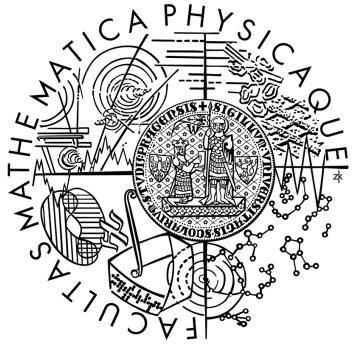
Univerzita Karlova v Praze
Matematicko-fyzikální fakulta

BAKALÁŘSKÁ PRÁCE



Robert Cesar

Rozvrhovací program pro základní a střední školy

Katedra softwarového inženýrství

Vedoucí bakalářské práce: RNDr. Michal Kopecký, Ph.D.

Studijní program: Informatika

Studijní obor: Programování

Praha 2013

Děkuji RNDr. Michalu Kopeckému, Ph.D. za vedení mé bakalářské práce a hodnotné připomínky k její podobě.
Také děkuji prof. RNDr. Romanu Bartákovi, Ph.D. za velmi inspirující přednášky.

Prohlašuji, že jsem tuto bakalářskou práci vypracoval(a) samostatně a výhradně s použitím citovaných pramenů, literatury a dalších odborných zdrojů.

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona v platném znění, zejména skutečnost, že Univerzita Karlova v Praze má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle § 60 odst. 1 autorského zákona.

V Praze dne 24 5. 2013

Název práce: Rozvrhovací program pro základní a střední školy

Autor: Robert Cesar

Katedra: Katedra softwarového inženýrství

Vedoucí bakalářské práce: RNDr. Michal Kopecký, Ph.D., KSI

Abstrakt: Rozvrhovátko je program určený základním a středním školám menší až střední velikosti. Jejich pracovníkům, odpovědným za vytváření rozvrhů, poskytuje jednoduchý nástroj pro každoroční vytváření rozvrhů pro celou školu. Primárním cílem programu je nabídnout generátor rozvrhů, který pracovníkům ušetří čas a bude mít parametry nastavitelné tak, aby tito pracovníci mohli jednoduše upravit délku výuky, požadované aprobace u předmětů, velikosti skupin žáků, volné hodiny pro konkrétní vyučující, nebo aby mohli změnit obsazení učebny vybranou třídou. Sekundárním cílem je usnadnit přechod k rozvrhování na počítači těm pracovníkům, kteří problém doposud museli řešit na papírech a lístečcích.

Klíčová slova: rozvrh, škola, plánování, administrace

Title: Scheduling program for basic and high schools

Author: Robert Cesar

Department: Departement of Software Engeneering

Supervisor: RNDr. Michal Kopecký, Ph.D., KSI

Abstract: Rozvrhovatko is a program dedicated for basic and high schools of small or medium size. The employees who are responsible for creating timetables for the school are provided with a simple tool which will save their time every year when they have to construct new timetables. Program can modify output of its generator by many parameters, such as number of lessons per day, required approbations for subjects, size of groups of students or free hours for chosen teachers. Secondary aim of program is to help those employees, who used to schedule everything on papers, to begin with automatic scheduling.

Keywords: timetable, school, scheduling, administration

Obsah

Úvod

Řada ředitelů základních a středních škol po celé republice, případně jimi pověření pracovníci, každoročně řeší problém, jak sestavit rozvrh vyučování pro všechny třídy školního zařízení a skloubit mnohdy protichůdné požadavky. Jejich zodpovědností je zajistit, aby učitelé měli kdy, kde, a koho učit. Musí proto seskládat toto velké puzzle hodinu po hodině, den po dni, třídu po třídě. Nemohou si přitom dovolit ubrat z dotace vyučovaných předmětů, držet žáky ve škole dlouho do večera, ani je nutit přijít příliš časně z rána. Přitom mají jen omezené množství místností, ve kterých se může vyučovat, a učitelů, kteří budou učit. Často je navíc nutné rozvrh operativně měnit i v průběhu školního roku, například pokud některá z učitelek dlouhodobě onemocní nebo odejde na mateřskou dovolenou.

Ještě dnes můžeme najít školy (drtivá většina škol do 200 žáků), ve kterých se rozvrhy plánují ručně na mnohokrát přepisovaných lístečcích. Některé jiné školy si nějaký rozvrhovací software pořídily, avšak nevyužívají jej naplno a část výuky stále plánují na papírech ručně. Jedná se například o využití tělocvičen.

Školy nemívají dostatek finančních prostředků pro najmutí specialisty či celého týmu pro sestavení rozvrhu, a proto je tato povinnost většinou přidělena zástupci ředitele či někomu z učitelů. Nemohu vztáhnout předpoklad na všechny školy, vím ale, že pokud někdo vytváří část rozvrhu na papírech, pak je to mnohdy způsobeno nedůvěrou osoby sestavující rozvrh k používanému programu. Proto je zapotřebí, aby program byl plně lokalizován a uživatelsky co nejpřívětivější. To umožní nechat veškerou manuální práci na počítači a odložit stohy stokrát přepisovaných papírů, které obsahovaly i dřívější verze rozvrhů. Administrativní pracovníci – ať již ředitel, jeho zástupce či jiný pověřený zaměstnanec – by měli programu dodat především informace o škole, předmětech a o lidech. Údaje o žácích a učitelích jsou obvykle potřeba v elektronické formě i bez podpory rozvrhování a jsou proto relativně snadno k dispozici, nebo bude naopak jejich elektronizace v rámci rozvrhování dalším přínosem. K těmto datům je potřeba doplnit omezení, jakými jsou dotace hodin a požadavky na výuku u předmětů. Za předpokladu, že nedojde k další reformě vzdělávání, se informace o předmětech během následujících let nemusí příliš upravovat. Program umožní správu dat a především vytváření přípustných rozvrhů a jejich upravování pomocí několika kliknutí myší. V případě následné ruční úpravy pak může snadno kontrolovat, zda změna neporušila některá z daných omezení. Dny strávené nad nekonečnými opravami tak mohou být při vhodné implementaci minulostí...

První kapitola se zabývá popisem modelové školy, pro kterou je vytvářený program primárně určený, a vlastnosti rozvrhovacího problému, na jehož vyřešení je program určen. Dále jsou zde uvedeny některé existující aplikace, jejich vlastnosti a vymezení vůči nim.

Další kapitola stanovuje požadavky na program, které vyplývají z analýzy provedené v první kapitole.

V následující kapitole jsem popsal některá rozhodnutí ohledně způsobu implementace, která bylo nutno při tvorbě programu učinit. Výsledky rozhodnutí

obvykle vycházely ze situace zjištěné při analýzou v první kapitole.

Navazující kapitola se zabývá podrobnějším vysvětlením důležitých částí programu a některých rozhodnutí zapsaných v předešlé kapitole. Kapitola končí popisem použitých rozhraní a možnostmi rozšíření pro případ dalšího vývoje aplikace.

Poslední kapitolou je uživatelská dokumentace, kde je s pomocí popisu a doprovodných obrázků vysvětleno používání aplikace pro jejího uživatele.

1. Analýza

1.1 Modelová škola

Primární cílovou skupinou jsou menší školy s velikostí přibližně do 200 žáků, ve kterých bývá vytvořením rozvrhu pověřena jediná osoba. Tento zaměstnanec pak postupně vytváří a přepisuje plány na papírech. Tyto školy nejspíše nevedou ani záznamy o svých studentech elektronicky, ale jen v papírové podobě, což činí jakoukoliv práci s daty značně obtížnější a i časově náročnější. Do cílové skupiny mohou patřit i školy, které sice patřičný software mají, ale ten buď nedokáže sestavit celý rozvrh, nebo mu pověřená osoba celé sestavení nesvěří. Těmto školám by bylo vhodné nabídnout plně lokalizovaný program, který obsahuje jednoduché administrativní nástroje pro data, jež škola potřebuje znát, který dokáže z poskytnutých dat vytvořit rozvrhy pro učitele, místnosti, třídy i pro skupiny ve třídách a který umí pomoci s realizací případných změn v rozvrhu v průběhu školního roku. Tyto rozvrhy, či jejich pozdější verze, by pak měl umět vytisknout či exportovat do formátu vhodného pro vyvěšení na webové stránky školy. Vhodným rozšířením požadavků by mohl být i export rozvrhů ve formátu pro mobilní telefony. Učitelé a žáci by pak mohli do svého telefonu nahrát informace o tom, kdy mají kde být. Po vytvoření a distribuci rozvrhů již program většinou nebude zapotřebí. Použije se pouze v případě změny ve vstupních požadavcích, kdy vygeneruje nový rozvrh nebo pomůže s ručními úpravami, a v případě potřeby dalšího vytištění rozvrhů. Pro tyto školy pak není zapotřebí zajišťovat přístup k aplikaci po síti. Postačí, když program poběží lokálně na počítači pověřené osoby a přístupné ostatním budou pouze výstupy programu.

Analýza vychází z mé zkušenosti s rozvrhováním na pražské základní škole, která má přibližně 600 žáků ve 25 třídách a 32 učitelů. Rozvrhy zde plánují dva zástupci ředitele pomocí programu *Bakaláři - systém pro školní administrativu*. Jenže jej nevyužívají pro generování rozvrhů, generátor totiž selhal, viz kapitola 1.4.1. V programu proto mají uloženou pouze evidenci žáků, zaměstnanců, tříd i vyučovaných předmětů. Sami nastavují úvazky učitelům, sestavují skupiny žáků a přiřazují vyučujícího pro každý předmět v každé třídě. Až poté vytvářejí rozvrhy „v režimu chytré nástěnky“, kdy seskládají rozvrh po jednotlivých hodinách v jednotlivých třídách.

Program jim nabízí zbývající hodiny u každé třídy, kontroluje, aby učitelé nebyli na více místech naráz, a barevně zvýrazňuje vhodné hodiny pro umístění lístečku s předmětem. Tento postup je velmi časově náročný, automatizace procesu by mohla ušetřit až týden práce. Svou bakalářskou prací bych chtěl čas potřebný na přípravu podobných rozvrhů zkrátit na minimum.

1.2 Vytváření vlastního rozvrhu

Program by samozřejmě měl umět pracovat v režimu „chytré náhrady nástěnky“ s popsanými lístky, kdy umožní ručně vytvořit rozvrh pro každého učitele, třídu i místnost tak, jak jsou to odpovědní pracovníci zvyklí dělat dosud. V tomto režimu, kdy by program pouze hlídal základní pravidla, například aby jeden učitel nebyl na dvou místech najednou, aby v místnosti byla pouze jedna třída, nebo

aby byl k dispozici požadovaný počet učitelů „v pohotovosti“[1]. Již splnění tohoto požadavku umožní nahradit program *Bakaláři*. Ve školách, které zatím žádný rozvrhovací software nemají, by program v tomto režimu mohl usnadnit přechod z plánování na papírech na elektronické rozvrhování.

Dále by však program měl umět vygenerovat kompletní rozvrhy podle požadavků stanovených uživatelem. Ten by měl mít možnost je postupně zadávat ručně nebo načíst ze zálohy či definice ve vhodném formátu. Požadavky by určovaly například skupiny žáků, maximální počet hodin daný den či hodinové dotace předmětů. Dále by měl být zohledněn preferovaný vyučující, například ten, kdo třídu učil předchozí rok, nebo třídní učitel, nejzazší čas pro výuku hlavních předmětů, jejich opakování v jeden den, upřednostněné rozložení hodin v rámci týdne a jiné. Tato omezení generátoru jsou pro program náhradou člověka, který by při plánování dával na všechny tyto požadavky pozor. Požadavek na kontinuitu učitelů navíc komplikuje přebírání rozvrhů z předchozích let. Je potřeba jej proto obvykle generovat vždy znovu od začátku.

Za třetí by program měl nabídnout přednastavené šablony, které by urychlily první plánování rozvrhů. Pověřená osoba si detaily různých omezení sama upraví tak, aby přesně vyhovovaly dané škole, ale nebude muset při jejich tvorbě stavět na zelené louce. Bude tak moci využít například maximální povolený počet hodin v daném dni, výčet předmětů, nebo jednoduché dělení stejných ročníků do skupin na tělocvik. V základní šabloně by mohly být nastaveny rovněž hodinové dotace předmětů podle již zmiňovaného Rámcového vzdělávacího programu[2], dále jen RVP.

Následně, nepůjde-li s danými omezeními rozvrh vyrobit kvůli triviálním nedostatkům, například kvůli nedostatku kvalifikovaných učitelů, by program měl nabídnout kriteria, která je potřeba změnit pro získání přípustného řešení.

Nakonec, po vytvoření jednotlivých rozvrhů, by program měl dokázat vytisknout celou sestavu, případně její vybrané části, s patřičnou úpravou na papíry formátu A4 i A3. Větší formát slouží k vyvěšení souhrnu rozvrhů na nástěnku ve sborovně či v ředitelně, menší formát se využívá pro tisk rozvrhů pro jednotlivé učitele, třídy a místnosti.

1.3 Uchovávaná administrativní data

Jak je výše uvedeno, program je určen především pro menší školy, které využijí programem udržovanou databázi pro základní školní administrativu. Program díky databázi nabízí alternativu i ke komplexním řešením, které používají ostatní školy. Aby se dala databáze takto použít, je potřeba, aby uchovávala informace o studentech, učitelích a dalších pracovnících, předmětech, třídách (skupinách) i o místnostech ve škole. Data by se měla do databáze vkládat ručně nebo nahráním z externího zdroje. Tím by mohl být zálohovací soubor z jiné instance tohoto programu, případně exportovaná data z aplikace pro správu základních škol *Bakaláři*, viz kapitola 1.4.1.

Dále je vhodné, aby databáze poskytovala samotnému programu informace, které využije při generování rozvrhu. Například je nutné znát kapacitu míst-

[1] Učitelé, kteří budou připraveni v danou hodinu zastoupit některého kolegu.

[2] viz *http : //www.msmt.cz/vzdelavani/skolskareforma*

ností, aby do malé učebny nebylo umístěno příliš mnoho žáků, nebo kvalifikaci učitelů, aby angličtinu nevyučoval pedagog, který vystudoval přírodní vědy, zatímco zkušený učitel anglického jazyka bude mít nedostatek hodin do úvazku. Úvazky učitelů jsou dalším významným faktorem, který program musí získat z databáze.

Nejdůležitější informací o studentech je jejich příslušnost k určité třídě. Při rozvrhování se počty žáků ve třídě mohou využít pro vytváření skupin, nebo třeba pro kontrolu nepřekročení kapacit místností. Dále by se mohlo využít profilování žáků v rámci výběrových předmětů, jejichž nabízení je pro školy povinné kvůli RVP. Nakonec je o studentech, ale i o učitelích, potřeba znát jejich osobní údaje a kontakty na ně, u žáků ještě kontakty na rodiče a příslušnost k určité třídě.

U učitelů je nezbytné znát hlavně jejich úvazky, ale také aprobace, z důvodů zmíněných dříve. Pokud to ředitel učitelům umožnil, bude nutné znát i požadavky učitelů na volné hodiny či dny v týdnu. Například učitel s částečným úvazkem bude chtít učit jen 3 konkrétní dny v týdnu. I s tímto požadavkem by měl generátor dokázat pracovat.

O místnostech, které mohou být identifikovány například číslem dveří, je potřeba znát především jejich vybavení a kapacitu. Pokud by systém znal požadované vybavení pro daný předmět a ročník, generátor by měl zajistit, aby třídy svými vlastnostmi splnily potřeby v nich vyučovaných předmětů a kapacitou vyhovovaly počtu žáků ve skupině.

Předměty mohou vyžadovat určité vybavení, jako například chemickou laboratoř, případně zkušenosti či kvalifikaci vyučujícího. Proto je nezbytné, aby se buďto manuálně předem nastavily všechny trojice učitel-předmět-třída, přičemž by kontrolu prováděl člověk, nebo aby měl program k dispozici data potřebná k vytvoření těchto trojic. Všechny předměty mají určitou hodinovou dotaci týdně či za dva týdny, která musí být rovněž zohledněna. Dotace se mezi školami může lišit podle jejich zaměření. Jako vhodný začátek pro vyplňování by však mohly sloužit požadavky MŠMT, plynoucí z RVP. Na nich založenou šablonu by stačilo jen upravit, případně rozšířit o další předměty.

Poslední částí jsou třídy, zde chápany obecněji ve smyslu „skupina žáků“. Je potřeba znát jejich označení, tj. „1.A“, určit žáky, kteří do nich patří, třídního učitele atd. Dala by se využít i znalost kmenové učebny dané třídy, mimo jiné proto, aby program mohl třídu umisťovat do dané učebny co nejčastěji. V základních třídách a napříč nimi se při některých předmětech budou tvořit obecnější skupiny. Proto by bylo užitečné, aby program podporoval určité množinové operace, například požadavek na disjunktnost určitých skupin či doplněk do celé třídy, které mohou složit jako pomůcka při vytváření skupin.

Předpokládá se, že se data nebudou měnit častěji než jednou ročně, tj. při zápisu nových žáků do školy. Výjimkami mohou být úpravy v případě personálních změn v průběhu školního roku, ale tyto situace nejsou příliš časté a změny nebývají příliš radikální. Lze proto předpokládat, že o údržbu databáze se bude starat pracovník pověřený sestavováním rozvrhu a program by tedy měl tuto údržbu co nejvíce usnadnit. Jelikož takto tvořený archiv obsahuje soukromá data, neměl by k němu mít přístup nikdo jiný, než pověřený pracovník kvůli údržbě dat a vedení školy. Tato omezená skupina by pak měla mít možnost pořizovat výpisy z databáze.

Alternativní variantou by bylo umožnění přístupu více lidem a vytvoření hi-

erarchie s různou mírou oprávnění. Ta by mohla na nejnižším stupni umožnit například zjištění kontaktu na žákovy rodiče, větší oprávnění by bylo potřeba pro vkládání či změnu údajů, nejvyšší pro tvorbu rozvrhů. Tato varianta by nalezla uplatnění především v komplexních programech, zahrnujících nejen rozvrhování, ale i další školní administrativu. Problematickou by se však mohla stát otázka ochrany osobních údajů, zda by každý uživatel programu měl mít přístup k záznamům se soukromými daty. Přístup by mohl být užitečný pro případ nečekaných naléhavých událostí, například při zranění žáka ve škole. Opačně se dá argumentovat, že v takové situaci musí být zkontaktováno vedení školy, které vyhodnotí situaci, zda je potřeba zavolat rodičům žáka, pro rychlou záchrannou službu, nebo zda stačí ošetřit odřeniny pomocí lékárničky. Vedoucí pracovníci školy by měli v každém případě přístup ke všem záznamům a také oni mají povinnosti vůči rodičům, tedy zřejmě by stačilo zpřístupnit data jenom jim a hierarchie oprávnění není zapotřebí.

1.4 Analýza existujících řešení

Problém hledání optimálního rozvrhu je skoro tak starý jako plánování času samo. Proto je již nyní k dispozici široká nabídka programů. Pokusíme-li se vyhledat existující řešení pomocí Google, dostaneme následující seznam programů[3]. Vzhledem k tomu, že hledání optimálního rozvrhu patří do kategorie NP-úplných úloh, implementované postupy hledání se liší v první řadě v účinnosti použitých heuristik. Samotné hledání optima vyžaduje rozličná data, popsaná v předchozí kapitole. Proto bývají programy pro jeho hledání často součástí větších balíků pro celou školní administraci. Tyto balíky pak mohou obsahovat služby od evidence žáků a učitelů, přes správu třídních knih, klasifikace, suplování a rozvrhování maturit až po nesouvisející programy jako podpora pro knihovnu, inventarizace majetku školy nebo vytváření rozpočtu školy. Volbu programu tedy lze usměrnit i vyjmenováním služeb, které budou od aplikace požadovány.

Vzhledem k tomu, že nelze obecně zajistit, aby osoba obsluhující program perfektně ovládala cizí jazyky, je možné uvažovat o použití pouze plně lokalizovaných programů. Tento požadavek omezí dlouhý seznamu existujících implementací jen na několik málo programů. Pokud by škola nevyužila evidenci žáků a zaměstnanců, pak může některý z freeware programů dostupných na internetu použít. V jejich případě ale chybí kontroly vyžadující evidenci, jako například jestli je v dané učebně dostatek míst pro danou skupinu, což by mohlo vést k nepoužitelnosti rozvrhu. V případě požadavku na evidenci škola potřebuje již nějaké komplexnější řešení, ta však jsou placená a možnost testování jejich vlastností je výrazně omezena.

Pro porovnání implementací obdobně komplexních, jako je plánovaná aplikace, je potřeba, aby vybraný program splňoval požadavky popsané v předcházejících kapitolách, protože existuje celá řada volně stažitelných lokalizovaných programů, které tyto požadavky nesplňují. Z existujících řešení jsem si vybral dvě v ČR používané aplikace, které by měly vyhovovat zadání, a sice program *Bakaláři - systém pro školní administrativu*, dále jen „*Bakaláři*“, který je podle

[3]Seznam programů pro vytváření rozvrhů na *directory.google.com/Top/Computers/ /Software/Educational/Administration_and_School_Management/Scheduling_Utilities*

internetových stránek distributora[4] nejznámější a nejoblíbenější software pro administraci na školách u nás, a dále program *aSc Rozvrhy*. Oba dva programy jsou komerční, minimální licence pro vytváření rozvrhu, suplování a jejich distribuci stojí přibližně 6 tisíc korun. Program *Bakaláři* jsem zvolil nejen pro jeho rozšířenost, ale také proto, že jde o komplexnější řešení pro celkovou správu. Program *aSc Rozvrhy* je u nás méně známý, je však dle internetových stránek výrobce[5] rozšířený do 114 zemí světa a získal tři zlatá ocenění na výstavách pro školství v Čechách a na Slovensku.

Pro otestování generátorů rozvrhů těchto programů jsou využita data z již zmiňované základní školy, ve které jsem se inspiroval. Velikost této školy umožňuje testovat rozvrhovací programy na hranici předpokládané velikosti škol cílové skupiny. Požadované rozvrhy pro třídy splňují kriteria RVP v minimálním rozsahu. Například volitelnými předměty pro 6. a 7. ročník jsou jen sportovní hry. Navzdory tomu není splnění podmínek triviální, jak je popsáno níže. V příloze na straně 50 je uveden souhrn vybraných vlastností popisovaných programů a plánované aplikace.

1.4.1 Bakaláři, program pro školní administrativu

Bakaláři představují komplexní balík pro správu veškeré školní administrace, poskytují podporu i pro oblasti jako jsou třídní knihy nebo školní rozpočet. Bohužel tato rozsáhlost má i negativní stránky - potřebuje společné rozhraní, podobné uživatelské prostředí,... Výrobce si také za každou část nechává zaplatit. Takže i když modul pro rozvrhy stojí 1600 Kč, uživatel musí koupit i další součásti tvořící základní balíček za celkem 5600 Kč, aby mohl program použít[6]. Jednotlivé moduly jsou na sobě relativně nezávislé, z čehož ale plyne další nepříjemnost - program vyžaduje přihlašování uživatele pro každý přechod mezi moduly, například mezi evidencí žáků a upravováním rozvrhu nebo suplování. Program při instalaci umožňuje nastavit údaje o škole, rozsah práv učitelů, zástupců a ředitele i vytvořit další účty typu „školník“ se specifickými omezeními přístupu. Také je možné importovat data z jiné instalace programu, např. ze souboru formátu `.dbf`. Exportování dat je umožněno ve třech formátech, a to `.dbf`, `.txt` a ve formátu pro tisk, pro zveřejnění na webu je zapotřebí samostatný modul. Ve všech verzích je možné jmenovitě vybrat, které informace budou přeneseny a které nejsou potřeba. Případně lze přímo v programu vybrat rozvrhy pro tisk a vytisknout je. Nicméně nemá-li uživatel k dispozici data k importu, musí vše nastavit sám, program nenabízí žádné šablony ani příklady použití. Modul zabývající se evidencí osob (žáků i zaměstnanců) by rovněž potřeboval vylepšit - program nepozná, když uživatel omylem jednoho žáka přidá do databáze dvakrát. Nevadí mu to, ani když se vyskytnou „oba“ žáci ve stejné třídě. Dalším nedostatkem je úprava dat - pokud si učitel rozšíří vzdělání, například získá Ph.D. titul, a správce upraví patřičné údaje v databázi, pak program namísto změny záznamu stávajícího uživatele (i s jeho přístupovými právy) vytvoří uživatele nového (bez práv) a původního uživatele nechá beze změny.

Prostředí programu není příliš uživatelsky přívětivé. Program nabízí mnoho

[4] *http://www.pachner.cz/bakalari/bakalari.htm*
[5] *http://www.asctimetables.com*
[6] *http://www.bakalari.cz/cenyprog.htm*

podpůrných funkcí, ale pro úpravy zadání a zvláště pro ruční skládání rozvrhu je potřeba věnovat čas na zorientování se ve velkém množství nepopsaných ikonek. Generátor pak požaduje vytvoření skupin a bloků, například více úrovní výuky angličtiny, chlapci a dívky na tělocvik, kde je potřeba určit i které skupiny jsou disjunktní a které se mohou prolínat. Pokud by chyběl jediný žák ve skupinách, tak se generátor bez upozornění vypne a chybu je nutné dohledat. Pokud je chybně upravena skupina žáků po vytvoření rozvrhů, pak jsou její rozvrhy bez oznámení vyjmuty. Dále je problém udělat skupinu z žáků patřících do různých tříd, například pro tělocvik, to je nutné obejít pomocí zvláštní skupiny pro každou třídu a tyto skupiny později provázat. Brzy po spuštění se generátor zacyklil při přiřazování hodin pro jazyky a skončil. Výsledkem bylo jen částečné řešení, bez možnosti úpravy či následného restartu generování, bez nápovědy jak dál pokračovat. Rozvrh bylo nutné dodělat ručně v režimu nástěnky třídu po třídě. Generátor jsem přitom spustil s mírnějšími omezeními, než jaká byla požadována pro rozvrh výše popsané školy, například bez podmínky, že učitel bude mít hodiny jen ve dvou dnech v týdnu. Při ručním sestavení byly i tyto náročnější požadavky splněny.

Program *Bakaláři* je vhodný, hledá-li škola komplexní řešení pokrývající všechny oblasti administrativy, se kterou se škola musí vypořádat. Bonusem určitě jsou dostupné plug-iny, například přihlášky na střední školy, nebo školení pro správce systému a pro učitele. Generátor rozvrhu však může selhat a vytvoření rozvrhu ručně zde není jednoduché.

1.4.2 aSc Rozvrhy

Program *aSc Rozvrhy* patří mezi programy specializované na nalezení a upravování rozvrhu. K tomu využije jen jména učitelů a počet či jména učeben. Nepoužívá žádné údaje o žácích, dokonce ani velikosti tříd. Nedokáže proto upozornit na řadu základních nesrovnalostí, například mezi velikostí místnosti a množstvím žáků ve třídě.

Pro vytvoření rozvrhu není potřeba žádné přihlašování uživatele, všechny zakoupené služby jsou k dispozici ihned po spuštění programu. Nejlevnější varianta, obsahující generátor rozvrhu a aplikaci pro správu suplování, je k dispozici za 6000 Kč. Varianta, která zahrnuje i podporu analytického týmu a exportní formát pro rozvrh na mobil, stojí dvojnásobek. Oproti výše popsaným „Bakalářům“ se však jedná o čistě rozvrhovací software, bez navazujících administrativních nástrojů[7].

Po nainstalování je možné rovnou začít tvořit rozvrhy. Při tom je možné, ale ne povinné, nastavit základní informace, jako např. jméno školy, počet vyučovacích dnů v týdnu nebo počet hodin denně. Dalším krokem, teď už nezbytným, je vložit místnosti a třídy s jejich názvy, jména učitelů a názvy předmětů spolu s týdenními dotacemi pro každou třídu. U všech těchto položek se dá upravit, kdy mohou být zapojeny v rozvrhu, kdy by radši neměly být použity a kdy nesmí být použity. Poté už může být spuštěn vlastní generátor. Program kompenzuje absenci jakýchkoliv šablon pomocí mnoha ukázkových sestav rozvrhů a podporou velkého množství formátů pro import. Nabízí celkem 14 formátů dat, včetně dat exportovaných z výše popisovaného programu *Bakaláři*. Bohatá je i podpora formátů pro export - celkem 13 variant, mezi nimi i formáty pro mobily

[7] $http://www.asctimetables.com/order2\_cz.php?school\_country=CZ$

a pro HTML. Dostupnost formátů pro import a export závisí na zakoupené licenci. Součástí programu je i vytištění hotových rozvrhů, jednotlivých i souhrnných, například pro třídy.

Prostředí programu je uživatelsky velmi přívětivé, rozdělení funkčních tlačítek do záložek připomíná MS Office 2007. Před samotným generováním je možné spustit testování zadání, které pomůže odhalit triviální nedostatky, takže je potřeba mít všechny eventuální skupiny připravené předem. Při testování je zkušebně vytvořen rozvrh samostatně pro každou třídu, učebnu, každého učitele i každý předmět. Toto testování proběhne rychle, a pokud selže, je nutné upravit požadavky, protože žádné řešení neexistuje. Dalším bonusem při nastavování je široká nabídka podmínek pro omezení výskytu předmětů, například nepovolit cizí jazyky v hodinách následujících po sobě, vybrané předměty vložit nejpozději na 4. vyučovací hodinu. V základním nastavení je možné rozšířit počet hodin během dne vyučovaných, například pro odpolední vyučování. Mezi negativa programu patří nedostatečná lokalizace (vyskytují se v něm i celé odstavce nepřeloženého textu) a odděleně zpracované suplování, vyžadující neustálé přihlašování při přechodu mezi rozvrhy a suplováním. Závažnou vadou by mohla být i absence záznamů o aprobaci učitelů, pokud nebudou učitelé všech předmětů u všech tříd nastaveni ručně, což je náročné, čili nepravděpodobné. Za hlavní nedostatek považuji selhání generátoru po 20 minutách počítání při zkušebním vytváření rozvrhu. Program poté nabídl urychlené doplnění maxima ze zbývajících hodin bez konfliktů, po kterém zbylo 60 nezařazených rozvrhových lístků pro ruční doladění v režimu nástěnky. Generátor selhal pravděpodobně na tom, že přednostně nedával dohromady disjunktní skupiny. Možná to bylo způsobeno špatným nastavením, ale tuto chybu jsem nedokázal najít.

Pracovat s programem *aSc Rozvrhy* je pohodlné a intuitivní, pokud škola nepotřebuje podporu pro další administrativu, tak je pro ni dobrou volbou.

2. Specifikace

Tato část shrnuje požadavky na vytvářenou aplikaci na základě výše uvedené analýzy.

2.1 Kompatibilita

Na základních a středních školách u nás se dají očekávat nejčastěji počítače s operačním systémem *Windows 7*, program proto musí podporovat tento systém. Avšak ne vždy je k dispozici tato verze *Windows*, v některých případech má škola na počítačích starší verzi *Windows Vista*, nebo dokonce *Windows XP*. Pro tyto případy by bylo vhodné, aby program dokázal pracovat i na těchto starších verzích operačního systému *MS Windows*, ale není to nezbytné. Lze předpokládat, že minimálně vedoucí pracovníci školy budou používat *Windows 7*.

Naopak není jisté, zda ve školách jsou počítače s 32 či 64bitovou architekturou. První varianta bude ve většině škol, ale po nákupu nových strojů se může stát, že se všechny staré odstraní a počítač s 32bitovou architekturou nebude k mání. Proto program musí zvládat obě varianty.

2.2 Uživatelské rozhraní

Jak vyplývá ze zkušenosti s modelovou školou, s programem budou obvykle pracovat zaměstnanci s běžnou či nižší počítačovou gramotností. Program proto musí mít grafické uživatelské rozhraní pro veškerou funkcionalitu s pokud možno intuitivním ovládáním. Všechny ovládací prvky jako tlačítka a položky nabídek musí proto obsahovat srozumitelný slovní popis, jednoznačnou ikonku, nebo ideálně obojí. Přechod mezi funkcemi programu by měl probíhat plynule, bez nutnosti opakovaného přihlašování či spouštění jiných programů.

2.3 Vrstvy aplikace

Program musí být schopen běžet celý na jediném počítači, bez připojení k internetu či místní síti. Možnost rozdělení aplikace na klienta a server na oddělených strojích je výhodou, avšak málokterá škola má dostatečné technické vybavení, aby tuto variantu vůbec mohla realizovat. Neměla by to proto být jediná varianta, kterou aplikace podporuje.

2.4 Rozvrhování

Hlavním účelem specifikovaného programu je vytvářet rozvrhy. Rozvrhem se rozumí množina lekcí, naplánovaná na určitou hodinu v určitý den týdne. Každá lekce spojuje učitele, třídu či skupinu, místnost a předmět v konkrétní čas. Dále pro rozvrhy platí, že učitel ani třída nemohou být na dvou místech naráz, v místnosti nemůže být najednou vyučováno více lekcí. Program musí také pohlídat, aby nebyla překročena kapacita místnosti.

Program musí být schopen automaticky vygenerovat podle uložených dat rozvrhy pro celou školu. Generování rozvrhu musí být uživatelem nastavitelné tak, aby mohl třídám přiřadit předměty a jejich hodinové dotace podle osnov a aby mohl i v rámci třídy tvořit skupiny žáků mající odlišné lekce. V ideálním případě by program měl podporovat i tyto osnovy, usnadnit jejich vytváření a modifikace. Ale není to nezbytné, pokud bude generování nastavitelné jiným způsobem. Výsledek generování pak musí být filtrovatelný podle tříd, učitelů i místností.

Program by dále měl nabízet poloautomatický režim tzv. „chytré nástěnky“. Tedy umožnit uživateli pracovat s lekcemi ve formě „lístků“, nechat jej „připíchnout“ tyto „lístky“ na konkrétní „nástěnku“ reprezentující učitele, místnost či třídu. Tento režim je potřeba především pro uživatele, kteří jsou zvyklí, že si rozvrhy tvoří sami ručně na papírech. Pomocí „chytré nástěnky“ je jim usnadněn přechod na rozvrhování s pomocí počítače. Program v tomto režimu uživateli nabídne aktuální rozvrh pro vybranou entitu, nerozvržené lísky pro manipulaci a především kontrolu uživatelových činností, aby nebyla porušena omezení stanovená výše.

2.5 Správa záznamů

Součástí programu by měla být databáze uchovávající seznamy učitelů, žáků, tříd a místností, pro které se budou dělat rozvrhy. Tyto záznamy musí jít jednoduše přidávat, upravovat i mazat skrze grafické rozhraní. Součástí záznamů by měly být údaje potřebné ke generování rozvrhů, například aprobace učitelů či kapacita místností, aby byly splněny požadavky kladené na rozvrhy.

2.6 Distribuce dat

Program musí umožnit přesun databáze či její kopie na jiný počítač tak, aby byla použitelná alespoň v jiné instanci tohoto programu. Bonusem by byla podpora formátů pro export do jiných programů pro rozvrhování či import z nich, pro základní funkci rozvrhování to však není nezbytné. Další výhodou by byl export výsledných rozvrhů do formátů pro kalendáře mobilních telefonů, pro vyvěšení na internet či pro tisk na tiskárnách, což by usnadnilo distribuci rozvrhů.

2.7 Úpravy funkcí

Program by mělo být možné rozšířit minimálně o podporu nových formátů pro export či import dat. Ať už kvůli vývoji nových technologií a jimi podporovaných standardů, nebo proto, že základní verze programu vybraný formát nepodporuje. Tato úprava by neměla vyžadovat novou instalaci programu ani manipulaci s databází.

3. Design

Po specifikaci požadavků na program bych nyní popsal implementaci vlastního programu.

3.1 Systémová rozhodnutí

3.1.1 Operační systém

Ze specifikace vyplývá, že aplikace nesmí být závislá na konkrétní verzi Windows. Program nevyžaduje konkrétní operační systém, jeho jednotlivé části však používají nástroje z platformy *.NET Framework* nejvýše ve verzi 4.0. Ta je k dispozici od jara roku 2010, a dá se proto předpokládat, že v rámci automatických aktualizací operačního systému bude nainstalována na všech počítačích. V opačném případě se dá velmi snadno doplnit[1]. Tato platforma podporuje *Windows XP* a všechny novější verze *Windows*. Domnívám se, že tím je zajištěna kompatibilita pro všechny počítače, které by mohly sloužit vedení školy.

Na druhou stranu je však nutné uznat, že na školách nebývají k dispozici nejnovější stroje a že obměna inventáře je dlouhodobý proces. Proto je značně pravděpodobné, že program skutečně bude muset podporovat *Windows* od verze XP a 32bitovou architekturu. Vzhledem ke zpětné kompatibilitě 64bitové architektury k té 32bitové a nijak extrémním nárokům na množství zpracovávaných dat není zapotřebí psát program pro obě varianty. Stačí 32bitová verze.

3.1.2 Způsob uložení dat

Jak vyplývá z analýzy modelové školy, program bude muset pracovat s navzájem silně provázanými daty v řádu stovek až tisíců záznamů. Z toho důvodu by použití vlastních souborů se zvláštní strukturou bylo nevhodné. Vhodnější bude použít databázi, která zajistí mj. i konzistenci a provázanost dat. Pro modelovou školu není potřeba zpracovávat žádné závratné množství dat, proto postačí jakýkoliv typ menší databáze, které jsou obvykle zdarma k dispozici.

Vybral jsem kompaktní edici Microsoft SQL databáze. Její hlavní výhodou je velmi snadné šíření či přesun databáze a jejích ovladačů společně s aplikací k uživateli, respektive mezi jednotlivými počítači uživatele. K programu tak stačí připojit knihovnu s nástroji a veškerá komunikace pak probíhá skrze *ADO.NET*. Pro distribuci je k dispozici malý soubor s knihovnou, který se nabídne operačnímu systému při instalaci vlastního programu. Nevýhodou této edice je velmi omezené množství nástrojů a redukce části obvyklé databázové funkcionality. Například neumí složené příkazy SELECT nebo uložené procedury. Toto omezení však nevadí, neboť propojení s databází může plně nahradit *ADO.NET*, které nabízí programátorovi velmi pohodlné pomůcky vygenerované na míru konkrétní databázi.

[1] Nejnovější verze platformy, .NET Framework 4.5, byla vydána 15. 8. 2012 a je k dispozici na stránkách *http* : *//www.microsoft.com/net*

3.1.3 Programovací jazyk a vývojové prostředí

Vzhledem k náročnosti řešeného problému předpokládám, že bych měl program napsat v některém z „managed languages“. Kvůli kvalitě programovacích nástrojů, značné propojenosti s jednotlivými částmi budoucí aplikace a osobní preferenci jsem zvolil jazyk C# a vývojové prostředí *Visual Studio*.

3.1.4 Uživatelské rozhraní

Specifikace vyžaduje grafické uživatelské rozhraní, po volbě jazyka a vývojového prostředí se výběr možných GUI, která půjdou rozumně implementovat, prakticky zužuje jen na *Windows Forms*, dále jen *WinForms*, a *Windows Presentation Foundation*, dále jen *WPF*. Obě možnosti poskytují dostatečně variabilní nástroje k zobrazení dat a realizaci jejich změn skrze GUI i pohodlnou práci se samotnou aplikací. *WPF* je daleko flexibilnější, umožňuje zobrazit prakticky cokoliv, avšak je také značně složitější. Vše, co je v aplikaci potřeba, podporují i jednodušší a pro programování příjemnější *WinForms*, proto jsem si pro GUI vybral je.

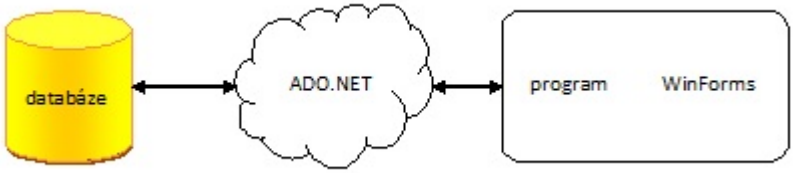
3.2 Vrstvy aplikace

Český termín „vrstvy“ se používá jak pro softwarový, tak pro hardwarový pohled na členění aplikace, což by mohlo být při popisu řešení matoucí. Proto si dovolím téma rozdělit do více sekcí a použít obšírnějšího popisu, bude-li to potřeba.[2]

3.2.1 Softwarové rozdělení

Při rozdělení na vrstvy v softwarovém významu je potřeba rozhodnout, jak moc nezávislé části programu chci, respektive kolik jich chci použít. Teoreticky je nejspíše možné mít celý program i s daty v jednom bloku, avšak takové řešení by podle mne bylo nepřehledné, značně krkolomné a nejspíše nerozšířitelné.

Dvě vrstvy, tedy rozlišení na databázi a „vše ostatní“, považuji za proveditelné, ale nikoli ideální řešení. Je potřeba počítat s tím, že postupem času může být program rozšířen, upraven či opraven. Nebo může narůst množství zpracovávaných dat a zvolená verze databáze pak již nemusí stačit. V každém z těchto případů by byly zapotřebí rozsáhlé úpravy kódu, pokud by výpočetní vrstva komunikovala přímo s databází.



Podle mého názoru jsou optimálním řešením tři softwarové vrstvy. Rozdělení programu by pak mělo být následující: databáze – databázové rozhraní – „tlustý klient“, kde klient obsahuje výpočetní část programu a GUI, viz obrázek. Přidání

[2] Rozlišení například viz *www.cleverandsmart.cz/vicevrstva−architektura−popis−vrstev*

databázového rozhraní umožňuje případnou změnu typu databáze bez úprav výpočetní části programu. Klient se pak postará o výpočty i o jejich zobrazení uživateli.

Uplatnění pro více vrstev by se zřejmě dalo najít, zejména v případě rozložení programu na více strojů, ale nepovažuji za žádoucí komplikovat program jen proto, abych získal menší vrstvy.

3.2.2 Hardwarové rozdělení

Použití více hardwarových vrstev se obvykle hodí v situacích, kdy se předpokládají vysoké požadavky na výkon stroje a daný problém se přitom dá rozčlenit podle poučky „rozděl a panuj“. Například jeden server by uchovával a zpřístupňoval data, další by nad nimi prováděl operace a třetí by řešil komunikaci s uživatelem.

U malých a středních škol se taková zátěž dá prakticky vyloučit. Aplikaci i databázi proto lze ponechat na jediném počítači, který veškerou práci vykoná sám. Tuto variantu přímo požaduje sama specifikace. Přesunutí databáze na samostatný server by mělo smysl až v případě, že by program umožňoval vzdálený přístup s přihlašováním, případně i s hierarchií různých oprávnění. Takové rozšíření však již značně přesahuje rozsah zamýšleného programu.

3.2.3 Moduly

Podle specifikace by program, nebo jeho vybrané části, měl být složen z nezávislých modulů tak, aby bylo možné upravit jeho funkce přímo u uživatele, bez nutnosti nové instalace. Požadavek jsem implementoval tak, že uživatel si za běhu programu bude moct sám vybrat, který modul chce použít například pro tisk rozvrhů. Defaultně jsou však přednastaveny základní varianty všech modulů, aby uživatel nemusel nic vybírat, pokud by o to sám neměl zájem.

Upravitelnost programu není implementována jen v minimální variantě, jak ji určuje specifikace. Kromě možnosti změnit moduly pro import, export a tisk jsou k dispozici rozhraní pro odlišné řešení požadavků na rozvrhy i pro samotný generátor rozvrhů, více viz kapitola 3.4.

3.3 Funkcionalita

3.3.1 Správa údajů

Veškeré záznamy jsou ukládány do databáze, komunikace s ní probíhá skrze *ADO.NET*. Jedná se o tzv. odpojenou aplikaci[3]. Taková aplikace se k databázi připojí, aby si načetla potřebná data do paměti, všechny úpravy, vkládání a mazání provádí pouze v paměti a až při ukládání se znovu spojí s databází a změny skutečně provede. Tento postup je velmi užitečný z mnoha důvodů, nejčastěji je uváděna velká úspora přenášených dat, pokud je databáze na jiném počítači. Tato výhoda není v mém případě využita, neboť program i data jsou

[3]Popis odpojené aplikace například zde: $http://www.zive.cz/clanky/poznavame-c-a-microsoft-net-60-dil--ado-net--implementace-odpojenych-aplikaci/sc-3-a-128948/default.aspx$

na jednom místě, jak už jsem popsal dříve. Nejedná se však o jedinou přednost. Odpojená aplikace zrychlí práci s daty, jelikož jsou v paměti a ne jen na disku, umožňuje jejich prohlížení v obou směrech, filtrování a jiné.

Uživatel může záznamy velmi jednoduše přidávat, upravovat i mazat v přehledných tabulkách, pro vkládání a změny mu je k dispozici dialogové okno se zvýrazněnými povinnými údaji. Zjednodušený ER diagram je v příloze na straně 51, kompletní seznam tabulek s vazbami mezi nimi na strane 52. Podrobnější informace jsou uvedené v dokumentaci na straně 23.

3.3.2 Osnovy

Se zavedením *Rámcových vzdělávacích programů*, dále jen *RVP*, musely všechny školy vypracovat nové osnovy. Hodinové dotace všech předmětů pro všechny třídy pak vycházejí z nich. V programu je přidaný jednoduchý nástroj pro práci s osnovami, jak žádá specifikace. Umožňuje jejich snadné vytvoření podle několika uložených šablon, které vycházejí právě z *RVP*, i úpravy již existujících variant. Pro rychlou orientaci nástroj zobrazuje součty hodin u jednotlivých předmětů, bloků předmětů, každého ročníku zvlášť i pro celý stupeň, kterému jsou osnovy určeny. Program ukazuje i zbývající disponibilní hodiny, které je možné v osnovách přidat. Stanovení hodinových dotací pro bloky předmětů, pro některé konkrétní předměty a pro ročníky v „rámci“ vychází z platné verze *RVP*. Po případné reformě je nutné čísla aktualizovat.

Osnov si uživatel může vytvořit libovolné množství, pojmenovat si v nich předměty, popsat k čemu slouží. U každých osnovy je potřeba říct, pro které ročníky jsou určeny.

3.3.3 Požadavky

Zvláštní, ale zcela zásadní částí pro generování rozvrhů jsou požadavky. Pomocí tlačítek lze nastavit požadavky pro učitele, třídy, místnosti i předměty. Na tomto nastavení pak závisí, jak budou rozvrhy vygenerovány. Pokud by například pro třídu „1.A“ bylo určeno, že může mít rozvrženy hodiny libovolně od první do osmé hodiny, pak se může stát, že třída bude mít značně „děravý“ rozvrh, tedy s mnoha volnými hodinami, což je nežádoucí. V programu jsou přednastaveny základní sady požadavků, další si může vyrobit a pojmenovat uživatel sám podle potřeby. Při vytváření tříd se automaticky nastaví i defaultní požadavky pro daný ročník. Toto nastavení pak lze samozřejmě změnit v požadavcích tříd.

3.3.4 Generování rozvrhů

Nejdůležitější část programu, vlastní vytváření rozvrhů, je úzce spjata s generováním plánů. Podle plánů a požadavků zúčastněných entit, tedy tříd, učitelů, místností a předmětů, jsou vygenerovány jednotlivé rozvrhy. Tento proces je podrobněji popsán v dokumentaci na straně 26. Vytvořené rozvrhy se pak dají zobrazit v různobarevném provedení pro jednotlivé učitele, třídy, místnosti i skupiny žáků, případně jsou k dispozici velké přehledy pro všechny učitele, třídy a místnosti.

3.3.5 Další

Program předpokládá i možnost importu, exportu a tisku všech uložených dat, rozvrhů, jednotlivých tabulek či aktuálně vyfiltrovaných záznamů. Tyto funkce však nebyly primárním požadavkem specifikace, proto jsou implementovány jen v nejjednodušší variantě, tedy export do XLS a import z něj, a případné rozšíření je možné pomocí specifikací vyžadovaných modulů.

3.4 Případná rozšíření

Základním cílem specifikace je napsat program, který dokáže vygenerovat kvalitní rozvrhy. Samozřejmě nemohu zaručit, že jde o optimální řešení, to by patrně bylo pro uživatele značně časově náročné. A také zbytečné, neboť pomocí zmíněné funkce požadavků lze přípustné řešení značně omezit a varianta, kterou generátor nabídne, už je té optimální velmi blízko. Při případné nespokojenosti s výsledkem se vždy dají zpřísnit podmínky a nechat vygenerovat nové rozvrhy.

Nicméně nechtěl jsem napsat program, který se nebude mít kam rozvíjet. Některé funkce by byly užitečné už teď, jiné by se mohly teoreticky hodit budoucímu uživateli. Proto je v programu mnoho „zadních vrátek“ pro úpravy či rozšíření aplikace, především pomocí modulů. Níže nabízím některá rozšíření, která mne napadla, s popisem jak je provést. Konkrétní implementace je poté popsána v následující kapitole.

3.4.1 Import a export

V současnosti je aplikace schopna přesouvat data do a z formátu XLS. Tato funkcionalita je užitečná pro zálohování, kopírování i přesun záznamů mezi jednotlivými instancemi programu, je požadována specifikací a pro uživatele samotné aplikace je postačující. Pro uživatele jiných programů řešících rozvrhování to však nestačí, proto prvním vylepšením aplikace by asi mělo být přidání podporovaných formátů, například pro převod z programů *Bakaláři* či *aSc Rozvrh*, které by zjednodušilo přechod k užívání tohoto programu. Pro export by bylo vhodné přidat podporu pro zobrazení rozvrhů na webu a pro převod rozvrhu na „události“ ve formátu, který by si učitelé či žáci mohli uložit v mobilu.

Pro import i export je k dispozici jednoduché rozhraní, přes které program dokáže rozšíření přijmout. Modul musí akorát obsahovat implementaci tohoto rozhraní, uživatel si jej pak může vybrat v nastavení programu.

3.4.2 Tisk

Tisknutí rozvrhů ani záznamů uložených v databázi není pro generování sestavy rozvrhů pro školu zapotřebí, proto není vůbec implementováno. Pro používání těchto rozvrhů je však vhodné, aby mohly být distribuovány mezi učiteli, případně mezi žáky vytištěné. Toto rozšíření je opět snadno realizovatelné pomocí modulu, stejně jako to bylo popsáno výše.

3.4.3 Suplování

V rámci usnadnění každodenní práce zaměstnanců vedení školy by program mohl podporovat i nabídku učitelů pro suplování. V podstatě by se jednalo pouze o jiný typ filtrace informací v databázi, než jaký provádí okno zobrazující rozvrhy. Toto rozšíření by vyžadovalo menší úpravu aplikace, a to přidání okna pro suplování. Okno by mělo v jedné části mřížku představující dny a hodiny v týdnu, v druhé části by se zobrazoval seznam učitelů, kteří mají ve vybranou dobu volno.

3.4.4 Změna či přesun databáze

Neočekávám, že by někdo chtěl vyměnit databázi se svými záznamy za jinou. K přenosu dat jednoduše slouží export a import. Pokud by takový zájem byl, je potřeba změnit „Connection string“ obsahující název souboru s databází a heslo k jejímu použití. Tento textový řetězec je kvůli bezpečnosti součástí aplikace, pro jeho změnu je potřeba nový „build“ aplikace. Stejná úprava je potřeba v případě přesunu databáze na jiné místo, ať už v rámci počítače či na oddělený server.

Daleko pravděpodobnější úprava je použití jiného typu databáze, například nahradit stávající *MS SQL Compact edition* za silnější *MS SQL Express* databázi. Ta má výkonnější nástroje a především zvládá větší množství dat. Otázkou zůstává, zda existují tak velké školy, které by větší databázi využily. V současnosti používaná databáze dokáže uchovat až 4 gigabyty dat, přičemž při běžném provozu v modelové škole program uloží maximálně jednotky megabytů dat. Každopádně, pro změnu typu databáze je potřeba zajistit, aby na počítači byly k dispozici její ovladače, tedy pořídit je a nainstalovat. Využívaná kompaktní edice má již zmiňovanou výhodu, že její ovladače se dají snadno šířit s programem. U silnějších typů databází je šíření daleko náročnější, především soubory s ovladači jsou mnohem větší.

3.4.5 Vlastní nastavení tlačítek

Vylepšením v jiném směru by bylo umožnit uživateli, aby si sám vybral, která tlačítka mu budou v horním panelu okna k dispozici. Program nabízí veškerou funkcionalitu přes menu, společné nástroje jsou dostupné ve všech oknech aplikace. Nyní ale mají jednotlivá okna navíc tlačítka, která zpřístupňují nejpoužívanější nástroje. Nebo alespoň tak, jak předpokládám, že budou využívány.

Změna těchto tlačítek „rychlé volby“ ve stávajícím řešení není možná, avšak úprava by nebyla náročná. Změnit by byla potřeba pouze funkce pro nastavení a zpřístupnit jí onen horní panel nesoucí tlačítka. Ikonky i funkcionalitu mohou tlačítka převzít přímo z menu, jeho ikonky by se na tlačítkách automaticky zvětšily.

3.4.6 Ukládání filtrů

Posledním urychlovacím vylepšením, které by se teoreticky mohlo hodit, je možnost uložit si používané filtry. Všechny základní tabulky, které zobrazují záznamy z databáze, jsem umožnil filtrovat. Filtry pro sloupečky obsahující prostý text hledají shodný podřetězec, filtry pro číselné sloupečky lze nastavit na konkrétní hodnotu, čísla větší či menší. Dokážu si například představit, že by uživatel

potřeboval opakovaně pracovat s konkrétní skupinou učitelů, kterou vybral pomocí filtrů. V tomto případě by se mu mohlo hodit si filtrovací příkazy uložit a později je načíst a rovnou používat, nemuset je znovu vymýšlet a vypisovat.

Tato úprava by šla zařídit rozšířením funkcí v menu, kde by se přidala položka pro uložení aktuálních filtrů. Zároveň by bylo nutné přidat funkce do editoru databáze a editoru plánů. Jedna funkce by pro menu tyto filtry zpřístupnila, druhá by dříve uložené hodnoty nastavila do filtrů.

3.4.7 Přihlašování uživatelů, hierarchie oprávnění

Posílení zabezpečení uložených dat by se dalo dosáhnout přidáním uživatelských účtů chráněných heslem. S jejich zavedením by bylo možné poskytnout přístup i pro další uživatele, například pro všechny učitele, kteří by měli menší oprávnění, jak jsem již psal výše, v kapitole 1.3. Tato změna by vyžadovala úpravu programu i databáze, kde by se tabulka učitelů rozšířila o hesla. Nejjednodušší a nejefektivnější by asi bylo řešení vyžadující přihlašovací údaje uživatele při startu programu.

S přidáním účtů se nabízí další rozšíření, a to vzdálený přístup. Program by pak bylo nutné značně upravit, aby dokázal komunikovat po síti. Databáze by se mohla přesunout na server, nebo spíše na výkonnější školní počítač, který by byl neustále připojený k místní síti.

3.4.8 Přidání administrativy

Budoucím vývojem programu by mohlo být jeho rozšíření o podporu dalších úkonů, například školní administrativy. V první fázi by mohl nabízet volné učitele pro suplování či hlídání na chodbách o přestávkách. Další fází by mohla být evidence známek, pozdních příchodů žáků, publikací v knihovně či dalšího inventáře školy. Program by se mohl rozšířit o účetnictví, tisk vysvědčení a jiných dokumentů. Současně s těmito rozšířeními by se mohlo uplatnit i přihlašování uživatelů popsané výše. Tento vývoj by však vyžadoval značný zásah nejen do programu, ale i do struktury samotné databáze.

4. Programátorská dokumentace

V návaznosti na rozhodnutí vypsaná v předchozí kapitole bych nyní podrobněji objasnil fungování jednotlivých částí programu.

4.1 Použité technologie

Celý program jsem napsal ve vývojovém prostředí programu MS Visual Studio 2010 Ulimate s využitím platformy MS .NET Framework verze 4.0. Z této platformy program využívá především technologií Windows Forms pro uživatelské rozhraní, ADO.NET pro komunikaci s databází a konečně databázi MS SQL Server Compact Edition verze 3.5, dále jen *SQL CE*, pro ukládání dat.

4.2 Databáze

Vytvoření databáze

Databázi jsem vytvořil v odděleném projektu, ve kterém byla naplněna i zkušebními záznamy. Výrobu samotného souboru má na starosti nástroj *SqlCeEngine*, který databázi po zadání „connection stringu“ sám vygeneruje. Následně jsem pomocí běžných SQL příkazů skrze *SqlCeCommand* vytvořil jednotlivé tabulky, jejich sloupečky, klíče a omezení. Pro její konkrétní složení bych odkázal na diagram v příloze na straně 52, případně na jeho elektronickou variantu s popisem vlastností a vztahů na přiloženém CD. Pro všechny tabulky platí, že jejich prvním sloupečkem je sloupeček s „ID“, který slouží jako primární klíč tabulky.

Dalším krokem bylo sestrojení adaptérů pro komunikaci tabulkou, konkrétně pro každou tabulku jeden adaptér, neboť *SQL CE* dokáže pojmout pouze jeden SELECT příkaz. Tyto adaptéry byly použity pouze pro vkládání iniciačních záznamů, například základních požadavků.

Vlastní program tento pomocný projekt nevyužívá, s výjimkou vyrobené databáze.

Komunikace s databází

Pro zapojení databáze do programu je zapotřebí vytvořit komunikační rozhraní. Zde jsem využil nástroj vývojového prostředí, které využilo *ADO.NET*, abych nechal vygenerovat sestavu silně typovaných tříd podle existující databáze. Tyto třídy odpovídají jednotlivým tabulkám a zastřešuje je „DataSet“, který charakterizuje strukturu celé databáze.

Přesun záznamů mezi databází na disku a pamětí probíhá v režimu tzv. odpojené aplikace, zmíněné v kapitole 3.3.1. Pomocí adaptérů vygenerovaných výše zmíněným nástrojem se data načtou do paměti v podobě „DataSetu“, s kterým pak program může pracovat. Poté je, pouze při spouštění programu, ve všech tabulkách nastavena automatická inkrementace sloupečku primárního klíče a omezení tabulek. Uložení dat zpět na disk proběhne opět skrze adaptéry, avšak aktualizují se výhradně ty záznamy, které byly přidány, změněny, či odebrány.

Změna databáze

Úprava „connection stringu“, zmíněná v předchozí kapitole kvůli výměně databáze, lze provést ve vývojovém prostředí v nastavení projektu, pak nechat sestavit nový „build“.

Pro změnu typu databáze je potřeba kromě získání ovladačů i tuto databázi vyrobit. K tomu se dá použít výše zmíněný pomocný projekt po modifikaci kreačních příkazů. Jedná se zejména o vytvoření souboru databáze a objektu pro SQL příkazy. Pokud by měl být projekt využit i pro vložení dat, je potřeba přidat patřičný typ adaptéru pro danou databázi do funkce GetAdapter třídy AdapterFactory. Ostatní nastavení lze ponechat.

Pro vlastní program je potřeba podle nové databáze nechat vygenerovat adaptéry, které jsou vázané na typ použité databáze. Toho lze docílit několika kroky v průvodci vývojového prostředí. Samotné třídy zastupující tabulky, respektive „DataSet“, jsou na typu databáze nezávislé a zůstanou stejné. Díky tomuto rozhraní není potřeba jakkoliv upravovat výpočetní část programu.

4.3 Rozdělení programu

Kromě výše uvedených vrstev databáze a komunikačního rozhraní se program jednoduše členit nedá. Každá funkcionalita ke svému provozu potřebuje databázi a vyjma modulů je integrovaná do aplikace. Nicméně se od sebe liší předmětem činnosti, z čehož vychází roztřídění mého popisu dále.

Základní členění

„Základní okna“ programu jsou čtyři, a sice editor administrativní dat, editor požadavků, editor plánů a editor rozvrhů. Tato okna v případě potřeby otevřou další, „vedlejší okna“, pro realizaci konkrétní činnosti, aby byla pro uživatele co nejpříjemnější. Tato vedlejší okna se chovají jako Dialogy, tedy do jejich zavření je vyvolávající základní okno zablokováno. To slouží především k tomu, aby uživatel nemohl spustit několik vedlejších oken a případně poškodit konzistenci dat.

Mezi základními okny se přepíná tak, že aktivní okno je viditelné, ostatní jsou skrytá. Při přechodu program zjistí velikost stávajícího okna a jeho pozici, pak toto okno skryje a se stejnými hodnotami zobrazí okno následující.

Z popsaného schématu se vymyká průvodce pro vložení osnov a tříd pro nově tvořenou databázi. Jelikož se jedná o složitější funkci, je tato implementována jako sled několika oken. Osnovy se vyrábějí ve zvláštním okně, při tvoření tříd se uživateli nabídne sekvence oken – pro výběr počtu ročníků, počet tříd v ročníku a pak osnovy pro každý ročník. Nakonec, po ukončení průvodce, jsou postupně data vkládána pomocí běžného dialogového okna, které uživatel potvrzuje.

Ovládání programu

Uživatel může veškeré úkony, vyjma poloautomatického „režimu nástěnky“, provádět pomocí tlačítek. Základní okna obsahují menu, ve kterém se vyskytují funkce společné pro jinak oddělené oblasti, například nabídku uložení dat, a funkce specifické pro konkrétní oblast, například rozložení plánu. Některé funkce

jsou přístupné i pomocí tlačítek v horním panelu okna, například přepnutí zobrazení na učitele, případně pomocí klávesové zkratky, například přepnutí na jiné základní okno. V každém případě se obsluha funkce provádí na jednom místě, ať byla funkce vyvolána klávesou, tlačítkem či z menu.

Menu Každé základní okno obsahuje lištu s nabídkou menu. Společné jsou záložky „Soubor“ (uložení, načtení, import, export, tisk dat, ukončení programu), „Výběr editoru“ pro změnu základního okna, „Generování a kontrola“ (různé kontroly připravenosti na generování, pak generování rozvrhových plánů, generování rozvrhů) a „Nastavení“. Obsluha těchto společných funkcí je pro všechny editory vykonávána ve třídě „MenuToolsClass“, která pak provádí další úkony.

Jednotlivé editory mají i své vlastní záložky, které se vztahují přímo k jejich specielním funkcím. Proto má editor administrativních dat záložku pro přepínání Panelů a záložku pro práci se záznamy. Záložka v editoru požadavků nechává vybrat, pro koho se budou požadavky zobrazovat, editor plánů má nástroje pro práci s plány a editor rozvrhů umožňuje přepnout způsob zobrazení rozvrhů.

Tlačítka v horním panelu U speciálních funkcí lze předpokládat, že budou u daného editoru obzvláště využívané, proto jsou dostupné nejen z menu, ale i pomocí tlačítek v horním panelu.

Klávesové zkratky Výběr použitelných zkratek se liší podle aktuálního základního okna, což vyplývá z jejich funkce. Odchytávání stisku kláves zajišťuje v každém editoru jedna funkce a ta také vyvolá patřičnou obsluhu. Pro úplný seznam podporovaných zkratek viz kapitola 5.2.3.

Zobrazení administrativních dat

Zobrazení dat uživateli probíhá skrze tabulky DataGridView, které čerpají data z DataSetu pomocí BindingSourců. Manipulaci se záznamy pak lze provádět buď přímo na DataSetu, nebo prostřednictvím BindingSource, který je na DataSet napojen. Na DataSetu se změny realizují ihned, avšak na BindingSourcu je potřeba úpravy potvrzovat, jinak se neprovedou a tento zdroj se může dostat do nekonzistentního stavu.

Administrativní data uživatel může upravovat v okně „EditDataForm“, které obsahuje 9 hlavních tabulek, například učitelé či třídy, a několik pomocných s přídavnými informacemi, například jména žáků patřících do vybrané třídy. Všechny tabulky jsou vloženy do 9 Panelů, v každém jedna hlavní tabulka. Mezi Panely, a tedy i mezi hlavními tabulkami, se přepíná skrýváním, respektive zobrazováním daného Panelu. Uživatel tak může činit skrze menu nebo tlačítka v horním panelu, obslužná funkce je stejná.

Uživateli je dále k dispozici filtrování zobrazených záznamů tím, že vyplní patřičná textová pole. Filtrace se pak provede nastavením BindingSourcu, což se v zápětí projeví v tabulce.

BindingSourcy, ze kterých čerpají pomocné tabulky, jsou napojené na konkrétní Foreign Key nějaké hlavní tabulky. Tento postup je velmi výhodný, neboť umožňuje automaticky zobrazovat záznamy vztahující se k vybranému řádku hlavní tabulky. Problém by mohl nastat v případě, kdy by pomocí filtrování

byly skryty záznamy v hlavní tabulce, ke které se klíč vztahuje. Tehdy by došlo k chybě, neboť pro pomocný BindingSource taková data neexistují. Dá se tomu předejít mazáním filtrů, nebo skrytím a zablokováním pomocné tabulky.

Správa administrativních dat

Uživatel může data měnit, vložit či mazat pomocí výše popsaných ovládacích prvků. Změnu lze také zahájit vstupem do políčka DataGridView. Všechny změny či vkládání vyvolají otevření dialogového okna, kde uživatel upraví údaje, okno je ověří a pak vrátí obsluze výsledný řádek. Ten je pak buď uložen, nebo vložen do DataSetu. Pokus o smazání vyvolá pouze okno, které se ujišťuje o úmyslu uživatele záznam odstranit.

Požadavky

Tento editor obsahuje mřížku dnů a hodin v týdnu. Každé políčko obsahuje Label, který zobrazuje konkrétní požadavek vybrané sady požadavků v podobě ikonky značící „ano“, „pokud je to nutné“ a „ne“. Tyto varianty jsou uloženy i v „Tagu“ Labelu, konkrétně hodnoty 0, 1 a 2. Hodnoty odpovídají údajům, které jsou uložené v databázi.

Dále editor nabízí dva ComboBoxy v horní liště. Jeden slouží k výběru jednotky, pro kterou jsou požadavky určeny, například pro místnost. Druhý je k výběru sady požadavků, jejíž ID je u jednotky uložené ve sloupečku pro požadavky. ComboBoxy získávají data z BindingSourců, proto je potřeba kontrolovat činnost uživatele, aby se tyto zdroje udržely v bezchybném stavu.

Zobrazení požadavků Pro zobrazení požadavků se postupuje tak, že při každé změně vybrané hodnoty v ComboBoxu pro sady požadavků se v Labelech, kde to je potřeba, přenastaví zobrazená ikonka a uloží patřičné číslo do Tagu.

Změny požadavků Labely v mřížce kromě zobrazení ikonek slouží i ke změně hodnot. Pokud uživatel na nějaký Label klikne, přenastaví se jeho ikonka, ale zároveň se i připraví nová sada. Tuto sadu pak lze uložit, pojmenovat nebo někomu přiřadit.

Rozvrhové plány

Plány se zobrazují stejně jako administrativní data v DataGridView tabulce. Uživatel může sám vytvářet, měnit, mazat či filtrovat existující záznamy obdobným způsobem, jako bylo popsáno dříve. Zde má navíc možnost vybraný plán „rozdělit“, což značí vytvořit další plán na základě stávajícího s možností upravit oba plány. Plány si také může nechat vygenerovat, o tom podrobněji níže.

Zobrazení rozvrhů

Rozvrhy jsou v databázi uložené v tabulce po jednotlivých lekcích, takže úkolem okna pro rozvrhy je tyto záznamy roztřídit a uživatelsky přívětivě zobrazit. Tento editor nabízí dva režimy zobrazení – rozvrh pro jednotlivce či rozvrh

souhrnný. Oba jsou řešeny vkládáním barevných Labelů do mřížky, přepínání mezi režimy se dělá pomocí skrývání a zviditelnění Panelů, obdobně jako bylo popsáno výše.

Režim pro jednotlivé rozvrhy navíc obsahuje Panel, ze kterého lze změnit zobrazený rozvrh. Panel se skládá z Buttonů pro všechny entity daného druhu, jako svůj text nese její název a jako Tag uchovává její ID. To se pak využije pro změnu zobrazeného rozvrhu.

4.4 Použité struktury

Nejdůležitější použitou strukturou je databáze a tabulky v ní, ale ty už jsou popsány výše. Zde tedy popíšu jiné významné stavební prvky programu.

Okna

Pro uživatele zásadní částí jsou okna, kterými s ním program komunikuje. Jedná se buďto o třídu Form, nebo MessageBox. Třída Form je využívána pro základní okna a v dialogové variantě pro vedlejší okna, jak je popsáno výše. MessageBoxy slouží k potvrzení úkonu či jednoduché informaci uživateli.

Další vizuální struktury

Visual Studio nabízí mnoho užitečných vizuálních komponent, avšak tato nabídka nepokryla všechny potřeby aplikace. Všechny nové třídy jsou v souboru VisualTypes.cs. Několik vizuálních prvků bylo potřeba pro průvodce tvorby osnov, aby si objekty předávaly užitečná data. Dále jsem toho využil, aby se při splnění podmínek pozadí kontrolky obarvilo na zeleno či červeno, podle splnění či nesplnění zadaných hodnot.

Další nové třídy byly nutné pro zobrazení rozvrhů, kdy lístečky nemohly být obyčejným Labelem, ale musely být schopné reagovat na vložení do mřížky změnou údajů „svojí“ lekce. Také je možné mít více lekcí v jednom políčku mřížky, na což je nutné reagovat vizuálně. Zvláštní třídu si vyžádal „odkládací Panel“, do kterého se vloží Labely pro nerozvržené lekce, i samotné mřížky, které musí umět pracovat s pohybem lekcí i přepínáním filtrované entity.

Výčtové typy

Začátek souboru DataTypes.cs obsahuje výčty charakterizující různé situace, které se v aplikaci vyskytnou. Jedná se například o typ zobrazeného rozvrhu.

Datové nosiče

Velmi početným druhem struktur jsou „nosiče“ informací, které přidávají minimum další obsluhy. V těchto případech šlo především o zjednodušení přenosu dat, například při úpravách záznamů ve „vedlejších oknech“. Do této skupiny patří většina tříd ze souboru DataTypes.cs.

Nosiče služeb

Posledním typem struktur, které se v programu vyskytují, jsou objekty, které zapouzdřují nějakou komplexní funkci a bývají využívané z různých oblastí programu. Nemají svoje vlastní data, pouze zpracovávají předané parametry. Patří sem třídy ze souboru ToolsClasses.cs.

Získávání zkratek Tento nástroj slouží k získání unikátní zkratky, respektive iniciálů, v dané tabulce. Pracuje tak, že předané jméno či název rozdělí podle určitých znaků na více slov, z těch pak použije počáteční písmena. Pokud by se takto vytvořená zkratka již vyskytovala, zkouší se připojit následující znaky názvu. Pokud ani to nepomůže, zkouší se ke zkratce přidat náhodné cifry.

Nástroj pro setřídění vybíraných hodin pro lekce Do této třídy jsou nejprve naskládány hodnoty pro „dobré“ či „přijatelné“ časy (den a hodina v týdnu), tyto hodnoty jsou pak náhodně zamíchány a uloženy jako posloupnost pro výběr času pro lekce. Zároveň je zajištěno, aby všechny „dobré“ časy přišly na řadu před těmi „přijatelnými“. Při následujících dotazech pak nástroj nastaví aktuální čas podle posloupnosti a inkrementuje index do ní.

Třídění vkládáním Nástroj slouží jako InsertSort, kdy se mu předají dvojice klíč a hodnota. On podle klíče dvojici zatřídí do svého seznamu a na konci pak vrátí pouze pole setříděných hodnot.

Získání úplného podgrafu disjunktních skupin Nástroj z předané tabulky skupin a tabulky disjunktních vztahů mezi skupinami postupně vysbírá všechny kliky pomyslného grafu, kde „vrcholy“ tvoří skupiny a „hrany“ vazby mezi skupinami. Postup je jednoduchý: nástroj vezme ze seznamu jednu skupinu, prochází postupně její „sousední vrcholy“ a tvoří z nich kliky. Pak jejich hrany odstraní ze seznamu a zkusí další sousední vrcholy. Nakonec nástroj vrátí seznam seznamů skupin, což je vlastně seznam nalezených klik.

Kontrola konzistence dat Tento nástroj, obvykle na žádost třídy MenuToolsClass, zkontroluje stav DataSetu a některé tabulky pro případné problémy, které by znemožnily rozvrhování. V případě, že nástroj nalezne problém, přesune Focus na patřičnou hlavní tabulku a nastaví filtr tak, aby uživatel mohl hned zjednat nápravu. Pro popis konkrétních zkoumaných oblastí bych odkázal na generovanou dokumentaci v příloze na CD.

4.5 Generátor

Program nabízí generování ve dvou případech – pro vytvoření rozvrhových plánů či samotných rozvrhů. Obě části jsou zásadní, protože bez připravených plánů nemohou být vzniknout rozvrhy. Obě části mají pro správné fungování určité požadavky na stav databáze, proto jsou součástí nástroje i volání kontrol.

Generování plánů

Nástroj pro sestavení rozvrhových plánů je přímočarý. U každé třídy zjistí z jejích osnov jaké předměty a s jakou hodinovou dotací má mít. Tím je určeno množství plánů a předměty v nich. Do plánů se nevkládají třídy, ale skupiny (aby mohla část třídy mít jinou výuku, nebo s jiným vyučujícím). Místo třídy se defaultně nastaví její „celotřídní skupina“. Volba místnosti a vyučujícího pak závisí na předmětu. Při porovnávání vytíženosti učitele či místnosti se hledí na doposud rozvržené plány.

Pokud je třída z prvního stupně, pak je dosazen třídní učitel. Jinak je potřeba vyřešit kumulativní podmínky na aprobaci a preferovaný stupeň učitele. Obecně je upřednostněn třídní učitel. Pokud však nesplňuje požadovanou aprobaci, zkusí nástroj najít vhodnějšího pedagoga. Při více kandidátech zvolí toho nejméně vytíženého. Funkce nezohledňuje úvazky, aby nebyla zasažena možnost přesčasů.

Předmět může mít i požadavky na vybavení místnosti. Při volbě se preferuje kmenová učebna, pokud to lze, jinak nástroj zvolí tu nejvhodnější místnost s nejmenším využitím.

Předpokládá se, že plány budou po vygenerování upraveny uživatelem, aby to odpovídalo požadované situaci ve škole. Zejména rozdělení plánů s využitím disjunktních skupin, například pro tělocvik, by mělo být obvyklé.

Generování rozvrhů

Obecně vzato generátor vychází z připravených rozvrhových plánů a požadavků jednotlivých entit. Podrobnější pohled odhalí přednostní rozvržení plánů s větším nebezpečím problémů, tedy výběr metodou „first-fail“. Do této kategorie patří především velmi vytížené místnosti, jako jsou například tělocvičny či odborné učebny. Další náročnou sadou jsou plány s vícetýdenním cyklem a nakonec plány pro disjunktní skupiny.

Přípravu na generování tvoří nejprve odstranění plánů s nulovou hodinovou dotací. Dalším krokem je vytvoření kombinovaných požadavků pro plány, tedy spojení požadavků zúčastněného učitele, třídy, místnosti a předmětu. Poté je z tohoto požadavku vytvořen nástroj pro setříděný výběr hodin zmiňovaný výše. Nakonec je pro každou skupinu vytvořen filtrovací řetězec pro seznam s ní konfliktních skupin.

První fázi generování tvoří správné uspořádání plánů, a to metodou popsanou výše. První v posloupnosti jsou plány nejvytíženější učebny, následované plány na více týdnů, pak plány pro disjunktní skupiny a poté zbytek plánů. Mezi rovnocennými plány v dané kategorii se vybírají nejprve plány pro nejnižší ročníky, protože ty mají nejnižší variabilitu. Podle takto uspořádané posloupnosti plánů je možné začít generovat rozvrhy.

Druhá fáze spočívá ve výběru aktuálního plánu, ověření, že nebyl celý rozvržen, určení jeho kategorie a pokus o rozvržení všech jeho lekcí. Kategorie je důležitá pro plány na více týdnů a plány s disjunktními skupinami, protože tyto plány je vhodné spojit s jinými plány a jejich lekce rozvrhnout v jeden čas. Při neúspěchu jsou všechny rozvržené lekce aktuálního plánu odstraněny, jeho nástroj pro výběr času restartován. Dále je jako aktuální nastaven předchozí plán, je odstraněna jeho poslední lekce a fáze se opakuje. Nástroj pro výběr času tedy určí další

možný čas pro rozvržení,... Pokud se index aktuálního plánu dostane pod nulu, znamená to, že rozvrhování selhalo.

Třetí fáze spočívá v kontrole volného času na oběd, případně se zde dá přidat nějaká evaluační funkce pro všechny rozvrhy. Konflikt rozvrhu dostatkem času na oběd se zjistí tak, že nějaký žák má během 5. 6. a 7. hodiny rozvrženy lekce. V takovém případě jsou konfliktní lekce odstraněny a generování se vrací do druhé fáze. Pokud generování projde i touto fází, jsou rozvrhy v pořádku.

4.6 Moduly

Všechna rozhraní pro moduly jsou k dispozici v souboru Interfaces.cs. Společným požadavkem na jejich implementaci je schopnost identifikace metodou GetModuleName. Tento údaj je totiž využit jako popis aktuálně použitého modulu viditelný v záložce Nastavení aplikace, přes kterou se moduly dají za běhu připojit k aplikaci.

Export

Předně je zapotřebí zjistit, do jakých formátů implementace zvládá data exportovat. Pro to slouží funkce zjišťující podporované koncovky a popis těchto formátů. Údaje jsou pak použity v dialogovém okně pro export, který je obdobný jako dialog „Uložit jako“, kdy si podle nich uživatel vybere formát pro export.

Následující funkce slouží k inicializaci nástroje, kdy se mu nastaví plný název souboru a index do seznamu podporovaných formátů. Zbylé dvě funkce slouží k samotnému exportu dat, kdy program nástroji předá buď celý DataSet, nebo jeho část.

Import

Nástroj pro import má jednodušší rozhraní, neboť je pouze potřeba zjistit, jestli podporuje formát vybraného souboru. Pokud ano, je nástroji předán plný název souboru a prázdný DataSet, do kterého má načíst data. Spojení s aktuální databází provede program sám. V podstatě převezme nový DataSet a tam, kde to bude možné, jej doplní původními záznamy.

Tisk

Nástroj pro tisk musí sám implementovat uživatelské rozhraní, pokud má umožnit nějaké nastavení, protože program to nedělá. Nástroji je předán celý DataSet, aby mohly být tisknuty nejen výsledné rozvrhy, ale i ostatní záznamy.

Požadavky

Tento modul má nahradit jedno ze základních oken, takže je mu pro iniciaci předána třída MenuToolsClass, aby mohl poskytovat stejnou funkčnost jako ostatní okna. Funkce UpdateAndShow() se volá pokaždé, kdy má být toto okno zobrazeno. Program nevolá přímo zděděnou funkci Show(), aby byly možné případné modifikace. Poslední funkce, GetFilteredData(), se volá v případě, že uživatel

chce exportovat aktuálně zobrazené informace. Tedy hodnoty aktuálně vybrané sady požadavků.

Na modul však nejsou kladené žádné podmínky ohledně úpravy požadavků. Může tak použít stávající informace z databáze, ale může také pracovat naprosto nezávisle na aplikaci.

Generátor

V případě, že by modul pro požadavky řešil svou oblast zásadně odlišně od současné verze, by bylo nutné přepsat i generátor. Proto je k dispozici jednoduchý interface, který si na generátoru nevynucuje nic víc, než schopnost zahájit generování, dogenerovat částečně rozvržené rozvrhy a přerušit proces generování.

4.7 Další rozšíření

Nakonec bych popsal kroky, které je potřeba učinit k implementaci navržených rozšíření. Náležitosti pro moduly řešící import, export a tisk jsem popsal v předcházející podkapitole, postup pro změnu databáze jsem vysvětlil na začátku této kapitoly. Níže proto uvádím jen zbývající možnosti rozšíření.

Suplování

Jak jsem psal již v designu, jedná se o jednoduché rozšíření. Do projektu by se přidala další třída dědící od Windows.Forms.Form, do MenuToolsClass by přibyla nová položka, která by se přidala do přepínání základních oken a jako ostatní základní okna by získalo přístup k databázi. Způsob vizuální úpravy by byl již jen otázkou vkusu.

Vlastní nastavení tlačítek

Toto rozšíření vyžaduje úpravu záložky Nastavení a možnost ukládat informace i volbě tlačítek. Samotné přidávání a odebírání tlačítek z horního Panelu by šlo udělat ve dvou funkcích. Náročnější by bylo vybírání tlačítek či určení jejich pořadí, na to bych asi potřebovat pomocné okno.

Ukládání filtrů

Rozšíření by mohlo být realizováno rozšířením společné funkcionality základních oken, které umožňují filtrování, a přidáním možnosti uložení hodnot získaných z filtrů. Obsluhu by pak mohla zařídit MenuToolsClass.

Přidání administrativy, přihlašování

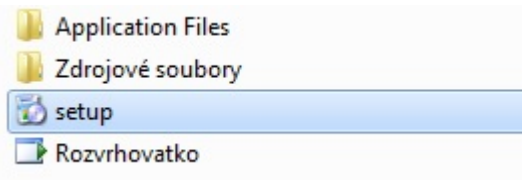
Tyto úpravy by vyžadovaly především značnou změnu databáze a přidání obslužných oken. Pro přihlašování by jedno okno zajistilo ověření uživatele například při startu aplikace, další by umožnilo práci s daty o uživatelích. Obdobné úpravy by pravděpodobně potřeboval každý přidaný úkon administrativy.

5. Uživatelská dokumentace

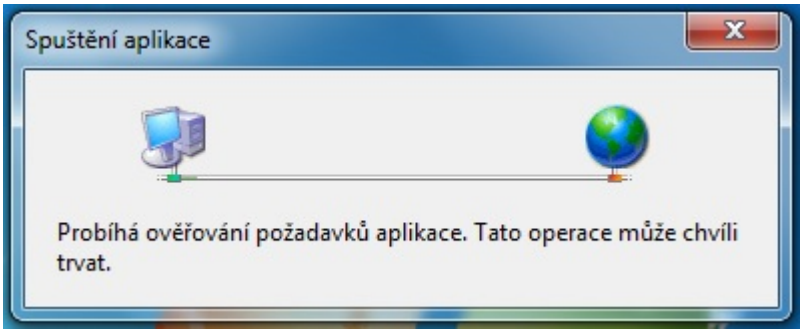
Tato část popisuje a s pomocí obrázků vysvětluje práci s aplikací.

5.1 První kroky

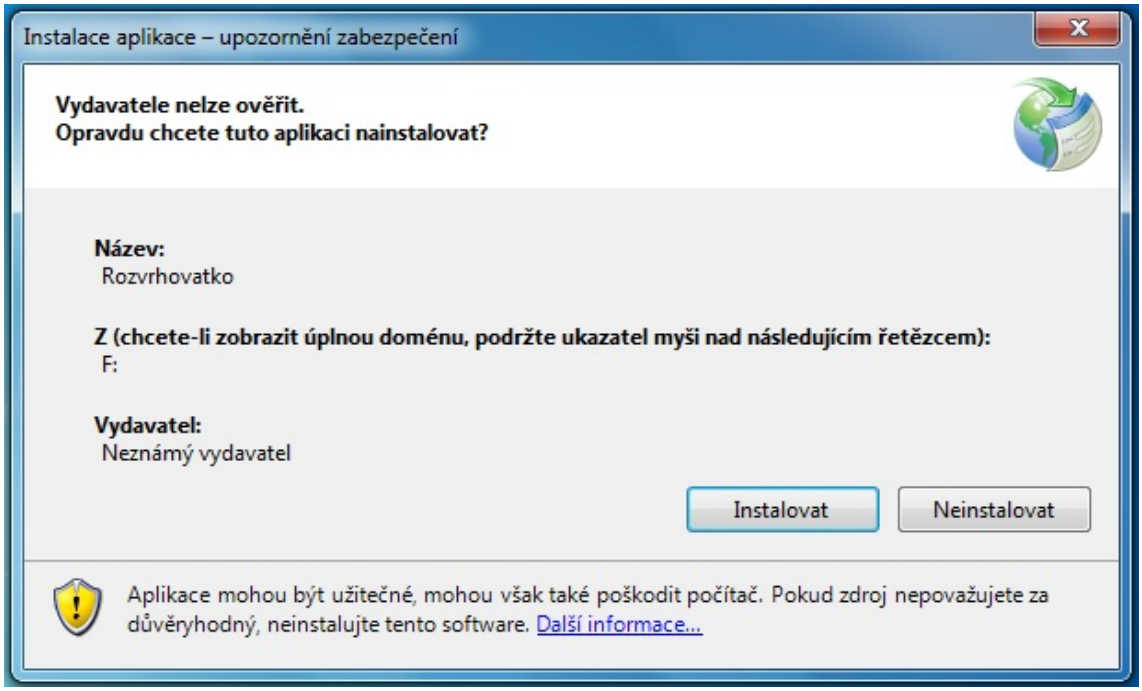
5.1.1 Instalace programu



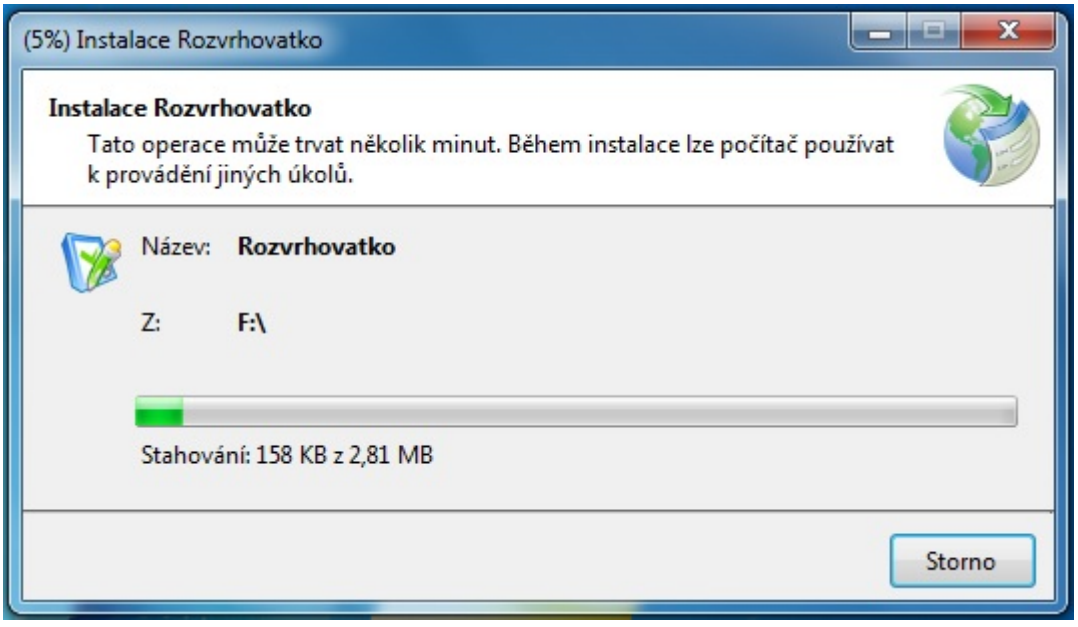
Instalaci aplikace lze zahájit poklepáním na soubor Setup.exe.



Tím se spustí instalátor, který ověří dostupnost nezbytných nástrojů, konkrétně *Microsoft .NET Framework 4.0* a *SQL Server Compact 3.5 SP2*, a který je v případě potřeby doinstaluje.

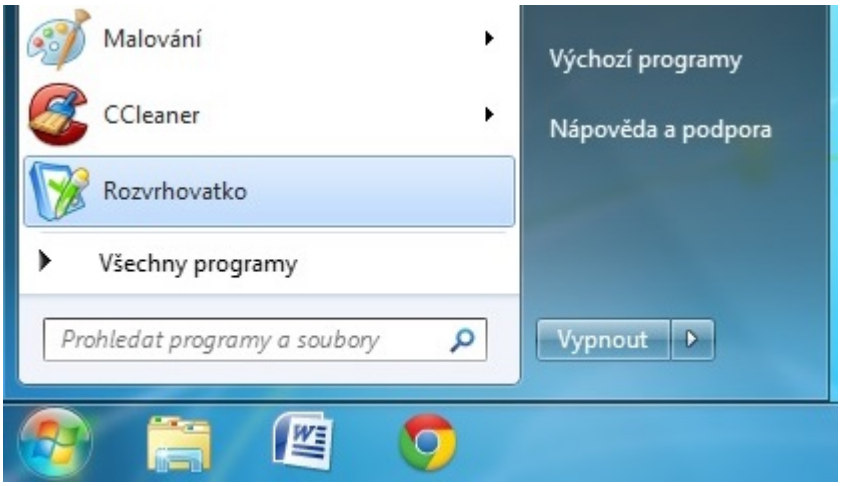


Instalátor používá technologii *ClickOnce* od Microsoftu. Od uživatele vyžaduje pouze ověření, že skutečně chce nainstalovat vybraný program. Všechna ostatní zbývající nastavení upraví sám a pak spustí instalaci vlastního programu.

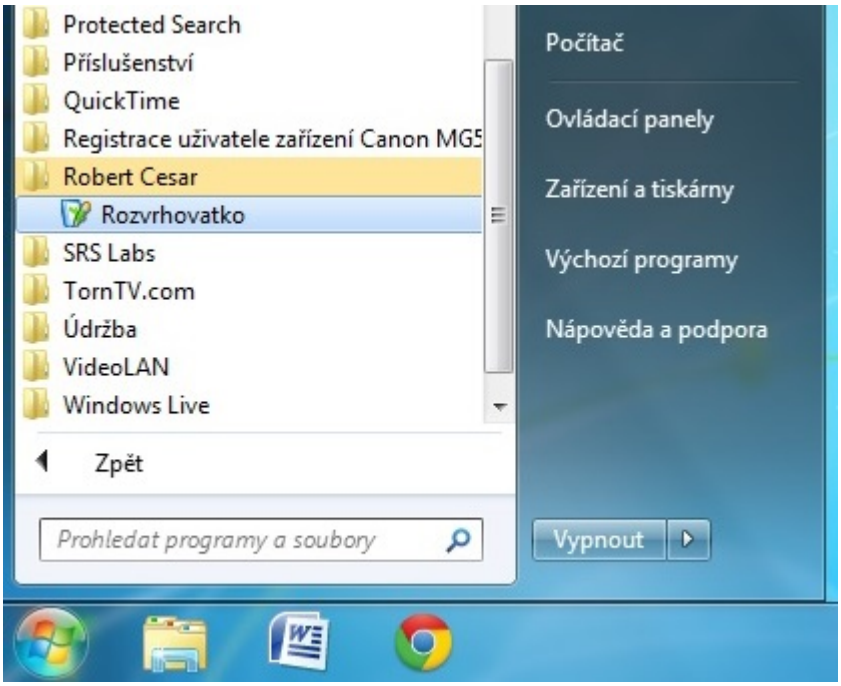


Po ukončení instalace je ihned spuštěn program. Instalátor také vloží položku do *Start Menu* pro pozdější spuštění programu. Pokud by uživatel vyvolal instalátor znovu, je místo vytvoření nové kopie spuštěn v počítači již nainstalovaný program.

5.1.2 Spuštění programu

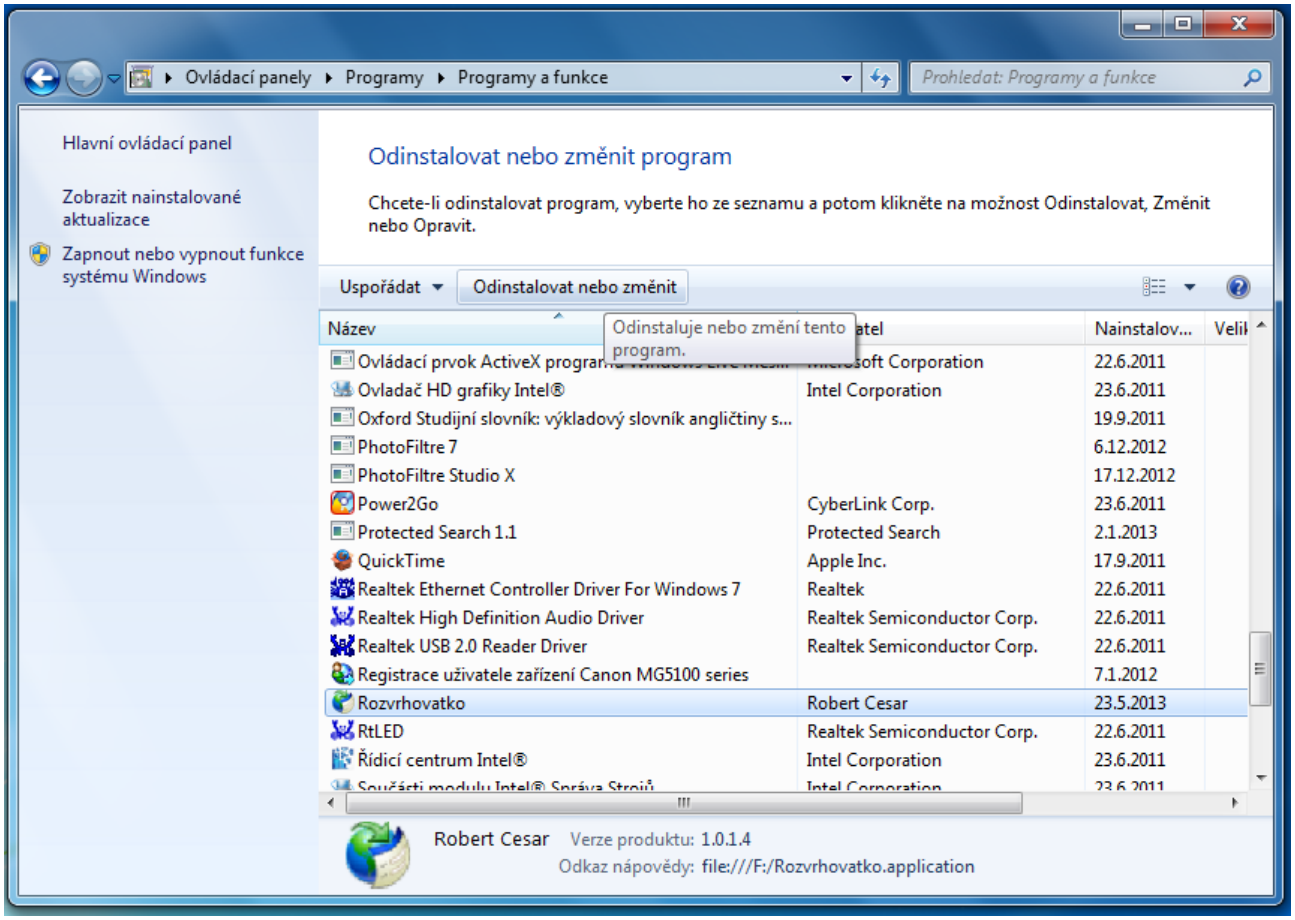


Poprvé je program automaticky spuštěn instalátorem. Další spouštění musí uživatel provést sám přes *Start menu*, kde je pro program vytvořená nová složka.

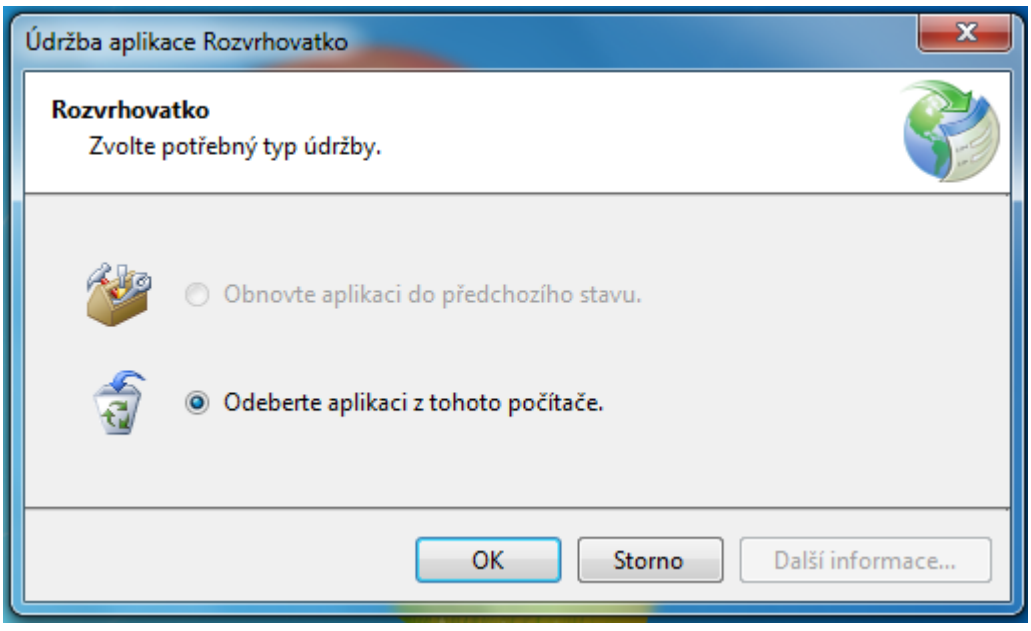


5.1.3 Odinstalování programu

Program lze z počítače odstranit pomocí nástroje *Přidat/odebrat programy*, který je součástí *Windows*.



Tento nástroj spustí odinstalační utilitu, která zařídí odstranění aplikace i jejích dat z počítače.

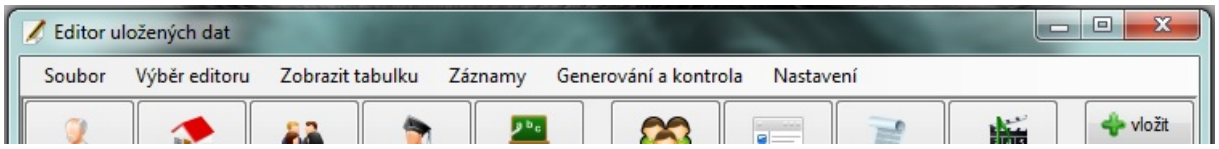


5.2 Ovládání programu

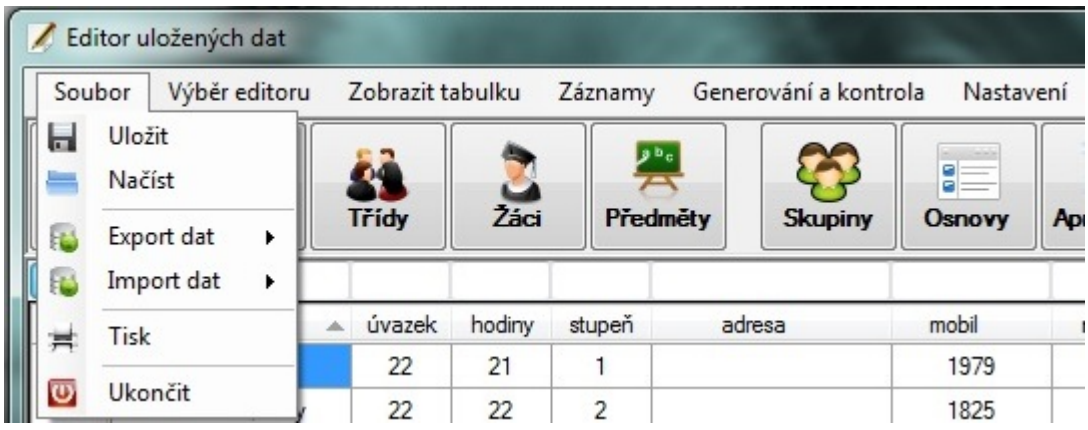
Pro maximální pohodlí uživatele jsou všechny funkce aplikace dostupné v grafickém rozhraní. Program lze ovládat pomocí velkých tlačítek, která nabízí obsluhu nejobvyklejších činností, položek v menu, ve kterém jsou obsaženy všechny dostupné funkce, nebo pomocí klávesových zkratek. Výskyt některých ovladačů je z pochopitelných důvodů omezen pouze na konkrétní část aplikace, kde je jím obsluhovaná funkcionalita použitelná.

5.2.1 Menu

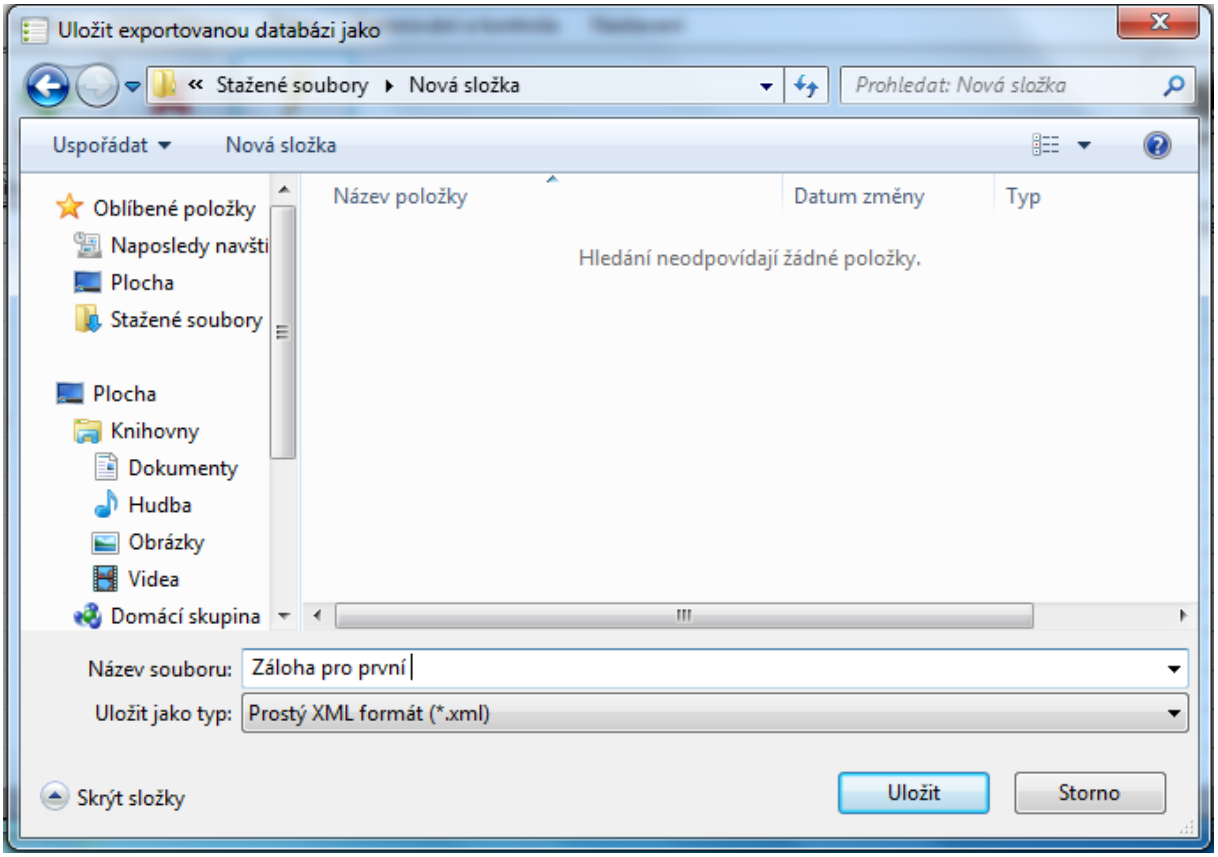
Lišta Menu je dostupná ve všech velkých oknech programu. Pro každou funkci programu jsou v Menu položky, které vyvolají jejich obsluhu. Tyto položky jsou členěny do záložek podle obvyklého použití, respektive podle blízkého vztahu obsluhovaných funkcí.



Soubor



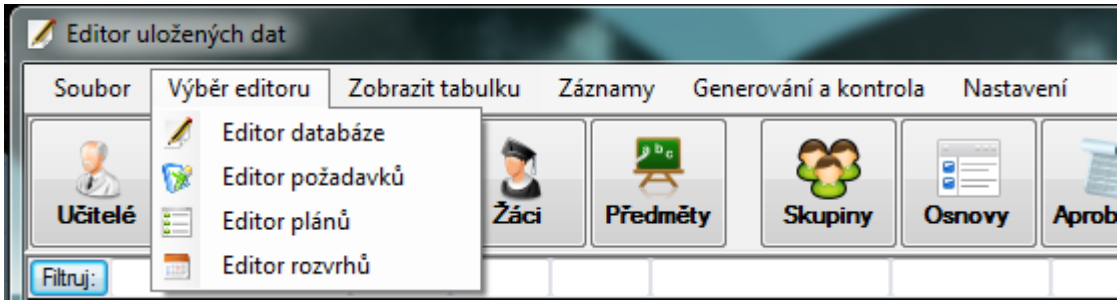
Záložka Soubor obsahuje položky, které se běžné vyskytují v programech. Jde o nabídku uložení informací do databáze či jejich načtení z databáze, která je napevno připojena k aplikaci. Import či export dat vůči jinému souboru proběhne po volbě destinace uživatelem.



Volba souboru probíhá stejně, jako je to u jiných aplikací pro *Windows*, tedy pomocí dialogu typu *Uložit jako*, respektive *Otevřít*. Tisk záznamů je řešen mimo prostředí aplikace, viz kapitola 4.6. Poslední položkou je ukončení aplikace. Vypnutí programu lze docílit i tlačítkem v pravém horním rohu.

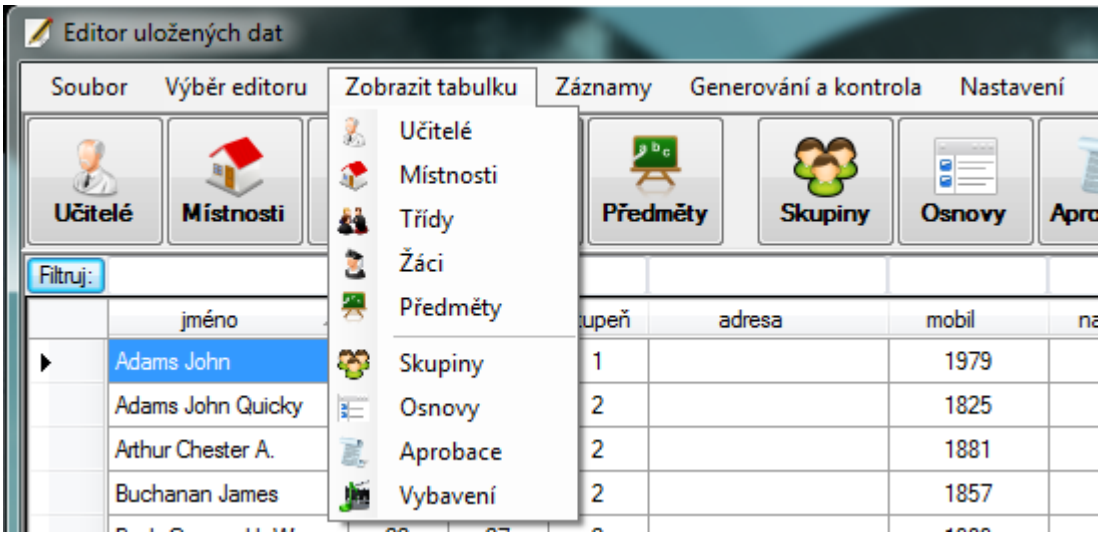
Výběr editoru

Záložka pro výběr editoru je také dostupná ve všech velkých oknech. Její položky slouží pro přímé přepínání mezi těmito okny, které uživateli zprostředkují různé funkce.



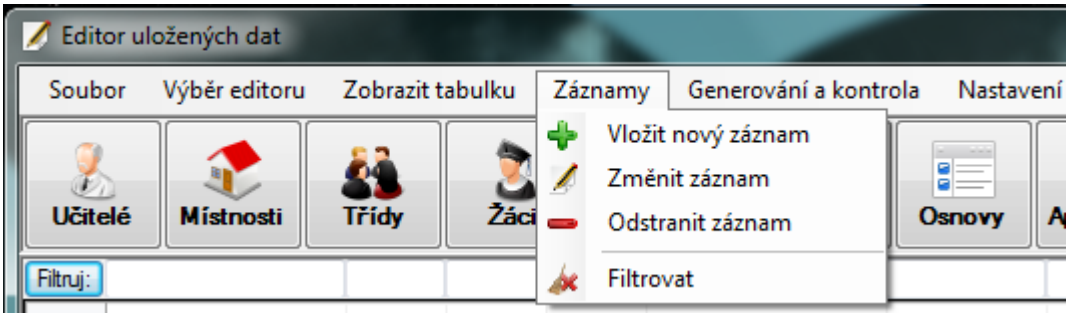
Zobrazit tabulku

Záložka Zobrazit tabulku nabízí přepínání mezi zobrazenými záznamy v oknu editoru dat. Z tohoto důvodu je dostupná pouze z okna tohoto editoru. Zobrazené záznamy odpovídají jednotlivým tabulkám v databázi, po přepnutí se záznamy ihned vyplní do mřížky okna.

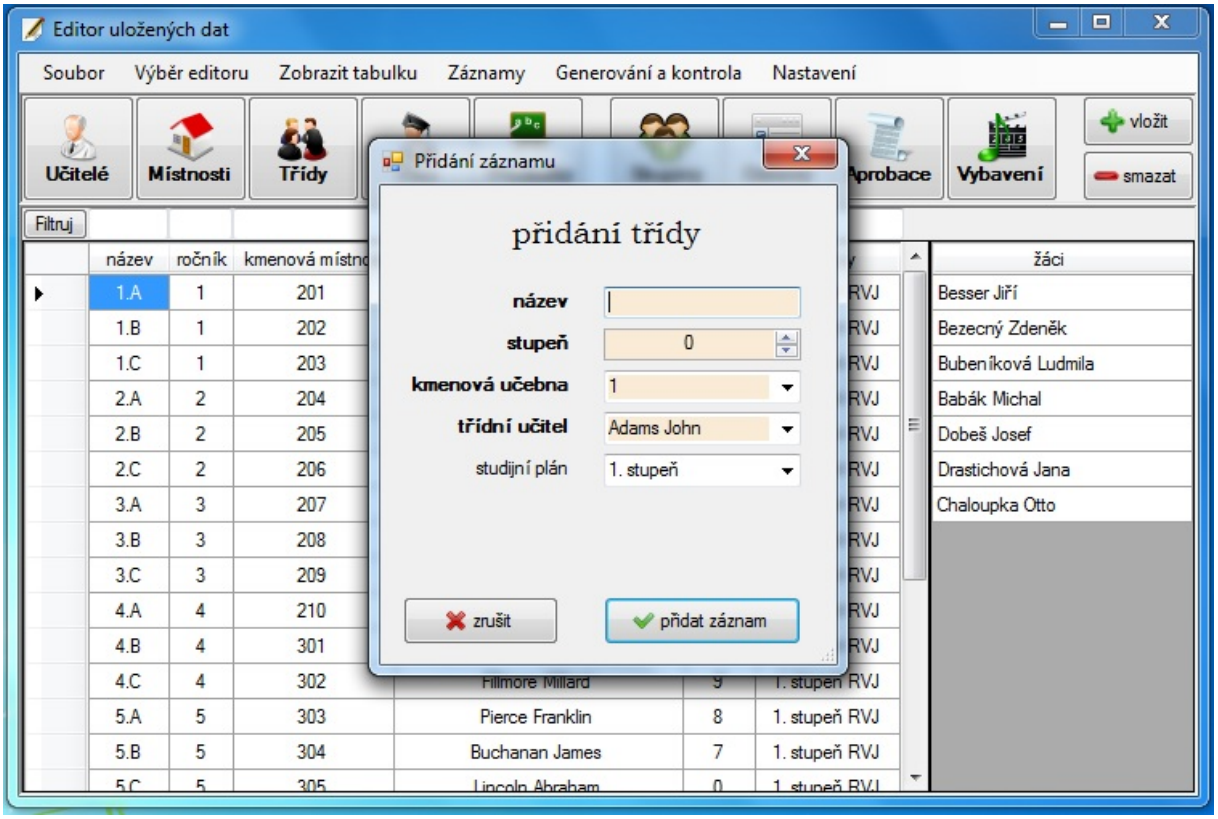


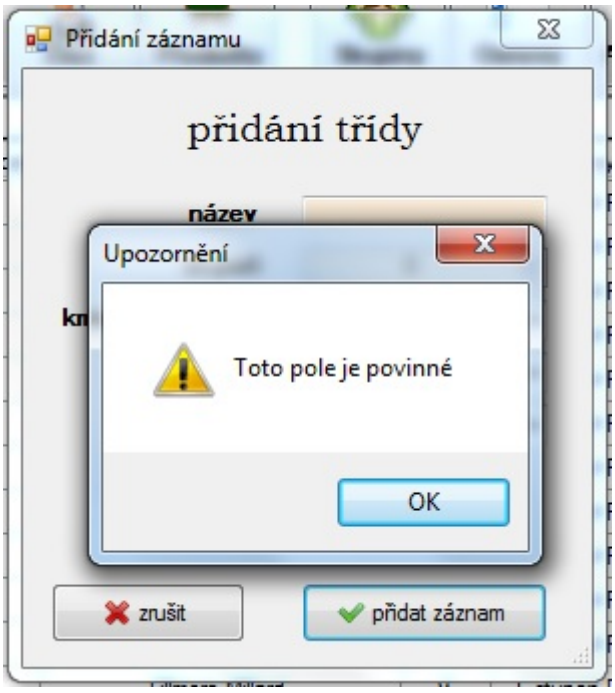
Záznamy

Záložka Záznamy poskytuje uživateli nástroje pro správu aktuálně zobrazených záznamů. Pomocí položek v záložce uživatel může přidávat údaje, odebírat či měnit aktuálně vybrané záznamy. Tato záložka je také k dispozici pouze z okna editoru dat.



Pro vkládání a úpravu záznamů vyskočí dialogové okno, které uživatele upozorní v případě chybně vložených či chybějících údajů.

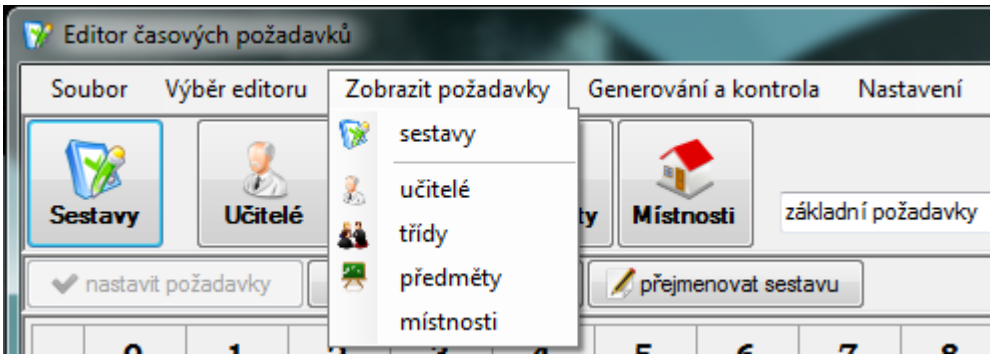




Zobrazit požadavky

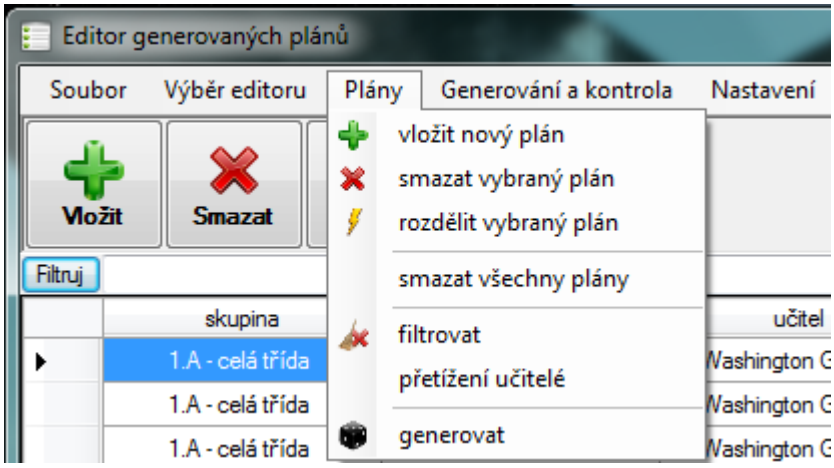
Záložka Zobrazit požadavky umožňuje změnit typ zobrazených a upravovaných požadavků. Přepínání uživateli dá buď úpravu samotných sad požadavků, nebo konkrétní časové požadavky učitelů, tříd, předmětů či místností. Tato zálož-

ka je k dispozici pouze pro editor požadavků.



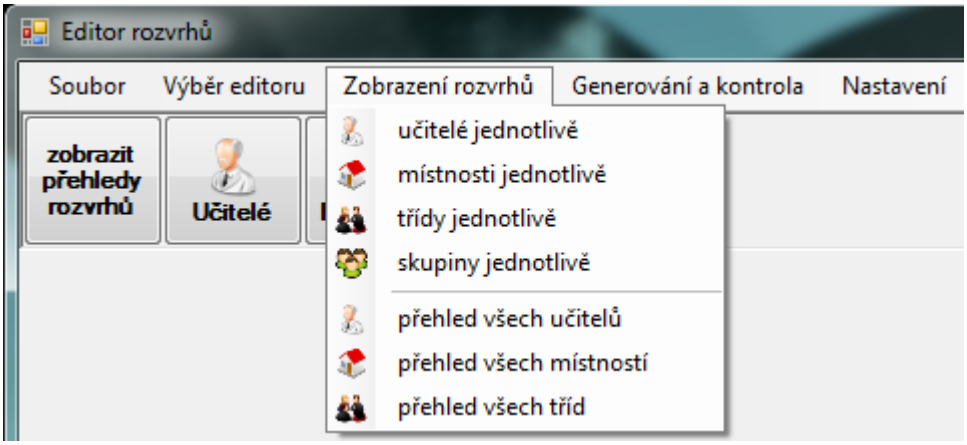
Plány

Záložka Plány je k dispozici v okně editoru plánů, kde nabízí podobnou funkcionalitu jako záložka Záznamy v editoru dat. Plány navíc přidávají položky pro smazání všech existujících plánů, respektive pro jejich vygenerování.



Zobrazení rozvrhů

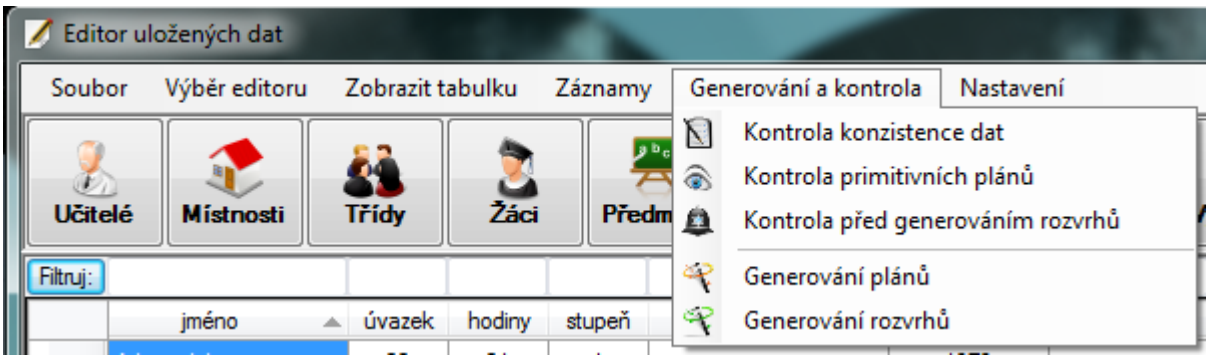
Záložka Zobrazení rozvrhů umožňuje změnit typ zobrazených a upravovaných rozvrhů. Tato zobrazení jsou pro jednotlivé rozvrhy i pro celkové přehledy všech lekcí.



Generování a kontrola

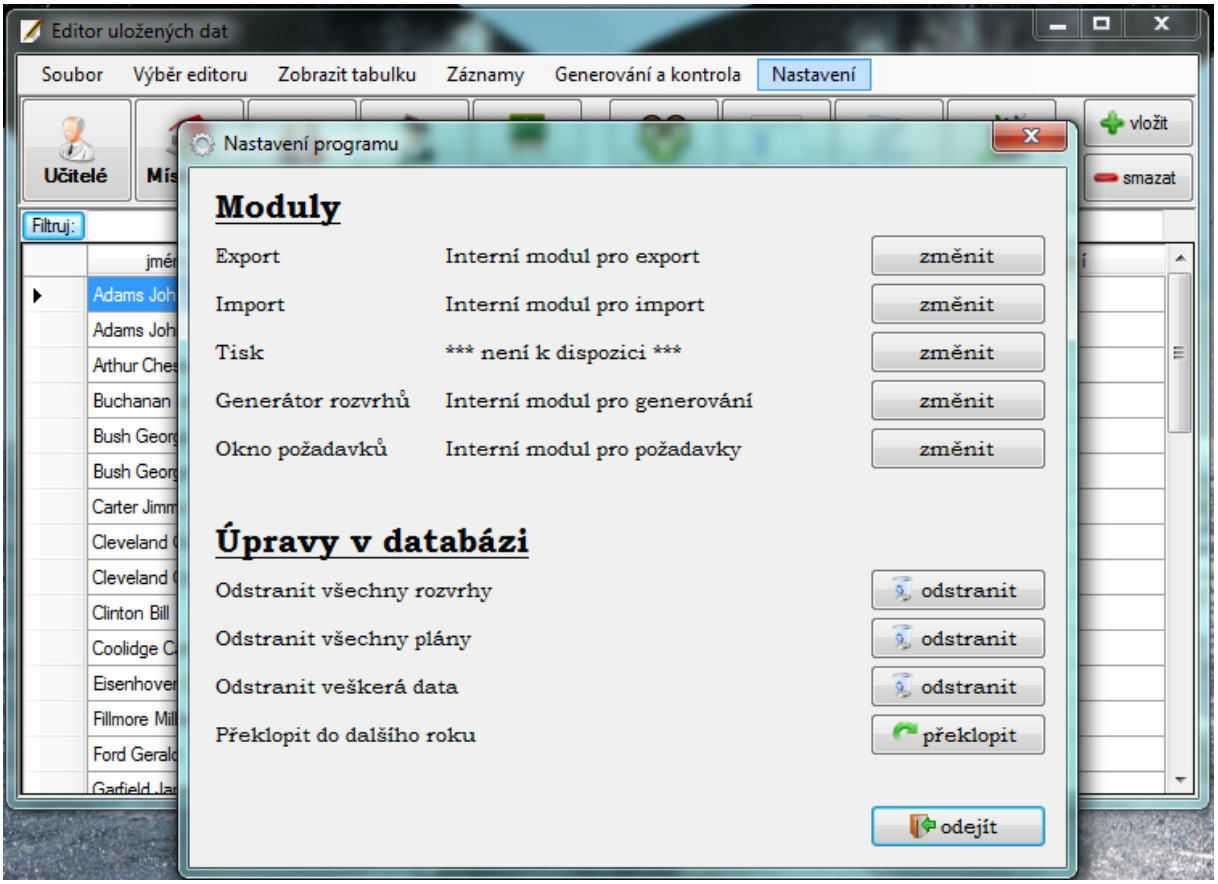
Záložka Generování a kontrola je dostupná ve všech oknech. Nabízí položky pro různou úroveň kontroly stavu databáze. První funkce zjistí, jestli je databáze připravena pro generování rozvrhových plánů ("konzistence dat"). Další zkontroluje, jestli jsou plány realizovatelné z pohledu zúčastněných jednotlivců. Poslední kontrola zjistí, zda jsou splněny náležitosti pro generování samotných plánů a zda v plánech není někdo příliš vytížený.

Položky pro generování samy vyvolají kontrolu databáze a až poté spustí generátor.



Nastavení

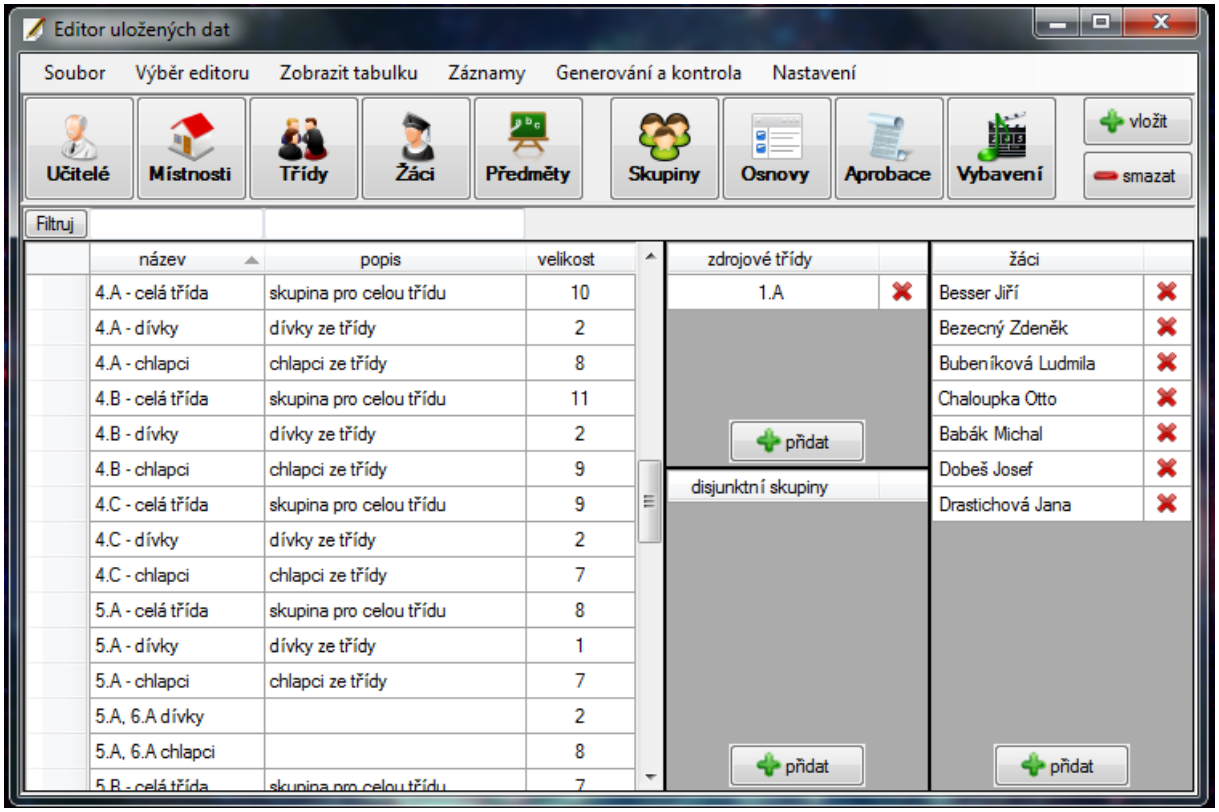
Nastavení je také dostupné ze všech oken, jedná se však o nástroj pro modifikaci aplikace samotné a ne jejích dat. Tato záložka vyvolá okno s další nabídkou, ve které je možné změnit moduly připojené do aplikace. Tyto moduly pak mohou jiným způsobem implementovat funkcionalitu programu, více viz kapitola 4.6;



5.2.2 Tlačítka

Tlačítka na rozdíl od Menu nabízí jen velmi omezené množství funkcí, byť těch nejpoužívanějších. Jelikož jsou ale přímo dostupná, dá se očekávat, že budou uživateli sloužit jako "rychlá volba", aby se nemusel proklikávat v menu.

Tlačítka se liší v každém editoru, protože poskytují obsluhu právě pro daný editor specielních funkcí.



5.2.3 Klávesové zkratky

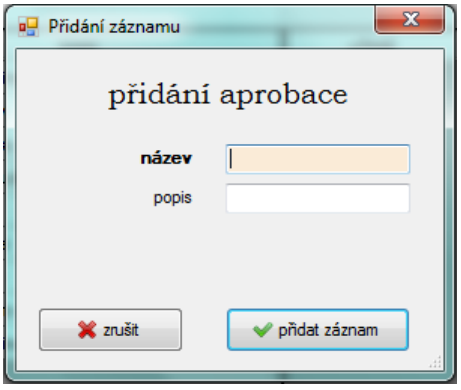
Klávesové zkratky jsou nejrychlejším možným ovládacím prvkem, protože s jejich využitím uživatel nemusí ani zvedat prsty z klávesnice. Přiřazená obsluha pro danou zkratku je pro všechna okna stejná, ale ne vždy je každá funkce dostupná. Podrobný přehled uvádím v následující tabulce.

zkratka	funkce	dostupnost pro editor
F2	změna aktuálního záznamu	dat a plánů
F3	vložení záznamu	dat a plánů
F4, Delete	odstranění aktuálního záznamu	dat, požadavků a plánů
F5	import databáze	všechny
F6	export databáze	všechny
F7	vygenerování rozvrhových plánů	všechny
F8	vygenerování rozvrhů	všechny
F9	rozdělení rozvrhového plánu na více plánů	plánů
Ctrl+S	uložení změn do databáze	všechny
Ctrl+L	načtení z databáze	všechny
Ctrl+P	vyvolání okna pro tisk	všechny
Ctrl+Tab	přepnutí na následující okno	všechny
Ctrl+Shift+Tab	přepnutí na předchozí okno	všechny
Esc	přepnutí na editor dat	všechny

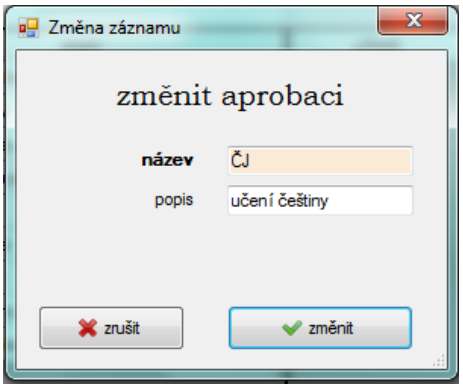
5.3 Správa dat

Prvním velkým oknem, které se po spuštění programu zobrazí, je editor dat. Ten v sobě integruje většinu tabulek pro zobrazení záznamů z databáze a skrze něj se také dát provést většina úprav.

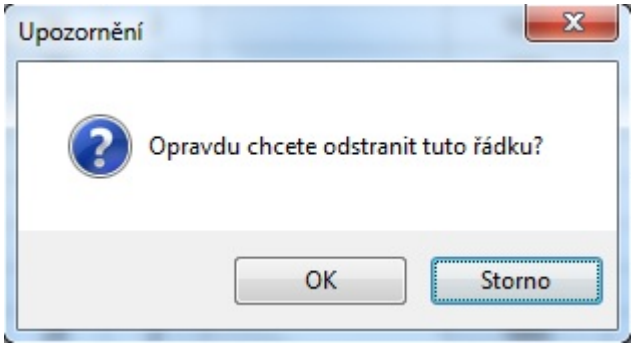
5.3.1 Vkládání, změna a odstranění záznamů



Pokud uživatel chce vložit nový záznam, stačí mu některým ze způsobů popsaných výše přepnout editor na příslušnou tabulku a zahájit vkládání dat. Program na to zareaguje vytvořením dialogového okna, které uživateli pomůže s následným vkládáním údajů.



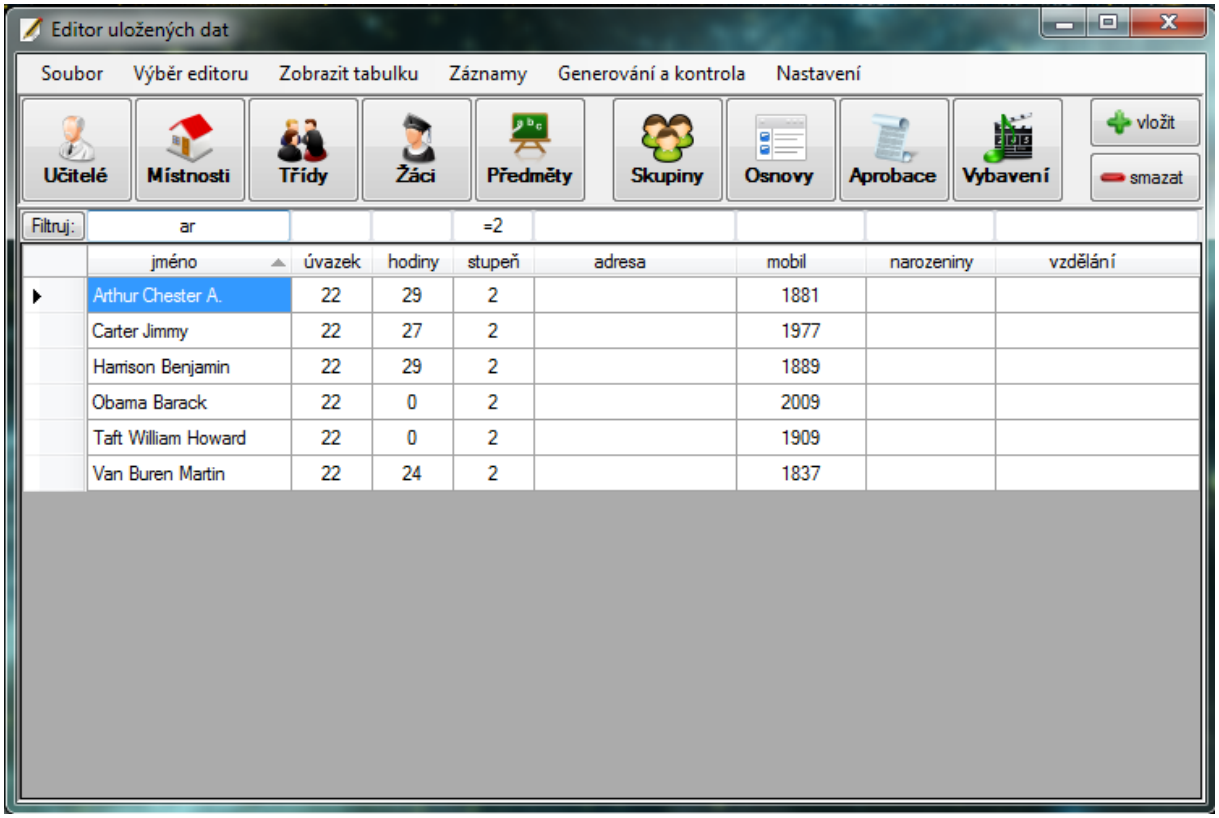
Pro změnu záznamů se užije podobný postup jako pro vkládání, po výběru tabulky je potřeba označit konkrétní záznam, který se má změnit. Poté uživatel buď začne psát na klávesnici, stiskne F2 či položku v Menu a program mu opět vytvoří dialogové okno. Tentokrát jsou políčka okna předvyplněná stávajícími záznamy. Uživatel hodnoty může libovolně upravit a pak změny přijmout či odmítnout pomocí tlačítek na tomto okně.



Pro odstranění záznamu je nutné vybrat tabulku, hledaný záznam a stisknout Delete, F4, příslušné tlačítko či položku v Menu. Program vyvolá dialogové okno pro ověření, zda uživatel skutečně chce záznam odstranit, zda se nejedná o omyl.

5.3.2 Filtrování záznamů

Zobrazené záznamy v tabulkách se dají snadno filtrovat. Do políčka nad odpovídajícím sloupečkem, ve kterém chce uživatel filtrování provést, uživatel napíše hodnotu pro filtrování. Pro textové řetězce se program snaží hledat shodný podřetězec, pro čísla lze nastavit hledání větších (>), menších (<) či přesných hodnot (=). Pokud je zadáno pouze číslo, hledá program stejně velká a větší čísla.

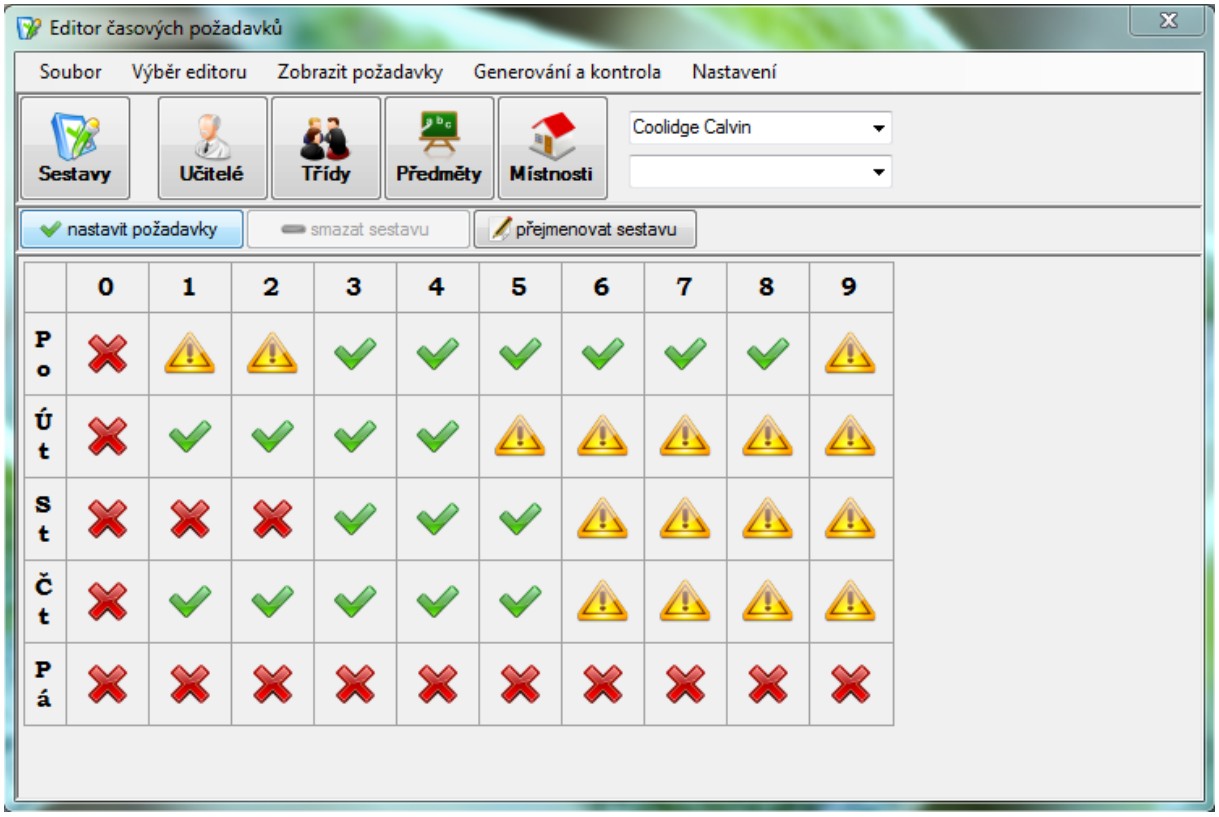


5.3.3 Osnovy

V případě, že uživatel nemá k dispozici potřebné osnovy, může si vytvořit nové s pomocí přednastavené šablony. Vkládání se spouští jako u všech ostatních záznamů. Program uživateli pomůže nastavit hodinové dotace pro jednotlivé předměty pro každý ročník, ověří dostatečné hodnoty pro jednotlivé bloky podle RVP a informuje jej o zbývajících disponibilních hodinách.

5.4 Organizace požadavků

Požadavky slouží k určení časů, kdy učitelé, třídy, místnosti a předměty chtějí či mohou mít rozvržené lekce.

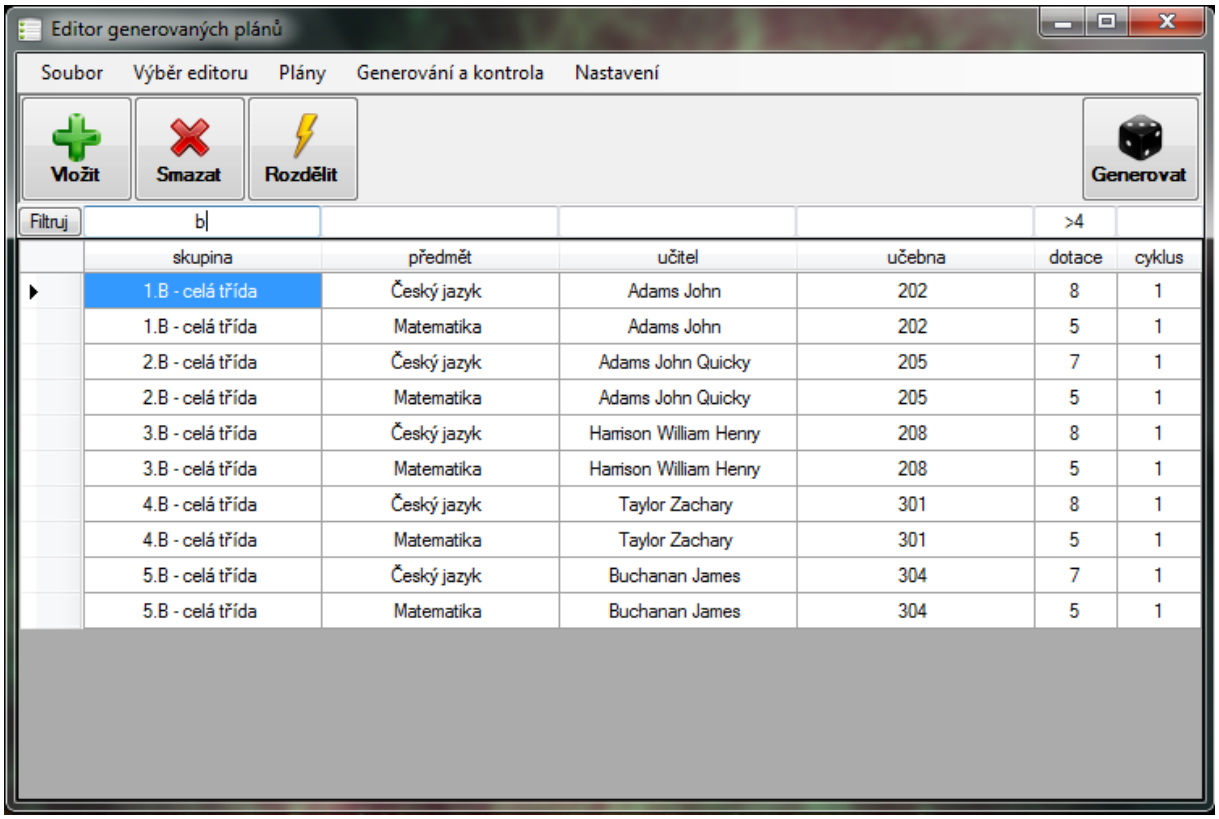


Defaultně má každý jako své požadavky nastavenou sadu "Základní požadavky". Nové sady lze vytvořit libovolnou změnou sady aktuálně zobrazené. Pokud uživatel klikne na ikonku pro určitou hodinu ve dne, změní se. Pokud klikne přímo na hodinu (první políčko ve sloupci), změní se všechny hodnoty v danou hodinu. Obdobně to platí pro změnu hodnot pro celý den.

Pro přiřazení konkrétní sady vybranému jedinci je nutné zvolené nastavení potvrdit tlačítkem "nastavit požadavky".

5.5 Rozvrhové plány

Dalším velkým oknem je editor plánů, které slouží jako hlavní podklad pro vygenerování rozvrhů. Plány lze rychle vytvořit pomocí generátoru, ale uživatel je také může vyrobit postupně sám. Vkládání, změny, odstraňování i filtrování plánů funguje stejně jako práce se záznamy v prvním popisovaném editoru.

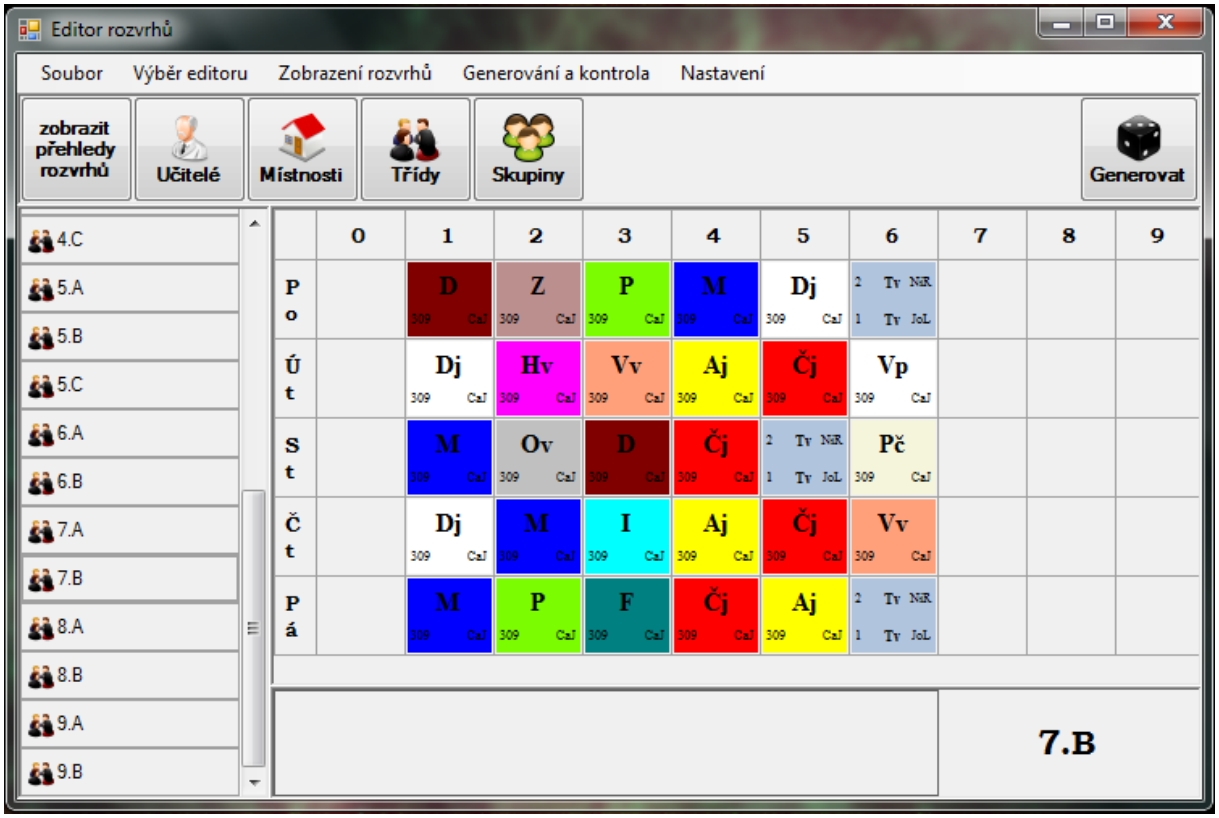


5.6 Správa rozvrhů

Poslední editor se zabývá zobrazením a úpravami samotných rozvrhů.

5.6.1 Prohlížení rozvrhů

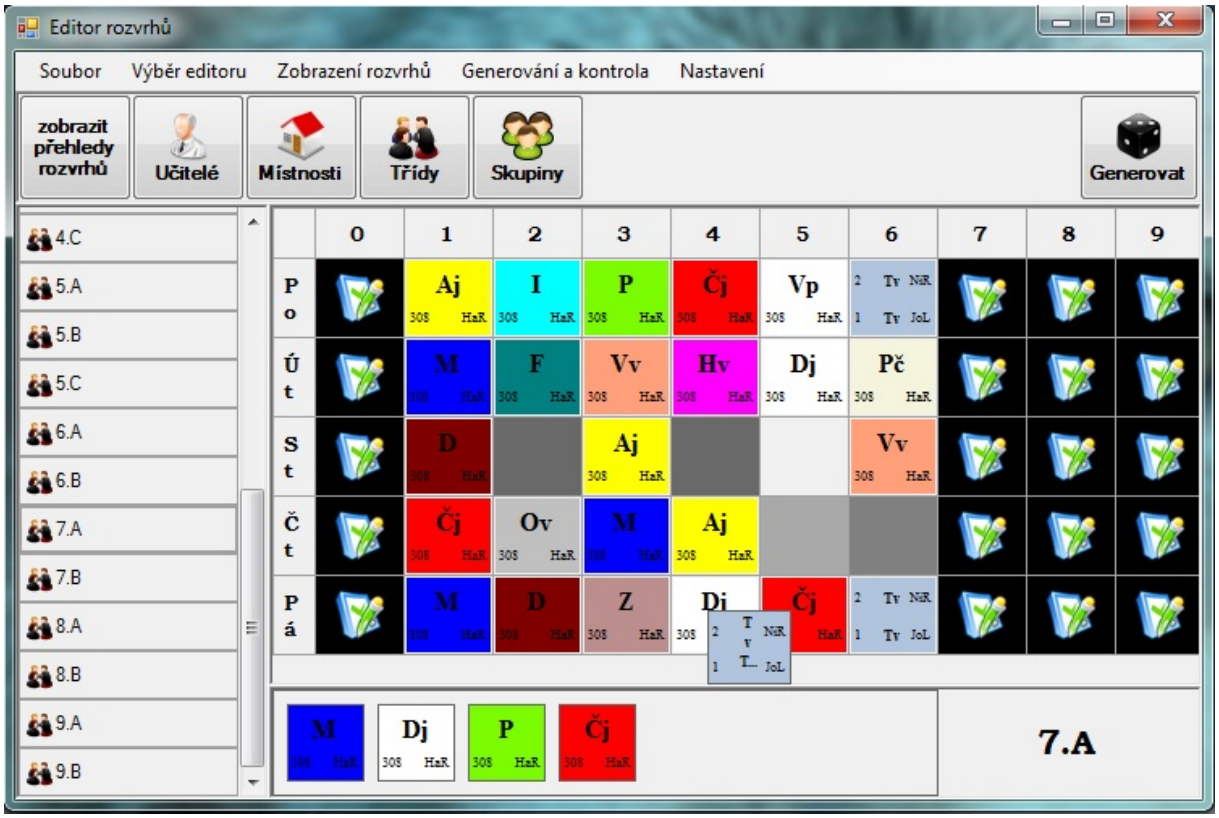
Vytvořené rozvrhy lze prohlížet v režimech pro jednotlivce, například třída 1.A, i v režimech celkových přehledů, například pro všechny místnosti. Uživateli jsou k dispozici tlačítka i Menu pro přepínání mezi těmito režimy.



5.6.2 Vytvoření rozvrhů

Nejčastěji bude uživatel rozvrhy vytvářet pomocí generátoru, který je dostupný z Menu, přes tlačítko i klávesovou zkratku. Uživatel je průběžně informován o postupu při generování menším oknem se zprávami. Je zde i možnost předčasně generování přerušit.

5.6.3 Režim chytré nástěnky



Uživateli je k dispozici i poloautomatický režim tzv. chytré nástěnky. Při něm může sám upravovat rozmístění lekcí, přičemž program pouze kontroluje základní omezení, například aby učitel neměl v jednu chvíli rozvržené dvě lekce. Pro usnadnění práce je mřížka pro vložení lekce podbarvena pro každé políčko jinak podle toho, jak moc je vhodné na toto místo lekci vložit.

Závěr

Provedl jsem analýzu modelové školy a její potřeby po rozvrhovacím programu. Z existujících aplikací jsem vybral dvě s nejvýznamnějším zastoupením. Tyto aplikace také charakterizují odlišné přístupy k řešení rozvrhování, avšak ani jedna nenaplnila očekávání. Navrhl jsem tedy požadavky pro vlastní řešení a program vytvořil.

Nový program má předpoklady pro splnění na počátku definovaných požadavků, avšak nemám k dispozici dostatečné množství dat pro důkladné ověření jeho výkonnosti. Testování jsem provedl na několika variantách menších škol, program při nich uspěl, tyto případy jsou ale pro generování jednodušší. Vytvořil jsem pro testování i větší imaginární školu, avšak nedokážu simulovat komplexnost vztahů, které v realitě vznikají, ani tato kontrola tedy není dokonalá.

Myslím si, že program nabízí kvalitní generátor rozvrhů a mnoho využitelných nastavení pro přiblížení se k optimálnímu řešení. Především pomocí nástroje časových požadavků je možné přijatelné řešení velmi omezit.

Užitečnou pomůckou jsou podle mne i šablony pro vytváření školních osnov, jelikož umožňují zkrátit práci na tomto problému z několika hodin na pár minut.

V případě, že by program měl úspěch, je možné jeho další rozšiřování. V této oblasti bych chtěl hlavně vyzdvihnout možnost v budoucnu oddělit databázi na samostatný server a prakticky neomezenou rozšiřitelnost podpory formátů pro import a export dat.

Seznam použitých zkratek

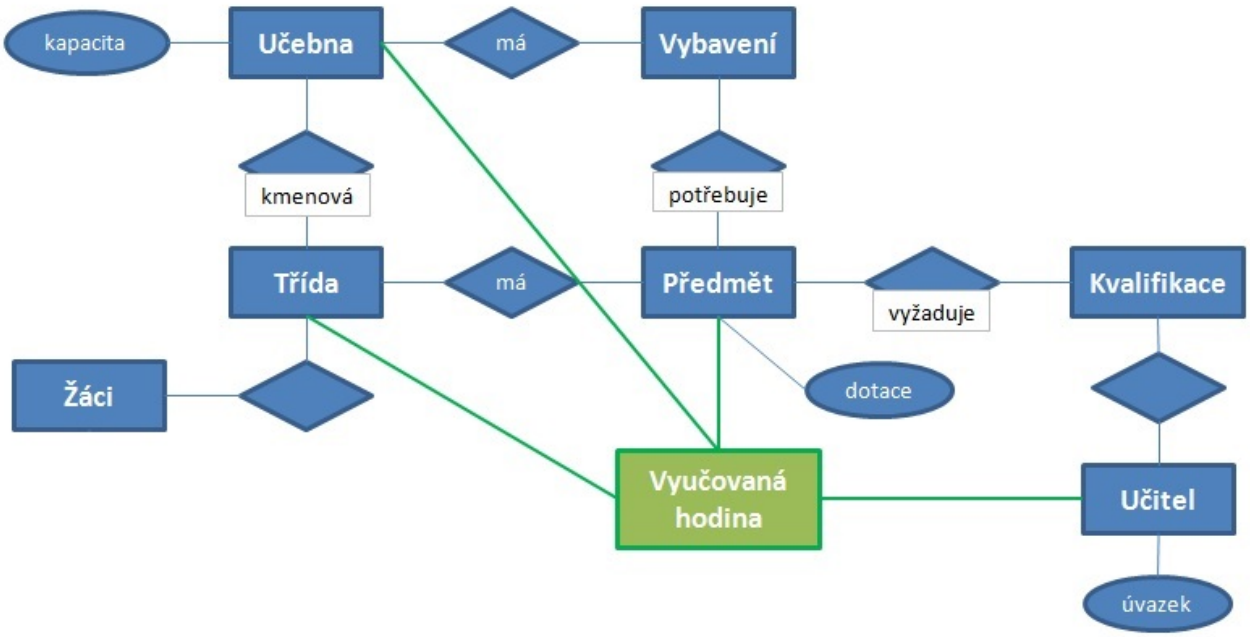
RVP	rámcový vzdělávací program viz www.msmt.cz/vzdelavani/skolskareforma
MS	Microsoft viz www.microsoft.com

Přílohy

1 Tabulka vybraných vlastností programů

	Bakaláři	aSc Rozvrhy	plánovaná aplikace
režim nástěnky	ano	ano	ano
generátor rozvrhů	ano	ano	ano
suplování	ano	ano	ano
úspěšný pokus o vygenerování sady rozvrhů	ne	ne	ano
šablony	ne	ne	ano
příklady použití	ne	ano	ano
kontrola trivialit s nápovědou	ne	ano	ano
tisk výsledku	ano	ano	ano
export výsledku pro web, mobil	ano	ano	ano
základní administrativa	ano	ne	ano
rozšířené nástroje pro správu školy	ano	ne	ne
cena zohledňované aplikace (pro ZŠ do 200 žáků, včetně všech slev)	11820 Kč[1]	5290 Kč[2]	–

2 Zjednodušený ER diagram



3 Plný diagram závislostí v databázi