Západočeská univerzita v Plzni
Fakulta aplikovaných věd
Katedra informatiky a výpočetní techniky

Bakalářská práce

Návrhový modul pro výpočet pevnosti šroubovaných spojů

Plzeň 2016 Jiří Hankovec

Místo této strany bude zadání práce.

Prohlášení

Prohlašuji, že jsem bakalářskou práci vypracoval samostatně a výhradně s použitím citovaných pramenů.

V Plzni dne 21. června 2016

Jiří Hankovec

Abstract

The main goal of this work is to design and implement interactive graphic designer for bolted joints. The application will be used to create a graphical design of a bolted joint and the application will be able to save this graphical design of a bolted joint into some data file. The data file will be used by software for firmness computation of a bolted joint.

Abstrakt

Hlavním cílem této práce je navrhnout a implementovat interaktivní vizuální návrhový modul šroubovaných spojů. Navržená aplikace bude umět vytvořit vizuální návrh šroubovaného spoje a tento vizuální návrh se bude moci uložit do nějakého datového souboru. Datový soubor bude dále využit výpočetním softwarem pro výpočet pevnosti šroubovaného spoje.

Obsah

1 Úvod

Cílem práce je naprogramovat interaktivní vizuální návrhový modul (vlastně jednoduchý *CAD*[1] systém), jehož výstupem bude datový soubor pro výpočetní software, který provádí výpočet pevnosti šroubovaného spoje.

Úkolem první části této práce je čtenáře seznámit se základními pojmy a ovládáním *CAD* aplikací. Existuje mnoho *CAD* aplikací, některé jsou obecné, jiné jsou specializované na různá odvětví (strojírenství, stavebnictví a architektura, elektrotechnika, atd.).

Další částí této práce je prozkoumat možnosti využití volně dostupných knihoven pro tvorbu uživatelských rozhraní. Poté navrhnout a implementovat, ve zvoleném jazyce a zvoleném vývojovém prostředí, aplikaci pro vizuální návrh šroubovaného spoje.

[1] *Computer Aided Design* – počítačem podporované projektování

2 Ovládání CAD aplikací

V této kapitole se seznámíme s obecnými principy ovládání *CAD* aplikací a s několika existujícími *CAD* aplikacemi a jejich ovládáním.

2.1 Pojmy

Níže jsou uvedeny základní pojmy a zkratky, které jsou důležité k porozumění celé této práci.

2.1.1 CAD

CAD [1] (*computer–aided design* nebo *computer–aided drafting*) je počítačem podporované projektování nebo kreslení, jedná se o systém, který se používá pro vytváření, úpravu nebo analýzu návrhů plošných výkresů a modelování objektů (např. strojních zařízení, stavebních celků, atd.). Jedná se o grafické uživatelské rozhraní, ve kterém uživatel projektuje namísto rýsovacího prkna.

CAD systémy mohou být *2D* i *3D*. Používají se v mnoha oborech, ve strojírenství, v architektuře, v elektrotechnice a v mnoha dalších. Ale jsou to pouze nástroje, proto samotná znalost libovolného *CAD* systému nezaručí, že uživatel systému bude dobrý konstruktér.

2.1.2 Uživatelské rozhraní

Uživatelské rozhraní [2] je způsob, jakým uživatel komunikuje s určitou aplikací. Uživatel může určitou aplikaci ovládat pomocí různých vstupů (např. kliknutí myší), reakcí na uživatelův vstup je buďto nějaká reakce aplikace nebo nějaký výstup.

Uživatelská rozhraní se dělí na rozhraní příkazové řádky, grafické uživatelské rozhraní, textové uživatelské rozhraní a další. Rozhraní příkazové řádky (*CLI*, *Command Line Interface*) používá na interakci s uživatelem pouze příkazovou řádku.

Grafické uživatelské rozhraní [3] (*GUI*, *Graphical User Interface*). Uživatel při zadávání vstupů používá myš, klávesnici a grafické vstupní prvky (ikony,

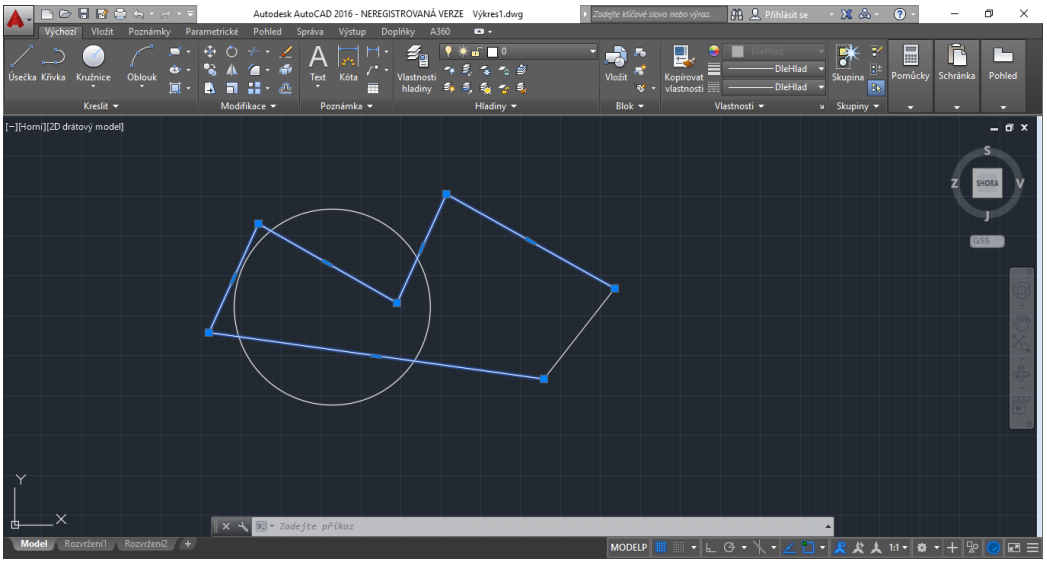
tlačítka, textová pole, formuláře, atd.). *GUI* se nepoužívají pouze v počítačích, ale používají se například také v herních zařízeních, přenosných hudebních přehrávačích, televizích, dokonce i v domácích spotřebičích a v dalších zařízeních.

2.2 Existující software

V této kapitole se seznámíme s několika existujícími *CAD* aplikacemi a jejich ovládáním. Je zde uvedeno pouze několik aplikací, *CAD* aplikací existuje velké množství. Zde uvedené *CAD* aplikace patří mezi nejznámější *CAD* aplikace.

2.2.1 AutoCAD

AutoCAD [4] je jeden z nejznámějších *CAD* systémů vyvinutý firmou *Autodesk*, pracuje v operačním systému OS Windows, dříve pracoval i v jiných operačních systémech, ale od roku 1994 pokračoval vývoj pouze pro OS Windows. První verze AutoCADu je z roku 1982.



Obrázek 2.1: Ukázka softwaru AutoCAD

AutoCAD se používá pro *2D* i *3D* projektování a konstruování.

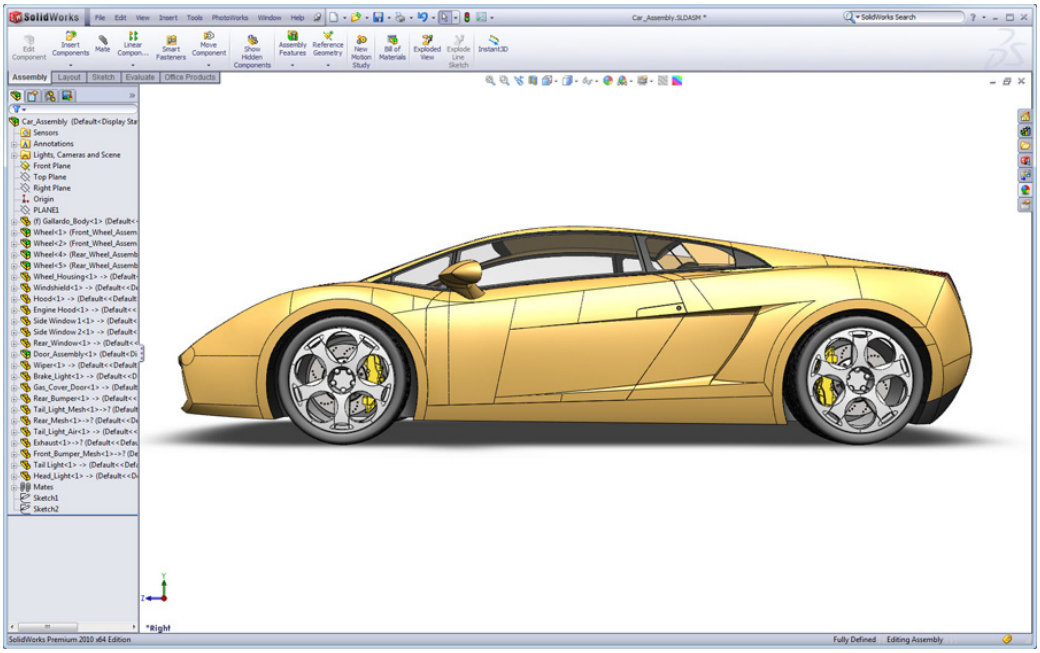
Ovládání

Příkazy můžeme zadávat několika způsoby. Je zde klasické hlavní menu, které nám především umožňuje pracovat se soubory. Poté zde máme několik

záložek s panely nástrojů, pomocí kterých lze s programem pracovat. Program můžeme ovládat také pomocí vložené příkazové řádky. Nakonec je zde kreslící (modelovací) plocha.

2.2.2 SolidWorks

SolidWorks [5] je *CAD* systém na *3D* projektování a konstruování ve strojírenském odvětví, pracující v OS Windows. Firma *SolidWorks Corporation*, která vyvinula tento software, byla založena v roce 1993. Tento software je k dispozici v několika verzích, ale všechny jsou placené.



Obrázek 2.2: Ukázka softwaru SolidWorks

Ovládání

Příkazy v tomto softwaru můžeme zadávat podobně jako u AutoCAdu, je zde hlavní menu, kreslící plocha, také jsou zde panely nástrojů. Na rozdíl od AutoCADu však nemá vestavěnou příkazovou řádku.

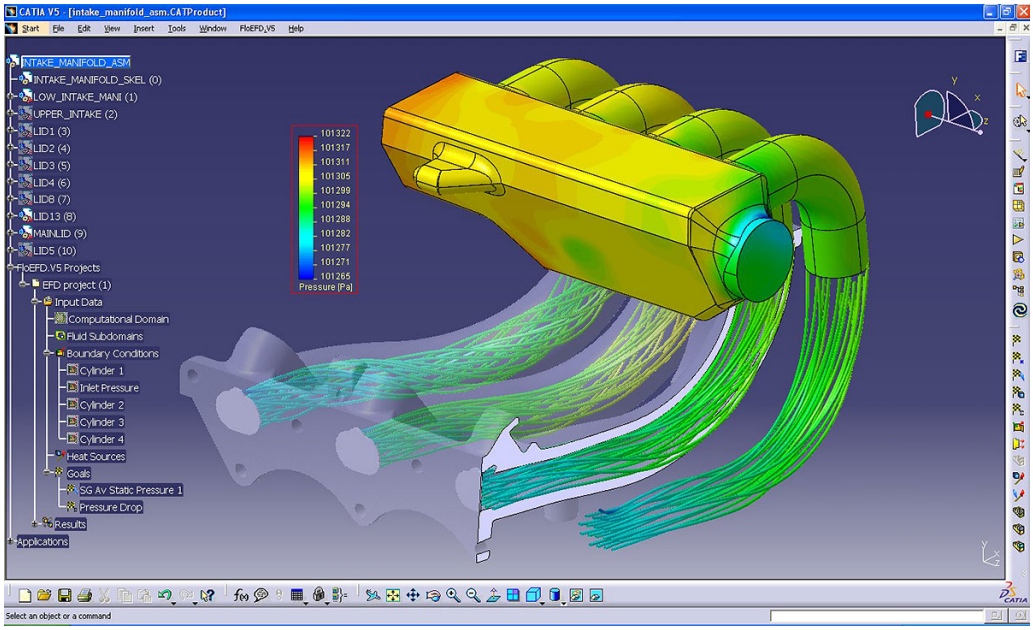
2.2.3 CATIA

CATIA [6] (*Computer Aided Three–dimensional Interactive Application*) je aplikace napsaná v jazyce `C++`. Aplikace byla vyvinutá francouzskou firmou *Dassault Systèmes*, vývoj aplikace začal v roce 1977 a v roce 1981 se začala

prodávat.

CATIA je také jako SolidWorks *CAD* systém na *3D* projektování a konstruování ve strojírenském oboru, využívaný převážně v leteckém a automobilovém průmyslu. Tento software může pracovat pod OS Windows nebo Linux.

Tento software využívají například světoznámé společnosti *AirBus Industries*, *Boeing Company* a jiné další společnosti.



Obrázek 2.3: Ukázka softwaru CATIA

2.3 Obecné ovládání CAD aplikací

V této kapitole je uvedeno několik základních ovládacích prvků *CAD* aplikací. Ve většině *CAD* aplikací jsou tyto ovládací prvky velmi podobné. Grafická uživatelská rozhraní *CAD* aplikací se mohou ovládat pomocí klávesnice a myši.

Velká většina *CAD* systémů se skládá z kreslící (rýsovací) plochy, na které je umístěna soustava souřadnic, buďto pouze dvojrozměrná nebo i trojrozměrná (u některých *CADů* možnost volby). Dále obsahují několik panelů nástrojů s různou funkcionalitou. Další důležitá komponenta je hlavní menu, které umožňuje hlavně práci se soubory.

- *ESC* – přerušuje probíhající akci nebo ruší vykonanou akci.
- „Scrollováním“ kolečkem myši na kreslící ploše se buďto přibližuje nebo oddaluje.
- Výběr z panelů nástrojů se provádí kliknutím levého tlačítka myši, přepínání mezi záložkami se provádí též levým tlačítkem myši.
- Po výběru nějakého objektu, který chceme vykreslit, kliknutím a tahem levým tlačítkem myši na kreslící plochu objekt vykreslíme.
- Kliknutí pravým tlačítkem myši na kreslící plochu nabízí další funkce, kliknutím pravým tlačítkem na vytvořené objekty nám umožňuje další práci s objekty (sloučení více objektů do jednoho, zjištění údajů o daném objektu, atd.).
- Pokud máme zvolený nástroj na výběr objektů, můžeme tahem se stisknutým levým tlačítkem myši vybrat více objektů najednou.

3 Analýza problému

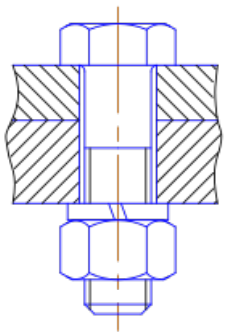
Katedra konstruování strojů Fakulty strojní ZČU požaduje návrhovou aplikaci, v podstatě jednoduchou *2D CAD* aplikaci, která bude umět vytvořit vizuální návrh šroubovaného spoje. Navržené šroubované spoje se budou moci uložit do nějakého datového souboru. Z datového souboru bude možné daný šroubovaný spoj načíst a vykreslit na kreslící plochu.

Tento datový soubor bude moci Katedra konstruování strojů Fakulty strojní ZČU dále použít pro svůj již existující výpočetní software, který provádí výpočet pevnosti šroubovaného spoje. Tento software sice umožňuje si daný šroubovaný spoj vytvořit, ale musí se ručně „bod po bodu“ vytvořit geometrie šroubovaného spoje, což je velmi zdlouhavé.

3.1 Šroubovaný spoj

Bylo by asi dobré stručně vysvětlit, co to vlastně šroubovaný spoj [7] je. Jedná se o základní konstrukční prvek, který se používá ke spojení dvou nebo více konstrukcí pomocí spojovacích prvků. Spojovací prvky mohou být různé, například šroub s maticí, kolík, péro nebo jiná obecná zátěž.

Tyto spoje můžeme dělit na spoje rozebiratelné a nerozebiratelné. Všechny výše vyjmenované spoje patří mezi spoje rozebiratelné. Rozebiratelné jsou proto, že je lze rozebrat a opět složit bez poškození spojovacího článku.



Obrázek 3.1: Ukázka šroubovaného spoje

3.2 Existující software

Existující software Katedry konstruování strojů Fakulty strojní ZČU pro výpočet pevností šroubovaných spojů pracuje s objekty sestavených z lineárních úseků, kružnic nebo jejich oblouků.

Objekt sestavený z lineárních úseků je tvořen několika vrcholy a přímkami vedenými mezi sousedními vrcholy. Proto lze spojitý objekt popsat N rovnicemi přímek a nespojitý objekt $N-1$ rovnicemi přímek, kde N představuje počet vrcholů objektu.

Rovnice přímky směrnicový tvar:

$$y = kx + q, \tag{3.1}$$

kde q je úsek na ose y a $k = \tan\alpha$, přičemž α je úhel, který svírá přímka s kladnou poloosou x. Ovšem při znalosti dvou vrcholů (bodů) využijeme rovnici přímky dané dvěma body $P_1[x_1, y_1]$ a $P_2[x_2, y_2]$.

Rovnice přímky dané dvěma body:

$$\frac{y - y_1}{x - x_1} = \frac{y_2 - y_1}{x_2 - x_1}, \tag{3.2}$$

kde x_1 a y_1 jsou souřadnice prvního vrcholu (bodu) a x_2 a y_2 jsou souřadnice druhého vrcholu (bodu). Více informací o rovnicích přímky lze nalézt například v [16].

Objekt sestavený z kružnic nebo jejich oblouků lze popsat rovnicí kružnice:

$$(x - x_0)^2 + (y - y_0)^2 = r^2, \tag{3.3}$$

kde x_0 a y_0 jsou souřadnice středu kružnice $S[x_0, y_0]$ a r je poloměr kružnice.

Objekty mohou být rovněž vytvořeny kombinací objektů sestavených z lineárních úseků a objektů sestavených kružnic nebo jejich oblouků.

3.3 Volba primitiv

Důležité je si správně zvolit základní primitiva a jejich reprezentaci. Zadavatel požaduje, aby aplikace uměla vykreslovat elipsy, kruhy, mezikruží,

obdélníky, čtverce, polygony a jejich různé kombinace. Proto by bylo asi výhodné si jako primitiva zvolit elipsu (kruh je jen speciální případ elipsy a mezikruží se dá zkonstruovat pomocí kruhu a díry v podobě kruhu), obdélník a polygon, tyto objekty budou reprezentovat plochu ke spojování. Zbývající primitiva budou spojovací prvky – šroub, kolík, péro ve směru osy X, péro ve směru osy Y a obecná zátěž.

3.3.1 Reprezentace objektů

Velmi důležité bude také zvolit vhodnou reprezentaci jednotlivých objektů, a dále jaké informace si o jednotlivých objektech budeme pamatovat a jaké informace si budeme muset spočítat. Více informací o rovnicích lze nalézt v [16].

Elipsa

Elipsa je reprezentovaná rovnicí:

$$\frac{x^2}{a^2} + \frac{y^2}{b^2} = 1, \tag{3.4}$$

nebo rovnicí:

$$b^2x^2 + a^2y^2 + a^2b^2 = 0. \tag{3.5}$$

Rovnice 3.4 je středová rovnice elipsy, což znamená, že střed elipsy leží v bodě $S[0, 0]$, ale pro náš problém elipsa nemusí mít střed v počátku.

Obecná rovnice elipsy je:

$$\frac{(x-c)^2}{a^2} + \frac{(y-d)^2}{b^2} = 1, \tag{3.6}$$

kde c a d jsou souřadnice středu elipsy $S[c, d]$, a je velikost hlavní poloosy elipsy (vzdálenost od středu elipsy k horizontálnímu vrcholu elipsy) a b je velikost vedlejší poloosy elipsy (vzdálenost od středu elipsy k vertikálnímu vrcholu elipsy).

Pokud bychom chtěli vytvořit kruh, což je pouze speciální tvar elipsy, rovnice je stejná. Akorát u kruhu bude velikost strany a a velikost strany b stejná.

Pro náš problém bude důležité si u elipsy pamatovat souřadnice středu $S[c, d]$, velikost hlavní poloosy a a velikost vedlejší poloosy b. Pokud bychom

elipsu vykreslovali pomocí opsaného obdélníku, u kterého bychom znali souřadnice počátečního bodu $P[x_0, y_0]$ (tedy souřadnice levého horního rohu obdélníku) a jeho velikost stranu a_{ob} a stranu b_{ob}, tak si lehce spočítáme střed elipsy a její strany a a b.

Strana a:

$$a = \frac{a_{ob}}{2}. \tag{3.7}$$

Strana b:

$$b = \frac{b_{ob}}{2}. \tag{3.8}$$

Střed elipsy $S[c, d]$:

$$c = x_0 + \frac{a_{ob}}{2}, \tag{3.9}$$

$$d = y_0 + \frac{b_{ob}}{2}. \tag{3.10}$$

Obdélník

Obdélník je rovnoběžník, jehož všechny vnitřní úhly mají velikost 90°. Obdélník je popsaný vrcholy A, B, C a D, strany AB, BC, CD a DA, protilehlé strany jsou stejně velké. Tyto strany můžeme popsat pomocí rovnic přímek daných dvěma body:

$$\frac{y - y_1}{x - x_1} = \frac{y_2 - y_1}{x_2 - x_1}, \tag{3.11}$$

kde x_1 a y_1 jsou souřadnice jednoho z vrcholů obdélníku $P[x_1, y_1]$ a x_2 a y_2 jsou souřadnice jednoho z vrcholů sousedícího s vrcholem $P[x_1, y_1]$. U obdélníku jsou také podstatné velikosti jeho stran. Velikost strany a je velikost úsečky AB a velikost strany b je velikost úsečky BC.

Speciální případ obdélníku, čtverec, má stranu a rovnu straně b. Pro náš problém podobně jako u elipsy budeme potřebovat znát stranu a a stranu b. Pamatovat si všechny souřadnice vrcholů je nadbytečné, stačí znát pouze souřadnice středu obdélníku. Ze souřadnic středu a stran a a b snadno dopočítáme souřadnice vrcholů. U vykreslovaného obdélníku budeme znát levý horní bod obdélníku $P[x_0, y_0]$ a velikost strany a a strany b. Z těchto údajů si dopočítáme střed obdélníku $S[m, n]$.

Střed obdélníku $S[m, n]$:

$$m = x_0 + \frac{a}{2}, \tag{3.12}$$

$$n = y_0 + \frac{b}{2}. \tag{3.13}$$

Polygon

Polygon se skládá z N vrcholů, minimální počet vrcholů jsou tři. Strany mezi sousedními vrcholy můžeme stejně jako u obdélníku popsat pomocí rovnice přímky dané dvěma body.

Jelikož se jedná o obecný polygon, nezbývá nám nic jiného než si pamatovat souřadnice všech jeho vrcholů (bodů).

Spojovací prvky

Spojovací prvky – šroub, kolík, péro ve směru osy X, péro ve směru osy Y a obecná zátěž – budeme muset brát jako samostatná primitiva. Tato primitiva nebude dále možné kombinovat s dalšími objekty.

3.4 Sloučení objektů

Návrhová aplikace má umožňovat i sloučení primitiv, které představují plochu ke spojování nebo díry v ploše. Proto musíme zjišťovat, kdy primitiva nebo kombinace primitiv spojit s ostatními primitivy nebo kombinací primitiv.

Spojovat budeme pouze primitiva nebo kombinaci primitiv, které mají nějaké společné body. Proto musíme nějak zjistit, kdy se primitiva mají společné body a kdy nikoliv. Jelikož u všech primitiv, které budeme moci spojovat, známe veškeré podstatné informace k sestavení rovnic přímek nebo rovnic elips, snadno z těchto rovnic vypočítáme existenci společných bodů u těchto primitiv.

U elipsy známe souřadnice středu, velikost její hlavní poloosy a a velikost její vedlejší poloosy b. Tyto informace nám stačí k sestavení obecné rovnice elipsy.

U obdélníku známe souřadnice jeho středu, velikost jeho strany a a velikost strany b. Z těchto informací si spočítáme souřadnice vrcholů A, B, C, D. Ze souřadnic sousedních vrcholů AB, BC, CD a DA sestavíme rovnice přímek.

U polygonu známe souřadnice všech jeho vrcholů. Proto stejně jako u obdélníku sestavíme rovnice přímek ze souřadnic všech sousedních vrcholů.

U primitiv, které budeme reprezentovat rovnicemi přímek, musíme počítat se všemi jeho rovnicemi. Například pokud budeme zjišťovat, zda se obdélník a elipsa protínají, musíme spočítat vzájemnou polohu všech rovnic přímek daného obdélníku s rovnicí elipsy.

Více informací o vzájemné poloze objektů lze nalézt v [16].

3.4.1 Vzájemná poloha přímek

U dvou přímek mohou nastat tři případy jejich vzájemné polohy, a to že přímky jsou navzájem rovnoběžné, různoběžné nebo jsou shodné. Známe rovnice obou přímek sestavené ze dvou sousedních vrcholů daného primitiva, z těchto dvou rovnic sestavíme soustavu rovnic.

Soustava rovnic dvou přímek:

$$y = k_1 x + q_1, \tag{3.14}$$

$$y = k_2 x + q_2, \tag{3.15}$$

u této soustavy rovnic mohou nastat tři případy řešení.

1. Soustava rovnic nemá žádné řešení – to znamená, že přímky jsou navzájem rovnoběžné, tedy nemají žádný společný bod.
2. Soustava rovnic má nekonečně mnoho řešení – přímky mají nekonečně mnoho společných bodů, navzájem se překrývají.
3. Soustava rovnic má právě jedno řešení – přímky jsou navzájem různoběžné, to znamená že mají právě jeden společný bod.

Pokud chceme zjistit zda primitiva mají společné body, bude nás zajímat pouze třetí případ, tedy pokud má soustava rovnic právě jedno řešení. Když budeme vědět, že soustava rovnic má pouze jedno řešení, můžeme si spočítat

souřadnice průsečíku dvou přímek $P[x_p, y_p]$.

Rovnice pro výpočet souřadnic průsečíku $P[x_p, y_p]$:

$$x_p = \frac{q_1 - q_2}{k_2 - k_1}, \tag{3.16}$$

$$y_p = \frac{q_1 k_2 - q_2 k_1}{k_2 - k_1}, \tag{3.17}$$

pokud průsečík přímek bude ležet na úsečkách sestrojených z bodů, ze kterých jsme vytvořili rovnice přímek, víme, že primitiva mají společné body.

3.4.2 Vzájemná poloha přímky a elipsy

Vzájemná poloha přímky a elipsy má tři možná řešení, přímka jde mimo elipsu, přímka se dotýká elipsy nebo přímka protíná elipsu. Z nám známé rovnice elipsy a rovnice přímky sestavené ze dvou sousedních vrcholů daného primitiva, sestavíme následující soustavu rovnic. Pro zjednodušení je uvedena pouze středová rovnice elipsy.

Soustava rovnic:

$$y = kx + q, \tag{3.18}$$

$$\frac{x^2}{a^2} + \frac{y^2}{b^2} = 1, \tag{3.19}$$

potom průsečíky elipsy s přímkou spočítáme z následujících rovnic:

$$x_{1,2} = -\frac{a^2 kq}{b^2 + a^2 k^2} \pm \frac{ab}{b^2 + a^2 k^2}\sqrt{a^2 k^2 + b^2 - q^2}, \tag{3.20}$$

$$y_{1,2} = \frac{b^2 q}{b^2 + a^2 k^2} \pm \frac{abk}{b^2 + a^2 k^2}\sqrt{a^2 k^2 + b^2 - q^2}, \tag{3.21}$$

z diskriminantu $a^2k^2 + b^2 - q^2 = D$ zjistíme vzájemnou polohu přímky a elipsy. Mohou nastat tři případy.

1. Pokud $D < 0$ – přímka neprotíná elipsu.
2. Pokud $D = 0$ – přímka se dotýká elipsy, je tedy její tečna.
3. Pokud $D > 0$ – přímka protíná elipsu, je tedy její sečna.

Stejně jako u vzájemné polohy přímek nás bude zajímat pouze třetí případ, tedy pokud diskriminant $D > 0$, když přímka je sečnou elipsy a má s ní tedy společné dva body. Z rovnic si spočítáme průsečíky přímky s elipsou $P_1[x_1, y_1]$ a $P_2[x_2, y_2]$. Pokud alespoň jeden z těchto průsečíků bude ležet na úsečce sestrojené z bodů, ze kterých jsme vytvořili rovnici přímky, znamená to, že primitivum, z jehož vrcholů jsme sestrojili rovnici přímky, má společné body s elipsou.

3.4.3 Vzájemná poloha dvou elips

U vzájemné polohy dvou elips může nastat několik možných řešení, a to že elipsy nemají žádný společný bod, elipsy se navzájem dotýkají, elipsy leží na sobě a elipsy se navzájem protínají. Opět sestavíme soustavu rovnic ze dvou nám známých rovnic elips.

Soustava rovnic:

$$\frac{(x - c_1)^2}{a_1^2} + \frac{(y - d_1)^2}{b_1^2} = 1, \tag{3.22}$$

$$\frac{(x - c_2)^2}{a_2^2} + \frac{(y - d_2)^2}{b_2^2} = 1, \tag{3.23}$$

jejímž vyřešením mohou nastat čtyři možná řešení.

1. Soustava nemá řešení v oboru reálných čísel – elipsy nemají žádný společný bod.
2. Soustava má nekonečně mnoho řešení – elipsy se navzájem překrývají, tedy leží na sobě.
3. Soustava má právě jedno řešení – elipsy se navzájem dotýkají, mají právě jeden společný bod.
4. Soustava má dvě řešení $x_{1,2}$ a $y_{1,2}$ – elipsy se navzájem protínají, mají dva společné body.

Zde nás bude zajímat pouze čtvrtý případ, když soustava rovnic má dvě řešení $x_{1,2}$ a $y_{1,2}$, které nám určují souřadnice průsečíků elips $P_1[x_1, y_1]$ a $P_2[x_2, y_2]$. V jiných případech elipsy spojovat nebudeme.

3.5 Výběr knihovny pro tvorbu GUI

Na začátku práce jsem se musel rozhodnout, jakou knihovnu a s ní spojený programovací jazyk použiji k vytvoření aplikace.

3.5.1 Qt

Qt [8] je framework[1], specializovaný převážně na tvorbu grafických uživatelských rozhraní. Jedná se o multiplatformní framework, který podporuje až patnáct různých platforem, s minimální nebo žádnou potřebou úpravy zdrojového kódu. Tento framework je napsán v programovacím jazyce `C++`. *Qt* má i vlastní vývojové prostředí *Qt Creator*, které běží v systémech Linux, Mac OS a Microsoft Windows.

Qt nabízí dva možné přístupy ke tvorbě grafického uživatelského rozhraní (*GUI*). Jedna z možností je *GUI* vytvořit přímo v jazyce `C++` s použitím *Qt Widgets* [9] modulu. Druhá možnost k vytvoření *GUI* je použít *QtQuick* [10] modul, který využívá k tvorbě deklarativní jazyk *QML* (*Qt Meta–Object Language*) založený na *JavaScriptu*.

Qt je velmi zajímavý framework pro tvorbu *GUI*, ale je napsaný v programovacím jazyce `C++`, se kterým nemám moc zkušeností, proto jsem ho nepoužil pro tvorbu aplikace.

3.5.2 Swing

Knihovna *Swing* [11] je napsaná v programovacím jazyce *Java* a poskytuje aplikační rozhraní pro tvorbu grafického uživatelského rozhraní (*GUI*). Knihovna *Swing* nám umožňuje vytvářet různé grafické komponenty, jako jsou tlačítka, různé záložky, rámečky, tabulky, stromy, panely a mnohé další. Knihovna *Swing* je napsáná pomocí programovacího jazyka *Java*, programovací jazyk *Java* je multiplatformní, proto i knihovna *Swing* je multiplatformní. Částečně vychází ze svého předchůdce *AWT* (*Abstract Window Toolkit*).

Knihovna je obsažena v běžné distribuci *Javy SE* [12]. Přestože s prací v jazyce *Java* a knihovně *Swing* mám zkušenosti, rozhodl jsem se tuto knihovnu nepoužít pro tvorbu aplikace, jelikož je tato knihovna již poměrně zastaralá a dnes je v mnoha případech nahrazována novější knihovnou *JavaFX*.

3.5.3 JavaFX

Knihovna *JavaFX* [13], jak již název napovídá, je vyvíjena v programovacím jazyce *Java*. *JavaFX* slouží pro vývoj grafických uživatelských rozhraní

[1] Aplikační rámec obsahující knihovny a nástroje, které slouží k vývoji aplikací.

desktopových aplikací a *RIA* (*Rich Internet Applications*), což je v podstatě webová aplikace. *JavaFX* se postupně používá i pro vývoj aplikací pro mobilní zařízení. Podobně jako u *Swingu* můžeme u *JavaFX* využívat mnoho grafických komponent, dále podporuje práci se základními mediálními formáty, jako je video nebo zvuk.

Knihovna, od verze *Java 8*, je obsažena v běžné distribuci *Javy SE* [12] a postupně nahrazuje starší knihovnu *Swing*. Pro tvorbu aplikace jsem se rozhodl použít tuto knihovnu, protože s jazykem *Java* mám největší zkušenosti a *JavaFX* mi přijde lepší pro tvorbu *GUI* než *Swing*. Podrobněji je tato knihovna popsána v kapitole 4.1.1.

4 Implementace

4.1 Použité technologie

Program je napsán v programovacím jazyce *Java*, pro tvorbu grafického uživatelského rozhraní (*GUI*) byla použita knihovna *JavaFX*. Zdrojové kódy jsem psal ve vývojovém prostředí (*IDE – Integrated Development Environment*) *Eclipse*. Program byl vyvíjen na operačním systému *Windows 10*, na tomto operačním systému byla provedena i kompilace a sestavení programu. Všechny využité technologie pro vývoj programu jsou obsaženy v běžné distribuci *Javy SE* [12] od verze *Java 8*.

4.1.1 Knihovna JavaFX

Knihovna *JavaFX* [13], jak již bylo popsáno v kapitole 3.5.3, se specializuje na tvorbu grafických uživatelských rozhraní (*GUI*). Tato knihovna je multiplatformní, to znamená, že pro správnou funkčnost aplikace, není třeba měnit zdrojový kód, při přechodu na jinou platformu .

Základní struktura aplikací napsaných pomocí knihovny *JavaFX* je následující (viz ukázka 4.1): hlavní třída aplikace dědí od třídy `javafx.application.Application` a metoda `start()` je hlavním vstupním bodem programu.

Každé okno aplikace je reprezentováno třídou `Stage` a veškerý obsah tohoto okna je obsažen ve třídě `Scene`. Obsah `Scene` je reprezentován hierarchickým grafem (stromem), uzly v tomto stromu představují jednotlivé objekty ve scéně. Kořenový uzel stromu je většinou nějaký panel, do kterého se vkládají další uzly. Každý uzel ve stromu, kromě kořenového uzlu, má svého rodiče a může, ale nemusí mít potomky.

```
1 import javafx.application.Application;
2 import javafx.scene.Scene;
3 import javafx.scene.control.Button;
4 import javafx.scene.layout.StackPane;
5 import javafx.stage.Stage;
6
7 public class HelloWorld extends Application {
8     @Override
```

```
9       public void start(Stage primaryStage) {
10          Button btn = new Button("Hello World!");
11
12          StackPane root = new StackPane();
13          root.getChildren().add(btn);
14
15          Scene scene = new Scene(root, 300, 250);
16          primaryStage.setTitle("Hello World!");
17          primaryStage.setScene(scene);
18          primaryStage.show();
19      }
20   public static void main(String[] args) {
21          launch(args);
22      }
23  }
```

Kód 4.1: Ukázka struktury *JavaFX* aplikace.

U *JavaFX* aplikací máme dvě možnosti, jak do vyvíjené aplikace vkládat komponenty (různé panely, tlačítka, textová pole, atd.). První z možností je vytvářet instance komponent přímo v kódu, například jak je vidět v ukázce 4.1, kde je kořenový uzel stromu `StackPane` a jeho potomek je `Button`.

Druhá možnost je využít jazyk pro návrh grafického uživatelského rozhraní (*GUI*) *FXML* (viz kapitola 4.1.2). V aplikaci je použita kombinace obou způsobů vytváření komponent.

4.1.2 FXML

FXML [14] je soubor *XML* (viz kapitola 4.1.3) s definicí grafického uživatelského rozhraní (*GUI*). Tento *FXML* soubor je načítán ve zdrojovém kódu. Jeho načtením vznikají všechny očekávané instance komponent nadefinované v *FXML* souboru, se kterými je dále možno pracovat. *FXML* soubory nemusíme psát ručně, existuje totiž grafický designer – *SceneBuilder*, který *FXML* soubor automaticky vygeneruje.

```
1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <?import java.lang.*?>
4  <?import java.util.*?>
5  <?import javafx.scene.*?>
6  <?import javafx.scene.control.*?>
7  <?import javafx.scene.layout.*?>
8
```

```
<AnchorPane id="AnchorPane" prefHeight="200" prefWidth="320" xmlns:fx="
    http://javafx.com/fxml" fx:controller="fxmlexample.
    FXMLExampleController">
    <children>
        <Button layoutX="126" layoutY="90" text="Click Me!" onAction="#
    handleButtonAction" fx:id="button" />
        <Label layoutX="126" layoutY="120" minHeight="16" minWidth="69"
    fx:id="label" />
    </children>
</AnchorPane>
```

Kód 4.2: Ukázka souboru *FXML*.

`FXMLExampleController` je třída, ve které obsluhujeme vytvořené komponenty. Příklad načítání *FXLM* souboru je vidět v ukázce 4.3.

```
    @Override
    public void start(Stage stage) throws Exception {
        //Nacitani FXML souboru na root uzel
        Parent root = FXMLLoader.load(getClass().getResource("
    FXMLDocument.fxml"));

        Scene scene = new Scene(root);

        stage.setScene(scene);
        stage.show();
    }

    public static void main(String[] args) {
        launch(args);
    }
```

Kód 4.3: Ukázka načítání *FXML* souboru.

4.1.3 XML

XML [15] (*Extensible Markup Language*) je obecný značkovací jazyk textových dokumentů. Struktura *XML* souboru je tvořena stromem, jednotlivé uzly stromu se nazývají elementy. Soubor musí mít právě jeden kořenový element, neprázdné elementy musí být označeny startovací a ukončovací značkou. Elementy mohou mít ve startovací značce uvedeny atributy elementu, hodnoty atributů musí být uzavřeny buďto v jednoduchých nebo dvojitých uvozovkách. Elementy mohou být vnořeny, ale nemohou se překrývat, celý element musí být obsažen v jiném elementu.

Jazyk je určen především pro výměnu dat mezi aplikacemi a ukládání dat pro pozdější programové zpracování. `Java` má k dispozici knihovny pro jednoduchou práci s těmito dokumenty (vytváření *XML* souborů, čtení z *XML* souborů).

4.2 Struktura aplikace

V aplikaci jsem se snažil dodržet vícevrstvou architekturu *MVC* (*Model–view–controller*), což je softwarová architektura, která rozděluje aplikaci na tři části: datový model aplikace, uživatelské rozhraní aplikace a řídící logiku aplikace.

- Datový model (`model`) – představuje nějakou datovou strukturu, reprezentaci informací (objektů), se kterými aplikace pracuje.
- Uživatelské rozhraní (`view`) – je vlastně samotný vzhled grafického uživatelského rozhraní. Zobrazuje objekty z datového modelu a také zobrazuje další prvky uživatelského rozhraní.
- Řídící logika (`controller`) – má na starost práci s objekty z datového modelu, stará se o jejich modifikaci, podle požadavků uživatele. V podstatě se jedná o komponentu, která se stará o funkčnost aplikace, má na starost aplikační logiku.

V poslední době je tato softwarová architektura velmi populární a dobře od sebe odděluje jednotlivé části aplikace, proto jsem se snažil o její použití.

4.3 Datový model

Datový model (`model`) se skládá ze tříd (objektů) reprezentujících objekty k vykreslování (plochy, díry a spoje). Tyto třídy (objekty) jsou obsaženy v balíčku `application.model`. Tento balíček obsahuje ještě výčtový typ (`enum`), pro určování typu objektu (plocha, díra, spoj).

4.3.1 Objekt

Třída `Objekt` je abstraktní třída, od které dědí všechny objekty, které je možné vykreslovat na kreslící plochu. Důležité proměnné třídy `Objekt` jsou zobrazeny v následující 4.4 ukázce.

```
1  public abstract class Objekt {
2
3      protected Shape tvarObjektu;
4      protected OvladaniObjektu ovlObjektu;
5      protected ObjektMenu objektMenu;
6      protected TypObjektu typObjektu;
7      protected double stredX;
8      protected double stredY;
9      protected double stredNaPlatneX;
10     protected double stredNaPlatneY;
11     private double posunutiX;
12     private double posunutiY;
13     private double scrollPosunutiX;
14     private double scrollPosunutiY;
15 }
```

Kód 4.4: Důležité proměnné třídy `Objekt`.

Proměnné `stredX` a `stredY` reprezentují souřadnice středu objektu v milimetrech, `stredNaPlatneX` a `stredNaPlatneY` je pozice středu objektu v pixelech.

Třídy (objekty), které dědí od třídy `Objekt` jsou:

1. Třída `Stred` – tato třída reprezentuje střed kreslící plochy s slouží ke snadné orientaci na kreslící ploše.

2. Třída `Elipsa` – tato třída si navíc uchovává velikost elipsy. Velikost hlavní poloosy elipsy – `stranaA` a velikost vedlejší poloosy elipsy – `stranaB`. `Elipsa` může reprezentovat buďto plochu ke spojování nebo díru v ploše. Proměnná `tvarObjektu` je vizuální znázornění objektu ve tvaru elipsy.

3. Třída `Obdelnik` – tato třída si navíc uchovává velikost obdélníku. Velikost strany a – `stranaA` a velikost strany b – `stranaB`. Stejně jako u elipsy může být `Obdelnik` buď plocha nebo díra v ploše. Proměnná `tvarObjektu` je vizuální znázornění objektu ve tvaru obdélníku.

4. Třída `Polygon` – u třídy `Polygon` střed nepředstavuje střed polygonu, ale střed opsaného obdélníku polygonu. Třída `Polygon` si ještě uchovává pole vrcholů polygonu a může být také buďto plocha nebo díra v ploše. Proměnná `tvarObjektu` je vizuální znázornění objektu ve tvaru uzavřeného polygonu.

5. Třída `Kompozitum` – je speciální třída, která si uchovává pouze pole objektů, ze kterých se skládá. Třídy, které mohou být obsaženy v poli objektů jsou: `Elipsa`, `Obdelnik` a `Polygon`. Všechny obsažené objekty musí být stejného typu, buďto plocha nebo díra v ploše.

6. Třída `Sroub` – tato třída reprezentuje spojovací prvek šroub, navíc si uchovává poloměr šroubu. Proměnná `tvarObjektu` je vizuální znázornění šroubu.

7. Třída `Kolik` – tato třída reprezentuje spojovací prvek kolík, navíc si uchovává poloměr kolíku. Proměnná `tvarObjektu` je vizuální znázornění kolíku.

8. Třída `PeroX` – tato třída reprezentuje spojovací prvek péro ve směru osy X, navíc si uchovává stranu a a stranu b péra. Proměnná `tvarObjektu` je vizuální znázornění péra ve směru osy X.

9. Třída `PeroY` – tato třída je skoro stejná jako třída `PeroX`, ale reprezentuje spojovací prvek péro ve směru osy Y.

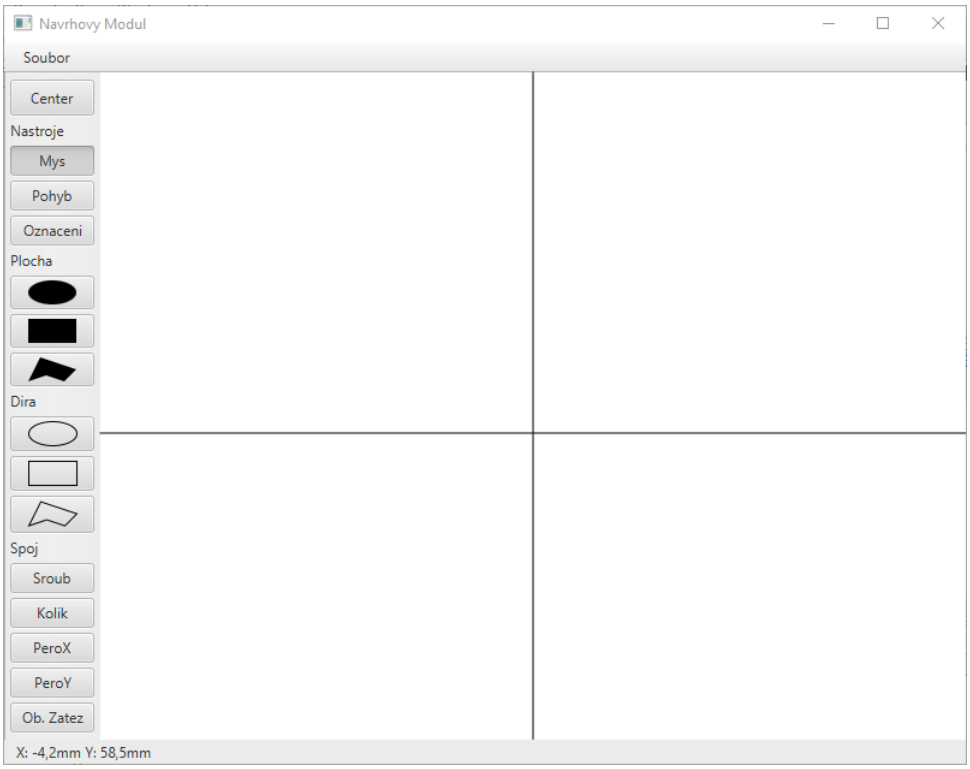
10. Třída `ObecZatez` – tato třída reprezentuje spojovací prvek obecná zátěž, navíc si uchovává stranu a a stranu b obdélníku, který představuje obecnou zátěž. Proměnná `tvarObjektu` je vizuální znázornění objektu ve tvaru obdélníku.

4.4 Grafické uživatelské rozhraní

Grafické uživatelské rozhraní aplikace (`view`) se skládá ze tříd obsažených v balíčku `application.view` a z *FXML* souborů obsažených ve složce `mapping`. Třídy reprezentují různá okna aplikace, komponenty těchto oken jsou nadefinovány v *FXML* souborech a načteny z příslušných tříd. Aplikační logika k těmto komponentám je obsažená ve třídách, které jsou nadefinovány v *FXML* souborech, jako `fx:cotroller` a bude popsána v kapitole 4.5.

4.4.1 Hlavní okno

Třída `HlavniOkno` představuje hlavní okno aplikace, načítá soubor s grafickými komponentami `HlavniOkno.fxml`, tento soubor obsahuje panel, který představuje kreslící plochu, hlavní menu, panel nástrojů a informační lištu, která zobrazuje souřadnice polohy myši na kreslící ploše.



Obrázek 4.1: Ukázka vzhledu hlavního okna aplikace

Hlavní okno je rozděleno na pět částí (horní, dolní, pravou, levou a prostřední). V horní části okna je umístěno hlavní menu, ve kterém jsou položky pro práci se souborem, a to vytvořit nový soubor, otevřít již existující soubor, uložit zpracovaný soubor a ukončení aplikace. V levé části okna je umístěn panel nástrojů, ve kterém si můžeme zvolit buďto nástroj (myš, pohyb po plátně nebo označení objektů), který budeme používat, nebo zvolit objekt, který chceme přidat na kreslící plochu. V prostřední části hlavního okna se nachází samotná kreslící plocha. V dolní části je umístěna informační lišta, která zobrazuje souřadnice polohy myši na kreslící ploše. Pravá část hlavního okna je prázdná.

4.4.2 Dialogová okna

Třídy `NovyVrchol`, `OdstranitVrcholy`, `VrcholyPolygonu`, `StredObjektu`, `PolomerObjektu` a `RozmeryObjektu` jsou dialogová okna, která slouží k úpravě různých objektů. Jejich komponenty jsou načteny z příslušných *FXML* souborů. Daný typ okna je zobrazen po kliknutí na položku v kontextovém menu objektu.

- Třída `NovyVrchol` představuje dialogové okno pro přidávání nového vrcholu objektu typu polygon. Grafické komponenty tohoto okna jsou načítány ze souboru `PridaniVrcholu.fxml`.
- Třída `OdstranitVrcholy` představuje dialogové okno pro odstraňování vrcholů objektu typu polygon. Komponenty tohoto okna jsou načítány ze souboru `OdstraneniVrcholu.fxml`.
- Třída `VrcholyPolygonu` představuje dialogové okno pro nastavování souřadnic vrcholů objektu typu polygon. Komponenty tohoto okna jsou načítány ze souboru `NastaveniVrcholu.fxml`.
- Třída `StredObjektu` představuje dialogové okno pro nastavování souřadnic středu objektu. Komponenty tohoto okna jsou načítány ze souboru `NastaveniStredu.fxml`.
- Třída `PolomerObjektu` představuje dialogové okno pro nastavení poloměru objektů typu šroub a kolík. Komponenty tohoto okna jsou načítány ze souboru `NastaveniPolomeru.fxml`.
- Třída `RozmeryObjektu` představuje dialogové okno pro nastavování rozměrů objektu, pro všechny objekty, které mají dva různé rozměry, stranu a a stranu b. Komponenty tohoto okna jsou načítány ze souboru `NastaveniVelikosti.fxml`.

4.4.3 Kontextové menu

Třída `ObjektMenu` představuje kontextové menu objektů, každý objekt si vytváří vlastní instanci této třídy. Tato třída grafické komponenty nenačítá ze žádného *FXML* souboru, ale vytváří si komponenty kontextového menu přímo v kódu. Kontextové menu se zobrazuje po kliknutí pravým tlačítkem myši na objekt. Různé typy objektů mají různé položky v kontextovém menu.

Položky kontextového menu můžeme rozdělit do tří skupin, podle toho pro jaké typy objektů se dané položky zobrazují:

1. První skupinou jsou položky `Stred`, `Velikost` a `Odstranit objekt`, které se zobrazují u většiny typů objektů – elipsa, obdélník, šroub, kolík, péro ve směru osy X, péro ve směru osy Y a obecná zátěž. Po kliknutí na položku `Stred` se zobrazí dialogové okno pro nastavení středu objektu, po kliknutí na položku `Velikost` se zobrazí dialogové okno pro nastavení velikosti objektu, buďto dialogové okno pro nastavení poloměru nebo dialogové okno pro nastavení rozměrů, záleží na

typu objektu. Po kliknutí na položku `Odstranit objekt` se vybraný objekt odstraní.

2. Druhou skupinou položek kontextového menu jsou položky `Pridej vrchol`, `Vrcholy`, `Odstran vrcholy` a položka `Odstranit objekt`, ta je stejná jako u ostatních typů objektů. Tyto položky se zobrazují u objektu typu polygon. Po kliknutí na položku `Pridej vrchol` se zobrazí dialogové okno pro přidání vrcholu polygonu, po kliknutí na položku `Vrcholy` se zobrazí dialogové okno pro nastavování souřadnic vrcholů polygonu a po kliknutí na položku `Odstran vrcholy` se zobrazí dialogové okno pro odstranění vrcholů polygonu.

3. Do poslední skupiny patří pouze položka `Rozdelit`, která se zobrazuje pouze u objektu typu kompozitum. Po kliknutí na tuto položku `Rozdelit` se objekt kompozitum rozdělí na původní objekty, ze kterých se skládá.

4.5 Aplikační logika

Aplikační logika (`controller`) se skládá ze tříd, které jsou obsaženy v balíčku `application.control`. Tyto třídy většinou představují `fx:controller` třídy k nějakým *FXML* souborům, `fx:controller` třídy jsou třídy, které mají přístup ke komponentám (mohou je ovládat) nadefinovaných v *FXML* souborech.

4.5.1 Ovládání hlavního okna

Třída `Ovladani` je základní třída s logikou pro ovládání hlavního okna aplikace. Tato třída představuje `fx:controller` třídu k souboru `HlavniOkno.fxml`, to znamená, že má přístup k jednotlivým komponentám hlavního okna aplikace. Třída `Ovladani` implementuje rozhraní `Initializable`. Při inicializaci deklaruje prázdné pole objektů, se kterým poté mohou pracovat i jiné třídy z balíčku aplikační logiky. Dále při inicializaci přidá na kreslící plátno objekt `Stred`.

Přidávání objektů

V balíčku tříd aplikační logiky je obsažen výčtový typ (`enum`) `Nastroj`, který obsahuje všechny nástroje i typy objektů, které mohou být vybrány z panelu nástrojů. Po zvolení určitého typu objektu z panelu nástrojů se nastaví `Nastroj` na vybraný nástroj nebo objekt. Pokud je `Nastroj` nastaven na

určitý objekt, je možné vybraný objekt přidat na kreslící plochu. Přidávání objektu probíhá kliknutím levým tlačítkem myši kamkoliv na kreslící plochu tam, kde chceme objekt umístit. Na souřadnicích, na kterých proběhlo kliknutí, se nastaví souřadnice středu objektu. Poté se samotný objekt přidá do pole objektů a na kreslící plochu.

Zobrazování souřadnic na informační liště

Po najetí myší na kreslící plochu se na informační liště zobrazují aktuální souřadnice, na kterých se nachází myš, v milimetrech. Souřadnice se vypočítají z aktuální polohy myši, z pozice středu kreslící plochy a z přiblížení nebo z oddálení kreslící plochy. Od středu kreslící plochy se odečte aktuální poloha myši a pomocí statické třídy `PrevodJednotek` se převedou pixely na milimetry. Pokud myš opustí kreslící plochu souřadnice zmizí.

4.5.2 Ovládání kreslící plochy

Třída `OvladaniPlatna` dědí od třídy `Ovladani` a slouží k ovládání kreslící plochy. Tato třída je `fx:controller` třída k souboru `Platno.fxml`, to znamená, že tato třída má přístup ke komponentě `Panel`, která představuje kreslící plochu. Stejně jako třída `Ovladani`, i třída `OvladaniPlatna` implementuje rozhraní `Initializable`. Při inicializaci se komponentě – kreslicí ploše nastaví příslušný `EventHandler` pro každý ovládací prvek kreslící plochy (stisknutí tlačítka myší, pohyb myši se stisknutým tlačítkem, uvolnění tlačítka myši a „scroll“ kolečkem myši).

Pohyb po kreslící ploše

Pokud je zvolen nástroj *POHYB*, je možné se pohybovat po kreslící ploše. Stisknutím levého tlačítka myši a pak pohybem myši se stisknutým levým tlačítkem se zdánlivě pohybuje kreslící plocha. Uvolněním levého tlačítka myši se ukončuje pohyb.

Ve skutečnosti se samotná kreslící plocha vůbec nepohybuje, pohybují se pouze všechny objekty na kreslící ploše. Ze souřadnic při stisknutí levého tlačítka myši se nastaví souřadnice ukotvení myši. Při pohybu s myší se stisknutým levým tlačítkem se pohybuje se všemi objekty na kreslící ploše, posunutí se vypočítá ze současných souřadnic myši a ze souřadnic ukotvení myši. Zdánlivě to potom vypadá, že se pohybuje samotná kreslící plocha.

Ukončení pohybu kreslící plochy nastane při uvolnění levého tlačítka myši. Při uvolnění levého tlačítka myši se všem objektům nastaví posunutí, o kolik byl daný objekt posunutý na kreslící ploše.

Přiblížení a oddálení kreslící plochy

Přibližování a oddalování kreslící plochy není závislé na žádném nástroji, je tedy jedno jaký nástroj je zvolen. „Scrollováním“ kolečkem myši se buďto kreslící plocha zdánlivě přibližuje nebo zdánlivě oddaluje.

Ve skutečnosti se samotná kreslící plocha nepřibližuje ani neoddaluje, pouze se všechny objekty na kreslící ploše posouvají a úměrně k tomu se zmenšují nebo zvětšují. Nejprve se nastaví měřítko kreslící plochy (`scale`) podle „scrollu“ kolečkem myši. Ověří se, zda toto měřítko nepřesáhlo maximální hodnotu přiblížení nebo minimální hodnotu oddálení. Pokud by měřítko mělo tyto hodnoty přesáhnout, nastaví se buďto maximální nebo minimální hodnota. Poté nastane samotné posouvání a zmenšování nebo zvětšování objektů na kreslící ploše.

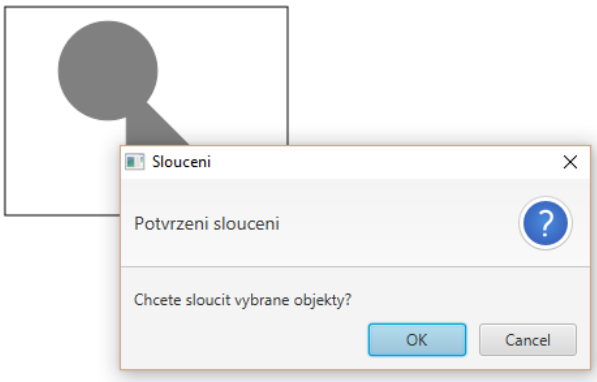
Objekty si kromě informace o posunutí uchovávají ještě informaci o „scroll“ posunutí. „Scroll“ posunutí objektu se vypočítá se souřadnic středu objektu a z aktuálního měřítka kreslící plochy. Objekty se zmenší nebo zvětší podle hodnoty měřítka. Aby byly objekty umístěny na správné pozici musí se ještě posunout o hodnotu „scroll“ posunutí.

Slučování objektů na kreslící ploše

Pokud je zvolen nástroj *OZNACENI* je možné slučovat objekty do jednoho objektu – vytváření objektů typu kompozitum.

Stisknutím levého tlačítka myši se nastaví levý horní bod označovacího obdélníku a pak pohybem se stisknutým levým tlačítkem myši se nastavuje velikost označovacího obdélníku z aktuálních souřadnic myši. Dále se kontroluje, zda objekty, které celé leží uvnitř označovacího obdélníku, mají nějaký průsečík.

Pokud alespoň dva objekty uvnitř označovacího obdélníku mají nějaký průsečík, dojde k samotnému slučování. Nejprve se zobrazí potvrzovací dialog, zda skutečně chceme vybrané objekty sloučit, viz obrázek 4.2. Pokud je zvolena možnost `Cancel` nebo je dialogové okno zavřeno, nedojde k žádné akci. Pokud je zvolena možnost `OK`, dojde ke sloučení vybraných objektů, které



Obrázek 4.2: Slučování dvou objektů

mají nějaké průsečíky.

Vytvoří se nový objekt typu kompozitum, který obsahuje všechny objekty, ze kterých byl vytvořen. Z pole objektů se odstraní všechny objekty, ze kterých byl objekt typu kompozitum vytvořen a poté se již přidá vytvořené kompozitum do pole objektů.

4.5.3 Ovládání objektů

Třída `OvladaniObjektu` dědí od třídy `Ovladani` a slouží k ovládání jednotlivých objektů. Tato třída implementuje příslušný `EventHandler` pro každý ovládací prvek objektu (stisknutí tlačítka myši, pohyb se stisknutým tlačítkem myši a uvolnění tlačítka myši). Ovládat objekty je možné pouze pokud je zvolen nástroj *MYS*.

Obrázek 4.3: Ukázka různých typů objektů

Pohyb s objekty

Pokud je zvolen nástroj *MYS*, je možné s vybraným objektem pohybovat po kreslící ploše. Stisknutí levého tlačítka myši na objektu nastaví souřadnice

ukotvení myši, při pohybu myší se stisknutým levým tlačítkem se objekt pohybuje po kreslící ploše. Při uvolnění levého tlačítka myši dochází k nastavení nových souřadnic středu objektu, tyto souřadnice jsou vypočítány z původních souřadnic středu objektu a posunutí objektu.

Zobrazování kontextového menu objektů

Stejně jako u pohybu s objekty musí být pro zobrazení kontextového menu vybraného objektu zvolen nástroj `MYS`. Stisknutím pravého tlačítka myši na objektu se zobrazí kontextové menu daného objektu.

4.5.4 Nastavení vlastností objektu

Nastavování různých vlastností objektů probíhá v dialogových oknech, které se zobrazují po kliknutí na položku kontextového menu objektu. Třídy `NastaveniPolomeru`, `NastaveniStredu`, `NastaveniVelikosti`, `NastaveniVrcholu`, `OdstraneniVrcholu` a `PridaniVrcholu` slouží k ovládání různých dialogových oken.

- Třída `NastaveniPolomeru` – tato třída je `fx:controller` třída k souboru `NastaveniPolomeru.fxml`. Dialogové okno, které představuje zmiňovaný *FXML* soubor má pouze jednu nastavitelnou komponentu, tou je poloměr objektu. Třída `NastaveniPolomeru` ověří, zda zadávaný poloměr je reálné číslo a je kladné. Pokud ano, nastaví objektu jako poloměr zadanou hodnotu a objekt se s novým poloměrem překreslí na kreslící ploše.

- Třída `NastaveniStredu` – tato třída je `fx:controller` třída k souboru `NastaveniStredu.fxml`. Dialogové okno, které představuje zmiňovaný *FXML* soubor má dvě komponenty k nastavení, tyto komponenty reprezentují souřadnice středu objektu. Třída `NastaveniStredu` ověří, zda zadávané souřadnice středu jsou reálná čísla. Pokud jsou to reálná čísla, nastaví objektu nové souřadnice středu a přemístí objekt na kreslící ploše na zadané souřadnice.

- Třída `NastaveniVelikosti` – tato třída je `fx:controller` třída k souboru `NastaveniVelikosti.fxml`. Dialogové okno, které představuje zmiňovaný *FXML* soubor má dvě komponenty k nastavení, kterými jsou rozměry objektu (strana a a strana b). Třída `NastaveniVelikosti` ověří zda zadávané rozměry objektu jsou reálná a kladná čísla. Pokud

jsou tyto podmínky splněny nastaví se objektu jako jeho rozměry zadané hodnoty a objekt se s novými rozměry překreslí na kreslící ploše.

- Třída `NastaveniVrcholu` – tato třída je `fx:controller` třída k souboru `NastaveniVrcholu.fxml`. Dialogové okno, které představuje zmiňovaný *FXML* soubor se zobrazuje pouze pro objekt typu polygon. V komponentách tohoto okna se nastavují souřadnice vrcholů polygonu. Třída `OdstraneniVrcholu` ověří zda zadávané souřadnice jsou reálná čísla, pokud jsou to reálná čísla nastaví nové souřadnice vrcholů objektu typu polygon a tento objekt se s novými souřadnicemi vrcholů překreslí na kreslící ploše.

- Třída `OdstraneniVrcholu` – tato třída je `fx:controller` třída k souboru `OdstraneniVrcholu.fxml`. Dialogové okno, které představuje zmiňovaný *FXML* soubor se zobrazuje pouze pro objekt typu polygon. U tohoto dialogového okna se dá nastavit, které vrcholy polygonu se odstraní. Třída `OdstraneniVrcholu` ověří zda počet neoznačených vrcholů jsou alespoň tři vrcholy, protože minimální počet vrcholů uzavřeného polygonu jsou tři vrcholy. Pokud jsou alespoň tři vrcholy neoznačené, odstraní z objektu typu polygon označené vrcholy a tento objekt se bez odstraněných vrcholů překreslí na kreslící ploše.

- Třída `PridaniVrcholu` – tato třída je `fx:controller` třída k souboru `PridaniVrcholu.fxml`. Dialogové okno, které představuje zmiňovaný *FXML* soubor se zobrazuje pouze pro objekt typu polygon. V komponentách tohoto okna se nastavují souřadnice přidávaného vrcholu polygonu. Třída `PridaniVrcholu` ověří, zda zadávané souřadnice jsou reálná čísla, pokud jsou to reálná čísla přidá, nový vrchol do objektu typu polygon se zadanými souřadnicemi a tento objekt se i s novým vrcholem překreslí na kreslící ploše.

4.5.5 Ukládání a načítání souboru s objekty

Logiku ukládání a načítání souboru s objekty obstarává třída `Ovladani`. Objekty se ukládají do souboru *XML*. V následujícím kódu (viz ukázka 4.5) je vidět struktura, jak se objekty ukládají do *XML* souboru

```
1 <?xml version="1.0" encoding="UTF-8" standalone="yes"?>
2 <Objekty xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:
      schemaLocation="">
3     <Objekt xsi:type="kompozitum" id="3" Typ="PLOCHA">
4         <Prvek xsi:type="obdelnik" id="1" Typ="PLOCHA">
```

```
            <StredX>0.0</StredX>
            <StredY>25.0</StredY>
            <StranaA>25.0</StranaA>
            <StranaB>20.0</StranaB>
        </Prvek>
        <Prvek xsi:type="elipsa" id="2" Typ="PLOCHA">
            <StredX>20.0</StredX>
            <StredY>25.0</StredY>
            <StranaA>10.0</StranaA>
            <StranaB>15.0</StranaB>
        </Prvek>
    </Objekt>
    <Objekt xsi:type="kolik" id="4" Typ="SPOJ">
        <StredX>3.4</StredX>
        <StredY>26.1</StredY>
        <Polomer>5.0</Polomer>
    </Objekt>
</Objekty>
```

Kód 4.5: Ukázka struktury *XML* souboru.

Kliknutím na položku hlavního menu `Ulozit` se otevře ukládací dialog, ve kterém se zvolí umístění souboru a jeho název. Po potvrzení ukládacího dialogu (kliknutím na tlačítko `Uložit`), se objekty na kreslící ploše uloží do souboru *XML* se zvoleným názvem, do zvoleného adresáře.

Kliknutím na položku hlavního menu `Otevrit` se otevře otevírací dialog, ve kterém se zvolí soubor *XML*, ve kterém je uložené nějaké schéma objektů. Po zvolení požadovaného souboru a po potvrzení otevíracího dialogu (kliknutím na tlačítko `Otevřít`), se do pole objektů načtou objekty obsažené ve vybraném *XML* souboru, poté se vykreslí na kreslící plochu.

Kliknutím na položku hlavního menu `Novy` se vymažou všechny objekty z pole objektů a zároveň se smažou i z kreslící plochy.

4.6 Testování

Program byl vyzkoušen na operačním systému *Windows 10* 64 bit. Základní funkce programu a jejich interakce byly otestovány při ladění programu. U programu velice záleží na grafickém zobrazování objektů na kreslící ploše. Z těchto důvodů bylo důležité otestovat, zda při posouvání objektu na kreslící ploše, se objekt vizuálně nachází na správných souřadnicích. Toto bylo testováno pro různé objekty, s různými rozměry a bylo zjištěno, že objekty

byly vždy vizuálně na správných souřadnicích.

Testování samotné kreslící plochy (pohyb po kreslící ploše, přibližování a oddalování kreslící plochy) bylo také velmi důležité. To znamená, bylo nutno otestovat, zda objekty při pohybu kreslící plochy mají správné souřadnice, a zda se vizuálně kreslící plocha pohybuje. To samé se týkalo otestování pro přibližování a oddalování kreslící plochy, to je zda se objekty vizuálně zmenšují nebo zvětšují a zároveň souřadnice jejich středu a jejich velikost jsou stále stejné. Tyto případy byly otestovány s různým počtem objektů na kreslící ploše s různými souřadnicemi středů objektů a různými rozměry objektů.

Dále bylo testováno pro nastavování souřadnic středu objektu, rozměrů objektu a dalších vlastností objektu, zda zadávané hodnoty jsou pouze reálná čísla a u rozměrů objektu navíc, zda zadávané hodnoty jsou reálná kladná čísla.

5 Závěr

V rámci této práce jsem se seznámil s některými existujícími *CAD* aplikacemi a jejich základním ovládáním. Také jsem si zlepšil znalosti v používání knihovny *JavaFX*, která se využívá pro tvorbu grafických uživatelských rozhraní.

Cílem této práce bylo navrhnout a naprogramovat interaktivní vizuální návrhový modul šroubovaných spojů, jehož výstupem bude datový soubor pro výpočetní software (tento software již existuje), který provádí výpočet pevnosti šroubovaného spoje. Vyvinutý software byl napsán v programovacím jazyce *Java* s použitím knihovny *JavaFX* pro tvorbu grafického uživatelského rozhraní. Vyvinutá aplikace splňuje zadání práce. Aplikace dokáže jednoduše vytvořit vizuální návrh šroubovaného spoje, tento vizuální návrh umí ukládat do *XML* souboru. Aplikace umožňuje tento vytvořený *XML* soubor otevřít, zobrazit obsažený vizuální návrh šroubovaného spoje a provádět na něm další úpravy.

Ovládání aplikace je velmi snadné a i nevyškolený uživatel dokáže aplikaci bez větších potíží ovládat. Tato aplikace usnadní Katedře konstruování strojů Fakulty strojní ZČU navrhování šroubovaných spojů pro jejich již existující software pro výpočet pevnosti šroubovaného spoje.

Vytvořená aplikace by mohla být v budoucnu rozšířena řadou možných vylepšení. Mezi tyto vylepšení patří například: lepší úprava grafického vzhledu oken aplikace, nebo přidání dalších ovládacích prvků sloužících pro snazší manipulaci s objekty. Dále přichází v úvahu možnost vylepšit aplikaci o souběžné pracování s vícero pracovními plochami zároveň.

Literatura

[1] *Computer aided design* [online]. [cit. 2016/01/22]. CAD. Dostupné z: http://cs.wikipedia.org/wiki/Computer_aided_design.

[2] *Uživatelské rozraní* [online]. [cit. 2016/01/22]. UI. Dostupné z: http://en.wikipedia.org/wiki/User_interface.

[3] *Grafické uživatelské rozhraní* [online]. [cit. 2016/01/22]. GUI. Dostupné z: http://cs.wikipedia.org/wiki/Grafick%C3%A9_u%C5%BEivatelsk%C3%A9_rozhran%C3%AD.

[4] *AutoCAD* [online]. [cit. 2016/01/23]. Dostupné z: http://www.autodesk.com/products/autocad/overview.

[5] *SolidWorks* [online]. [cit. 2016/01/23]. Dostupné z: http://www.solidworks.com/.

[6] *CATIA* [online]. [cit. 2016/01/23]. Dostupné z: http://www.3ds.com/products-services/catia.

[7] *Bolted joints* [online]. [cit. 2016/05/25]. Dostupné z: http://en.wikipedia.org/wiki/Bolted_joint.

[8] *Qt* [online]. [cit. 2016/05/28]. Dostupné z: http://www.qt.io/.

[9] *Qt Widgets* [online]. [cit. 2016/05/28]. Dostupné z: http://doc.qt.io/qt-5/qtwidgets-index.html.

[10] *Qt Quick* [online]. [cit. 2016/05/28]. Dostupné z: http://doc.qt.io/qt-5/qtquick-index.html.

[11] *Swing (Java)* [online]. [cit. 2016/05/29]. Dostupné z: http://en.wikipedia.org/wiki/Swing_(Java).

[12] *Java Platform Standard Edition 8 Documentation* [online]. Oracle. [cit. 2016/05/29]. Dostupné z: http://docs.oracle.com/javase/8/docs/.

[13] *Getting Started with JavaFX, Release 8* [online]. Oracle. [cit. 2016/06/05]. Dostupné z: http://docs.oracle.com/javase/8/javafx/get-started-tutorial/jfx-overview.htm.

[14] *Introduction to FXML* [online]. Oracle. [cit. 2016/06/05]. Dostupné z: http://docs.oracle.com/javase/8/javafx/api/javafx/fxml/doc-files/introduction_to_fxml.html.

[15] *Extensible Markup Language* [online]. [cit. 2016/06/05]. XML. Dostupné z: `http://cs.wikipedia.org/wiki/Extensible_Markup_Language`.

[16] Bartsch, H.-J. *Matematické vzorce*. Státní nakladatelství technické literatury, 1963. ISBN 04-010-63.

A Uživatelská příručka

A.1 Překlad programu

Přeložit program lze několika způsoby, potřebný software pro překlad programu je operační systém Windows a nainstalovaná *Java SE* verze `8u40` a vyšší. První způsob přeložení programu je prostřednictvím nějakého integrovaného vývojového prostředí *IDE* (*Integrated Development Environment*) (např. *Eclipse*). Druhý způsob překladu je pomocí nástroje *Ant*, v adresáři `build` je soubor `build.xml`, program pomocí tohoto souboru příkazem `ant -buildfile build.xml` přeložíme.

A.2 Spouštění programu

Pro spuštění aplikace je potřeba stejný software jako pro překlad programu. V adresáři `build/dist` je soubor `NavrhovyModul.jar`, soubor lze spustit dvojklikem na tento soubor. Další možnost spuštění aplikace je pomocí příkazové řádky příkazem `java -jar NavrhovyModul.jar`.

A.3 Obsluha programu

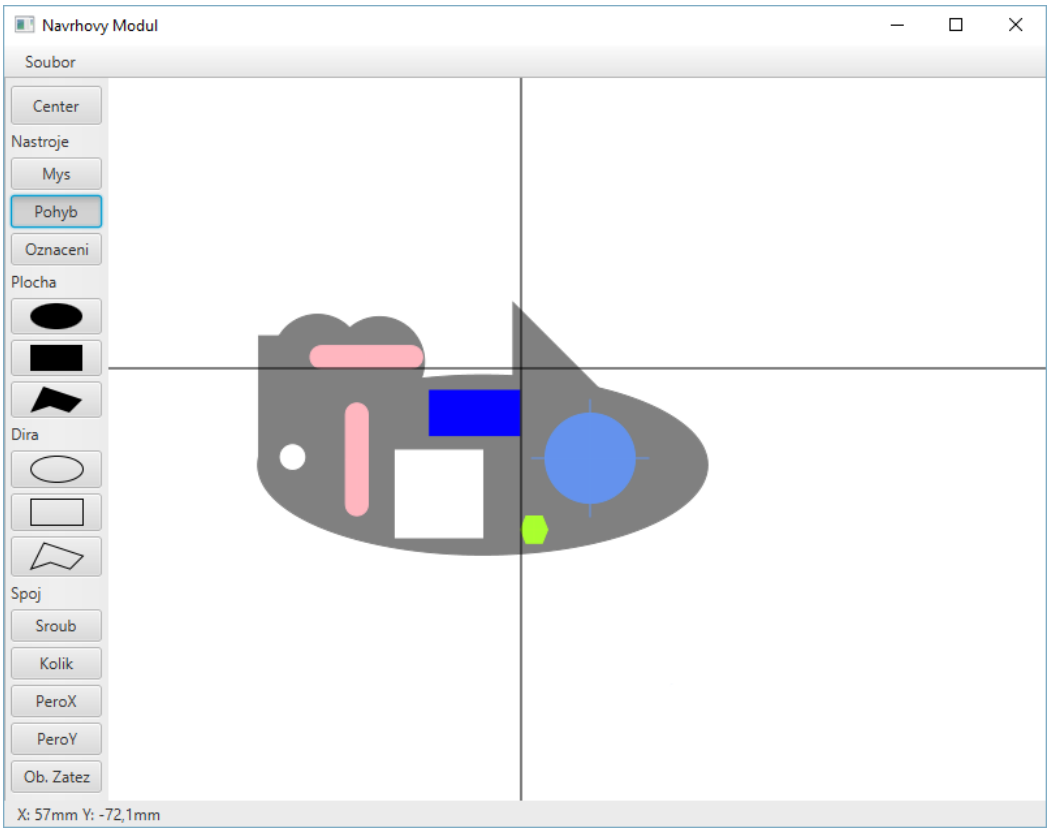
Po spuštění programu se objeví hlavní okno aplikace, viz obrázek A.1. Hlavní okno aplikace se skládá z menu, panelu nástrojů, kreslící plochy a informační lišty.

A.3.1 Menu

Menu obsahuje pouze jednu položku menu `Soubor`, v této položce jsou další čtyři položky.

- `Novy` – tato položka „vyčistí“ (smaže všechny objekty) kreslící plochu a střed kreslící plochy se nastaví na střed plátna.
- `Otevrit` – tato položka otevře dialog otevření souboru. Po vybrání *XML* souboru s reprezentací objektů, se objekty vykreslí na kreslící plochu.

- `Ulozit` – tato položka otevře dialog pro uložení souboru. Po zadání názvu a umístění souboru, uloží vykreslené schéma objektů do *XML* souboru se zadaným názvem a umístěním.
- `Konec` – tato položka ukončí aplikaci.



Obrázek A.1: Hlavního okno aplikace s vykreslenými objekty

A.3.2 Přidávání objektů na kreslící plochu

V panelu nástrojů, v levé části hlavního okna aplikace, můžeme vybrat objekty, které se dají přidat na kreslící plochu. Přidávání objektu probíhá kliknutím na vybraný objekt v panelu nástrojů, poté kliknutím levým tlačítkem myši na kreslící plochu se vykreslí vybraný objekt.

Typy objektů:

- *Elipsa plocha* – představuje plochu ke spojování ve tvaru elipsy.
- *Obdélník plocha* – představuje plochu ke spojování ve tvaru obdélníku.

- *Polygon plocha* – představuje plochu ke spojování ve tvaru polygonu.
- *Elipsa díra* – představuje díru v ploše ve tvaru elipsy.
- *Obdélník díra* – představuje díru v ploše ve tvaru obdélníku.
- *Polygon díra* – představuje díru v ploše ve tvaru polygonu.
- *Sroub* – představuje spojovací prvek šroub.
- *Kolik* – představuje spojovací prvek kolík.
- *PeroX* – představuje spojovací prvek péro ve směru osy X.
- *PeroY* – představuje spojovací prvek péro ve směru osy Y.
- *ObecZatez* – představuje spojovací prvek obecná zátěž.

A.3.3 Nástroje pro práci s objekty a kreslící plochou

Přiblížení a oddálení kreslící plochy

Přibližování nebo oddalování kreslící plochy se provádí „scrollováním“ kolečkem myši, pokud se kurzor myši nachází na kreslící ploše. Přibližují nebo oddalují se i všechny objekty na kreslící ploše.

Posouvání kreslící plochy

Pro posouvání kreslící plochy je třeba vybrat nástroj `Pohyb` z panelu nástrojů. Pokud je vybraný nástroj `Pohyb`, můžeme kreslící plochu posouvat stlačením levého tlačítka myši na kreslící ploše a následným tažením myši se stisknutým levým tlačítkem v požadovaném směru posouváme kreslící plochu.

Tlačítko `Center` v panelu nástrojů slouží k vycentrování kreslící plochy. To znamená, že střed kreslící plochy se posune na střed plátna.

Manipulace s objekty na kreslící ploše

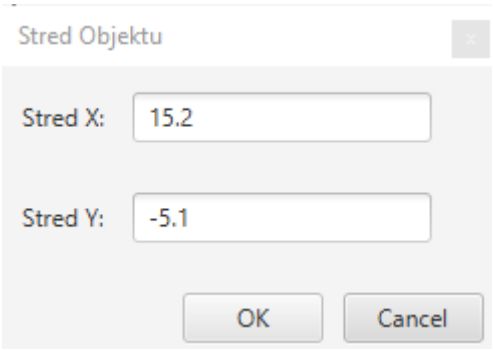
Pro pohybování s objekty nebo pro nastavování jejich vlastností je třeba vybrat nástroj `Mys` z panelu nástrojů. Pokud je vybraný nástroj `Mys`, můžeme stlačením levého tlačítka myši na vybraném objektu a následným tažením myši se stisknutým levým tlačítkem v požadovaném směru pohybovat s objektem po kreslící ploše.

S vybraným nástrojem `Mys` můžeme nastavovat i různé vlastnosti objektu (velikost objektu, střed objektu, atd.). Kliknutím pravým tlačítkem na vybraný objekt se otevře kontextové menu objektu, ve kterém si zvolíme co chceme nastavit. Více v kapitole A.3.4.

Nástroj `Oznaceni` umožňuje vybrat více objektů a sloučit je do jednoho. Slučovat lze pouze objekty, které představují plochu nebo díru. Výběr objektů se provádí stisknutím levého tlačítka myši na kreslící ploše a následným tažením myši se stisknutým levým tlačítkem se vykresluje označovací obdélník. Pokud po uvolnění levého tlačítka myši se nějaké objekty uvnitř označovacího obdélníku protínají, zobrazí se dialogové okno, zda chceme vybrané objekty (objekty uvnitř obdélníku) sloučit. Potvrzením tohoto dialogového okna se objekty sloučí.

A.3.4 Nastavení vlastností objektu

Po zvolení položky z kontextového menu objektu, se buďto zobrazí dialogové okno pro nastavení požadované vlastnosti nebo se pouze provede požadovaná akce (například se objekt odstraní).



Obrázek A.2: Dialogové okno pro nastavení středu objektu

Typy dialogových oken:

- Nastavení středu objektu – v tomto dialogovém okně se nastavují souřadnice středu objektu. Vzhled dialogového okna můžeme vidět na obrázku A.2.
- Nastavení rozměrů objektu – v tomto dialogovém okně se nastavují rozměry objektu, buďto velikost strany a a strany b nebo poloměr objektu.

- Nastavení souřadnic vrcholů polygonu – v tomto dialogovém okně se nastavují souřadnice vrcholů polygonu.
- Odstranění vrcholů polygonu – v tomto dialogové okně se odstraňují vrcholy polygonu.
- Přidání vrcholu polygonu – v tomto dialogové okně se nastaví souřadnice přidávaného vrcholu polygonu a následně se vrchol přidá.