ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE

FAKULTA DOPRAVNÍ

Ústav letecké dopravy

Hodnocení spolehlivosti metodou FMEA s využitím ontologického inženýrství

Bakalářská práce

Markéta Adamcová

Vedoucí práce: Ing. Oldřich Štumbauer, Ing. Andrej Lališ Ph.D.

Praha 2020

ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE
Fakulta dopravní
děkan
Konviktská 20, 110 00 Praha 1

K621 ..... **Ústav letecké dopravy**

ZADÁNÍ BAKALÁŘSKÉ PRÁCE
(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení studenta (včetně titulů):

Markéta Adamcová

Kód studijního programu a studijní obor studenta:

B 3710 – LED – Letecká doprava

Název tématu (česky): **Hodnocení spolehlivosti metodou FMEA s využitím ontologického inženýrství**

Název tématu (anglicky): Reliability Assessment by Means of FMEA Method and Ontology Engineering

Zásady pro vypracování

Při zpracování bakalářské práce se řiďte následujícími pokyny:

- Cíl práce: Realizace automatizovaného hodnocení spolehlivosti letadlových komponent s pomocí existujících ontologických modelů metodiky FMEA
- Analýza metody FMEA a dostupných ontologických modelů
- Výběr a popis letadlového celku pro hodnocení spolehlivosti
- Výběr vhodného ontologického řešení pro automatizované hodnocení spolehlivosti
- Realizace hodnocení spolehlivosti vybraného letadlového celku
- Vyhodnocení celkového řešení

ČVUT FD

Rozsah grafických prací: dle pokynů vedoucího bakalářské práce

Rozsah průvodní zprávy: minimálně 35 stran textu (včetně obrázků, grafů a tabulek, které jsou součástí průvodní zprávy)

Seznam odborné literatury:

Aerospace Recommended Practice ARP4754A. Guidelines For Development Of Civil Aircraft and Systems. SAE International, 2010

Arlow, J. a Neustadt, I. UML 2 a unifikovaný proces vývoje aplikací: objektově orientovaná analýza a návrh prakticky. 2., Computer Press, 2007.

Vedoucí bakalářské práce: **Ing. Oldřich Štumbauer**

Ing. Andrej Lališ, Ph.D.

Datum zadání bakalářské práce: **9. října 2019**
(datum prvního zadání této práce, které musí být nejpozději 10 měsíců před datem prvního předpokládaného odevzdání této práce vyplývajícího ze standardní doby studia)

Datum odevzdání bakalářské práce: **10. srpna 2020**
a) datum prvního předpokládaného odevzdání práce vyplývající ze standardní doby studia a z doporučeného časového plánu studia
b) v případě odkladu odevzdání práce následující datum odevzdání práce vyplývající z doporučeného časového plánu studia

L. S.

doc. Ing. Jakub Kraus, Ph.D.
vedoucí
Ústavu letecké dopravy

doc. Ing. Pavel Hrubeš, Ph.D.
děkan fakulty

Potvrzuji převzetí zadání bakalářské práce.

Markéta Adamcová
jméno a podpis studenta

V Praze dne 9. října 2019

Prohlášení

Prohlašuji, že jsem předloženou práci vypracovala samostatně a že jsem uvedla veškeré informační zdroje v souladu s Metodickým pokynem o etické přípravě vysokoškolských závěrečných pracích.

Nemám závažný důvod proti užití tohoto školního díla ve smyslu zákona § 60 Zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon).

V Praze dne 10.8. 2020 Markéta Adamcová

Poděkování

Ráda bych poděkovala vedoucím své práce Ing. Oldřichu Štumbauerovi a Ing. Andreji Lališovi, Ph.D. za jejich pomoc, podporu a cenné rady poskytované během psaní této práce. Také bych ráda poděkovala všem zaměstnancům Ústavu letecké dopravy na Fakultě dopravní ČVUT za sdílení jejich odborných znalostí během celého mého studia. Nakonec bych chtěla poděkovat své rodině a přátelům za jejich trvalou podporu během studia a neustálou víru ve mně.

Abstrakt

Cílem této bakalářské práce je realizace automatizovaného hodnocení spolehlivosti letadlových komponent s pomocí existujících ontologických modelů metodiky FMEA. V první části práce je popsána současná situace na poli spolehlivostních analýz, dále následuje popis vybraného letadlového systému – systému odmrazování vstupního hrdla motoru. V praktické části je na vybraný systém vypracována analýza FMEA nejprve s tradičním přístupem, poté s ontologickým přístupem. Výsledky jsou porovnány a jsou navrženy změny vybraného ontologického modelu, které vedou ke zkvalitnění a zlepšení automaticky generovaného výsledku ontologického modelu.

Klíčová slova: FMEA, ontologie, konceptuální model, spolehlivost, spolehlivostní analýza, bezpečnost

Abstract

The objective of this bachelor thesis is the implementation of automated evaluation of aircraft components reliability using existing ontological models of FMEA methodology. The first part describes the current situation in the field of reliability analysis, as well as a description of the selected aircraft system - engine anti-ice system. In the practical part, the FMEA analysis of the selected system is first done in a traditional approach and then with an ontological approach. The results are compared and evaluated, and changes are made to selected ontology, which lead to the improvement of the automatically generated output by means of the ontology.

Keywords: FMEA, ontology, conceptual model, reliability, reliability analysis, safety

Obsah

Seznam obrázků

Seznam tabulek

Seznam příloh

Seznam použitých zkratek

A-FMEA	Aplikovaná FMEA
	Application FMEA
APU	Pomocná pohonná jednotka
	Auxiliary power unit
D-FMEA	Konstrukční FMEA
	Design FMEA
EASA	Agentura Evropské unie pro bezpečnost v letectví
	European Union Aviation Safety Agency
FMEA	Analýza možného výskytu a vlivu vad
	Failure Mode and Effects Analysis
FTA	Analýza stromu poruchových stavů
	Fault Tree Analysis
NASA	Národní úřad pro letectví a vesmír
	National Aeronautics and Space Administration
OWL	Webový ontologický jazyk
	Web ontology languague
P-FMEA	Procesní FMEA
	Process FMEA
RPN	Číslo priority rizika
	Risk priority number
STPA	Systémově-teoretický model nehod a procesů
	Systems Theoretic Process Analysis
UML	Unifikovaný modelovací jazyk
	Unified Modeling Language

Úvod

Tato práce se zabývá spolehlivostními analýzami. Jsou to analýzy, které hodnotí systém a snaží se nalézt kroky k minimalizaci rizik a ke zvýšení bezpečnosti. V letecké dopravě je toto obzvláště důležité. Důsledkem spolehlivostní analýzy může být vylepšení návrhu systému, lze zvýšit povědomí o kritických součástech a celkově spolehlivost, a bezpečnost lze zvýšit vyloučením určitých poruchových stavů nebo přijetím zmírňujících opatření. Jedna z analýz, která se aplikuje během procesu výroby letadlového systému je FMEA (analýza možného výskytu a vlivu vad). Byla vyvinuta v polovině 20. století a používá se tak už desetiletí.

V poslední době se ale zvyšují nároky na kvalitu a efektivitu spolehlivostních analýz nejen ze strany kontrolních úřadů ale i samotných firem. Způsob provádění spolehlivostních analýz od svého vzniku neprošel žádnou razantní změnou, naopak vývoj letadlové techniky jde prudce dopředu a systémy jsou stále složitější. Klasický přístup může být zdlouhavý a neefektivní proces. Jeden ze způsobů, jak zvýšit kvalitu provedené analýzy, je ontologický přístup. Ontologie, jako odvětví filozofie, se snaží o vytvoření modelu reálného systému a dnes má tento přístup vysokou počítačovou podporu.

Cílem této práce je ověřit vhodný ontologický model FMEA a případně ho modifikovat tak, aby byl vhodný pro použití v leteckém průmyslu. Toto by mělo zajistit konzistenci mezi sdílenými a opakovaně používanými informacemi, snížení časové náročnosti analytických procesů, a tak zlepšit celkovou účinnost analýzy. Tímto způsobem lze provést analýzu spolehlivosti snadněji, rychleji a díky tomu je možné nebezpečné režimy selhání identifikovat, zmírnit a eliminovat dříve.

1 Spolehlivost v letecké dopravě

Pojem spolehlivost je možné použít téměř v jakémkoli odvětví, ale ne vždy musí znamenat to samé. V letectví je to jedena z hlavních vlastností, na kterou se musí brát zřetel při návrhu, vývoji, údržbě a provozu letadlové techniky. Je zde ale potřeba od sebe odlišit dva termíny.

Spolehlivost (z anglického dependability) jako vlastnost objektu definujeme jako souhrnný termín pro popis pohotovosti a činitelů, které ji ovlivňují-bezporuchovost, udržovatelnost, zajištění údržby. [1]

Bezporuchovost (z anglického reliability) definujeme jako schopnost sledovaného objektu plnit požadovanou funkci v daných podmínkách a v daném časovém intervalu za předpokladu, že na začátku sledovaného období je objekt ve stavu schopném plnit požadovanou funkci a schopnost plnit funkci končí s nástupem jevu porucha. [1]

Reliability lze z angličtiny přeložit i jako spolehlivost, a pro potřeby práce bude dále termín spolehlivost používán ve smyslu termínu výše definované bezporuchovosti.

1.1 Inženýrské metody spolehlivosti

Technický systém je spojení dvou a více komponentů, které jsou mezi sebou propojeny a interagují spolu na základě jejich funkcí. Interakce všech komponentů mezi sebou tvoří požadovanou funkci celého systému. Použitím metod hodnocení spolehlivosti můžeme v takovém systému nalézt slabé propojení a zhodnotit vliv poruchy jednoho komponentu na funkci systému.

Hlavní cíle použití metod hodnocení spolehlivosti jsou:

-Prevence a snížení pravděpodobnosti poruchy nebo četnosti poruch

-Identifikace příčin poruch a přijetí opatření k jejich prevenci

-Stanovení zmírňujících opatření

Důraz na spolehlivost a bezpečnost se kontinuálně zvyšuje. Stále se zvětšující intenzita leteckého provozu klade větší a větší důraz na spolehlivost letadlové techniky. Národní a nadnárodní organizace, které se problematikou zabývají, jako je např. Agentura Evropské unie pro bezpečnost v letectví (EASA) kladou při certifikaci stále větší důraz na použití dostupných spolehlivostních metod.

Trend posledního desetiletí s důrazem na systémové selhání a selhání lidského faktoru vedl ke vzniku několika nových analýz, jako je např. STPA (Systémově-teoretický model nehod a procesů). [2]

Význam spolehlivostních analýz ale neklesá, naopak s vývojem nové techniky a stále složitějšími elektronickými systémy v letadle význam jejich použití roste.

V letectví se pro analýzu technického systému používají dvě základní analýzy. Obě byly vyvinuty v 60. letech pro potřebu analýzy složitých technických systémů a nalezení kritické poruchy. Jsou to desetiletími prověřené metody, prošly řadou úprav a modifikací v závislosti na použití. Jejich využití není jen v letectví. V letectví jsou tyto metody používány několikrát v kombinaci s dalšími metodami během různých fází vývoje a návrhu.

FTA (Analýza stromu poruchových stavů) využívá deduktivního přístupu k analýze, tedy přístup shora dolů založený na identifikaci selhání konečné úrovně a analýza všech možných režimů selhání, které by jej mohly způsobit. [1]

FMEA (analýza možného výskytu a vlivu vad) s induktivním přístupem zdola nahoru založeným na analýze funkcí a poruch komponentů na místní úrovni a rozvinutí poruchy až k efektu na celý systém. [1]

1.2 FTA

FTA je strukturovaná shora dolů, obvykle začíná nežádoucí událostí nejvyšší úrovně a poté klesá dolů ve formě stromu a ukazuje, jaké události k vrcholové události vedou. FTA graficky znázorňuje události a logické operátory. Události popisují poruchový stav komponentu, který svým výskytem zapříčiňuje vznik události o úroveň výše. Například událost *žádné zásobení elektrickou energií* zapříčiní vznik události *žádné světlo z lustru*. Grafické znázornění možných typů událostí je ukázáno na obrázku 1.

Obrázek 1:Příklad grafického znázornění událostí [3]

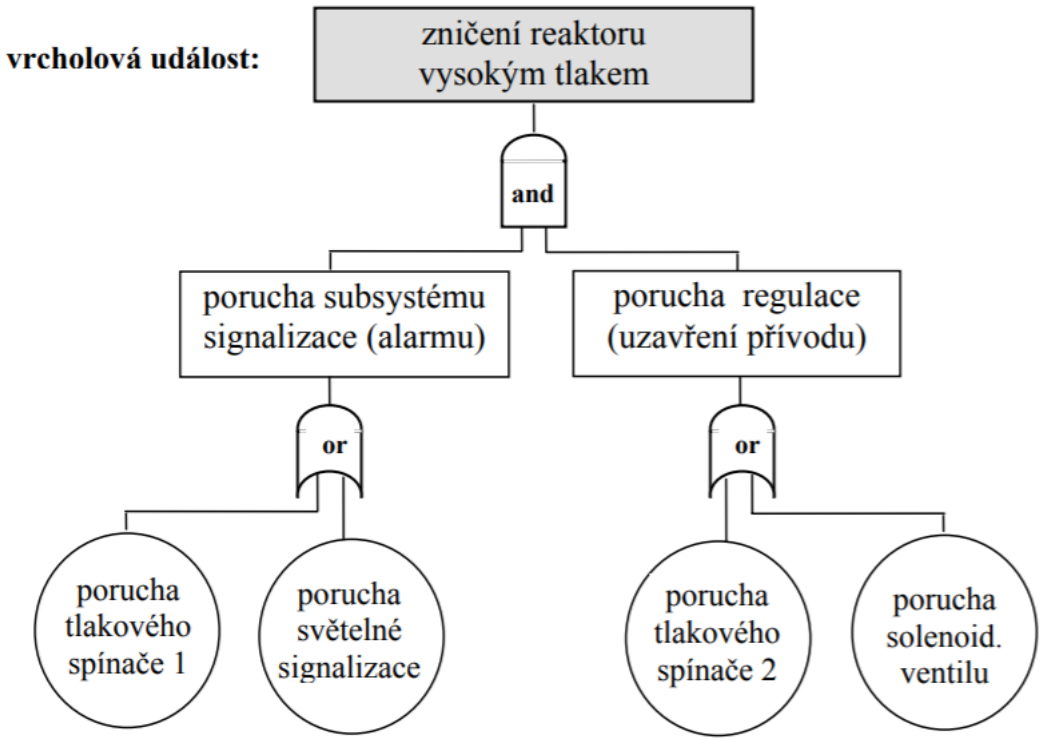
Takovou kombinaci událostí, která zapříčiní vznik události o úroveň výše nám znázorňují logické operátory. Situaci, kdy je potřeba ke vzniku hlavní události více podřazených událostí najednou, nám reprezentuje operátor AND. Naopak situaci, kdy ke vzniku stačí pouze jedna z několika událostí znázorníme OR. Například událost *žádné* světlo z lustru může zapříčinit *žádné zásobení elektrickou energií* nebo (OR) vypínač *na pozici vypnuto*. Grafické znázornění logických operátorů je ukázáno na obrázku 2.

Obrázek 2: Příklad grafického znázornění logických operátorů [3]

FTA nabízí komplexní a vizualizovaný pohled na zkoumaný systém. Díky tomu pomáhá uživatelům rychle pochopit výsledky a určit chyby v procesu. Je vhodná pro složité komplexní systémy. Do FTA lze zakomponovat i prvek pravděpodobnosti a to tak, že se základním nejspodnějším a mezilehlým událostem přiřadí hodnota pravděpodobnosti, získaná buď měřením nebo odhadem, a pravděpodobnost události o úroveň výše se získá v závislosti na logických operátorech buď součtem nebo součinem nebo jejich kombinací. Díky tomu lze získat pravděpodobnost vrcholové události a pracovat tak s matematickým vyjádřením odhadu rizika. Příklad FTA je na obrázku 3. [3]

1.3 FMEA

FMEA je jednou z nejpoužívanějších analýz v oblasti rizik. Na rozdíl od FTA je tvořena zdola nahoru. Hlavním cílem je identifikovat všechny možné poruchové stavy a rozpoznat jejich efekt na funkci komponentů nebo celkovou funkci systému. Ve své finální podobě navrhuje řešení nebo zmírňující opatření k možným scénářům. Podobně jako do FTA i do FMEA lze zakomponovat kvantitativní vyjádření rizika, a na základě tohoto atributu lze určit prioritu jednotlivým scénářům a nápravným opatřením.



Obrázek 3:Přiklad grafického znázornění FTA [4]

FMEA se obvykle skládá z následujících informací:

- seznam všech komponentů
- potenciální poruchové stavy komponentů
- příčiny poruchových stavů
- popis lokálních/systémových/finálních efektů všech poruchových stavů
- stupeň závažnosti, detekce a výskytu všech poruchových stavů
- nápravná opatření ke zmírnění

Poznámka: v FMEA se dá, dle oblasti použití a rozlišovací schopnosti, jako základní kámen použít místo komponentu například systém, subsystém nebo proces. Pro potřeby zkoumané oblasti, to jsou letadlové systémy, a pro rozsah této práce bude výčet atributů FMEA zjednodušen jen na komponenty zkoumaného systému.

1.3.1 Tvorba FMEA

Ne vždy musí mít FMEA přesně stejnou formu jako je popsaná níže. Některé procesy lze při její tvorbě vynechat. Vždy záleží na tom, za jakým účelem FMEA tvoříme, jestli chceme

znát jen slabá místa systému, nebo chceme možným scénářům přiřadit závažnost a definovat případná opatření. Rozhodnutí o tom, jaké informace do své FMEA zahrnout nebo nezahrnout, je na společnosti, resp. konkrétním analytikovi. Hlavním cílem je ale vždy hodnocení spolehlivosti a bezpečnosti analyzovaného systému formou definování všech možných způsobů selhání a jejich vliv na hlavní funkci.

FMEA netvoří jen jedna osoba nebo jeden tým. Do práce by se mělo zapojit co nejvíce týmů z různých oblastí záběru FMEA, např. designu, výroby montáže, údržby, kvality atd. Je to z důvodu zahrnutí co největšího pole působnosti, dojde tak k překrytí znalostí jednotlivých expertů a minimalizuje se tím riziko přehlédnutí detailů.

Základní kroky tvorby FMEA jsou: [5]

1. *Sběr informací*

Prvním krokem k tvorbě jakékoli analýzy je vždy sběr dat. Kvalitní a správná data jsou pro tvorbu FMEA kritická. Nedostačená a chybějící data mohou vést v procesu tvorby k fatálním chybám. Informace můžeme získat ze schémat, nákresů, výkresů, manuálů, revizí, záznamů údržby, hlášení poruch, předchozí vytvořené FMEA atd.

Z těchto zdrojů získáme výčet komponentů a jejich funkcí v systému.

2. *Vyplnění informací v hlavičce*

Jedná se vyplnění základních informací v záhlaví, které by nám měly zjednodušit hledání a orientaci. Jedná se například o název projektu, datum zahájení a ukončení projektu, organizaci, jméno odpovědného vědoucího atd.

3. *Identifikace poruchových stavů, jejich efektů a příčin*

Z předchozího bodu (sběru dat) je k dispozici seznam komponentů a jejich funkcí v systému. Dalším krokem je ke každému komponentu přiřadit jeho poruchový stav. Ten se vždy odvíjí od jeho funkce a jedná se o takový stav, který komponentu znemožňuje vykonat jeho funkci nebo vykonání funkce ovlivňuje. Například v palivovém systému bude v komponentu nádrž poruchový stav *díra*, protože funkcí nádrže je zadržovat palivo. Naopak poruchovým stavem nebude *škrábnutí*, protože to funkci komponentu neovlivní.

Po identifikaci všech poruchových stavů (failue modů) komponentů určíme jejich efekt na funkci komponentu. Protože jsou komponenty mezi sebou propojené, bude se přenášet i efekt poruchy. Tímto způsobem lze rozvětvit poruchové stavy až k jejich

efektu na systémovou funkci. Efekty tedy rozlišujme na úrovni lokální, systémové a finální.

Lokální efekt je takový efekt, který ovlivňuje pouze funkci komponentu, jehož porucha efekt vyvolala. Systémový efekt ovlivňuje funkci více spojených komponentů a finální efekt ovlivňuje hlavní funkci systému. Například v palivovém systému se v komponentu nádrž vyskytne porucha *díra*. *Díra* ovlivní funkci nádrže, což je *uchování paliva*. Ta dále způsobí systémový efekt *snížení množství paliva*. Pokud všechno palivo unikne, na systém bude působit finální efekt *žádné palivo*, protože to ovlivňuje hlavní funkci systému, což je *uchování a distribuce paliva*.

Dále se ke každému poruchovému stavu identifikuje jeho příčina. Každý poruchový stav má minimálně jednu příčinu selhání, přičemž FMEA na rozdíl od FTA nepočítá se selháním několika věcí najednou.

4. *Závažnost, detekce, výskyt*

Jedná se o číselné ohodnocení poruchových stavů. Díku tomu lze ve finále priorizovat jednotlivé kroky ke zmírnění následků a rozdělit komponenty na kritické a nekritické. Přiřazené číslo se nazývá RPN (z anglického risk priority number) a skládá se ze tří složek. RPN se počítá jako součin hodnot jeho tří složek-závažnosti, výskytu a detekce a jeho rozsah je 1-1000.

$$RPN=S*O*D$$

Závažnost (S-z anglického severity) je ohodnocení poruchy spojené s nejzávažnějším následkem. Rozsah se obvykle volí 1 (bez následku) -10 (rychlá realizace následku bez varování).

Výskyt (O-z anglického occurrence) je číslo spojené s pravděpodobností výskytu poruchového stavu. Nejedená se ale o přímé vyjádření pravděpodobnosti ani četnosti výskytu, ale pouze o kvalitativní škálu. Rozsah se volí od 1 (téměř nemožné) -10 (téměř nevyhnutelné).

Detekce (D-anglického detection) je číslo asociované s nejlepší možností preventivní kontroly konstrukce, která zabrání vzniku poruchy. Rozsah je opět od 1 (téměř jistý) – 10 (absolutně nejistý).

5. *Doporučená zmírňující opatření*

Na základě výsledků RPN lze doporučit postup k prevenci nebo zmírnění rizika poruchy. Toho lze dosáhnout snížením pravděpodobnosti, zlepšením detekční metody nebo úplným přepracováním konstrukčního řešení.

6. *Přijatá opatření*

Po provedení zmírňujícího opatření se vloží přesný popis akce. Přepracuje se S, O, D a přepočítá RPN.

7. *Výstup*

Grafická reprezentace FMEA je obvykle ve formě tabulky, která obsahuje informace řazené do sloupců (komponenty, mody selhání, následky, příčiny selhání, závažnost, výskyt, detekce, RPN, doporučené akce, přijaté akce, nové RPN, komentáře)

Prvek -------- Funkce	Možná vada	Možné následky vady	Význam	Kritičnost	Možné Příčiny (mechanismy vady)	Výskyt	Stávající opatření pro prevenci	Stávající řízení procesu	Odhalitelnost	RPN	Doporučená opatření	Odpovědnost --------- Termín	Provedená opatření	Význam	Výskyt	Odhalitelnost	RPN
Zadání objednávky	Nekompletní objednávka	Objednávka se musí vrátit k doplnění	5		Neznalost postupu	8	Metodika	Není	5	200	Školení 1x za rok	1.10.2017 Karel	Zavedeno pravidelné školení	5	4	4	80
					Neinformování klienta o nutných dokumentech	9	Žádné	Není	6	270	Použít checklist	1.9.2017 Novák	Checklist vytvořen	5	2	4	40
	Chybný údaj v objednávce	Nemožnost vyřídit úvěr	8		Úmysl	2	Čerpání Přes CAP	Kontrola 4 očí	1	16	Žádné						
					Překlep	2	Žádné	Vizuální kontrola	3	48	Žádné						

Obrázek 4: Příklad výstupu FMEA [6]

1.4 Ontologie

Jedním z hlavních důvodů, proč vznikly analýzy FMEA a FTA bylo, že finální testování výrobku ne vždy dokázalo odhalit možné režimy selhání. Letecký průmysl je na takové chyby obzvláště citlivý. V minulosti špatně navržené letadlové systémy stály mnoho lidských životů. Proto je důležité provést správně a kvalitně spolehlivostní analýzy v několika fázích návrhu a výroby, aby se podobné vady odhalily a odstranily před finálním testováním a uvedením do provozu.

Není pravda, že by význam analýz s ústupem nedokonalých konstrukcí minulého století klesl. Naopak s dalšími a dalšími inovacemi a vývojem nových technologií význam kvalitních bezpečnostních analýz roste. Např. chyba v softwaru MCAS v letadlech B737

MAX byla odhalena až po dvou fatálních nehodách z let 2018 a 2019 a celosvětovém uzemnění všech letadel tohoto typu.

Avšak tvorba FMEA je velice složitý a komplexní proces. Vyžaduje velmi dobrou znalost systému, spolupráci lidí z různých oborů, kteří se na vývoji podílejí. Při její tvorbě dochází k obrovským datovým tokům. Její forma tabulky se sestává z tisíců dat. Všechny tyto faktory společně s lidským faktorem zvyšují riziko vzniku chyby.

Nejkritičtějšími místy v procesu je:

- rozhraní komunikace: člověk-člověk

- špatná orientace v tabulce: velké množství dat

Řešením těchto kritických míst je vytvoření automatizovaného nebo částečně automatizovaného systému. Ideálním nástrojem pro vytvoření takového systému je použití ontologického datového modelu.

Slovo ontologie pochází z řečtiny v překladu znamená „věda o bytí“. Jedná se tedy o filozofickou vědu. Dle Encyklopedie Britannica chápeme termín ontologie jako studii o bytí obecně nebo o tom, co se neutrálně vztahuje na všechno, co je skutečné. [7]

Vždy ale závisí na kontextu, ve kterém termín použijeme. Můžeme mu rozumět jako: [8]

- odvětví metafyziky zabývající se povahou a vztahy bytí

- konkrétní teorie o povaze bytí nebo druzích věcí, které existují

V posledních letech se termínu věnuje velká pozornost v oblasti počítačových a informačních věd. Forma aplikované ontologie tak vytváří ideální nástroj pro sdílení informací. V této oblasti definujeme ontologii jako způsob, jak určit specifický obsah, sdílení a opětovné použití znalostí o pojmenování definic a reprezentaci entit, jejich vlastností a vztahy.

Většina takových ontologií se zaměřuje na určení tříd, ty popisují koncepty v doméně.

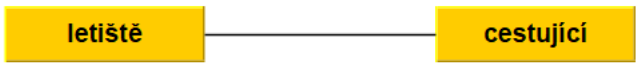
1.4.1 Tvorba ontologie

Vytvoření zahrnuje několik kroků: [9]

1. *Definování tříd v ontologii*

Jedná se o nejzákladnější krok, ve kterém definujeme, jakou skutečnost chceme ontologií popsat. Například chci vytvořit model letiště. Na letiště chodí cestující. Jsou

tedy definované 2 třídy a vztah mezi nimi. Pro potřeby této práce a pro lepší srozumitelnost zde bude použité grafické znázornění UML (unifikovaný modelovací jazyk). Třídu pomocí UML zobrazíme obdélníkem a vztah čárou, jak je zobrazeno na obrázku 5. Jednoduchá čára v UML reprezentuje nejzákladnější vztah, tzv. asociaci. Entity jsou v ontologii ve vztahu ale mohou existovat i nezávisle na sobě. [10]



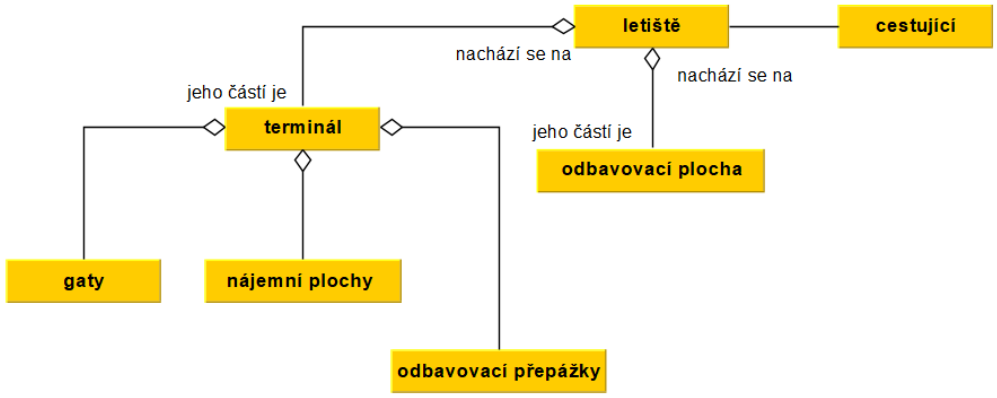
Obrázek 5:Grafická reprezentace tříd a vztahů

Dále je důležité určit rozlišovací schopnost. Jedná se o určení toho, do jaké míry chci skutečnost zkoumat, jaké detaily chci ještě zahnout. Například se rozhodnu, že do třídy *letiště* chci zahrnout terminál a odbavovací plochu. Dále do třídy *terminál* gaty, nájemní plochy a odbavovací přepážky, ale už mě nebudou zajímat zaměstnanci, kteří na terminále pracují.

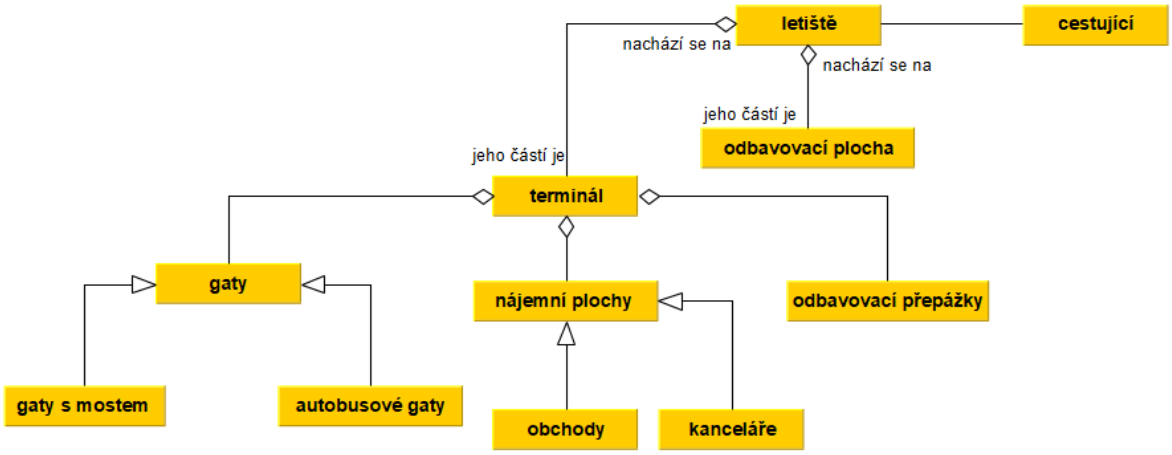
Částmi letiště jsou *terminál* a *odbavovací* plocha. Tento vztah se nazývá agregace a reprezentuje vztah typu část-celek. V UML jej znázorníme jako plnou čáru zakončenou kosočtvercem u třídy reprezentující celek. Stejný vztah vzniká i mezi třídami *terminál* a *gaty, odbavovací přepážky, nájemní plochy*. Tento vztah reprezentuje obrázek 6.

2. *Uspořádání tříd do hierarchie*

Vztahy mezi třídami jsou někdy více specifické. Tyto vztahy často fungují na principu nadřazenosti a podřazenosti, a to nám umožňuje definovat nadtřídy a podtřídy. Například, jak už bylo zmíněno výše, chci do ontologie zahrnout gaty, nájemní plochy a odbavovací přepážky. Do třídy *nájemní plochy* chci dále zahrnout obchody a kanceláře. Pro všechny zmíněné je třída *nájemní plochy* nadtřídou a všechny zmíněné jsou podtřídami třídy *nájemní plochy*. To samé mohu udělat s třídou *gaty* a přiřadit ji podtřídy *autobusové gaty* a *gaty s mostem*. Vztah podřazenosti znázorníme graficky čárou s šipkou, jak je ukázáno na obrázku 7.

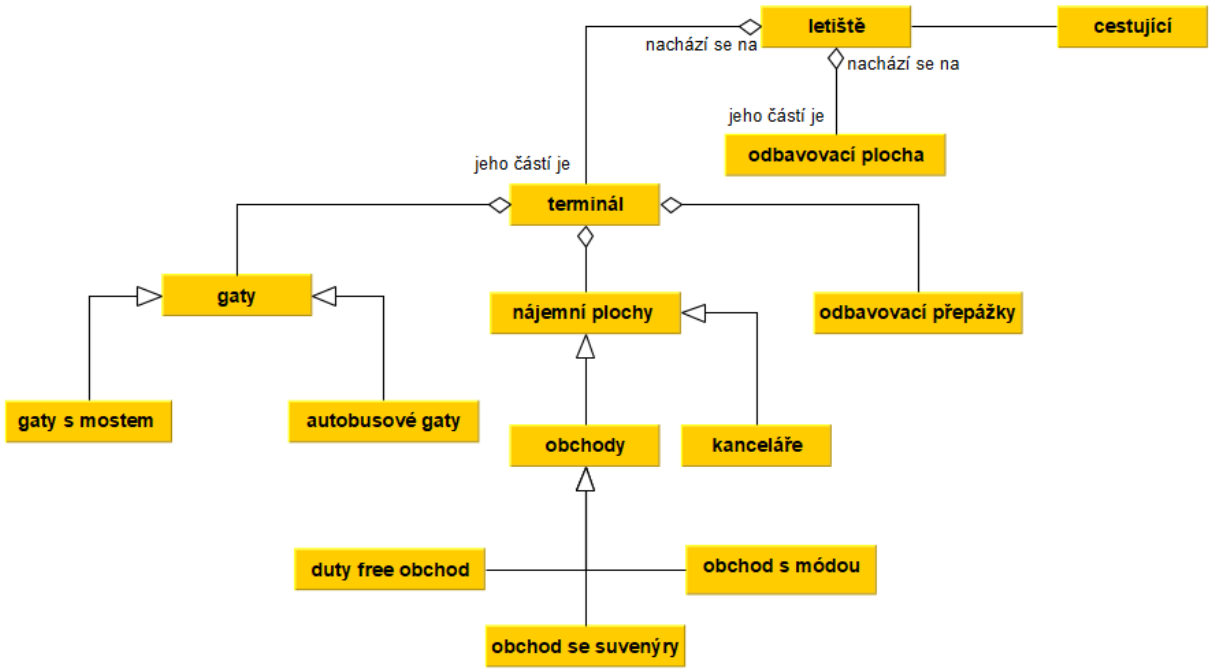


Obrázek 6: Agregace: vztah typu část-celek



Obrázek 7: Grafické znázornění podtřídy a nadtřídy

Neplatí, že by v ontologii musel být jen jeden stupeň nadřazenosti. Může nastat situace, kdy podtřída v jednom vztahu, může být nadtřídou v jiném vztahu. Příklad je na obrázku 8. Třída *obchody je* podtřídou třídy *nájemní plochy*, naopak ve vztahu k třídám *obchody s módou, obchody se suvenýry, duty free obchody* je nadtřídou.



Obrázek 8: Grafické znázornění podtřídy a nadtřídy 2

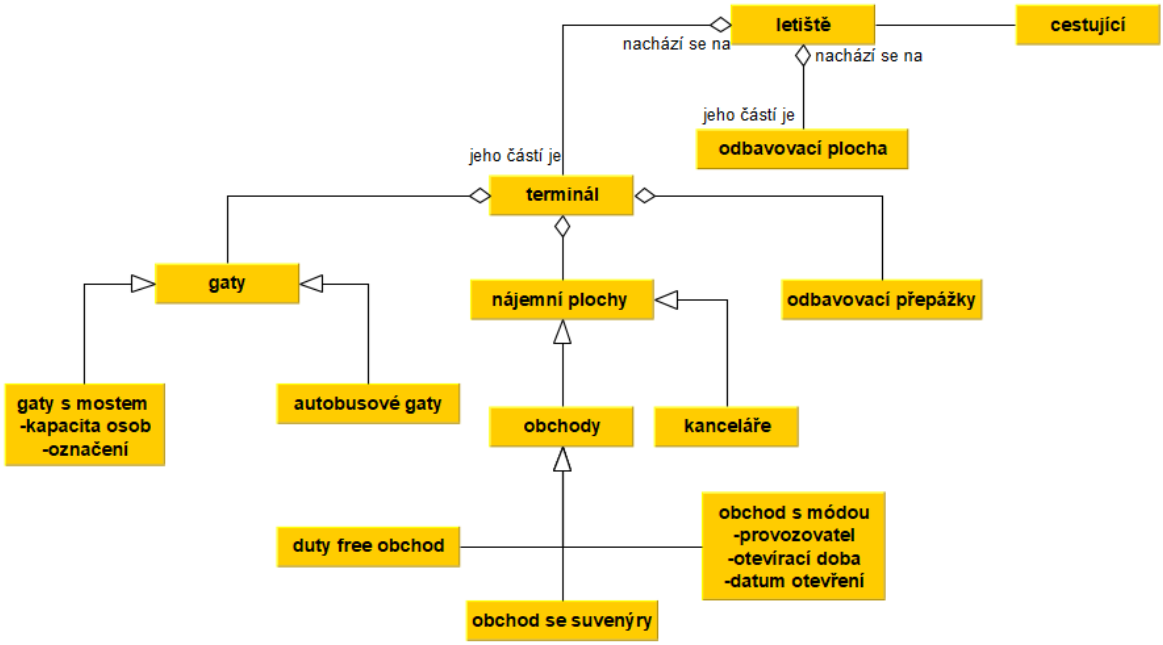
3. *Definice vlastností tříd*

Samotné třídy neposkytují dostatek informací. Jakmile je definujeme, musíme popsat i jejich vnitřní strukturu. Neplatí, že by všechny definované třídy měly stejné vlastnosti. Naopak u každé třidy definujeme zvlášť to, co nás zajímá, jaké parametry jsou důležité. Všechny instance (konkrétní data), které do třídy zařadíme, budou nositeli těchto vlastností. Příklad je zobrazen na obrázku 9.

Dalším důležitým krokem je určit povolenou hodnotu vlastností. Jedná se hlavně o typ hodnoty vlastnosti. V našem případě vlastnost kapacita osob bude definovaná datovým typem *číslo*, datum otevření datovým typem *datum* a provozovatel nebo označení datovým typem *text*.

4. *Vytvoření instancí*

Posledním krokem je nahrání dat. Tedy vytvoření instance vyžaduje výběr třídy, vytvoření individuální instance a vyplnění hodnot vlastností. Vytvoříme instance třídě *gate s mostem* a vyplníme hodnoty slotů, jak je ukázáno v tabulce 1.



Obrázek 9: Příklad zobrazení vlastností tříd

Tabulka 1: Instance třídy gate s mostem

Gate s mostem	první vpravo	druhý vpravo
Označení	B1	B2
Kapacita osob	200	149

1.4.2 Současné ontologie FMEA

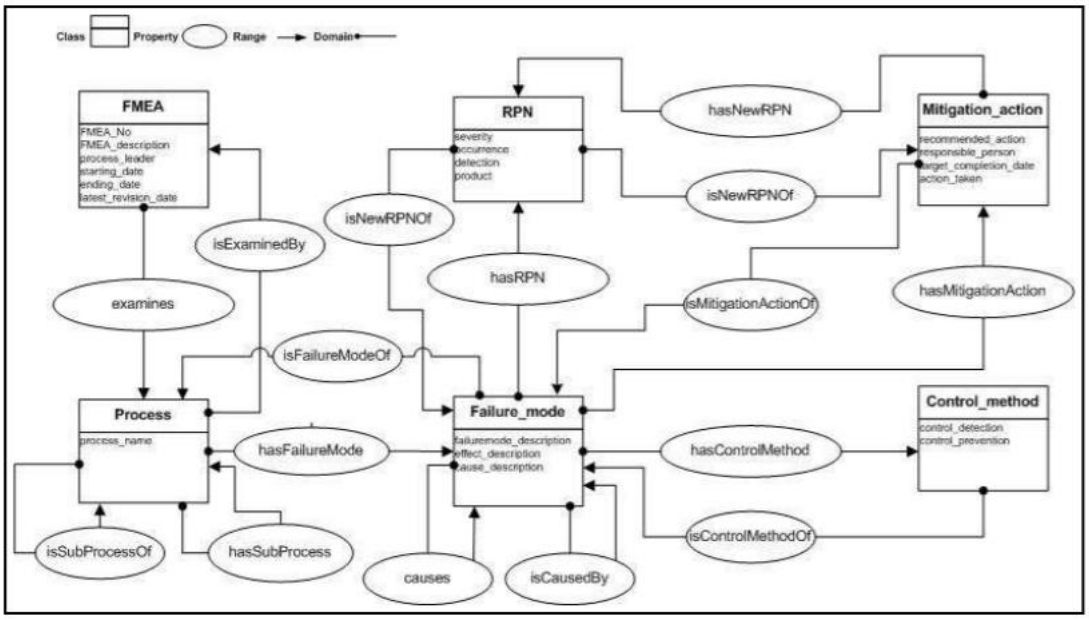
Jak bylo ukázáno výše, ontologický model je jeden z nejvhodnějších nástrojů pro vytvoření modelu skutečnosti. Lze v něm snadno definovat vztahy mezi třídami a určit vlastnosti, které jsou důležité. Po nahrání instancí se tak ontologie stává vhodným nástrojem pro přenos ukládání a práci s velkým množstvím dat.

V minulosti se už několik pracovních skupin snažilo vytvořit ontologický model FMEA.

V roce 2019 se tomuto problému věnovala Simona Bolčeková ve své diplomové práci „Reliability analysis of mechanical and lubrication system of an aircraft engine"[11]. V práci analyzovala všechny dostupné ontologické modely FMEA s cílem vytvořit model, který by byl vhodný pro použití na olejový systém motoru M601 společnosti GE Aviation Czech, s.r.o.[1] Dostupné modely jsou popsány níže.

[1] https://www.geturboprops.com/cz/uvod

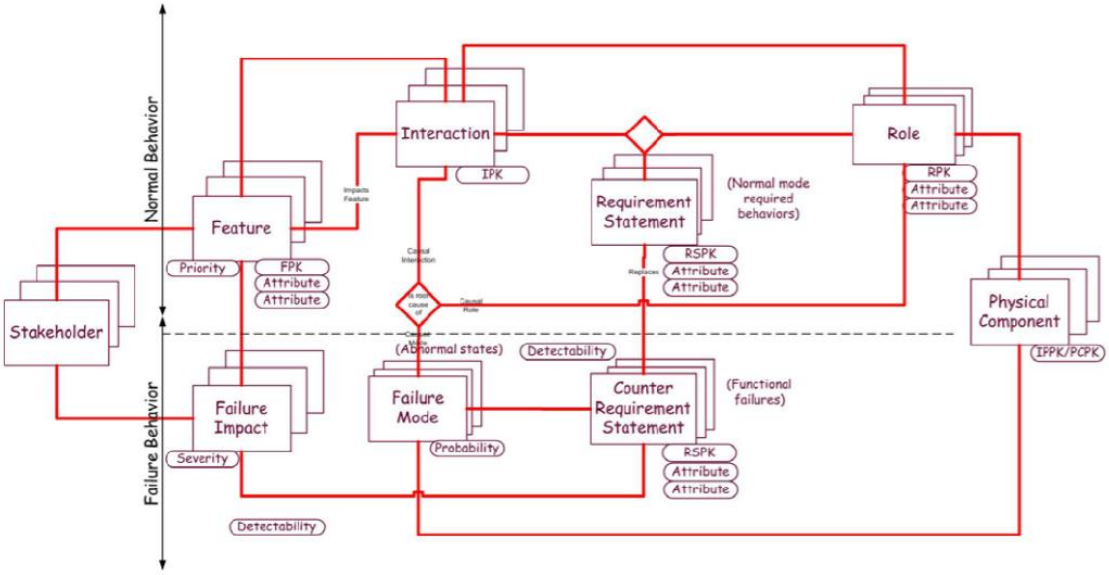
První ze současně dostupných ontologických modelů FMEA představili ve své práci v roce 2006 Z. Rehman a C. V. Kifor. V práci s názvem „An Ontology to Support Semantic Management of FMEA Knowledge“ představili model v reakci na zvýšený zájem firem se analýzami zabývat. Navrhli model, který by firmám zmírnil náklady a úsilí vynaložené na realizaci metody FMEA, zmenšil by chybovost, zlepšil by grafickou interpretaci a organizovanost. Grafické znázornění jejich ontologického modelu je na obrázku 10. [12]



Obrázek 10:Grafické znázornění modelu FMEA z práce "An Ontology to Support Semantic Management of FMEA" [12]

Jak je vidět na obrázku 10, model má pouze 6 tříd. Je zaměřený hlavně na procesy, poruchové stavy a zmírňující akce.

Další dostupný model FMEA pochází z práce „Failure Risk Analysis: Insights from Model-Based Systems Engineering“ představené v roce 2010. V práci byla FMEA rozdělena na tři druhy. D-FMEA (design FMEA), která je zaměřená na technické systémy, A-FMEA (application FMEA), která se zaměřuje na abnormální chování externího systému (typicky člověka) a P-FMEA (process FMEA), která se zaměřuje na výrobní proces zájmového subjektu. Snaha autora byla spojit tyto tři druhy do jednoho metamodelu, který je na obrázku 11. [13]

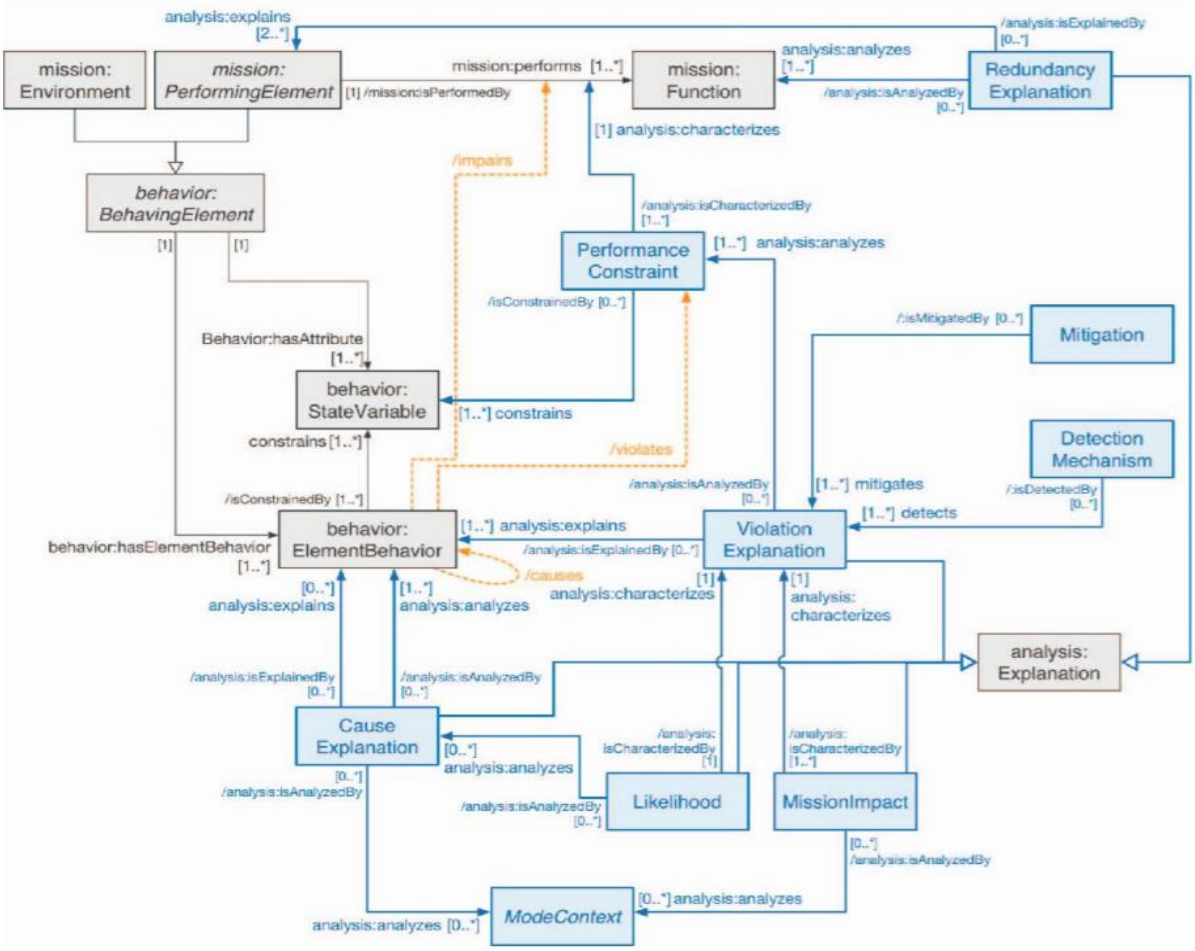


Obrázek 11: model FMEA z práce "Failure Risk Analysis: Insights from [13]"

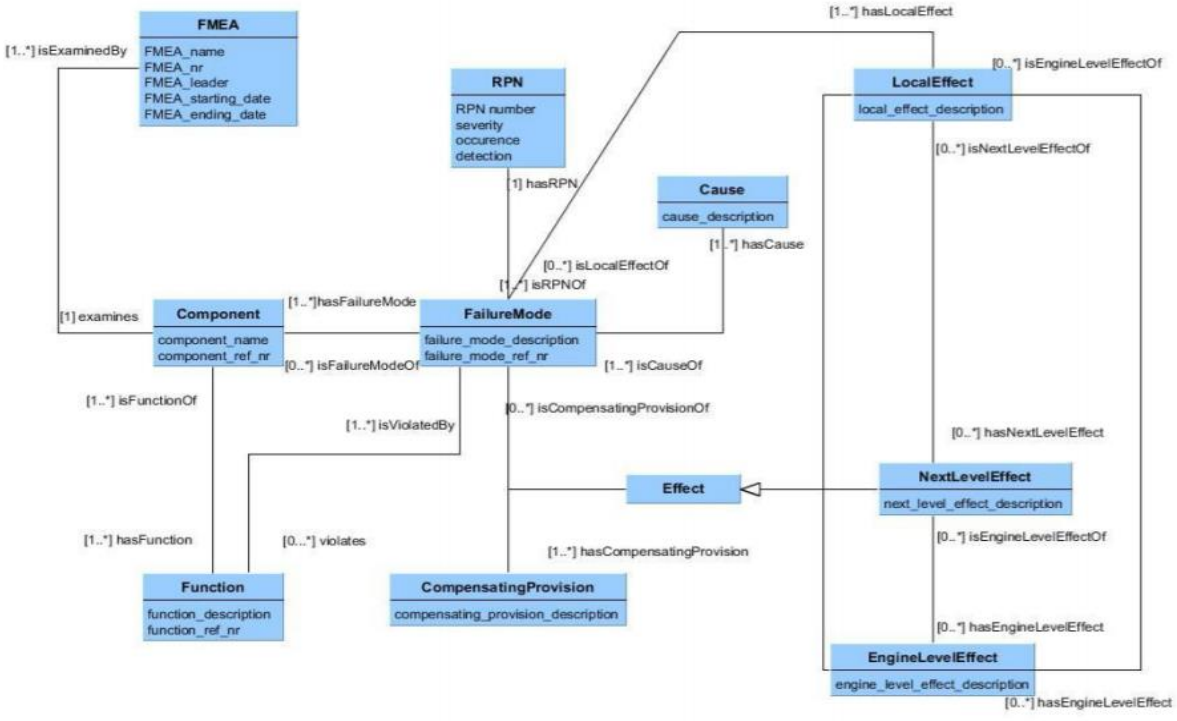
Třetí „Fault Management" ontologie byla vytvořena týmem expertů z NASA (National Aeronautics and Space Administration). Cílem bylo vytvořit jednotný model správy chyb s FMEA a FTA doménou. Zahrnuje problematické chování včetně režimů selhání, jeho důsledků, šíření, a příčin selhání. Ontologie byla navržena pro vesmírný program a správu vesmírné stanice, takže pro oblast působnosti velice podobnou letecké technice. „Fault management" ontologie je na obrázku 12. [14]

Simona Bolčeková ve své diplomové diplomové práci pracovala s výše zmíněnými modely. Po uvážení jejich vhodností byl vybrán model NASA jako nejvíce vhodný a modifikován tak, aby se dal použití na letadlový systém. Jeho grafické znárodnění je na obrázku 13. [11]

Po prohledání dostupných zdrojů a vědeckých článků jsem došla k závěru, že od doby publikace práce Simony Bolčekové nebyly publikované další modely, které by v kontextu této práce byly relevantní. Její model je nadále nejvíce vhodný a bude v praktické části použit. Tento model byl navržen cíleně pro práci s letadlovými systémy. Avšak byl designován na míru jen jednomu systému, a to olejovému systému motoru M601 společnosti GE Aviation Czech, s.r.o. Je tedy na místě dále ověřit, zda je tento model použitelný i pro jiné systémy. V další části práce tedy dojde k výběru vhodného systému a jeho podrobnému popisu.



Obrázek 12: "Fault management" ontologie [14]



Obrázek 13: Modifikovaný model NASA [11]

V dalších částech práce bude provedena FMEA na jiný systém nejprve tradiční metodou, poté pomocí zmiňovaného modelu. Výsledky se porovnají a z toho vyvodím patřičné závěry. Za tímto účelem bude použit stejný nástroj pro práci s ontologiemi, Protégé[2] (podrobně bude popsán dále dále), jako použila Simona Bolčeková. K dispozici je také originální soubor s modelem. K realizaci výsledku analýzy FMEA s pomocí ontologie FMEA lze stejným způsobem využít webovou aplikaci Apache Jena Fuseki[3], která byla navržena ve zmiňované diplomové práci.

[2] https://protege.stanford.edu/
[3] https://jena.apache.org/

2 Odmrazovací systémy

Pro ověření vhodnosti použití modelu FMEA z práce Simony Bolčekové pro použití i na jiné letadlové systémy než je olejový systém motoru, byl vybrán systém odmrazování vstupního hrdla motoru letadla B737. Jedná se o část pneumatického systému letadla, který zajišťuje odstranění námrazy a ledu z motoru a zabraňuje jejímu vzniku. V této kapitole popíši princip fungování tohoto systému.

Odmrazování letadel je velice důležité. Rozlišujeme odmrazování před letem, kdy se na letadlo nanese speciální kapalina, která letadlo námrazy zbaví a na určitou dobu zamezí vzniku nové, a odmrazovaní během letu. To může být zajištěno několika systémy: [15]

- termální: Pomocí ohřátého vzduchu nebo elektrického ohřevu se přivede teplo do požadovaného prostoru. To rozpustí už nahromaděný led a dále zabraňuje jeho vzniku.

- pneumatický: Tento systém využívá gumové zarážky umístěné na náběžných hranách. Ty se po aktivaci rychle nafouknou a vyfouknou, čímž rozbijí nahromaděný led. Tento postup je velice citlivý na načasování, protože pokud je vrstva ledu při aktivaci moc tenká nebo naopak silná, systém nepůsobí správně.

Námraza na náběžných hranách a na trupu letadla zhoršuje obtékání vzduchu a narušuje aerodynamické proudění. V minulosti bylo již několik nehod zapříčiněno námrazou. Například v roce 2004 se letadlo Bombardier Challenger 601 zřítilo krátce po startu na letišti Montrose (Spojené státy americké) poté, co posádka nevyužila pozemní odmrazovací stanice. Dále v roce 2014 letadlo McDonnell Douglas MD-83 Air Algerie havarovalo poté, co během letu na autopilota došlo k nahromadění ledu na motorech. To vedlo ke snížení výkonu motorů a tahu a nakonec pádu.

2.1 Pneumatický systém

Pneumatický systém je systém, který uchovává a transportuje vzduch po letadle k různým účelům. Dříve byl tento systém využíván i k ovládání systému řízení a fungoval na podobném principu jako systém hydraulický. V moderních letadlech se už k řízení nepoužívá. V některých typech letadel se pneumatický systém využívá k ovládání podvozku, dveří nebo k provozu nouzových zařízení.

Hlavní funkce pneumatického systému ale je odebrání vzduchu z kompresorové části motoru, jeho uchování, distribuce a tlaková a tepelná úprava. Takovýto odebraný vzduch se používá při spouštění motoru, k odmrazování motoru a křídel, k tlakování zásobníku hydraulického systému nebo ke klimatizování kabiny. [16]

2.1.1 Pneumatický systém B737

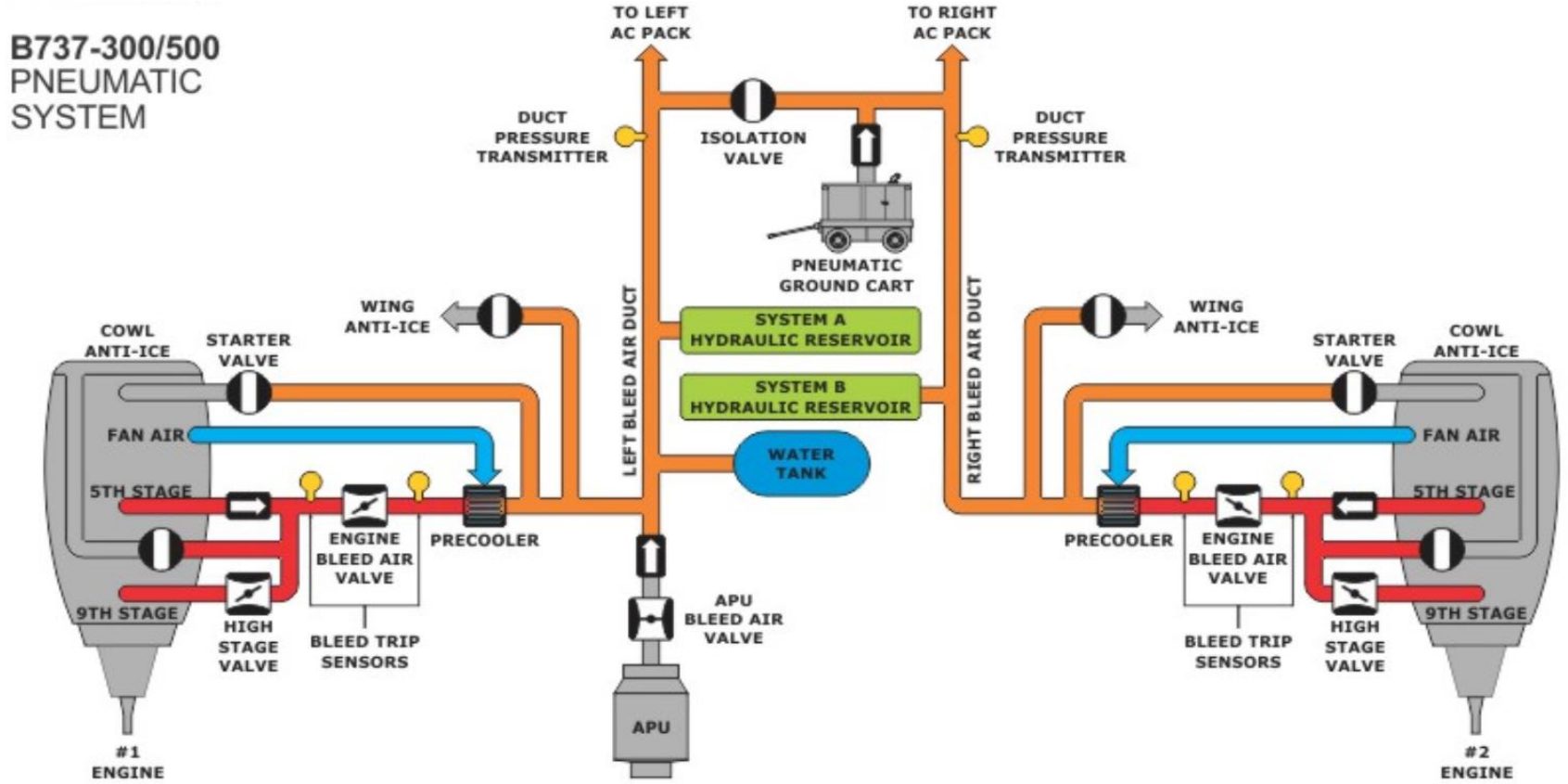
Pneumatický systém letadla B737 může získávat vzduch z motoru, APU (pomocná pohonná jednotka) nebo externího zdroje. Systémy, které závisí na dodávce vzduchu jsou klimatizace a přetlakování kabiny, protinámrazové systémy na křídle a motoru, spouštění motoru, tlakování hydraulické nádrže a nádrže s vodou. [17]

Jedná se o systém trubek a ventilů, které transportují vzduch po celém letadle. V systému se také nacházejí součásti, které upravují tlak a teplotu proudícího vzduchu. Dalšími důležitými částmi jsou kontrolní ventily a senzory, které monitorují stav proudícího vzduchu. Schéma pneumatického systému B737 je na obrázku 14.

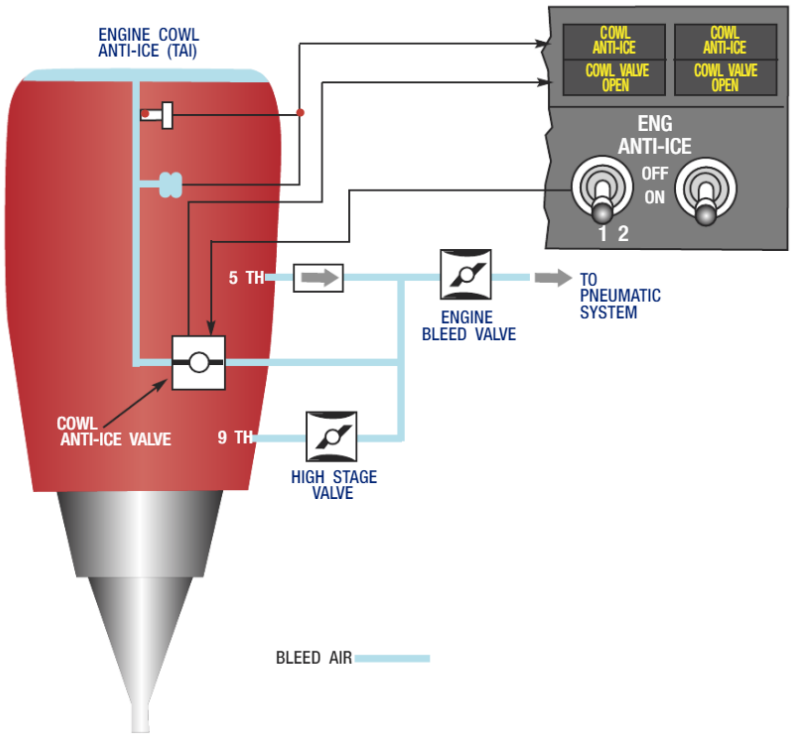
2.2 Odmrazování vstupního hrdla motoru

Systém odmrazovaní vstupního hrdla motoru je součástí pneumatického systému letadla. Jeho funkcí je odstranit námrazu ze vstupního hrdla motoru a zamezit jejímu dalšímu vzniku. Případná námraza by mohla ovlivnit plynulost proudění vzduchu do motoru a tím snížit výkon. Schéma tohoto systému je na obrázku 15.

Systém odebírá vzduch z kompresorové části motoru na pozici 5. a 9. stupně. Za 5. stupněm se nachází jednosměrný ventil, který zabraňuje proudícímu vzduchu v systému proniknout zpět do motoru. Za 9. stupněm se nachází vysokotlaký ventil. Ten má podobnou funkci jako jednosměrný ventil. Odebraný vzduch se pak dostává k ventilu *anti-ice*. Ten svojí polohou dále propouští nebo nepropouští vzduch k odmrazovací části. Tento ventil je spojen s kontrolním panelem v kokpitu, kde přepínač v poloze „ON“ nebo „OFF“ otevírá/zavírá ventil. Jako kontrolní prvek je zde senzor, který svým rozsvícením potvrzuje souhlasnou polohu přepínače a ventilu. Dále se nachází tlakové čidlo, které při překročení určitého tlaku sepne výstrahu v kokpitu. Dále se nachází už jen odmrazovací část a vzduch zde odchází výduchem ven. Celý systém je pak od zbytku pneumatického systému oddělen oddělovacím ventilem, jeho poloha ale nemá na funkci odmrazovacího systému vliv, a proto není na zjednodušeném schématu na obrázku 16 uveden.

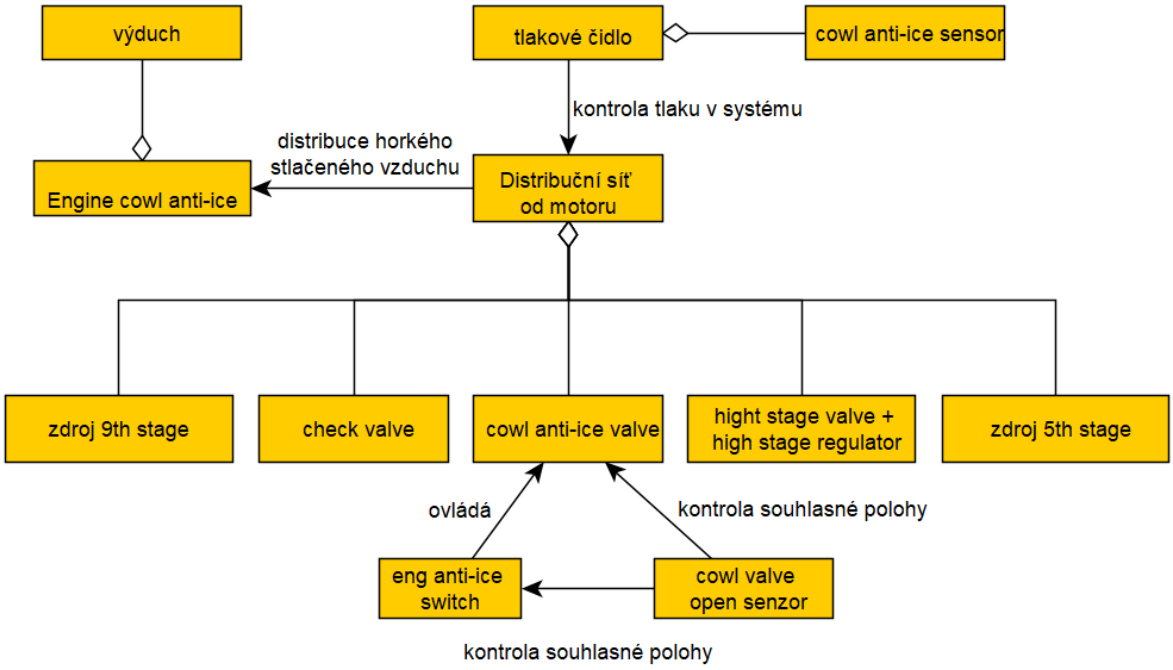


Obrázek 14: Pneumatický systém letadla B737 [18] (engine-motor; 9th stage-zdroj na pozici 9. stupně; 5th stage-zdroj na pozici 5. stupně; fan air-vzduch ventilátoru; cowl anti-ice-místo na vstupním hrdle, kde dochází k odmrazování; starter valve-startovací ventil; high stage valve-vysokotlaký ventil; engine bleed air valve-odvzdušňovací ventil motoru; bleed trip senzors-odvzdušňovací senzory; precooler-předchlazovač; APU bleed air valve-odvzdušňovací ventil APU; wing anti-ice-odmrazování křídla; water tank-nádrž s vodou; left bleed air duct-levé pneumatické potrubí; systém A hydraulic reservoir-hydraulické nádrž systému A; systém B hydraulic reservoir-hydraulické nádrž systému B; duct pressure transmitte-snímač tlaku v potrubí; *isolation valve-izolační ventil; to left/ right AC pack-do klimatizace; pneumatic ground cart-pneumatický pozemní vozík; right bleed air duct-pravé pneumatické potrubí)*



Obrázek 15: Systém odmrazování vstupního hrdla motoru [19] (bleed air-proudící vzduch; 9th-zdroj na pozici 9. stupně; high stage valve-vysokotlaký ventil, cowl anti-ice valve-ventil ovládající odmrazování; 5th-zdroj na pozici 5. stupně; engine cowl anti-ice- místo na vstupním hrdle, kde dochází k odmrazování; engine bleed valve-odvzdušňovací ventil motoru; to pneumatic system-dále do pneumatického systému; eng anti ice-přepínač pro ovládání ventilu pro ovládání odmrazování)

Na schématu na obrázku 16 je schéma systému zobrazeno pomocí UML. U některých komponentů jsou použity anglické výrazy. Je to z toho důvodu, že některé výrazy nelze přesně přeložit do českého jazyka a jejich použití by bylo nepřesné. Z tohoto důvodu budou dále v této práci a v přílohách názvy komponentů použity v takové formě jako na obrázku 16.



Obrázek 16: Schéma odmrazování vstupního hrdla motoru (engine cowl anti-ice-místo kde dochází k odmrazování na vstupním hrdle; cowl anti-ice senzor-senzor na kontrolním panelu indikující zvýšení tlaku; cowl anti.ice valve-ventil ovládající průchod vzduchu k odmrazovací části; eng anti-ice switch-přepínač na kontrolním panelu ovládající cowl anti-ice valve; cowl valve open senzor-senzor na kontrolním panelu potvrzující souhlasnou polohu cowl anti-ice valve a eng anti-ice switch; check valve-jednosměrný ventil, zdroj 5th stage-zdroj vzduchu na pozici 5. stupně kompresorové části motoru; high stage valve +high stage regulator-vysokotlaký ventil + vysokotlaký regulator; zdroj 9th stage- zdroj vzduchu na pozici 9. stupně kompresorové části motoru

3 Spolehlivost systému odmrazování vstupního hrdla motoru

V této části budu vybraný systém odmrazování vstupního hrdla motoru analyzovat nejprve tradiční metodou, poté pomocí ontologie. Ve finále budou oba výsledky porovnány.

3.1 Tvorba FMEA tradiční metodou

V praktické části práce došlo nejprve k tvorbě FMEA tradiční metodou. Níže popíšu postup tvorby této FMEA.

1. *Studium materiálů*

Všechny materiály ke studiu vybraného systému, odmrazovacího systému vstupního, pocházejí z volně dostupných zdrojů. Během této fáze tvorby analýzy došlo ke sběru informací k dosáhnutí velmi dobré znalosti tohoto systému. Identifikace komponentů a funkcí lze poté určit přímo.

2. *Identifikace komponentů a jejich funkcí*

Po nastudování materiálů došlo k určení všech komponentů v systému. Všechny komponenty tvoří dohromady systém. Každý z komponentů má v systému jednu a více funkcí.

Identifikace funkcí komponentu vychází z jeho povahy. Pokud bychom se podívali na komponent *tlakové čidlo* bez toho, abychom měli přehled o zbytku systému, i bez hlubšího studia bychom získali povědomí o funkci tohoto komponentu. Nebo komponent *ventil*, ten ze své povahy propouští nebo nepropouští proudící substanci. V systému se nachází tří ventily, ale ne všechny mají jen tuhle jednoduchou funkci. Např. *high stage valve,* kromě už zmíněné funkce, také reaguje na určitý tlak proudícího vzduchu, což zapříčiní jeho uzavření. Z tohoto důvodu je pro objevení těchto na první pohled skrytých funkcí důležité pečlivé nastudování systému. Výsledný seznam komponentů pneumatického systému B737 a jejich funkcí je uveden v tabulce 2.

Tabulka 2: Seznam komponentů a jejich funkcí

KOMPONENT	FUNKCE
engine cowl anti-ice	zahřívá vstupní hrdlo motoru
výduch	odvádí vzduch ze systému ven po zahřátí náběžné hrany motoru
tlakové čidlo	citlivé na hodnotu tlaku nad 65 psi
cowl anti-ice senzor	upozorňuje na zvýšený tlak vzduchu v systému rozsvícením
cowl anti-ice valve	propouští vzduch dále do systému
	reguluje tlak na max 50 psi
eng anti-ice switch	ovládá polohu cowl anti-ice valve
cowl valve open senzor	potvrzuje souhlasnou polohu eng anti-ice switch a cowl anti-ice valve
	upozorňuje na nesouhlasnou polohu anti-ice switch a cowl anti-ice valve
zdroj 5th stage	odebírá ohřátý vzduch z motoru na pozici 5. stupně kompresoru
zdroj 9th stage	odebírá ohřátý vzduch z motoru na pozici 9. stupně kompresoru
check valve	jednosměrně propouští vzduch
	otevře se, pokud tlak vzduchu z 5th stage je vetší než 32 psi
	zabraňuje vniknutí zpětného toku do zdroje 5th stage
High stage valve + hihg stag regulator	zajišťuje dostatečný přísun vzduchu do pneumatického systému při nízkorychlostním režimu motoru (do výkonu motoru cca 47%)
	reguluje tlak vzduchu na 32psi
	odstavuje 9th stage zdroj při vysokorychlostním režimu motoru (zavře se pokud tlak v systému je větší než 32psi)
	zabraňuje vniknutí zpětného toku do zdroje 9th stage
trubka	propojuje jednotlivé části systému
	rozvádí vzduch

3. *Identifikace poruchových stavů*

Poté, co se podařilo identifikovat všechny komponenty a jejich funkce, je identifikace poruchových stavů (dále používán původní termín failure mode) snadnější. Připomeňme, že failure mode je takový poruchový stav komponentu, který mu zabraňuje vykonat jeho funkci, nebo ovlivňuje správné vykonání funkce. Failure mode souvisí jak s funkcí, tak s komponentem. Failure mode se objevuje u komponentu (jeho stav) a ovlivňuje jeho funkci. Tabulka 3 ukazuje příklad komponentů, funkcí a failure modů.

Tabulka 3: Příklad komponentů, funkcí a failure módů

KOMPONENT	FUNKCE	FAILURE MODE
tlakové čidlo	citlivé na hodnotu tlaku nad 65 psi	citlivé na špatnou hodnotu
		nedetekuje tlak
cowl anti-ice senzor	upozorňuje na zvýšený tlak vzduchu v systému rozsvícením	kontrolka nefunguje
eng anti-ice switch	ovládá polohu cowl anti-ice valve	přepínač nefunguje
cowl valve open senzor	potvrzuje souhlasnou polohu a upozorňuje na nesouhlasnou eng anti-ice switch a cowl anti-ice valve	citlivé na špatnou polohu
		nedetekuje polohu
zdroj 5th stage	odebírá ohřátý vzduch z motoru na pozici 5. stupně kompresoru	na pozici 5. stupně není odebírán z motoru žádný vzduch
		na pozici 5. stupně odebírá menší množství vzduchu
check valve	zabraňuje vniknutí zpětného toku do zdroje 5th stage	check valve se nezavírá
		check valve netěsní
	jednosměrně propouští vzduch	check valve se neotevírá
		check valve netěsní
	otevře se, pokud tlak vzduchu z 5th stage je vetší než 32 psi	check valve se neotevírá
		check valve reaguje na nesprávný tlak

Z tabulky 3 je vidět, že k jedné funkci je možné identifikovat více failure modů, dále také to, že větší komponenty s více funkcemi mají hodně failure modů, ale ne všechny ovlivňují všechny funkce.

4. *Rozvinutí efektů*

Dalším krokem v tvorbě FMEA je nalezení řetězce efektů. Každý failure mode má efekt na úrovni komponentu, ten označíme jako lokální efekt. Protože komponenty mezi sebou mají vazby, přenáší se i efekty. Lokální efekt se může rozvinout a ovlivnit funkci skupiny komponentů a vytvořit tak efekt další úrovně. Takto se může rozvinout až k efektu, který ovlivňuje systémovou funkci a vzniká finální efekt. Efektů další úrovně může vzniknout mnoho, pro zjednodušení a pro shodu při porovnávání s výsledkem, který vznikne pomocí ontologie, se rozvinutí efektů omezí na tři stupně. Tento třístupňový systém může v jakékoli fázi končit efektem *no effect*, to znamená, že daný failure mode, od kterého se cesta větvila, nemá na daném stupni efektu žádný vliv na funkci. Tímto způsobem můžeme ve výsledku identifikovat kritické failure mody, tedy takové, které mají ve finálním efektu nějaký efekt. Příklad návaznosti efektů je v tabulce 5.

Tabulka 4: Příklad třístupňového systému efektů

LOKÁLNÍ EFEKT	EFEKT DALŠÍ ÚROVNĚ	FINÁLNÍ EFEKT
špatná indikace detekce tlaku v systému	zvýšení tlaku v systému	no effect
žádná indikace detekce tlaku v systému	zvýšení tlaku v systému	no effect
žádná indikace zvýšeného tlaku v systému	zvýšení tlaku v systému	no effect
nelze uzavřít přívod vzduchu	nelze vypnout odmrazovaní	no effect
průnik vzduchu anti-ice ventilem	snížená funkce	no effect
nelze otevřít přívod vzduchu	nefunkční odmrazovaní náběžné hrany	nefunkčnost odmrazovacího systému
vzduch prochází anti-ice ventilem bez regulace tlaku	zvýšení tlaku v systému	no effect
přepínačem nelze ovládat přívod vzduchu	nefunkční odmrazovaní náběžné hrany	nefunkčnost odmrazovacího systému

5. *Příčiny*

Dalším krokem je určení příčin selhání. Failure mody se neobjevují jen tak z ničeho, takže musí mít každý alespoň jednu příčinu. Tabulka 6 ukazuje příklad určení pravděpodobných příčin. V našem případě se bavíme pouze o pravděpodobných příčinách. Kdyby k uvedené situaci skutečně došlo, příčina selhání by se určila exaktní

metodou. Já jsem ale schopna pouze odhadnout nejpravděpodobnější příčinu selhání a tu do řešení zahrnout.

Tabulka 5: Příklad pravděpodobných příčin

FAILURE MODE	PŘIČINA
trhlina	vada materiálu
	vada výroby
	koroze
	opotřebení
ucpání cizím předmětem zvenčí	špatná údržba
	kolize s ptákem
citlivé na špatnou hodnotu	špatná konfigurace
	zkrat
nezavírá se	opotřebení
	kontaminace
	nedostatečné promazání
neotevírá se	opotřebení
	kontaminace
	nedostatečné promazání
nereguluje tlak	opotřebení
	vada výroby
neodebírá z motoru žádný vzduch	ucpání

6. *Výsledek*

Celá výsledná tabulka z tvorby FMEA tradiční metodou je v příloze 1. Třídy komponent, funkce, failure mode, lokální efekt, efekt další úrovně, finální efekt, a příčiny jdou po sobě tak jak bylo ukázáno výše v postupu tvorby.

Do tvorby nebyla zahrnuta práce s RPN číslem. Je to volitelný atribut s hodnotou, kterou určuje společnost nebo bezpečnostní analytik, který danou analýzu tvoří v závislosti na daných potřebách. V případě potřeby se dá zpětně doplnit.

3.2 Tvorba FMEA pomocí ontologie

V této části práce budu modelovat stejný systém pomocí vybraného ontologického modelu FMEA navrženého v diplomové práci Simony Bolčekové. [9]

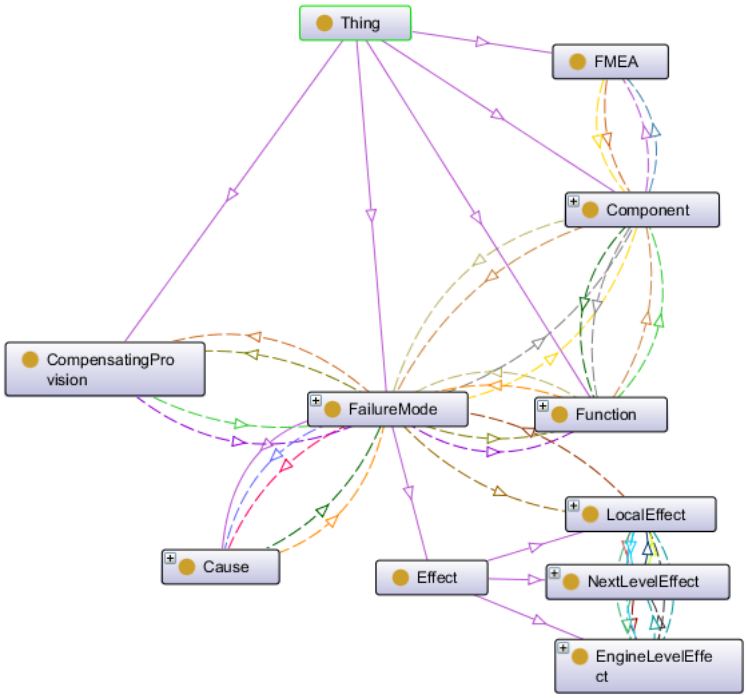
3.2.1 Protégé

Pro práci s ontologií budu používat program Protégé. Je to volně dostupný ontologický editor, který pracuje s formátem OWL (web ontology languague) s RDF/XML formátováním. Výhoda počítačového prostředí je snadnější práce s velkým množstvím

dat. Uložená data mohou být taky dále zpracována dalšími aplikacemi. Protégé byl vyvinut na Stanfordské univerzitě a první verze byla spuštěna už v roce 1999. Dnes je to jeden z nejpoužívanějších ontologických editorů.

Níže popíšu postup práce s daty v tomto editoru. Ontologie, která byla výstupem již zmíněné diplomové práce, byla také zpracována v programu Protégé a v této práci byla využita.

Na obrázku 17 můžeme vidět grafické znárodnění této ontologie zvané ontograf. Je to jedna z funkcí nástroje Protégé. Protégé dokáže na základě definovaných tříd a jejich vazeb automaticky vygenerovat grafické znázornění. Na obrázku stojí za povšimnutí třída *thing-věc*. Je to třída, která stojí na začátku tvorby každé ontologie a vychází přímo z definice ontologie, která je modelování skutečnosti, a co je skutečné, musí být věcí. Dále vidíme všechny třídy, které každá FMEA musí mít a byly zmíněny již v části práce o tvorbě FMEA tradiční metodou.

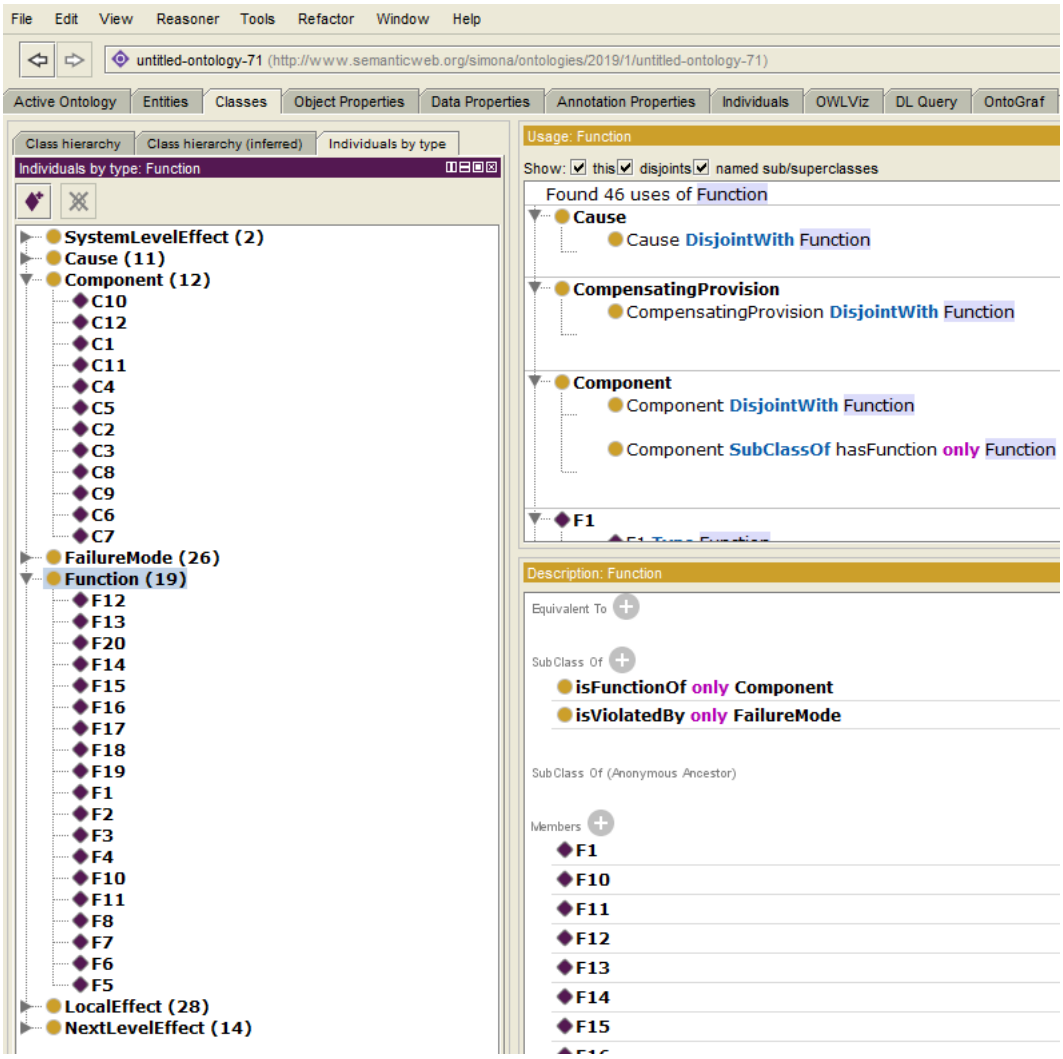


Obrázek 17: Ontograf (thing-věc; component-komponent; failure mode-poruchový stav; function-funkce; compensating provision-protiopatření; cause-příčina; effect-efekt; local effect-lokální efekt; next level effect-efekt další úrovně; engine level effect-efekt na úrovni motoru)

Třída *EngineLevelEffect*-efekt na motorové úrovni v mém případě nemá smysl, protože výstupem má být vliv na systémovou funkci. Důležité ale je, že je to efekt nejvyšší úrovně, stejně jako systémový efekt, který mě zajímá. Jde tedy jen o název, ten z důvodu praktičnosti a možného dalšího využití bude přejmenován na *SystemLevelEffect*.

Postup tvorby FMEA je velice podobný jako při tvorbě tradiční metodou, jen prostředí je jiné. Jako první krok je opět studium systému. Poté je jednoduché určit komponenty a jejich funkce.

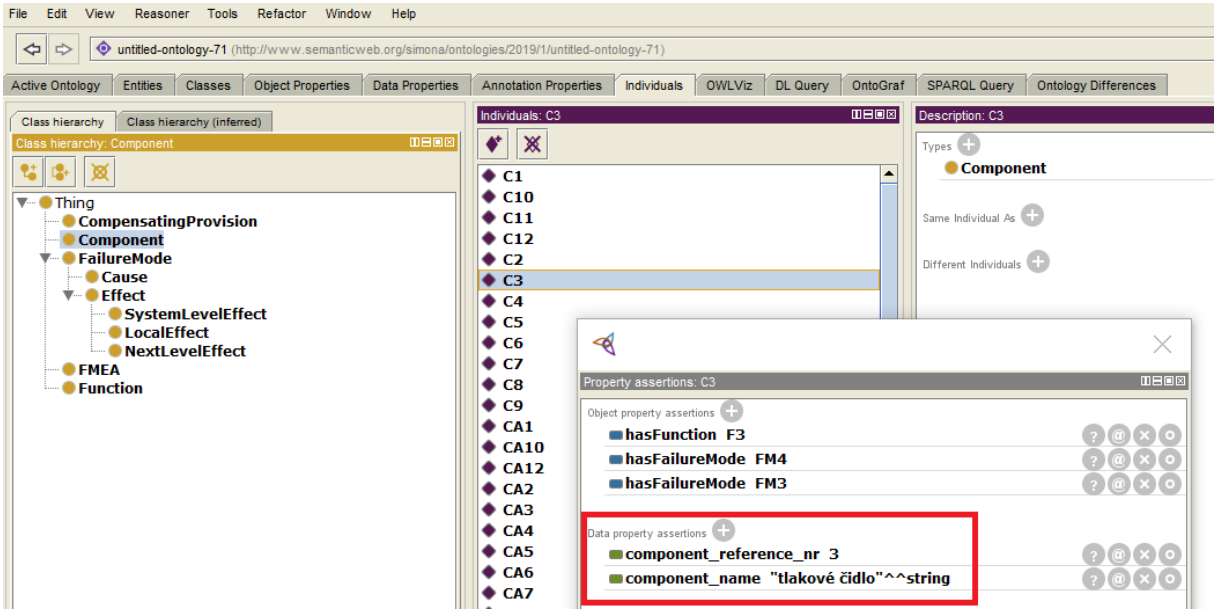
Na obrázku 18 je ukázáno, že data typu *Component* jsou reprezentována řadou C a data typu *Function* řadou F.



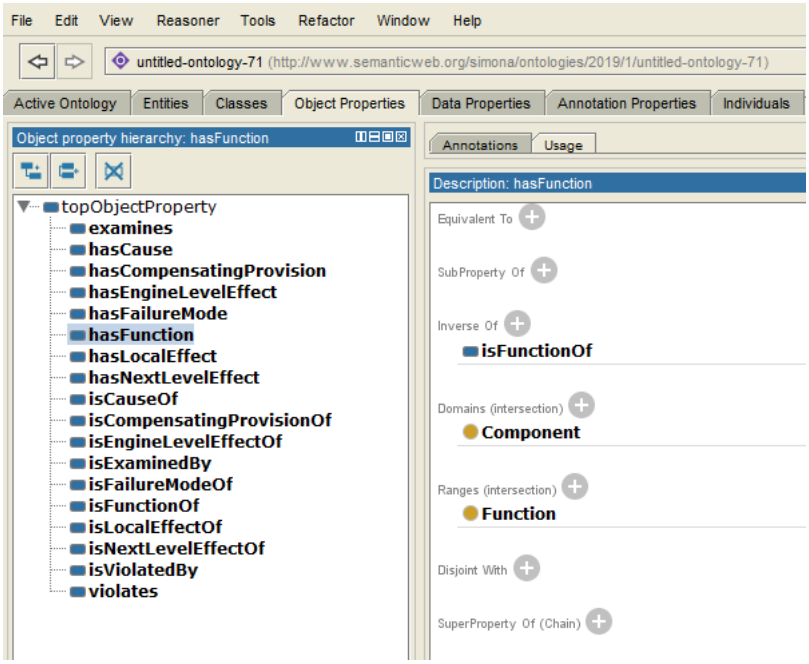
Obrázek 18: Komponenty a funkce

Při práci s ontologickým editorem je důležité dbát na správné definování dat. Při vytvoření instance se vždy nejprve definuje typ, tedy pod jakou třídu nová instance patří. U tříd definujeme vlastnosti a jak bylo zmíněno v kapitole o tvorbě ontologií, instance jsou nositeli těchto vlastností. Je tedy potřeba zadat i tyto vlastnosti. Příklad je uveden na obrázku 19. Je vybrána instance C3, která je typu *Component*. Komponent má definované vlastnosti *component_referenc_nr* – referenční číslo, u instance C3 je to číslo 3, a *component_name* – název komponentu, u instance C3 je to tlakové čidlo.

Stejně důležité jako definovat u instance typ a vlastnosti je vložit vazby mezi instancemi. Ty kopírují definované vazby mezi třídami. Všechny definované vazby původní ontologie jsou na obrázku 20. Jako příklad byla vybrána vazba *hasFuntion*-má funkci. Ta vychází z třídy *Component* a končí ve třídě *Funtion*. Tedy celá vazba je *komponent má funkci*. Můžeme vidět i definování inverzní funkce jako *funkce je funkcí komponentu*.

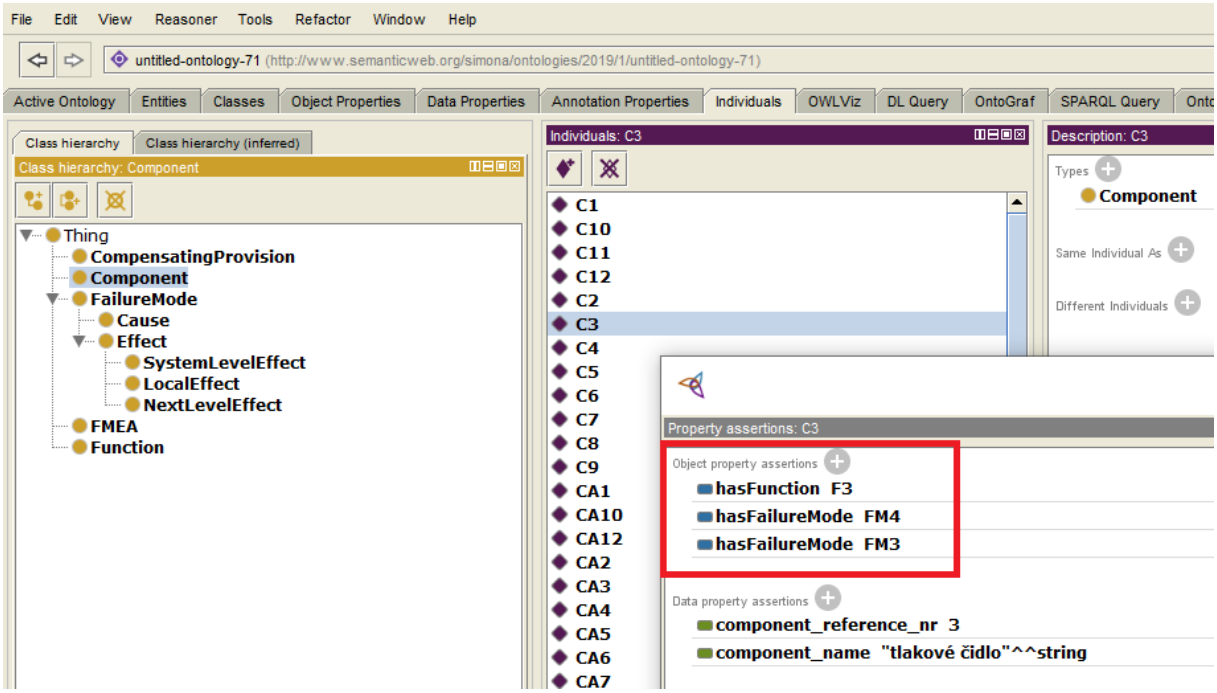


Obrázek 19: Vlastnosti instancí



Obrázek 20: Vazby mezi třídami

Na obrázku 21 už je vidět definovaná vazba mezi konkrétními instancemi. Instance C3 typu *Component* má vazbu *hasFuntion*-má funkci na instanci F3 a vazbu *hasFaliureMode*-má poruchový stav na instance FM3 a FM4.



Obrázek 21: Vazby instancí

Je velice důležité tyto vazby definovat správně. Pokud bychom např. již zmíněnou vazbu *hasFuntion* definovanou mezi třídami *Component* a *Fuction* definovali omylem např. mezi třídami *Component* a *Cause*, celá ontologie by byla nekonzistentní a hlásila by chybu.

Práci usnadňuje funkce Protégé *reasoner*. Tato funkce po zapnutí zkontroluje konzistenci ontologie, zkontroluje vazby mezi třídami a instancemi. Dokáže doplnit chybějící vazby, např. nedoplněnou inverzní funkci. V případě, že nalezne nějakou chybu, určí místo vzniku a ve výstupu se objeví seznam problematických implikací, který je třeba projít, aby se chyba odstranila.

Stejným způsobem popsaným výše byla do programu vložena všechna data. Na základě vytvořených tříd vazeb a vlastností v původní ontologii Simony Bolčekové byl vytvořen soubor ve formátu OWL.

3.2.2 Apache Jena Fuseki

Protégé je velmi užitečná aplikace při vývoji ontologie, umožňuje přehledné prostředí při práci s třídami a instancemi, avšak nedokáže exportovat zpracovaná data ve formátu, který by byl uživatelsky příjemný a snadno čitelný, jako je např. formát XLSX nebo XML,

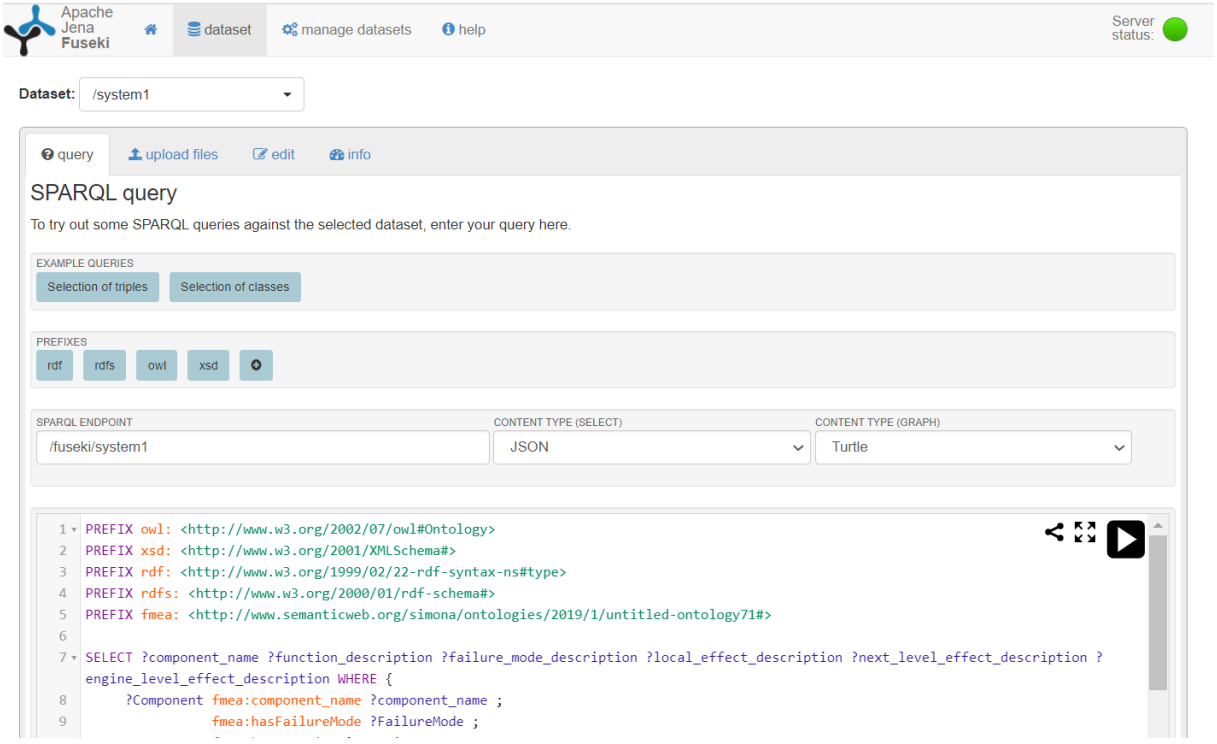
který je zpracovatelný programem Excel, jehož prostředí bylo použito i pro tvorbu FMEA tradiční metodou.

Za tímto účelem bude použita webová aplikace od projektu Tomcat[4] nazvaná Apache Jena Fuseki.

Je to webová aplikace, která pracuje na principu JAVA a běží na servletu Tomcat. Umožňuje uživatelsky příjemné prostředí při práci se soubory formátu OWL. Umožňuje nahrání ontologie do paměti a dále její zpracování. Lze také získat konkrétní údaje z ontologie pomocí dotazování SPARQL.

SPARQL je sémantický dotazovací jazyk určený speciálně pro databáze, který je schopen načíst a manipulovat s daty ve formátu RDF. Výsledky lze stáhnout v několika formátech, z nichž jeden je CSV formát, který lze zobrazit v prohlížeči CSV nebo jednoduše převést do formátu XLS nebo XLSX a otevřít jako soubor Excel, čehož využila i tato práce. [20]

Obrázek 22 ukazuje prostředí Apache Jena Fuseki s vytvořeným datovým souborem pojmenovaným *System1*. Dále je zde příkazový řádek na zadání SPARQL dotazu.



Obrázek 22: Apache Jena Fuseki

[4] https://tomcat.apache.org/

3.3 Porovnání výsledků

V této části práce budu mezi sebou porovnávat výsledek FMEA vytvořené tradiční metodou a FMEA vytvořené pomocí ontologie.

Soubor vytvořený v nástroji Protégé pomocí původní ontologie FMEA byl nahrán do webové aplikace Apache Jena Fuseki. V její práci byl také vytvořen SPARQL dotaz za účelem získání efektů a failure modů v jejím systému olejového systému motoru v tradiční formě. Na nahraný soubor byl aplikován tento dotaz, avšak výsledek vykazoval oproti výsledku z postupu tradiční metodou (příloha 1) chybovost v některých aspektech. Na obrázku 23 je vidět část výsledku po aplikování tohoto původního dotazu. Červeně jsou zvýrazněná data, která jsou zde nadbytečná. Tedy jak lze vidět, k první funkci patří 3 failure mody, a druhé funkci jen jeden failure mode. Ve výsledku se ale všechny failure mody objevují u obou funkcí.

component_name	function_description	failure_mode_description
cowl anti-ice valve	propouští vzduch dále do systému	nereguluje tlak
		nezavírá se
		neotevírá se
		netěsní
	ruguluje tlak na max 50 psi	nereguluje tlak
		nezavírá se
		neotevírá se
		netěsní

Obrázek 23: Nedostatky v původním dotazování se nad ontologií FMEA

3.3.1 Navržené změny

Po zvážení chyb ve výsledku automaticky generované FMEA jsem zjistila, že tyto chyby byly způsobeny strukturou samotného dotazu. Původní dotaz je příliš jednoduchý a je vhodný pro menší systémy s menším počtem dat.

Výsledek vykazuje nejvíce chyb na úrovni *failure_mode_despcription*-popis failure modu. Původní dotaz identifikoval failure mody jen na základě komponentu. Problém nastal hlavě u větších komponentů s více jak jednou funkcí. Tam docházelo k situaci, kdy ne všechny failure mody komponentu ovlivňovaly všechny jeho funkce. Např. komponent *cowl anti-ice valve* má funkce *propouští vzduch dále do systému* a *reguluje*

tlak na 50 psi. K funkci *propouští vzduch dále do systému* patří failure mody *nezavírá se*, *neotevírá se*, *netěsní* a k funkci *reguluje tlak na 50 psi* failure mode *nereguluje tlak*. Ale dotaz přiřadil k oběma funkcím všechny failury mody, jak je vidět na obrázku 23.

Proto došlo v tomto místě k úpravě dotazu. Při změně byla využita koncepce samotné ontologie. V té má totiž třída *FailureMode* vazbu na *Component* (komponent má failure mode) ale i na třídu *Function* (failure mode ovlivňuje funkci). Dotaz byl tedy upraven do podoby, kde se identifikuje failure mode ke komponentu jen tehdy, je-li failure mode spojený s funkcí komponentu. Podoba nového SPARQL dotazu je na obrázku 24.

```
PREFIX owl: <http://www.w3.org/2002/07/owl#Ontology>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#type>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX fmea: <http://www.semanticweb.org/simona/ontologies/2019/1/untitled-ontology-71#>

SELECT ?component_name ?function_description ?failure_mode_description
?local_effect_description ?next_level_effect_description ?engine_level_effect_description
WHERE
{
?Component fmea:component_name ?component_name ;
           fmea:hasFailureMode ?FailureMode ;
           fmea:hasFunction ?Function .
?FailureMode fmea:failure_mode_description ?failure_mode_description ;
             fmea:hasLocalEffect ?LocalEffect ;
             fmea:violates ?Function .
?Function fmea:function_description ?function_description .
?LocalEffect fmea:local_effect_description ?local_effect_description ;
             fmea:hasNextLevelEffect ?NextLevelEffect .
?NextLevelEffect fmea:next_level_effect_description
?next_level_effect_description ;
                fmea:hasEngineLevelEffect ?EngineLevelEffect .
?EngineLevelEffect fmea:engine_level_effect_description
?engine_level_effect_description .
}
ORDER BY ?component_name ?Function ?FailureMode ?LocalEffect ?NextLevelEffect
```

Obrázek 24: Nový SPARQL dotaz

Prvních pět řádků obsahuje prefixy, které jsou nezbytné k deklarování použité ontologie, jazyka, kterým je kódována, a jeho syntax. Část „SELECT“ obsahuje dotazované informace. Část „WHERE“ definuje požadavky na vybrané informace.

Po aplikování nového dotazu na stejný nahraný soubor vykazoval výsledek menší chybovost než první, ale nepodařilo se odstranit veškeré chyby. Na obrázku 25 je část výsledku po aplikování nového dotazu. Červeně jsou opět vyznačena nadbytečná data.

Problém popisu failure mode tady již nenastává, avšak nadbytečná data se nyní objevují u popisu lokálního efektu. Tento problém nelze řešit změnou dotazu jako v minulém případě, protože lokální efekt má v ontologii pouze jednu vazbu, a to na failure mode (failure mód má lokální efekt).

Po uvážení nastalé situace jsem dospěla k závěru, že failure mode je natolik kritická třída v celé ontologii, že zde vzniká požadavek na jeho unikátnost. Např. definice failure modu *nefunguje*, je pro tento systém příliš obecná. Objevuje se kromě komponentu *cawl valve open senzor* u třech dalších komponentů. Proto dochází k nadbytečnému rozvinutí lokálního efektu, jak je vidět na obrázku 25.

component_name	function_description	failure_mode_description	local_effect_description
cawl valve open senzor	potvrzuje souhlasnou polohu eng anti-ice switch a cowl anti-ice valve	citlivé na špatnou hodnotu	špatná indikace zvýšeného tlaku v systému
			špatná indikace
		nefunguje	žádná indikace zvýšeného tlaku v systému
			nelze ovládat přívod vzduchu
			žádná indikace
	upozorňuje na nesouhlasnou polohu anti-ice switch a cowl anti-ice valve	citlivé na špatnou hodnotu	špatná indikace zvýšeného tlaku v systému
			špatná indikace
		nefunguje	žádná indikace zvýšeného tlaku v systému
			nelze ovládat přívod vzduchu
			nelze ovládat přívod vzduchu
			žádná indikace

Obrázek 25: Chybovost na pozici lokálního efektu

Došlo tedy k úpravě failure modů, tak aby u dvou různých komponentů nedocházelo k jejích shodě. Příklad této změny je uveden na failure modu *nefunguje* v tabulce 6.

Tabulka 6: Úprava failure modů

komponent	failure mode původně	nový failure mode
cowl anti-ice senzor	nefunguje	kontrolka nefunguje
eng anti-ice switch	nefunguje	přepínač nefunguje
cawl valve open senzor	nefunguje	nedetekuje polohu
tlakové čidlo	nefunguje	nedetekuje tlak

Tato úprava byla provedena jak v tabulce FMEA, která byla tvořena tradiční metodou, tak v programu Protégé proto, aby se ve výsledku výstupy shodovaly a nedocházelo k záměně identických instancí.

Po této úpravě byl znovu pomocí Jena Fuseki aplikován nový dotaz. Tato tabulka už se už od FMEA tvořené tradiční metodou liší pouze v jednom místě. Chyba je zobrazena na obrázku 26.

Chyba se vyskytla u komponentu *high stage valve+high stage regulator*, což je největší komponent v systému, má nejvíce funkcí a failure modů. Problém nelze řešit pomocí změny dotazu, protože jak již bylo zmíněno výše, lokální efekt má v ontologii vazbu pouze na failure mode. Problém u tohoto konkrétního komponentu je ten, že má u více svých funkcí stejné failure mody, které ale generují jiné lokální efekty. Např. funkce *odstavuje 9th stage zdroj při vysokorychlostním režimu motoru* má failure mode *nezavírá se* stejně jako funkce *zabraňuje vniknutí zpětného toku do zdroje 9th stage*. V prvním případě se lokální efekt rozvíjí do *průnik vzduchu přes high stage valve směrem do motoru*, v druhém případě *průnik vzduchu přes high stage valve směrem do systému*. Protože SPARQL dotaz tyto dvě situace nerozezná, přiřadí k failure modu oba zmíněné lokální efekty.

Tato chyba ale není závažná v kontextu celého systému, protože její působení je pouze v rámci jednoho komponentu. Nemíchají se sem lokální efekty jiných komponentů, a to díky předchozí úpravě požadavku na unikátnost failure modu u každého komponentu. Tudíž případné další chyby na úrovni efektu další úrovně nebo systémovém efektu se budou týkat zase původního komponentu. V mém případě se lokální efekty rozvíjejí do stejného systémového efektu a chyba se tak dále nebude šířit.

Tento problém lze dále řešit pomoci rozlišení problematických failure modů. Nejedná se ale rozlišení dvou failure modů u dvou různých komponentů, čehož bylo využito výše. Jedná se o rozlišení stejných failure modů pomocí kontextu, ve kterém se vyskytují. V tabulce 7 je tato úprava ukázána. V tomto případě došlo k úpravě pomocí fáze režimu motoru, během které se daný failure mode může vyskytnout.

Nyní vzniká konkrétnější požadavek na formu vkládaného failure modu. Nejedná se jen o požadavek na unikátnost v rámci komponentu, nyní je požadavek na unikátnost u každé definované funkce. Na takto upravenou ontologii byl opět aplikován dotaz. Nyní je výstup pomocí ontologie FMEA identický s tabulkou v příloze 1 a v ničem se neliší. Obě výsledné tabulky se po této úpravě shodují.

component_name	function_description	failure_mode_description	local_effect_description
high stage valve+high stage regulator	zajišťuje dostatečý přísun vzduchu do pneumatického systému při nízkorychlostních operacích motoru(do výkonu motoru cca 47%)	high stage valve netěsní	průnik vzduchu přes high stage ventil směrem do motoru
			průnik vzduchu přes high stage ventil směrem do systému
		high stage valve se neotevírá	zastavení přívodu vzduchu do systému přes hihg stage valve
	reguluje tlak vzduchu na 32psi	high stage regulator nereguluje tlak	vzduch procházíhigh stage valve+high stage regulator bez regulace tlaku
	odsatvuje 9th stage zdroj při vysokorychlostním režimu motoru (zavře se pokud tlak v systému je větší než 32psi)	high stage valve se nezavírá	průnik vzduchu přes high stage ventil směrem do motoru
			průnik vzduchu přes high stage ventil směrem do systému
		high stage valve netěsní	průnik vzduchu přes high stage ventil směrem do motoru
			průnik vzduchu přes high stage ventil směrem do systému
		high stage valve reaguje na nesprávný tlak	high stage valve zavře se při vyšším tlaku
			high stage valve zavře se při nižší tlaku
	zabraňuje vniknutí zpětného toku do zdroje 9th stage	high stage valve se nezavírá	průnik vzduchu přes high stage ventil směrem do motoru
			průnik vzduchu přes high stage ventil směrem do systému
		high stage valve netěsní	průnik vzduchu přes high stage ventil směrem do motoru
			průnik vzduchu přes high stage ventil směrem do systému

Obrázek 26: Chyba ve výsledné tabulce

Tabulka 7: Failure mody rozlišené pomocí kontextu

komponent	funkce	failure mode původně	nový failure mode
high stage valve+high stage regulator	odstavuje 9th stage zdroj při vysokorychlostním režimu letu	high stage valve se nazavírá	high stage valve se nezavře během přechodu z nízkorychlostního na vysokorychlostní režim letu
		high stage valve netěsní	high stage valve netěsní během nízkorychlostního režimu letu
	zabraňuje vniknutí zpětného toku do zdroje 9th stage	high stage valve se nazavírá	high stage valve je otevřený během vysokorychlostního režimu
		high stage valve netěsní	high stage valve netěsní během vysokokorychlostního režimu letu

4 Diskuse

Správné výsledky analýz spolehlivosti jsou velmi důležité a cenné pro všechny výrobce. Na základě výsledků lze zabránit možným nebezpečným poruchovým režimům, design systému může být zlepšen, nebo mohou být získány informace o kritických součástech. Pokud jde o tradiční přístup analýzy spolehlivosti, spolehlivostní inženýr potřebuje shromáždit všechny potřebné informace pro provedení analýzy, a to jsou informace obvykle dokumentovány jinými inženýry. Jak se design vyvíjí, kontinuální shromažďování informací a komunikace mezi inženýry se stává složitější, projekt se může stát neefektivním. Navíc předávání informací prostřednictvím řetězce osob může mít za následek horší kvalitu informací a jejich správnost. Ontologický přístup je jedna z možností jak zlepšit dnešní způsob provádění spolehlivostních analýz.

Nevýhoda ontologického přístupu je v samotném nástroji Protégé. Jeho používání vyžaduje pokročilé počítačové dovednosti a jeho prostředí se běžnému uživateli osobního počítače může zdát složité a nepřehledné. Osvojení si tohoto nástroje vyžaduje několik hodin cvičného tvoření ontologií.

Cílem praktické části bylo vygenerovat výslednou tabulku tvořenou pomocí ontologie tak, aby se shodovala s tabulkou tvořenou tradiční metodou. V mém případě se mohou eliminovat rozdíly v tabulkách způsobené rozdílným názvem některých instancí, protože obě tabulky tvořila jedna osoba. Pokud byla v jednom případě upravená instance, stalo se tak i v druhém případě, aby nedocházelo k záměně stejných instancí.

Výhodou tvorby FMEA pomocí ontologie je, že se všechny instance vkládají pouze jednou. Dále je možné opravovat jejich název a vazby mezi ostatními instancemi bez toho, aniž by se některá instance musela mazat nebo znova vytvářet. Navíc funkce *reasoner* automaticky odvozuje některé informace. To je proč ontologický přístup zajišťuje porozumění informacím a jejich konzistentnost i při spolupráci více pracovních skupin a minimalizuje tak riziko vzniku chyby.

Velkým přínosem je i úspora času. V mém případě nebyl potenciál nástroje Protégé úplně využitý. Nástroj totiž dokáže načítat data i z externích souborů. Tudíž by stála za povšimnutí výhoda ukládání informací o systémech výrobců letadel a jejich systémů v patřičném formátu pro případně budoucí využití podobného modelu a jeho implementaci do běžné praxe. V mém případě došlo ke vkládání dat do programu ručně, tudíž časová úspora nebyla taková, jak by mohla být. Nejvíce času zabralo samotné vytváření instancí a definování hodnot vlastností. Např. u třídy komponent to byly

vlastnosti *component_reference_nr – referenční číslo* a *component_name* – název komponentu. Při zadání 12 komponentů bylo ještě potřeba definovat 24 vlastností. Cekově bylo do programu vloženo 115 instancí. Pokud by program načetl data z externího souboru, kde by tyto informace byly uvedeny od expertů, kteří soubor vytvořili, např. od konstruktérů, program by tyto vlastnosti doplnil sám a ušetřil by tak dalším pracovním skupinám čas. Časová úspora u mě nebyla veliká, ale v porovnání s tvorbou FMEA tradiční metodou, kde bylo využito pouze prostředí Excel bez podpory programu určených pro tvorbu FMEA, stále znatelná. Při tvorbě bylo potřeba využít několik dílčích tabulek. Finální tabulka pak vznikla jejich spojením. V případě potřeby provést změnu, mnohdy muselo dojít dalšímu přeformátování celého dotčeného řádku, což práci velice ztěžovalo.

Jak již bylo zmíněno, do Protégé je nutné nahrát každou instanci pouze jednou. Tudíž, pokud se některá instance opakuje, vloží se právě jen jednou. Např. instance třídy *lokální efekt – propouští vzduch dále do systému* se v systému odmrazování vstupního hrdla motoru objevuje několikrát. Tato vlastnost bude o to více znatelná při analýze opravdu velkého systému, jako je například celý motor, kde se při velkém množství komponentů a failure modů efekty hodně opakují.

Výhodou této vlastnosti je jakási forma predikce. Pokud se v systému opakují lokální efekty, ontologie předpoví i opakování efektů dalších úrovní. Toto je ale také důvodem proč dílčí tabulky v této práce nejsou úplně shodné. Jak bylo prokázáno výše, v praxi může nastat situace, kdy shodné lokální efekty nevyvozují v různých situacích stejné efekty další úrovně. V tu chvíli program nabízí nesprávnou predikci. Pokud ale bude dodržena jedinečnost failure modu, jejíž důležitost také byla prokázána výše, rozvinutí chyby se omezí pouze na jeden komponent a nebude se šířit dále.

V mém případě se chyba zastavila na úrovni systémového efektu, oba indikované lokální efekty se vyvinuly ve stejný systémový efekt - *no effect*. Chyba tedy neměla vliv na správnost výsledku, finální efekt se shodoval u obou výsledných tabulek. Chyba stejného typu se nemusí vyskytnout jen na úrovní lokálního efektu, ale i u efektu další úrovně a finálního efektu. Povaha vazby v ontologii a SPARQL dotazu je totiž stejná jako u lokálního efektu. Zde má ale výskyt chyby menší pravděpodobnost, protože efektů další úrovně a finálních efektů bývá méně než lokálních efektu a méně často se stane, že by byl efekt o úroveň výše ve dvou různých cestách jiný. Proto považuji za nekritičtější místo v celém modelu indikaci lokálního efektu. Pro potvrzení tohoto mého tvrzení by bylo potřeba provést stejnou metodou několik desítek dalších analýz různých systémů, aby

se zjistila četnost výskytu této chyby, místo výskytu a také závažnost v kontextu celé analýzy. Tedy jak často se stalo, že se chyba vyskytne, kde se vyskytne, jak často se stane, že se chyba rozvine až do systémového efektu a jak závažná tato chyba je. Z toho by se dala vyvodit potřeba další modifikace modelu, nebo jen zběžná kontrola výsledků bezpečnostním analytikem.

Pozitivní je, že model nezapomíná označit kritické komponenty a failure mody. V nejhorším případě rozvinutí chyby „pouze" označí komponent nebo failure mode za kritický, i když kritický není. Naopak se to však nemůže stát.

Výše však byl navržen způsob, jak tuto chybu eliminovat pomocí dalšího rozlišení kontextem. V této práci se to týkalo pouze failure modu, pokud by se ale tento model aplikoval na jiné systémy, bylo by třeba hlídat unikátnost i dalších dat (lokální efekt, efekt další úrovně), protože i u nich se tato chyba může vyskytnout.

Rozhodnutí, jakou z těchto dvou možností využití modelu použít je na společnosti, příp. spolehlivostním analytikovi. První způsob umožňuje volnější způsob vkládání dat a jeho limitace je pouze v unikátnosti failure modů u každého komponentu. Ve výsledku je ale nutné počítat s rizikem výskytu chyby a je tedy nutná kontrola výsledku. Druhý způsob vyžaduje striktní dodržení unikátnosti vkládaných dat. Tvorba a vkládání takových dat do programu zabere více času, avšak ve výsledné tabulce si můžeme být jisti správností.

V závislosti na výše uvedených důvodech bych vytvořený model označila za použitelný v běžné praxi. Avšak s nutností zkušebního provozu na několika dalších systémech a s manuální kontrolou výsledku, s potenciálem dalšího zlepšení.

Závěr

Cílem této práce bylo vytvoření automatizovaného hodnocení spolehlivosti letadlových komponent metodiky FMEA. Tomuto účelu nejvíce vyhovuje ontologický přístup. Byl popsán současný stav a existující modely, které k problematice přistupují ontologickým řešením. Ze všech existujících modelů byl vybrán ten z práce *Reliability analysis of mechanical and lubrication system of an aircraft engine* Simony Bolčekové jako nejvíce vhodný pro účel této práce. Avšak tento model potřeboval další validaci, protože byl vytvořen jednomu systému na míru a na tento systém aplikován. Pro svojí práci jsem vybrala systém odmrazování vstupního hrdla motoru na letadle typu B737. Na tento systém byla nejprve vytvořena spolehlivostní analýza FMEA tvořená tradiční metodou a to proto, aby mohl být výsledek analýzy s ontologickým řešením s tímto porovnán. Stejná data byla nahrána do ontologického editoru Protégé a s pomocí webové aplikace Apache Jena Fuseki byl vygenerován výsledek ve formě tabulky. Tyto tabulky byly porovnány a bylo v nich nalezeno několik odchylek. Ty se však podařilo odstranit pomocí změny dotazu a vytvoření požadavku na unikátnost dat.

V práci byl modelován pouze jeden systém. Ontologie také nebyla ověřena implementací v reálném provozu a nebyla předložena žádnému spolehlivostnímu analytikovi, aby s ní pracoval a její vhodnost zhodnotil, a to proto, že tvorba této práce neprobíhala ve spolupráci s žádnou společností. Další limitací práce je, že FMEA nebyla vytvořena kompletně, nebylo hodnoceno RPN. Jak již bylo uvedeno výše, hodnota RPN závisí na společnosti nebo spolehlivostním analytikovi a lze doplnit i zpětně. Ontologie je plně připravená na práci s RPN, je schopna přijmout tato data. K tomu, aby bylo RPN součástí výsledné tabulky, stačí drobná úprava SPARQL dotazu.

Pro zajištění správnosti výsledku doporučuji dva možné přístupy pro práci s modelem. Za prvé přístup kdy je vkládání dat limitované pouze požadavkem na unikátnost failure modu mezi komponenty. Tento přístup ale vyžaduje zpětnou kontrolu výsledné tabulky, protože je zde riziko vzniku chyby. Za druhé přístup při kterém je vkládání dat limitované unikátností dat. Výsledek tohoto přístupu je ale správný.

Ontologický přístup ke spolehlivostním analýzám v sobě má velký potenciál. Eliminuje limitace tvorby těchto analýz tradiční metodou jako je časová náročnost, nebo riziko vzniku chyby. Elektronické prostředí ontologického editoru umožňuje i automatické nahrávání dat, např. ve formě databáze. Je zde také prostor pro vytvoření ontologie, která by spojovala více spolehlivostních analýz do jednoho modelu. Například spojení

analýzy FMEA a FTA by rapidně snížila čas potřebný pro celkové hodnocení letadlového celku. Takový model by totiž vyžadoval nahrání pouze jedné sady dat.

Zdroje

1. HOLUB, R.; VINTR, Z. Spolehlivost letadlové techniky: elektronická skripta. Brno: VUT FSI, 2001. pp. 233 http://lu.fme.vutbr.cz/files/SpolehlivostLetadloveTechniky.pdf
2. LEVESON, NANCY G. a JOHN P. THOMAS. STPA Hanbook [online]. MIT, 2018 [cit. 2020-08-10]. Dostupné z: https://psas.scripts.mit.edu/home/get_file.php?name=STPA_handbook.pdf
3. What is a Fault Tree Analysis (FTA)? - The Beginner's Guide. Edrawsoft [online]. [cit. 2020-07-13]. Dostupné z: https://www.edrawsoft.com/what-is-fault-tree-analysis.html
4. BERNATÍK, Aleš. *Prevence závažných havárií I*. Ostrava: Sdružení požárního a bezpečnostního inženýrství, 2006. ISBN 8086634892
5. SUBRIADI, Apol Pribadi. The consistency analysis of failure mode and effect analysis (FMEA) in information technology risk assessment. Heliyon [online]. 2020 [cit. 2020-08-02]. DOI: 10.1016/j.heliyon. 2020.e03161. ISSN 24058440.
6. FMEA – Vyhodnocení rizik. *Lean6SIgma.cz* [online]. [cit. 2020-07-14]. Dostupné z: https://lean6sigma.cz/fmea/
7. Ontology: metaphysics. Britannica.com [online]. [cit. 2020-07-15]. Dostupné z: https://www.britannica.com/topic/ontology-metaphysics
8. Merriam-Webster [online]. [cit. 2020-08-09]. Dostupné z: https://www.merriam-webster.com/
9. Ontology Development: A Guide to Creating Your First Ontology. Protege.stanford.edu [online]. Stanford University [cit. 2020-07-15]. Dostupné z: https://protege.stanford.edu/publications/ontology_development/ontology101-noy-mcguinness.html
10. AMBLER, Scott W. The Elements of UML(TM) 2.0 Style. New York: Cambridge University Press, 2005. ISBN 0521616786. 9780521616782.
11. BOLČEKOVÁ, Simona. *Reliability analysis of mechanical and lubrication system of an aircraft engine*. Prague, 2019. Diploma thesis. CZECH TECHNICAL UNIVERSITY IN PRAGUE.
12. REHMAN, Z. a C. V. KIFOR. An Ontology to Support Semantic Management of FMEA Knowledge. *INTERNATIONAL JOURNAL OF COMPUTERS COMMUNICATIONS & CONTROL*. 2016. ISSN ISSN 1841-9836.

13. SCHINDEL, William. *Failure Risk Analysis: Insights from Model-Based Systems Engineering* [online]. Chicago: INCOSE Symposium, 2010 [cit. 2020-07-22]. Dostupné z: http://www.omgwiki.org/MBSE/lib/exe/fetch.php?media=mbse:patterns:pbse__wg__meeting__dec__17__2014__attachment__2.pdf
14. CASTET, Jean-Francois. Failure Analysis and Products in a Model-Based Environment. IEEE Aerospace Conference Proceedings [online]. 2018 [cit. 2020-08-02]. DOI: 10.1109/AERO.2018.8396736. ISSN 1095323X. Dostupné z: https://ieeexplore.ieee.org/document/8396736
15. Aircraft Ice Protection Systems. Https://www.skybrary.aero/index.php/Aircraft__Ice__Protection__Systems [online]. [cit. 2020-07-18]. Dostupné z: https://www.skybrary.aero/index.php/Aircraft__Ice__Protection__Systems
16. B737 NG Air Systems. *Slideshare.net* [online]. [cit. 2020-07-21]. Dostupné z: https://www.slideshare.net/theoryce/b737-ng-air-systems
17. MOIR, Ian. Aircraft Pneumatic Subsystems. *Encyclopedia of Aerospace Engineering* [online]. 2013 [cit. 2020-08-07]. DOI: 10.1002/9780470686652.eae615. Dostupné z: https://doi.org/10.1002/9780470686652.eae615
18. Boeing b737 300-500 system diagrams. Avsoft.com [online]. [cit. 2020-07-21]. Dostupné z: https://www.avsoft.com/product/boeing-b737-300-500-system-diagrams/ssd-b73a-3/
19. 737 SYSTEMS SCHEMATICS POWERPLANT. *Vdocuments.mx* [online]. [cit. 2020-07-21]. Dostupné z: https://vdocuments.mx/737-systems-schematics-powerplant.html
20. SPARQL Query Language for RDF. *W3.org* [online]. [cit. 2020-08-06]. Dostupné z: https://www.w3.org/TR/rdf-sparql-query/

Příloha 1: Výsledná tabulka FMEA tvořená tradiční metodou

KOMPONENT	FUNKCE	FAILURE MODE	LOKÁLNÍ EFEKT	EFEKT DALŠÍ ÚROVNĚ	FINÁLNÍ EFEKT
engine cowl anti-ice	zahřívá vstupní hrdlo motoru	trhlina na vstupním hrdle	únik vzduchu ze systému trlinou na vstupním hrdle	snížení objemu vzduchu v odmrazovací části	nefunkčnost odmrazovacího systému
výduch	odvádí vzduch ze systému ven po zahřátí náběžné hrany motoru	ucpání cizím předmětem z venčí	vzduch ze systému výduchem neodchází	zvýšení tlaku v systému	no efekt
tlakové čidlo	citlivé na hodnotu tlaku nad 65 psi	citlivé na špatnou hodnotu	špatná indikace detekce tlaku v systému	zvýšení tlaku v systému	no efekt
		nedetekuje tlak	žádná indikace detekce tlaku v systému	zvýšení tlaku v systému	no efekt
cowl anti-ice senzor	upozorňuje na zvýšený tlak vzduchu v systému rozsvícením	kontrolka nefunguje	žádná indikace zvýšeného tlaku v systému	zvýšení tlaku v systému	no efekt
cowl anti-ice valve	propouští vzduch dále do systému	cowl anti-ice valve se nezavírá	nelze uzavřít přívod vzduchu	nelze vypnout odmrazovaní	no efekt
		cowl anti-ice valve netěsní	průnik vzduchu ant-ice ventilem	snížená funkce	no efekt
		cowl anti-ice valve se neotevírá	nelze otevřít přívod vzduchu	nefunknční odmrazovaní náběžné hrany	nefunkčnost odmrazovacího systému
	reguluje tlak na max 50 psi	nereguluje tlak	vzduch prochází anti-ice ventilem bez regulace tlaku	zvýšení tlaku v systému	no efekt
eng anti-ice switch	ovládá polohu cowl anti-ice valve	přepínač nefunguje	přepínačem nelze ovládat přívod vzduchu	nefunknční odmrazovaní náběžné hrany	nefunkčnost odmrazovacího systému
cowl valve open senzor	potvrzuje souhlasnou polohu a upozorňuje na nesouhlasnou eng anti-ice switch a cowl anti-ice valve	citlivé na špatnou polohu	špatná indikace polohy cowl valve	no efekt	no efekt
		nedetekuje polohu	žádná indikace polohy cowl valve	no efekt	no efekt

KOMPONENT	FUNKCE	FAILURE MODE	LOKÁLNÍ EFEKT	EFEKT DALŠÍ ÚROVNĚ	FINÁLNÍ EFEKT
zdroj 5th stage	odebírá ohřátý vzduch z motoru na pozici 5. stupně kompresoru	na pozici 5. stupně není odebírán z motoru žádný vzduch	zastavení přívodu vzduchu do systému na pozici 5. stupně	nedojde odstavení přívodu z 9th stage	no efekt
		na pozici 5. stupně odebírá menší množství vzduchu	snížení přívodu vzduchu do systému na pozici 5. stupně	k odstavení přívodu z 9th stage dojde později	no efekt
zdroj 9th stage	odebírá ohřátý vzduch z motoru na pozici 9. stupně kompresoru	na pozici 9. stupně není odebírán z motoru žádný vzduch	zastavení přívodu vzduchu do systému na pozici 9. stupně	systém bude bez vzduchu než zdroj 5th stage získá dostatečný tlak	no efekt
		na pozici 9. stupně odebírá menší množství vzduchu	snížení přívodu vzduchu do systému na pozici 9. stupně	v systému bude dočasně měnší množství vzduchu	no efekt
check valve	zabraňuje vniknutí zpětného toku do zdroje 5th stage	check valve se nezavírá	průnik vzduchu check valve směrem do motoru	vniknutí zpětného toku do motoru	no efekt
		check valve netěsní	průnik vzduchu check valve směrem do motoru	vniknutí zpětného toku do motoru	no efekt
	jednosměrně propouští vzduch	check valve se neotevírá	zastavení přívodu vzduchu do systému přes check valve	nedojde odstavení přívodu z 9th stage	no efekt
		check valve netěsní	průnik vzduchu check valve směrem do systému	do systému jde vzduch z obou zdrojů	no efekt
			průnik vzduchu check valve směrem do motoru	vniknutí zpětného toku do motoru	no efekt
	otevře se pokud tlak vzduchu z 5th stage je vetší než 32 psi	check valve se neotevírá	zastavení přívodu vzduchu do systému přes check valve	nedojde odstavení přívodu z 9th stage	no efekt
		check valve reaguje na nesprávný tlak	check valve se otevře při nižším tlaku	k odstavení přívodu z 9th stage dojde dřív	no efekt
			check valve se otevře při vyšším tlaku	k odstavení přívodu z 9th stage dojde později	no efekt

KOMPONENT	FUNKCE	FAILURE MODE	LOKÁLNÍ EFEKT	EFEKT DALŠÍ ÚROVNĚ	FINÁLNÍ EFEKT
high stage valve+high stage regulator	zabraňuje vniknutí zpětného toku do zdroje 9th stage	high stage valve je otevřený během vysokorychlostního režimu	průnik vzduchu přes high stage ventil směrem do motoru	vniknutí zpětného toku do motoru	no efekt
		high stage valve netěsní během vysokokorychlostního	průnik vzduchu přes high stage ventil směrem do motoru	vniknutí zpětného toku do motoru	no efekt
	zajišťuje dostatečý přísun vzduchu do pneumatického systému při nízkorychlostním režimu motoru(do výkonu motoru cca 47%)	high stage valve netěsní	průnik vzduchu přes high stage ventil směrem do systému	do systému jde vzduch z obou zdrojů	no efekt
			průnik vzduchu přes high stage ventil směrem do motoru	vniknutí zpětného toku do motoru	no efekt
		high stage valve se neotevírá	zastavení přívodu vzduchu do systému přes hihg stage valve	systém bude bez vzduchu než zdroj 5th stage získá dostatečný tlak	no efekt
	reguluje tlak vzduchu na 32psi	high stage regulator nereguluje tlak	vzduch procházíhigh stage valve+high stage regulator bez regulace tlaku	zvýšení tlaku v systému	no efekt
	odsatvuje 9th stage zdroj při vysokorychlostním režimu motoru (zavře se pokud tlak v systému je větší než 32psi)	high stage valve se nezavře během přechodu z nízkorychlostního na na vysokorychlostní režim letu	průnik vzduchu přes high stage ventil směrem do systému	do systému jde vzduch z obou zdrojů	no efekt
		high stage valve netěsní během nízkorychlostního režimu letu	průnik vzduchu přes high stage ventil směrem do systému	do systému jde vzduch z obou zdrojů	no efekt
		high stage valve reaguje na nesprávný tlak	high stage valve zavře se při vyšším tlaku	k odstavení přívodu z 9th stage dojde později	no efekt
			high stage valve zavře se při nižší tlaku	k odstavení přívodu z 9th stage dojde dřív	no efekt
trubka	propojuje jednotlivé části systému	trhlina	únik vzduchu ze systému	snížení objemu vzduchu v odmrazovací části	nefunkčnost odmrazovacího systému
	rozvádí vzduch		únik vzduchu ze systému	snížení objemu vzduchu v odmrazovací části	nefunkčnost odmrazovacího systému