prostředků vložených do informačních technologií. Cílem MDA je zachovat investice vložené do informačních technologií tím, že vývojáři se zaměří na vytváření platformově nezávislých modelů, které představují věcnou oblast, ale nejsou dotčeny změnami v technologiích. Z těchto modelů potom s využitím MDA nástrojů, které se dnes začínají již objevovat, budou automatizovaně generovány platformově specifické modely i vlastní implementace.

7.6 Architektura orientovaná na služby (Service Oriented Architecture, SOA)

Důraz na přínosy v podnikových procesech a snaha o zefektivnění informačních technologií vede stále více k prosazování konceptu služeb. Služby představují spojovací článek mezi podnikovými procesy a informačními technologiemi. Hlavní zájem architektů se soustředil v posledním desetiletí zejména na tzv. aplikační služby. Cílem aplikačních služeb je poskytování funkcionality softwarových aplikací byznys procesům – viz obrázek 8. Příklady služeb jsou: „založ účet“, „sjednej pojištění“ a podobně. Současný zájem o aplikační službu je především spojen s technologií webových služeb a architekturou orientovanou na služby.

Technologie webových služeb je tvořena množinou standardů (průmyslových) založených na XML, jejž specifikují jazyk pro definici služby (Web Service Definiton Language, WSDL), komunikační protokol (Simple Object Access Protocol, SOAP) a registr pro publikování a vyžádání služby (Universal Description, Discovery and Integration, UDDI). SOA je aplikační architekturou, kde je funkcionalita aplikace definována jako množina služeb, které mají přesně definovaný interface (vstupy, výstupy). Služba může být volána přímo z jednotlivých činností byznys procesu. Podstatou SOA na standardech založené služby a jsou volně spřažené (loosely coupled). Volná spřaženost znamená, že jednotlivé služby jsou na sobě nezávislé. Uživatel služby se nezajímá o to, na kterém počítači počítačové sítě je provozována, nebo jak je služba implementovaná. Tato skutečnost je známa jen registru UDDI.



Obrázek 8 Standardní aplikace s upraveným interface podle pravidel SOA[9]

8 Architektury podle MMDIS (Multidimensional Management and Development of Information System)

Architektury v MMDIS určují pro daný systém (organizace, informační systém organizace, softwarová aplikace a podobně), z jakých konkrétních komponent se bude skládat a jaké vzájemné vazby budou. Současně určuje principy provozu a vývoje IS, jejichž cílem je dosáhnout požadovaných vlastností systému. Vlastnosti jsou například: pokrytí požadované funkcionality, dostupnost, včasnost, správnost a důvěryhodnost funkcí a informací, výkonnost a efektivita.

Metodika MMDIS při návrhu architektur vychází z konceptuálního modelu SPSPR[10], který jako hlavní komunikační nástroj mezi byznysem a IT uvažuje definici ICT služeb a jejich využití byznysem. Metodika aplikuje na architektury využívané při vývoji Informačního systému princip vrstevnosti a rozlišuje tři úrovně architektur. Úrovně reprezentují byznys úroveň, ICT úroveň a úroveň řešící vztah mezi byznysem a ICT – viz obrázek 10. V první úrovni je pohled na celý podnik. Ve druhé úrovni je pohled na služby informačního systému poskytované byznysu a ve třetí nám pohled dekomponuje globální architekturu IS/ICT na dílčí IS/ICT architektury. Tento přístup určuje zodpovědnosti různých typů manažerů za architektury, přičemž tyto zodpovědnosti jsou ve shodě se zodpovědnostmi

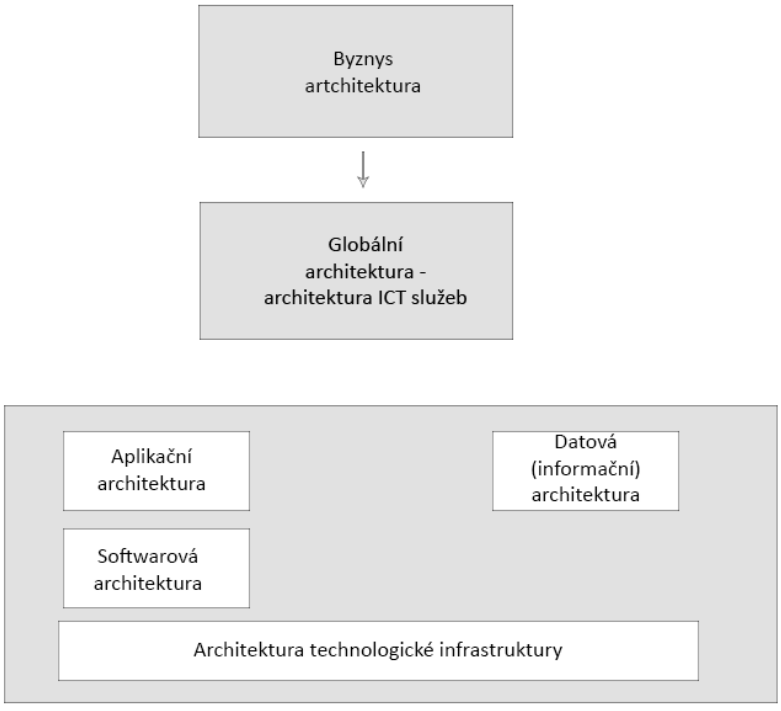
[9] Převzato a doplněno z Buckner T, Voříšek J., Buchalcevová A., Stanovská I., Chlapek D., Řepa V.: Tvorba informačních systémů: Principy, metodiky, architektury, 2012

[10] Model SPSPR řeší vztah mezi řízením podnikových procesů a řízením podnikové informatiky. Definuje základní zodpovědnosti byznys a ICT manažerů při řízení vztahu byznys – podniková informatika. V modelu jsou explicitně využity principy multidimenzionality, vrstevnosti, standardizace, kooperace, procesního přístupu a měřitelnosti.

v modelu SPSPR. Hlavní zodpovědnost za byznys architekturu mají byznys manažeři (vlastníci byznys procesů a vrcholový management).

Za dílčí IS/ICT architektury mají zodpovědnost ICT manažeři. Zodpovědnost za globální architekturu ICT služeb mají společně ICT a byznys manažeři. Byznys architektura nepatří do oblasti informatiky. Představuje celkový pohled na podnik, byznys modely, byznys procesy a na jeho cíle.

Byznys architektura je velmi významná pro podnikovou informatiku, protože je základním východiskem pro architektury informačního systému. ICT manažeři na její tvorbě úzce spolupracují. V metodice MMDIS se ICT architektury navrhují na dvou úrovních. První informatickou úrovní architektur je globální IS/ICT. Zachycuje jednotlivé vrstvy ICT služeb a jejich vzájemné vztahy. Druhou informatickou úrovní architektur jsou dílčí architektury IS, které dekomponují globální architekturu z několika pohledů – softwarového, aplikačního, datového a technologického. Návrh architektur je zakončen definováním vazeb mezi architekturami. Například kterými ICT službami je podporován daný proces, kterými aplikacemi jsou zajišťovány jednotlivé ICT služby atd.



Obrázek 9 Tři úrovně architektur v MMDIS[11]

[11] Převzato z Buckner T, Voříšek J., Buchalcevová A., Stanovská I., Chlapek D., Řepa V.: Tvorba informačních systémů: Principy, metodiky, architektury, 2012

9 Byznys architektura

V případě byznys architektury je systém, na kterém architekturu definujeme celá organizace a její významné okolí (dodavatelé, instituce státní správy, zákazníci). Primárními komponentami, jejichž vztahy a uspořádání architektura definuje, jsou podnikové funkce, podnikové procesy a podnikové útvary. Tvorba byznys architektury úzce navazuje na globální podnikovou strategii, jež určuje:

- Strategické podnikové cíle
- Hlavní předmět podnikání
- Podnikatelský model, tedy model tvorby hodnoty pro zákazníka
- Sourcing strategii
- Základní podniková pravidla (business rules)
- A identifikovala hlavní zdroje potřebné pro dosažení strategických cílů

Byznys architektura dále rozpracovává výše uvedené body. Definuje architektonické pohledy a s nimi související architektonické modely:

- Model významného okolí podniku
- Funkční model podniku
- Procesní model podniku
- Model organizační struktury a model rozmístění útvarů v lokalitách podniku

Součástí architektury je návrh vazeb mezi modely. Jedná se zejména o vazby funkčního/procesního modelu na model organizační struktury. Vazba definuje pravomoci a zodpovědnosti jednotlivých útvarů a funkčních míst za podnikové funkce a podnikové procesy. Zodpovědnosti mohou být definované jako centralizované (jeden útvar zodpovídá za danou podnikovou funkci v rámci celého podniku) nebo distribuované (zodpovědnost je dislokována do nižších organizačních úrovní).

10 Globální architektura IS/ICT

Globální architektura je systém, na kterém definujeme architekturu, informační systém organizace a jeho významné okolí (IS dodavatelů, zákazníků, institucí státní správy). Primárními komponentami, které definuje architektura a jejich uspořádání a vztahy, jsou ICT služby. V globální architektuře je primárním modelem model ICT služeb a jejich vztahů:

- Vztahy k uživatelům ICT služeb (k zaměstnancům, zákazníkům, dodavatelům apod.)

- Vztahy k funkčním oblastem organizace
- Vztahy k podnikovým procesům, na jejichž podporu služba slouží
- Vztahy k jednotlivým aplikacím, které zajišťují funkcionalitu služby
- Vztahy k poskytovatelům ICT služeb (určuje, kdo ICT službu dodává)

Hodnocení navržené globální architektury lze hodnotit následujícími kritérii:

- Jestli služby ICT poskytují potřebnou funkcionalitu a veškeré informace potřebné pro hladký běh a efektivní průběh podnikových procesů.
- Objemové a kvalitativní.
- Charakteristika ICT služeb v souladu s požadavky byznysu.
- Cena služeb ICT je taková, že nepřevyšuje cenu srovnatelných služeb na trhu a nesnižuje konkurence schopnosti vůči hlavním konkurentům.
- Náklady na správu služeb jsou přijatelné. Počet služeb je přijatelný a může růst zejména v případě, kdy je zvolená granularita služeb velmi jemná. To znamená, že jedna služba obsahuje malý počet funkcí.

10.1 ICT služby a jejich definice

ICT služba jsou konzistentní aktivity a/nebo informace dodávané poskytovatelem služby příjemci služby. Služba je vytvářena procesy, které při svém průběhu spotřebovávají ICT zdroje (hardware, software, data, lidé). Poskytovatel dodává službu příjemci na základě dohodnutých obchodních a technických podmínek. SLA (Service Level Agreement) je část smlouvy, kde jsou obchodní a technické podmínky dodávky služby a obvykle upřesňují tyto charakteristiky ICT služby:

- Komu, kde, kdy, kým je služba poskytována.
- Co je předmětem, jak, jaký je objem, Jaké jsou kvalitativní charakteristiky a jaká je cena poskytované služby.
- Jakými znalostmi a/nebo technologiemi musí disponovat příjemce služby, aby mohl službu konzumovat, mechanismy zajištění kontinuity služby v případě havárie, bezpečností pravidla a mechanismy
- Forma, obsah a cyklus reportování o průběhu dodávky služby a pravidla pro realizaci změn služby

Nad službou jsou definovány operace, ze kterých se skládá životní cyklus služby ICT:

- Identifikace a popis potřeby
- Definice, návrh/design, vytvoření/sestavení, testování a přizpůsobení služby

- Instalace/nasazení služby
- Zahájení provozu, provoz dodávání služby zákazníkovi včetně podpůrných služeb
- Monitor a řízení kvality, škálování objemu, údržba/modifikace/kontinuální zlepšování služby
- Zpoplatnění
- Ukončení provozu

Nad vybranými službami ICT poskytovatele se realizují tyto operace/aktivity:

- Strategie, architektura, sourcing, integrace a standardizace služeb

10.2 Typy ICT služeb

Při návrhu architektury ICT služeb je klasifikačním hlediskem předmět služby. Toto hledisko nám říká a soustřeďuje se na to, co poskytovatel příjemci v rámci služby dodává. Dále se soustřeďuje na to, jaký vztah má tato dodávka k byznysu příjemce. Podle předmětu služby dělíme do dvou skupin a to do ICT služeb byznysu a ICT služeb rozvoje IS/ICT viz tabulka 1.

ICT služby byznysu (koncovým uživatelům)	ICT služby rozvoje IS/ICT
Informační	Vývoj softwaru
Aplikační	Implementace aplikace
Infrastrukturní	Integrace IS
Podpůrná	Rozvoj technologické infrastruktury
smíšená	Poradenství atp.

Tabulka 1 Členění služeb podle předmětu služby[12]

První skupinu služeb tvoří služby, které bezprostředně podporují koncové uživatele a podnikové procesy. Mezi ně patří informační, aplikační, infrastrukturní a podpůrné služby.

Ve druhé skupině jsou zahrnuty služby, které pomáhají k rozvoji IS/ICT. Tyto služby nejsou konzumovány byznysem, ale slouží k vývoji nových ICT služeb byznysu či ke změně stávajících služeb. Především zahrnují vývoj a dodávku požadovaného softwaru či ICT infrastruktury. Řeší se za pomoci klasického informatického projektu s definovaným cílem,

[12] Převzato z Buckner T, Voříšek J., Buchalcevová A., Stanovská I., Chlapek D., Řepa V.: Tvorba informačních systémů: Principy, metodiky, architektury, 2012

rozpočtem a dobou řešení. Na každý projekt se uzavírá smlouva, ve které je definováno podíl případného externího dodavatele, informatického útvaru a uživatelského útvaru na projektu.

Po ukončení projektu dojde u dané služby k „překlopení“do nové nebo změněné ICT služby byznysu. Do této skupiny patří široká škála ICT služeb, z nichž nejvýznamnější jsou: vývoj softwaru, implementace aplikace, integrace IS, rozvoj technologické infrastruktury a poradenství. V globální architektuře IS/ICT vystupují v roli komponent ICT služby byznysu. To je jedním z důvodů, proč je budeme dále podrobněji charakterizovat.

10.3 Informační služba

Touto službu dodává poskytovatel příjemci požadovanou informaci, například stav kurzů na burze cenných papíru, předpověď počasí a podobně. Informace jsou dodávány v požadované struktuře, formátu a čase. Dodaná informace může být produktem softwarové aplikace, takže funkcionalita aplikace je pro příjemce služby nepodstatná. Je však podstatná pro poskytovatele služby, jelikož funkcionalitou aplikace požadovanou informaci získává a dodává. Poskytovatel informační služby na rozdíl od poskytovatele služeb aplikačních odpovídá za správnost a aktuálnost dodaných informací. U některých informačních služeb je problém v tom, že se na ně vztahuje autorský zákon, který omezuje příjemce služby v tom, jak může s poskytnutou informací nakládat. Výhodou této služby je jejich snadná replikace s nízkými náklady, tedy poskytovatel velmi lukrativní informační služby může dosáhnout velmi vysokých ekonomických efektů – například Google.

10.4 Aplikační služba

Předmětem této služby je funkcionalita byznys aplikace, například účetnictví a podobně. Uživatel užívá funkcionalitu aplikace, která je provozována na vhodné ICT infrastruktuře. Data, jež jsou aplikací zpracovány, jsou buď zákazníka (účetnictví…) anebo poskytovatele (vyhledávač Google a podobně). Může se též jednat i o kombinaci dat poskytovatele a zákazníka (objednávky, Google Earth s objekty označenými zákazníkem). Poskytovatel služby odpovídá za správnost transformace dat, majitel dat (ten, jenž data vkládá) odpovídá za správnost vstupních dat. V případě služeb zaměřených na podnikovou sféru funckionalita aplikace realizuje jednu nebo více aktivit podnikového procesu. V tomto případě mohou nastat dvě situace:

- Služba podporuje jen vybrané aktivity podnikového procesu
- Služba realizuje celý podnikový proces

Aplikační služba je velmi často dodávána v balíčku. Tento balíček obsahuje služby, které jsou bezprostředně spojeny s aplikací (školení uživatelů, help desk a podobně). Významným představitelem aplikačních služeb je SaaS (Software as a Service). Zde externí poskytovatel poskytuje velkému počtu zákazníků přes internet funkcionalitu aplikace, jenž běží na jeho technologické infrastruktuře. Příkladem může být Škoda auto, jenž díky této aplikaci umožňuje zákazníkům nakonfigurovat si ze standardních dílů svůj nový vůz.

10.5 Infrastrukturní služba

Předmětem této služby je vybudování a provoz ICT infrastruktury (servery, koncové stanice a tak dále), jež jsou potřebné pro bezchybný chod aplikace nebo aplikací. Do infrastrukturních služeb patří:

- Služby správy technologických zdrojů, jenž zahrnují například pořizování a správu koncových zařízení.
- Služby komunikačních kanálů, jenž zahrnují řízení a integraci všech elektronických komunikačních kanálů, které organizaci propojují se zákazníky a partnery.
- Služby komunikační, orientované na zajištění přenosových tras mezi jednotlivými místy zpracování byznys aplikací. Prostřednictvím počítačových a telekomunikačních sítí.
- Služby správy dat. Cílem je nabídnout takové prostředí, jež nám umožní řídit data (uchovávat, zpřístupňovat…) nezávisle na aplikacích.
- Služby spojené s řízením rizik a bezpečností ICT. Zde je zahrnuto zajištění stanovené úrovně informační bezpečnosti. Je reprezentována sadou vlastností (důvěrnost, integrita, dostupnost…) a také vytvoření prostředí důvěryhodnosti ICT pro všechny zainteresované strany.

Jiný přístup ke členění infrastrukturních služeb přišel s nástupem cloud computingu, ve kterém se obvyklé infrastrukturní služby člení na IaaS (Infrastructue as a Service) a PaaS (Platform as a Service). V případě IaaS jsou zahrnuty služby bez vývojového prostředí a integračních nástrojů, ale u PaaS jsou zahrnuty včetně vývojového prostředí a integračních nástrojů.

10.6 Podpůrné služby

Jsou to takové služby, jež jsou potřebné či vhodné pro zajištění služeb informačních, aplikačních a infrastrukturních služeb. Jedná se především o školení, implementaci, přizpůsobení (customizaci) a integraci aplikací, služby help desku, ale i o pomocné služby poradců při návrhu služby a podobně. V praxi bývají výše zmíněné služby úzce provázány a vznikají tak smíšené služby. Příkladem může posloužit služba typu e-learning. Tuto službu na základně výše uvedené klasifikace lze přiřadit do skupiny aplikačních služeb.

Za předpokladu, že si ji představíme jako webovou službu, jenž poskytuje svojí funkcionalitou uživateli možnost vybrat požadovaný kurz, studovat kurz, psát testy, hodnotit testy a podobně pak nepochybně této kategorii náleží. Její součástí je, ale i obsah dostupných materiálu pro studium. Aktivity spojené s tvorbou a poskytování těchto materiálu lze řadit mezi služby informační. K požadovanému fungovanání e-learningové služby bude zapotřebí rovněž provozování aplikace na určité ICT infrastruktuře. Zajištění tohoto provozu je předmětem služeb infrastrukturních a konečně je pro fungování služby typu e-learning zapotřebí i vyškolení uživatelů v přípravě a publikaci studijních materiálu, jenž patří mezi služby podpůrné.

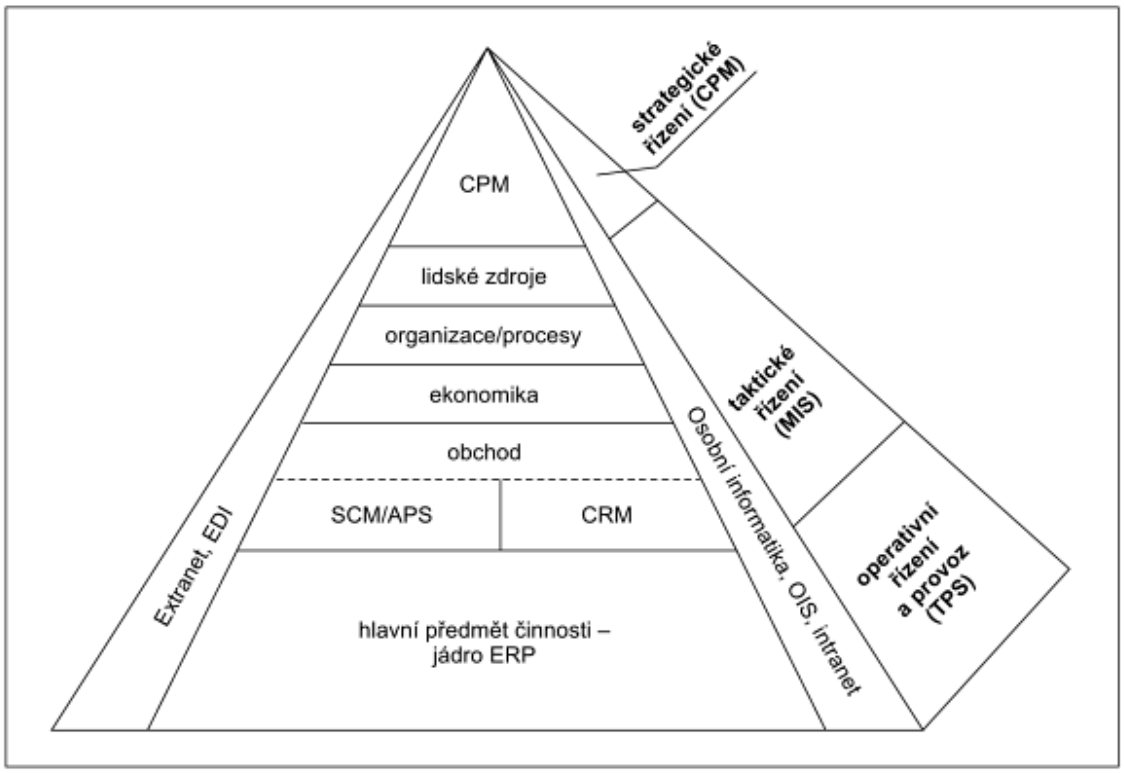
11 Definice globální architektury IS/ICT

Definice této architektury zahrnuje:

- Model služeb ICT
- Katalog služeb ICT (id, název služby, skupina služeb v modelu služeb ICT, do které služba patří, stav služby – plánovaná, v provozu, základní parametry služby).

11.1 Model služeb ICT

Model služeb ICT znázorňuje graficky hlavní skupiny služeb ICT (stavební bloky globální architektury). Každá skupina služeb se obvykle vztahuje k určité skupině podnikových funkcí. Dále se vztahu k určité úrovni podniku (strategické, taktické, operativní) – viz obrázek 10.



Obrázek 10 Obvyklá struktura modelu ICT služeb v podniku[13]

Globální architektura IS/ICT organizací se obvykle skládá z pěti základních bloků. Většinou z TPS, MIS, CPM, osobní informatika, OIS, intranet a externí komunikace (EDI, extrakt). V dalším textu stručně popíši jednotlivé bloky.

11.2 TPS – Transaction Processing Systém

Je to blok, jenž je zaměřený na podporu hlavní činnosti podniku na operativní úrovni. Je nejspecifičtějším blokem celkové architektury. Závislý je na charakteru podniku (výrobní, obchodní, dopravní atd.). U obchodních podniků je závislý na typu prodávaných komodit, na teritoriích nákupu a prodeje a podobně.

11.3 MIS – Management Information Systém

Je to blok, jenž je orientovaný na řízení podniku na taktické úrovni, která zahrnuje ekonomická, organizační a obchodní hlediska. Struktura toho bloku je značně standardizována a je velmi podobná i pro podniky různých typů. Pojetí MIS je založeno na integraci procesů ve třech základních liniích – obchodně-logistické, finančně-účetní a průřezové. Mezi služby obchodně-logistické patří zejména: řízení vztahů s dodavateli, řízení

[13] Buckner T, Voříšek J., Buchalcevová A., Stanovská I., Chlapek D., Řepa V.: Tvorba informačních systémů: Principy, metodiky, architektury, 2012

vztahů se zákazníky, nákup a podobně. Mezi finančně-účetní patří: hlavní kniha, závazky, pohledávky, controlling (nákladové účetnictví), majetek, pokladna, práce a mzdy a finanční řízení. Průřezové služby mají celopodnikový charakter a řadíme mezi ně: organizaci a správu, řízení lidských zdrojů, marketing, legislativu a řízení jakosti.

11.4 CPM – Corporate Performance Management

Je to blok orientovaný na strategické řízení výkonnosti podniku. Služby řízení podniku na vrcholové úrovni získávají data z TPS a MIS, dále z externích informačních zdrojů. Tato data agregují a vytvářejí časové řady a vzájemné vazby. Výstupy z CPM pak slouží jako podklady pro strategická rozhodnutí členů vrcholové managementu.

11.5 Osobní informatika, OIS (Office Information Systém), intranet

Mezi služby tohoto bloku patří:

- Vytváření, distribuce dokumentů
- Řízení týmové práce na dokumentu
- Podpora řízení projektů
- Plánovací kalendář
- Sledování úkolů
- Evidence odešlé a došlé pošty, elektronická pošta, videokonference, elektronické diskusní skupiny
- Tvorba a prohlížení webových stránek, archivace dokumentů
- Workflow, intranet

Služba workflow je průřezová přes všechny úrovně řízení organizace. Tato služba řídí tok dokumentů podnikem a případně i průběh celopodnikových procesů. Intranet je služba, jenž zprostředkovává komunikaci mezi zaměstnanci podniku. Dále zpřístupňuje jednotlivým rolím v podniku jim přiřazené funkce.

11.6 EDI (Electronic Data Interchange) a extranet

Je to blok, který zajišťuje komunikaci podniku s jeho významným okolím, tedy se zákazníky, dodavateli, státními institucemi atd. Role v rámci globální architektury IS/ICT

podniku roste spolu se stále větším prosazováním aplikací využívajících internet a aplikací pro standardizovanou výměnu dat mezi obchodními partnery. Model architektury IS/ICT (globální) vychází obvykle z funkčního modelu podniku, jenž byl zpracován v rámci byznys architektury. Krom návrhu jednotlivých stavebních bloků globální architektury zahrnuje též i hrubý návrh vazeb mezi stavebními bloky. V návrhu jsou uváděny vazby, jež musí být respektovány nehledě na to, o jak je daný stavební blok řešen.

Další krokem návrhu globální architektury IS/ICT je návrh katalogu služeb ICT. Katalog vychází z hrubého návrhu neboli modelu služeb ICT, jenž dekomponuje bloky služeb do jednotlivých služeb a pro každou službu specifikuje základní parametry:

- Zákazník služby (službu objednává a platí)
- Uživatel služby (službu využívá)
- Dodavatel služby (zodpovídá za vývoj a provoz služby)
- Obsah služby (funkcionalita, školení, služby help desku)
- Objem služby (počet uživatelů, objem dat, počet transakcí)
- Kvalita služby (dostupnost, doba odezvy, spolehlivost, bezpečnost)
- Cena služby

11.7 Katalog ICT služeb a parametry služeb

Parametry, jež výrazně ovlivňují dílčí návrh architektur jsou obsah, objem, kvalita a cena služby. Ovlivňuje nám to vývoj aplikací, které budou služby ICT zajišťovat včetně nároků na provoz podnikového IS/ICT. Poslední tři parametry výrazně ovlivňují dimenzování a zabezpečování technologické infrastruktury, na kterých budou aplikace provozovány.

12 Dílčí architektury

Dílčí architektury IS/ICT navazují na globální architekturu, jenž dále prohlubuje návrh budoucího stavu IS/ICT. Jedná se o tyto architektury:

- Aplikační
- Softwarová
- Datová/informační
- Technologické infrastruktury

Je zapotřebí předem upozornit, že v podniku je nutné navrhovat všechny dílčí architektury pouze v případě, kdy podnik vyvíjí i provozu informační systém vlastními silami.

V případě outsourcování, tedy když je vývoj nebo provoz aplikace outsourcován tak některé, nebo všechny dílčí aplikace se v podniku nenavrhují, jelikož odpovědnost za ně přebírá externí poskytovatel.

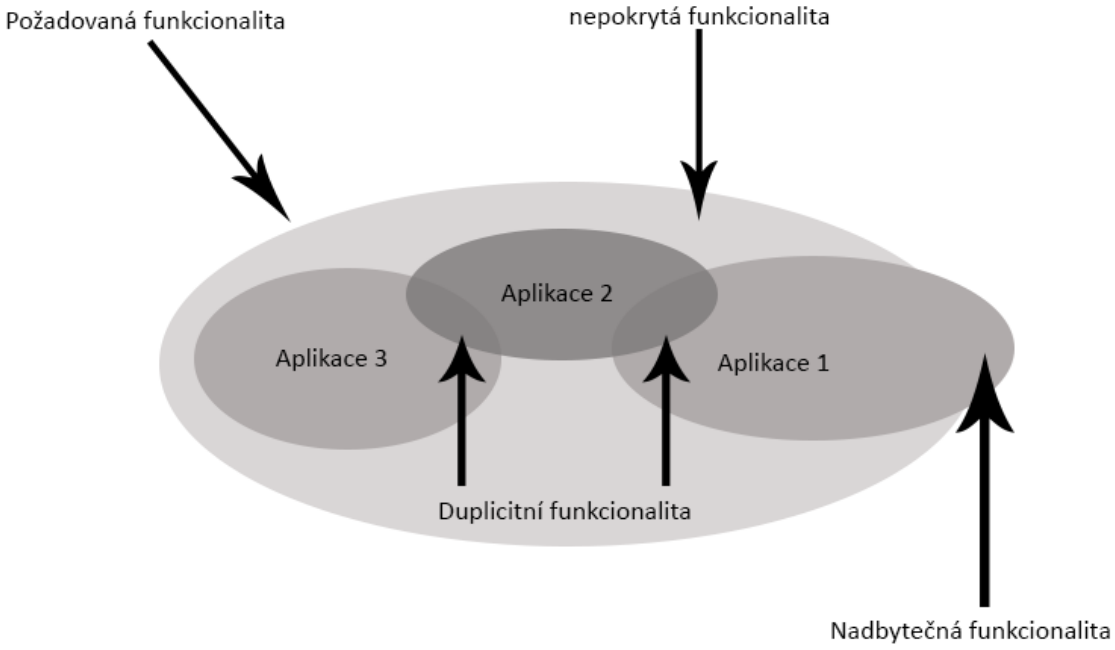
12.1 Aplikační architektura

V této architektuře definujeme architekturu na informační systém. Hlavními komponentami, jenž definujeme jsou softwarové aplikace a definujeme jejich strukturu a vztahy. Aplikační architektura vychází z architektury služeb ICT a říká, kterými aplikacemi je pokryta celková funkcionalita IS a jaké vazby jsou mezi aplikacemi. Mezi aplikacemi a službami ICT je obecně vztah M:N, neboli funkcionalita ICT služby může být zajišťována jednou nebo více aplikacemi. Současně může jedna aplikace svojí funkcionalitou podporovat jednu nebo více služeb ICT.

12.1.1 Kritéria pro hodnocení aplikační architektury

Kvalitu a vhodnost navržené architektury můžeme hodnotit následujícími kritérii:

- Zda stupeň pokrytí splňuje pokrytí požadované funkcionality služeb ICT aplikacemi – viz obrázek 11. V případě, že dojde k tomu, že některá z funkcí aplikačních služeb není zajištěna žádnou aplikací, jedná se o fatální chybu. V případě, že je daná funkcionalita zajišťována dvěma či více aplikacemi tak se nejedná o fatální chybu, nýbrž jen o neekonomické řešení, jež komplikuje uživatelské rozhraní a komplikuje to integraci aplikací. Jestli některá aplikace obsahuje funkcionalitu, která není požadována, nebo jí nikdo nevyužívá tak se jedná o neekonomické řešení, jelikož jsme nakoupili nebo vyvinuli funkcionalitu, jež nikdo nepotřebuje. Vyplývá z toho tedy, že cílem architektury by mělo být pokrýt 100% požadovanou funkcionalitu s minimální duplicitou či minimem nadbytečných funkcí.



Obrázek 11 Hodnocení stupně pokrytí funkcionality požadované ICT službami[14]

- Počet aplikací je dalším kritériem. Je-li informační systém dané organizace řešen velkým počtem aplikací (stovky aplikací), komplikuje a prodražuje to správu, integraci a údržbu aplikací. Kvůli tomu, by mělo být cílem architekta zajistit požadovanou funkcionalitu z malého počtu aplikací.
- Velmi důležité kritérium z hlediska byznysu i z hlediska ICT je integrace aplikací. Lze ho rozdělit do tří dílčích požadavků:
    - Požadavek, aby jeden datový objekt byl v celé datové základně uložen pouze jednou a to i v případě, že s ním pracuje více aplikací. Jedná se o datovou integraci. V případě, že byly informace o klientovi uloženy vícekrát, je zde vysoká pravděpodobnost, že se hodnoty různých výskytů budou lišit a pak není jasné, které údaje odpovídají realitě. Dojde-li k případu, že dvě aplikace mají v datové základně tentýž objekt, musí architekt zajistit synchronizaci těchto duplicitně uložených údajů.

[14] Vlastní tvorba na podkladech z Buckner T, Voříšek J., Buchalcevová A., Stanovská I., Chlapek D., Řepa V.: Tvorba informačních systémů: Principy, metodiky, architektury, 2012

    - Aplikace musejí nabízet svoji funkcionalitu i přes API (Application Program Interface) tak, aby již existující funkcionalitu mohly využívat i jiné aplikace.
    - Integrace uživatelského rozhraní. Jedná se o to, aby principy komunikace v různých aplikacích byly stejné. Tímto se zajistí, že uživatel se nemusí učit různé způsoby komunikace a ani nepozná, že jednotlivé funkce jsou zajišťovány různými aplikacemi.
- Jednotná technologická platforma pro všechny aplikace. Cílem je snížení nákladů na technologické zdroje, ale i na správce těchto zdrojů.
- Náklady na pořízení a provoz jsou kritériem, jež zohledňuje požadavky byznysu na nízkou cenu ICT služeb. Pro architekta to znamená, že musí vybírat takové aplikace, jež vyhovují výše uvedeným kritériím a zároveň mají přijatelné náklady na pořízení a provoz.

12.1.2 Integrace aplikací

Ideálním cílem architekta informačního systému je, aby systém byl zcela integrován. To znamená, že všechny technologie a aplikace spolupracují bez nutnosti manuálních administrativních zásahů. Jak ukazuje praxe, tak tento stav není dosažitelný a pokud se mu chceme přiblížit tak je to velmi nákladné a pracné. Základním důvodem obtížnosti dokonalé integrace jsou takzvané legacy systémy.

Legacy systémy jsou takové systémy, jež byly zprovozněny v minulosti a jsou dosud užívány. Vzhledem k technologickému rozvoji jsou obvykle vytvořeny na již zastaralých technologiích. V případě, že zastaralá aplikace stále vyhovuje, je vhodné ji v podniku zachovat, místo jejího nákladného a pracného nahrazení nově vytvořenou funkcionalitou, na kterou uživatelé nejsou zvyklí, a následně je tedy obtěžuje. Vzhledem k faktu, že k vývoji nových aplikací se obvykle používají spíše aktuální technologie tak v průběhu času v systému narůstá počet různých použitých technologií a roste technologická složitost a náročnost na údržbu. Legacy systém je tedy často z pohledu údržby a úprav zdrojem obtíží, ale na rozdíl aplikace vytvořené na jednoduchých platformách v minulosti jsou na rozdíl od sofistikovaných distribuovaných aplikací provozovatelné a udržovatelné velmi levně.

Nárůst počtu odlišných legacy systémů a provozovaných platforem může vést k tomu, že se systém stává nezvladatelný nebo nerozšiřitelný. Nezvladatelný v tom smyslu, že se objevují problém a výpadky, jenž se nedaří v dostatečné době identifikovat a odstraňovat. Nerozšiřitelný v tom smyslu, že zvážit dopad i drobné změny do funkcionality je velmi

náročné a stává se, že zavedení změn má neočekávané dopady do fungování systému. V takových to případech dochází na integrační projekty. Jsou to projekty, jejichž účelem není konkrétní přínos pro byznys, tedy změny aplikační funkcionality, ale zavedení nových technologií, jenž zajistí lepší spolupráci aplikací. Jako příklad může sloužit zavedení integrační platformy, jenž například pomocí zasílání zpráv nahradí přímé vazby mezi jednotlivými aplikace. Zde bohužel může dojít k tomu, že zavedení této integrační technologie také přispěje k růstu složitosti systému, jelikož časem se z ní stane další legacy systém.

Při tvorbě nové aplikace, jenž má být součástí stávajícího systému je nezbytné kromě byznys analýzy brát v úvahu stávající architekturu systému, aby výsledkem byl dostatečně integrovaný informační systém jako celek. To je však poměrně snadné ve srovnání s tím, aby aplikace nedělala v budoucnosti, například za 10 let obdobné potíže, jako dnes dělají současné legacy systémy. Tedy odpovědné osoby by měly při plánování projektů a zadávání zakázek na budoucí rozvoj a dlouhodobou udržovatelnost a integrovatelnost systémů. Můžeme při tom využít řadu různých metod a přístupů:

- Manuální integrace – data jsou získávána z jedné aplikace, člověk je přečte a ručně je vloží jako vstup do aplikace druhé
- Posílání souborů – jedna aplikace vytvoří soubor s daty jako výstup a uloží jej na smluvené místo nebo jej pošle jiné aplikaci, jež jej použije jako vstup
- Sdílená databáze – aplikace využívají společný datový model a přistupují ke společné databázi
- Vzdálené spouštění procedur – aplikace zpřístupní část své funkcionality pro spuštění jinými aplikacemi, ty ji spustí a vloží vstupní data a získají výstupy
- Posílání zpráv – aplikace mají smluvený komunikační kanál, do něhož posílají asynchronně podle potřeby zprávy a odkud si v definovaných situacích zprávy vyzvedávají

Každý z přístupů k integraci aplikací má své výhody. Nejlevnější, nejsnadnější na správu a s nejnižším dopadem do komplexity je manuální integrace, jenž klade nároky na uživatele, který se stává sluhou informačních technologií, i když by tomu mělo být naopak. Nejsofistikovanější, náročné na standardizaci, ale také s nízkým dopadem do komplexity je integrace pomocí posílání zpráv. Dopad do komplexity, neboli náročnost a nákladovost na úpravy integrace při budoucích změnách je patrně nejdůležitější hledisko těsnosti integrační vazby.

Je nezbytné v případě těsné vazby, tedy tam, kde je práce jedné aplikace závislá na struktuře či datech jiné aplikace, aby při změně jedné aplikace došlo také ke změnám všech závislých aplikací. Nezávislé aplikace jsou v případě volné vazby. Integrace pomocí těsné vazby je velmi rychlá a levná, ale v dlouhodobém pohledu se prodraží i drobné změny. Podíváme se na integraci a souvislosti s relační databází ERP, ESB, EAI, SOA, datawarehouse a ontologiemi. V menších podnicích se začíná využívat software pro podnikové procesy bez ohledu na integraci a složitost. S růstem podniku roste složitost IS a už s existencí dvou aplikací nastává potřeby tyto aplikace nějak propojit. Z hlediska ceny bývají nejvhodnějšími řešeními způsoby pomocí předávání souborů anebo prostřednictvím jedné aplikace k funkcionalitě druhé.

S rostoucím počtem aplikací v systému tyto vzájemné vazby aplikací jedna ku jedné narůstají a stávají se obtížně udržovatelné. Tento způsob se nazývá špagetovou integrací. Nazývá se tak, jelikož pohled na aplikační architekturu se zobrazenými vazbami mezi více aplikacemi připomíná změť těstovin na talíři. Po tzv. ostrůvcích automatizace, jež začaly být propojovány špagetovou integrací vznikla strukturovaná analýza a myšlenka jednotné datové základny pro celý systém. Jednotná datová základna měla být reanalizována nejlépe prostřednictvím relační databáze, jenž tvoří speciální vrstvu v architektuře informačního systému nezávislou na aplikační logice. Jednotlivé aplikace sdílí společná data a přistupují k nim za použití SQL. Ve skutečnosti se nedaří zajistit to, aby všechny aplikace používaly společnou databázi, jelikož nakupované hotové softwary nebo dodavatelské metodiky často počítají s jinou strukturou dat a technologií.

Další myšlenkou bylo použít pro řízení podnikových zdrojů jednotný dodavatelský softwarový systém ERP (Enterprise Resource Planning). Jeho ambicemi bylo pokrýt celý systém kompletně integrovaným řešením a případně být alespoň hlavním systémem mezi ostatními aplikacemi. ERP obvykle neobsáhne veškeré potřeby podniku, a jelikož je technologicky zastaralý tak se tento systém nedá udržet do budoucna jako jediný v podniku. Některé balíky ERP také pracují s proprietární datovou strukturou a není je tedy možné použít pro integraci jejich databázi. S tím, jak začalo býti více zřejmé, že integrace aplikací je závažný a nákladný problém tak začala být tato disciplína podrobněji zkoumána a začaly se objevovat přístupy a technologie zaměřené speciálně na integraci aplikací. Většina přístupů se týká metodických a technologických přístupů k integraci a tyto pohledy jsou jen dílčí částí problému. Shrnujícím pojmem pro aplikační integraci je EAI (Enterprise Application Integration). Je to souhrn metodik a technologií, které umožňují zjednodušit prostředí systému se špagetovou integrací do lépe spravovatelné architektury.

Technologie bývá založena obvykle na middleware neboli softwaru, jenž je použit jako mezivrstva pro nahrazení přímých vazeb mezi aplikacemi. Aplikace komunikují s middlewarem a při nahrazení jedné aplikace není nutné upravovat rozhraní všech napojených aplikací, ale pouze jedno rozhraní vedoucí k middlewaru. Dále zahrnuje nástroje pro zajištění průběhu procesů nebo transakcí přes více aplikací, řešení konfliktů zpráv, zajištění konzistence dat a někdy i pro sjednocení přístupu a uživatelského rozhraní. Dodávkou se zaobírá řada firem například IBM, Tibco, WebMethods a jiné.

Jednou z využití middlewaru je architektura založené na sběrnici zvaná ESB (Enetprise Service Bus). Principem je sběrnice, jež jednotlivé aplikace používají pro vzájemné zasílání zpráv. Centrálně je pak řízeno předávání zpráv, řešení konfliktů, překlady zpráv a podobně.

Vhodnou architekturou pro integraci aplikací je SOA (Servisně orientovaná architektura). Je založená na principech slabé vazby a posílání zpráv mezi malými aplikačními celky zvanými služby. Tyto služby lze propojovat do celků rozsáhlejších neboli procesů a tyto celky řídit, aniž by musely být navrženy s ohledem na tento celek, například pomocí standardu BPEL (Business Process Execution Language), anebo specifikovat jejich aktivní spolupráci na dosažení daných cílů. Tyto principy jsou použity hlavně v integračních nástrojích pro aplikace založené na technologii Web Services. Vzhledem k tomu, že webové služby jsou z principu obvykle podstatně méně rozsáhlé než například aplikační softwarové balíky typu ERP, je jejich integrace značně náročnější na složitost, například na správu verzí, testování a podobně. Dokonce i drobné změny mohou mít vysoké dopady do uživatelské zkušenosti s funkcionalitou a do doby odezvy aplikace. Vhodné je tedy v podnikovém prostředí používat automatizované nástroje pro řízení a monitorování aplikací postavených na SOA.

ESB a SOA nejsou v rozporu, konkrétní implementace může stavět na obou architekturách, tedy obě se vhodně doplňují. Součástí integrace aplikací je zajištění jednotného přístupu uživatelů k datům a více aplikacím na základě oprávnění vycházejícího z rolí, jež v byznysu zastávají a to za pomoci přístupů nazývaných Identity Management. Datawarehouse je další z možností integrace aplikací, respektive jejich dat a je to integrace pomocí analytických manažerských systémů. BI (Business Intelligence) je systém získávající data z většiny ostatních aplikací, aby umožnily různé analytické pohledy na podniková data zejména pro manažery a jejich rozhodování. Současné technologie datawarehouse umožňují

získávat data z on-line transakčních systému a není tedy nezbytné nahlížet do více aplikací pro získávání souvislostí mezi daty.

Toto lze využit v pouze v případě, že není potřeba aplikace integrovat na transakční úrovni a kdy jsou data v organizaci vhodně řízena a je udržována jejich alespoň základní konzistence a integrita. Trendem poslední doby je práce s nestrukturovanými informace a poznávání logicky jejich souvislostí automatizovaně, zkoumáním jazyka prostřednictvím tzv. ontologií. Tato technologie je vhodná pro hledání souvislostí v existujícím obsahu nestrukturovaných textových podnikových dat, nikoliv pro transakční systémy. Principy pro integraci aplikací v rámci jednoho podniku lze ovšem využít i pro integraci aplikací mezi více podniky. V takovém případě je vhodné, aby jeden z účastníků byl autoritou, jež určuje standardy integrace. Novou obtíží v integraci, nebo poměrně novou je trend SaaS (Software as a Service).

Skutečnost, že jednotlivou aplikaci z mnoha těch, jenž je potřeba vzájemně integrovat provozuje externí poskytovatel někde v cloudu a má ji zcela v rukách včetně instalace nových verzí, včetně vnitřní struktury a včetně specifikace rozhraní, značně ztěžuje rozhodování o technologických možnostech její integrace do podnikového systému a vytváří pochybnosti o dlouhodobé trvanlivosti a nákladech na udržování integračních vazeb na takovou aplikaci.

12.2 Softwarová architektura

V případě této architektury definujeme jeden softwarový produkt, tedy jedna softwarová aplikace. Hlavními komponentami jsou programové moduly aplikace a jejich strukturu a vztahy definuje architektura. Softwarová architektura určuje:

- Jestli aplikace bude provozována jako víceuživatelská nebo jednouživatelská
- z kolika modulů (programových) se bude aplikace skládat
- jak budou moduly aplikace uspořádány a specializovány – jedno, dvou a třívrstvá architektura (lineární, hierarchická, vrstvená a síťová architektura)
- funkcionalitu, jež bude každý modul zajišťovat
- každé funkci vstupní a výstupní data, interní data modulu
- algoritmus, jenž předepisuje způsob transformace vstupních dat na výstupní data a způsob ošetření mimořádných stavů
- vazby na ostatní moduly aplikace a interface modulu
- vývojové prostředí modulu
- provozní prostředí modulu

Provozní a vývojové prostředí bývá stejné pro všechny moduly dané aplikace, ale není to kategorickou podmínkou. V softwarové architektuře se zabýváme v plném rozsahu pouze u aplikací, jenž byly pro podnik vyvinuty na klíč (IASW – Individuální Aplikační Software). V případě, že se jedná o aplikaci typu TASW (Typový Aplikační Software) tak je její architektura pro podnikové informatiky většinou v plném rozsahu nedostupná, jelikož výrobce TASW zákazníkům obvykle v plném rozsahu nesděluje. Pro integraci TASW do celého IS jsou podstatné aplikační programový interface (API) a provozní prostředí celé TASW. Pro výrobce TASW je softwarová architektura TASW velice podstatná, jelikož její kvalita značně ovlivňuje flexibilitu aplikace a ovlivňuje také náklady na tvorbu a údržbu aplikace. To, jestli je architektura dobře navržena lze hodnotit podle následujících kritérií:

- **Neduplicitní funkcionalita a znovu použitelnost modulů aplikace** – Jedná se o to, aby potřebná funkcionalita ve více modulech byla řešena v systému pouze jednou místo řešení na několika místech.
- **Správné dimenzování aplikace** – Zda aplikace zvládne s požadovanou dobrou odezvy zpracovávat požadavky plánovaného počtu uživatelů a zda to zvládne i při maximálně očekávatelném objemu dat.
- **Snadnost údržby a dalšího rozvoje aplikace** - Jako příklad lze uvést, zda na místech očekávatelných změn jsou změny realizovatelné pomocí změn hodnot parametrů.
- **Shoda vývojového a provozního prostředí dané aplikace** – Posuzujeme softwarovou architekturu s vývojovým a provozním prostředím ostatní aplikací IS.
- **Náklady tvorby a provozu aplikace**

Dále charakterizuji různé varianty řešení softwarové architektury včetně jejich výhod a nevýhod.

12.2.1 Jednouživatelská a víceuživatelská aplikace

Jednouživatelská aplikace je navržena pro situaci, kdy je aplikace provozována zvlášť pro každého uživatele. Taková aplikace má nižší náklady na tvorbu, ale má vyšší náklady na provoz (vyšší celkové nároky na počítačové zdroje a jsou případy, kdy jsou takovéto aplikace nepoužitelné). Architektura víceuživatelská je navržena tak, aby tutéž aplikaci mohly ve stejné době využívat stovky a tisíce uživatelů z jedné nebo dokonce z mnoha organizací. Příkladem může být e-shop, rezervace letenek, Google Apps či CRM od SalesForce. V případě víceuživatelské aplikace je často typické, že všichni uživatelé sdílejí tutéž datovou základnu. Tyto aplikace se široce využívají zejména pro realizace SaaS (Software jako služba).

12.2.2 Klient/server architektura

V této architektuře je aplikace rozdělena na dvě nebo více kooperujících programů. Klientem je takový program, jenž vyžaduje provedené určité služby, zatímco serverem je ten, jenž danou službu na požádání poskytne. Jeden program přitom jednou může vystupovat jako klient a jindy jako server. Klient i server mohou být umístěny na tomtéž počítači, nebo mohou být a pracovat na různých počítačích počítačové sítě. Mezi výhody architektury, jenž využívá více počítačů patří:

- Oproti centralizovanému zpracování nižší nároky na server.
- Snadnější růst kapacit, jenž jsou pro zpracování aplikace zapotřebí.
- Lze volit zvlášť vhodnou provozní platformu pro jednotlivé aplikace.
- Jednodušší zabezpečení ochrany dat pomocí specializovaného datového serveru.

Mezi nevýhody lze zařadit:

- Vyšší prvotní investice nutné pro zprovoznění aplikací
- Problémy s integrací různých platforem.
- Vyšší nároky na kvalifikaci řešitelů.

Klient/server architektura se aplikuje ve dvou a třívrstvé softwarové architektuře.

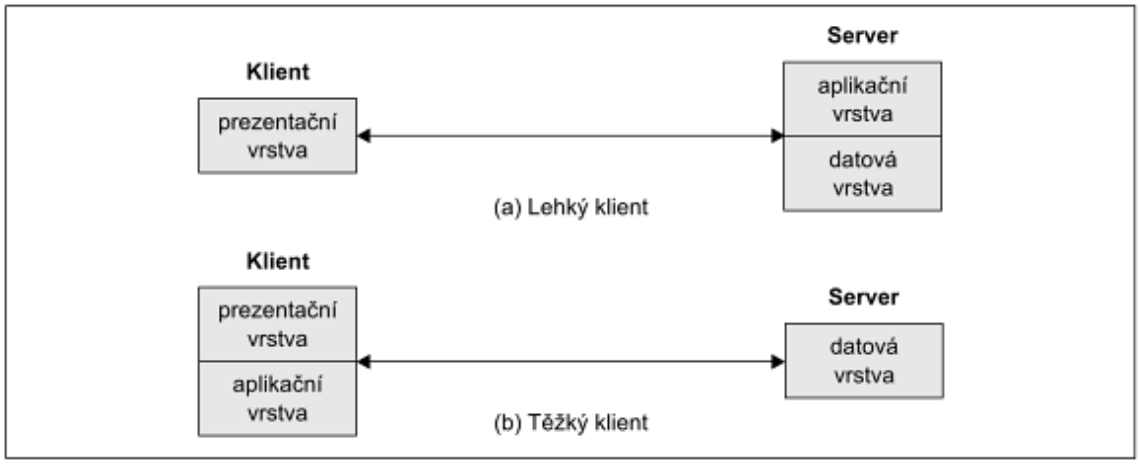
Monolitická, dvouvrstvá a třívrstvá architektura. Obvyklá aplikace obsahuje tři základní skupiny funkcí:

- Datové funkce (ukládání dat, ochrana dat, aktualizace dat a výběry dat)
- Věcně orientované funkce, jež zajišťují vlastní byznys logiku aplikace, neboli zajišťují požadovanou transformaci vstupních dat na data výstupní.
- Komunikační funkce (zobrazování výsledků na obrazovce koncové stanice, komunikace s uživatelem atd.).

Základním rozdílem mezi monolitickou, dvouvrstvou a třívrstvou architekturou je v tom, zda jsou tyto skupiny funkcí odděleny do samostatných programů, či nikoliv. V případě, že jsou všechny skupiny funkcí realizovány jedním programem tak se jedná o jednovrstvou monolitickou architekturu. V případě, že je jedna ze skupin funkcí oddělena od dvou zbývajících tak se jedná o dvouvrstvou architekturu a jsou-li všechny tři skupiny odděleny do samostatných programů, jde o třívrstvou architekturu. Jednotlivé vrstvy mezi sebou komunikují jako klient se serverem, to znamená, že dvouvrstvá a třívrstvá architektura aplikuje současně klient/server architekturu.

Pro centralizované zpracování je typická monolitická architektura. V monolitické architektuře jsou tři skupiny funkcí v programu vzájemně propleteny a samotný program běží na jednom počítači. Výhodou tohoto řešení je snadné zajišťování ochrany funkcí a dat aplikace před neautorizovaným použitím a snadné zajišťování aplikace proti výpadkům. Přináší sebou i řadu problémů:

- Obtížná údržba a obtížná přenositelnost aplikace mezi různými platformami.



Obrázek 12 varianty dvouvrstvé architektury[15]

U dvouvrstvé architektury jsou možné dvě varianty řešení aplikace a to lehký nebo těžký klient – viz obrázek 12. Obě varianty sledují hlavní cíl a to oddělit komunikační a datové funkce aplikace. Oddělením těchto dvou skupin funkcí se získává řada výhod:

- V aplikaci se snadněji zajišťují různé formy komunikace s různými koncovými stanicemi a s různými uživateli. Dochází k tomu tak, že pro každou formu komunikace se vytvoří samostatný program realizující prezentační vrstvu pro daný účel. Díky tomu roste uživatelský komfort aplikace a usnadňuje se údržba aplikace.
- Zvyšuje se přenositelnost aplikace
- U dvouvrstvé architektury však přetrvává problém přílišného propojení věcně orientovaných funkcí s jednou z dalších skupin funkcí.

Třívrstvá architektura odstraňuje poslední nevýhodu. Každou ze skupin funkcí lze udržovat a rozvíjet zcela samostatně a pro každou z nich je také možné zvolit nejvýhodnější vývojové a provozní prostředí. Tato architektura je proto ideální architekturou pro tvorbu

[15] Buckner T, Voříšek J., Buchalcevová A., Stanovská I., Chlapek D., Řepa V.: Tvorba informačních systémů: Principy, metodiky, architektury, 2012

otevřených, flexibilních a distribuovaných informačních systémů, jenž lze pružně přizpůsobovat:

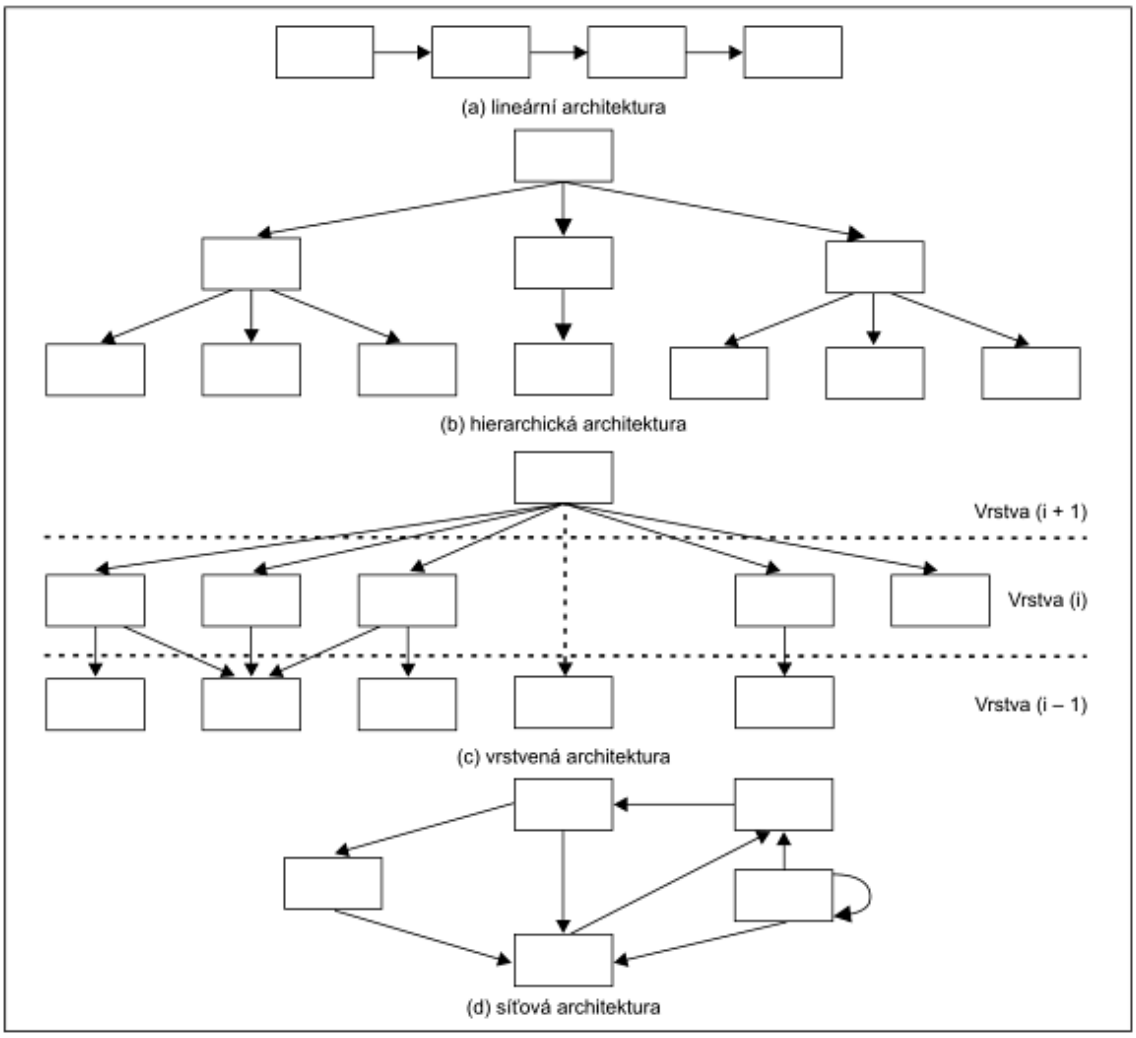
- Změnám na trhu s hardwarem a základním softwarem.
- Změnám v uživatelských požadavcích na funkce a uživatelské rozhraní aplikací.
- Změnám v počtu uživatelů a změnám v intenzitě práce uživatelů.

12.2.3 Lineární, hierarchická, vrstvená a síťová architektura

Aplikace typu ERP, CRM jsou tvořeny desítkami až stovkami programových modulů. Znamená to tedy, že například byznys logika aplikace nebo prezentační funkce nejsou tvořeny pouze jedním programovým modulem, ale mnoha moduly. Cílem dekompozice aplikace na malé programové celky je snadnější vývoj a údržba aplikace. Dekomponujeme-li aplikace na mnoho modulů tak musí být navrženo, jaké budou platit principy pro jejich vzájemnou komunikaci.

12.2.4 Lineární architektura

V této architektuře je požadovaná funkcionalita systému dosažena sekvenčním uspořádáním elementárních modulů. Příkladem lineární architektury je uspořádání elementárních komponent klasického textového systému, jenž obsahuje tři základní funkční komponenty: editor, reeditor a formátor. Výhodou systému s lineární architekturou je, že nevyžaduje komplikovanou organizaci pracovních týmů a že se snadno testuje. Nevýhodou je fakt, že nepodporuje strukturovaný přístup k řešení problému a že změna v jedné funkci/modulu může vyvolat řetěz úprav navazujících modulů či funkcí. V případě použití na vhodném problém pak má tato aplikace oproti následujícím architekturám nejnižší náklady vývoje. Z čistě lineární architekturou se dnes moc nesetkáme, jelikož podstat problému obvykle vyžaduje strukturovaný přístup k jeho řešení, a i když má systém na nejvyšší úrovni abstrakce architekturu lineární, tak obvykle na nižších úrovních má funkce či moduly uspořádány podle některé z následujících architektur.



Obrázek 13 Lineární, hierarchická, vrstvená a síťová architektura[16]

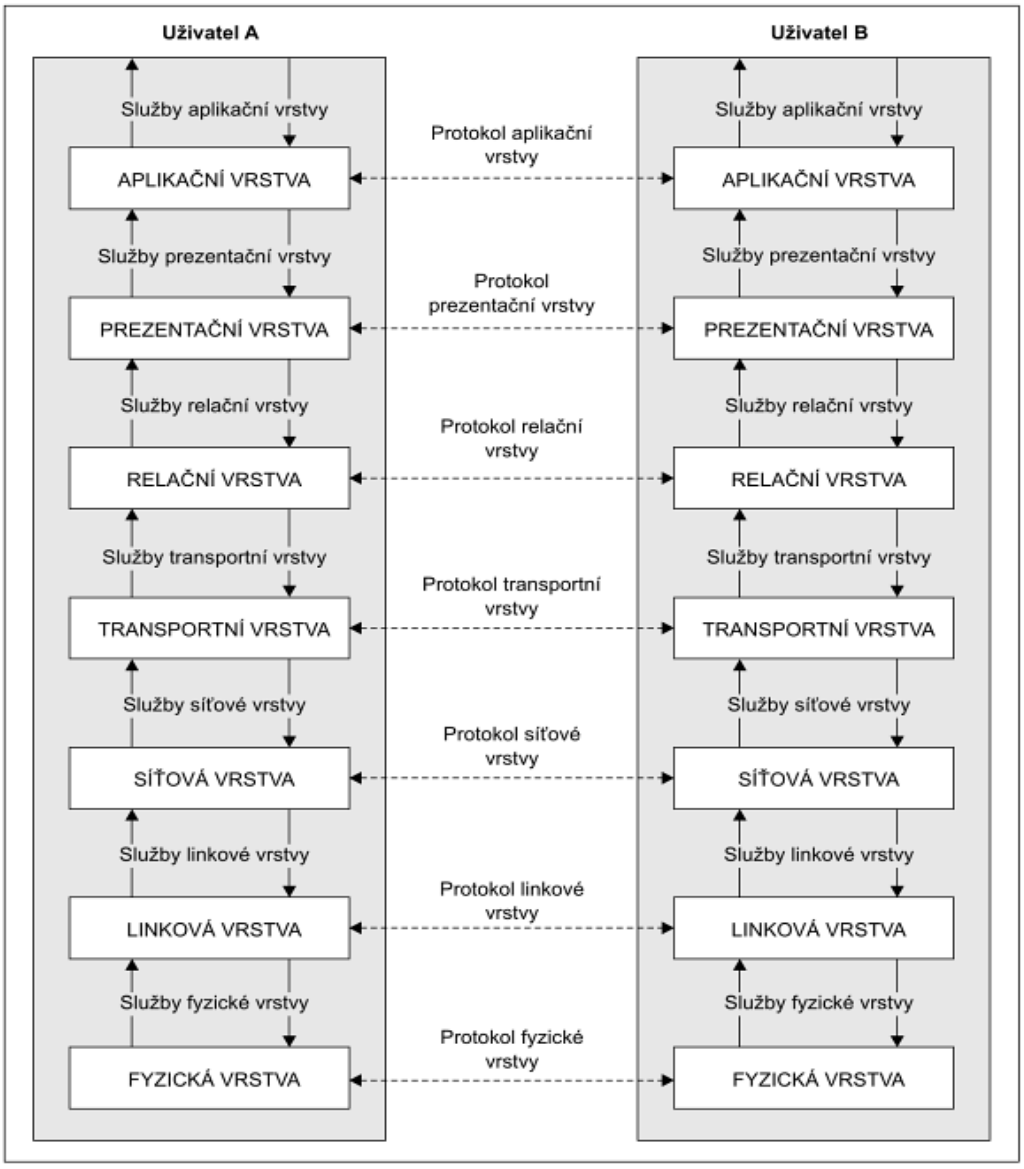
12.3 Hierarchická architektura

V této architektuře jsou jednotlivé funkce/moduly systému upořádány tak, že jejich vazby jsou reprezentovány stromovým grafem. Znamená to, že každá elementární funkce je vždy využita v právě jedné funkci vyšší úrovně. Tato architektura se dá s úspěchem použít pouze tehdy, jsou-li na systém kladeny takové požadavky, které splnit množinou vzájemně disjunktních funkcí. Funkce musí být na nižší úrovni abstrakce opět dělitelné na vzájemně disjunktní množiny elementárních funkcí. Tento předpoklad bývá málokdy v praxi splněn, takže striktní dodržení hierarchické architektury obvykle vede k duplicitním pracím. Dále k nárůstu nákladů na tvorbu a údržby systému. Výhodou hierarchické architektury je přehledná struktura systému, jenž povede k poměrně snadnému testování a snadné údržbě systému.

[16] Buckner T, Voříšek J., Buchalcevová A., Stanovská I., Chlapek D., Řepa V.: Tvorba informačních systémů: Principy, metodiky, architektury, 2012

12.3.1 Vrstvená architektura

Tato architektura překonává hlavní nevýhodu hierarchické architektury. Grafickou reprezentací je acyklický graf. Funkce softwarového systému jsou uspořádány do několika vrstev s tím, že funkce vyšší vrstvy mohou využívat jen funkce vrstev podřízených. Příkladem vrstvené architektury jsou vrstvy komunikačního softwaru podle standardu OSI – viz obrázek 14.



Obrázek 14 vrstvy komunikačního softwaru podle OSI[17]

12.3.2 Síťová architektura

Tato architektura je posledním typem používaným v praxi. Je reprezentována obecným orientovaným grafem a to znamená, že zde neplatí závazná pravidla podřízenosti a nadřízenosti jednotlivých modulů. Síťová architektura je typická pro řadu současných

[17] Buckner T, Voříšek J., Buchalcevová A., Stanovská I., Chlapek D., Řepa V.: Tvorba informačních systémů: Principy, metodiky, architektury, 2012

rozsáhlých softwarových systémů. De facto je to jediná použitelná architektura pro systémy budované „za pochodu“, jelikož hlavní předností je otevřenost pro přidávání nových funkcí. Předchozí architektury tak pružné nejsou, jelikož neočekáváný požadavek na přidání určité funkce může být v rozporu s dosavadní strukturou funkcí. Další výhodou této architektury je fakt, že v porovnání s hierarchickou a vrstvenou architekturou má obvykle nižší náklady provozu.

Na druhé straně má z uvedených architektur obvykle nejvyšší náklady užití, jelikož její zvládnutí uživatelem je nejnáročnější. Náklady na tvorbu a údržbu mohou být nízké jen za předpokladu dostatečně vysoké zkušenosti tvůrců a jejich dokonalé organizace. Síťová architektura vede k vysoké vzájemné závislosti jednotlivých funkcí. V případě, že se tyto závislosti vymknou přísné kontrole tak to má za důsledek velmi těžko zjistitelné šíření chyb po celém systému a rapidní narůstání nákladů údržby. U rozsáhlých systémů je tato pravděpodobnost značně vysoká. V případě opravení jedné chyby se často ukazuje, že to vede ke vzniku předem těžko odhadnutelného počtu chyb jiných. Síťová architektura je vhodná tehdy, když je zapotřebí preferovat nízké náklady provozu před nízkými náklady tvorby a údržby a před nízkými náklady užití.

12.3.3 Architektonické a návrhové vzory

V posledních letech se začíná prosazovat vývoj založená na architektonických a návrhových vzorech, jenž umožňuje navrhovat software rychleji, v lepší kvalita a znovu využívat již exitujících zdrojů. Vývojář může pracovat na vyšší úrovni abstrakce a aplikuje vzory, jež popisují obecná řešení opakujících se problému při návrhu. Díky tomu se může soustředit na řešení věcných problémů, místo toho, aby řešil problémy architektonické a technologické. Vzory zapouzdřují specifickou znalost architektury a technologie a pomáhají vytvářet znovupoužitelný kód. Jako příklad lze uvést Broker.

Broker je vzor pro distribuované systémy sestávající z nezávislých komponent. Tato komponenta zajišťuje jejich komunikaci prostřednictvím zasílání požadavků, výsledků a vyjímek.

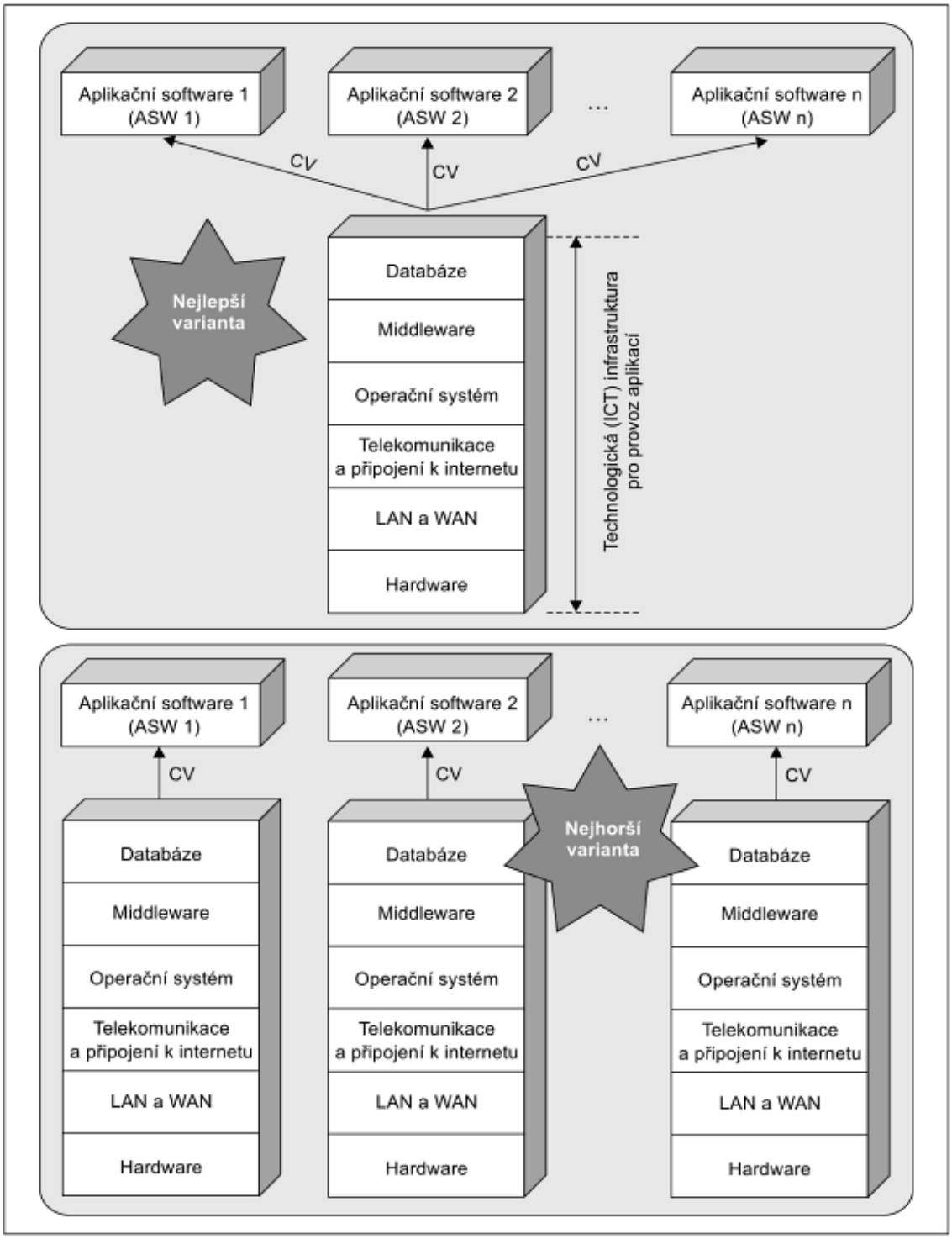
13 Datová/Informační architektura

V případě této architektury je systém, na němž architekturu definujeme informační a datová základna IS organizace. Architektura definuje vztahy a strukturu mezi hlavními komponentami, jež jsou datové objekty. Datová architektura vychází z analýzy potřebných

typů datových objektů a jejich vazeb. Podle této analýzy se provede konceptuální návrh a poté logický návrh datové základny. Navrhují se tedy nové datové entity, jejich vazby a atributy. Datová architektura je návrhem databázových souborů a jejich fyzického uložení, neboli jedná se o finalizovaný fyzický návrh datové základny.

14 Architektura technologické infrastruktury

V tomto případě je systém, na němž architekturu definujeme, provozní platforma aplikací IS organizace. Hlavními komponentami jsou hardwarové komponenty a komponenty základního programového vybavení. Snahou této architektury tedy je, aby technologická infrastruktura mohla být pro všechny aplikace jednotná – viz obrázek 15. V případě jednotné technologické infrastruktury dochází k výraznému snížení provozních nákladu informačního systému.



Obrázek 15 varianty řešení technologické infrastruktury pro provoz aplikací[18]

Dimenzování technologické infrastruktury neboli počet a výkonnost serverů, kapacita diskových polí apod. se odvozuje od požadovaného objemu (počet uživatelů, dat….) a kvality (dostupnost, spolehlivost…), jež souvisí se službami ICT. Jestliže mají být zajištěny

[18] Buckner T, Voříšek J., Buchalcevová A., Stanovská I., Chlapek D., Řepa V.: Tvorba informačních systémů: Principy, metodiky, architektury, 2012

dohodnuté parametry aplikačních služeb, především dostupnost a doba odezvy aplikací, musí být technologická infrastruktura dimenzována na období s nejvyšším vytížením. V případě, kdy dojde k poklesu uživatelských požadavků, dochází k neoptimálnímu využití technologické infrastruktury. To je jedna z podstatných nevýhod, když podnik provozuje svoje aplikace na vlastní technologické infrastruktuře. Je to jeden z důvodů proč celosvětově sílí trend cloud computingu (PaaS a IaaS).

Technologická architektura vychází z informační strategie, jenž určila standardy pro jednotlivé komponenty technologické architektury a dále z požadavků jednotlivých ICT služeb na kvalitu.

15 Praktická část

Pro praktickou část jsem se rozhodl uvést podstatu a účel IT architektury v pojišťovací společnosti a dále uvedu ukázku dílčích architektur, jelikož řešení architektury a použití různých metod a způsobu jak ji tvořit je několik a je na každém, co využije a co nikoliv. Vzhledem k omezenému rozsahu práce uvedu pouze jeden příklad skladby IT architektury v pojišťovací společnosti s tím, že rozepíši u dané vrstvy, co se v té konkrétní vrstvě vyskytuje. Zaměřím se pouze na popis nejzákladnějších vrstev architektury, jelikož v každé organizaci se to může lišit.

15.1 Podstata a účel

Za pomocí IT architektury dochází k efektivnímu využití prostředků ICT a ke zvýšení celkové výkonnosti organizace. V pojišťovně je vhodné, aby bylo v architektuře řešeno vše potřebné především ohledně hardwaru, databází, webových službách a aplikacích. V dobře fungující pojišťovací společnosti potřebujeme, aby vše fungovalo jak má a například při nějaké akci, kdy vzroste vytížení dané webové služby, tak aby nedocházelo ke zkolabování systému.

V případě pojišťovací společnosti si troufám říci, že je nezbytné, aby tato organizace měla na míru navrženou IT architekturu. V mnoha případech dochází k podceňování významu a přínosnosti architektury, ale jak čas ukazuje, tak je to nezbytné pro správné fungování organizace.

Jak jsem již zmiňoval v teoretické části této práce, tak v případě pojišťovny, jenž pojišťuje například automobily, nemovitosti a podobně je velice důležité, aby aplikace, byly na dostatečné hardwaru. Je nezbytné, aby architekt při tvorbě architektury měl po ruce dostatek informací o dané organizaci a aby tyto informace byly, pokud možno, maximálně komlexní..

15.2 Kritéria při výběru architektury

Mezi hlavní kritéria, která jsou při výběru vhodné architektury použity jsou následovné:

- spolehlivost,
- možnosti dalšího rozšíření,
- cena,

- technologické požadavky.

Každá organizace, či pojišťovna má svá kritéria podle, kterých rozhoduje a hodně záleží na druhu a oblasti podnikání. Kritéria výše vypsaná jsem napsal na základě prostudování literatury a subjektivního názor při vzniku nové pobočky či rovnou nové pojišťovací společnosti.

15.3 Aplikační architektura

V této architektuře se definuje architektura na informační systém. Podle mého názoru proto v této architektuře nalezneme jednotlivé softwarové aplikace a jejich vazby a strukturu. Jedná se tedy o architekturu, kde máme aplikace pro výpočet pojištění, účetnictví, různé kalkulační a podpůrné aplikace pro chod pojišťovací společnosti. Jako příklad aplikační architektury může sloužit informační systémy ERP.

„Informační systém kategorie ERP definujeme jako účinný nástroj, který je schopen pokrýt plánování a řízení hlavních interních podnikových procesů (zdrojů a jejich transformací na výstupy), a to na všech úrovních, od operativní až po strategickou.“[19]

V praxi to znamená, že firma od firmy se bude lišit, jednotlivé aplikace se budou e v této architektuře lišit, jelikož i každá pojišťovací společnost se liší v tom, jaké služby poskytuje, případně jak moc široký záběr má. Aplikace nám tedy realizuje jednu nebo více aktivit podnikového procesu. Často jsou aplikační služby dodávány v balíčku, například balíček aplikací pro komunikaci se zákazníky a podobně.

V ideálním případě, by zde měly být jen takové aplikace, aby bylo pokryto vše potřebné a tedy nedocházelo k duplicitě funkcionality. Například systém CRM by neměl obsahovat funkcionalitu jiné aplikace. Jak jsem již psal výše, jedná se o veškeré aplikace implementované do informačního systému. Aplikace pro výpočet pojistného, aplikace s nabídkou produktů, jejž nám zároveň kontrolují, zda daný klient má nárok na konkrétní produkt a podobně. Vzorce na výpočet produktu, tedy pojištění má každá společnost jiné a tudíž se následně podle toho může lišit architektura pojišťovacích společností.

Jsou zde dvě možnosti poskytování aplikací a to buď nákupem a následným používáním, nebo využitím externího poskytovatele, jenž nám poskytne vzdálený přístup. Podle mého názoru není moc vhodné využívat externího dodavatele a tedy mít vzdálený

[19] Sodomka P., Klčová H.: Informační systémy v podnikové praxi, 2010

přístup k aplikacím, jelikož v rámci pojišťovny je zde trocha know-how a informace k nimž by měli mít přístup pouze pracovníci dané společnosti.

15.4 Softwarová architektura

„V případě softwarové architektury je systémem, na kterém architekturu definujeme, jeden softwarový produkt, tedy jedna softwarová aplikace.“ [20] Z toho vyplývá, že hlavními komponentami jsou moduly aplikace. Tato architektura je vlastně „podsekce“ aplikační architektury, kde pracujeme se softwarovými produkty, jež vznikají v této architektuře. Zde tedy dochází přímo ke skládání modulů do jednotné aplikace podle přesných požadavků. V této architektuře dochází k zakomponování know-how a konkrétních vzorců a dalších osobitých částí dané organizace. Vzhledem k podstatě účelu podnikání zde může být spousta modulů, jež tvoří potřebné aplikace a vždy to záleží na dané organizaci jaké má potřeby.

15.5 Datová / Informační architektura

V této architektuře dochází k analýze a následnému návrhu potřebné datové základny. Zde tedy dochází k návrhu a poté k realizaci datových základen, jež jsou vhodné a potřebné pro pojišťovací společnost. V praxi pak tedy v této architektuře budou řešeny datové objekty a jejich vazby. Například zde bude databáze klientu a řešeno, které všechny informace o nic potřebujeme vědět.

15.6 Technologická architektura

V této architektuře dochází k řešení hardwarových komponent a komponent základního programového vybavení. Snahou je, aby infrastruktura mohla být pro všechny aplikace jednotná.

Podle mého názor a na základě poznatků získaných po prostudování literatury se tedy jedná se o to, aby například počítače na jedné pobočce pojišťovací společnosti byly jednotné a tudíž nebylo zapotřebí řešit problémy s nefunkčními aplikacemi vzhledem k rozličnému hardwarovému vybavení.

Po jednotném hardwaru se zde řeší provedení zasíťovaní pobočky pojišťovací společnosti a následné propojení se sítí internetu a intranetu v rámci celé organizace. V rámci

[20] Buckner T,Voříšek J., Buchalcevová A., Stanovská I., Chlapek D., Řepa V.: Tvorba informačních systémů: Principy, metodiky, architektury, 2012

zasíťování dané budovy se také provádí a počítá se zavedením telekomunikace a případně provedení intercomu. Řeší se zde také rozmístění zásuvek s přípojkou do sítě LAN či WAN, elektrické sítě. Dále se zde řeší rozmístění switchů a serveru v dané pobočce tak, aby došlo k optimálnímu pokrytí vše potřebných prostor. V případě serverů hodně záleží na tom, jak je to v rámci celé firmy řešené. V případě, že na lokálním serveru jsou pouze základní prostředky a zbytek je na serverech vzdálených (umístěných v serverovnách), tak nám zde odpadá spousty problémů a nároků jak na prostor, tak správu a finanční náročnost.

Podle mého názor v rámci jedné pobočky je nutné mít server s aplikacemi a dále se zálohovanými daty pro konkrétní pobočku. Webové služby by měly běžet na serveru mimo danou pobočku umístěném v příslušně vybavené a zabezpečené serverovně. Co se týče technických parametrů jednotlivých PC, je zde potřeba rozlišovat pro koho jsou určeny. Pracovníci, kteří nepotřebují vysoký výkon mohou využívat jednotných All-in-One PC, naproti tomu zaměstnanci, kteří k práci potřebují výkonější PC (pro svou práci potřebují více výkonu), by měli mít jiné stroje ovšem se stejnými komponentami, aby nebylo zapotřebí odlaďovat vše zvlášť na každém PC.

V rámci celé firmy by měl být jednotný operační systém. Jde především o podporu potřebných aplikací, odladění daného hardwaru (dostatečný výpočetní výkon a plná podpora daných komponent) a také o to, aby bylo možné všechny potřebné počítače připojit do dané sítě (domény).

Databáze by měli být řešeny tak, aby pro všechny aplikace byly jednotné a nebylo potřeba duplikovat data. Například údaje o klientech by měly být v jedné databázi, jenž by měla být přístupná všem pobočkám v rámci dané firmy, tudíž je potřeba takového důležité data mít na serveru, který zvládne vysokou zátěž. V rámci pobočky a případně jejich lokální databáze by měly být uloženy pouze takové informace, které nejsou potřebné pro vytváření a uzavírání smluv, ale jsou to doplňkové informace. Podle mého názoru by zrovna v případě pojišťovací společnosti měly být data uloženy pouze na serveru, ke kterému mají přístup všichni v rámci organizace, jelikož vzhledem k využití dat pro práci není přímo nezbytné mít lokální databázi. Ovšem lokální server by měl mít uložena data ohledně aplikací a případně přihlašovací informace do systému. Vše záleží na tom, jak se rozhodne daná organizace problém řešit.

15.7 Shrnutí

V praktické části jsem chtěl poukázat na to, co se v dané konkrétní architektuře schovává, jelikož pro nezasvěcené lidi to vždy nemusí být předem jasné. V případě, že by se postupovalo od úplného základu, tedy od koupě budovy přes budování infrastruktury až po následný provoz tak je nezbytné postupovat od zdola. Jednalo by se tedy nejdříve o technologickou architekturu, kde by došlo právě k zavedení potřebné kabeláže, instalace zásuvek a serverovny včetně veškeré potřebné hardwarové základny. Dále by se pokračovalo přes datovou architekturu, kde by došlo k vytvoření všech potřebných datových objektů a vazeb, tedy přípravou pro aplikace.

Teď máme připravený základ pro fungující společnost, ale stále nám schází to nejdůležitější. Schází nám aplikace, které by obsluhovaly připravená data. Tudíž je dále nezbytné pokročit na softwarovou architekturu. V softwarové architektuře tedy připravíme potřebné moduly a jednotlivé softwarové aplikace pro chod společnosti. Máme hotovy veškerý software a teď už jen zbývá jej implantovat do informačního systému. V aplikační architektuře dochází k implementaci softwarových aplikací do informačního systému a jejich vzájemných vazeb. Pro představu tedy dochází k tomu, že zaměstnanci dostanou své potřebné aplikace k práci a také k zavedení webových služeb, tedy možnosti klientů, nebo potencionálních klientů si sjednat pojištění on-line.

V konečné fázi zbývá jen sehnat dostatečně kvalifikované zaměstnance, kteří se budou aktivně věnovat IT architektuře a upravovat, či rozvíjet ji podle aktuální potřeby.

Výše uvedený postup je velice zjednodušený, jelikož jak již bylo v teoretické části zmíněno lze použít různé rámce a jiné technologie, jenž nám usnadní či zpřehlední a celkově pomohou při tvorbě kvalitní IT architektury společnosti nehledě na její zaměření.

16 Závěr

Cílem mé bakalářské práce bylo seznámit s IT architekturou a přiblížení používaných technologií a rozdělení na dílčí architektury. Bylo vysvětleno, jaké technologie je možné použít a jaké architektury a rámce lze použít. Bohužel vzhledem k rozsahu tohoto tématu bylo nutno vše vysvětlit jen do určité hloubky. Problematika IT architektury je značně rozsáhlá a tak jsem se rozhodl zde vybrat a popsat dílčí architektury a různé technologie, které je možné využít.

Integrace IT architektury do společnosti je velice významná a přináší v rámci podniku úspory nákladů, vyšší bezpečnost a větší spolehlivost. Problematika IT architektury je řešena již řadu let a neustále se vyvíjí. V teoretické části jsem se nejdříve zaměřil na vysvětlení základních pojmů, například co je architekt, co tato pozice obnáší a jaké jsou požadavky. Dále jsem pokračoval přes objasnění, proč je vlastně vhodné mít architekturu ve společnosti, jaké to má přínosy a dále na kritéria (liší se podle konkrétní společnosti a druhu podnikání) a osvětlení pojmu IT architektura. První část teoretické části se zabývá objasněním základních informací a jak vybrat správně řešení pro danou společnost.

Po vysvětlení těchto pojmů a objasnění, jak vybrat správně řešení jsem se rozhodl vysvětlit základní technologie. Ať se již jedná o různé přístupy k architektuře, k různým rámcům nebo různým druhům architektury tak je vše vysvětleno pouze částečně, jelikož vysvětlení každého pojmu, či součásti dané architektury by dalo na velice rozsáhlé knihy a tudíž zde není prostor pro vysvětlení všeho.

Podle mého názoru by měla být tato problematika mnohem více probírána, ať už se jedná o střední či vysoké školy. V praxi dochází ve větších podnicích k rozdělení IT architektury na více úrovní a každé úrovni je přidělen daný architekt. Navštívení a základní prostudování chodu a rozdělení architektur ve společnosti, jež si nepřála býti jmenována mi ukázalo, jak moc je veliký rozdíl mezi články, knihami a celkově teorií o IT architektuře a praxi.

Domnívám se, že stanovené cíle v mé bakalářské práci byly splněny. Tato práce obsahuje vše, co bylo stanoveno v cíli. Příklady, které jsou zde uvedeny jsem se snažil zasadit do kontextu a do pojišťovací společnosti s tím, že jsem se snažil, aby byl daný příklad co nejsnadněji pochopitelný.

V mém případě mi tato práce dala mnoho v tom ohledu, že jsem si prohloubil své znalosti o IT architektuře. Tato oblast informatiky mi přijde velice zajímavá a myslím si, že v budoucnu by to mohla být oblast, které bych se chtěl naplno věnovat.

17 Seznam použitých zdrojů

1. BRUCKNER, Tomáš. *Tvorba informačních systémů: principy, metodiky, architektury*. 1. vyd. Praha: Grada, 2012, 357 s. ISBN 978-80-247-4153-6.
2. DOHNAL, Jan a Jan POUR. *Architektury informačních systénů v průmyslových a obchodních podnicích: principy, metodiky, architektury*. 1. vyd. Praha: Ekopress, 1997, 357 s. Management v informační společnosti. ISBN 80-861-1902-5.
3. SODOMKA, Petr a Hana KLČOVÁ. *Architektury informačních systénů v průmyslových a obchodních podnicích: principy, metodiky, architektury*. 2. aktualiz. a rozš. vyd. Brno: Computer Press, 2010, 501 s. Management v informační společnosti. ISBN 978-80-251-2878-7.
4. SODOMKA, Petr a Hana KLČOVÁ. *Informační systémy v podnikové praxi: principy, metodiky, architektury*. 2. aktualiz. a rozš. vyd. Brno: Computer Press, 2010, 501 s. Management v informační společnosti. ISBN 978-80-251-2878-7.