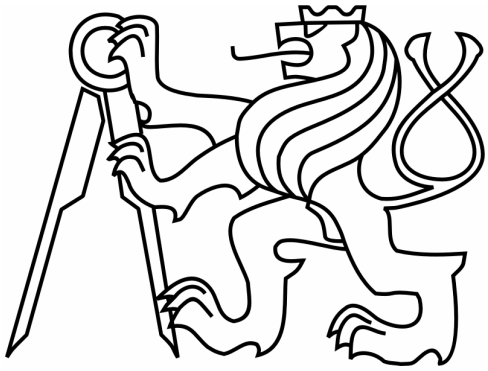
ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE

Fakulta elektrotechnická

Katedra řídicí techniky



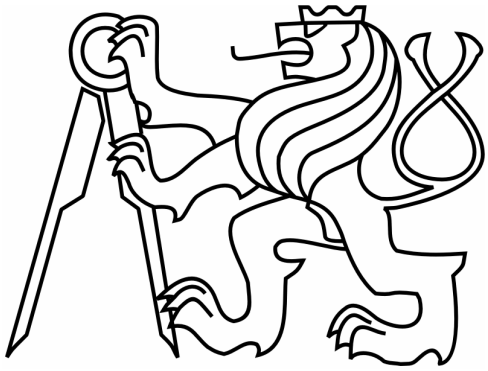
Bakalářská práce

2007 Miroslav Janáček

ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE

Fakulta elektrotechnická

Katedra řídicí techniky



Bakalářská práce

Ganttovy diagramy v SVG

Vypracoval:	Miroslav Janáček
Vedoucí bakalářské práce:	Ing. Jan Kelbel
Rok:	2007

Prohlášení

Prohlašuji, že jsem svou bakalářskou práci vypracoval samostatně a použil jsem pouze podklady (literaturu, projekty, SW atd.) uvedené v přiloženém seznamu.

V Praze dne ……. …….

podpis

Poděkování

Na tomto místě bych rád poděkoval vedoucímu této bakalářské práce Ing. Janu Kelblovi za odborné vedení, rady a čas, který mně a této práci věnoval. V průběhu vypracovávání bakalářské práce mě vedl k samostatnosti, věnoval se mi se zájmem a byl mi skutečně všestrannou oporou.

Děkuji.

České vysoké učení technické v Praze, fakulta elektrotechnická

Katedra řídicí techniky

Školní rok: 2006/2007

Zadání bakalářské práce

Student: Miroslav Janáček

Obor: Kybernetika a měření

Název tématu: Ganttovy diagramy v SVG

Zásady pro vypracování:

1. Seznamte se s pojmy z oblasti rozvrhování.
2. Seznamte se s jazyky XML a SVG.
3. Vytvořte program generující Ganttovy diagramy pro rozvrhy popsané v jazyce XML.

Seznam odborné literatury: Dodá vedoucí práce

Vedoucí bakalářské práce: Ing. Jan Kelbel

Datum zadání bakalářské práce: zimní semestr 2006/07

Termín odevzdání bakalářské práce: 15. 8. 2007

L.S.

Prof. Ing. Michael Šebek, DrSc.
vedoucí katedry

Prof. Ing. Zbyněk Škvor, CSc.
děkan

V Praze, dne 5. 3. 2007

Anotace

Tato bakalářská práce se zabývá Ganttovými diagramy, pojmy z oblasti rozvrhování, jazyky XML (*eXtensible Markup Language*) a SVG (*Scalable Vector Graphics*). Hlavní částí práce je program SVG Gantt, který umožňuje v uživatelském prostředí upravovat vzhled rozvrhů.

Program načítá rozvrhy popsané v jazyce XML. Následně uživateli umožňuje měnit parametry vykreslování, vzhled rozvrhu a možnosti zobrazení. Upravený diagram může být uložen ve formátu SVG.

SVG Gantt lze použít pro dávkový převod rozvrhů z XML dokumentů na vektorové obrázky SVG.

Program je napsán v jazyce Java a pro generování SVG kódu byla použita knihovna Batik.

Annotation

This bachelor thesis deals with Gannt Charts, notions from branch scheduling, languages XML and SVG. Main part is software SVG Gantt, which is used to edit the schedule design in a graphical user interface.

Program loads charts defined in language XML. User can also change parameters of drawing, a schedule design and different ways of view. The edited chart is posibble to save in format SVG.

SVG Gantt can be used for bunch conversion schedules from XML documents into vector pictures SVG.

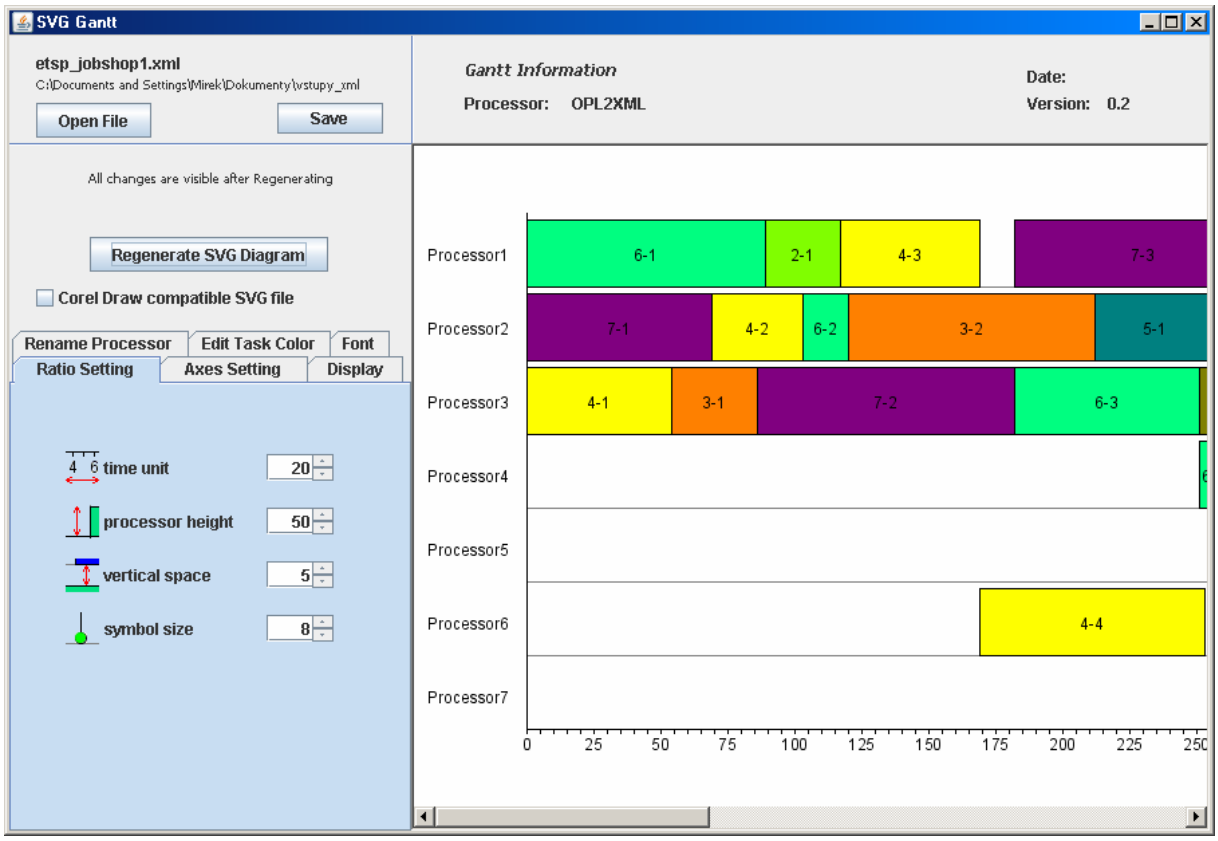
Program is written in Java language and to generating SVG code is used library Batik.

Obsah

Seznam obrázků

1 Úvod

Ganttovy diagramy se používají ke grafickému znázornění rozvrhů. V dnešní době se rozvrhovací algoritmy spouštějí na počítači a pro přehledné znázornění výsledků se zobrazují ve formě Ganttových diagramů. Hlavní částí této bakalářské práce je vytvoření programu **SVG Gantt** (obrázek 1.1), který umožňuje zobrazovaní a editaci Ganttových diagramů popsaných v jazyce XML (*eXtended Markup Language*) a následné uložení diagramů do vektorového grafického formátu SVG (*Scalable Vector Graphics*). Také je možné SVG Gantt použít jako dávkový program pro převod rozvrhů z formátu XML do SVG.



Obrázek 1.1: SVG Gantt

Vstupem programu je textový soubor obsahující data rozvrhu. Tyto data jsou uložena pomocí konstrukcí jazyka XML (kap. 5.1). Takový soubor může být vytvořen například pomocí programu Matlab s nainstalovaným TORCHE Scheduling Toolboxem nebo ručně v jakémkoliv textovém či XML editoru.

Program data zpracuje a následně vykreslí rozvrh v okně s uživatelským rozhraním (*GUI*). Zobrazeny jsou pouze objekty definované ve zdrojovém XML.

V uživatelském prostředí program umožňuje měnit měřítka, barvy úloh, fonty, popisky procesorů, nastavení os. Uživatel si také může zvolit, jaké součásti diagramu chce vykreslovat (čas dostupnosti – *release time*, termín dokončení – *due date*, nejzazší termín dokončení - *deadline*, perioda - *period*, precedence - *precedence*, popisky úloh - *item label*). Zobrazený diagram je možné zvětšovat, posouvat a otáčet.

Výstupem je vektorový grafický formát SVG (kap. 5.2), který je generován pomocí knihovny Batik (kap. 5.3). Formát SVG je prozatím před svým největším rozmachem. V budoucnu měl být velmi rozšířený, zejména na Internetu.

2 Motivace

Program SVG Gantt vychází z diplomové práce Antonína Karolíka *Nástroj na kreslení Ganttových diagramů s využitím Metapostu* [4]. Pan Karolík ve své práci vytvořil XML formát, pomocí kterého mohou být popsány Ganttovy diagramy (tento formát využívá SVG Gantt). A také vyvinul program PS Gantt. Tento výkonný program umožňuje automaticky generovat Ganttovy diagramy ze vstupního XML souboru do formátu EPS nebo PDF. Vzhled diagramu určují kaskádové styly (*CSS*). Program je určen pro dávkový převod a tudíž není moc vhodný pro úpravu vzhledu jednotlivých rozvrhů.

SVG Gantt je zaměřen právě na možnost editace vzhledu rozvrhu v uživatelském prostředí. Po změně některého parametru nebo vlastnosti rozvrhu se změna ihned projeví na zobrazeném diagramu.

Zvolený výstupní formát SVG je snadno upravitelný v grafických nebo dokonce i v textových editorech. Dá se očekávat, že v blízké době se stane velmi rozšířeným.

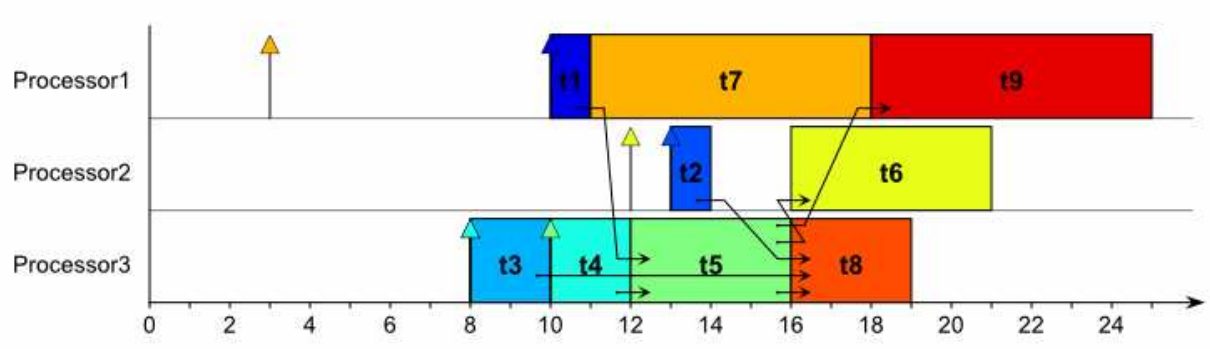
SVG Gantt je napsán v jazyce Java z důvodu snadné přenositelnosti na různé operační systémy.

3 Ganttův diagram

3.1 Úvod o Ganttových diagramech

Henry Laurence Gantt (1861 – 1919), americký strojní inženýr, se zabýval zdokonalováním řízení. Na základě analýzy pracovních postupů v průmyslové výrobě vynalezl Ganttův diagram (*Gantt Chart*). Z původně pruhového (lineárního) se s vývojem moderních nástrojů pro plánování a řízení projektů stal dokonalejší a účelnější diagram, zejména pro různé prezentace síťových grafů a hierarchických datových struktur.

Ganttovy diagramy umožňují přehledně prezentovat aktuální stav projektu, směrný a aktuální plán, především údaje časového rozvrhu, nákladů, práce, financování a zisku na projektu.



Obrázek 3.1: Ganttův diagram

V současné době Ganttův diagram patří k nejpoužívanějším formám prezentace projektových modelů pro plánování a řízení rozsáhlých projektů. Jeho hlavní výhodou je přehlednost projektových ukazatelů na časové ose a přehlednost hierarchické struktury.

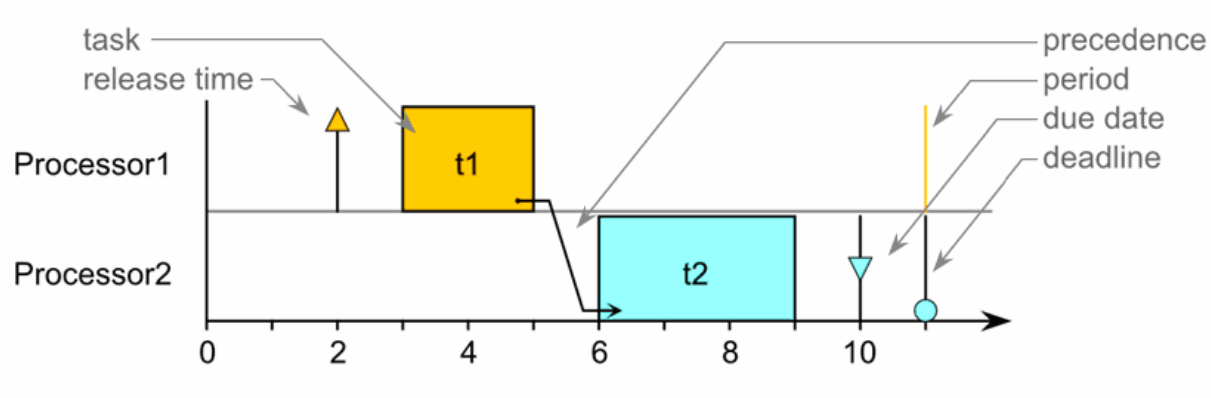
3.2 Popis Ganttových diagramů

V Ganttově sloupcovém diagramu reprezentuje vodorovná osa čas a na svislé ose jsou procesory (*processor*), které vykonávají příslušné úlohy (*task*). **Processor** a **task** jsou základní prvky diagramu [4].

Jednotlivé úlohy jsou v rozvrhu znázorněny obdélníčkem (*taskbox*) s popisem, který reprezentuje identifikátor úlohy. Úlohy, jejichž vykonávání může být přerušeno, se nazývají *preemptivní* a vykreslují se ve více taskboxech stejné barvy. V jazyce XML bývají části preemptivních úloh označovány tagy *item*.

Poloha taskboxu je určena procesorem, který úlohu vykonává, a časem (*start time*), kdy se daná úloha začíná zpracovávat. Délka boxu odpovídá době zpracování (*processing time*), resp. času ukončení (*completion time*).

Obrázek 2.1 zobrazuje jednoduchý Ganttův diagram obsahující všechny prvky, které SVG Gantt podporuje.



Obrázek 3.2: Popis Ganttova diagramu

Další prvky diagramu představují omezení, která mohou být na daný rozvrh kladena. Těmito prvky jsou:

Release time – čas dostupnosti úlohy, doba kdy nejdříve může být úloha započata.

Due date – termín dokončení úlohy, doba kdy by měla úloha být dokončena.

Deadline – nejzazší termín dokončení úlohy, v tuto dobu musí být úloha bezpodmínečně dokončena.

Precedence – vzájemná návaznost jednotlivých úkolů, první úkol (všechny jeho itemy) musí být dokončen před započetím druhého úkol, ke kterému směřuje šipka.

Period – perioda, hodnota používaná v cyklickém rozvrhování (zároveň udává deadline), periodicky se opakuje.

4 Ukázkový příklad

V této kapitole jsou pro lepší pochopení činnosti programu SVG Gantt uvedeny vstupní a výstupní soubory jednoho ukázkového diagramu.

Program nejprve vybere data ze vstupního XML souboru (kap.3.1). Ovšem ne všechny data z XML jsou důležitá pro vzhled diagramu. SVG Gantt nejvíce zajímají *start*, *length*, *processor*, *label*, atd.

Podle těchto dat následně kreslí rozvrh (kap. 3.2). Veškeré nakreslené objekty se vkládají do DOM[1] (*Document Object Model*). Při absenci tagu *releasetime*, *duedate* nebo *deadline* se příslušné značky nekreslí. V případě, že rozvrh neobsahuje informace o barvě úkolu (*graphicparam – color*), program přiřadí úkolům vlastní barvy (barvu úkolu může následně změnit i uživatel).

Při uložení do formátu SVG program pomocí knihovny Batik generuje kód (kap 3.3) z DOM, vzniklého při vykreslování rozvrhu.

4.1 Vstupní XML soubor

Jde o jednoduchý rozvrh s pouze jedním úkolem, složeným z pouze jednoho itemu a dalších tří symbolů (releasetime, duedate a deadline).

```
<?xml version="1.0" encoding="utf-8" ?>
- <matlabdata date="16-Jan-2007 00:00:18" proccessor="TORSCHE Scheduling
Toolbox for Matlab" ver="0.2">
  - <taskset id="TS" ver="1.0">
    - <task id="t1">
        <name>t1</name>
```

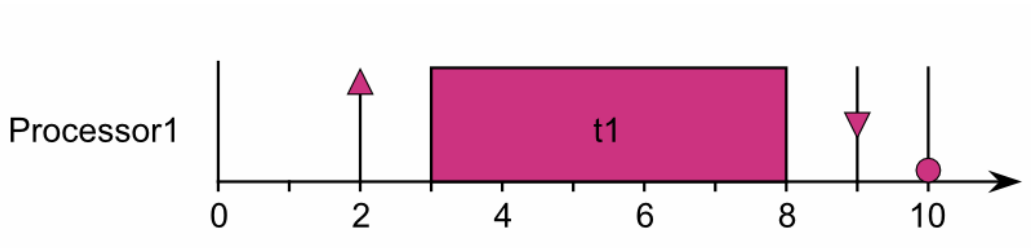
[1] Document Object Model – je jazykově a platformě nezávislé rozhraní pro dynamický přístup k obsahu, struktuře a stylu dokumentu [17].

```
        <proctime>5</proctime>
        <releasetime>2</releasetime>
        <deadline>10</deadline>
        <duedate>9</duedate>
      - <schedule>
       - <item order="1">
          <start>3</start>
          <length>5</length>
          <processor>1</processor>
          <label>t1</label>
         </item>
        </schedule>
      - <graphicparam>
       - <color type="rgb">
          <r>0.8</r>
          <g>0.2</g>
          <b>0.5</b>
         </color>
        </graphicparam>
    </task>
  </taskset>
</matlabdata>
```

Podrobný rozbor všech tagů vstupního XML souboru popisuje diplomová práce Antonína Karolíka [4].

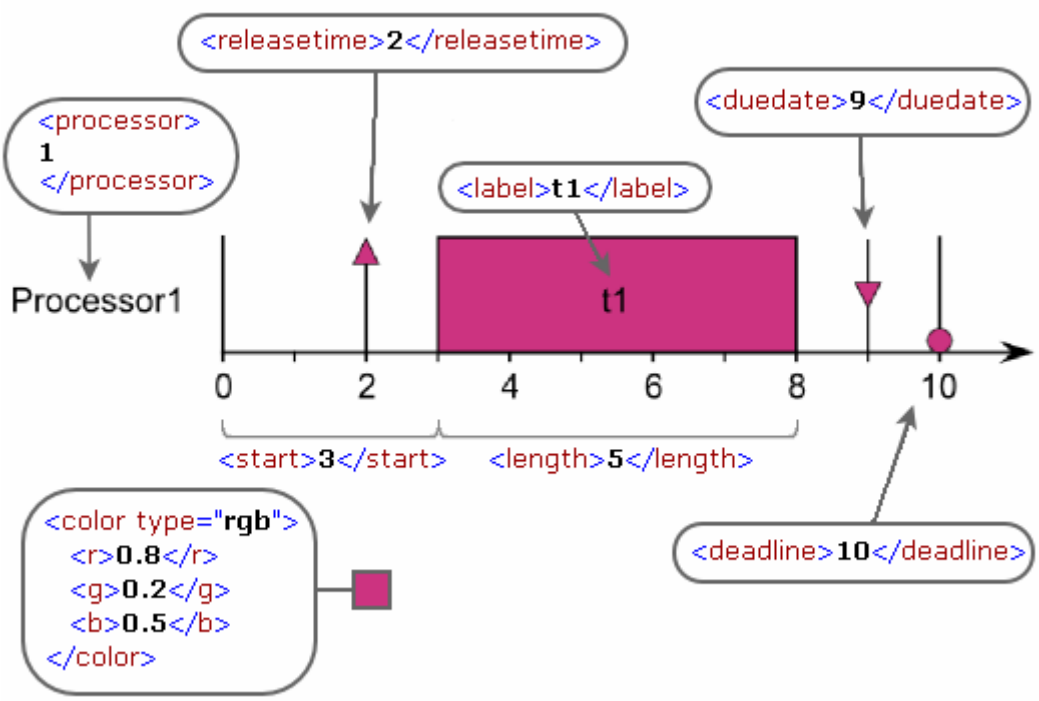
4.2 Výsledný Ganttův diagram

Obrázek 3.1 zobrazuje výsledný diagram nakreslený programem SVG Gantt. Rozvrh se skládá z jednoho itemu a symbolů releasetime, duedate a deadline.



Obrázek 4.1: Výsledný diagram

Na obrázku 3.2 je Ganttův diagram popsán tagy s hodnotami, které mu ve vstupním XML souboru náleží a udávají jeho vzhled.



Obrázek 4.2: Výsledný diagram s XML popisky

4.3 Část výstupního kódu SVG obrázku

Níže uvedená část kódu obsahuje informace, jak mají být součásti rozvrhu (item, releasetime, duedate, deadline a item label) zobrazeny. Osy, popisky os, atd. jsou v kódu vynechány.

```
<svg ...
  <g>
  ...
        <!-- item -->
    <rect x="159" y="52" fill="rgb(204,51,128)" width="125"
      height="40" stroke="none"/>
    <rect fill="none" width="125" x="159" height="40" y="52"/>
        <!-- release time -->
    <line y2="92" fill="none" x1="134" x2="134" y1="60"/>
    <polygon fill="none" points=" 134 53 138 61 130 61"/>
    <polygon fill="rgb(204,51,128)" points=" 134 53 138 61 130 61"
      stroke="none"/>
        <!-- due date -->
    <line y2="68" fill="none" x1="309" x2="309" y1="52"/>
    <line y2="92" fill="none" x1="309" x2="309" y1="76"/>
    <polygon fill="none" points=" 309 76 313 68 305 68"/>
    <polygon fill="rgb(204,51,128)" points=" 309 76 313 68 305 68"
```

```
      stroke="none"/>
        <!—- deadline time --!>
    <line y2="84" fill="none" x1="334" x2="334" y1="52"/>
    <circle fill="none" r="4" cx="334" cy="88"/>
    <circle fill="rgb(204,51,128)" r="4" cx="334" cy="88"
      stroke="none"/>
        <!—- item label --!>
    <text xml:space="preserve" x="216" y="78"
      stroke="none">t1</text>
    ...
  </g>
</svg>
```

5 Použité technologie

5.1 XML

XML (*eXtensible Markup Language* - rozšiřitelný značkovací jazyk) [15, 18] je obecný značkovací jazyk, který byl vyvinut a standardizován konsorciem W3C[1] (World Wide Web Consortium). Umožňuje jednoduše, přehledně a efektivně pracovat s daty. Jeho uplatnění je nejen na internetu, ale prakticky všude, kde je potřeba pracovat se strukturovanými daty.

Tento jazyk obsahuje pouze čistá data bez informací o jejich zobrazení. Jeho výhodou je, že není vázán na žádný software. Je možné ho vytvořit i v nejjednodušším textovém editoru a je vhodný pro převod do řady jiných formátů. Možnost přenosu dat mezi aplikacemi je také jeho hlavní přínos.

XML a HTML (*HyperText Markup Language* – značkovací jazyk pro hypertext) jsou si velmi podobné díky stejnému předchůdci SGML (*Standard Generalized Markup Language* – univerzální značkovací jazyk). Hlavní rozdíl mezi těmito dvěma jazyky je v tom, že HTML má definovanou sadu značek, které udávají význam označených dat. V XML nejsou žádné definované značky. Každý uživatel nebo program si může vymyslet vlastní. Pro definování zobrazení dat z XML pak můžou být použity stylové jazyky (CSS, XSL ...). Tím jsou oddělená čistá data od způsobu jejich interpretace.

K usnadnění porozumění XML mezi více programy slouží jazyk DTD (*Document Type Definition* – definice typu dokumentu), který obsahuje informace o tom, co který tag v XML označuje. Existují i standardní DTD, mezi které se dá zařadit např. HTML.

[1] World Wide WEB Consorcium – je mezinárodní konsorcium, jehož cílem je rozvíjet World Wide Web (celosvětovou síť počítačů = web) pro vylepšení jeho potenciálu vývojem protokolů a směrnic [21].

Ukázka XML kódu:

```
<?xml version="1.0" ?>
- <xml>
  - <osoba id="01">
      <jmeno>Martin</jmeno>
      <prijmeni>Nahodil</prijmeni>
      <vek>2...9</vek>
    </osoba>
    <osoba id="02">
    ...
    </osoba>
  </xml>
```

5.2 SVG

SVG (*Scalable Vector Graphics* - škálovatelná vektorová grafika) [6, 14] vychází z XML a je to formát popisující dvojrozměrnou vektorovou grafiku. Za jeho vznikem také stojí také konsorcium W3C a měl hlavně sloužit pro zobrazovaní vektorové grafiky na Internetu. Jeho primární účel zatím není moc rozšířený, protože ne všechny internetové prohlížeče zobrazují všechny vlastnosti a funkce stejně.

Typy objektů:

- vektorové tvary (vector graphic shapes – obdélník, kružnice, elipsa, úsečka, lomená čára, mnohoúhelník a křivka)
- textové objekty
- rastrové obrazy (raster images)

Všechny tyto objekty lze seskupovat, formátovat pomocí atributů nebo stylů CSS (*Cascading Style Sheets*) a polohovat pomocí obecných prostorových transformací. SVG podporuje skriptovací jazyky počínaje ECMAScript a plně podporuje animace, dále ořezávání objektů alpha masking, interaktivitu, filtrování obrazu (konvoluce, displacement mapping, atd.

Jelikož SVG vychází z jazyka XML, stačí pro vytvoření jakéhokoliv SVG obrázku pouze textový editor. Pro pohodlnější tvorbu a editaci existují desítky grafických vektorových editorů.

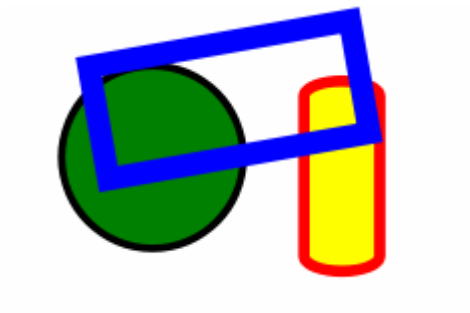
Ukázkový kód popisující obrázek 5.1:

```
<?xml version="1.0" standalone="no"?>
<!DOCTYPE svg PUBLIC "-//W3C//DTD SVG 1.1//EN"
  "http://www.w3.org/Graphics/SVG/1.1/DTD/svg11.dtd">
<svg width="6cm" height="4cm" viewBox="0 0 600 400"
     xmlns="http://www.w3.org/2000/svg" version="1.1">
   <title>Ukazka</title>

   <rect x="400" y="100" width="100" height="250"
      fill="yellow" stroke="red" stroke-width="15" rx="50" ry="20" />

   <circle cx="200" cy="200" r="120"
      fill="green" stroke="black" stroke-width="10" />

   <g transform="translate(0 0) rotate(-10)">
      <rect x="100" y="100" width="350" height="150"
         fill="none" stroke="blue" stroke-width="30" />
   </g>
</svg>
```



Obrázek 5.1: Ukázka SVG

5.3 Batik SVG Toolkit

5.3.1 Úvod o Batik SVG Toolkit

Batik [5, 8, 9, 13] je **open-source software**[1] vytvořený společností **Apache Software Foundation.**

Batik SVG Toolkit je na Javě založená skupina nástrojů pro aplikace a applety, které pracují s obrázky ve formátu SVG (*Scalable Vector Graphics*). Obsahuje nástroje pro zobrazovaní, generovaní, editaci a konverzi SVG.

Účelem Batiku je poskytnout programátorům základní moduly, které můžou být použity dohromady nebo samostatně.

Moduly:

SVG DOM API – objektový model dokumentu (*Document Object Model*) s rozhraním pro programování aplikací (*Application Programming Interface*) pracujících s XML dokumenty. Definuje logickou strukturu dokumentu, způsob přístupu a manipulace s dokumentem.

SVG Parser – modul pro syntaktický přepis složitých SVG atributů.

Scripting module – modul pro přidávání skriptů (ECMAScript) do SVG dokumentů.

SVG Generator – modul, který umožňuje používat grafickou třídu Graphics2D pro malovaní objektů a následně z ní generovat strukturu SVG.

Swing SVG component – obsahuje swingové komponenty pro Javu.

Transcoder module – poskytuje obecný nástroj pro překódování vstupu na výstup.

Kromě modulů vytvořil Batik i pár samostatných programů pro práci s SVG.

Nástroje a aplikace:

Squiggle – prohlížeč SVG obrázků.

SVG Rasterier – nástroj pro převod SVG souborů na rastrové obrázky.

[1] Open-source software - software s volně přístupným (technicky i legálně) zdrojovým kódem.

SVG Font converter – aplikace na převod TrueType Font (standart pro popis vektorových písem) do fontu SVG.

SVG Pretty Printer – nástroj pro úpravu syntaxe SVG dokumentů.

5.3.2 Moduly použité v programu SVG Gantt

Pro vytvoření programu SVG Gantt jsem použil tyto moduly z knihovny Batik SVG Toolkit:

SVG DOM API – pro uložení objektového modelu dokumentu.

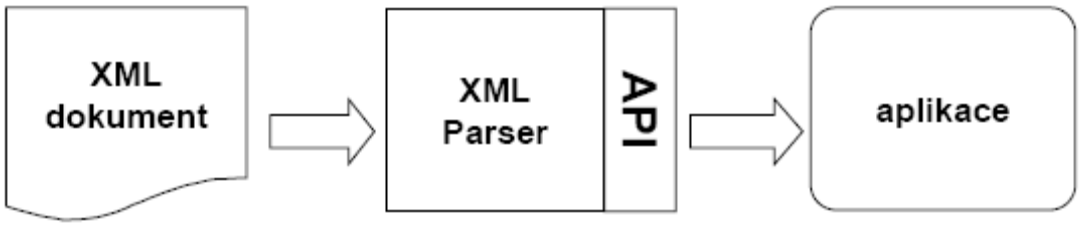
SVG Generator – k vykreslování objektů a generování dokumentu.

Swing SVG component – komponentu *JSVGCanvas* jako plátno pro vykreslování a zobrazování

5.4 XML parser

5.4.1 Úvod o parserech

Pro získání dat z dat z XML dokumentů je potřebný XML parser [10, 11, 16]. Samozřejmě je též možné číst XML dokument znak po znaku a vyhodnocovat, zda se jedná o atribut, element nebo komentář. Pro tuto práci však existuje spousta knihoven, které usnadňují čtení XML. Těmto knihovnám se říká parser.



Obrázek 5.2: Schéma práce XML parseru

Parsery při procházení dokumentu kontrolují, jestli je dokument správně formulovaný (*well-formed*) – kontrolují spárované tagy, hodnoty atributů. Pomocí DTD (kap.5.1.1) je možné kontrolovat, zda XML dokument odpovídá schématu, nebo-li provádět *validaci* dokumentu.

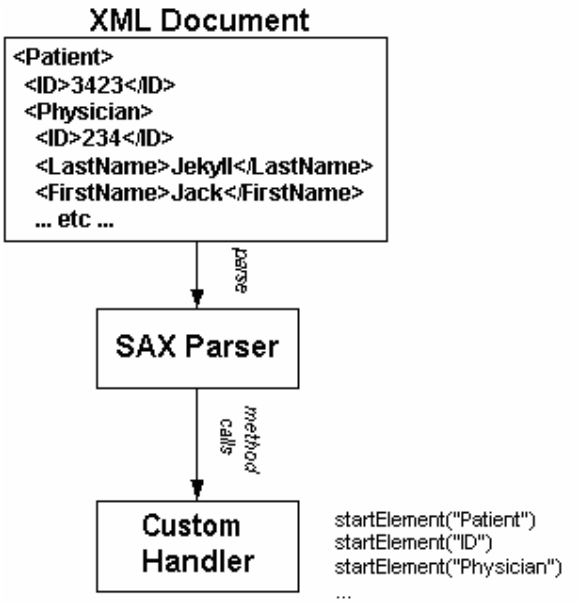
Syntaktická analýza XML dokumentů se nazývá parsování (*parsing*). Parser čte dokument a za menšítkem „<“ očekává název elementu, vybírá hodnotu a název atributů. Ze získaných dat vytváří abstraktní model XML dokumentu – tzv. *infoset*.

Různé parsery se liší přístupem k infosetu. Základní dva typy parserů jsou SAX (*Simple API for XML*) a DOM (*Document Object Model*).

5.4.2 SAX parser

Tento parser je řízený událostmi. Postupně čte dokument a podle přečtených elementů a atributů volá naše funkce, kterým předává parametry (obrázek 5.3).

Toto zpracování má dvě velké výhody – rychlost a malé paměťové nároky. Nevýhodou je nutnost zpracování celého XML během jednoho sekvenčního průchodu.

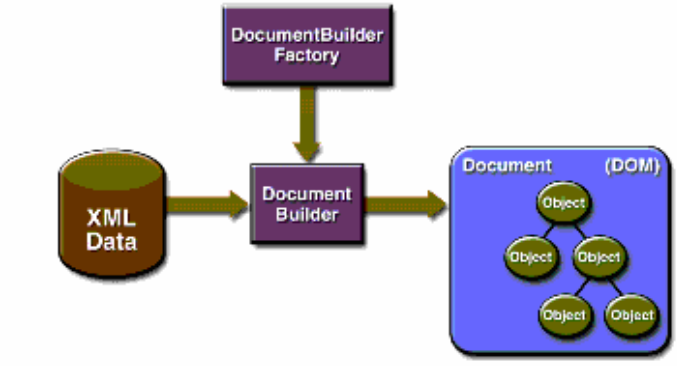


Obrázek 5.3: SAX parser, událostmi řízené rozhraní

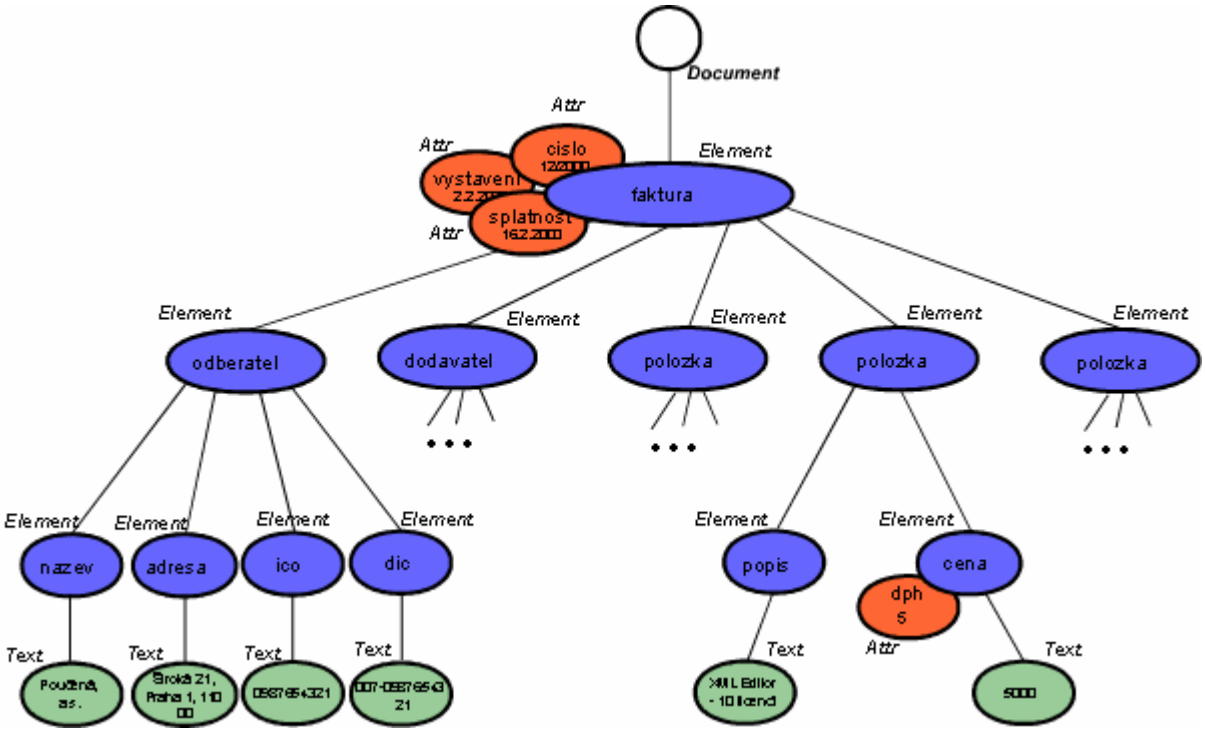
5.4.3 DOM parser

DOM parser načte celý XML dokument a ze získaných dat vytvoří v paměti stromovou strukturu dokumentu (obrázek 5.5).

Výhoda DOM parseru je jednodušší zpracování a hlavně možnost libovolného a opakovaného procházení stromové struktury. Je to ovšem vykoupeno nižší rychlostí zpracování a velkou paměťovou náročností.



Obrázek 5.4: DOM parser



Obrázek 5.5: DOM parser, stromová struktura reprezentující XML dokument

5.4.4 Další parsery

Během posledních několika let se ukázalo, že použití parserů SAX a DOM je zbytečně komplikované, proto je snaha vytvořit rozhraní, které by práci s XML dokumenty co nejvíce usnadnilo. Příkladem může být jazyk *XPath*, který dokáže pomocí jednoduchých dotazů, podobných jako při popisu cesty v adresářové struktuře, vybrat jen určité uzly DOM stromu (tedy jen části ze struktury XML dokumentu) a tím celý proces výrazně zjednodušit. V oblasti sekvenčních parserů se začínají objevovat tzv. *pull-parsery*, pomocí nichž je možno dokument zpracovávat po sekvenčních částech.

5.4.5 Parser použitý v programu SVG Gantt

V programu SVG Gantt jsem použil DOM parser z důvodu jednoduššího zpracování dokumentu. Paměťová náročnost a rychlost zpracování se u dokumentů popisující Ganttovy diagramy negativně neprojevila.

Konkrétně jsem upravil zdrojový kód od Martina Glogara [11]. Program SVG Gantt dokáže správně zpracovat jen XML dokument odpovídající DTD popsané v diplomové prácí Antonína Karlíka [4]. Dokumenty s jinou definici typu je nutné převést. Toto DTD odpovídá dokumentům generovaných programem Matlab s TORCHE Scheduling Toolboxem.

6 Popis Programu

6.1 Vnitřní popis

Program SVG Gantt je napsán v jazyce Java. K jeho vývoji bylo použito programovací prostředí NetBeans IDE 5.5. Pro generování výstupu ve formátu SVG program využívá knihovnu Batik (kap. 5.3).

Po spuštění se program podle počtu parametrů rozhoduje, v jakém módu se spustí. Při zadání dvou parametrů (vstupního a výstupního souboru) program běží v módu dávkového převodu (kap 6.2.5). Po jiném zadání parametrů se program pouští s uživatelským rozhraním.

Vstupní XML soubor je parsování třídou **ParseXMLFile**, která vznikla upravením zdrojového kódu od Martina Glogara [11]. Data důležitá pro vykreslení Ganttova diagramu jsou ukládána do objektů (Task, ScheduleItem a PrecedenceConstrains) a některá data jsou rovnou předány třídě Creator, která se stará o veškerou práci programu.

Po uložení informací z XML do objektů, třída Creator postupně volá funkce, které počítají umístění a vykreslují symboly na plátno (*JSVGCanvas*). Nejdříve se vykreslují osy, a obdélníčky znázorňující itemy všech úkolů. Pokud rozvrh obsahuje symboly release time, due date, deadline nebo period, pak je vykresluje hned po itemech a následně případné precedence. U precedencí program počítá nejvhodnější místo pro vykreslení, aby se v rozvrhu pokud možno nepřekrývaly. Na závěr vykreslování se doplňují popisky os a názvy itemů.

Popisky itemů mohou obsahovat horní a dolní index, při použití syntaxe sytému **T$_E$X**[1].

Vykreslování se provádí pomocí knihovny *Graphics2D*. Před vykreslením prvního symbolu daného rozvrhu program vytvoří SVG dokument, do kterého se ukládají všechny nakreslené symboly.

[1] T$_E$X – systém pro sazbu textu (např. pro psaní složitých matematických vzorců)

O ukládání se pak postará knihovna Batik, která z SVG dokumentu vygeneruje kód ;v jazyku SVG.

Při editaci rozvrhu uživatelem se přepisují hodnoty uložené v objektech a znovu se provádí vykreslování.

6.1.1 Objekty

6.1.1.1 Task

Do této třídy se ukládají data o vlastnostech každého úkolu. Jsou to data jako id úkolu, čas dostupnosti, termín dokončení, nejzazší termín dokončení, perioda, barva úloh. Každý úkol obsahuje další objekty třídy **ScheduleItem**. Další data uložená v tomto objektu slouží pro řízení vykreslování.

6.1.1.2 ScheduleItem

Mezi třídou **Task** a **ScheduleItem** se vyskytuje vazba typu *kompozice* (objekt ScheduleItem nemůže existovat samostatně bez objektu Task). V objektech této třídy se uchovávají informace o itemech daného úkolu a také informace důležité pro generování diagramu. Data o itemech jsou ID itemu, startovní čas, doba trvání, přiřazený processor, popisek.

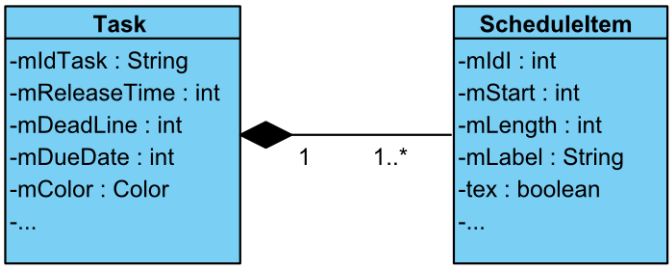
6.1.1.3 PrecedenceConstrains

Tato třída je *asociační* pro objekty třídy **Task** (popisuje vztah mezi dvěma objekty třídy Task). Z XML se do ní vkládají ID počátečního a koncového úkolu. Pro usnadnění určování pozice vykreslovaných symbolů a počítání nejvhodnějšího vedení symbolu jsou v této třídě uloženy další důležité informace.

6.1.2 UML modely

6.1.2.1 Kompozice tříd Task a ScheduleItem

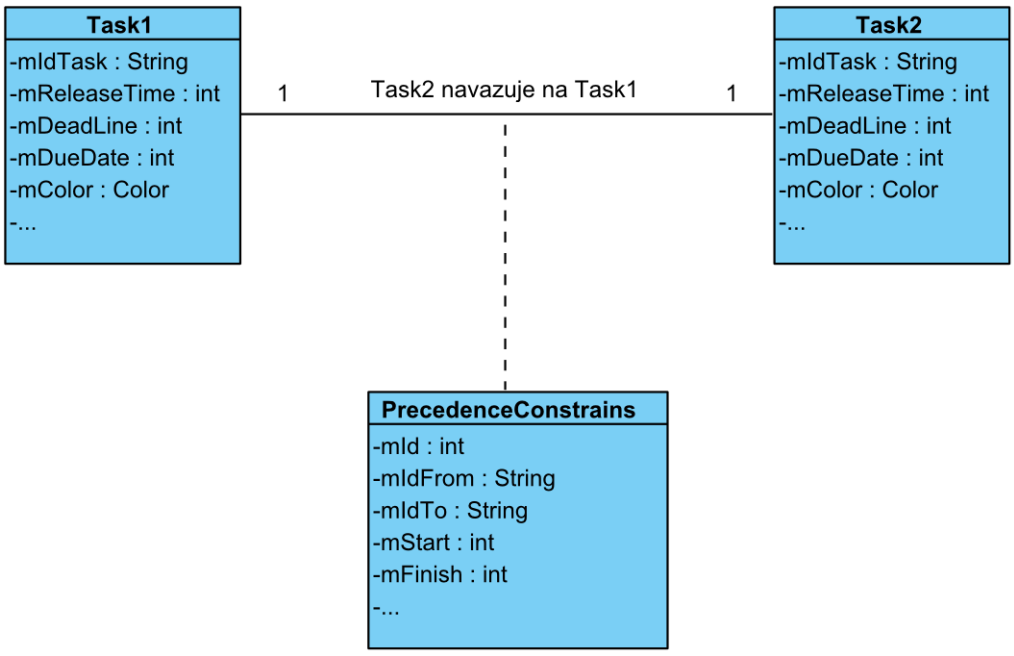
Třída **Task** a **ScheduleItem** mají mezi sebou vztah typu *agregace* (třída ScheduleItem je součástí třídy Task) a dokonce se jedná o speciální případ zvaný *kompozice* (třída ScheduleItem nemůže existovat bez třídy Task). Tento vztah je znázorněn na obrázku 6.1.



Obrázek 6.1: Kompozice tříd

6.1.2.2 Asociace tříd Task a PrecedenceConstrains

Precedence, které udávají vztah mezi dvěma úkoly, jsou uloženy v třídě **PrecedenceConstrains** a tudíž vztah k třídě **Task** je *asociace* (obrázek 6.2).

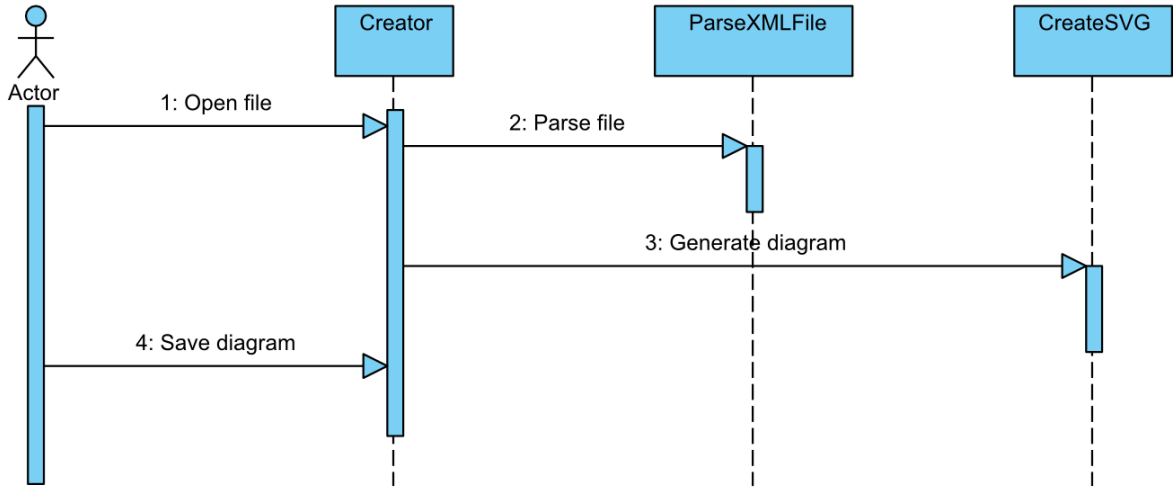


Obrázek 6.2: Asociační třída

6.1.2.3 Sekvenční diagram

Tento sekvenční diagram (obrázek 6.3) znázorňuje pracovní postup programu, když uživatel (Actor) zvolí otevření souboru a následně ho uloží.

Krok	Role	Akce
1	Actor	zvolí vstupní XML soubor
2	SVG Gantt	vybere a zpracuje data z XML souboru
3	SVG Gantt	zobrazí diagram
4	Actor	dá pokyn k uložení
5	SVG Gantt	vygeneruje výstupní SVG soubor

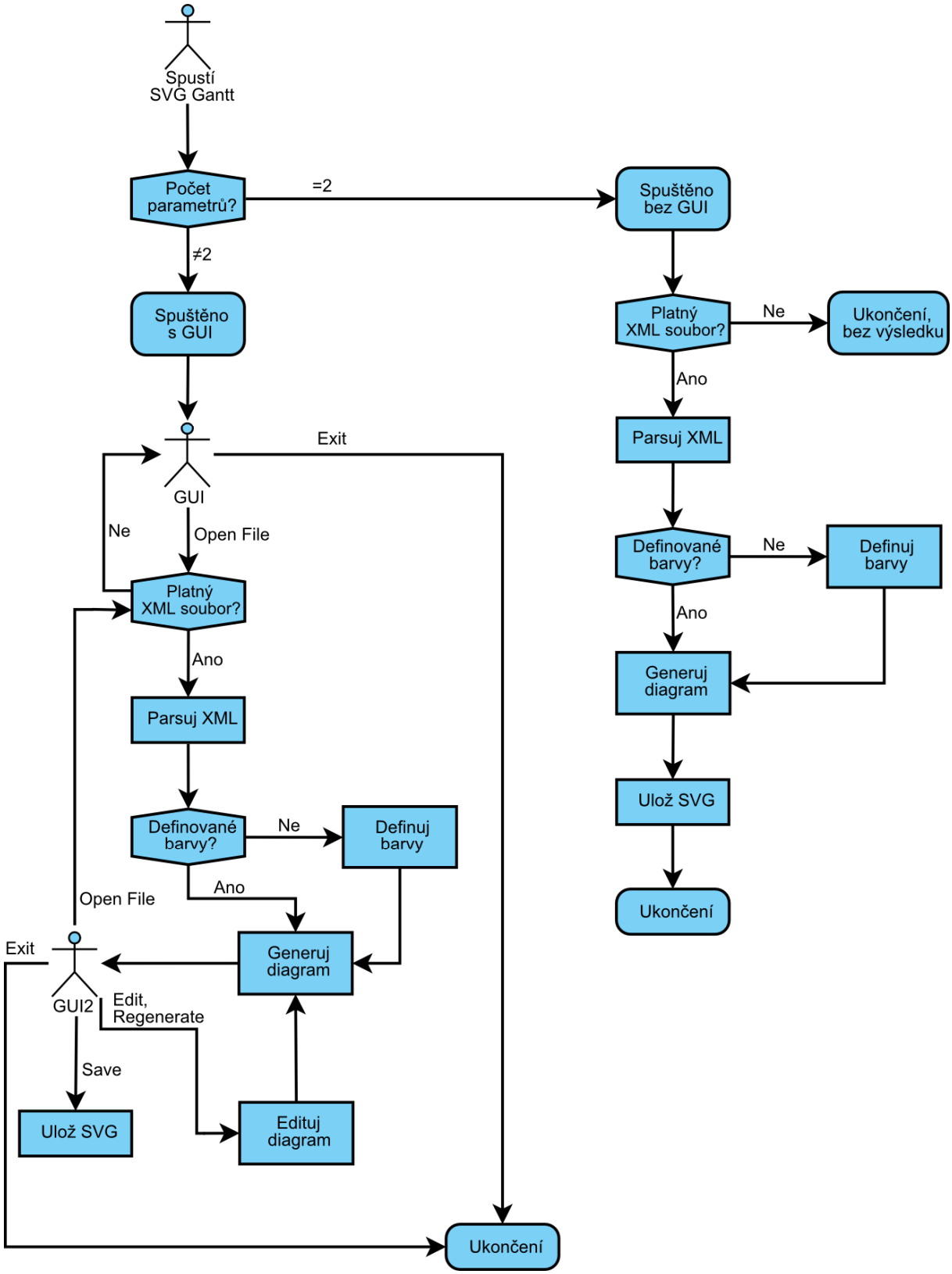


Obrázek 6.3: Sekvenční diagram

6.1.2.4 Vývojový diagram

Vývojový diagram (obrázek 6.4) znázorňuje reakce programu na všechny možné podněty od uživatele.

Celá pravá větev znázorňuje chování programu při spuštění v módu dávkového převodu rozvrhů. Zbytek popisuje reakce programu na příkazy od uživatele při spuštění programu s GUI.



Obrázek 6.4: Vývojový diagram

6.2 Popis užívání programu

6.2.1 Spuštění programu

Program je možné spustit více způsoby. Nejjednodušší a univerzální způsob je spustit soubor „**run.bat**“. Pokud nemá uživatel v úmyslu otvírat velmi rozsáhlé rozvrhy, je stejně vhodně spustit program souborem „**SVG_Gantt.jar**“.

U velkých souborů se může stát, že se všechny vykreslované objekty nevejdou do paměti programu. Tomu se dá předejít např. parametrem „–Xmx512m“ při spuštění z příkazové řádky (tento parametr zvětší paměť pro program na 512MB). S tímto parametrem se spouští program právě souborem „run.bat“.

Další způsob je spuštění z příkazové řádky:

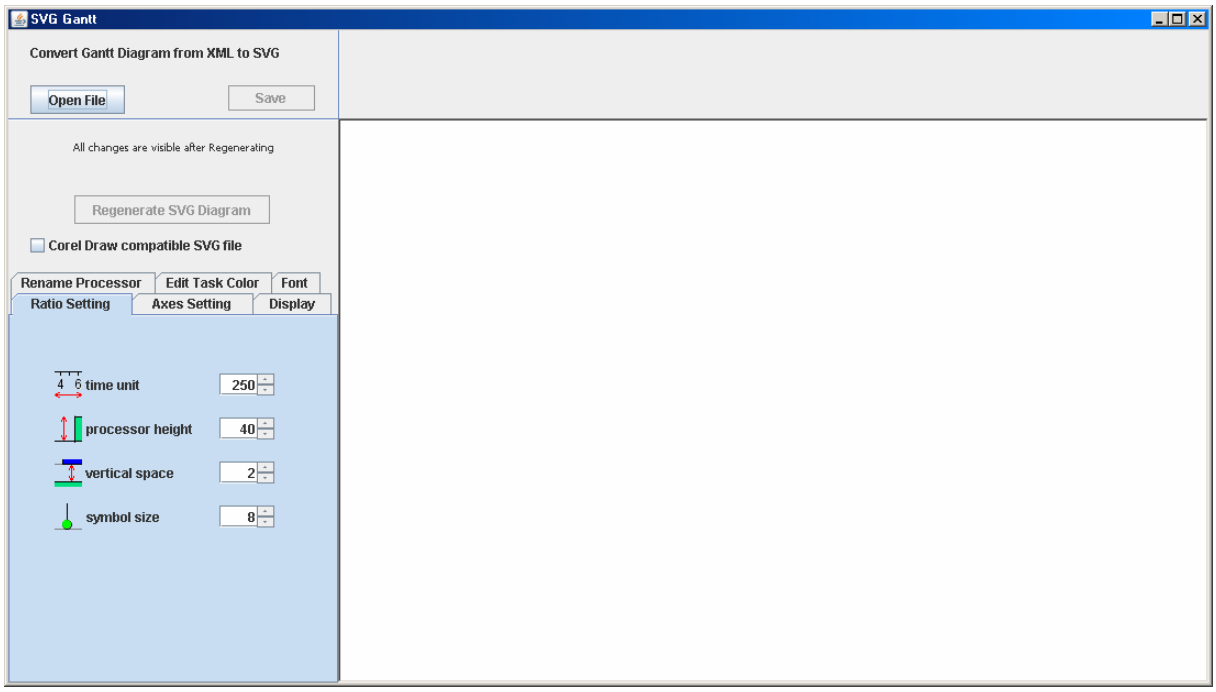
```
java -jar -Xmx512m SVG_Gantt.jar
```

(parametr –Xmx512m je jen pro zvětšení paměti, není nutný pro standardní rozvrhy)

Poslední způsob je spuštění bez uživatelského rozhraní pro dávkovou úpravu (kap. 6.2.5).

6.2.2 Základní operace

Po spuštění programu se uživateli zobrazí GUI (obrázek 6.5). Pro otevření rozvrhu je nutné kliknout na tlačítko „**Open File**“. Program je možné kdykoliv ukončit křížkem v pravém horním rohu.



Obrázek 6.5: Screenshot – spuštění programu

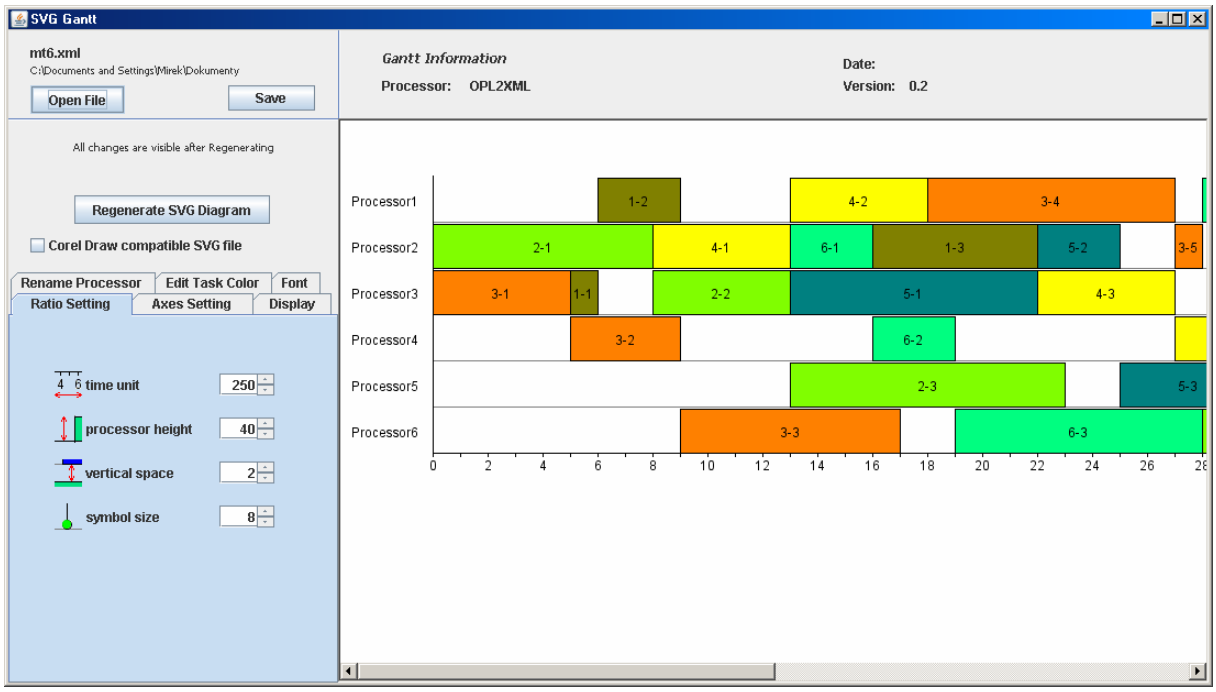
Pokud byl otevřen XML soubor, který není validní, pak se nad tlačítkem „**Regenerate SVG Diagram**“ zobrazí hláška „**This XML File is NOT VALID**“ a na plátno se nic nevykreslí. V případě validního souboru program vykreslí Ganttův diagram na plátno (obrázek 6.6).

Uživatel pak může editovat vzhled a zobrazení rozvrhu. Po změně vzhledu je nutné kliknout na tlačítko „**Regenerate SVG Diagram**“ pro znovu vykreslení rozvrhu.

Další volbou uživatele může být otevření nového souboru nebo uložení zobrazeného rozvrhu. Uložení se provádí tlačítkem „**Save**“.

Speciální zaškrtávací volba „**Corel Draw compatible SVG file**“ mění způsob vykreslení rozvrhu, aby byl správně zobrazen i v programu Corel Draw (kreslí rámečky itemu, jako úsečky místo obdélníků).

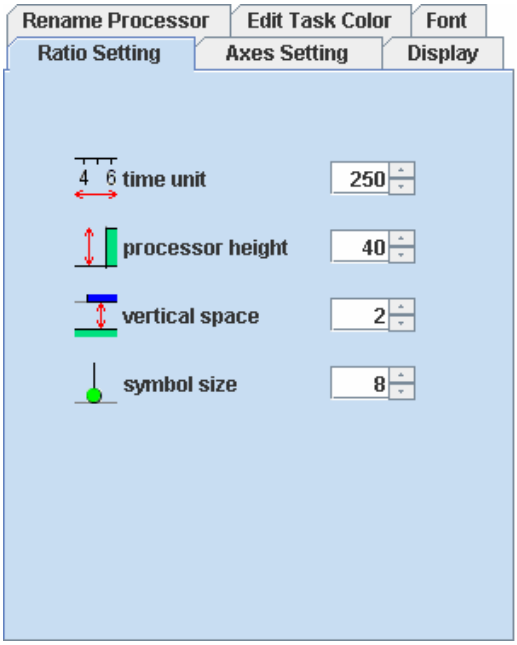
Pokud v otevřeném XML souboru nejsou definované barvy úkolů, program je sám přiřadí a nad tlačítkem „**Regenerate SVG Diagram**“ zobrazí hlášku „**Non-Defined Task Color was automatically set**“.



Obrázek 6.6: Screenshot – vykreslení rozvrhu

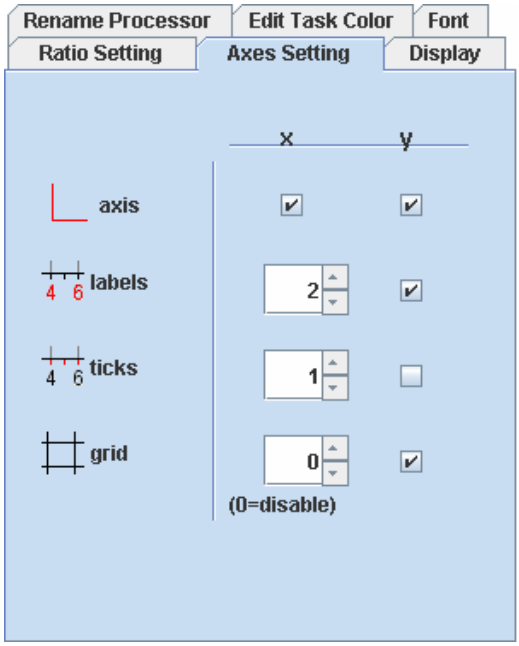
6.2.3 Editace vzhledu

Pro editaci vzhledu slouží šest záložek v GUI. První záložka „**Ratio Setting**“ umožňuje měnit měřítka a velikosti rozvrhu.



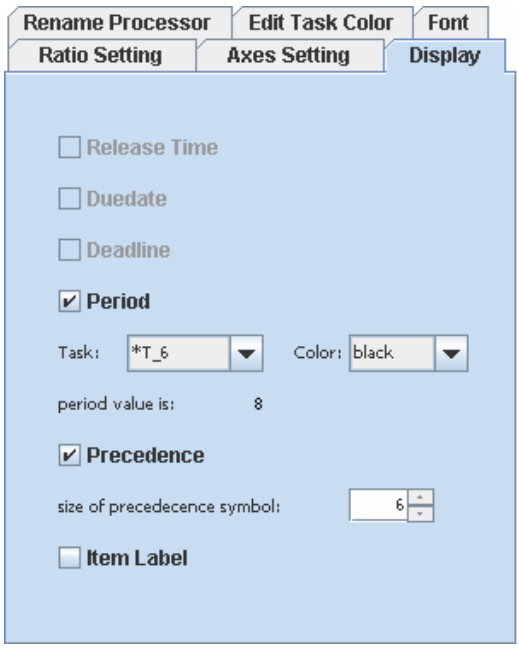
Obrázek 6.7: Screenshot –Ratio Setting

Na kartě „**Axes Setting**“ si uživatel volí nastavení os. Může měnit zobrazení popisků os, mřížky, samotných os a ticků (krátké čárky na osách).



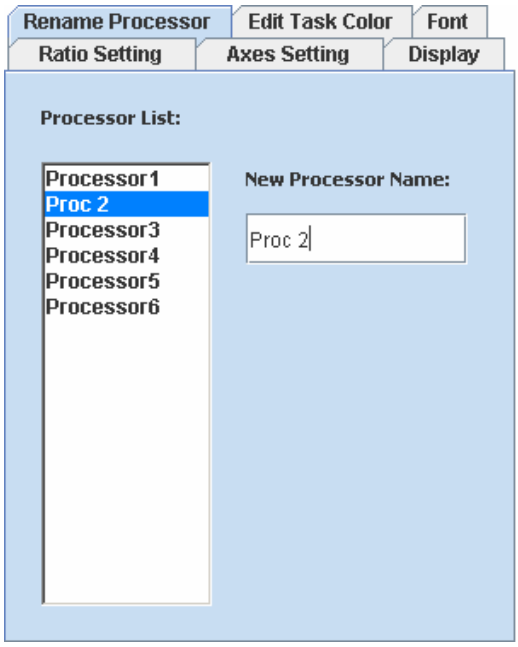
Obrázek 6.8: Screenshot - Axes Setting

Záložka „**Display**“ obsahuje volby, které symboly mají být vykresleny. V případě „**Period**“ je možné měnit barvu symbolu pro zvolený úkol. A u „**Precedence**“ lze zvolit velikost symbolu.



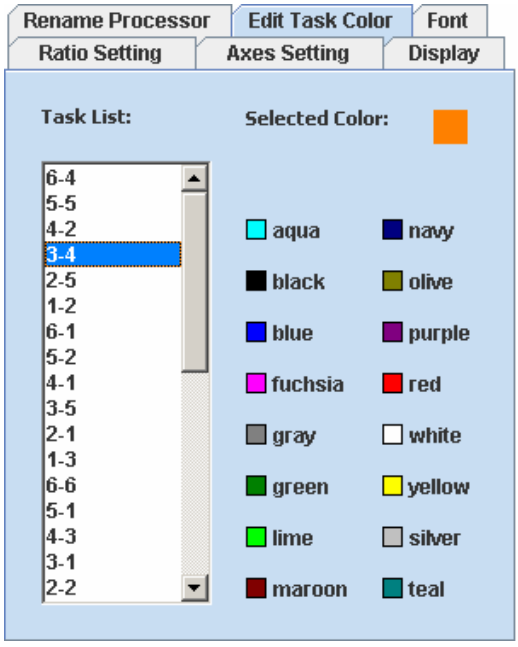
Obrázek 6.9: Screenshot – Display

„**Rename Processor**“ umožňuje přejmenování procesorů vykonávajících úlohy.



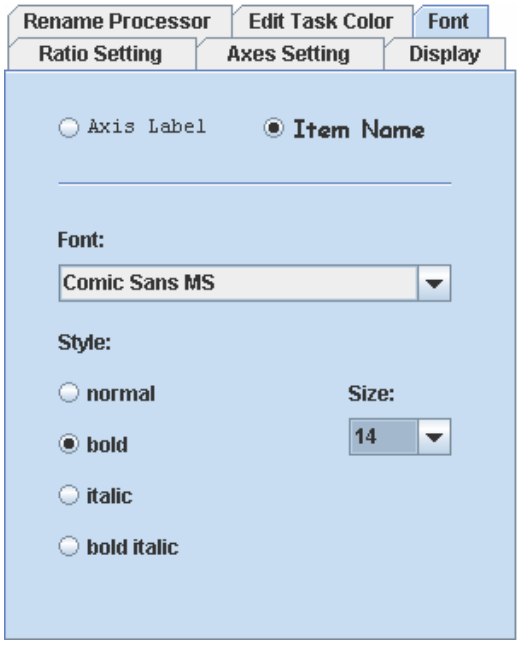
Obrázek 6.10: Screenshot – Rename Processor

Na záložce „**Edit Task Color**“ může uživatel zvolit úkol a změnit barvu, kterou se itemy a symboly vykreslují.



Obrázek 6.11: Screenshot – Edit Task Color

Karta „**Font**“ umožňuje měnit fonty popisku os i popisků itemů.



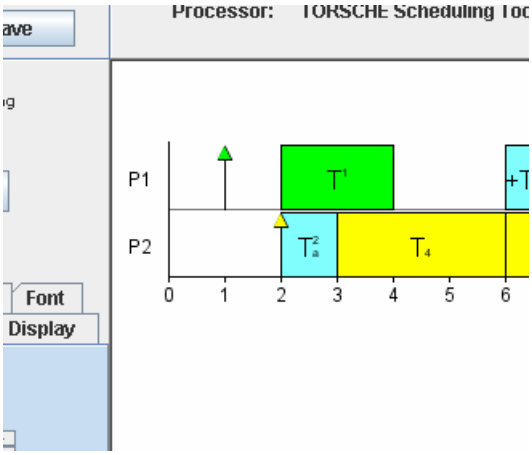
Obrázek 6.12: Screenshot – Font

6.2.4 Možnosti zobrazení

Plátno (*JSVGCanvas*), na které se rozvrh vykresluje umožňuje manipulaci s rozvrhem. Kombinací současného stisknutí klávesy Shift nebo Ctrl, levého nebo pravého tlačítka myši a pohybem myši, lze rozvrh posouvat, otáčet, přibližovat, oddalovat.

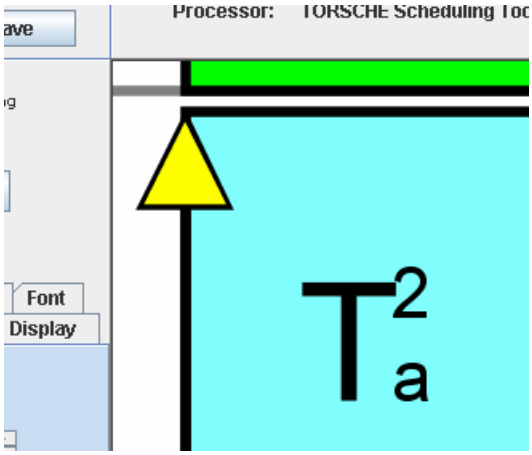
Na obrázku 6.13 je zobrazen výřez z okna SVG Ganttu při normálním pohledu (bez manipulace). Na obrázku 6.14 je rozvrh přiblížen a posunut. A obrázek 6.15 otočený rozvrh.

Změna zobrazení na plátně nemá vliv na vzhled rozvrhu uloženého do SVG. Po stisknutí klávesy „**Regenerate SVG Diagram**“ se rozvrh překreslí a zobrazí se bez změny zobrazení (toto tlačítko je vhodné pro resetování zobrazení).

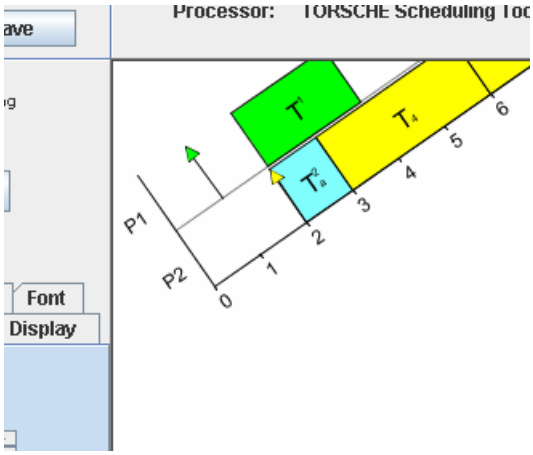


Obrázek 6.13: Zobrazení před manipulací

klávesa	*tlačítko myši*	*událost*
Shift	levé	posouvání rozvrhu
Shift	pravé	zoomování rozvrhu
Ctrl	levé	volba výřezu pro zvětšení
Ctrl	pravé	libovolné otáčení



Obrázek 6.14: Zoomování a posun

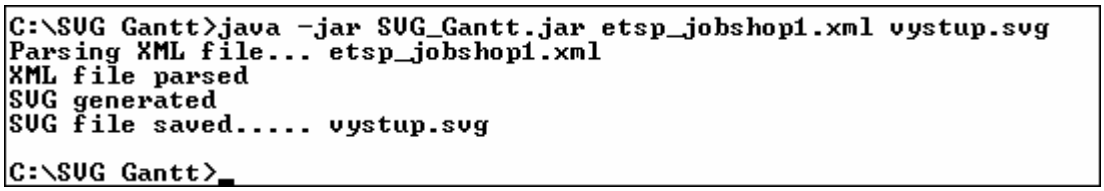


Obrázek 6.15: Otáčení rozvrhu

6.2.5 Dávkový převod rozvrhů

Při spuštění programu z příkazové řádky s parametry *vstupní XML soubor* a *výstupní SVG soubor*, program nespouští uživatelské prostředí (GUI). *Vstupní XML soubor* se rozparsuje, vygeneruje se kód a uloží se jako *výstupní SVG soubor*.

Takto se dá program použít pro dávkový převod rozvrhů.



Obrázek 6.16: Spuštění z příkazové řádky

7 Závěr

Hlavní částí této bakalářské práce je vytvoření programu SVG Gantt. Další část práce se zabývá teoretickým popisem Ganttových diagramů, jazyky XML a SVG a rozborem způsobu naprogramování programu SVG Gantt.

Přes komplikovaný začátek se podařilo vytvořit program SVG Gantt, který načítá Ganttovy diagramy z XML souborů, umožňuje v uživatelském prostředí editovat vzhled rozvrhů a ukládat rozvrhy do grafického vektorového formátu SVG. Program také umožňuje dávkově převádět Ganttovy diagramy z XML do SVG.

Výstupní SVG soubory mají trošku nešikovný formát (většina objektů je popsána dvakrát – okraj a výplň). Je to způsobeno použitím modulu SVG Generator a grafické třídy Graphics2D. Toto řešení má ovšem velkou výhodu ve zjednodušení programu. Na zobrazení SVG souborů nebo jejich editaci v grafických editorech nemá „zdvojení“ některých tagů žádný vliv. Negativně se to projeví na velikosti souboru, ale i v případě velmi rozsáhlého rozvrhu se sto úkoly je velikost výstupního souboru pouze 130kB (velikost vstupního XML je 51kB), což podle mě není znepokojivé.

Všechny rozvrhy, které byly k dispozici, program bez problému dokázal zobrazit, editovat a uložit.

Jediným problémem, který program způsoboval, bylo přetečení standardně přiřazené paměti při zpracovávání rozsáhlých rozvrhů. Při vykreslování na plátno se nevešly všechny objekty do paměti, program nahlásil chybu a nevykreslil rozvrh (uložit rozvrh do SVG souboru ale možné bylo). Tato chyba je odstraněna vytvořením spouštěcího souboru „run.bat“, který programu zajistí přirazení většího místa v paměti.

Program nebyl testován v jiných operačních systémech než Microsoft Windows, ale díky zvolenému programovacímu jazyku Java lze předpokládat, že SVG Gantt bude pracovat bezchybně ve všech operačních systémech, pro které bude zkompilován.

SVG Gantt by mohl být v budoucnu rozšířen o podporu kaskádových stylů CSS.

Využití tohoto programu může být všude, kde je potřeba editovat vzhled Ganttových diagramů nebo převádět diagramy do SVG. Například při výuce či pro práci lidí na Katedře řídící techniky FEL ČVUT.

Seznam použité literatury

[1] HEROUT, Pavel. *Java – grafické uživatelské prostředí a čeština.* 1. vyd. České Budějovice: Kopp, 2001. 316 s. ISBN 80-7232-150-1

[2] ECKEL, Bruce. *Myslíme v jazyku Java: knihovna zkušeného programátora.* Přeložil Bogdan Kiszka. 1. vyd. Praha: Grada, 2000. 470 s. ISBN 80-247-0027-1

[3] *ECKEL, Bruce. Thinking in Java* [online]. 2007 [cit. 2007-03-17]. <http://www.mindview.net/Books/TIJ/>.

[4] KAROLÍK, Antonín. *Nástroj na kreslení Ganttových diagramu s využitím Metapostu.* Praha, 2007. 90 s. Diplomová práce na Elektrotechnické fakultě Českého vysokého učení technického na katedře řídící techniky. Vedoucí diplomové práce Ing. Michal Kutil.

[5] *Batik SVG Toolkit* [online]. c2000, last published 30th of March 2007 [cit. 2007-06-29]. <http://xmlgraphics.apache.org/batik/>.

[6] HEJRAL, Martin. *Průvodce SVG* [online]. 2006. [cit. 2007-03-10]. <http://interval.cz/clanky/pruvodce-svg/>.

[7] SEMECKÝ, Jiří. *Naučte se Javu* [online]. 2002. [cit. 2007-03-11]. <http://interval.cz/clanky/naucte-se-javu-uvod/>.

[8] DeWEESE, Thomas. *Introduction to the Batik project* [online]. 2002. [cit. 2007-03-16]. <http://www.svgopen.org/2002/papers/deweese_hardy__batik_intro/>.

[9] KORMANN, Thierry. *Developing SVG Application with Batik* [online]. 2002. [cit. 2007-03-16]. <http://www.svgopen.org/2002/papers/kormann__developing_svg_apps_with_batik/index.html>.

[10] KADLEC, Václav. *JAXP – Java API for XML Processing* [online]. 2002. [cit. 2007-05-06]. <http://nb.vse.cz/~zelenyj/it380/eseje/xkadv02/JAXP.htm>.

[11] GLOGAR, Martin. *Parsing XMl in Java 1.4* [online]. 2003 [cit. 2007-05-06]. <http://lynx1.felk.cvut.cz/pte/doc/java/xml/java-xml.htm>.

[12] ZUKOWSKI, John. *Java AWT Reference* [online]. 1997 [cit. 2007-04-20]. <http://www.oreilly.com/catalog/javawt/book/index.html>.

[13] KOLESNIKOV, Alexander. *Java Drawing with Apache Batik: A Tutorial.* 1th printing. Toronto: BrainySoftware, 2007. ISBN 0975212893

[14] *Scalable Vector Graphics (SVG)* [online]. 2004. [cit. 2007-08-02]. <http://www.w3.org/Graphics/SVG/About.html>.

[15] SOLDÁT, Marek. *Slabikář XML* [online]. 2002. [cit. 2007-08-01]. <http://interval.cz/clanky/slabikar-xml-uvod-do-problematiky/>.

[16] KOSEK, Jiří. *XML API* [online]. 2002. [cit. 2007-08-03]. <http://www.kosek.cz/xml/api/>.

[17] *W3C Document Object Model* [online]. 2005. [cit. 2007-08-04]. <http://www.w3.org/DOM/>.

[18] *Extensible Markup Language (XML)* [online]. 2003. [cit. 2007-08-03]. <http://www.w3.org/XML/>.

[19] BENEŠ, Miroslav. *Programovací jazyk Java* [online]. 2007. [cit. 2007-03-20]. <http://www.cs.vsb.cz/benes/vyuka/pte/texty/java/index.html>.

[20] KOTALA, Zdeňek. TOMAN, Petr. *Java* [online]. 2007. [cit. 2007-03-15]. <http://dione.zcu.cz/java/sbornik/toc.html>.

[21] *About W3C* [online]. 2007. [cit. 2007-08-04]. < http://www.w3.org/Consortium/>.

A CD-ROM

Přílohou bakalářské práce je CD-ROM, který obsahuje:

- ☐ program SVG Gantt
- ☐ všechny zdrojové soubory programu SVG Gantt
- ☐ bakalářskou práci ve formátu PDF
- ☐ zdrojové soubory této bakalářské práce
- ☐ ukázkové zdrojové soubory XML a jejich výstupy ve formátu SVG
- ☐ knihovnu Batik, potřebnou pro vývoj programu SVG Gantt