ZS 2016/2017

ADAPTABLE OR PREDICTABLE?

4IT421 - ZLEPŠOVÁNÍ PROCESŮ BUDOVÁNÍ IS

MONIKA TOMÁNKOVÁ (XTOMM59)
ZDENĚK TURECKÝ (XTURZ03)
ADAM UHLÍK (XUHLA01)

DATUM ODEVZDÁNÍ: 18. 12. 2016

Abstrakt

Tato semestrální práce se zabývá problematikou adaptivního a predikovaného přístupu k vývoji softwaru a Cynefin frameworkem, který napomáhá ke kombinaci obou těchto způsobů. Práce se dá rozdělit na dvě hlavní části. V první části práce jsou rozebrány oba výše zmíněné přístupy a možnosti využití obou současně. V druhé části je pak podrobně popsán Cynefin Framework a jeho praktické využití. V závěru je shrnuto, co je klíčové si ze semestrální práce zapamatovat.

Klíčová slova

Adaptivní, predikovaný, vývoj, software, Cynefin, framework

Obsah

1. Úvod

Při vývoji softwaru si můžeme zvolit mezi dvěma, na první pohled zcela odlišnými, přístupy. Abychom zlepšili kvalitu vývoje, musíme si položit otázku, na jaký aspekt se chceme zaměřit. Měli bychom se spíše snažit o předvídatelnost a maximalizaci efektivity nákladů na úkor hodnoty a omezené přizpůsobivosti na vnitřní i vnější vlivy? Nebo naopak upřednostnit hodnotu a odezvu na nečekané situace před předvídatelností a efektivitou nákladů?

Situace ve skutečnosti není tak černobílá, jak by se na první pohled mohlo zdát. Dle odborného článku Adaptable or Predictable? Strive for Both – Be Predictably Adaptable! (Bakardzhiev, 2016) je z hlediska zdokonalení vývoje softwaru nejefektivnější být „předvídatelně přizpůsobiví“, protože jednoho bez druhého lze v reálném světě dosáhnout jen velice obtížně.

Dá se tedy jednoznačně říci, jaký z těchto přístupů je lepší? Nebo je nejefektivnější využít kombinace obou? A pokud bychom chtěli kombinovat oba přístupy, jak bychom toho nejlépe dosáhli?

Právě těmito otázkami se zabývá tato semestrální práce.

1.1. Cíl práce

Cílem této semestrální práce je uvést čtenáře do tématu adaptivního a predikovaného vývoje a vysvětlit mu rozdíly mezi těmito dvěma přístupy k vývoji softwaru a popsat možnosti jejich vzájemné kombinace.

Dalším cílem je pak výše zmíněné demonstrovat za využití Cynefin frameworku, který je v práci nejprve popsán, poté se vysvětluje jeho vztahu k oběma způsobům vývoje, objasňuje jeho užitečnosti v praxi a rozebírají se jednotlivé kroky při jeho používání.

1.2. Struktura práce

Semestrální práce je rozdělena na dvě části. V první části jsou nejprve popsány jednotlivé přístupy, jejich výhody a nevýhody a možnosti jejich kombinace. V části druhé, která je rozsáhlejší, je objasněn pojem „Cynefin framework“, jeho principy, praktické využití a výhody, které přináší.

2. Adaptable or Predictable?

Článek definuje pojmy adaptivní a prediktivní z hlediska Komplexních adaptivních systémů (*Complex Adaptive Systems* zkráceně CAS – pozn. autora). Tyto systémy se vyznačují velkým počtem komponent zvaných agenti. Tito agenti spolu nepředvídatelně interagují. Agenti se, na základě změněného prostředí v závislosti na proběhlých interakcí, učí nebo přizpůsobují nastalým změnám prostředí. Takovýmto systémům lze porozumět pouze na základě přímého seznámení s daným systémem. (Holland, 2006)

Jednotlivé komponenty a systém sám se vzájemně vyvíjejí a nelze tak předvídat jejich budoucnost. Lidé jsou však schopni komplexnost daného systému nahradit řízenou formou, tím se systém stává předvídatelný. Příkladem CAS může být vývojářský tým z pohledu klienta.

Hlavní charakteristiky CAS (Gell-Mann, 1994)

- získávání informací o prostředí a jednotlivých interakcích
- identifikování podobností ve sbíraných datech. Tyto podobnosti následně přepracovat do vzorů
    - vzory lze chápat jako metody pro interakce, podle kterých se jednotlivé komponenty v CAS řídí
- výsledky v reálném světě získané za pomoci užití vzorů představují zpětnou vazbu, kterou systém bere jako nové informace

2.1. Přizpůsobivost (Adaptablility)

Přizpůsobivost systému lze charakterizovat jako schopnost adekvátně reagovat na měnící se podmínky trhu, a zároveň požadavky zákazníků, aby co nejvíce odpovídal účelu, pro který byl vytvořen. Hlavní výhodou adaptivního přístupu je především výše popsaná rychlá reakce na měnící se podmínky a důraz na výslednou hodnotu systému.

Přizpůsobivost lze rozdělit na dvě úrovně. První úroveň označovaná jako přímá (anglicky *direct* – pozn. aut.) představuje případ, kdy CAS využívá současných vzorů. Druhá úroveň představuje situaci, kdy CAS může upravit, popřípadě nahradit, vzory, pokud neposkytují dostatečné výsledky. Tato úroveň je označována jako komplexní.

2.2. Předvídatelnost (Predictability)

Předvídatelnost systému lze charakterizovat jako schopnost správného předvídání budoucnosti daného systému na základě dat získaných v minulosti. Hlavními výhodami prediktivního přístupu je možnost dosažení efektivnějšího vynakládání zdrojů a právě předvídatelnost budoucího stavu systému.

2.3. Předvídatelně přizpůsobivý

Pokud má daný CAS být dlouhodobě udržitelný a produktivní, musí se přizpůsobovat danému prostředí. Právě udržitelnost CAS závisí především na úspěšném dosažení obou úrovní přizpůsobivosti, tedy využívat současných vzorů a měnit je či vytvářet v závislosti na potřebě. V ideálním případě by CAS měl dosáhnout předvídatelné přizpůsobivosti.

Pro dosažení předvídatelné přizpůsobivosti je nutné zjistit, kdy stávající vzory již neodpovídají současným podmínkám, respektive již nejsou tak produktivní, a je potřeba provést jejich úpravu či výměnu za nové. K tomuto úkolu lze využít Cynefin framework.

Pohledem Cynefin frameworku lze říci, že rozhodování mezi adaptivním a prediktivním přístupem je bezpředmětné, protože v reálném světě nelze dosáhnout jednoho přístupu bez druhého. Cílem pro vylepšování vývoje softwaru by tedy mělo být dosažení prediktivní adaptability.

3. Cynefin framework

Cynefin framework nám umožňuje popsat realitu[1] a dává nám techniky a praktiky, které se používají především při řešení komplikovaných a komplexních problémů a při rozhodovacích úkolech. Jeho implementace pomáhá především manažerům a tvůrcům standardů podniku. Dává jim jednotný rámec pro jejich rozhodování a poznávání (vyjasňování si) reality a na zjištěné skutečnosti poté vhodně zareagovat (přizpůsobit se jim) případně je již predikovat dopředu. (Brougham, 2015)

Cynefin framework byl vytvořen na počátku roku 2000 Davem Snowdenem and Cynthia Kurtzem v rámci IBM Cynefin Centre for Organizational Complexity. Tehdy byl ještě tento framework nazýván sense-making device[2]. V roce 2002 Snowden a Kurtz zakládají společnost Cynefin Centre, která se nadále věnuje rozvoji tohoto frameworku. V roce 2004 se tato společnost stává nezávislá na IBM. (Snowden, Boone, 2007)

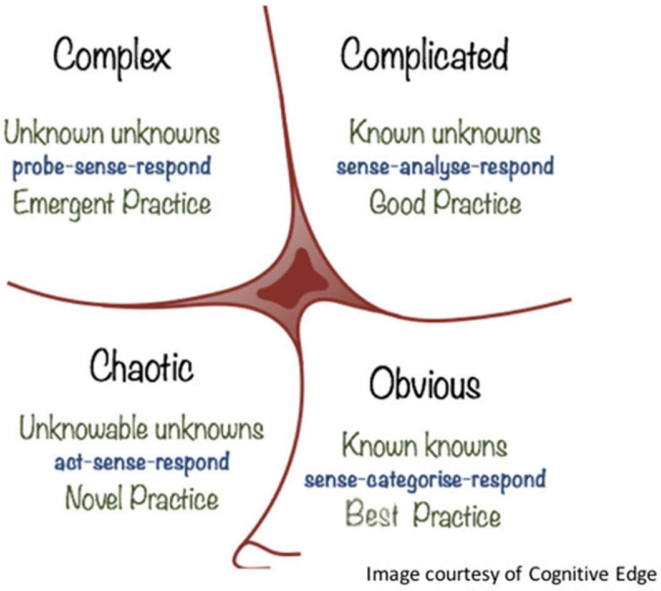
3.1. Princip Cynefin frameworku

Jak již bylo zmíněno výše, Cynefin framework nám pomáhá při rozhodování tím, že nejprve přiřadíme význam jednotlivým zkoumaným aspektům reality. Poté co známe významy aspektů, můžeme aspekty rozřadit do jednotlivých kvadrantů (domén) Cynefin Frameworku. Pro každou doménu jsou potom už definovány řešení, které radí, jak daný aspekt bude obtížné řešit (řádově kolik času zabere jeho řešení, zda vůbec je aspekt řešitelný, zda známe všechny potřebné okolnosti, abychom mohli aspekt vyřešit apod.).

[1] Přiřazením významů různým objektům světa, ve kterém se nacházíme.
[2] Volně přeloženo jako „Nástroj pro přiřazování významů“.

Cynefin framework má celkem pět domén (jak zobrazuje *Obrázek 1: Domény Cynefin frameworku* (zdroj: Bakardzhiev, 2016). Jedná se o domény problémů: *Jasné,*

, ***Error! Reference source not found.****, Chaotické* a *Centrální doména*.



Obrázek 1: Domény Cynefin frameworku (zdroj: Bakardzhiev, 2016)

Pro následující text k vysvětlení domén byla použita kniha The Cynefin Mini-Book (Brougham, 2015) a článek Adaptable or Predictable? Strive for Both – Be Predictably Adaptable! (Bakardzhiev, 2016).

3.1.1. Jasné

Jasné (anglicky Obvious) aspekty[3] lze také vysvětlit jako **známe známé**. Je to doména těch aspektů, se kterými se setkáváme nejčastěji, a tudíž už máme vydefinované jasné postupy a řešení, jak na ně reagovat. Navíc na rozdíl od ostatních domén, u této domény máme možnost vysoce spolehlivé predikce – umíme přesně odhadnout průběh i následky aspektu. Při řešení aspektu této domény se nejčastěji používá tento specializovaný postup:

1. Přiřadíme význam jednotlivým aspektům.
2. Roztřídíme aspekty do kategorií.
3. Aplikujeme řešení na základě best practices[4] definovaných pro danou kategorii.

[3] Dříve se tato kategorie nazývala Simple (jednoduché).
[4] Tj. nejlepší/nejosvědčenějších postupů.

3.1.2. Komplikované

Komplikované (neboli Complicated) aspekty lze charakterizovat jako **známe neznámé**. Jsou to ty typy aspektů, u kterých víme, že nastaly, ale zároveň nevíme, o co přesně jde (co je přesně problém), nebo nemáme přesné informace či zdroje, pro jeho vyřešení. Ale pomocí výsledků analýzy, lze použít nejlepší dostupné řešení.

Protože řešení tohoto druhu aspektu je založeno hodně na schopnostech analytika a jeho dedukcí, klade výběr vhodného řešení velké nároky na zkušenosti a vědomosti analytika. Je to způsobeno tím, že pokud se pouze odhaduje, jak daný problém s nedostatkem informací řešit, jeví se hned několik nejlepších způsobů řešení a je zapotřebí vybrat to nejefektivnější (optimální).

Při řešení aspektu této domény se nejčastěji používá tento specializovaný postup:

1. Přiřadíme význam jednotlivým aspektům.
2. Analyzujeme možnosti řešení pro dané aspekty.
3. Na základě výsledků analýzy se vybere nejvhodnější řešení.

3.1.3. Komplexní

Komplexní (anglicky Complex) aspekty, jsou takové, které můžeme definovat jako **neznáme neznámé**. Při určitých výsledcích (dopadech) neumíme určit, co je způsobilo. Jinými slovy, nevíme, že nastal nějaký problém a ani nevíme, jak ho řešit, známe jen jeho následky.

Protože se v této doméně neznáme neznámé, lze aspekty identifikovat a nalézt správná řešení pouze metodou pokus-omyl. Díky tomuto postupu získáme spoustu hypotéz, které poté můžeme ověřovat a aplikovat různá řešení. Je také nutné zdůraznit, že ze začátku bez dostatečných znalostí a informací, neexistuje žádná správná nebo špatná odpověď, je tedy nutné prověřit každou hypotézu zvlášť. Až postupem času se zjistí, zda naše hypotéza byla správná a kategorizace aspektu se může posunout do domény ***Error! Reference source not found.***, nebo dokonce problém přesně určíme a nalezneme k němu optimální řešení a překlasifikujeme ho tak na doménu *Jasné*.

Pro urychlení odhalení aspektu se doporučuje, aby běželo několik experimentů pro ověření hypotézy současně. Také je nutné, aby postupy i výsledky každého experimentu byly zaznamenány a mohli jsme tak zaměřit a zkusit aplikovat ještě neprověřené postupy.

3.1.4. Chaotické

Chaotické (anglicky Chaotic) aspekty, jsou aspekty, které můžeme nazvat také jako **neznáme nepoznané**. Neexistují zde žádné vztahy mezi příčinou a souvislostí. Pro tuto doménu aspektů je typické, že zareagujeme okamžitě a aplikujeme první řešení, které se naskýtá bez žádných časových prodlev.

Při řešení aspektu této domény se nejčastěji používá tento specializovaný postup:

1. Rychle a rozhodně zareagovat na aspekt.
2. Přiřadíme význam jednotlivým aspektům.
3. Na základě významu zkusíme na aspektu zareagovat lépe.

3.1.5. Centrální doména

Někdy také nazývána jako nezařazené (anglicky disrodered nebo také not determined) jsou aspekty, které ještě nebyly zařazeny do žádné z předchozích domén, a které tak musíme ještě rozřadit.

3.2. Použití Cynefin frameworku v praxi

Následující text čerpá informace z odborného článku Adaptable or Predictable? Strive for Both – Be Predictably Adaptable! (Bakardzhiev, 2016).

Cynefin framework se používá především při vývoji softwaru, ale díky jeho universálnímu principu ho lze využít pro rozhodování managementu i mimo oblast IT.

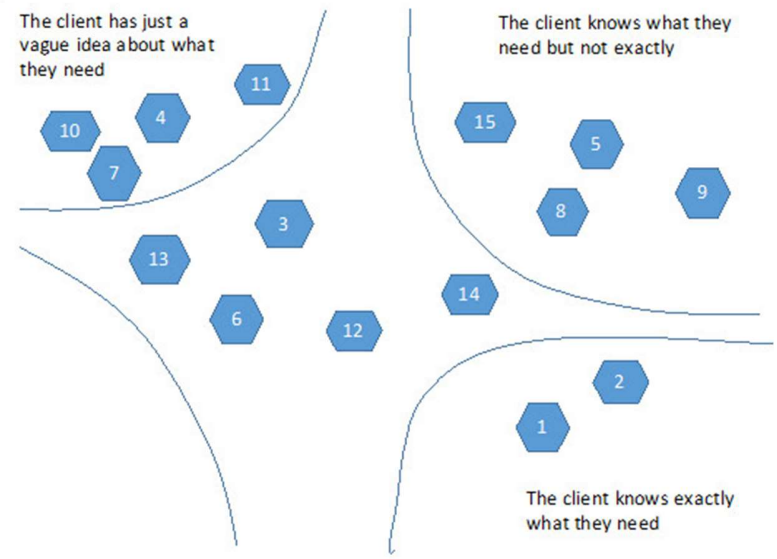
Při vývoji softwaru se framework používá jako komunikační nástroj, který pomáhá si ujasnit, co klient vlastně požaduje a potřebuje a na druhé straně, co dodavatel umí (co může nabídnout) a v čem má naopak vývojový tým mezery. Výsledkem *Kontextualizace* a *Kategorizace* je, vyjasnění si se zákazníkem, jak přesně bude vypadat výsledný produkt (jaké části/funkce se přesně budou realizovat) a z tohoto můžeme predikovat požadavky na znalosti a zkušenosti členů vývojového týmu, časový harmonogram apod. a pokud nedisponujeme nějakými zdroji, tak na to vhodně zareagovat (adaptovat se na požadavky).

3.2.1. Kontextualizace

Proces kontextualizace slouží k vyjasnění si uživatelových požadavků na vyvíjený software a zda náš tým umí vytvořit požadované řešení. Jinými slovy, co vlastně bude realizováno.

Nejprve se začíná s definicí uživatelských scénářů[5]. Tuto skutečnost ukazuje *Obrázek 2: Kontextualizace klientových požadavků* (zdroj: Bakardzhiev, 2016). Na kartičky se napíší scénáře (funkcionality), které zákazníci chtějí v rámci softwaru využívat[6]. Poté se jednotlivé scénáře rozdělují do domén (kvadrantů) vytvořeného Cynefin frameworku. Jak je vidět na obrázku, při kontextualizaci využíváme jen tří domén frameworku (*Chaotickéá* doména se nepoužívá). Jedná se konkrétně o domény:

- *Jasné* – zákazník přesně ví, jaké funkcionality požaduje a už si je detailně specifikoval.
- ***Error! Reference source not found.*** – zákazník ví, že něco potřebuje, ale neví, co to přesně je. Je zapotřebí expertní analýzy, aby se vyjasnila požadovaná funkcionalita.
- ***Error! Reference source not found.*** – zákazník má jen vágní tušení, co potřebuje. Musí se vyzkoušet řada alternativ, aby bylo možné učit, jaká funkcionalita je potřeba.



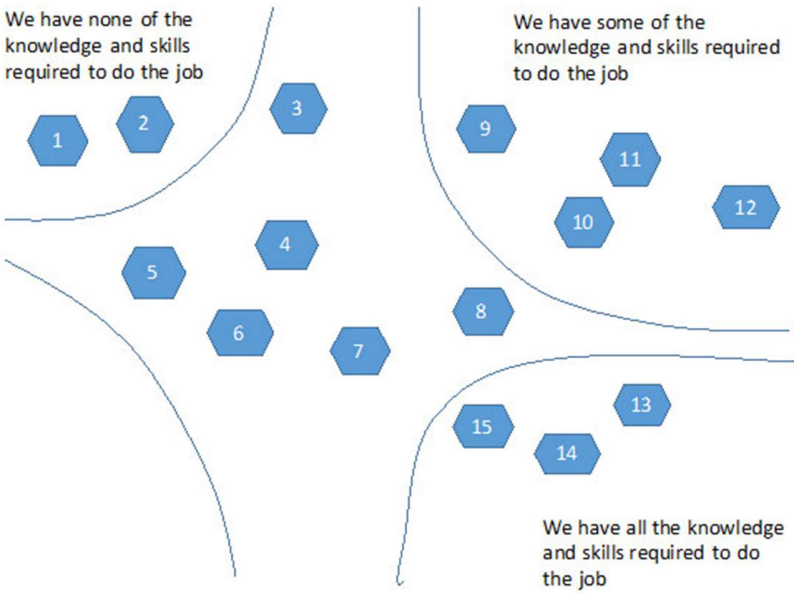
Obrázek 2: Kontextualizace klientových požadavků (zdroj: Bakardzhiev, 2016)

[5] Anglicky User stories.
[6] V případě obrázku jsou scénáře znázorněny modrými šestiúhelníky.

Dále je třeba si stejným principem definovat znalosti a zkušenosti vývojového týmu tak, aby bylo možné uspokojit zákazníkovy požadavky (viz *Obrázek 3: Kontextualizace schopností vývojového týmu* (zdroj: Bakardzhiev, 2016)).

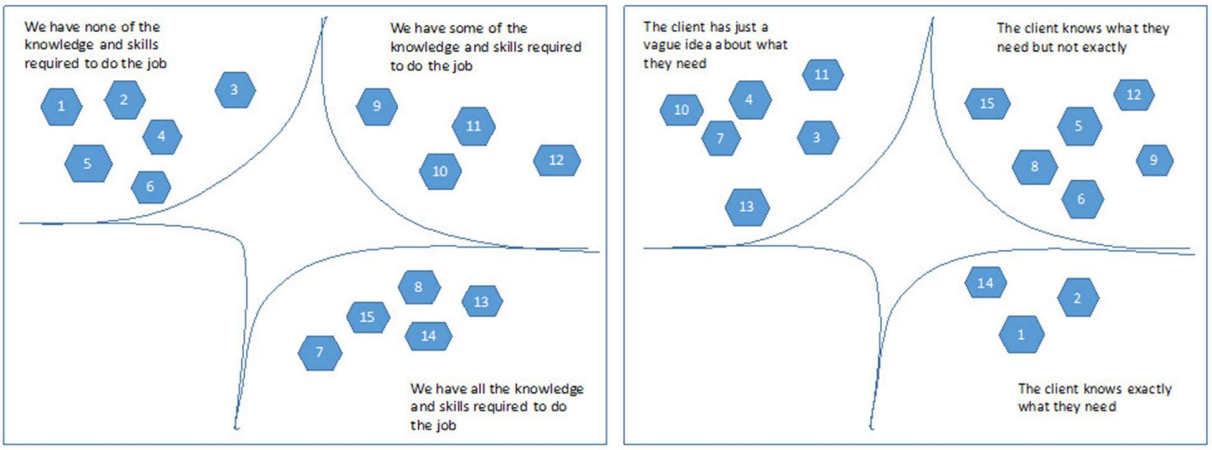
Opět používáme pouze tři rozřazovací domény:

- *Jasné* – vývojový tým má všechny potřebné znalosti a zkušenosti k vývoji softwaru.
- ***Error! Reference source not found.*** – vývojový tým disponuje pouze některými znalostmi a zkušenostmi, které jsou požadovány. Pro uspokojení ostatních požadavků musíme provést výzkum a analýzu, protože někdo před námi už tyto požadavky řešil a lze se tak inspirovat jejich řešeními.
- ***Error! Reference source not found.*** – pro uspokojení požadavků nemá vývojový tým žádné potřebné znalosti ani zkušenosti. Není známo, že by někdo tyto požadavky někdy řešil, a proto lze nalézt jen podobné požadavky (alternativy), na základě jejich řešení si vybudovat znalost a zkusit daný požadavek s těmito nově nabytými znalostmi řešit.



Obrázek 3: Kontextualizace schopností vývojového týmu (zdroj: Bakardzhiev, 2016)

Výsledkem těchto dvou procesů je to, že nakonec má každý uživatelský scénář přiřazenou svoji doménu – viz *Obrázek 4: Finální rozřazení uživatelských scénářů do domén* (zdroj: Bakardzhiev, 2016).



Obrázek 4: Finální rozřazení uživatelských scénářů do domén (zdroj: Bakardzhiev, 2016)

Pokud by se stalo, že některý scénář se pořád nachází v *Centrální doména* nebo naopak je na pomezí dvou domén, musí proběhnout diskuze, která poskytne rozřešení těchto konfliktů. Z principu Cynefin frameworku totiž vyplývá, že každý uživatelský scénář musí mít určenou právě jednu doménu.

3.2.2. Kategorizace

Pokud jsme si již rozřadili uživatelské požadavky a možnosti vývojového týmu, lze přejít k dalšímu kroku, kterým je kategorizace.

Kategorizace se provádí především kvůli tomu, abychom u jednotlivých uživatelských scénářů mohli určit nejistotu (nespolehlivost) při jeho dodání. Jinými slovy, jak moc je nepravděpodobné, že dokážeme požadovanou funkcionalitu zákazníkovi dodat.

Pro kategorizaci se využívá Kategorizační matice (viz *Obrázek 5: Kategorizační matice* (zdroj: Bakardzhiev, 2016)

		Client Context		
		Complex	Complicated	Obvious
Capability Context	Complex	High	High	High
	Complicated	High	Medium	Medium
	Obvious	High	Medium	Low

Obrázek 5: Kategorizační matice (zdroj: Bakardzhiev, 2016)

Jednotlivé scénáře se ohodnotí na základě přiřazení do domén v rámci uživatelových požadavků[7] a schopností vývojového týmu[8]. Hodnota Vysoká (High) znamená, že dodání je velice nepravděpodobné. Hodnota Střední (Medium) v sobě nese jistou neurčitost dodání. Hodnota Nízká (Low) v sobě nenese téměř žádné pochyby o dodání funkcionality.

Všechny vlastnosti uživatelského scénáře se poté zapisují přehledně do tabulky – viz *Obrázek 6: Tabulka vlastností uživatelských scénářů* (zdroj: Bakardzhiev, 2016)

User Story	Capability Context	Client Context	Uncertainty profile
#1	Complex	Obvious	High
...	...	...	
#5	Complex	Complicated	High
...			
#14	Obvious	Obvious	Low
#15	Obvious	Complicated	Medium

Obrázek 6: Tabulka vlastností uživatelských scénářů (zdroj: Bakardzhiev, 2016)

Pokud v budoucnu přibydou nové uživatelské scénáře, je třeba dodržovat stejné postupy a principy při kontextualizaci a kategorizaci, jaké byly použity předtím. Dále je třeba si uvědomit, že jakékoliv změny se v kontextualizaci či kategorizaci provedou, jsou sledovány. Tyto skutečnosti jsou ověřovány pomocí pravidelných feedbacků.

3.3. Výhody pomocí Cynefin frameworku

V této podkapitole je vysvětleno, jaké výhody má použití Cynefin frameworku při vývoji softwaru a jak lze na základě rozdělní do příslušných domén predikovat jednotlivé zdroje projektu.

Následující text čerpá informace z odborného článku Adaptable or Predictable? Strive for Both – Be Predictably Adaptable! (Bakardzhiev, 2016).

3.3.1. Lidské zdroje

Na základě různých hodnot nejistoty potřebujeme i různě zkušené členy týmu. Například pro nízkou hodnotu nejistoty se hodí juniorní členové týmu, kterým stačí dodržovat dohodnuté postupy a nástroje a při vývoji se nemění požadavky.

Naopak v případě, kdy máme často se měnící požadavky, potřebujeme seniorní členy, kteří se dokáží adaptovat na tyto měnící se požadavky i bez návodů.

Klíčem k co největší míře adaptability je mít co nejvíce diverzifikovaný tým – tj. tým, který bude mít různou úroveň zkušeností (tým obsahuje juniory, tak i seniory).

[7] Na *Obrázek 5: Kategorizační matice* (zdroj: Bakardzhiev, 2016) zaznamenáno jako Client Context.
[8] Na *Obrázek 5: Kategorizační matice* (zdroj: Bakardzhiev, 2016) zaznamenáno jako Capability Context.

3.3.2. Plánování

Hodnota nejistoty má negativní vliv na plánování projektů. Čím vyšší je totiž hodnota nejistoty, tím vyšší je risk, že se zpozdí projekt nebo daný uživatelský scénář nebude splněn.

Z tohoto důvodu se snažíme řešit nejprve scénáře s vysokou hodnotou nejistoty, abychom mohli naplánovat projekt co možná nejpřesněji (po vyřešení scénářů s vysokou nejistotou totiž dokážeme celkem přesně určit zbývající časovou náročnost pro scénáře se střední a nízkou nejistotou).

3.3.3. Provedení

Při aplikování Cynefin frameworku se snažíme posunout z domény ***Error! Reference source not found.****ch* aspektů do domény ***Error! Reference source not found.****ých.* To provádíme pomocí řady pokusů, kdy se snažíme poznat realitu a identifikovat různé vlivy, které působí na výsledný aspekt. To, zda je výsledek našeho pokusu správný, si ověříme tím, že námi vyzkoumané vlivy následně umíme ovlivňovat a predikovat.

Pokud zjistíme, že po zařazení některého ze scénářů do domény Komplikovaných aspektů, se odhadované vlastnosti zkoumaného jevu vyvíjí jinak, než jak jsme předpokládali, je možné scénář posunout zpět do domény Komplexních aspektů. Zde může zůstat napořád, protože u některých scénářů je možné, že nikdy nebudeme schopni predikovat některé průběhy a dopady daných aspektů.

Poté, co jsme spokojeni se zařazením všech uživatelských scénářů do příslušných domén, lze začít s vývojem softwaru, který bude odpovídat vydefinovaným uživatelským scénářům odsouhlaseným jak ze strany zákazníka, tak ze strany vývojového týmu.

3.4. Příklad využití

I přes vynaložené úsilí se nám nepodařilo dohledat zmapovaný případ využití Cynefin frameworku na reálném vývojovém projektu. Vysvětlujeme si to tak, že data použitá při aplikaci tohoto frameworku byla příliš citlivá než aby se objevila veřejně.

Existující zmínka o jeho využití se týká jiné oblasti, konkrétně Agentury amerického ministerstva obrany pro pokročilé výzkumné projekty (DARPA), která aplikovala framework na boj proti terorismu. Důvodem bylo, že se zaměstnanci chovali ke komplexním problémům jako by byly jasné, což mělo za následek několik selhání americké zahraniční politiky. Integrace frameworku spočívala ve varování lidí, aby se na problémy koukali více zblízka, raději než předpokládali, že strukturované procesy přinesou předvídané výstupy (Snowden, 2016).

4. Závěr

Ačkoliv se na první pohled mnohou zdát oba přístupy k vývoji softwaru naprosto odlišné, ve skutečnosti je z hlediska efektivnosti vývoje a dlouhodobé udržitelnosti CAS klíčová kombinace obou.

Pro ulehčení dosažení předvídatelné přizpůsobivosti lze využít Cynefin framework.

Cynefin framework poskytuje přístup a soubor praktik pro řešení nejistoty, které v dnešní době čelí management společností čím dál více.

Poskytuje jim prostředky pro to, aby si uvědomily, že stojí proti chaotickému, obtížně řešitelnému problému a zároveň i nástroje, které jim pomohou se s těmito problémy vypořádat.

Jedná se o znatelnou změnu oproti tradičním přístupům, které se snažili problém rozebrat na množinu racionálních akcí a připouští, že v některých případech není možné předvídat výsledky.

Místo posedlosti předpovídáním budoucnosti se můžeme soustředit na její kontrolování, díky čemuž pak nemusíme predikovat vše. A to je hodnota Cynefin frameworku.

5. Zdroje

BAKARDZHIEV, Dimitar. Adaptable or Predictable? Strive for Both – Be Predictably Adaptable!. In: InfoQ [online]. Waterloo: C4Media Inc., 2016 [cit. 2016-11-26]. Dostupné z: https://www.infoq.com/articles/predictably-adaptable?utm_source=infoqWeeklyNewsletter&utm_medium=WeeklyNL_EditorialContent_culture-methods&utm_campaign=09132016news

BROUGHAM, Greg. *The Cynefin Mini-Book*. First Edition. USA: InfoQ, 2015. ISBN 9781329508644.

SNOWDEN, David J. a Mary E. BOONE. A Leader's Framework for Decision Making. *Harvard Business Review* [online]. 2007, **85**(11), 68-76. ISSN 00178012. Dostupné z: http://ar9zk7pw9f.search.serialssolutions.com/?ctx_ver=Z39.88-2004&ctx_enc=info%3Aofi%2Fenc%3AUTF-8&rfr_id=info%3Asid%2Fsummon.serialssolutions.com&rft_val_fmt=info%3Aofi%2Ffmt%3Akev%3Amtx%3Ajournal&rft.genre=article&rft.atitle=A+Leader%27s+Framework+for+Decision+Making&rft.jtitle=Harvard+Business+Review&rft.au=David+F+Snowden&rft.au=Mary+E+Boone&rft.date=2007-11-01&rft.pub=Harvard+Business+Review&rft.issn=0017-8012&rft.volume=85&rft.issue=11&rft.spage=68&rft.externalDocID=1370899951&paramdict=cs-CZ

HOLLAND, John H. Studying Complex Adaptive Systems. Journal of Systems Science and Complexity. 2006, 19(1), 1-8. DOI: http://dx.doi.org/10.1007/s11424-006-0001-z. ISSN 1009-6124.

GELL-MANN, Murray. Complex adaptive systems. *Complexity: Metaphors, models and reality* [online]. 1. Los Alamos: Santa Fe Institute, 1994, s. 17-45 [cit. 2016-11-22]. Dostupné z: http://tuvalu.santafe.edu/~mgm/Site/Publications_files/MGM%20113.pdf

FOWLER, Martin. The New Methodology. martinfowler.com. [online]. 13.12.2005 [cit. 2016-12-01]. Dostupné z: http://www.martinfowler.com/articles/newMethodology.html

SNOWDEN, David, Anna Royzman, Justin Rohrman a Michael Larsen. Sensemaking with Cynefin. QUALITEST. [online]. 17.11.2016 [cit. 2016-12-18]. Dostupné z: https://www.qualitestgroup.com/The-Testing-Show/sensemaking-with-cynefin/

6. Seznam obrázků