Univerzita Hradec Králové

Fakulta informatiky a managementu

Nástroje automatizace testování API webové aplikace

Diplomová práce

Autor práce: Bc. Lenka Bártová
Vedoucí práce: Ing. Karel Mls Ph.D.
Studijní obor: Informační management

Prohlášení:

Prohlašuji, že jsem diplomovou práci zpracovala samostatně a s použitím uvedené literatury.

V Hradci Králové dne 29.4.2022 Lenka Bártová

Poděkování:

Na tomto místě bych chtěla poděkovat panu Ing. Karlovi Mlsovi, Ph.D., za cenné připomínky, rady a metodické vedení mé diplomové práce.

Anotace

Práce se zabývá možnostmi automatizovaného testování aplikačního webového rozhraní v kontextu decision-making problematiky.

V první části práce je vymezena problematika a základní pojmy, týkající se testování softwaru obecně, dále automatizovaného testování, systémů pro podporu managementu a rozhodovacích procesů, zásadní pro pochopení dané problematiky. Tato část slouží jako východisko pro následující část práce.

Ve výzkumné části práce je zpracována analýza konkrétní modelové situace, na jejímž základě jsou v další části práce stanovena výběrová kritéria, vybrány automatizační nástroje pro testování aplikačního rozhraní webové aplikace, a na základě toho je následně sestaven rozhodovací model pro podporu rozhodnutí o výběru vhodného nástroje. Tento model je poté na tuto modelovou situaci aplikován a vybrán vhodný nástroj pro danou situaci.

Závěr práce shrnuje nabyté teoretické a praktické poznatky vyplývající z této práce.

Klíčová slova: Testování software, Automatizace testování software, Systémy pro podporu rozhodování, Vícekriteriální rozhodování

Annotation:

The diploma thesis deals with the possibilities of automated testing of the application web interface in the context of decision-making issue.
The first part of the thesis defines the issues and basic concepts related to software testing in general, automated testing, decision support systems and decision-making processes, essential for understanding the issue.
This part serves as a basis for the next part of the thesis.
In the research part of the thesis, an analysis of a specific model situation is elaborated, based on which selection criteria are determined in the next part of the thesis, automation tools are selected for testing the application interface of the web application, and on the basis of this, a decision model is subsequently compiled to support the decision on the selection of a suitable tool. This model is then applied to this model situation and the appropriate tool for the situation is selected.
The conclusion of the thesis summarizes the acquired theoretical and practical knowledge resulting from this theses.

Keywords: Software testing, Automated software testing, Decision support systems, Multicriterial decision making

Obsah

1. Úvod

V současné době je software zaváděn do všech oblastí lidské činnosti. Již téměř neexistují zařízení či stroje, jejichž ovládání není řízeno prostřednictvím softwaru, jako je tomu například u automobilů, ve kterých je stále více funkcí řízeno mikroprocesory, od řízení motoru po převodovku či brzdy. Na zajištění kvality softwarového systému by tudíž měl být kladen velký důraz v každém vývojovém procesu. [37]

Automatizované testování softwarových produktů se stalo již v minulosti nedílnou součástí vývoje softwaru, a to zejména v agilních projektových metodikách, a je mu věnováno stále více pozornosti a prostředků, což také bylo jedním z motivů autorky pro zvolení tohoto tématu, stejně tak jako zájem rozšířit si povědomí o dalších metodách, technologiích a nástrojích pro automatizovaného testování. Protože správně zvolená strategie, resp. správné stanovení, které části systému je výhodné automatizovat, automatizace testování může být způsobem, jak zvýšit efektivitu celého testovacího procesu, včasnějšího odhalení defektu, snížení nákladů a v konečném důsledku zvýšení kvality produktu a tím pádem i spokojenost zákazníka.

Tomuto tématu již v minulosti byla věnována pozornost v rámci práce s názvem „Volba nástroje pro funkční automatizované testování webových aplikací“ studenta Jana Faška z Vysoké školy ekonomické. Ve výše zmíněné práci dochází k výběru nejvhodnějšího nástroje pomocí manuálních výpočtů bodovací metodou hodnocení variant.

V této práci jsou varianty jednotlivých kritérií porovnávány párovou metodou vzájemně mezi sebou a na základě toho je vždy danému kritériu pro vybraný nástroj stanovena váha. Tato práce dále rozšiřuje problematiku o využití technologie systémů pro podporu manažerského rozhodování, pomocí kterého je možné vytvořit znovupoužitelný model, který je dále možný snadno modifikovat pro různé alternativy a různá kritéria, a dosažení výsledného rozhodnutí pak může být snazší.

První část práce je věnována teoretickým východiskům. V kapitole 4 je představen základní vhled do problematiky testování obecně, jeho význam a základní rozdělení typů testů z různých hledisek. Dále je zde nastíněna problematika automatizovaného testování, zaměřená na programové aplikační rozhraní webové aplikace.
Kapitola 5 je věnována rozhodovacím procesům a metodám vícekriteriálního rozhodování.
Kapitola 6 vymezuje pojem systémy pro podporu managementu a jejich princip.
V kapitole 7 je definována modelová organizace, sloužící jako východisko pro výzkumnou část diplomové práce. Na základě takto definované situace a dodatečné analýzy zdrojů zabývajícími se danou problematikou jsou dále v kapitole 8 stanovena výběrová kritéria pro výběr vhodného automatizačního nástroje, stanoveny preference dílčích alternativ a na základě toho jsou vypočteny váhy jednotlivých alternativ

V kapitole 9 jsou vybrány vhodné nástroje pro automatizaci testů aplikačního programového rozhraní. Následně jsou nástroje zhodnocena z hlediska naplnění stanovených kritérií.
Kapitola 10 se zabývá softwarovým řešením rozhodovacího problému o výběru vhodného automatizačního nástroje. Na základě stanovených kritérií a alternativ je vytvořen rozhodovací model v nástroji Criterium Decision Plus, pomocí kterého je stanoveno optimální řešení pro danou modelovou situaci.
Tato práce a její výsledky může mít praktický přínos pro vznikající či již existující podniky, které se nachází teprve ve fázi plánování zavedení automatizace testování do procesu vývoje aplikace a může tak významně napomoci zvolit nejvhodnější řešení.

2. Cíl práce

K dosažení hlavního cíle práce byly stanoveny následující dílčí cíle:

1. Stanovení modelové situace, na kterou bude následně aplikován rozhodovací model.
2. Stanovení výběrových kritérií pro volbu vhodného nástroje pro automatizaci testování.
3. Výběr automatizačních nástrojů k porovnání.
4. Návrh rozhodovacího modelu pro podporu rozhodnutí o vhodném automatizačním nástroji.
5. Výběr vhodného nástroje pro danou modelovou situaci.

3. Metodika zpracování

V průběhu zpracovávání diplomové práce je využito více různých zdrojů informací. Ke zpracování teoretických východisek je využita metoda literární rešerše relevantních zdrojů, zejména knižních publikací, odborných článků a vybraných internetových zdrojů.

V navazující výzkumné části je definována modelová situace, na základě které jsou dále analyzovány požadavky, ze kterých byla odvozena kritéria pro výběr vhodného automatizačního nástroje. Další kritéria byla stanovena pomocí rešerše internetových zdrojů, zabývající se problematikou výběru vhodného automatizačního nástroje. Dále byly stanoveny hodnoty, kterých daná kritéria mohou nabývat. Na základě jejich přípustných kombinací byly stanoveny dílčí alternativy. Tyto alternativy byly následně seřazeny dle stupně preference – od nejvyšší po nejnižší za účelem výpočtu jejich vah pomocí Saatyho metody párového porovnávání, které jsou dále aplikovány v rozhodovacím modelu.

Výběr vhodných alternativ byl založen rovněž na analýze zdrojů představující dostupné automatizační nástroje. Byly vybrány nástroje, které svým rozšířením zaujímají významnou část trhu.

Na základě definovaných kritérií a alternativ byl vytvořen rozhodovací model v nástroji Criterium Decision Plus, jehož pomocí byl metodou AHP vypočítán konečný výsledek. Pro hodnocení jednotlivých alternativ vůči kritériím byla aplikována přímá metoda, byly zadány vypočtené váhy. Pro stanovení vah jednotlivých kritérií byla využita Saatyho porovnávací metoda.

Teoretická východiska

4. Testování softwaru

4.1. Vymezení pojmu

Testování softwaru je způsob, jak zhodnotit kvalitu vyvíjeného softwaru a současně tak významně přispívá k jejímu zvyšování a eliminaci rizika selhání systému v provozu. Kvalita softwaru je však více než jen eliminace defektů nalezených během testování. Dle ISO standardu 9126-1 tento pojem zahrnuje následující faktory: [37]

- Funkčnost – je podstatou každého produktu či služby a podstatou je ověření správné funkčnosti všech funkcionalit. Výsledkem testování funkčnosti je pak potvrzení, že byl systém implementován dle požadavků popsaných ve specifikaci produktu. [11]
- Spolehlivost – charakteristika spolehlivosti definuje schopnost systému udržovat poskytování služeb za definovaných podmínek neboli správně fungovat v konkrétní situaci po určitou dobu. [11]
- Použitelnost – tento aspekt je klíčový pro přijetí systému uživatelem. Použitelnost je závislá na funkčnosti a spočívá ve snadnosti použití uživatelem dané funkce. Dále je ovlivněna také výkonností. Například při delší době odezvy systému by nemohl být snadno použitelný. [11]
- Výkonnost – charakteristika výkonnosti se týká systémových zdrojů využívaných při poskytování požadovaných funkcionalit. Například množství místa na disku, paměti a další. [11]
- Udržovatelnost – týká se snadnosti, s jakou můžete opravit, zlepšit a pochopit softwarový kód. Údržba softwaru je fáze cyklu vývoje softwaru, která začíná poté, co zákazník obdrží produkt. [11]
- Přenositelnost – tato charakteristika se týká toho, jak dobře se software může přizpůsobit změnám ve svém prostředí nebo svým požadavkům. [11]

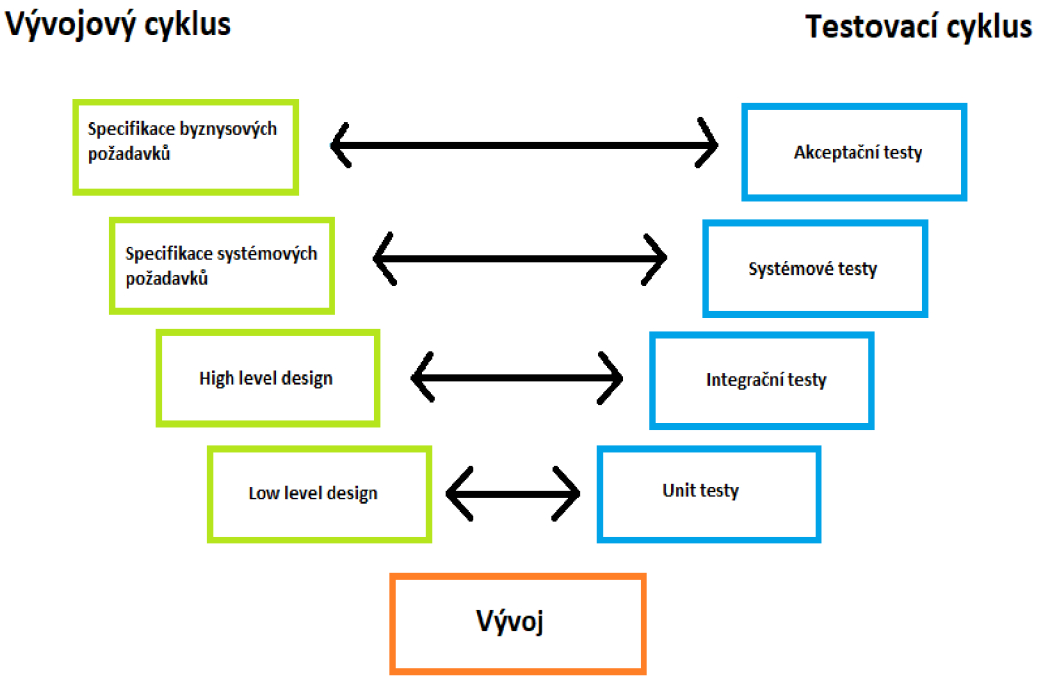
Testování softwaru však nespočívá pouze ve spouštění testovaného softwaru a následné hodnocení výsledků. Je to pouze jedna z činností v rámci celého procesu. Tento proces je dle organizace ISTQB definován těmito aktivitami:

- Plánování testování a analýza
- Monitoring a řízení testování
- Návrh a implementace testů
- Reportování průběhu testů a jejich výsledků
- Hodnocení kvality testovaného objektu

Existuje mnoho různých typů softwarových testů, z nichž každý má své specifické cíle a strategie. Jednotlivé úrovně a typy testů budou představeny v následujících kapitolách. [18]

4.2. Úrovně testování

Úrovně testování jsou skupiny testovacích činností, které jsou organizovány a řízeny společně. Jedná se o rozdělení dle doby, kdy a kým je aplikace testovaná. [37]
Podstatu úrovní testování a vztah mezi fázemi vývoje a druhy testování znázorňuje obrázek 1. V průběhu procesu vývoje slouží jednotlivé typy testů k ověření, že daná fáze vývoje proběhla správně a výsledky odpovídají očekávání. Jedná se o jednoduchý V-Model využívaný v agilních metodikách vývoje.



Obrázek 1 V-model (Zdroj: vlastní zpracování, inspirováno https://www.tutorialscampus.com/sdlc/img/v-model.png)

Levá část obrázku poukazuje na jednotlivé fáze, kterými prochází vývoj softwaru. V pravé části pak naleznete jednotlivé druhy testování, které se využívají v jednotlivých fázích vývoje.

4.2.1. Unit testy

Unit testy jsou využívané v úvodní fázi testování. Testují se jednotlivé komponenty na úrovni modulů, objektů a tříd. V této fázi testů mohou testeři a vývojáři pomocí této metody izolovat každý modul, identifikovat a opravit systémové vady ve velmi rané fázi životního cyklu vývoje softwaru (SDLC). Déle je cílem těchto testů ověřit, že implementovaný kód funguje správně dle specifikace a rovněž správnost a konzistentnost při rozšiřování funkčnosti. [40]

4.2.2. Integrační testy

Integrační testování je druhou úrovní testů, zaměřeno na interakce mezi komponentami nebo systémy. Cílem integračního testování je ověřit, zda funkcionální i nefunkcionální rozhraní odpovídá specifikaci, nalézt defekty v interakci mezi integrovanými jednotkami, v komponentách nebo systémech a prevence jejich proniknutí do dalších úrovní testování. Na této úrovni jsou jednotlivé jednotky / komponenty integrovány a testovány jako celek. Typické testované objekty pro integrační testování jsou subsystémy, databáze, infrastruktura, rozhraní, aplikační rozhraní (API). [34]

4.2.3. Systémové testy

Systémové testování je úroveň testování softwaru, kde je testován kompletní a integrovaný software. Účelem této zkoušky je vyhodnotit shodu systému se specifikovanými požadavky. Typickými objekty integračního testování jsou aplikace, hardwarové/softwarové systémy, operační systémy, testovaný systém, konfigurace systému a konfigurační data. Systémové testování by mělo být zaměřeno na celkové chování systému jako celku, jak funkcionální, tak nefunkcionální. Testování systému jako celku je nejkomplexnější úrovní testování a provádí se mnoho typů testů. [34]

- Smoke testy
- Funkční testy
- Regresní testy
- Výkonnostní testy – viz. kapitola 4.1.2.2

- Bezpečnostní testy

4.2.4. Akceptační testy

Poslední úrovní testování na straně poskytovatele softwaru jsou akceptační testy, které probíhají těsně před nasazením do produkce. Zde je produkt testován z uživatelské perspektivy. Někdy mohou být tyto testy prováděny již na nižších úrovních testování. Předmět akceptačních testů se odvíjí od funkčních a systémových požadavků zákazníka, na základě kterých jsou definovány případy užití. Jejich rozsah se dále může lišit na základě identifikovaných rizik vyvíjeného softwaru. [37]

4.3. Obecné typy testů

Typy testů jsou obecné skupiny aktivit zaměřené na konkrétní charakteristiky softwarového systému nebo jeho částí na základě konkrétních testovacích cílů. Organizace ISTQB definuje následující čtyři typy tesů dle způsobu testování [18]:

4.3.1. Funkcionální testy

Účelem funkcionálního testování tzv. testování černé skříňky je ověřování funkcí, který by měl daný systém vykonávat a neuvažuje vnitřní strukturu systému. Tyto požadavky jsou obvykle definovány ve specifikaci byznysových požadavků, či funkční specifikaci. V případně agilního vývoje často bývají i nedokumentované. [18] Zjednodušeně řečeno jsou u tohoto typu testů hodnoceny výstupy pro dané vstupy, viz. obrázek 2. Funkcionální testování by mělo být prováděno na všech úrovních testování. Důkladnost funkcionálního testování může být měřena rozsahem funkcionálního pokrytí. Funkcionální pokrytí je míra, do jaké je určitá funkcionalita prozkoumána testy a vyjadřuje se jako procento pokrytí testy pro daný typ prvku. Techniky testování černé skříňky jsou například analýza hraničních hodnot, nebo rozdělení tříd ekvivalence. [39]

Obrázek 2 Testování černé skříňky (zdroj: vlastní zpracování, inspirováno https://softwaretestingfundamentals.com/wp-content/uploads/2012/02/black-box-testing.jpg)

4.3.2. Nefunkcionální testy

Nefunkcionální testování se soustředí na aspekty vyvíjeného systému, které nejsou přímo spojeny s funkcionalitou aplikace. Testuje charakteristiky jako jsou použitelnost, výkonnost, odezva nebo bezpečnost. Testování je obvykle prováděno na všech úrovních testování a včasné odhalení nefunkcionálních defektů je klíčové pro úspěch celého projektu. Příkladem takových testů jsou performance, load nebo stress testy.

Nefunkcionální požadavky bývají často opomíjeny, což může vést k selhání dané aplikace. Proto by tyto požadavky měli být zdokumentovány stejně jako požadavky funkční. Funkční i nefunkční požadavky jsou stejně důležité a musí být důkladně testovány pro úspěch aplikace. [10]

4.3.3. Testy bílé skříňky

Techniky testování bílé skříňky odvozují testy z interních struktur nebo implementace systému. Interní struktura může zahrnovat kód, architekturu, pracovní toky anebo datové toky v rámci systému. [18]. Na rozdíl od testování černé skříňky se tento typ nezaměřuje na funkčnost. Podstatou testování bílé skříňky je ověření že jsou dané moduly správně implementovány a operace jsou prováděny dle specifikace, na základě revize kódu. [38]

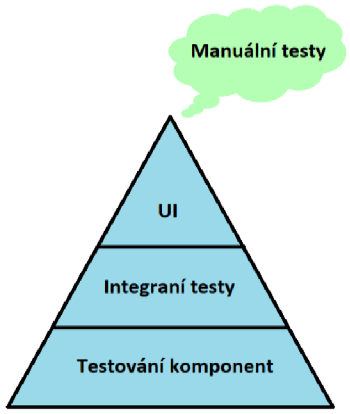
4.3.4. Testy související se změnami

Testy související se změnami následují po provedení změny v systému z důvodu opravy defektu, implementace nové funkcionality, nebo úprava stávající implementace. Je nutno tedy zjistit, že oprava defekt odstranila, nebo že daná funkcionalita byla správně naimplementovaná a nebyly přitom způsobeny žádné nežádoucí následky. Po opravě defektu následuje tzv. Konfirmační testování na nové verzi systému, kde je třeba otestovat všechny případy užití, které selhaly v důsledku defektu. Hlavním cílem konfirmačních testů je tedy potvrdit, že byl daný defekt odstraněn. V návaznosti na konfirmační testování musí být provedeny tzv. Regresní testy. Přestože byl daný defekt úspěšně odstraněn, existuje riziko, že změna provedená v části kódu, ať už v rámci opravy nebo jiné změny, může náhodně

ovlivnit chování jiné části kódu. Změna může mít dopad na stejnou komponentu, na které byl defekt nalezen, ale i na jiné komponenty, či dokonce jiný systém. Proto je třeba identifikovat možné důsledky a na základě toho provést regresní testy. [37]

4.4. Manuální vs. Automatizované testování

Pro účely této práce je neméně důležité rozdělení podle toho, kdo testy vykonává. Rozlišujeme tedy testy manuální a automatizované. Vztah těchto dvou typů vystihuje obrázek 3 – rozšířená Cohnova pyramida o podstatnou část, a tou je manuální testování, které by vždy mělo být nad každou z těchto úrovní a nikdy nelze nahradit zcela. [26]



Obrázek 3 Pyramida úrovní automatizace (zdroj: vlastní zpracování, inspirováno Layers of Test Automation | QA Matters (qa-matters.com))

Často se totiž objevuje tendence považovat manuální testování za pouhé klikání do aplikace, a lze tak tento proces plně zautomatizovat. Není však zcela správná. Manuální testy provádí tester a využívá k tomu veškeré své předchozí zkušenosti. Jsou to právě zkušenosti, kreativita a intuice testera, které nelze nahradit a bez čehož by nebylo možné některé defekty odhalit. Nicméně dobře navrhnutá a implementovaná automatizace však může velmi významně přispět ke zkvalitnění testovacího procesu. Pokud není prováděna pravidelná údržba automatizovaných testů, časem mohou přestat odhalovat nové chyby v systému. Automatizovanému

testování, které je hlavním předmětem této práce, budou věnovány následující kapitoly, ve kterých bude podrobněji popsáno. [7]

5. Automatizované testování

5.1. Vymezení pojmu

V této kapitole je vymezen pojem automatizované testování.

Dle ISTQB je automatizované testování definováno jako *„Použití softwaru k provádění nebo podpoře testovacích činností, např. správa testů, design testů, provádění testů a kontrola výsledků.“*

Automatizované testování je proces, který ověřuje, zda software funguje správně a splňuje požadavky před tím, než je uvolněn do produkčního prostředí. Tato metoda testování softwaru používá skriptované sekvence, které jsou vytvářeny a spouštěny pomocí testovacích softwarových nástrojů. Automatizační testovací nástroje provádí testy softwaru, poskytují reporty výsledků a porovnávají výsledky s předchozími testovacími běhy. [6]

Proces automatizace testů zahrnuje činnosti, které jsou prováděny během automatizace testů softwarových aplikací. Tento proces probíhá v následujících krocích:

- Porozumění požadavkům a rozhodování o automatizaci – první a nejdůležitější činností je porozumět požadavkům a usnadnit tak další fáze definování rozsahu automatizace spolu s výběrem správného nástroje. [6]
- Definice rozsahu automatizace
- Výběr vhodného nástroje pro podporu automatizace – Výběr nástroje závisí na různých faktorech, jako je cíl projektu, odborné znalosti (například programování), rozpočet projektu (bezplatný x placený) atd. Více v kapitole 6, která bude věnována právě stanovení kritérií pro výběr vhodného nástroje. [6]
- Vytvoření frameworku pro automatizaci a definování infrastruktury – k vytvoření znovupoužitelných, udržovatelných a robustních automatizovaných testů je vhodné vytvořit automatizační rámec, který je chápán jako souhrn pravidel, stanovení priorit provádění testů, různé

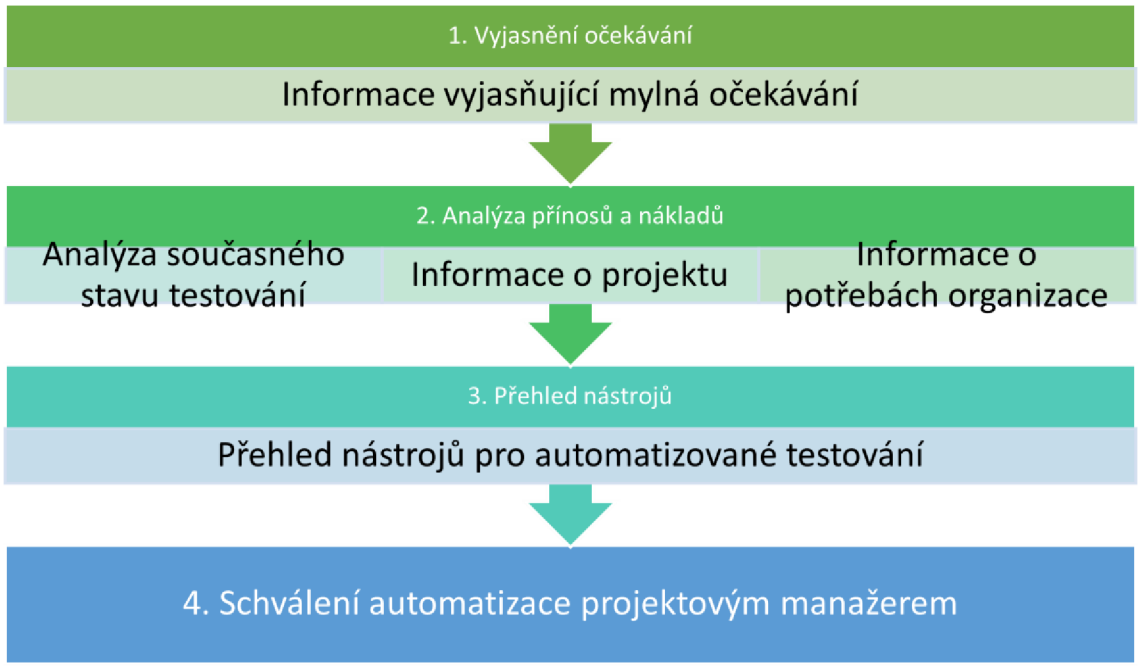
postupy pro optimalizaci celého procesu testování, správa testovacích dat, nebo formát reportů. [6]

- Vývoj testovacích scriptů
- Provedení testů a analýza – Během této fáze se spouštějí automatizační skripty. Po provedení testů jsou dostupné výsledky a reporty, které jsou poté analyzovány. [6]
- Údržba automatizovaných testů

5.2. Rozhodování o automatizaci

Zavedení automatizace je vždy nutné začít procesem, který vede k rozhodnutí, zda automatizovat, či nikoli. Dále definovat, co se od automatizace očekává, a zda jsou tato očekávání reálná. Stanovit očekávané přínosy implementované automatizace a porovnat je s očekávanými náklady.

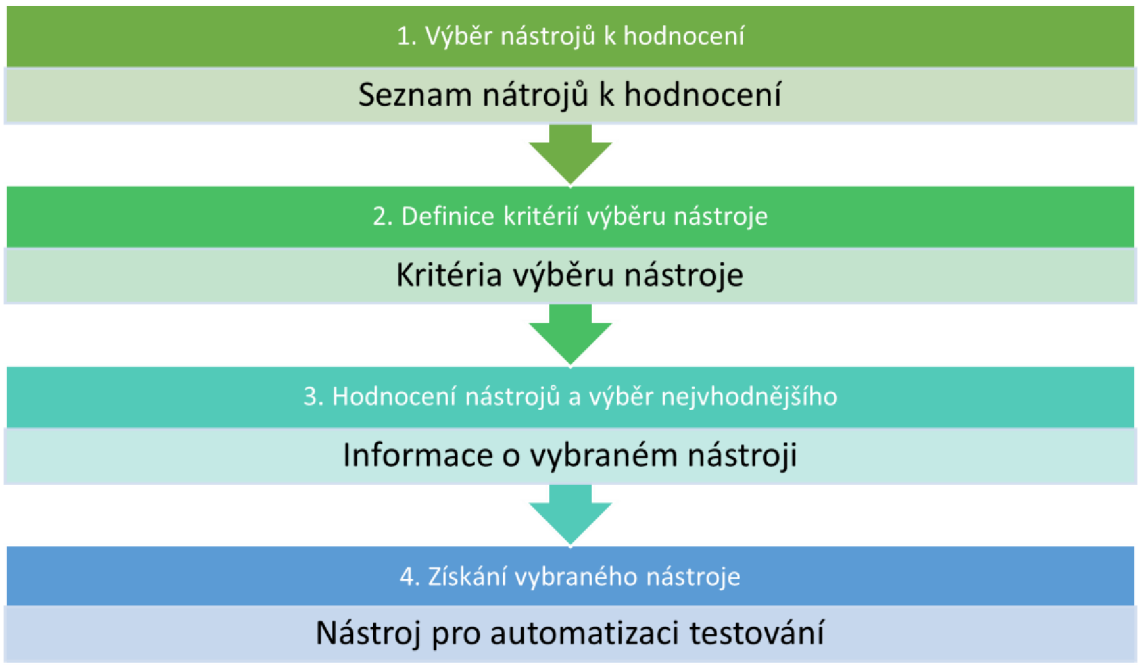
Jednotlivé aktivity procesu rozhodování o automatizaci a jejich výstupy znázorňuje obrázek č. 4. Aktivity 1–3 jsou zodpovědností test manažera, schvalování rozhodnutí (krok 4) provádí projekt manažer.



Obrázek 4 Procecs rozhodování o automatizaci (Zdroj: vlastní zpracování, inspirováno: DUSTIN, Elfriede. Automated software testing: introduction, management, and performance. Boston, Massachusetts, USA: Addison-Wesley, 1999. ISBN 0201432870. strana 31.)

5.3. Výběr automatizačního nástroje

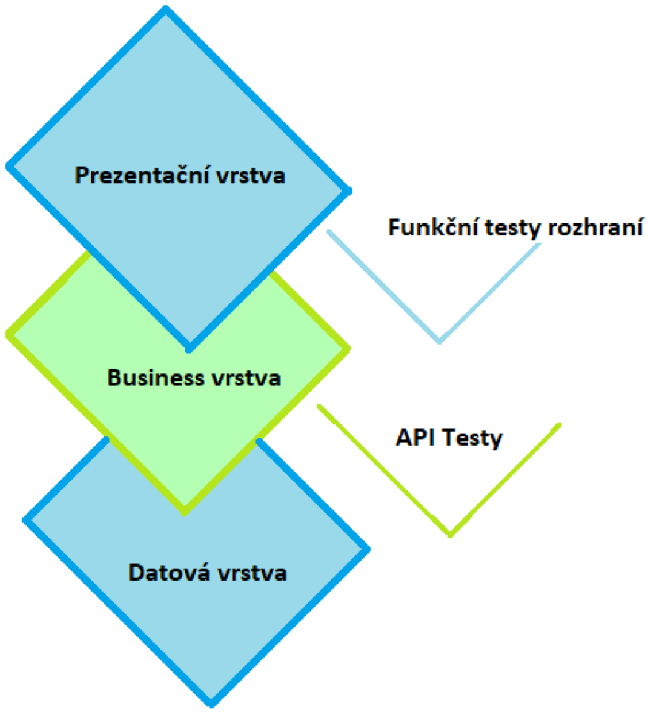
V případě, že bylo rozhodnuto, že bude automatizace zavedena, je dále nutné vybrat vhodný testovací nástroj. V rámci tohoto procesu je nutné varianty pečlivě zvážit a zhodnotit s ohledem na potřeby daného projektu. Nástroje by měli být hodnoceny podle předem stanovených kritérií tak, aby bylo dále možné je seřadit dle vhodnosti pro daný projekt. Aktivity procesu výběru nástroje jsou znázorněny na obrázku č.5.



Obrázek 5 Proces výběru automatizačního nástroje (zdroj: vlastní zpracování, inpirováno dle: DUSTIN, Elfriede. Automated software testing: introduction, management, and performance. Boston, Massachusetts, USA: Addison-Wesley, 1999. ISBN 0201432870. strana 69)

5.4. Testování aplikačního programového rozhraní

Testování API je součástí testování softwaru, které zahrnuje testování aplikačních programových rozhraní na úrovni vrstvy dat a business logiky, na které probíhá interakce mezi uživatelským rozhraním a datovou vrstvou. V kontextu webových aplikací zahrnuje testování API hlavně testování webových REST/SOAP API. [4]



Obrázek 6 Aplikační vrstvy (zdroj: vlastní zpracování, inspirováno https://www.katalon.com/api-testing/)

Existuje mnoho aspektů testování API, nicméně z obecná podstata spočívá v odesílání požadavků na jeden, či několik koncových bodů API (endpointů) a ověřování, tzv. validace jejich odpovědi. Na rozdíl od testování uživatelského rozhraní, kdy se ověřuje zejména vzhled a chování webového rozhraní, se během testování API klade důraz zejména na testování funkčností, obchodní logiky, odezvy dat, zabezpečení či výkon dané části aplikace. [5]

V rámci testování API je taktéž možno provádět různé typy testů již zmíněné –unit testy, funkční testy, zátěžové testy či bezpečnostní testy. Automatizovat je vhodné v případech, kdy je třeba testovací případy provádět opakovaně, či před každým vydáním. [32]

5.5. Důvody k automatizaci testování

- Schopnost testovacích nástrojů zaznamenávat a reportovat výsledky a možnost je porovnat s výsledky předchozích běhů testů. – [37]
- Aplikační programové rozhraní bývá podstatně méně častěji měněno oproti uživatelskému rozhraní, které i při častých změnách rozhraní zůstává nezměněno. Proto testování API představuje hlavní roli v oblasti automatizace testování. [12]
- Testy mohou být prováděny opakovaně a nezávisle na uživatelském prostředí a mohou tak sloužit jako užitečný nástroj pro regresní testy. [12]
- Schopnost odhalení defektu ještě před integrací, či dokonce před samotným vývojem uživatelského rozhraní. [12]
- Snížení nákladů na testování systému z dlouhodobého hlediska. [12]
- Možnost vysokého pokrytí rozsahu testování díky postupnému zvyšování počtu testovacích případů. [12]

5.6. Automatizační testovací rámce

Testovací rámec je sada pokynů nebo pravidel používaných pro vytváření a navrhování testovacích případů. Rámec se skládá z kombinace postupů a nástrojů, které jsou navrženy tak, aby pomohly testerům testovat co nejefektivněji.
Tyto postupy zahrnují standardy kódování, metody zpracování testovacích dat, úložiště objektů, procesy pro ukládání výsledků testů nebo informace o tom, jak získat přístup k externím zdrojům. [31]

5.6.1. Typy automatizačních testovacích rámců:

- **Linear Automation Framework**

Lineární skriptovací rámec je základní úroveň testovacího automatizačního rámce, který je ve formě "Record and playback". Tento typ architektury se používá k testování menších aplikací, jelikož se vytváření a provádění testovacích skriptů provádí jednotlivě pro každý testovací případ. Jednotlivé kroky jsou nahrávány – například navigace, přihlášení uživatele, nebo vyžádání přístupových údajů. Tyto skripty jsou pak opakovaně přehrávány za účelem testování. [31]

- **Modular Based Testing Framework**

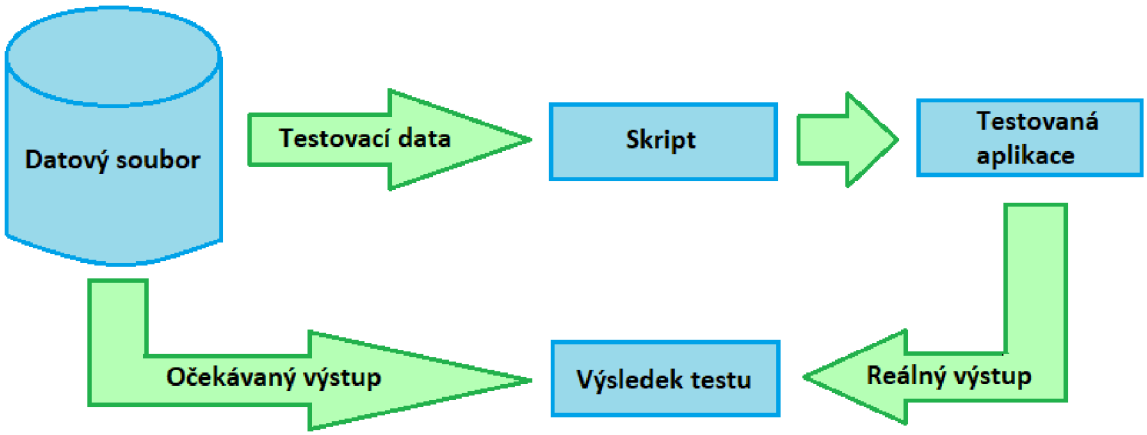
Implementace modulárního rámce vyžaduje rozdělení testované aplikace do samostatných jednotek, funkcí nebo sekcí, z nichž každá bude testována izolovaně. Po rozdělení aplikace na jednotlivé moduly je pro každou část vytvořen testovací skript, které jsou následně zkombinovány pro vytvoření větších testů hierarchickým způsobem. Tyto větší sady testů pak představují různé testovací případy. [31]

- **Library Architecture Testing Framework**

Tento rámec je založen na modulárním rámci rozšířený o další výhody. Testovací skripty jsou zpočátku zaznamenávány metodou "Record & Playback". Později jsou běžné úlohy uvnitř skriptů identifikovány a seskupeny do funkcí. Tyto funkce jsou volány hlavním testovacím skriptem nazvaným Driver různými způsoby pro vytvoření testovacích případů. [31]

- **Data-driven testing framework**

Technika automatizace testů řízená daty (Též nazývaná parametrizované testování), umožňuje otestovat aplikaci oproti různým datovým sadám a datovým hodnotám a odděluje testovací logiku od testovacích dat a je tak možné spouštět jeden testovací skript s různými vstupními hodnotami uloženými vstupními hodnotami uloženými v excelu, CSV nebo XML formátech. Umožňuje tak testovat pozitivní i negativní scénáře. Nástroje založené na této technice bývají dodávány spolu s inteligentními řešeními správy testovacích dat. [31]



Obrázek 7 Data-driven testing framework (zdroj: vlastní zpravocání, inspirováno https://www.guru99.com/images/1/032318_1019_WhatisDataD1.png)

- **Keyword-driven framework**

V rámci řízeném klíčovými slovy je každá funkce testované aplikace rozložena v tabulce s řadou instrukcí v po sobě jdoucím pořadí pro každý test, který je třeba spustit. S tímto přístupem jsou klíčová slova také uložena jako externí data. Klíčová slova jsou součástí skriptu představujícího různé akce prováděné za účelem testování aplikace. Ty mohou být označeny jednoduše jako *click* nebo *login*. [31]

- **Hybrid framework – kombinovaný přístup**

Stejně jako u většiny testovacích procesů dnes se automatizované testovací rámce začaly integrovat a vzájemně se překrývat. Jak název napovídá, hybridní rámec je kombinací kteréhokoli z výše uvedených rámců, které byly vytvořeny tak, aby využívaly výhod některých a zmírňovaly slabiny jiných. Většina existujících nástrojů nabízí tento přístup. [31]

6. Aplikační programové rozhraní – API

API je zkratka pro Application Programming Interface, které software používá pro přístup k datům, serverovému softwaru nebo jiným aplikacím. Zjednodušeně řečeno se jedná o rozhraní, které umožňuje dvěma stranám (například aplikacím) vzájemně komunikovat. Rozhraní API používají definované protokoly, které vývojářům umožňují rychle a ve velkém měřítku vytvářet a integrovat aplikace. Protokoly si lze představit jako soubor pravidel, která definují, jakým způsobem spolu mohou aplikace, či přístroje komunikovat. Příkladem toho, jak API funguje, může být přihlášení do internetového bankovnictví. Pro získání přístupu k účtu je voláno rozhraní API, aby získalo přihlašovací údaje. Daná aplikace poté k těmto informacím přistupuje ze svého serveru a vrací data do aplikace. To je příkladem webového rozhraní API, která jsou velmi často využívaná, ale omezená pouze na web. Existují rozhraní API pro prakticky každý stroj nebo systém, který očekává interakci s jinými počítači nebo systémy. Například REST API, API na bázi klient-server, XML-RPC, JSON-RPC, SOAP API. V souvislosti s webovými aplikacemi, budou pro účely této práce detailněji popsána rozhraní REST a SOAP. [13]

6.1. REST API

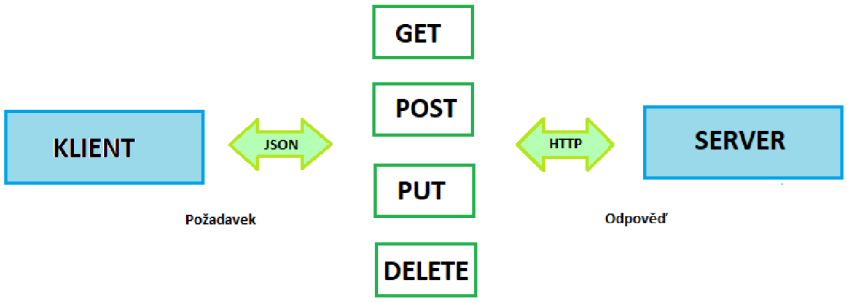
REST je sada architektonických omezení, nikoli protokol nebo standard. Vývojáři rozhraní API můžou REST implementovat různými způsoby.

Když je požadavek klienta proveden prostřednictvím rozhraní REST API, přenese reprezentaci stavu na endpoint. Tyto informace či reprezentace jsou přenášeny v jednom z několika formátů prostřednictvím protokolu http – JSON (Javascript Object Notation), HTML, XLT, Python, PHP nebo prostý text. JSON je obecně nejoblíbenější formát souboru, protože navzdory svému názvu je jazykově nezávislý a čitelný pro lidi i stroje. Někdy jsou data také vytvářeny například ve formátech XML nebo YAML. [30]

Podstata fungování REST API je znázorněna na obrázku 8.

Pro komunikaci jsou využívané tyto metody:

- GET – získání, čtení dat
- POST – Přidávání nových dat
- PUT – Aktualizace existujících dat
- DELETE – Smazání existujících dat

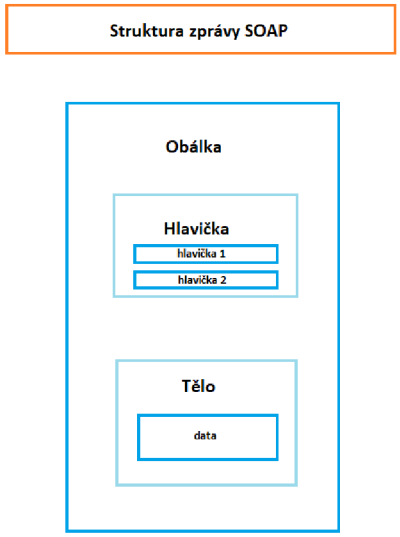


Obrázek 8 Rest API

6.2. SOAP API

Simple Objects Access Protocol je webový komunikační protokol. Používá se pro zasílání zpráv, výměnu strukturovaných dat, nejčastěji přes HTTP/HTTPS. K přenosu využívá jediný formát XML. SOAP umožňuje procesům komunikovat napříč různými platformami, jazyky a operačními systémy. V porovnání s REST API má striktně stanovené standardy. Musí splňovat určitou strukturu zprávy, pravidla kódování a standardy pro požadavky a odpovědi. [1]
Struktura zprávy SOAP je znázorněna na obrázku 9.

- Obálka (envelope) - je základním a nezbytným prvkem každé zprávy, která určuje začátek a konec SOAP zprávy. [1]
- Hlavička (header) - určuje specifika, dodatečné požadavky na zprávu, například autentizaci. [1]
- Tělo (body) – obsahuje request nebo response. [1]



Obrázek 9 Struktura zprávy SOAP (zdroj: vlastní zpracování, inspirováno https://content.altexsoft.com/media/2019/08/word-image-8.png)

7. Rozhodovací proces

Rozhodování je nedílnou součástí managementu a je považováno za primární a klíčovou funkci řízení. Rozhodování hrají důležitou roli, definuje organizační a manažerské činnosti. Rozhodnutí může být definováno jako možnost zvolená z množiny přípustných alternativ k dosažení organizačních nebo manažerských cílů, a jsou přijímána na všech úrovních řízení. [24]

Právě v kontextu organizací je nutno poznamenat, že rozhodování vyžaduje přístup ke správným a úplným informacím, a zároveň schopnost dávat těmto informacím správný význam. V opačném případě tzv. "asymetrie informací", může vést ke špatným rozhodnutím. Rozhodovací proces je nepřetržitou a nepostradatelnou součástí řízení a prostupuje všemi činnostmi organizace. Důležitými předpoklady úspěšného manažera jsou tedy odborné znalosti spolu s manažerskými dovednostmi a zkušenostmi, a v neposlední řadě také schopnost vzít v úvahu různé názory a odlišné perspektivy a dospět tak ke správnému rozhodnutí. Výběr z množiny možných variant usnadňuje vícekriteriální analýza. Strukturování variant do hierarchie, popis jednotlivých kritérií rozhodování, jejich kvantifikace a vyhodnocení, to umožňuje vícekriteriální rozhodovací proces. [15]

Řešení každého rozhodovacího problému by mělo splňovat následující logický postup, který je také znázorněn na obrázku 10:

- Definování problému
- Detailní analýza problému a shromažďování informací
- Stanovení a hodnocení alternativ řešení
- Výběr optimální možnosti
- Plánování a realizace
- Případné přijetí následných opatření

Obrázek 10 Rozhodovací proces (zdroj: vlastní zpracování, inspirováno https://th.bing.com/th/id/OIP.1msCGu9291oO6KjD9iiNkwHaHa?pid=ImgDet&rs=1)

7.1. Vícekriteriální rozhodování

Vícekriteriální rozhodovací analýza (MCDA) spočívá v hodnocení možných alternativ podle různých, často konfliktních kritérií. Přičemž alternativa hodnocená podle jednoho kritéria, nebývá hodnocena stejně podle kritéria jiného. Výstupem takového procesu je uspořádání variant dle určitých preferencí s cílem stanovení nejvýhodnější neboli optimální varianty.

Účelem využití MCDA je pouze pomoc při vykonávání rozhodnutí, jelikož neexistuje nic jako správná odpověď ani optimum ve vícekriteriální oblasti. Dále je jejím účelem lépe strukturovat daný problém a lépe mu porozumět. Svou roli také vždy hrají postoje a priority zúčastněných stran.

V první fázi tohoto procesu jsou stanoveny váhy jednotlivým kritériím, čímž jsou jednotlivá kritéria příslušně kvantifikovány a je definován jejich číselný význam, resp. Jejich preference. Čím vyšší váha, tím je jeho preference vyšší. Obvykle jsou tyto váhy normovány tak, aby byl jejich součet roven jedné. Součet všech vah je vydělen jejich počtem. Ve druhé fázi dochází k hodnocení variant a výběru té nejoptimálnější. [15]

7.2. Analytický hierarchický proces – AHP

AHP je zkratka pro analytický hierarchický proces a patří k multikriteriálním rozhodovacím metodám (MCDM) založených na matematických základech, a je jednou z nejvyhledávanějších technik v dnešní manažerské vědě.

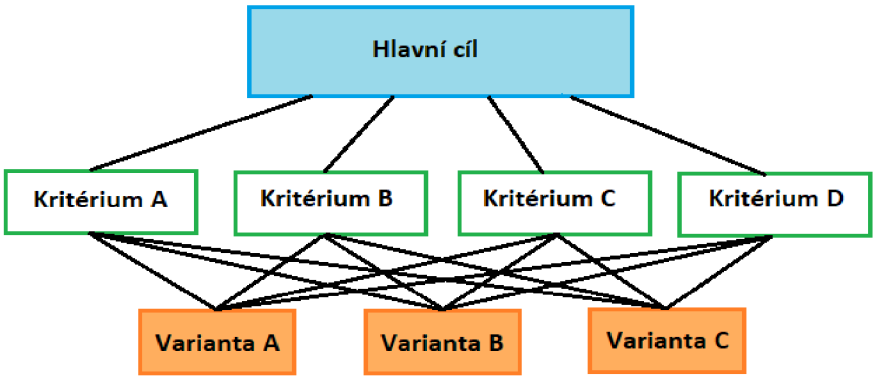
Tato metoda připouští existenci více kritérií, ale zároveň význam či vnímání důležitosti rozhodovacím subjektem každého z nich, nemusí být stejný. V AHP mohou být hodnoty jako cena, váha nebo plocha, nebo dokonce subjektivní názory, jako jsou pocity, preference nebo spokojenost, převedeny do měřitelných číselných hodnot.

Proto je třeba při hodnocení kritérií jim přikládat váhu, aby se zajistilo dosažení správného závěru a konzistentního řešení, což právě implementace a automatizace těchto výpočtů do systémů pro podporu rozhodování, významně usnadňuje.

AHP metoda reprezentuje rozhodovací problém pomocí hierarchického uspořádání, jak znázorňuje obrázek 9. Skládá se ze tří částí – Cíl nebo řešený problém, možná

řešení nazývaná alternativy a kritéria, podle kterých jsou jednotlivé alternativy posuzovány.

Analytický hierarchický proces zohledňuje důležitost kritérií prostřednictvím párových srovnání, kdy je určeno, které ze dvou kritérií je považováno za důležitější. Tato hodnocení na čísla, která lze porovnat se všemi možnými kritérii. Tato kvantifikační schopnost odlišuje AHP od jiných rozhodovacích technik. [25]



Obrázek 11 AHP hierarchie (zdroj: vlastní zpracování, inspirováno https://people.revoledu.com/kardi/tutorial/AHP/image/AHP-Example_clip_image002.jpg)

7.3. Metody vícekriteriálního rozhodování

Existuje široký výčet metod vícekriteriálního rozhodování, nicméně v této kapitole jsou představeny základní metody vícekriteriálního rozhodování a jejich princip, Těmi jsou bodovací metoda, Saatyho metoda a Fullerova metoda. [23]

- **Bodovací metoda**

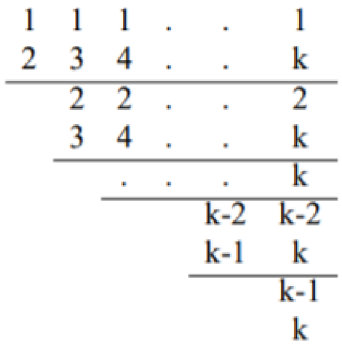
Tato metoda je jednou z nejjednodušších. Pro využití této metody je předpokladem možnost kvantifikovat důležitost jednotlivých kritérií. Princip této metody spočívá v přiřazení určitého počtu bodů ze zvolené stupnice na základě zhodnocení důležitosti daného kritéria vzhledem k hodnocené variantě. Čím je kritérium významnější, tím větší počet bodů je kritériu přiřazeno. Stupnice mohou být definovány různě, například desetibodová stupnice.
Na základě tohoto hodnocení je provedeno stanovení vah jednotlivých kritérií podílem počtu bodů daného kritéria a součtem všech bodů.
K dosažení výsledného rozhodnutí je třeba každé dílčí bodové hodnocení vynásobit vypočtenou vahou kritéria. Po sečtení hodnot pro každou variantu je dle nejvyšší této hodnoty stanovena optimální varianta. [23]

- **Fullerova metoda**

Fullerova metoda je metodou párového porovnávání. Zde se pro stanovení vah vyžívá informace, které ze dvou kritérií je významnější. Postupně jsou porovnávána každá dvě kritéria mezi sebou, počet srovnání je tedy následující:

$$N = \binom{k}{2} = \frac{k(k-1)}{2}$$

Stanovení preferencí se provádí na základě trojúhelníkové matice, tzv. Fullerova trojúhelníku (obrázek 10). Kritéria jsou očíslována pořadovými čísly 1, 2, …,k. Na základě toho je sestavena matice, jejíž dvojřádky se skládají z dvojic pořadových čísel uspořádaných tak, aby se každá dvojice vyskytovala právě jednou. [23]



Obrázek 12 Fullerův trojúhelník (zdroj: https://korviny.cz/Korviny/soubory/teorie_mca.pdf)

Následně je z každé dvojice označeno to kritérium, které je považováno za důležitější. Počet výskytů označení i-tého kritéria je označeno n_i. Váha je vypočtena pomocí tohoto vzorce [23]:

$$v = \frac{n_i}{N} \; ; \; i = 1,2, \dots k$$

- **Saatyho metoda**

Saatyho metoda je taktéž metodou párového porovnávání a v několika krocích lze dojít k výpočtu vah kritérií. Rozdíl oproti Fullerově metodě spočívá v tom, že se neurčuje pouze které kritérium je významnější než druhé, ale je také zohledněn stupeň významnosti. [23]

Nejprve je třeba stanovit matice intenzit preferencí S, které mají ve sloupcích i řádcích kritéria k = {K1, K2, …., K_x}, hlavní diagonála tedy obsahuje jedničky. Jednotlivé prvky, označovány jako s_{ij} nabývají reálných hodnot, které jsou stanoveny podle toho, kolikrát je varianta i významnější než varianta j v případě že platí že K_i je významnější, nebo stejně významné jako K_j. Podle toho, kolikrát je kritérium K_i významnější, než K_j jsou jednotlivým prvkům matice přiřazovány hodnoty 1-9, jejichž význam je vysvětlen v tabulce 1. Pokud ale platí, že K_j je významnější než K_i, prvky s_{ij} mají tento tvar [23]:

$$s_{ij} = \frac{1}{s_{ji}}$$

Matice má tedy následující tvar:

$$S = \begin{array}{c} \\ K_1 \\ K_2 \\ \vdots \\ K_i \end{array} \begin{array}{c} \begin{array}{cccc} K_1 & K_2 & \dots & K_j \end{array} \\ \begin{pmatrix} 1 & s_{12} & \cdots & s_{1j} \\ \frac{1}{s_{12}} & 1 & \dots & s_{2j} \\ \vdots & \vdots & \ddots & \vdots \\ s_{i1} & s_{i2} & \cdots & 1 \end{pmatrix} \end{array}$$

Tabulka 1 Tabulka preferencí (Zdroj: vlastní zpracování)

Stupeň významnosti	
1	Kritéria jsou stejně významná
3	První kritérium je slabě významnější než druhé
5	První kritérium je dosti významnější než druhé
7	První kritérium je prokazatelně významnější než druhé
9	První kritérium je absolutně významnější než druhé

Dalším krokem je výpočet normované váhy kritérií, který lze získat podílem geometrického průměru j-tého kritéria součtem geometrických průměrů všech kritérií. [23]

Geometrický průměr j-tého kritéria je dán tímto vztahem [23]:

$$G_j = \sqrt[m]{s_{j1} \times s_{j2} \times \dots \times s_{jm}}$$

Váha kritéria je dána tímto vztahem [23]:

$$v_j = \frac{g_j}{G_1 + G_2 + \cdots + G_m}$$

8. Systémy pro podporu rozhodování

8.1. Vymezení pojmu

Systémy pro podporu rozhodování jsou systémy, které slouží pro podporu ve všech fázích rozhodování, zejména o strukturovaných a polostrukturovaných rozhodovacích problémů v organizaci. Tyto problémy nastávají, když se osoby s rozhodovacími pravomocemi ocitnou v situaci, kdy nejsou známé všechny, pouze některé aspekty daného problému. [27]

Termín „Decision Support Systems“ byl poprvé použit v 70. letech 20. století, a byl představen jejich koncept. Průkopníky těchto systémů byli P.G.W. Keen a Scott Morton, kteří v roce 1978 společně vydali knihu „Decision Support Systems: An Organizational Perspective“, ve které systémy pro podporu rozhodování definují jako počítačové systémy mající vliv na rozhodování na základě dat a analytických nástrojů, přičemž je zde stále zásadní úsudek manažera. Systém pro podporu rozhodování může informace prezentovat graficky a také může zahrnovat expertní systém, potažmo umělou inteligenci. [27]

Systémy pro podporu rozhodování nachází své využití v řadě různých odvětví. Příklady některých z nich mohou být využití DSS při plánování nejrychlejší a nejlepší trasy mezi dvěma body analýzou dostupných možností, díky schopnosti monitorovat provoz v reálném čase. Nebo například v lékařství může DSS pomoci stanovit diagnózu. [9]

Se stále rostoucím pokrokem v oblasti výpočetní techniky se rodily další nové způsoby a prostředky podpory rozhodování za pomoci počítače a zároveň vznikaly různé definice a pohledy na DSS.

- Little (1970) definuje DSS jako modelový soubor procedur pro zpracování dat a manažerských úsudků za účelem asistence manažerovi při rozhodování. [9]
- Klasická definice Keena a Stotta Mortona definuje systémy pro podporu rozhodování jako spojení potenciálu lidského intelektu s možnostmi výpočetní techniky, za účelem zlepšení kvality manažerského rozhodnutí,

v podobě počítačového systému pro manažery zabývající se polostrukturovanými rozhodovacími problémy. [9]

- Podle Manna a Watsona (1984) – je DSS interaktivní systém, který uživateli poskytuje snadný přístup k rozhodovacím modelům a datům s cílem podpořit řešení polostrukturovaných a nestrukturovaných rozhodovacích problémů. [9]
- Bidgoli (1989) definuje DSS jako počítačový informační systém skládající se ze softwarové části, hardwarové části a lidského prvku, sloužící manažerům na různých úrovních vedení a je zaměřen na polostrukturované a strukturované rozhodovací problémy. [9]
- Sprague, Watson (1996) - počítačový systém, který pomáhá osobám s rozhodovací pravomocí čelit strukturovaným problémům prostřednictvím přímé interakce s daty a analytickým modelem. [9]

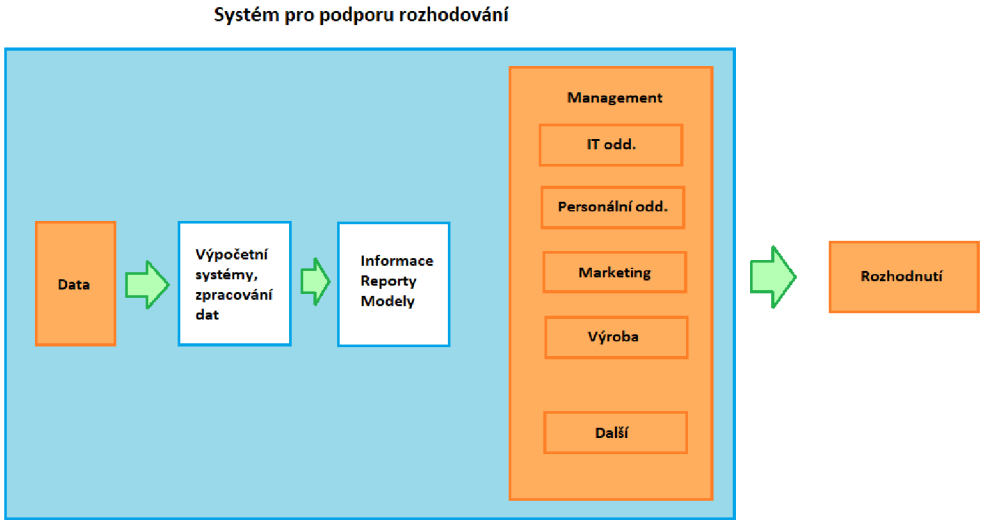
8.2. Obecná architektura DSS

Systémy pro podporu managementu se vždy skládají ze tří klíčových komponent – Databáze, softwarový systém a uživatelské rozhraní. [27]

Pro vyřešení jakéhokoli problému jsou vždy nezbytná data, která jsou tedy jednou ze základních komponent DSS. Tyto data jsou dále zpracovávána další z komponent – softwarovým systémem, který je založen na modelu. Počet a typy modelů závisí na účelu DSS. Mez běžně používané modely patří [27]:

- Statistický model – model využívaný k určení vztahů mezi událostmi a faktory související s touto událostí
- Model analýzy citlivosti – model využívaný pro analýzu „Co kdyby“
- Model optimalizační analýzy – model využívaný k nalezení optimální hodnoty pro cílovou proměnnou ve vztahu k ostatním proměnným.
- Model pro určení prognózy – zahrnuje regresní modely, analýzu časových řad a další modely využívané k analýze dat a vytváření prognóz
- Model zpětné citlivostní analýzy – podstata tohoto modelu spočívá v hledání

Samozřejmostí je uživatelské rozhraní, přes které do systému vstupuje uživatel. Souvislost těchto komponent je znázorněna na obrázku 13.



Obrázek 13 Obecná architektura systému pro podporu rozhodování (zdroj: vlastní zpracování, inspirováno http://2.bp.blogspot.com/-3isuMDcKpxI/Vmu1pQuAYdI/AAAAAAAAAGQ/xbkXKV-AYZk/s1600/decision%2Bsupport%2Bsystem.jpg)

Výzkumná část

9. Specifikace modelové situace

Východiskem pro následné stanovení kritérií pro výběr automatizačního nástroje, a výběr konkrétních nástrojů k porovnání, je definování výchozí modelové situace a jejích požadavků na testovací nástroj. Tato specifikace je zaměřena zejména na aspekty, které s výběrem nástroje přímo souvisejí. V této kapitule je naplněn dílčí cíl č.1.

9.1. Subjekt rozhodování – modelová organizace

Modelovou organizací pro tuto diplomovou práci je softwarová firma XY, poskytující softwarové řešení v oblasti energetiky. Modelová organizace vyvíjí webovou aplikaci, kterou je registr zelených certifikátů a je určený pro obchodování s těmito certifikáty. Toto softwarové řešení je vyvíjeno jako produkt, založený na společném zdrojovém kódu pro všechny verze, které se pak od sebe dále liší v parametrizaci a konfiguraci. Aplikace je poskytována do zemí Švédska, Norska a Chorvatska. Jednotlivé organizace jsou v roli zákazníka a zadavatele funkčních požadavků.

9.2. Metodika vývoje

Vzhledem k rozsahu projektu a velikosti vývojového týmu je využívána agilní metodika vývoje, od které se dále odvíjí procesy a pracovní postupy a organizační struktura organizace. Vývoj tedy probíhá iterativně a produkt je v jednotlivých iteracích postupně rozšiřován o nové funkcionality, případně stávající funkcionality jsou vylepšovány, nebo jsou řešeny případné nastalé problémy. Vzhledem k tomuto faktu je nutné zahrnutí automatizovaných testů do procesu kontinuální integrace.

9.3. Úrovně testování a testovací strategie

Režim vývoje je tedy stanovený na dvoutýdenní sprint. Pro každý sprint (verzi) je vytvořen nový projekt v nástroji JIRA, ve kterém jsou ve spolupráci se zákazníkem stanoveny požadavky, které mají být v rámci daného sprintu realizovány. Následně jsou zde evidovány také nalezené defekty. V rámci tohoto sprintu tedy probíhá i první úroveň testování, kterou jsou jednotkové (unit) testy prováděny vývojáři.

První den následujícího sprintu je verze určena k vydání do produkce nasazena na testovací prostředí, čímž je zahájena další úroveň testování – FAT (Factory Acceptance Tests) interním týmem pro zajištění kvality na straně poskytovatele produktu. Tento tým se skládá z celkem dvanácti testerů, přičemž pro každou zemi (zákazníka) jsou alokováni čtyři testeři. Test architekt, který je zodpovědný za organizaci a koordinaci testovacích aktivit, úzce spolupracuje s projektovým manažerem a testovacím týmem na straně zákazníka. V rámci této úrovně – FAT testů (Factory Acceptance Tests) jsou prováděny integrační, funkční, výkonnostní a zátěžové testy. Tyto testy jsou prováděny manuálně – manuální tester zodpovědný za testy uživatelského rozhraní, o jehož automatizaci se prozatím neuvažuje, z důvodu jeho častých změn. Back-endová část aplikace je testována automatizovaně. Tato část je zodpovědností dvou automatizačních testerů, kteří mají na starosti tvorbu, údržbu a spouštění automatizovaných skriptů. Na úvod je proveden takzvaný smoke test, který ověří základní funkce, a že je aplikace testovatelná jako celek. Dále následují regresní testy související se změnami v aplikaci, přičemž nalezené defekty jsou dále paralelně s vývojem opravovány v pořadí dle stanovených priorit. Následně jsou prováděny průzkumné neboli exploratory testy, jejichž cílem je důkladné otestování aplikace a nalezení dalších chyb.

Po dokončení této úrovně jsou vytvořeny detailní reporty o stavu verze, průběhu a výsledcích testů. Otestovaná verze aplikace v rámci FAT je odeslána zákazníkovi k uživatelskému akceptačnímu testování (UAT) provedených změn. Pro každou zemi je na straně zákazníka právě jeden tester.

9.4. Nástroje využívané na projektu

Již zmíněný nástroj JIRA slouží k řízení projektu, vývojových i testovacích aktivit, správě chyb a defektů, a je integrován s nástrojem PractiTest pro správu testovacích scénářů. Výhodou této integrace automatizovaných testů spočívá v reportování výsledků a informací o průběhu testů pomocí API do těchto nástrojů, kde jsou pak automaticky označeny stavy jednotlivých testovacích případů, což lze dále využít k vytváření test reportů a také je možnost sledovat pokrytí funkcionalit automatizovanými testy.
Pro správu verzí kódu aplikace je využíván nástroj Gitlab, proto pro zavedení procesu integrace správy verzí kódu, automatického sestavení a nasazení verzí na prostředí je nutné zavést nástroje pro CI/CD, které je možné s nástrojem Gitlab integrovat.

Ostatní technické parametry aplikace:

- Jazyk, v němž je aplikace napsána – Java
- Platforma vyvíjené aplikace – webová aplikace
- Operační systémy, na kterých běží vyvíjená aplikace – Windows, Linux
- Využité typy API – SOAP, HTTPS, FTP

10. Stanovení kritérií pro výběr automatizačního nástroje

V této kapitole je naplněn dílčí cíl č.2.

Při výběru správného nástroje je důležité si nejprve stanovit rozhodovací kritéria. Na základě důkladné analýzy modelové situace a z toho plynoucích požadavků na automatizační nástroj, je v této kapitole stanoveno deset kritérií, na kterých bude založen výběr vhodného automatizačního nástroje pro tuto modelovou organizaci. Jedná se zejména o podporované technologie a kompatibilita nástroje s vyvíjenou aplikací, možnost integrací s dalšími nástroji, typy testů, které je možné pomocí nástroje provádět, možnosti reportování výsledků, velikost a složení týmu pro zajištění kvality softwaru, s čímž přímo souvisí náklady na pořízení nástroje.

Správné stanovení výběrových kritérií však může být poměrně náročný úkol, existuje riziko, že některá důležitá kritéria mohou být opomenuta. Dílčím cílem této práce je vytvořit znovupoužitelný rozhodovací model pro podporu rozhodování o výběru vhodného automatizačního nástroje. Každý projekt má však svá specifika, různé potřeby a nároky na automatizaci, a pro každý projekt tak může být vhodný jiný nástroj. Z tohoto důvodu se výběr kritérií v této diplomové práci zakládá nejen na analýze samotné modelové situace, ale bude doplněn o další kritéria na zákadě analýzy zdrojů zabývající se kritérii pro výběr vhodného automatizačního nástroje.

[2], [8], [21], [35], [36],

Na základě výše uvedených zdrojů jsou pro účely této práce brány v potaz tato další kritéria:

- Správa testovacích dat
- Kontrola obsahu odpovědí
- Udržovatelnost testovacích skriptů
- Podporované skriptovací jazyky
- Programovací jazyk vyvíjené aplikace
- Uživatelská podpora nástroje

Na základě výše zmíněného byla stanovena kritéria, jejichž výčet je uveden v kapitole 10.1. Pro každé kritérium byly stanoveny hodnoty, kterých dané

kritérium může nabývat a na základě toho byly stanoveny dílčí alternativy, jako přípustné kombinace jednotlivých hodnot. Alternativy byly stanoveny tak, aby bylo možné je seřadit dle priorit – od nejvyšší po nejnižší, které se mohou lišit na základě požadavků pro konkrétní situaci. Poté byly Saatyho metodou vypočteny váhy jednotlivých kritérií.

Tato kritéria a jim stanovené váhy dále slouží jako součást podkladu pro vytvoření softwarového řešení – rozhodovacího modelu pro podporu rozhodování o výběru vhodného automatizačního nástroje.

10.1. Kritéria

K1 Cena nástroje

- Toto kritérium definuje požadavky týkající se vynaložení nákladů na licenci, předpokládá se zde využití v rámci firmy, nikoli jednotlivce.
- Může záviset na počtu uživatelů.
- Alternativy:
  - **A1** Nástroj je bezplatný – open-source
  - **A2** Nástroj je bezplatný, s možností placeného rozšíření
  - **A3** Komerční nástroj je placený – roční náklady na licenci nepřesahují 20 000 Kč
  - **A4** Nástroj je placený – roční náklady na licenci přesahují 20 000 Kč

K2 Kompatibilita nástroje s vyvíjenou aplikací, programovací jazyk vyvíjené aplikace

- Je nutné, aby nástroj pro automatizaci podporoval a byl schopen pracovat s technologií, na které je založena vyvíjená aplikace.
- mnoho nástrojů podporuje pouze určitý programovací jazyk vyvíjené aplikace. Například Abbot a Maveryx se používají pro testování aplikací Java. Mnoho dalších nástrojů, jako je TestComplete a Ranorex, může pracovat s libovolnými jazyky za předpokladu, že může běžet na podporovaných platformách.
- Alternativy:
  - **A1** Nástroj je kompatibilní s vyvíjenou aplikací
  - **A2** Nástroj není kompatibilní s vyvíjenou aplikací

K3 Podpora operačních systémů

- Operační systémy, které nástroj podporuje. Nástroj vždy musí podporovat verzi OS, na kterém aplikace běží.
- Hodnoty, kterých kritérium může nabývat:
  - **A2** Windows
  - **A3** Mac
  - **A4** Linux
- Alternativy:
  - **A1** Všechny uvedené možnosti
  - **A2** Windows
  - **A3** Mac
  - **A4** Linux
  - **A5** Windows, Mac
  - **A6** Windows, Linux
  - **A7** Mac, Linux

K4 Podpora platforem

- Toto kritérium definuje platformy, které lze pomocí nástroje testovat. Možnost webových aplikací je vzhledem k tématu diplomové práce obsažen ve všech možnostech.
- Hodnoty, kterých kritérium může nabývat:
  - Desktopové aplikace
  - Webové aplikace
  - Mobilní aplikace
  - Nástrojem lze testovat jakoukoliv platformu
- Alternativy:
  - **A1** Všechny uvedené možnosti
  - **A2** Webové aplikace, Mobilní aplikace
  - **A3** Webové aplikace, Desktopové aplikace
  - **A4** Pouze webové aplikace

K5 Podporované typy protokolů

- Podporované typy požadavků, které vyžaduje vyvíjená aplikace. Typy API, které nástroj může testovat. Možnost webových služeb je vzhledem ke kritériím výběru nástrojů je obsažen ve všech možnostech.
- Hodnoty, kterých kritérium může nabývat:
  - Webové služby (REST, SOAP, HTTP a další)
  - Poštovní služby (POP, IMAP, SMTP a další)
  - Přenos souborů (FTP, FTPS, SFTP a další)
  - Přístupy k databázi (JDBC a další)
- Alternativy:
  - **A1** Všechny zmíněné typy
  - **A2** Webové a poštovní služby
  - **A3** Webové služby, Přenos souborů
  - **A4** Webové služby, Přístupy k databázi
  - **A5** Webové služby, Přenos souborů, Poštovní služby
  - **A6** Webové služby, Přístupy k databázi, Poštovní služby
  - **A7** Webové služby, Přístupy k databázi, Přenos souborů
  - **A8** Pouze Webové služby

K6 Typy testů

- Možnost využití nástroje pro další typy testů nad rámec funkčních testů.
- Hodnoty, kterých kritérium může nabývat:
  - Funkční testy
  - Zátěžové testy
  - Výkonnostní testy
  - GUI testy
- Alternativy:
  - **A1** Všechny zmíněné typy testů
  - **A2** Funkční, Zátěžové, Výkonnostní
  - **A3** Funkční, Zátěžové, GUI
  - **A4** Funkční, Výkonnostní, GUI
  - **A5** Funkční, Zátěžové
  - **A6** Funkční, Výkonnostní
  - **A7** Funkční, GUI
  - **A8** Pouze funkční

K7 Nutnost znalosti programování

- Některé nástroje vyžadují pro tvorbu skriptů znalost programovacího jazyka, jiné umožňují méně zkušenému uživateli vytvořit skript metodami, které znalost programování nevyžadují.
- Alternativy:
  - **A1** Nástroj mohou využívat i uživatelé bez znalosti programování
  - **A2** Nástroj mohou využívat uživatelé se základní znalostí programování
  - **A3** Nástroj umožnuje pouze základní možnosti automatizace bez znalosti programování
  - **A4** Nástroj neumožňuje využití pro uživatelé bez znalosti programování

K8 Vyžadovaná míra zkušenosti testera

- Míra znalosti, jakou vyžaduje použití nástroje.
- Alternativy:
  - **A1** Tester junior – bez předchozí zkušenosti s testováním
  - **A2** Mírně pokročilý – má předchozí zkušenost, ale nedisponuje aktivní znalostí programování
  - **A3** Středně pokročilý – má předchozí zkušenost, disponuje základní znalostí programování
  - **A4** Tester senior – zkušený tester, disponuje aktivní znalostí programování

K9 Možnost integrace s dalšími nástroji a systémy

- Toto kritérium definuje, zda je nástroj možný integrovat s těmito nástroji a systémy.
- Hodnoty, kterých kritérium může nabývat:
  - Nástroje pro průběžnou integraci
  - Nástroje pro test management
  - Nástroje pro správu verzí
- Alternativy:
  - **A1** Nástroj umožňuje integraci se všemi zmíněnými
  - **A2** Nástroje pro průběžnou integraci a test management
  - **A3** Nástroje pro průběžnou integraci a správu verzí
  - **A4** Nástroj neumožňuje integraci s žádným z těchto nástrojů

K10 Možnosti reportování výsledků

- Toto kritérium definuje možnosti definování reportů výsledků testů a logování informací o jednotlivých krocích. Pro každý testovaný krok by měl být jasně vidět výsledek a informace o kvalitě, což dále umožňuje analyzovat příčinu případného negativního výsledku.
- Alternativy:
  - **A1** Nástroj disponuje textovými i grafickými reporty, s možností exportování do různých formátů a customizace reportů
  - **A2** Nástroj disponuje textovými i grafickými reporty
  - **A3** Nástroj disponuje pouze možnostmi základního textového reportování

K11 Správa testovacích dat, možnost parametrizace testů

- Možnost importu testovacích dat z externích souborů, jako je CSV, soubor aplikace Excel, soubor XML, JSON, nebo databáze, z čehož plyne možnost testovat různé varianty jednoho testovacího scénáře, které se liší vstupními daty.
- Alternativy:
  - **A1** Ano – nástroj podporuje import externích dat v požadovaných formátech
  - **A2** Ne – nástroj nepodporuje import externích dat

K12 Kontrola obsahu odpovědí

- Nástroj disponuje funkcí aserce odpovědí.
- Alternativy:
  - **A1** Ano – nástroj disponuje možností aserce odpovědí
  - **A2** Ne – nástroj nedisponuje možností aserce odpovědí

K13 Udržovatelnost testovacích skriptů

- Údržba testovacích skriptů je časově velmi náročná činnost u projektů, kde jsou požadavky často měněny, což je specifikum zejména agilních projektů. Některé nástroje dokážou částečně tyto změny ve skriptech provádět automaticky při změně v kódu softwaru.
- Alternativy:
  - **A1** Ano – nástroj umožňuje automatickou aktualizaci skriptů
  - **A2** Ne – nástroj neumožňuje automatickou aktualizaci skriptů

K14 Podporované skriptovací jazyky

- Toto kritérium hodnotí flexibilitu možností skriptování. Nástroj, který podporuje různé skriptovací jazyky, umožňuje lepší přizpůsobení testovacím týmům, které tak mohou psát testovací skripty v jazycích, jehož znalostí disponují.
- Alternativy:
  - **A1** Nástroj disponuje možností skriptování v různých jazycích
  - **A2** Nástroj podporuje pouze nativní skriptovací jazyk daného nástroje

K15 Podpora nástroje

- Dostupnost dokumentace, tutoriálů, školení, telefonická či emailová podpora společnosti, uživatelská fóra
- Kritérium může být významné pro méně zkušené testery
- Alternativy:
  - **A1** Uživatelská fóra, články autora, online dokumentace, online materiály
  - **A2** Telefonická/emailová podpora, možnost školení a webinářů
  - **A3** Kombinace obou možností

10.2. Výpočet vah dílčích alternativ

Po stanovení kritérií následuje výpočet jejich vah. Nejdříve byly vypočteny váhy jednotlivých dílčích alternativ, které budou následně aplikovány na jednotlivá kritéria. Tyto dílčí výpočty byly provedeny Saatyho párovou porovnávací metodou, pomocí MS Excel.

10.2.1. Postup výpočtu

- Nejdříve je třeba seřadit stanovené alternativy dle priorit (nejvhodnější možnosti pro danou modelovou situaci). Viz tabulka č.2.

Tabulka 2 Seřazení dle preferencí dílčích alternativ (Zdroj: vlastní zpracování)

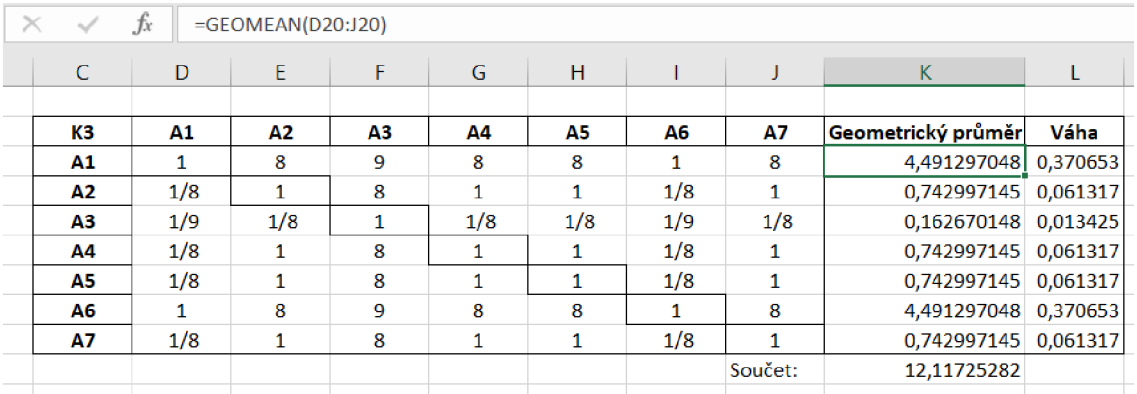
Kritérium	Pořadí dle priority	Alternativa		Kritérium	Pořadí dle priority	Alternativa
K1	1.	A1		**K7**	1.	A1
	2.	A2			2.	A2
	3.	A3			3.	A3
	4.	A4			4.	A4
K2	1.	A1		**K8**	1.	A4
	2.	A2			2.	A3
K3	1.	A1			3.	A2
	1.	A6			4.	A1
	2.	A2		**K9**	1.	A1
	2.	A4			2.	A2
	2.	A5			2.	A3
	2.	A7			3.	A4
	3.	A3		**K10**	1.	A1
K4	1.	A1			2.	A2
	1.	A2			3.	A3
	1.	A3		**K11**	1.	A1
	1.	A4			2.	A2
K5	1.	A1		**K12**	1.	A1
	1.	A7			2.	A2
	2.	A3		**K13**	1.	A1
	2.	A4			2.	A2
	2.	A5		**K14**	1.	A1
	2.	A6			2.	A2
	3.	A2		**K15**	1.	A3
	3.	A8			2.	A2
K6	1.	A1			3.	A1
	1.	A2				
	2.	A3				
	2.	A4				
	2.	A5				
	2.	A6				
	3.	A7				
	3.	A8				

- V MS Excel jsou vytvořeny matice pro jednotlivá kritéria, kde se počet sloupců a řádků rovná počtu alternativ. Na diagonále tabulky se zapíšou jedničky. Do zbylých buněk jsou zapisovány hodnoty dle Saatyho matice (tabulka 1). Pokud je preferován prvek ve sloupci nad prvkem v řádku, je zapsána převrácená hodnota. Viz. obrázek 14.

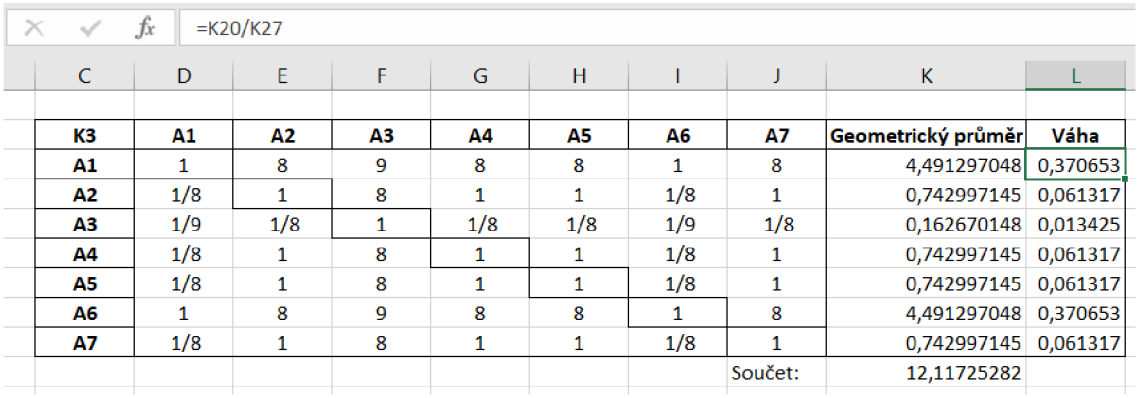
K3	A1	A2	A3	A4	A5	A6	A7
A1	1	8	9	8	8	1	8
A2	1/8	1	8	1	1	1/8	1
A3	1/9	1/8	1	1/8	1/8	1/9	1/8
A4	1/8	1	8	1	1	1/8	1
A5	1/8	1	8	1	1	1/8	1
A6	1	8	9	8	8	1	8
A7	1/8	1	8	1	1	1/8	1

Obrázek 14 Saatyho matice (Zdroj: vlastní zpracování)

- Dále jsou přidány další dva sloupce pro výpočet geometrického průměru, jehož výpočet je znázorněn na obrázku 15. Z jejich celkových součtů jsou následně vypočteny váhy, jejichž výpočet je znázorněn na obrázku 16. Takto byl proveden výpočet pro všechna kritéria. Hodnoty vah dílčích alternativ zaokrouhlené na dvě desetinná místa, jsou zaznamenány v tabulce 3. Kompletní výpočty viz. příloha.



Obrázek 15 Výpočet geometrického průměru (Zdroj: vlastní zpracování)



	C	D	E	F	G	H	I	J	K	L
	K3	**A1**	**A2**	**A3**	**A4**	**A5**	**A6**	**A7**	**Geometrický průměr**	**Váha**
	A1	1	8	9	8	8	1	8	4,491297048	0,370653
	A2	1/8	1	8	1	1	1/8	1	0,742997145	0,061317
	A3	1/9	1/8	1	1/8	1/8	1/9	1/8	0,162670148	0,013425
	A4	1/8	1	8	1	1	1/8	1	0,742997145	0,061317
	A5	1/8	1	8	1	1	1/8	1	0,742997145	0,061317
	A6	1	8	9	8	8	1	8	4,491297048	0,370653
	A7	1/8	1	8	1	1	1/8	1	0,742997145	0,061317
								Součet:	12,11725282	

Obrázek 16 Výpočet vah dílčích kritérií (Zdroj: vlastní zpracování)

Tabulka 3 Vypočtené váhy dílčích alternativ (Zdroj: vlastní zpracování)

Kritérium	Alternativa	Váha		Kritérium		Váha
K1	A1	0,6		**K7**	A1	0,43
	A2	0,26			A2	0,1
	A3	0,11			A3	0,16
	A4	0,03			A4	0,06
K2	A1	0,9		**K8**	A1	0,04
	A2	0,1			A2	0,1
K3	A1	0,37			A3	0,29
	A2	0,06			A4	0,57
	A3	0.01		**K9**	A1	0,69
	A4	0,06			A2	0,14
	A5	0,06			A3	0,14
	A6	0,37			A4	0,03
	A7	0,06		**K10**	A1	0,69
K4	A1	0,25			A2	0,25
	A2	0,25			A3	0,05
	A3	0,25		**K11**	A1	0,9
	A4	0,25			A2	0,1
K5	A1	0,34		**K12**	A1	0,9
	A2	0,01			A2	0,1
	A3	0,07		**K13**	A1	0,8
	A4	0,07			A2	0,2
	A5	0,07		**K14**	A1	0,9
	A6	0,07			A2	0,1
	A7	0,34		**K15**	A1	0,06
	A8	0,01			A2	0,21
K6	A1	0,32			A3	0,74
	A2	0,32				
	A3	0,08				
	A4	0,08				
	A5	0,08				
	A6	0,08				
	A7	0,02				
	A8	0,02				

11. Výběr nástrojů pro automatizaci testování API webových aplikací

V této kapitole byl proveden výběr nástrojů pro automatizaci testů aplikačního programového rozhraní webových aplikací a byly vybrány čtyři nástroje, které svým rozšířením zaujímají významnou část trhu. V následující kapitole bude zhodnoceno naplnění jednotlivých kritérií pro jednotlivé nástroje, což bude dále sloužit jako podklad pro řešení rozhodovacího problému pomocí vybraného softwarového nástroje.
[16], [22], [41]

V současnosti je na trhu velké množství nástrojů pro automatizaci testování, přičemž jeho správný výběr může hrát významnou roli v konečném úspěchu procesu zavádění automatizace testů do projektu.

Poté, co jsou známy požadavky projektu na automatizační nástroj, je definován rozsah automatizace a jsou stanovena výběrová kritéria, přichází na řadu samotný výběr vhodného automatizačního nástroje.

Vybrané nástroje:

- Jmeter
- PostMan
- SoapUI
- Katalon Studio

Vyhodnocení kritérií – JMeter

Tabulka 4 Vyhodnocení kritérií – JMeter [3]

Kritérium	Alternativa	Váha	Komentář
K1	A1	0,6	Nástroj je bezplatný – open-source
K2	A1	0,9	Nástroj podporuje jazyk Java
K3	A1	0,37	
K4	A1	0,25	
K5	A1	0,34	Webové služby: REST, SOAP, HTTP, HTTPS Databázové servery: JDBC, LDAP, Poštovní služby: POP3, IMAP, SMTP Přenos souborů: FTP
K6	A1	0,32	
K7	A2	0,1	
K8	A3	0,29	
K9	A1	0,69	
K10	A1	0,69	Textové/grafické reporty Informace o odeslaném požadavku/odpovědi Chyby, upozornění Graf, tabulka, strom Export
K11	A1	0,9	CSV, JSON, XML
K12	A1	0,9	
K13	A2	0,2	
K14	A1	0,9	Podporované jazyky: Java, JavaScript, Scala, Beanshell, Groovy , PHP, ASP.NET
K15	A1	0,06	

Vyhodnocení kritérií – PostMan

Tabulka 5 Vyhodnocení kritérií – PostMan [29]

Kritérium	Alternativa	Váha	Komentář
K1	A3	0,11	Cena licence Enterprise - 99$/uživatel/měsíc 6 uživatelů – celkové náklady – 162 000 Kč/rok
K2	A1	0,9	Nástroj podporuje jazyk Java
K3	A1	0,37	
K4	A1	0,25	
K5	A8	0,01	Webové služby: HTTP, REST, SOAP
K6	A8	0,02	
K7	A2	0,1	
K8	A2	0,1	
K9	A1	0,69	BitBucket, GitHub, GitLab, Dropbox, Slack, Microsoft Teams či APIMatic, JIRA,
K10	A1	0,69	Textové/grafické reporty Generování reportů v různých formátech Možnost vytvoření vlastní šablony reportu Informace o odeslaném požadavku/odpovědi
K11	A1	0,9	CSV, JSON
K12	A1	0,9	
K13	A2	0,2	
K14	A1	0,9	
K15	A3	0,74	

Vyhodnocení kritérií – SoapUI

Tabulka 6 Vyhodnocení kritérií – SoapUI [33]

Kritérium	Alternativa	Váha	Komentář
K1	A1	0,6	Cena licence – 43 000 Kč/rok/uživatel 6 uživatelů – 257 000 Kč/rok
K2	A1	0,9	
K3	A1	0,37	
K4	A2	0,25	
K5	A4	0,07	Webové služby: http, HTTPS, SOAP/WSDL, REST, Databázové služby: JDBC
K6	A2	0,32	
K7	A3	0,16	Pro pokročilou automatizaci je nutná znalost spritovacího jazyka Groovy
K8	A3	0,29	
K9	A1	0,69	
K10	A2	0,25	
K11	A1	0,9	Zdroje dat: csv, xml soubory, databáze
K12	A1	0,9	
K13	A2	0,2	
K14	A2	0,1	
K15	A3	0,74	

Vyhodnocení kritérií – Katalon

Tabulka 7 Vyhodnocení kritérií – Katalon [20]

Kritérium	Alternativa	Váha	Komentář
K1	A1	0,6	Cena licence – 17 000/rok
K2	A1	0,9	
K3	A1	0,37	
K4	A1	0,25	
K5	A8	0,01	Rest/SOAP
K6	A1	0,32	
K7	A1	0,43	
K8	A1	0,04	
K9	A1	0,69	
K10	A1	0,69	
K11	A1	0,9	
K12	A1	0,9	
K13	A1	0,8	
K14	A2	0,1	
K15	A3	0,74	

12. Softwarové řešení rozhodovacího problému

12.1. Volba nástroje pro podporu rozhodování

Pro softwarové řešení rozhodovacího problému byl zvolen nástroj Criterium Decision plus, který svými funkcionalitami a využívanými metodami výpočtů splňuje požadavky pro realizaci hlavního cíle této práce.

Tento systém je vyvíjen společností InfoHarvest Inc.,a je dostupný na virtuálních učebnách UHK.

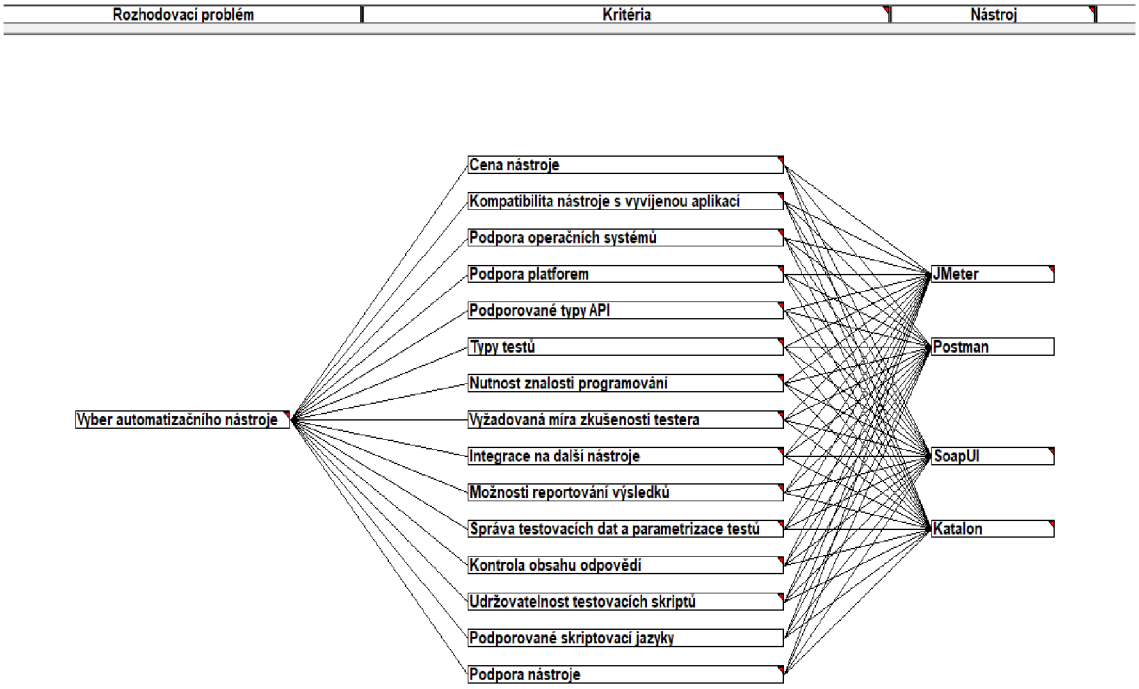
V tomto nástroji jsou implementovány dvě metody – Analytic Hierarchy Process (AHP) a Simple Multi-Attribute Rating Technique (SMART).

Byla využita metoda AHP, která je vhodná v případech, kdy je k dispozici konečný počet alternativ.

12.2. Návrh systému

V této kapitole je sestaven rozhodovací model, jehož hierarchie se skládá z rozhodovacího problému, stanovených kritérií a vybranými alternativami řešení, viz. obrázek 17.

Je zde naplněn dílčí cíl č. 4, kterým je vytvoření rozhodovacího modelu pro podporu rozhodnutí o vhodném nástroji pro automatizaci testů API pro konkrétní modelovou organizaci.



Obrázek 17 CDP – Hierarchie (Zdroj: vlastní zpracování)

12.3. Využití rozhodovacího systému v modelové situaci

Proces výběru vhodného řešení je proveden v těchto následujících krocích:

- Definice problému
- Stanovení kritérií pro výběr vhodného nástroje a jejich alternativ
- Výběr alternativ řešení
- Výpočet vah jednotlivých alternativ vybraných kritérií
- Sestavení rozhodovacího modelu
- Provedení výpočtu pomocí Criterium Decision Plus
- Zhodnocení výsledků

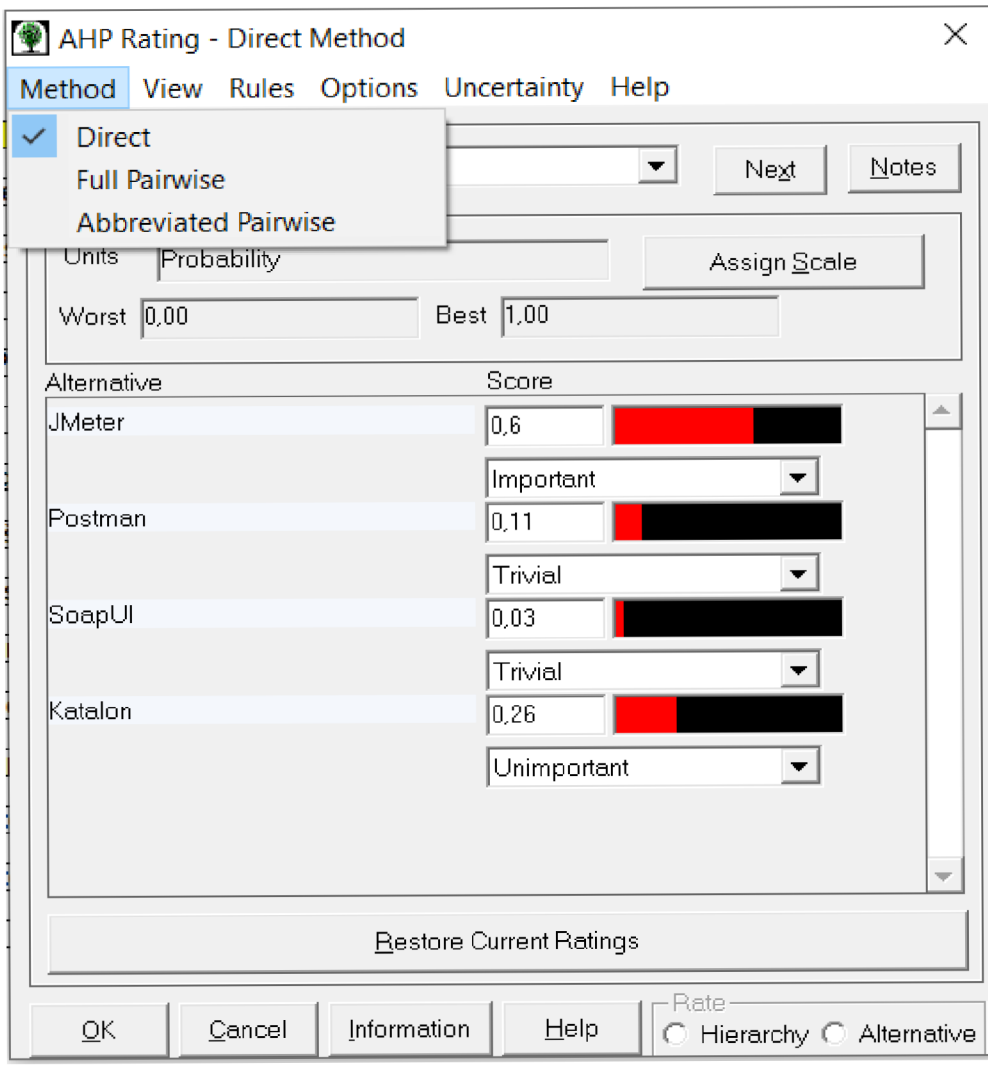
Pro další využití tohoto rozhodovacího modelu v praxi je možné tento model modifikovat, přizpůsobit kritéria a možné alternativy dané situaci.

Pro hodnocení jednotlivých alternativ z hlediska stanovených kritérií byla využita přímá metoda, kdy byly aplikovány vypočtené váhy viz. tabulky č.3. Škála byla stanovena na 0-1, kde nula představuje nejnižší váhu. (obrázek 18). Pro stanovení vah jednotlivých kritérií byla využita opět Saatyho metoda párového porovnávání, kdy se jednotlivá kritéria porovnávají mezi sebou z hlediska preference daného kritéria, na základě seřazení kritérií od nejdůležitějšího po nejméně důležitého pro danou modelovou situaci. Škála pro hodnocení kritérií byla stanovena na 1-9, jejichž význam je vysvětlen v kapitole Saatyho metoda (Tabulka 1, Obrázek 20).

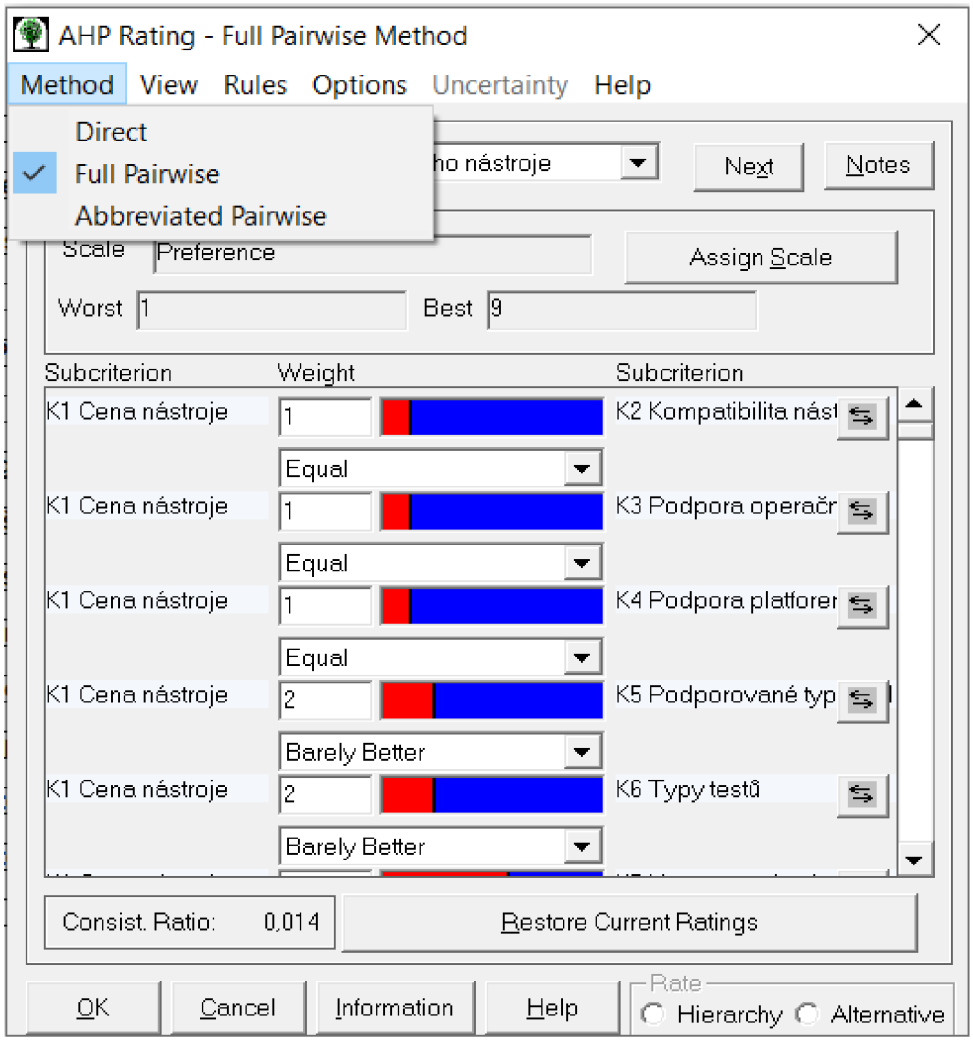
Pro stanovení důležitosti jednotlivých kritérií byla vytvořena matice v programu MS excel (obrázek 18), kde je rovněž zohledněn stupeň preference jednotlivých kritérií. Na základě které byly porovnávány kritéria v rozhodovacím modelu, a byla tak zajištěna konzistence hodnocení jednotlivých kritérií.

	K1	K2	K3	K4	K5	K6	K9	K10	K11	K12	K7	K8	K13	K14	K15
K1	1	1	1	2	2	2	3	4	4	5	5	5	6	6	6
K2	1	1	1	2	2	2	3	4	4	5	5	5	6	6	6
K3	1	1	1	2	2	2	3	4	4	5	5	5	6	6	6
K4	1/2	1/2	1/2	1	1	1	2	3	3	4	4	4	5	5	5
K5	1/2	1/2	1/2	1	1	1	2	3	3	4	4	4	5	5	5
K6	1/2	1/2	1/2	1	1	1	2	3	3	4	4	4	5	5	5
K9	1/3	1/3	1/3	1/2	1/2	1/2	1	2	2	3	3	3	4	4	4
K10	1/4	1/4	1/4	1/3	1/3	1/3	1/2	1	1	2	2	2	3	3	3
K11	1/4	1/4	1/4	1/3	1/3	1/3	1/2	1	1	2	2	2	3	3	3
K12	1/5	1/5	1/5	1/4	1/4	1/4	1/3	1/2	1/2	1	1	1	2	2	2
K7	1/5	1/5	1/5	1/4	1/4	1/4	1/3	1/2	1/2	1	1	1	2	2	2
K8	1/5	1/5	1/5	1/4	1/4	1/4	1/3	1/2	1/2	1	1	1	2	2	2
K13	1/6	1/6	1/6	1/5	1/5	1/5	1/4	1/3	1/3	1/2	1/2	1/2	1	1	1
K14	1/6	1/6	1/6	1/5	1/5	1/5	1/4	1/3	1/3	1/2	1/2	1/2	1	1	1
K15	1/6	1/6	1/6	1/5	1/5	1/5	1/4	1/3	1/3	1/2	1/2	1/2	1	1	1

Obrázek 18 Matice – stanovení priorit kritérií (Zdroj: vlastní zpracování)



Obrázek 19 Hodnocení nástrojů (Zdroj: vlastní zpracování)



Obrázek 20 Hodnocení kritérií v systému Criterium Decision Plus (Zdroj: vlastní zpracování)

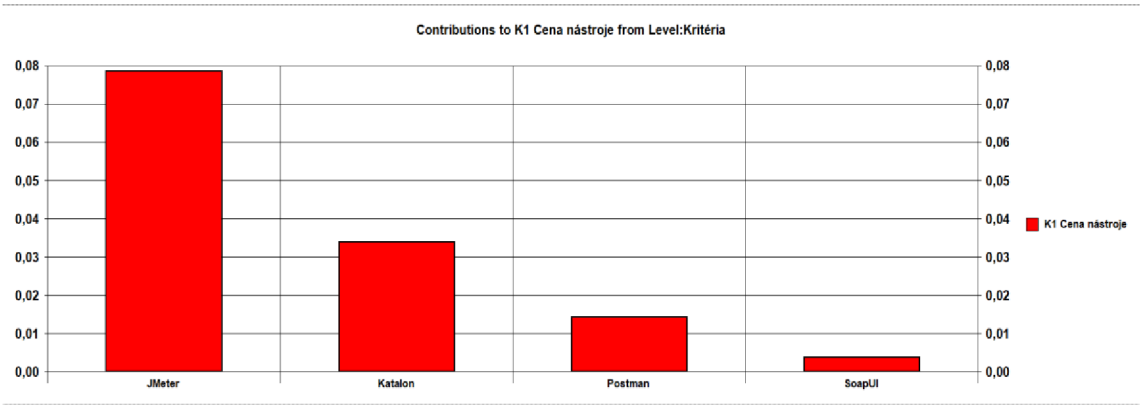
13. Shrnutí výsledků

V této kapitole je zhodnocen výsledek výběru vhodného automatizačního nástroje, a tím je naplněn dílčí cíl č. 5.

Na základě hodnocení a stanovených vah, poskytuje nástroj Criterium Decision Plus jednoznačný výsledek.

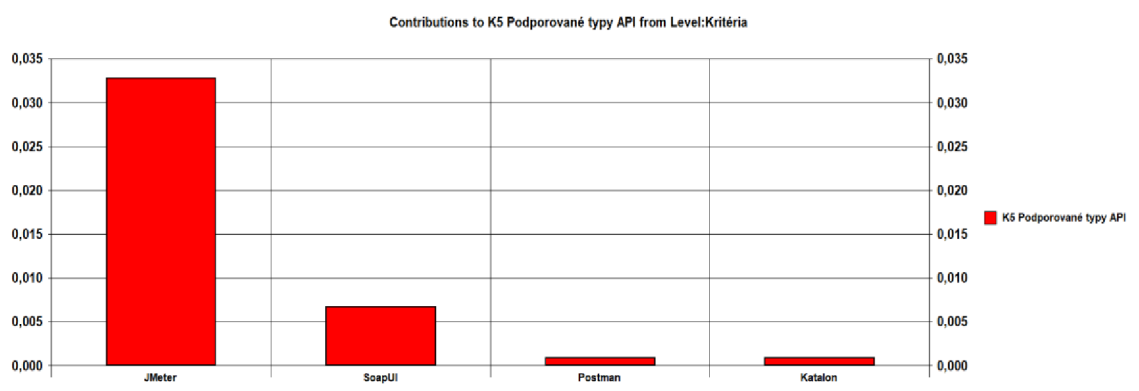
Z dílčích výsledných grafů jednotlivých kritérií lze vidět, že nejpodstatnější vliv na konečné rozhodnutí měla tato kritéria, ve kterých se vybrané nástroje nejvíce liší:

- K1 Cena nástroje



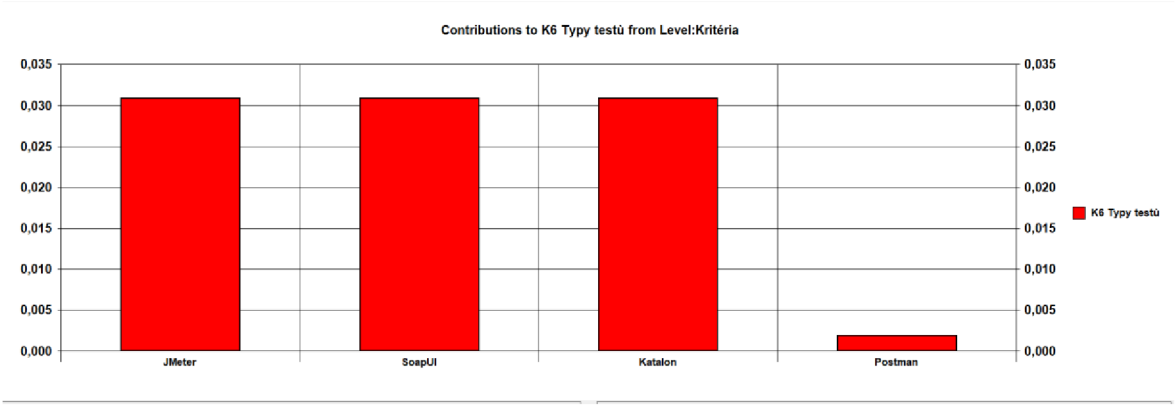
Obrázek 21 Výsledek – K1 (Zdroj: vlastní zpracování)

- K5 Podporované typy API



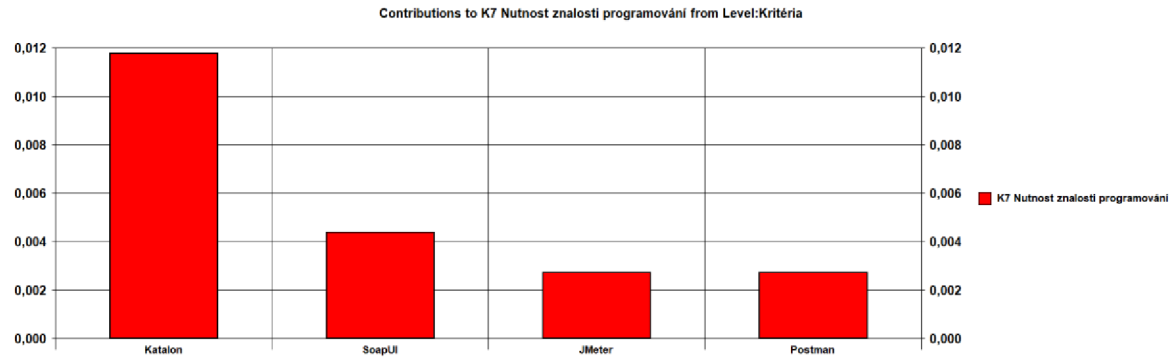
Obrázek 22 Výsledek – K5 (Zdroj: vlastní zpracování)

- K6 Typy testů



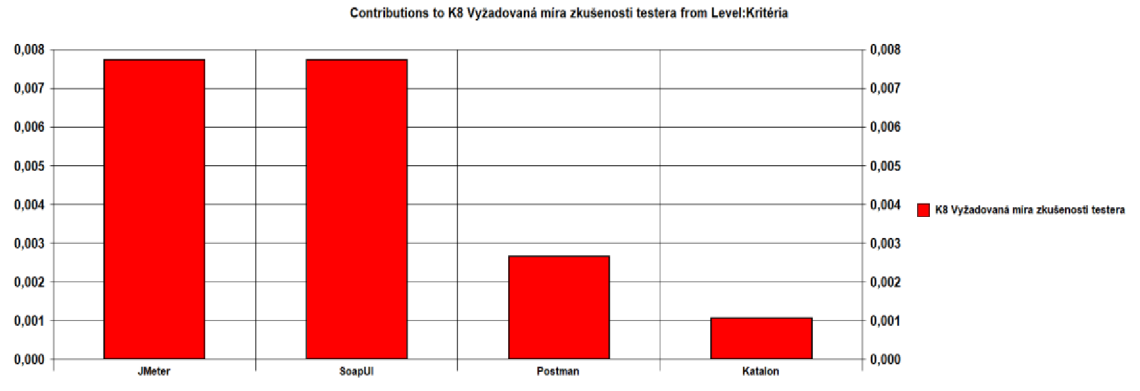
Obrázek 23 Výsledek – K6 (Zdroj: vlastní zpracování)

- K7 Nutnost znalosti programování



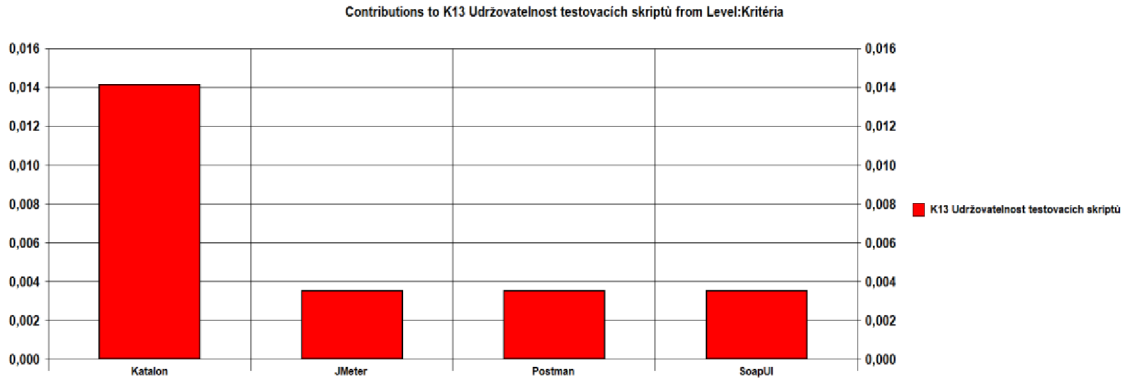
Obrázek 24 Výsledek – K7 (Zdroj: vlastní zpracování)

K8 Vyžadovaná míra zkušenosti testera



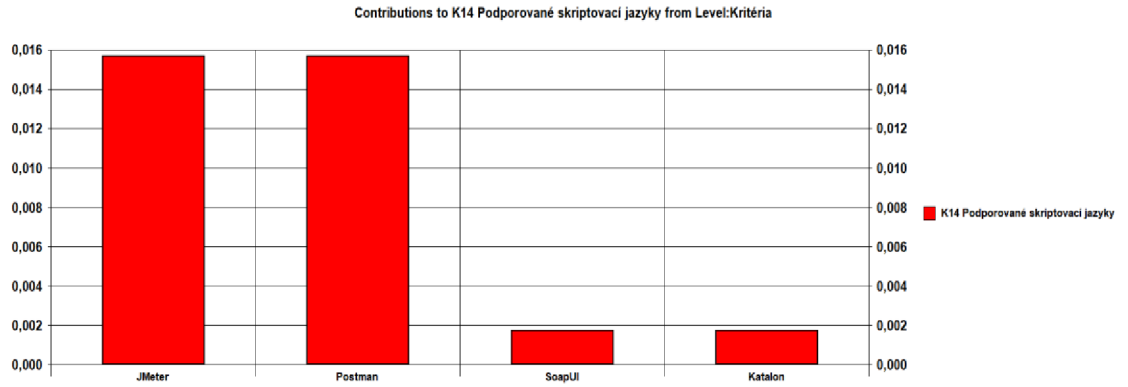
Obrázek 25 Výsledek – K8 (Zdroj: vlastní zpracování)

- K13 Udržovatelnost testovacích skriptů



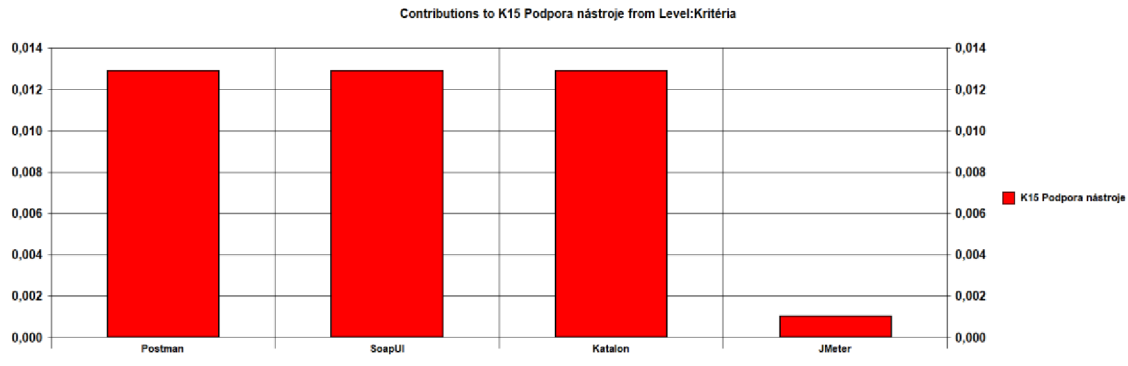
Obrázek 26 Výsledek – K13 (Zdroj: vlastní zpracování)

- K14 Podporované skriptovací jazyky



Obrázek 27 Výsledek – K14 (Zdroj: vlastní zpracování)

- K15 Podpora nástroje



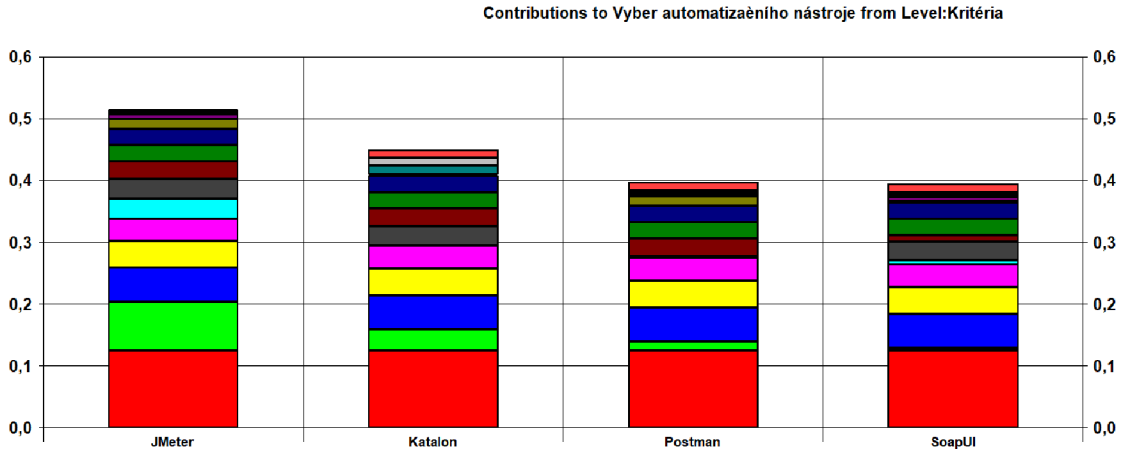
Obrázek 28 Výsledek – K15 (Zdroj: vlastní zpracování)

Následující kritéria byla splněna byla splněna shodně pro všechny vybrané nástroje:

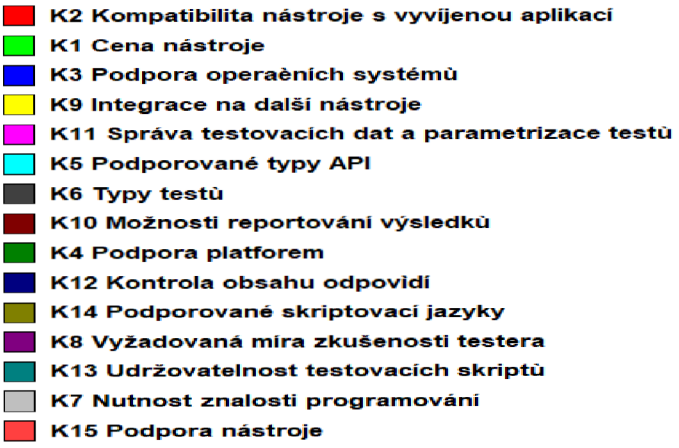
- K2 Kompatibilita nástroje s vyvíjenou aplikací
- K3 Podpora operačních systémů
- K4 Podpora platforem
- K9 Integrace na další nástroje
- K10 Možnosti reportování výsledků
- K11 Správa testovacích dat a parametrizace testů
- K12 Kontrola obsahu odpovědí

Celkový výsledek

Nejlépe hodnocenou variantou je automatizační nástroj JMeter, který je oproti ostatním třem nástrojům výrazně upřednostňován. Dále následují dvě vyrovnané varianty Katalon Studio a SoapUI a nejnižší hodnocení získal nástroj PostMan.



Obrázek 29 Celkový výsledek výběru nástroje



Obrázek 30 Legenda

14. Závěr

V rámci diplomové práce byla představena problematika testování softwaru aplikačního rozhraní webové aplikace v kontextu problematiky manažerského rozhodování a s tím souvisejícího využívání nástrojů pro podporu managementu.
Hlavním cílem diplomové práce bylo vytvoření systému pro podporu rozhodování, za účelem doporučení vhodného nástroje pro konkrétní situaci, která byla definována v rámci prvního dílčího cíle.
V rámci druhého a třetího dílčího cíle byla stanovena výběrová kritéria a zvoleny vhodné alternativy automatizačních nástrojů. Na základě toho byl naplněn čtvrtý dílčí cíl, vytvoření modelu pro podporu rozhodnutí o výběru automatizačního projektu. Pro vytvoření tohoto modelu byl využit nástroj Criterium Decision Plus a byl zde názorně vysvětlen způsob jeho využití.
Pomocí vytvořeného modelu bylo učiněno konečné rozhodnutí a jako nejvhodnější nástroj byl zvolen JMeter a byl tak naplněn pátý dílčí cíl. Tento nástroj byl zvolen jako nejvhodnější, zejména z pohledu nákladů na pořízení nástroje a podporovaných technologií. Nicméně vybrané alternativy se ukázaly ve značném počtu výběrových kritériích jako velmi podobné, což potvrzuje fakt, že dostupná nabídka automatizačních nástrojů si může snadno konkurovat nabízenými funkčnostmi a nabízí se zde provedení detailnější analýzy nástrojů pro identifikaci dalších rozdílů v nabízených funkcionalitách, na základě specifikace požadavků konkrétního projektu.
Přínosem této práce může být reálná využitelnost tohoto rozhodovacího modelu v praxi, při řešení této důležité otázky výběru vhodného automatizačního nástroje, při zavádění automatizace testů do projektu. Zde byla nastíněna pouze modelová situace, požadavky a kritéria různých projektů se mohou lišit. Na základě toho může být tento model modifikován pro konkrétní situaci.

15. Seznam použitých zdrojů

[1] *Altexsoft: What is SOAP: Formats, Protocols, Message Structure, and How SOAP is Different from REST* [online]. [cit. 2021-12-05]. Dostupné z: https://www.altexsoft.com/blog/engineering/what-is-soap-formats-protocols-message-structure-and-how-soap-is-different-from-rest/

[2] *Analytics Week: Evaluation Criteria for Selecting Test Automation Tools* [online]. [cit. 2021-12-20]. Dostupné z: https://analyticsweek.com/evaluation-criteria-for-selecting-test-automation-tools/

[3] *Apache Software foundations: Apache JMeter* [online]. [cit. 2021-12-05]. Dostupné z: https://jmeter.apache.org/

[4] API Testing. Soap UI [online]. [cit. 2021-10-18]. Dostupné z: https://www.soapui.org/learn/functional-testing/api-testing-101/

[5] API Testing. Testing Xperts [online]. [cit. 2021-10-18]. Dostupné z: https://www.testingxperts.com/category/api-testing/

[6] *Art of testing: Automation Testing Tutorial* [online]. 6.9.2021 [cit. 2021-11-20]. Dostupné z: https://artoftesting.com/automation-testing

[7] ASHMAN, Stuart. *QA Matters: Layers of Test Automation* [online]. 28.12.2014 [cit. 2022-04-29]. Dostupné z: https://qa-matters.com/2014/12/28/layers-of-test-automation/

[8] *Automation Tool Selection Criteria* [online]. 24.8.2016 [cit. 2021-12-20]. Dostupné z: https://hsc.com/Resources/Blog/Automation-Tool-Selection-Criteria

[9] AVERWEG, Udo Richard Franz. *Decision-making support systems: Theory & practice*. Bookboon, 2012, 148 s. [cit. 2021-11-20]. ISBN 978-87-403-0176-2.

[10] AXELROD, Arnon. Complete guide to test automation: techniques, practices, and patterns for building and maintaining effective software projects. [cit. 2021-10-28]. New York: Apress, 2018, xxix, 529 stran ; 26 cm. ISBN 978-1-4842-3831-8.

[11] BUENAFLOR, Leanard. *ISO 9126 Software Quality Characteristics* [online]. 2.9.2017 [cit. 2021-11-22]. Dostupné z: https://medium.com/@leanardbuenaflor/iso-9126-software-quality-characteristics-a25a26e7d046

[12] CELESTIAL SYSTEMS. *API Testing in Web Apps* [online]. [cit. 2021-10-28]. Dostupné z: https://celestialsys.com/blog/api-testing-in-web-apps/

[13] DAVIS, Thomas. *What is An API and How Does It Work?* [online]. 23.12.2019 [cit. 2021-12-02]. Dostupné z: https://towardsdatascience.com/what-is-an-api-and-how-does-it-work-1dccd7a8219e

[14] FLYNN, Alexis. *GREEDHEAD: What is document-driven?* [online]. 19.10.2020 [cit. 2021-11-12]. Dostupné z: https://greedhead.net/what-is-document-driven/#What_is_document-driven

[15] FOTR, Jiří, DĚDINA, Jiří. Manažerské rozhodování. 1. vyd. Praha : Vysoká škola ekonomická v Praze, 1993. 170 s. ISBN 80-7079-939-0.

[16] *Gartner: Products In Software Test Automation Market* [online]. [cit. 2021-12-20]. Dostupné z: https://www.gartner.com/reviews/market/software-test-automation

[17] *IBM: Application Programming Interface (API)* [online]. [cit. 2021-12-05]. Dostupné z: https://www.ibm.com/cloud/learn/api#toc-types-of-a-DMAvqUq6

[18] International Software Testing Qualifications Board. *Certified Tester, Foundation Level Syllabus* [online]. Version 2018. 2018 [cit. 2021-11-09]. Dostupné z: https://istqb.in/2018version/CTFL-Syllabus-2018-GA.pdf

[19] *Katalon: A Structured Evaluation for Selecting a Right Automated Testing Tool* [online]. [cit. 2021-12-20]. Dostupné z: https://katalon.com/resources-center/blog/a-structured-evaluation-for-selecting-a-right-automated-testing-tool

[20] *Katalon docs* [online]. [cit. 2021-12-05]. Dostupné z: https://docs.katalon.com/katalon-studio/docs/index.html

[21] *Katalon: How do People Select Test Automation Tools?* [online]. [cit. 2021-12-18]. Dostupné z: https://katalon.com/resources-center/blog/select-test-automation-tools-criteria

[22] *Katalon: Top 15 Automated API Testing Tools in 2022 | Latest Update* [online]. [cit. 2022-04-05]. Dostupné z: https://katalon.com/resources-center/blog/top-5-free-api-testing-toolsGartner

[23] KORVINY, Petr. *Teoretické základy vícekriteriálního rozhodování* [online]. [cit. 2022-04-05]. Dostupné z: https://korviny.cz/Korviny/soubory/teorie_mca.pdf

[24] *Management Study Guide: Decision Making in an Organizational Context* [online]. [cit. 2021-12-16]. Dostupné z: https://www.managementstudyguide.com/decision-making-in-organizational-context.htm

[25] *Management Study Guide: What is Analytical Hierarchy Process (AHP) and How to Use it ?* [online]. [cit. 2021-12-05]. Dostupné z: https://www.managementstudyguide.com/analytical-hierarchy-process.htm

[26] Manuální testování. Testování softwaru [online]. [cit. 2019-03-30]. Dostupné z: http://testovanisoftwaru.cz/manualni-testovani/

[27] OLAVSRUD, Thor. *CIO: Decision support systems: Sifting data for better business decisions* [online]. 29.05.2020 [cit. 2021-11-12]. Dostupné z: https://www.cio.com/article/193521/decision-support-systems-sifting-data-for-better-business-decisions.html

[28] *PostMan: About PostMan* [online]. [cit. 2022-04-05]. Dostupné z: https://www.postman.com/company/about-postman/

[29] *Postman Learnign Center* [online]. [cit. 2022-04-05]. Dostupné z: https://learning.postman.com/docs/getting-started/introduction/What is JMeter? Advantages and Limitations of JMeter (artoftesting.com)

[30] *Red Hat: What is a REST API?* [online]. 08.05.2020 [cit. 2021-12-02]. Dostupné z: https://www.redhat.com/en/topics/api/what-is-a-rest-apiWhat is SOAP API: Formats, Protocols, and Architecture | AltexSoft

[31] SMARTBEAR*: Test Automation Frameworks* [online]. [cit. 2021-12-16]. Dostupné z: https://smartbear.com/learn/automated-testing/test-automation-frameworks/Sprague, Ralph H. a Watson, Hugh J. 1993. Decision support system putting theory into practice. New Jersey : Prentice-Hall, 1993. ISBN:0-13-042235-5.

[32] SMARTBEAR. *What Is API Testing?* [online]. [cit. 2021-10-28]. Dostupné z: https://smartbear.com/solutions/api-testing/

[33] *SoapUI* [online]. [cit. 2021-12-05]. Dostupné z: https://www.soapui.org/docs/soapui-projects/

[34] *Software testing fundamentals: System Testing* [online]. 21.9.2020 [cit. 2021-12-05]. Dostupné z: https://softwaretestingfundamentals.com/system-testing/

[35] *Software Testing Help: How To Choose The Best Automation Testing Tool (A Complete Guide)* [online]. [cit. 2022-04-05]. Dostupné z: https://www.softwaretestinghelp.com/automation-testing-tutorial-4/

[36] *Software Testing Help: How To Choose The Best Automation Testing Tool (A Complete Guide)* [online]. [cit. 2021-12-20]. Dostupné z: https://www.softwaretestinghelp.com/automation-testing-tutorial-4/

[37] SPILLNER, Andreas, Tilo LINZ a Hans SCHAEFER. *Software Testing Foundations*. 4th edition. [cit. 2021-10-28]. Rocky Nook, 2014. ISBN 9781492001485.

[38] *Testbytes: What is White Box Testing? Techniques, Examples and Types* [online]. 17.12.2019 [cit. 2021-11-09]. Dostupné z: https://www.testbytes.net/blog/white-box-testing/

[39] *Testbytes: Black Box Testing Techniques with Examples* [online]. 25.11.2019 [cit. 2021-11-09]. Dostupné z: https://www.testbytes.net/blog/black-box-testing/

[40] *TestingXperts: Unit Testing – What is Its Importance in Software Testing?* [online]. 9.6.2020 [cit. 2021-12-05]. Dostupné z: https://www.testingxperts.com/blog/unit-testing#What%20is%20Unit%20Testing?

[41] *Top 12 Best API Testing Tools (for Developers) [2021]* [online]. 23.8.2021 [cit. 2022-04-05]. Dostupné z: https://rapidapi.com/blog/best-api-testing-tools/

16. Seznam Obrázků

17. Seznam Tabulek

UNIVERZITA HRADEC KRÁLOVÉ
Fakulta informatiky a managementu
Akademický rok: 2021/2022

Studijní program: Informační management
Forma studia: Prezenční
Specializace/kombinace: Informační management (im2-p)

Podklad pro zadání DIPLOMOVÉ práce studenta

Jméno a příjmení: **Bc. Lenka Bártová**
Osobní číslo: **I1900812**
Adresa: Labská Louka 650, Hradec Králové – Třebeš, 50011 Hradec Králové 11, Česká republika

Téma práce: Nástroje automatizace testování API webové aplikace
Téma práce anglicky: Web application API testing automation tools

Vedoucí práce: Ing. Karel Mls, Ph.D.
Katedra informačních technologií

Zásady pro vypracování:

Cíl: Zmapovat stávající stav nástrojů pro automatizované testování aplikací a navrhnout systém pro podporu výběru nejvhodnějšího nástroje pro konkrétní situace.
Osnova:
Analýza problematiky
Přehled nástrojů pro automatizované testování
Stanovení výběrových kriterií
Návrh řešení rozhodovacího systému
Zhodnocení výsledků

Seznam doporučené literatury:

ED-DOUIBI, Hamza; IZQUIERDO, Javier Luis Cánovas; CABOT, Jordi. Automatic generation of test cases for REST APIs: A specification-based approach. In: 2018 IEEE 22nd international enterprise distributed object computing conference (EDOC). IEEE, 2018. p. 181-190.
Alyoubi, Adel. (2015). Decision Support System and Knowledge-based Strategic Management. Procedia Computer Science. 65. 278-284. 10.1016/j.procs.2015.09.079.
API Testing. Testing Xperts [online]. [cit. 2021-10-18]. Dostupné z: https://www.testingxperts.com/category/api-testing/
AXELROD, Arnon. Complete guide to test automation: techniques, practices, and patterns for building and maintaining effective software projects. New York: Apress, 2018, xxix, 529 stran ; 26 cm. ISBN 978-1-4842-3831-8.
API Testing. Soap UI [online]. [cit. 2021-10-18]. Dostupné z: https://www.soapui.org/learn/functional-testing/api-testing-101/